한국정보보호학회
Korea Institute of Information Security & Cryptology

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

일시 2019. 11. 30(토)

장소 중앙대학교(흑석동)

Proceedings



주최 한국정보보호학회 Korea Institute of Information Security & Cryptology

주관 중앙대학교 CHUNG-ANG UNIVERSITY

후원 국가정보원 National Intelligence Service Korea, 과학기술정보통신부, 행정안전부, KISA 한국인터넷진흥원, ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute, NSR 국가보안기술연구소 National Security Research Institute

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

목 차

- 구두발표 트랙 A ····· 001
- 구두발표 트랙 B ····· 068
- 구두발표 트랙 C ····· 132
- 구두발표 트랙 D ····· 199
- 구두발표 트랙 E ····· 299
- 구두발표 트랙 F ····· 365
- 포스터 트랙 P ····· 432

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 A

자동차 소프트웨어 보안 개발 프로세스를 위한 요구사항*

정승연**,강수영†, 김승주†

**고려대학교 자동차융합학과, † 고려대학교 정보보호대학원

Requirements for Automotive SDL

Seung-yeon Jeong**, Soo-young Kang†, Seung-joo Kim†

**Department of Automotive Convergence, Korea University

† Center for Information Security Technologies(CIST), Korea University

요 약

기능 안전에 초점을 맞추어왔던 기존 자동차 소프트웨어 개발은 자동차 사이버보안 위협에 대해 충분히 보안성을 확보하지 못한다. 이를 해결하기 위해 표준 기관들은 자동차 사이버보안을 위한 표준을 제정 중이고, 2022년 발효될 UNECE 사이버보안 규제가 대표적이다. UNECE 규제에서는 자동차 개발 생명주기 전체에 대한 보안 정책을 요구하는데, 기존에는 보안 개발 생명주기가 존재하지 않는다. 이러한 자동차와 다르게 기존 소프트웨어 개발의 경우 효과적인 보안 개발 생명주기를 통해 시스템의 보안성을 향상시켰다. 본 논문에서는 자동차 안전 개발 생명주기를 기준으로 기존 소프트웨어 보안 개발 프로세스인 NIST SSDLC와 Microsoft SDL 및 UNECE 규제의 요구조건을 비교 분석 후 Automotive SDL 요구사항을 도출한다.

I. 서론 및 관련 연구

기존 자동차 산업에서 소프트웨어 개발은 기능 안전에 초점을 맞추어왔다. 기능 안전의 목표는 시스템 내부의 전기 전자 시스템 오작동을 방지하는 것으로 기능 안전이 제대로 지켜지지 않으면 소비자 피해뿐만 아니라 막대한 경제적 손해까지 발생시킬 수 있게 된다. 따라서 국제 표준화 기구 ISO(International Organization for Standardization)에서는 ISO 26262라는 자동차 기능 안전 표준을 통한 자동차 개발을 요구하고 있다. 특히 ISO 26262에서는 안전 생명주기를 통해 개발 프로세스 전반에 대해 기능 안전을 다룸으로써 자동차 개발에서의 안전성을 확보한다.[1]

안전성과 다르게 사이버보안은 자동차 개발에서 배제되어왔다. 사이버보안의 목표는 외부 보안 위협으로부터 시스템을 보호하는 것으로, 기존 자동차는 물리적인 접근을 통한 정보 유출 정도의 위협이 존재했다면 최근 커넥티드 카의 등장과 함께 위협 노출 가능성이 커졌고 사이버보안의 중요성 또한 높아졌다. 이에 대해 여러 표준 기관에서는 자동차 사이버보안 관련 표준 및 규제 제정에 돌입하였고, 대표적으로 UNECE(United Nations Economic Commission for Europe)에서는 자동차 사이버보안에 대한 규제를 제정 중이다.[2] UNECE 사이버보안 규제는 2022년부터 발효될 예정이며, 이에 인증을 받지 못한 차는 유럽에 수출할 수 없게 된다. UNECE 규제에서는 개발 생명주기 전체에 대한 자동차 보안 정책을 요구하는데 기존 자동차 개발에서는 보안 개발 생명주기가 적용되지 않았기 때문에 자동차 소프트웨어 개발 보안 생명주기 필요성이 강조되고 있다.

안전성 위주의 프로세스에 보안 활동을 통합시키기 위한 연구는 많이 이루어졌지만, 자동차 표준 개발 프로세스나 UNECE 규제를 준수하는 완전한 프로세스는 제안되지 않았다. [3],[4],[5],[6],[7]에서는 일부 단계에 대해서만 다뤄졌고, [3],[6],[7],[8]에서는 자동차가 아닌 CPS(Cyber Physical System) 대상의 프로세스를 구축하였다. 이러한 자동차 소프트웨어 개발과 다르게 기존 소프트웨어 개발의 경우 효과적인 보안 개발 생명주기를 통해 보안성을 향상시켰다. 대표적으로 미국 국립 표준 기술 연구소 NIST는 2008년 발간한 SSDLC(Secure System Development Life Cycle) 표준화 문서[9]를 통해 개발 프로세스 전반에 대한 보안 활동의 중요성을 강조하였고, Microsoft는 2002년부터 SDL(Security Development Lifecycle)을

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터 육성지원사업의 연구결과로 수행되었음(IITP-2019-2015-0-00403)

통해 소프트웨어의 보안성을 확보하였다.[10]

본 논문에서는 기존 소프트웨어 보안 개발 생명주기를 기반으로 자동차 소프트웨어 보안 개발 생명주기 Automotive SDL을 제안하고자 한다. 논문에서는 NIST SSDLC와 Microsoft SDL, UNECE 규제의 요구조건을 비교 분석한 후 이를 기반으로 Automotive SDL이 가져야 할 단계별 요구사항을 도출함으로써 설계의 기반을 마련한다.

				ISO 26262 (기반)	NIST SDLC	Microsoft SDL	UNECE 사이버보안 규제 (필수)
1. 교육	1.1 기본 교육			X	-개발 일정/자원에 영향이 있는 활동	-설계/코딩/테스팅, 위협모델링, 프라이버시	-보안 인식 확보
	1.2 심화 교육			X	-사업소유자/개발팀 교육	-특정 역할 대상 세부 교육	-데이터 자산 관리/유지보수 교육 -설계/개발 역할 교육
2. 초기	2.1 역할 부여	2.1.1 보안책임자 역할		X	-정보 시스템 보호 책임자(ISSO)	-보안 팀, 보안 어드바이저	-보안 역할 및 책임
		2.1.2 업체 간 역할 및 연결성		X	X	X	X
	2.2 도구/기술 선정			-방어 기술	-테스팅/평가 도구	-버그 트래킹(Tracking) 도구	X
	2.3 보안 목표			-기능 안전 개념 설계	-보안 범주화 및 노력 수준 측정	-품질 게이트 및 버그 바 지정	-환경에 대한 보호 정책 설정
3. 설계	3.1 구조 설계	3.1.1 전체 구조		O	O	O	O
		3.1.2 구성요소 간 구조		- 구성요소 간 계층/결합 관계 설계	X	X	X
		3.1.3 안전/보안 상호 의존성		X	X	X	X
		3.2 설계 요구사항		O	O	O	O
	3.2 위협 분석	3.2.1 대상 선정		O	O	O	-서버, 통신 채널, 업데이트 과정, 외부 연결성
		3.2.2 위협 식별/평가/완화		-안전성 평가 (ASIL)	O	-위협 평가(STRIDE) -위험도 분류(DREAD)	- Extended CIA 대상 위협 도출 - 위협에 대해 level 부여
		3.2.3 보안/안전 요구사항		-기능/기술 요구사항으로 소프트웨어 요구사항 도출	O	O	O
	3.3 공격/위협 표면 감소			X	X	-접근 제한, 최소권한의 법칙, 계층 방어	O
	3.4 영향 평가	3.4.1 프라이버시 영향 평가		X	O	-프라이버시 수준 평가(P1,P2,P3)	O
		3.4.2 사업상 영향 평가		X	O	X	X
4. 구현	4.1 도구 및 기술 선정			-승인된 언어 구성 사용	-자동화 도구 선정	-승인된 도구 사용 및 -불안정한 기능 사용 배제	X
	4.2 문서화	4.2.1 문서 작성 및 활용		O	-보안 코드 관행 및 저장소 -보안 관련 문서 (C&A를 위한)	-사용자/관리자/개발자 매뉴얼 -개발자를 위한 코딩 매뉴얼	-안전한 코드 모범사례
		4.2.2 각 조직의 매뉴얼 준수		X	X	X	X
	4.3 테스트 케이스			-테스트 케이스 추출 -커버리지 파악	O	O	O
5. 검증	5.1 검증 (정적 테스트)	5.1.1 소스 코드 검증		-정적 코드 분석 -코드 의미 분석	- 정적 코드 분석	-정적 코드 분석 -수동 코드 검토	X
		5.1.2 설계 검증	5.1.2.1 단위 설계 검증	-통합/단위 모델 설계 검증	-성능 사양 검토 (타 시스템에 대한 평가 수행)	X	X
			5.1.2.2 합성 설계 검증	-구성요소 간 응집도 검증 -데이터/제어 흐름 분석	X	X	X
		5.1.3 요구사항 검증		-시스템 설계 일치성 입증 -다양한 환경에서 검증 (ECU/네트워크/자동차)	- 보안 정책 위반 점검	X	X
	5.2 재정 리뷰			X	O	X	X
	5.2 테스트 (동적 테스트)	5.2.1 단위 테스트		-요구사항 기반 테스트 -구성요소 간 연동 테스트	X	X	-보안 기능 테스트
		5.2.2 통합테스트		-요구사항 기반 테스트 -인터페이스 테스트 -오류 주입 테스트 -자원 사용 테스트	O	-런타임 검증 테스트(기능 평가) -퍼즈 테스트	O
		5.2.3 침투 테스트		X	X	O	X
	5.4 설계 적용 검토			O	-구조/설계 리뷰 -최종 프로젝트 상태/재무 검토 -정보 시스템 승인 -기존 환경/시스템에 보안 통합	X	X
	5.5 위험평가 검토			O	-위험 관리 리뷰 -시스템 보안 평가	-위협 모델 및 공격 표면 검토	-위험평가 검증
	5.5 보안/안전 푸시			X	X	O	X
6. 운영	6.1. 배포			-확인 및 배포	-배포 준비 검토 -배포 또는 연결 승인	-배포 전 최종 보안 검토 -배포	X
	6.2 사고 대응 프로세스			O	-비상 및 재해 복구 계획	-대응 센터 설립 및 사용자와의 지속적 의사소통	O
	6.3 모니터링			-외부 모니터링 시설 -흐름 모니터링 제어	-지속적인 모니터링 수행	O	O
	6.4 업데이트	6.4.1 업데이트 프로세스		X	X	-보안 이슈에 대한 패치 시기 고려	O
		6.4.2 업데이트 관련 문서		X	X	-보안 게시판	-시스템 버전 관리
7. 폐기	7.1 폐기(제거)			O	-제거/전이 계획 수립 -소프트웨어 제거 및 매체 삭제	X	-데이터 제거/전이 계획 -자산의 안전한 폐기
	7.2 정보 보존/저장			X	-정보보호의 보장	X	-데이터 자산 처리

[표 1] 프로세스 비교 분석

II. Automotive SDL

2.1 기존 프로세스 비교 분석

본 논문에서는 자동차 안전 개발 프로세스인 ISO 26262, 소프트웨어 보안 개발 프로세스인 Microsoft SDL 및 NIST SSDLC, UNECE 자동차 사이버보안 규제를 대상으로 프로세스를 비교 분석하며 [표 1]과 같이 교육, 초기, 설계, 구현, 검증, 운영, 폐기의 7단계로 분류된다.

먼저, 교육 단계에서는 기본적인 보안 인식 개선 교육 및 특정 업무 담당자를 위한 심화 교육이 이루어진다. 초기 단계에서는 역할 및 전반적인 프로젝트 목표를 설정하고 발생 가능한 위협이나 버그들에 대한 수준을 결정한다. 설계 단계에서는 시스템의 보안 및 설계 요구사항을 도출하는데, 이 과정에서 위협 평가가 포함된다. 구현 단계에서는 자동화 도구나 매뉴얼을 통해 요구사항을 기반의 소프트웨어를 개발한다. 검증 단계에서는 구현된 시스템을 기반으로 테스트 및 보안성 검토를 한다. 운영 단계에서는 시스템을 배포하고 주기적인 모니터링 및 사고 대응 프로세스를 통해 소프트웨어에 발생하는 위협에 대해 빠르게 대처한다. 마지막으로 폐기 단계에서는 기존 시스템의 데이터를 저장 및 전달하고 시스템을 폐기한다.

2.2 Automotive SDL 요구사항

본 논문에서는 자동차 소프트웨어 보안 개발 프로세스 Automotive SDL을 제안하고자 한다. 프로세스 비교 분석[표 1] 및 선행 연구를 바탕으로 단계별 Automotive SDL 요구사항을 도출하였고, 그 결과는 [그림 1]과 같다. Automotive SDL은 Microsoft SDL과 NIST SSDLC를 기반으로 설계되며 UNECE 규제의 요구조건을 모두 충족한다. 또한, 안전 개발 프로세스를 기반으로 하므로 기존 자동차 개발 프로세스에 통합하기 효율적이다.

1) 교육

기본적으로 보안 인식 확보를 위한 교육이 이루어져야 하며 이는 SDL 및 NIST에서도 볼 수 있듯이 위험 분석/프라이버시와 같은 프로세스 계획 및 진행에 도움이 되는 주제를 포함해야 한다. 또한, 개발/테스트 팀의 경우 추가적인 보안 교육이 별도로 이루어져야 하며 UNECE 요구조건에 따라 유지보수를 담당하거나 데이터 자산을 다루는 직원에 대해서도 추가 교육이 수행되어야 한다.

2) 초기

프로젝트를 진행하기 전에 전체 프로세스에서 보안 팀과 책임자를 지정해야 하며, 프로젝트 전반에서 다룰 보안 수준에 대해 명시를 해놓은 후 이를 따라야 한다. 추가적으로 자동차 개발의 경우 다양한 업체의 산출물을 통합하는 과정을 통해 시스템을 설계하므로, 각 업체 간의 역할 및 연결성을 프로세스 초기에 설계해놓아야 한다.

3) 설계

기능 안전 프로세스에서 수행되는 안전성 위험평가와 같이 보안성에 대해서도 위험평가가 이루어져야 한다. 이때 UNECE에서 언급하는 Extended CIA 대상의 위협을 다뤄야 하는데, 이는 Microsoft SDL의 STRIDE 모델을 활용함으로써 가능하다. STRIDE를 통해 다룰 수 없는 책임성 (Accountability)의 경우에는 추가 고려되어야 한다. 또한, 자동차 소프트웨어는 사용자 개인 정보를 다루는 경우가 다수이기 때문에 프라이버시 수준에 따른 평가도 이뤄져야 한다.

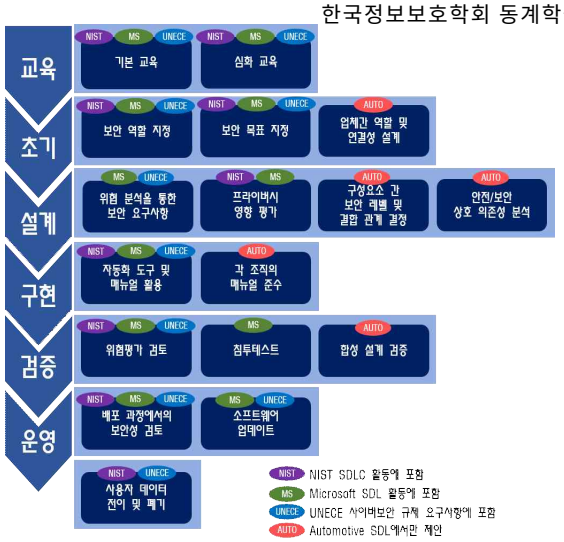
안전성과 보안성 사이에는 상호 의존성이 존재한다. 상호 의존성에는 보안이 안전을 위한 조건이 되거나 그 반대인 경우, 안전과 보안 대책이 서로를 강화하는 경우, 서로를 약화하는 경우, 서로 상호 독립적인 경우의 4가지가 포함된다[3]. 따라서 보안과 안전 사이에 존재하는 상호 의존성을 분석하고 이에 대해 통합 관점의 시스템을 설계하는 과정이 설계 단계에 포함되어야 한다. 또한, 모듈 단위의 산출물을 통합하는 자동차 개발 환경에 따라, 설계 과정에서 구성요소 간의 보안 수준/결합 관계 등이 설정되어야 한다.

4) 구현

구현 과정에서는 매뉴얼이나 도구 사용을 권장해야 한다. 특히, 자동차 소프트웨어 개발의 경우 각 업체를 통해 구현한 후 통합되기 때문에, 프로세스에서 초기에 결정한 사항들에 문서화를 통해 동일한 준수를 요청해야 함으로써 전체 시스템의 통일성을 유지해야 한다. 이때 앞서 권장되는 세부적인 자동화 도구나 코딩 매뉴얼을 통해 이 과정이 이루어져야 한다.

5) 검증

앞서 설계 단계에서 수행된 위협 평가는 검증 단계를 통해 재검토되어야 한다. 코드 분석은 안전 프로세스에서 수행되고 있으므로 중복 테스트는 효율성을 떨어트린다. 다만 침투 테스트의 경우, 기존에 수행되지 않았고 실제 소프트웨어의 적용 이후에 발생 가능한 위협을 발견할 수 있으므로 수행되어야 하며



[그림 1] Automotive SDL 단계별 요구사항

이때 Microsoft SDL을 활용할 수 있다. 또한, 소프트웨어를 합성 과정에서 발생 가능한 위협에 대해 검증하는 과정을 통해 보안성이 유지되어야 한다.

6) 운영

소프트웨어를 배포하는 과정에서도 마찬가지로 보안성이 유지되어야 하며 이후 모니터링을 통해 보안 위협 발생 여부를 점검해야 한다. 이때 ISO 26262에서 기존에 수행하던 모니터링을 활용할 수 있다. 소프트웨어 업데이트의 경우, UNECE 규제에서 별도로 요구하는 상세화 문서를 활용해야 한다.

7) 폐기

소프트웨어 폐기 시 사용자 데이터의 안전한 전이 및 폐기를 통해 이후 발생 가능한 보안 문제에 대처가 가능해진다. 이 과정의 경우, NIST SSDLC를 활용해야 한다.

III. 결론 및 향후 연구

증가하는 자동차 사이버보안 위협으로 인해 자동차 소프트웨어 개발에서는 보안 정책이 요구된다. 본 논문에서는 자동차 안전 개발 생명주기를 기준으로 기존 소프트웨어 보안 개발 프로세스인 NIST SSDLC와 Microsoft SDL 및 UNECE 규제의 요구조건을 비교 분석한 후, 이를 기반으로 Automotive SDL의 요구사항을 도출한다.

교육, 초기, 설계, 구현, 검증, 운영, 폐기의 총 7단계에 대한 요구사항을 도출하였으며, 각 요구사항에 대해 활용가능한 기존 소프트웨어 보안 개발 프로세스를 명시함으로써 자동차 보안 소프트웨어 개발 프로세스 설계의 발판을 마련한다. 나아가 본 논문에서는 자동차 소프트웨어 개발 환경에서 다루어져야 할 추가적인 요구사항 5가지를 도출함으로써, 보다 보안성이 향상된 Automotive SDL 개발을 가능하게 한다.

향후에는 요구사항을 기반으로 프로세스를 설계한 후 각 단계에 대해 상세화를 진행할 예정이다. 더 나아가 설계된 프로세스를 기반으로 성숙도 모델을 활용하여 Automotive SDL이 가지는 보안성을 검증할 예정이다.

[참고문헌]

[1] Debouk, Rami. "Overview of the 2nd Edition of ISO 26262: Functional Safety - Road Vehicles." General Motors Company, Warren, MI, USA (2018).

[2] Draft Recommendation on Cyber Security of the Task Force on Cyber Security and Over-the-air issues of UNECE WP.29 GRVA

[3] Sabaliauskaite, Giedre, and Aditya P. Mathur. "Aligning cyber-physical system safety and security." Complex Systems Design & Management Asia. Springer, Cham, 2015.

[4] Amorim, Tiago, et al. "Systematic pattern approach for safety and security co-engineering in the automotive domain." International Conference on Computer Safety, Reliability, and Security. Springer, Cham, 2017.

[5] Macher, Georg, et al. Integrated safety and security development in the automotive domain. SAE Technical Paper, 2017.

[6] Riel, Andreas, et al. "Integrated design for tackling safety and security challenges of smart products and digital manufacturing." CIRP annals 2017

[7] Schmittner, Christoph, Zhendong Ma, and Erwin Schoitsch. "Combined safety and security development lifecylce." 2015 IEEE 13th International Conference on Industrial Informatics (INDIN). IEEE, 2015.

[8] Kriaa,Siwar, et al. "A survey of approaches combining safety and security for industrial control systems." Reliability engineering & system safety 2015

[9] NIST SP 800-64, Security Considerations in the System Development Life Cycle

[10] Microsoft Security Development Iifecycle (SDL)

스마트 자동차 전자제어장치(ECU)의 보안취약점 분석 및 해결방안 마련

이기옥*, 장남수**

*세종사이버대학교 정보보호대학원

Vulnerability Analysis and Solutions for Smart Vehicle Ectronic Control Unit

Ki Ok Lee*, Nam Su Chang**

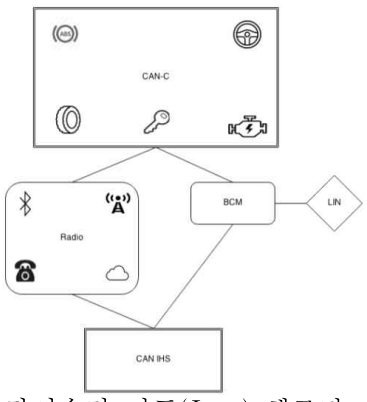
*Graduate school of Information Security, Sejong Cyber University.

요 약

스마트 자동차는 기계 중심 기술에서 전자, 정보통신 기술을 융합한 기술이 접목되어 “운전자 편의”와 “차량 안전”을 제공하기 위하여 발전하고 있다. 그러나 스마트 자동차가 발전하면 할수록 자동차 내 전장화 비율이 증가하면서 다양한 보안 위협에 증가하게 된다. 본 연구는 스마트 자동차 내 존재하는 보안 위협 및 각종 전자제어장치에서 발생할 수 있는 보안 취약점들에 대한 이해를 바탕으로 전자제어장치 펌웨어 변조 방지를 위하여 펌웨어 분석방법을 도출하고 정보보호 기술을 접목하여 전자제어장치(ECU)에 안전하고 보안성이 높은 펌웨어가 적용될 수 있는 방안을 제시하고자 한다.

I. 서론

최근 스마트 자동차 시장이 급격하게 발전함에 따라 자동차의 전장화 비율이 증가하면서 스마트 자동차를 대상으로 하는 해킹사례가 점차 증가하고 있고, 그 심각성도 점차 가시화되고 있다. 대표적인 피해 사례가 크라이슬러 지프(Jeep) 체로키 자동차의 사이버 해킹 사건이다. 지프 체로키는 자동차의 모든 조작이 무선 네트워크 해킹을 통해 가능하다. 인포테인먼트, 에어컨, 엔진, 브레이크 조작까지 가능할 뿐만 아니라 탑재된 GPS를 통한 위치 추적까지 가능하다. 이러한 해킹이 가능한 이유는 사용자 편의 기능으로 탑재된 유커넥트라는 인포테인먼트 시스템이 장착되어 있기 때문이다. 인포테인먼트 시스템은 그림1과 같이 사용자 편의를 위하여 무선 네트워크와 블루투스 기능이 탑재되어 스마트폰으로 전화를 걸거나, 음악을 듣고, 무선 인터넷을 이용할 수 있으며 자동차의 다양한 전자제어장치와 연동되어 자동차의 상태 및 직접 제어할 수 있도록 구성되어 있다. 하지만 인포테인먼트 시스템은 스마트 자동차의 핵심 부분일 수 있으나 무선 인터넷 등 다양한 보안 취약점을 가지고 있다.



[그림1] 크라이슬러 지프(Jeep) 체로키 네트워크 구성도, 참고문헌[5]

이 문제점을 해결하기 위해 크라이슬러는 인포테인먼트 시스템과 외부와의 접점을 최소화하며 펌웨어 업데이트를 USB커넥터를 이용한 접속을 통해서만 보안패치가 가능하도록 제한하였으며 그 피해 대상 차량 약 140만 대가 리

콜된 사례가 있다. 이러한 스마트 자동차의 사이버 보안위협에 대응하기 위해 북미, 유럽을 중심으로 세계 각국의 정부와 기관에서 적극적으로 보안 가이드라인과 규제사항들을 공표해 나가고 있고, 글로벌OEM社는 대내외적으로 모의해킹을 실시하면서 자동차의 보안성 강화를 내재화하고 있다.

본 논문은 스마트 자동차가 가지고 있는 다양한 보안 취약점 식별하고 더 나아가 자동차 내 전자제어장치의 보안 취약점 분석 방법과 해결방안을 제시한다.

II. 스마트 자동차 보안

2.1 스마트 자동차 보안취약점

스마트 자동차는 지능적이고 다양한 ICT 기술들이 적용되어 있다. 그러므로 스마트 자동차에 대한 사이버 공격을 받을 수 있는 가능성이 존재한다. 일반적으로 자동차를 세가지로 분류해보면 물리적인 움직임을 담당하는 구동부와 자동차의 엔진이나 변속기를 제어하는 전자제어장치(Electronic Control Unit, ECU), 차 안에서 외부를 연결시켜 주는 인포테인먼트(Infotainment) 시스템으로 구분된다. ECU는 CAN(Controller Area Network)를 통해서 제어가 가능하다. 이로 인해서 공격자는 CAN을 이용하여 ECU 영역을 침범해 자동차를 급제동을 가능케 하거나 브레이크 페달을 무력화시킬 수 있다. 인포테인먼트 부분에서는 자동차 내부의 네비게이션에 스마트폰 화면을 그대로 비춰주는 '미러 링크' 기술이 취약점으로 발견되었다. 또한 인포테인먼트 시스템 중 하나인 AVN(Audio, Video, Navigation) 시스템에서도 악성코드로 이루어진 미디어 파일, CD 등을 통해 취약점 공격이 가능하고 DMB, DAB, GPS 등 무선주파수를 활용한 해킹에도 취약하다. 또한 블루투스를 이용하여 외부로부터 전송된 파일들을 통해서 악성코드가 감염될 수 있다. 표1은 실제 자동차 보안 취약점을 진단했던 경험을 바탕으로 스마트 자동차에서 발생될 수 있는 보안 취약점을 분류하여 정리하였다.

[표1] 스마트 자동차의 주요 보안 취약점

구분	시스템 명칭	보안 취약점
전자제어장치(ECU)	자동차 엔진 제어 시스템	CAN 신호 변조 공격에 의한 급발진, 급가속, 급제동
	바퀴 회전 및 브레이크 제어 시스템	
	무선 장금 장치 시스템	무선 신호 해킹을 통한 차량 문 열림 제어
	타이어 공압 모니터 시스템	펌웨어 변조 등 타이어 공기압 값 조작에 의한 타이어 상태 변조
	기타 전자제어장치	자동차 자가 진단 시스템(OBD II)로 각종 ECU 접근 및 펌웨어 변조
인포테인먼트(Infotainment)	AVN (Multimedia /Navigation)	악성코드가 담긴 미디어 재생 공격을 통한 시스템 장악 후 ECU 연계 공격
	Telematics (원격 제어 시스템)	무선 모의해킹 공격에 인하여 원격 시동, 원격 문열림
	ADAS (운전자 지원 시스템)	CAN 신호 변조 등의 해킹 공격 전방추돌경보시스템 : 전방 차량 거리 조작 차선 이탈 방지 : 임의 핸들 조작
	DMB, DAB, GPS (디지털 방송 시스템 및 위치정보 시스템)	임의 무선 신호 생성 해킹 공격 (HackRF 등 이용 - 무선 신호 해킹) - 디지털 방송 데이터 임의 송출 : 익스플로잇 공격을 시스템 장악
	블루투스 장치	블루투스 취약점으로 임의의 주소록 데이터 획득 악성코드 내제된 파일 전송 후 설치/실행

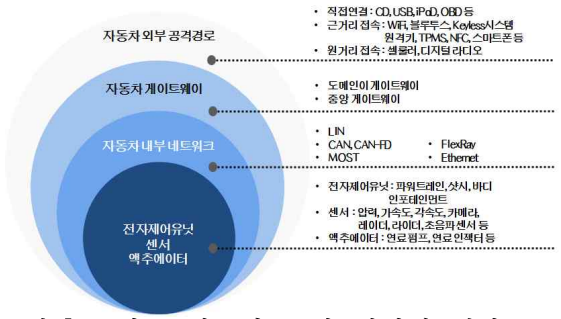
스마트 자동차의 보안 취약점은 ECU 중 자동차 사용자 편의 사양인 전방 추돌 방지 시스템 펌웨어 변조로 오동작하도록 조작하여 운전자의 생명과 직결되는 심각한 문제부터 각종 인포테인먼트 시스템에 다양한 방법으로 악성코드를 설치하여 운전자의 위치 추적, 탑승자의 대화내용 도감청 등 다양한 문제가 존재하므로 정확한 보안 취약점을 파악하고 보안 취약점에 대한 대응방법이 제시되어야 된다.

2.2 스마트 자동차 보안위협

스마트 자동차는 다양한 유무선 인터페이스 증가에 따라 확대된 공격 벡터를 통한 내부 네트워크 불법 침투의 가능성이 증대되고 있다.

보안 설계가 미흡한 내부 네트워크 프로토콜 특성을 이용한 임의 접근. 스푸핑 및 악의적 데이터 주입 등 공격 탐지의 한계성과 전자제어장치(ECU), 자동차에 탑재된 다양한 센서, 자동차 연료 관련 엑추에이터에 대한 다양한 물리 공격, 펌웨어 위변조, 메모리 공격 등을 통한 임베디드 장치의 오작동 유발 사례 증가 스마트 자동차에 대한 표준화된 보안성 시험/진단 시스템이 부재하고 사이버 공격에 의한 사고 발생 시 원인분석 방법이 명확하지 못한 것이 현실이다.

그림2는 스마트 자동차 보안 위협에 대한 공격경로를 나타낸 것으로 스마트 자동차가 얼마나 많은 보안위협을 가지고 있는 알 수 있다.

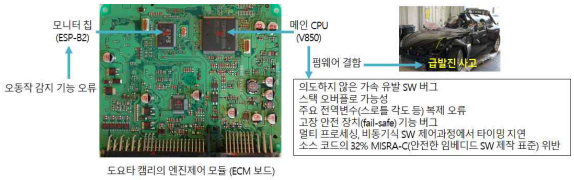


[그림2] 스마트 자동차 보안 위협에 대한 공격경로, 참고문헌[6]

III. 전자제어장치(ECU) 보안

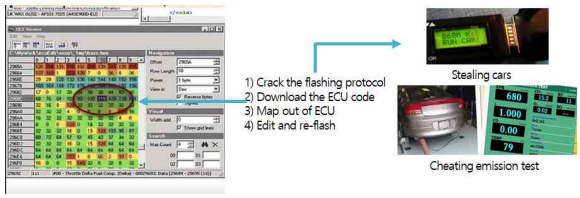
3.1 전자제어장치(ECU)의 보안위협

전자제어장치(ECU)의 보안위협은 대부분의 ECU는 임베디드 시스템으로 자원제약으로 인한 보안 설계가 미흡하여 불법 접근/권한상승, 인증우회, 펌웨어 변조, 위장 ECU 등을 통한 다양한 공격에 취약하다. 그림3은 도요타 캠리의 엔진제어 모듈 결함으로 급발진 사망사고가 발생한 사례이다.



[그림3] ECU 펌웨어 결함 사례, 참고문헌[6]

ECU 펌웨어 내 자체 결함 외 무작위 값 대립과 같은 반복적인 공격을 통해 ECU 접근제어 기능 무력화 및 권한 획득이 가능하며 ECU 플랫폼에 대한 버퍼 오버플로우, 메모리 오류 주입, 역공학 분석 등 ECU 메모리에 대한 공격으로 보안 위협이 존재한다. 그림4는 ECU 펌웨어 변조 사례로 악의적인 기능을 ECU에 삽입하거나 고유 기능을 바꾸기도 한다.



[그림4] ECU 펌웨어 변조 사례, 참고문헌[6]

추가적으로 인증되지 않는 위장 ECU를 자동차에 연결하여 자동차 내부 네트워크에 악의적인 기능을 수행하도록 할 수 있으며, ECU에는 다양한 센서와 연결하게 되는데 센서에 직접적으로 물리 공격을 통해 센서의 측정 값 오류 발생 및 이를 통한 자동차 기능 오작동 사고발생을 유도할 수 있다.

스마트 자동차는 기본적으로 차량고장 상황이나 내부 상태를 알기 위하여 진단도구를 연결하여 다양한 전자제어장치(ECU)의 상태 값을 확인하여 차량의 상태를 확인 가능하다. 이 때 사용되는 연결방법이 OBD-II 포트이다.

OBD-II 포트는 기본적으로 인증 절차 없이 접근 가능하도록 설계되어 악성코드에 감염된 자동차 보안·안전 진단 장비(전용장비, 노트북, 스마트폰 등)의 OBD-II 연결로 인해 자동차 내부로 악의적인 데이터 주입이 가능하다. OBD-II 포트로 연결되는 진단도구가 무선장치로 연결된 경우에는 원격 공격이 가능하다.

3.2 전자제어장치(ECU) 보안취약점 점검

ECU의 펌웨어 조작 취약점은 다양한 기능을 가지고 있는 각종 전자제어장치에 모두 적용되며 예를 들어 스마트 자동차의 주요 보안 취약점 중 타이어 공기압 모니터 시스템(TPMS)이

라는 전자제어장치의 펌웨어 조작이 성공한다면 타이어의 공기압 값 조작으로 자동차 운행 중 운전자에서 생명에 큰 위험을 초래할 수 있다. 펌웨어 조작 공격을 수행하는데 있어 필요한 세부 공격 과정은 자료수집, 디버그 포트 연결, 펌웨어 추출, 보안 기능 우회 등으로 진행된다.

ECU 보안 취약점 진단 절차

1. 하드웨어 분석 및 MCU 등 관련 칩 정보 수집
2. 디버그 포트 연결 및 펌웨어 추출
3. 펌웨어 역공학 분석
   (내부 데이터 / 함수 분석)
4. 보안 기능 및 펌웨어 리프로그래밍 절차 분석

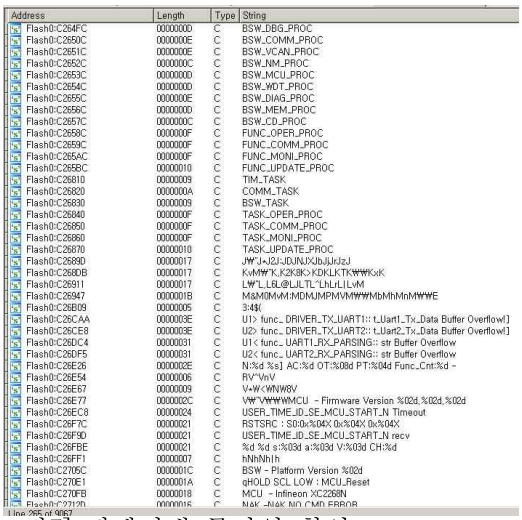
전자제어장치(ECU)의 보안 취약점 진단 절차와 더불어 표2와 같이 진단 항목을 분류하였다.

[표2] ECU 보안 취약점 진단 항목

분류	보안 취약점 진단 항목
하드웨어	1. 하드웨어 정보 노출
	2. 칩간 통신 스니핑
	3. 디버그 포트
	4. 외부 부트로더
	5. 외부 메모리 덤프
소프트웨어 (펌웨어)	6. 펌웨어 정보 노출
	7. 보안 기능 코드 노출
	8. 펌웨어 삭제 및 변조
	9. 코드 신뢰성
	10. 입출력 데이터 검증

3.3 전자제어장치(ECU) 펌웨어 분석방법

펌웨어는 바이너리로 구성되어 있기 때문에 펌웨어 내에서 취약점 정보를 바로 찾아내는 것은 어렵다. 개발자가 아닌 사람이 제어기 펌웨어 내에서 취약점을 찾기 위해서는 바이너리 ROM 데이터의 역어셈블, 함수 식별, 데이터 식별 등 여러 가지 전문적인 분석 과정이 요구된다. 그러나 펌웨어 내 그림5와 같이 디버그 정보나 문자열 정보가 그대로 바이너리에 포함된 경우, 이러한 분석 과정을 좀 더 용이하게 만들 수 있다. 해당 정보를 통해 시스템 기능이나 데이터 영역 등 여러 가지 유용한 정보를 추측해 낼 수 있고, 이러한 정보를 이용해 치명적 기능 관련 부분을 집중 분석하여 공격 시나리오를 구성할 수 있다. 예를 들면, 바이너리 분석으로 통해 획득한 CAN ID, UDS Service ID 등을 바탕으로 제어기에서 제공되는 서비스를 예상할 수 있으며, 퍼징 공격의 효율을 높이는 정보로 이용될 수 있다. 각 암호 알고리즘에서 고정적으로 사용하는 상수 값을 통해 제어기에서 어떤 암호 알고리즘을 사용하는지 유추할 수 있으며, 문자열을 통해 예외처리 메시지나 시스템 버전, 함수명 등이 노출되어 추가적인 펌웨어 분석에 사용될 수도 있다. 이러한 정보가 직접적으로 공격에 활용되지 않으나 공격자의 지식 수준과 경험에 의해 공격 시간을 단축시키는데 활용될 수 있다.



[그림5] 펌웨어내 문자열 확인

3.4 전자제어장치(ECU) 펌웨어 보안 방안

ECU 펌웨어의 보안을 강화하기 위하여 두가지 보안 방안을 제시한다. 첫 번째 보안 방안으로 주요 정보를 인지할 수 있는 문자열을 사용하지 않는 것이다. 디버그 관련 정보 및 중요 데이터 정보에 대한 문자열이 펌웨어에 포함되지 않도록 한다. 대부분의 펌웨어에서 문자열이 발견되지 않았으나, 일부 제어기는 디버그 기능

이나 소프트웨어 정보를 유추할 수 있는 문자열이 삽입되어 있다. 이런 문자열 중 내부 기능을 추측할 수 있다고 판단되는 중요 정보에 대해서는 노출되지 않도록 양산 시 반드시 제거하도록 해야 한다. 모든 정보가 공격자에게 유용하지는 않으나, 개발자 입장에서 단순하게 넘기는 정보도 때에 따라서 공격에 유용하게 활용될 수 있다 (예: 소프트웨어 버전, 날짜). 반드시 필요한 경우에는 각 상태마다 특정 코드를 붙여서 사용하고 코드 정보를 대외비로 관리한다.

두 번째 보안 방안으로 바이너리 코드 난독화하는 방법이 있다. 특정 상수나 문자열, 코드 흐름을 통해 펌웨어 기능을 유추해낼 수 있다. 이러한 역분석을 완전히 방지하는 것은 불가능하나 공격자로 하여금 시간을 소비하게 하거나, 어렵게 만들 수는 있다. 이러한 방법을 소위 난독화 기법이라고 부른다. 난독화 기법은 여러 가지가 있다. 데드 코드(dead code) 또는 더미 코드(dummy code)를 삽입하는 방법, 유사 기능 함수를 여러 개 만들어 여기저기 복사하는 방법, 관련 코드의 배치를 의도적으로 멀리 하는 방법, 비 관련 코드를 의도적으로 묶어 처리하는 방법, 코드를 RAM 영역에 복사, 실행하고 지우는 방법, 암호화 같은 인코딩(encoding)을 적용하고 필요한 경우만 디코딩(decoding)해서 사용하는 방법 등 다양하다. 하지만, 이러한 방법은 MISRA 검사나 중복 코드 테스트, 커버리지 테스트 검사와 같은 코드 신뢰성 품질 검사 시 문제가 생길 수 있으며, 코드 사이즈가 증가하거나, 개발자 가독성 문제, 소프트웨어 재사용률 저하, 성능 저하 문제 등이 발생할 수 있다. 보안적으로 중요한 코드나 데이터를 먼저 판별하고, 신중한 설계, 개발 시 이슈, 품질 관리, 사후 관리에 대한 대책 등 충분한 검토 후에 난독화를 적용해야 한다.

IV. 결론

본 논문에서는 스마트 자동차는 운전자의 별다른 조작 없이 스스로 목적지에 도착하는 자율주행 자동차로 진화되고 있다. 다른 사이버 보안 사고와 다르게 운전자 또는 탑승자의 생명까지 위협할 수 있어 사회에 큰 위협으로 다가올 수 있다. 그 동안 웹 어플리케이션 또는 앱, 리눅스 또는 안드로이드가 탑재된 임베디드 시스템에 대한 보안 취약점 점검 방법은 많이 나왔으나 본 논문에서는 MCU를 기반으로 하는 스마트 자동차의 전자제어장치(ECU)에 보안 취약점 점검 방법과 대응 방안에 대하여 제시하였다. 향후 전자제어장치의 보안 취약점 점검 방법에 대하여 좀 더 세분화하여 점검 표준에 대한 연구가 계속되어야 한다.

[참고문헌]

[1] 최성호, 이승형, 장보근, 박희주, 정기현 (2016) “스마트 자동차 네트워크 통신의 보안 취약점 분석 및 개선 연구”, 대한전자공학회 학술대회, 1819-1822.

[2] 김예지, 이영숙 (2017) “자율주행자동차의 취약점 및 보안 고려사항에 대한 연구”, 한국컴퓨터정보학회 학술발표논문집, 25(2), 165-168.

[3] 김진묵, 문정경, 황득영 (2018) “스마트 자동차 네트워크의 보안취약점 분석 및 해결방안 마련”, 융합보안눈문지, 18:3, 69-76.

[4] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces", USENIX Conference on Security, 2011.

[5] Dr. Charlie Miller, Chris Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle", DEF CON 23, 2015.

[6] 보고서, 과학기술정보통신부, 정보통신기술진흥센터, “지능형 자동차 보안 위협 및 대응 방안 보고서”, 2017.

[7] 한주엽, "[MCU강의②] MCU의 내부 구조와 동작 프로세스“, 2014. (http://www.ddaily.co.kr/news/article/?no=117571, 웹사이트 참고)

[8] 자동차 주요구성요소 ECU와 CAN 등의 통신기술(http://wiki.wikisecurity.net/issue:자동차_해킹_위협, 웹사이트 참고)

V2X환경에서의 차량 내 인증서 관리시스템 보안성 평가항목 연구*

김태경[1], 지청민[1], 김지민[1], 홍만표[2]

[1]아주대학교 컴퓨터공학과 [2]아주대학교 사이버보안학과

A Study on the Security Assessment Items of the In-Vehicle Certificate Management System in the V2X

Taekyung Kim[1], Cheongmin Ji[1], Jimin Kim[1], Manpyo Hong[2]

[1]Department of Computer, Ajou University, [2]Department of Cyber security, Ajou University.

요 약

4차 산업혁명과 자율주행이 가능한 차량의 개발로 도로 상에서 자율주행차량이 실제 주행을 위한 연구가 국내외에서 진행되고 있다. 차량과 차량, 차량과 인프라에 대한 연결을 위해 차량에서는 안전메시지(BSM)을 생성, 송신하여 차량과 차량의 충돌을 막고 교통상황 분석 등 도로상황 분석을 가능하게 한다. 주행환경의 특수성과 신속한 정보처리를 위해 안전메시지는 암호화가 되지 않은 상태에서 송신되어 안전성과 개인 정보보호 침해가 우려되고 있다. 이러한 취약점을 보완하기 위해 차량 내 인증서 관리시스템(SCMS) 도입하였는데 현재 차량 내 인증서 관리시스템의 보안성을 평가하기 위한 평가항목에 대한 연구가 필요하다.. 본문에서는 차량 내 인증서 관리시스템에 대한 취약점과 공격 유형을 분석하고 이를 기반으로 인증서 관리시스템의 보안성을 평가할 수 있는 항목을 제시하였다.

I. 서론

4차 산업혁명의 핵심인 IoT 기술의 발전으로 자동차는 기존의 운송수단에서 다양한 환경을 연결하는 디바이스로 진화 중이다. 이러한 자동차의 변화는 자동차 뿐 아니라 자동차 주행 환경의 변화를 가져오게 되었다. 친환경 기술 및 자율주행 기술을 기반으로 지능형 교통시스템의 연구도 지속적으로 수행중이다.

현재 개발 중인 자율주행 차량의 자율주행 수준을 미국자동차공학회(이하 SAE)에서 5단계로 정의하고 있고 미국도로교통안전국(이하 NHTSA)에서는 4단계로 정의한다. <표1>은 SAE에서 정의한 자율주행 단계이며 NHTSA와의 분류 차이는 단계 무인운전의 지원여부에 대한 차이이다.[1]

이러한 자율주행 기능을 원활히 수행하기 위해 차량에는 카메라, 레이더 라이더 등의 센서를 탑재하여 운영한다. 그러나 차량의 센서 의 인식 가능 범위가 넓고 오 인식으로 인한 문제를 방지하고자 차량은 상호 안전메시지(Basic Safety Message, 이하 BSM)를 발송하고 도로 인프라와의 상호 통신을 통해 도로조건과 교통상황 등을 제공하는 협력형 주행(Cooperative driving) 형태로 발전 중이다.[1][2]

<표 1> 자율주행 단계(SAE)

단계	기 능	특 징
1	단일주행 보조	- 운전 보조기능 탑재 운전자가 직접 운전
2	복수주행 / 융합 보조	- 2개 이상의 주행 기능이 융합보조 - 운전자는 주행상황 주시
3	제한된 자율주행	-특정 도로 조건에서 자율주행 가능 -운전자는 위급상황시 개입
4	완전 자율주행	모든 조건에서 자율주행 - 운전자 탑승
5	무인 완전 자율주행	- 모든 조건에서 자율주행 - 무인운전

차량의 안전을 위해 SAE-J2735 표준에서 v2X환경에서 사용되는 메시지 셋을 정의하고 있으며, 미국교통부(USDOT)은 2017년 1월 12일 NPRM(Notice of Proposed Rulemaking, 이하 NPRM)을 통해 해당

* 본 연구는 국토교통부 및 국토교통과학기술진흥원의 연구비지원(19PQOW-B152473-01)으로 수행된 연구임.

기술을 포함시켰다. 차량과 차량의 충돌 방지를 위해 초당 최대 10회까지 BSM이 송신되며 송신자의 시간, 위치, 속도, 경로이력 등의 정보를 포함하여 송신한다.[3]

이러한 BSM은 암호화되지 않았으므로 악의적인 공격자가 거짓 메시지를 삽입하거나 다른 차량의 메시지를 탈취 및 도용하여 공격할 수 있다. 또한 BSM에는 송신자의 위치 및 경로이력 등 식별정보를 분석하여 개인 사생활을 침해할 수 있는 위협요소가 존재한다.

Benedikt Brecht 등[4]은 연구에서 PKI를 기반으로 하는 인증서 관리 시스템(Security Credential Management System, 이하 SCMS)을 제안하였고 이를 기반으로 NHTSA와 차량 제조사 안전 컨소시엄(Crash Avoidance Metrics Partners LLC, 이하 CAMP)에서 기반 구축 및 검증을 진행하고 있다.

본 연구에서는 V2X 환경에서의 차량 내 인증서 관리시스템의 보안성을 위협하는 요소를 도출하고 차량의 보안성을 평가하기 위한 평가항목을 제안한다.

II. 관련연구

PKI를 기반으로 한 인증서 관리 시스템에 대한 표준은 IEEE 및 SAE에서 제정하였다. 이를 기반으로 미국은 미국교통국(USDOT) 산하 기관인 NHTSA와 FHWA에서 기반 구축을 위한 연구와 이를 평가하기 위한 테스트를 수행하고 있다.

2.1 관련 표준

IEEE 및 SAE에서는 V2X 환경에서 사용되는 차량의 통신규격 및 보안에 대한 표준을 <그림1>과 같이 제정하고 있다.

이 중 차량 보안과 관련된 표준은 IEEE 1609.2[6]이며 차량 및 인프라에서 생성 및 송신하는 메시지의 표준은 SAE-J2735를 따른다.

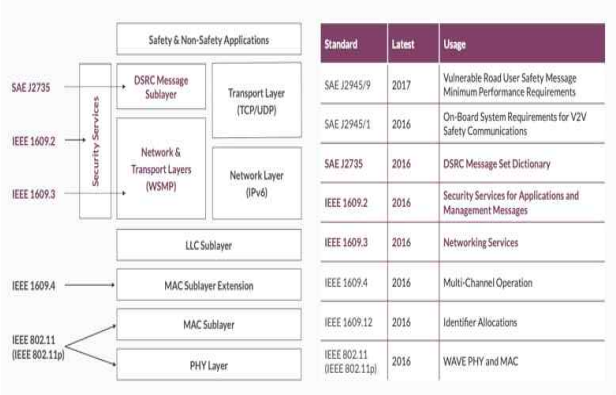
2.2 USDOT

미국교통국 산하 NHTSA는 NPRM[3]을 통해 자율주행 차량 중 경량차에 대한 차량과 차량간 통신(V2V) 메시지와 형식을 표준화하기 위한 기술제안을 하였고 이를 통해 연방 자동차 안전 표준(Federal Motor Vehicle Safety Standard, 이하 FMVSS) No.150을 제정하였다.

CAMP에서는 CAMP WIKI[7]를 통해 지속적으로 시스템과 차량에 대한 요구사항을 테스트를 통해 연구를 진행하고 있다.

NHTSA에서는 SCMS 검증을 위해 정보컨설팅 기업인 BAH(Booz Allen Hamilton, 이하 BAH)에게 V2X 환경에서 사용하는 통신 규격 중 하나인 DSRC(Dedicated Short Range Communications, 이하 DSRC) 시스템의 작동을 보장하기 위한 통신성능 및 보안 요구사항을 테스트를 통해 권장사항을 제시하였다.[8]

<그림 1> V2X 관련 표준



III.취약점 분석

3.1 평가 대상 및 범위

보안성 평가를 위한 대상은 자율주행 기술이 도입된 차량이며 BSM을 생성 및 송신할 수 있는 기반 장치인 OBU(On Board Unit, 이하 OBU)를 탑재한 차량이다.

평가 범위는 자율주행차량이 주행을 위해 등록인증서(Enrollment Certificate) 및 익명인증서(Pseudonym Certificate)를 발급하고 사용하는 것에서부터 비정상 행동을 탐지하여 보고하는 절차까지 선정하여 평가 요소를 분석 및 평가항목을 도출하였다.

3.2 예상되는 취약점

William Whyte 등[5]은 연구에서 SCMS 환경에서 예상되는 취약점을 다음과 같은 3가지로 분류하였다.

- 외부에서 사용자 개인 정보에 대한 공격
- 내부에서 사용자 개인 정보에 대한 공격
- 거짓 경고로 이어지는 인증 메시지

발생 가능한 공격 유형으로는 메시지를 위조 및 변조하여 차량 주행 정보에 혼란을 주어 사고를 유발하는 공격과 대량으로 가짜 메시지를 생성하여 차량 기능을 정지시키거나 오작동을 유발하는 공격이 있다. 또한 BSM을 수집하여 마다 생성되는 임시ID를 사용하는데 임시ID를 통해 차량을 특정하여 주행 정보를 수집하는 개인정보 침해가 발생할 수 있다.[8]

IV. 보안성 평가항목

V2X 환경에서 차량 내 인증서 관리시스템 보안성을 평가하기 위해 3절을 기초로 항목을 도출하였고 이를 유형별로 정리하여 평가방법을 수립해 보았다.

4.1 평가항목 도출

인증서 보안성을 평가하기 위해 CAMP WIKI에서 명시한 Use-case를 기반으로 차량의 등록인증서 및

익명인증서의 발급 및 폐지, 사용과 같은 요구사항을 수집하였다.

또한 3.2에서 예상한 취약점을 대비하는 방법들과 구성요소의 요구사항을 만족할 수 있는 항목들을 여러 참고자료에 명시된 요구사항을 수집하였다.

4.2 평가항목 분류

평가항목은 관련된 연구와 차량의 보안성을 검증하기 위한 항목을 연관시켜 다음과 같이 분류하였다.

- 저장능력 평가
- 인증서 발급절차 준수 평가
- 인증서 사용 평가
- 비정상행동 탐지 및 보고 평가

4.2 평가방법

평가방법은 완성된 임의의 자율주행차량을 대상으로 해당 평가항목을 검증하기 위한 방법을 의미하며 다음과 같다.

- 해당 기능 및 부속에 대한 외부기관 인증
- 정상 절차 수행을 통한 평가
- 특정 조건 주입을 통한 평가

외부기관 인증은 표준에서 요구하는 보안 모듈과 저장소에 대한 역공학이나 분해가 불가하기 때문에 해당 항목에 대해서는 CC인증이나 FIPS 140-2인증과 같은 외부기관의 인증을 필요로 한다.

정상 절차 수행을 통한 평가는 각 Use-Case에서 수행한 절차를 완료하면 정상적으로 BSM이 송신되는데 유효한 인증서가 식별되면 정상으로 평가 가능하다.

특정 조건 주입을 통한 평가는 비정상 행동을 하는 차량의 메시지를 송신(하거나 유효하지 않은 인증서를 송신하는 경우, Use-Case 수행 간 오류조건을 조성하여 해당 항목을 평가하는 것을 의미한다.

4.3 평가항목

평가항목은 <표2>와 같이 24개의 항목으로 도출하였으며 이를 4.1과 4.2를 참조하여 유형을 약식으로 분류하였다.* 관련된 인증서는 등록인증서 혹은 익명인증서가 관련되었는지 여부를 확인하기 위해 명시하였다.

<표2> 차량 인증서 관리시스템 평가항목

유형	평가항목	관련 인증서	평가 방법
1	OBU는 FIPS 140-2 Level 2 이상의 저장장치에 private keys, 인증서 등과 같은 제한된 정보를 안전하게 저장할 수 있어야 한다.	등록	외부 인증
1	OBU는 최소 하나의 CRL, 3년치 익명 인증서를 주당 20개 저장할 수 있는 용량을 가져야 한다.	등록 익명	상황 조성
1	OBU는 제한된 정보를 보호 메모리 내에서만 사용해야 한다.	등록 익명	외부 인증
1	OBU는 로컬 인증서 체인 파일을 변조 방지 또는 이와 동등한 성능을 가진 저장소에 저장할 수 있어야 한다.	등록 익명	외부 인증
1	OBU는 1600KB의 Misbehavior Report를 저장하기 위해 충분한 영구 메모리 저장장치를 할당해야 한다.	등록 익명	상황 조성
1	OBU는 신뢰성 수준 레벨 1에 대해 지속적으로 실패한 차량의 연속 BSM메시지를 10세트 기록할 수 있어야 한다.	익명	외부 인증
2	등록인증서 생성 요청을 위한 응답 파일을 작성 및 압축하여 전송할 수 있어야 한다.	등록	절차 수행
2	OBU는 RA와의 통신을 설정한 후 업데이트된 Local Certificate Chain File을 확인해야 한다.	등록 익명	상황 조성
2	Hash 알고리즘으로 오직 SHA-256만을 사용하여야 한다.	등록 익명	상황 조성
2	Public Key 암호 알고리즘은 오직 ECIES를 사용하여야 한다.	등록 익명	상황 조성
2	대칭 알고리즘은 오	등록	상황

* 평가유형은 4.1 분류 순서대로 번호로 표기하였으며 평가방법은 외부인증, 절차수행, 상황조성으로 약식 표기하였다.

	직 AES-CCM을 사용하여야 한다.	익명	조성
3	OBU는 최소 10개의 BMS/sec를 생성하고 전송할 수 있어야 한다.	익명	절차 수행
3	식별 방지를 위해 일정 100분 동안 식별된 인증서가 최소 20개여야 한다.	익명	상황 조성
3	OBU가 잘못된 BSM이 생성될 수 있는 고장을 감지하면 BSM 전송을 중단해야 한다.	익명	상황 조성
3	익명 인증서는 식별 방지를 위해 주기적으로 다른 익명 인증서로 교체할 수 있어야 한다.	익명	상황 조성
3	익명 인증서는 식별 방지를 위해 주기적으로 다른 익명 인증서로 교체할 수 있어야 한다.	익명	상황 조성
3	BSM 내 임시 ID는 익명인증서 교체와 동시에 변경되어야 한다.	익명	상황 조성
4	OBU는 검증이 필요하다고 경고 발생되는 수신된 모든 메시지를 확인할 수 있어야 한다.	익명	상황 조성
4	Misbehavior Report는 20주 이상 보관할 수 있어야 한다.	등록 익명	상황 조성
4	용량이 초과되어도 새로운 보고서의 저장이 이루어질 수 있어야 한다.	등록 익명	상황 조성
4	보고서를 암호화되지 않은 저장매체에 저장하려면 내용을 암호화 할 수 있어야 한다.	등록 익명	외부 인증
4	로컬에서 탐지된 잘못된 장치의 인증서가 저장된 CRL에 있는 경우에는 MA로 다시 보고하지 않아야 한다.	등록 익명	상황 조성
4	OBU는 침입 탐지 활동을 수행해야 하며 침입으로 탐지된 메시지를 잘못된 것으로 표시해야 한다.	익명	상황 조성
4	OBU는 연결이 가능한 경우 저장된 Misbehavior Report를 MA로 보낼 수 있어야 한다.	익명	상황 조성

V.결론 및 향후 연구

본 연구에서는 V2X환경에서의 차량 내 인증서 관리시스템이 보안성을 갖기 위한 요구사항을 도출하였고 이를 기반으로 차량에 평가 가능한 항목을 제시하였다.

향후에는 이 평가항목을 기반으로 한 평가기술에 대한 연구가 진행되어야 한다. 이를 통해 자율주행 차량의 개발과 검증에 기여할 것이다.

[참고문헌]

[1] 권혁찬 외 5명, “자율주행 자동차 보안기술 동향”, 한국전자통신연구원 전자통신 동향분석 Vol.33.No.1, 2018

[2] 최종찬, “자율주행자동차 V2X 통신 표준화 현황”, 한국표준협회, 2019

[3] U.S. Department of Transportation - National; V2V Communications; NPRM.

[4] Benedikt Brecht, Dean Therriault, Andre Weimerskirch, William Whyte, Virendra Kumar, Thorsten Hehn, Roy Goudy, “A security credential management system for V2X communications”, IEEE Trans. Intell. Transp. Syst. 99, 1 - 22 (2018)

[5] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, “A security credential management system for V2V communications,” inProc. IEEEVeh. Netw. Conf., Boston, MA, USA, Dec. 2013, pp. 1 - 8.

[6] IEEE. std 1609.2, 2016

[7] CAMP WIKI, “https://wiki.campllc.org”

[8] U.S. Department of Transportation - National; FHWA-JPO-17-483, “Development of DSRC Device and Communication System Performance Measures” 2016

V2V 통신 안전성 평가를 위한 DoS 공격 연구 동향

이숭영* 지청민* 김지민* 홍만표**

*아주대학교 컴퓨터공학과 **사이버보안학과

A Survey on Research Trend of DoS Attack for V2V Communication Safety Evaluation

Sung-Young Lee* Cheong-Min Ji* Ji-Min Kim* Man-Pyo Hong**

*Dept. of Computer Engineering, **Cyber security, Ajou University.

요 약

최근 큰 관심을 받는 자율주행환경은 외부 네트워크와 연결이 되기 때문에 외부의 위협이 차량에 침투할 수 있다. 차량은 이동하는 특성으로 인해 실시간으로 메시지를 처리하는 것이 중요하다. 실시간으로 메시지를 처리하기 위해서는 가용성이 중요한 요소이기 때문에 가용성을 위협하는 DoS 공격은 자율주행환경에 치명적이다. 본 논문은 자율주행환경에서 차량 간 통신을 위해 사용되는 WAVE 프로토콜을 살펴보고 이와 관련된 프로토콜에서 발생 가능한 DoS 공격 연구 동향을 살펴본다.

I. 서론

최근 자율주행 기술의 발달로 자동차의 개념이 단순 이동수단이 아닌 이동성을 확보한 생활 공간으로 변화하고 있다. 자율주행 기술은 운전자의 운전 부담을 줄이고 차내에서의 생산이나 휴식을 확보하는 등 삶의 질 개선에 이바지할 것으로 예상이 되어 많은 관심을 받고 있다. 자율주행 자동차는 차량에 장착된 다양한 센서를 사용하여 기본적인 자율주행 기능을 수행할 수 있다. 하지만 더욱 완성도 높은 자율주행을 위해서는 통신이 매우 중요하다. 자율주행 자동차는 통신을 통해 다양한 곳과 연결되어 자율주행에 도움이 되거나 필요한 서비스를 받을 수 있다. 예를 들어, 사전에 통신을 통해 도로정보, 교통신호 정보, 전방 충돌 및 사고 정보를 받는다면 더욱 안전하고 효율적인 자율주행이 가능하다[1].

자율주행환경에서는 차량 간 통신을 수행하기 위해 WAVE(Wireless Access for Vehicular Environments)를 사용한다. WAVE는 차량 간 통신을 통해 외부 네트워크와 연결 하여 다양한 서비스를 제공한다. 하지만 동시에 외부 네트워크에 존재하던 사이버 위협이 차량에서도 발생 가능하다. 특히 차량 환경의 경우 사이버 위협이 발생하게 되면 이는 단순히 자산의 손실뿐만이 아닌 인명피해와도 직결되므로 차량 간 통신에서의 보안에 각별한 관심이 요구된다. 또한 자율주행에서는 송수신되는 메시지에 대해 실시간으로 처리가 이루어져야 하므로 가용성을 확보하는 것이 매우 중요하다. 기존 네트워크 환경에서 가용성을 위협하는 대표적인 공격은 DoS 공격이 있다. 본 논문에서는 차량 간 통신에 사용되는 WAVE를 살펴보고 WAVE에서 사용되는 프로토콜과 관련된 DoS 공격 연구 동향에 대해 살펴본다. 본 논문의 구성은 다음과 같다. Ⅱ장에서는 WAVE와 DoS 공격에 대한 관련 연구를 소개하고, Ⅲ장에서는 DoS 공격 관련 연구에서 소개된 공격을 분류한다. 마지막으로 Ⅳ장에서 결론 및 추후연구를 제시한다.

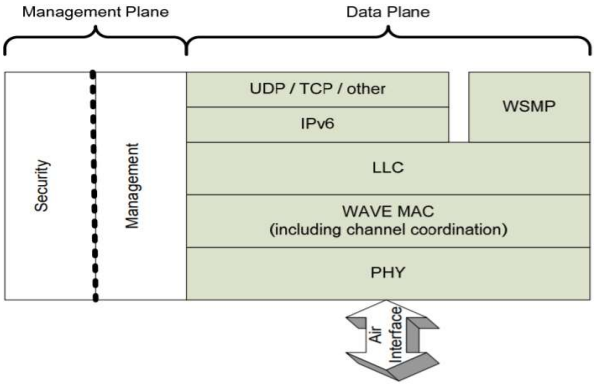
II. 관련연구

본 장에서는 WAVE와 WAVE와 관련된 DoS 공격 및 분류방법에 대한 연구에 관해서 설명한다.

2.1 WAVE

WAVE는 ITS(Intelligent Transportation System) 환경에서 원활하고 상호운용 가능한 서비스를 제공하기 위한 무선통신 시스템이다. WAVE와 관련된 표준은 IEEE 1609와 IEEE 802.11p가 있다. WAVE 프로토콜 스택은 아래 (그림 1)과 같다. IEEE 1609는 WAVE 아키텍처를 설명하는 IEEE 1609.0, WAVE 식별자 할당 기능을 설명하는 IEEE 1609.12, WAVE

애플리케이션과 관리 메시지 보안 서비스를 설명하는 IEEE 1609.2, WAVE 네트워킹 서비스를 위한 IEEE 1609.3, 다중 채널 기능을 설명하는 IEEE 1609.4로 구성되어 있다[2]. IEEE 802.11p는 WAVE를 위한 물리계층과 MAC 계층을 정의한다. 본 논문에서는 WAVE 네트워킹 시스템과 연관된 IEEE 1609.3에 대해 중점적으로 설명한다.



(그림 1) WAVE 프로토콜 스택 [1]

IEEE 1609.3은 무선차량 환경에서 최적화된 작동을 위해 설계된 표준 인터넷 프로토콜 버전 6(IPv6)과 WSMP(WAVE Short Message Protocol)을 정의하는 표준이다[3]. IPv6에서 전송계층으로 사용되는 프로토콜에는 TCP 및 UDP 등이 있다. TCP는 포트 번호 주소 지정 및 종단 간 신뢰성 및 선택적 재전송 기능을 제공하고 UDP는 포트 번호 주소 지정 및 WSMP에서 제공되지 않는 체크섬 기능을 제공한다. 또 WSMP와 IPv6 메시지를 구분하기 위해서 LLC 헤더에 있는 2-octet 필드에 해당하는 Ethertype 필드가 사용된다.

2.2 IPv6 DoS 공격 연구 동향

WAVE는 IP 기반 애플리케이션 서비스를 위해 IPv6를 사용하고, 전송계층으로는 TCP, UDP 등을 사용한다. Schuba C. L. 등은 TCP 프로토콜에서 사용되는 Three-way Handshake를 악용한 TCP-SYN Flooding 공격에 대한 분석과 대응방안을 소개한다[4]. 소스와 목적지 간 TCP 연결과정은 다음과 같다. 소스는 목적지에 연결 요청을 전송하고, 연결 요청을 받은 목적지는 자원을 할당하여 half-open 연결을 진행한다. 이 때 TCP는 지정된 시간 동안 포트당 가능한 half-open 연결의 수를 제한하기 때문에 목적지의 half-open 연결의 수가 한계에 다다르게 되면 이후 연결 시도를 거부한다. 공격자는 위와 같은 TCP의 half-open 특성을 악용하여 공격을 수행하는데, 먼저 다량의 연결 요청을 대상에 전송한다. SYN 요청을 받은 대상은 SYN+ACK 메시지를 공격자에 전송한다. 공격자는 연결을 끝내지 않은 채 계속해서 SYN 연결 요청을 공격 대상에게 전송해 DoS 공격을 수행한다. Mirkovic J. 등은 DDoS 공격과 방어 메커니즘에 대한 분류 방법을 제시한다[5]. 해당 연구에서는 DDoS 공격을 DA(Degree of Automation), EW(Exploited Weakness to Deny Service), SAV(Source Address Validity), ARD(Attack Rate Dynamics), PC(Possibility of Characterization), PAS(Persistence of Agent Set), VT(Victim Type), IV(Impact on the Victim)에 따라 분류하고 AL(Activity Level), CD(Cooperation Degree), DL(Deployment Location),에 따라 DDoS 방어 메커니즘을 분류한다. 가령 IPv6와 관련된 공격인 TCP, UDP Flooding 공격은 (표 1)과 같이 분류된다. TCP Flooding은 EW에서 프로토콜

(표 1) TCP, UDP Flooding 공격 분류 [5]

공격 분류	세부 항목
EW (Exploited Weakness to Deny Service)	Semantic
	Brute-force
PC (Possibility of Characterization)	Characterizable
	Non-Characterizable
VT (Victim Type)	Application
	Host
	Resource
	Network
	Infrastructure
IV:PDR (Impact on Victim: Possibility of Dynamic Recovery)	Self-Recoverable
	Human-Recoverable
	Non-Recoverable

이나 애플리케이션의 특정 기능을 악용하는 Semantic에 해당하고, PC에서 특정 프로토콜을 대상으로 하는 Characterizable에 해당한다. 그리고 VT는에서호스트를 대상으로 하는 host 분류에 속한다. UDP Flooding은 PC의 Characterizable, IV의 PDR(Possibility of Dynamic Recovery)에서 정상상태로 복구될 때 사람의 도움이 필요 없는 Self-Recoverable 분류에 속한다. Zargar, S. T. 등은 DDoS 공격의 주요 방법을 기형 패킷을 전송하여 혼란을 일으키는 방법과 Flooding 공격을 사용하여 자원 및 대역폭을 고갈시키는 방법으로 분류한다. 그리고 Flooding 공격을 중점으로 DDoS 공격을 설명하는데[6], 해당 연구에서 분류한 공격을 (표 2)에 나타내었다. TCP 및 UDP에

(표 2) DDoS Flooding 공격 분류 [6]

Layer	Attack
Transport Network	Flooding
	Protocol Exploitation Flooding
	Reflection-based Flooding
	Amplification-based Flooding
Application	Reflection/Amplification -based Flooding
	HTTP Flooding

대한 DoS 공격의 또 다른 종류로 분할된 패킷을 전송할 때 악의적으로 크기를 조절하여 DoS를 일으키는 Teardrop 공격이 존재한다[7]. 공격자는 분할된

패킷의 오프셋을 조절하여 패킷들을 서로 겹치도록 하여 대상에 전송하는데, 이렇게 되면 공격 대상은 분할된 패킷을 재조립하지 못하게 되어 DoS가 발생하게 된다. 한편 TCP와 UDP 같은 전송계층이 아닌 IPv6의 NDP(Neighbor Discovery Protocol)를 악용한 공격도 존재한다[8][9]. IPv6 NDP에는 같은 링크에 주소 충돌이 발생하지 않도록 하는 중복 주소 검출인 DAD(Duplicate Address Detection) 기능이 있다. DAD 과정은 처음 활성화된 호스트가 링크 로컬 주소를 생성할 때 이루어지는데, 호스트는 인접 노드를 대상으로 자신의 출발지 주소는 미정으로 하고 목적지 주소를 자신이 사용할 주소로 지정하여 인접한 노드에서 중복된 주소를 사용하는지 확인한다. 공격자는 이 DAD 메시지를 악용하여 다른 노드가 주소를 얻지 못하도록 하는 DoS 공격을 발생시킬 수 있다. DAD 대상 Dos공격에서 악의적인 노드는 자신에게 들어오는 모든 DAD에 대해 자신이 해당 주소를 가지고 있다는 스푸핑 된 응답을 전송하여 합법적인 호스트가 새로운 주소를 얻지 못하게 한다.

2.3 WSMP DoS 공격 연구 동향

WAVE는 non-IP 기반 애플리케이션 서비스를 위해 WSMP를 사용는데, WSMP는 네트워크와 전송계층 프로토콜을 모두 지원한다[2]. Kaur, M. 등은 차량 네트워크에서 발생 가능한 위협을 완화하기 위해 사용할 수 있는 보안 메커니즘과 잠재적인 보안 위협 및 취약점에 대한 연구를 수행하였다[10]. 해당 연구에 따르면 V2V와 V2I 환경에서의 위협은 애플리케이션 수준과 기반시설 수준 두 가지로 분류되며, 이 중 기반시설 수준에서 WAVE BSM을 대상으로 DoS 공격이 발생할 수 있다. WAVE 환경에서 차량은 주기적으로 브로드캐스트 되는 BSM을 통해 V2V 통신을 수행하는데. 이를 위해 WAVE는 IEEE 1609.4에서 기술하는 다중 채널 조정 기능을 사용한다. 이 기능에서 전송 시간은 제어 채널 간격과 서비스 채널 간격으로 나뉜다. 각 간격의 시작에는 보호 간격이 있으며, 보호 간격에서는 전송을 허용하지 않는다. 공격자는 이러한 채널 동기화 속성을 악용하여 채널 혼란을 일으켜 DoS 공격을 발생시킬 수 있다. Li, Y. 등은 V2X 네트워크에 MEC(Mobile Edge Computing)가 적용된 환경에서 발생 가능한 새로운 형태의 DoS 공격과 이를 탐지하고 방어하는 방법에 대한 연구를 수행하였다[11]. 해당 연구에서 제시하는 환경에서 공격자는 전방 충돌 경고, 차량 군집, 정보 공유 등과 같은 우선순위가 높은 서비스를 자주 요청한다. 그 결과로 공격자가 자원을 선점함으로써 다른 차량은 자원을 할당받지 못하게 되어 DoS가 발생한다. Whyte, W 등은 WAVE에서 사용되는 WSA (WAVE Service Advertisement)에 대한 위협 분석과 현존하는 보안 취약점의 대응방안을 소개한다[10]. 해당 연구에서는 WSA에 대한 보안 위협을 가용성에 대한 위협과 프라이버시에 대한 위협으로 분류하는데, 본 논문에서는 가용성에 대한 위협을 중점적으로 기술한다. 가용성에 대한 위협은 Misconfiguration과 Force-multiplier denial of service로 분류된다. Misconfiguration 위협은 장치 재구성의 취약점을 악용하는데 Country String 공격과 2D Location 공격으로 다시 분류된다. County String은 시스템이 작동하는 규제영역과 관련되는데, IEEE 1609.3에서는 Country String을 제공자에 대한 정보 혹은 사용자를 재구성하기 위한 정보로 사용한다. 이 중 취약점이 존재하는 상황은 Country String이 사용자를 재구성하기 위한 정보로 사용될 때이다. 2D Location는 WSA 송신 안테나의 위치를 제공하기 위해 사용되는 정보인데, 2D Location 또한 Country String과 유사하게 채널 매핑이 잘못되면 시스템이 정상적으로 작동하지 않는 위험이 있다. 다음으로 Force-multiplier denial of service 위협은 비교적 적은 양의 데이터를 전송하여 채널에 많은 트래픽을 발생시켜 유효한 서비스를 받거나 소비하지 못하게 하는 공격이다. 해당 위협에 대한 예시로 불법 서비스를 통한 공격이 있다. 특정 채널에 대한 접근 규칙이 있다고 가정할 때, 악의적인 공급자는 주어진 채널에서 광고되지 않아야 할 서비스를 광고한다. 이 때 사용자가 해당 서비스에 연결하는 경우 Force-multiplier 위협이 발생할 수 있다.

III. WAVE 환경에서의 DoS 공격 분류

본 논문에서는 2장에서 살펴본 연구 동향을 바탕으로 WAVE를 사용한 V2V 통신환경에서 발생 가능한 DoS 공격유형을 (표 3)과 같이 분류한다. 각 공격 별로 공격방법, 관련 프로토콜, 공격 대상, 증상을 나타냈다. DoS on DAD는 IPv6의 NDP 특성을 악용한 공격이다. Spoofing을 통해 공격하고 대상 네트워크의 패킷을 손실시켜 DoS 공격을 발생시킨다. TCP Flooding는 TCP SYN, ACK, SYN-ACK, RST, FIN Flooding 공격으로 분류되는데. 공격자는 Flooding을 통해 대상 서버의 패킷을 손실시켜 DoS 공격을 발생시킨다. Teardrop 공격도 TCP와 관련된 DoS 공격인데, 공격자는 패킷을 전송할 때 분할하는 방법을 악용하여 DoS 공격을 발생시킨다. UDP Flooding 공격은 UDP 메시지를 대량전송하여 대상 서버의 패킷을 손실시키는 DoS 공격이다. WSMP에 대한 공격 사례들은 다음과 같다. 먼저 BSM의 채널 동기화를 악용한 WAVE BSM DoS 공격이 있다. 해당 공격은 BSM을 전송하기 위한 WSMP 프로토콜을 대상으로 하며, 서비스 채널과 관리 채널 간격의 동기화를 악용하여 공격을 수행한다. Reserve Resource는 우선순위가 높은 서비스를 자주 요청하여 자원을 선점하여 고갈시키는 DoS 공격이다. Misconfiguration과 Force-multiplier은 WAVE에서 사용되는 WSA 메시지의 취약점을 악용한 공격이다. Misconfiguration은 악의적인 설정으로 대상 시스템을 정상적으로 작동하지 못하게 하며, Force-multiplier는 접근 정책을 악용하여 서비스의 DoS를 발생시키는 공격이다.

IV. 결론 및 추후연구

본 논문에서는 WAVE에서 사용되는 IPv6와 WSMP 계층에서 발생 가능한 DoS 공격에 관해 살펴보

(표 3) V2V 통신환경에서 발생 가능한 DoS 공격유형

주요 공격		공격 방법	관련 프로토콜	공격 대상	증상
DoS on DAD		Spoofing	IPv6 DAD	Network	패킷 손실
TCP Flooding	SYN	Flooding	TCP	Target Server	패킷 손실
	ACK				
	SYN+ACK				
	FIN				
	RST				
Teardrop Attack		Exploit packet Fragment	TCP	Target Server	패킷 손실
UDP Flooding		Flooding	UDP	Target Server	패킷 손실
WAVE BSM		Exploit channel synchronization	WSMP	Channel	시간 지연
Reserve Resource		Flooding	WSMP	Channel	패킷 손실
Misconfiguration		Exploit configuration synchronization	WSA	Target System	패킷 손실
Force-multiplier		Exploit access policy	WSA	Service	패킷 손실

았다. 본 논문에서 조사한 관련 연구 중 WSMP를 제외한 IPv6 대상 DoS 공격들은 모두 차량 네트워크보다는 기존 네트워크 환경을 대상으로 연구가 진행되었다. IPv6의 경우 기존 네트워크와 차량 네트워크가 완벽하게 같지 않기에 기존 네트워크에서 고려되지 않는 다른 공격이 존재할 수 있으며 이미 존재하는 공격의 경우도 차량 간 통신에는 적합하지 않을 수 있다. 공격뿐만 아니라 기존 연구들에서 소개하는 방어 메커니즘도 차량 간 통신에는 적용되지 않을 수 있다. 그렇기 때문에 차량 간 통신에서 발생 가능한 DoS 공격을 식별하고 방어할 수 있는 방안에 대한 추가적인 연구가 필요할 것으로 판단된다.

V. Acknowledgement

본 연구는 국토교통부 및 국토교통과학기술진흥원의 연구비지원(19PQOW-B152473-01)으로 수행된 연구임.

[참고문헌]

[1] 자율주행과 V2X 통신 기술 동향, 정보통신기술진흥센터, 주간기술동향 1820호, 2017
[2] IEEE Guide for Wireless Access in Vehicular Environments (WAVE)-Architecture, IEEE Standard 1609.0-2013, 2013.
[3] IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Networking Services, IEEE Standard 1609.3-2016, 2016
[4] Schuba, Christoph L. et al. Analysis of a denial of service attack on TCP, IEEE Symposium on Security and Privacy, 1997.
[5] Mirkovic, Jelena, and Peter Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms, ACM SIGCOMM Computer Communication Review, 2004.
[6] Zargar, Saman Taghavi, James Joshi, and David Tipper, A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks, IEEE communications surveys & tutorials, 2013.
[7] Wankhede, Sonali B, Study of network-based DoS attacks, Nanoelectronics, circuits and communication systems. Springer, Singapore, 2019.
[8] AlSa'deh, Ahmad, and Christoph Meinel, Secure neighbor discovery: Review, challenges, perspectives, and recommendations, IEEE Security & Privacy, 2012.
[9] Ahmed, Amjed Sid Ahmed Mohamed Sid, Rosilah Hassan, and Nor Effendy Othman. "IPv6 neighbor discovery protocol specifications, threats and countermeasures: a survey." IEEE Access 5 (2017): 18187-18210.
[10] Kaur, Manveen, James Martin, and Hongxin Hu, Comprehensive view of security practices in vehicular networks, 2016 International Conference on Connected Vehicles and Expo (ICCVE), 2016.
[11] Li, Yang, et al, An MEC-Based DoS Attack Detection Mechanism for C-V2X Networks, 2018 IEEE Global Communications Conference(GLOBECOM), 2018.
[12] Whyte, William, et al, Threat and Countermeasures Analysis for WAVE Service Advertisement, 2015 IEEE 18th International Conference on Intelligent Transportation Systems. 2015.

STRIDE 위협 모델링 기법을 이용한 Digital Signage 플랫폼 취약점 분석 및 대응방안

홍현경, 신현욱, 고준영, 권동현, 소찬영, 김지섭, 심기욱, 한철규
한국정보기술연구원 차세대 보안리더 양성 프로그램(Best of the Best)

Digital Signage Platform Vulnerability Assessment Using STRIDE Threat Modeling Technique

Hyeon-Kyeong Hong, Hyun-Wook Shin, Jun-Yeong Ko, Dong-Hyun Kwon, Chan-Young So, Ji-Sub Kim, Ki-Wook Shim, Cheol-Gyu Han
Best Of the Best, Korea Information Technology Research Institute.

요 약

Digital Signage란 사람들과의 양방향 커뮤니케이션이 가능한 인터넷 기반의 광고용 디스플레이로서 IoT 원리로 구동된다. Digital Signage는 다양한 분야에서 광고를 제공하는 용도로 쓰이고 있으며 세계적으로 시장규모가 꾸준히 증가하고 있다. 하지만 이를 악용한 사이버 공격이 지속적으로 발생하고 있으나 관련 보안 연구는 상대적으로 미흡한 실정이다. 본 논문에서는 STRIDE 위협 모델링 기법을 이용해 Digital Signage 플랫폼에 발생할 수 있는 취약점 도출 및 대응방안을 제시하여 안전한 Digital Signage 사용을 도모하고자 한다.

I. 서론

Digital Signage는 주로 공공, 상업공간에 설치되어 광고를 제공해주고 있으며, 네트워크를 통해 원격제어가 가능한 디스플레이다.[1] IoT의 원리로 구동되며 교통, 쇼핑, 교육 등 다양한 분야에서 활용되고 있다.[2] 단순하게 광고만 보여줄 수 있었던 기존 디스플레이의 역할을 넘어 현재는 양방향 커뮤니케이션이 가능한 디스플레이로 발전하였다.

2019년도에 발행된 'Digital Signage Market Size Analysis Report'에 따르면 Global Digital Signage의 시장 규모는 USD 185억 5천만 달러에서 2025년 USD 317억 1천만 달러까지 증가할 것으로 예상된다.[3] 실생활 곳곳에서 Digital Signage를 찾아볼 수 있지만 이를 악용한 사이버 공격이 증가하고 있어 보안 사고가 발생했을 경우 큰 파급력을 가져올 수 있다. 그럼에도 불구하고 현재 관련 보안 연구는 상대 적으로 미흡한 실정이다.

따라서 본 논문에서는 Digital Signage 플랫폼의 구성을 도식화하고 STRIDE 위협 모델링 기법을 이용해 발생할 수 있는 위협을 식별하여 취약점 및 대응방안 도출을 목표로 한다.

II. Digital Signage 위협 모델링 기법을 통한 취약점 도출 및 대응방안 수립

STRIDE 위협 모델링 기법은 공격자의 관점에서 발생 가능한 잠재적 위협을 식별하고 우선순위를 정할 수 있는 일종의 프로세스이다. STRIDE는 응용 프로그램에서 발생할 수 있는 위협 유형들을 위장(Spoofing), 변조(Tampering), 행위 부인(Repudiation), 정보 유출(Information disclosure), 서비스 중단(Denial of service), 권한 상승(Elevation of privilege) 해당 여섯 가지로 식별한다. 본 장에서는 취약점 리스트 도

출을 위해 Digital Signage 플랫폼을 도식화하고 DFD 기반으로 STRIDE 위협 모델링 기법을 적용한다.

2.1 Digital Signage 플랫폼 도식화

Digital Signage 플랫폼을 도식화하였고 Fig. 1과 같이 운영된다.[4] Manager는 Digital Signage 내 구동되고 있는 Web 서버를 이용하여 기기를 관리할 수 있다. 또한, 각 기기들은 CMS(Contents Management System)라는 관리 솔루션을 통해 원격에서 콘텐츠를 배포하거나 기기를 제어할 수 있는 권한을 가진 CMS Manager로부터 통합 관리된다.

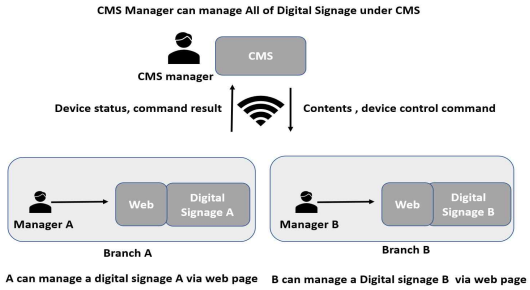


Fig. 1. Digital Signage Platform Concept Diagram

2.2 DFD(Data Flow Diagram) 작성

DFD를 이용해 Digital Signage 플랫폼의 데이터 흐름을 시각적으로 나타낼 수 있으며 체계적으로 추적이 가능하다. 이를 통해 프로세스

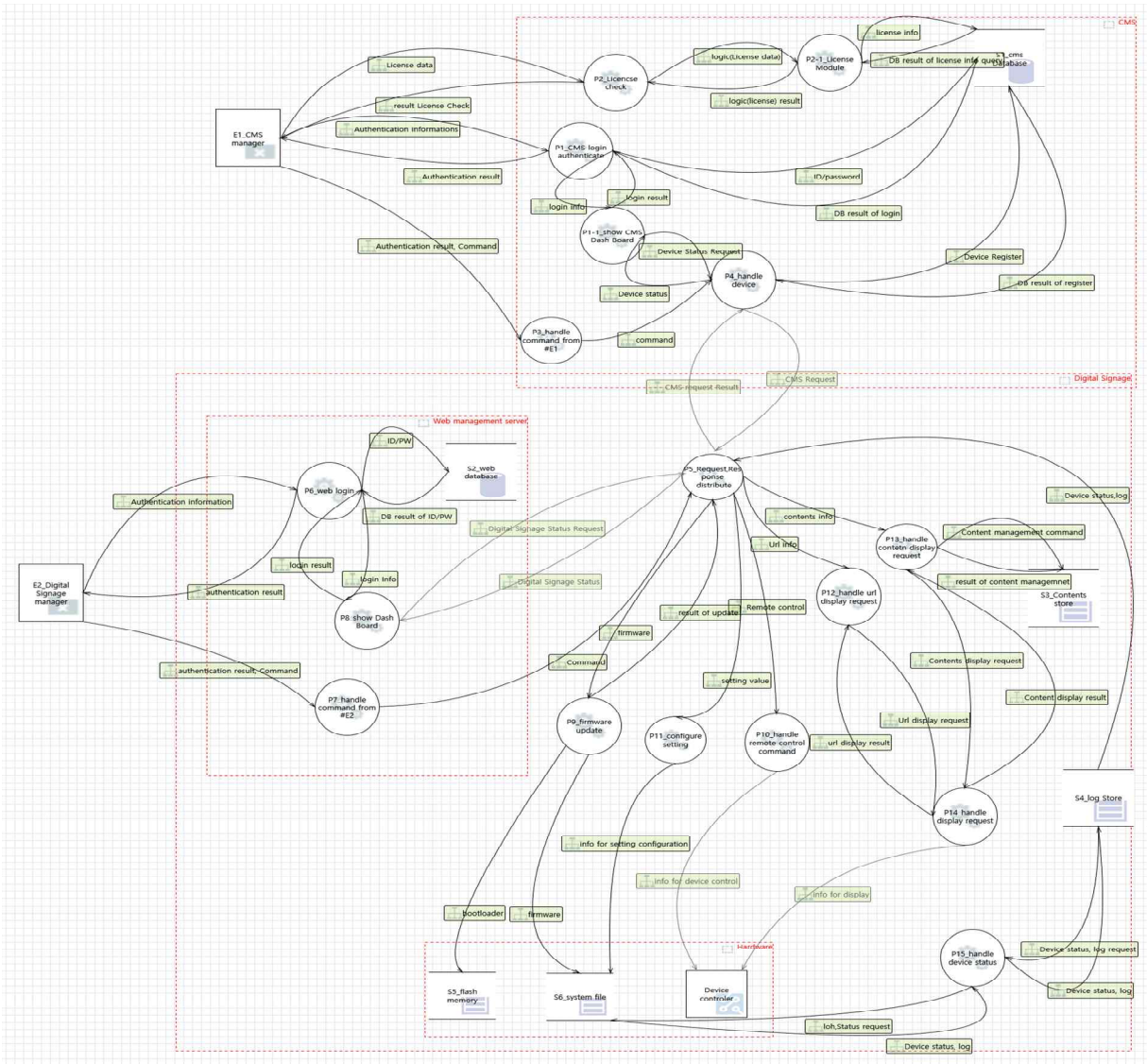


Fig. 2. DFD for Digital Signage Platform

들에 대한 위협을 사전에 식별할 수 있다. 앞서 설명한 것처럼 위협을 사전에 식별하기 위해 Fig. 2와 같이 Digital Signage 플랫폼의 DFD를 작성하였다.[5] 다음 Table 1은 앞서 확인한 Fig. 2의 DFD 구성요소들을 설명하고 있다.

Type	No	Name	Description
Entity	E1	CMS manager	Manager of CMS
Entity	E2	Digital Signage manager	Manager of Digital Signage
Process	P1	CMS login authenticate	Authenticate login info for CMS
Process	P1-1	Show CMS dash board	After login Success, show dashboard to user
Process	P2	Check licencse exsist	License file check
Process	P2-1	Check license logic	License logic check
Process	P3	Handle command from #E1	If CMS got command from Admin, It relay to #p4
Process	P4	Handle device	If Command is related to device then relay to #p5
Process	P5	Request/response distribute	Distribute commands to handler
Process	P6	Web login	Authenticate logininfo at web
Process	P7	Handle command from #E2	If webserver got command from Signage owner, It relay to #p5
Process	P8	Show app dash board	After login Success, show dashboard to Signage owner
Process	P9	Firmware update	Update firmware
Process	P10	Handle remote control command	Handle Remote control payload
Process	P11	Configure setting	Configure setting
Process	P12	Handle URL display request	Handle URL display payload
Process	P13	Handle content display request	Handle contents display payload
Process	P14	Handle display request	Relay all about display payload to device controller
Process	P15	Handle device status	Get device status and relay to dashboard
Data Store	D1	CMS DB	Store data for CMS
Data Store	D2	Web DB	Store data for Web Server
Data Store	D3	Contents Store	Store data for contents
Data Store	D4	Log Store	Store data for log
Data Store	D5	Flash memory	Store bootloader
Data Store	D6	System file	Store file system
flow		CMS manager ->CMS	Control & command to Signage via CMS
flow		CMS -> Signage	Relay CMS manager's command to signage
flow		Signage manager -> Signage	Control & command to Signage

Table 1. Description of Digital Signage DFD Element

2.3 STRIDE 위협 모델링 기법을 이용한 취약점 도출

STRIDE를 이용한다면 초기에 잠재적인 보안 문제를 식별할 수 있어 공격으로부터 시스템을 보호할 수 있다. Digital Signage 플랫폼에 STRIDE 위협 모델링 기법을 적용하여 도출된 취약점 리스트를 Table 2에서 제시한다.

Type	No	Name	STRIDE	Threat Description	No
Entity	E1	CMS manager	S	Anonymous user claim them is manager	T1
Entity	E2	Digital Signage manager	R	Attacker deny Authorized user access	T2
Process	P1	CMS login authenticate	S	Attacker make fishing login site	T3
			S	Spoofing for MITM attack	T4
			T	Tampering authentication parameters	T5
			R	Attacker repudiate sniffing, brute forcing	T6
			I	Sniffing parameter for Authentication	T7
			I	Sensitive information be leaked	T8
			D	Attacker can make login function lock by tying and exceeding available number of login attempt	T9
			D	Attacker can make access login page impossible.	T10
			E	Attacker got admin privilege	T11
			E	Attacker can access to specific page without authentication	T12
Middle omission					
Data Store	D6	System file	I	Extract file system	T96
			D	Denial of system service	T97
			T	Tampering system file	T98
Flow		CMS manager <->CMS	T	Tampering request/response payload	T99
			I	Leaking of login info	T100
			I	Leaking of communication	T101
			D	Denial of communication	T102
Flow		CMS <-> Signage	T	Tampering request/response payload	T103
			I	MITM via Spoofing	T104
			T	Send tampered Request / Response	T105
			D	Denial of communication	T106
Flow		Signage manager <-> Signage	T	MITM via Spoofing	T107
			I	Leaking of login info	T108
			I	Leaking of communication	T109
			T	Send tampered Request / Response	T110
			D	Denial of communication	T111

Table 2. STRIDE per Element for Digital Signage Platform DFD

STRIDE 유형별 취약점 개수는 S:21, T:23, R:15, I:28, D:17, E:7로 도출된 취약점은 총 111가지이다.

2.4 STRIDE 위협 모델링 기법으로 도출된 취약점 대응방안

다음 Table 3은 STRIDE 위협 모델링 기법으로 도출된 취약점 대응방안을 제시한다.[6][7][8][9] Table 3의 열 머리글을 살펴보면 'Signage/CMS'는 도식화한 Fig. 1의 Digital Signage와 CMS 자체 내부에 발생할 수 있는 위협들의 대

응방안이며, 'Digital Signage<->CMS'는 Digital Signage와 CMS간의 통신에서 발생할 수 있는 위협들의 대응방안이다.[10]

Threat	DIgital Signage/CMS	Digital Signage<->CMS
Spoofing	• Packet Filtering • Access Control List • Packet Sender Authentication • Strong Password • Security login logic	• Packet Encryption • Secure Session Token • Secure communication channel • PKI
Tempering	• One-Way Hash Function • Secure Boot • Read-Only File System • Database Encryption	• Time Stamp • Parity Bit • Secure communication channel
Repudiation	• Digital Signing • Strong Authentication • Use the logging features	• Packet dump
Information Disclosure	• Debug Port Disabling • Firmware Code Signing • Database Encryption • Secure Encryption Protocol • Code Obfuscation	• Packet Encryption • Certificate Pinning
Denial of Service	• Higher Bandwidth • Abnormal status detection	• Traffic Analyzer • Monitering • NAT • Time Stamp
Elevation of Privilege	• RBAC • Data Execution Prevention • Least Privilege • Address space layout randomization • Application Sandboxing • Antivirus software. • Scan the application for known vulnerabilities • Multi-factor authentication and authorization	• Packet Encryption • Certificate Pinning

Table 3. Countermeasure for STRIDE threat

III. 결론

Digital Signage는 보안 전문가들이 일반적인 경로로 구하기 어려운 B2B(Business to Business) 제품이므로, 현재 Digital Signage 플랫폼에서 발생 가능한 위협 연구가 활발하게 이루어지지 않고 있다.

따라서 본 논문에서는 Digital Signage의 플랫폼을 도식화한 개념도를 기반으로 DFD를 작성하였고 STRIDE 위협 모델링 기법을 적용시켜 취약점 리스트 및 대응방안을 제시하였다.

본 논문에서 제시한 취약점 리스트를 활용하여 향후 Digital Signage 플랫폼의 보안성 강화를 위해 취약점 점검 체크리스트와 상세 대응방안을 도출하고, 이를 기반으로 보안 가이드라인을 제작할 수 있도록 연구가 필요하다.

[참고문헌]

[1] Jong Hwan Cha, "'Korean style Time square' Digital Signage seize spotlight", http://www.koit.co.kr/news/articleView.html?idxno=74874, 2019.

[2] Yong Seok Cheon, "The Status and implication of Digital Signage Industry Promotion", 2016.

[3] Grand View Research, "Digital Signage Market Size Analysis Report", 2019.

[4] Korea Telecommunications Technology Association, "HTML5 based Digital Signage Device Service Platform Standard", 2018.

[5] Seung Joo Kim, "Basic principles When Write DFD(Data Flow Diagram)", http://blog.naver.com/amhoin/221660252139, 2019.

[6] OWASP, "Embedded Application Security TOP10", 2018.

[7] OWASP, "Internet of Things (IoT) Top 10", 2018.

[8] KISA, "Home and Appliances IoT Security Guide", 2017.

[9] KISA, "IoT Common Security Guide", 2016.

[10] Adam Shostack, "threat modeling", WILEY, pp. 145-165, 2014.

Syma X8 드론에 대한 취약점 분석*

김예준, 강수영, 김승주
고려대학교 정보보호대학원

Vulnerability Analysis for X8SW Drones

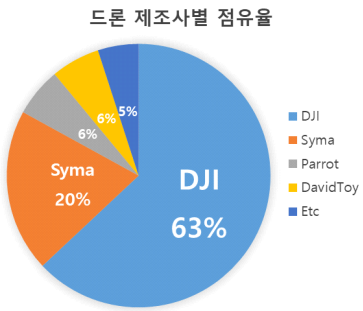
Ye-jun Kim, Soo-young Kang, Seung-joo Kim
Center for Information Security Technologies (CIST), Korea University

요 약

현재 IoT 장비들의 인기가 높아지면서 그중 하나인 드론 역시 군사용부터 상업용까지 매년 시장규모가 지속해서 증가하고 있다. 하지만 여러 사용자가 생기고 대중화되는 만큼 내부 시스템 분석을 이용한 취약점들이 발견되며 문제가 발생하고 있다. 또한 진입장벽이 높은 하드웨어적인 분석의 경우 분석가들의 관심이 낮아 상대적으로 활발한 연구가 이루어지지 않는다. 따라서 본 논문에서는 Syma사의 X8SW 드론을 대상으로 체계적인 펌웨어 획득 절차에 대하여 기술하고 그를 이용한 취약점 분석을 통해 드론 시장의 보안성 향상에 도움을 주고자 한다.

I. 연구 배경 및 관련 연구

현재 드론 산업은 군사용으로 사용될 뿐만 아니라 민간산업까지 확대되며 농업, 영상, 재해관측, 스포츠 등 다양한 분야에서 활용되고 있다. 해외 통계 사이트인 Satista 에 따르면 상업용 드론의 세계 시장 규모는 2019년 기준 판매량은 약 392,000대이며 가치는 16억 달러로 추정되고 있다. 이에 대한 시장규모는 매년 지속해서 상승하고 있다[1].



[그림 1] 드론 제조사별 점유율

또한 커지는 드론 시장에서 2019년도 국내 시장 점유율은 DJI, Syma, Parrot 순서로 자리 잡고 있다[2]. 이렇게 점점 대중화되는 드론 대상의 네트워크 패킷이나 파일 시스템 등을 분석해주는 도구개발, WiFi나 RF 신호 등과 같은 무선 네트워크를 통해 조작이 가능한 드론의 특성을 이용한 재전송 공격과 패킷 조작과 같은 취약점들이 발견되며 관련 연구물들이 지속해서 발표되고 있다[3][4]. 또한 과거부터 하드웨어 장비의 분석기법 및 취약점 분석에 대한 연구는 꾸준히 진행되어 왔다. 해외 유명 해킹 컨퍼런스인 Black-Hat USA 2013에서는 IoT 장비에서 일반적으로 사용되는 펌웨어에 대한 분석 방법에 대한 발표가 진행되었다[5]. 또한 2018년 IEEE에서는 IP 카메라를 대상으로 IoT 장비의 분석 기술 및 방법론이 발표되었으며[6], IoT 장비와 연결된 Application에 대한 Fuzzer를 제작하여 발표된 연구도 존재하였다[7]. 본 연구에서는 드론을 대상으로 분석 절차를 구성하고 이를 이용한 취약점 분석과 그에 대한 대응 방안을 제시하고자 한다. 본 논문의 2장에서는

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2019-2015-0-00403)

드론분석에 필요한 분석절차와 과정을 다루며, 3장에서는 이를 바탕으로[1] 진행한 취약점 분석 결과와 그에 대한 대응방안을 다룬다. 마지막으로 4장에서는 본 논문의 결론 및 향후 연구 방향에 대한 내용으로 끝을 맺는다.

II. 분석 절차

장비 획득 후 내부에 저장되어있는 펌웨어를 분석하기 위해서는 펌웨어를 획득하거나, 별도의 분석환경을 구성해야 한다. 이러한 과정에서 체계적인 분석 절차가 존재하여 제공될 경우 분석을 시도하는 사람이 누구냐에 따라 발생할 수 있는 결과물의 격차를 줄일 수 있다. 드론에서 펌웨어를 획득하기 위해 필요한 분석 절차는 [그림 2]와 같다.

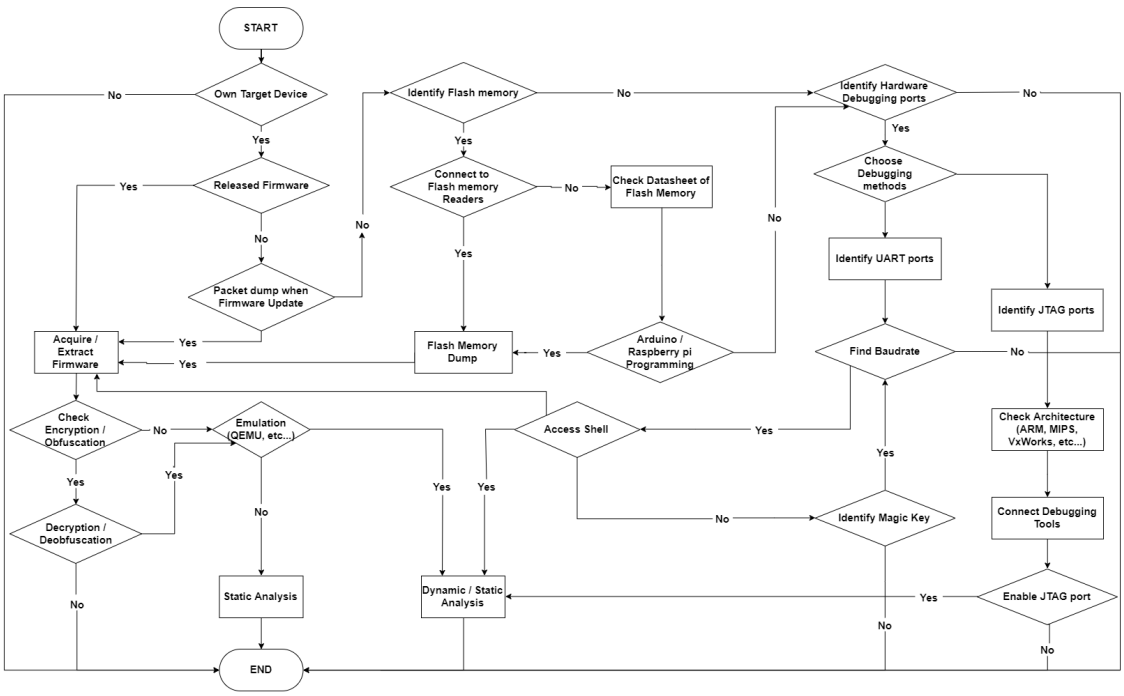
2.1 장비 외 펌웨어 획득

분석 대상 장비의 공식 홈페이지를 통해 펌웨어 다운로드가 가능한 경우가 존재한다. 현재 상용 드론 중 가장 높은 점유율을 가지고 있는 DJI사의 Phantom 시리즈, Inspire 등이 이에 해당하며 제품명을 알고 있는 경우 인터넷 검색을 통해 쉽게 펌웨어 획득이 가능하기 때문에 바로 소프트웨어 분석단계 진행이 가능하다. 또는 드론의 펌웨어 업데이트 기능의 동작을 패킷캡쳐도구인 WireShark나 Fiddler 등을 이용하여 Sniffing[8]하여 펌웨어 획득이 가능하다.

2.2 디버깅 포트를 이용한 분석

데이터 전송, 디버깅 등을 목적으로 사용되는 UART[9], JTAG[10][11] 포트를 이용하여 펌웨어 추출 및 명령어 쉘 획득, 동적 분석 등이 가능하다.

UART는 Gnd, Rx, Tx 포트를 USB-ISP와 같은 장비와 연결하여 Putty, Xshell과 같은 도구로 Serial 통신이 가능하다. 이때 Serial 전송 속도인 Baudrate를 식별해야하며 이는 Jtagulator를 이용하여 전수조사가 가능하다. 또한 기존에 자주 쓰이는 Baudrate로는 9600, 19200, 57600, 115200 등이 있어 Jtagulator를 이용하여 식별이 불가능 할 경우 사용이 가능하다. Serial 통신이 성공한 경우 즉시 Shell에 대한 접근이 가능하거나 Booting 메시지 혹은 연결한 장비에 따라 일부 정보에 대한 읽기만 가능하다. 이때는 펌웨어 내부에 Debug 모드로 연결해주는 숨겨진 명령어가 존재할 수 있어 내부 펌웨어 분석을 통해



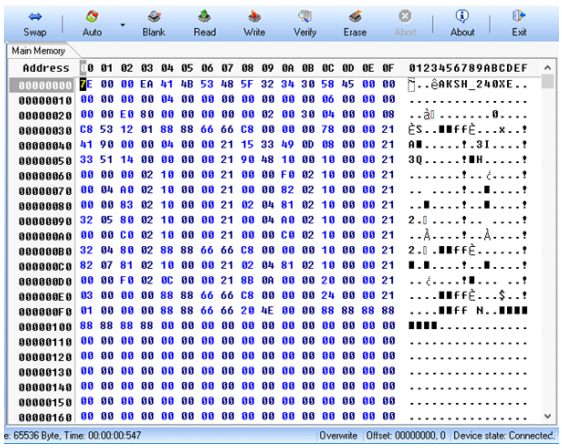
[그림 2] 드론 기반의 펌웨어 획득 절차

명령어를 추출하거나 특정 조건을 만족시킬 경우 Shell을 획득할 수 있다.

JTAG은 펌웨어 획득은 물론 동적, 정적 분석이 모두 가능하여 다양한 용도로 사용될 수 있는 강력한 방법이다. 하지만 장비에서 사용되는 펌웨어의 아키텍처에 따라 사용되는 연결 장비의 종류가 다르다. 따라서 DataSheet, 검색엔진을 통하여 사전 정보 수집을 통해 해당하는 아키텍처에 필요한 전문 장비가 필요하다.

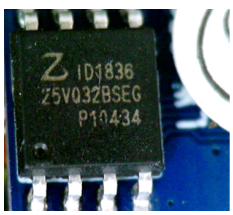
2.3 Flash Memory를 이용한 펌웨어 획득

내부 PCB 보드에 있는 Flash Memory의 Datasheet를 획득한 뒤, Arduino나 Raspberry pi를 이용하여 직접 프로그래밍을 하여 추출이 가능하다. 또한 Chip에 대한 모델명에 대한 식별이 가능한 경우 CH341A와 같은 Flash Memory Reader를 사용하여 [그림 3]과 같이 펌웨어 추출이 가능하다.



[그림 3] CH341A를 이용한 펌웨어 획득

Flash Memory의 Datasheet를 획득하지 못할 경우에는 [그림 4]와 같이 Chip의 모델명을 통해서 종류와 특징을 파악하여 장비연결 및 프로그래밍을 시도할 수 있다.



Flash Memory (ZB25VQ32BSEG)

ZB : Zbit (Manufacturer)
25 : SPI (Product Family)
VQ : 2.5V 4KB, Quad Mode (Series)
32 : 32Mbit (Device Density)

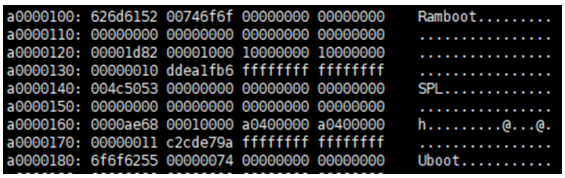
[그림 4] 모델명을 통해 얻을 수 있는 정보

III. 분석절차를 이용한 취약점 분석

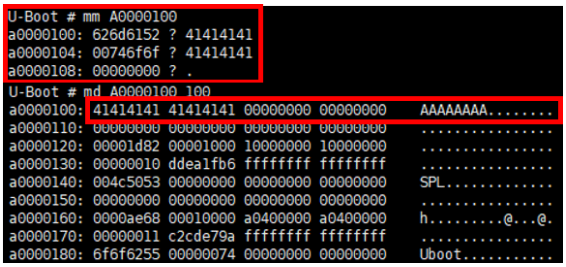
2장에서 언급한 절차를 실제 드론에 적용한 분석 결과는 아래와 같다.

3.1 메모리 위·변조

PCB 분석으로 UART 포트를 식별 후 연결하여 U-Boot Shell을 얻을 수 있다. 이를 통해 [그림 5, 6]과 같이 사용되는 메모리에 대한 읽기/쓰기가 모두 가능하여 자유로운 메모리 수정이 가능하다.

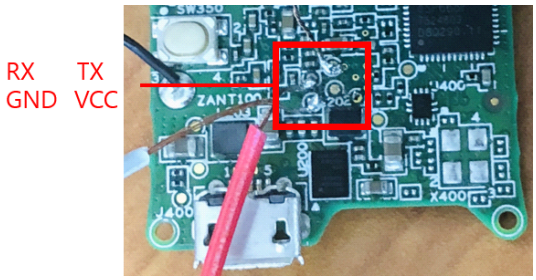


[그림 5] U-Boot를 이용한 메모리 접근



[그림 6] U-Boot를 이용한 메모리 수정

따라서 UART를 통한 메모리 변조를 막기 위해서는 디버깅 포트를 비활성화하거나 식별이 어렵게 만들어 접근을 제한해야 한다.

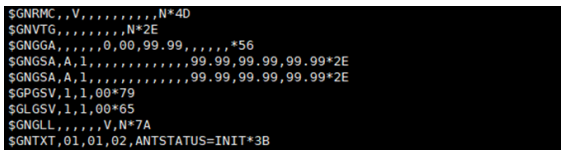


[그림 7] 숨겨져 있는 UART 포트

3.2 GPS 신호 획득

위치정보의 전송은 [그림 8]과 같이 드론과 연결되어있는 GPS 모듈에서 NMEA를 사용하여 위치정보를 전송하고 있는 것을 확인할 수 있으며 이는 UART 통신을 이용하여 전송된다. 전송되는 GPS 신호를

변조하여 드론에 보낼 경우 GPS Jamming[12]을 발생시킬 수 있으며, 이를 이용하여 드론 탈취가 가능하다.



[그림 8] NMEA를 이용한 GPS 정보 전송

IV.결론 및 향후 연구 방향

실제 취약점 분석 분야는 많은 분석가가 연구를 진행하고 있지만, 취약점 분석을 위해 필요한 이전 단계인 펌웨어 획득 등과 같은 절차는 많은 부분이 생략되어 하드웨어 장비에 대한 진입장벽이 높다. 따라서 드론의 분석 절차를 구성하여 취약점을 분석함으로써 드론에 대한 보안성 향상에 도움이 될 것이라고 기대한다. 본 논문에서는 특정 드론에 대한 분석 절차를 구성하고 취약점 분석을 진행하였다. 하지만 각 장비별로 분석 절차의 우선순위와 결과가 다르게 도출될 수 있다. 그렇기에 향후 진행하고자 하는 연구에서는 다양한 장비를 분석하여 각 장비의 특성에 따라 적합한 분석 절차를 수립하여 하드웨어 장비 분석에 대한 접근성을 높이고자 한다.

[참고문헌]

[1] Katharina Buchholz, “Commercial Drones are Taking Off” 2019 [Internet], https://www.statista.com/chart/17201/commecial-drones-projected-growth/

[2] Danawa research, “드론 가이드 끝판왕! 드론의 모든 것을 파헤쳐보자.” 2019 [Internet], http://dpg.danawa.com/news/view?boardSeq=63&listSeq=3871439

[3] Clark, Devon R., et al. "DROP (DRone Open source Parser) your drone: Forensic analysis of the DJI Phantom III." Digital Investigation 22 (2017): S3-S14.

[4] Mantas, Evangelos, and Constantinos Patsakis. "GRYPHON: Drone Forensics in Dataflash and Telemetry Logs." International Workshop on Security. Springer, Cham, 2019.

[5] Zaddach, Jonas, and Andrei Costin. "Embedded devices security and firmware reverse engineering." Black-Hat USA (2013).

[6] Shwartz, Omer, et al. "Reverse engineering IoT devices: Effective techniques and methods." IEEE Internet of Things Journal 5.6 (2018): 4965-4976.

[7] Chen, Jiongyi, et al. "IoTFuzzer: Discovering Memory Corruptions in IoT Through App-based Fuzzing." NDSS. 2018.

[8] Okmianski, Anton, Mickael Graham, and Joshua B. Littlefield. "Method of identifying a home gateway using network traffic sniffing and apparatus employing the same." U.S. Patent No. 7,505,464. 17 Mar. 2009.

[9] Lu, Chung-Ming. "Communication system for devices with UART interfaces." U.S. Patent No. 7,650,449. 19 Jan. 2010.

[10] Zadigian, Timothy, Jonathan Stroud, and Michael Moriarty. "JTAG-based programming and debug." U.S. Patent No. 8,856,600. 7 Oct. 2014.

[11] Rosenfeld, Kurt, and Ramesh Karri. "Attacks and Defenses for JTAG." IEEE Design & Test of Computers 27.1 (2010): 36-47.

[12] Van den Bergh, Bertold, and Sofie Pollin. "Keeping UAVs Under Control During GPS Jamming." IEEE Systems Journal 13.2 (2018): 2010-2021.

DDR4 메모리의 보안 취약점을 통한 보안 위협 검증

강지훈,* 류재철**
충남대학교 컴퓨터공학과

A Study on Verification of Security Threat Through Security Vulnerabilities of DDR4 Memory

Ji-hun Kang,* Jae-cheol Ryou**
Department of Computer Engineering, Chungnam National University

요 약

최근 기존의 운영체제 내부에서 발생하는 보안 취약점이 아닌 새로운 영역을 통한 사이버 공격이 시도되고 있다. 이는 기술이 고도화됨에 따라 PC의 각 부품 또한 사용자의 수요에 따른 다양한 기능을 제공하게 되었고, 이로 인해 공격 벡터가 확장된 결과라고 할 수 있다. 본 논문에서는 2019년 현재 가장 대중적으로 사용되고 있는 RAM에 해당하는 DDR4 메모리의 보안 취약점에 대해 분석하고, 이로 인해 발생 가능한 보안 위협에 대해 설명한다.

I. 서론

메인보드의 바이오스를 통한 공격 기법이 발표 되고, GPU 기반의 루트킷이 크게 이슈가 되는 등 기존의 운영체제 내부에서 발생하는 보안 취약점이 아닌 새로운 영역을 통한 사이버 공격이 시도되고 있다. 이는 기술이 고도화됨에 따라 PC의 각 부품 또한 사용자의 수요에 따른 다양한 기능을 제공하게 되었고, 이로 인해 공격 벡터가 확장된 결과라고 볼 수 있다.

컴퓨터 RAM(Random Access Memory)은 주기억 장치로써 PC의 구성요소 중 필수적인 부품이며 일반적으로 알려진 보안 위협이 거의 존재하지 않는다. 본 논문에서는 2019년 현재 가장 대중적으로 사용되고 있는 RAM에 해당하는 DDR4 메모리의 보안 취약점에 대해 검증하고, 이로 인해 발생 가능한 보안 위협에 대해 설명한다.

본 논문의 구성은 다음과 같다. 2장에서 연구 배경에 대해 소개하고 3장에서 DDR4 메모리 모듈에 존재하는 보안 취약점을 증명하고, 이로 인해 발생 가능한 보안 위협에 대해 설명한다. 마지막으로 4장에서 결론을 맺는다.

II. 연구배경

2.1 메모리 오버클럭

PC를 구성하는 부품들은 보통 기본 연산속도에 해당하는 클럭(Base clock * Multiplier)을 기반으로 작동한다. 오버클럭이란 이러한 클럭을 사용자가 임의로 조작하여 기본 클럭 속도를 초과하여 작동하게 만드는 것을 의미한다. 오버클럭이 가능한 부품은 대표적으로 CPU, 메모리, GPU 등이 있다.

최근 메모리 오버클럭은 고사양 PC 게임을 보다 원활하게 실행하기 위해서 많이 사용하는 추세이다. 특히 AMD 사의 CPU를 사용할 경우 인피니티 패브릭(Infinity Fabric) 기술로 인하여 메모리 클럭 속도가 CPU 성능까지 영향을 주

게 된다. 이러한 메모리 오버클럭을 적용하기 위해서는 과부하를 견딜 수 있는 오버클럭용 메인보드와 메모리가 필수적이다. 메인보드에 따라 메모리 오버클럭 지원 유무가 다르므로 반드시 메모리 오버클럭을 지원하는 메인보드를 사용해야 한다. 또한 메모리가 동작 가능한 최대 클럭 속도는 메모리 자체의 내구성과 밀접한 관련이 있다. 따라서 오버클럭용 메모리 제조사들은 발열을 해소하여 내구성을 높이기 위해 메모리 위에 방열판을 부착하여 판매한다.

현재 시중에서 판매되고 있는 오버클럭용 메모리는 대부분 2133Mhz 클럭 속도로 작동하는 PC4-17000을 기반으로 제작된다. 이를 메모리 오버클럭을 통해 3000Mhz, 3200Mhz, 3600Mhz 이상으로 작동시키는 형태이다. 이러한 고 클럭 메모리는 보통 XMP(Extreme Memory Profile) 기술을 사용하여 오버클럭이 적용된다. 해당 기술에 대해서는 아래 절에서 자세하게 설명한다.

2.2 XMP(Extreme Memory Profile)

XMP란 인텔에서 출시한 메모리 오버클럭 기술을 의미한다. 이러한 XMP는 메모리의 클럭과 램타이밍, 전압 정보 등을 사전에 프로파일 형태로 저장해두어, 사용자가 직접 해당 정보를 테스트할 필요 없이 메모리 오버클럭을 적용할 수 있도록 개발된 기술이다. XMP를 사용하지 않고 수동으로 오버클럭을 적용할 경우 적절한 클럭과 램타이밍 등을 설정하는 작업은 매우 까다로운 작업이 될 수 있다. 반면에 XMP를 사용하게 되면 제조사가 사전에 테스트를 거쳐 검증된 클럭과 램타이밍을 사용하기 때문에 보다 안정적이고 간편하게 메모리 오버클럭이 가능하다. 이러한 XMP 프로파일은 크게 클럭과 램타이밍, 전압 정보 등으로 구성되어 있다. 프로파일 상에 표기되는 램타이밍 정보는 아래 순서대로 tCL(Column-Address Strobe Latency), tRCD(Row-address to Column-address Delay), tRP(Row Precharge), tRAS(Row Active Time)에 해당한다. 따라서 오버클럭용 메모리를 살펴보면 아래 [그림 1]과 같이 클럭과 램타이밍, 전압 정보를 확인할 수 있다.

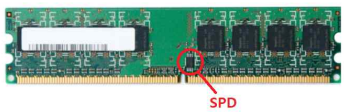


[그림 1] 오버클럭용 메모리

III. 보안 취약점 분석

3.1 DDR4 메모리의 SPD 구조

SPD(Serial Presence Detect)는 PC가 부팅될 때 메모리 모듈의 정보를 바이오스로 전달하는 JEDEC 규격이다. 이러한 SPD 데이터는 아래 [그림 2]와 같이 메모리 모듈 표면에 존재하는 EEPROM(Electrically Erasable Programmable Read-Only Memory)에 저장되어 있다.



[그림 2] DDR4 메모리의 SPD

SPD 데이터의 용량은 총 512바이트며, 내부는 제조사, 클럭, 전압 정보 등 메모리의 동작과 관련된 데이터로 구성되어 있다.

Block	Range		Description
0	0~127	0x000~0x07F	Base Configuration and DRAM Parameters
1	128~255	0x080~0x0FF	Module Specific Parameters -- See subsections L1, L2, and L3 for details
2	256~319	0x100~0x13F	Reserved -- must be coded as 0x00
	320~383	0x140~0x17F	Manufacturing Information
3	384~511	0x180~0x1FF	End User Programmable

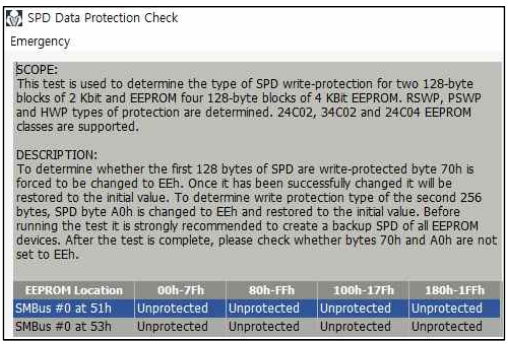
[그림 3] SPD address map for DDR4

[그림 3]은 DDR4 메모리 내부 SPD의 메모리 맵이다. 이 중 offset 384부터 511까지의 128 바이트 영역은 End User Programmable 영역에 해당한다. 오버클럭용 메모리의 제조사는 해당 영역에 XMP 프로파일을 저장한다. XMP 프로파일은 통상 64바이트 미만이므로 오버클럭용 메모리에는 1개 또는 2개의 XMP 프로파일이 저장되어 있다.

3.2 SPD의 보안 취약점

SPD의 offset 0-383 영역은 메모리의 작동을 위해 필수적인 정보가 포함되어 있다. 하지만 383 이후 영역부터는 End User Programmable

영역으로써 XMP와 같은 추가적인 기능을 사용하기 위한 공간으로 사용된다. 이는 해당 영역에 어떠한 값이 저장되더라도 메모리의 기본 동작과는 관계가 없음을 의미한다. 즉 해당 영역이 보호되어 있지 않다면 악의적인 사용자에 의해 악성코드와 같은 임의의 코드가 삽입될 수 있으며, 이는 심각한 보안 위협을 초래할 수 있다. 이러한 보안 취약점을 검증하기 위하여 본 논문에서는 오버클럭용 메모리로 유명한 G 사의 오버클럭용 메모리를 대상으로 보안 취약점을 확인하고 코드 삽입 공격을 진행하였다.



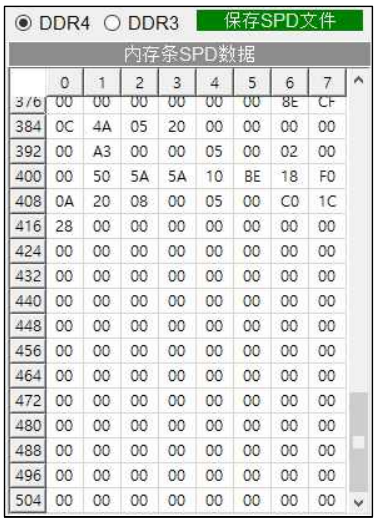
[그림 4] SPD 보안 취약점 확인

[그림 4]는 메모리의 상세 정보를 확인하기 위해 사용되는 프로그램인 Thaiphoon Burner를 사용하여 분석 대상 메모리의 SPD 영역에 대한 쓰기 권한을 확인한 화면이다. 이를 확인한 결과 SPD의 전 영역은 Unprotected 상태로써 쓰기가 가능한 상태임을 확인할 수 있다.

3.3 코드 삽입 공격

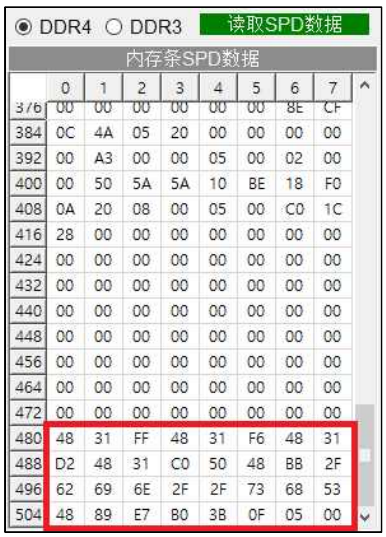
SPD의 전 영역에 쓰기 금지 Lock이 설정되어 있지 않음을 확인하였고, 이를 검증하기 위해 DDR4 memory programmer를 사용하여 End User Programmable 영역에 리눅스를 기반으로 동작하는 쉘 코드를 삽입하였다.

우선 End User Programmable 영역(384-511)을 확인한 결과 [그림 5]와 같이 1개의 XMP 프로파일이 존재하였다.



[그림 5] XMP 프로파일

DDR4 memory programmer를 사용하여 리눅스에서 실행 가능한 31 바이트 크기의 쉘 코드 "\x48\x31\xff\x48\x31\xf6\x48\x31\xd2\x48\x31\xc0\x50\x48\xbb\x2f\x62\x69\x6e\x2f\x2f\x73\x68\x53\x48\x89\xe7\xb0\x3b\x0f\x05"를 offset 480에 삽입한 결과 [그림 6]과 같이 쉘 코드가 정상적으로 삽입된 것을 확인할 수 있다.



[그림 6] 쉘 코드 삽입 결과

이처럼 제조사가 출시 전에 SPD 영역에 쓰기 금지 Lock을 설정하지 않을 경우 이를 악용하여 임의의 코드를 삽입할 수 있게 된다. 특히 SPD 영역은 운영체제에서 접근 가능한 영역이기 때문에 실행 가능한 형태의 코드가 삽입될 수도 있고, 코드은닉 공간으로 사용되어 다른

사이버 공격의 통로로 악용될 가능성 또한 존재한다. 추가적으로 오버클럭용 메모리가 아닌 S사의 DDR4 메모리를 대상으로 동일한 실험을 진행한 결과 위와 동일하게 코드 삽입이 가능한 것을 확인하였다. 이러한 일반적인 DDR4 메모리는 기본적으로 End User Programmable 영역이 비어있기 때문에 128 바이트를 초과하는 임의의 코드가 삽입될 수 있다. 그리고 해당 영역은 메모리의 기본 동작에 영향을 주지 않기 때문에 임의의 코드가 삽입되더라도 이를 탐지하는 것은 매우 어려운 일이 될 수 있다.

IV. 결론

본 논문을 통해 SSD나 HDD와 같은 일반적인 데이터 저장 장치가 아닌 DDR4 메모리 모듈 공간에도 임의의 코드를 영구적으로 저장할 수 있다는 것을 증명하였다. 이는 운영체제 환경에서 접근 가능한 영역이므로 코드은닉을 통한 심각한 보안 위협이 될 수 있다. 따라서 메모리 제조사는 SPD 영역에 쓰기 금지 Lock을 설정하여 임의의 데이터를 쓰지 못하도록 조치해야 한다. 또한 본 논문을 통해 백신 프로그램이나 사용자에 의한 보안 점검이 이뤄지지 않는 영역에도 보안 위협이 존재할 수 있다는 것을 확인하였다. 향후 이와 같은 다양한 영역의 사이버 공격에 대한 연구를 진행할 계획이다.

[참고문헌]

[1] Xeno Kovah, Corey Kallenberg, How Many Million BIOSes Would you Like to Infect?, CanSecWest, March, 2015.

[2] New GPU-based Linux Rootkit and Keylogger with Excellent Stealth and ComputingPower, HackerNews, May, 2015. https://thehackernews.com/2015/05/gpu-rootkit-linux-Keylogger.html

[3] SPD Annex L: Serial Presence Detect (SPD) for DDR4 SDRAM Modules, Memory Configurations: JESD21-C

[4] Thaiphoon Burner, http://softnology.biz/

코드 인젝션을 통한 데이터 탈취 가능성 연구

김찬진*, 염흥열*
*순천향대학교 정보보호학과

A study on the possibility of data sniffing by code injection

Chan-Jin Kim*, Heung-Youl Youm*
*Division of information security, Soonchunhyang University.

요 약

악성코드들은 탐지를 회피하기 위해 DLL 인젝션과 코드 인젝션을 통해 정상적인 프로세스에 원하는 공격 코드들을 주입한다. 그 중 코드 인젝션 기법은 2008년도부터 악성코드들이 사용하기 시작하였고, 현재까지 유포되고 있는 악성코드들도 사용하고 있다. 코드 인젝션 기법은 상대적으로 DLL 인젝션보다 메모리를 덜 차지하고, 흔적을 잘 남기지 않기 때문에 탐지가 어렵다. 본 논문에서는 코드 인젝션 기법들의 종류에 대해 알아보고, 각 기법들의 실제 구현을 함으로써 그에 대한 데이터 탈취 가능성에 대해서 논하고자 한다.

I. 서론

악성코드들은 백신 등의 탐지 솔루션으로부터 탐지되는 것을 회피하기 위해 인젝션 기법을 사용하여 정상적인 프로세스에 어태치(Attach)하는데, 이러한 인젝션 기법은 DLL을 대상 프로세스 메모리에 삽입시키는 DLL 인젝션(DLL Injection)과 필요한 코드와 데이터만 삽입시키는 코드 인젝션(Code Injection) 크게 두 가지로 나눌 수 있다.

대략 2008년도부터 악성코드들은 코드 인젝션 기법을 사용하기 시작했고[1], 최근까지도 유포되고 있는 랜섬웨어나 파밍 등의 악성코드들 또한 코드 인젝션 기법을 사용하고 있다. 코드 인젝션은 DLL 인젝션과 달리 필요한 소량의 코드와 데이터만 인젝션하기 때문에 메모리를 상대적으로 덜 차지하며, 별도의 파일 없이 인젝터(Injector) 프로그램만 있으면 쉽게 메모리에 인젝션할 수 있기 때문에 쉽게 흔적을 남기지 않고, 따라서 탐지하기가 어렵다. 또한 현재까지도 지속적으로 새로운 코드 인젝션 기법들이 발견되고 있고, 각각의 메커니즘들이 다른 경우가 많기 때문에 대응하기도 까다롭다.

본 논문에서는 현재까지 발견된 코드 인젝션 기법들을 소개하고, 직접 구현하여 데이터 탈취 가능성을 확인하였다. 이를 통하여 공격자가 원하는 공격을 할 수 있음을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 코드 인젝션의 기법들을 기술하고, 3장에서는 기술한 기법들의 구현을 통해 데이터 탈취 가능성에 대해 논한다. 그리고 4장에서 결론으로 마무리한다.

II. 코드 인젝션 기법

본 장에서는 코드 인젝션의 기법들을 기술하였다.

2.1 쓰레드 인젝션(Thread Injection)

쓰레드 인젝션은 가장 일반적인 형태의 코드

인젝션 기법으로, CreateRemoteThread를 사용하여 원격 쓰레드의 형태로 실행을 시키는 기법이다.

우선, 인젝션할 공간을 마련하기 위해 인젝터 프로그램에서 VirtualAllocEx를 통해 대상 프로세스의 메모리 공간에 버퍼를 할당해준다. 공간 할당에 성공했을 경우, WriteProcessMemory를 통해 페이로드를 인젝션한다. 마지막으로, 인젝터 프로그램에서 CreateRemoteThread를 통해 쓰레드를 실행시킴으로써 페이로드가 작동하고 코드 인젝션에 성공한다. 쓰레드 인젝션을 시도할 때 주의해야 할 점은, 인젝터로부터 인젝션된 페이로드에서 사용되는 API가 대상 프로세스에 없을 경우 크래시(Crash)가 발생한다. 따라서 인젝터에서 구조체와 API를 새로 선언해 기존 형태의 API와 같은 형태로 설정한 후 호출해야 한다.

2.2 아톰바밍(AtomBombing)[3]

아톰바밍 기법은 2016년 10월 말에 공개된 코드 인젝션 기법[4]으로, 쓰레드 인젝션과는 다르게 CreateRemoteThread나 WriteProcessMemory 없이도 코드 인젝션이 가능하다. 또한 시스템 레벨의 아톰 테이블(Atom Table)을 악용한 기법으로, 최신 버전 Windows 10을 포함한 Windows OS의 모든 버전이 취약하다[5]. 취약한 코드에 의존하는 것이 아닌 OS 메커니즘이 설계된 방식을 악용했기 때문에[5] 패치를 하기 위해서는 OS의 설계 방식을 변경해야 하므로 사실상 패치가 불가능하다.

아톰 테이블이란 어플리케이션들이 정보를 문자열, 정수 및 다른 유형의 데이터를 저장할 수 있도록 하는 윈도우의 기능이다. 아톰 테이블은 크게 두 종류로 나눌 수 있는데, 로컬 아톰 테이블과 글로벌 아톰 테이블이다. 글로벌 아톰 테이블은 공유된 테이블이기 때문에 모든 종류의 어플리케이션들이 테이블에 저장된 데이터에 접근하거나 변경할 수 있다[5]. 따라서 아톰바밍 기법은 이러한 특징을 지닌 글로벌 아톰 테이블을 사용한다.

우선, 인젝터 프로세스는 대상 프로세스에서 실행시킬 쉘코드를 아톰 테이블에 추가한다. 그 후, 대상 프로세스가 아톰 테이블로부터 쉘코드를 가지고 오게 하도록 APC(Asynchronous Procedure Call)를 사용한다. 마지막으로 APC를 통해 조작된 대상 프로세스는 쉘코드를 가지고 온 후, DEP(Data Execution Prevention) 우회를 하기 위해 ROP(Return Oriented Programming) Chain을 사용한다. DEP란 데이터 실행 방지의 약자로, 메모리 영역에서 코드가 실행되는 것을 방지하는 것이다. 아톰바밍 기법은 총 3가지 단계가 존재한다[4].

1) Write-What-Where

아톰바밍 기법은 GlobalAddAtom과 GlobalGetAtomName을 사용한다. GlobalAddAtom은 글로벌 아톰 테이블에 문자열을 더하고, 해당 문자열을 식별하기 위한 고유한 값인 아톰을 반환하고, GlobalGetAtomName은 해당 글로벌 아톰에 해당하는 문자열의 복사본을 반환한다. 따라서 GlobalAddAtom 호출을 통해서 쉘코드를 글로벌 아톰 테이블에 저장하고, 대상 프로세스가 GlobalGetAtomName을 호출하게 만들면 WriteProcessMemory 없이 코드 인젝션이 가능하다[4]. 대상 프로세스에서 GlobalGetAtomName을 통해 쉘코드를 호출할 때 APC를 사용하는데, 임의의 APC 콜백 함수를 선언한 후 QueueUserAPC를 호출함으로써 콜백 함수를 실행한다. 하지만 해당 콜백 함수는 인자의 개수가 1개여야 하기 때문에 3개의 인자를 갖는 GlobalGetAtomName을 대상 프로세스에서 호출할 수 없다. QueueUserAPC는 내부에서 NtQueueApcThread를 호출하고, NtQueueApcThread는 내부에서 ntdll!RtlDispatchAPC를 호출한다. 이 때, 해당 API는 3개의 인자를 받는다. 따라서 GlobalGetAtomName을 대상 프로세스에서 호출하려면 NtQueueApcThread를 통해 호출할 수 있다.

2) Execution

1단계에서 GlobalGetAtomName을 통해 인젝션한 쉘코드는 DEP로 인해 실행을 할 수 없다. 따라서 DEP를 우회하기 위해 ROP Chain을 사

용한다. 우선, 읽기/쓰기/실행 권한이 있는 RWX메모리를 할당한다. 메모리 할당에 성공했을 경우, 쉘코드를 실행 권한이 없는 RW메모리에서 새로 할당된 RWX메모리로 복사한다. 따라서 쉘코드를 실행시킬 수 있게 된다.

3) Restoration

쓰레드를 하이재킹 한 후, 하이재킹 전의 동작으로 복원하지 않을 경우 어떤 형태의 충돌이 일어날지 알 수 없기 때문에 기존의 동작으로 복원해야 한다. APC 함수들을 분배하는 역할을 하는 함수인 ntdll!KiUserApcDispatcher를 사용해서 복원할 수 있다. ntdll!KiUserApcDispatcher는 내부에서 ZwContinue를 호출하는데, ZwContinue는 CONTEXT 구조체를 인자로 실행을 재개시킨다[6]. CONTEXT 구조체는 APC 함수를 호출하기 전의 EDI가 저장되어 있는데, 이 EDI를 쉘코드 도입부에서 백업한 후, 쉘코드 종료 직전에 복원을 해줌으로써 쓰레드를 복원할 수 있다[6].

2.3 PROPagate

2017년 10월에 Hexacorn에서 PROPagate라는 이름의 코드 인젝션 기법이 발표되었다[7]. PROPagate 기법 또한 모든 최신 Windows 버전에서 작동하며, Windows GUI나 유명한 GUI 프레임워크를 사용하는 어플리케이션에서만 작동된다. PROPagate는 새로운 서브 클래스 헤더를 삽입하거나 다른 프로세스에서 제어할 수 있는 콜백 함수를 포함하는 기존 서브 헤더를 수정하는 방식으로 작동한다. 서브 클래스 윈도우는 SetProp을 사용하여 업데이트할 수 있고, SetWindowLong이나 SetWindowLongPtr를 사용하여 Windows 콜백 프로시저를 업데이트하는 것과 매우 유사하다.

우선, Windows 7과 Windows 10 모두에서 작동하는 부모 윈도우인 “Progman”과 자식 윈도우인 “SHELLDLL_DefView”를 각각 반환하고[8], 서브클래스는 “UxSubclassInfo”를 반환한다. 대상 프로세스인 “explorer.exe”의 pid를 반환한 후, VirtualAllocEx를 통해 RW메모리와 RWX메모리를 할당함으로써 각각 서브클래스 헤더와 페이로드를 인젝션한다. 인젝션 후, 서브클래스의 헤더인 pfnSubclass 필드를 페이로드의 주소로 설정하고 SetProp API를 통해 서브클래스의 프로시저를 페이로드의 주소로 업데이트한다. PostMessage API를 통해 “explorer.exe”에 WM_CLOSE 메시지를 보내면 페이로드가 실행된다.

2.4 기타 코드 인젝션 기법

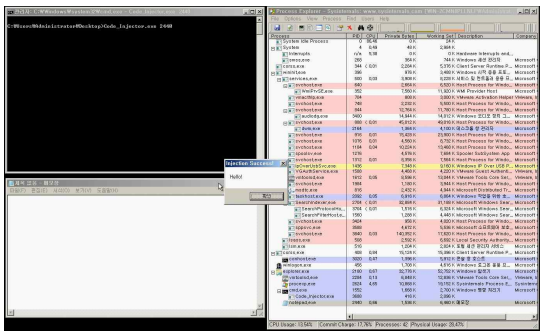
2018년 4월에 발견된 Early Bird라는 새로운 코드 인젝션 기술이 발견되었다[9]. 해당 기법은 악성코드를 쓰레드 초기화의 매우 이른 단계에서 로드하여, 많은 보안 솔루션들이 훅을 배치하기도 전에 로드될 뿐만 아니라 탐지되기 쉬운 API에 의존하지 않아 탐지를 회피한다[9]. 아톰바밍 기법과 유사하게 프로세스에 메모리를 할당받고, APC를 사용한다. 차이점은 Early Bird에서는 NtTestAlert를 통해 메인 쓰레드가 재시작되기 직전에 커널이 악성코드를 실행하도록 한다. 이외에도 ConsoleWindowClass를 사용하는 Conhost 기법, 추가적인 윈도우 바이트를 사용하는 ExtraBytes 기법, 서비스 컨트롤 핸들러를 사용하는 SvcCtrl 기법 등이 존재한다.

III. 구현

본 논문에서 기술한 코드 인젝션 기법 중 쓰레드 인젝션과 아톰바밍 기법, PROPagate 기법을 실제로 구현하였다. 구현 환경은 Windows 7 x86이다.

3.1 쓰레드 인젝션

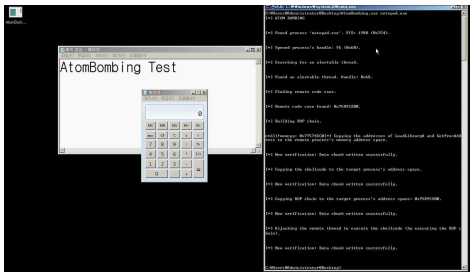
인젝터를 실행시킬 때 인자 값으로 대상 프로세스의 pid를 입력하면 [그림 1]과 같이 인젝션에 성공했다는 메시지 박스가 띄워진다. 만약 메시지 박스를 띄우는 페이로드가 아닌 악성코드 드롭, DNS 변조, 컴퓨터 정보 탈취 등의 악의적인 행위였을 경우, 충분히 데이터 탈취가 가능했을 것이다.



[그림 1] 쓰레드 인젝션 구현

3.2 아톰바밍

인젝터를 실행시킬 때 대상 프로세스명을 입력하면 대상 프로세스의 pid와 핸들을 반환하고, ROP Chain을 시작한다. ROP Chain에 성공했을 경우, 코드 인젝션에 성공했다는 표시로 계산기가 띄워진다. 만약 계산기를 띄우는 페이로드가 아닌 악성코드 드롭, DNS 변조, 컴퓨터 정보 탈취 등의 악의적인 행위였을 경우, 충분히 데이터 탈취가 가능했을 것이다.



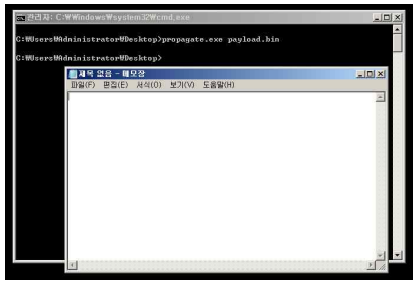
[그림 2] 아톰바밍 구현

3.3 PROPagate

인젝터를 실행시킬 때 인자 값으로 페이로드 파일의 경로를 입력하면, 코드 인젝션에 성공했다는 표시로 메모장이 띄워진다. 만약 메모장을 띄우는 페이로드가 아닌 악성코드 드롭, DNS 변조, 컴퓨터 정보 탈취 등의 악의적인 행위였을 경우, 충분히 데이터 탈취가 가능했을 것이다.

```
v4 = 'EniW';                                    // WinExec
v5 = 'cex';
v6 = 'eton';                                    // notepad
v7 = 'dap';
v2 = sub_DC((int)&v4, a1, a2) - 1;
if ( v2 )
  ((void (__fastcall *)(char *, signed int))(v2 - 1))((char *)&v6 + 2, 5);
return -1;
```

[그림 3] payload.bin 동작 루틴



[그림 4] PROPagate 구현

IV. 결론

본 논문에서는 코드 인젝션 기법들을 상세히 알아보았고, 코드 인젝션 기법의 일부를 실제로 구현하면서 데이터 탈취 가능성을 연구하였다. 구현 결과, 세 가지 기법 모두 Windows 7에서 동작하였다. 쓰레드 인젝션의 경우, 가장 흔한 기법이기 때문에 많은 보안 전문가들이 활발히 탐지 및 대응 방법을 연구하고 있다. 쓰레드 인젝션에서 주로 사용되는 API인 WriteProcessMemroy와 CreateRemoteThread를 후킹 하는 방법이 탐지 및 대응 방안이 될 수 있지만, 기본적으로 여러 경우의 수를 따진 후 화이트리스트를 구성해야 부작용이 덜 할 것이다. 또한, 메모리를 검사해서 코드 인젝션의 여부를 검사하는 방법도 존재하지만, 이 방법 또한 상당히 구현이 어렵다. 뿐만 아니라 Early Bird와 같은 새로운 메커니즘의 코드 인젝션 기법이 지속해서 발견되고 있으므로, 코드 인젝션에 대한 지속적인 연구가 필요하다. 게다가, 최근 악성코드에서 코드 인젝션을 자주 사용하고, 특히 피해가 큰 금융 분야에서도 정보 탈취 등의 공격이 이루어지고 있으므로 코드 인젝션의 대응 방안에 대한 연구 또한 같이 이루어져야 한다.

Acknowledgment

"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음"(IITP-2019-2016-0-00304*)

[참고문헌]

[1] ThreatExpertBLOG, “Malware Employs a Memory Injection Technique Patented by Microsoft”, http://blog.threatexpert.com/200

8/02/malware-employs-microsofts-patent-on.html

[2] k3y6reak's lab, "Code Injection", http://kblab.tistory.com/289

[3] atom-bombing, https://github.com/BreakingMalwareResearch/atom-bombing

[4] BreakingMalware, "AtomBombing: Brand New Code Injection for Windows", https://breakingmalware.com/injection-techniques/atombombing-brand-new-code-injection-for-windows/

[5] ESTsecurity, "모든 윈도우 버전 공격 가능한 코드 인젝션 기술 'Atom Bombing' 발견", http://blog.alyac.co.kr/861

[6] reversenote, "AtomBombing - Stage 3", http://www.reversenote.info/atombombing-stage3/

[7] hexacorn, "PROPagate - a new code injection trick", http://www.hexacorn.com/blog/2017/10/26/propagate-a-new-code-injection-trick/

[8] modexp, "Windows Process Injection: PROPagate", https://modexp.wordpress.com/2018/08/23/process-injection-propagate/

[9] ESTsecurity, "해커들, 탐지를 피하기 위해 새로운 코드 인젝션 기술 사용하는 것으로 나타나", http://blog.alyac.co.kr/1628

Key-value쌍 입력을 통한 CGI Fuzzing

정재영[1] ,조성준[1], 김창현[1]

1. 한양대학교

A Study on CGI Fuzzing via key-value input

Jae-Young Jeong[1] Seong-Joon Cho[1] Chang-Young Kim[1]

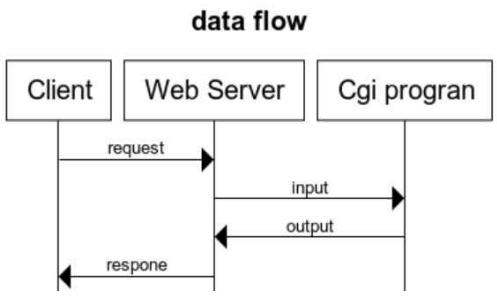
1. Hanyang University

요 약

공용 게이트웨이 인터페이스 (이하 CGI)는 웹서버 상에서 사용자의 프로그램을 실행시키기 위한 기능이다. 웹서버는 클라이언트의 요청에 따라 CGI를 실행시킨 후 그 결과값을 클라이언트에게 전송할 수 있다. CGI에 취약점이 존재할 경우 서버의 제어권을 클라이언트에게 탈취당할 수 있다. 따라서 CGI의 취약점을 발견하여 패치하는 작업이 필요하다. AFL은 취약점을 발견하는 가장 인기있는 도구 중 하나임에도, AFL을 이용해 CGI을 퍼징하는 방법에 대해서 충분한 연구가 이루어지지 않고 있다. 해당 연구에서 CGI의 취약점을 점검하기 위해 key - value 쌍을 통한 퍼징을 제안한다. 본 방식의 유효성을 검증하기 위해 My Cloud Expert Series EX2 Ultra[4]의 펌웨어에 대하여 해당 기술을 적용시켜본 결과 32가지의 유니크한 크래시를 얻었다. 이중 10가지 버그가 익스플로잇 가능한 취약점이었다.

I. 서론

cgi는 클라이언트와 웹서버가 상호간의 정보교환을 통해 동적인 웹페이지를 만들기 위해 도입되었다. 웹서버는 클라이언트에게 전달받은 파라미터를 CGI에 전달한다. CGI가 실행된 이후 웹서버는 CGI의 결과값을 클라이언트에게 전송한다. 이러한 방식을 통해 웹서버는 클라이언트의 요청에 따라 데이터를 생성하여 응답할 수 있게 된다.



[그림 1] CGI를 사용하는 웹서버의 데이터 흐름

웹서버는 클라이언트의 요청에서 특정한 값들을 환경변수의 형태로 CGI에 전달한다. [1] 클라이언트가 GET 형식으로 요청한 파라미터들은 key - value 쌍으로 환경변수 QUERY_STRING에 등록되어있다. 예를 들어 클라이언트가 “http://url/test.cgi?key1=value1&key2=value2”으로 요청을 보낸 경우를 가정하자. 이 경우 key - value 쌍은 {key1:value1, key2=value2} 으로 나타낼 수 있으며, “QUERY_STRING=key1=value1&key2=value2”가 환경변수로 등록된다.

CGI에 취약점이 존재할 경우 서버의 제어권이 클라이언트에게 넘어갈 수 있다. [2] 보안 사고를 미연에 방지하기 위해 CGI에 존재하는 취약점들을 탐지한 후 패치해야 한다.

American Fuzzer Lop[2]는 프로그램의 취약점을 점검하는 가장 인기있는 도구 중 하나이지만, 이를 이용하여 CGI를 퍼징하는 방법에

대해서는 충분히 연구가 이루어 지지 않았다. AFL는 환경변수가 아닌 표준입력을 통해 입력하기 때문에 CGI에게 파라미터를 전달 할 수 없다. 또한 AFL은 key-value 입력이라는 특수성을 고려하지 않는다. 본 논문의 본론에서는 ① 환경변수를 통해 CGI에게 파라미터를 전달하고 ② key – value 쌍을 입력을 퍼징하기 위한 퍼저를 구현하는 방법에 대하여 서술한다.

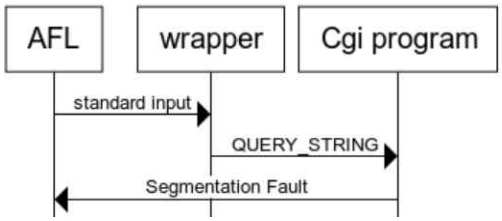
II. Cgi 퍼징

2.1 cgi wrapper for afl

AFL과 CGI 간의 입력 방식 차이 문제를 해결하기 위해 wrapper를 도입하였다. 부모 프로세스가 execve를 통해 자식 프로세스를 생성하면 부모 프로세스의 환경변수는 자식 프로세스에게 상속된다. 표준입력으로 입력받은 값을 환경변수 *QUERY_STRING*로 등록한 후 execve로 CGI를 실행시키는 wrapper를 통해 표준입력을 환경변수로 변환할 수 있다. AFL이 표준입력으로 파라미터를 전달하면 wrapper가 파라미터를 환경변수에 등록한 후 CGI에게 전달해준다.

```
... 생략
        scanf("%s",parameter);
        setenv("QUERY_STRING",parameter,1);
        execv("./victim.cgi",NULL,environ);
... 생략
```

[그림 2] 표준입력으로 입력받은 값을 CGI에 환경변수로 전달하는 코드



[그림 3] wrapper를 통해 AFL이 CGI를 퍼징할 때 데이터 흐름

2.2 wrapper generator

wrapper를 생성을 자동화하기 위해 wrapper generator을 이용한다. wrapper generater가 wrapper를 생성하는 방식은 다음과 같다. cgi 파라미터의 value의 최대길이와 key를 입력받는다. 입력받은 key를 하드코딩하여 코드에 저장한다. 표준입력을 통해 value 를 입력받는 코드를 생성한다. 입력받은 key와 value를 '=' 과 '&' 를 이용하여 연결하고 QUERY_STRING에 등록하는 코드를 생성한다. execve로 CGI를 실행시키는 코드를 생성한다. 이전의 과정을 통해 만들어진 코드를 컴파일한다.

Wrapper generator의 wrapper 생성 방식을 보면 key와 value를 다른 방식으로 전달하는 것을 볼 수 있다. 파라미터의 value는 사용자의 입력에 따라 바뀌지만, key는 웹서버에서 미리 지정한 값만이 사용되기 때문이다. AFL에서 key를 변이를 할 경우 key가 웹 사이트에서 사용하는 값에서 벗어나게 된다. 웹서버는 사용범위에서 벗어난 key를 사용하지 않기 때문에, key가 변이되어도 CGI의 실행흐름는 바뀌지 않는다. 따라서 AFL이 key를 제외한 value만을 변이하는 것이 바람직하다. wrapper generator를 통해 wrapper 코드를 생성할 때 key를 하드코드하면 key가 AFL를 통해 전달되지 않음으로 key에 대한 변이는 이루어지지 않는다.

```
def makeWrapperFile(self, funcArgs, cgiName, outdir):
    cgiPath = cgiAbsPath+"/"+cgiName
    wrapper_code ='''
#include<stdio.h>
#include<stdlib.h>

extern char ** environ;
void main(int argc, char *argv[]){
  char s[0x60000] = {};
  char q[0x60000] = {};
'''[1:]
    for size, key in funcArgs.items():
      wrapper_code += '  scanf("%%%ds", q);\n'%(size)
      wrapper_code += '  strcat(s, "&%s=");\n'%(key)
      wrapper_code += '  strcat(s, q);\n'

    wrapper_code += '  setenv("QUERY_STRING",s,1);\n'
    wrapper_code += '  setenv("REQUEST_METHOD","GET",1);\n'
    wrapper_code += '  execv("%s", NULL, environ);\n}\n'%(cgiPath)
    open(outdir + "/output.c",'w').write(wrapper_code)
    os.system('arm-linux-gnueabi-gcc -o %s/wrapper %s/output.c'%(outdir,outdir))
```

[그림 4] wrapper generator에서 wrapper를 생성하는 코드. value는 표준입력으로 입력받고 key는 하드코딩한다. value와 key를 연결한 후 환경변수에 등록하고, execve로 CGI를 실행한다.

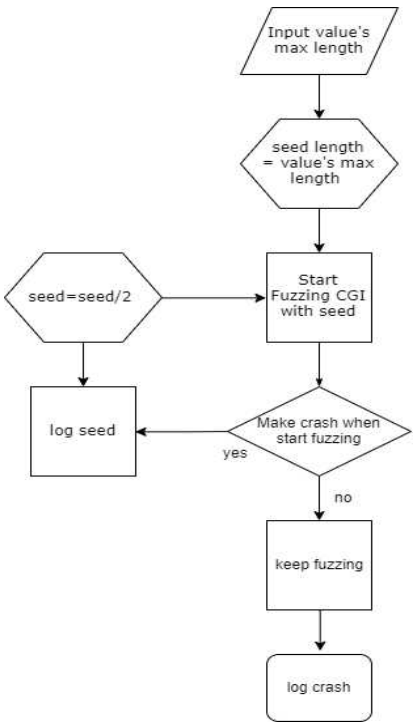
2.3 시드 생성

value에 대한 시드는 사용자가 직접 선정하는

것을 권장한다. 하지만 다수의 CGI에 대하여 일일이 적절한 시드를 선정하는 것은 쉽지 않은 일이다. 따라서 본 논문에서는 시드가 주어지지 않았을 때 적당한 효율을 가지는 시드를 생성하는 방법을 제안한다.

value의 최대 길이를 가지는 seed를 생성한다. 해당 시드를 이용하여 CGI를 퍼징한다. 시드를 입력했을 때 CGI에서 크래시가 발생할 경우 AFL이 에러와 함께 종료되기 때문에 value값을 로그로 남긴다. seed의 길이를 절반으로 줄인 후 다시 퍼징을 시작한다. 이것을 AFL이 에러없이 실행될 때까지 반복한다.

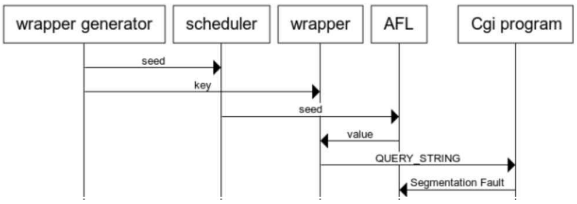
이 방법은 버퍼 오버플로우 버그를 찾는데 뛰어난 효과를 보인다. 버퍼 오버플로우는 다수의 프로그램에서 발견되는 버그이며[3], 공격가능할 가능성이 높은 심각한 취약점이다. 따라서 이러한 시드 생성 방법은 사용자가 적절한 시드를 설정할 수 없을 때 차선책이 될 수 있을 것으로 보인다.



[그림 5] seed를 생성하는 순서도.

2.4 스케줄러

한번에 실행되는 AFL의 개수, 하나의 CGI에 대해서 퍼징하는 시간을 지정한 후 입력된 폴더 내의 모든 CGI에 대하여 퍼징하는 scheduler를 구현하였다. scheduler를 통해 다수의 CGI에 대하여 효율적으로 퍼징할 수 있게 된다.



[그림 6] 스케줄러가 적용되었을 시 데이터의 흐름.

2.4 방법론 유효성 검증

앞에서 제시한 방법들의 유효성을 테스트하기 위해 My Cloud Expert Series EX2 Ultra의 최신버전(2.31.195) 펌웨어[4]에서 추출한 CGI 파일을 대상으로 테스트하였다. 시드로 유효하지 않은 시드값인 “aaa “를 넣었을 때에 5개의 크래시를 얻을 수 있었다. 반면 본논문에서 제시한 시드 생성 방법을 사용한 경우 32개의 유니크한 크래시를 얻을 수 있었다. 크래시 분석 결과 32 개중 10개의 크래시는 익스플로잇 가능한 취약점으로 판별되었다.

III. 결론

본 논문에서는 CGI의 파라미터 전달방식에 대해 밝힌 후 key – value 쌍 입력을 통해 AFL을 이용하여 CGI를 퍼징할 수 있는 방법을 제안하였다. 제안한 방법의 유효성을 검증하기 위하여 My Cloud Expert Series EX2 Ultra의 펌웨어에 존재하는 CGI에 해당 방법을 적용하여 퍼징하였다. 그 결과 33개의 크래시를 얻을 수 있었고, 그 중에서 10개는 익스플로잇 가능한 취약점이었다.

[참고문헌]

[1] International Business Machines Corporation . CGI 스크립트의 환경 변수 . https://www.ibm.com/support/knowledgecenter/ko/SSSHTQ_8.1.0/com.ibm.netcool_OMNIbus.doc_8.1.0/webtop/wip/reference/web_cust_envvariablesincgiscripts.html

[2]M. Zalewski. american fuzzy lop . http:/

/lcamtuf.coredump.cx/afl, 2018
[3] Steve Christey, Robert A. Martin. http://cwe.mitre.org/documents/vuln-trends/vuln-trends.pdf, 2007
[4] Western Digital Corporation. https://support-en.wd.com/app/products/product-detail/p/130#WD_downloads

< 저자 소개 >

정재영 (Jae-Young Jeong)
2017년~현재 : 한양대 공과대학 재학
조성준 (Seong-Joon Cho)
2017년~현재 : 한양대 공과대학 재학
김창현 (Chang-Young Kim)
2017년~현재 : 한양대 공과대학 재학

RAW 이미지 뷰어 취약점 분석

황선홍, 이병천*

*중부대학교 정보보호학과

Vulnerability Analysis of RAW Image Viewer

Sunhong Hwang, Byoungcheon Lee*

*Department of Information Security, Joongbu University.

요 약

RAW 이미지 파일은 이미지 파일 포맷 중 하나로 디지털 카메라나 이미지 스캐너의 이미지 센서로부터 얻어진 데이터를 최소한만 처리한 원본 데이터를 포함하고 있다. RAW 이미지 파일은 일반적으로 널리 사용되는 손실 압축 이미지 파일 포맷인 JPEG, PNG 보다 높은 화질을 가지고 있지만 파일 크기가 크고 해당 파일을 PC에서 열어보기 위해서는 별도의 소프트웨어가 필수적으로 요구된다.

본 논문에서는 이러한 RAW 이미지 파일 처리를 위한 이미지 뷰어 소프트웨어들의 취약점을 효과적으로 분석하기 위해 RAW 이미지 파일의 구조를 분석하고, 어떤 필드의 값이 이미지 뷰어의 취약점을 도출하는데 유용한 값인지 조사하였으며 결론적으로 이를 통해 X사의 이미지 뷰어에서 임의 코드를 실행할 수 있는 취약점을 도출하는데 성공하였다.

I. 서론

RAW 이미지 파일[1]은 이미지 파일 포맷 중 하나로 DSLR(Digital Single-Lens Reflex camera), 디지털카메라, 이미지 스캐너 등의 이미지 센서로부터 얻은 데이터를 최소한만 처리한 원본 데이터를 포함하고 있다. RAW 이미지 파일들은 가공되지 않은 원본 상태이며 디지털 카메라 제조사에 따라 여러 가지 다른 확장자를 가지고 있다.

이러한 RAW 이미지 파일을 사용하는 데에는 많은 이점이 있는데, 가공되지 않은 상태의 파일이기 때문에 일반적인 이미지 파일보다 더 높은 화질을 갖고 있다. 예를 들어 JPEG(Joint Photographic coding Expert Group)와 비교하면 JPEG은 손실 압축 포맷인데 반해 RAW 포맷은 압축되지 않거나 무손실 압축을 사용하므로 이미지 원본을 그대로 가지고 있다. 또한 카메라의 RAW 파일들은 12bit 혹은 14bit의 명암 정보를 가지고 있으며, 가공된 TIFF나 JPEG 파일에 저장되어있는 감마 압축 8bit를 사용하지 않는다. 따라서 데이터는 그림자, 밝은 부분, 채도가 깊은 색에 더 정확성을 제공한다는 장점이 있다.

하지만 이러한 RAW 파일들은 일반적으로 JPEG 파일보다 약 2~6배 정도 용량이 크고, 카메라 제조업체 대부분이 RAW 포맷을 사용하고 있지만 표준 RAW 포맷이 별도로 지정된 것이 아니기 때문에 각자 다른 포맷을 사용하고 있다. 따라서 일반적인 PC에 설치된 이미지 뷰어를 이용하여 파일을 열어볼 수가 없기 때문에 별도의 소프트웨어 설치가 요구되는데, 본 논문에서는 이러한 RAW 파일을 열어보기 위해 널리 사용되는 이미지 뷰어들에 대한 취약점 분석을 위해 RAW 파일을 hex단위로 분석하여 각 필드의 정보를 조사하였고, 필드의 데이터를 고의적으로 조작하는 공격을 시도하였다. 결과적으로 X사의 이미지 뷰어에서 임의 코드를 실행할 수 있는 취약점을 발견하였다.

II. RAW 이미지 파일 포맷

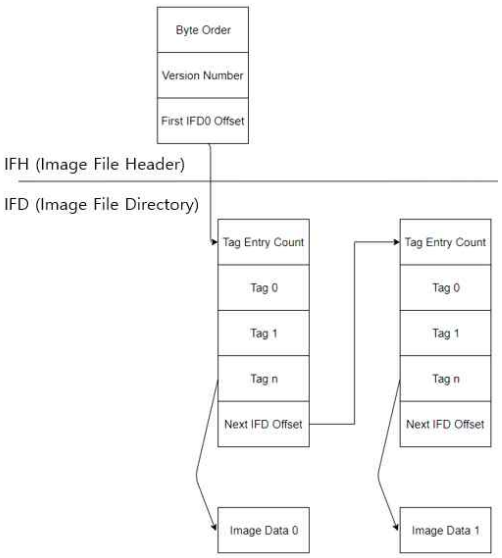
RAW 이미지 파일의 포맷은 [표 1]과 같이 DSLR 카메라 제조사별로 다르고, 정확한 포맷에 대한 정보는 카메라 제조업체의 지적재산권을 이유로 공식적으로 공개되지 않고 있다. 하지만 대부분의 RAW 이미지 파일이 TIFF(Tag Image File Format)를 기반으로 변형되었기 때문에 이를 토대로 분석이 가능하다.

[표 1] 제조사에 따른 RAW 이미지 파일 포맷

제조사	파일 확장자
FUJIFILM	.raf
Canon	.crw .cr2 .cr3
SAMSUNG	.srw
Nikon	.nef .nrw
Olympus	.orf
Adobe	.dng
PENTAX	.ptx .pef

2.1 TIFF 파일 분석

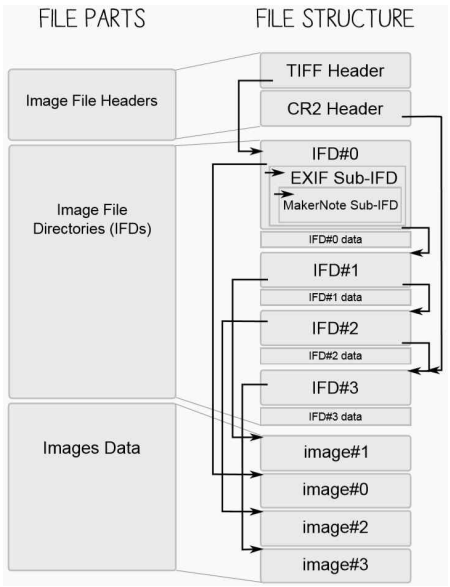
TIFF는 Aldus사와 마이크로소프트사가 공동으로 개발한 이미지 저장 포맷이며 사용자가 고쳐서 쓸 수 있는 유연한 특징이 있다. TIFF는 [그림 1]과 같이 IFH(Image File Header), IFD(Image File Directory), Bitmap Data등으로 구성된다.



[그림 1] TIFF 파일 구조

IFH는 2byte의 Byte-Order Identifier와 2byte의 TIFF 버전 정보, 4byte의 첫 번째 IFD에 관한 Offset정보를 가지고 있으며 IFD는 해당 IFD에 태그가 몇 개 존재하는지를 나타내는 값인 2byte의 NumDirEntries, 12byte의 태그 배열, 다음 IFD의 Offset정보를 나타내는 4byte의 값이 존재한다.

2.2 Canon사의 CR2 포맷 분석



[그림 2] CR2 포맷의 구조

CR2 포맷은 Canon사의 RAW 이미지 파일 포맷으로 Canon RAW Version 2를 의미하고, Canon사의 디지털 카메라에 의해 생성된 파일이다. CR2 포맷은 TIFF를 기반으로 하며 고품질, 비 압축, 비교적 크기가 큰 특징을 갖는다. CR2 포맷은 [그림 2]와 같이 크게 IFH, IFD, Image Data로 나뉘고, IFH에 TIFF Header 정보와 CR2 Header 정보가 위치한다. 또한 총 4개의 IFD를 가지고 있는 구조를 가지며 TIFF 구조에서 첫 번째 IFD인 IFD0에 Maker Note 등의 내용을 추가하였다. IFD0~2에 3개의 JPEG 이미지를 포함하고 있고, IFD3에 실제 RAW 이미지 파일이 위치한다.

III. RAW 이미지 뷰어 취약점 분석

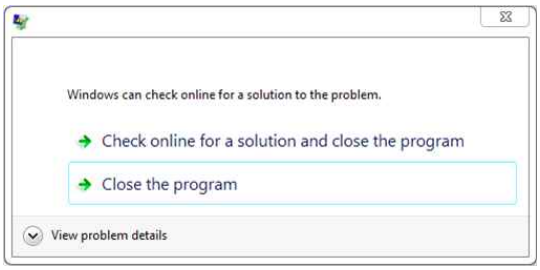
3.1 RAW 이미지 파일 변조 및 공격

앞서 조사한 내용을 바탕으로 CR2 파일을 이용하여 Smart 퍼징을 수행하였다. 취약점 분석의 대상이 되는 소프트웨어는 [표 2]와 같이 RAW 이미지 파일을 다루는 10개의 이미지 뷰어로 선정하였으며 본 논문에서는 CR2 포맷의 IFD0영역을 변조하며 Crash를 모니터링하였다.

[표 2] 취약점 분석 대상 이미지 뷰어

개발사	소프트웨어 명
Irfan Skiljan	IrfanView
FastStone Soft	FastStone
Sharpened Productions	File Viewer Plus
개인	Free Raw Viewer
SONY	RAW Viewer
개인	RawViewer
XnSoft	Xnview
GNU General Public License	UFRaw
Graphic-Region Development	Able RAWer
ideaMK	RAW Image Viewer

3.2 Crash 발생 및 원인



[그림 3] 이미지 뷰어에서 발생한 Crash

앞서 변조한 CR2 파일에 의해 10개의 이미지 뷰어 소프트웨어 중 5개의 소프트웨어에서 [그림 3]과 같이 운영체제의 Crash 메시지를 확인하였는데, Crash가 발생하는 전형적인 원인은 PC(Program Counter)레지스터가 부정확한 주소로 설정되거나 Buffer Overflow와 같은 버그 때문에 발생한다. 해당 Crash는 Model 태그의 Count 및 Tag_value를 정상적인 파일보다 큰 값으로 변조하였을 때 발생하였다. 해당 소프트웨어들에는 이에 대한 예외처리가 마련되어있지 않아 파일에서 정상적인 값보다 더 많은 값을 읽어오게 되었고 이는 Stack-Based Buffer Overflow를 야기하게 되었으며 X사의 소프트웨어에서는 이를 통해 PC레지스터를 조작하여 임의 코드를 실행할 수 있었다.

[그림 4]는 X사에서 발생한 Stack-Based Buffer Overflow 취약점을 이용하여 PC레지스터를 조작하는 것이 가능한 행위라는 것을 계산기를 실행함으로서 증명한 예시이다. 만약 공격자가 이를 악의적으로 조작한다면 X사의 소프트웨어가 설치되어 있는 모든 사용자를 대상으로 공격자가 원하는 모든 행위를 수행할 수 있다.

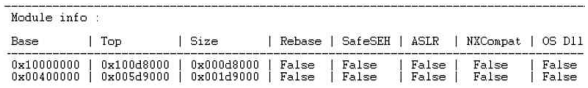


[그림 4] 이미지를 이용한 임의 코드 실행

3.3 취약점 개선 방안

Crash가 발생한 이미지 뷰어는 모두 CR2 포맷의 RAW 이미지 파일에서 변조된 Model 태그의 Count, Tag_value 값에 의해 발생하였는데 소프트웨어가 변조된 태그에 해당되는 값을 parsing할 때 보다 큰 값을 읽어오면서 발생하였다. 따라서 개발사는 이와 같이 비정상적인 값이 세팅되었을 때 parsing을 진행하지 않고 예외처리를 할 수 있는 루틴을 추가해야 될 것이다.

또한 일부의 이미지 뷰어에서 [그림 5]와 같이 메모리 보호기법을 확인하였을 때 DEP(Data Execution Prevention)와 같은 메모리 보호 기법이 적용되어있지 않아 쉘코드를 이용한 공격이 가능하였는데, 만약 DEP가 적용되어 있다면 쉘코드를 Stack 영역에 저장하더라도 실행권한이 없기 때문에 쉘코드가 실행되는 것을 방지할 수 있다. 따라서 소프트웨어의 메모리 보호기법의 적용 또한 요구된다.


Module info :

Base	Top	Size	Rebase	SafeSEH	ASLR	NXCompat	OS Dll
0x10000000	0x100d8000	0x000d8000	False	False	False	False	False
0x00400000	0x005d9000	0x001d9000	False	False	False	False	False

[그림 5] 메모리 보호기법 확인

IV. 결론 및 향후 연구

4.1 결론

RAW 이미지 파일의 포맷은 DSLR 카메라의 제조사에 따라 상이하며 제조사의 지적재산권을 침해한다는 이유로 일반인에게 공개되지 않고 있다. 하지만 대부분의 RAW 이미지 파일이 TIFF를 기반으로 변형된 포맷이기 때문에 이를 통해 분석이 가능하였고, 본 논문에서 조사한 CR2 포맷 RAW 이미지 파일의 IFD0의 Model 태그의 Count, Tag_value 값을 비정상적인 값으로 변조하여 10개의 이미지 뷰어 소프트웨어 중 5개의 소프트웨어에서 취약점 도출의 가능성을 확인하였으며 X사의 소프트웨어에서는 임의 코드를 실행이 가능하다는 것을 증명하였다. 이를 통해 개발사는 이미지의 비정상적인 필드 값에 대한 예외를 처리하는 루틴을 추가하고, 메모리 보호기법을 적용하여 사용자에게 좀 더 신뢰도 높은 소프트웨어를 제공하는 노력이 필요하다.

4.2 향후 연구

본 논문의 실험에서는 RAW 이미지 파일의 구조에 대해 조사하고, data model을 직접 세팅한 뒤 퍼징을 수행하여 취약점 분석을 진행하였다. 하지만 이러한 진행 방식은 파일 구조에 대해 직접 조사를 하는 과정에서 많은 시간을 필요로 하기 때문에 다양한 필드 및 파일을 조사하기에는 어려움이 있었다. 따라서 자동으로 필드의 값을 변경하며 코드 커버리지를 판단하여 취약점을 도출하는 방식의 퍼저를 개발할 계획이다. 이를 통해 취약점을 도출하는 과정에서 시간이 단축될 것으로 예상되며 이미지 파일 뿐만 아니라 다른 포맷의 파일을 이용한 소프트웨어 취약점 분석에도 유용하게 사용될 것으로 예상된다.

또한 본 논문의 실험 결과로 도출된 취약점은 KISA에 제보하여 보안 패치를 요구할 계획이다.

[참고문헌]

[1] 노광현. DSLR 카메라의 RAW 파일 포맷 분석. 한국컴퓨터정보학회 학술발표논문집, 2009, 16.2: 89-92.

[2] 김동진; 조성제. 멀티미디어 플레이어에 대한 퍼징기반 취약점 분석. 정보과학회논문지: 컴퓨팅의 실제 및 레터, 2011, 17.2: 98-107.

[3] OKAMOTO, Takeshi. SecondDEP: Resilient computing that prevents shellcode execution in cyber-attacks. Procedia Computer Science, 2015, 60: 691-699.

[4] GRIECO, Gustavo; CERESA, Martín; BUIRAS, Pablo. QuickFuzz: An automatic random fuzzer for common file formats. In: ACM SIGPLAN Notices. ACM, 2016. p. 13-20.

최신식의 화이트 박스 퍼징 도구에 대한 성능 비교 평가

김주환, 이성호, 최세호, 윤주범*

*세종대학교 정보보호학과

Performance Comparison Evaluation for State-of-the-art White box Fuzzing Tools

Juhwan Kim, Sungho Yi, Seho Choi, Joobeom Yun*

*Department of information security, Sejong University.

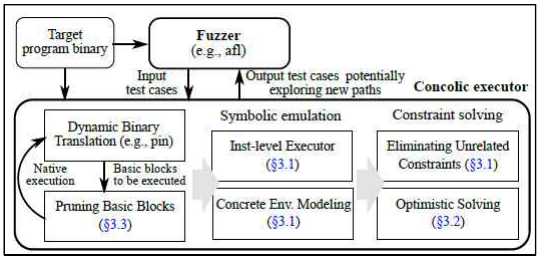
요 약

컴퓨팅 세계에 취약점을 탐지하는 것에 대한 중요성이 대두되면서 이를 대표하는 기법으로 퍼징이 존재한다. 퍼징은 특정 대상에게 특정 입력 값 또는 무작위 입력 값을 입력하여 대상의 상태를 모니터링하는 기법으로 취약점을 탐지하는데 주로 사용된다. 전통적인 퍼징은 대상에 대한 내부를 고려하지 않고 퍼징을 수행하기 때문에 입력 값이 대상에 깊은 공간까지 탐색이 되기에는 확률이 매우 적다. 이에 따라 본 논문에서는 소프트웨어 및 어플리케이션을 대상으로 화이트 박스 퍼징을 수행하는 최신식의 퍼징 연구들에 대해 분석하며, 실제 세계의 바이너리를 대상으로 퍼징을 수행하여 평가를 수행한다.

I. 서론

현대의 컴퓨터 보안에서는 소프트웨어(어플리케이션), 임베디드 디바이스, 네트워크 프로토콜 등 다양한 대상에 대해 취약점을 탐지하는 연구들[1][2][3]이 진행되어 왔고, 현재도 진행되고 있는 중이다. 사람이 수동으로 취약점을 탐지하는 것이 아닌 자동화된 시스템들이 연구되고 있으며, 미국의 방위 고등 연구 계획국인 DARPA는 사람의 개입 없이 자동화된 시스템들이 서로 주어진 바이너리를 대상으로 취약점을 탐지하고 익스플로잇 코드를 생성하면 점수를 획득하는 CGC(Cyber Grand Challenge)라는 대회를 개최했다. 이렇게 컴퓨팅 세계에 취약점을 탐지하는 것에 대한 중요성이 대두되면서 이를 대표하는 기법으로 퍼징(fuzzing)이 존재한다. 퍼징은 특정 대상에게 특정 입력 값 또는 무작위 입력 값을 입력하여 대상의 상태를 모니터링하는 기법으로 취약점을 탐지하는데 주로 사용된다. 퍼징은 퍼즈 테스팅(fuzz testing)이라고도 불리며, 전통적인 퍼징은 블랙박스 기반의 퍼징을 사용한다. 즉, 대상에 대한 내부를 고려하지 않고 퍼징을 수행하는 방식이다. 이는 간편하고 특별한 규격이 없다는 장점이 있지만, 입력 값이 대상에 깊은 공간까지 탐색이 되기에는 확률이 매우 적다. 이에 따라 화이트 박스 퍼징 기법이 제안되었으며, 이는 기호 실행(symbolic execution)과 같은 기법과 퍼징 기법을 결합하여 퍼징을 수행하는 방식이다. 이는 대상에 특정 공간이 퍼징이 탐색하기에 어려운 제약 조건이 있는 경우, 기호 실행을 호출하여 제약 조건을 해결해주고, 다시 퍼징으로 돌아와 더 깊은 공간을 탐색할 수 있게 된다.

본 논문에서는 위와 같이 소프트웨어 및 어플리케이션을 대상으로 화이트 박스 퍼징을 수행하는 최신식의 퍼징 연구들에 대해 분석하며, 실제 세계의 바이너리를 대상으로 퍼징을 수행하여 평가를 수행한다.



[그림 1] QSYM의 개요[4]

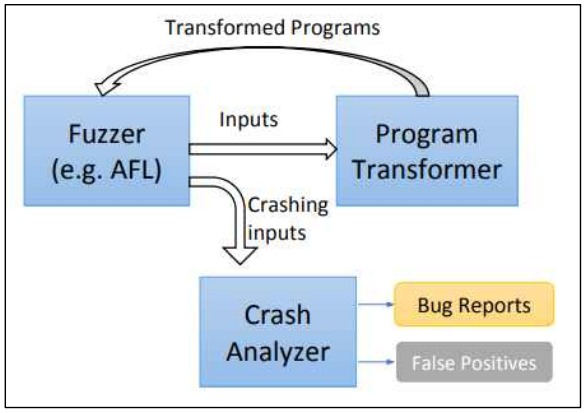
II. QSYM

QSYM[4]은 하이브리드 퍼징을 지원하는 콘콜릭(동적 기호) 실행 엔진으로 퍼저의 발판이 되는 역할을 한다. 즉, 퍼저가 탐색하기 어려운 경로에 대해 QSYM이 제약 조건을 해결하고 해당 경로로 향하는 입력 값을 생성해 퍼저에게 전달을 한다. 기존의 하이브리드 퍼징인 Driller[5]의 한계점을 파악하고 개선한 QSYM은 기존의 콘콜릭 실행 엔진이 에뮬레이션을 수행할 때 구현 복잡성을 줄이기 위해 중간 표현인 IR(Intermediate Representation)로 변환하는 과정이 에뮬레이션을 매우 느리게 만든다는 점을 입증했다. 이에 QSYM은 콘콜릭 실행 엔진의 최적화를 위해 새로운 접근법을 제안했으며, [그림 1]과 같이 핵심 아이디어는 DBT(Dynamic Binary Translation) 도구를 사용하여 instruction-level에서의 에뮬레이션을 수행하는 접근법을 제안했다. 실험 평가 부분에서 LAVA-M[6] 데이터 세트에서 VUzzer[7]보다 14배 이상의 버그를 찾았고, CGC 데이터 세트의 126개 바이너리 중 104개에서 버그를 찾아 Driller의 평가 결과를 능가했다. 뿐만 아니라, 실제 세계의 라이브러리에서 알려지지 않은 버그인 13개를 찾았다.

III. T-Fuzz

```
void main() {
  int fd  = open(...);
  char *hdr = read_header(fd);
  if (strncmp(hdr, "ELF", 3) == 0) {
    // main program logic
    // ...
  } else {
    error();
  }
}
```

[그림 2] NCC(Non-Critical Checks)의 예제



[그림 3] T-Fuzz의 개요[8]

T-Fuzz[8]는 QSYM처럼 콘콜릭 실행 엔진을 사용하여 제약 조건을 해결하는 특징을 갖지 않고, 전통적인 퍼징 접근법에 NCC(Non-Critical Checks)를 탐지하고 제거하는 특징을 가진 화이트 박스 퍼저이다. 즉, 원본 바이너리를 변환하여 퍼징을 수행한다는 특징을 가지고 있으며, NCC는 [그림 2]와 같이 magic value, checksum 검사처럼 버그를 방지하기 위한 검사가 아닌 것을 의미한다. 바이너리 내 NCC를 제거하여도 오류가 발생하지 않으며, 이와 동시에 퍼징이 단순해지는 장점을 갖게 된다.

[그림 3]은 T-Fuzz의 전체 구성 요소 및 흐름을 보여주며, 퍼저는 커버리지 가이디드(coverage guided) 퍼저이자 그레이 박스 퍼저인 AFL(American Fuzzy Lop)[9]을 사용한다. 먼저 퍼징이 수행되고 더 이상 퍼저가 새 경로를 찾지 못하는 'stuck' 상태에 빠지면, 프로그램 변환기가 호출되어 NCC 후보를 탐지하고, 탐지된 특정 NCC 후보들을 제거해 변형된 대상의 사본들을 생성한다. 이후 변형된 사본들을 대상으로 퍼징을 수행하며, 이는 계속해서 반복된다. 마지막 구성 요소인 크래시 분석기는 변형된 프로그램에서 발견된 크래싱 입력 값을 검증하여 오진(false-positive)의 여부를 확인하는 역할을 한다.

IV. 성능 평가

본 논문에서는 최신식의 화이트 박스 퍼저들

<표 1> 화이트 박스 퍼저들의 버그 발견 수

	cjpeg /libjpeg(v9c)	jpegtran /libjpeg(v9c)	wrjpgcom /libjpeg(v9c)	pngfix /libpng(1.6.37)	file /unix(5.31)
QSYM(AFL)	0	0	0	0	0
T-Fuzz(AFL)	25(opaque)	1(opaque)	80(opaque)	1(opaque)	0

<표 2> QSYM의 입력 값 생성 수

cjpeg	452
jpegtran	531
wrjpgcom	72
pngfix	593
file	546

의 성능을 비교하기 위해 평가를 수행했다. 실험을 수행한 환경은 Ubuntu 16.04, Intel i5-4460 @ 3.20GHz, 메모리 32GB의 호스트 컴퓨터에서 수행되었다.

퍼징의 대상은 실제 세계의 라이브러리로 선정했으며, 또한 라이브러리의 버전을 기존 연구에 실험했던 버전과 다르게 수행하고자 했다. 퍼징의 시간 제한은 대상마다 5시간을 수행했으며, QSYM과 T-Fuzz 모두 퍼저는 AFL을 사용했다.

실험의 결과는 <표 1>과 같으며, 대상은 프로그램/라이브러리(버전)로 총 5개의 프로그램을 퍼징했다. QSYM은 평가를 기대했던 것과 달리, 버그를 발견하지 못하였다. 그러나 <표 2>와 같이 QSYM의 에뮬레이터는 제대로 동작하였으며, 실제로 제약 조건을 해결해 입력 값을 생성한 개수이다. T-Fuzz는 file 프로그램을 제외한 나머지 모두에서 버그를 발견했고, wrjpgcom 프로그램에서 제일 많은 크래싱 입력 값을 발견했다. 하지만 T-Fuzz는 앞서 말했듯이, NCC를 제거한 바이너리를 복제하여 복제된 바이너리에 퍼징을 수행하기 때문에 크래시가 유발되는 입력 값을 찾아도 원본 바이너리에서도 크래시가 발생하는지 검증이 필요하다. 그러나 T-Fuzz의 크래시 분석기는 현재 오진으로 인해 작동이 중지되어 원본 바이너리에서의 크래시 유발을 확인할 수 없었다.

V. 결론 및 향후 계획

본 논문에서는 세계적인 보안 컨퍼런스에서 입증되고 발표된 화이트 박스 퍼저들의 성능 비교를 위해 분석하고 실제 세계의 프로그램을 대상으로 퍼징을 수행했다. QSYM의 경우에는 취약점이 패치된 마지막 버전의 라이브러리들을 대상으로 수행했기 때문에 버그가 쉽게 발견되지 않았으며, 게다가 file 프로그램은 QSYM에서 주장했던 것과 달리 동일한 버전임에도 불구하고 버그가 발견되지 않았다. 시간적인 제한과 시스템의 자원에 대한 차이로 인해 발견되지 않았을 것으로 추정하고 있다. T-Fuzz의 경우에는 현재 구성 요소 모듈 중 크래시 분석기가 중단되어 확실한 검증을 할 수 없었다. 이러한 한계점에 따라 우리의 향후 계획은 QSYM에 시간 제한을 두지 않고, 동일한 라이브러리에 버그가 발견되기까지 소요되는 시간을 평가해볼 것이며, 좀 더 최적화된 연구를 진행할 예정이다. 또한 T-Fuzz는 현재 크래시 분석기 구성 요소가 작동되지 않기 때문에 이를 해결하는 방향의 연구도 함께 진행할 예정이다.

[참고문헌]

[1] P. Godefroid, M. Y. Levin and D. A. Molnar, “Automated whitebox fuzz testing,” in Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, Feb. 2008.

[2] S. K. Cha, T. Avgerinos, A. Rebert, and D. Brumley, “Unleashing mayhem on binary code,” in Proceeding of the 33rd IEEE Symposium on Security and Privacy

(Oakland), San Francisco, CA, May 2012.

[3] M. Muench, J. Stijohann, F. Kargl, A. Francillon and D. Balzarotti, “What you corrupt is not what you crash: challenges in fuzzing embedded devices,” in Proceedings of the 25th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, Feb. 2018.

[4] I. Yun, S. Lee, M. Xu, Y. Jang, and T. Kim, “QSYM: A practical concolic execution engine tailored for hybrid fuzzing,” in Proceedings of the USENIX Security Symposium, 2018.

[5] N. Stephens, J. Grosen, C. Salls, A. Dutcher, R.Wang, J. Corbetta, Y. Shoshitaishvili, C. Kruegel, and G. Vigna, “Driller: Augmenting fuzzing through selective symbolic execution,” in Proceedings of the Symposium on Network and Distributed System Security, 2016.

[6] B. Dolan-Gavitt, P. Hulin, E. Kirda, T. Leek, A. Mambretti, W. Robertson, F. Ulrich, and R. Whelan, “LAVA: Large-scale automated vulnerability addition,” in Proceedings of the 37th IEEE Symposium on Security and Privacy (Oakland), San Jose, CA, May 2016.

[7] S. Rawat, V. Jain, A. Kumar, L. Cojocar, C. Giuffrida, and H. Bos, “VUzzer: Application-aware evolutionary fuzzing,” in Proceedings of the 2017 Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, Feb. - Mar. 2017.

[8] H. Peng, Y. Shoshitaishvili, and M. Payer, “T-Fuzz: fuzzing by program transformation,” in IEEE Symposium on Security and Privacy, ser. Oakland, 2018.

[9] M. Zalewski, “american fuzzy lop,” http://lcamtuf.coredump.cx/afl/, 2015.

커버리지 기반 퍼징 최신 기술 분석*

정세연, 정수창, 김서영, 조민기, 권태경
연세대학교 정보대학원 정보보호연구실

A Study on Analysis of Recent Coverage-based Fuzzing

Seyeon Jeong, Soochang Chung, Seoyoung Kim, Mingi Cho, and Taekyoung Kwon
Information Security Lab., GSI, Yonsei University

요 약

퍼징 (Fuzzing)은 대상 프로그램에 무작위 값을 입력하여 소프트웨어의 결함을 찾는 테스트 기법이다. 소프트웨어의 규모가 커짐에 따라 그 복잡도가 증가하고 있으며 이로 인해 기존 퍼징 기술이 새로운 취약점을 발견하는데 어려움이 발생한다. 따라서 복잡도가 높은 소프트웨어를 보다 효과적으로 테스트하기 위해서는 기존 퍼징 기술의 한계점을 분석하고, 이것을 극복하는 보다 나은 퍼징 기술 개발이 지속적으로 필요하다. 본 논문에서는 최신 퍼징 기술인 VUzzer, REDQUEEN, Angora, QSYM, Intriguer의 구조를 알아보고, 한계점에 대해 분석한다.

I. 서론

퍼징은 대상 프로그램에 무작위 값을 입력하여 소프트웨어의 결함을 찾는 테스트 기법 중 하나이다. 퍼징은 동적 분석으로 실행 상태를 관찰하고 분석함으로써 프로그램의 버그를 빠르게 발견할 수 있다. 하지만 소프트웨어 규모가 커짐에 따라 복잡도가 증가하고 있으며, 이로 인해 기존 퍼징 기술이 새로운 취약점을 발견하는데 어려움이 발생한다. 따라서 복잡도가 높은 소프트웨어를 보다 효과적으로 테스트하기 위해 새로운 퍼징 기술에 대한 연구가 지속적으로 필요하다.

본 논문에서는 최신 퍼징 기술인 VUzzer, REDQUEEN, Angora, QSYM, Intriguer의 구조에 대하여 알아보고, 한계점을 분석한다.

* 본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2019-2016-0-00304)

II. 배경 지식

2.1 퍼징 (Fuzzing)

소프트웨어 취약점 검증의 방법으로는 동적 분석과 정적 분석이 존재한다. 정적 분석은 프로그램을 실행하지 않고 소스코드 등을 사용하여 분석하는 것을 말한다[1]. 그러나 정적 분석은 높은 오탐 비율이 존재한다. 따라서 직접 프로그램의 실행을 통해 프로그램 버그를 정확하게 감지하는 동적 분석이 필요하다.

동적 분석을 사용하여 소프트웨어 취약점을 검증 하는 방법 중 하나로 퍼징이 있다. 퍼징은 대상 프로그램에 무작위 값을 입력하여 소프트웨어 결함을 찾는 테스트 기법 중 하나이다. 가장 대표적인 퍼징 기술로 AFL(American Fuzzy Lop)이 존재한다[2]. 그러나 AFL은 랜덤 변이 전략을 사용하기 때문에 새로운 분기문을 실행하기 위해 특정 조건이 필요한 경우 새로운 분기문을 실행하는데 어려움을 가진다. 또한 위치

와 값에 대한 정보를 갖지 않고 랜덤 변이를 수행하므로, 매직 바이트(Magic byte) 문제를 해결하는데 어려움이 존재한다.

높은 커버리지를 확보하지 못한다는 문제점이 존재하여 이러한 문제점을 해결 가능한 커버리지 기반 퍼징(Coverage-based Fuzzing)이 연구되었다.

2.2 커버리지 기반 퍼징

시간 대비 효율적인 퍼징을 수행하기 위하여 최근의 많은 퍼징 기술은 커버리지 기반 퍼징을 제안한다. 코드 커버리지가 높을수록 프로그램 실행 상태의 범위가 넓어지고 자세한 검사가 이루어지기 때문에 커버리지 기반 퍼징은 프로그램 계측과 같은 프로그램 분석 기술의 도움으로 변이 전략을 수립한다. 그러나 커버리지 기반 퍼징은 해결하기 어려운 제약 조건들이 존재한다.

2.3 하이브리드 퍼징

하이브리드 퍼징은 해결하기 어려운 제약 조건 문제로 인하여 깊은 경로를 탐색하기 어렵다는 한계점을 가진 퍼징 기술과 실행속도가 느리고 자원의 소모가 크다는 한계점이 존재하는 기호 실행 기술을 혼합한 퍼징 기술이다[3]. 기호 실행은 특정한 입력값을 기호 값으로 지정하고 프로그램의 취약점이 존재하는 경로에 도달할 수 있는 값을 찾는 기법을 말한다. 얕은 경로에서는 퍼징 기술을 활용하여 취약점을 빠르게 탐색하고, 더 이상의 경로 탐색이 어려울 경우 기호 실행을 사용하여 취약점을 탐색한다.

III. 최신 퍼징 기술 연구 동향

3.1 커버리지 기반 퍼징 연구

2017년 Rawat 등의 연구에서는 콘콜릭 실행의 단점을 보완하기 위해 동적 오염 분석을 사용하여 퍼징을 수행하는 VUzzer를 제안하였다[4]. 정적 분석과 동적 분석을 같이 수행하여 퍼징을 더욱 효과적으로 수행한다.

```
if( ((ox^(ox+dxw))
        | (ox^(ox+dxh))
        | (ox^(ox+dxw+ dxh))
        | (oy^(oy+dyw))
        | (oy^(oy+dyh))
        | (oy^(oy+dyw+ dyh))) >> (16 + shift)
        || (dxx | dxy | dyx | dyy) & 15
        || (need_emu && (h > MAX_H || stride > MAX_STRIDE)))
{ ... return; }
// the bug is here
```

그림 1 FFmpeg에서 복잡한 제약 조건을 만족하였을 때 발견할 수 있는 버그 [7]

정적 분석을 통해 cmp 명령어에 대한 정보와 control-flow에 대한 정보를 얻은 뒤 이를 활용하여 동적 분석을 수행한다. 동적 분석에서는 실행된 cmp 명령어들을 분석하여 매직 바이트 탐지를 수행하고 실행된 basic block을 분석하여 실행 빈도가 낮은 입력 파일에 대해 더 높은 우선순위를 주고 예외 처리 코드에 대해서는 낮은 우선순위를 주어 퍼징을 수행한다.

VUzzer는 복잡한 제약 조건을 해결하지 못한다는 한계점이 존재한다. 그림1은 VUzzer가 탐지하지 못하는 예시로, FFmpeg에서 존재하는 복잡한 제약 조건이다. VUzzer에는 복잡한 제약 조건을 해결하는 새로운 방법이 포함되지 않아 추가로 이를 해결하는 방안이 필요하다. 이러한 문제를 해결하기 위해 이후 연구에서는 콘콜릭 실행을 사용한다. VUzzer는 베이직 블록의 추적을 위해 실행 중에 발생하는 각 베이직 블록의 빈도 수를 기록하는 Pin tool을 구현한다. 그러나 VUzzer가 구현한 Pin의 테스트 속도는 매우 느려 오버헤드가 발생한다.

2018년 Aschermann 등의 연구에서는 입력의 일부분이 프로그램에서 거의 변화되지 않은 상태로 프로그램에 직접 사용되는 input-to-state correspondence 특성을 기반으로 하여 퍼징에서 발생하는 문제점을 해결하는 REDQUEEN을 제안하였다[5]. 이 때 REDQUEEN이 해결하는 기존 퍼징의 문제는 그림 2와 같은 매직 바이트 문제와 체크섬(Checksum) 문제가 있다. Taint tracking 과 기호 실행을 사용하지 않기 때문에 다른 퍼징 기술에 비해 더욱 빠르게 퍼징을 수행할 수 있다.

```
/* magic number example */
if(u64(input)==u64("MAGICHDR"))
  bug(1);

/* nested checksum example */
if(u64(input)==sum(input+8, len-8))
  if(u64(input+8)==sum(input+16, len-16))
    if(input[16]=='R' && input[17]=='Q')
        bug(2);
```

그림 2 매직 바이트 및 체크섬 문제 [5]

그러나 복잡한 조건으로 만족해야하는 분기문이 존재하는 경우 이에 대한 정확한 값을 도출하지 못한다는 한계점이 존재한다. 또한 문자열이 해시맵의 인덱스로 사용되는 경우 이를 제대로 처리하지 못한다.

2018년 Chen 등의 연구에서는 Mutation-based Coverage guided 퍼징으로, 기호 실행을 사용하지 않고 제약 조건을 해결하여 커버리지 범위를 향상시키는 Angora를 제안하였다[6]. 이 연구에서는 taint tracking, shape and type inference, gradient descent, input length exploration의 핵심기술을 제안하고 퍼징 능력을 향상시킨다.

Angora는 경로 제약 조건을 해결하기 위해 실행 시간이 오래 걸리는 기호 실행을 하지 않고, 기계학습에서 널리 사용되는 경사 하강법(Gradient descent)을 사용한다. 모든 제약 조건을 세가지 유형으로 단순화 시키고, 논리연산자가 포함되어 있으면 명령문을 여러 조건문으로 나눠 경사 하강법을 사용한다.

그러나 angora-clang 이라는 특수한 컴파일러로 taint tracking 버전과 instrument를 위한 버전으로 대상 바이너리를 두 번 컴파일하는 과정이 필요하다. 이것을 위해 퍼징 대상 바이너리의 소스코드가 요구된다. 이러한 사항은 pin tool을 구현하여 개선 가능하다.

3.2 하이브리드 퍼징 연구

2018년 Yun 등의 연구에서는 기존 콘콜릭 실행(concolic execution)의 문제점을 개선하여 기존 방법보다 더욱 빠르게 하이브리드 퍼징을 수행하는 QSYM을 제안하였다[7]. 기존 콘콜릭 실행 방법이 가지는 병목현상을 해결하고 Dynamic Binary Translation (DBT)를 사용하여 기호 실행을 순수한 실행과 밀접하게 통합하였다. 또한 optimistic solving과 Basic block pruning을 통해 기존 콘콜릭 실행의 엄격한 요구사항을 완화하였다.

QSYM의 동작 구조로는 우선 instrument를 한 뒤 Dynamic Binary Translation (DBT)와 커버리지 기반 퍼징에서 제공하는 입력을 사용하여 대상 프로그램을 실행한다. DBT는 그대로 실행되는 베이직 블록을 만들고, 그 중 기호 실행을 위한 블록들을 선별하여 두 실행 모델 사이의 전환을 원활하게 한다. 그다음 기호 제약 생성을 위해 필요한 명령어만 선택적으로 에뮬레이션한다. 이러한 효율적인 구조를 활용하여 스냅샷 대신 기호 실행을 반복적으로 수행할 수 있다. 더불어 QSYM은 콘크리트 방식으로 외부 환경과 상호작용할 수 있다. 생성된 테스트 케이스에 대해서는 휴리스틱을 통해 불필요한 것을 제거한다. 이를 통해 기존 퍼징 기술보다 더욱 빠르게 취약점을 발견할 수 있다.

그러나 이전 연구보다 실행 속도가 개선되었지만, 여전히 느린 실행 시간을 지닌다. 또한 불필요한 제약 조건 해결에 시간을 낭비하여 많은 타임아웃이 발생한다. 기호 실행은 오염된 명령어가 실행될 때마다 기호 표현을 관리해야 하므로 높은 오버헤드가 발생한다.

2019년 Cho 등의 연구에서는 QSYM과 같은 하이브리드 퍼징에 사용되는 기존의 콘콜릭 실행의 문제점을 개선하기 위해 필드 레벨 정보를 사용하여 콘콜릭 실행을 최적화한 Intriguer를 제안하였다[8]. 기존의 하이브리드 퍼징의 콘콜릭 실행 엔진이 지니는 느린 기호 실행 속도, 불필요한 제약 조건 해결에 많은 시간 사용, 복잡하지 않은 제약 해결에 많은 자원 할당과 같은 문제를 개선하였다.

Intriguer는 동적 오염 분석을 수행하여 mov와 같은 데이터 전송 명령어를 제외한 명령어만 기록한 실행 추적 리스트를 생성한다(Execution Monitor), 생성된 추적 기록에서 넓은 범위의 입력 데이터를 사용하는 기록을 최소화하고(Trace Reduction), 입력 필드의 정보를 유추하

여 필드 전환 트리를 생성 후 제약조건을 해결한다(Field-level Constraint Solving). 이러한 과정을 통해 Intriguer는 보다 관련성이 높은 명령어에 대해 기호 에뮬레이션을 수행하고 복잡한 제약 조건에 대해서만 Solver를 호출하여 효율성을 향상 시켰다.

그러나 Intriguer는 간접 점프 또는 조건부 점프 명령어와 같은 제어 흐름 종속성을 고려하지 않는다. 동적 오염 분석 과정에서 데이터의 오염이 직접 전파되는 데이터 이동 명령어와 달리 데이터 오염이 간접적으로 전파되는 조건부 점프 명령어 또는 간접 점프 명령어가 있다. 입력 값에서 간접적으로 오염 전파 된 데이터를 재생산할 가능성이 없기 때문에 간접 전파의 추적 기록은 필드 전이 트리에 추가되지 않을 수 있다.

IV. 최신 퍼징 기술 성능 비교

LAVA (Large-scale Automated Vulnerability Addition)는 프로그램 소스코드에 자동으로 버그를 삽입하는 기술로써, 소스코드에 두 개 이상의 버그를 삽입한 데이터 셋을 LAVA-M이라고 지칭한다[9]. 표 2는 각 퍼징 기술들이 LAVA-M의 4가지 바이너리를 대상으로 5시간 퍼저를 돌렸을 때 발견한 버그 개수를 나타낸 것이다[4, 5, 6, 7, 8]. 여기서 괄호는 LAVA-M에 리스팅되어 있는 총 버그 개수 외에 추가로 찾은 버그 개수를 나타낸다.

	uniq	base64	md5sum	who
Total Bugs	28	44	57	2,136
VUzzer	27	17	1	50
Angora	29 (+1)	48 (+4)	57 (+0)	1,541 (+98)
QSYM	28	44	57	1,238
REDQUEEN	28 (+1)	44 (+4)	57 (+4)	2,134 (+328)
Intriguer	28	44	57	2,136

표 2 각 퍼저들의 LAVA-M 버그 발견 개수

Intriguer는 모든 버그를 발견하였으며, 나머지 퍼저들도 who를 제외하고는 거의 모든 버그를 발견한 것을 알 수 있다. 그러나 VUzzer의 경우 매직 바이트가 다수 존재하는 uniq 프로그램에서는 좋은 성능을 보였지만 나머지 프로그램에서의 성능은 좋지 않았다.

V. 결론

본 논문에서는 최신 퍼징 기술의 연구 동향에 대하여 분석을 해보았다. 최신 퍼징 기술인 VUzzer, REDQUEEN, Angora, QSYM, Intriguer를 하이브리드 퍼징을 기준으로 분류하여 그 구조를 알아보고, 한계점을 분석하였다. 우리의 향후 계획은 이러한 최신 퍼징 기술의 한계점을 극복하고, 나아가 보다 빠르고 정확한 퍼징 기술을 연구하는 것이다.

[참고문헌]

[1] Wichmann, B. A., et al. "Industrial perspective on static analysis." Software Engineering Journal 10.2 (1995): 69-75.

[2] AFL, http://lcamtuf.coredump.cx/afl/

[3] Pak, Brian S. "Hybrid fuzz testing: Discovering software bugs via fuzzing and symbolic execution." School of Computer Science Carnegie Mellon University (2012).

[4] Rawat, Sanjay, et al. "VUzzer: Application-aware Evolutionary Fuzzing." NDSS. Vol. 17. 2017.

[5] Aschermann, Cornelius, et al. "REDQUEEN: Fuzzing with Input-to-State Correspondence." NDSS. 2019.

[6] Chen, Peng, and Hao Chen. "Angora: Efficient fuzzing by principled search." 2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018.

[7] Yun, Insu, et al. "QSYM: A practical concolic execution engine tailored for Hybrid Fuzzing." 27th USENIX Security Symposium (USENIX Security 18). 2018.

[8] Cho, Mingi., et al. "Intriguer: Field-Level Constraint Solving for Hybrid Fuzzing", 2019 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2019.

[9] Dolan-Gavitt, Brendan, et al. "Lava: Large-scale automated vulnerability addition." 2016 IEEE Symposium on Security and Privacy (SP). IEEE, 2016.

음악 스트리밍 프로파일링 기반의 불법 음악 스트리밍 사이트 탐지 방안

정해선*, 곽 진**

*아주대학교 컴퓨터공학과 정보보호응용및보증연구실

**아주대학교 사이버보안학과

Illegal Music Streaming Site Detection based on Music Streaming Profiling

Hae-Seon Jeong*, Jin-Kwak**

*ISAA Lab., Department of Computer Engineering, Ajou University.

**Department of Cyber Security, Ajou University.

요 약

음악 저작물 복제는 간단하고 쉽게 할 수 있는 작업이기 때문에 불법 음악 저작물이 빈번하게 유포되고 있다. 이에 따라 내부 서버에 음악 복제물을 저장하고, 불법으로 음악 스트리밍을 제공하는 사이트가 운영되고 있다. 불법 음악 스트리밍 사이트들은 무료로 서비스를 제공하기 때문에 많은 사용자들이 사용하며, 해당 사이트들은 광고를 통해 수익을 얻기도 한다. 이에 따라 본 논문에서는 음악 저작물을 보호하기 위하여 합법적인 사이트와 불법 사이트를 비교 분석한 후 이에 기반한 불법 사이트 탐지 기법을 제시하였다.

I. 서론

재생한 음악 녹음하기, 유튜브에서 mp3 파일 추출하기 등 음악을 복제하는 것은 쉽게 할 수 있다. 실제로 2018년 12월 기준으로 음악은 두 번째로 많이 복제된 컨텐츠로 분류되었다[1]. 불법 음악 스트리밍 사이트는 쉽게 복제한 음악 저작물을 파일의 형태로 무료 유포하며 광고로 수익을 얻기도 한다. 이에 따라 본 논문에서는 음악 저작물을 보호하기 위해 불법 음악 스트리밍 사이트 탐지 기법을 제시하였다.

II. 관련 연구

2.1 저작권 침해 기준

저작권법에 따르면 저작권자, 저작권 신탁관리자의 이용 허가를 받지 않고 음악 저작물을 공유하는 경우 저작권법에서 규정한 전송권을 침해한다고 판단한다. 또한 권리자의 허가 없이 음악 파일을 통해 광고 수익을 보는 것 또한 저작권법을 위반하는 행위이다. 1분 이내로 편집하여도 명백한 전송권 침해 행위이며 면책 사유가 될 수 없다. “멜론” 등의 사이트에서 1분 미리듣기를 무료 제공하는 것은 이미 권리자의 허가를 받은 상태에서 서비스를 제공하는 것이므로 저작권 침해가 아니다. 이와 같은 저작권법 위반은 업로더의 책임이지만, 저작권 침해에 대해 온라인서비스제공자가 확인하여 방조한 경우에는 방조 책임이 있다.

본 연구는 문화체육관광부 및 한국저작권위원회의 2019년도 저작권기술개발사업의 연구결과로 수행되었음.

2.2 저작권 침해 사이트 사례

□ A 사이트

A 사이트는 서버 내에 저장되어 있는 음악 파일을 무료로 제공하고 있다. 무료 음악 스트리밍, 다운로드뿐만 아니라 웹페이지에 음악을 넣기 위한 태그도 제공하고 있다. 또한, 저작권자의 허가 없이 복제된 음악 파일도 제공하고 있다. 이는 저작권자의 이용 허가를 받지 않고 음악 저작물을 공유하는 것이기 때문에 저작권법 위반 행위에 해당한다.

□ B 사이트

B 사이트는 서버 내에 저장되어 있는 음악 파일에 대하여 스트리밍과 다운로드 기능을 제공하고 있다. B 사이트는 사용자에게 이와 같은 서비스를 제공하면서 광고를 통해 수익을 얻는다. 이는 저작권자의 이용 허가를 받지 않고 음악 저작물을 공유하며 저작물을 통해 수익을 얻는 저작권 위반 행위에 해당한다.

III. 음악 스트리밍 사이트 비교·분석

본 논문에서는 합법적인 음악 스트리밍 사이트와 불법 음악 스트리밍 사이트를 비교·분석함으로써 불법 음악 스트리밍 사이트를 탐지할 수 있는 특징을 도출하였다. 합법적인 음악 스트리밍 사이트로 이용량이 가장 높은 "멜론", "지니", "벅스"를 대상으로 분석하였으며[2], 불법 음악 스트리밍 사이트로 2.2절의 A 사이트와 B 사이트를 대상으로 분석하였다.

3.1 합법적인 음악 스트리밍 사이트 분석

합법적인 음악 스트리밍 사이트는 사용자가 음악 파일에 대해 사용 권한이 없는 경우에는 1분 미리 듣기만 제공한다. 이를 위해 서버는 음악 재생 요청 시 음악 정보. 사용자 정보 및 가사를 JSON 형태로 먼저 전송한 후에 음악 파일을 전송한다.

음악 재생 요청부터 음악 파일 전송까지의 과정은 3개 사이트 모두 대략 30개의 세션으로 구성되어 있다. "벅스"와 "지니" 서버는 암호화되어 있지 않은 평문 정보를 JSON 형태로 전송한다. 음악 정보, 사용자 정보, 사용자의 이용권 결제 정보, 가사 등을 평문의 JSON 형태로 전송한다. 그 후에 음악 파일을 전송한다. 하지만 "멜론"의 경우 JSON 내 key, value 값이 모두 암호화되어 있어 정보를 식별할 수 없었다. 하지만 요청 URL을 통해 멜론 서버에 요청한 정보를 추정할 수 있었다.

음악 재생 요청 시 서버의 구체적인 정보 전송 과정은 다음과 같다.

□ 벅스

Step 1. 음악 정보 및 사용자 정보 전송

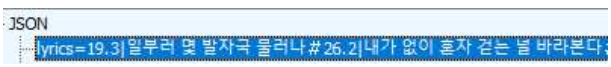
"https://music.bugs.co.kr/**player**/**track**/'노래 ID'?" 형태의 URL을 요청하면 서버는 [그림 1]과 같이 암호화되지 않은 평문의 JSON 형태로 음악 정보, 사용자 정보 및 사용자의 이용권 결제 정보를 전송한다. 해당 URL 내에서 음악 정보 요청에 해당하는 "player", "track" 키워드를 볼 수 있다.



[그림 1] 음악 정보 및 사용자 정보 일부

Step 2. 가사 전송

"https://music.bugs.co.kr/player/**lyrics**/T/'음악 ID'" 형태의 URL을 요청하면 서버는 [그림 2]와 같이 사용자가 요청한 음악의 가사를 암호화되지 않은 평문의 JSON 형태로 전송한다. 해당 URL 내에서 가사에 해당하는 "lyrics" 키워드를 볼 수 있다.



[그림 2] 서버가 전송한 음악 가사 일부

Step 3. 음악 파일 전송

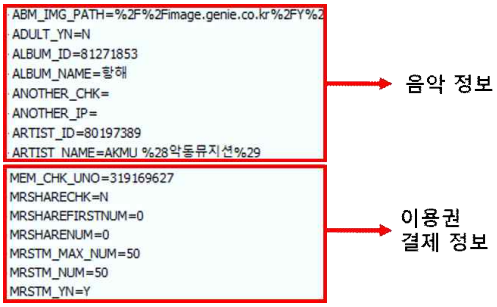
"https://w-aod.bugs.co.kr/raout/aac128/'음악

ID'.m4a?mediatype=TRACK&**mediaid**='음악 ID'&" 형태의 URL을 요청하면 서버는 음악 파일을 전송한다. 해당 URL 내에서 요청한 음악을 명시하기 위해 사용되는 "mediaid" 키워드를 볼 수 있다.

□ 지니

Step 1. 음악 정보 및 사용자 정보 전송

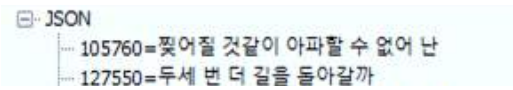
"https://www.genie.co.kr/**player**/play**Stm**Info.json음악 ID'?" 형태의 URL을 요청하면 서버는 [그림 3]과 같이 암호화되지 않은 평문의 JSON 형태로 음악 정보, 사용자의 이용권 결제 정보를 전송한다. 해당 URL 내에서 음악에 해당하는 "player"와 스트리밍의 약자 "stm" 키워드를 볼 수 있다.



[그림 3] 음악 정보 및 사용자 정보 일부

Step 2. 가사 전송

"https://dn.genie.co.kr/app/purchase/get_msl.asp?path=a&**songid**='음악 ID'&...." 형태의 URL을 요청하면 서버는 [그림 4]와 같이 사용자가 요청한 음악의 가사를 암호화되지 않은 평문의 JSON 형태로 전송한다. 해당 URL 내에서 가사에 해당하는 "lyrics" 키워드를 볼 수 있다. 해당 URL 내에서 요청한 음악을 명시하는 "songid" 키워드를 볼 수 있다.



[그림 4] 서버가 전송한 음악 가사 일부

Step 3. 음악 파일 전송

"https://web-hls.genie.co.kr/web/_definst_/ND_O/SONG/..../mp3: '음악 파일 명'/playlist.m3u8?token=...." 형태의 URL을 요청하면 서버는 음악 파일을 전송한다.

□ 멜론

step 1. 사용자 인증 정보 전송

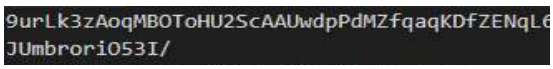
"http://appsvr.melon.com/cds/user/player/user_authInfoUser5.json"형태의 URL을 요청하면 서버는 [그림 5]와 같이 암호화된 JSON 데이터를 전송한다. 해당 JSON 데이터는 식별할 수 없지만 URL 내에서 사용자 및 인증과 관련된 "user", "player", "author" 키워드를 볼 수 있다.



[그림 5] user_authInfoUser5.json의 일부 내용

Step 2. 음악 정보 전송

"http://appsvr.melon.com/cds/delivery/**strea**ming/**player**/streaming_getTgtInfo.json" 형태의 URL을 요청하면 서버는 [그림 6]과 같이 암호화된 JSON 데이터를 전송한다. 해당 JSON 데이터는 식별할 수 없지만 URL 내에서 음악과 관련된 "player", "stream" 키워드를 볼 수 있다.



[그림 6] streaming_getTgtInfo.json 내용 일부

Step 3. 가사 전송

"http://appsvr.melon.com/cds/delivery/streaming/player/streaming_getstn**lyrics**info2.json?_cid=..." 형태의 URL을 요청하면 서버는 암호화된 JSON 데이터를 전송한다. 해당 JSON 데이터는 식별할 수 없지만 URL 내에서 가사와 관련된 "lyrics" 키워드를 볼 수 있다.



[그림 7] streaming_getstnlyricsinfo2.json 내용 일부

Step 4. 음악 파일 전송

"http://sts.melon.com/p3/.../'음악 파일명'..." 형태의 URL을 요청하면 서버는 음악 파일을 전송한다.

3.2 불법 음악 스트리밍 사이트 분석

불법 음악 스트리밍 사이트의 경우 별도의 사용자 인증 없이 자체 서버 내에 있는 음악 파일을 바로 제공한다. 음악 재생 요청 시 음악 정보, 사용자 정보, 가사를 전송하지 않고 바로

음악 파일을 전송하기 때문에 전체 세션 수는 2, 3개로 매우 적다. [표 1]의 URL을 요청함으로써 단순히 서버 내부의 음악 파일을 요청한다.

[표 1] 불법 음악 스트리밍 사이트의 음악 파일 요청 URL

사이트 명	요청 URL
A 사이트	https://"A 사이트 도메인"/.../"음악 파일 명"
B 사이트	https://"B 사이트 도메인"/.../"음악 파일 명"

IV. 불법 음악 스트리밍 사이트 탐지

3장에서 합법적인 음악 스트리밍 사이트와 불법 음악 스트리밍 사이트를 분석한 결과 도출한 불법 음악 스트리밍 사이트의 특징을 기반으로 제안하고자 하는 불법 음악 스트리밍 사이트 탐지 기법 동작 과정은 다음과 같다.

Step 1. 합법적인 음악 스트리밍 사이트 목록과 비교

불법 사이트를 탐지하기 위해 가장 먼저 합법 사이트 목록과 비교한다. 일치하지 않는 경우 다음 단계들을 통해 불법 사이트인지 탐지한다.

Step 2. 통신 세션 수 확인

음악 재생 시 불법 음악 스트리밍 사이트는 음악 정보, 사용자 정보 및 가사 전송 없이 음악 파일만 전송하기 때문에 세션 수가 5개 미만으로 적게 나타난다. 따라서 적은 세션 수를 통해 음악을 재생한다면 불법 음악 스트리밍 사이트로 탐지할 수 있다.

Step 3. 요청 URL 키워드 DB 비교

합법적인 음악 스트리밍 사이트의 경우 음악 파일 전송에 앞서 음악 정보, 사용자 정보 및 가사를 전송한다. 음악 정보 요청 URL에는 음악을 뜻하는 "track", "player"와 스트리밍에 해당하는 "stream", "stm"이 포함되어 있었다. 요청하는 음악을 명시하여 요청하는 경우에는 음악 ID를 "mediaid", "songid" 등을 사용하여 URL 내에 명시하였다. 또한, 3개의 합법적인 음악 스트리밍 사이트 모두 가사 요청 URL에는 "lyrics"가 포함되어 있었다.

이에 따라 음악을 의미하는 키워드, 음악 ID를 의미하는 키워드, 가사를 의미하는 키워드 항목으로 구성된 DB 기반으로 불법 음악 스트리밍 사이트를 탐지하고자 한다. 해당 DB 내에는 3.1절에서 도출한 키워드 외에도 DB 키워드 항목에 대응 가능한 키워드가 포함되어 있다. 음악 재생 시 요청 URL을 [표 2]의 키워드 DB와 매칭하여 해당 사이트가 합법적인 스트리밍 사이트처럼 음악 파일 전송에 앞서 음악 정보, 사용자 정보 및 가사를 전송하는지 확인한다. [표 2]의 키워드 DB와 매칭되지 않으면 불법 음악 스트리밍 사이트로 탐지한다.

[표 2] 키워드 DB

키워드 항목	키워드
음악	track, media, player, stream, stm, song
음악 ID	trackid(track_id) mediaid(media_id) playerid(player_id) streamid(stream_id) stmid(stm_id) songid(song_id)
가사	lyrics

V.결론

본 논문에서는 음악 저작물을 보호하기 위하여 합법 음악 스트리밍 사이트와 불법 음악 스트리밍 사이트를 비교·분석하였고, 이를 통해 도출한 불법 음악 스트리밍 사이트의 특징 기반 탐지 기법을 제시하였다. 만약 불법 사이트가 의도적으로 합법 사이트와 비슷한 세션 수, 비슷한 키워드를 가지는 통신을 한다면 해당 방법은 사용할 수 없기 때문에 이를 보완하는 방법과 함께 사용한다면 더 높은 탐지율을 보일 수 있을 것이다.

[참고문헌]

[1] 문화체육관광부(저작권정책과), "저작권법", October, 2018.

[2] 컨텐츠진흥원, "국내 음악이용자 실태 조사", February, 2019.

공공 인프라 영상보안을 위한 보안 프레임워크 연구

유진용*, 김영갑*†

*세종대학교 정보보호학과

A Study on Security Framework for Video Security of Public Infrastructure

Jin-Yong Yu*, Young-Gab Kim*†

*Division of information security, Sejong University.

요 약

최근 공공 인프라 환경에서 보안 카메라의 증가율을 크게 늘어나고 있다. 각 종 범죄·사건으로부터 사회질서를 확립하기 위하여 이러한 현상은 자연스럽게 받아들여지고 있다. 그러나 방대한 양의 실시간 영상데이터를 처리해야하는 공공 인프라 환경의 특성으로 인하여 영상을 보호하기 위한 적절한 보안기술 적용이 어려운 상황이다. 따라서 본 논문에서는 공공 인프라 환경에 알맞은 보안 메커니즘이 갖춰질 수 있도록 영상보안 프레임워크를 제안하고, 이를 위한 최신 영상보안기술들에 대한 체계적인 분류와 분석을 수행한다.

I. 서론

오늘날 일상생활 속 보안 카메라 사용이 나날이 급증하고 있다. 수많은 범죄들이 만연함에 따라 공공사회의 질서를 확립하기 위하여 보안 카메라의 증가는 자연스러운 것이라 여겨진다. 그러나 공공 인프라 환경에서 운용하고 있는 보안 카메라의 증가 추세를 고려할 때, 실시간으로 저장되는 방대한 양의 영상데이터에 대한 보호는 점점 어려워지고 있다. 영상감시(video surveillance)측면에서도 방대한 양의 영상데이터는 모니터링에 대한 비효율성을 야기한다. 이는 특정구간에 대한 영상데이터를 열람하기 위해서는 시간, 장소, 카메라의 위치 등의 정확한 정보가 필요하며 해당정보가 없을 경우에는 모든 영상에 대한 조사가 필요하기 때문에 많은 시간이 소비될 것임을 의미한다. 또한, 개인정보보호의 중요성에 대한 사회적 인식이 높아짐에 따라 개인을 식별할 수 있는 민감한 정보가 담긴 영상에 대한 보안은 선택이 아닌 필수가 되었다. 특히, CCTV, 열 감지 카메라, X-ray영상 등의 민감한 개인정보를 수집할 수 있는 공공인프라 영상에 대한 접근제어(access control), 암호화(encryption), 비식별화 등의 보안기술의 적용은 법·제도적으로도 규정하려 하는 노력을 지속적으로 하고 있다 [1]. 더욱이 많은 연구기관을 비롯한 전문가들이 이러한 문제를 인식하고 해결하기 위해 광범위한 연구를 수행하고 있다.

그러나 각 종 범죄들이 지능화 되면서 더욱 높은 수준의 보안과 효율적인 영상시스템을 요구하고 있다. 현재, 이와 관련된 표준으로는 ITU-T (International Telecommunication Union Telecommunication Standardization Sector)에서 제정한 H.530-H.539 및 H.540-H.549 등의 영상 멀티미디어에 대한 보안표준이 있다. 그러나 해당 표준은 실시간

† 교신저자

(real-time) 영상들에 대한 보안이 전혀 고려되지 않았으며 공공 인프라 영상과 같이 방대한 양의 영상데이터를 효율적으로 보호할 수 있는 연구들이 여전히 부족한 상황이다 [2].

따라서 본 논문에서는 공공 인프라 환경에서 영상을 보호하기 위한 보안 메커니즘으로 영상 보안 프레임워크를 제안한다. 또한, 공공 인프라 영상에 대한 특성을 고려하여 다양한 영상 보안기술들에 대한 체계적인 분류와 분석을 수행한다.

제 2장에서는 영상보안 관련 기술들에 대한 체계적인 분류와 분석을 수행한다. 제 3장에서는 안전한 공공 인프라 환경을 조성하기 위해 갖추어야 할 필수요소를 포괄하는 영상보안 프레임워크를 제안한다. 최종적으로 제 4장에서는 결론과 향후연구를 기술한다.

II. 영상보안 관련 기술 분석

영상시스템을 보호하기 위한 보안기술들에 대한 연구는 일찍이 수행되어 왔다. 그러나 광범위하게 연구되어진 영상보안기술들에 대한 체계적인 분류와 분석을 수행한 연구는 거의 없으며 기술발전 속도와 지능화된 범죄들이 나날이 증가하는 것을 고려할 때 이에 대처할 수 있는 최신 보안기술들에 대한 동향 분석은 필수적이다. 따라서 본 장에서는 영상시스템에 대한 최신 보안기술들의 동향을 살피고 이를 체계적으로 분류하고 분석한다.

2.1 암호화(encryption)

영상 멀티미디어에 대한 암호화 연구는 일찍이 다양한 측면에서 수행되어 왔으며 현재에도 지속적으로 연구되고 있는 분야이다 [3]. 그러나 실시간 영상에 대한 암호화 및 표준에 대한 기술과 연구는 많지 않다. 특히, 공공 인프라 환경에서는 많은 영상이 실시간으로 처리되기 때문에 실시간으로 영상 데이터를 암호화할 수 있는 기술들이 요구되고 있다. 본 연구에서는 영상 암호화에 대한 최신연구들과 관련문헌들을 분석하여 영상 암호화기술을 암호화 시점, 영상 암호화 유형, 암호화 알고리즘, 암호화 평가로 분류 하였다. 영상이 암호화되는 시점은 인코딩 전과 후 그리고 인코딩과 동시에 이루어 질 수 있다. 그러나 암호화 시점에 따라 적용 가능한 암호화 유형과 그렇지 못한 유형이 존재한다. 일부분만 선택적으로 암호화 하는 유형일 경우에는 인코딩이 진행 중일 때만 가능하며, 전체를 암호화하는 경우에는 인코딩 전, 후에만 암호화가 적용 가능하다. ROI (Region of Interest)기반으로 영상 암호화를 수행할 경우 인코딩 전에만 적용 가능하다. 암호화 알고리즘은 DES와 AES 같은 전통적(traditional)인 방식과 보다 진화된(advance) 방식으로 나뉠 수 있으며, 진화된 방식의 암호화 알고리즘 같은 경우는 실시간 암호화, 방대한 양의 데이터 암호화 등과 같이 특정 시나오리에서 더욱 효율적으로 적용될 수 있다. 영상데이터가 암호화된 이 후, 이를 평가할 수 있는 요소로는 엔트로피 분석, 픽셀 변화율, 픽셀 상관 계수등이 있다.

2.2 접근제어(access control)

일반적으로 접근제어는 식별과 인증, 인가로 분류되며 제안된 프레임워크에 대한 구체적인 접근제어 기술을 적용하기 위해 정의되어야 할 선행조건이 필요하다. 우선, 수집된 영상에 대하여 접근할 수 있는 사용자를 명확히 정의할 필요가 있다. 이는 모니터링 인력과 같은 정당한 이유 또는 권한을 가진 사용자를 식별하기 위함이다 [4]. 또한, 사용자마다의 접근할 수 있는 영상 데이터 영역이 정의함으로써 앞서 언급한 모니터링 인력과 같이 영상데이터에 쉽게 접근할 수 있는 사용자들의 부정적인 접근을 방지할 필요가 있다. 공공 인프라에서 수집되는 영상들은 안면, 홍채등과 같은 개인정보 즉, 개인식별이 가능한 정보들을 다수 포함하고 있기 때문에 각 영상에 대하여 올바른 접근권한이 있는 사용자만이 해당 영상을 열람할 수 있도록 조치를 취해야 한다.

본 연구에서는 영상시스템에 대한 접근제어 기술을 분석하기 위해 최신 연구, 문헌을 기반으로 식별과 인증, 인가, 접근제어 정책으로 분류하였다. 식별 및 인증체계는 식별 및 인증 수단, 사용자를 비롯한 메시지와 디바이스 등과 같은 인증할 요소, 단일인증과 복합인증, one-way 방식과 two-way방식, 인증 유닛, 이

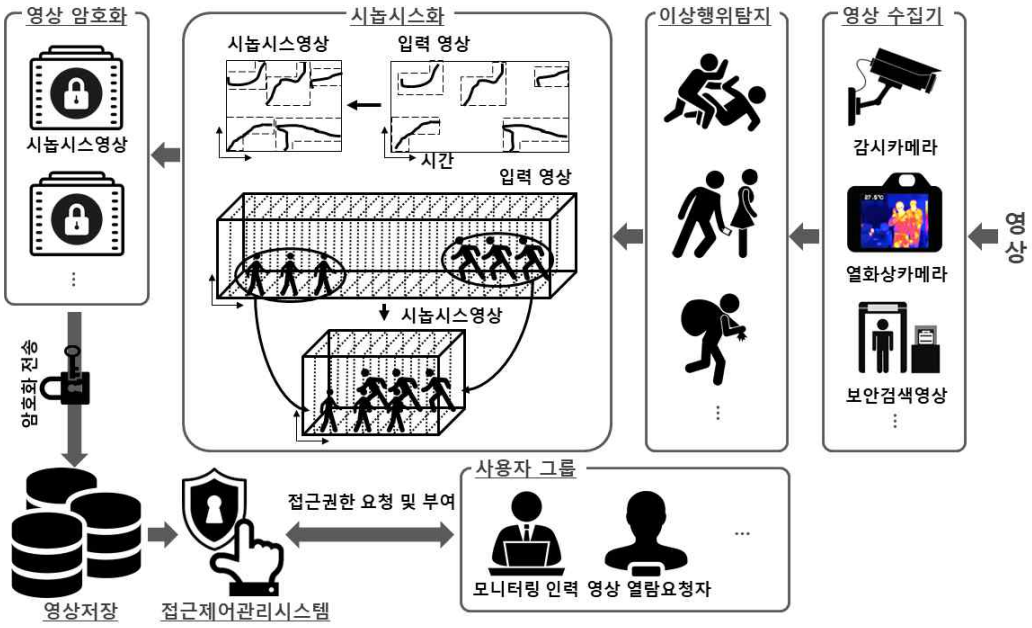


그림 1 공공 인프라 환경에서의 효율적인 영상보안을 위한 프레임워크

종 도메인에 따른 인증 및 식별로 분류될 수 있다. 인가체계는 임의적 접근제어(Discretionary Access Control), 강제적 접근제어(Mandatory Access Control) 등과 같은 다양한 접근제어 모델들과 각 모델들이 기반하고 있는 참조 아키텍처, 정책평가전략, 앞서 인증 및 식별체계에서 언급한 도메인으로 분류할 수 있다. 정책체계는 정책을 작성하는 언어와 시나리오에 따라 정책을 추출해주는 Derivation/Extraction, 이종간의 체계에서 정책을 협의하는 과정인 협의(negotiation)으로 분류된다.

III. 시놉시스 기반 영상보안 프레임워크

현재, 공공 인프라에서 보안 카메라의 수가 기하급수적으로 늘어나고 있는 것에 반해 모니터링 인력에 대한 문제는 크게 이슈화 되고 있다. 미국 법무성에서는 모니터링 인력의 주의력 지속시간이 시간이 지남에 따라 급격히 저하되며 25분이 지나면 주의력이 10%이하로 내려간다는 연구 사례가 발표된 바 있다. 수집되는 영상데이터의 양이 비대해진 것을 감안했을 때 정상적인 모니터링에 대한 효과를 기대하기 어려우며 영상데이터에 대한 관리에도 보안적 문제가 발생할 수 있음을 암시한다. 따라서 본 논문에서는 이를 보완하기 위하여 그림 1과 같은 영상 시놉시스(video synopsis)기반 영상보안 프레임워크를 제안한다.

영상 시놉시스란 그림 1에서 볼 수 있듯이 활동 기반 영상압축기법으로 짧은 시간동안 가능한 많은 객체의 활동을 동시에 보이는 것을 목표로 하는 영상요약기술이다 [5]. 영상 시놉시스의 기술은 객체간의 충돌을 최소화 시키고 짧은 시간동안 더 많은 객체활동을 표현할 수 있도록 조정하는 과정인 최적화, 카메라의 위상에 따른 배치, 입력 데이터 도메인, 행위 클러스터링을 기준으로 분류될 수 있다. 방대한 양의 영상 데이터를 다뤄야 하는 공공 인프라 환경에서의 영상 시놉시스 기술의 적용은 모니터링 인력문제에 대한 부담을 줄일 수 있으며 보안측면에서도 보호되어야 할 데이터 양이 대폭 감소함으로 이점이 있다.

제안된 영상보안 프레임워크는 영상 시놉시스를 기반으로 공공 인프라 환경에서의 영상 시스템이 갖춰야할 보안요소를 내포하고 있다. 공공 인프라 환경이라는 방대한 양의 데이터를 실시간으로 처리하는 도메인의 특수성을 고려할 때 모든 영상에 대하여 시놉시스화 또는 보안처리를 하는 것은 시스템으로 하여금 부하를 가중시키고 시간지연(time delay)가 발생시킨다.

따라서 공공 인프라 환경에서는 사건·사고와

같은 핵심적인(critical) 또는 이상적인(abnormal)장면을 탐지 또는 색출하여 시놉시스 또는 암호화하는 것이 효율적이다. 그러나 객체가 취할 수 있는 수많은 경우의 수에서 이상행위라고 판단할 수 있는 기준이 매우 모호하기 때문에 실시간으로 이상행위를 탐지하는 것은 쉽지 않다 [6]. 이를 해결하기 위해 AI를 이용하여 이상행위를 학습시키고 높은 정확도(accuracy)를 도출하기 위한 방안이 필요하다. 시놉시스와 이상행위탐지는 방대한 양의 데이터가 모이는 공공 인프라 환경에서 관리되어야 할 데이터를 대폭적으로 줄여줌으로써 보안측면으로 기여될 수 있다.

시놉시스화 이후에 기밀성 보호를 위한 암호화 및 보안프로토콜과 안전한 채널을 통한 전송이 이루어져야 한다. 암호화 등을 통한 기밀성을 보호하는 것도 중요하지만 최종적으로 영상데이터에 대한 구체적인 접근제어 방안을 구체화 하는 것이 보안핵심이다. 현재, 공공 인프라 영상에 대한 접근제어 방안은 구체화되어 있지 않으며 대다수의 모니터링 인력들이 모든 영상에 접근할 수 있다고 해도 과언이 아닐 것이다. 따라서 영상데이터에 따른 사용자의 접근제어 방안이 구체화 되어야 하며 영상 사용자들에게 적절한 권한을 부여할 수 있는 체계가 필요하다.

IV. 결론 및 향후연구

본 논문에서는 영상보안과 관련된 연구들이 광범위하게 수행되고 있음에도 불구하고 공공 인프라 환경에서 수집되는 영상들에 대한 적절한 보안이 이루어지지 않고 있다는 것을 지적하였다. 이에 따라 공공 인프라에서의 영상보안을 위한 메커니즘으로 시놉시스 기반 영상보안 프레임워크를 제안하였다. 또한, 영상보안기술(암호화, 접근제어)에 대한 최신연구와 문헌 등을 통하여 체계적인 분류와 분석을 수행하여 최신보안기술에 대한 동향을 살펴보았다.

그러나 제안된 프레임워크는 하나의 시스템에서 시놉시스화, 이상행위탐지를 비롯한 많은 동작과정을 수행하기엔 성능적인 측면에 한계가 존재한다. 따라서 향후에는 제안된 프레임워크를 기반으로 fog와 edge-cloud 컴퓨팅과 같이 단말(endpoint)에서 데이터처리를 수행할 수 있는 분산된 시스템 체계에 이를 적용할 방안에 대한 연구가 필요하다.

[Acknowledgement]

이 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00231, 공공인프라 안전을 위한 인공지능 기반 영상보안 기술 및 시스템 개발)

[참고문헌]

[1] Ministry of the Interior of Republic korea, ‘Bill of Video Security for Privacy’, December, 2016.

[2] Boyer M., Veigl, S., Privacy preserving video surveillance infrastructure with particular regard to modular video analytics. 2015.

[3] Saleh, Mohammed A., et al. "An analysis and comparison for popular video encryption algorithms." 2015 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE). IEEE, 2015.

[4] Billau, Ronald L., et al. "Automated authorization to access surveillance video based on pre-specified events." U.S. Patent No. 9,041,812. 26 May 2015.

[5] Kemal Batuhan Baskurt, Refik Samet, Video synopsis: A survey, Computer Vision and Image Understanding, Pages 26-38, Volume 181, April, 2019.

[6] MABROUK, Amira Ben; ZAGROUBA, Ezzeddine. Abnormal behavior recognition for intelligent video surveillance systems: A review. Expert Systems with Applications, 2018

클라우드 공유 기능의 저작권 침해 방지를 위한 연구

박소희*, 곽 진**

*아주대학교 컴퓨터공학과 정보보호응용및보증연구실

**아주대학교 사이버보안학과

The Prevention of Copyright Infringement in Cloud Sharing Function

So-Hee Park*, Jin Kwak**

*ISAA Lab., Department of Computer Engineering, Ajou University.

**Department of Cyber Security, Ajou University.

요 약

클라우드는 사용자의 요구 및 수요 변화에 따라 데이터를 시간과 장소에 제약받지 않으며 신축적으로 이용할 수 있도록 하는 정보처리체계이다. 매년 클라우드 데이터센터 트래픽은 증가하고 있으며 클라우드 내에 주요정보를 저장하는 개인 사용자 및 기업 또한 늘고 있는 추세이다. 또한 공유된 데이터를 언제 어디서나 다운로드할 수 있는 환경이 형성되면서 사용자의 편리성이 증대되었다. 그러나 클라우드 내에서 발생하는 공유된 데이터의 저작권 침해 행위가 증가하는 반면 이에 대한 대응은 미흡하다. 따라서 클라우드 환경에 적용할 수 있는 보안 서비스에 대한 연구가 필요하다. 본 논문에서는 데이터 유출 방지 기반 모듈을 구성하여 클라우드 환경에서 저작권 침해 행위를 대응하기 위한 아키텍처를 연구한다.

I. 서론

클라우드(Cloud)란 집적·공유된 정보통신기기, 정보통신설비, 소프트웨어 등 정보통신자원을 사용자의 요구나 수요 변화에 따라 정보통신망을 통하여 신축적으로 이용할 수 있도록 하는 정보처리체계이다[1].

클라우드는 비용 절감, 업무 효율성 및 생산성 향상 효과 등을 가진다. 클라우드 데이터센터 트래픽이 2016년 기준 연간 6.0ZB에서 2021년에는 19.5ZB까지 연평균 27%의 성장률을 보일 것이며, 2021년 전체 데이터센터 트래픽의 95%를 차지할 것으로 전망되고 있다[2]. 또한 최근 클라우드 내에 주요정보를 저장하는 개인 사용자·기업이 늘고 있는 추세이다.

클라우드 서비스가 증가하면서 공유되는 데이터를 언제 어디서나 다운로드할 수 있는 환경이 형성되었다. 클라우드 서비스는 사용자의 편리성을 추구하지만 이를 이용한 저작권 침해 행위 또한 증가하고 있다. 그러나 클라우드 환경에서의 저작권 침해에 대한 인식 및 대응 방안은 부족한 실정이다. 따라서 클라우드 환경에서 활용 가능한 보안 서비스 제공을 위해 다양한 연구가 필요하다.

본 논문에서는 데이터 유출 방지 (DLP, Data Loss Prevention) 기반 모듈을 구성하여 클라우드 환경에서 발생 가능한 저작권 침해 행위에 대응하기 위한 아키텍처를 제시하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서 클라우드 저작권 및 데이터 유출 방지에 대하여 연구하고, 3장에서 저작권 침해 행위에 대한 시나리오를 분석한다. 4장에서 보안 서비스를 적용한 클라우드 아키텍처에 대하여

본 연구는 문화체육관광부 및 한국저작권위원회의 2019년도 저작권기술개발사업의 연구결과로 수행되었음.

분석하고 5장에서 결론을 내린다.

II. 클라우드에 의한 저작권 침해

2.1 클라우드 저작권 침해 유형

저작권법은 저작자의 권리와 이에 인접하는 권리를 보호하고 저작물의 공정한 이용을 도모함으로써 문화 및 관련 산업의 향상발전을 목적으로 한다[3]. 저작권법의 목적이 문화 및 관련 산업의 향상발전에 있는 만큼 저작권을 지적재산으로 인정하기 때문에 저작자가 아닌 자가 저작물로써 영리를 목적으로 할 때 저작자의 권리를 해한 것으로 간주하며 해당하는 행위를 제재한다.

저작권법 제27조에 따르면 저작자가 아닌 자가 공표된 저작물을 영리를 목적으로 하거나, 가정 및 이에 준하는 한정된 범위를 넘어서 불특정 다수와 복제한 저작물을 이용하는 경우 이는 사적이용을 위한 복제가 아니므로 저작권 침해에 해당한다[3]. 따라서 정당한 값을 지불하지 않은 자에게 저작물을 이용할 수 있는 권리를 부여하는 것은 클라우드 서비스에 의한 저작권 침해 행위이다. 클라우드 서비스의 아이디 및 패스워드를 불특정 다수인과 공유하는 경우, 저작물이 업로드된 폴더에 불특정 다수인을 멤버로 초대하는 경우, 저작물을 다운받을 수 있는 링크를 생성하여 저작물을 타인에게 배포하는 경우 등으로 저작권 침해 유형을 나눌 수 있다.

따라서 사용자가 저작물을 클라우드에 업로드 할 때, 다른 사용자들과 저작물 공유를 위한 배포 경로의 모니터링과 사적 복제의 가능 여부를 확인하여 저작권 침해 행위를 판별할 수 있다. 본 논문에서는 배포 경로의 감시와 사적 복제의 가능 여부 확인을 위해 클라우드 환경 보안 서비스 모듈에 DLP 시스템을 사용한다.

2.2 데이터 유출 방지 (Data Loss Prevention)

데이터 유출 방지는 정보 유출을 막기 위해 콘텐츠의 이동을 모니터링하며 이동하는 데이터에 기밀정보 및 개인정보 등 주요정보가 포함되어 있을 경우 이를 차단하여 데이터를 보호한다. 구성원과 프로세스 및 기술의 결합을 통해 주요정보가 외부로 유출되는 것을 방지한다. 사용되는 데이터를 위한 데이터 유출 방지는 사용자 인증과 자원에 대한 액세스를 제어하고 응용 프로그램이나 엔드포인트에 의해 처리되는 데이터를 수집한다. 이동되는 데이터를 위한 데이터 유출 방지는 주요정보가 네트워크를 통해 전송되는 경우 외부로의 주요정보 라우팅 여부와 저장소 영역의 안전 여부에 대해 확인한다. 유휴 데이터를 위한 데이터 유출 방지는 권한을 가진 사용자만 데이터에 액세스할 수 있도록 하고 데이터가 유출될 경우 액세스를 추적하는 기능 등으로 클라우드를 포함한 다양한 저장 매체의 데이터를 보호한다[4].

III. 저작권 침해 시나리오

[그림 1]은 클라우드 환경에서 사용자가 저작물 공유를 통해 저작권을 침해하는 과정에 대하여 나타낸 것이다. 과정은 다음과 같다.



[그림 1] 저작권 침해 시나리오

Step 1. 클라우드 사용자의 저작물 업로드

사용자가 클라우드 서버에 저작물을 업로드한다. 사용자 본인이 업로드한 저작물의 저작자일 경우 특정·불특정 다수인과 클라우드 서비스를 이용하여 공유하는 것은 저작권법상 문제가 되지 않는다. 그러나 저작자가 아닐 경우에 저작권 침해의 우려가 있다.

Step 2. 제3자 대상의 저작물 배포

사용자가 클라우드의 공유 기능을 통하여 불특정 다수인과 저작물을 공유하며 클라우드

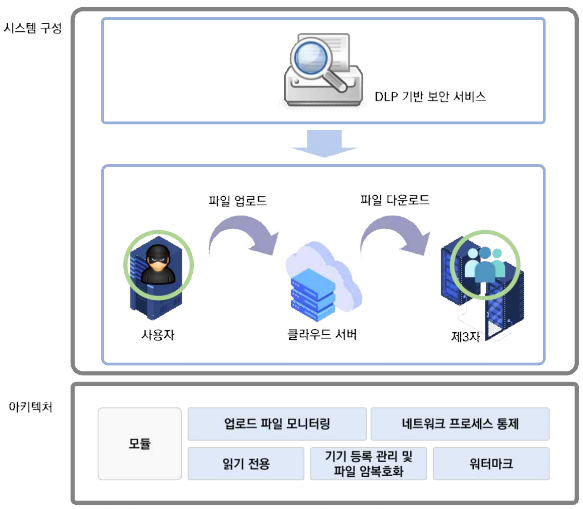
환경에서 저작권 침해는 대부분 이와 같은 형태이다. 따라서 클라우드에 저작물을 업로드하는 행위는 저작권 침해가 일어나는 원인이다. 그러나 클라우드의 정의에 따르면 언제 어디서나 필요한 파일을 다운로드 할 수 있는 서비스로써 해당 행위를 제한하는 것은 클라우드의 정의와 맞지 않는다.

Step 3. 저작물 공유로 인한 저작권 침해

제3자가 정당한 대가를 지불하지 않고 저작물을 다운로드 받음으로써 저작권을 침해한다.

IV. 제안 사항

클라우드 환경에서 일어나는 저작권 침해를 방지하기 위해서 저작물의 업로드를 제한하는 것은 클라우드의 정의와 맞지 않기 때문에 불가능하다. 클라우드에 동시 접속할 수 있는 기기의 수를 제한하거나 사용자가 기기를 등록하여 클라우드에 대한 이용을 제한하는 등의 방법이 사용되고 있다. 그러나 논문에서 다루는 클라우드 서비스의 분류상 저장 공간 제공이라는 특성 때문에 데이터 복제 및 다운로드 등의 서비스 이용에 대한 저작권 침해를 막을 수 없다. 따라서 저작물의 통신상 이동에 중점을 두고 데이터 유출 방지 솔루션을 사용하여 클라우드 보안 서비스를 제공할 때 필요한 기능을 제안한다.



[그림 2] 제안하는 클라우드의 공유 기능에 의한 저작권 침해 행위 대응 아키텍처

본 절에서는 [그림 2]의 구성으로 데이터 유출 방지 기반의 클라우드 저작권 침해를 방지하는 보안 서비스 아키텍처를 제안한다. 보안 서비스 아키텍처는 다음과 같은 기능을 가진다.

4.1 업로드 파일 모니터링

클라우드에 업로드되는 모든 파일에 대한 암호화된 데이터를 저장한다. 또한 파일의 헤더값을 비롯한 위변조 발생을 모니터링하며 위변조를 발견할 경우 파일 업로드에 제재를 둔다.

4.2 네트워크 프로세스 통제

화이트 리스트 (White List) 방식으로 사전에 프로세스를 등록하여 허용된 프로세스에 대한 정책을 설정한다. 등록된 프로세스는 사용자가 주로 접속하는 IP주소 정보로 프로세스에 따라 파일의 업로드와 다운로드에 대한 허용 및 차단 설정을 관리하며 모든 업로드된 파일을 모니터링한다. 해당 로그를 수집하여 프로세스가 등록된 화이트 리스트에 항목을 추가하거나 삭제한다.

4.3 읽기 전용

클라우드 환경의 저작물이 업로드된 폴더를 이용하는 공유 기능에 대하여 제한을 설정한다. 읽기 전용 서비스로써 업로드된 파일의 다운로드, 편집 및 저장이 불가하며 웹을 이용한 파일의 열람이 가능하다.

4.4 기기 등록 관리 및 파일 암복호화

클라우드를 사용하는 기기를 미리 등록한다. 또한 클라우드에 저장되는 모든 파일을 자동 암호화한다. 등록된 기기를 통한 서비스 이용만을 가능하게 하며 등록된 기기가 아닐 시 암호화된 파일의 열람이 불가하다.

4.5 워터마크

클라우드에 업로드된 파일을 다운받을 경우 워터마크를 함께 출력한다. 로고 이미지, 저작자 정보, 저작물에 대한 명시 등의 정보를 표시한다.

[표 1] 제안하는 클라우드의 공유 기능에 의한 저작권 침해 행위 대응 아키텍처의 기능

모듈	설명
업로드 파일 모니터링	- 업로드 파일 모니터링 및 위변조 파일 업로드 제한 - 저작권 침해 방지에 대한 우회 탐지를 통해 저작권을 보호함
네트워크 프로세스 통제	- IP주소 정보 수집 및 프로세스 정책 설정 - 저작물 공유 정책의 강화를 통해 저작권을 보호함
읽기 전용	- 공유 기능의 파일 이용 제한 - 저작물의 재배포 방지를 통해 저작권을 보호함
기기 등록 관리 및 파일 암복호화	- 사용자 기기 등록 및 파일 암복호화 - 저작물 사용 정책의 강화를 통해 저작권을 보호함
워터마크	- 파일 대상 워터마크 출력 - 저작권에 대한 확고한 명시를 통해 저작권을 보호함

V. 결론

최근 클라우드 서비스 기술이 지속적으로 발전함에 따라 클라우드 내에 주요정보를 저장하는 개인 사용자·기업이 늘고 있는 추세이다. 클라우드 서비스 사용이 증가하면서 클라우드 환경에서 발생하는 보안 위협에 대한 연구가 지속되고 있다. 본 논문에서는 클라우드 서비스를 통한 저작권 침해 유형에 대해 분석하였다. 또한 저작권 침해 행위의 방지를 위하여 데이터 유출 방지 기반 모듈로 구성한 클라우드 환경의 보안 서비스 아키텍처를 제안하였다. 클라우드 환경에서 저작물의 배포 경로를 모니터링하고 사적 복제의 가능 여부를 확인하기 위해 데이터 유출 방지 기술을 사용하였으며 제안한 모델을 보안 서비스로 제공할 경우 클라우드 보안 서비스에 필요한 복합적인 요소를 고려하여 저작권에 대한 보호가 이루어질 것이다.

향후, 제안한 모델의 연구 및 구현을 통해 기능을 개선하고, 실제 서비스 제공시 효율성 및 안전성을 보장할 수 있는 시스템에 대한 연구가 지속될 것이다.

[참고문헌]

[1] 과학기술정보통신부, “클라우드컴퓨팅 발전 및 이용자 보호에 관한 법률”, July, 2017.

[2] 한국인터넷진흥원, “클라우드 정보보호 안내서”, December, 2017.

[3] 문화체육관광부, “저작권법”, October, 2018.

[4] Cisco, “What Is Data Loss Prevention(DLP)?”, October, 2019.

공공장소 CCTV 프라이버시 확보 기법

권혁동* 장경배* 권용빈* 서화정*†

*한성대학교 IT융합공학부

A Method for CCTV privacy secure in public places

Hyeok-Dong Kwon* Kyoung-Bae Jang*, Yong-Been Kwon*, Hwa-Jeong Seo*†

*Division of IT convergence engineering, Hansung University.

요 약

CCTV(Closed Circuit Television, 폐쇄회로)는 침입 감시, 범죄 억제, 사고 예방 등의 다양한 목적으로 설치되는 카메라로 인가된 사용자만이 내용을 확인할 수 있기 때문에 CCTV라는 명칭을 사용한다. CCTV는 식당과 카페 같은 개인 사업장이나 지하철, 공항 등 다양한 공공장소에서 사용한다. 시간이 흘러 기술이 발전함에 따라 CCTV의 해상도 역시 향상되었으며 CCTV에 찍힌 피사체를 구분하기가 쉬워졌다. 하지만 발전한 성능으로 인해 의도치 않은 프라이버시 침해가 발생할 수 있다. 사람들이 사용하는 휴대폰이나 노트북 등의 전자기기가 CCTV에 촬영되며 고해상도 CCTV는 영상을 통해 화면의 내용을 대략적으로 또는 명확하게 확인할 수 있다. 이는 프라이버시 침해로 이어지며 CCTV 관리자에 한해 사람들의 전자기기 화면을 볼 수 있다. 본 논문에서는 제기한 문제를 해결하기 위해 CCTV가 전자기기 화면을 인식하여 비식별 단계를 거치는 시스템을 제안하며 프라이버시 침해 가능성을 예방하고자 한다.

I. 서론

현대 사회에서는 다양한 사고가 발생하며 자연스레 사고를 탐지 및 예방하기 위한 여러 수단이 발전하게 되었다. 그 중 하나로 CCTV(Closed Circuit Television, 폐쇄회로)는 강력한 예방 수단의 하나로 작용한다. CCTV는 흔히 말하는 감시카메라로서 지속적인 감시가 필요한 국지적인 지역을 촬영하는데 사용한다. 이는 감시에 사용되는 인력을 줄이고 불확실한 사람의 기억에 의존하지 않고 정확한 기록을 남길 수 있다. CCTV를 설치하는 것으로 얻을 수 있는 이점은 굉장히 많다. 가령 제한구역에 침입하는 비인가자를 감지하고 범죄의 억제 및 범죄자 추적에도 사용할 수 있으며, 화재 등 각종 재해 발생 감시 및 재해의 조기 파악과 같은 다양한 순기능을 가져온다.

기술이 발전함에 따라서 CCTV의 성능도 비약적으로 향상되었다. 하지만 이로 인해서 프라이버시 침해가 발생할 가능성이 생겼다. 일반적인 사람들은 CCTV의 유무를 그다지 의식하지 않으며, 자유롭게 자신들의 전자기기를 사용한다. 이 과정에서 CCTV에 사람들의 전자기기 화면이 촬영되며 고성능의 CCTV는 촬영한 전자기기 화면 내용을 파악할 가능성이 생긴다. CCTV는 인가된 사용자만이 영상을 열람할 수 있으나, 촬영된 사람들은 자신의 스마트폰과 같은 전자기기 화면이 자기도 모르는 사이에 촬영되고 노출된다는 사실에 불쾌할 것이다.

따라서 CCTV의 본연의 목적에 충실하도록 촬영을 방해하지는 않되, 촬영되는 사람들의 프라이버시는 지켜줄 수 있도록 전자기기 화면에 대한 비식별 조치가 필요하다. 이를 통해 CCTV 설치자는 CCTV 설치 목적을 달성할 수 있으며 동시에 사람들은 자신의 전자기기에 노출된 사생활이 CCTV에 촬영되는 것을 방지할 수 있다.

본 논문에서는 CCTV 프라이버시와 관련된 문제점에 대해 서술하며 이에 대한 해결책을

제시한다. 논문의 구성은 다음과 같다. 2장에서 CCTV의 발전상과 이에 관한 동향을 확인하며 3장에서 구체적인 문제점을 점검한다. 4장에서는 문제점을 해소할 방안에 대해 서술하며 5장에서 결론을 맺는다.

II. CCTV(Closed Circuit Television)

CCTV는 기술의 발전에 따라 자연스럽게 발전하게 되었다. 초기 CCTV는 아날로그 방식을 사용했으나 이후 디지털 방식의 CCTV가 등장하였고, 인터넷과 연결되는 네트워크 CCTV에 이르기까지 점점 진화하였다. CCTV는 예방 및 감시 용도로 사용되기 때문에 영상 감시 시스템 또는 영상 보안 시스템으로도 호칭된다.

현재 CCTV는 세대 분류상 3세대로 분류되며 각 세대별로 적용되는 기술을 정리하면 표 1과[1] 같다.

표. 1. CCTV 세대별 분류

Generation	Gen 1	Gen 2	Gen 3
Camera	Analogue, Digital	Digital, Network	Network, 3D, Thermal image
Resolution	SD	HD, FHD	UHD
Codec	MPEG2	MPEG4, H.264	HEVC
Medium	Coaxial cable	10/100M IPNetwork	Giga Network, Wireless
Video analysis	Monitoring	Motion/Fire detection	Object analysis
Storage	VCR, DVR	DVR, NVR	NVR, Cloud

CCTV를 감시 목적으로 사용할 경우에는 확보한 영상의 내용을 정확하게 분석할 수 있어야 한다. 이를 위해서는 더 좋은 화질의 영상이 필요할 것이며, 이는 CCTV가 제공하는 해상도와 직결된다. 초기 CCTV, 그 중에서도 아날로그 방식의 CCTV는 매우 낮은 해상도를 지원하였으며 이는 표 2[2]에서 확인할 수 있다.

표. 2. 아날로그 CCTV의 지원 해상도

QCIF	176 x 120 pixels
CIF	352 x 240 pixels
2CIF	704 x 240 pixels
4CIF	704 x 480 pixels
D1/WD1/Full D1	720 x 480 pixels

CCTV의 방식이 개선됨에 따라 디지털 방식이 등장하였고, 디지털 방식의 CCTV는 기존 아날로그 방식 CCTV보다 훨씬 더 높은 해상도를 지원하기 시작하였다. 이는 표 3[2]에서 확인할 수 있다.

표. 3. 디지털 CCTV의 지원 해상도

720p HD	1280 x 720 pixels
960p HD	1280 x 960 pixels
1.3 MP	1280 x 1024 pixels
2 MP	1600 x 1200 pixels
1080p HD	1920 x 1080 pixels
3 MP	2048 x 1536 pixels
5 MP	2592 x 1944 pixels
4K	3840/4096 x 2160 pixels

III. CCTV 프라이버시 침해 과정

2장에서 CCTV의 발전상을 확인할 수 있었다. 현재 사용되는 고해상도 CCTV는 상당히 선명한 화질의 영상을 제공할 수 있으며 영상 내부의 각종 사물뿐만 아니라 작은 글씨까지 구분이 가능하다. 가령 4K 해상도 카메라를 사용한 경우에는 딥러닝을 사용하여 차량 번호판 인식 학습을 시키는 것이 가능했으며[3], SD 해상도 카메라를 사용하여 얼굴 인식이 가능했다[4]. 이외에도 SD 해상도의 영상에 입혀진 문자를 인식에도 성공한 사례가 존재한다[5].

특히 [3]과 같은 고해상도 CCTV 카메라를 사용한다면 작은 문자도 인식할 수 있으며, 영상을 사람이 육안으로 확인한다면 내용을 파악할 가능성이 생긴다. 이에 따라 다음과 같은 상황이 발생한다면 프라이버시 침해 가능성이 생긴다.

우선 고해상도 CCTV가 설치된 카페나 공항 같은 공공장소가 있음을 가정한다. 이때 피해자는 CCTV의 촬영범위를 등지는 자리에 위치한다. 피해자는 자신의 위치에서 스마트폰이나 노트북 같은 전자기기를 사용할 수 있으며 해당 전자기기의 화면에는 피해자가 열람중인 각종 정보가 나타난다. CCTV는 피해자의 전자기기 화면을 촬영할 수 있고 후에 CCTV 관리자는 영상을 열람하는 것으로 전자기기 화면의 내용을 확인할 수 있다. 이를 도식화한 것이 그림 1 이다.

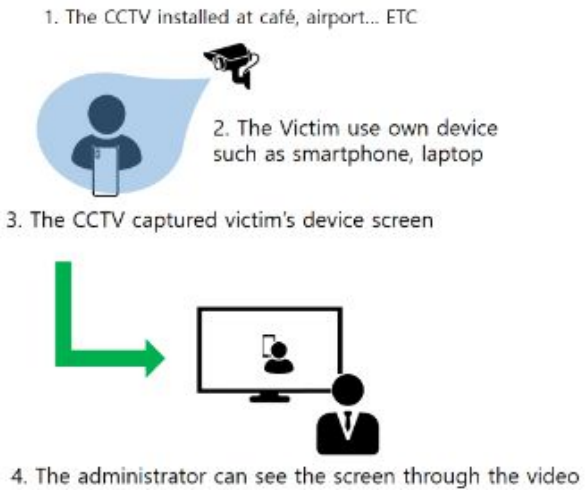


그림. 1. CCTV 프라이버시 침해 상황

즉, 공공의 목적으로 설치한 CCTV가 우연히 스마트폰이나 노트북 같은 전자기기의 화면을 촬영할 수 있다. 그리고 CCTV 관리자는 영상을 관리하면서 사람들의 전자기기 화면을 볼 가능성이 생긴다. 특히 스마트폰과 같은 극도로 개인화된 전자기기는 사용자 개인과 관련된 내용을 많이 저장하고 있으며, 사용자도 메신저, 커뮤니티 등의 사생활과 밀접하게 관련된 부분을 사용한다. 따라서 CCTV가 일반 사용자의 스마트폰을 촬영한다면 사생활과 관련된 부분이 촬영될 가능성이 매우 높으며, 사용자에게 매우 달갑지 않을 것이다.

IV. 제안 방안

본 논문의 3장에서 CCTV가 일반인들의 전자기기 화면을 촬영하여 CCTV의 관리자가 사람들의 화면을 확인할 수 있음을 알았다. 따라서 본 장에서는 문제의 해결 방안을 제안하고자 한다.

문제의 쟁점은 CCTV가 일반 사람들의 전자기기 화면을 촬영한다는 점이다. CCTV의 본연의 목적을 달성함과 동시에 사람들의 프라이버시를 지켜주기 위해서는 촬영한 영상에서 사람들의 전자기기 화면에 비식별 조치를 취하는 것으로 프라이버시를 확보할 수 있다. 구체적인 과정은 다음과 같이 정리할 수 있다.

CCTV가 특정 공간을 지속적으로 촬영하고 있을 때, 누군가가 스마트폰 등의 전자기기를 사용하면서 지나가는 장면이 촬영된다. CCTV가 확보한 영상에서 전자기기로 보이는 물체를 자동으로 블러, 모자이크 등의 비식별 조치를 취한다. 이를 도식화하면 그림 2와 같다.

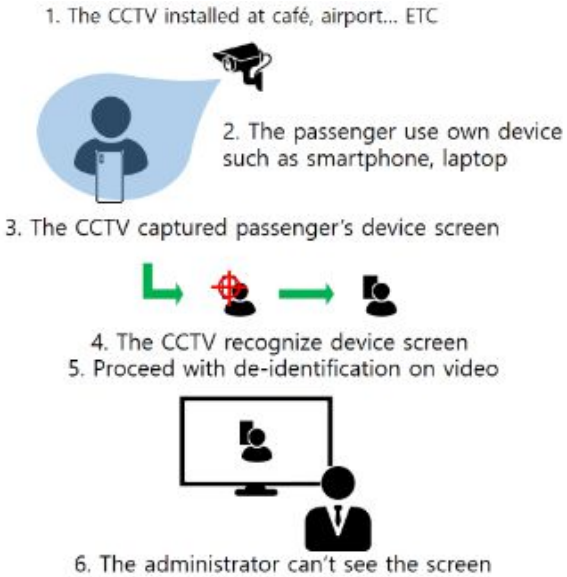


그림. 2. CCTV 비식별 조치

CCTV가 비식별 조치를 자동으로 취하기 위해서 크게 두 가지 방법을 제안한다.

우선 비식별 대상 기준을 설정한다. 그리고 설정된 기준에 따라 CCTV가 촬영한 영상에서 기준에 만족하는 물체에 자동으로 비식별 조치를 취한다. 가령 일정 크기 이하와 특정 형체를 지니는 발광체는 전자기기로 취급하고 비식별 조치를 취할 수 있다. 이 방법은 간단하며 CCTV의 성능의 영향을 덜 받는다는 장점이 있으나 전자기기가 아닌 다른 사물들도 비식별 조치를 거치는 오작동을 할 가능성이 존재한다.

때문에 단순한 물체 인식 방식에서 조금 더 발전한 두 번째 방법은, 머신러닝을 통해 전자기기로 보이는 물체를 학습시키는 것이다. 정확한 학습을 통해 전자기기를 식별하게 된다면 사물 인식 방식보다 훨씬 정교한 비식별 조치가 가능할 것이다. 하지만 고성능의 CCTV가 요구될 것이며 이에 따라 CCTV 관리자의 부담이 가중될 수 있다.

V. 결론

본 논문에서는 CCTV가 일반인의 프라이버시를 침해할 가능성이 있음을 제기하고 이에 관한 해결책을 제시하였다. CCTV 같은 영상 보안 시스템은 범죄 예방과 같은 본래의 역할을 충실하게 수행하고 있으나 본 논문에서 확인한 문제점으로 인하여 관리자가 악의적인 의도 하에 다른 사람들의 스마트폰이나 노트북 화면을 훔쳐볼 수 있다.

본 논문에서 제안하는 방법을 통해 의도치 않은 프라이버시 침해를 예방할 수 있다. 다만 논문에서 제안하고자 하는 방안은 구현물이 없으며 제안 방안이 효과적인지를 보이기 위해서는 정밀한 실험 결과가 필요할 것으로 보인다. 따라서 본 논문의 제안 방안을 조금 더 구체적으로 개선한 다음, 실제로 구현을 통해 프라이버시 침해 예방에 어느 정도의 기여를 할 수 있는지 확인해야 한다. 이를 위해서 본 논문의 주제에 관해 지속적으로 연구와 개선이 진행될 필요가 있다.

[참고문헌]

[1] B.S.Choi, The Trends of Video Surveillance Systems market and Technologies, S&T Market Report, vol.51, September, 2017.

[2] CCTV Video Resolution Comparison Chart. Mammoth Security Incorporation [Internet]. Available: https://mammothsecurity.com/comparing-cctv-video-resolution/

[3] K.S.Son, J.W.Kim, and J.H.Lim, Convergence CCTV camera embedded with Deep Learning SW technology, Commercializations Promotion Agency for R&D outcomes, Journal of Korea Convergence Society, vol. 10, No. 1, pp. 103-113, October, 2019.

[4] T.Marciniak, A.Chmielewska, R.Weychan, M.Parzych, and A.Dabrowski, Influence of low resolution of image on reliability of face detection and recognition, Multimedia Tools and Applications, vol.74, No. 12, pp 4329-4349, June, 2015.

[5] B.C.Kim, C.H.Choi, and B.J.Son, Character Recognition of Low Resolution CCTV Images of Sewer Inspection, Journal of the Korea Institute for Structural Maintenance and Inspection, vol.20, No.5, pp 58-65, September, 2016.

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 B

Efficient Lightweight Authentication Scheme for Internet of Things Application

Abhijeet Thakare, Euijong Lee, Young-Gab Kim*

Dept. of Computer and Information Security, Sejong University

Abstract

To meet the need for modern Internet of Things (IoT) devices, many researchers attempted to produce a lightweight, secure authentication scheme. However, existing schemes unable to completely satisfy the lightweight definition due to high computation, communication, and storage cost. To overcome this issue, we proposed an efficient, secure lightweight authentication scheme for IoT applications. Our experimental results show that the proposed scheme holds successfully lightweight criteria by reducing the number of hash and elliptic-curve cryptography multiplication operations, communication, and the storage parameters in comparison to other existing schemes. The security analysis using AVISPA and ProVerif tool shows that the proposed scheme is secure against security attack.

I. Introduction

Currently, the Internet of Things (IoT) devices lacks in limited computation power, storage, and power capability. Given this fact, many researchers attempted to design a lightweight elliptic-curve cryptography (ECC) based authentication scheme. However, the current cryptographic method does not take the resource-constrained limitation of IoT devices into consideration. In addition, to ensure secure communication between millions of nodes, the unique secret key is needed every time. Therefore, the existing high computation, authentication scheme consumes too much time, storage, and computation energy for key distribution and management. This is results in current IoT devices suffer from multifaceted computational nature, memory, and communication overhead [1]. This turn into inspiration to proposed efficient, lightweight, ECC based authentication scheme for IoT applications.

* Corresponding author

II. Survey

We analyze various research papers from 2011 to 2018 [2-5]. The analysis state that the researchers claimed their schemes are secure; however, it fails to support lightweight expectations [6]. The above analysis infers that lightweight design, privacy and security are major concerns for designing an efficient and secure IoT system. Several studies [3-7] have been done in this context; however, hardly of them satisfy the complete definition of the lightweight. In this regard, we presented a lightweight authentication scheme in the next section.

III. Proposed Scheme

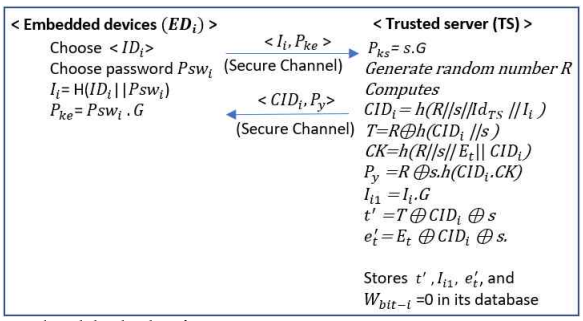
The proposed scheme consists of two phases, which include the registration phase and the other is the password authentication phase, which includes the session key distribution phase. In the registration phase,

to prevent the proposed scheme from device anonymity attack, we concealed the device identity into using a hash function. We also used one server-generated identity to authenticate each device by a server. The detail description of the overview of the proposed architecture approach can be found in [5]. We depicted the schematic of the proposed scheme in Figures 1 and 2. For the convenience of description, we used different terminology and notations in the paper, which is summarized in Table I.

Table I. Notations guide

Notations	Description
Psw_i, P_{ke},P_{ks}	Secret password, Public key of the devices, Public key of the server
I_i,CID_i	Device hash ID, Server generated ID
CK,s,	Session cookie, Server private key
E_{ki}, D_{ki}, S_K	Encryption/Decryption key, Session key
h, (x_{i1}, x_{j1})	Hash function, Ephemeral secretes
(X_i, Y_i, X_j, Y_j)	Challenges
W_{bit}	Working bit represent status
E_t, (A, T, P_y)	Expiration time of the cookie, Equations
$W_{bit-i}=0$	Working status of devices

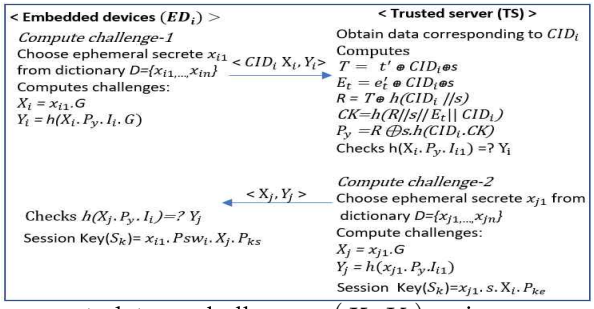
The explanation of the proposed scheme for the registration phase in Figure 1 are as follows: Initially, devices ED_i registered with the server using wrapped identity I_i with their public key P_{ke}. The server computes another unique identity CID_i and compute other parameters (T, CK) and wraps the session cookies CK inside the challenge Py. Server send (CID_i, Py) as response to



embedded devices.

Figure 1.: Registration Phase

Then, as depicted in Figure 2, devices



computed two challenges (X_i, Y_i) using

Figure 2.: Authentication of scheme

ephemeral secret x_i. Using identity CID_i and computed challenge (X_i, Y_i) the device requests an authentication request to the server. The server verifies devices challenge using some computation of variables. Once verification comes true, the server computes another challenge (X_j, Y_j) for devices and sends it back as the response to an earlier request. The devices verify the challenge using some computation, and if verification is correct, devices compute a session key S_k . Whenever the devices communicate with the server, the working bit W_{bit-i} is set to one for the corresponding connected communication; otherwise and after the termination of communication, it is set to zero. It helps to prevent many logged-in users attack [5].

IV. Cryptoanalysis

In this section, we presented how the proposed scheme is free from all known cryptographic attack. The proposed scheme used the elliptic curve discrete logarithm problem (ECDLP) and the elliptic curve Diffie - Hellman problem (ECDHP), which strengthens the hardness of the proposed scheme and make it more secure [3]. We also perform the formal security investigation using a widely used AVISPA tool [5] with CL- Atse backend. This analysis helps us to check the security strength of the proposed scheme against attacks. Figure 3 shows that

the scheme is safe, which infer that the

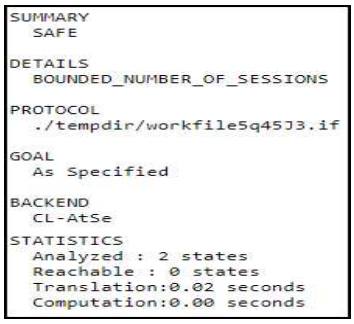


Figure 3: Analysis result using AVISPA

proposed scheme is secure against a replay attack and all active and passive attacks. We also tested the scheme using a widely used ProVerif tool [8]. Figure 4 depicted analysis using the ProVerif tool confirms that the initial two server and user events are executed successfully. The third query execution verifies that the attacker is not able to find a session key. Hence, the proposed scheme possesses authentication property.

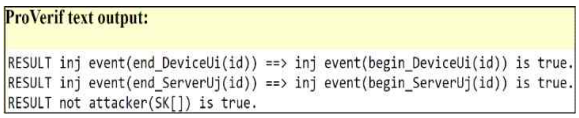


Figure 4.: Output of query execution

V. Evaluation

Let T_h represent the execution time of a hash operation, T_{ecm} represent the execution time of an ECC point multiplication operation and T_{eca} represent the executation time of an ECC point addition operation. The standard execution time (computation cost) of T_h, T_{ecm}, and T_{eca} are $2.3\mu s$, $2226\mu s$ and $28.8\mu s$ respectively [9]. The other operations (i.e., concatenation, comparison, and XOR) has been overlooked, because of inexpensive computation. The proposed scheme consumes $7T_h+6T_{ecm} = 13.3721\times10^3\mu s$ for computational operation. Likewise, we computed the computation cost of other existing scheme (i.e., Liao et al. [2], Kalra et.al. [3], Chang et.al. [4], and Kumari et.al. [5]). The Table II shows, to save computation cost, the proposed scheme used a minimum number of the hash and ECC point multiplication operation in comparison to other existing mentioned scheme.

Table II: Computation cost comparison

Schemes	Total Cost
Liao et al. [2]	$5T_{eca}+6T_{ecm} = 13.5000\times10^3\mu s$
Kalra et.al. [3]	$7T_h+7T_{ecm} = 15.5981\times10^3\mu s$
Chang et.al. [4]	$8T_h+7T_{ecm} = 15.6004\times10^3\mu s$
Kumari et.al. [5]	$7T_h+8T_{ecm} = 17.8241\times10^3\mu s$
Proposed	$\mathbf{7T_h+6T_{ecm} = 13.3721\times10^3\mu s}$

We consider some assumptions from the existing research [9] for computing the communication and storage cost. An ECC point $(X, Y) = (Xp, Yp) \epsilon\, Ep\,(a, b)$ takes (160+160)=320 bits. In the proposed scheme, the communication parameters are (X_i, Y_i, CID_i) and (X_j, Y_j).Thus, communication cost is (320+320+160+ 320+320)=1440, which is lower as compared to other existing schemes depicted in Table III. In the proposed scheme, devices store two-variable (CID_i, Py) and consume (160+160) = 320 bits, which is almost lower as compared to other existing schemes depicted in the Table III.

Table III: Communication and storage cost comparison

Schemes	Communication cost/bit	Storage cost/bit
Liao et al. [2]	2260	815
Kalra et.al. [3]	1760	320
Chang et.al.[4]	1760	320
Kumari et.al [5]	1760	480
Proposed	**1440**	**320**

VI. Conclusion and Future Work

Based on research investigation, we found that existing schemes are secure, however they unable to stand on the lightweight computation, communication, and storage over the other existing scheme. In addition, analysis of the proposed scheme using AVISPA and ProVerif tool indicate that the proposed scheme is secure against all active

and passive attack with successful mutual authentication. Therefore it makes the proposed scheme efficient and practical for over IoT applications. In future work, we would like to include the practical implementation of the proposed scheme in medical application.

VII. Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00231, Development of artificial intelligence-based video security technology and systems for public infrastructure safety).

References

[1] Chen, C.M., Wang, K.H., Fang, W., Wu, T.Y. and Wang, E.K., 2019. Reconsidering a lightweight anonymous authentication protocol. Journal of the Chinese Institute of Engineers, 42(1), pp.9-14.

[2] Liao, Y.P. and Hsiao, C.M., 2014. A secure ECC-based RFID authentication scheme integrated with ID-verifier transfer protocol. Ad Hoc Networks, 18, pp.133-146.

[3] Kalra, S. and Sood, S.K., 2015. Secure authentication scheme for IoT and cloud servers. Pervasive and Mobile Computing, 24, pp.210-223.

[4] Chang, C.C., Wu, H.L. and Sun, C.Y., 2017. Notes on “Secure authentication scheme for IoT and cloud servers”. Pervasive and Mobile Computing, 38, pp.275-278.

[5] Kumari, S., Karuppiah, M., Das, A.K., Li, X., Wu, F. and Kumar, N., 2018. A secure authentication scheme based on elliptic curve cryptography for IoT and cloud servers. The Journal of Supercomputing, 74(12), pp.6428-6453.

[6] Shieh, W.G. and Horng, W.B., 2008, June. Efficient and complete remote authentication scheme with smart cards. In 2008 IEEE International Conference on Intelligence and Security Informatics (pp. 122-127). IEEE.

[7] El-hajj, M., Fadlallah, A., Chamoun, M. and Serhrouchni, A., 2019. A survey of internet of things (IoT) Authentication schemes. Sensors, 19(5), p.1141.

[8] Mahmood, K., Chaudhry, S.A., Naqvi, H., Kumari, S., Li, X. and Sangaiah, A.K., 2018. An elliptic curve cryptography based lightweight authentication scheme for smart grid communication. Future Generation Computer Systems, 81, pp.557-565..

[9] Kilinc, H.H. and Yanik, T., 2013. A survey of SIP authentication and key agreement schemes. IEEE Communications Surveys & Tutorials, 16(2), pp.1005-1023

임베디드 OS에서의 메모리 보호 메커니즘 적용 한계성

권순홍*, 이종혁*

*상명대학교 프로토콜공학연구실

Limitation of Memory Protection Mechanisms Applied in an Embedded OS

Soonhong Kwon*, Jong-Hyouk Lee*

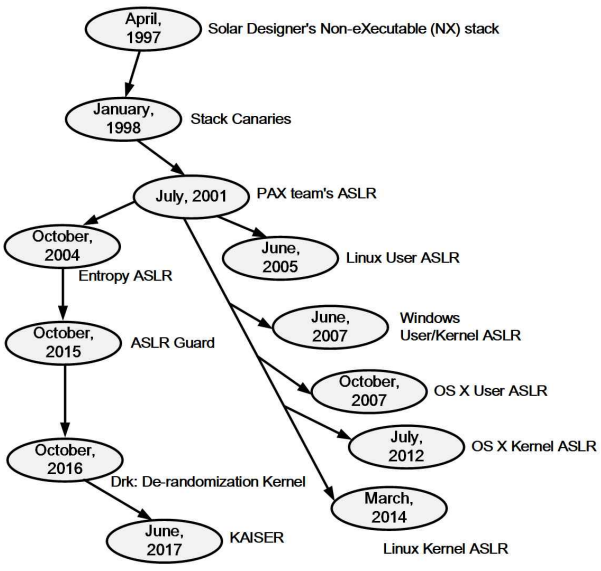
*Protocol Engineering Lab., Sangmyung University.

요 약

임베디드 시스템은 소비자 전자제품에서 중요한 인프라에 이르기까지 어디에나 있으며, 사물인터넷(IoT)이 증가하면서, 임베디드 시스템은 일상생활의 모든 측면에 걸쳐 확산되기 시작했다. 이러한 시스템은 특정 목적을 위해 설계됨에 따라 보안과 프라이버시에 대한 우려를 가지고 있지만, 임베디드 세계에서 이러한 측면에 대한 적절한 보안이 매우 부족한 경우가 많다. 이에 본 논문에서는 임베디드 시스템에 적용되어 있는 메모리 보호 기법이 GP(General Purpose) 시스템에 적용되어 있는 메모리 보호 메커니즘과의 차이점에 대해 설명하며, 임베디드 시스템에 GP 시스템의 메모리 보호 메커니즘 적용의 어려움에 대해 설명한다.

I. 서론

최근 특정 목적을 위해 설계된 임베디드 장비의 사용이 증가함에 따라, 임베디드 장비 보안에 대한 관심 또한 증가하고 있는 실정이다. 임베디드 시스템과 관련하여 메모리 손상 문제는 카스퍼스키의 ICS(Industrial Cyber Security) 취약점 연구에서 말한 것처럼 가장 널리 알려진 취약점 범주 중에서 일관되게 순위를 매기고 있다. 일반 PC나 서버와 같은 GP(General Purpose) 시스템의 경우, 메모리 공격에 따른 메모리 보호 메커니즘이 [그림 1]과 같이 지속적으로 개발되고 있지만, 임베디드 OS의 경우, CSPRNG 부재, OS Virtual Memory 부재, MMU/MPU의 부재로 인한 메모리 보호 메커니즘을 지원하지 않는 OS가 대다수이다. 이에 따라 본 논문에서는 임베디드 OS의 메모리 보호 메커니즘의 적용의 한계성에 대해 상세히 설명한다 [1]. 본 논문의 2장에서는 메모리 보호 메커니즘 및 공격 기법에 대해 설명하며, 3장에서는 임베디드 OS에서의 메모리 보호 메커니즘의 적용 한계에 대해 설명한다. 4장에서는 본 논문의 결론을 맺는다.



[그림 1] GP 시스템의 메모리 보호 기법 발전

II. 관련 연구

본 장에서는 일반 PC나 서버와 같은 GP(General Purpose) 시스템에 적용되어 있는 메모리 보호 메커니즘에 대해 설명한다. GP 시스템에는 Stack Canary, NX bits, Relro, PIE, ASLR등의 많은 보호 메커니즘이 적용되어 있지만, 임베디드 OS와 비교하여 공통점을 가지고 있는 메모리 보호 메커니즘인 Stack Canary, Nx bits 및 ASLR에 대해 설명한다.

2.1 스택 카나리(Stack Canary) [2]

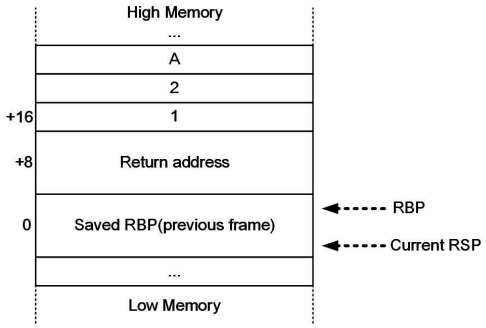
RedHat사의 엔지니어들이 IBM사의 ProPolice에 대한 문제점을 확인하고, 2005년 GCC 4.1에 스택 스매싱 보호 기법을 보완하여 SSP(Stack Smashing Protector) 패치를 적용하였다. 스택 카나리는 GCC 4.1에 포함된 SSP 기법 중 하나로써, 버퍼 오버플로우 발생 시, 공격 시도를 탐지하는 메모리 보호 메커니즘이다.

스택 카나리는 프로그램 실행 시 main() 함수 이전에 실행되는 _libc_start_main() 함수에 의해 설정된다. 반환 주소와 스택 프레임의 포인터 정보를 저장할 때, 반환 주소 하위 메모리 영역에 카나리 값을 추가한다. 해당 함수가 종료되며 반환 주소로 이동할 때, 카나리 값이 변경되었다면 버퍼 오버플로우 공격이 발생되었다고 판단한다. 이후, Stack Smashing Detected handler가 동작하여 프로그램을 중단한다. 카나리 기법을 사용하면 함수 프롤로그 과정에서 '카나리 값 '(Linux)' 또는 '쿠키(Window)'라고 불리는 랜덤 값을 추가한다. 함수 프롤로그는 함수가 시작될 때 스택에서 가장 먼저 수행되는 동작이다. 새로운 함수가 추가될 때 스택 프레임이 추가된다. 함수 프롤로그 과정은 [그림 2]와 같다.

1	push %rbp
2	push %rbp, %rsp

[그림 2] 함수 프롤로그

1번째 라인의 'push %rbp'에서 rbp 레지스터는 호출된 함수(caller) 스택 프레임의 기준 주소를 저장한다. 이는 호출된 함수(callee) 종료 후 돌아올 주소를 저장하는 과정이다. 호출한 함수의 기준 주소는 아래 [그림 3]의 'SFP(Saved Frame Pointer)' 영역에 저장된다. 이후 2번째 라인의 'mov %rbp, %rsp' 인스트럭션에서, 'rsp'에 저장된 값을 'rbp'에 저장한다. 다시 말해, 호출된 함수와 기준 주소가 저장된 'rsp' 레지스터 값은 'rbp'에 저장해 'rbp'가 호출된 함수의 기준 주소를 가리키도록 한다. 스택 프레임에서 함수 프롤로그 동작은 아래 [그림 3]과 같다.



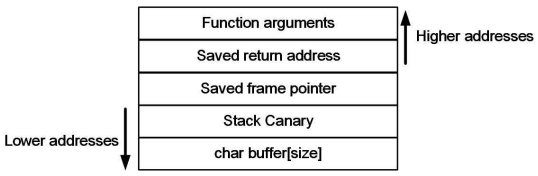
[그림 3] 함수 프롤로그의 동작

함수가 모두 실행되면 함수 에필로그(epilogue)가 수행된다. 아래 그림 4는 함수의 에필로그를 나타낸다.

1	mov %rsp, %rbp
2	pop %rbp
3	ret

[그림 4] 함수 에필로그

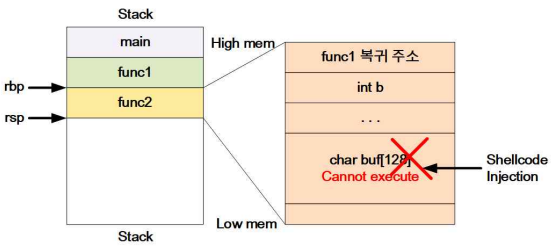
함수 에필로그 동작은 1번째 라인의 'mov %rsp, %rbp' 인스트럭션에서, 호출된 함수의 기준 주소를 갖는 'rbp' 레지스터 값을 'rsp;로 복사한다. 즉, 'rsp'가 'rbp'와 같은 위치를 가리키도록 한다. 이후 2번째 라인의 'pop %rbp'에서, 프롤로그 과정 시 SFP에 저장된 호출한 함수의 기준 주소를 복구한다. 그리고 3번째 라인의 'ret' 인스트럭션을 통해 호출한 함수로 복귀한다. 스택 카나리 값을 생성하는 방법은 크게 랜덤, 랜덤 XOR, 터미네이터등이 있다. StackGuard에서 구현된 스택 카나리는 '/dev/random' 또는 '/dev/urandom'에서 생성된 의사 난수 값(OS CSPRNG)을 사용한다. 스택 카나리는 SFP 직전 하위 메모리에 삽입된다. 공격자가 스택스매싱 공격을 수행할 때, 스택 프레임의 반환 주소를 공격자가 생성한 공격 코드(e.g., 쉘 코드)가 있는 주소로 변경한다. 반환 주소를 변경하기 위해 버퍼 오버플로우에 취약한 문자열 입력 함수를 이용한다. 반환 주소 영역 전까지 임의의 문자로 채워지는 과정에서 카나리 값도 변경된다. 스택 카나리 보호 기법이 적용되는 시스템에서 함수가 종료될 때, 에필로그 과정에서 프롤로그 시 생성된 카나리 값(원본 카나리 값)과 비교한다. 만약 호출 시 생성되었던 카나리 값과 다르다면, "Stack Smashing Detected"가 출력되고, 프로그램이 종료된다. 다음 [그림 5]는 카나리가 삽입된 스택 프레임의 구조를 나타낸다.



[그림 5] 카나리가 삽입된 스택 프레임

2.2 NX bits(No-eXecutable bits)/ESP

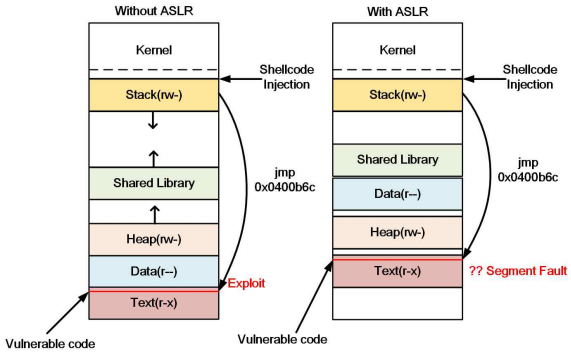
공격자에 의해 삽입된 악의적인 코드가 실행될 수 없도록 메모리 영역을 실행 전용과 쓰기 전용으로 분리하는 기법으로써, 2003년 OpenBSD 3.3에서 W^X(Write XOR eXecution)로 처음 구현되었다. 그 후 마이크로소프트사는 윈도우즈 환경을 위한 W^X기법으로 Windows XP SP2부터 DEP(Data Execution Prevention)을 제공한다. 리눅스에서는 PaX에 의해 Kernel 2.6.8부터 NX bits(No-eXecutable bits)로 제공된다. 초창기에 개발된 W^X는 소프트웨어적으로 구현되었기에 오버헤드가 발생한다. 이러한 오버헤드를 줄이기 위해 DEP와 NX bits는 CPU에서 하드웨어적으로 제공된다. NX bits는 메모리 영역에 대해서 W^X의 권한으로 관리하며, 데이터 변경이 가능할 경우(W) 실행이 불가능하게 하며, 코드 실행이 가능할 경우(X) 해당 영역에 데이터 변경을 수행하지 못하도록 함으로써 메모리를 보호한다 [3]. 다음 [그림 6]은 NX bits의 동작을 보여준다.



[그림 6] NX bits 동작

2.3 ASLR(Address Space Layout Randomization)

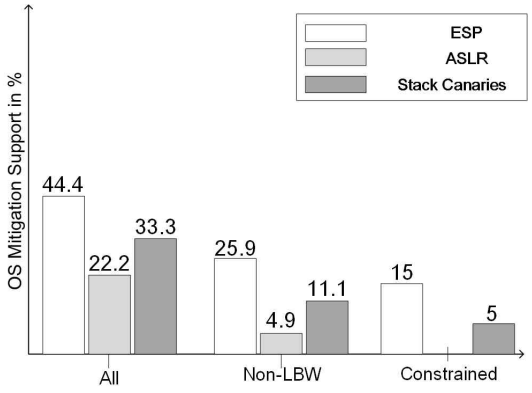
ASLR(Address Space Layout Randomization)은 주로 메모리 손상 공격을 방어하는데 사용되는 메모리 보호 메커니즘 중 하나이다. ASLR이 적용됨에 따라 공격자는 메모리 손상 취약점(e.g., 버퍼 오버플로우)을 쉽게 악용하여 제어 흐름을 자신이 원하는대로 변경하거나 데이터 흐름을 조작하고 악성 코드를 실행시킬 수 없게 한다. ASLR은 메모리 레이아웃(e.g., 실행 파일 및 데이터 위치)를 무작위로 지정하여 제어 흐름 하이재킹 및 인 메모리 데이터 조작이 프로세스 메모리 레이아웃을 이해해야 하기 때문에 공격자가 취약점을 악용하기 어렵게 만든다. 따라서 ASLR은 확률적인 보호 메커니즘으로 표현 또한 가능하다. 특히, ASLR을 사용하면 운영체제가 로드시 코드와 데이터를 임의로 배치하므로 공격자가 코드 및 데이터의 위치를 유추하기 어렵게 된다. 공격자는 코드 및 데이터의 위치를 유추하기 어렵기 때문에 랜덤화 엔트로피에 대해 전수 조사(brute force)를 수행할 수 있으나, 현실적으로 불가능하다. ASLR은 스택 카나리, NX bit등과 같은 메모리 보호 기법과 함께 대부분의 최근 운영체제에 탑재되어 메모리 보호 메커니즘으로 사용되고 있다 [4]. 다음 [그림 7]은 ASLR의 동작에 대해 설명한다.



[그림 7] ASLR 동작

III. 임베디드 OS에서의 메모리 보호 메커니즘 한계

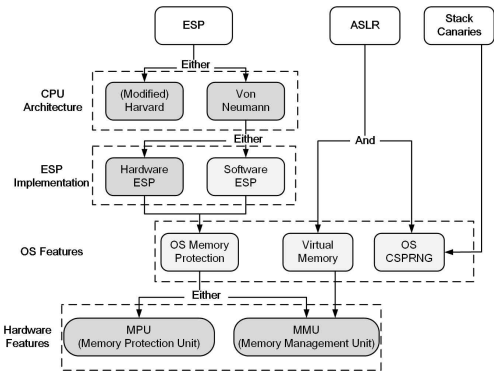
본 장에서는 일반 PC나 서버와 달리 특정 목적을 위해 만들어진 임베디드 시스템에서의 메모리 보호 메커니즘 적용의 한계에 대해 설명한다. 임베디드 장비의 경우, 일반 PC나 서버와 달리 제한된 리소스(e.g., 처리 속도, 메모리 및 전력 소비량) 및 낮은 엔트로피 환경으로 인해 GP 시스템의 메모리 보호 메커니즘을 적용하는데 한계가 존재한다. WETZELS, Jos 논문 [1]을 통해 확인할 수 있듯이, Non-LBW(Non-Linux, BSD 및 윈도우 기반 OS)를 제외하고, 임베디드 시스템을 위해 설계된 초소형 OS인 Constrained OS의 경우 소수만이 메모리 보호 메커니즘을 지원하고 있음을 [그림 8]을 통해 확인할 수 있다. 다음은 임베디드 시스템에 GP 시스템의 메모리 보호 메커니즘을 적용의 한계에 대해 설명한다.



[그림 8] OS에 따른 메모리 보호 메커니즘 지원

3.1 하드웨어 및 소프트웨어 의존성

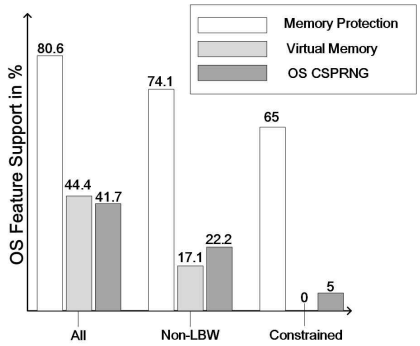
임베디드 시스템에 GP 시스템의 메모리 보호 메커니즘을 적용하기 힘든 이유 중 첫 번째는 하드웨어 및 소프트웨어의 의존성 문제이다. [그림 9]는 메모리 보호 메커니즘의 하드웨어 및 소프트웨어 종속성을 매핑하여 메모리 보호 메커니즘의 채택이 복잡해지는 몇가지 제약에 대해 나타내고 있다. 임베디드 장치는 특정 작업을 위해 설계되었기 때문에 headless(모니터와 키보드 같은 엔트로피를 발생 시킬 수 있는 장치가 필요 없는 장비) 및 diskless와 같은 제한된 리소스를 가지게 된다. 하드웨어의 경우, 단순하고 고급 기능이 부족한 경향이 대다수이며, 소프트웨어의 경우, 하드웨어의 제약 조건에 맞게 설계된다 [1].



[그림 9] 하드웨어와 보호 메커니즘 의존성

3.2 CSPRNG, OS Virtual Memory, MMU/MPU

[그림 9]를 통해 알 수 있듯이, Stack Canary의 경우, OS CSPRNG를 지원해야만 적용이 가능하며, ESP의 경우, CPU 아키텍처가 폰 노이만 방식 또는 (Modified) 하버드인지에 따라 적용 방식이 다르다. 폰 노이만 아키텍처의 경우, 하드웨어 ESP 및 소프트웨어 ESP 구현이 가능하며, MMU/MPU를 지원해야 적용이 가능하다. ASLR의 경우, 적용을 위해서는 OS에서 OS Virtual Memory, OS CSPRNG, MMU/MPU를 지원해야한다. Constrained OS의 경우, [그림 10]과 같이 5%만이 OS CSPRNG를 제공하며, Virtual Memory는 지원하지 않으므로 Stack Canary, ESP, ASLR 메모리 보호 메커니즘의 적용에 한계가 존재한다 [1].



[그림 10] OS별 기능 지원

IV. 결론

IoT 환경에서 특정 목적을 위해 설계된 임베디드 장비의 사용이 증가함에 따라, 공격자들로 하여금 많은 공격에 노출되어 있다. 이에 본 논문에서는 임베디드 시스템의 보안 취약성 및 GP(General Purpose) 시스템의 메모리 보호 메커니즘을 임베디드 시스템에 적용하는데 존재하는 한계점에 대해 설명하였다. 임베디드 시스템를 위해 설계된 Constrained OS의 경우, 메모리 보호 메커니즘 지원을 위한 OS CSPRNG, OS Virtual Memory, MMU/MPU를 지원하지 않아 Stack canary, ESP, ASLR의 적용에 한계가 있음을 설명하였다. 추후, 임베디드 시스템을 위해 설계된 OS인 Constrained OS를 위한 메모리 보호 메커니즘을 설계하고자 한다.

Acknowledgement

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2019-2015-0-00403)

[참고문헌]

[1] WETZELS, Jos. Internet of Pwnable Things: Challenges in Embedded Binary Security. ; login:, 2017, 42.2.

[2] COWAN, Crispan, et al. Stackguard: Automatic adaptive detection and prevention of buffer-overflow attacks. In: USENIX Security Symposium. 1998. p. 63-78.

[3] http://index-of.es/EBooks/NX-bit.pdf, 2019년 10월 접속

[4] SHACHAM, Hovav, et al. On the effectiveness of address-space randomization. In: Proceedings of the 11th ACM conference on Computer and communications security. ACM, 2004. p. 298-307.

IoT 플랫폼 상호운용을 위한 OCF 게이트웨이 구현

수란토 나우팔*, 이스마일*, 라라사티 하라스타*, 허신욱*, 김호원*
*부산대학교 정보컴퓨터공학부

OCF Gateway Implementation for IoT Platforms Interoperability

Naufal Suryanto*, Ismail*, Harashta Tatimma Larasati*, Shinwook Heo*, Howon Kim*

*Department of Computer Science and Engineering,
Pusan National University.

요 약

OCF IoTivity is one of the Internet of Things (IoT) platforms that have gained popularity for IoT deployment in recent years. However, since the use of other IoT platforms also grows as high, realizing an interoperable environment to other IoT platforms will eventually be essential. Unfortunately, OCF Iotivity acts in a device-to-device manner and currently only supports the CoAP protocol for its communication. A system that can translate OCF communication protocol to other IoT platforms needs to be established in order to achieve interoperability. In this study, we propose a method to bridge the connectivity between OCF IoTivity and other platforms by constructing a gateway in an OCF client. This gateway will act as a relay to connect different IoT platforms to OCF servers. Furthermore, we take into account the functions in each platform to avoid overlapping. Our proposed method implemented and tested among three different platforms: OCF IoTivity, IBM Watson, and oneM2M as a proof-of-concept (PoC), can be utilized further as the first step to achieving interoperability.

I. Introduction

In the time of the ever-increasing adaptation of Internet of Things (IoT), more and more vendors are entering the market to offer their own IoT platforms. As reported by IoT Analytics[1], the number of IoT platforms has reached 450 in 2016, almost doubling from around 260 in the previous year and is predicted to keep arising. Even though this brings advantages for users to choose the most suitable platform from the wide variety of options, the lack of interworking among platforms[2] will eventually be a restraint since users are softly forced to remain using the same platform for future use (e.g. when they wish to expand or add new services in complement to the existing ones) even when another platform may provide better support for a distinct scenario. Therefore, interoperability among IoT platforms is said to be one of the most essential requirements to be achieved in the field of IoT[3].

Among others, OCF IoTivity is one of the emerging IoT platforms that has risen in popularity in recent years. OCF IoTivity is

the open-source implementation of the OCF standard. It works in a device-to-device manner, with every device is a server by default. However, by design, OCF works only within the same network. Hence, there will be a difficulty when it needs to be accessed from other platforms. To address this issue, we propose to construct an OCF IoTivity gateway that can facilitate the interoperability among different platforms.

The contribution of this paper is as follows: (1) We build a proxy to connect CoAP protocol in OCF to MQTT and HTTP protocols utilized in other platforms; (2) We expose OCF device availability status to non-OCF platforms; (3) We implement access control method to secure the connection from and to OCF IoTivity.

II. Proposed Method

3.1 Architecture Design

In order to enable interworking among OCF and Bridged Devices, we propose to build an OCF gateway that will act as a bridge to connect different IoT platforms. We adopt the concept of Interworking Proxy Entity (IPE) presented in oneM2M standard [6] with addition of HTTP and MQTT proxy to facilitate the communication process.

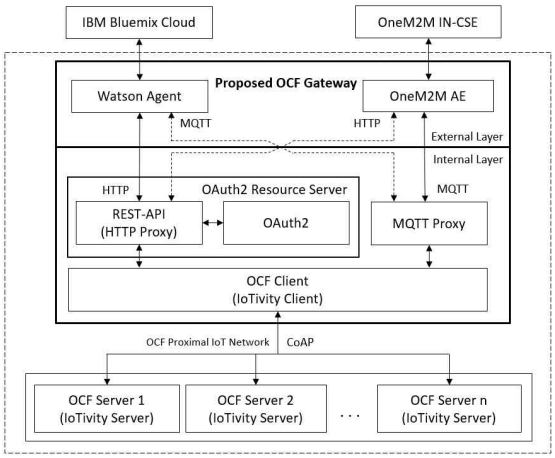


Fig. 1. Architecture of the Proposed Gateway

Figure 1 depicts the architecture of the proposed gateway. The gateway comprises two different layers: internal and external layer. Internal layer handles the essential operations of OCF gateway and consists of three different entities: OCF Client, MQTT Proxy, and OAuth2 Resource Server which itself contains HTTP Proxy and OAuth2. The external layer can be thought as the extension of other IoT platforms which wish to interwork with OCF Platform and is composed of other platform's components functioning as an agent: in our case, Watson Agent to connect to IBM Bluemix Cloud and oneM2M Application Entity to connect to oneM2M infrastructure. The entities comprising an internal layer are fixed while external layers can be modular depending on what platforms need to be connected to OCF IoTivity. For the time being, our architecture supports MQTT and HTTP protocols, which is handled by MQTT Proxy and HTTP Proxy within OAuth2 Resource Server.

The core entity in our architecture is the OCF Client, which has two main functions: device discovery and data forwarding. In terms of device discovery, when there is a newly connected OCF Server in the network, OCF Client will detect and then record the device information and attributes (i.e. UUID, IP address, resource path/address, owned resources) in its memory. A more detailed sequence diagram is presented in Fig. 2. Data forwarding functions to facilitate requests from other IoT platforms. It is done by first performing a resource mapping from UUID to IP address and vice versa. Since OCF Client keeps the attributes of each OCF Server, data forwarding can then be performed using CoAP protocol according to the mapping.

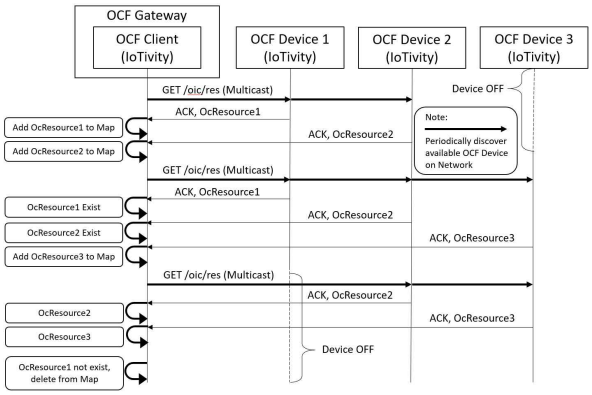


Fig. 2. Device Discovery and Auto Registration Flow

3.2 Communication Flow

The proposed OCF gateway supports HTTP and MQTT protocol for communication with other platforms as shown in Fig. 3 and Fig. 4, respectively. In order to forward OCF data to either oneM2M or IBM Watson, there are proxies inside the gateway which will translate from CoAP into the corresponding protocols for other IoT platforms (HTTP or MQTT) and forward the message to the resource requester (oneM2M AE or IBM Watson Agent).

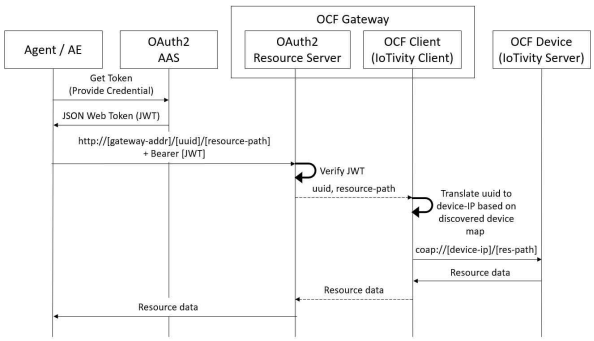


Fig. 3. Communication Flow via HTTP

Streaming data and acquiring OCF device status is provided by MQTT protocol. In this mode, OCF gateway will periodically check the availability of OCF Servers via the OCF native notification system and publish the updated status and resource values to the MQTT broker. Watson Agent or oneM2M AE can obtain the status and resources value by subscribing to the same topic via the broker. In terms of HTTP, data will be provided by previously requesting the data.

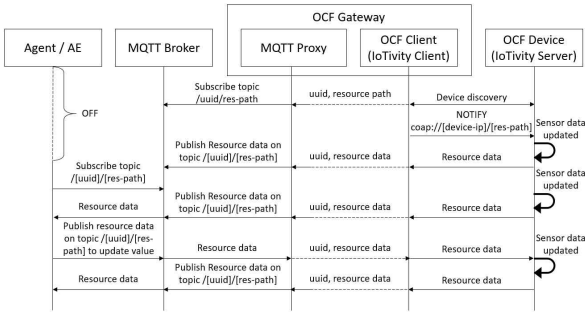


Fig. 4. Communication Flow via MQTT

In addition to facilitating interoperability, the proposed OCF gateway also provides security for communication between platforms. For HTTP connection, we employ a token mechanism using OAuth 2.0 whereas for MQTT, it has already equipped with the security option by using TLS. The communication between OCF gateway and OCF Servers utilizes standard DTLS protocol.

Table 1. Hardware Requirement for PoC

Criteria	Item
Software	Java 1.8, Maven, Spring boot v.2.1.6 IoTivity v1.3 Ubuntu 16.04 LTS
Hardware	PC: Processor Intel Core I7-8700 CPU@ 3.20 GHz Device: Raspberry Pi 3

III. Result and Discussion

```
: Found resource for the first time on server with ID: f5c4a650-eb6e-3827-e63c-e67892e6da46
:     URI of the resource: /temp
:     Host address of the resource: coap://[fe80::698c:fed4:f3d9:710d%25enp0s3]:41976
:     List of resource types:
:             oic.r.temperature
:     List of resource interfaces:
:             oic.if.baseline
:             oic.if.s
:     Host of resource:
:             coap://[fe80::698c:fed4:f3d9:710d%25enp0s3]:41976
:     List of resource endpoints:
:             coap://192.168.0.40:39823
:             coap://[fe80::698c:fed4:f3d9:710d%25enp0s3]:41976
:     List of resource connectivity types:
:             CT_ADAPTER_IP
:             CT_IP_USE_V4
```

Fig. 5. Device Discovery

Fig. 5, shows the OCF gateway broadcasts /oic/res for discovering the available OCF Servers. OCF Servers will then reply with their information on their resources.

```
=== OCF Endpoint Called ==
== 1. Getting JWT Token from OAuth 2.0 Authorization Server using Client Cridential ==
request : <[Accept:"application/json", Authorization:"Basic b2NmR2F0ZXdheTpvY2ZHYXRld2F5
url : http://192.168.0.17:9051/oauth/token?grant_type=client_credentials | method : POST
response : <200,
{"access_token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOlsib2NmUmVzb3VyY2VzIl0sIn
lYWQiLCJ3cm10ZSJdLCJleHAiOjE1NjQ0MjgxOTcsImp0aSI6ImNmMzJkYTE4LWZhMTgtNGUwNi1iYjBkLTAzNzk
ImNsaWVudF9pZCI6Im9jZkdhdGV3YXkifQ.3DQ9zyip9jGx1xp8DrbAK5RyLIzRxGUmxZsPhXBy5HQ","token_t
,"expires_in":24999,"scope":"read write","jti":"cf32da18-fa18-4e06-bb0d-03793f5a0068"},[
cache", Cache-Control:"no-store", X-Content-Type-Options:"nosniff", X-XSS-Protection:"1;
X-Frame-Options:"DENY", Content-Type:"application/json;charset=UTF-8", Transfer-Encoding
Date:"Mon, 29 Jul 2019 12:26:37 GMT"]>
```



Fig. 6. HTTP Flow on Agent/AE

Fig. 6 shows the process executed on the Agent to obtain resource data from OCF devices. The steps are obtaining a token from OAuth AAS, and then use the obtained token to access the OCF gateway. OCF gateway will then translate the UUID to the corresponding OCF Server's IP address and forward the request accordingly.

When OCF gateway discovers a new device, it sends Notify request to the corresponding device. As described in Fig. 7, OCF Server will then notify OCF gateway when there is a change of data. OCF gateway will consequently publish the obtained data to MQTT broker. Therefore, the subscriber will keep receiving updated data without requesting.



Fig. 7. MQTT Flow on OCF Gateway

Our implementation result is shown in Fig. 5-7. The presented figures are the realization of sequence diagrams on previous sections. The OCF gateway has already supported HTTP and MQTT protocols, which brings the advantage of connecting various heterogeneous IoT platforms. Additionally, we employ OAuth2.0 for securing the communication between platforms. However, our current OCF gateway only supports for asymmetric bridging, which enables other platforms to have the full privilege (e.g. CRUDN) to access the OCF device, but not vice versa.

IV. Conclusion and Future Work

In this paper, we have implemented an OCF gateway in order to bridge OCF to other IoT platforms such as oneM2M and IBM Watson IoT by adapting interworking proxy approach. Since the current gateway implementation only supports asymmetric bridging method, we will extend the functionality to symmetric bridging and support for other IoT platforms in the future. Additionally, we will compare our gateway implementation with other implementation results.

Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00097, Development of Security Chip and Real Time Control Protocol Security Technology for Smart Factory Network Infrastructure).

[참고문헌]

[1] IoT Analytics, "IoT Platform Comparison: How the 450 providers stack up - IoT Analytics," 2017. [Online]. Available:https://iot-analytics.com/iot-platform-comparison-how-providers-stack-up/. [Accessed: 29-Jul-2019].

[2] S. Cirani, M. Picone, P. Gonizzi, L. Veltri, and G. Ferrari, "IoT-OAS: An oauth-based authorization service architecture for secure services in IoT scenarios," IEEE Sens. J., vol. 15, no. 2, pp. 1224-1234, 2015.

[3] S. Soursos, I. P. Zarko, P. Zwickl, I. Gojmerac, G. Bianchi, and G. Carrozzo, "Towards the cross-domain interoperability of IoT platforms," EUCNC 2016 - Eur. Conf. Networks Commun., pp. 398-402, 2016.

[4] Paul Sawers, "Microsoft, Intel, Samsung, & others launch IoT standards group: Open Connectivity Foundation | VentureBeat," 2016. [Online]. Available: https://venturebeat.com/2016/02/19/microsoft-intel-samsung-others-launch-iot-standards-group-open-connectivity-foundation/. [Accessed: 29-Jul-2019].

[5] O. Connectivity and F. Ocf, "OCF Bridging specification," no. November, pp. 0-58, 2017.

[6] oneM2M Partners Type 1, "oneM2M TS-0024-V3.2.2: OCF Interworking," 2019.

IoT 기기 보안 기술 검증을 위한 사물 봇 공격 자동화 도구 연구

황송이, 김정녀*
과학기술연합대학원대학교 정보보호공학과, 한국전자통신연구원*

A Study on the Automated Attack Tool of IoT Bot for Verification about Security Technique of IoT device

Song Yi Hwang, Jeong Nyeo Kim*
Department of information Security Engineering, Graduate School, University of Science and Technology,
Electronic and Telecommunications Research Institute*

요 약

최근 들어 인터넷에 연결되는 IoT 기기의 수가 급증하면서, 이로 인한 잠재적인 IoT 보안 위협이 계속해서 증가하고 있지만 IoT에 대한 보안 의식은 아직도 현저히 부족하다. 특히, 저전력, 저성능 구조로 설계된 IoT 기기의 특성상 기존 컴퓨터에서 사용하던 보안 기술을 탑재하여 사용하기에는 매우 제한적이거나 어렵다. 이러한 이유로 IoT 환경에서는 잠재된 보안 위협이 기존 컴퓨터보다 훨씬 많이 존재한다. IoT 기기 보안 기술은 취약한 IoT 기기가 악용되는 것을 예방하고 차단하기 위해 IoT 기기 보안 기술은 IoT 보안 위협과 공격 양상을 고려하여 개발되어야 한다. 이렇듯, 새롭게 개발되는 IoT 기기 보안 기술을 실제 IoT 환경에 적용하기 전에 기술의 성능을 판단하거나 인지하지 못한 문제점을 발견하기 위해 기술 검증을 반드시 거쳐야 한다. 현재까지는 IoT 기기 보안 기술의 기능과 성능을 검증할 방안을 수동으로 구현하기 때문에, 이를 위한 검증 환경을 구축하기 위한 많은 시간과 노력이 소모된다. 따라서 본 논문에서는 새롭게 개발된 IoT 보안 기술을 검증하기 위한 사물 봇 공격 자동화 도구에 대해 제안한다.

I. 서론

최근 들어, 인터넷에 연결되는 많은 사물들이 증가하면서 연결된 IoT 기기의 잠재적인 보안 위협 또한 계속 증가하고 있다. 특히 IoT 기기를 대상으로 하는 분산 서비스 거부 공격(DDoS, 이하 DDoS)은 인터넷에 공개되어 있는 소스코드를 기반으로 해당 공격의 변형 공격들이 많이 생겨나고 있다.

그 중, Mirai(이하 미라이) 악성코드는 IoT 기기로 DDoS 공격을 일으켜 미국의 주요 웹사이트를 마비시켜서 등장하자 마자 엄청난 파괴력을 입증했다. 이러한 미라이 악성 코드의 소스코드가 인터넷에 공개되었기 때문에 IoT 기기를 위협하는 변형 미라이 악성코드가 현재까지도 계속해서 생겨나고 있다. 최근 IoT 기기를 악용한 DDoS 공격 사례가 급증하고 있는 만큼 IoT 기기를 보호할 보안 기능 및 기법이 필요하다.

현재 취약한 IoT 기기들을 위협하는 IoT 악성 코드에 방어하기 위해, 멀웨어에 감염되는 것을 예방하는 기술, 감염된 기기 혹은 멀웨어를 탐지하는 기술, 멀웨어의 전파를 막는 기술 등과 같이 다양한 IoT 기기 보안 기술이 개발되고 있다. 하지만 새로 개발된 IoT 보안 기술을 실제 IoT 환경에 적용하기 전, IoT 보안 기술의 기능과 성능을 평가하기 위한 검증 과정을 거쳐야 한다.

기존에는 개발된 IoT 기기 보안 기술을 검증하기 위해 테스트베드 환경을 구성하는데 필요한 프로그램을 일일이 내려 받거나 검증에 필요한 시험 환경을 자체적으로 구축해야 한다. 이렇듯, IoT 기기 보안 기술을 위한 검증용 도구가 없기 때문에 IoT 보안 기술의 기능과 성능을 검증할 때마다 테스트베드 환경을 구축하기 위해 많은 시간과 노력이 든다. 더군다나 IoT 기기 보안 기술을 검증할 땐 많은 수의 IoT 기기가 네트워크에 연결된 IoT 환경에서 검증을 해야 하기 때문에, 환경이 바뀔 때마다 일일이 수동으로 IoT 기기 보안 검증을 위한 테스트베드 환경을 구축해야 한다면 매우 번거로울 뿐만 아니라 비효율적이다.

따라서, IoT 기기 보안 기술 검증을 위한 테스트베드 환경을 일일이 수동으로 구축하는 비효율적인 작업을 제거하고 불필요하게 소모되던 시간과 노력을 줄이기 위해 실제 사물 봇과 동일하게 동작하는 환경을 구축하고 공격을 수행할 수 있는 자동화 도구가 필요하다. 따라서 본 논문에서는 IoT 기기 보안 기술의 검증을 위해 실제 사물 봇과 동일한 동작을 수행하는 사물 봇 공격 자동화 도구를 제안하고자 한다.

II. 요구사항

미라이는 처음 등장하자 마자 대규모 DDoS 공격을 일으켜 Netflix, CNN, GitHub, Reddit, Twitters 등 유명 웹 사이트에 접속 장애를 발생시킴으로써 엄청난 파괴력을 입증했다. 이 대규모 DDoS 공격에서 100,000가 넘는 미라이에 감염된 IoT 기기가 악용되었을 것으로 추정된다. 이토록 많은 IoT 기기가 DDoS 공격에 악용될 수밖에 없었던 이유는 그 당시 IoT 기기에 대한 보안 인식이나 기술이 아직 낮은 수준에 머물러 있었기 때문이다.

초기 미라이는 공장에서 설정한 디폴트 비밀번호

를 변경없이 그대로 사용하는 IoT 기기를 대상으로 감염시켰다. 대다수가 IoT 기기의 보안 위협에 크게 주의하지 않고 기존에 설정되어 있는 초기 비밀번호를 그대로 사용했기 때문이다. 하지만 현재는 IoT 기기에 대한 보안 인식과 보안 기술 수준이 높아져서 초기 미라이의 디폴트 비밀번호를 공격하는 익스플로잇 방식은 통하지 않을 가능성이 높다.

하지만 이런 대응에도 불구하고 IoT 기기는 아직도 위협에 노출되어 있다. 안나 센파이(Anna-senpai)라는 닉네임을 가진 해커가 미라이 악성 코드의 소스코드를 공개했기 때문이다[1]. 공격자들은 공개된 미라이 악성 코드의 소스코드를 이용해 계속해서 변형 미라이 악성 코드를 생성하고 있다. 그렇기 때문에 IoT 기기 보안 기술을 개발할 때는 IoT 기기의 기존에 존재하는 취약점은 물론 잠재적 보안 위협까지 고려해야 한다. 또한, IoT 기기의 보안 기술이 계속해서 새롭게 개발되는 만큼, 이러한 보안 기술을 실제 IoT 환경에 적용해도 되는지 판단하기 위한 기능과 성능 검증 기술 또한 구현되어져야 한다.

개발한 IoT 기기 보안 기술의 정확한 검증을 위해선 실제 사물 봇넷이 동작하는 환경과 동일하게 테스트베드 환경을 구축해야 한다. 따라서 여러 다양한 사물 봇 중 소스코드가 공개된 미라이 악성 코드를 선정했으며, 미라이 악성 코드의 소스코드로 실제 미라이 봇넷이 동작 과정과 동일한 공격 환경을 쉽게 구축할 수 있다.

III. 설계

사물 봇 공격 자동화 도구는 IoT 기기 보안 기술의 검증을 위한 공격 환경을 구축하는 도구이기 때문에, 정확한 기능과 성능 검증을 위해선 실제 미라이 악성 코드와 동일하게 동작해야 한다. 사물 봇 공격 자동화 도구는 공개된 미라이 악성 코드의 소스코드를 사용하기 때문에, 우선 미라이 악성 코드가 실행될 수 있는 환경을 구축하는 작업을 수행하면 된다.

그러나 사물 봇 공격 자동화 도구를 동작 시키기 전에 먼저 C&C 서버와 DNS 서버로 쓰일 PC, 그리고 미라이 악성 코드에 감염될 IoT 기기가 네트워크에 연결되어 있어야 한다. 기기들이 네트워크에 연결되어 있다는 전제하에 사물 봇 공격 자동화 도구는 C&C서버, DNS 서버, 초기 감염 미라이 봇의 환경을 설정한다.

1) 기본 화면

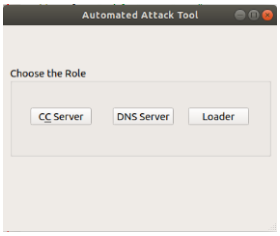


Fig. 1. 사물 봇 공격 자동화 도구 메인 화면

사물 봇 공격 자동화 도구를 실행시키면 가장 먼저 보이는 화면이다. 기기가 동작할 역할을 클릭하면 해당 역할을 수행하기 위한 환경 설정 화면으로 이동한다. 예를 들어, C&C 서버로 동작할 기기에서 사물 봇 공격 자동화 도구를 실행시킨 뒤, 기본 화면에서 'C&C 서버'라 적힌 버튼을 클릭하면 C&C 서버로 동작하도록 환경 설정 화면으로 이동한다.

2) C&C 서버 환경 설정

C&C 서버에서는 다른 기기들보다 설정해야 하는 작업이 많다. C&C 서버로 동작하기 위해 필요한 패키지들을 다운로드하고, C&C 서버를 사용하는 사용자와 감염된 미라이 봇을 관리할 DB를 구축해야 한다.

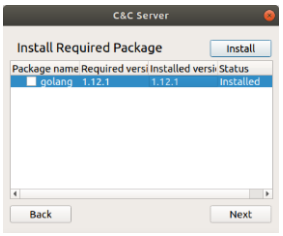


Fig. 2. C&C 서버 필수 패키지 설치 화면

이 화면에서 가장 먼저 수행할 작업은 'Install' 버튼을 클릭하여 C&C 서버에 필요한 패키지를 다운로드 받는 것이다. 설치해야 하는 패키지에 대한 버전 정보, 필수 요구 버전 정보, 설치 여부 정보를 나타내 원하는 패키지만 설치하여 패키지를 설치하는데 소요되는 시간을 줄인다.

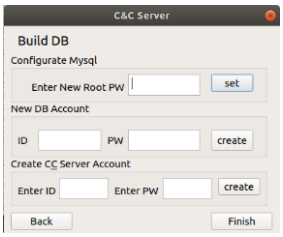


Fig. 3. C&C 서버 DB 구축 화면

다음으로 'next' 버튼을 클릭해 C&C 서버를 사용하는 사용자와 감염된 미라이 봇을 관리할 DB 구축 화면으로 이동한다. 이 화면에서는 C&C 서버에 접속하는 사용자 계정을 관리하는 테이블과 감염된 IoT 기기를 관리하는 테이블을 생성하는 작업을 수행한다.

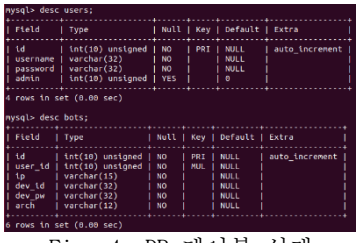


Fig. 4. DB 테이블 설계

위의 Fig.4에서의 테이블 설계를 바탕으로 DB를 구축하고 난 뒤, 미라이 봇에 공격 명령을 전달하는 C&C 서버에 접속하기 위해 필요한 사용자 계정

을 생성한다.

앞의 작업을 모두 수행하여 C&C 서버로 동작할 수 있는 환경으로 구축되면, ‘Execute’ 버튼을 클릭하여 미라이 봇을 제어하고 공격 명령을 내릴 수 있는 C&C 서버를 실행시킨다.

3) DNS 서버 환경 설정

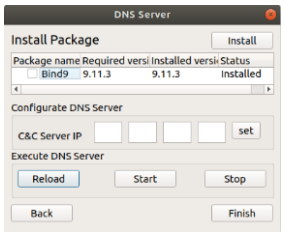


Fig. 5. DNS 서버 환경 설정 화면

DNS 서버로 동작할 기기에서는 DNS 서버로 동작하기 위한 BIND 패키지를 설치하는 작업과 미라이 봇이 C&C 서버에 접속할 수 있도록 환경을 설정하는 작업만 수행하면 된다. DNS 서버 환경 설정 화면에서도 C&C 서버 환경 설정 화면과 동일하게 패키지에 대한 버전 정보, 필수 요구 버전 정보, 설치 여부 정보를 나타내 패키지를 설치하는데 소요되는 시간을 단축시킨다.

4) 초기 감염 미라이 봇

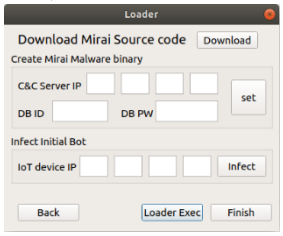


Fig. 6. IoT 기기 감염 화면

C&C 서버, DNS 서버와 달리 IoT 기기에서는 사물 봇 공격 자동화 도구를 실행시킬 수 없다. 그렇기 때문에, 미라이 악성 코드를 전파해 IoT 기기를 감염시키는 로더가 필요하다. 사물 봇 공격 자동화 도구는 로더 역할을 하는 기기가 미라이 악성 코드 바이너리를 취약한 IoT 기기에 전파할 수 있도록 환경을 설정한다.

우선 ‘Download’ 버튼을 클릭하여 미라이 악성 코드 소스코드를 로더로 동작할 기기에 다운로드 한다. 그 뒤, ‘Infect’ 버튼을 클릭하여 초기 감염 IoT 기기에 미라이 악성 코드를 전송한다. 미라이 악성 코드가 다 전송되면 해당 IoT 기기에 접속하여 미라이 악성 코드를 실행시킨다. 마지막으로 로더가 미라이 봇으로부터 스캔 결과를 받아 취약한 IoT 기기를 감염시킬 수 있도록 ‘Execute’ 버튼을 클릭하여 로더를 실행시킨다.

IV. 기능 검증 방법

Fig. 7은 사물 봇 공격 자동화 도구의 기능을 검증할 실제 환경이다. 해당 공격 환경에는 C&C 서버, DNS 서버, 로더, IoT 기기인 5개의 Raspberry Pi가 네트워크에 연결되어 동작 중이다.

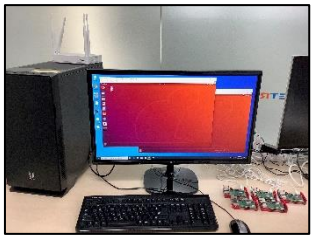


Fig. 7. 미라이 봇넷 실제 구조

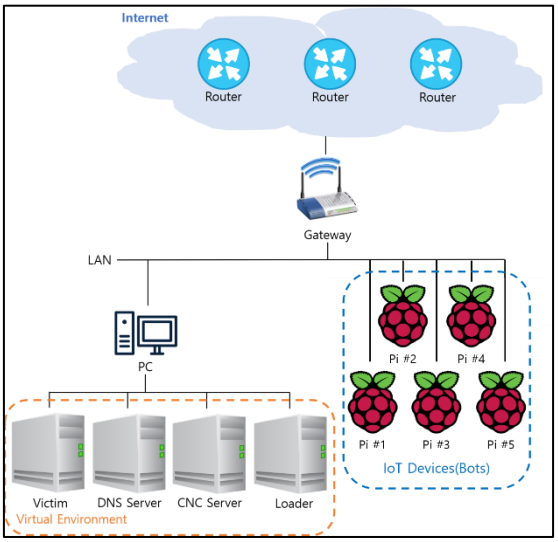


Fig. 8. 공격 환경 구조도

사물 봇 공격 자동화 도구로 구축한 공격 환경이 정상적으로 동작하는지 검증하기 위해 5개의 Raspberry Pi 중 하나의 기기만 미라이에 감염시킨 후 로더가 미라이 봇으로부터 받은 스캔 정보를 이용해 감염된 상태가 아니었던 Raspberry Pi를 감염시키는지, C&C 서버가 미라이 봇과 통신을 하여 제어하고 공격 명령을 전달하는지를 확인한다.



Fig. 9. 미라이 C&C 서버 동작 화면

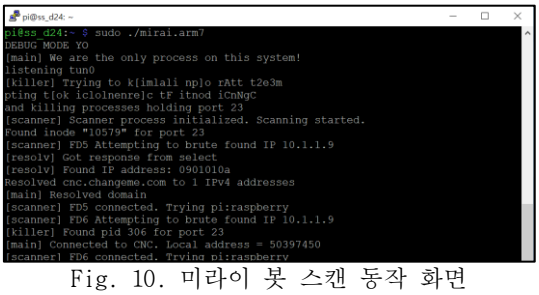


Fig. 10. 미라이 봇 스캔 동작 화면



Fig. 11. 로더 동작 화면

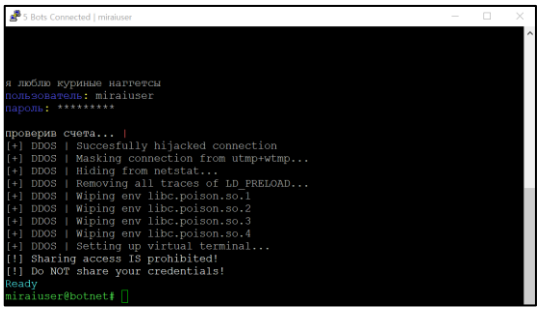


Fig. 12. C&C 서버 접속 화면

Fig. 9는 C&C 서버를 동작 시킨 화면이고, Fig. 10은 IoT 기기에서 미라이 악성 코드를 실행한 화면이다. 감염된 기기가 하나밖에 없기 때문에 C&C 서버 접속 화면 상단 바에 1 Bots Connected라고 감염된 IoT 기기의 숫자가 나타난다. 이 후, Fig. 11과 같이 취약한 IoT 기기를 감염시키는 로더를 동작 시키면 Fig. 12 상단 바에 보여지는 것처럼 감염된 IoT 기기의 수는 증가할 것이다. 해당 환경에서는 5개의 Raspberry Pi를 연결했기 때문에 5 Bots Connected라고 나타난다.

위와 같이 스캔, 감염 기능을 테스트했으면 실제로 미라이 봇이 C&C 서버로부터 공격 명령을 전달받아 공격을 수행하는지 확인해야 한다.

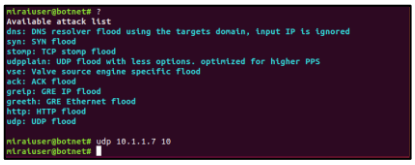


Fig. 13. 미라이 봇넷 공격 화면

위의 Fig. 13와 같이 동작중인 C&C 서버에 접속하여 '?' 명령어를 입력하여 사용할 수 있는 공격 벡터를 확인하고, '[공격 벡터] [타겟 PC IP] [Duration]' 형식으로 공격 명령을 입력하여 타겟 PC를 공격할 수 있다.

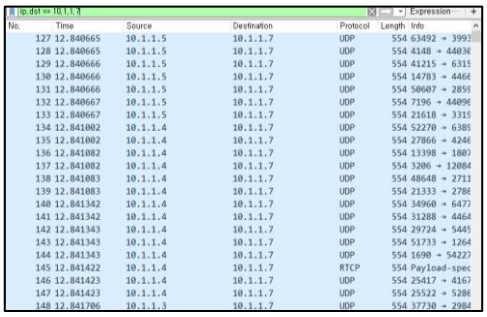


Fig. 14. 미라이 봇넷 공격 수행 화면

공격 명령을 입력한 뒤, 미라이 봇이 실제로 공격을 수행하는지 확인하기 위해 Fig. 14처럼 타겟 PC에서 네트워크 패킷을 확인해본다.

이렇듯, 사물 봇 공격 자동화 도구를 사용하여 공격 환경을 구축한 뒤, C&C 서버에 접속하여 공격 명령을 입력했을 때 Fig. 14와 같이 미라이 봇이 타겟 PC를 공격하는 결과가 나오는지 확인하면 된다. 미라이 봇이 타겟 PC를 공격하면 정상적으로 미라이 봇넷 공격 환경이 구축된 것이다.

V. 결론

기존에는 새롭게 개발된 IoT 기기 보안 기술의 기능과 성능을 검증해 볼 테스트베드 환경을 수동으로 구축했기 때문에 IoT 기기 보안 기술을 검증하는데 불필요하게 많은 시간과 노력을 소모했다. IoT 기기의 보안 기술을 검증하기 위한 공격 환경을 구축할 때마다 매번 수동으로 작업하는 것은 매우 비효율적이기 때문에, IoT 기기 보안 기술의 기능과 성능을 검증할 공격 환경을 자동으로 구축하고 실제 사물 봇처럼 동작하는 사물 봇 공격 자동화 도구가 필요하다.

본 논문에서 제안하는 사물 봇 공격 자동화 도구는 초기 미라이 악성 코드와 똑같이 동작하면서 자동화된 형태로 공격을 유발하여 새롭게 개발된 IoT 보안 기술의 기능과 성능을 검증하여 준다. 그러나 추후에는 미라이 악성 코드를 비롯한 많은 사물이 계속해서 진화하여 보다 강력한 변형 악성 코드를 생성하기 때문에, IoT 기기 보안 기술은 물론 사물 봇 공격 자동화 도구 또한 진화하는 변형 사물 봇에 따라 발전해야 한다. 따라서 향후 연구에서는 IoT 기기를 감염시키는 공격 벡터와 공격 기법을 더 연구하여, 다양한 공격 기법이 내재되고 공격 기능 또한 향상된 사물 봇 공격 자동화 도구를 연구하고자 한다.

[ACKNOWLEDGEMENT]

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥 센터의 지원을 받아 수행된 연구임 (No.2018-0-00231, (IoT 2세부) IoT 인프라 공격 확산 방어를 위한 상황 적응형 보안 자율제어 기술개발)

[참고문헌]

[1] "DDoS공격을 위한 IOT 악석코드 'Mirai'의 소스코드 공개", KISA 인터넷 보호나라 & KrCERT, 2016년 10월 21일 수정, 2019년 11월 11일 접속, https://www.krcert.or.kr/data/trendView.do?bulletin_writing_sequence=24654

[2] Meghan Carole Riegel. 2017. TRACKING MIRAI: AN IN-DEPTH ANALYSIS OF AN IOT BOTNET, Master's thesis, The Pennsylvania State University,

모바일 포렌식 도구를 통한 효과적인 iPhone 내 디지털 증거 수집 절차에 대한 연구

안재혁*, 손태식*

*아주대학교 정보통신대학원

A study of effective digital evidence collection procedures in iPhones through mobile forensics tools

Jaehyuk An*, Taeshik Shon*

*Graduate School of Information and Communication Technology, Ajou University.

요 약

최근 스마트폰이 급격히 대중화되며 범죄에 악용되는 사례가 급증하면서, 개인의 민감 정보와 활동 내역을 다수 포함하고 있는 스마트폰을 디지털 포렌식 관점에서 분석하는 연구의 중요성은 날이 갈수록 커지고 있다. 그러나 지금까지의 연구들은 오픈소스로 상대적으로 추출이 용이한 안드로이드 OS 기반 스마트폰들을 대상으로 데이터를 수집하는 경우가 대부분이었다. 이에 본 논문에서는 높은 보안성을 이유로 소비자들이 구매하는 iPhone을 대상으로, 디지털 포렌식 관점에서 증거로서의 가치가 높은 데이터를 우선적으로 수집하는 도구 및 절차를 제안하였다.

I. 서론

국내 모바일 시장에서 Apple 사의 iOS는 국내에서 약 21%의 점유율을 차지하고 있으며, 이는 안드로이드 OS의 점유율에 이은 2위다[1]. 또한 Apple 사는 개인정보보호와 보안성을 최우선으로 고려한 iPhone 설계를 표방하고 있으며, 이러한 Apple 사의 방침은 종종 수사기관의 iPhone 내 디지털 증거 수집을 어렵게 하고 있다. iPhone은 iOS 운영체제가 폐쇄적이며 날이 갈수록 다양하고 높은 수준의 암호화 수단(Face ID, 보안코드 등)을 제공하기에, 디지털 포렌식 분야에서 iPhone 기기 내 잠재적으로 디지털 증거가 될 수 있는 데이터를 수집하고 분석하는 일의 어려움과 중요성은 증가하고 있다.

지금까지 국내 iPhone 포렌식 연구는 스마트폰 내부 데이터에 직접 접근하여 데이터를 수집하는 Jailbreak 기법, iTunes 또는 iCloud와 동기화된 스마트폰 내 백업 데이터를 추출하는 기법 등 데이터 수집 기법 소개에 초점을 맞춘 연구들이 대부분이었다. 반면 잠재적으로 디지털 증거가 될 수 있는 수집된 방대한 데이터들을 효율적으로 수집하는 세부 절차에 대한 연구는 아직 미비하다. 문헌[2]는 iPhone을 대상으로 한 포렌식 수사 절차를 제안하였으나, 실제 포렌식 도구를 실험에 적용하여 효과성을 보여주지는 못했다는 한계를 지닌다. 또한 문헌[3]은 iTunes를 활용해 백업 데이터를 수집하고 iPhone을 대상으로 한 디지털 증거 분석 도구를 설계하여 실험에 적용하였으나, 수집된 데이터의 SQlite 포맷을 출력하는 기능만을 가지고 있다는 한계가 있다.

따라서 본 논문은 최신 iOS(ver. 13.1.3) iPhone에 상용 포렌식 도구를 적용하여 미디어 파일의 Metadata, 설치된 앱 및 Wi-fi 접속 내역, GPS 정보 등 포렌식 수사 관점에서 가치 있는 정보들을 우선적으로 수집하는 도구 및 세부 절차를 고찰하였다.

II. 분석 방법

2.1 분석 환경

본 연구에서 사용한 PC와 iPhone, 모바일 포렌식 도구는 Table 1과 같다.

Table 1. Experimental Circumstance

	사용기종	OS	적용도구
PC	Intel(R) Celeron(R) N4000 CPU	Windows 10 Home (64 bit)	Elcomsoft Mobile Forensic Bundle
iPhone	iPhone XR	iOS 13.1.3	

2.2 실험 절차

데이터 무결성 보장을 위해 iPhone 설정에서 Wi-fi, Bluetooth 등 무선 연결 네트워크를 차단하고 비행기 모드로 전환하였다. 본 연구에서는 입수한 iPhone의 6자리 보안코드를 획득한 상태에서 실험을 진행하였다. 실험 절차는 다음과 같다.

1. Elcomsoft iOS Forensic Toolkit 프로그램에 포함되어 있는 Hardware USB Key를 실험용 PC에 연결한다.

2. 입수한 iPhone을 Lightening Cable로 PC와 연결한다.

3. Toolkit.exe 프로그램을 실행한다.

4. 프로그램에서 “I” 명령어를 사용하여 iPhone의 정보를 확인한다. 이 정보는 생성된 “ideviceinfo.xml” 파일, “applications.xml”에 기록된다.



Fig. 1. Extracted informations of the iPhone

5. “M” 명령어를 입력하여 연결된 iPhone의 미디어 파일(이미지, 비디오, 음악 파일)과 관련된 메타데이터를 추출한다. 이때 추출되는 파일들은 Photodata 폴더 내 CPLAssets, Metadata, Thumbnails 폴더에 저장되며, iPhone의 백업 파일이 암호로 보호되어 있어도 Device의 Camera Roll에 직접 접근하여 추출 가능하다.

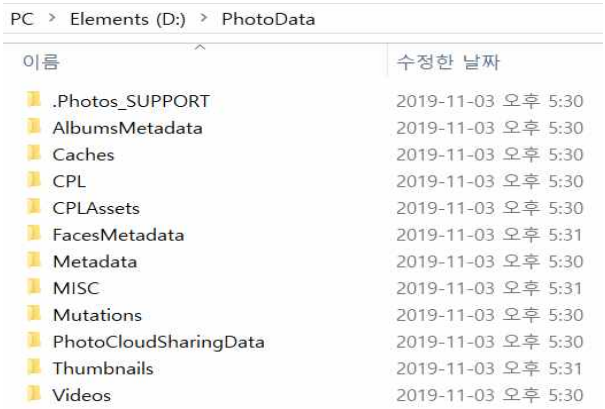


Fig. 2. Extracted Media files in folders

추출한 미디어 파일들은 HEIC, JPG, PNG, MOV 등의 확장자를 지니며, EXIF 정보가 내재되어 있다. 본 실험에서는 Kuso Exif Viewer 프로그램으로 추출한 이미지 파일의 EXIF 정보를 확인하며 사진의 GPS 정보, 촬영 및 수정 시간 등 사진에 대한 메타데이터를 얻을 수 있었다.

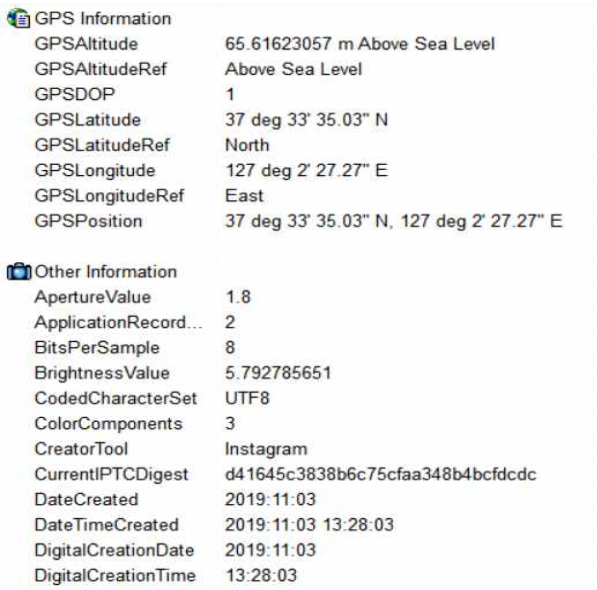


Fig. 3. Metadata information of an image file

6. “B” 명령어를 입력하여 iPhone이 연결된 컴퓨터에 기기 백업 파일을 생성한다. 이 때 백업 데이터는 status.plist, manifest.plist, info.plist, manifest.db 등에 기록되며, 백업 파일이 암호화 되어 있지 않다면 프로그램상 백업파일 암호가 “123”으로 설정된다.

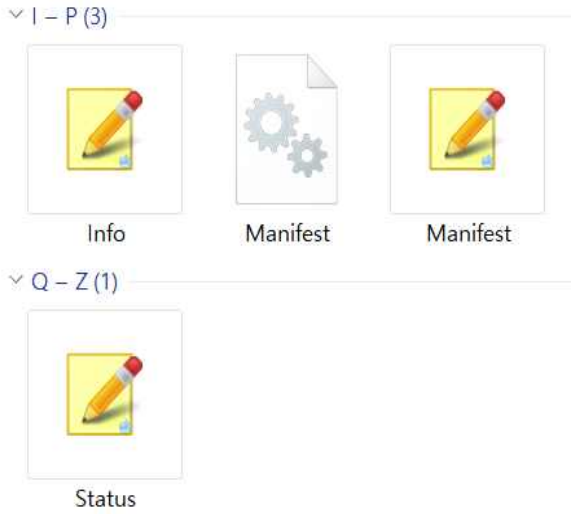


Fig. 4. Generating backup-file of the iPhone

7. Elcomsoft Phone Viewer 프로그램을 실행 후 생성된 iOS 백업파일 manifest.plist를 불러온다. 이 파일은 설치된 애플리케이션에 대한 정보, 연락처, 일정표 등을 포함한다. 이 백업파일을 상기 임의 설정된 암호를 입력해 복호화 한다.

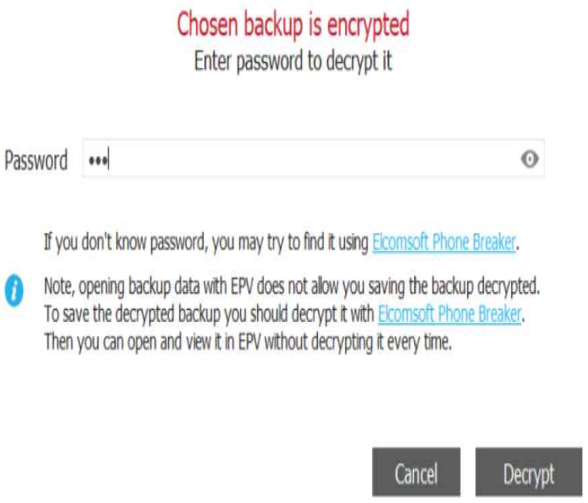


Fig. 5. Decrypting backup-file of the iPhone

8. 복호화가 완료되면 논리적으로 추출한 iPhone 백업 데이터가 프로그램을 통해 조사관이 분석하기 용이한 형태로 나타난다. 이 데이터는 사용자의 Wi-fi 연결, Web 연결, 위치 및 결제 정보 등을 포함하고 있다.

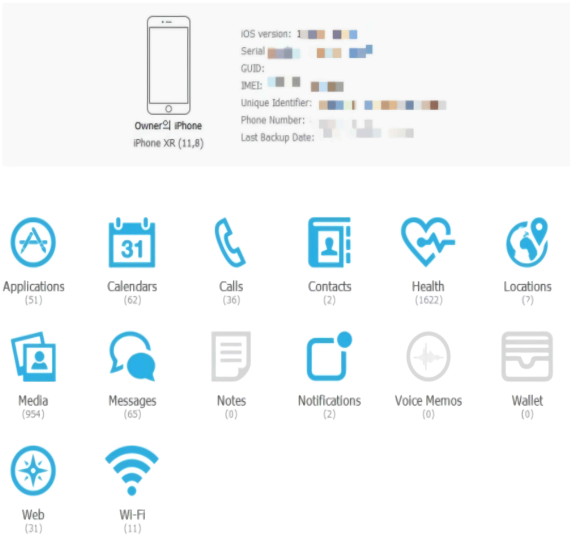


Fig. 6. Viewing backup-file of the iPhone

9. 기기의 iOS가 탈옥(Jailbreak)이 가능한 버전인지 확인한다. 탈옥이 적용된다면 toolkit.exe 프로그램에서 “D” 명령어를 통해 기기의 자동 잠금을 무력화시키고, “F” 명령어를 통해 애플리케이션 데이터, 키 체인 정보 등의 내부 데이터에 직접적으로 접근하여 물리적 추출이 가능하다[4]. 본 실험에 사용된 iPhone은 iOS가 가장 최신 버전으로 탈옥 적용이 불가능하다.



Fig. 7. Deploying unc0ver jailbreaking tool

위 실험 순서를 포함하는 본 연구의 절차 제안을 순서도로 나타내면 다음과 같다.

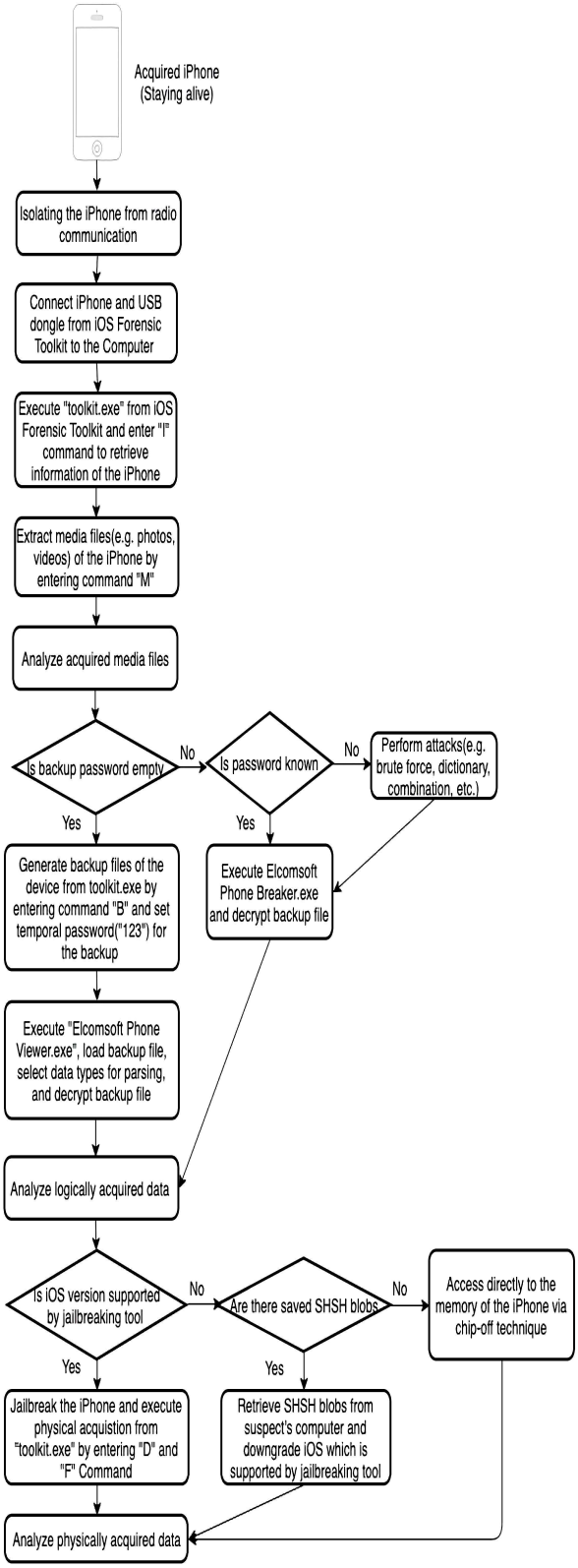


Fig. 8. iPhone data acquisition flowchart

III. 결론

본 논문에서는 상용 모바일 포렌식 도구를 사용하여 iPhone 내 디지털 증거 수집 시 포렌식 수사 관점에서 가치가 높은 미디어 파일을 먼저 추출하는 방안을 제시하였다. iPhone 내 미디어 파일과 관련 메타데이터는 사건 용의자의 인간 관계, 행동 분석, 이동 경로 확인 등에 이용되므로 가장 가치가 높은 데이터 중 하나라고 할 수 있다. 실험에서 사용한 Elcomsoft iOS Forensic Toolkit는 디바이스 백업 파일의 암호화 여부와 상관없이 미디어 파일을 우선적으로 추출할 수 있기에 유용하다. 이후 미디어 파일 외의 데이터 추출 시 디지털 증거로 사용될 수 있는 Wi-fi 접속 내역, 위치 및 결제 정보를 파악하기 용이하다는 장점이 있다.

iOS의 주기적인 업데이트로 인해 본 실험에 사용한 모바일 포렌식 도구 또한 지속적으로 그에 맞춰 업데이트되어야 하며, 제안한 절차를 더욱 효과적으로 수행하는 도구가 등장한다면 그것과 조합하여 모바일 포렌식 수사를 진행할 수 있다. 본 실험 또한 iPhone의 탈옥이 제한되어 디바이스 내 물리적 데이터 수집 절차까지 완벽히 수행하지는 못했다. 향후에는 수집된 미디어 파일들을 다각도로 자동 분석하고 복구하여 디지털 증거화하는 도구, 그리고 그 도구를 포렌식 수사에 활용하는 방안을 연구하고자 한다.

[참고문헌]

[1]http://www.koreanclick.com/insights/newsletter_view.html?code=topic&id=532&page=1

[2] 이방협. “아이폰 디바이스 환경에서 효과적인 디지털 포렌식 분석.” 공학석사학위 논문, 아주대학교, 2011.

[3] 김주영, 구본민, 이태림, 신상욱 (2010). 아이폰을 위한 디지털 증거 분석 도구 설계. 한국멀티미디어학회 학술발표논문집, 456-459

[4] 정진형,변근덕,이상진 (2011). iPhone의 SNS 데이터 수집 및 디지털 포렌식 분석 기법. 정보처리학회논문지 18-C권 제4호, 217-226

UI 은닉 기반의 안드로이드 Inter-Procedural Control Flow Graph 난독화 기법

심현석*, 윤홍선*, 정수환**

*숭실대학교 정보통신공학과 **숭실대학교 전자정보공학부

Android Inter-Procedural Control Flow Graph Obfuscation based on UI concealing method

Hyunseok Shim*, Hongsun Yoon*, Souhwan Jung**

*School of Information and Telecommunication Engineering, Soongsil University.
**School of Electronic Engineering, Soongsil University.

요 약

안드로이드 운영체제에서 Flow Analysis의 난독화를 위해서는 실행되지 않는 흐름의 코드를 생성하여 Flow Graph의 크기를 크게 만들어 분석이 어렵게 만들 수 있다. 이를 위해 논문에서는 aar 형태의 라이브러리를 구현하여 외부 라이브러리의 형태로 애플리케이션에 삽입이 가능하도록 하였다. 라이브러리는 더미 코드에서의 진입점에서부터 최대 4개의 child node를 가질 수 있도록 설계되었으며, child node의 클래스는 각 node마다 100개, 300개, 500개, 900개의 메서드를 가지고 있으므로 총 1,800개의 진입점으로 구성된다. 또한 진입점은 XML에서 총 150개의 뷰로 구성되며, 각각의 진입점은 *OnClickListener*와 같은 인터페이스를 통해 연결된다. 따라서 Inter-Procedural 기반의 Control Flow Graph를 생성하는 과정에서는 최대 2조 250억개의 추가적인 경우의 수를 가지게 된다. 이를 애플리케이션에 적용한 결과 Inter Procedural Control Flow Analysis 툴에서 평균 4,983개의 Edge와 1,980개의 Node가 추가 생성되었으며 평균 29.78%의 그래프 크기 증가율을 갖는다. 또한 APK를 분석 시에는 최대 평균 96.25MB의 오버헤드가 발생하였지만, 사용자의 ART 환경에서는 최대 평균 0.98MB의 실행 오버헤드만을 가지며 실행 가능한 것을 확인하였다.

I. Introduction

안드로이드 운영체제에서 개인정보 유출을 분석하는 방법은 정적분석과 동적분석 방법이 있으며, 이 중 정적분석 방법에서는 흐름 분석을 통한 방법이 주된 방법으로 제시되고 있다. 흐름 분석은 각 함수 호출을 비롯한 실행 흐름을 알아내기 위한 분석 방법으로, Control Flow Analysis나 Data Flow Analysis와 같이 여러 종류를 가지고 있다. 안드로이드 애플리케이션은 여러 개의 procedure를 가지므로 Control Flow Graph(CFG)를 생성할 때 이를 고려한 Inter-Procedural 기반 CFG (ICFG)를 생성하고 분석하는 데에 초점을 맞추고 있다. ICFG는 UI component에서의 이벤트나, *broadcastReceiver*와 같은 비동기 지점에서의 흐름을 분석하는 장점을 가지고 있지만, 현재 ICFG 분석 툴에서는 UI 컴포넌트에 대한 면밀한 분석이 이루어지지 않는다. 기존에 존재하는 툴에서는 XML의 분석을 통한 UI component의 분석이 이루어지지만 이는 UI component에 대한 존재 여부를 파악하기 위함이다. 따라서 기존의 툴에서는 inter-procedural 환경에서 entry point의 존재 여부만을 확인하며 해당 entry point의 진입 가능 여부는 파악이 불가능하다. 반면 UI component 상에서의 사용자 접근 가능 여부에 대한 확인 과정이 이루어지지 않는다면 정적 분석 환경에서는 모든 entry point로의 진입이 유효한 것으로 파악하게 된다. 우리는 이러한 점을 이용하여 ICFG에 대한 난독화가 가능한 것을 확인하였으며, 이를 구현하였다.

구현을 위한 기법은 UI component를 배치한 후 은닉하는 것으로, 일반 사용자는 실행이 불가능하지만 코드 상에서는 존재하는 흐름을 만드는 것이다. APK는 UI component를 통한 수많은 entry point를 가지게 되지만, 이러한 UI component는 크기가 0px이거나 화면 밖에 배치되므로 실행이 불가능하다. 이러한 흐름을 통해 APK는 실행 상에서 최소한의 오버헤드를 가지며 실행되지만, 분석가의 관점에서는 2차, 3차적인 분석이 불가능할 정도로 큰 그래프를 가지게 된다.

논문은 다음 장부터 배경지식을 설명하며, 3장에서 현재 흐름 분석의 문제점에 대해 설명할 것이다. 이후 4장에서 논문에서의 툴을 구현하는 방법에 대해 소개하며, 5장에서 툴에 대한 평가가 이루어진다. 논문은 마지막으로 6장에서 결론을 통해 끝맺는다.

II. Background

2.1. Android Application

안드로이드 애플리케이션은 classes.dex, res, lib, assets, META-INF, resources.arsc, 그리고 AndroidManifest.xml로 구성된 Android package, 즉 APK의 형태를 가지고 있다. 안드로이드의 정적 분석 과정에서는 APK를 디컴파일하여 내부의 리소스 및 API들을 추출하며, 주로 classes.dex와 AndroidManifest.xml의 정보를 활용한다[1].

안드로이드 애플리케이션은 뷰를 생성하기 위해 리소스 파일 중 하나인 XML 파일에 의존하는데, 이는 res/ 디렉토리의 layout/ 디렉토리에 위치한다. 각각의 뷰 구성요소는 뷰 내부의 리소스 배치를 위하여 이미지 리소스에 접근하며, 이는 res/ 디렉토리의 drawable-*/과 mipmap-*/ 경로에 포함되어 있다.

2.2. Control Flow Graph

Control Flow Graph(CFG)는 각 basic block으로 구성된 절차적 그래프로, 각 노드는 statement에 대응된다. CFG의 목적은 실행가능한 모든 경로를 알아내는 것으로, 모든 경로는 statement에 대응하는 노드와 실행 경로를 의미하는 edge로 구성된다. 그림 1과 같이 어떤 노드 A를 진입점으로 하는 CFG에서 서브그래프의 노드 B로의 흐름이 존재한다면 B는 A가 실행된 이후 특정 조건하에서 반드시 실행된다는 것을 의미한다.

안드로이드 운영체제에서 대부분의 CFG는 Interprocedural Control Flow Graph(ICFG)와 Callback Control Flow Graph(CCFG)로 나타난다. 고전적인 ICFG는 main procedure에서의 진입점에 의존하지만, CCFG에서는 사용자가 실행하는 환경에서의 이벤트와 이벤트 핸들러에 의해 나타난다. 안드로이드에서의 CFG 생성은 프레임워크 기반의 특징과, 이벤트에 의해 작동하는 특징으로 인해 ICFG 보다 CCFG가 더 적합하다. 반면, 안드로이드에서는 CFG를 시작하는 main procedure가 존재하지 않는데, 이런 문제를 해결하기 위해 Steven Arzt et al.은 **FlowDroid**[2]와 같은 오픈소스 툴을 개발하였다.

III. Related work

최근 ICFG 분석을 수행하는 수많은 툴들이 연구되어왔으며, 논문을 통해 공개되었다. 그러나 이러한 작업들은 진입점의 진입 가능여부를 판단하지 않으며, 최대한 많은 진입점을 찾는 것에 초점을 두고 있다. **FlowDroid**는 *dummyMainMethod*를 이용하여 여러 진입점을 가지는 문제와 비동기로 실행되는 코드 및 *callback*까지 분석하는 툴이다. **FlowDroid**에서는 inter-component 분석을 위해 XML에서의 뷰 컴포넌트를 분석하며, 어떤 뷰가 어떤 코드로 연결되었는지 파악한다. 따라서 각 *button* 혹은 *textView*와 같은 뷰에서 연결되는 진입점을 찾는 것은 가능하지만, 이러한 진입점이 실제로 user-level에서 진입 가능한지 확인하는 부분은 누락되어있다. **IccTA**[3]는 안드로이드 운영체제에서 inter-component 간의 흐름을 포함한 flow analysis를 수행하기 위한 툴로, lifecycle 메서드와 콜백 메서드를 분석하는 기능을 포함한다. **IccTA**에서는 *onClick*과 같은 뷰에서 전달되는 콜백 메서드의 흐름을 추적하여 분석하지만, 해당 뷰와 연결된 java 코드에서만 콜백 메서드를 분석하므로 뷰의 접근 가능 여부는 확인하지 않는다. **AmanDroid**[4]는 Inter/intra-component 간의 Data Flow Graph(DFG)와 CFG를 모두 지원하며, XML 리소스 기반의 분석 또한 가능하다. 반면 XML 리소스에서 특정 뷰의 접근 가능 여부를 파악하지 않으므로 접근 가능하지 않은 뷰에서는 false positive를 가진다.

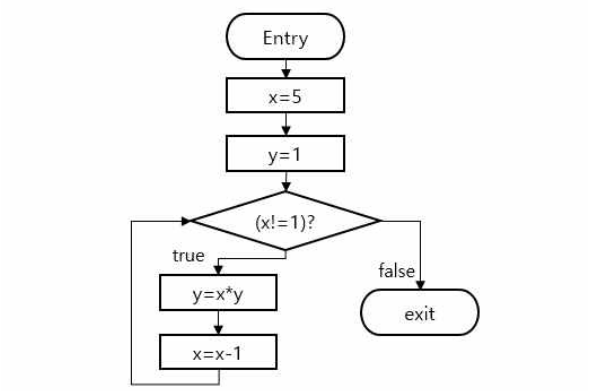


그림 1. CFG 예시. 각 노드는 basic block에 대응하며, 그래프는 하나의 entry point와 하나의 exit point를 가진다

논문에서 제안하는 툴은 이러한 문제를 해결하기 위해 구현되었으며, 다음 섹션부터 구현 방법에 대한 상세한 기술이 이루어진다.

IV. Implementation

4.1. Dummy View Insertion

본 논문에서는 dummy view를 삽입하기 위해 약 150개의 dummy view를 가지는 XML 코드와 java 코드 내에서 동적으로 뷰를 추가하는 기능을 구현하였다. Dummy view를 포함하는 XML의 경우, *focusable=false* 혹은 *cliackable=false* 등을 이용하여 뷰에 이벤트를 발생시키는 것이 불가능하게 하거나, 혹은 *visibility=gone*으로 정의하거나 width와 height가 0px로 정의하여 뷰에 대한 접근이 불가능하도록 숨겨놓았다. 또한 java 코드에서는 개발자가 지정한 complexity에 따라 XML의 뷰를 inflating하는 기능을 가지고 있으며, 만약 complexity가 XML에서 정의된 최대 개수인 150개를 초과한다면 동적으로 뷰를 추가한다. 뷰의 동적 추가 과정은 하나의 ViewGroup에 complexity에 대응하는 수만큼의 뷰를 추가하고, 이를 *visibility=gone*으로 설정하여 숨기는 방식으로 이루어진다.

4.2. Dummy Entry Point Insertion

이 섹션에서는 각 뷰에 대해 임의의 이벤트 리스너를 부착하여 랜덤으로 entry point를 생성하는 과정에 대해 소개한다. 각 뷰는 *AnonymousInterface*로 정의되는 custom interface를 가지고 있으며, 각각은 4단계로 이루어진 depth에 따라 임의의 그래프 흐름을 생성한다. 그림 2는 랜덤으로 생성된 dummy entry point와 해당 entry point에서 이어질 수 있는 가능한 흐름을 나타낸 것이다. 그림에서 각 depth에 따라 생성 가능한 흐름은 단계별로 100개, 300개, 500개, 900개로, 총 2조 250억 개의 경우의 수를 가지며, 이는 각 진입점으로부터 4단계의 depth를 가지는 서브그래프로 나타난다. 따라서 2조 250억 개의 경우의 수 만큼 임의로 생성된 진입점부터의 흐름에 대응할 수 있다.

각 진입점에서 연결되는 실행 흐름은 추상클래스로 구성된 메서드를 랜덤으로 호출하며 생성된다. 이를 위해 *onClickListener*, *onLongClickListener*, *onTouchListener*를 랜덤으로 반환하는 인터페이스인 AnonymousInterface로부터 4단계의 클래스 호출이 이루어진다. 그래프에서 노드에 해당되는 요소인 뷰와 dummy entry point를 각각 $v \in V$, $e \in E$ 로 정의하며, dummy method를 m∈M로 정의하였을 때 대응되는 서브그래프는 $subgraph(v) \subseteq e \rightarrow m_1 \rightarrow m_2 \rightarrow m_3 \rightarrow m_4$로 나타난다. 따라서 CFG를 생성하는 과정에서 각 노드는 진입점 혹은 dummy method를 의미한다. 이러한 ICFG의 전체 dummy flow 생성 과정에 대한 pseudo code는 그림 3와 같이 나타난다.

4.3. Library Extraction

본 논문에서 개발된 툴은 aar 형태의 라이브러리로 추출되어 개발자의 gradle 환경에서 import 및 implementetion이 가능하도록 한다. aar 라이브러리는 15이상 28이하의 SDK level을 타겟으로 하였으며, 각 리소스 및 레이아웃을 포함하고 있어 single line으로 사용이 가능하도록 하였다. 라이브러리는 구현 위치의 Activity나 Fragment, 혹은 공통으로 사용되는 Base Context 상에서 라이프사이클의 첫 번째 순서인 *onCreate*()에서 호출하여 사용한다.

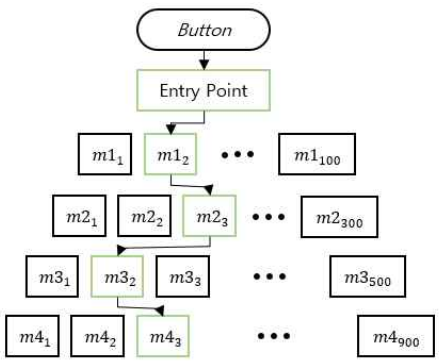


그림 2. 랜덤으로 생성된 dummy entry point와 가능한 흐름.

```
ALGORITHM 1: Creating Dummy Flow
    Input: C : complexity.
    Input: maxC : number of views in XML.
    Input: maxD : maximum depth of dummy flow.
    Output: dummyFlow : dummy flow of control flow graph.
    Variable: dummyViews : set of dummy views inflated by XML.
    Variable: dynamicDummyView : set of dummy views inflated dynamically.
    Variable: DEP : Dummy enty point.
1   procedure InflateDummyView(C)
2       IF C is smaller than given complexity C DO
3           dummyViews <- inflateXMLs(C);
4       ELSE
5           dummyViews <- inflateXMLs(maxC);
6           extraC <- subtract(C, maxC);
7           currentC <- extraC;
8           WHILE currentC is not reached extraC DO
9               dynamicDummyViews <- inflateDynamic(currentC);
10              re-compute current complexity;
11          setObfuscation(dynamicDummyViews);
12      setObfuscation(dummyViews);
13  procedure setObfuscation(v)
14      FOREACH view of set of views v DO
15          aInterface <- get anonymous interface;
16          DEP <- attachListener(view);
17  procedure setDummyFlow(DEPS)
18      FOREACH dummy entry point DEP of set of entry point DEPS DO
19          dummyFlow <- Ø;
20          SETcurrentDepth 0;
21          dummyFlow <- dummyFlow∪ setDummyFlow(DEP,currentDepth);
22  procedure setDummyFlow(DEP, depth)
23      dummyFlow<- Ø;
24      WHILE depth is smaller than maxD DO
25          dummyMethods <- dummyMethods∪ dummy methods of depth;
26          assign one of dummyMethods to DEP randomly;
27          re-compute currentDepth;
28      dummyFlow <- DEP with dummyMethods;
29      return dummyFlow;
```

그림 3. Entry Point 생성 과정부터 dummy flow 생성까지의 알고리즘

V. Evaluation

본 논문에서는 뷰의 개수에 따른 난독화 여부의 확인을 위해 Activity를 비롯한 레이아웃 개수를 변화시켜가며 성능 평가를 진행하였다. 또한 실제 서비스 중인 애플리케이션에 해당 라이브러리를 적용하여 적용 전후의 성능을 평가하여 표 1과 표 2에 나타내었다. 성능 평가 지표는 애플리케이션을 실행하는 환경에서의 메모리 오버헤드와 분석 시 메모리 오버헤드, 그리고 그래프 간의 차이로 구성된다.

그래프 간의 차이를 계산하기 위해 Graph Edit Distance(GED)를 사용하였다. 일반적으로 GED는 두 그래프 G, G′에 대해 G를 G′으로 만들기 위해 필요한 비용으로 나타낸다. 이 과정에서 추가되고 삭제된 에지를 각각 E_I, E_D로, G, G'의 모든 노드를 V, V'로 정의하면 $GED(G, G')$는 식 1과 같이 나타난다.

표 1. 메모리 소모량 기준 난독화 성능 평가
(max.mem.cons. = maximum memory consumption), (R* = Running, A* = Analyzing)

# of activities	Original		ICFGO applied		R* Overhead (MB)	A* Overhead (MB)
	R*max.mem.cons.(MB)	A*max.mem.cons.(MB)	R*max.mem.cons.(MB)	A*max.mem.cons.(MB)		
Single activity	3.04	1,357	3.98	1,486	0.94	129
2 activities	3.88	1,444	4.54	1,472	0.66	28
11 activities (ServiceApp)	10.06	1,577	11.10	1,672	1.04	95
15 activities (ServiceApp)	7.52	1,859	8.79	1,992	1.27	133

표 2. Graph Edit Distance 기준 난독화 성능 평가

# of activities	Original			ICFGO applied			GED(Ø, G) - GED(Ø, G')	Increase ratio
	Graph Nodes	Graph Edges	GED (Ø, G)	Graph Nodes	Graph Edges	GED (Ø, G)		
Single activity	2,336	9,242	11,578	4,284	13,092	17,376	5,798	50.08%
2 activities	2,311	9,149	11,460	4,245	13,026	17,271	5,811	50.71%
11 activities (ServiceApp)	8,450	45,494	53,944	10,428	49,813	60,241	6,297	11.67%
15 activities (ServiceApp)	10,319	139,472	149,791	12,377	147,358	159,735	9,944	6.64%

$$GED(G,\ G') = min(\sum(v_I)+\sum(v_D)+|E_I|+|E_D|) \quad (1)$$
$$(v_I \in \{V'-V\},\ v_D \in \{V-V'\})$$

식 1에 따라, 두 그래프에서의 생성을 위한 비용 차이는 GED(Ø, G)와 GED(G', Ø)의 차이로 나타낼 수 있다. 우리는 **FlowDroid**를 이용하여 테스트를 진행하였다. 결과에 의하면 모든 난독화 과정은 같은 complexity를 통해 수행되며, **FlowDroid**의 특성 상 XML 코드의 경우 단한번의 분석만을 진행하기 때문에 activity의 개수에는 무관한 특징을 보였다. 이로 인해 크게 다르지 않은 GED 차이를 가지므로, 레이아웃의 수가 적은 애플리케이션에서 더 높은 GED 증가율을 가지는 것을 확인하였다. 또한 실행 루틴이 같더라도 기기가 소모하는 메모리는 다르게 나타나므로, 정확한 최대 메모리 소모량을 측정하는데 어려움이 있다. 결과에서는 실행 중 오버헤드의 증가 비율은 높지만, 실제로 발생하는 오버헤드는 1.5MB 미만으로 적은 메모리 증가율을 보인다.

VI. Conclusion

본 논문에서는 관련 연구들에서 다루지 않은 뷰 및 entry point에 대한 접근 가능 여부 파악에 대한 취약점을 파악하여 이를 이용한 ICFG 난독화를 진행하였다. 논문에서는 한가지의 툴만을 이용하여 평가가 이루어졌지만, 좀 더 다양한 툴을 이용한 연구가 필요하다. 이 연구는 ICFG에 대한 난독화가 가능한 것 뿐 만이 아니라, 실제로 ICFG 생성 툴에서도 false positive를 줄이기 위한 XML 분석이 필요함을 나타낸다. 연구는 난독화에 대해 진행되었지만 향후에는 XML 분석과 이를 이용한 ICFG 생성의 고도화에 대한 연구가 이루어 질 것이다.

Acknowledgment

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-01853, 머신러닝 기반 지능형 악성코드 분석 통합 플랫폼) 또한 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발)

[참고문헌]

[1] N. Peiravian and X. Zhu, "Machine Learning for Android Malware Detection Using Permission and API Calls", 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, pp.300-305, 2013.

[2] A. Steven, et al. "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps.", Acm Sigplan Notices, vol.49, no.6 pp.259-269, 2014.

[3] L. Li, et al, "IccTA: Detecting Inter-Component Privacy Leaks in Android Apps", 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, pp.280-291, 2015

[4] F. Wei, et al, "Amandroid: A Precise and General Inter-component Data Flow Analysis Framework for Security Vetting of Android Apps", ACM Transactions on Privacy and Security, vol.21, no.3, pp.14:1-14:32, 2018

Android 어플리케이션에서 사용된 Third-party Library 탐지 기술 동향 분석*

이석원† , 이연준‡ , 오희국

*한양대학교 컴퓨터공학과

A Survey on Android Third-party Library Detection Approaches

SeokWon Lee† , Yeonjoon Lee‡ , Heekuck Oh‡

*Department of Computer Science and Engineering, Hanyang University.

요 약

현대 사회에서 모바일 기기의 사용량은 지속적으로 증가하고 있다. 특히 Android의 점유율은 모바일 시장을 압도하고 있다. 또한 Android 어플리케이션 시장의 규모는 매우 거대하며, 어플리케이션 개발을 위한 라이브러리의 종류 역시 다양하게 존재하고 있다. 특히 Third-party 라이브러리를 이용한 어플리케이션의 개발이 대다수를 이루고 있다. 그러나 취약하거나 유해한 Third-party 라이브러리를 포함한 어플리케이션의 확산은 새로운 보안 문제를 야기하게 되었다. 이와 같은 문제를 해결하고자 Third-party 라이브러리 탐지 기술에 대한 많은 연구들이 진행되었다. 본 논문에서는 Third-party 라이브러리 탐지에 대한 기존 연구의 동향과 사용된 여러 기법들을 분석하고자 한다.

I. 서론

현대 사회에서 모바일 기기의 사용량은 지속적으로 증가하고 있다. 그에 따라 모바일 어플리케이션 시장 규모의 성장 또한 급격한 상승세를 보이고 있다. 그 중에서도 모바일 기기에 사용되는 Android의 점유율 76.67%[1]에 달할 만큼 모바일 시장을 전체를 압도하고 있다. Android 어플리케이션 시장의 규모 또한 매우 거대하며, Google Play에 등록된 어플리케이션의 수는 280만개[2]에 이른다. 이에 따라 어플리케이션 개발을 위한 라이브러리의 종류 역시 다양하게 존재한다.

그 중에서도 Third-party Library (TPL)는 광고, 위치 정보, SNS 연동 등과 같은 다양한 기능들을 제공하며 어플리케이션 개발의 편의성을 증진시켰다. 이에 따라 어플리케이션 개발의 생산성을 크게 향상되었으며, 지금까지 개발된 어플리케이션은 상당수의 TPL을 포함하고 있다. 하지만 취약하거나 유해한 TPL을 포함한 어플리케이션의 확산은 다양한 보안 문제를 발생시키는 원인이 되었다. 따라서 이를 식별하는 것은 중요한 문제로 대두되었고 리패키징 탐지, 취약성 식별, 악성코드 분석 등과 같은 많은 연구들이 지속적으로 진행되어 왔다.

본 논문에서는 TPL 탐지에 사용된 원리와 기법 및 최근까지 진행된 클러스터링 기반의 연구에 대해 분석하고자 한다. 이후 논문의 구성은 다음과 같다. 2장에서는 TPL에 관한 배경지식을 소개한다. 3장에서는 TPL 탐지에 사용된 원리를 소개한다. 4장에서는 클러스터링 탐지 기법과 관련된 연구를 소개한다. 5장에서는 결론을 맺으며 논문을 마무리한다.

* “이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A2C2003045).”

† 주 저자 : sevenshards@hanyang.ac.kr

‡ 교신저자 : yeonjoonlee@hanyang.ac.kr

II.배경 지식

2.1 Third-Party Library (TPL)

TPL은 개발자가 편리하게 자신의 어플리케이션에 포함시킬 수 있는 일련의 기능 (사진 촬영, SSL 연결 설정, 광고, 위치 제공, SNS 등)을 제공하는 비휘발성 리소스(서브루틴, 클래스 또는 데이터 포함)이다. 다시 말하자면 잘 정의된 API 세트로, 이를 통해 누구나 제공하는 기능을 쉽게 이용할 수 있다. 이와 같은 편의성으로 인해 오늘날 대부분의 어플리케이션은 TPL을 사용하여 개발되었다. 인기 있는 라이브러리의 예로는 OkHttp, GSON, Crashlyticse 등이 있다.

2.2 Third-party 라이브러리로 인한 문제점

앞에서 말한 바와 같이 TPL은 어플리케이션 개발 생산성 향상에 큰 기여를 했다. 하지만, 이로 인해 발생하는 문제점도 존재한다.

첫 번째는 취약점 또는 악성코드를 포함한 TPL이 확산되는 것이다. 일부 라이브러리는 개발자가 공식 웹 사이트를 통해 배포하므로 유지관리가 잘 되어있지만 대부분의 라이브러리는 Github나 온라인 포럼 등을 통한 개방형 플랫폼을 통해 배포되고 있다. 또한, 어플리케이션 개발 과정에서 대다수의 개발자는 라이브러리의 취약점이나 악성코드를 내포하고 있는지에 대한 검증을 하지 않고 그대로 사용하는 경향이 크다. 따라서 개발자도 모르는 사이에 결함이 있는 TPL을 포함한 어플리케이션을 배포할 가능성이 높다. 이외에도 악의적인 목적의 사용자에 의해 악성코드를 포함하는 라이브러리를 포함한 어플리케이션을 리패키징을 통해 배포할 수도 있다.

두 번째는 어플리케이션 분석을 어렵게 만드는 것이다. 예를 들어, 어플리케이션 복제 여부를 탐지할 때는 라이브러리 코드를 배제해야 한다. 라이브러리를 포함한 어플리케이션을 탐지하려고 하면 분석에 불필요한 라이브러리의 코드가 추가됨으로 인해 탐지 체계의 정확도가 왜곡된다. 또한, 라이브러리를 포함한 어플리케이션을 정적 분석하게 될 경우 추가된 라이브러리의 코드에 의해 분석에 필요한 시간이 늘어나게 된다.

마지막으로는 TPL이 최신화가 되어도 기존 어플리케이션에서는 이를 잘 반영하지 않는 점이다. 대부분의 어플리케이션에서는 TPL을 사용하고 있지만 보안 수정 사항을 적용한 TPL의 최신 버전이 있음에도 이를 반영하는 비율은 높지 않으며 과거의 버전을 사용하는 경우가 상당수를 차지한다. 그로 인해 개발자가 기존의 어플리케이션에 최신화된 TPL을 적용하지 않은 상태로 어플리케이션을 스토어에 노출시킴에 따라 보안 취약점이 지속적으로 드러나게 된다.

2.3 난독화 (Obfuscation)

Android 어플리케이션에서의 코드 난독화는 라이브러리 탐지를 어렵게 만드는 요소 중 하나다. Android 어플리케이션에서 지원하는 난독화 도구는 ProGuard, Allatori, DashO, 및 Legu 등이 있으며, 그 중에서도 ProGuard는 Android Studio라는 IDE에 기본적으로 탑재되어 있기 때문에 가장 많이 사용된다. Android에서의 난독화는 크게 이름 기반 난독화, 흐름 제어 난독화, 문자열 암호화, Dead code 제거의 네 가지 유형으로 나뉜다. 이름 기반 난독화는 패키지, 클래스, 메소드 및 필드의 이름을 바꾸는 것이며 흐름 제어 난독화는 프로그램의 제어 흐름을 바꾸는 것이다. 문자열 암호화는 문자열을 암호화하는 것이며, Dead code 제거는 데이터 흐름 그래프(DFG)를 통해 사용하지 않는 코드를 찾아서 제거하여 최종 어플리케이션 파일의 크기를 줄인다.

또 다른 난독화 기법은 패키지 및 클래스 병합으로 패키지 병합은 바로 아래에 파일이 있는 모든 패키지를 하나의 공통 상위 패키지에 배치하는 것이며, 클래스 병합은 모든 클래스 파일을 공유 패키지에 배치하는 것을 말한다. 이는 패키지 관계 분석을 어렵게 만든다.

III. TPL 탐지 원리

이와 같은 문제점들로 인해 TPL을 식별하는

것은 Android 보안에서 중요한 부분이 되었으며, TPL을 탐지하기 위한 다양한 연구들이 진행되어 왔다. 여기서는 지금까지 탐지 기법들에 사용된 원리를 소개하고자 한다.

3.1 White-list 기반 탐지

초기의 TPL 탐지는 White-list를 기반으로 하였으며 White-list에 등록된 라이브러리의 이름과 어플리케이션의 패키지 이름 또는 전체 라이브러리의 해시 값의 비교를 통해 TPL을 탐지하는 방식이다. 이 기법을 사용한 대표적인 연구의 예로는 Centroid[3]와 AdRisk[4]가 있다.

3.2 White-list 기반 탐지의 한계

그러나 White-list 기반의 탐지는 다음과 같은 문제가 존재했다. 첫 번째는 White-list는 항상 최신화된 상태를 유지해야 할 필요가 있다는 것이다. 그러나 White-list는 수동 분석에 의해 생성되며 현재 Android 어플리케이션의 수는 수백만 개에 이른다. 이를 수동으로 분석하는 데에는 수많은 시간이 소요된다. 두 번째는 패키지의 이름에 의존한 탐지 기법이기 때문에 이름 기반의 난독화로 인해 패키지의 이름이 난독화될 경우에는 White-list에서 탐지를 할 수 없으므로 정확성이 현저히 떨어진다. 이와 같은 두 가지의 문제점은 White-list 기반 탐지의 한계를 명확히 드러냈다.

3.4 클러스터링 기반 탐지

앞서 말한 바와 같이 White-list 기반의 탐지에는 한계가 존재했다. 또한 난독화로 인한 TPL 탐지 정확성의 저하는 해결해야 할 중요한 과제가 되었다. 이를 해결하기 위해 클러스터링 기반으로 한 탐지 기법이 고안되었다. 이 기법은 코드가 난독화되어도 안정적이며 변경되지 않는 코드의 feature를 유지하는 “stable code feature”를 선택하고 이를 추출하여 클러스터링한 후 유사도를 비교하는 방식으로 라이브러리를 탐지한다. 비교 방식은 크게 두 가지로, 완전 비교와 부분 비교 방식이 있다. 완전 비교는 feature의 집합들이 정확하게 일치하는지 비교하는 것이며, 부분 비교는 유사한 두 기능 집합의 유사성을 통해 점수를 매기는 방식이다. 클러스터링을 통한 탐지 기법이 제안된 이후 지금까지 진행된 연구의 대다수가 클러스터링을 기반으로 진행되었다.

IV. 클러스터링 기반 관련 연구

클러스터링 기법을 사용한 초기의 연구는 WuKong[5]과 LibRadar[6]가 있으며, 해당 연구에서는 API 호출의 빈도를 stable code feature로 사용하였다. 두 연구는 완전 비교를 통해 Third-party 라이브러리를 탐지하였으며 다른 점은 LibRadar는 해싱 기능을 도입하여 stable code feature를 경량화한 것이다. LibRadar 이후의 연구는 보다 정확한 난독화 처리와 정확성에 중점을 두었으며, 다음과 같은 연구들이 진행되었다.

LibD[7]는 API의 호출 빈도가 아닌 복구된 TPL의 인스턴스 feature를 해싱한 값을 완전 비교를 통해 탐지한다. 이는 LibRadar보다 높은 정확성을 보였으나 Runtime에서는 큰 차이가 없었다. LibSift[8]는 feature를 사용하지 않고 패키지 의존성 그래프를 통해 어플리케이션을 각각의 모듈로 분해하여 TPL을 탐지하는 방식이다. 하지만 패키지 및 클래스 병합에 의한 난독화에 취약하다는 단점이 있다. LibScout[9]는 TPL의 원본에서 class profile을 추출하여 fuzzy descriptor를 해싱한 값을 feature로 사용한다. 또한 부분 비교를 통해 탐지하는 방식으로 이전의 연구들과 비교하였을 때 많은 난독화 기법들을 처리 가능하며 한정적으로나마 dead code 제거를 해결한 것이 큰 특징이다. LibPecker[10]는 LibScout의 장점을 토대로 class profile이 아닌 라이브러리 내부의 클래스 종속성을 활용한다. 이를 토대로 완전한 class signature를 구성하여 fuzzy class matching을 통해 라이브러리를 탐지하는 방식이다. 이 연구는 가중치 접근 방식을 통해 dead code 제거에 대한 복원력을 향상시킨 것이 특징이다.

가장 최근의 연구는 LIBBANDAID[11]로, 개발자들이 어플리케이션을 업데이트할 때, 라이브러리를 최신화하지 않는 경향이 있으며, 최신

화를 하지 않을 경우 이전 버전에 존재하는 취약점이 그대로 남게 된다. 해당 연구에서는 TPL 업데이트를 자동으로 생성하여 어플리케이션에서는 코드를 수정할 필요가 없고, 특히 로컬 및 원격에서의 호출을 통해 다른 구성 요소와 라이브러리의 상호작용에 영향을 미치지 않는 것이 특징이다.

V. 결론

본 논문을 통해 Android 어플리케이션에 사용되는 TPL에 대해 알아보았다. 또한 TPL로 인해 생기는 문제점에 대해 알아보았으며, 이를 탐지하기 위한 기법들에 대해 초기의 연구부터 지금까지 진행된 연구에 대해 분석하였다. 향후 연구에서는 탐지만이 아닌 TPL 업데이트 지연에 의한 취약점 노출과 같은 다양한 관점으로의 연구에 대해 분석할 예정이며, 더 나아가 새로운 방향을 제안하는 연구를 하고자 한다.

[참고문헌]

[1]https://gs.statcounter.com/os-market-share/mobile/worldwide#monthly-201810-201910

[2]https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

[3] Chen, Kai, Peng Liu, and Yingjun Zhang. "Achieving accuracy and scalability simultaneously in detecting application clones on android markets." Proceedings of the 36th International Conference on Software Engineering. ACM, 2014.

[4] Grace, Michael C., et al. "Unsafe exposure analysis of mobile in-app advertisements." Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks. ACM, 2012.

[5] Wang, Haoyu, et al. "Wukong: A scalable and accurate two-phase approach to android app clone detection." Proceedings of the 2015 International Symposium on Software Testing and Analysis. ACM, 2015.

[6] Ma, Ziang, et al. "LibRadar: fast and accurate detection of Third-party libraries in Android apps." Proceedings of the 38th international conference on software engineering companion. ACM, 2016.

[7] Li, Menghao, et al. "LibD: scalable and precise third-party library detection in android markets." 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE). IEEE, 2017.

[8] Soh, Charlie, et al. "LibSift: Automated detection of third-party libraries in Android applications." 2016 23rd Asia-Pacific Software Engineering Conference (APSEC). IEEE, 2016.

[9] Backes, Michael, Sven Bugiel, and Erik Derr. "Reliable third-party library detection in android and its security applications." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.

[10] Zhang, Yuan, et al. "Detecting third-party libraries in Android applications with high precision and recall." 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2018.

[11] Duan, Yue, et al. "Automatic Generation of Non-intrusive Updates for Third-Party Libraries in Android Applications." 22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019). 2019.

안드로이드 어플리케이션 레벨에서 호출 가능한 보안 컨테이너 환경 구현 기법

윤재현*, 차오녹투*, 응웬휘황*, 정수환**

*숭실대학교 정보통신공학과 **숭실대학교 전자정보공학부

Implementation of Secure Container Environment Callable from Android Application Level

Jaehyeon Yoon*, Ngoc-Tu Chau*, Hoang Nguyen Huy*, Souhwan Jung**

*School of Information and Telecommunication Engineering, Soongsil University.
**School of Electronic Engineering, Soongsil University.

요 약

보안 컨테이너는 가상 머신의 보안성과 컨테이너 기술의 퍼포먼스를 모두 충족시켜 줄 수 있는 새로운 가상 황경 구축 기술로서 현재 다양한 클라우드 시스템에서 활용되고 있다. 하지만 안드로이드 환경은 앱 샌드박싱, 기존 리눅스와 다른 실행 코드 및 사용자에게 제한된 권한 등의 이유로 인해 보안 컨테이너 및 가상환경 등에 대한 연구가 제대로 이루어지지 않고 있었다. 안드로이드 어플리케이션 레벨에서 가상머신을 구현하기 위한 기존의 방법은 루트권한을 획득하고 QEMU 바이너리를 이용하는 방식이었으며, 이는 매우 큰 오버헤드를 초래하였다. 본 연구에서는 기존 가상머신보다 빠르고, 별도의 커널을 이용하여 기존 컨테이너 기술보다 안전한 보안 컨테이너 구현 기법 및 안드로이드 어플리케이션 레벨에서의 호출 방법을 제안한다. 제안된 기법은 기존 보안 컨테이너 기술과 비교하여 빠른 퍼포먼스를 보여주며, 루트 권한 없이 어플리케이션 레벨에서 호출 가능하여 안드로이드 프레임워크를 침범하지 않고 구현될 수 있다는 장점을 가진다. 또한 추후 연구될 수 있는 안드로이드 환경에서의 컨테이너 서비스 활용 방안의 기반 연구가 될 수 있을 것으로 예상한다.

I. 서론

컨테이너 기술은 가상머신과 유사하지만 격리 기술을 간소화 하여 호스트 운영체제와 커널을 공유할 수 있다. 이로 인해 기존의 가상머신보다 빠르고 가벼운 기술로서 많은 관심을 받고 있다. 현재 컨테이너 구현 도구인 도커와 컨테이너 관리 도구인 쿠버네티스 등의 기술은 클라우드 환경에서 주목받고 있는 기술로서 많은 기업에 의해 활용되고 있다. 컨테이너가 호스트 운영체제와 커널을 공유한다는 것은 많은 오버헤드를 줄일 수 있는 장점이지만 다양한 취약점을 제공할 수도 있다. 호스트 커널에 취약점이 있는 경우 컨테이너는 내부에서 해당 취약점을 이용하여 권한 상승을 이룰 수 있으며, 이는 컨테이너의 제어권만이 아닌 호스트 운영체제에 영향을 미칠 수 있다[1]. 또한 호스트 운영체제에서 컨테이너는 하나의 프로세스로 구동이 된다. 컨테이너 내부에서 기밀성이 필요한 작업을 수행하는 경우, 호스트 운영체제로부터의 접근을 제한할 수 있는 방법이 요구된다. 가상머신의 경우 컨테이너 기술보다는 안전한 환경을 구현할 수 있지만 무겁고 오버헤드가 크다는 단점을 가진다.

이러한 문제점들을 해결하기 위하여 경량화된 가상머신을 이용한 보안 컨테이너의 개념이 등장 하였다. 보안 컨테이너는 경량화된 가상머신을 이용하여 격리환경을 구축하고 가상머신의 안전성을 상속받을 수 있다. 또한 가상머신 내부의 운영체제를 간소화하고 사용할 라이브러리와 어플리케이션만을 실행하여 오버헤드를 줄일 수 있다.

하지만 아직 안드로이드 환경에서 사용자 어플리케이션 레벨에서 직접 활용할 수 있는 컨테이너 환경을 제공하는 프레임워크는 없는 실정이다. 본 논문에서는 안드로이드 환경에서 경량화된 가상머신을 기반으로 보안 컨테이너를 구현하여 앱이 활용할 수 있도록 하는 방법에 대하여 설명한다.

본 논문에서는 2장에서 보안 컨테이너에 대한 배경을 설명하고, 3장에서 관련된 연구들을 다룬다. 4장에서 경량 가상머신 기반의 보안 컨테이너 구현 및 실행 방안에 대하여 설명하며, 5장에서는 기존 가상머신들과의 성능 및 기능성 평가를 수행하고, 마지막 6장에서 결론을 내린다.

II. 배경

2.1. 컨테이너 기술과 가상머신의 차이

컨테이너 기술과 가상머신의 차이점은 격리되는 레벨과 제어 기술에 있다. 컨테이너 기술의 경우 리눅스에서 사용하고 있는 Namespaces와 Cgroups을 이용하여 호스트 환경과 컨테이너 환경을 격리 시킨다. Namespace는 컨테이너를 위해 호스트와 별개의 사용자(USER), 프로세스관리(PID), 네트워크(NET), 마운트(NS), 프로세스 간 통신(IPC) 등의 시스템을 구현하여 준다. 또한 리눅스에서 제공하는 Cgroups을 이용하여 cpu와 메모리를 할당하여 격리시켜 줄 수 있다. 위 기술들은 리눅스 커널에서 제공되는 격리 기술로서 리눅스 환경에서만 사용할 수 있다.

일반적인 가상머신의 경우 가상화된 CPU, 가상화된 메모리 등의 다양한 디바이스를 에뮬레이션하고 그 위에 새로운 커널을 올려 하나의 컴퓨트 시스템을 구축할 수 있다. QEMU 등의 하이퍼바이저는 호스트 CPU와 다른 아키텍처를 사용하는 CPU도 구현이 가능하며, 하이퍼바이저에서는 상이한 아키텍처간의 명령어 변환을 위한 서비스도 제공한다. 하지만 모든 시스템을 가상화하여 실행시키는 방법은 많은 오버헤드를 초래하기 때문에 리눅스에서는 KVM(Kernel-based Virtual Machine) 기능을 커널에 포함시켜 제공한다. 가상머신을 실행시킬 때 KVM을 사용하는 경우 호스트 커널과 공유할 수 있는 일부 기능들을 활용할 수 있으며, 모든 시스템에 대한 가상화 작업을 하지 않아도 된다. 또한 호스트와 다른 커널을 사용함으로 호스트에서는 가상머신에게 자원만 할당하여 제공하며, 가상머신 내부에서 동작하는 작업에 대한 접근이 어렵게 된다.

2.2. 경량 가상머신기반 보안 컨테이너

앞서 설명한 컨테이너의 속도와 가상 머신의 보안성을 모두 제공하기 위하여 보안 컨테이너(Secure Container)의 개념이 제안되었다. Intel에서 개발된 Clear Container는 kvmtool과 QEMU-Lite를 기반으로 컨테이너를 위한 환경을 구축하고 해당 환경 내부에서 어플리케이션을 실행 시키는 방법을 사용하였다. 이후 등장한 Kata Container는 Intel의 Clear Container와 HYPER.SH의 runV를 통합하여 가장 유명한 보안 컨테이너 기술을 제공하고 있다. Kata Container의 경우 QEMU-Lite를 이용하여 보안컨테이너 환경을 구축하고 내부에서 docker 이미지를 활용한 서비스를 수행할 수 있다. 초기에는 Clear Container와 같이 Intel 계열의 CPU만을 지원하였으나 2018년 9월부터 ARM을 위한 Kata Container 개발이 진행되고 있다. [표 1]은 기존의 컨테이너 기술, 가상머신 과 경량가상머신 기반의 보안컨테이너의 특징을 비교하여 보여준다.

III. 관련 연구

많은 연구들이 가상머신의 경량화와 보안 컨테이너에 대하여 다루고 있으며, 논문을 통하여 공개되었다.

Roberto Morabito등[2]은 보안 컨테이너와 기존 컨테이너 기술의 성능 비교를 위한 연구를 수행하였다. 해당 연구에 따르면 클라우드 환경을 위해 개발된 OSv는 KVM 기능을 이용하여 운영체제레벨의 가상화를 수행하였고, 기존의 컨테이너 기술인 LXC와 Docker에 비하여 CPU, 디스크 IO, 네트워크 IO에 대한 성능 평가를 수행하였다. 수행된 결과에 따르면 CPU성능은 기존 컨테이너 기술 대비 동일하거나 빠른 성능을 보여주었으나, 디스크 IO와 네트워크 IO의 성능은 훨씬 떨어지는 것으로 나타났다.

Wenzhi Chen 등[3]은 안드로이드 디바이스를 위한 경량 가상화 기법에 대한 연구를 진행하였다. 하지만 해당 연구에서는 디바이스에서 2개의 안드로이드 운영체제를 올리기 위한 기법으로 실제로는 호스트와 동일한 커널을 활용하였고, namespace와 cgroup을 이용하여 2개 이상의 가상화된 안드로이드 운영체제를 위한 컨테이너 기술을 개발하였다.

Uri Kanonov 등[4]은 안드로이드에서 보안서비스를 제공하기 위한 knox에 대하여 연구하였다. knox의 주 목적은 신뢰실행환경의 하드웨어 백엔드로부터 상속받은 Root of Trust를 이용하여 안드로이드 어플리케이션들에게 보안 서비스를 제공하는 것이다. 이를 위해 knox 앱들은 기존 사용자앱과 동일하게 Zygote로부터 파생되어 생성되며, 커널 레벨 접근 제

[표 1] 하이퍼바이저 기반 가상머신, 컨테이너, 보안 컨테이너 기술적 특징 비교

	관련 대표 기술	특징	오버헤드	보안성
Hypervisor	QEMU, KVM 등	• 가상 하드웨어, 새로운 커널 및 운영체제 이용 • 비교적 많은 오버헤드와 뛰어난 보안성 보유	높음	높음
Container	Docker, lxc 등	• 호스트 OS와 동일한 커널 사용하여 커널 취약점에 의해 보안 훼손 가능성 존재 • Cgroups, Namespace를 이용한 프로세스 격리로 낮은 오버헤드 제공	매우 낮음	보통
Secure Container	Clear Container, Kata Container 등	• 경량 하이퍼바이저와 커널을 이용하여 가상환경 구현 • 가상환경 내부에서 Container 기술 이용 • 적은 오버헤드와 높은 보안성을 가짐	낮음	높음

어 기능인 SEAndroid, 루팅 및 시스템 변조를 막기 위한 eFuse 및 신뢰실행환경 기술들이 사용되고 있다.

기존 진행된 연구들은 보안 컨테이너의 성능을 증명하고, 안드로이드 환경에서 컨테이너를 활용하는 방법 등을 제안하고 있다. 하지만 안드로이드 환경을 위한 보안 컨테이너는 아직 연구되지 않고 있는 실정이다. 본 논문에서는 안드로이드 환경에서 어플리케이션에게 보안 컨테이너를 제공하기 위한 시스템 구현을 위한 방안들을 설명한다.

IV. 구현 방법

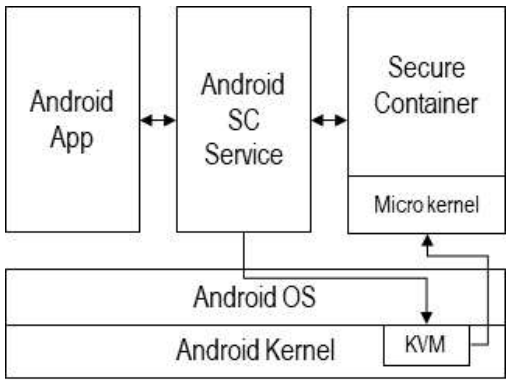
본 논문에서는 안드로이드 프레임워크에서 보안 컨테이너를 구현하기 위하여 ARM을 위한 경량 가상머신 툴, 경량 커널 및 루트파일시스템 경량화 작업, 안드로이드 시스템 서비스를 구현하였다. 구현은 안드로이드 레퍼런스 디바이스인 HIkey 960을 기준으로 개발되었다.

4.1. ARM 기반 경량 가상머신 도구

안드로이드에서 사용하는 커널은 리눅스 커널로부터 파생되어 개발되었으며, 리눅스 커널에서 제공하는 기능들을 활용할 수 있다. 경량 가상머신 환경을 구현하기 위하여 Kata Container와 Clear Container에서 활용한 QEMU-Lite와 kvmtool에 대하여 연구를 진행하였다. QEMU-Lite의 경우 다양한 환경 및 기능 지원을 위해 QEMU-Lite를 사용하지만 클라우드 서버와 비교하여 낮은 성능을 가진 모바일 디바이스 환경을 위하여 kvmtool을 활용하였다. 리눅스의 커널 모듈인 KVM을 안드로이드 커널에 적용하여 가상화 환경을 위한 vcpu 및 vmem 등을 지원하였으며, ARM 환경에서 제공하는 HYP mode를 통해 하드웨어 레벨에서 각 VM을 위한 시스템 버스 가상화 및 CPU 인터럽트 가상화 기능 등을 적용하였다[5]. 안드로이드 커널에 설정된 KVM을 활용하기 위해 lkvm 바이너리를 사용하였다. lkvm 소스코드를 받아 ARM 아키텍처를 위해 크로스 컴파일하고 동적 라이브러리들을 모두 통합하여 정적 라이브러리로 함께 컴파일을 수행하였다.

4.2. 커널 및 루트파일시스템 경량화

보안 컨테이너에서 사용될 커널로서 Zircon kernel을 활용하였다. Zircon kernel은 임베디드 디바이스를 위한 Google Fuchsia OS를 위해 개발된 커널로서 기본적인 스케쥴링 및 시스템콜들을 제공해주고 있다. 보안 컨테이너를 위해 사용될 루트파일시스템으로서 busybox 기반의 minimum root file system을 구현하였다. 해당 루트 파일 시스템은 리눅스 운영체제에서 활용하는 데몬이나 추가 서비스들이 없으며, 기본적인 명령어만을 제공하는 busybox를 제공한다. 이후 docker 컨테이너 이미지처럼 base root file system을 기반으로 컨테이너 안에서 실행될 어플리케이션과 필요 라이브러리들을 통합하여 컨테이너를 실행 시킬 수 있도록 하였다.



[그림 1] 안드로이드 환경에서 구현된 보안 컨테이너 실행 기법 아키텍처

4.3. 컨테이너 실행 서비스

안드로이드 프레임워크는 안드로이드 어플리케이션이 리눅스 커널에서 제공하는 기능을 직접적으로 사용할 수 있도록 허용하지 않는다. 안드로이드의 모든 어플리케이션들은 샌드박싱 되어있으며, 각 프로세스간의 통신은 안드로이드에서 제공하는 프로세스간 통신 디바이스인 Binder를 활용하여야 한다. 안드로이드 앱이 Binder를 통하여 컨테이너를 실행시키기 위해서는 특정 서비스를 Binder에 등록시킬 필요가 있다.

기존의 리눅스 운영체제와 다른 안드로이드 환경을 고려하여 안드로이드 프레임워크에 포함될 수 있는 시스템 서비스를 추가적으로 구현하였다. 구현된 서비스는 보안 컨테이너를 만들기 위해 ARM기반 경량 가상머신 환경 구축 도구에 대한 접근권한을 가지며, 안드로이드 어플리케이션의 요청에 따라 보안 컨테이너를 생성 및 제거할 수 있다. 또한 생성된 컨테이너와 연결된 콘솔을 각 어플리케이션들에게 제공해주는 프록시 역할도 수행하도록 개발되었다.

V. 평가

[그림 1]에서 볼 수 있듯이 구현된 안드로이드 프레임워크에서 안드로이드 어플리케이션은 Binder에 등록된 보안 컨테이너 실행 서비스(4.3에서 설명, Android SC Service)에 등록된 API를 이용하여 보안 컨테이너 실행 요청을 전송한다. 안드로이드 보안 컨테이너 서비스는 경량 가상머신 도구(4.1에서 설명,)를 이용하여 보안 컨테이너를 실행하며, 실행된 보안 컨테이너는 micro kernel과 minimum root file system(4.2에서 설명)을 가진다. 안드로이드 어플리케이션은 서비스 호출시 파라메터를 이용하여 특정 실행 바이너리를 보안 컨테이너에 탑재할 수 있으며, 탑재된 바이너리의 실행결과 및 과정은 안드로이드 보안 컨테이너 서비스를 통하여 어플리케이션에게 전달된다.

본 논문에서는 구현된 보안 컨테이너 기술과 Kata Container 등의 기존 기술들과 비교하여 환경 구축에 걸리는 소요시간 및 기능 특징에 대한 평가를 수행

[표 2] sha1sum에 대한 구현된 안드로이드용 보안컨테이너와 Kata Container 성능 비교

	소요 시간 (10회 평균)	실험 환경 및 특징
안드로이드용 보안 컨테이너	2167 ms	• Hikey 960 디바이스 환경 • 안드로이드 버전 9.0 • 안드로이드 어플리케이션에서 busybox기반 보안 컨테이너 구축 및 sha1sum 명령을 통한 해시 값 계산 수행
Kata Container	2362 ms	• Intel i7-6700, 16G RAM 데스크탑 PC 환경 • Ubuntu 16.04 • 쉘스크립트를 이용하여 busybox기반 docker 이미지 구축 및 sha1sum 명령을 통한 해시 값 계산 수행 • docker 설정을 통하여 kata-container를 이용한 환경 설정 • 다양한 docker 이미지 실행 가능
Native	3 ms	• Intel i7-6700, 16G RAM 데스크탑 PC 환경 (Ubuntu 16.04) • 리눅스의 기본 sha1sum 명령을 통해 산출된 평균 소요시간

하였다. 안드로이드 환경을 위한 보안 컨테이너 기술이 아직 없는 관계로 성능 평가를 위한 테스트 환경은 서로 다른 환경에서 소요시간을 산출하였다.

구현된 보안 컨테이너의 경우 Hikey960 디바이스 (4*Cortex A73 + 4*Cortex A53 Big.Little CPU architecture, 4GB LPDDR4 DRAM, 32GB UFS flash, Android 9.0) 환경에서 안드로이드 어플리케이션 레벨에서 측정 되었다. 기존 보안 컨테이너 기술인 Kata container의 경우 Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, 16GB RAM, Ubuntu 16.04 환경에서 에서 수행되었다. 테스트에 사용된 이미지는 busybox 이미지를 활용하였으며 각 환경에서 sha1sum 값을 수행하는 동작에 대하여 소요시간을 산출하였다. 소요시간은 어플리케이션 레벨에서 호출 이전 타임스탬프를 기록하고 호출 이후 타임스탬프를 기록하여 계산하였다. 각 환경에서 10회에 대한 반복수행을 통하여 산출된 소요시간의 평균값과 특징은 [표 2] 와 같다. Native의 경우 리눅스 환경에서 보안 컨테이너 없이 동일한 작업을 수행한 경우 걸리는 소요시간을 나타내었으며, 매우 짧은 작업을 사용하여 각 보안 컨테이너가 가지는 오버헤드를 산출 할 수 있다.

VI. 결론

본 논문에서는 기존에 다루지 않았던 안드로이드 환경에서 실행 가능한 보안 컨테이너를 구현하고 기존 보안 컨테이너와 성능 및 특징적 측면에서 비교하였다. 구현된 보안 컨테이너는 기존 보안 컨테이너 기술들과 비교하여 심플한 컨셉으로 안드로이드 프레임워크에서 컨테이너를 실행시키는 방법에 초점을 맞추었다. 하지만 비교적 기존 보안 컨테이너보다 비교적 빠른 성능을 보여준다. 안드로이드 어플리케이션 레벨에서 활용 가능한 보안 컨테이너에 대한 연구는 이후 docker 등의 컨테이너 환경과 통합하여 어플리케이션에게 다양한 서비스를 제공해 줄 수 있을 것으로 기대 된다. 또한 본 연구를 기반으로 안드로이드 보안 서비스를 제공하기 위한 프레임워크에 대한 연구가 이루어 질 것이다.

Acknowledgment

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00477, 가상화된 신뢰실행환경을 이용한 안드로이드 보안 프레임워크 기술 개발)

[참고문헌]

[1] CVE Details, "CVE-2019-5736", https://www.cvedetails.com/cve/CVE-2019-5736

[2] Morabito Roberto, Jimmy Kjällman, and Miika Komu, "Hypervisors vs. lightweight virtualization: a performance comparison," In 2015 IEEE International Conference on Cloud Engineering, pp. 386-393, March, 2015

[3] Wenzhi Chen, Lei Xu, Guoxi Li, and Yang Xiang, "A lightweight virtualization solution for android devices," IEEE Transactions on Computers vol.64, no.10, pp.2741-2751, October, 2015

[4] Kanonov Uri and Avishai Wool, "Secure containers in Android: the Samsung KNOX case study," Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices, pp.3-12, October, 2016.

[5] Dall Christoffer and Jason Nieh, "KVM/ARM: the design and implementation of the linux ARM hypervisor," ACM SIGARCH Computer Architecture News, Vol. 42, No. 1, pp.333-348, March, 2014

새로운 인증 패러다임 오라 (ORA: one round authentication) - 보안 USB를 활용한 단말 대 단말 1 라운드 인증 프로토콜 설계 및 구현✝

강병민[1], 안홍규[1,2], 변진욱[1]

평택대학교 정보통신학과[1]

㈜인젠트 ECM 사업본부 ECM 서비스팀[2]

A new proposal of new authentication paradigm, ORA : Design and Implementation of End-to-End One Round Authentication protocol with Secure USBs

Byeong Min Kang[1], Hong Kyu Ahn[1,2], Jin Wook Byun[1]

Department of Information and Communication, Pyeongtaek University[1]

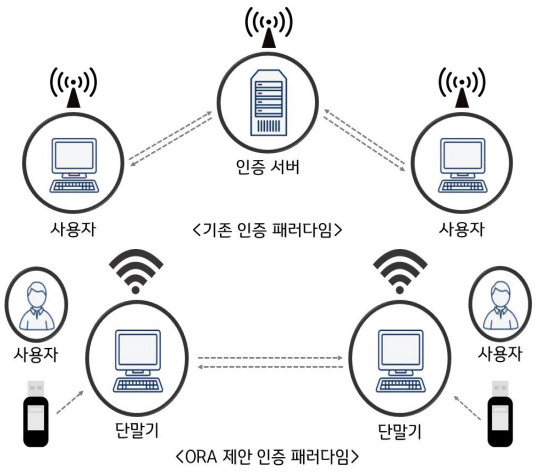
Inzent Co., Ltd. ECM Business Headquarters, ECM Service Team[2]

요 약

본 논문에서는 사용자와 사용자들이 각각 보유한 단말기(태블릿, 스마트폰)와 보안 USB를 활용하여 상호 인증할 수 있는 안전하고 실용적인 ORA 프로토콜을 제안한다. 가장 많이 활용되는 두 가지 환경 (안드로이드와 자바 플랫폼 환경)에서 ORA 프로토콜을 각각 구현하여 실용성 및 편리함을 보인다. 제안된 프로토콜은 단말 대 단말의 통신이 빈번히 발생하는 IoT환경에서 사용자들을 상호인증하고 안전하게 데이터를 전송하는데 널리 활용될 수 있다.

I. 서론

사물인터넷(Internet of Things, IoT) 환경에서 단말 대 단말(device-to-device, end-to-end) 간의 안전한 보안 채널 형성은 매우 중요한 보안 이슈로 간주되고 있다. 단말 대 단말 간 환경이란 사용자들이 IoT 환경에서 본인들의 단말기들을(예: 태블릿, 스마트폰, 노트북) 통해 데이터를 상호 주고받을 수 있는 환경을 의미한다. [그림 1]처럼 사용자들은 본인의 고유 단말기들을 통해 다른 사용자들을 인증할 수 있으며 메시지를 안전하게 전달한다. 일반적인 네트워크 환경에서의 인증은 단말과 서버와의 인증이 주를 이루지만 IoT 환경은 수많은 단말기 및 센서들을 가정하므로 단말과 단말 간의 인증이 훨씬 빈번히 발생할 수밖에 없는 구조이다. 하지만 이러한 필요성에도 불구하고 현재 최신 인증 트렌드는 아직도 서비스를 제공하는 클라우드 서버, 웹 서버가 사용자를 직접 인증하는 패러다임을 고수하고 있다.



[그림 1] 기존 인증 패러다임 및 ORA 제안 인증 패러다임 비교

일례로, 최신 FIDO(fast identity online) 기술도 사용자의 생체정보 등을 활용하여 서버로

✝ 이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. 2017R1D1A1B03032424)

부터 인증을 받는 구조이다 [1]. 또한, 최근에 구글의 보안키를 활용한 인증 기술도 사용자가 자신의 보안 USB를 활용하여 구글 계정에(구글 서버) 접근하여 안전한 인증 작업 후 구글 서비스를 받도록 하는 구조이다 [2].

본 논문에서는 IoT 환경에 더욱 적합한 단말 대 단말 인증 구조를 사용자들 간에 어떻게 효율적으로 새롭게 설계하고 쉽게 구현할 수 있는가를 연구한다. 이러한 목표를 위해 다음의 두 가지 환경적 시나리오를 가정한다. 첫째, 사용자 A, B는 상호 인증하기 위해서 소유하고 있는 단말기 및 보안 USB를 활용한다. 보안 USB란 사용자의 생체정보 (지문, 홍채)를 통해서만 접근 가능한 USB를 의미한다. 가령, 본 논문에서는 사용자의 홍채 인식을 통한 인증 기능이 탑재된 경량 USB를 활용하였다 [3]. 즉, 사용자들 간의 다중인증정보(패스워드, 생체정보, USB)를 바탕으로 상호인증을 효율적으로 유도하는 방법을 제안한다. 둘째, 사용자가 상호인증을 수행하기에 앞서 한번은 서로 만나서 인증정보를 단말기를 통해 주고받는 초기화 과정을 거친다. 이는 전통적인 인증방법에서도 사용자 등록을 동일하게 요구한다.

본 연구에서는 홍채 인식 보안 USB를 활용한 단말 대 단말 간의 상호인증 프로토콜을 제시하고 안전성 및 효율성에 대해 분석한다. 또한, 제안된 방법에 대해 안드로이드와 일반 JDK 환경에서 각각 구현하여 실용 가능성 및 효율성을 검증하였다.

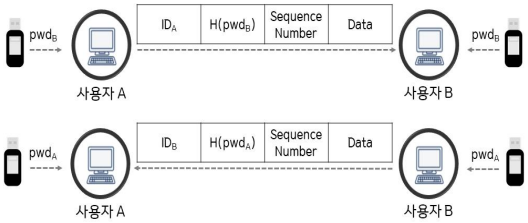
기호	설 명
A, B	프로토콜 참여자 (사용자) A, B
ID_A, ID_B	사용자 A, B의 ID
pwd_{AB}, pwd_{BA}	사용자 A, B의 쌍별 패스워드
SK_A, PK_A, SK_B, PK_B	사용자 A, B의 개인키 공개 키
$PE(k,m)$	공개키 k로 m을 공개키 암호화
M_A, M_B	사용자 A, B가 전달하려는 메시지
$r(\cdot)$	RSA 암호화
$H(\cdot)$	해시함수 연산
$USB_A(\cdot), USB_B(\cdot)$	사용자 A, B의 보안 USB
Seq	Sequence Number
$\|\|$	concatenate 연산

[표 1] 프로토콜에 사용되는 기호 정의

II. ORA 프로토콜 제안 및 분석

2.1 ORA 프로토콜 설명

[사전정보 등록단계] ORA는 두 사용자 A, B의 보안 USB_A, USB_B의 지원을 각각 받아 사전 인증정보 등록단계를 거친다. 제안된 ORA는 두 사용자 A, B가 물리적으로 직접 접촉한 뒤 각 사용자의 ID 및 일회용 패스워드, 그리고 공개키를 각각 상대편의 USB에 저장한다. 패스워드는 해시 처리하여 저장된다.



[그림 2] ORA 프로토콜 동작 과정

A	setup	B
	$ID_A,\ pwd_{AB},\ PK_A$ →	
		$H(pwd_{AB})$
	← $ID_B,\ pwd_{BA},\ PK_B$	
$H(pwd_{BA})$		

[표 2] ORA 프로토콜 사전정보 등록단계

A	authentication	B
	$PE(PK_B, ID_A\|\|H(pwd_{BA})\|\|Seq\|\|M_A)$ →	
		ID_A
		$H(pwd_{BA})$
		M_A
	← $PE(PK_A, ID_B\|\|H(pwd_{AB})\|\|Seq\|\|M_B)$	
ID_B		
$H(pwd_{AB})$		
M_B		

[표 3] ORA 프로토콜 인증단계 교환 과정

[인증단계] 제안된 ORA는 상호인증을 위해 각자 한 번의 통신을 요구한다.

- A는 본인의 ID_A와 USB_A에 저장된 사용자 B의 쌍별 패스워드 해시값 $H(pwd_{BA})$, 그리고 연속번호 Seq를 선택한다. 그리고 M_A와 결합하여 $ID_A||H(pwd_{BA})||Seq||M_A$를 생성한

다. 이를 B의 공개키 PK_B로 암호화하여 B에게 전달한다.

- B는 암호화 값을 전달받은 후 SK_B를 통해 복호화하여 $ID_A \| H(pwd_{BA}) \| Seq \| M_A$를 얻는다. 그리고 ID_A와 연속번호가 정당한지 확인, 본인의 패스워드 해시값 $H(pwd_{BA})$과 복호화로부터 얻은 $H(pwd_{BA})$이 일치하는지 비교하여 메시지 인증을 수행한다. 성공적이면, A가 보내고자 하는 메시지 M_A를 확인한다. 실패하면 프로토콜을 종료한다. 사용자 B는 $ID_B \| H(pwd_{AB}) \| Seq \| M_B$를 같은 방법으로 생성하고 A의 공개키 PK_A로 암호화하여 A에게 전달한다. 암호문을 전달받은 A는 자신의 개인 키로 암호문을 복호화하여 $ID_B \| H(pwd_{AB}) \| Seq \| M_B$를 얻는다. ID_A와 연속번호가 정당한지 확인하고, 본인의 패스워드 해시값 $H(pwd_{AB})$과 복호화로부터 얻은 $H(pwd_{AB})$이 일치하는지 비교하여 메시지 인증을 수행한다. 하나라도 정당하지 않으면 프로토콜을 종료한다. [그림 3]에 제안된 ORA 프로토콜의 패킷 포맷을 제시하였다. 음영은 암호화가 되어있음을 의미한다.

16byte	16byte	16byte	
ID_A	$H(pwd_B)$	Sequence Number	Data

[그림3] ORA의 프로토콜 패킷 포맷

2.2 ORA 안전성 분석

상호인증 프로토콜에서 반드시 고려되어야 할 공격은 가장 공격과 패스워드 사전공격이다. 이를 분석한 결과는 다음과 같다.

[요구 1] 제안된 ORA 프로토콜은 오프라인 패스워드 사전공격에 안전하다.

우선, 사용자 A는 사용자 B의 쌍별 패스워드 pwd_{BA}를 등록단계에서 얻어서 $H(pwd_{BA})$를 보관한다. 그러므로 당연히 사용자 A는 사용자 B의 쌍별 패스워드를 사전공격할 수 있다. 하지만, 쌍별 패스워드는 사용자 A와 사용자 B 간에만 사용되는 일회용 패스워드이므로 사용자 A가 사용자 B의 패스워드를 알더라도 그것을 이용해 사용자 B를 가장 할 수 없다. 둘째로 사용자 A,B가 아닌 외부 공격자가 전달되는 메시지를 통해 pwd_{AB}혹은 pwd_{BA}를 사전공격하는 경우를 고려할 수 있다. 하지만 모든 메시지는 공개키 암호화되어 있으므로 개인 키를 모르는 외부 공격자는 추측된 패스워드가 옳은지 옳지 않은지 검증할 방법이 없다. 물론, 온라인으로 프로토콜을 진행하면서 직접 $PE(PK_B, ID_A \| H(pwd_{BA}') \| Seq \| M_A)$를 보냈을 때 프로토콜 인증 실패 여부를 통해 pwd_{BA}'를 사전공격할 수 있지만, 이는 어디까지나 온라인 공격으로 최대 정책 횟수(예:3회)까지 가능하다.

[요구 2] 제안된 ORA 프로토콜은 재생 공격 및 가장 공격에 안전하다.

공격자가 예전 프로토콜 메시지를 그대로 보내는 재생 공격을 한다고 했을 때, ORA의 각 참여자가 연속번호 Seq를 비교함으로써 재생 공격을 쉽게 막을 수 있도록 설계하였다. 또한, 사용자 A를 가장하기 위해서는 pwd_{BA}를 반드시 알아야 한다. 이는 프로토콜 사전 단계에서 직접 사용자를 보면서 얻는 정보이므로 공격자에게 알려질 수 없다. 다만, 사전공격을 통해 알 수 있는데, 이는 요구 1에서 분석했듯이 공격자는 오프라인 사전공격을 할 수 없다.

I. 구현 및 결과 분석

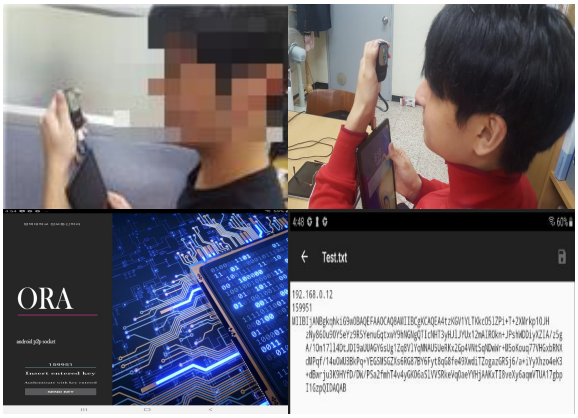
3.1 Android 기반 구현

태블릿 PC 및 홍채 인식 USB를 활용하여 안드로이드 기반 플랫폼에서 구현하였다. [그림 4]는 프로그램 최초 구동 시 사전정보 등록 단계로서 사용자 A, B의 패스워드, IP주소, 공개키를 각각 교환하여 각각의 홍채 인식 USB에 저장하는 과정이다. [그림 5]는 인증단계로 각자 개인 키로 복호화 후 패스워드 해시값을 비교 후 인증하고 메시지를 전달받는 화면이다.

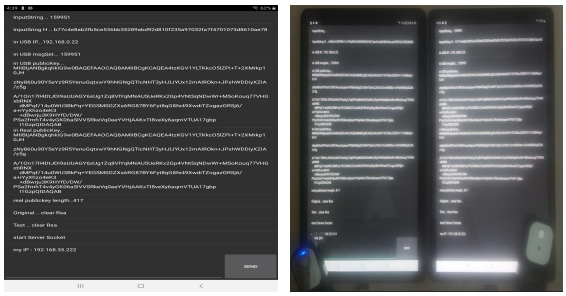
3.2 JVM 기반 일반 플랫폼 구현

JVM 기반 플랫폼은 Java Crypto API를 활용하여 구현하였다 [4]. Chatting Window 구현을 위해 Swing 그래픽 컴포넌트를 활용하였다.

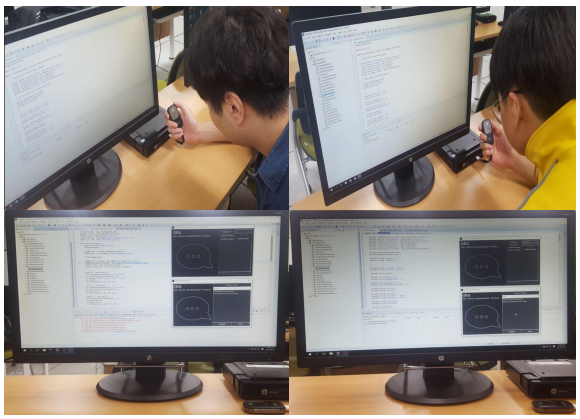
[그림 6]은 사전 정보 교환 단계에서 보안 USB 인증 및 교환된 패스워드 해시를 불러오는 과정을 거친다. [그림 7]은 JDK 기반 인증 및 메시지 교환 단계로서 1:1 네트워크 연결을 위해 일반적인 네트워크 Socket을 사용하였다.



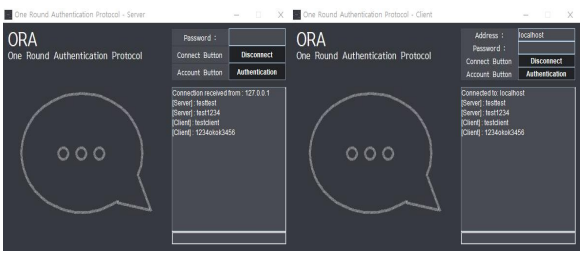
[그림 4] Android 기반 사전정보 등록단계



[그림 5] Android 기반 인증 및 메시지 교환 과정



[그림 6] JDK 기반 사전정보 등록단계



[그림 7] JDK 기반 인증 및 메시지 교환 과정

3.3 구현결과 분석

Android 및 JVM 환경에서 구현된 ORA 프로토콜의 동작 시간을 분석하였다. [표 4]는 각 환경에서 인증 시 걸리는 시간을 공개키 및 메시지 교환 관점으로 10회 측정 후 평균값을 계산한 결과이다. JVM은 유선, 안드로이드는 무선 환경을 가정하였으며, 모두 현실적인 시간이 소요됨을 확인하였다. 다양한 환경에서 향상된 성능을 위해 코드 최적화를 수행할 예정이다.

종류 / 횟수	Android 공개키 교환 (sec)	Android 메시지 교환 (sec)	JVM 공개키 교환 (sec)	JVM 메시지 교환 (sec)
1회	0.992	0.426	1.414	0.116
2회	0.764	0.133	1.338	0.117
3회	0.978	0.178	1.271	0.117
4회	0.207	0.255	1.490	0.118
5회	0.123	0.318	1.487	0.116
6회	0.428	0.133	1.418	0.116
7회	0.976	0.676	1.415	0.116
8회	0.179	0.201	1.512	0.113
9회	0.191	0.926	1.324	0.114
10회	0.226	0.023	1.332	0.114
평균	0.506	0.327	1.400	0.115

[표 4] 프로토콜 동작 시각 측정값

II. 결론 및 향후 연구 과제

본 논문에서는 사용자들이 보유한 단말기(태블릿, 스마트폰)와 보안 USB를 활용하여 상호 인증할 수 있는 ORA 프로토콜을 제안하고 구현하였다. ORA 프로토콜의 안전성 모델 설립과 증명 가능한 안전성 결과 도출은 향후 연구 과제로 진행하고 있다.

[참고문헌]

[1] "FIDO UAF Architectural Overview", FIDO Alliance Proposed Standard 08 December 2014, http://fidoalliance.org.

[2] "Introducing Titan Security Keys", https://support.google.com/titansecuritykey/answer/9115487?hl=ko&ref_topic=9115486.

[3] “Irisys LOCKIT 홍채 인식 보안 USB (LU-100)”, https://www.irisys.co.kr/clients.

[4] David Bishop, “Introduction to Cryptography with Java Applets” (김형준·손태식 공역). 홍릉과학출판사. (원서출판 2003).

표준 웹 기능을 활용한 디바이스 핑거프린팅 기반 사용자 식별 모델*

† 박소희, ‡ 최대선

공주대학교 융합과학과 대학원, 공주대학교 의료정보학과

User Identification Model based on Device Fingerprinting Using Web Standard Technology.

Sohee Park, Daeseon Choi

Department of Convergence Science, Graduate School of Kongju National University,
Department of Medical Information, Kongju National University

요 약

사용자의 편리한 인터넷 이용환경을 조성하기 위한 웹사이트의 플러그인 제거 정책 추진에 따라 기존 플러그인은 웹 표준 기술 및 무설치 방식의 대안 서비스 제공 방식으로 대체되고 있다. 하지만 금융서비스를 제공하는 은행 및 카드사의 웹사이트는 보안 및 이상거래탐지를 위해 사용자의 단말정보 및 환경정보를 수집하는 소프트웨어 설치가 필수적이다. 따라서 별도의 소프트웨어 설치 없이 사용자의 정보를 수집할 수 있는 대체 기술 및 대응정책이 필요하며 본 연구는 표준 웹 기본 기능을 통해 사용자의 단말정보를 수집 및 조합하여 사용자를 식별하는 기술인 디바이스 핑거프린팅에 대해 소개하고 이를 활용한 사용자 식별 및 이상거래탐지를 위한 모델을 제시한다.

I. 서론

인터넷 기술이 발전함에 따라 기존 대면 방식을 통한 업무 절차들이 온라인으로 전환되면서 대부분의 웹사이트에서는 다양한 서비스 제공, 편의성 제고 및 보안 강화를 목적으로 플러그인 기능을 사용해왔다. 플러그인은 브라우저가 제공하지 않는 기능을 사용하기 위해 별도로 PC에 설치해야하는 소프트웨어를 말하며 주로 비대면 본인확인, 전자서명, 개인정보보호, 방화벽, 키보드 보안 등 인증 및 보안 기능을 제공하기 위해 사용되거나 웹브라우저에서의 문서 열람, 동영상 재상, 증명서 출력 등 이용자의 편의를 위해 사용된다. 하지만 웹사이트마다 필요한 플러그인이 다르기 때문에 매번 다른 플러그인을 설치해야하고 중복 설치로 인한 PC 속도 저하, 설치로 인한 브라우저 강제 종료 등 사용자의 웹사이트 이용에 많은 불편함을 초래한다[1].

최신 웹 표준 기술인 HTML5에서 플러그인이 제공하던 다수의 기능을 제공하고 있으며 이에 따라 다수의 플러그인을 제거함으로써 사용자의 편리한 인터넷 이용환경을 조성하고자 정부에서는 웹사이트 플러그인 제거 정책을 추진 중이며 기존의 플러그인은 웹 표준 기술로 대체하거나 이용절차 변경 및 무설치 방식의 대안 서비스를 제공하는 방식을 도입하도록 하고 있다[2]. 그러나 노플러그인 정책은 웹사이트를 통해 전자금융거래서비스를 제공하는 기존 금융회사의 웹사이트의 보안정책과 충돌한다. 보안 및 이상거래탐지를 위해 사용자 단말정보/

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원(No.2016-0-00173, 핀테크 서비스 금융사기 방지를 위한 비대면 본인확인)을 받아 수행된 연구임

† 주저자, soheeee0803@smail.kongju.ac.kr
‡ 교신저자, sunchoi@kongju.ac.kr

환경정보/거래정보 등을 수집하고 있으며 사용자 단말정보 수집 모듈, 보안 소프트웨어 등의 설치가 필수적이다. 따라서 노플러그인 환경에 따른 별도의 소프트웨어 설치 없이 사용자의 단말정보를 수집할 수 있는 대체 기술 및 대응 정책 마련이 필요하다.

본 연구는 대체 기술로서 사용자의 단말정보를 수집 및 조합하여 특정 사용자를 식별하는 기술인 디바이스 핑거프린팅[3]을 소개하고 노플러그인 환경에 맞춰 표준 웹 기본 기능만을 사용해 수집 가능한 디바이스 핑거프린팅을 활용하여 사용자 식별 및 이상거래탐지모델을 제시한다.

II. 디바이스 핑거프린팅

디바이스 핑거프린팅은 웹사이트를 방문한 사용자의 디바이스 속성 정보, 네트워크 정보, OS정보, 브라우저의 구성이나 설정 등을 기반으로 장치를 트래킹 하는 방법[4]을 말하며 사용자의 단말로부터 수집한 여러 정보를 조합하여 고유한 식별자를 생성하고 이를 사용하여 사용자를 식별할 수 있다. 사용자로부터 수집 가능한 정보는 매우 다양하며 소프트웨어 설치 여부에 따라 수집이 가능한 정보가 상이하다. 본 연구는 별도의 프로그램 설치 없이 표준 웹 기본 기능만으로 수집할 수 있는 정보에 중점을 둔다.

디바이스 핑거프린팅 정보는 주로 JavaScript를 통해 수집하며 IP, Javascript Attribute, HTTP Request Header, User Agent[5] 등이 있다. 또한 웹 표준 HTML5 API를 활용하여 Canvas, WebGL, AudioContext, WebRTC[6], CSS3 Attribute[7], Font Metrics[8] 등을 수집할 수 있으며 디바이스의 특정 계산을 실행하는 프로세스 시간차이에 대한 값을 핑거프린팅으로 사용하는 CryptoFP[9]가 있다. 디바이스 핑거프린팅 속성 값은 운영체제, 브라우저, 그래픽카드, 사용자의 설정 등에 따라 정해지며 속성 값의 조합을 통해 고유한 키를 생성한다. 생성된 키를 사용자의 식별키로 사용함으로써 추가 인증방법이나 블랙리스트 기반의 계정관리 등의 사용자 식별 및 이상거래탐지에 활용할 수 있다. 표 1은 디바이스 핑거프린팅과 수집 정보를 나타내었으며 그림 1은 하나의 유니코드를 Canvas API(2D 그래픽을 그리기 위한 수단)를 사용하여 렌더링할 때 디바이스마다 렌더링 되는 결과의 차이가 있다는 것을 보여주는 예시이다.

Fingerprinting	Information
IP	IP address
JavaScript	Platform, Lists of Fonts/ Plugins, Cookie Enabled, Do Not Track, Screen Resolution, Etc.
HTTP Header	UserAgent, Accept, Encoding, Language
Canvas	Rendering Image
WebGL	Rendering 3d image
Web Audio	AudioContext (Audio Signal)
WebRTC	Local IP, Public IP
CSS3	Property, Selector, Filter (CSS3 Attribute)
Font metrics	width/height of Font (Glyph Rendering Value)
CryptoFP	Timing Matrix (Processing time)

표 1 . 디바이스 핑거프린팅

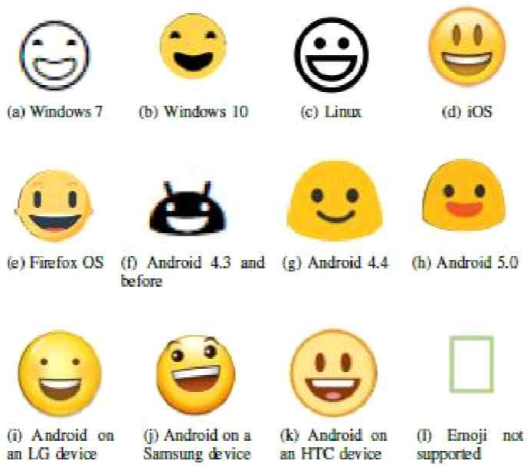


그림 1 . Canvas API를 사용하여 같은 유니코드를 렌더링한 결과

III. 디바이스 핑거프린팅 기반 사용자 식별 모델

3.1 사용자 식별 모델

노플러그인 환경 하에서 표준 웹 기본 기능을 사용한 디바이스 핑거프린팅을 통해 사용자를 식별하고 이상거래탐지를 위한 모델을 제시한다. 그림 2는 디바이스 핑거프린팅 수집/생성 과정 및 사용자 인증 과정을 나타낸다. 사용자 식별 모델은 디바이스 핑거프린팅 과정과 인증 부분으로 나뉘며 사용자가 특정 웹사이트에 로그인을 시도할 시 디바이스 핑거프린팅을 통해 사용자가 계정 소유자 본인이 맞는지 판단하는 디바이스 핑거프린팅 과정과 판단 결과에 따라 사용자의 접근 허용 및 추가 인증을 요구하는 인증과정으로 나뉜다.

전체 적인 과정은 다음과 같다. 사용자가 웹사이트에서 본인 계정을 입력 후 로그인을 시도하면 사용자의 단말정보를 수집하고 이를 조합하여 고유한 디바이스 핑거프린팅을 생성한다. 가장 먼저, 블랙리스트와의 비교를 통해 블랙리스트 목록에 있다면 사용자를 거절하거나 추가적인 인증을 요구하고 만약 블랙리스트 목록에 없는 경우 기존의 디바이스 핑거프린팅을 요청한다. 웹사이트의 처음으로 방문하는 사용자는 기존의 디바이스 핑거프린팅 값이 존재하지 않으며 따라서 새로운 사용자로 판단하여 생성된 핑거프린팅을 저장한다. 기존 디바이스 핑거프린팅 값이 존재하는 경우 새로 생성된 값과 비교하여 정확하게 일치한다면 본인임을 확인하고 서비스 이용을 허용한다. 하지만 디바이스 핑거프린팅이 일치하지 않는다면 두 핑거프린팅의 유사도 측정 과정을 수행한다. 이는 핑거프린팅 속성 값의 일부가 변화할 가능성이 충분히 있기 때문이다. 유사도를 측정하여 유사도점수가 허용 임계값 이상이라면 서비스 이용을 허용하고 새로운 디바이스 핑거프린팅을 저장 및 업데이트를 한다. 하지만 유사도점수가 임계값 미만이라면 사용자가 본인이 아닐 위험이 존재한다고 판단하여 추가적인 인증을 요청하거나 접속을 차단한다. 인증은 사용자의 디바이스 핑거프린팅을 통해 위험 수준에 따라 차등적으로 인증수단을 적용한다. 위험수준이 높은 경우에는 브라우저인증서나 모바일 인증서 등을 통해 사용자에게 인증을 요구하고 위험수준이 낮은 경우에는 SMS/ARS같은 휴대폰 본인확인 및 이메일 인증 등 비교적 간단한 인증방식을 통해 사용자의 신원을 확인한다.

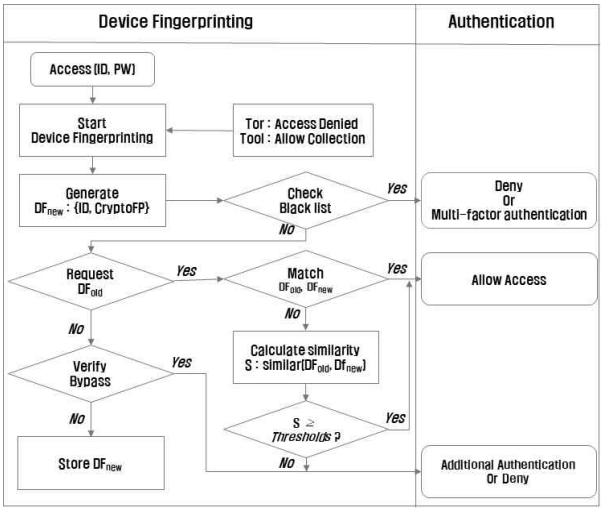


그림 2 . 디바이스 핑거프린팅 기반 사용자 식별 모델

3.2 엔트로피 기반 유사도 스코어링

디바이스 핑거프린팅을 통해 사용자 본인임을 확인하는 경우 핑거프린팅을 정확히 매칭하는 것뿐만 아니라 유사도 스코어링을 하는 이유는 핑거프린팅 정보가 변화할 가능성이 있기 때문이다. 사용자로부터 수집되는 정보는 디바이스 정보뿐만 아니라 네트워크, 운영체제, 브라우저 설정 등이 있다. 따라서 하드웨어나 소프트웨어를 업그레이드하거나 변경하는 경우, 사용자의 상황이 변하는 경우 혹은 임의적으로 설정을 변경하는 등의 다양한 이유로 핑거프린팅 정보는 충분히 바뀔 수 있는 가능성이 있기 때문에 유사성을 계산하여 동일한 사용자인지 아닌지 판단하는 과정이 필요하다.

유사도 스코어링 방법은 판단 기준이나 고려사항에 따라 달라질 수 있으며 본 논문에서는 디바이스 핑거프린팅의 식별성을 반영하기위해 엔트로피 기반 유사도 스코어링을 제시한다. 이는 사용자의 유사도 점수를 계산하는 과정에서 디바이스 핑거프린팅의 엔트로피 값을 계산 가

중치로 사용하는 방법으로 아래의 식과 같이 계산한다.

$$Entropy(x) = -\log_2 \Pr(X=x) = -\log_2(\frac{x}{X})$$

$$DF_{score} = \sum_{n=1}^{i} df_{i'score} \times Entropy(df_i)$$

식별성이란 특정 속성을 가지는 핑거프린팅이 얼마의 확률로 고유한지를 뜻하며 이는 정보 엔트로피를 측정하여 나타낼 수 있다. 엔트로피는 평균 정보량을 말하며 특정 사건이 일어날 확률이 낮을수록 불확실성은 커지고 엔트로피, 즉 정보량은 커지기 때문에[10] 엔트로피가 클수록 핑거프린팅이 중복될 확률은 낮아지고 식별성은 높다는 것을 의미한다. 따라서 엔트로피를 가중치로 사용하면 고유한 특성이 높은 핑거프린팅이 일치할수록 높은 점수를 부여할 수 있다. 예를 들어 단순히 디바이스 핑거프린팅 A의 엔트로피가 1이고 핑거프린팅 B의 엔트로피가 2일 때 A의 이전 값과의 유사도가 0.8이고 B의 유사도가 0.8 이라면 총 스코어가 2.4(A=0.8, B=1.6)가 나온다. 둘의 유사도가 같지만 엔트로피가 높을수록 반영되는 스코어는 높아진다.

IV. 결론

본 논문은 사용자의 편리한 인터넷 이용환경을 조성하기 위한 노플러그인 환경에 따라 기존의 플러그인을 대체할 수 있는 기술로서 디바이스 핑거프린팅을 설명하고 표준 웹 기본 기능만을 사용하여 수집할 수 있는 디바이스 핑거프린팅 정보를 소개하였다. 또한 이를 활용한 사용자 식별 및 이상거래탐지 모델을 제시하고 이에 활용할 수 있는 엔트로피 기반 유사도 스코어링 방법을 제시하였다. 디바이스 핑거프린팅을 대체 방안으로 사용하기 위해서는 식별성 뿐만 아니라 브라우저 호환성, 불변성, 구현 상의 오버헤드와 같은 기준을 고려해야한다. 디바이스 핑거프린팅을 다양한 기준에 따른 결과를 조사하고 있으며 실제로 핑거프린팅 구현 및 수집을 통해 사용자 식별을 위한 연구를 진행 중에 있다.

[참고문헌]

[1] “Guidelines for removal of public website plugins,” Ministry of the Interior and Safety, Nov. 2018.

[2] “Guidelines for improving private public website plugins,” Ministry of Science and ICT , Nov. 2018.

[3] P. Laperdrix et al. Browser Fingerprinting : A survey, arXiv:1905.01051, 2019.

[4] S. Jang et al, A Study on Online Fraud and Abusing Detection Technology using Web-Based Device Fingerprinting, Journal of The Korea Institute of Information Security & Cryptology, 28(5), Oct. 2018.

[5] N. Nikiforakis et al. Cookieless Monster : Exploring the Ecosystem of Web-based Device Fingerprinting, In IEEE Symposium on Security and Privacy, IEEE, 2013.

[6] The Spanish Data Protection Agency (AEPD), Survey on Device Fingerprinting, Feb. 2019.

[7] T Unger et al. Shpf: Enhancing http(s) session security with browser fingerprinting, 2013 International Conference on Availability, Reliability and Security. IEEE, 2013.

[8] D Fifield et al. Fingerprinting web users through font metrics, International Conference on Financial Cryptography and Data Security. Springer, 2015.

[9] I Sanchez-Rola et al. Clock around the clock: time-based device fingerprinting, Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2018.

[10] Wikipedia, Entropy(information theory), accessed Nov 07 2019, https://en.wikipedia.org/wiki/Entropy_(information_theory).

Permissioned Blockchain에서의 Privacy를 위한 OKSA기반 익명인증 기법에 관한 연구+

라경진*, 이임영**

순천향대학교 컴퓨터학과

A Study on OKSA based Anonymous Authentication Scheme for Privacy in Permissioned Blockchain

Gyeong-Jin Ra*, Im-Yeong Lee**

Dept of Computer Science and Engineering, Soonchunhyang University

요 약

Blockchain은 탈중앙성과 투명성, 확장성으로 각광 받아 여러 Application에 적용되고 있다. 하지만 사용자 인증 과정에서 다시 SPOF(Single Point of Failure)의 위협으로 Blockchain의 탈중앙성이 붕괴하는 문제가 발생하고 있다. 따라서 사용자 인증 또한 Blockchain기반의 DPKI(Decentralized Public Key Infrastructure), DID(Decentralized IDentity)가 주목받고 있다. 그러나 Permissioned Blockchain은 사용자의 신원이 확인된 멤버로 구성되므로 사용자의 식별정보가 Consortium Blockchain의 원장으로 공유 될 경우 Privacy의 문제가 생긴다. 따라서 Private Blockchain과 Consortium Blockchain에서의 사용자의 신원 연결성이 분리되어야 한다. 본 논문에서는 이를 위해 pseudonym ID를 사용하여 익명성을 강화한다. 따라서 Permissioned Blockchain에서의 OKSA(Oblivious Keyword Search Authority)기반의 pseudonym ID 획득 및 사용을 통한 익명인증 기법을 제안한다.

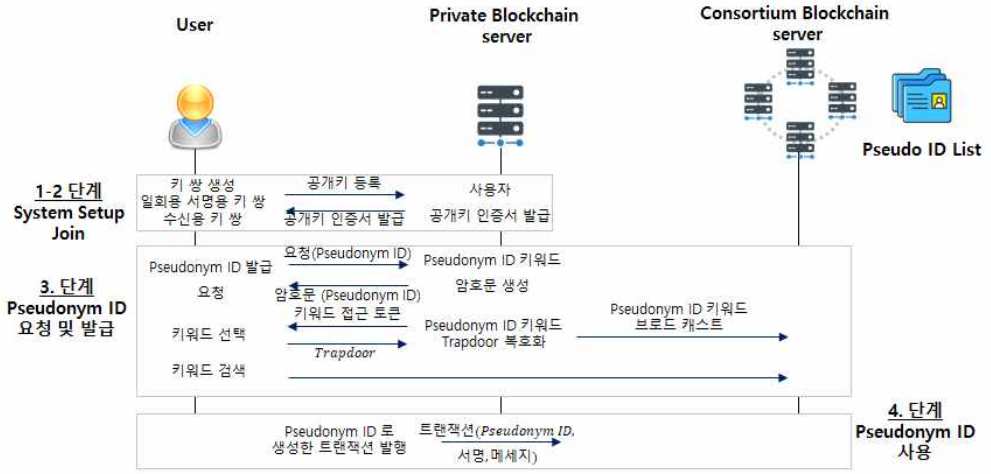
I. 서론

최근 Blockchain은 탈중앙성과 투명성, 확장성으로 각광 받아 여러 Application에 적용되고 있다. 이 중 Permissioned Blockchain은 신원이 허가된, 즉 확인된 사용자의 참여로 이루어지며, PKI(Public Key Infrastructure)의 공개키 인증서를 통해 사용자 인증을 수행한다. 하지만 인증서 발급과 유효검증 과정에서 다시 중앙집중형태가 될 경우 단일지점장애(SPOF, Single Point Of Failure)의 문제가 발생한다 [1]. 이를 위해 최근 사용자 인증 형태 또한 Blockchain기반의 DPKI(Decentralized Public Key Infrastructure), DID(Decentralized IDentity)가 주목받고 있다 [2]. 그러나 Private Blockchain에서의 사용자 신원정보가 Consortium Blockchain의 원장으로 공유될 경우 Privacy의 문제가 생긴다. 따라서 Private Blockchain과 Consortium Blockchain에서의 사용자의 신원은 연결성이 분리되어야 한다. 그러나 기존 제안방식은 Public Blockchain에 초점이 맞춰져 있거나 비효율적으로 제안되었다. 따라서 Permissioned Blockchain을 위한 안전하고 효율적인 익명인증 방식이 필요하다. 본 논문에서는 익명 신용장 및 Pseudonym ID를 통해 사용자 식별정보의 비연결성을 주어 Anonymity를 통한 Privacy를 보장하고자 한다. 이때 사용자는 Pseudonym ID를 OKSA(Oblivious Keyword Search Authority)를 통해 비밀스럽게 획득한다 [3].

II. 관련연구

익명성(Anonymity)은 프라이버시(Privacy)의 한 부분으로 블록체인에서 메시지(거래)가 공개되더라도 공개키가 개인의 오프체인 ID와

+ 이 논문은 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(No. NRF-2019R1A2C1085718) 및 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음 (IITP-2019-2015-0-00403)



(그림 1) 제안방식의 전체 시나리오

관련되지 않는 것을 의미한다 [4]. 또는 메시지를 통해(혹은 이를 모아) 개인 신원을 확인 및 추론 할 수 없음을 의미한다. 이를 위해선 사용자 신원 정보의 연결성 해제인 Un-linkability와 메시지의 기원 해제인 Un-traceability를 만족해야한다. 최근 Public Blockchain에서의 Privacy를 위한 기법은 “Hidden in a crowd”를 통해 무리 속에 사용자의 트랜잭션을 숨김으로서 익명성을 제공하는 방식이다. Mixing Service는 그의 한 예로 중간의 서버가 트랜잭션을 취합하여 발신자의 주소를 mixing함에 따라 pseudonym을 제공할 수 있다. 하지만 서버는 이를 확인할 수 있다는 Privacy 문제 및 서버의 SPOF 및 신뢰 문제가 발생한다. 한편, Ring Signature는 사용자의 공개키를 통해 누구나 서명을 생성할 수 있어 별도의 서버가 필요 없지만 참여자가 많을수록 서명키가 길어지는 단점이 있다. 또한 이러한 방식들은 Permissioned Blockchain에서의 적절한 멤버 관리, 추적 등의 기능을 제공하지 못한다. 이를 위해 Group Signature는 미리 생성한 그룹 키를 통해 서명을 생성하여, 서명한 사람이 어떤 그룹의 멤버라는 사실만을 알고 있을 뿐 그 사람이 누구인지에 대한 정보는 드러내지 않는다. 하지만 그룹매니저의 단일 수행으로 생기는 연산오버헤드, 그룹 키 관리(갱신, 분배 등)의 문제가 발생한다 [5]. 이는 익명 신용장 시스템(Anonymous credential system)을 통하여 사용자 신원인증과 실제 서비스 이용을 분리 할 수 있다. 익명 신용장 시스템(Anonymous credential system)이란, 사용자가 기관들로부터 신용장(credentials)을 획득하되, 각 기관마다 다른 token을 사용하여 신용장을 발급하고, 그 기관에서만 사용한다. 따라서 동일한 사용자가 다른 기관에서 사용하는 신용장이나 token을 자신의 기관에서 사용하는 신용장이나 token과 연결 지을(linked) 수 없는 기법이다 [6]. 이는 Permissioned Blockchain에서의 사용자 관리, 추적, 익명성, 접근제어 등을 효율적으로 제공할 수 있다. 또한, Secret Computation을 통해 키 없이 대화를 통해 비밀의 소유를 증명하기도 한다. Zero-Knowledge Proof(Zkp)나 ZK-SNARKs 등이 한 예이다. 하지만 이는 매우 속도가 느리고 검증 메세지를 별도로 생성해야 한다 [4]. OKSA는 Oblivious Transfer을 통해 확률적으로 데이터를 획득하여, 데이터의 검색 및 데이터의 접근을 비밀스럽게 진행한다. 이는 PEKS(Public Encryption Keyword Search)에서 메시지 키워드에 비례하여 증가하는 암호문의 길이를 고정 시켜 보다 효율적이고 안전하게 데이터를 획득하게 한다 [3].

III. 제안방식

본 논문에서는 Permissioned Blockchain에서의 Privacy를 위한 익명인증을 위해 익명신용장 기반의 익명 프로토콜을 제안한다. 사용자의 pseudonym ID는 익명신용장 시스템에 따라 여

러 pseudonym ID를 소유할 수 있고 다른 그룹은 사용자를 pseudonym ID로 인식한다. 사용자는 일회용 서명용 공개키, 개인키와 수신용 공개키, 개인키 쌍을 이용한다. 이로 인해, 제안방식에서는 Aggregator Signature를 통한 효율적인 일회용 키 검증 및 트랜잭션의 서명 압축을 수행한다. 서버는 사용자에게 OKSA를 통해 신용장을 발급해주며, 사용자는 ID 노출 없이 신용장의 소유를 다른 기관에게 증명함으로써, 자신이 최초에 성공적으로 인증 수행했음을 보인다. 또한 일회용 키와 검증 값으로 바인딩한 공개키 인증서 및 PKI 통해, Private Blockchain은 사용자를 관리 할 수 있으며, 추후 정당한 요청자에 의해 사용자의 정보를 제공할 수 있다. 본 제안방식의 전체 시나리오는 (그림 1)에 따라 다음과 같다. 전체 단계는 4단계(System setup 단계, Join 단계, Pseudonym ID 요청 및 발급 단계, Pseudonym ID 사용 단계)로 구성된다.

1) System setup 단계

이 단계에서 Private Blockchain과 사용자는 보안 정책에 따라 적절한 보안 강도를 설정하고, 서명용 일회용 공개키 및 개인키와 수신용 공개키 및 개인키를 생성한다.

2) Join 단계

이 단계에서 사용자는 가입을 위해 공개키인증서를 발급받고자 한다. Aggregator Signature의 수신자를 사용자 자신으로 하여, 일회용 공개키를 압축함으로서 일회용 공개키의 바인딩을 수행한다. 이후 사용자는 일회용 공개키 확인 값과 일회용 공개키를 모두 Private Blockchain에 전송하고, Private Blockchain Server는 사용자 목록 List = (List [1]… List [n]을 생성한다.

3) Pseudonym ID 요청 및 발급 단계

이 단계에서 사용자는 Private Blockchain Server에서 List를 요청하고 Private Blockchain의 OKSA를 통해 이를 얻는다. 이 단계는 3개의 소단계로 구성된다.

3-1). Keyword Set Negotiation

사용자는 수신용 키를 사용하여 추후 Private Blockchain Server에서 사용할 가명 ID를 검색하기 위한 키워드를 암호화한다. Private Blockchain Server는 키워드를 포함하여 사용자의 Pseudonym ID를 생성한다.

3-2). Pseudonym ID Issuance

Private Blockchain Server는 유효한 사용자가 pseudonym ID를 검색하고 사용할 수 있도록 사용자의 수신용 공개키로 암호화를 적용한다. 이는 사용자 요청과의 트랜잭션으로 발생하여 Consortium Blockchain 네트워크로 브로드캐스트 된다.

3-3). Pseudonym ID Search

사용자는 수신용 개인키를 통해 가명 ID를 검색하고 액세스한다.

4) Pseudonym ID 사용 단계

사용자는 주소 대신 발행 된 pseudonym ID을 사용하여 일회용 서명키로 서명한 메시지 트랜잭션을 생성하고, Private Blockchain는 이를 공개키 인증서와 pseudonym ID의 유효성을 확인한다. 이후 Aggregator Signature를 통해 Consortium Blockchain으로 트랜잭션을 생성한다.

IV. 제안방식 분석

이 장에서는 4장에서 제안한 방식과 기존방식의 비교분석을 통해 제안방식의 강점에 대해 논의해 본다. <표 1>에 따라 제안방식은 다음과 같은 보안요구사항을 만족한다.

Authorization: Private Blockchain의 사용자가 생성한 서명은 정당한 사용자의 일회용 서명용 개인키로 생성되어 Blockchain으로 Commit 되었으며, 다른 사용자는 이를 통해 검증을 수행 할 수 있다. 일회용 개인키에 해당하는 공개키는 공개키 검증 값과 공개키 인증서를 통해 유효하다. 이는 모두 ECDLP(Elliptic Curve Discrete Logarithm Problem) 안전성에 기인한다.

Confidentiality : OKSA의 암호화 수행을 통해 pseudonym ID의 획득 및 키워드의 기밀성을 만족한다. 또한 수신용 키 쌍을 통해 민감한 정보의 암호화 수행이 가능하다.

<표 1> Blockchain에서의 Privacy를 위한 기존과 제안방식 비교

	Authorization	Confidentiality	Guess the key	Anonymous		Traceability (Conditional)
				Non-Linkability	Non-traceability	
Hardjono, T	Private Issuer	Not Considered	Strong	ZKP	ZKP	Private Issuer
Liu, Y	Private Issuer	Not Considered	Weak	Mixing Server	Mixing Server	Not offer
Mercer,R	Public Issuer	Not Considered	Weak	Not offer	Ring Signature	Tag+ZKP
Axon, L. M.	Public Issuer	Offer	Strong	Off Chain	Off Chain	Not offer
Proposed Scheme	Private Issuer	Considered	Strong	Pseudo ID + OKSA	Aggregator Signature	Credential System + PKI

Unforgable: Private Blockchain은 공개키 인증서 기반으로 사용자를 제어하고, 이를 통해 유효한 그룹만이 서명생성이 가능하다.

Anonymity: Anonymous Credential System에 따라, 사용자는 사용자의 pseudonym ID와 일회용 키를 사용하므로 사용자의 신원을 연결 할 수 없으며 (Un-linkability), Aggregator Signature를 통해 Consortium Blockchain의 메세지의 기원을 알 수 없다(Un-traceability).

Anti-collision attack: Private Blockchain 51% 미만의 사용자들이 공모를 통해 공모에 참여하지 않은 정당한 사용자의 서명을 위조하는 것은 Hash Function의 Collision attack과 Pre-image attack을 시도하는 것과 같으며 이는 현실적으로 매우 어렵다.

Traceability(Conditional): Private Blockchain의 Credential System과 PKI의 신원정보 관리기능을 통해 필요에 따라 악의적인 사용자의 추적, 관리가 가능하다.

V. 결론

최근 Blockchain은 탈중앙성과 투명성, 확장성으로 각광 받아 여러 Application에 적용되고 있다. 이 중, Permissioned Blockchain은 신원이 허가된, 즉 확인된 사용자의 참여로 이루어진다. 하지만 Permissioned Blockchain에 신원정보가 포함 될 경우 사용자 Privacy 문제가 발생할 수 있다. 그러나 기존 제안방식은 Public Blockchain에 초점이 맞춰져 있거나 비효율적으로 제안되었다. 따라서 본 제안방식은 Permissioned Blockchain에서 일어날 수 있는 Privacy 문제와 이를 위한 해결방안으로 익명인증기법을 제안하고 이를 보안요구사항에 따라 기존방식과 비교 분석하였다. 향후 지속적인 연구와 실제 환경에 적용을 통해 본 제안방식의 효용성을 입증 할 수 있도록 수행할 예정이다.

[참고문헌]

[1] Biswas, K., & Muthukkumarasamy, V. “Securing smart cities using blockchain technology”, IEEE 14th international conference on smart city, IEEE, pp. 1392-1393, 2016.

[2] Dunphy, P., & Petitcolas, F. A., “A first look at identity management schemes on the blockchain”, IEEE Security & Privacy, 16(4), pp. 20-29, 2018

[3] Jiang, P., Wang, X., Lai, J., Guo, F., & Chen, “Oblivious keyword search with authorization”, In International Conference on Provable Security, Springer, pp. 173-190, 2016.

[4] Mercer, Rebekah. "Privacy on the blockchain: Unique ring signatures.", arXiv preprint arXiv:1612.01188, 2016.

[5] J. Hwang, S. Lee, B. Chung, H. Cho and D. Nyang, “Short group signatures with controllable linkability”, Proc. of the 2011 Workshop on Lightweight Security & Privacy: Devices, Protocols, and Applications, pp. 44-52, 2011.

[6] E. F. Brickell, J. Camenisch and L. Chen, “Direct anonymous attestation”, 11th ACM Conference on Computer and Communications Security, 2005.

최소행위 중 EEG 기반 사용자인증 연구

남승수*, 최대선**

*공주대학교 지능보안 연구실, *공주대학교 의료정보학과

User Authentication Based on EEG Data During Minimum Action

Seungsoo Nam*, Daeseon Choi**

*Intelligent Security Lab, Kongju National University.

**Department of Medical Information, Kongju National University.

요 약

뇌파는 수많은 신경 세포의 활동 전위들을 두피의 전극 또는 뇌에 삽입한 전극으로 감지된 뇌의 전기적인 활동이다. 이러한 뇌파는 가만히 있을 때보다 어떠한 행위를 했을 때 사용자의 개인적 특징이 두드러지게 나타날 수 있어 인증시스템에 관한 연구가 활발히 이루어지고 있다. 본 논문에서는 마우스 클릭행위 중 뇌파데이터를 이용하여 사용자인증을 하는 방법을 제안한다. 수집된 뇌파데이터의 특징을 추출하기 위해 밴드패스필터와 푸리에 변환으로 특징을 추출하였다. 마지막으로 사용자인증을 위해 CNN(Convolution Neural Network)을 이용하여 뇌파 데이터를 학습하고 테스트 데이터로 사용자 인증 여부를 확인할 수 있게 했다. 실험결과 마우스 클릭 행위 상태에서 84.5%의 정확도를 보였으며, 정지상태보다 6% 높은 결과를 보였다.

I. 서론

뇌파(Electroencephalography, EEG)는 한스 베르거(Hans Berger)에 의해서 발견되었으며, 뇌에서 발생된 전위의 변화이다. 뇌에서 어떤 일이 발생될 때 뇌를 구성하는 신경세포(뉴런)의 상태가 변하게 된다. 이때 상태 변화는 신경세포의 막 주변의 이온 이동에 의해 발생되게 되며, 이 결과로 세포막전위의 변화가 발생되게 된다. 이 세포막전위의 변화를 활동 전위라고 하며 수많은 신경세포의 활동전위들을 두피(Scalp)의 전극 또는 뇌에 삽입한(Intracranial) 전극으로부터 감지할 수 있다. 이렇게 감지된 뇌의 전기적 활동이 뇌파이다. 이는 전기적 신호를 통해 뇌의 활동을 볼 수 있기 때문에 최근 뇌파데이터를 통한 사용자 인증연구기술이 주목받고 있다[1].

현재 많이 상용화 되고 있는 인증방법인 지문, 안면인식이 대표적이다. 하지만 생체인증시스템 또한 실리콘을 이용한 지문복제[2], SNS(Social Network Service)를 악용하여 타인의 사진을 도용한 안면인식[3] 시스템의 해킹 등 여전히 보안의 위험성이 존재한다. 이처럼 사용자의 특징이 밖으로 들어나게 되면 공격자들에게 표적이 되기 싶다. 그러나 뇌파는 사용자의 특징이 겉 표면에 들어나지 않는다. 혹, 뇌파의 전기적인 신호를 훔치더라도 노이즈가 심한 뇌파데이터가 수집되어 뇌파데이터를 분석하는 것은 쉽지 않다.

본 논문에서는 마우스클릭 행위 중 뇌파데이터를 수집하여 사용자 인증방법을 제안한다. 행위는 뇌파에 많은 영향을 미치고 있고 사람마다 다르게 생각하는것과 마찬가지로 행위를 통해 전극 신호도 다르기 때문에 사용자 인증 정확도를 높일 수 있다 행위는 마우스를 클릭하는 행위로 정하여 수집했다. 마우스 클릭행위는 데스크탑이나, 노트북을 사용할 때 많이 이용하

고 뇌파데이터 수집에 대해 거부감이 적게 들기 때문이다. 수집된 뇌파데이터는 밴드패스 필터(Band-Pass Filter)와 푸리에 변환을 통해 사용자에 맞는 뇌파 특징데이터를 수집하고 딥러닝 방식중 하나인 CNN(Convolution Neural Network)을 이용하여 학습 및 검증에 사용한다.

순서로는 2장에서는 관련 연구, 3장에서는 제안하는 EEG신호를 이용한 특징추출 방법, 인증에 사용될 CNN구조, 4장에서는 실험구성 및 결과 마지막 5장은 결론 및 향후연구계획에 대해 기술한다.

II. 관련 연구

2.1 뇌파 측정 장치의 구성

뇌파 측정 장치는 크게 전극부 (Electrode panel), 컴퓨터 (Computer), 뇌파를 보여주는 모니터 (Signal display)로 구성되어진다. 그림 1은 뇌파 측정 기계의 구성을 도식화한 그림이다.

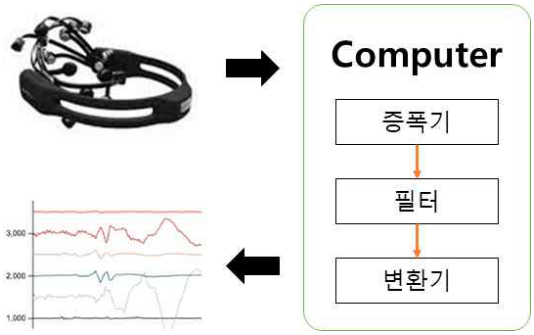


그림 1 뇌파 측정기계구성도

컴퓨터 부분은 교정기 (Calibrator), 필터 (Filter), 아날로그-디지털 변환기(Analog to digital converter; ADC)로 이루어진다. 교정기는 뇌파의 전압이 제대로 측정되었는지 확인하는 역할을 한다. 뇌파는 두피의 여러 위치에서 측정할 경우 여러 채널(Multi-channel)의 신호를 획득하는데 모든 채널에서 신호가 측정되었는지 확인하는 역할 또한 교정기가 한다. 변환기는 아날로그-디지털 변환기는 컴퓨터가 신호를 저장하기 위해 디지털 신호로 변환하는 역할을 한다.

2.2 EEG 기술연구

뇌파를 분석하여 사용자를 구분하는 연구를 수행할 것이기 때문에 기존 연구의 뇌파 분류 성능을 분석하고 실용적인 방법을 제안한다.

Rodrigues [5]는 뇌파 신호에 기초로 채널 개수를 줄이면서 사용자인식을 위한 가장 높은 인식률을 찾기 위해 이진 제한 Flower Pollination Algorithm을 사용했다. 뇌파데이터는 표준뇌파 데이터세트인 BCI2000 데이터이며 채널수를 64채널에서 32채널까지 줄이면서 실험결과 인식률이 거의 87%의 정확도를 보였다. B. C. Armstrong[6]은 실험자는 텍스트를 자동으로 읽으면서 사용자 인증에 사용되었다. 실험자는 텍스트를 읽으면서 15명의 실험자들에게 1주일~6개월 동안 수집되었다. 학습과 검증 알고리즘은 SVM을 시용하였다. 실험결과 89%의 정확도를 보였다. 이처럼 EEG로 정확도를 높이기 위해 실험자들에게 여러 가지 조건을 부여하는 방법으로 실험을 하고 편리성을 위해 전극 수를 줄이면서 동일한 정확도를 표현하는 연구가 진행된다. 본 실험에서는 Emotiv 사의 Epoc+ 14ch을 가진 뇌파데이터 기기를 사용했으며, 실생활에 많이 사용하는 마우스 행위로 사용자 인증연구를 한다.

III. 특징 추출

3.1 특징 추출

데이터 수집을 위해 사용한 EEG 측정기기인 Epoc+로 측정된 뇌파데이터는 내부적으로 2048 Hz 주파수를 샘플링한 후, 채널당 128 Hz 주파수로 다운샘플링 되어 뇌파기기와 연결된 PC에 저장된다. 저장된 데이터는 노이즈를 포함한 데이터이므로 노이즈제거 후 실험에 사용될 사용자 특징을 가진 EEG신호만 추출해야한

다. 특징추출 방법은 밴드패스필터로 1차 필터링을 한다. 밴드패스필터는 입력주파수에서 특정한 주파수 영역만 통과시키도록 주파수이하 신호와 주파수이상신호를 제거하고 출력하는 필터이다. 1차 필터링을 마치고 나면 두 번째로 푸리에 변환을 통해 2차 필터링을 마친 특징을 추출 한다. 1차 필터링된 데이터정보를 푸리에 변환했을 경우 20초간 측정되었던 신호 전체에 대한 주파수 정보를 얻을 수 있다[7]. 주파수의 정보를 식으로 나타내면 식 (1)로 표현할 수 있다.

$$x(t) \underset{\text{역푸리에변환}}{\overset{\text{푸리에변환}}{\rightleftharpoons}} A(f), \Phi(f) \tag{1}$$

이 변환을 푸리에 변환이라고 부른다. 푸리에 변환은 진폭A와 위상 Φ를 같이 나타내므로 복소수 형식으로 사용한다. 이 푸리에 변환과정의 결과는 식(2)과 같다.

$$x(f) = \int_{-\infty}^{+\infty} x(t) e^{-\Omega \Pi} dt \tag{2}$$
$$x(t) = \int_{-\infty}^{+\infty} X(t) e^{-\Omega \Pi} df$$

푸리에 변환된 그래프는 EEGdata신호의 주파수 성분별 세기를 파악할 수 있다.

3.2 CNN 기반 사용자 인식

일반적인 인공신경망은 분류의 측면에서 그림의 이동(Shift), 크기 변화(Translation), 왜곡(Distortion)에 취약하다는 단점을 가지고 있다(8). 이러한 단점을 보완한 합성곱신경망은 인공신경망에서 발전된 형태의 모델이다[8]. 합성곱신경망은 합성곱연산을 특징으로 가지며, 합성곱레이어(Convolution layer)와 풀링레이어(Pooling layer) 그리고 전체결합레이어(Fully connected layer)로 이루어진다. 각 노드들은 가중치의 행렬(Matrix)인 필터(Filter)들로 구성되어 있다. 각각의 필터들은 입력데이터와 합성곱연산이 수행된다. 이렇게 연산이 수행된 결과 각각의 필터들은 입력데이터의 여러 특징들을 반영한다. 그 후 특징들이 반영된 각각 필터 내의 값들 중 대표 값을 정하여 서브 표본화를 진행한다. 이는 여러 특징들이 계산되는 차원을 줄여주며 과적합화(Overfitting)를 막아준다. 마지막으로 이전층의 여러 특징 값을 인공신경망에서 사용된 신경망을 통해 연결한다. 이를 통해 데이터의 분류를 진행할 수 있다. 그림 8은 합성곱 신경망의 학습 방법을 도식화한 그림이다.

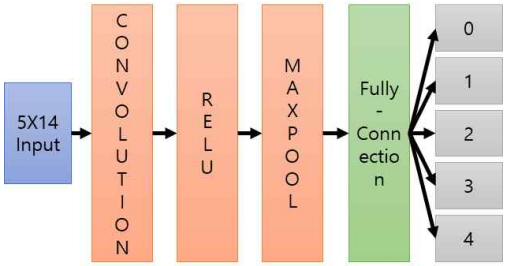


그림 2. CNN 구조

IV. 실험방법 및 결과

실험데이터 수집을 위해 실험자는 뇌파측정에 영향을 미칠 수 있는 정신적 또는 신체적 질환이 없고, 모든 실험자는 본 실험에 대한 충분한 설명과 주의 사항을 인지하고 실험에 참여하였다. 실험자로는 건강한 20~30대 남성을 대상으로 5명을 선정하여 진행했으며, 하루에 10초씩 20번 데이터를 수집을 했다. 처음 10번은 정지된 환경에서 한곳을 응시한 상태에서 EEG수집, 다음 10번은 최소의 행위 측정을 위해 1초 간격으로 마우스를 클릭한 상태로 EEG를 수집 했다.

수집된 데이터는 CNN을 통해 학습 및 검증에 사용되었는데 1~5일차데이터로 학습을 하고 6~10일차 데이터로 검증에 사용했으며, 비교실험모델로는 시계열 데이터에 강인 하다는 RNN 모델로 같은 비율로 나누어 비교했다.

표 1. 인증 비교 실험결과

Model / State	RNN	CNN
Think	73.75%	**78.5%**
Mouse	77.5	**84.5%**

실험 결과는 표 1과 같으며, 생각만 하고 있을 때 RNN모델에서는 73.75%, CNN 모델은 78.5%의 정확도를 보였으며 RNN모델보다 CNN모델이 4.75%높은 정확도를 보였으며, 마우스클릭 행위 중에 정확도도 CNN모델에서 84.5%를 보여 RNN모델보다 7% 높은 정확도를 보이고 생각과 행위의 정확도를 비교하면 행위를 통한 뇌파가 약 4.8% 높았다. 이는 사용자가 작은 행위라도 했을 때 미세한 뇌파 및 근육의 움직임으로 인해 사용자를 특징이 발생되는 것을 알 수 있다.

V.결론

본 논문에서는 사용자인증 성능정확도를 높이기 위해 마우스 클릭행위 중 EEG데이터를 수집해 밴드패스필터로 노이즈를 제거하고 FFT로 2필터링을 했다. 추출된 뇌파데이터는 CNN을 통해 학습 시키고 사용자인증 판별율을 산출했다. 실험결과 생각만 하고 있을 때보다 6%높은 결과를 보였다.

향후 연구에는 키보드 입력 중에 대한 뇌파 데이터를 수집하여 더 큰 행위를 통해 사용자 인증을 높이는 연구를 할 것이다.

[참고문헌]

[1] L. Ma, J.W. Minett, T. Blu, and W. Wang, “Resting State EEG-Based Biometrics for Individual Identification Using Convolutional Neural Networks,” *IEEE EMBC*, pp. 2848-2851, Aug. 2015.

[2] A. Chatterjee, V. Bhatia, and S. Prakash, “Anti-spoof touchless 3D fingerprint recognition system using single shot fringe projection and biospeckle analysis,” *Opt. Lasers Eng.*, vol. 95, pp. 1 - 7, 2017.

[3] K. Patel, H. Han, A. K. Jain, and G. Ott, “Live face video vs. spoof face video: Use of moire patterns to detect replay video attacks,” *IEEE International Conference on ICB*, pp. 98 - 105, 2015.

[4] https://www.theverge.com/2018/12/31/18162541/vein-authentication-wax-hand-hack-starbug

[5] D. Rodrigues, G. F. Silva, J. P. Papa, A. N. Marana, and X.-S, “Yang. Eegbased person identification through binary flower pollination algorithm,” *Expert Systems with Applications*, vol. 62, pp. 81 - 90, 2016.

[6] B. C. Armstrong, M. V. Ruiz-Blondet, N. Khalifian, K. J. Kurtz, Z. Jin, and S. Laszlo, “Brainprint: Assessing the uniqueness, collectability, and permanence of a novel method for erp biometrics,” *Neurocomputing*, vol. 166, pp. 59 - 67, 2015.

[7] Acharya, U.R., et al., “Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals”, p. 270-278, 2018.

스마트 그리드 환경에서 안전한 통신을 위한 사용자 인증 및 키 합의 방식의 보안 취약점 분석 및 대응방안

박기성*, 노성기*, 윤대근*

*한국전자통신연구원

Cryptanalysis and countermeasure of user authentication and key agreement scheme in smart grid environment

KiSung Park*, SungKee Noh*, DaeGeun Yoon*

*Electronics and Telecommunications Research Institute.

요 약

2019년 Zhang 등은 스마트 그리드 환경에서 안전한 통신과 사용자의 프라이버시를 보호하기 위한 인증 및 키 합의 방식을 제안하였다. 본 논문에서는 Zhang 등의 인증 및 키 합의 방식의 안전성을 분석하여 단일 요소 안전성 의존 문제, 스마트 미터 도난 및 위장 공격, 세션 키 노출 공격 등 다양한 공격에 취약함을 입증하고 이에 대한 대응 방안 및 향후 연구 방향을 제시한다.

I. 서론

최근 스마트 그리드 환경에서 사용자의 데이터를 안전하게 보호하고 신뢰성 있는 전력 서비스를 제공하기 위한 인증 방식에 대한 연구가 활발히 이루어지고 있다[1-3]. 2016년 Odelu 등은[1] 스마트 그리드 환경에서 안전한 통신을 위한 인증 및 키 합의 방식을 제안하였다. 그러나 2017년 Chen 등은[2] Odelu 등의 방식이 위장 공격에 취약하며 불추적성을 보장하지 못함을 보이고 이를 개선한 방식을 제안하였다. 또한 2019년 Zhang 등은[3] tamper-resistant 디바이스를 사용하여 기존의 제안된 방식들의[2-3] 보안 취약점과 성능을 개선한 인증 및 키 합의 방식을 제안하였다. 그러나 Zhang 등의 방식은 tamper-resistant 디바이스가 이상적으로 안전하다는 강한 가정을 가지고 개인키와 같은 사용자의 민감한 정보를 tamper-resistant 디바이스에 그대로 저장하는 문제점이 있다. 이러한 이상적인 가정에 안전성을 전적으로 의존하는 것은 tamper-resistant 디바이스가 공격에 노출되는 경우 시스템 전체의 안전성을 침해하고 심각한 피해를 일으킬 수 있다[4-5].

또한 초연결 시대가 다가옴에 따라 스마트 미터와 같은 사물인터넷 기기들의 데이터 발생량이 폭발적으로 증가하고 있으며 이러한 사물들의 안전성을 tamper-resistant와 같은 강한 가정 없이 보장할 수 있어야 하므로 블록체인을 활용하여 Trusted Third Party (TTP) 없이 효율적으로 데이터를 교환 및 관리 할 수 있어야 한다.

본 논문에서는 2019년 Zhang 등이 제안한 인증 및 키 합의 방식이 단일 요소에 안전성을 의존하는 문제점이 있음을 보이고 이 요소가 공격자의 공격에 노출되었을 때 발생할 수 있는 보안 취약점을 분석하며 이에 대한 대응방안 및 향후 연구 방향을 제시한다.

II. Zhang 등이 제안한 방식

Zhang 등이[3] 제안한 인증 및 키 합의 방식은 등록 단계, 인증 및 키 합의 단계로 구성되며 본 논문에서 사용하는 시스템 매개 변수는 다음 표 1과 같다.

표 1. 시스템 매개 변수

기호	의미
SM_i, SP_j	스마트 미터, 서비스 제공자
ID_i	스마트 미터의 아이디
ID_j	서비스 제공자의 아이디
s	서비스 제공자의 마스터 키
r_1, r_2, r_3	랜덤 넘버
k	대칭 키
Q_i	스마트 미터의 동적 아이디
$h(.)$	단 방향 해시 함수
$\|$	결합 연산
$E_k()/D_k()$	대칭키 암호화/복호화 연산
$\oplus$	XOR 연산

2.1 Zhang 등의 스마트 미터 등록 단계

Zhang 등이 제안한 방식의 스마트 미터 등록 단계는 다음 그림 1과 같다.

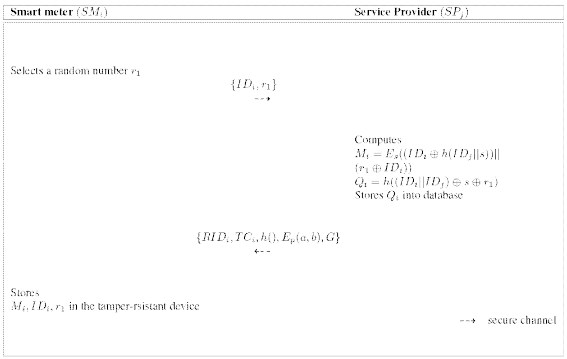


그림 1. Zhang 등이 제안한 스마트 미터 등록 단계

2.2 Zhang 등의 인증 및 키 합의 단계

Zhang 등이 제안한 방식의 인증 및 키 합의 단계는 다음 그림 2와 같다.

III. Zhang 등이 제안한 방식의 보안 취약점 분석

본 논문에서는 Zhang 등의 방식에서 이상적으로 안전하다고 가정된 tamper-resistant 디바

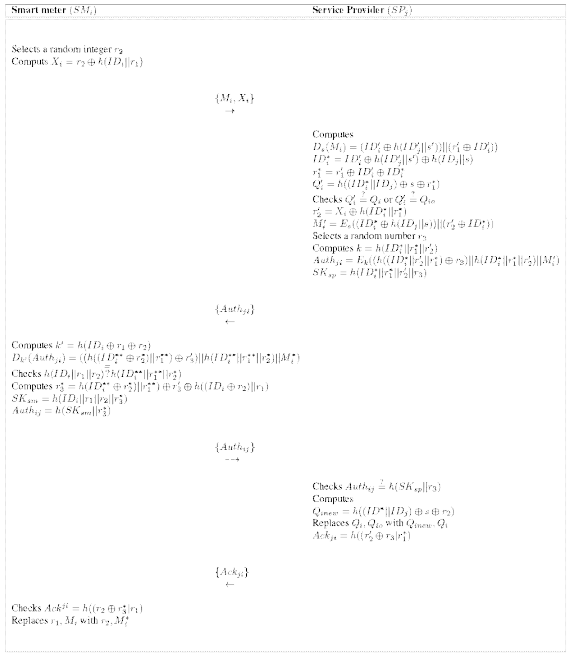


그림 2. Zhang 등이 제안한 인증 및 키 합의 단계

이스가 공격자에게 노출될 경우 스마트 미터 도난 및 위장 공격, 세션 키 노출 공격 등에 다양한 공격에 취약하고 상호 인증을 제공하지 않음을 informal 분석을 통하여 입증하였다.

Zhang 등의 방식의 보안 취약점은 다음과 같다.

3.1 단일 요소 안전성 의존 문제

Zhang 등의 인증 및 키 합의 방식은 tamper-resistant 디바이스가 이상적으로 안전하다고 가정하고 사용자의 중요한 정보 ID_i, r_1을 그대로 저장하고 있다. 그러나 이러한 강한 가정은 tamper-resistant 디바이스가 공격에 노출될 경우 시스템 전체의 안전성을 침해하고 개인 및 국가적인 손실을 야기할 수 있다[4-5].

3.2 스마트 미터 도난 공격

스마트 미터 도난 공격은 공격자가 사용자의 스마트 미터를 획득하고 스마트 미터안의 저장된 값을 활용하여 사용자의 정보를 탈취하거나 변조하는 공격이다. Zhang 등의 방식에서 공격자는 스마트 미터로부터 저장된 값 M_i, ID_i, r_1를 얻고 이를 활용하여 스마트 미터 위장 공격, 세션 키 노출 공격 등 다양한 공격을 시도할 수 있다.

3.3 스마트 미터 위장 공격

공격자는 스마트 미터 도난 공격을 통하여 획득한 M_i, ID_i, r_1을 이용하여 로그인 요청 메시지 M_i, $X_i = r_2 \oplus h(ID_i \| r_1)$을 성공적으로 생성할 수 있으므로 Zhang 등의 방식은 스마트 카드 위장 공격에 취약하다.

3.4 세션 키 노출 공격

Zhang 등의 방식에서 공격자는 사전에 획득한 정보 M_i, ID_i, r_1과 공개 채널로 전송되는 값 X_i, $Auth_{ji}$을 이용하여 해당 세션에서 사용된 세션 키를 생성할 수 있다. 먼저 공격자는 $r_2 = X_i \oplus h(ID_i \| r_1)$를 얻고 r_1, ID_i를 이용하여 대칭키 $k = h(ID_i \oplus r_1 \oplus r_2)$를 생성한 후 $Auth_{ji}$를 복호화하고 암호화된 값들을 얻는다. 따라서 공격자는 세션 키 생성에 사용되는 모든 값 ID_i, r_1, r_2, r_3를 얻을 수 있으므로 Zhang 등의 방식은 세션 키 노출 공격에 취약하다.

3.5 상호 인증

Zhang 등의 방식에서 공격자는 스마트 미터 도난 공격 및 세션 키 노출 공격을 이용하여 세션 키를 얻을 수 있으며 로그인 요청 메시지를 생성하여 합법적인 스마트 미터로 위장할 수 있으므로 Zhang 등의 방식은 안전한 상호 인증을 보장하지 않는다.

IV. Zhang 등이 제안한 방식의 보안 취약점 대응방안 및 향후 연구

Zhang 등의 방식의 보안 취약점을 개선하기 위한 대응 방안과 향후 연구 방향은 다음과 같다.

4.1 대응 방안

Zhang 등의 인증 및 키 합의 방식의 안전성은 이상적인 tamper-resistant 디바이스에 전적으로 의존하고 있으므로 tamper-resistant 디바이스가 악의적인 공격에 노출될 경우 전체 시스템을 취약하게 만들 수 있다. 이에 대응 방안으로 스마트 미터안의 저장된 민감한 값들을 랜덤넘버와 XOR 연산하여 저장하고 기존의 스마트 미터와 서비스 제공자간의 2 party 인증에 사용자의 스마트 디바이스를 추가하여 3 party 인증으로 보안성을 향상 시키는 것을 제시한다. 따라서 공격자가 tamper-resistant 디바이스의 데이터와 공개된 채널로 전송되는 값들을 도청하더라도 스마트 미터의 민감한 정보들을 바로 얻을 수 없으므로 로그인 요청 메시지 및 세션 키를 계산 할 수 없다.

4.2 향후 연구

기존의 TTP 기반의 중앙집중식 시스템을 블록체인을 활용하여 데이터를 관리 및 교환하는 연구를 수행하고 이를 스마트 그리드 시스템에 접목하여 효율적이고 안전한 인증 및 키 합의 방식을 제안 할 수 있다.

V. 결론

본 논문에서는 Zhang 등이 제안한 스마트 그리드 환경에서 안전한 통신을 위한 인증 및 키 합의 방식의 안전성을 informal 분석으로 분석하였으며 제안된 방식이 단일 요소 안전성 문제, 스마트 미터 도난 및 위장 공격, 세션 키 노출 공격에 취약하고 상호 인증을 제공하지 않음을 보였다. 따라서 tamper-resistant 디바이스에 값을 저장할 때 랜덤넘버와 XOR 연산하고 기존의 스마트 미터와 서비스 제공자간의 2 party 인증을 사용자의 스마트 디바이스를 추가한 3 party 인증으로 변경하여 보안성을 향상하는 대응 방안을 제시하였다. 또한 블록체인을 활용하여 효율적인 데이터 교환 및 관리 기법에 대한 향후 연구를 통하여 TTP 없이 데이터를 관리하고 보안성 및 효율성을 향상할 수 있다. 이러한 대응 방안 및 향후 연구를 통하여 실제 스마트 그리드 환경에서 보다 효율적이고 안전한 인증 및 키 합의 방식을 제안 할 예정이다.

Acknowledgement

본 연구는 한국전자통신연구원 연구운영비지원사업의 일환으로 수행되었음. [19ZH1200, 데이터 안심사회를 위한 트러스트 데이터 커넥톰 핵심 원천 기술 개발]

[참고문헌]

[1] V.Odelu, A.K.Das, M.Wazid and M.Conti,

Provably secure authenticated key agreement scheme for smart grid. ***IEEE Trans. Smart Grid,*** 2016, pp. 1900-1910.

[2] Y.Chen, J-F.Martinze, P.Castillejo and L.Lopez, An anonymous authentication and key establish scheme for smart grid: FAuth, ***Energies***, 2017, pp. 1354-1374.

[3] L.Zhang, L.Zhao, S.Yin, C.Chi, R.Liu and Y.Zhang, A lightweight authentication scheme with privacy protection for smart grid communications, ***Future Generation Computer System***, 2019, pp. 770-778.

[4] Z.Liu, L.Xiong, T.Peng, D.Peng and H.Liang, A realistic distributed conditional privacy-preserving authentication scheme for vehicular ad hoc networks, ***IEEE Access***, 2018, pp.26307-26317.

[5] H.Zhong, S.Han, J.Cui, J.Zhang and Y.Xu, Privacy-preserving authentication scheme with full aggregation in VANET, ***Information Sciences***, 2019, pp.211-221.

Concepts on Improving Nuclear Control Inspection through Augmented Reality

Poe-il Park*

*Korea Institute of Nuclear non-proliferation And Control

Summary

The paper is written with a concern of limited augmented reality applications exists today which are to the field of education and games. Hence the author has twisted augmented reality concepts to those fields of nuclear control such that inspectors hopefully utilize cutting-edge technology of augmented reality one day.

Three different concepts of augmented reality application which can be applicable to inspection will be introduced. Even though this paper specifically discuss nuclear control field inspection, similar concepts of augmented reality application can be applied to various industry as well.

I. Introduction

Virtual Reality(VR) and Augmented Reality(AR) are one of most charming technology that arose from last few years. However the applications of VR is quite limited to field of education, movie and games which are more of consumption oriented way.

[Picture1. VR game application]



Similarly AR applications can be easily found through many different industries that are related to the consumer industry. For example, recent cell phones have AR features such that customers can AR their clothes, furnitures before the purchase.

[Picture2. AR phone application]



Indeed, it is time for AR to directly contribute to industry itself for example through AR improvement of safety or product efficiency can be achieved. Of course there are various products, Google Glass and Microsoft Holo-lens, that can be directly apply to industry. However, the actual applications are less and it needs to be boosted to accomplish VR usability.

Recently there has been a case where Boeing

is using AR to guide product line employees. In this application, employees don't need to look through the blueprint or keep there blueprint at their workplace. AR will show what colored line has to be inserted to such numbered hole. Boeing applied this to production of Boeing 737-900ER and they successfully reduced their assembly time over 25%[1]

[Picture3. Boeing airplane build AR application using Google Glass]



Likewise, AR application on Nuclear control inspection could be found and to be studied to improve efficiency of the working time. The application specifically bounded to those works done in field.

II. Possible applications

2.1 Different Nuclear Control Inspections

There are few different nuclear inspection areas; safeguard, physical protection and cyber security. Inspection is done through expert Korea Institute of Nuclear non-proliferation And Control inspectors or International Atomic Energy Agency inspectors.

Safeguard inspection is about nuclear material inspection at nuclear facilities or safeguarded facilities to prevent any misuse of nuclear material. Physical protection inspection is about inspecting the appropriate level of physical protection of nuclear facilities or safeguarded facilities. Likewise, Cyber security inspection is about inspecting the appropriate level of cyber security of nuclear plant, especially focused on electric components in nuclear plant.

2.2 The Goal of Inspection Applications

The born nature of AR is supporting not operating.(This is why AR is vigorously used in marketing, education and so) Hence, the goal of inspection applications for AR should be concentrated to supporting features that possibly reduce the burden that inspectors hold.

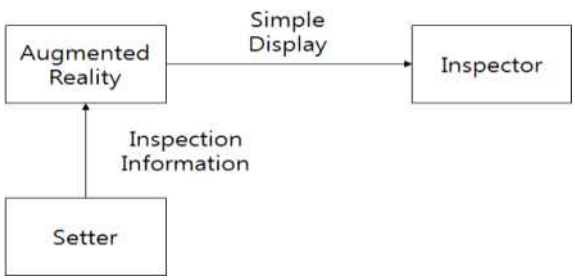
2.3 Different levels of AR applications

2.3.1 AR application level 1 : Simple Display

Easiest way and the simplest way of utilizing AR would be just simply display inspection data or inspection guideline to inspector through AR. Easy example of this level 1 application can displaying target number of spent fuel rod that need to be inspected or displaying Critical Assets through AR glasses.

However, the effectiveness of the level 1 applications can be doubted hence application can only depict limited information to inspectors which most skilled inspectors find them less worthy.

[Figure 1. Level 1 application process diagram]



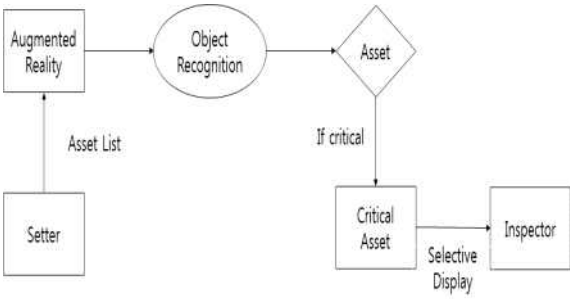
2.3.2 AR application level 2 : Case selective display

Level 2 application of AR would be case selective display of information. Explicit meaning of this word is showing information with respect to the inspector's current sight. For example, if the inspector is inspecting cyber security on certain nuclear plant, AR could selectively show critical asset information which inspector is currently inspecting. This could help inspector by informing what inspector might have missed while performing the inspection.

Level 2 application requires object recognition technology since AR need to distinguish different object especially critical assets, the assets that need to be inspected. Therefore, level 2 application need way more effort to develop than those of level 1. As a consequence, level 2 will be more effective than those of level 1.

However, as it is just level 2 application, it possess much more to be improved. What it could go further will be discussed at level 3 application.

[Figure 2. Level 2 application process diagram]



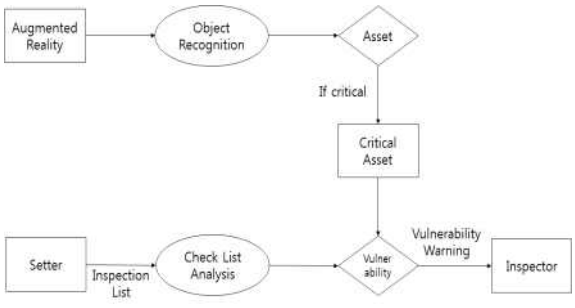
2.3.3 AR application level 3 : Be Inspector

We saw simple display and selective display from previous application. What is left is level 3 application which is 'Be inspector'. This level of application not only requires object recognition technology but data analysis as well.

The goal of level 3 application is AR being work as inspector actively. AR need to recognize critical asset and should be able to tell inspector what vulnerability it has. Then the inspector will manually recheck what AR called vulnerable. This might look like AI but this is far from AI rather close to utilizing matching tables. AR doesn't need to learn anything or make any judgement. AR will be informed by a setter what inspector need to inspect at certain site with certain critical assets and will cross check input table and what AR sees.

Level 3 definitely requires much more development effort than those of level 2 or level 1. We certainly can doubt possibility of level 3 development and real-life application productivity because possibility of mis-detective always exists and such behaviour will bother inspector.

[Figure 3. Level 3 application process diagram]



III. Conclusion

Even though virtual reality and augmented reality are uprising technology, there are no studies of AR applications on nuclear control

field, at least there were none found from RISS site.

Through this paper, three different concepts of AR applications were discussed.

Table 1. The keynote of exercise evaluation

Levels	Concept examples
1	Simple display of information
2	Case selective display
3	Vulnerability analysis

Since this is different concepts only, further implementation to improve AR applications on nuclear control need to be done.

[References]

[1]Chosun news, “There’s no need for blueprint, AR glass is enough”, (2019.08.19).

국내 전력 시스템에서의 SSL VPN 가용성 분석

오진혁, 이옥연
국민대학교 금융정보보안학과

Analysis of SSL VPN Availability in Domestic Power System

Jin-Hyeok Oh, Okyeon Yi
Department of Financial Information Security

요 약

최근 전력 시스템에 IoT(사물인터넷, Internet of Things) 기술을 접목하여 에너지 이용 효율을 높일 수 있는 융합기술이 빠르게 도입되고 있다. 이에 따라 기존의 ICT환경에서 존재하였던 보안 위협이 전력 시스템에서도 그대로 나타날 수 있다. 전력 시스템에서 사이버 공격이 발생할 시 천문학적인 경제적 손실이 발생할 수 있다. 하지만 무조건적인 보안기술의 적용은 전력 시스템의 오작동을 초래할 수 있으므로 적용하고자 하는 보안기술의 가용성을 분석하여 전력 시스템 서비스 환경에 맞는 보안기술을 적용하여야 한다. 따라서 본 논문에서는 국내 전력 시스템에서의 사이버보안 표준을 분석하여 보안 요구사항을 서술하고 이에 적용할 수 있는 SSL VPN의 가용성을 분석한다.

I. 서론

최근 전력의 생산 및 공급을 담당하는 국가기반시설에 IoT(Internet of Things, 사물인터넷) 기술을 접목하여 에너지 이용 효율을 높일 수 있는 융합기술이 빠르게 도입되고 있다. IoT 기반의 전력 시스템을 운영할 시 전력 사용 효율성 증가, 전력 품질 개선과 같은 장점이 존재한다. 하지만 기존에 폐쇄적으로 운영되어 사이버위협으로부터 안전한 전력 시스템이 이제는 인터넷과 연결되어 수많은 사이버위협이 존재하게 된다. 전력 시스템과 같은 국가기반시설에서 사이버 공격이 발생할 시 천문학적인 경제적 손실이 발생할 수 있으며 이는 국민의 생활에 직접적인 영향을 줄 수 있다. 이에 정부에서도 사이버보안의 중요성을 파악하여 대한민국 최초로 '국가사이버안보전략'의 수립을 통해 사이버공간에서의 보안을 중요하게 여기고 있다[1]. 하지만 무조건적인 보안기술의 적용은 전력 시스템의 오작동을 초래할 수 있으므로 적용하고자 하는 보안기술의 가용성을 분석하여 전력 시스템 서비스 환경에 맞는 보안기술을 적용하여야 한다. 따라서 본 논문에서는 국내 전력 시스템에서의 사이버보안 표준을 분석하여 보안 요구사항을 서술하고 이에 적용할 수 있는 SSL VPN의 가용성을 분석한다.

II. 국내 전력 시스템에서의 사이버보안 요구사항

2.1. KS C IEC 62351-Part 3

KS C IEC 62351 표준은 IEC 62351 표준을 기초로 기술적 내용 및 구성을 변경하지 않고 작성한 한국산업표준이다[2].

KS C IEC 62351-Part 3에서는 전력 시스템에서 데이터 전송 프로토콜로 TCP/IP를 사용할 때 TLS를 적용하여 안전한 보안 통신 채널을

구성하도록 기술되어 있다. 전력 시스템에서의 데이터 통신에 TLS를 적용함으로써 데이터에 대한 기밀성, 무결성 보장, 출처 인증 및 사용자 인증 기능을 제공할 수 있다. 따라서 전력 시스템에서 발생할 수 있는 Man-in-the-middle 공격, Replay 공격, Eavesdropping으로부터 보호할 수 있다. 해당 표준에서는 TLS 사용 시 cipher suite, 버전 협상, 세션 재개, 세션 재협상, 메시지 인증 코드, 인증서 지원 총 5가지의 필수 요구사항들이 기술되어 있다.

cipher suite의 필수 요구사항은 다음과 같다.

- 다른 수단으로 통신 암호화를 하지 못하는 경우 NULL cipher 사용을 금지한다.
- 다른 수단으로 통신 암호화 할 수 있는 경우 NULL cipher 사용을 할 수 있지만 종단간 인증 및 무결성을 보장하여야 한다.

버전 협상의 필수 요구사항은 다음과 같다.

- TLS v1.2 이상 지원되어야 하며 호환성을 보장하기 위해 TLS v1.0, v1.1 역시 지원하여야 한다.

세션 재개는 최대 마지막 세션 종료 후 24시간 이내에 가능하며 기존의 세션 ID를 기반으로 full handshake를 진행하지 않아도 되며 새로운 세션 키를 생성하여 세션을 재개한다. 세션 재협상은 RFC 5746에 정의된 메커니즘을 사용하고 메시지 인증 코드는 man-in-the-middle 공격을 방지하기 위해 사용되어야 한다.

인증서 지원의 필수 요구사항은 다음과 같다.

- 하나 이상의 CA를 지원하여야 한다.
- 인증서를 사용한 인증은 상호인증이어야 하며 만약 한쪽의 엔티티가 인증서를 제공하지 않으면 연결은 종료되어야 한다.
- 서명 알고리즘으로는 RSA와 DSS 알고리즘을 지원해야 한다. RSA-2048은 필수적으로 지원되어야 하며, RSA-1024의 지원은 선택 가능하다. ECDSA 또는 ECGDSA는 SHA-256 알고리즘과 사용되어야 하며 키 길이는 256비트 이상이어야 한다. ECDSA에 사용되는 커브는 secp256r1 커브이며 ECGDSA의 경우 ISO/IEC 15946-2에 정의되어 있는 규격을 따라야 한다.

한국인터넷진흥원의 암호 알고리즘 및 키 길이 이용 안내서에 의하면 현재 권고하는 보안강도는 112비트 이상이며, 해당 표준에서 권고하는 ECC-256, RSA-2048, RSA-1024의 보안강도는 각각 128비트, 112비트, 80비트로 RSA와 ECC의 보안강도가 서로 상이하다[3]. 따라서 해당 표준에서는 RSA-1024의 사용은 권장하지 않도록 수정되어야 하며, 해당 표준에서 권고하는 ECC-256과 RSA-2048의 보안강도가 상이하므로 해당 알고리즘들의 보안강도를 동일하게 적용해야 한다. 따라서 RSA-2048과 동일한 보안강도를 가지는 ECC-224의 지원 또는 ECC-256과 동일한 보안강도를 가지는 RSA-3072를 필수적으로 지원하도록 수정되어야 한다.

2.2. TTAK.KO-12.0182

해당 표준은 국내표준단체인 TTA에서 발간한 표준으로, 안전한 스마트 그리드 서비스 제공을 위한 기술적인 측면에서의 보안 요구사항을 제공하고 있는 표준이다[4]. 본 표준은 NISTIR 7628 표준을 참조한 표준으로 스마트 그리드 보안 모델, 스마트 그리드 공통 보안 요구사항, 스마트 그리드 구간별 보안 요구사항으로 구성되어 있다. 스마트 그리드 구성요소는 아래 그림과 같으며 스마트 그리드 공통 보안 요구사항은 아래 [표 1]과 같다.

스마트 그리드 사이버보안 요구사항
접근제어
식별 및 인증
침해 대응
미디어 보호
스마트 그리드 정보 시스템 개발 및 유지
스마트 그리드 정보 시스템 및 서비스 구입
스마트 그리드 정보 시스템 및 통신 보호
스마트 그리드 정보 시스템 및 무결성

[표 1] 스마트 그리드 사이버보안 요구사항

III. 국내 전력 시스템에서의 SSL VPN

국내 전력 시스템은 국가기반시설이므로 전자정부법 56조에 의해 전력 시스템에서의 정보보호를 위해 IT보안인증사무국에서 검증받은 검증필암호모듈을 의무적으로 사용하여야 한다. 즉, 국내 전력 시스템에서의 사이버보안을 위해 본 논문의 2장에서 명시된 보안 요구사항을 만족시키면서 검증필암호모듈을 사용한 보안기술이 적용되어야 한다. 이에 검증필암호모듈을 연동한 SSL VPN 기술을 도입할 수 있다. SSL VPN은 TLS의 Crypto 라이브러리를 호출하여 암호 알고리즘들을 사용하는데 이때 Cipher suite에 검증필암호모듈의 보호함수 목록에 존재하는 암호 알고리즘을 추가하여 해당 암호 알고리즘 호출 시 검증필암호모듈을 사용하도록 구성하게 되면 검증필암호모듈을 연동하여 SSL VPN을 사용할 수 있다. 또한 SSL VPN은 접근제어, 인증서 기반의 상호인증, 기밀성, 무결성 보장, 재전송공격 방지, 송수신 분석 보호, 보안 로그 관리, 안전한 키 관리 기능을 제공할 수 있으며 이는 2장에서 명시된 보안 요구사항을 만족하므로 국내 전력 시스템에 적용할 수 있는 보안기술이 될 수 있다[5]. 본 장의 세부 절에서는 SSL VPN 성능분석을 위한 실험 환경, 실험 결과를 제시한다.

3.1. 실험 환경

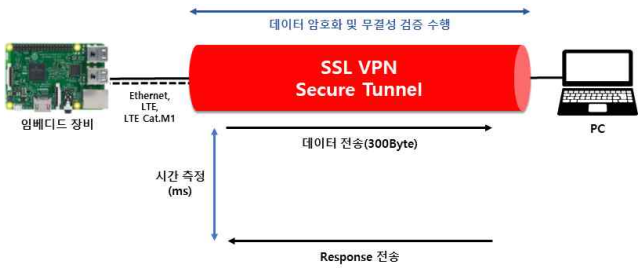
전력 시스템에서는 전력계통 장비와 전력계통 장비를 제어하고 전력 데이터를 수집 및 관리하는 서버와의 통신이 이뤄진다. 이때 사용되는 통신기술은 따로 규격화되어 있지 않으므로 Ethernet, LTE, LTE Cat.M1 등과 같이 다양한 통신이 사용될 수 있다. 따라서 본 실험에서는 전력 시스템과 유사한 환경을 구성하기 위해 임베디드 장비와 PC를 사용하여 유사한 환경을 구성하였으며, 임베디드 장비에서 서버와 통신하기 위해 Ethernet, LTE, LTE Cat.M1 통신기술을 사용하여 임베디드 장비에서 SSL VPN의 성능을 측정하였다. 실험에 사용한 임베디드 장비와 PC의 성능은 아래 표와 같으며, 실험 방법은 아래 [그림 1]과 같다.

구분	PC 사양
CPU	Intel Celeron Processor N3160 1.60GHz
Memory	4GB DDR3 1600MHz PC3 12800 x 2
OS	Ubuntu 16.04.6 LTS

[표 2] PC 사양

구분	임베디드 장비 사양
CPU	ARM Cortex-A53 1.20GHz 64-bit
Memory	SDRAM 1GB
OS	Raspbian Stretch (Linux Kernel 4.14.79)

[표 3] 임베디드 장비 사양



[그림 1] SSL VPN 성능 측정 방법

임베디드 장비에서 PC에게 300byte 크기의 데이터를 송신한 후 그에 대한 PC의 응답을 수신할 때까지의 걸린 시간을 측정하였으며 SSL VPN을 적용하였을 때와 SSL VPN을 적용하지 않았을 때의 동작 시간을 측정하였다. 전력 시스템에서 송수신되는 계측 데이터의 크기는 약 300byte 이하의 저용량의 데이터이므로 실험에서 송수신되는 데이터의 크기는 300byte로 고정하여 실험을 진행하였다[6]. 실험 결과의 정확성을 위해 10,000번의 실험을 진행하여 평균 동작 시간을 측정하였다.

본 실험에서는 OpenVPN과 mbedTLS 라이브러리를 사용한 SSL VPN을 사용하였으며 TLS버전은 1.2를 사용하였다. SSL VPN과 검증필암호모듈의 연동을 위해 TLS Cipher suite에 검증필암호모듈의 보호함수 목록을 추가하였고 KMULiB v2.0 (CM-111-2021.9) 검증필암호모

듈을 사용하였다. SSL VPN에서는 RSA-2048 기반 인증서를 사용한 상호인증 및 DH-2048 알고리즘을 사용한 키 일치를 하여 미리 Secure Tunnel을 형성한 뒤 실험을 진행하였다. 데이터 암호화에는 ARIA-128-CBC, 데이터 무결성 검증에는 SHA-256 알고리즘을 사용하였다. 본 실험에서는 상호 인증 및 키 일치 과정의 동작 시간은 측정하지 않고 데이터 암호화 및 무결성 검증을 수행하여 통신할 때의 시간을 측정하였다.

3.1. 실험 결과

아래 [표 4]는 다양한 통신기술에 따른 SSL VPN을 적용하였을 때와 SSL VPN을 적용하지 않았을 때의 데이터 송수신 동작 시간 및 성능 비교를 정리한 표이다.

구분	SSL VPN 적용	SSL VPN 비적용	성능 저하율
Ethernet	1.744ms	1.535ms	약 13%
LTE	51.647ms	45.257ms	약 14%
LTE Cat.M1	199.121ms	178.326ms	약 11%

[표 4] SSL VPN 적용/비적용 데이터 송수신 동작 시간 비교

Ethernet, LTE, LTE Cat.M1에서의 SSL VPN을 적용하였을 때의 상황이 SSL VPN을 적용하지 않았을 때의 상황 대비 성능 저하율은 대략 11~13%이다. 성능 저하율 지표만 보았을 때는 SSL VPN을 적용 시 서비스에 영향이 있다고 판단될 수 있다. 하지만 동작 시간의 차이는 Ethernet에서 약 0.209ms, LTE에서 6.39ms, LTE Cat.M1에서 20.795ms로 측정되었다. 초저지연 서비스를 지향하고 있는 5G의 latency가 1ms인 것을 고려하면 Ethernet에서 약 0.209ms의 차이는 전력 시스템에서 SSL VPN의 적용이 미치는 영향이 미비한 것으로 판단된다. 또한 LTE와 LTE Cat.M1의 latency가 각각 약 25ms, 15ms인 것을 고려하였을 때 LTE에서의 약 6.39ms, LTE Cat.M1에서의 약 20.795ms의 차이 역시 전력 시스템에서 SSL VPN 적용이 미치는 영향이 미비한 것으로 판단된다.

IV. 결론

현재 전력 시스템에 IoT 기술을 기반한 융합기술이 빠르게 도입되고 있다. 이에 전력 시스템에서의 사이버 공격이 발생할 시 천문학적인 경제적인 피해가 일어날 수 있으므로 이에 대한 보안 적용이 시급한 상황이다. 하지만 국내 전력 시스템에서 적용할 수 있는 보안 규격안은 제정되어 있지 않은 상황이므로 무분별한 보안기술의 적용으로 인해 시스템의 오작동을 초래할 수 있다. 따라서 전력 시스템에서의 사이버보안 요구사항은 만족시키면서 보안기술의 가용성을 분석하여 본래의 시스템에 악영향은 미치지 않는 보안기술의 적용이 필요한 상황이다.

본 논문에서는 국내 전력 시스템에서의 사이버보안 표준을 분석하여 전력 시스템에서의 보안 요구사항을 서술하였다. 그리고 전력 시스템에서 해당 보안 요구사항을 만족하는 검증필 암호모듈을 연동한 SSL VPN의 가용성을 분석하여 전력 시스템에서의 SSL VPN 적용이 본래의 시스템에 미치는 영향이 미비하다는 결론을 맺었다.

[참고문헌]

[1] 대한민국 청와대, “국가사이버안보전략”, 2019.04

[2] 산업표준심의회, “KS C IEC 62351”, 2018.10

[3] 한국인터넷진흥원, “암호 알고리즘 및 키 길이 이용 안내서”, 2013.01

[4] TTA, “스마트 그리드 보안 요구사항”, 2011.12

[5] NIST, “NIST Special Publication 800-113, Guide to SSL VPNs“, 2008.07

[6] 한주홍, 오진혁, 이재훈, 윤승환, 이인호, 이옥연, “에너지 인터넷 환경에서의 보안 설계 및 실험”, 2019.06

내부자 위협에 의한 정보유출 방지를 위한 기술적, 관리적 보안대책에 관한 연구

변예은*

*한국원자력통제기술원

A Research on Technical and Management Security Measures to Prevent Information Leakage by Insider Threat

Ye-Eun Byun*

*Korea Institute of Nuclear Nonproliferation and Control

요 약

다양한 분야에서 디지털화가 급속도로 진행되면서 국내외에서 정보보안 사고도 끊임없이 일어나고 있고, 그중에서도 내부자에 의한 사고가 높은 비율을 차지하고 있다. 이에 많은 기업에서는 내부자에 의한 정보유출에 대응하기 위하여 각 기업의 상황에 맞게 기술적, 관리적 보안대책을 운용하고 있다. 본 논문에서는 해당 보안대책들에 어떠한 것들이 있는지 알아보고, 본 기술원에서 내부자에 의한 정보유출 사고에 대응하기 위해 적용할 수 있는 보안대책은 어떠한 것이 있는지 알아보고자 한다.

I. 서론

2014년 카드사 개인정보 유출, 2012년 대기업 첨단기술 유출, 2011년 금융업체 개인정보 유출 등 국내외에서 내부자에 의한 정보유출 사고가 빈번하게 발생하고 있다. 또한, 포스포인트 조사에 따르면 보안사고의 90%가 내부에서 발생한다고 한다. 이에 많은 기업에서는 내부자에 의한 정보유출에 대응하기 위해 기업 내에 존재하는 내부자 위협의 종류를 파악하고, 이에 대응하기 위한 여러 정책을 마련하고 있으며, 이를 통하여 기술적, 관리적 보안대책을 적용하여 운용하고 있다. 뿐만 아니라 내부 직원들의 보안 인식 제고 또한, 내부자 위협에 대응하기 위한 중요한 요소로 인식하여 다양한 활동을 수행하고 있다.

이에 본 논문에서는 우선 내부자 위협(Insider Threat)의 정의에 대해 살펴보고, 내부자 위협에 의한 정보유출 사고를 방지하기 위한 기술적, 관리적 보안대책에 대해 자세히 알아보고자 한다. 이를 통해 현재 기술원에서 적용할 수 있는 방안들을 살펴보고자 한다.

II. 내부자 위협 개요

1.1 내부자 위협 정의

최근 개인정보, 첨단기술 등 기업에서의 정보유출 사고가 빈번하게 일어나고 있으며, 많은 사고가 내부자를 통해서 일어났다. 미국 국토안보부에서는 내부자 위협(Insider Threat)을 현재 또는 이전 직원, 협력업체 또는 회사의 네트워크, 시스템, 데이터에 승인받은 접근 권한을 가지거나(혹은 가졌거나) 고의로 접근 권한을 악용하여 조직의 정보나 정보시스템의 기밀성, 무결성 가용성에 악영향을 미칠 수 있는 사업 파트너로 정의하고 있다. 즉, 내부 직원뿐만 아니라 내부 직원과 연계가 될 수 있는 사람, 접근 권한 등이 모두 내부자 위협이 될 수 있는 것이다. 따라서, 각 기업에서는 어떠한 종류의 내부자 위협이 기업에 존재할 수 있는지를 파악하여 대응하는 것이 필요하다. 다음으로는 내부자 위협에 대응하기 위해 적용할 수 있는 기술적, 관리적 보안대책에 대해 살펴보고자 한다.

1.2 내부자 위협에 대응하기 위한 기술적, 관

리적 방안

미국 국토안보부에서는 내부자 위협에 대응 및 예방을 위한 기술적, 관리적 방법에 대해 다음과 같이 9가지로 분류하고 있다.

· 자료 및 파일 암호화
· 접근 관리(IAM)
· 자료 접근 모니터링
· 자료 접근 제어
· SIEM 및 기타 로그 분석
· 침입 탐지 및 방지 시스템(IDS/IPS)
· 자료 손실 방지(DLP)
· 디지털 권리 관리 솔루션
· Data redaction

각 보안대책에 대해서는 다음 장에서 상세히 알아보고자 한다.

III. 내부자 위협에 대응하기 위한 기술적, 관리적 보안대책

1.1 자료 및 파일 암호화

기업에서는 사용자들이 기업 내부 서버 등에 보안 접속을 할 수 있도록 메일 및 통신 암호화, 키보드 암호화 솔루션 등을 설치해야 하며, 내부 직원에 의한 중요 자료 유출 방지를 위해 데이터베이스 암호화 및 파일, 디스크 암호화 솔루션을 설치해야 한다.

1.3 접근 관리(IAM)

접근 관리(IAM, Identify and Access Management)는 식별 관리(Identify Management)와 접근 관리(Access Management)을 합친 것으로, 계정과 접근 권한을 관리한다는 개념이다. 즉, 조직 내의 직원이 권한 내의 접근을 하는지를 확인하기 위한 것으로 볼 수 있다. 많은 조직에서는 직원별로 시스템에 접근할 수 있는 권한에 차등을 두어 운영하고 있는 것이 접근 관리의 대표적인 운영 방식이라고 볼 수 있다.

1.4 자료 접근 모니터링

자료 접근 모니터링은 사용자가 시스템에 접근한 로그, 사용자가 자료에 접근하여 활용한 로그 등을 포함하여 데이터베이스 관리시스템 등에 대한 전반적인 모니터링을 포함하는 활동이라 할 수 있다.

1.5 SIEM 및 기타 로그 분석

SIEM(Security Information and Event Management)는 네트워크와 보안 장비로부터 정보를 모으고 분석하여 제시하는 시스템으로, 관리 프로그램의 식별과 접근, 취약점 관리, 운영 체제, 데이터베이스, 프로그램 로그, 그리고 외부위협을 포함하는 것이다. 조직 내에서 발생하는 모든 자원의 정보 및 보안 이벤트를 통합 관리하는 것이라고 할 수 있다.

1.6 침입 탐지 및 방지 시스템(IDS/IPS)

침입 탐지 시스템(IDS, Intrusion Detection System)이란 네트워크 환경에서 발생하는 불법 침입 및 시스템의 자원을 정당한 권한 없이 불법적으로 사용하는 시도나, 내부 사용자가 자신의 권한을 오·남용하는 시도를 탐지하고 대응하며 관리자에게 통보하는 것을 목적으로 하는 소프트웨어나 하드웨어를 의미한다. 또한, 침입 방지 시스템(IPS, Intrusion Prevention System)은 기존 침입 탐지 시스템의 실시간 침입 차단 기능이 없고 공격 트래픽과 정상 트래픽을 구별하는 데 어려움이 따라 오탐지와 미탐지 문제가 발생하는 한계점을 극복하기 위해 개발된 시스템이다.

1.7 자료 손실 방지(DLP)

자료 손실 방지(DLP, Data Loss Prevention)은 조직 구성원, 프로세스, 기술의 결합을 통해 조직의 주요 정보가 유출되는 것을 방지하는 솔루션이다. 자료 손실은 외부로부터 악의적인 공격보다는 조직 구성원의 부주의, 조직 프로세스 위반, 내부자의 의도적인 유출 등으로 주로 발생함에 따라 DLP 솔루션에 대한 관심이 증가하고 있다.

1.8 디지털 권리 관리 솔루션

디지털 권리 관리(Digital Right Management)는 출판사 또는 저작권자의 디지털 컨텐츠를 보호하고 불법적인 이용을 막기 위해 활용하는 모든 기술과 서비스를 일컫는다.

디지털 컨텐츠의 복제와 배포가 쉬워진 환경에서 저작권자는 자신이 허가한 상황에서만 디지털 컨텐츠가 이용되도록 하고자 하는 것이 디지털 권리 관리의 목적이다.

1.9 Data redaction

Data redaction의 개념은 주로 개인정보를 취급하는 시스템에서는 필수적으로 사용하고 있는 기술로, 중요 데이터들에 대하여 마스킹 처리 등을 하여 운영하는 것을 의미한다.

IV. 기술원 적용 필요에 대한 고찰

앞서 미국 국토안보부에서 제시한 내부자 위협에 대응하기 위한 9가지의 기술적, 관리적 보안대책에 대해 살펴보았고 이를 토대로 기술원에 적용이 필요한 부분에 대해 살펴보고자 한다. 우선, 기술원에서는 출장이 많은 기관 업무 특성상, 업무 자료를 노트북, 보안 USB 등에 저장하여 다니는 일이 빈번하므로 해당 자료 및 파일에 대하여 암호화 솔루션을 적용할 필요가 있다. 이를 통해, 분실 등의 이유로 인해 파일이 유출되는 사고가 발생하더라도 암호화된 자료에 복호화가 불가능하므로 정보유출 사고를 방지할 수 있을 것이다. 또한, 시스템별로 접근 권한에 차등을 두어 불필요한 접근 권한을 부여하는 것을 방지하는 것도 원내 정보에 대한 접근 횟수를 줄이므로 사고 방지에 효과가 있을 것이다.

또한, 업무 소통을 위해 보조기억매체, 이메일, 메신저 등 다양한 정보 전달 경로가 존재한다. 이는 정보유출 경로도 다양해졌음을 의미하며, 이러한 경로를 통해 전달되는 자료의 종류를 확인하는 솔루션을 통해 내부 중요 자료의 전달 경로 파악 및 향후 사고 발생 시, 유출 경로 추적에도 도움이 될 것이다. 또한, 네트워크 보안을 위해 운영하는 여러가지 보안장비에 대한 모니터링을 강화함으로써 외부에서의 침입 시도에 대해 탐지 및 대응을 할 수 있을 뿐만 아니라 내부에서의 비정상 행위 등에 대해 탐지하여 내부자 위협에 대응할 수 있는 기본적인 수단이 될 수 있을 것이다.

V. 결론

본 논문에서는 최근 늘어나고 있는 개인정보, 산업기밀 등의 유출 사고가 내부자 위협에 의해 발생하고 있다는 점에 주목하여 본 기술원에서도 내부자 위협에 대응할 수 있는 방안에 대해 고찰해보고자 하였다. 이를 위하여, 먼저 내부자 위협의 정의에 대해 알아본 후, 미국 국토안보부에서 제시한 내부자 위협에 대응하기 위한 9가지의 기술적, 관리적 방법에 대해 자세히 알아보았다. 마지막으로 기술원에서 적용이 필요한 보안대책에 대해 살펴보았다.

원내 망분리가 구축되어 있고 보안장비들로 네트워크에 대한 보안을 강화하고 있지만, 지속적으로 증가하고 있는 사이버 공격에 대응하기 위해서는 내부자 위협에 대응하여 정보유출 사고 등을 방지하는 것도 원내 정보보안 강화를 위해 고려하여야 할 부분이다. 이를 위해서는 자료 암호화 솔루션 적용, 시스템별 접근 권한 차등 부여, 자료 손실 방지 솔루션 적용, 모니터링 강화 등 다양한 기술적 방안이 적용되어야 함과 동시에 원내 사용자들의 인식 또한 매우 중요한 부분이다. 본인이 인식하지 못하는 사이에 내부에 접근하려는 외부의 위협에 협조할 수 있는 내부자 위협이 될 수 있는 존재임을 인식하여 보안대책들을 준수하는 보안 활동이 매우 중요하다고 할 수 있다.

[참고문헌]

[1] US-CERT, Combating the Insider Threat, 2014

[2] KAIST 사이버보안연구센터, 내부자 사이버 위협 방어대책 연구, 2016

[3] 한국인터넷진흥원, 암호이용 안내서, 2010

[4] 한국정보통신기술협회, 네트워크 보안관리 지침, 2006

[5] 정보통신산업진흥원, 내부정보 유출방지(DLP) 기술 동향, 2009

[6] 국립중앙도서관 도서관연구소, 도서관용어해설, 2010

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 C

딥러닝 기술을 활용한 S-box 구조 식별 방법

김동훈*, 김성겸*, 홍득조**, 성재철***, 홍석희*

*고려대학교 정보보호학과, **전북대학교 IT정보공학과, ***서울시립대학교 수학과

An Approach to Identify S-box Structure using Deep Learning

Donghoon Kim*, Seonggyeom Kim*, Deukjo Hong**,
Jaechul Sung***, Seokhie Hong*

*Graduate School of Information Security, Korea University,

**Department of Information Technology, Chunbuk National University,

***Department of Mathematics, University of Seoul.

요 약

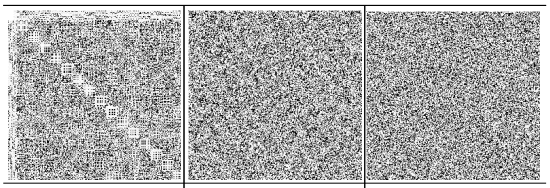
CRYPTO 2019에 발표된 Gohr의 연구결과는 딥러닝 기술이 암호설계 및 분석에 응용될 수 있음을 보여주었다. 본 논문에서는 특정 S-box로 확장된 S-box를 구성한 경우, 딥러닝 기술을 이용하여 S-box의 설계구조를 식별할 수 있는지를 실험한 결과를 제시한다. 이를 위해, 경량암호 설계에 주로 사용하는 *Feistel* 및 *MISTY* 구조를 가진 S-box의 DDT 및 LAT로 학습 데이터를 구성하고 딥러닝 알고리즘으로 식별하는 실험을 수행한 결과, 이론적으로 제시된 라운드 수 보다 많은 라운드 수에서 S-box의 구조를 식별할 수 있음을 확인하였다. 군사용 등 고도의 기밀성 유지를 위해 사용되는 암호들은 공격이나 해독을 근본적으로 차단하기 위해 설계 정보를 공개하지 않는 것이 원칙이다. 본 논문에서 제시된 방법은 딥러닝 기술이 이처럼 공개되지 않은 설계정보를 분석하는 하나의 도구로 사용 가능하다는 것을 보여준다.

I. 서론

암호는 Kerckhoffs 원리에 따라 설계원리가 공개되어도 안전하게 구현되어 오직 키를 통하여서만 암호 안전성을 보장받도록 한다. 그러나, 군이나 국가기관에서 사용하는 암호는 해독을 방지하기 위해 설계정보를 일체 미공개 하고 있다. 이러한 암호를 분석하기 위한 역공학 등을 수행하기 위해서는 암호 구조와 같은 초기정보 획득이 중요하다.

블록암호의 구성요소 중 하나인 S-box는 치환(Substitution)을 통하여 입력값을 특정 출력값으로 변경시켜 비선형적 성질을 만든다. 한편, NSA의 DES 및 Skipjack의 경우, 비선형 부분을 나타내는 S-box의 구체적인 설계원리를 미공개하여 학계에서는 이를 분석하기 위한 연구가 진행되어왔다.

Biruykov 및 Perrin[1]은 Skipjack S-box 설계원리 복원을 위해 다양한 방법을 사용하였다. 그 중 Pollock's Pattern 기법은 DDT 및 LAT를 이미지화하여 육안으로 한번에 식별가능하도록 구성한 방법이다. 특히, 이 방법으로 3라운드 *Feistel*구조 DDT를 [그림1]과 같이 나타내어 살펴보면 3라운드 *Feistel*에서는 *random*함수와 육안으로 구별되게 보이지만 5라운드에서는 구별이 되지 않는 것을 알 수 있다. 한편,



3R *Feistel* | 5R *Feistel* | *random*함수

[그림 1] DDT's Pollock's Pattern

딥러닝 기술은 육안으로 식별되지 않는 특징점들을 잘 식별할 수 있는 장점을 가지고 있다. 이에, 본 논문에서는 경량암호에 많이 사용되는 *Feistel* 및 *MISTY* 구조의 확장된 S-box에 대하여, 딥러닝 기술이 S-box의 구조를 식별할 수 있는지를 실험하였다. 이를 위해 *Feistel* 및 *MISTY* 구조를 가진 S-box의 DDT 및 LAT에 대한 학습데이터를 구성하고, 딥러닝 알고리즘을 적용하여 학습을 시킨 결과 7라운드에서 각 구조를 구별하였다. 본 연구결과는 딥러닝 기술을 이용하여 암호알고리즘의 공개되지 않은 설계정보를 분석하는 것이 가능하다는 것을 보여준다.

본 논문은 구성은 다음과 같다. Ⅱ장에서 확장 S-box 설계원리와 DDT, LAT에 대해서 소개하였으며, 딥러닝 기술을 기술하였다. Ⅲ장에서는 실험에 필요한 데이터 셋 구성 방법과 실험환경 및 딥러닝 네트워크 구성을 소개하였으며, 실험 결과를 Ⅳ장에서 분석해보았다. 마지막 장에서는 본 논문의 결론을 도출하였다.

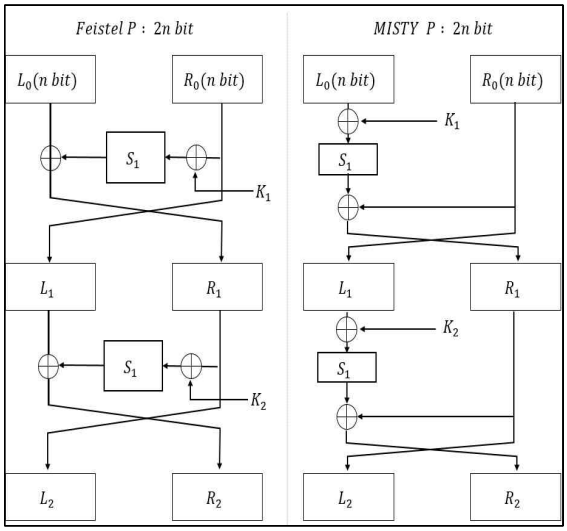
* 본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임. (No.2017-0-00520 (ICT 기초연구실) SCR-Friendly 대칭키 암호 및 응용모드 개발)

II. 배경지식

2.1 Feistel 기반 확장 S-box

Robin, CRYPTON schem의 확장 S-box에서 사용된 *Feistel* 구조는 [그림2]에서 보는 것처럼 $2n$bit 입력값을 좌・우 $n$bit씩 분리하고 F함수로 불리는 연산을 수행한 후, 라운드를 지날 때마다 좌・우 값을 교환한다. 1라운드를 지나면 $R_1 = S_1(R_0) \oplus L_0$, $L_1 = R_0$ 값을 가진다. R라운드 *Feistel*로 확장한 S-box는 $\mathcal{F}^R$로 표현하며, F함수는 $s^r (1 \le r \le R)$로 표현하고 S-box 1개만 사용한다. 본 논문에서는, $n = 4$로 설정하여 4-bit 내부 S-box를 통해 8bit로 확장된 모델을 사용하였다.

또한, Ruby and Rackoff[2] 및 Patarin[3]에 의해 *Feistel* 구조는 이론적으로 6라운드 이상에서 *random*한 함수와 구별이 어려운 것으로 알려져있다.



[그림2] Feistel & MISTY structure

2.2. MISTY 기반 확장 S-box

Fantomas schem의 확장 S-box에 사용된 *MISTY* 구조는 [그림 2]처럼 *Feistel*과 동일하게 $2n$비트 입력 후, n비트씩 나누어지며 *Feistel*과 달리 좌・우를 번갈아 가면서 F함수에 입력되는 구조를 가진다. 따라서 1라운드에서는 *Feistel*과 동일한 형태를 가지지만 2라운드부터는 다른 형태를 보인다. 아래에는 *Feistel*과 *MISTY*의 2라운드 값이다.

$$Feistel: L_2 = S_1(R_0) \oplus L_0$$
$$R_2 = S_1(S_1(R_0) \oplus L_0) \oplus R_0$$
$$MISTY: L_2 = S_1(L_0) \oplus R_0$$
$$R_2 = S_1(R_0) \oplus S_1(L_0) \oplus R_0$$

R라운드 *MISTY*로 확장한 S-box는 $\mathcal{M}^R$로 표현하며, 내부 S-box는 *Feistel*과 동일한 s^r로 표현한다. *MISTY* 구조도 Gilbert and Minier[4]에 의해서 4라운드부터 *random*한 함수와 구별이 어려운 것이 알려져 있다.

2.3. DDT・LAT

S-box의 DDT는 차분분석[5]을 위해서 필수적으로 구성해야 하는 가장 기본적인 요건으로, 입력쌍의 차분과 대응하는 출력쌍의 차분을 구한 후, mapping되는 차분 값들을 나타낸 테이블이다. $s: \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$일 때, $2^{2n} \times 2^{2n}$크기 테이블의 계수 $d_{i,j}$를 구하는 방법은 아래 식과 같다.

$$d_{i,j} = \#\ \{x \in \{0,1\}^{2n} | s(x \oplus i) \oplus s(x) = j\}$$

또한, LAT는 선형분석[6]을 위해서 구성해야 하는 기본적인 요건이다. 선형분석은 S-box 사이의 특정 입・출력 관계를 통해 선형근사식을 구성하고 이러한 관계를 통해 키를 추측하는 분석방법이다. 이 때, 선형식을 구성하는 입・출력 마스킹 사이의 관계가 기준점으로부터 얼마나 차이가 나는지를 계산하여 나타낸 표가 LAT이다. $2^{2n} \times 2^{2n}$ matrix의 계수값 $c_{i,j}$를 구하는 식은 아래와 같다.

$$c_{i,j} = \#\ \{x \in \{0,1\}^{2n} | x \cdot i = s(x) \cdot j\} - 2^{2n-1}$$

2.4. 딥 러닝을 활용한 암호분석

딥 러닝 기술은 컴퓨터가 학습 과정을 통해서 특정 데이터 집단의 숨겨진 패턴을 파악하고 데이터를 분류하는 기술이다. 최근에는 GPU 기술의 발전으로 병렬처리가 가능하게 되어 많은 양의 데이터를 학습하고 분류할 수 있게 되었다. 암호분석에서는 머신러닝을 이용하여 암호문을 통하여 암호알고리즘의 종류를 구분하는 연구들이 수행되었으나 실제 높은 수준의 정확도를 보여주지 못하였다.[7] 최근에는 딥러닝 기술이 발전하면서 부채널 분야에서 다양한 방법으로 활용되고 있으며[8], CRYPTO 2019에서 Ahron Gohr가 딥러닝을 이용한 Speck 32/64 공격 논문을 발표[9]하는 등 암호분석에도 활용되고 있다.

III. 실험 방법

3.1 dataset 준비

실험은 8-bit $\mathcal{F}^R$,$\mathcal{M}^R$의 R값을 3에서 8까지 증가시키며 DDT 및 LAT를 구성하고, 내부 4-bit S-box s^r은 3가지 종류로 구성하였다. 첫째, 같은 S-box를 사용한 경우로 $\mathcal{F}_e^R$, $\mathcal{M}_e^R$으로 표현한다. 두 번째는 동일 S-box에 라운드 키를 포함한 경우로 $\mathcal{F}_k^R$, $\mathcal{M}_k^R$으로 표현하며, 세 번째로는 서로 다른 S-box를 사용한 경우로 $\mathcal{F}_d^R$, $\mathcal{M}_d^R$로 표현한다. 또한, 대조군으로는 *random*한 함수를 사용하였으며 $\mathcal{R}$로 표기한다.

학습을 위한 입력 데이터는 $\mathcal{F}_e^R$ vs $\mathcal{M}_e^R$ vs $\mathcal{R}$, $\mathcal{F}_k^R$ vs $\mathcal{M}_k^R$ vs $\mathcal{R}$, $\mathcal{F}_d^R$ vs $\mathcal{M}_d^R$ vs $\mathcal{R}$ 3가지 상황에 대해서 3~8 라운드 별로 총 18개의 상황을 가정하였다. 또한, DDT($\mathcal{D}_D$), LAT($\mathcal{D}_L$)와, 두 개의 테이블을 합친 DDT&LAT ($\mathcal{D}_{D\|L}$)까지 총 3개의 테이블 구별자를 통하여 54가지 상황을 실험하였다. 본 논문에서 $\mathcal{D}_{D\|L}$를 사용하는 이유는 딥러닝 모델이 특징점을 잘 파악하기 때문에 두 테이블을 합하여 하나로 구성할 경우 각 테이블의 장점을 반영하여 훨씬 더 좋은 구별자 역할을 할 것으로 가정했기

때문이다. 입력 데이터는 $\mathcal{F}^R$,$\mathcal{M}^R$, $\mathcal{R}$ 구조별로 1,000개씩 총 3,000개를 생성하고 각 구별자들이 *Feistel*·*MISTY*·*random*을 구분하는지 딥러닝 모델을 이용하여 학습시킨다. 훈련 데이터는 입력 데이터의 50%인 1,500개를, 검증 데이터는 30%인 900개를, test 데이터는 20%인 600개를 사용하였다. 또한, 딥러닝 모델이 테이블을 잘 인식하도록 각 테이블을 계수의 최고값으로 나누어 scaling하였다.

단위 : 개

구 분	$\mathcal{D}_D$	$\mathcal{D}_L$	$\mathcal{D}_{D\|\|L}$
$\mathcal{F}_e^3$ vs $\mathcal{M}_e^3$ vs $\mathcal{R}$	*input data* : 3,000(*each* 1,000) *train* : 1,500(*each* 500) *validation* : 900(*each* 300) *test* : 600(*each* 200)		
$\mathcal{F}_k^3$ vs $\mathcal{M}_k^3$ vs $\mathcal{R}$			
$\mathcal{F}_d^3$ vs $\mathcal{M}_d^3$ vs $\mathcal{R}$			
$\mathcal{F}_e^4$ vs $\mathcal{M}_e^4$ vs $\mathcal{R}$			
…			
$\mathcal{F}_d^8$ vs $\mathcal{M}_d^8$ vs $\mathcal{R}$			

[표1] Proprecessing input data

3.2. 실험환경

실험에는 NVIDIA GeForce GTX 1660 Ti(6GB) GPU, Intel Core i7-9700 CPU와 32GB RAM을 탑재한 PC 1대로 사용하였다. 1개의 상황당 학습을 수행하는데 약 1.2GB의 GPU 메모리를 사용하였으며 학습에는 5분이 소요되었다.

3.3. 모델 구성

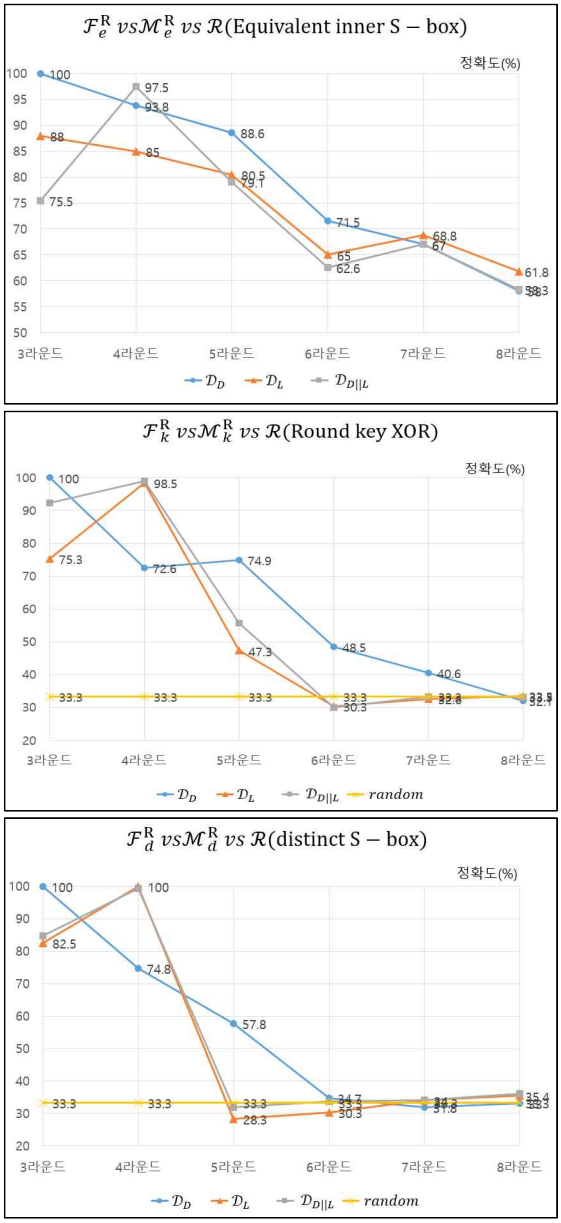
딥러닝 프로그램으로는 keras를 사용하였으며, DDT 및 LAT가 256 × 256 Matrix로 입력되기 때문에 2차원 데이터 학습에 좋은 Conv2D(Convolution Neural Network)를 5개층으로 구성하였다. 각 층 마다 Maxpooling과 과적합을 최소화하기 위해 dropout을 사용하였다. 입력된 데이터는 Shuffle함수를 통해 *random*과 *Feistel*·*MISTY* 구조를 섞어서 동일한 구조가 연속으로 학습되거나 평가되어 결과에 영향을 미치는 상황을 최소화시켰다. Epoch는 30, batch size는 10으로 설정하였으며, optimizer는 RMSprop을 사용하였다. loss function은 분류에 적합한 categorical cross entropy를 사용하였다.

Ⅳ. 실험 결과

4.1. 데이터 이미지화를 통한 식별

우선 DDT와 LAT로 표현한 데이터를 육안으로 한눈에 식별해보기 위해서 Pollock's Pattern 방법으로 이미지화 해보았다. 이미지는 테이블의 계수 값이 존재할 경우 흰색으로 표시되며, 계수 값이 0일 경우에는 흑색으로 표시하여 계수의 존재 여부에 따른 차이가 명확히 드러나도록 구성하였다. [그림1]에서 $\mathcal{F}_e^3$의 경우를 살펴보면 육안으로도 식별될 만큼 선명한 패턴이 나타났다. 이는, *Feistel*구조가 bijective한 함수일 경우 5라운드까지 불능차분이 나타나기 때문이다. 한편, *Feistel*은 6라운드 이상, *MISTY*에서는 4라운드 이상부터 *random*함수와 구별 가능한 패턴들이 식별되지 않았다.

4.2. 실험결과



[그림 3] Result Identification S-box Structure

실험결과는 $\mathcal{F}^R$ vs $\mathcal{M}^R$ vs $\mathcal{R}$을 구분할 확률이 33.3%일 때, Test data의 정답을 딥러닝 모델이 맞추는 정확도로 [표2]와 같이 구별 성공을 판단하였다. 각 상황별로 실험결과는 아래와 같다.

【$\mathcal{F}_e^R$, $\mathcal{M}_e^R$: 동일한 S-box 사용】

$\mathcal{F}_e^R$, $\mathcal{M}_e^R$의 경우 Pollock's Pattern이 5라운드 이상부터는 *random*함수와 눈으로 식별이 불가한 것에 비해 딥러닝 모델은 8라운드까지 높은 확률로 각 구조를 식별하였다. 이는 DDT, LAT 계수들 사이의 관계나 구조가 반복되면서 발생하는 특정 연산에서 딥러닝이 구별 가능한 특징점이 존재한다는 것을 확인하였다.

구별가능 라운드(정확도(%))

구 분	$\mathcal{D}_D$	$\mathcal{D}_L$	$\mathcal{D}_{D\|\|L}$
$\mathcal{F}_e^R$ vs $\mathcal{M}_e^R$ vs $\mathcal{R}$	8R(58)	8R(61.8)	8R(58.3)
$\mathcal{F}_k^R$ vs $\mathcal{M}_k^R$ vs $\mathcal{R}$	7R(40.6)	5R(47.3)	5R(55.6)
$\mathcal{F}_d^R$ vs $\mathcal{M}_d^R$ vs $\mathcal{R}$	5R(57.8)	4R(100)	4R(99.3)

[표2] Maximum Identification Round S-box Structure

【$\mathcal{F}_k^R$, $\mathcal{M}_k^R$: 동일한 S-box ⊕ 라운드 key】

안전한 S-box는 라운드별로 다른 S-box를 사용하는 것이 가장 이상적이나, 암호 설계시 가용성 문제로 동일한 S-box에 라운드 키를 적용하여 설계한다. 이를 적용하여 $\mathcal{F}_k^R$, $\mathcal{M}_k^R$는 $s^r \oplus key$를 통해 라운드별로 다른 S-box를 적용한 효과를 가진다고 가정하였다. 실험 결과, $\mathcal{D}_D$ 는 6라운드까지 상당히 높은 수준으로 구별을 하였으며, 7라운드에서도 33.3%보다 높은 정확도로 구별을 하는 것을 알 수 있었다. 반면 $\mathcal{D}_L$ 의 경우 6라운드부터 33.3%에 가까운 수준을 나타냈다. 이를 볼 때, 차분 특성을 지닌 $\mathcal{D}_D$ 가 선형 특성을 나타내는 $\mathcal{D}_L$ 보다 구조를 식별하는데 좋은 성능을 보인다는 것을 알 수 있었다. 또한, $\mathcal{D}_{D\|L}$의 경우 우리의 예상과는 달리 정확도가 낮은 $\mathcal{D}_L$ 와 유사한 모습을 보였다. 이것으로 볼 때, $\mathcal{D}_{D\|L}$ 모델은 정확도가 상대적으로 낮은 $\mathcal{D}_L$ 의 영향을 많이 받는 것을 알 수 있었다.

【$\mathcal{F}_d^R$, $\mathcal{M}_d^R$: 서로다른 s^r 사용】

$\mathcal{F}_d^R$, $\mathcal{M}_d^R$ 의 경우 $\mathcal{F}_k^R$, $\mathcal{M}_k^R$의 경우보다 정확도가 낮았다. 이를 볼 때, 딥러닝 모델은 $\mathcal{F}_k^R$, $\mathcal{M}_k^R$의 s^r에 라운드 키가 xor 되기 전 동일한 S-box의 정보를 DDT나 LAT에서 특징점으로 추출한 것으로 분석할 수 있었다. 또한, $\mathcal{F}^R$ vs $\mathcal{R}$과 $\mathcal{M}^R$ vs $\mathcal{R}$의 경우를 실험해보면 상대적으로 $\mathcal{F}^R$ vs $\mathcal{R}$의 경우가 잘 식별되었다. 이는, *MISTY*설계간 선형 공격 및 차분공격에 저항성을 가지도록 설계하였다[9]고 알려져 있는데, 이 실험에서 사용한 특성이 LAT와 DDT인 점이 영향을 미친 것으로 판단된다.

Ⅴ. 결론

이번 실험을 통하여 S-box의 알려지지 않은 설계 원리를 딥러닝 모델이 식별할 수 있는지를 알아보았다. *Feistel* 및 *MISTY* 구조를 사용한 S-box의 경우 기존에 이론적으로 주장한 라운드 수보다 2라운드 증가한 라운드에서 한 번에 식별 가능함을 실험적으로 알 수 있었다. 딥러닝의 특성상 구별자가 구별에 활용한 특징점을 정확히 알 수는 없지만, 구조별 특성에서 나오는 불능 차분이나 LAT, DDT 계수들 간의 관계를 통하여 S-box를 구별해내는 것으로 추측해본다.

최근 경량암호에 대한 수요가 많아지면서 작은 bit S-box가 암호설계간 많이 사용되고 있다. 이러한 경량암호 분석간 우리가 설계한 도구를 사용시 설계 원리가 미공개된 S-box에 대한 근본적인 정보를 획득할 수 있다. 이러한 정보는 암호분석을 위한 역공학을 수행할 경우 하나의 Trapdoor역할을 할 수 있을 것으로 생각된다. 또한, S-box 설계간에도 검증시 우리의 딥러닝 구별자 도구를 사용한다면 설계간 알지 못한 특성을 찾아내어 암호의 안전성을 강화하는데 활용할 수 있을 것으로 생각된다.

한편, 실험적 측면에서 LAT와 DDT 특성을 혼합한 경우 오히려 실험에 부정적 영향을 주는 것도 식별하였는데 차후 암호분석에 활용할 수 있는 좋은 계기가 되었다. 향후에는 도출된 결과를 바탕으로 더 큰 S-box나 많은 라운드, 다른 설계원리들에 대해서도 식별 가능한 딥러닝 구별자를 구현하는 방법에 대해서 연구를 진행해 볼 예정이다.

[참고문헌]

[1] Biryukov, Alex, and Léo Perrin. "On reverse-engineering S-Boxes with hidden design criteria or structure." Annual Cryptology Conference. Springer, Berlin, Heidelberg, 2015.

[2] Luby, Michael, and Charles Rackoff. "How to construct pseudorandom permutations from pseudorandom functions." SIAM Journal on Computing 17.2 (1988): 373-386.

[3] Patarin, Jacques. "Generic attacks on Feistel schemes." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2001.

[4] Gilbert, Henri, and Marine Minier. "New results on the pseudorandomness of some blockcipher constructions." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 2001.

[5] Biham, Eli, and Adi Shamir. "Differential cryptanalysis of DES-like cryptosystems." Journal of CRYPTOLOGY 4.1 (1991): 3-72.

[6] Matsui, Mitsuru. "Linear cryptanalysis method for DES cipher." Workshop on the Theory and Application of of Cryptographic Techniques. Springer, Berlin, Heidelberg, 1993.

[7] Tan, Cheng, and Qingbing Ji. "An approach to identifying cryptographic algorithm from ciphertext." 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN). IEEE, 2016.

[8] 권동근, et al. "비프로파일링 기반 전력 분석의 성능 향상을 위한 오토인코더 기반 잡음 제거 기술." 정보보호학회논문지 29.3 (2019): 491-501.

[9] Gohr, Aron. "Improving Attacks on Round-Reduced Speck32/64 Using Deep Learning." Annual International Cryptology Conference. Springer, Cham, 2019.

[10] Matsui, Mitsuru. "New block encryption algorithm MISTY." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 1997.

언어 규칙 기반 DGA-DNS 머신러닝 분류 모델링

이동주*, 강성묵*

*순천향대학교 정보보호학과, *강남대학교 컴퓨터공학과

DGA-DNS Machine Learning Classification Based on Language Rules

Dong-Ju Lee*, Seong-muk Gang*

*Department of Information Security Engineering, Soonchunhyang University.
*Computer Engineering, Kangnam University, Computer engineering

요 약

오늘날의 국내·외 기업들은 시스템으로부터 유입되는 악성코드에 대한 사이버 위협에 직면하고 있다. APT(Advanced Persistent Threat) 공격엔 악성코드에 감염된 클라이언트와 C&C(Command and Control) 서버 간의 통신으로 클라이언트의 자산에 침입하고 공격 행위를 지시할 수 있다. 하지만, C&C 서버 정보가 분석가들에 의해 차단되면 감염된 클라이언트와 공격자의 통신을 방지함으로서 침해대응에 성공할 수 있다. 때문에, 최근 악성코드 개발자는 접속 방지 기법을 우회하기 위한 DGA(Domain Generation ALgorithm) 방식을 사용해 도메인 주소를 무작위로 생성하여 DNS(Domain Name Server)를 짧은 시간 간격으로 교체하여 탐지 및 차단을 어렵게 한다. 본 논문은 이러한 문제점을 해결하기 위해 문자의 언어 규칙 요소를 활용하여 DGA-DNS 탐지를 판단하는 머신러닝 모델을 제시하고 유효성을 검증한다.

I. 서론

악성 DNS를 활용한 악성코드 공격은 감염된 클라이언트와 공격자의 C&C 서버가 지속적으로 통신을 함으로써 감염자의 자산을 침입하거나 공격 행위를 지시할 수 있다. 하지만 이 단계에서 공격자의 IP 주소 및 도메인을 차단함으로써 악성코드에 감염된 클라이언트가 C&C 서버에 접속하는 것을 방지할 수 있다. 이러한 정적 기반의 탐지 및 차단 방식을 우회하기 위해 공격자의 C&C서버는 IP 통신을 시작으로, Fast Flux를 거쳐 DGA를 사용하고 현재는 이 둘을 합친 형태로 진화한 기법을 사용하고 있어 정적 기반 방식을 어렵게 하고 있다.

따라서 본 논문에서는 머신러닝을 기반으로 DGA-DNS로 생성된 도메인과 정상 도메인 탐지 모델을 설계하기 위해, 학습 데이터 샘플로 인텔리전스 37개 그룹에서 생성된 DGA-DNS와 정상 도메인 상위 약 2만 개를 대상을 선정하였다. 머신러닝 모델링에서 DGA-DNS로 생성된 악성 도메인과 정상 도메인을 판별하기 위해 어휘의 자음과 모음의 배치 관계도, 문자의 복잡도, 문자의 길이, 문자의 벡터화를 통한 기울기, Histogram 등의 요소를 학습요소로 활용한 결과 97.53% 이상의 탐지율을 보였다.

최종적으로 위와 같은 머신러닝 모델을 설계함으로써 DGA-DNS로 제작된 C&C 서버를 탐색 및 구별하여 보안사건/사고를 예방하는데 기여한다.

II. 관련연구

한국정보보호학회 논문지의 김동현 저자의 논문에 “N-gram을 활용한 DGA-DNS 유사도 분석 및 APT 공격 탐지” 논문은 DGA-DNS의 형태적 측면을 활용하여 N-gram을 통해 DGA-DNS를

판별하였다. 반면에, 형태적인 측면뿐만이 아닌 언어적 특징을 활용한 “Raghuram”, “Wei-Wei” 연구에서는 언어학적 특징을 통해 분석으로 DGA-DNS를 판별하였다. 이러한 형태적인 측면을 활용하여 딥러닝 모델을 제시한 ‘Ubin’ 등의 연구에서는 DGA-DNS의 텍스트 분류를 위해 CNN, RNN, LSTM, MLP 등의 모델을 활용하였으며, 평균적으로 97%의 정확도를 보였다. 하지만, 어휘의 특징이 아닌 자연어 처리 모델을 기반으로 유사한 벡터와 의미론적으로 유사한 언어 형태를 매핑하는 방식으로 공격자가 정상 도메인과 유사한 DNS를 DGA-DNS로 활용할 때 무력화가 될 가능성을 보여준다.

본 논문에서는 기존 딥러닝 모델의 악성 DNS를 탐지하기 위한 언어 형태의 복잡도를 계산하는 방식과 언어의 특징을 활용한 방식을 앙상블 함으로써 효과적인 DGA-DNS 판별 머신러닝 모델 설계를 제안한다.

III. 머신러닝 모델링 프로세스

3.1 데이터 수집 및 전처리

학습 데이터의 특징을 분석하기 위해 총 37개의 악성코드 그룹에 대한 DGA-DNS 데이터 20만개를 수집하였다. DGA-DNS는 공격 그룹마다 알고리즘에 대한 차이가 있어 형태가 다른 양상을 나타낸다. 각 그룹별 생성된 DGA-DNS 알고리즘으로 인해 생성된 데이터는 다음 [표 1]과 같다

Groups	DGA-DNS
banjori	earnestnessbiophysicalohax.com
chinad	8f6bacmw30xxv6sc.cn
corebot	63234wluv5v365bwp5.ddns.net
dircrypt	rauggyguyp.com
dnschanger	aktklyvbiu.com
fobber	gbjtyyhrhk.com
gozi	quodpresidentemaxsagit.com
ranbyus	jxbdxeyxttdmcjagi.me
reconyc	E5zHail0Mw.com
shiotob	wtipubctwiekhir.net
sisron	mdiwnjiwmtya.com
suppobox	mariabellachauncey.net
symmi	ogovugtuipawi.ddns.net

[표 1] 악성 DGA-DNS 데이터 예

그다음으로 정상 DNS 정보를 수집하기 위해 도메인 순위를 나타내 주는 [그림 1]의 “quantcast” 사이트에서 자료를 수집하였다. 상위 1위부터 20,000위까지의 도메인 이름을 수집하여 정상 DNS에 대한 데이터셋을 확보하였다.

3.2 데이터 분석

수집된 37개의 공격 그룹에서 생성된 DGA-DNS 20만 개를 수집하였다. 머신러닝 학습 요소를 추출하기 위해 DGA-DNS의 언어적인 특징과 언어의 형태를 기반으로 공통된 사항을 추출해보면 다음과 같다.

1) 문자의 복잡도

DGA-DNS로 생성된 것은 일방적으로 정상 DNS보다 복잡도가 높다. 이러한 복잡도를 계산하기 위해 엔트로피 수치를 사용하였으며, 정상 DNS의 경우 평균적으로 2~3의 수치가 나오는 반면 DGA-DNS로 작성된 경우 엔트로피 수치가 평균 4~5의 수치가 나왔다.

2) 문자의 길이

DGA-DNS로 생성된 도메인은 정상 도메인보다 평균적으로 길다. 하지만, 일부 DGA-DNS는 정상 도메인보다 짧을 수 있다.

3) 자음과 모음의 배열

일반적으로 정상 DNS는 접근하기 쉬운 이름으로 지정하기에 기억하기 쉽거나 발음하기 쉬운 단어로 한다. 이는 언어학적인 측면으로 정상 DNS에 대한 문자열을 보면 자음과 모음의 배열에 대한 규칙성이 균일함을 알 수 있지만, DGA-DNS 경우 자음과 모음의 배열이 가변적이다.

4) TLD

정상 DNS 같은 경우 .com, .org와 같은 정상적인 TLD를 활용하는 반면에, 악성코드 같은 경우 .tf, .mx와 같은 알려지지 않은 TLD를 일부 활용함을 알 수 있다.

5) 숫자 사용 여부

생성된 DGA-DNS 중 숫자를 2~3개 이상을 갖고 있는 형태가 존재한다. 반면에, 정상적인 DNS의 경우는 사용자가 기억하기 쉬운 형태를 유지하기 위해 대부분 숫자를 삽입하지 않는다.

6) dot(.)의 갯수

피싱 사이트 분류 모델에서 활용되는 방법 중 하나로 정상적인 DNS의 경우 '.'의 개수가 DNS내에 1~2개이며 그 이상인 경우는 없다. 하지만 일부 DGA-DNS가 2개 이상의 '.'을 삽입한 형태를 나타내고 있음을 알 수 있다.

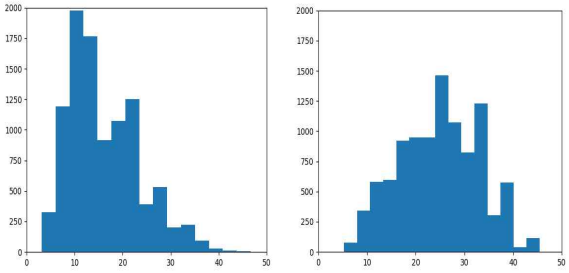
이러한 규칙에 대한 머신러닝 앙상블 효과는 각각의 학습 요소가 갖고 있는 단점을 극복할 수 있는 시너지 효과를 발휘할 수 있기에, 이를 활용할 수 있는 머신러닝 모델링의 필요성을 나타낸다.

3.3 엔진 요소

3.2의 데이터 분석 과정을 통해 머신러닝 엔진에 활용할 수 있는 총 요소는 19개의 요소에 해당한다. 머신러닝 엔진의 각 요소는 언어의 특징 및 형태 그리고 정상 도메인과 비정상 도메인에 대한 일반적인 차이를 기반으로 설계하였다. 활용되는 머신러닝 엔진 요소는 다음과 같다.

1) 문자열 간 기울기

기울기의 확실한 결과를 위해 자모음 간의 큰 차이를 두어 인덱싱 후, 인덱싱한 숫자를 바탕으로 생성된 직선의 기울기를 생성한다.



[그림 1] 좌) [DGA-DNS 기울기 비율]
우) [정상 DNS 기울기 비율]

inclination	DGA-DNS	DNS
25 ~ 30	902	2597
20 ~ 25	1678	1817
15 ~ 20	1574	1139
10 ~ 15	3043	1218
5 ~ 10	2055	205

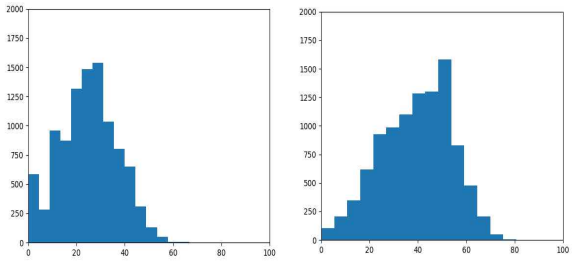
[표 2] 기울기에 따른 DNS 분포

악성 DGA-DNS의 기울기를 확인하면 약 10~20 정도이며 정상 DNS의 히스토그램의 경우 20 ~35 정도 기울어져 있는 것을 확인할 수 있었다.

이러한 통계를 통해서 자음과 모음의 언어적 조화는 DNS 설정 시 많은 고려사항이며 벡터의 기울기 값에 의해 확인하였다. 즉, 자음과 모음의 조화는 DGA-DNS보다 많은 사람이 이해하기 쉬운 일반 DNS에 많이 적용되어 높은 벡터 간 기울기가 형성되었음을 확인할 수 있다.

2) 두 직선의 사잇각

자음과 모음의 임의의 인덱싱으로부터 직선을 생성하였다. 이러한 직선의 사잇각을 이용하여 언어적 특징인 자음과 모음의 활용도를 구분하고자 한다.



[그림 2] 좌) [정상 DNS의 각도 비율]
우) [DGA-DNS의 각도 비율]

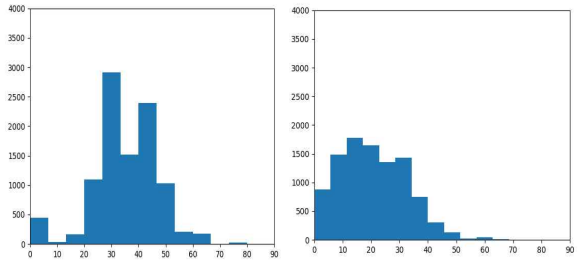
inclination	DGA-DNS	DNS
60 ~ 70	642	7
50 ~ 60	1792	188
40 ~ 50	2849	936
30 ~ 40	1998	2330
20 ~ 30	1668	3306

[표 3] DNS 사잇각 분포

정상 DNS의 각도 비율을 확인하면 다음과 같이 40° 이하를 가진 각도 비율이 88%로 나타났다. 반면에 DGA-DNS의 경우 40° 이상 각도의 비율이 53%로 나타났다. 자음과 모음의 순차적 이동이 활발히 일어나 사잇각이 40° 이하인 비율이 DGA-DNS보다 월등히 높다는 결론을 얻을 수 있었다.

3) 언어 자모음의 비율

기존 DNS로부터 사람이 사용할 수 있는 언어 성격을 가졌는지 특징 분석을 위해 자음과 모음 비율을 활용한다. 영어의 알파벳은 총 24개의 문자이며 21개의 자음과 5개의 모음으로 이루어져 있다. 이러한 구성요소를 이용하여 사람의 발음이 가능하고 문법에 이상이 없으며 문자의 사용을 조사, 자음, 모음 비율을 확인한다.



[그림 3] 좌) [일반 DNS 자모음 비율]
우) [DGA-DNS 자모음 비율]

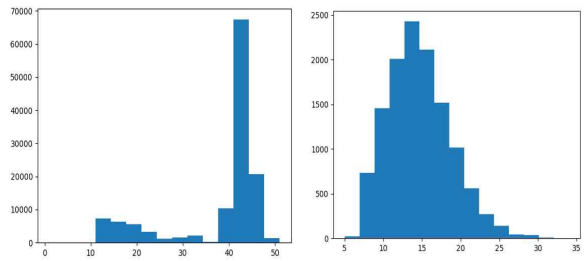
inclination	DGA-DNS	DNS
50% ~ 60%	171	1215
40% ~ 50%	320	2416
30% ~ 40%	1845	3700
20% ~ 30%	2683	1827
10% ~ 20%	3045	171

[표 4] 자모음 비율 분포

4) 문자열 엔트로피

엔트로피를 활용하여 문자열의 무질서도를 측정합니다. 만약 해당 DGA가 문법에 맞는 영어 문장만을 담고 있다면 엔트로피의 수치는 더 줄어들게 된다. 하지만 반대로 랜덤하게 적혀 있는 영어 글자의 엔트로피 수치는 상대적으로 높은 엔트로피 수치를 기록하였다.

5) 도메인 길이



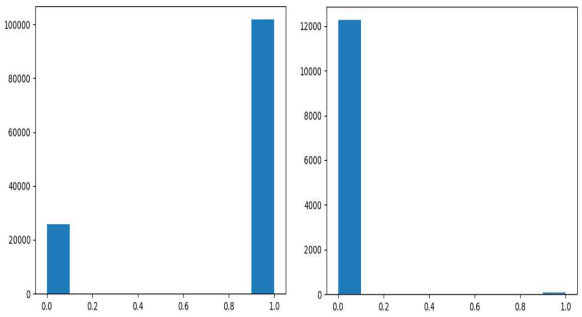
[그림 4] 좌) [악성 DGA-DNS 도메인 길이 분포도]
우) [일반 DNS 도메인 길이 분포도]

40 Byte 이상 DGA-DNS 개수가 전체의 90%를 차지할 정도로 압도적으로 많은 것을 확인할 수 있었다. 반대로 일반 DNS는 20Byte 이하의 길이가 전체의 90%를 차지하는 것을 확인할 수 있다. 위와 같이 상반된 결과는 머신러닝 엔진 요소로 활용되어 일반 DNS와 악성 DGA-DNS를 구분 짓는 데 매우 효과적이다.

6) TLD [Top Level Domain]

TLD는 .com .net과 같은 형태로 존재하며 그 이외에도 지역으로 분류하면. kr(한국), .jp(일본), .Cn(중국)도 존재한다. 하지만 악성 DGA-DNS의 경우 알려진 TLD를 사용을 하는 경우가 적었으며 이를 이용하여 판별에 사용하였다. 여러 개의 서브 도메인을 가지는 것이기 때문에 피싱 사이트로 분류하였다.

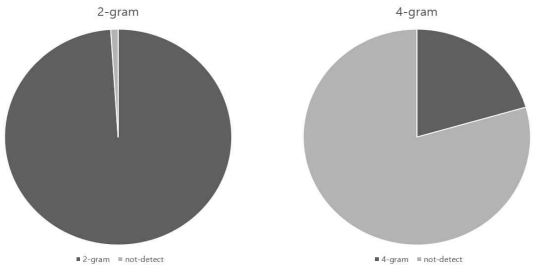
7) 문자열 분포도



[그림 5] 좌) [악성 DGA-DNS 도메인 문자열 분포도]
우) [일반 DNS 도메인 문자열 분포도]

문자열 분포도는 중복으로 사용하는 단어의 개수를 뜻한다. 중복된 문자 3개 이상 존재하면 악성 DGA-DNS라 판별을 시도하였으며 약 127000개의 악성 DGA-DNS와 약 120000개의 일반 DNS의 문자열 분포도 실험 결과 다음과 같은 결과를 얻었다. 악성 DGA-DNS 도메인 문자열 분포도의 80%는 3개 이상의 문자를 중복적으로 사용하였다. 반면에 일반 DNS 도메인 문자열 분포도는 95%의 도메인이 문자열을 중복 사용 안 하는 것으로 나타났다.

8) Histogram



[그림 6] N-gram N값 탐지 실험 결과

약 1만 개의 악성 DGA-DNS를 대상으로 2-gram과 4-gram을 진행하여 어떠한 n 값이

유사도 판별 결과가 높은지 확인하는 실험을 진행하였다. 그 결과 총 1만 개의 악성 DGA-DNS 중 2-gram은 9403개의 DGA-DNS가 매치가 되었다. 반면에 4-gram은 약 2053개의 DGA-DNS가 매치되어 위 결과와 상반된 모습을 보였다.

IV. 모델링 실험

학습 모델은 여러 머신러닝 알고리즘을 경쟁시킨 뒤 좋은 성능을 자랑하는 모델을 선택하였다. 학습하는 과정에서 Gradient Boosting, Random Forest, XGBoost, SVC, LightGBM등 다양한 함수를 활용한다. 머신러닝 알고리즘에 대해 최적의 파라미터를 삽입하기 위해 파이썬 Scikit-learn 모듈의 GridSearchCV 함수를 활용하였다. 또한, 각 머신러닝 알고리즘을 경쟁하여 높은 수치가 나온 알고리즘을 선별하여 Slearn 모듈에서 제공하는 ensemble 함수의 투표 기반 분류를 활용하여 각 머신러닝 모델들이 예측 결과를 이용해 하나의 모델보다 더 나은 결괏값을 예측할 수 있도록 한다.

4.1 실험 및 환경

실험 단계는 총 3단계로 구성된다. 실험의 단계는 '데이터셋 선정'-'모델링 테스트'-'모델링 앙상블'로 구성된다. 실험의 최종적인 목적은 DGA-DNS와 정상 DNS를 구분하는데 있어 정확도 높은 머신러닝 알고리즘을 선별함을 목표로 한다. 실험에 활용된 데이터는 DGA-DNS 20만개 샘플과 정상 DNS 2만개(상위 도메인)로 구성된다.

1) 머신러닝 데이터셋

데이터셋을 구축하기 위해 설계한 엔진을 기반으로 정상 DNS와 DGA-DNS에 대한 요소를 CSV 파일로 기록한다.

URL	earnestnessbiophysicalohax.com
Length	30
dot_len	1
TLD	2
Repet_num	2
Entropy	3.898069
2gram(11)	[1,1,1,0,1,0,1,0,0,0,0]
4gram(20)	[1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
Collection Percent	43.3333333333333
Collection Count	1
Collection inclination	23.076923076923
label	1

[표 5] DGA-DNS 학습 데이터셋 예시

2) 머신러닝 모델링

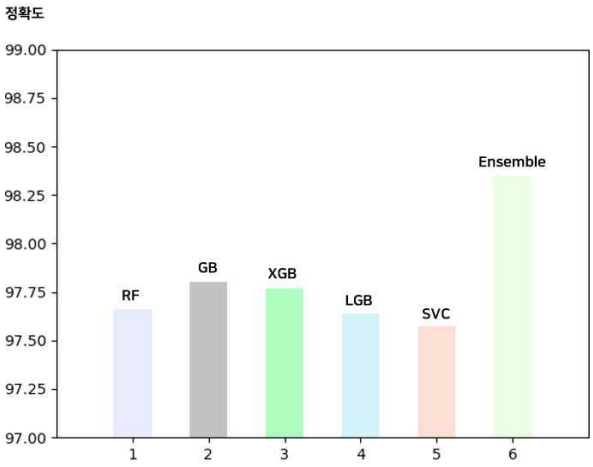
Average Score가 가장 높은 머신러닝 분류 모델을 선정하기 위해 각 알고리즘을 경쟁하여 Best Case를 선정하다. 경쟁을 위해 활용되는 머신러닝 모델은 RandomForest, GradienBoost, XGBoost, LightGBM, SVC, Ensemble 이다.

4.2 모델링 선택 및 앙상블

머신러닝 모델을 경쟁하여 각 모델 중 Score 점수가 높은 모델을 선택한다. 선택된 모델은 투표 기반 분류기 학습을 통해 모델을 학습시킨다. 그 뒤 다수결 투표를 진행하여 최종 Classifier를 예측한다.

4.3 모델링 학습 결과

각 머신러닝 알고리즘과 이중 가장 높은 수치를 반환한 알고리즘을 활용한 결과 다음 아래 [그림 7]과 같았다.



[그림 7] 전체 알고리즘 실험 결과 수치

V. 결과

학습 데이터로 활용된 악성 DGA 도메인 20만 개와 정상 도메인 상위 2만 개는 악성 DGA와 정상 도메인을 구별하기 위해 37개의 머신러닝 학습 요소를 추출하였다. 추출된 각종 요소는 기존에 알려지지 않은 방식인 언어의 특징과 언어의 형태를 활용하였다. 알려진 논문과 발표된 자료를 인용하면 도메인에 대한 whois를 통해서 직접 조사를 한 '네트워크 방식'과 언어의 형태나 특징 같은 문자열 복잡도를 이용한 '언어의 특징' 및 '언어의 형태'로 구분된다. '네트워크 방식'은 정확성은 높으나 하루 평균 500만 개가 신규로 생성되고 있는 악성 DGA를 분류하는 것은 하드웨어적 부담감이 있으며, '언어의 특징' 및 '언어의 형태'는 빠르게 분류하고 분석하는 것이 가능하지만 정상 도메인과 비슷한 유형의 악성 DGA를 분류하기 어렵다는 특징을 갖는다. 본 보고서에서는 각 특징을 모두 추출하여 RandomForest, GradientBoosting, XGBoosting 알고리즘을 활용한 결과 97% 이상의 평균 정확도를 자랑하였다.

5.1 독창성

DGA-DNS를 구별하는 기존 방법은 Histogram, 자연어 처리 모델, 네트워크 조사 방식이 대표적이었다. 하지만, 본 보고서에서는 언어의 자연어 특성을 활용한 '언어의 형태' 기법, 언어의 복잡도를 활용한 '언어의 특징' 기법을 통해 문자의 복잡도, 어휘의 자음과 모음의 배치, 문자의 벡터화를 통한 기울기, 자음과 모음의 기울기 등 언어의 복합적인 특성을 활용한 획기적인 머신러닝 모델링 방식을 제안 및 구현하였으며, 높은 정확도를 산출해냄으로써 모델링의 성공적인 입증을 하였다.

5.2 기대 효과

기존 한국인터넷진흥원에서 소개된 DGA-DNS 구별 방법은 '네트워크 조사' 방식을 활용한다. '네트워크 조사' 방식은 'Alexa'를 통해서 사이트 통계를 확인하고 URL에 대한 whois를 활용하여 도메인 정보를 요청하여 정보를 확인한다. 이러한 '네트워크 조사' 방식은 정확성은 뛰어나지만, 현대에서 DGA를 활용한 방식은 하루 평균 500만 개가 신규로 생성되고 있다는 점에서 '네트워크 조사' 방식은 분석 하드웨어의 자원 소모량이 커 가용성에 대한 문제점이 있다.

본 보고서에서는 '언어의 특징'과 '언어의 형태'의 복합적인 특성을 활용한 획기적인 머신러닝 모델링 방식을 제안하고 구현하였으며, 높은 정확도 및 빠른 분석은 산업현장에서의 침해위협 대응에 효과적일 것임을 기대한다.

[참고문헌]

[1] Elmar Gerhards-Padilla (2016) A Comprehensive Measurement Study of Domain Generating Malware

[2] 한국인터넷진흥원 (2019) '딥러닝을 이용한 악성 도메인 탐지 기법'

[3] Kim DongHyeon (2018) DGA-DNS Similarity Analysis and APT Attack Detection Using N-gram

[4] Raghuram&Wei-Wei (2016) DGA-DNS Similarity Analysis and APT Attack Detection Using N-gram

진화-생산적 적대 신경망에 기반한 악성 소프트웨어 탐지

장용남*, 원동옥*, 이성환*,**

*고려대학교 뇌공학과, **고려대학교 인공지능학과

Malware Detection based on Evolutionary Generative Adversarial Networks

Yong-Nam Jang*, Dong-Ok Won*, Seong-Whan Lee*,**

*Department of Brain and Cognitive Engineering, Korea University,

**Department of Artificial Intelligence, Korea University

요 약

악성 소프트웨어(Malware)는 컴퓨터 보안 관련하여 여전히 심각한 문제로 지적되고 있으며 그 피해는 해마다 증가하고 있다. 특히, 악성 소프트웨어는 기존의 악성 소프트웨어를 변형시켜서 쉽게 생성될 수 있는데, 많은 안티바이러스 시스템들은 사전 정보가 없는 악성 소프트웨어들에 대해서는 탐지를 못 하는 문제가 있다. 따라서, 변형된 악성 소프트웨어들에 대한 데이터를 사전에 학습할 수 있다면 탐지 확률을 높일 수 있다. 본 논문에서는 진화-생산적 적대 신경망(Evolutionary Generative Adversarial Networks, E-GAN) 모델에 기반하여 악성 소프트웨어의 다양한 변형을 사전에 생성하고 분류 정확도를 높이기 위해 이러한 데이터로 모델을 훈련하는 방법과 악성 소프트웨어를 자동으로 분류하는 방법을 제안한다. 제안하는 방법의 평균 분류 정확도는 97.2%로 다른 모델들보다 성능이 높은 것을 확인하였다.

I. 서론

악성 소프트웨어(Malware)는 사용자의 컴퓨터를 감염 시켜 정보를 유출하거나 시스템을 파괴하기 위해 설계된 소프트웨어를 말한다. Kaspersky Lab 기관의 2018년 한 해 동안 전체 컴퓨터 중 30%에 가까운 컴퓨터들이 최소 1번 이상의 위협을 받았다는 조사에서 볼 수 있듯이[1], 악성 소프트웨어 탐지는 컴퓨터 보안에서의 중요한 이슈이다. 악성 소프트웨어 탐지를 위해 기존의 많은 안티바이러스 시스템들은 악성 소프트웨어 특징을 추출한 뒤 저장하여 사용한다. 그러나 악성 소프트웨어 개발자들은 이러한 안티바이러스 시스템의 탐지를 피하고자 주로 기존 악성 소프트웨어를 변형시켜 같은 기능을 하는 유형이라도 다른 특징을 가지도록 만든다. 이렇게 되면 저장된 특징들을 사용하는 기존의 안티바이러스 시스템들은 변형된 특징을 샘플링하고 분류하기 전까지 같은 유형의 악성 소프트웨어라도 변형된 악성 소프트웨어를 탐지 할 수 없다. 따라서 본 논문에서는 악성 소프트웨어를 자동으로 분류하는 방법을 제안하여 분류 작업에서 발생하는 비용을 절감하고 변형된 악성 소프트웨어를 사전에 생성한 후 학습하여 분류 정확도를 높이고자 한다.

II. 관련 연구

악성 소프트웨어 분석 및 분류를 위해서 기계학습을 이용한 여러 시도가 있었으나 최근에는 딥러닝을 이용한 연구가 늘고 있다[2]. 딥러닝을 이용한 악성 소프트웨어 탐지는 크게 두 부분으로 이루어진다. 첫째는 악성코드로부터 딥러닝 모델 학습에 사용할 특징 데이터를 추출하는 것이다. 특징 데이터를 추출하는 방법으

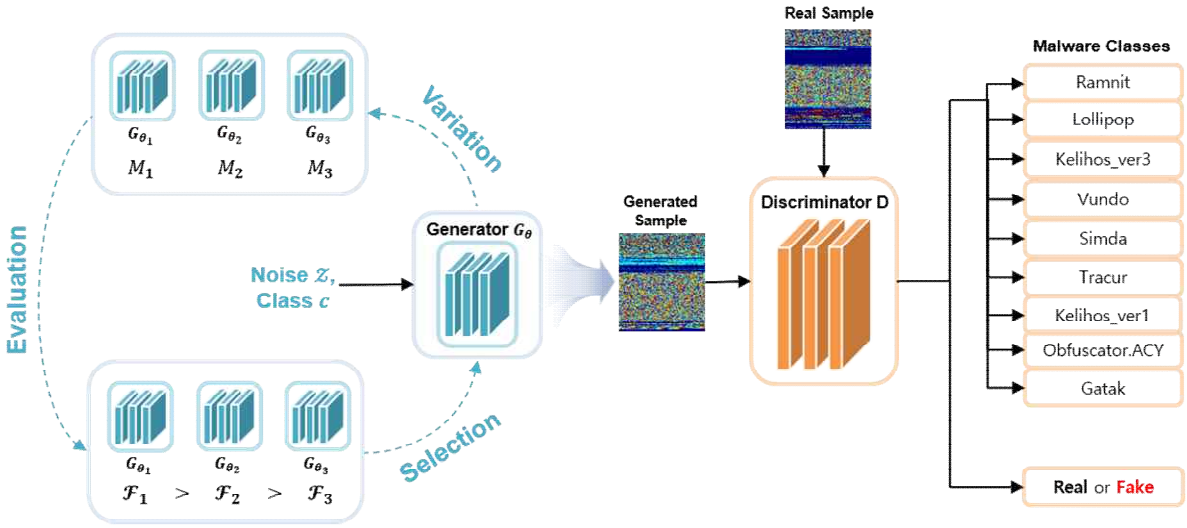


그림 1. 제안하는 방법의 전체 구조

로는 악성코드를 이미지로 간주하거나[3] API를 사용한다[4]. 둘째는 추출한 특징 데이터를 사용하여 딥러닝 모델을 만들고 학습하는 것이다. 악성 소프트웨어 탐지를 위한 딥러닝 모델로 Recurrent Neural Networks(RNN)[5]이나 Convolutional Neural Networks(CNN)[6]이 사용되었다. 표1은 이러한 연구들의 사례를 정리한 것이다.

표 1. 악성 소프트웨어 분석 및 분류에 관한 연구

저자	방법
Nataraj, L. (2011)[3]	코드 벡터화를 통한 시각화
Dahl, G. E. (2013)[4]	API call
Pascanu, R. (2015)[5]	RNN
Kabanga, E. K. (2018)[6]	CNN

III. 방법

3.1 생산적 적대 신경망

생산적 적대 신경망(Generative Adversarial Networks, GAN)은 실제 데이터와 비슷한 데이터를 생성해내는 생성 모델과 생성한 데이터와 실제 데이터를 구분해내는 분류 모델로 나누어진다. 본 논문에서 생성 모델은 진화 알고리즘[7]을 사용하였으며 분류 모델은 악성 소프트웨어의 실제 여부는 물론 유형 구분을 위해 최종 산출은 두 개의 층으로 구성하였다[8]. 악성 소프트웨어 유형 c기준으로 임의로 생성된 z에서 생성 모델로 $G(z|c)$데이터를 생성해내고 분류 모델을 통해 실제 여부와 유형을 구분해낸다. 표2는 생성 모델과 분류 모델의 자세한 구조이다. 전체 목적어 함수는 실제와 생성한 데이터

표 2. 생성 모델과 분류 모델의 자세한 구조

Generative Network G
Input: Noise z + Class c
[layer 1] Fully connect and Reshape to (4 x 4 x 2048); Leaky ReLU;
[layer 2] Transposed Conv. (4, 4, 2048), stride=2; Leaky ReLU;
[layer 3] Transposed Conv. (4, 4, 1024), stride=1; Leaky ReLU;
[layer 4] Transposed Conv. (4, 4, 1024), stride=2; Leaky ReLU;
[layer 5] Transposed Conv. (4, 4, 512), stride=1; Leaky ReLU;
[layer 6] Transposed Conv. (4, 4, 512), stride=2; Leaky ReLU;
[layer 7] Transposed Conv. (4, 4, 256), stride=1; Leaky ReLU;
[layer 8] Transposed Conv. (4, 4, 256), stride=2; Leaky ReLU;
[layer 9] Transposed Conv. (4, 4, 128), stride=1; Leaky ReLU;
[layer 10] Transposed Conv. (4, 4, 128), stride=2; Leaky ReLU;
[layer 11] Transposed Conv. (3, 3, 3), stride=1; Tanh;
Output: Generated Image, (128 x 128 x 3)
Discriminative Network D
Input: Real image x or Generated image $G(z\|c)$ (128 x 128 x 3)
[layer 1] Conv. (3, 3, 128), stride=1; Batchnorm; Leaky ReLU;
[layer 2] Conv. (3, 3, 128), stride=2; Batchnorm; Leaky ReLU;
[layer 3] Conv. (3, 3, 256), stride=1; Batchnorm; Leaky ReLU;
[layer 4] Conv. (3, 3, 256), stride=2; Batchnorm; Leaky ReLU;
[layer 5] Conv. (3, 3, 512), stride=1; Batchnorm; Leaky ReLU;
[layer 6] Conv. (3, 3, 512), stride=2; Batchnorm; Leaky ReLU;
[layer 7] Conv. (3, 3, 1024), stride=1; Batchnorm; Leaky ReLU;
[layer 8] Conv. (3, 3, 1024), stride=2; Batchnorm; Leaky ReLU;
[layer 9] Conv. (3, 3, 2048), stride=1; Batchnorm; Leaky ReLU;
[layer 10] Conv. (3, 3, 2048), stride=2; Batchnorm; Leaky ReLU;
[layer 11-1] Fully connected (1); Sigmoid; [layer 11-2] Fully connected (9); Softmax;
Output: Real or Fake (Probability), Classes of Malware (Probability)

를 구분하기 위한 로그 가능도 함수(L_S), 유형 분류를 위한 로그 가능도 함수(L_C)로 두 가지이다. 분류 모델은 L_S+L_C를 최대화하는 방향으로 훈련되고, 생성 모델은 L_C-L_S를 최대화하는 방향으로 훈련된다.

$$L_S = E[\log P(S = real \mid x)] + E[\log P(S = fake \mid G(z|c))] \quad (1)$$

$$L_C = E[\log P(C = c \mid x)] + E[\log P(C = c \mid G(z|c))] \quad (2)$$

학습이 진행될수록 생성 모델은 진화 알고리즘에 의해 실제 악성 소프트웨어와 비슷하고 다양성이 높은 데이터를 생성하게 되며 분류

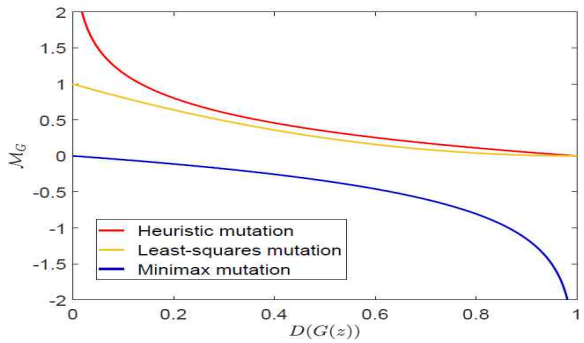


그림 2. 변형 함수들의 차이

모델은 이러한 생성 데이터와 실제 데이터를 학습하게 된다. 악성 소프트웨어가 실제 데이터를 변형시켜서 생성된다는 사실로 볼 때 이와 은 방법은 악성 소프트웨어 분류에 효과적임을 알 수 있다.

3.2 진화-생산적 적대 신경망

생성 모델의 훈련을 안정화하면서 실제 악성 소프트웨어와 유사하면서도 다양성이 높은 데이터를 생성해내기 위해 진화 알고리즘을 이용하는 진화-생산적 적대 신경망(Evolutionary Generative Adversarial Networks, E-GAN)을 사용하였다. 그림 1은 본 논문에서 제안하는 방법의 전체 구조를 나타낸 것이다. 손실 함수를 하나만 쓰는 일반적인 GAN과 달리 E-GAN은 손실 함수를 여러 방식으로 변형(Mutation, M) 시키면서 생성 모델을 진화시키는 방법이다. 본 논문에서는 3개의 변형 (minimax, heuristic, lease-square)함수를 사용하였으며 그림2는 이러한 변형 함수들의 차이를 보여주고 있다.

$$M_G^{\text{minimax}} = \frac{1}{2}E_{z\sim p_z}[\log(1-D(G(z)))] \quad (3)$$

$$M_G^{\text{heuristic}} = -\frac{1}{2}E_{z\sim p_z}[\log(D(G(z)))] \quad (4)$$

$$M_G^{\text{least-square}} = E_{z\sim p_z}[(D(G(z)) - 1)^2] \quad (5)$$

손실 함수의 변형에 따른 여러 생성 모델은 분류 모델에 의해 평가를 받는데 평가 기준으로 생성되는 이미지의 품질과 다양성을 측정한다. 품질은 분류 모델에서 실제 여부의 산출 값을 사용하고 다양성은 분류 모델 매개변수들의 기울기를 사용한다. 산출 값이 높다면 품질이 높은 것이고 기울기가 작다면 분류 모델을 크게 변화시키지 않으면서 데이터를 다양하게 만든다고 볼 수 있다. 적절한 비율로 산출 값과 마이너스 로그 기울기를 더하여 성능이 가장 높은 생성 모델이 선택되고 다음 학습 단계로 넘어간다. 데이터 생성과 이렇게 생성된 데이터로 학습하는 과정을 충분히 하기 위해 1,000세대 반복 진행하였다.

IV. 실험 결과

4.1 데이터

학습과 검증에 사용되는 데이터는 Microsoft에서 Kaggle에 공개한 데이터로써 표3과 같다. 바이너리 코드로 되어있는 원본 데이터를 코드 벡터화를 통하여 이미지로 만들었다. k를 바이너리 코드의 길이, C를 변형된 열의 크기, R을 변형된 행의 크기로 하였을 때 열과 행의 크기를 계산하는 방법은 다음과 같다.

$$C = 2^{\frac{\log\sqrt{16k}}{\log 2}+1} \quad (6)$$

$$R = \frac{16k}{C} \quad (7)$$

최종적으로 변형된 행과 크기의 이미지를 128 x 128 크기로 맞춰주었다.

표 3. 사용되는 데이터 종류와 개수

종류	개수	종류	개수	종류	개수
Ramnit (R)	1541	Lollipop (L)	2478	Kelihos_ ver3(K3)	2942
Vundo (V)	475	Simda(S)	42	Tracur (T)	751
Kelihos _ver1 (K1)	398	Obfuscato r.ACY(O)	1228	Gatak (G)	1013

4.2 생성 이미지 분석

E-GAN을 이용하여 실제 악성 소프트웨어의 특징을 잘 학습하고 비슷한 데이터를 생성해내는지 알아보기 위해 생성 모델이 생성한 이미지와 실제 이미지를 그림3에 나타내었다. 다양한 악성 소프트웨어의 대략적인 특징을 잘 생성해냈음을 알 수 있다.

4.3 분류 정확도

분류 모델의 유형 분류 층을 이용하여 악성 소프트웨어 분류를 실험하였다. 전체 데이터 중 10%의 데이터로 최종 성능을 평가하였으며, 표 4는 결과에 따른 혼동행렬을 나타내었다.

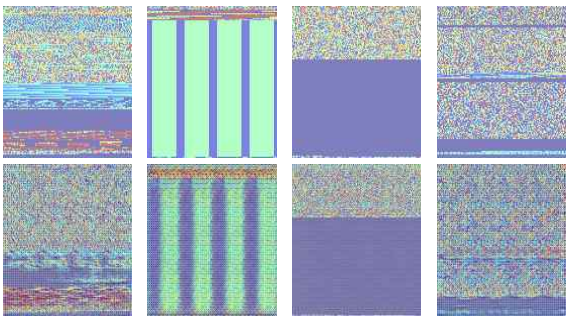


그림 3. 실제 이미지(위)와 생성해낸 이미지(아래)

표 4. 분류 결과 혼동 행렬

		Predicted								
		R	L	K3	V	S	T	K1	O	G
Actual	R	154	0	0	2	0	0	0	1	1
	L	0	247	0	0	0	0	0	0	0
	K3	0	0	294	0	0	0	0	0	0
	V	1	1	0	47	0	0	0	0	0
	S	0	0	0	0	4	0	0	0	0
	T	1	1	0	1	0	75	1	0	1
	K1	2	0	0	0	0	0	39	0	0
	O	0	1	2	1	0	2	1	122	1
	G	3	1	0	0	0	1	1	3	101

목적어 함수를 하나만 쓰는 다른 GAN 모델들과 CNN 분류기와의 비교를 표5에 나타내었다. 수치는 5-fold 교차 검증한 결과이다. 비교 결과, 제안하는 방법이 97.23%로 가장 높은 분류 성능을 보여주었음을 확인할 수 있었다.

표 5. 다른 분류기와의 비교 결과

모델	CNN	DCGAN	LSGAN	WGAN	**Ours**
정확도(%)	94.33	95.01	95.48	96.1	97.23
표준편차	0.045	0.082	0.058	0.069	0.082

V. 결론

본 논문에서는 악성 소프트웨어의 분류 정확도를 높이기 위해 진화-생산적 적대 신경망에 기반하여 악성 소프트웨어의 다양한 변형을 사전에 생성하는 방법과 악성 소프트웨어를 자동으로 분류하는 방법을 제안하였다. 해당 방법은 기존의 다른 모델들과 비교하여 높은 분류 성능을 보였으며 악성 소프트웨어의 변형을 생성해서 학습한다는 점에서 Zero-day 공격 방어를 기대할 수 있다. 추후 모델이 생성하는 데이터를 이용하여 Zero-day 관련 연구를 진행할 예정이다.

[참고문헌]

[1] Kaspersky Lab, "Kaspersky Security Bulletin 2018. Statistics," https://securelist.com/kaspersky-security-bulletin-2018-statistics/89145/, Nov. 2018.

[2] 최선오, 김영수, 김종현, 김익균, “딥러닝을 이용한 악성코드 탐지 연구동향,” 정보보호학회지, 제 27권 3호, pp. 20-26, 2017년 6월.

[3] Nataraj, L., Karthikeyan, S., Jacob, G., and Manjunath, B. S., "Malware Images: Visualization and Automatic Classification," In Proc. IEEE Symposium on Visualization for Cyber Security. ACM, 2011.

[4] Dahl, G. E., Stokes, J. W., Deng, L., and Yu, D., "Large-Scale Malware Classification using Random Projections and Neural Networks," IEEE International Conference on Acoustics, Speech and Signal Processing, 2013.

[5] Pascanu, R., Stokes, J. W., Sanossian, H., Marinescu, M., and Thomas, A., "Malware Classification with Recurrent Networks," IEEE International Conference on Acoustics, Speech and Signal Processing, 2015.

[6] Kabanga, E. K., and Kim, C. H., "Malware images classification using convolutional neural network, " Journal of Computer and Communications, vol. 06, no. 01, pp. 153 - 158, 2018

[7] Wang, C., Xu, C., Yao, X., and Tao, D., " Evolutionary Generative Adversarial Networks," IEEE Transactions on Evolutionary Computation, 2019.

[8] Odena, A., Olah, C., and Shlens, J., "Conditional image synthesis with auxiliary classifier gans," In Proc. International Conference on Machine Learning, 2017.

디지털 점 분장을 통한 딥러닝 기반 얼굴인식 모델 기만공격†

류권상*. 최대선**

공주대학교 융합과학과 대학원, 공주대학교 의료정보학과

Evasion Attack Using Digital Mole-Makeup on Deep Learning-Based Face Recognition Model

Gwonsang Ryu*, Daeseon Choi**

*Graduate School of Convergence Science, Kongju National University,
** Dept. of Medical Information, Kongju National University.

요 약

본 논문은 현실적인 얼굴인식 모델 기만 공격을 위해서, 실제 얼굴에 분장이 가능한 적대적 예제(adversarial examples)를 생성하는 방안을 제안한다. 얼굴인식 모델의 오인식을 유도하기 위한 대부분은 기존 공격 방안들은 디지털 환경에서만 가능한 노이즈를 추가하는 변조 방법으로 실제 얼굴에 적용하기 어렵다. 또한, 최근 실제 얼굴 변조를 통한 공격 방법들은 자연스럽지 못한 LED 조명 및 노이즈 안경을 활용하기 때문에, 사람들에게 변조 여부를 들킬 확률이 높다. 따라서, 제안방안은 얼굴에 점 분장을 통해 실제 얼굴에 분장 가능한 동시에 자연스러운 얼굴 변조를 시도한다. 이를 위해 얼마나 많은 점을 어디에 찍을지 찾고, 점이 부자연스럽지 않도록 색의 변화를 조절하였다. 점 분장 기만 공격 실험을 통해 최대 50개의 점을 찍었을 때 90.32%의 공격 성공률을 보였으며 평균 24.65개의 점으로 타겟 얼굴인식 모델을 속일 수 있었다.

I. 서론

딥러닝 기술은 이미지 분류, 음성 인식, 악성코드 탐지, 자연어 처리 등 다양한 분야에서 주목할만한 발전을 이끌고 있다. 그러나 그 눈부신 성공에도 불구하고, 최근 연구들은 딥러닝 모델이 정교하게 변조된 입력 샘플 즉, 적대적 예제(adversarial examples)에 취약하다는 것을 입증했다[1,2]. 다시 말해, 원본 데이터에 노이즈를 추가함으로써 모델의 오인식을 유도하는 기만 공격이 가능한 것이다. 추가되는 노이즈는 매우 적은 양이므로 사람들은 적대적 예제의 변조 여부를 눈으로 인식하기 어렵다. 최소한의 노이즈로 최대한의 오인식을 유도하는 것이 기만 공격의 두 가지 목표인 것이다.

본 논문은 현실적인 얼굴인식 모델 기만 공격을 위해서, 실제 분장이 가능한 적대적 예제를 생성하는 방안을 제안한다. 따라서, 다음과 같은 두 가지 목표를 동시에 만족할 수 있어야 한다: 1) 실제 사람 얼굴에 분장이 가능한 변조를 통해 타겟 얼굴인식 모델을 기만한다, 2) 해당 분장은 사람들이 이상하게 느끼지 않도록 자연스러워야 한다. 목표를 만족하기 위해 제안방안은 얼굴에 점을 찍는 분장을 통한 공격을 시도했다. 우선, 공격에 성공하기 위해 점을 찍는 위치의 우선순위를 정하고 해당 우선순위에 따라 점의 개수를 늘리며 공격을 시도한다. 또한, 일반적인 노이즈처럼 보이는 점을 자연스러운 색으로 변환하는 과정을 거쳤다. 물론 이 과정에서도 공격 성공은 유지되어야 한다.

† 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원(No.2016-0-00173, 핀테크 서비스 금융사기 방지를 위한 비대면 본인확인)을 받아 수행된 연구임

**교신저자: 최대선 (sunchoi@kongju.ac.kr)

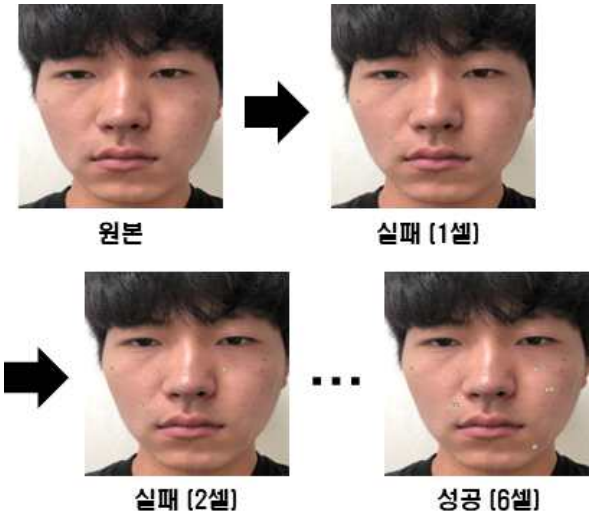


그림 1. 노이즈 점 증가 공격의 예

II. 제안방안

2.1 노이즈 점 증가 공격

혼란을 피하기 위해 제안방안의 범위를 명확히 하면, 얼굴 이미지를 통한 디지털 환경에서의 공격 방법을 제안하는 것이다. 다만 실제 얼굴에서의 분장이 가능하도록 이미지 전체에 노이즈를 추가하는 방식이 아닌 눈에 보이는 점을 추가하는 방식을 택한 것이다. 실제 점 분장을 통한 얼굴인식 모델 기만공격을 위해, 제안방안은 분장 영역 증가 공격을 시도한다. 다시 말해, 공격에 성공할 때까지 점의 개수를 늘리며 얼굴의 다른 영역은 제외하고 점 영역만의 변조를 시도하는 것이다. 실제로 변장할 때는 점만을 찍을 것이기 때문이다. 얼굴에 점을 찍기 위해 우리는 여러 픽셀을 하나의 단위로 묶은 셀의 개념을 사용한다. 예를 들어 5픽셀X5픽셀의 영역을 하나의 셀이라고 지정한다. 공격 이미지 생성 알고리즘은 현재까지 가장 좋은 성능을 보이는 Carlini and Wagner 알고리즘[3]을 활용하여 타겟 클래스를 지정하는 공격 방식을 사용한다. CW 공격 알고리즘은 다음과 같이 정의된다.

$$\text{minimize} \ \| x' - x \|_2^2 + c \cdot f(x') \qquad (1)$$

여기서, f는 [16]에서 정의한 7가지의 목적함수 (objective function) 중 가장 성능이 좋았던 목적함수이고, 다음과 같이 정의된다.

$$f(x') = \max(\max\{Z(x')_i : i \neq t\} - Z(x')_t, -k) \qquad (2)$$

여기서, $Z(x')$은 적대적 예제를 타겟 딥러닝 모델에 입력하였을 때, 적대적 예제에 해당하는 출력 logits를 의미한다. 우선순위는 CW 공격 알고리즘으로 생성된 적대적 예제에 대해 노이즈 값의 합을 구해 높은 값을 가지는 위치에 노이즈 점으로 공격을 시도한다.

2.2 노이즈 점 색 조정

그림 1을 보면 공격 셀이 부자연스럽게 보이는 것을 알 수 있다. 셀 내부의 각 픽셀 색이 제멋대로이므로 어지럽게 노이즈처럼 보이는 것이다. 이는 부자연스러움 뿐만 아니라 실제 점 분장도 어렵게 하는 요인이 된다. 따라서 셀의 색을 좀 더 자연스럽게 조정할 필요가 있다. 이른 위해 제안방안은 공격 알고리즘의 손실 함수에 total variation loss (TV_loss)를[4] 추가한다. 해당 손실 함수는 주변 픽셀과의 색 차이를 감소시켜 주는 역할을 수행한다. 따라서 공격 알고리즘의 손실 함수는 다음과 같다.

$$Loss = Loss1 + \alpha Loss2$$
$$Loss1 = softmax\ cross\ entropy(pred, target) \qquad (3)$$
$$Loss2 = \sum_{i,j}\left((r_{i,j} - r_{i+1,j})^2 + (r_{i,j} - r_{i,j+1})^2\right)^{\frac{1}{2}}$$

loss2가 TV_loss에 해당하며 α는 TV_loss의 반영 비율을 나타낸다.

III. 점 분장을 통한 기만공격 실험

본 장에서는 자연스러운 점 분장을 통해 딥러닝 기반 얼굴인식 모델을 속이는 기만 공격에 대한 실험을 설명한다. 본 실험에서는 공격자가 공격 대상 얼굴인식 모델에 대한 모든 정보를 알고 있는 상태에서 공격을 수행하는 화이트 박스 공격을 수행하였다.

3.1 얼굴 이미지 수집 및 전처리

공격대상 얼굴인식 모델을 학습하기 위해, 연구실 9명 (남자 7명, 여자 2명)에 대해 하루에 50장 이상 12일 동안 얼굴 이미지를 수집하였다. 얼굴 이미지는 각도, 조명, 표정 등 다양한 변화를 주며 수집되었으며, 각자 사용하고 있는 스마트폰을 통해 사람마다 약



그림 2. 전처리된 얼굴 이미지

500장의 얼굴 이미지가 수집되었다. 수집된 얼굴 이미지에서 얼굴 추출 및 정렬하기 위해 사전에 학습된 Multi-task CNN (Convolutional Neural Network) [5] 을 사용하였으며, 전처리한 얼굴 이미지의 크기는 224×224로 통일시켰다. 그림 2는 전처리된 얼굴 이미지의 예를 보여준다.

3.2 공격대상 얼굴인식 모델 학습

전처리한 얼굴 이미지 데이터 세트를 이용하여 공격대상 얼굴인식 모델을 학습하였다. 전처리한 얼굴 이미지 데이터 세트에 대해서 학습 데이터와 테스트 데이터로 7:3으로 나눈 후 학습 데이터에 대해서만 좌우 반전시켜 데이터를 증폭시켰다. 공격대상 얼굴인식 모델은 50개의 레이어 (layer)를 가진 ResNet 버전 2 [6] 를 이용하여 학습하였으며 손실 함수로는 소프트맥스 크로스 엔트로피 (softmax cross entropy)에 0.1×Ring loss [7]을 더하여 사용하였다. 그림 2는 소프트 맥스 크로스 엔트로피와 소프트맥스 크로스 엔트로피에 Ring loss를 추가로 사용하여 학습된 딥러닝 모델에서 추출된 특징 분포의 차이를 보여준다. 학습된 공격대상 얼굴인식 모델은 테스트 데이터 셋에 대해 99.77%의 정확도를 보였다.

3.2 실험 결과

테스트 데이터에서 사람마다 한 장의 얼굴

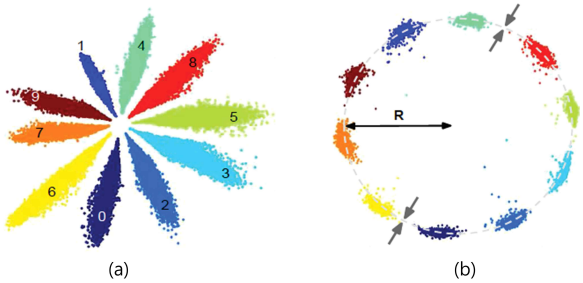


그림 3. MNIST 데이터 세트를 손실 함수로 (a) 소프트맥스 크로스 엔트로피 (b) 소프트맥스 크로스 엔트로피+Ring loss로 학습된 딥러닝 모델에서 추출된 특징 분포에 대한 예 [7]

이미지를 선택하여 다른 8명으로 인식하도록 얼굴인식 모델에 대해 표적 공격을 수행하였다. 또한, 디지털 노이즈 점 크기는 5×5 픽셀로 설정하였으며 최대 50개 점까지 점 개수를 증가시켜가면서 공격을 시도하였다.

표 1. TV_loss 반영 비율에 따른 점 분장을 통한 기만 공격 성공률 비교

		TV_loss×0.1	TV_loss×0.5
점 10개 공격 성공률		30.65%	11.11%
점 20개 공격 성공률		45.16%	38.89%
최대 50개	성공률	90.32%	83.33%
	평균 점 개수	24.65	28.34

최대 점 개수를 10개로 제한하고 TV_loss를 10% 반영한 결과 30.65%의 공격 성공률을 보였으며 TV_loss를 50% 반영한 결과 11.11%의 공격 성공률을 보였다. 최대 점 개수를 20개로 제한하였을 때, TV_loss를 10% 반영한 결과 45.16%의 공격 성공률을 보였으며 TV_loss를 50% 반영한 결과 48.89%의 공격 성공률을 보였다. 최대 점 개수를 50개로 제한하였을 때, TV_loss를 10% 반영한 결과 90.32%의 공격 성공률을 보였으며 TV_loss를 50% 반영한 결과 83.33%의 공격 성공률을 보였다. 또한, 최대 50개로 제한하였을 때 TV_loss를 10% 반영한 결과 공격 성공에 필요한 평균 점 개수는 24.65개였으며 TV_loss를 50% 반영한 결과 공격 성공

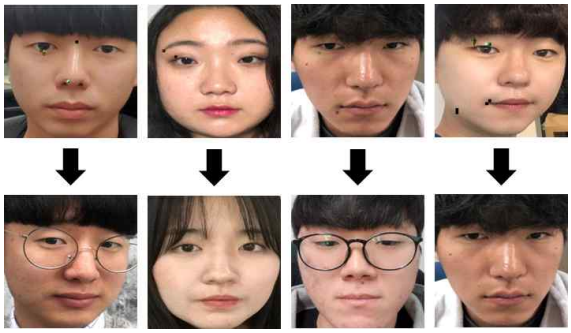


그림 4. 점 분장 기만 공격 성공 예

에 필요한 평균 점 개수는 28.34개였다.

따라서, TV_loss를 많이 반영할수록 공격 성공률은 낮아지고 공격 성공에 필요한 점 개수를 더 늘어났으며 최대 점 개수가 많을수록 공격 성공률이 더 높아지는 것을 볼 수 있었다. 그림 4는 점 분장을 통한 딥러닝 기반 얼굴인식 모델을 속이는 기만 공격에 성공한 예를 보여준다.

IV. 고찰

본 논문은 공격자가 타겟 얼굴 모델에 대한 모든 정보를 알고 있는 상태에서 공격하는 화이트박스 공격을 수행하였다. 최대 점 개수를 50개로 한정하여 공격을 수행하였을 때 90.32%의 공격 성공률을 보였으며 공격 성공까지 필요한 평균 점 개수는 24.65개였다. 하지만, 실제 분장 공격할 때 다른 사람들 눈에 띄지 않기 위해서 점의 색을 조절하는 것도 중요 하지만 적은 개수의 점으로 딥러닝 기반 얼굴인식 모델을 속일 수 있어야 한다. 또한, 공격자가 타겟 얼굴인식 모델에 대한 정보를 알지 못하는 상태에서 공격하는 블랙박스 공격에서는 공격 성공률이 현저히 떨어질 것으로 예상된다.

V. 결론

본 논문에서는 딥러닝 기반 얼굴인식 모델을 속이기 위해 실제 분장이 가능한 공격 이미지를 생성하는 방안을 제안하였다. 제안방안의 영역 증가 공격을 시도함으로써 점이라는 작은 영역의 변조를 통해 공격 실험을 수행하였으며, 최대 50개의 점을 찍었을 때 90.32%의 공격 성공률을 보였으며 평균 24.65개의 점으로 타겟 얼굴인식 모델을 속일 수 있었다. 본 연구를 기반으로 실제 얼굴에 점을 찍어 딥러닝 기반 얼굴인식 모델을 속이는 연구를 수행할 예정이다.

[참고문헌]

[1] C. Szegedy, et al., Intriguing properties of neural networks. International Conference on Learning Representations, Banff, Canada, 2014.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, Explaining and harnessing adversarial examples. International Conference on Learning Representations, San Diego, CA, 2015.

[3] N. Carlini, and D. Wagner, Towards evaluating the robustness of neural networks. IEEE Symposium on Security and Privacy, 2017.

[4] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[5] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks,” IEEE Signal Processing Letters, 23(10), pp. 1499-1503, 2016.

[6] K. He, X. Zhang, S.Ren, and J. Sun, “Identity Mappings in Deep Residual Networks,” European Conference on Computer Vision, Springer, pp. 630-645, 2016.

[7] Y. Zheng, D.K. Pal, and M. Savvides, “Ring loss: Convex Feature Normalization for Face Recognition,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp. 5089-5097, 2018.

위협 인텔리전스를 이용한 위협 경보와의 연관분석 및 결과구축 방법론

이종진, 배근우, 한인성, 조성영*

*국방과학연구소 제2기술연구본부

Correlation analysis and deployment with threat alert using cyber threat intelligence

Jongjin Lee, Geunwoo Bae, Insung Han, Sungyoung Cho*

*Agency for Defense Development

요 약

본 논문에서는 여러 보안 센서에서 발생한 위협 이벤트를 SIEM(Security Information and Event Management)에서 수집하고 상관분석함으로써 생성된 위협 경보에 대해, 연관된 사이버 위협 인텔리전스를 분석하는 방법을 제안한다. 먼저 SIEM에서 생성하는 위협 경보의 데이터 모델과 위협 인텔리전스의 데이터 모델을 정의한 후, 두 데이터 모델이 가지는 속성들을 비교 분석하는 연관분석 방법론을 제안한다. 마지막으로 연관분석 결과 중 의미 있는 정보를 향후에도 연관분석할 수 있도록 위협 인텔리전스를 구축하는 방안을 제안한다. 이를 통해 SIEM에서 지속적으로 발생하는 위협 경보에 대해 보다 넓은 컨텍스트를 제공하여 현재 직면한 사이버 위협에 대해 분석가가 보다 통찰력 있게 대응할 수 있도록 돕는다.

I. 서론

사이버 공격은 점점 고도화되고 지능적으로 진화하고 있다. 기업과 기관, 군에서는 직면하고 있는 사이버 공격을 방어하기 위해서 방화벽과 IPS과 같은 네트워크 경계 보안 센서를 설치하고 있으며, 최근 엔드포인트 보안을 강화하기 위해 EDR(Endpoint Detection and Response) 등의 엔드포인트 보안 솔루션을 설치, 운영하고 있다. 보안 관제 차원에서는 SIEM(Security Information and Event Management)를 운영하여 네트워크 경계 보안 센서 및 엔드포인트 보안 솔루션들로부터 발생하는 보안 이벤트(event) 및 로그들을 수집하고 있다.

그러나 수집된 보안 이벤트 및 로그에는 많은 오탐(false positive)을 포함하고 있으며, 분석해야 할 보안 이벤트가 실제 위협에 비해 과도하게 발생한다. 또한 인력의 수급 및 전문성 등의 격차로 인하여 실제 위협이 발생하였을 위협 이벤트 간 상관분석을 수행하는 과정에 오랜 시간이 걸린다. 이를 극복하기 위해 사이버 위협 인텔리전스를 수집하여 새로운 위협과 관련된 정보를 검색하여 분석한다[1].

본 논문에서는 위협 인텔리전스와의 연관분석을 통해 위협 경보에 대한 상세 정보를 제공하는 모델을 제안한다. 위협 경보(alert)는 사전에 정의된 룰셋에 의해 SIEM에서 수집한 보안 이벤트 및 로그 간의 상관분석을 통해 생성된 정탐(true positive)으로 판명된 보안 이벤트 및 로그 정보를 융합한 상위 개념의 경보를 말한다. 이는 위협 인텔리전스와의 연관분석을 위해 검토해야 할 보안 이벤트 및 로그의 양을 감소시킨다.

2장에서는 위협 경보의 데이터 모델과 위협 인텔리전스의 데이터 모델을 제시한다. 위협 경보 데이터 모델은 다이아몬드 모델[2]을 기반으로 설계되었으며 위협 인텔리전스 데이터 모델은 MISP[3]에서 제시하는 데이터모델을 인용하였다. 3장에서는 위협 경보의 데이터 모델과

MISP의 데이터 모델을 이용하여 위협 경보와 위협 인텔리전스와 연관분석하는 방법을 제시한다. 4장에서는 연관분석 결과를 위협 인텔리전스에 저장하여 추후 발생할 위협 경보와의 연관분석에 사용하게 된다.

II. 데이터 모델

2.1 위협 경보 데이터 모델

2.1.1 Diamond model

다이아몬드 모델[2]은 사이버 침해사고를 판단하는 기본 단위인 이벤트(event)를 공격자(adversary), 공격 능력(capacity), 공격 인프라(infrastructure), 피해자(victim)의 네 가지 핵심 요소로 구성한다. 각 요소들은 다이아몬드 형태로 구성할 수 있기 때문에 다이아몬드 모델이라 불린다. 추가로 메타 특징(meta-feature)을 정의하여 이벤트의 통합을 지원한다. 위협 이벤트 간의 상관관계를 분석하고 위협 인텔리전스와 연관분석을 수행하는데 적합한 모델이다.

2.1.2 위협 경보 데이터 모델 설계

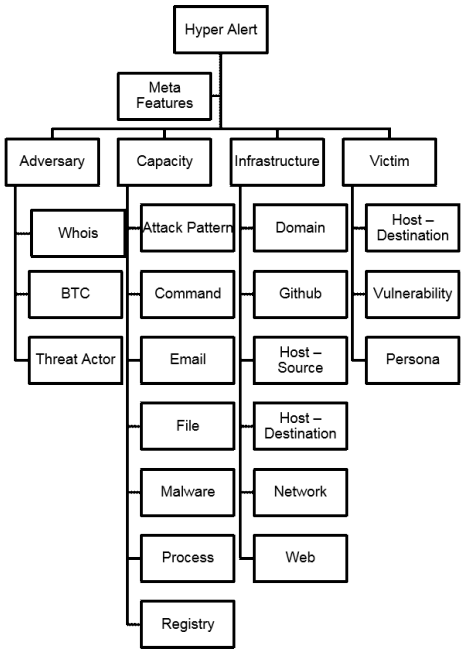


그림 1. 위협 경보 데이터 모델

본 논문에서는 다이아몬드 모델을 기반으로 위협 경보에 대한 데이터 모델을 설계하였다. 탐지된 위협 경보는 다이아몬드 모델을 기반으로 공격자, 공격 능력, 공격 인프라, 피해자의 4가지 요소로 구성되며 분류되지 않는 항목은 다이아몬드의 메타 특성에 대응되는 기본 정보로 분류된다. 이는 그림 1과 같이 주요 특징들을 분류하였으며, 세부 항목에 대한 예시는 표 1과 같다.

표 1. 위협 경보 데이터 모델 예시

분류	예시
Attack Pattern	• technique_name • action_name
Host-Source	• source_ip • source_port
Host-Destination	• destination_ip • destination_port

2.2 위협 인텔리전스 데이터 모델

2.2.1 MISP

MISP(Malware Information Sharing Platform)는 위협 인텔리전스 공유 플랫폼 중 하나로 유럽 연합(EU)의 재정적 지원 하에 룩셈부르크 침해대응센터(Computer Incident Response Center Luxembourg)에서 개발, 유지하고 있다. 표적 공격의 중요 침해 지표를 수집 및 공유하고 사기에 사용되는 취약점 또는 재무 지표와 같은 위협 정보도 제공한다[4]. 오픈소스 도구로 제공되기 때문에 누구나 쉽게 접근하여 사용 가능하다.

2.2.2 MISP 데이터 모델

본 논문에서는 MISP에서 제시하는 데이터 모델을 사용한다. MISP 데이터 모델은 Core format, Taxonomy, Galaxy, Object 네 가지로 구성된다[3]. Core format은 침해사고 및 악성코드 분석 결과의 보고서 형태와 같은 이벤트(event)와 이들에 포함된 침해지표, 방어방책(course of action), 참고자료 등과 같은 속성(attribute)들로 구성되어 있으며 속성은 정해진 유형과 값으로 이루어져 있다. Object는 속성의 집합으로 실제 사이버 환경에서 관련 있는 속성들을 묶어서 새로운 Object를 정의한다. Taxonomy는 이벤트에 대한 태그를 정의한 라이브러리를 제공한다. Galaxy는 클러스터를 표현하기 위한 방법이다.

후술한 연관분석에서는 위협 인텔리전스의

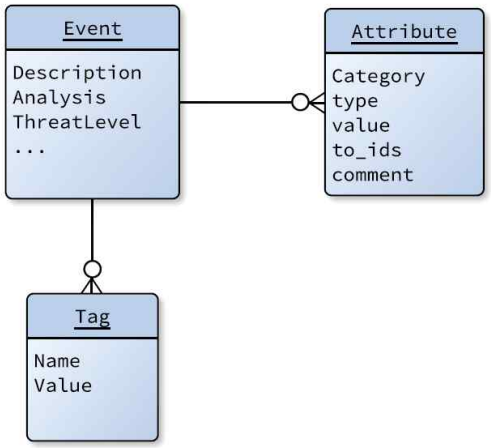


그림 2. MISP 데이터 모델[4]

이벤트에 대한 실질적인 정보가 들어있는 core format을 중점으로 분석한다.

III.위협 경보와 인텔리전스 연관분석

2장에서 위협 경보에는 여러 데이터 속성들이 정의되어 있으며 각 속성들에 매치되는 위협 인텔리전스를 검색하여 위협 인텔리전스들을 위협 경보에 통합하는 방식으로 연관분석을 수행한다.

표 2. 위협 경보의 데이터 모델과 위협 인텔리전스(MISP)의 데이터 모델 매칭

	위협 경보와 연관된 속성	의미 있는 MISP 속성	예시
Type 1	o	o	• filename • regkey • domain|ip
Type 2	x	o	• whois • campaign-id
Type 3	x	x	• gender • nationality • passport

이를 위해 2장에서 살펴 본 MISP의 데이터 모델의 속성은 표 2와 같이 위협 경보의 데이터 모델의 속성과 매치되는지 여부와 실제 분석가에게 의미 있는지 여부의 2가지 기준을 이용하여 3가지 유형으로 분류하였다. Type 1은 연관분석에 의미가 있는 속성이면서 위협 경보의 데이터 모델에 정의된 속성이다. Type 2는 연관분석에 의미가 있는 속성이면서 위협 경보의 데이터 모델에 정의되지 않은 속성이다. Type 3은 연관분석에 의미가 없는 속성이면서 위협 경보의 데이터 모델에도 정의되지 않은 데이터 속성이다.

위 분류에 따라 위협 경보에 정의된 데이터 필드의 값을 위협 인텔리전스에서 검색할 때 Type 1의 MISP의 데이터 모델의 속성을 검색한다. 그리고 매칭된 위협 인텔리전스를 반환할 때에 Type 1과 Type 2에 해당하는 MISP의 데이터 모델의 속성을 JSON 형태로 반환한다. 이러한 연관분석 방법을 그림 3의 시퀀스 다이어그램에 정리하였다.

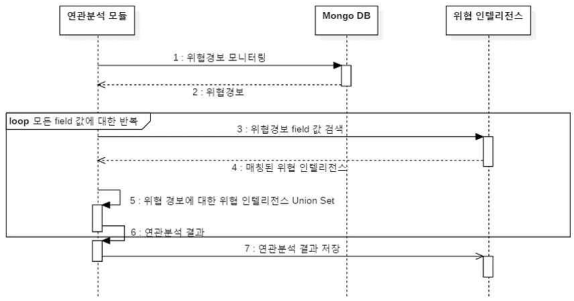


그림 3. 위협 경보와 위협 인텔리전스 연관 분석 방법을 정의한 시퀀스 다이어그램

연관분석은 위협 경보가 생성되는 시점에 수행된다. 위협 경보의 데이터 모델의 속성의 값에 대해 위협 인텔리전스를 검색하여 매칭되는 위협 인텔리전스를 반환하고, 반환된 위협 인텔리전스는 중복된 값을 제외하고 통합되어 연관분석 결과로 저장된다. 연관분석 결과는 위협 인텔리전스의 형태로 저장되며 분석가의 판단하에 위협 인텔리전스 데이터베이스에 저장될 수 있다.

IV.위협 인텔리전스 구축

지속적으로 연관분석에 활용할 위협 인텔리전스는 외부 및 내부로부터 위협 인텔리전스를 수집하여 저장한다. 외부에서는 그림 4와 같이 MISP 플랫폼을 포함한 다른 위협 인텔리전스 플랫폼(threat intelligence platform)을 통해 공유되는 위협 인텔리전스를 수집하고 STIX (Structured Threat Information eXpression) 형태로 변환하여 데이터베이스에 저장한다. 해당 기능은 백그라운드에서 동작하여 자동으로 수집할 수 있다.

내부에서는 단일 위협 경보에 대한 연관분석 결과를 저장하는 것과 여러 위협 경보에 대해 시나리

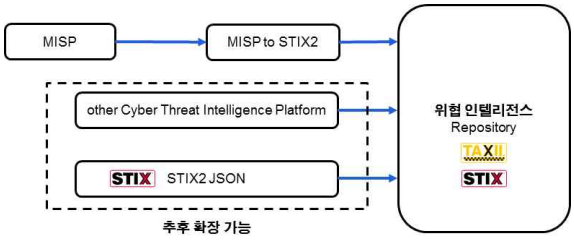


그림 4. 외부 위협 인텔리전스 플랫폼을 이용한 위협 인텔리전스 구축

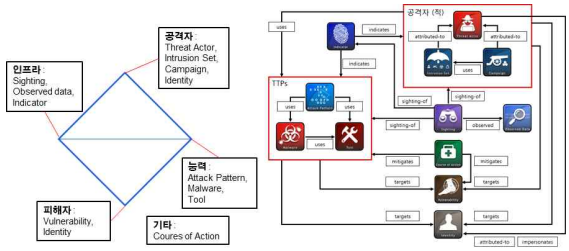


그림 5. 위협 경보와 위협 인텔리전스 연관분석 결과를 이용한 위협 인텔리전스 저장

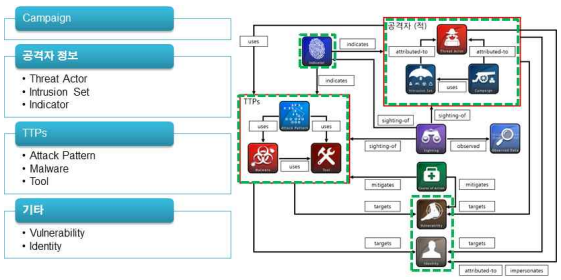


그림 6. 공격 캠페인 분석 결과를 이용한 위협 인텔리전스 저장

오를 만들어 인텔리전스에 저장하는 방안이 있다.

먼저 단일 위협 경보에 대한 연관분석 결과는 그림 5와 같다. 이는 2장에서 정의한 위협 경보에 대한 데이터 모델이 다이아몬드 모델을 기반으로 정의되어 있으며, 이를 STIX 데이터 모델에 매칭하여 향후 위협 인텔리전스 연관분석에 활용할 속성만 사용자가 지정하여 저장할 수 있다.

여러 위협 경보로 구성된 재구성된 공격 캠페인 분석 결과는 그림 6과 같이 정의된 항목들을 STIX 데이터 모델에 매칭하여 향후 위협 인텔리전스 연관분석에 활용할 속성만 사용자가 지정하여 저장할 수 있다. 내부적으로 구축된 위협 인텔리전스는 MISP 플랫폼을 포함한 위협 인텔리전스 플랫폼을 통해 공유할 수 있다.

구축된 위협 인텔리전스를 본 논문에서는 MISP 데이터 모델으로 저장한다고 제시하였다. 현 시점에서는 JSON 형태의 STIX(Structured Threat Information eXpression) 2.0 규격이 사실상의 표준으로 정의되고 있기 때문에 MISP 데이터 모델을 STIX 데이터 모델로 변환하여 구축하는 방안을 제안한다.

V. 결론

시간이 경과함에 따라 복잡해지는 사이버 공격이 발생함에 따라 이를 분석하는 데 소요되는 시간이 점점 증가하고 있다. 사이버 공격을 대응하기 위해 실시간으로 분석하는 방법 또한 발전하고 있으며 분석 대상이 되는 모든 데이터를 실시간으로 수집, 분석, 대응하는 것이 요구되고 있다. 때문에 여러 위협 이벤트들을 하나의 데이터 모델에 융합하고 위협 인텔리전스와 연관분석할 수 있도록 함으로써 분석가가 분석에 필요한 시간을 감소시키고 위협에 대한 신속한 대응이 가능할 것으로 기대한다.

[참고문헌]

[1] G.Settanni, Y.Shovgenya, F.Skopik, R.Graf, M.Wurzenberger, R.Fiedler, “Acquiring cyber threat intelligence through security information correlation”, 2017 3 IEEE International Conference on Cybernetics (CYBCONF), pp. 1-7, 2017.

[2] Sergio Caltagirone, Andrew Pendergast and Christopher Betz, "The diamond model of intrusion analysis", Center For Cyber Intelligence Analysis and Threat Research Hanover Md, 2013.

[3] MISP Core Format, https://www.misp-project.org/datamodels/ (2019. 11. 8. 확인)

[4] C.Wagner, A.Dulaunoy, G.Wagener, A.Iklody, “MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform”, Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security (WISCS‘ 16), pp. 49-56, 2016.

음성 인식 시스템에 대한 공격과 방어*

김예은† , 이연준‡ , 오희국
한양대학교 컴퓨터공학과

Attacks and Defenses on Voice Interface Systems: A Survey

Ye Eun Kim, Yeonjoon Lee, Heekuck Oh*
*Department of Computer Science and Engineering, Hanyang University.

요 약

음성 인식을 이용한 시스템들이 최근 크게 발전하고 있다. 이에 음성 인식 시스템들은 여러 공격의 표적이 되고 있으며, 이에 맞선 다양한 방어 전략 또한 나오고 있다. 음성 인식 시스템은 현재 금전 및 개인 정보를 다루는 영역까지 점점 그 분야를 넓히고 있으며, 이는 음성 인식 시스템에 대한 보안의 필요성을 입증한다. 이에 본 논문에서는 2017년부터 2019년 현재까지 발표된 음성 인식 시스템을 겨냥한 공격 전략과 방어 전략을 조사 및 정리함으로써, 현 음성 인식 시스템 보안의 지표를 제공한다.

I. 서론

기술이 발전하며 인간과 컴퓨터가 상호 작용을 할 수 있는 방법 또한 다양해지고 있다. 사용자 음성 인터페이스, 즉 VUI(Voice User Interface) 역시 그 방법 중 하나이다. VUI는 음성 인식을 이용하여 사용자가 컴퓨터에게 명령을 내리면, 컴퓨터가 그 명령을 수행할 수 있도록 만들어주는 인터페이스이며, 최근 아마존사의 알렉사(Alexa), 구글사의 구글 어시스턴트(Google Assistant), 애플사의 시리(Siri), 마이크로소프트사의 코르타나(Cortana) 등 이런 인터페이스를 사용한 시스템들이 현대인의 일상에서 적지 않은 비중을 점점 차지해 나가고 있다[8]. 그 중 하나인 알렉사에서 사용 가능한 어플리케이션, '스킬'의 개수는 현 시점 기준으로 대략 5만에 달하며, 스킬 카테고리 또한 24개에 달한다[1]. 그러나 이러한 음성 인식 시스템들의 어플리케이션들은 사용자의 건강 및 피트니스 정보, 쇼핑 정보, 미디어 취향 등 이러한 정보를 담고 있으므로 만일 유출될 시 사용자 특정이 가능해질 위험이 존재하며, 또한 최근 미국의 간편 결제 서비스 페이팔(PayPal)이 알렉사 스킬로 연동되는 것이 가능해지며 이제는 음성 인식 시스템이 금전적 정보를 다루게 됨으로써, 이러한 음성 인식 시스템들의 보안성을 고려해야할 필요성은 더욱 높아졌다[2].

음성 인식 시스템들을 노리는 공격들은 지금도 꾸준히 개발되고 있으며, 이러한 공격들을 막기 위한 방어 전략들 또한 계속하여 개발되고 있다. 이에 본 논문은 2017년부터 2019년 현재까지 발표된 음성 인식 시스템을 겨냥한 공격 전략과 방어 전략을 조사 및 정리하여 기술

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A2C2003045).

† 주저자: yeeunkim1216@gmail.com
‡ 교신저자: yeonjoonlee@hanyang.ac.kr,

한다. 본 논문의 내용은 음성 인식 시스템 보안 연구를 계획하는 연구자들에게 이 분야에 대한 지표를 제공할 것이다.

II. 공격 전략

음성 인식 시스템, 특히 스마트 스피커와 스마트폰을 겨냥한 공격들의 주 핵심 아이디어는 바로 사람은 들을 수 없는 소리를 이용하는 것이다. 이러한 공격을 본 논문에서는 불가청음 신호 공격(inaudible signal attack)이라고 한다. 불가청음 신호 공격에서 중요한 점은 시스템의 사용자가 자신이 사용하는 시스템이 공격에 놓여 있다는 점을 절대 알지 못하도록 신호를 사용해야 한다는 점이다[8].

2.1 음악

스피커에 신호를 은밀히 전송하기 위한 방법은 여러 가지 존재하는데, 그 중 하나는 음악 속에 불가청음 신호를 숨기는 것이다. 음악 속에 불가청음 신호를 숨길 경우 사용자가 신호를 알아채는 것이 어려워지는 효과와 동시에, 음악의 특성 상 인터넷과 라디오로 쉽게 배포하는 것도 가능하다는 장점이 있다. 이러한 공격에는 소리에 미세한 변화를 일으키면 그 변화의 정도가 사람이 알아챌 수 있을 정도로 크지 않더라도 음성 인식에 사용되는 머신러닝 분류기(classifier)가 예상치 못한 방향으로 오작동을 일으킬 수 있다는 사실을 이용하여 스마트 스피커가 공격자가 의도하는 대로 작동하도록 만드는 공격과[8], 스마트 스피커나 스마트폰을 해킹하여 시스템이 음악을 재생하면 음악과 동시에 음파를 방출 및 탐지하여 방 안에 사용자가 어디에 있는지, 어떤 움직임을 보이는지 파악이 가능하도록 만드는 공격이 있다[5]. 이러한 공격들은 현재 음악에 잡음이 섞여 들어가거나, 입력된 소리의 샘플링 정도를 낮추면 성공률이 경감한다는 단점이 존재한다.

2.2 초저주파

공격자가 사용할 수 있는 다른 방법으로는 원하는 명령을 인간은 인식할 수 없지만 음성 인식 시스템은 감지할 수 있는 초저주파로 변환하여 보내는 방법이 있다. 이 방법은 음성 인식 시스템이 음성을 받기 위해 사용하는 마이크 하드웨어 회로의 비선형성을 이용하여, 초저주파로 변환한 명령을 스피커에 보내면 마이크가 받아들인 초저주파를 원래의 명령으로 다시 복원하여 인식하도록 만든다[4]. 현 시점 기준으로 이 공격을 반경 7.62 미터까지에서 실험했을 때 성공한 연구가 존재한다[11]. 초저주파 공격의 단점으로는 성능이 좋거나 불가청음 캔슬링을 사용한 하드웨어를 사용한 시스템에는 효과를 볼 수 없다는 것과, 원 신호와 마이크 내에서 복원된 신호 사이에는 500에서 1000Hz의 큰 주파수 차이를 보이는 현상이 발생하기 때문에 머신러닝을 사용한 분류기가 쉽게 탐지 가능하다는 점이 있다.

2.3 음향 심리학

초저주파 공격과 반대로 인간이 들을 수 있는 음역대 안에서, 인간이 인식하기 특히 어려운 특정 소리를 이용하여 공격하는 방법이 존재한다. 이 방법은 음원이 여러 개 존재할 때 인간의 뇌는 주로 한 음원에 집중하기 때문에 관련이 없다고 판단한 소리는 쉽게 걸러내게 된다는 사실과, 같은 소리라도 높은 주파수로 들려줄 경우 청각 인식력이 약해진다는 사실을 이용한다. 연구[12]는 이러한 음향 심리학과, 최근 음성 인식 시스템들이 데이터 기반 접근 방법을 차용하며 특정 기능 공학을 포기했다는 사실을 조합하여 시스템 모델에 구애받지 않는 공격을 개발하였다. 이 공격 역시 음악을 이용한 공격과 비슷하게 잡음이 심한 환경에서는 성공률이 떨어진다는 단점이 존재한다.

2.4 빛

연구[13]은 가장 최근에 발표된 최신 공격으로서 레이저 빛을 MEMS 마이크에 쏘았을 때 마이크가 빛을 소리로 인식하는 현상을 발견, 이용하여 레이저 빛으로 음성 인식 시스템에 신호를 주입하는 공격을 개발하였다. 이 공격은 망원 렌즈를 사용하여 레이저의 초점을 맞출 시, 최대 110미터에 달하는 장거리에서도 공격이 가능한 것으로 확인되었다. 이 공격의 단점은 빛을 사용하기 때문에 소리와 달리 불투명한 물체를 통과할 수 없다는 점, 그리고 빛의

조준이 확실해야하기 때문에 스마트 워치, 폰, 타블렛같은 모바일 기기에서는 사용이 힘들다는 점과, 원격 공격이기 때문에 시스템이 활성(liveness) 테스트를 수행할 시 통과가 어렵다는 점이 있다.

III. 방어 전략

음성 인식 시스템 방어 전략의 핵심은 바로 시스템이 받은 명령이 과연 정말로 사용자에게서 온 명령인지 확인하는 것이다. 현재까지 발표된 확인 방법은 크게 세 종류로 나눌 수 있다. 이러한 방어 전략들은 공통적으로 사용자에게 제약 조건을 만족할 것을 요구하는 단점이 존재한다.

3.1 제스처

제스처란 인간이 말의 효과를 더하기 위하여 하는 몸짓이나 손짓을 뜻한다. 연구[3]과 연구[6]은 음파 탐지 기법을 이용하여 음성이 입력되었을 때 음원의 제스처를 탐지한다. 연구[3]은 탐지한 제스처의 결과로 지금 시스템이 놓여있는 공간 내에 명령을 말한 사람이 정말로 존재하는지를 판별하고, 인증을 위해 사용자에게 확인용 제스처를 요구하여 확인할 수 있는 방법을 제시한다. 연구[6]은 연구 대상 플랫폼이 스마트폰이라는 사실과 사람이 말을 할 때 나오게 되는, 사람마다 각각 다른 특정 제스처가 있다는 사실을 이용하여 사용자를 확인한다. 제스처 방어 전략은 사용자가 추가로 움직여야 하거나, 정확한 인식을 위해 스마트폰을 매우 가까이 들어야 하는 등의 제약이 존재한다.

3.2 활성 테스트

아무리 공격자가 인간의 소리를 정교히 구현할 수 있다고 하더라도, 현 기술로는 도저히 구현할 수 없는 범위가 있다. 이 점에 입각하여 시스템에 입력된 소리가 인간이 낸 소리인지, 기계에 녹음된 소리가 재생된 것인지를 구별하는 과정을 활성 테스트라고 한다. 연구[7]은 대부분의 인간은 아무리 복잡한 것을 요구받았다 하더라도 그 요구에 1초 이내에 응할 수 있다는 사실을 이용하여, 인증 방법을 요구 받았을 때 확인 대상이 보이는 반응 속도를 계산하고 인간인지 아닌지를 판별하는 자동 로그인 방지 시스템, 활성 캡차(CAPCHA)를 제안하였다. 연구[10]과 연구[11]은 인간이 아닌 기계에서 나오는 음성은 인간이 낸 음성과 다르게 명확히 낮은 주파수의 음성 파형이 존재한다는 점에 입각한 활성 테스트 방어 전략을 발표하였다. 활성 테스트 또한 인간의 소리가 활성 테스트 기준을 충족한다는 추정에 기반하고 있기 때문에, 특정한 경우에 대하여 테스트가 오판할 수 있다는 단점이 있다.

3.3 명령 인증

연구[9]에서는 음성 인식 시스템이 있는 공간 내에 최소 2개의 마이크가 존재할 가능성이 높다는 사실을 이용하여 명령을 확인한다. 연구[9]가 제안하는 2MA 시스템은 2개의 마이크가 존재할 시 음원의 위치를 파악할 수 있는 DoA(Direction of Arrival)라는 기술과, 입력된 소리의 해쉬값을 계산하는 RSH(Robust Sound Hash)를 사용한 기술이다. 이 기술은 음성 인식 시스템과 그 방에 있을 확률이 높은 다른 모바일 기기(주로 스마트폰) 간의 통신을 통해, 명령이 내려졌을 때 각 기기가 받은 명령이 과연 둘 다 동일한지 아닌지 따지고, 그와 동시에 명령을 내린 음원이 그 모바일 기기에 더 가까이 있을 때에만 그 명령을 진짜 사용자가 내린 명령으로 판단한다. 명령 인증 또한 시스템이 존재하는 공간 안에 또 다른 모바일 기기가 있어야 한다는 점과, 사용자가 그 모바일 기기에 가까이 있어야 한다는 제약이 존재한다.

IV. 결론

본 논문에서는 2017년부터 현재까지 발표된 주요 음성 인식 시스템 보안 공격과 방어 전략들을 조사하였다.

음성 인식 시스템을 겨냥한 공격 전략으로는 음악, 초저주파, 빛, 인간의 청각 인지능력 특성을 이용하여 사용자가 인식할 수 없는 신호를 음성 인식 시스템 내에 심는 전략들이 존재한다. 이들의 주요 단점은 공격 환경에 잡음이 존재하거나(음악, 음향 심리학), 걸러질 수 있는 명확한 특성이 존재하거나(초저주파, 빛), 빛의

특성으로 인해 공격이 가능한 환경이 한정된다는(빛) 것이 있다[4, 5, 8, 11, 12, 13].

음성 인식 시스템을 위한 방어 전략으로는 인간이 소리를 낼 때 나오는 제스처를 사용한 제스처 방어 전략, 인간만이 낼 수 있는 특성의 유무를 검사하는 활성 테스트, 입력된 명령이 과연 사용자에게서 온 명령인지를 검사하는 명령 인증 전략이 있다. 방어 전략 또한 사용자가 만족해야 할 제약 조건이 있다는 단점이 존재한다[3, 6, 7, 9, 10, 11].

[참고문헌]

[1] [Online] https://www.amazon.com/s?i=alexa-skills

[2] [Online] https://www.paypal.com/us/webapps/mpp/amazon-echo-and-alexa

[3] Amr Alanwar, Bharathan Balaji, Yuan Tian, Shuo Yang, Mani Srivastava, “EchoSafe: Sonar-based Verifiable Interaction with Intelligent Digital Agents”, ACM Workshop on the Internet of Safe Things, 2017, 5 pages.

[4] Guoming Zhang et al., “DolphinAttack: Inaudible Voice Commands”, ACM Conference on Computer and Communications Security (CCS), 2017, 14 pages.

[5] Rajalakshmi Nandakumar, Alex Takakuwa, Tadayoshi Kohno, Shyamnath Gollakota, “CovertBand: Activity Information Leakage using Music”, ACM Interact. Mob. Wearable Ubiquitous Technol, 1, 1, Article 87, 2017, 14 pages.

[6] Linghan Zhang, Sheng Tan, Jie Yang, “Hearing Your Voice is Not Enough: An Articulatory Gesture Based Liveness Detection for Voice Authentication”, ACM Conference on Computer and Communications Security (CCS), 2017, 15 pages.

[7] Erkam Uzun, Simon Pak Ho Chung, Irfan Essa, Wenke Lee, “rtCaptcha: A Real-Time CAPTCHA Based Liveness Detection System”, Network and Distributed Systems Security (NDSS) Symposium, 2018, 15 pages.

[8] Xiejing Yuan et al., “CommanderSong: A Systematic Approach for Practical Adversarial Voice Recognition”, 27th USENIX Security Symposium, 2018, 15 pages.

[9] Logan Blue, Hadi Abdullah, Luis Vargas, Patrick Traynor, “2MA: Verifying Voice Commands via Two Microphone Authentication”, Asia Conference on Computer and Communications Security, 2018, 12 pages.

[10] Logan Blue, Luis Vargas, Patrick Traynor, “Hello, Is It Me You’re Looking For? Differentiating Between Human and Electronic Speakers for Voice Interface Security”, ACM Conference on Security & Privacy in Wireless and Mobile Networks, 2018, 12 pages.

[11] Nirupam Roy, Sheng Shen, Haitham Hassaneh, Romit Roy Choudhury, “Inaudible Voice Commands: The Long-Range Attack and Defense”, NSDI, 2019, 15 pages.

[12] Hadi Abdullah, et al., “Practical Hidden Voice Attacks against Speech and Speaker Recognition Systems”, The Network and Distributed System Security Symposium (NDSS) 2019, 2019, 15 pages.

[13] [Online] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, Kevin Fu, “Light Commands: Laser-Based Audio Injection Attacks on Voice-Controllable Systems”, https://lightcommands.com/, 2019

CNN과 바이트 평균을 활용한 데이터 조각 파일타입 분류에 관한 연구

조현수* 김견우* 김재헌* 원종은* 윤현호* 조재현* 김재희* 김영옥** 한철규***

*한국정보기술연구원 차세대 보안리더 양성 프로그램 (Best of the Best),

**현대 자동차, ***LG CNS

A Study on the File Type Classification of Data Fragment Using CNN and Byte Average

Hyunsoo Cho* Gyeonwoo Kim* Jaeheon Kim* Jongeun Won*

Hyunho Yoon* Jaehyeon Cho* Jaehui Kim* Youngok Kim** Cheolkyu Han***

*Best of the Best, Korea Information Technology Research Institute,
**Hyundai Motor Company ***LG CNS

요 약

본 논문은 CNN과 바이트 평균을 활용하여 시그니처가 존재하지 않는 데이터 조각의 파일 타입을 분류하는 연구에 관한 논문이다. 파일 타입마다 압축 방식과 특징이 모두 다르므로 조각 분류 시 기존 단일 기법만으로는 정확도가 낮다. 이를 개선하기 위해 본 논문에서는 CNN을 사용하여 조각을 분류하고, 분류되지 않은 조각들을 BFA 기법으로 분류하는 복합 기법을 제시한다. 본 연구는 복합 기법을 통해 기존 연구의 정확도를 높일 수 있는 방향을 제시하며 이를 활용한다면 높은 정확도로 파일 타입을 분류할 수 있을 것이다.

I. 서론

파일이 저장매체에 단편화되어 저장될 때, 파일시스템의 파일 메타정보가 존재하지 않으면 해당 파일을 복구하는 것은 쉽지 않다[8]. 이러한 파일들을 복구하기 위해 먼저 단편 데이터의 파일 타입을 분류해야 하는데 시그니처가 없는 단편 데이터들의 파일 타입을 식별하는 것은 매우 힘든 일이다. 이러한 문제를 해결하기 위해 통계기법이나 AI를 활용하여 단편 데이터의 파일 타입을 식별하는 연구가 이전부터 진행되어 왔다[5,6,7]. 하지만 파일 타입 분류에 낮은 정확도를 보이며 단편 데이터들을 파일 타입별로 정확히 분류하지 못하고 있다.

따라서 본 논문은 CNN(Convolutional Neural Network)과 바이트 평균을 활용해 시그니처가 없는 단편 데이터의 파일 타입을 분류하는 방법론을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 파일의 데이터만으로 파일 타입을 식별하기 위해 선행된 연구들을 알아본다. 3장에서는 연구 절차에 대해 설명한다. 4장에서는 학습 방법과 분류 정확도를 보여주고, 5장에서는 결론을 내며 본 논문을 끝낸다.

II. 관련연구

가장 대표적인 연구는 파일 타입마다 바이트 빈도가 다른 특성을 이용하여 분류를 시도한 연구이다. BFA(Byte Frequency Analysis), BFC(Byte Frequency Cross-correlation), FHT(File Header/Trailer)로 JPG와 WAV를 포함한 30개 확장자의 분류를 시도하여 각 알고

리즘마다 27.50%, 45.83%, 95.83%의 정확도를 보였다[1].

BFA를 이용한 다양한 연구가 진행되었지만 파일 지문을 생성하거나 교차상관관계의 계산 복잡성 때문에 일부 확장자만 분류 가능한 단점을 보였다. 본 논문에선 BFA와 더불어 CNN을 함께 사용하여 분류 가능한 확장자를 한층 넓힐 것이다.

또한 파일 지문을 Gray-Scale 이미지로 생성해 CNN으로 학습한 연구도 존재한다. 헤더/푸터를 제거하고 4096byte 단위로 나누었고, 확장자 구분 없이 훈련 셋에 약 81만 조각, 시험 셋에 31만 조각으로 연구를 진행하였으며 연구의 결과로 JPG를 포함한 16개 확장자의 분류를 시도하여 평균 70.9%의 정확도를 보였고 TEXT는 3%, JPG 15%, PNG는 21% 등 큰 편차를 보였다[2].

하지만 데이터 구조의 시그니처를 제거하지 않았으며 JPG를 포함한 몇몇 확장자에서 낮은 정확도를 보이며 확장자 간 큰 편차를 보였다. 본 논문에서는 헤더/푸터와 데이터 구조의 시그니처도 제거하며 오직 파일 내용만으로 분류를 시도한다.

III. 연구 절차

3.1 데이터 전처리

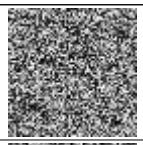
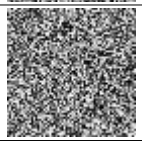
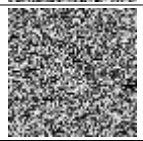
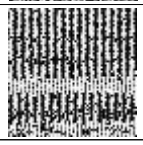
본 연구에서는 파일 타입마다 서로 다른 압축방법이 적용되므로 압축방법에 따라서 특징적인 바이트 값이 등장할 것으로 예상하였다. 이러한 개념증명을 위하여 사진, 동영상, 음성 분야에서 각각 JPG, H.264, WAV를 선정하였고 동영상의 경우 코덱에 따라서 파일의 특성이 정해지므로 파일 타입이 아닌 코덱으로 지정하였다.

또한 BFA로는 다양한 파일 타입에 적용하기 힘든 것을 알게 되었고 데이터 파일을 Gray-Scale 이미지화한 후 CNN 모델을 이용하여 BFA의 한계를 벗어나고자 하였다. Gray-Scale은 0~255까지의 경우의 수로 색을 표현할 수 있는데 4096byte의 데이터를 64x64 픽셀의 이미지로 변환하면 0x00~0xFF까지 총 256개의 바이트와 1 대 1 매핑이 될 수 있으므로 데이터의 손실을 최소화할 수 있다.

3.2 데이터 셋

데이터 셋을 수집하기 위해 우선 인터넷상에서 크롤링을 통해 파일을 무작위로 수집하였다. 이후 파일의 헤더/푸터와 데이터구조에 존재하는 시그니처를 제거하여 완전한 데이터만 추출하고, Windows10, NTFS, 256GB ~ 2TB기준으로 클러스터의 최소 입출력 크기인 4096byte로 자른다. 또한 데이터를 무작위로 섞어 저장하였다. 이러한 방식으로 산출된 파일의 데이터를 1byte마다 1픽셀로 매핑될 수 있도록 64x64 픽셀 Gray-Scale 이미지로 변환한다. 위 방식으로 가공된 이미지는 아래 [표 1]과 같다.

[표 1] 각 파일 타입의 데이터 셋 가공 현황

JPG			
H.264			
WAV			

3.3 분류 방법

파일 타입 분류가 필요한 데이터들을 CNN을 사용하여 우선적으로 분류한다. 본 연구에서는 JPG, H.264, WAV를 분류하며, CNN의 결과로 90% 이하의 정확도를 보이는 파일 타입은 추가적으로 BFA를 사용하여 단편 데이터 분류를 진행한다.

IV. CNN 학습 방법 및 분류 결과

4.1 CNN 학습 방법

CNN 모델의 구성은 세 번의 Convolution과 MaxPooling을 거쳐 Gray-Scale로 학습을 시도한다. Dropout은 학습 때 0.25, Flatten 단계는 0.5로 설정하였고 총 Epoch는 16, batch-size는 128로 설정하였다.

학습 방법은 시그니처가 없는 단편 데이터를 가공하여, 파일 타입마다 훈련 셋을 10,000개씩, 시험 셋을 1,000개씩 무작위로 선정한다. 여기서 훈련 셋과 시험 셋은 완벽히 독립된 상태이다.

훈련은 훈련 셋의 조각 데이터를 폴더로 레이블 하여 진행하고, 시험은 레이블 없이 한 폴더에 무작위로 저장하여 연구를 진행했다.

4.2 분류 결과

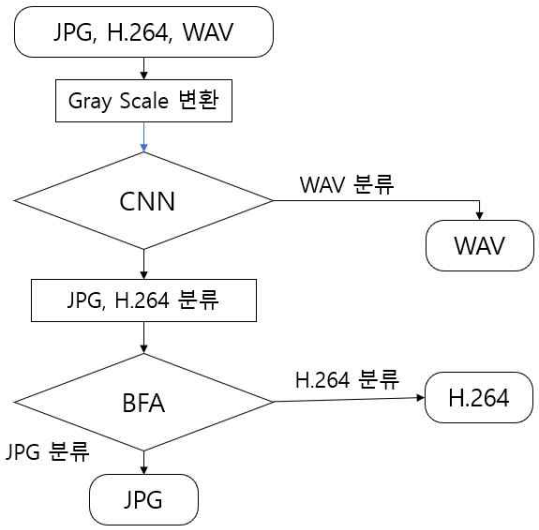
파일 타입 분류를 하였을 때 첫 번째 단계로 진행한 CNN 모델은 WAV를 94.2%의 높은 정확도로 분류했다. JPG의 정확도는 이전 연구에서 각각 25%[1], 15%[3]를 보였는데 본 연구에선 78.5%의 높은 정확도를 보였다. 이에 반해 H.264는 이미지 프레임이 중첩되어 저장하기 때문에 일부 데이터가 JPG로 판별되어 28.1%의 낮은 정확도를 보였다.

따라서 두 번째 단계로 JPG, H.264의 바이트 빈도를 비교해본 결과, JPG는 0x00, 0x20, 0x92의 평균값이 순서대로 약 33개, 24개, 21개로 높게 나왔고 나머지 바이트의 평균은 대체적으로 14~15개의 값을 보였다. H.264는 0x00 만이 18로 높게 나왔으며, 나머지 바이트의 평균은 15~16개에 수렴하였다. 이 차이점을 피처로 추출한 뒤 이어서 분류를 진행하였고 2단계의 분류 결과는 JPG는 97.8%, H.264는 97.4%의 높은 결과를 보였다.

최종적으로 CNN과 BFA를 복합적으로 사용한 최종 결과를 도출해 냈으며 각각의 확장자는 아래 [그림 1]와 같은 정확도를 내었고, 평균 96.46%의 높은 정확도를 확인할 수 있다.

[표 2] 최종 분류 결과

	JPG	H.264	WAV
JPG	97.8%	2.0%	0.2%
H.264	2.1%	97.4%	0.5%
WAV	3.4%	2.4%	94.2%



[그림 1] 단계별 분류 방법

V.결론

본 연구는 BFA+CNN 기법을 이용하여 평균 96.4%의 분류 정확도를 보였으며 동일한 파일 타입을 기준으로 단일 BFA를 이용하여 평균 49.7%, 단일 CNN을 이용하여 평균 64.4%의 정확도를 확인하였고, 복합 기법을 적용하여 각각 46.7%, 32% 개선되는 모습을 보였다.

현재 기술이 발전하면서 효율, 용도 등에 따라 새로운 파일 타입이 개발되고 있으며 데이터 저장 방식, 압축 알고리즘 등에 따라 파일 타입의 특징이 모두 다르므로 단일 기법으로는 모든 파일 타입을 분류할 수 없다.

따라서 다양한 파일 타입의 특징을 고려하여 식별 목적, 선정 파일 타입 등에 따라 다양한 모델을 구성하고 복합으로 설정하면 정확도가 높아질 것이다.

본 연구를 통하여 파일 타입 식별에 대한 연구들이 활성화된다면 파일 복구의 새로운 연구 방향이 시도될 수 있을 것이다. 향후 단편 데이

터의 파일 복구가 수월해진다면 관련 업무 및 포렌식 수사에 큰 도움이 될 것이다.

[참고문헌]

[1] Mason McDaniel and M.Hossain Heydari, “Content Based FIle Type Detection Algorithms.” Hawaii International Conference on System Sciences, pp. 10-19, January, 2003.

[2] Qian Chen, Qing Liao, Zoe L.Jiang, Junbin Fang, Siuming Yiu, Guikai Xi, Rong Li, Zhengzhong Yi, Xuan Wang, Lucas C.K.Hui, Dong Liu and En Zhang, “File Fragment Classification Using Grayscale Image Conversion and Deep Learning in Digital Forensics.” IEEE Symposium on Security and Privacy Workshops, pp. 140-147, May, 2018.

[3] 원종훈, 강민지, 박지성, 김지홍, “Apriori 알고리즘 기반의 N-그램 선정과 이를 이용한 파일 조각 타입 결정.” 한국소프트웨어종합학술대회 논문집. pp. 1459-1461, December, 2018.

[4] Kristijan Vulinović, Lucija Ivković, Juraj Petrović, Kristian Skračić and Predrag Pale, “Neural Networks for File Fragment Clasification.” 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics, pp. 1395-1399, May, 2019.

[5] Tantan Xu, Ming Xu, Yizhi Ren, Jian Xu, Haiping Zhang and Ning Zheng, “A File Fragment Classification Method Based on Grayscale Image.” Journal of Computers, vol. 9, no .8, pp. 1863-1870, August, 2014.

[6] Qiming Li, Alvin Y.Ong, Ponnuthurai N.Suganthan and Vrizlynn L.L.Thing, “A novel support vector machine approach to high entropy data fragment classification.” Proceedings of the South African information security multi-conference, pp. 236-247, January, 2010.

[7] Irfan Ahmed, Kyung-Suk Lhee, Hyun-jung Shin and Man-Pyo Hong, “Fast Content-Based File Type Identification.” 7th Annual IFIP WG 11.9 International Conference on Digital Forensics, pp. 65-75, January, 2011

[8] 김진국, 이승봉, 박정흠, 방제완, 이상진, “파일카빙 기법에 관한 연구.” 디지털포렌식연구, (2), 19-31, June, 2008.

LSTM Neural Network를 기반으로 한 전자문서 보안등급 산정 기준 수립에 관한 연구

김정욱*, 송성욱** 장항배***

*중앙대학교 일반대학원 융합보안학과

**중앙대학교 소프트웨어대학 소프트웨어학부

**중앙대학교 경영경제대학 산업보안학과

A Study on the Classification of Document Security Level Based on LSTM Neural Network

Jeongwook Kim*, Seongwook Kim**, Hangbae Chang***

*Dept. of Security Convergence, Chung-Ang University

**Dept. of Computer Science and Engineering, Chung-Ang University

***Dept. of Industrial Security, Chung-Ang University

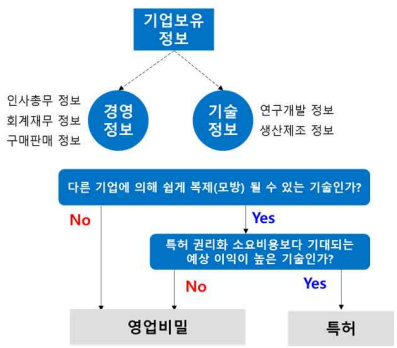
요 약

기업 내부의 정보를 분류하고 가치를 측정하는 작업은 기업에서 가장 중요한 의사결정 프로세스 중 하나이며 이를 위한 분류기를 구현하는 문제는 컴퓨터공학, 문헌정보학, 정보공학 등 여러 분야에 걸쳐 논의 되고 있는 사항이다. 본 연구에서는 순차데이터 처리에서 강점을 보이는 LSTM을 활용해 전자문서의 범주를 기반으로 보안등급을 산정하는 방법에 대하여 논의하고자 한다. 전자문서의 분류는 텍스트 전처리, 특징선택, 분류 예측 단계를 거치며 특징 선택 단계에서 추출된 가중치를 기반으로 LSTM을 통해 미래의 분류를 예측하고자 하며, 연구를 통해 기업이 가지고 있는 특징(Feature) 및 정보문맥(Information Context)에 맞는 정보등급을 산출하는 방법에 대해 논의하고자 한다.

I. 서론

1.1 전자정보의 분류

전자문서의 양이 방대하게 증가함으로 인하여 여러 텍스트로부터 중요한 정보를 추출해내는 텍스트마이닝과 문서의 범주를 구별해내는 분류 문제의 중요성이 지속적으로 증가하고 있다. 기업 내부에서 비정형의 텍스트데이터는 보고서, 도서, 인스턴스 메시지 등 다양한 형태로 존재하며 기업이 보유정보에 대한 보호수단을 채택하기 위해서는 우선 대상 정보가 보호할 가치가 있는지 분류하여야한다. 분류과정에서 타 회사에 의한 모방위협이 크고, 예상 이익이 높은 기술의 경우 특허권 취득을 통해 해당 기술을 보호할 수 있지만, 이외의 정보는 영업비밀로 관리하여야한다. 지금까지 영업비밀의 보안등급 분류는 전문가에 의한 수작업이 주를 이루고 있으며, 보안등급 산정을 위한 평가항목 역시 기업의 특성을 반영하지 못한 공통적인 항목만을 제시하고 있다.

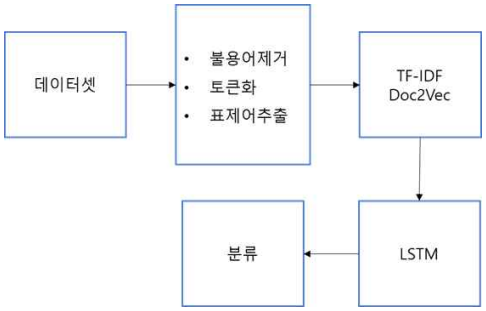


[그림 1] 영업비밀 분류

정확하지 않은 보안등급 분류체계는 보호할 자산에 대한 불명확성을 가져오며 과평가된 보안등급으로 인한 업무의 비효율이 증가하는 결과를 초래하게된다.

1.2 LSTM을 통한 분류

텍스트 범주화 작업은 데이터전처리, 인덱싱, ,특징선택, 분류, 분류결과분석 5단계로 나누어지며 분류과정에서 문서의 문맥에 대한 정보를 고려할 수 있어야한다.



[그림 2] LSTM을 통한 문서 분류 프로세스

이중 문서 내용을 임베딩(Embedding) 하는 기법으로 기존에는 BOW와 TF-IDF 기법을 활용하여 문서 분류를 수행하였으나 단어간의 순서를 고려하지 않는 단점으로 인해 동일한 말뭉치(Corpus)를 포함하고 있는 문서를 같은 범주로 판단하게 되는 특징이 있어 말뭉치 전체의 공기정보(Co-occurrence)를 반영할 수 잇는 Word2Vec 모델을 사용하여 의미론적으로 비슷한 문서들을 군집화하여 TF-IDF와 비교하고자 한다.

II. 결론

이번 연구를 통해 분야가 다른 산업군에 대해 공통적인 보안등급 분류를 수행하는 것이 아닌 각 산업군의 핵심 키워드와 문맥에 맞는 보안등급기준을 생성할 수 있는 분류기를 통해 노동집약적인 내부 문서 분류의 과정의 시간을 줄일 수 있을 뿐만 아니라 분류 품질의 향상을 기대할 수 있다.

[사사]

이 논문은 2019년도 정부(산업통상자원부)의 재원으로 한국산업기술진흥원의 지원을 받아 수행된 연구임(P0008703, 2019년 산업전문인력역량강화사업

[참고문헌]

[1] Li, Yuefeng, et al. "Relevance feature discovery for text mining." IEEE Transactions on Knowledge and Data Engineering 27.6 (2014): 1656-1669.

[2] Kehagias, Athanasios, et al. "A comparison of word-and sense-based text categorization using several classification algorithms." Journal of Intelligent Information Systems 21.3 (2003): 227-247.

[3] Cohen, William W., and Yoram Singer. "Context-sensitive learning methods for text categorization." ACM Transactions on Information Systems (TOIS) 17.2 (1999): 141-173.

[4] Nihar M. Ranjan “Document Classification using LSTM Neural Network” Journal of Data Mining and management Vol 2 Issue 2

NIST 2라운드 Ring LWE 기반 공개키 암호 기법 동향 분석*

박소현[1†] , 이동훈[1‡]

[1]고려대학교 정보보호학과

A Study of Ring-LWE based Public Key Encryption in NIST PQC 2-round submissions*

So-Hyun Park[1†] , Dong-Hoon Lee[1‡]

[1]Division of information security, Korea University.

요 약

양자 컴퓨터에 대한 연구가 활발히 진행됨에 따라 양자 컴퓨팅 환경에서도 안전한 암호 기법들에 대한 연구가 활발히 진행되고 있다. 2016년 미국 국립표준기술연구소(NIST)에서는 양자 내성 공개키 암호, 서명 등에 관한 표준화 공모를 시작하였다. 본 논문에서는 2019년 2라운드에 진출한 17개의 공개키 기반 기법 가운데, RLWE 계열 공개키 기법 4가지를 소개하고, 각 기법의 특징과 비밀키 분포를 중심으로 장·단점에 대해 분석하였다.

I. 서론

양자 컴퓨터 개발 기술의 발전과 동시에 현재 상용되는 많은 공개키 암호 시스템이 양자 컴퓨팅 환경에서 안전하지 않음이 알려지면서 양자 내성 암호에 대한 연구가 활발히 진행되고 있다. 미국 국립표준기술연구소(NIST)에서는 2016년 양자 내성 공개키 암호, 서명 등에 관한 표준화 공모를 시작하였고, 2017년 69개의 후보들이 1차 라운드에 진출하였고, 2019년 17개의 공개키 기법과 9개의 서명 기법, 총 26개의 후보가 2차 라운드에 진출하였다.

2005년 Regev에 의해 처음 제시된 LWE 문제는 비밀키의 분포가 Z_q에서 uniformly 랜덤하였지만, 이후에는 복호화 에러를 줄이기 위해 비밀키가 작은 값을 가지는 분포를 선택하고 있다. 본 논문에서는 2차 라운드에 진출한 공개키 기법 가운데, Ring Learning with errors (RLWE) 계열 중 SABER[1], Kyber[2], LAC[3], Round5[4]를 소개하고 비밀키 분포를 중심으로 특징을 분석한다.

II. 배경지식

2.1 다항식환(Polynomial ring)

차수가 n인 기약 다항식 $f(x) \in Z[x]$에 대하여 다항식환(polynomial ring) $R = Z[x]/\langle f(x) \rangle$을 정의하고, 모듈러(modulo) q에 대한 상환(quotient ring) $R_q = Z_q[x]/\langle f(x) \rangle$를 정의한다.

많은 래티스 기반 공개키 기법들은 해당 기법의 환(ring)을 정의할 때, 2의 지수승인 n에 대하여 $f(x) = x^n + 1$을 선택한다. 주어진 에러 분포와 비밀키 분포에 대해, 복호화시 에러를

* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.
† 주저자, shparg@naver.com
‡ 교신저자, donghlee@korea.ac.kr

가장 작게 하고, 효율적인 구현이 가능하다는 이점이 있기 때문이다. 그러나 이것은 n이 2의 지수승으로 제한된다는 단점을 가지고 있다. 이런 문제를 해결하기 위해서 $f(x)$에 대한 대안들이 제시되었는데, NTRU[7]와 Round5[4]에서는 $f(x)$을 n+1이 소수인 n+1차 원분 다항식(cyclotomic polynomial) $f(x)=x^n+\cdots x+1$으로 선택하였다. 그리고 NTRU Prime[8]에서는 소수 n에 대해 $f(x)=x^n-x-1$으로 선택하였다.

2.2 Ring learning with errors (RLWE)

양의 정수 q가 주어졌을 때, 랜덤한 다항식 $a\leftarrow R_q, s\leftarrow R_q, u\leftarrow R_q$와 각 계수가 이산 정규분포를 따르는 에러 $e\leftarrow GD$에 대하여, 두 분포 $(a, a\cdot s+e)\in R_q\times R_q$와 $(a,u)\in R_q\times R_q$를 구분하는 문제를 결정 RLWE 문제(decisional RLWE problem)라고 한다.

2.3 Ring learning with rounding (RLWR)

$p|q$인 두 양의 정수 p,q가 주어졌을 때, 랜덤한 다항식 $a\leftarrow R_q$와 랜덤한 두 다항식 $s\leftarrow R_q, u\leftarrow R_p$에 대하여, 두 분포 $\left(a, \left\lfloor \frac{p}{q}a\cdot s \right\rfloor\right)\in R_q\times R_p$와 $(a,u)\in R_q\times R_p$를 구분하는 문제를 결정 RLWR 문제(decisional RLWR problem)라고 한다.

2.4 여러 가지 분포들

- **중앙 이항 분포 (Central Binomial Distribution)**

파라미터 μ를 가지는 중앙 이항 분포의 확률 질량 함수(probability mass function)는 $bin_{\mu/2}(k)=\binom{\mu}{k+\mu/2}2^{-\mu}$이다.

- **대칭 삼진 분포 (Symmetric Ternary Distribution)**

대칭 삼진 분포란 $0\le\alpha\le 1$인 α에 대해 $\Pr(0)=1-\alpha$, $\Pr(1)=\Pr(-1)=\alpha/2$인 분포이다.

- **고정된 웨이트를 가지는 삼진 분포 (Fixed-weight Ternary Distribution)**

고정된 해밍웨이트 h에 대해 n차 벡터 x가 이 분포를 따른다면 $n-h$개의 원소들은 0이고, h개의 원소들은 1 혹은 -1이다.

III. NIST 2라운드 Ring-LWE(R) 기반 공개키 기법

3.1 CRYSTALS-KYBER

- **키생성** (1^λ)

① NTT 도메인에서 각 행렬의 원소가 R_q에 있는 2×2 행렬 $A\in R_q^{k\times k}$을 생성한다.

② 중앙 이항분포에서 각 원소를 선택하여 비밀 값 $s\in R_q^k$을 생성한다.

③ 중앙 이항분포에서 각 원소를 선택하여 비밀 값 $e\in R_q^k$을 생성한다.

④ 공개키 $pk=As+e$와 비밀키 $sk=s$를 출력한다.

- **암호화** (pk,m)

① NTT 도메인에서 각 행렬의 원소가 R_q에 있는 2×2 행렬 $\hat{A}\in R_q^{k\times k}$을 생성한다.

② 각 원소가 중앙 이항분포를 따르는 난수 벡터 $r\in R_q^k$를 생성한다.

③ 각 원소가 중앙 이항분포를 따르는 에러 벡터 $e_1\in R_q^k$를 생성한다.

④ 중앙 이항분포를 따르는 에러 값 $e_2\in R_q$ 을 생성한다.

⑤ $u=A^Tr+e_1 \bmod q$와

$v=(As+e)^Tr+e_2+\lfloor (q/2)\cdot m\rceil \bmod q$를 계산한다.

⑥ 암호문 $c=(\lfloor (2^{d_u}/q)\cdot u\rceil, \lfloor (2^{d_v}/q)\cdot v\rceil)$를 출력한다.

- **복호화** (sk,c)

① $m=\lfloor (2/q)\cdot (v-s^Tu)\rceil \bmod 2$를 출력한다.

3.2 LAC

- **키생성** (1^λ)

① 씨드 $seed_a$로부터 R_q의 랜덤한 원소 a를 생성한다.

② 차수가 n이고, 분산이 σ, 해밍웨이트가 h로

고정된 중앙 이항분포를 따르는 비밀키 $s \xleftarrow{\$} \Psi_\sigma^{n,h}$를 생성한다.

③ 마찬가지로 에러 $e \xleftarrow{\$} \Psi_\sigma^{n,h}$를 생성한다.

④ 공개키 $pk=(seed_a, b=as+e \in R_q)$와 비밀키 $sk=(s)$를 출력한다.

- **암호화** (pk,m)

① 씨드 $seed_a$로부터 공개키 a를 복원한다.

② 메시지 m을 인코딩하여 $\hat{m} \in \{0,1\}^{l_v}$를 생성한다.

③ 각각 $\Psi_\sigma^{n,h}, \Psi_\sigma^{n,h}, \Psi_\sigma^{l_v}$를 따르는 (r,e_1,e_2)를 생성한다.

④ $c_1 = ar+e_1 \in R_q$와

$c_2 = \langle br \rangle_{l_v} + e_2 + \left\lfloor \frac{q}{2} \right\rceil \cdot \hat{m} \in Z_q^{l_v}$를 계산한다.

⑤ 암호문 $c=(c_1,c_2) \in R_q \times Z_q^{l_v}$를 출력한다.

- **복호화** (sk,c)

① $\hat{m} = c_2 - \langle c_1 s \rangle_{l_v} \in Z_q^{l_v}$를 계산한다.

② $\hat{m}$을 디코딩하여 메시지 m을 출력한다.

3.3 Round5

- **키생성** (1^λ)

① 랜덤한 씨드 σ로부터 $a \in R_q$를 생성한다.

② 랜덤한 씨드 sk로부터 고정된 웨이트를 가지는 삼진 분포를 따르는 랜덤한 $s \in R_q$를 생성한다.

③ $b = \left\lfloor \frac{p}{q} \cdot a \cdot s \right\rceil \in R_p$를 계산한다.

④ 공개키 $pk=(\sigma,b)$와 비밀키 $sk=(sk)$를 출력한다.

- **암호화** (pk,m)

① 씨드 σ로부터 $a \in R_q$를 복원한다.

② 고정된 웨이트를 가지는 삼진 분포를 따르는 랜덤한 $r \in R_q$를 생성한다.

③ $u = \left\lfloor \frac{p}{q} \cdot a \cdot r \right\rceil \in R_p$를 계산한다.

④ $v = \left\lfloor \frac{t}{p} Sample_\mu(b \cdot r \bmod \xi) \right\rceil + \frac{t}{b} Encode(m) \in Z_t^\mu$를 계산한다.

⑤ 암호문 $c=(u,v)$를 출력한다.

- **복호화** (sk,c)

① 씨드 sk로부터 $s \in R_q$를 복원한다.

② $y = \left\lfloor \frac{b}{p}\left(\frac{p}{t}v - Sample_\mu(u \cdot s \bmod \xi)\right) \right\rceil$를 계산한다.

③ $\hat{m} = decode(y)$를 출력한다.

3.4 Saber

- **키생성** (1^λ)

① 씨드 $seed_A$로부터 $A \in R_q^{l \times l}$를 생성한다.

② 분산이 $\frac{\mu}{4}$인 중앙 이항분포로부터 비밀키 $s \in R_q^{l \times 1}$를 생성한다.

③ $b = \left\lfloor \frac{p}{q} \cdot A^T s \right\rceil \in R_p^{l \times 1}$를 계산한다.

④ 공개키 $pk=(seed_A,b)$와 비밀키 $sk=(s)$를 출력한다.

- **암호화** (pk,m)

① $seed_A$로부터 A를 복원한다.

② 분산이 $\frac{\mu}{4}$인 중앙 이항분포로부터 난수 $s' \in R_q^{l \times 1}$를 생성한다.

③ $b' = \left\lfloor \frac{p}{q} \cdot As' \right\rceil \in R_p^{l \times 1}$를 계산한다.

④ $c_m = \left\lfloor \frac{T}{p} \cdot (b^T(s' \bmod p) - (p/2)m) \right\rceil \in R_T$를 계산한다.

⑤ $c=(c_m,b)$를 출력한다.

- **복호화** (sk,c)

① $v = b'^T(s \bmod p) \in R_p$를 계산한다.

② $m' = \left\lfloor \frac{2}{p} \cdot \left(v - \frac{p}{t}c_m\right) \right\rceil \in R_2$를 출력한다.

IV. 파라미터 비교 및 분석

4.1 파라미터 비교 및 분석

KYBER와 Saber의 경우, n을 2의 지수승으로 설정해야 하기 때문에, 128-bit 안전성을 가지는 파라미터를 계산하기 위해 n을 512로 설정하였다. 그런데 일반적으로 비밀키가 삼진 분포를 따를 경우 $n=512$인 파라미터는 128-bit보다 낮은 안전성을 가진다. 그래서 안전성을 증가시키기 위해 두 기법 모두 비밀키의 분포를 삼진 분포보다 엔트로피가 큰 중앙 이항 분포

로 설정하였다. 하지만, 그 결과 안전성은 증가하였으나 비밀키의 절대값이 커지면서 복호화 시 발생하는 에러의 크기가 커진다. 그래서 Round5가 $q=2^{11}, p=2^8$를 설정한 것에 반해 Saber는 $q=2^{13}, p=2^{10}$로, KYBER는 $q=3329$로 비교적 큰 모듈러스를 가지게 되었고, 암호문과 공개키의 크기가 증가하였다.

	n	q	p	t	h	ring	Security (quantum)	Error rate	CT (byte)	PK (byte)	Secret distribution	ECC
KYBER	512	3329	-	-	-	$\mathbb{Z}[x]/x^n+1$	100	2^{-178}	736	800	CBD_2	X
LAC	512	251	-	-	256	$\mathbb{Z}[x]/x^n+1$	133	2^{-116}	712	544	$HWT(n,h)$	O
Round5	618	2^{11}	2^8	2^4	104	$\mathbb{Z}[x]/\Phi_{n+1}(x)$	131	2^{-65}	682	634	$HWT(n,h)$	X
Saber	512	2^{13}	2^{11}	2^5	-	$\mathbb{Z}[x]/x^n+1$	114	2^{-120}	736	672	CBD_5	X

[그림 1]. 파라미터 비교

LAC의 경우, $n=512$에서 비밀키가 삼진 분포를 따르지만 에러 정정 코드(Error-correcting code)를 사용하기 때문에 모듈러스로 $q=251$와 같은 작은 값을 설정할 수 있어 128-bit 안전성을 만족할 수 있다.

Round5의 경우, $n+1$이 소수라는 조건을 가지기 때문에 512와 1024 사이에 많은 n의 후보가 존재한다. 따라서 128bit 나 192bit 안전성을 가지는 파라미터를 설정하고자 할 때, 적절한 n 값을 선택할 수 있다. 하지만 다항식 간의 덧셈과 곱셉에 모듈러 n+1차 원분 다항식 $(=1+x+\cdots+x^n)$을 하기 때문에 모듈러 x^n+1를 하는 경우에 비해 복호화 시 발생하는 에러가 2배가 된다. 이를 해결하기 위해 Round5에서는 $x^{n+1}-1=(x-1)(1+x+\cdots x^n)$이라는 사실을 이용하여 $Z_p[x]/\Phi_{n+1}(x)$ 대신 $Z_p[x]/x^{n+1}-1$에서 복호화를 함으로써 에러를 줄이는 방법 또한 제시하였다. 그러나 $Z_p[x]/x^{n+1}-1$에서 올바른 복호화가 진행되기 위해서는 비밀키 s와 암호화시 사용되는 난수 r이 모두 $(x-1)$의 배수가 되어야 한다는 조건이 필요하기 때문에, s와 r 생성 이후에 각 다항식에서의 1과 -1의 개수를 같게 만드는 추가적인 과정이 요구된다.

본 논문과 Bhattacharya등이 발표한 [6]의 논문에서도 알 수 있듯이, RLWE계열 암호 기법에서는 비밀키 다항식의 각 계수가 $\{-1,0,1\}$의 원소로 삼진 분포를 따를 때 중앙 이항 분포나 이산 정규분포보다 더 효율적인 파라미터를 도출해낼 수 있다. 같은 n에 대하여 비밀키가 삼진 분포를 따를 경우 중앙 이항 분포보다 안전성은 낮다. 하지만 같은 안전성을 갖기 위해 n을 약간 증가시키더라도 복호화시 발생하는 에러의 크기가 월등히 작아지므로 모듈러스를 줄일 수 있고, 결과적으로 암호문, 공개키의 크기가 더 작아진다. Round5[4]의 경우 n의 선택에 있어 제약이 적기 때문에 삼진 분포를 선택하여 다른 기법보다 암호문, 공개키가 작다는 것을 알 수 있다.

V. 결론

본 논문에서는 NIST 표준화 공모 2라운드에 진출한 4가지의 RLWE 계열 공개키 기법에 대하여 살펴보고, 분석하였다.

암호문과 공개키의 크기가 작고 보다 효율적인 암호기법을 설계하기 위해서는 n 선택에 대한 제약이 적고, 복호화 실패 확률을 줄일 수 있는 환(ring)에 대한 연구가 필요할 것으로 보인다.

[참고문헌]

[1] J. D'Anvers, A. Karmakar, S. S. Roy, and F. Vercauteren. SABER. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

[2] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

[3] X. Lu, Y. Liu, Z. Zhang, D. Jia, H. Xue, J. He, and B. Li. LAC. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

[4] H. Baan, S. Bhattacharya, S. Fluhrer, O. Garcia-Morchon, T. Laarhoven, R. Player, R. Rietman, M. Saarinen, L. Tolhuizen, J. L. Torre-Arce, and Z. Zhang. Round5. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

[5] D.J. Bernstein. Comparing proofs of security for lattice-based encryption. Cryptology ePrint Archive, Report 2019/691, 2019.

[6] S. Bhattacharya, O. Garcia-Morchon, R. Player, and L.Tolhuizen. Achieving secure and efficient lattice-based public-key encryption: the impact of the secret-key distribution. Cryptology ePrint Archive, Report 2019/389, 2019.

[7] C. Chen, O. Danba, J. Hostein, A. Hülsing, J. Rijneveld, J.M. Schanck, P. Schwabe, W. Whyte, and Z. Zhang. NTRU. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

[8] D.J. Bernstein, C. Chuengsatiansup, T. Lange, and C. Vredendaal. NTRU Prime. Technical report, National Institute of Standards and Technology, 2019. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions.

NIST 경량 암호 공모사업 2라운드 후보에 대한 구조 분석†

백승준*, 김영범*, 전용진**, 김종성*,**

*국민대학교 정보보안암호수학과, **국민대학교 금융정보보안학과

Analysis of Structures Used in 2-Round Candidates Selected by NIST Lightweight Cryptography Standardization Project

Seungjun Baek*, Youngbeom Kim*, Yongjin Jeon**, Jongsung Kim*,**

*Dept. of Information Security, Cryptology and Mathematics, Kookmin University

**Dept. of Financial Information Security, Kookmin University

요 약

정보통신기술의 발달로 인해 디바이스는 소형화되는 추세다. 경량 디바이스에서는 자원이 제한되므로, 기존의 보안 기술을 적용하기에는 많은 제약이 있다. 최근 미국 국립표준기술연구소에서는 경량 디바이스에 적합한 암호 알고리즘을 표준화하는 작업을 시작했다. 총 57팀이 지원하여 2019년 4월 18일에 56팀이 1라운드 후보로 선정되었고, 2019년 8월 30일에 32개 팀이 2라운드 후보로 선정되었다. 본 논문에서는 2라운드 후보들의 경량 암호 알고리즘을 내부 핵심 함수를 기준으로 분류하고 각각의 특징을 분석한다.

I. 서론

최근 정보통신기술(Information and Communication Technology)의 발달로 인해 사물인터넷(Internet of Things, IoT) 디바이스가 빠른 속도로 증가하고 있다[1]. IoT 환경에서 디바이스는 소형화되고 있으며, 이에 따라 가용한 자원은 제한적이다[2]. 경량 디바이스에서 처리되는 정보의 크기 및 가치를 고려할 때, 서버나 PC 환경에서 사용되는 암호 알고리즘들은 필요 이상의 암호학적 안전성을 제공한다. 또한, 인증과 암호화를 동시에 제공하는 CCM(Counter with CBC-MAC), GCM(Galois Counter Mode)도 경량 환경에 적합하지 않다[3]. 이로 인해 더 적은 비용으로 적정 수준의 보안을 보장할 수 있는 경량 암호 알고리즘의 필요성이 높아졌다.

† 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

미국 국립표준기술연구소(The National Institute of Standards and Technology, NIST)는 암호기술 표준화를 다루는 대표적인 곳이다. NIST에서는 2018년 4월, 제한된 환경에 적합한 경량 암호 알고리즘을 공모 및 표준화하는 사업을 시작했다. 제출되는 알고리즘은 관련 데이터로 인증된 암호화(Authenticated Encryption with Associated Data, AEAD)를 기능적으로 지원해야 한다[4]. 2019년 4월 18일에 56개 팀의 알고리즘이 1라운드 후보로 선정됐고, 현재 32팀의 알고리즘이 2라운드 후보에 올랐다[5].

본 논문의 구성은 다음과 같다. 2장에서는 인증된 암호화(Authenticated Encryption, AE)와 AEAD에 대해 설명한다. 3장에서는 2라운드 후보들의 경량 암호 알고리즘을 치환 기반, 블록 암호 기반, 트위커블 블록 암호 기반(Tweakable Block Cipher, TBC), 스트림 암호 기반으로 분류하고 각각의 특징을 분석한다. 4장에서는 본 논문에 대한 결론으로 마무리한다.

II. AE 및 AEAD

본 장에서는 AE와 AEAD에 대해 서술한다.

2.1 AE 알고리즘

단순히 데이터를 안전하게 저장하는 목적이 아니라 통신 매체를 통해 안전하게 전달하고자 할 때는 공격자의 데이터 탈취 및 변조를 반드시 고려해야 한다. 그런 경우 기밀성뿐만 아니라 무결성(인증)이 함께 요구된다.

AE는 메시지의 기밀성과 인증을 동시에 제공하는 암호 알고리즘이다. 이 알고리즘은 인증값을 메시지 또는 암호문에 태그 형식으로 삽입하며, 그 결과로서 인증을 제공하는 출력물을 생성한다.

AE는 기밀성 제공 알고리즘과 무결성 제공 알고리즘을 조합하여 일반적 구성(Generic Composition)을 형성한다. 기밀성 제공 알고리즘으로는 대칭 암호, 무결성 제공 알고리즘으로는 메시지 인증 코드(Message Authentication Code, MAC)를 주로 사용한다. 두 알고리즘의 결합 구조에 따라서 Encrypt-then-MAC, Encrypt-and-MAC, MAC-then-Encrypt로 분류할 수 있다[6].

2.2 AEAD 알고리즘

이후 AE 알고리즘은 관련 데이터(Associated Data)를 암호문이나 중간 데이터에 결합하는 방식을 통해 더 안전한 암호 알고리즘인 AEAD로 발전했다. 관련 데이터는 헤더(Header)라고도 불리며 인증이 필요하지만, 기밀로 유지할 필요는 없는 정보이다. 예를 들어 AEAD를 사용하여 네트워크 프로토콜을 보호할 때, 관련 데이터는 주솟값, 포트, 순서 번호(Sequence Number), 프로토콜 버전 번호 등이 될 수 있다.

AEAD 알고리즘은 암호화 과정과 복호화 과정으로 나뉜다. 암호화 과정의 입력은 키, 넌스(Nonce), 평문, 관련 데이터이며 모종의 알고리즘을 거친 후 암호문과 인증을 위한 태그를 반환한다. 이때, 평문에는 기밀성과 인증이 모두 제공되고 넌스와 관련 데이터에는 인증만이 제공된다. 복호화 과정의 입력은 키, 넌스, 암호문, 관련 데이터, 태그이며 복호화 과정이 정상적으로 완료된 경우는 평문, 문제가 있는 경우는 오류를 반환한다[7].

III. 제출된 AEAD들의 내부 함수에 따른 분류 및 특징 분석

본 장에서는 2라운드 후보, 총 32개 AEAD 알고리즘을 내부의 핵심 함수를 기준으로 분류하고 각각의 특징을 분석한다.

3.1 치환 기반 AEAD

치환 기반 AEAD로서 NIST 공모사업에 참여한 팀은 총 26팀이고 2라운드에 진출한 팀은 16팀이다[표 1]. 치환 기반 AEAD는 알고리즘을 구성하는 핵심 함수가 치환 함수로 구성되어 있다. ACE, ASCON, DryGASCON, KNOT, SPARKLE, Subterranean 2.0, Oribatida, WAGE 팀은 자체적으로 설계 혹은 이미 공개된 알고리즘을 활용하여 치환 함수로 사용했다.

[표 1] 치환 기반 AEAD 스펙

AEAD	핵심 함수	함수 작동 모드
ACE	ACE-320	Duplex Sponge
ASCON	ASCON-320	Monkey Duplex
DryGASCON	GASCON-128/256	Duplex Sponge
Gimli	Gimli-384	Duplex Sponge
KNOT	KNOT-256/384/512	Monkey Duplex
ORANGE	PHOTON-256	Sponge
PHOTON-Beetle	PHOTON-256	Sponge
SPARKLE (SCHWAEMM)	Sparkle-256/384/ 512	Sponge
Subterranean 2.0	Subterranean	Sponge
Xoodyak	XOODOO-384	Sponge
Elephant	Spongent-160/176, Keccak-200	Sponge
ISAP	Keccak-400, ASCON-320	Duplex Sponge
Oribatida	SimP-256-2/4 SimP-192-2/4	Sponge
SPIX	sLiSCP-light-256	Monkey Duplex
SpoC	sLiSCP-light-256	Monkey Duplex
WAGE	WAGE-259	Monkey Duplex

제출된 치환 기반 AEAD의 함수 작동 모드는 모두 스펀지(Sponge) 구조에 기반을 두고 있다. 스펀지 구조는 두 가지 작동 과정으로 이루어진다. 먼저 흡수 단계(Absorbing Phase)에서는 메시지를 초기 상태(State), 치환을 거쳐 생성되는 내부 상태와 XOR 연산시킨다. 모든 메시지를 흡수한 후, 압착 단계(Squeezing Phase)에서는 내부 상태의 일부 비트들을 추출하여 출력값을 생성한다.

스펀지 구조의 안전성은 내부 상태의 길이에 의존한다. 또한, 스펀지 구조에 쓰이는 치환 함수가 랜덤 오라클(Random Oracle)과 구분할 수 없어야 한다[8].

스펀지 구조 기반으로 AEAD를 설계하면, bitrate와 capacity를 적절히 설정하여 트레이드 오프(Trade-off) 관계에 있는 효율성과 안전성을 유연하게 조절할 수 있다[8]. 또한, 별도의 키 스케줄이 필요하지 않으므로 메모리에 라운드 키를 저장하지 않아도 되며, 블록 암호와 달리 데이터 부분과 키 스케줄 부분을 하나의 상태로 취급하여 확산을 전체적으로 줄 수 있다[9].

3.2 블록 암호 기반 AEAD

블록 기반 AEAD로서 NIST 공모사업에 참여한 팀은 총 15팀이고 2라운드에 진출한 팀은 9팀이다[표 2]. 블록 암호 기반 AEAD는 알고리즘을 구성하는 핵심 함수가 블록 암호로 구성되어 있다. Saturnin, Pyjamask, TinyJAMBU 팀은 자체 설계한 블록 암호를 사용했다.

[표 2] 블록 암호 기반 AEAD 스펙

AEAD	핵심 함수
Saturnin	SATURNIN-256/256
COMET	AES-128, Speck-64/128, CHAM-128/128, CHAM-64/128
GIFT-COFB	GIFT-128
HyENA	GIFT-128
mixFeed	AES-128
Pyjamask	Pyjamask-96, Pyjamask-128
SAEAES	AES-128, AES-192, AES-256
SUNDAE-GIFT	GIFT-128
TinyJAMBU	JAMBU-128

Saturnin 팀의 경우 특이하게도 3차원 구조의 블록 암호를 사용하는데, 이 블록 암호는 효율적인 비트 슬라이스(Bit Slice) 기법을 사용하도록 개발되었다. 따라서 이전에 발표된 3차원 구조의 AES-like 알고리즘들보다 경량 환경에 더 적합하다. 아울러 양자 컴퓨터가 있더라도 높은 안전성을 보장할 수 있다고 주장한다. 이것은 키 길이를 늘이고 추가적인 매개변수를 사용하는 방식으로 실행된다.

블록 암호는 안전성이 증명된 것이 많으며, PRESENT나 AES의 경우는 표준으로 제정되어 있다. 따라서 블록 암호 기반으로 AEAD 설계하면 블록 암호의 안전성을 토대로 설계할 수 있고, 원하는 조건에 따라 핵심 함수를 다양하게 선택할 수 있다. 예를 들어 GIFT-COFB 팀의 경우, 128-bit의 블록 크기를 가진 블록 암호 중 하드웨어 공간을 적게 차지하는 저전력 블록 암호 GIFT-128을 핵심 함수로 선택했다.

3.3 TBC 기반 AEAD

TBC 기반 AEAD로서 NIST 공모사업에 참여한 팀은 총 10팀이고 2라운드에 진출한 팀은 6팀이다[표 3]. TBC 기반 AEAD는 알고리즘을 구성하는 핵심 함수가 TBC로 구성되어 있다. Spook 팀은 자체 설계한 TBC를 사용했다.

TBC는 일반적인 블록 암호에서 트윅(Tweak)이라는 입력을 하나 추가한 알고리즘이다. 트윅을 이용하면 공격자에게 가변성을 제공하여 공격을 어렵게 할 수 있다. 일반적으로 트윅은 비밀로 할 필요가 없으나, 키를 바꿔 가변성을 제공하는 비용보다 트윅을 바꾸는 비용이 더 저렴해야 한다.

[표 3] TBC 기반 AEAD 스펙

AEAD	핵심 함수
SKINNY-AEAD	SKINNY-128-256, SKINNY-128-384
ForkAE	SKINNY-64-192, SKINNY-128-192, SKINNY-128-256, SKINNY-128-288
ESTATE	TweGIFT-128, TweAES-128, TweAES-128-6
LOTUS-AEAD and LOCUS-AEAD	TweGIFT-64
Romulus	SKINNY-128-256, SKINNY-128-384
Spook	Clyde-128

보통 AEAD 설계 시, 각 암호화 과정마다 입력을 다르게 만들기 위해 많은 노력을 기울인다. TBC를 기반으로 AEAD를 제작할 경우 그러한 문제를 방지할 수 있으므로 알고리즘 분석이 훨씬 쉬워진다는 장점이 있다[10].

3.4 스트림 암호 기반 AEAD

스트림 암호 기반 AEAD로서 NIST 공모사업에 참여한 팀은 총 5팀이고 2라운드에 진출한 팀은 단 1팀이다[표 4]. Grain-128AEAD 팀

을 제외하고는 모두 위조 공격에 취약한 것으로 드러나 1라운드에서 탈락했다. 스트림 암호 기반 AEAD는 알고리즘을 구성하는 핵심 함수가 스트림 암호로 구성되어 있다.

[표 4] 스트림 암호 기반 AEAD 스펙

AEAD	핵심 함수
Grain-128AEAD	Grain-128a

Grain-128AEAD 팀의 핵심 함수로 쓰인 스트림 암호 Grain-128a는 Grain-128의 향상된 버전으로, 2011년 Symmetric Key Encryption Workshop에서 처음 제안됐다[11]. Grain-128AEAD 팀의 알고리즘은 Grain-128a와 달리, 인증 과정이 제거된 암호화 유일 모드(Encryption Only Mode)를 지원하지 않는다. 또한, 레지스터와 태그의 크기가 32-bit에서 64-bit로 증가했고 키와 넌스의 길이를 80-bit로 제한했다.

스트림 암호는 블록 암호와 달리 다양한 길이의 메시지를 허용한다. 이는 스트림 암호를 기반으로 AEAD 설계하면 별도의 운영 모드가 필요하지 않으며 상태의 크기를 절약할 수 있음을 의미한다. 또한, 스트림 암호는 블록 암호보다 에너지 절약 측면에서 유리하므로 경량 디바이스에 더 적합할 수 있다[12].

IV. 결론

본 논문에서는 NIST 경량 암호 공모사업 2라운드 후보들의 알고리즘을 내부 핵심 함수를 기준으로 분류했다. 각 기반 AEAD가 가질 수 있는 특징도 분석했다. 이를 참고하여 유사한 구조의 알고리즘들을 빠르게 비교할 수 있으며, 각각의 구조의 안전성을 수월하게 분석할 수 있다.

2019년 11월 4일부터 6일까지 미국 게이더스버그에서 세 번째 NIST 경량 암호 워크숍이 열렸다. 이 자리에서 2라운드 진출 알고리즘들에 대한 디자인 평가, 실행 성능, 암호 해독, 적용 대상 등에 대해 논의했다. 앞으로 진행될 경량 암호 공모사업을 통해 여러 IoT 환경에 가장 적합한 알고리즘이 채택되길 기대한다.

[참고문헌]

[1] 배상태, 김진경, “사물인터넷(IoT) 발전과 보안의 패러다임 변화,”KISTEP InI 14호, 2016.

[2] S.-K. Lee, “A study on lightweight block cryptographic algorithm applicable to IoT environment,”Journal of the Korea Academia-Industrial, vol. 19, no. 3, pp.1-7, Mar. 2018.

[3] T.-H. Park, H.-J. Seo, S.-W. Heo, H.-W. Kim, “The recent research trend and prospective on authenticated encryption schemes,”KIISE Transactions on Computing Practices, vol. 24, no. 10, pp.563-568, Oct. 2018.

[4] NIST, https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf

[5] NIST, https://csrc.nist.gov/projects/lightweight-cryptography/round-2-candidates

[6] M. Bellare and C. Namprempre, “Authenticated encryption: relations among notions and analysis of the generic composition paradigm,”Journal of Cryptology, vol. 21, no. 4, pp.469-491, Oct. 2008.

[7] D. McGrew, “An interface and algorithms for authenticated encryption,”RCF 5116, Jan. 2008.

[8] G. Bertoni, J. Daemen, M. Peeters, G.V. Assche, “Cryptographic sponge functions,”STMicroelectronics, 2011.

[9] G. Bertoni, J. Daemen, M. Peeters, G.V. Assche, “Permutation-based encryption, authentication and authenticated encryption,”STMicroelectronics, 2012.

[10] M. Liskov, R.L. Rivest, D. Wagner, “Tweakable block ciphers,”Advances in Cryptology, CRYPTO’02, LNCS 2442, pp.31-46, 2002.

[11] M. Ågren, M. Hell, T. Johansson, W. Meier, “Grain-128a: a new version of Grain-128 with optional authentication,”International Journal of Wireless and Mobile Computing, vol. 5, no. 1, pp.48-59, 2011.

[12] S. Banik, V. Mikhalev, F. Armknecht, T. Isobe, W. Meier, A. Bogdanov, Y. Watanabe, F. Regazzoni, “Towards low energy stream ciphers,”IACR Transactions on Symmetric Cryptology, pp.1-19, Jun. 2018.

RLWE 기반 분산형 그룹키 교환 방식의 성능 분석[1)]

한성호* 최락용** 김광조*,**

* KAIST 정보보호대학원 ** KAIST 전산학부

Performance Analysis of RLWE-based Distributed Group Key Exchange Protocols

Seongho Han* Rakyong Choi** Kwangjo Kim*, **

*Graduate School of Information Security, KAIST **School of Computing, KAIST

요 약

통신 기술이 발달하면서 한 그룹에서 동일한 키를 공유하는 프로토콜의 수요가 늘고 있다. 그러나 현재 대부분의 그룹키 교환 방식은 이산대수 기반 문제를 사용함으로써 양자 컴퓨터 공격에 취약하다. 양자내성을 갖는 문제 중 하나인 RLWE를 활용하여 다양한 그룹키 교환 방식이 제안되었다. 분산형 그룹키 교환 방식은 모든 노드가 동등한 권한으로 그룹키 형성에 기여하는 방식이다. 본 논문에서는 RLWE 기반 그룹키 교환 방식 중 분산형 방식으로 구현된 프로토콜의 성능을 정량적으로 비교한다. 이를 통해 더 효율적인 RLWE 기반 분산형 그룹키 교환 방식을 파악한다.

I. 서론

통신 기술이 발달하면서 한 그룹에서 같은 키를 공유하는 프로토콜의 수요가 증가하고 있다. Bresson 등[1], Steiner 등[2]과 Dutta, Barua[3] 등이 대표적인 그룹키 교환 방식이다. 그러나 기존 프로토콜 대부분은 이산대수 기반 문제의 어려움에 기반을 두고 있으므로 양자 컴퓨터 공격에 취약하다[4].

양자 컴퓨팅 시대에 대응하여 NIST는 양자 컴퓨터 공격에 안전한 양자내성암호 표준을 채택하기 위해 공모를 시행하고 있다. 격자 기반 키 교환 방식은 양자내성암호에서 가장 유망한 후보 중 하나이다. 초기에는 Regev[5]에 의해 Learning with Error (LWE) 문제를 기반으로 하는 격자 기반 암호가 주목되었다. 그러나 연산량이 많아 효율성이 저하된다는 비판이 제기되었으며 이를 보완하기 위해 Ring Learning with Error (이하 RLWE) 문제[6]가 Lyubashevsky 등에 의해 제안되었다. 대표적으로 구현된 RLWE 기반 키 교환 프로토콜은 NewHope[7], Bos 등[8] 등이 있다.

RLWE 기반 키 교환 방식을 확장한 여러 RLWE 기반 그룹키 교환 방식이 제시되었다. Apon 등[9] (이하 ADGK19), Ding 등[10] (이하 DXL12)은 분산형 RLWE 기반 그룹키 교환 방식을 제안하였다. 분산형 그룹키 교환 방식에서는 모든 노드가 동등한 권한으로 그룹키 형성에 기여한다. 지금까지 알려진 바에 의하면 분산형 RLWE 기반 프로토콜의 효율성을 서로 비교한 적은 없다.

본 논문에서는 RLWE 기반 그룹키 교환 방식 중 분산형 프로토콜인 ADGK19와 DXL12의 효율성을 정량적으로 비교하여 효율적인 RLWE 기반 분산형 그룹키 교환 방식을 분석한다.

1) 본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행되었습니다. (No.2017-0-00555, 양자 컴퓨터 환경에서 래티스 문제를 이용한 다자간 인증키교환 프로토콜 연구)

1.1 논문의 구성

본 논문의 구성으로 Ⅱ장에서는 RLWE 문제 및 분산형 그룹키 교환 방식에 대하여 설명하고, Ⅲ,Ⅳ장에서는 AGDK19과 DXL12의 키 교환 방식을 설명한다. Ⅴ장에서는 ADGK19와 DXL12의 효율성을 정량적으로 분석한다. 끝으로 Ⅵ장에서는 결론과 추후 연구에 대하여 기술한다.

II. 배경지식

이 장에서는 RLWE와 분산형 그룹키 교환 방식에 대해 설명한다.

2.1 RLWE 문제

RLWE 문제에는 search RLWE와 decisional RLWE가 있다. NewHope[7], Ding 키 교환[10] 등에 널리 사용되는 decisional RLWE의 정의는 다음과 같다.

RLWE instance의 차원 n, ring polynomial의 집합 $R=Z[X]/(X^n+1)$, quotient ring을 정의하는 modulus q, R의 q로 quotient한 R_q, R_q에서 error를 추출하는 분포 χ를 정의하자. s를 χ에서 뽑자. 오라클 $O_{\chi,s}$를 다음과 같이 정의하자.

1) $a \leftarrow U(R_q)$, $e \leftarrow \chi$를 추출한다.

2) $(a, as+e) \in R_q \times R_q$를 반환한다.

n, q, χ에 대한 decisional RLWE 문제는 $R_q \times R_q$로부터 균일하게 랜덤 샘플을 반환하는 오라클과 오라클 $O_{\chi,s}$를 구별하는 것이다.

2.2 분산형 그룹키 교환 방식

분산형 그룹키 교환 방식은 키 교환에 참여한 모든 노드가 동등한 권한으로 키 생성에 참여하는 방식을 의미한다. 대표적인 분산형 그룹키 교환 방식은 Steiner 등[2]과 Dutta, Barua[3] 등이 있다. 대부분의 분산형 그룹키 교환 방식은 Diffie-Hellman 문제에 기반을 두고 있으므로, 양자 컴퓨터에 대비한 그룹키 교환 방식의 필요성이 제기되어 왔다.

ADGK19와 DXL12은 RLWE 문제를 기반으로 분산형 그룹키 교환 방식을 제안하였다. ADGK19는 방송형 방식을 이용하며, DXL12은 환 구조와 일방향 통신 방식을 사용한다. 본 논문에서는 두 프로토콜의 효율성을 비교한다.

III. ADGK19의 그룹키 교환 방식

ADGK19는 Dutta, Barua의 그룹키 교환 방식[3]에서 Diffie-Hellman 문제를 RLWE 문제로 변형하여 프로토콜을 설계했다. 노드의 숫자와 관계없이 고정적으로 3 round를 거쳐 키 교환이 이루어지도록 설계하였다. 프로토콜의 진행 과정은 다음과 같다.

(Initialization) 통신할 노드의 수 N, RLWE instance의 차원 n, quotient ring을 정의하는 modulus q, 균일한 값 $a \leftarrow R_q$, R_q에서 error를 추출하는 분포 χ_{σ_1}, χ_{σ_2}, statistical security parameter ρ를 결정한다. 노드의 번호는 0부터 N-1까지이다.

(Round 1) 각 노드 P_i에서 병렬적으로 수행

1. 노드 P_i는 '작은' 비밀 값인 $s_i \leftarrow \chi_{\sigma_1}$를 선택하고 '작은' error $e_i \leftarrow \chi_{\sigma_1}$를 추출한다. 그리고 $z_i = as_i + e_i$를 계산한다.

2. 다른 모든 노드들에게 z_i를 방송한다.

(Round 2) 각 노드 P_i에서 병렬적으로 수행

1. 노드 P_0는 $e'_0 \leftarrow \chi_{\sigma_2}$를 추출한다. 나머지 노드 P_i는 $e'_i \leftarrow \chi_{\sigma_1}$를 추출한다.

2. 각 노드 P_i는 $X_i = (z_{i+1} - z_{i-1})s_i + e'_i$를 계산 후 X_i를 방송한다.

(Round 3)

1. 노드 P_{N-1}는 $e''_{N-1} \leftarrow \chi_{\sigma_1}$를 추출한다.

그리고 $b_{N-1} = z_{N-2}Ns_{N-1} + (N-1) \cdot X_{N-1} + (N-2) \cdot X_0 + \cdots + X_{N-3} + e''_{N-1}$을 계산한다.

2. $(rec,\ k_{N-1}) = recMsg(b_{N-1})$를 계산한다.

3. rec를 방송한다.

4. 세션키 $sk_{N-1} = H(k_{N-1})$을 도출한다.

(Key Computation) 각 노드 P_i (P_{N-1}제외)에서 병렬적으로 수행

1. 노드 P_i는 $b_i = z_{i-1}Ns_i + (N-1) \bullet X_i + (N-2) \bullet X_{i+1} + \cdots + X_{i+N-2}$를 수행한다.

2. $k_i = recKey(b_i, rec)$를 계산한다.

3. 세션키 $sk_i = H(k_i)$를 도출한다.

(Key Computation) 과정이 끝나면 모든 노드 P_i는 1에 가까운 확률로 동일한 세션키를 공유한다.

IV. DXL12의 그룹키 교환 방식

DXL12 방식은 통신에 주로 일방향 통신을 사용한다는 특징이 있다. DXL12는 Bresson 등 [1]의 키 교환 방식에서 Diffie-Hellman 문제를 RLWE 문제로 바꾸어 설계했다. 프로토콜의 진행과정은 다음과 같다.

(Initialization) ADGK19 방식과 유사하다. 통신할 노드의 수 N, RLWE instance의 차원 n, quotient ring을 정의하는 modulus q, 균일한 값 $a \leftarrow R_q$, R_q에서 error를 추출하는 분포 χ_σ를 결정한다. 노드의 번호는 0부터 N-1까지이다.

(Round 1) 각 노드 P_i에서 병렬적으로 수행

1. 노드 P_i는 '작은' 비밀 값인 $s_i \leftarrow \chi_\sigma$를 선택하고 '작은' error $e_i^0 \leftarrow \chi_\sigma$를 추출한다. 그리고 $z_i = as_i + 2e_i^0$를 계산한다.

2. 노드 P_i는 인접한 노드 P_{i+1}에게 z_i를 전송한다.

(Round j (j=2, 3,..., N-1)) 각 노드 P_i에서 병렬적으로 수행

1. 노드 $P_{i+j-1 \pmod N}$는 j번째 Round에 $z_i^{j-1} = s_{i+j-1 \bmod N} \bullet z_i^{j-2} + 2e_i^{j-1}$를 계산한다.

2. 노드 $P_{i+j-1 \pmod N}$는 계산한 z_i^{j-1}를 $P_{i+j \pmod N}$에 전송한다.

(Round N)

1. 노드 P_0는 $e_0^{N-1} \leftarrow \chi_\sigma$을 추출한다.

그리고 $K_0 = s_0 \bullet z_1^{N-2} + 2e_0^{N-1}$을 계산한다.

2. Reconciliation Method S를 사용하여, $\sigma \leftarrow S(K_0)$를 계산한다.

3. 노드 P_0는 세션키 $SK_0 = recKey(K_0, \sigma)$를 얻는다.

4. rec인 σ를 방송한다.

(Key Computation) 각 노드 P_i (P_0 제외)에서 병렬적으로 수행한다.

1. 노드 P_i는 $e_i^{N-1} \leftarrow \chi_\sigma$를 추출한다. 그리고 $K_i = s_i \bullet z_{i+1}^{N-2\ (\bmod N)} + 2e_i^{N-1}$를 계산한다.

2. 각 노드 P_i는 $SK_i = recKey(K_i, \sigma)$를 계산하여 세션키를 도출한다.

(Key Computation) 과정이 끝나면 모든 노드 P_i는 1에 가까운 확률로 동일한 세션키를 공유한다.

V. ADGK19와 DXL12의 그룹키 교환 방식의 효율성 비교

본 논문에서는 ADGK19와 DXL12의 키 교환 방식을 통신 방식, round의 수, ring polynomial addition 연산 횟수, ring polynomial multiplication 연산 횟수, 비밀 값 샘플링 횟수, error 샘플링 횟수의 관점에서 비교하였다. Ring polynomial addition 연산 횟수는 +가 있을 때마다, ring polynomial multiplication 연산 횟수는 •가 있을 때마다 1회로 계산하며 ring polynomial 간의 연산만 고려한다.

비밀 값 샘플링과 error 샘플링은 추출할 때마다 1회로 간주한다.

이 중 ring polynomial multiplication과 각 샘플링 횟수는 프로토콜의 효율성 측정에 가장 중요한 요소이다.

표 1. ADGK19와 DXL12의 그룹키 교환 방식의 성능 비교

	ADGK19[9]	DXL12[10]
통신 방식	방송형	일방향 통신 + 방송형
Round 수	***3***	N
Ring Polynomial Addition 연산 횟수	$(N+1)^2$	N^2
Ring Polynomial Multiplication 연산 횟수	***3N***	N^2
비밀 값 샘플링 횟수	N	N
Error 샘플링 횟수	***2N+1***	N^2

[표 1]은 노드 숫자 N에 따른 ADGK19와 DXL12 키 교환 방식의 연산 횟수이다. Ring polynomial addition 연산을 제외하고 N이 클수록 ADGK19의 연산 횟수가 DXL12보다 더 적음을 알 수 있다. 결론적으로 통신할 노드의 숫자 N이 커질 때, ADGK19 키 교환 방식이 효율성 측면에서 우월함을 도출할 수 있다.

VI. 결론 및 향후 과제

본 논문에서는 두 개의 RLWE 기반 분산형 그룹키 교환 방식인 ADGK19와 DXL12의 효율성을 정량적으로 계산하였다. 그 결과 ADGK19 방식이 효율적임이 확인되었다.

향후 연구에서는 두 프로토콜을 실제 구현한 후 연산 시간을 제시하고 RLWE와 그 외의 양자내성을 가지는 분산형 그룹키 교환 방식의 효율성을 비교한다.

[참고문헌]

[1] E. Bresson, *et al.* “Provably authenticated group Diffie-Hellman key exchange.” ACM CCS 2001, 2001

[2] M. Steiner, G. Tsudik, and M. Waidner. "Key agreement in dynamic peer groups." IEEE Transactions on Parallel and Distributed Systems 11.8: pp. 769-780, 2000.

[3] R. Dutta and R. Barua. "Constant round dynamic group key agreement." ISC 2005, 2005.

[4] P. W. Shor. "Algorithms for quantum computation: Discrete logarithms and factoring." FOCS 1994, 1994.

[5] O. Regev. "On lattices, learning with errors, random linear codes, and cryptography." J. of the ACM 56.6: 34, 2009.

[6] V. Lyubashevsky, C. Peikert, and O. Regev. "On ideal lattices and learning with errors over rings." EUROCRYPT 2010, 2010.

[7] E. Alkim, *et al.* “Post-quantum key exchange - a new hope.” Cryptology ePrint Archive, Report 2015/1092, 2015

[8] J. W. Bos, *et al.* "Post-quantum key exchange for the TLS protocol from the ring learning with errors problem." IEEE S&P 2015, 2015.

[9] D. Apon, *et al.* "Constant-Round Group Key Exchange from the Ring-LWE Assumption." IACR Cryptology ePrint Archive 2019/398, 2019.

[10] J. Ding, X. Xie, and X. Lin. "A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem." IACR Cryptology ePrint Archive 2012/688, 2012.

NIST 경량암호 공모사업 1라운드 탈락 AEAD에 대한 안전성 분석†

송진교*, 이종혁*, 박보선*, 전용진**, 김종성*,**

*국민대학교 정보보안암호수학과, **국민대학교 금융정보보안학과

Security Analysis of AEADs Eliminated in 1-Round Candidates by NIST Lightweight Cryptography Standardization Project

Jingyo Song*, Jonghyeok Lee*, Bosun Park*, Yongjin Jeon*, Jongsung Kim*,**

*Dept. of Information Security, Cryptology and Mathematics, Kookmin University

**Dept. of Financial Information Security, Kookmin University

요 약

현대 사회는 임베디드 시스템 기술의 발전에 의한 IoT 기기 증가로 제한된 환경에서도 사용할 수 있는 경량암호의 필요성이 화두가 되고 있다. 현재 미국 국립표준기술연구소에서는 제한된 환경에서 사용하기에 적합한 경량암호 알고리즘을 표준화하는 프로젝트를 시작하였다. 총 57개의 제출물을 받았고 56개의 알고리즘이 1라운드 후보자로 선정되었다. 2019년 8월 30일, 1라운드에서 24개의 알고리즘이 탈락하였다. 본 논문에서는 1라운드에서 탈락한 알고리즘들의 구조의 취약점을 분석하였고, 취약점에 따른 분류기준을 제시해 탈락 원인에 대해 분석하였다.

I. 서론

최근 컴퓨터뿐만 아니라 사물(Things)에 인터넷을 연결하여 사물이 정보를 수집하고 통신하는 IoT (Internet of Things) 기술의 활용 범위가 점차 넓어지고 있다. 통신 되는 정보의 규모가 확대됨에 따라 정보의 노출에 의한 사건·사고에 대응하고 예방할 수 있는 기술의 중요성이 강조되고 있다. 정보를 보호하기 위해서는 암호 알고리즘의 사용은 필수적이다. IoT 환경에서는 기기가 작아진 만큼 전력, 프로세서 성능, 메모리 등의 자원이 제한되어 있다. 우리가 사용하는 암호 알고리즘의 대부분은 PC, 서버 환경을 위해 설계되었기 때문에, 제한된 환경을 가진 IoT 기기에 그대로 사용하기에는 어려움이 따른다. 따라서 제한된 환경에서 적합한 경량암호 알고리즘의 필요성이 대두되고 있다.

모든 IoT 기기들이 네트워크처럼 연결되기 때문에 기존 인터넷이 가진 다양한 위협요소들이 모두 존재한다. 예를 들어 공격자는 중간자 공격을 통해 데이터의 무결성을 침해할 수 있고 올바른 사용자가 보낸 것처럼 위장할 수 있다. 가장 대중적인 AEAD(Authenticated Encryption with Associated Data)인 GCM(Galois Counter Mode)은 태그 계산 시 128-bit 유한체 연산을 사용한다. 그러나 IoT 환경에서는 8-bit에서 32-bit 사이의 저성능 프로세서를 사용하기 때문에 연산에 부하가 많이 발생한다. 이를 해소하기 위해, IoT 환경에 적합한 데이터의 무결성과 인증을 제공하는 경량 AEAD가 필요하다.

현재 국립표준기술연구소(National Istitute of Standards and Technology, NIST)는 경량암호에 대해 공모 및 표준화 사업을 진행하고 있다. NIST는 관련 데이터 기능으로 AEAD를 구현하고 선택적으로 해싱 기능도 제공하는 표준화 사업을 위한 사양을 제시했다.

† 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

표준화 사업에 총 57개의 알고리즘이 제출되었고, 56개의 알고리즘이 1라운드의 후보가 되었다. NIST는 제출물의 안전성 평가와 제한된 환경에서 구현 효율성을 평가 기준으로 2019년 8월 30일 2라운드 후보를 발표했다.

본 논문의 2장에서는 AEAD 공격법 일부를 소개한다. 3장에서는 1라운드에서 탈락한 24개의 알고리즘의 구조의 취약성에 대해 분석하여 4가지로 분류했다. 결론으로 각 알고리즘의 탈락한 원인에 대해 분석한 결과를 서술하며 마무리한다.

II. AEAD에 대한 공격법

AEAD의 암호화 과정은 (N, K, M, A)쌍을 입력받고 (C, T)쌍을 출력한다. 이때 N은 논스, K는 키, M는 메시지, A는 인증데이터, C는 암호문, T는 태그이다. 복호화 과정은 (N, K, C, A, T)쌍을 입력값으로 하여, 계산한 태그와 입력받은 T가 같은지 확인한 후 M을 출력하거나 오류를 출력한다. 본 장에서는 AEAD에 대한 공격 기법들을 설명한다.

2.1 위조 공격 (Forgery Attack)

위조 공격은 서로 다른 입력 쌍에 대하여 동일한 인증값 (태그)를 생성하여 검증을 우회하는 기법이다. (N, K, M, A)의 M 또는 A에 변화를 주어 같은 T를 생성한 후 검증을 우회할 수 있다.

2.2 차분 공격 (Differential Attack)

차분 공격이란, 두 입력값 사이에 XOR 차분 관계를 주고 그에 따른 출력 차분이 나올 확률이 높을 때, 이 차분 관계를 활용하는 선택 평문 공격이다. n-bit 완전 랜덤 함수의 경우, 입력 차분이 정해졌을 때 모든 출력 차분의 확률이 $1/2^n$으로 균일한 확률 분포를 갖는다. 만약 출력 차분의 확률이 균일하지 않다면, 완전 랜덤 함수와 구분되는 특성을 갖게 되므로 이를 이용한 공격이 가능하다.

2.3 길이 확장 공격 (Length Extension Attack)

메시지 또는 인증 데이터의 길이를 확장하는 공격으로, 암호 알고리즘에서 사용하는 패딩의 취약성을 이용한다. 대표적인 예로, CBC모드의 패딩을 공격한 패딩 오라클 공격이 있다.

III. AEAD의 구조와 안전성 분석

본 장에서는 1라운드에서 탈락한 알고리즘의 취약성에 대해 분석하고 취약점에 따라 분류기준을 제시한다. Fountain, Yarará and Coral은 설계에 대해 공개적인 제 3자 보안 분석이 없었고, 제출물 내의 보안 분석은 보안 요구를 충분히 지원하지 않은 이유로 고려대상에서 제외됐다.

3.1 구조의 취약성을 이용한 위조 공격

Bleep64는 LFSR(Linear Feedback Shift Register)방식 스트림 암호에 기반한 AEAD이다. $2i$번째 암호문 블록 C_{2i}에 $\pm d$을 연산하면, $2i+2j$번째 암호문 블록 C_{2i+2j}에 $\mp d$가 된 꼴이 되어 높은 확률로 같은 태그를 출력하게 된다. 또한, K와 N을 고정했을 때 메시지, 인증데이터 쌍 (empty M, empty A), (empty M, A = 0x00), (M = 0x00, A = 0x00) 세 경우는 같은 암호문이 된다는 취약점이 존재한다.

CLAE은 스트림 암호를 기반으로 하여 LWE (Learning With Errors) problem과 유사한 스타일을 사용한 AEAD이다. 랜덤한 M, N, K에 대해 인증데이터 empty A와 A = 0x00 (1-byte)로 각각 암호화하면 SpillA 함수에 의해서 1/256 확률로 C와 T가 같아진다. 또한, 랜덤한 K에 대해, 두 개의 논스 N_1, N_2(N_1 = 0x01******01*****, N_2 = 0x00******00*****)에 대해 메시지와 인증데이터를 empty M, empty A로 사용했을 경우 VecIV 함수에 의해서 암호문과 태그가 2^{-16}의 확률로 같아지는 취약점이 존재한다.

GAGE and InGAGE는 치환을 기반으로 하여 SPN(Substitution Permutation Network)구조를 사용해 AEAD를 지원한다. |T| = 128, |S| = b = 232 or 240, b-|T| < |T|인 경우 문제가 발생한다. 주어진 (A, T, N, empty M) 쌍에 대해 |A| > b-|T|라 가정하면 공격자는 마지막

(b-|T|)-bit에 대해 A를 사용하여 잘못된 추측을 빠르게 걸러낼 수 있다. 따라서, 마지막 M은 손쉽게 생성할 수 있다.

HERN and Heron은 LFSR을 응용한 치환 기반 AEAD를 지원한다. (M = 0x00, empty A)와 (empty M, A = 0x00)을 암호화한 후 얻은 태그가 2^{-8} 확률로 충돌한다. 또한, A와 N을 처리한 후 치환되기 전 중간 상태(State) 8-bit가 노출되어 키 복구 공격에 취약하다.

SIV-Rijndael256은 인증모드 SIV(Synthetic Initialization Vector)에 블록암호 AES-256을 사용한 AEAD를 지원한다. 메시지 길이가 128-bit 미만일 경우 패딩된 관련 데이터가 Pad(A) = Pad(A')을 만족하는 |A|=256, |A'|<256에 대하여 동일한 태그 위조 공격이 가능하다.

Limdolen은 ARX(Addition Rotation XOR)구조를 기반으로 한 블록암호를 PMAC-SIV 인증모드에 사용한 AEAD이다. $A = A_0 \| A_1 \| A_0 \| A_1$ |A| = 4*128, |M| = 128라 하자. $A' = 4^{*}j, j \geq 2$ 조건을 만족할 때, 확률 1로 검증을 통과하게 할 수 있다.

Qameleon은 Tweakable 블록암호를 기반으로 하여 PanORAMa와 QARMA를 사용한 AEAD이다. N과 A를 고정하고, $m = |M|/128 < 2^{28}$, $M = M_1 \| \cdots \| M_m$, $M_1 \oplus \cdots \oplus \mathrm{pad}(M_m) = 0$인 경우 M의 체크섬 값과 empty M의 체크섬 값이 0으로 같게 된다. 이로 인해 tweak 값이 같게 되므로 1의 확률로 태그의 위조가 가능하다.

Remus, TGIF는 Tweakable 블록암호를 기반으로 Skinny(TCB)를 이용한 AEAD이다. 각 K, N으로 128-bit의 일회용 비밀값 L을 생성하여 인증에 사용한다. 그러나 L을 복구하여 위조에 사용되는 공격이 개발됐다.

Simple은 블록암호 기반으로 CTR모드로 메시지를 암호화하고 CBC-MAC을 통해 태그를 생성하는 AEAD이다. Simple은 태그를 만들 때 인증데이터와 메시지를 구분하지 않아 충돌쌍이 발생한다.

SIV-TEM-PHOTON는 인증모드 SIV에 치환암호 TEM-PHOTON을 사용한 AEAD를 지원한다. Pad(A) = Pad(A')을 만족하는 |A|=384, |A'|<384에 대하여 태그 위조 공격이 가능하다.

Sycon은 320-bit 경량 치환 기반으로 AEAD를 지원한다. 태그 생성 시 메시지와 인증데이터를 구분하지 않아 empty M과 M = 0x00일 때 쉽게 같은 태그를 생성할 수 있다.

Triad는 스트림암호를 기반으로 한 AEAD인 Triad-AE를 지원한다. Triad는 태그 계산 시 인증데이터와 메시지를 분리하지 않아 충돌쌍이 발생한다.

3.2 차분공격

SNIEK는 NLFSR(Non-Linear Feedback Shift Register) 구조의 치환을 기반으로 스트림암호와 구조적 유사성을 보인다. 1의 확률을 가진 차분반복특성이 발견 되었고, 이를 이용하여 태그를 위조하는 데 성공했다.

Quartet는 스트림암호를 기반으로 한 AEAD이다. 첫 번째 메시지 블록과 두 번째 메시지 블록의 차분이 각각 0x0000000000000040, 0x0000000006100000일 때 내부 상태의 차분이 2^{-6}의 확률로 0이 되는 것을 발견했다. 그리고 이를 이용하여 2^{-9}의 확률을 가진 차분경로가 생성됐고, 위조됐다.

Lilliput-AE는 Tweakable 블록암호를 기반으로 SC2모드와 CB3모드를 적용한 AEAD이다. Lilliput-TBC에 대한 확률 1의 연관 tweak 차분 특성을 발견하여 태그를 위조할 수 있다.

FlexAEAD는 블록암호를 기반으로한 FlexAE의 변형으로 AEAD를 지원한다. FlexAEAD-64, 128, 256 각각에 대해 확률이 2^{-66}, 2^{-79}, 2^{-108}인 차분 특성이 발견됐다. 그리고 이를 이용하여 2^{-46}, 2^{-54}, 2^{-70}의 확률로 위조 공격에 성공했다.

Limdolen은 구조상 이론적으로 16라운드의 반복 차분이 2^{-16}의 확률을 가져야 한다. 그러나 8번째 키블록 K_8의 MSB(Most Significant Bit)가 0일 때 2^{-9}의 확률을 가진 차분이 발견됐다.

3.3 길이 확장 공격

FlexAEAD는 관련 데이터를 패딩할 때 0으로만 덧붙인다. 이 점을 이용하여 길이 확장 공격에 성공했다.

Cilipadi는 MonkeyDuplex 구조를 가진 AEAD이다. Cilipadi가 관련 데이터와 평문의 길이를 One-and-Zeroes 패딩으로 블록 길이의 배수로 만드는 점을 이용하여 임의의 7-byte 데이터와 그 뒤에 0x80을 추가한 8-byte 데이터의 태그가 일치한다.

3.4 그 외

LAME은 블록암호 기반으로 SIMON 경량암호를 이용한 AEAD이다. 암호문 생성 시 빈 메시지에 대해서 빈 태그를 생성하는 취약점이 존재한다.

CLX는 치환을 기반으로한 Duplex모드를 사용한 AEAD를 지원한다. CLX의 안전성은 Sliding Property를 갖고있는 Permutation에 의존한다. 그러나 이 Property를 이용한 공격이 발표됐다.

TRIFLE은 블록암호를 기반으로 SPN구조와, CA-rule based S-box를 사용하고, MAC-then-Encrypt 방식으로 CBC모드를 통해 태그를 계산한다. S-box가 4개의 고정점을 가져(0->0, 5->5, A->A, F->F), 부분공간 공격이 가능하다. 긴 단일 활성 비트 차분/선형 공격 트랙이 존재하여 차분 및 선형 공격에 취약하다.

IV. 결론

본 논문에서는 AEAD의 공격법 소개와 NIST 경량암호 알고리즘 공모사업 1라운드에서 탈락한 AEAD를 조사하고 취약점을 분석했다. 주된 공격은 구조에 의한 위조 공격, 차분 공격, 그리고 길이 확장 공격이었다. 이를 토대로 분석된 AEAD를 [표 1]과 같이 결과를 분류했다.

[표 1] AEAD 공격법에 따른 분류

공격법	AEAD
위조 공격(구조의 취약성)	Bleep64, CLAE, GAGE and InGAGE, HERN and HERON, Limdolen, Qameleon, Remus, Simple, SIV-Rijndael256, SIV-TEM-PHOTON, Sycon, TGIF, Triad
위조 공격(차분 공격에 대한 안전성 부족)	SNIEK, Quartet, Lilliput-AE, FlexAEAD, Limdolen
길이 확장 공격	FlexAEAD, Cilipadi
그 외	LAME, SNIKE, CLX, TRIFLE

[참고문헌]

[1] Léo Perrin, “Probability 1 Iterated Differential in the SNEIK Permutation”, IACR Cryptology ePrint Archive, 374, April.2019

[2] Mustafa Khairallah, “Forgery Attack on SNEIKEN”, IACR Cryptology ePrint Archive, 408, April.2019

[3] Orr Dunkelman, Nathan Keller, Eran Lambooij and Yu Sasaki, “A Practical Forgery Attack on Lilliput-AE”, IACR Cryptology ePrint Archive, 867, July.2019

[4] Maria Eichlseder, Daniel Kales and Markus Schofnegger, “Forgery Attacks on FlexAE and FlexAEAD”, IACR Cryptology ePrint Archive, 679, June.2019

비선형 불변 공격에 저항성을 갖는 최적의 비트 순열에 관한 연구*

정건상*, 김성겸*, 홍득조**, 성재철***, 홍석희*

*고려대학교 정보보호학과, **전북대학교 IT정보공학과, ***서울시립대학교 수학과

A Study on the Optimal Bit Permutation Resistant to Nonlinear Invariant Attack

Keonsang Jeong*, Seonggyeom Kim*, Deukjo Hong**, Jaechul Sung***, Seokhie Hong*

*Graduate School of Information Security, Korea University

**Department of Information Technology, Chonbuk National University

***Department of Mathematics, University of Seoul

요 약

비선형 불변 공격은 비교적 간단한 구조의 키 스케줄을 갖는 경량 블록암호에서 필수적으로 고려되어야 할 공격이다. 간단한 구조의 키 스케줄을 갖는 경량 블록암호가 비선형 불변 공격에 대응하는 방법으로 가장 잘 알려진 것은 여러 개의 라운드 상수를 사용하는 것이다. 그러나, 이 방법은 라운드 상수 선택에 제약이 많고 경량성을 저하시킨다. 본 논문에서는 다음과 같은 연구 결과를 제시함으로써 이 문제를 개선한다. 설계자 관점에서 비트 순열을 선형계층으로 사용하는 SPN 구조 경량 블록암호는 한 쌍의 라운드 상수만을 사용하여도 비선형 불변 공격에 안전할 수 있음을 증명하고, 그러한 비트 순열의 개수와 형태를 제시한다. 또한, PRESENT 구조 블록암호는 비선형 불변 공격에 저항성을 갖기 위해 적어도 두 쌍의 라운드 상수가 필요함을 전수조사를 통해 보이며, 두 쌍의 라운드 상수가 적용되어도 차분 공격에 대한 저항성이 오히려 증가할 수 있음을 보인다.

I. 서론

경량 디바이스의 암호화 서비스 수요가 증가하면서 새로운 경량 블록암호가 지속적으로 제안되고 있다. 경량 블록암호는 경량성을 설계 목표로 하여 주로 단순한 키 스케줄 또는 동일한 키를 사용하는 경향이 있다. 하지만 단순한 구조의 키 스케줄에 대한 취약성으로 인해 2016년에 새로운 구조적 공격인 비선형 불변 공격이 제안되었다[1]. 비선형 불변 공격은 차분 공격, 선형 공격과 달리 블록암호의 라운드 수와 관계없이 공격을 수행할 수 있다. 따라서 비교적 간단한 구조의 키 스케줄을 갖는 경량 블록암호에서는 필수적으로 고려되어야 할 공격이다.

단순한 구조의 키 스케줄을 사용하는 경량 블록암호의 키 스케줄 구조는 주로 마스터키와 라운드 상수만을 이용하여 라운드 키를 생성한다. 해당 암호의 선형계층에 대해 적절한 라운드 상수를 선택하면 비선형 불변 공격에 저항성을 가질 수 있음이 2017년에 제안되었다[2]. [2]에서는 라운드 키 간의 차분을 알 수 있는 집합에서 선형계층에 대해 불변인 최소의 선형공간의 크기가 클수록 비선형 불변 공격에 저항성을 보일 수 있음을 증명하였다. 예를 들어 라운드 키를 동일한 마스터키에 라운드 상수를 라운드 카운터로 사용하는 경우($k_i = k + i$), 라운드 키 차분을 알 수 있는 집합의 크기는 사용한 라운드 상수의 개수보다 하나 작다. 이처럼 라운드 키 간의 차분을 알 수 있는 집합의 크기가 크려면 많은 라운드 상수를 사용해야 하고, 즉 비선형 불변 공격에 저항성을 보이려면 많은 라운드의 수가 필요하다. 한 쌍의 라운드 상수만을 사용하여 하나의 라운드 상수 차분으로 비선형 불변 공격에 저항성을 보일 수 있으면 적은 라운드의 수로도 공격에 저항성을 보일 수 있다. 또한, 설계자 관점에서 라운드 상수 선택에 대한 제약을 줄일 수 있다.

본 논문에서는 설계자 관점에서 한 쌍의 라운드 상수만을 사용하여도 비선형 불변 공격에 저항성을 갖는 비트 순열의 개수와 형태를 제안한다. 비트 순열을 선형계층으로 가지는 경우, 해당 비트 순열에 대응되는 행렬의 최소다항식과 특성다항식이 같으면 라운드 상수 차분을 하나만 알 수 있어도 비선형 불변 공격에 저항성을 갖는다. 이러한 특성을 만족하는 비트 순열

* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

을 **최적의 비트 순열**이라 표기한다. 또한, PRESENT 구조 블록암호는 공격에 저항성을 갖기 위해 적어도 두 쌍의 라운드 상수가 필요함을 전수조사를 통해 보이며, 두 쌍의 라운드 상수가 적용되어도 차분 공격에 대한 저항성이 오히려 증가할 수 있음을 보인다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 표기법과 비선형 불변 공격, 해당 공격에 대한 저항성 입증방법을 소개한다. Ⅲ장에서는 최적의 비트 순열 도출 방법에 관해 기술하고, PRESENT에 적용하여 도출한 결과를 Ⅳ장에서 제시한다. 끝으로 Ⅴ장에서 결론을 맺는다.

II. 관련 연구

2.1 표기법

본 논문에서 사용하는 표기법은 다음과 같다.

n : 블록암호의 블록 크기

L : SPN 구조의 선형계층, L-Layer

$\mathcal{S}$: SPN 구조의 비선형계층, $\mathcal{S}$-Layer

π : 비트 순열

P_π : 비트 순열 π에 대응되는 행렬($\pi \Leftrightarrow P_\pi$)

$(i_1 i_2 \cdots i_k)$: 위수가 k인 사이클 γ

$$\begin{cases}\gamma(i_1)=i_2, \gamma(i_2)=i_3, \cdots, \gamma(i_k)=i_1 \\ \gamma(i)=i, \forall i \notin \{i_1, i_2, \cdots, i_k\}\end{cases}$$

모든 비트 순열(π)은 대응되는 행렬(P_π)이 존재하고, 사이클들의 조합으로 나타낼 수 있다.

예시. $\pi = \begin{pmatrix}1&2&3&4&5\\1&4&2&5&3\end{pmatrix} \Leftrightarrow P_\pi = \begin{bmatrix}1&0&0&0&0\\0&0&0&1&0\\0&1&0&0&0\\0&0&0&0&1\\0&0&1&0&0\end{bmatrix} \Leftrightarrow (1)(2453)$

$C(Q_i)$: 다항식 Q_i에 대한 동반행렬

$A \sim B$: Similar 행렬 A, B ($\exists P s.t\ P^{-1}AP = B$)

$\chi_A(x)$: 행렬 A의 특성다항식

$m_A(x)$: 행렬 A의 최소다항식

$W_L(c)$: 선형계층 L에 대해 불변인 선형공간 중 c를 포함하는 최소의 선형공간

D : 라운드 키 간의 차분을 알 수 있는 집합 (i, j번째 라운드 상수 RC_i, RC_j에 대해 라운드 키가 $k_i = k + RC_i, k_j = k + RC_j$면, $k_i + k_j = RC_i + RC_j \in D$)

$$W_L(D) := \sum_{c \in D} W_L(c)$$

$$LS(g) := \{\alpha \in F_2^n | f(x) + f(x+\alpha) = Const, \forall x \in F_2^n\}$$

2.2 비선형 불변

비선형 불변 공격은 블록암호 $E_k : F_2^n \to F_2^n$의 비선형 불변을 이용한 공격으로 비선형 불변은 다음과 같이 정의한다.

블록암호 $E_k : F_2^n \to F_2^n$에 대해, 불함수 $g : F_2^n \to F_2$가 $g(x) + g(E_k(x)) = Const, \forall x \in F_2^n$을 다수의 키 k에 대해 만족시킬 때, g를 E_k의 비선형 불변이라 한다.

블록암호의 비선형 불변이 존재하는 경우에는 구별 공격, 평문 복원 및 키 복원을 시행할 수 있음이 2016년에 비선형 불변 공격을 통해 알려졌다[1].

2.3 비선형 불변 공격의 저항성 입증방법

블록암호가 비선형 불변 공격에 저항성을 갖는다는 것은 비선형 불변 g가 상수함수인 $\mathbf{0}, \mathbf{1}$로만 존재할 경우이다. 선형계층 L에 대해 불변이고, 라운드 키 간의 차분을 알 수 있는 집합 D를 포함한 최소 선형공간인 $W_L(D)$의 차원을 통해 비선형 불변 공격에 대한 저항성을 입증할 수 있다.

정리1 [2]. $W_L(D) \subseteq LS(g)$

정리2 [2]. $\dim LS(g) \geq d \Leftrightarrow \deg(g) \leq \begin{cases}n-d & \text{if } d \neq n \\ 1 & \text{if } d = n\end{cases}$

정리1, 2를 통해 $\dim W_L(D) = n$일 경우, 선형계층 L에 대한 비선형 불변 g는 선형함수 또는 상수함수이다. 하지만 비선형계층 $\mathcal{S}$에서 비선형 불변 g가 선형함수일 경우에는 선형 공격에 취약하다. 따라서 $W_L(D)$ 차원이 블록 크기와 같을 때, 해당 블록암호는 비선형 불변 공격에 저항성을 갖는다.

2.4 비선형 불변 공격의 저항성과 라운드 상수 차분 간의 관계

비선형 불변 공격의 저항성을 $W_L(D)$의 차원을 통해 보일 수 있고, 선형계층 L이 주어진 경우에는 $W_L(D)$의 차원이 블록 크기와 같게 되는 최소 라운드 상수 차분의 개수가 정해진다. 이는 선형계층 L에 대한 Rational Canonical Form의 불변 성분(Q_i)의 개수와 같다[2, 3].

$$\begin{pmatrix} C(Q_r) & & \\ & \ddots & \\ & & C(Q_1) \end{pmatrix}$$

그림 1 선형계층 L에 대한 Rational Canonical Form

정리3 [2]. $Q_1, \cdots, Q_r$이 선형계층 L의 불변 성분이고 $t \leq r$이면, $\max \dim W_L(c_1, \cdots, c_t) = \sum_{i=1}^{t} \deg Q_i$

III. 최적의 비트 순열의 개수와 형태 도출 방법

본 장에서는 선형계층 중 최적의 비트 순열의 개수와 형태를 제안한다. 한 쌍의 라운드 상수만을 사용하여도 비선형 불변 공격에 저항성을 갖는 블록암호 라운드 함수의 선형계층 L은 정리3에 의해 최소

다항식과 특성다항식이 같다.

정리4 [3]. 행렬 A, B에 대해 $A \sim B$이면 $\chi_A(x) = \chi_B(x), m_A(x) = m_B(x)$

정리5 [3]. 모든 P_{π_i}에 대해, π_i를 구성하는 사이클의 개수와 각 사이클 내 위수가 같으면서 순서가 정렬된 P_{π_j}로 변환하는 행렬 P가 존재하고 $P_{\pi_i} \sim P_{\pi_j}$이다.

예시. 주어진 비트 순열(π_1)에 대해 사이클의 순서가 정렬된 비트 순열(π_2)로 변환하는 행렬 P는 아래와 같다.

$$\pi_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 5 & 1 & 3 & 2 \end{pmatrix} \Leftrightarrow P_{\pi_1} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \Leftrightarrow (143)(25)$$

$$\pi_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 1 & 5 & 4 \end{pmatrix} \Leftrightarrow P_{\pi_2} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \Leftrightarrow (123)(45)$$

$$\exists P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} s.t\ P_{\pi_2} = P^{-1} P_{\pi_1} P$$

정리5에 의해 모든 비트 순열은 사이클 내 원소의 순서가 정렬된 비트 순열로 변환되고, 이들의 최소다항식과 특성다항식은 같다. 모든 비트 순열에 대해 두 식이 같은 비트 순열을 살펴보는 것과 정렬된 비트 순열에 대해 두 식이 같은 비트 순열을 살펴보는 것은 동치이다. 따라서 정렬된 비트 순열에 대해 최소다항식과 특성다항식이 같아지는 비트 순열의 개수와 형태를 보인다.

3.1 비트 순열이 하나의 사이클로 구성된 경우

정리5에 의해 $(12 \cdots n)$은 하나의 사이클로 구성된 모든 비트 순열을 대표하는 사이클이고, 이를 행렬로 표현하면 동반행렬의 특별한 형태이다.

$$\pi = \begin{pmatrix} 1 & 2 & \cdots & n \\ 2 & 3 & \cdots & 1 \end{pmatrix} \Leftrightarrow P_\pi = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ 1 & & & & 0 \end{pmatrix} \Leftrightarrow (12 \cdots n)$$

동반행렬은 동반행렬을 만드는 다항식과 최소다항식, 특성다항식이 같다. 따라서 비트 순열이 하나의 사이클로 구성된 경우, P_π의 최소다항식과 특성다항식이 같다($\chi_{P_\pi}(x) = m_{P_\pi}(x) = x^n - 1$). 그리고 그 개수는 원순열의 개수와 같으므로 $(n-1)!$개이다.

3.2 비트 순열이 여러 개의 사이클로 구성된 경우

$(12 \cdots i_1)(i_1+1 \cdots i_1+i_2) \cdots (n-i_k+1 \cdots n)$은 정리5에 의해 여러 개의 사이클로 구성된 모든 비트 순열을 대표한 사이클이다. 전체 사이클에 대응되는 비트 순열을 π, 각 사이클에 대응되는 비트 순열을 $\pi_1, \cdots, \pi_k$로 표현하면 다음과 같다.

$$\pi_1 = \begin{pmatrix} 1 & 2 & \cdots & i_1 \\ 2 & 3 & \cdots & 1 \end{pmatrix}, \cdots, \pi_k = \begin{pmatrix} n-i_k+1 & \cdots & n-1 & n \\ n-i_k+2 & \cdots & n & n-i_k+1 \end{pmatrix}$$

$$\Rightarrow \pi = (\pi_1 \pi_2 \cdots \pi_k)$$

이때 P_π는 대각 성분이 동반행렬로 이루어진 블록 행렬로, (1)과 같이 나타낸다.

$$P_\pi = \begin{pmatrix} P_{\pi_1} & & & \\ & P_{\pi_2} & & \\ & & \ddots & \\ & & & P_{\pi_k} \end{pmatrix} \quad (1)$$

예시. 비트 순열이 사이클 $(165)(27)(384)$로 이루어진 경우, 정리5에 의해 사이클 내 원소의 순서가 정렬된 사이클 $(123)(45)(678)$로 변환된다. 이때 정렬된 사이클에 대응되는 행렬은 아래와 같다.

$$P_\pi = \begin{pmatrix} 0 & 1 & 0 & & & & & \\ 0 & 0 & 1 & & & & & \\ 1 & 0 & 0 & & & & & \\ & & & 0 & 1 & & & \\ & & & 1 & 0 & & & \\ & & & & & 0 & 1 & 0 \\ & & & & & 0 & 0 & 1 \\ & & & & & 1 & 0 & 0 \end{pmatrix}$$

정리6 [3]. 블록 행렬 $M = \begin{pmatrix} M_1 & & & \\ & M_2 & & \\ & & \ddots & \\ & & & M_r \end{pmatrix}$의 최소다항식은 각 블록인 M_i $(i = 1, \cdots, r)$의 최소다항식들의 최소공배수와 같다.

(1)에서 행렬 $P_{\pi_1}, \cdots, P_{\pi_k}$는 크기가 각각 $i_1 \times i_1, \cdots, i_k \times i_k$ $(i_1 + \cdots + i_k = n)$이다. 또한, 정리6에 의해 P_π의 최소다항식은 $P_{\pi_1}, \cdots, P_{\pi_k}$의 최소다항식들의 최소공배수이다. $P_{\pi_1}, \cdots, P_{\pi_k}$의 최소다항식은 각각 $x^{i_1} - 1, \cdots, x^{i_k} - 1$이고, 이들은 적어도 공통인수로 $x-1$을 가진다. 따라서 P_π의 최소다항식은 차수가 n보다 낮다. 하지만 P_π의 특성다항식의 차수는 n이므로 비트 순열이 여러 개의 사이클로 구성된 경우에는 최소다항식과 특성다항식이 같지 않다.

즉 라운드 상수 한 쌍을 사용해도 비선형 불변 공격에 저항성을 갖는 비트 순열은 $(n-1)!$개이고, 그 형태는 하나의 사이클로 구성되어야 한다.

IV. 최적의 비트 순열 적용 결과 : PRESENT에 적용

본 장에서는 Ⅲ장에서 구한 최적의 비트 순열을 PRESENT의 설계 논리에 따라 적용한 결과를 보인다.

즉 하나의 사이클로 구성된 비트 순열이 PRESENT 구조에 존재하는지를 전수조사를 통해 확인한다.

PRESENT는 16비트 순열로부터 설계 논리에 맞게 확장한 64비트 순열을 사용하고, 16개의 S-box를 각각 4개씩 나누고, 이를 각각 그룹이라 지칭한다. PRESENT의 설계 논리를 살펴보면 다음과 같다.

1. S-box의 입력 비트는 동일한 그룹의 4개의 서로 다른 S-box로부터 선택된다.

2. 한 그룹의 4개의 S-box에 대한 입력 비트는 16개의 서로 다른 S-box로부터 선택된다.

3. 특정 S-box에 대한 4개의 출력 비트는 다음 라운드에서 각각 다른 그룹의 4개의 S-box의 입력 비트로 보낸다.

4. 다른 그룹에 속하는 S-box의 출력 비트는 서로 다른 S-box의 입력 비트로 보낸다.

위와 같은 설계 논리를 따르면, S-box의 디퍼런셜 브랜치 넘버가 3인 모든 5라운드 디퍼런셜 특성들이 적어도 10개의 활성 S-box를 갖는다는 사실이 증명되었다[4]. 본 논문에서는 동일한 설계 논리를 갖는 비트 순열을 탐색한 결과 하나의 사이클로 구성된 비트 순열은 존재하지 않음을 전수조사를 통해 확인하였다. 즉 PRESENT와 동일한 설계 논리를 갖는 암호의 경우 비선형 불변 공격에 저항성을 가지려면 적어도 두 쌍의 라운드 상수를 사용해야 한다.

PRESENT의 설계 논리를 따르며, 두 쌍의 라운드 상수가 적용된 비트 순열로 변경한 PRESENT를 Opt-PRESENT라 표기한다. Opt-PRESENT에 적용한 비트 순열을 대표적으로 3가지만 제안하면 그림 2와 같다. 그림 3은 PRESENT와 Opt-PRESENT의 차분 공격에 대한 저항성 비교 결과이다. PRESENT의 경우에는 차분 공격과 비선형 불변 공격에 저항성을 갖기 위해서는 각각 최소 15라운드와 24개의 라운드 상수 차분이 필요하다. Opt-PRESENT의 경우, 차분 공격과 비선형 불변 공격에 저항성을 갖기 위해서는 각각 최소 13라운드와 2개의 라운드 상수 차분이 필요하다. 이에 대한 정리는 표 1과 같다.

Opt-PRESENT를 탐색하여 차분 공격과 비선형 불변 공격에 대한 저항성을 확인한 결과, PRESENT보다 적은 라운드와 적은 라운드 상수 쌍을 이용하여 저항성을 보일 수 있음을 확인하였다.

V. 결론

본 논문에서는 경량 블록암호의 선형계층이 비트 순열일 경우, 라운드 상수 한 쌍만을 사용해도 비선형 불변 공격에 저항성을 갖는 비트 순열의 개수와

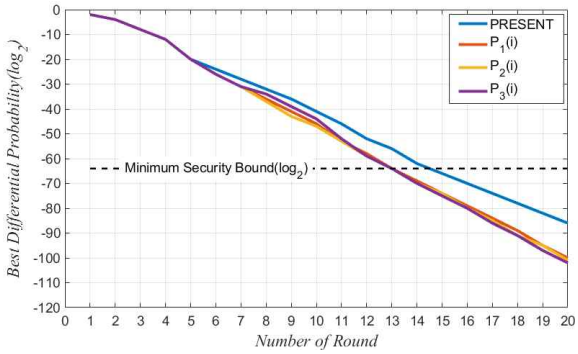


그림 3 PRESENT와 Opt-PRESENT의 최대 차분 확률 비교

공격법 / 블록암호	차분 공격 저항성	비선형 불변 공격 저항성
PRESENT	15라운드	최소 24개의 라운드 상수 차분
Opt-PRESENT	13라운드	최소 2개의 라운드 상수 차분

표 1 PRESENT와 Opt-PRESENT 저항성 비교

형태를 도출하였다. 또한, PRESENT 구조 블록암호의 경우 비선형 불변 공격에 저항성을 가지려면 적어도 라운드 상수 두 쌍을 사용해야 함을 전수조사를 통해 확인하고, 라운드 상수 두 쌍을 적용하여 탐색한 결과 PRESENT보다 적은 라운드와 적은 라운드 상수 쌍을 이용하여 차분 공격과 비선형 불변 공격에 저항성을 보일 수 있었다.

향후에 일반적인 선형계층에서 비선형 불변 공격에 저항성을 갖는 최적의 선형계층을 연구하고, 이러한 선형계층이 차분 공격, 선형 공격에도 저항성을 갖는지를 연구해볼 예정이다.

[참고문헌]

[1] Todo, Yosuke, Gregor Leander, and Yu Sasaki. "Nonlinear invariant attack." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2016.

[2] Beierle, Christof, et al. "Proving resistance against invariant attacks: How to choose the round constants." Annual International Cryptology Conference. Springer, Cham, 2017.

[3] Dummit, David Steven, and Richard M. Foote. Abstract algebra. Vol. 3. Hoboken: Wiley, 2004.

[4] Bogdanov, Andrey, et al. "PRESENT: An ultra-lightweight block cipher." International workshop on cryptographic hardware and embedded systems. Springer, Berlin, Heidelberg, 2007.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P_1(i)$	0	17	33	49	2	18	34	50	1	19	35	51	3	16	32	48	4	21	37	53	6	22	38	54	5	23	39	55	7	20	36	52
$P_2(i)$	1	18	35	49	0	19	33	50	3	17	51	32	2	16	48	34	5	22	39	53	4	23	37	54	7	21	55	36	6	20	52	38
$P_3(i)$	2	17	32	50	1	16	33	48	0	19	35	49	3	18	51	34	6	21	36	54	5	20	37	52	4	23	39	53	7	22	55	38
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P_1(i)$	8	25	41	57	10	26	42	58	9	27	43	59	11	24	40	56	12	29	45	61	14	30	46	62	13	31	47	63	15	28	44	60
$P_2(i)$	9	26	43	57	8	27	41	58	11	25	59	40	10	24	56	42	13	30	47	61	12	31	45	62	15	29	63	44	14	28	60	46
$P_3(i)$	10	25	40	58	9	24	41	56	8	27	43	57	11	26	59	42	14	29	44	62	13	28	45	60	12	31	47	61	15	30	63	46

그림 2 Opt-PRESENT에 적용한 비트 순열 (i번째 비트가 $P(i)$번째 비트로 이동)

LWE 기반 퍼지 추출기 연구 동향*

임일환[1†], 이동훈[1‡]

[1]고려대학교 정보보호대학원

A Study on LWE based Fuzzy Extractors

IlHwan Lim[1†], DongHoon Lee[1‡]

[1]Graduate School of information security, Korea University.

요 약

사용자 인증 수단으로 생체 정보의 이용이 늘고 있으며 생체 정보 기반 인증은 사용자가 어떠한 정보도 소유하거나 기억할 필요가 없으므로 매우 편리하다. 하지만 생체 정보는 추출할 때마다 측정값이 달라지므로 인증 수행 시 허용 범위 내의 오차를 처리할 수 있어야 한다. 때문에 이런 문제를 해결한 퍼지 추출기가 개발되었고, 생체 정보의 고유성과 불변성 때문에 한 번 노출되면 영구적으로 사용할 수 없게 되는 문제를 해결하기 위해 재사용이 가능한 퍼지 추출기가 개발되었다. 또한, 양자 컴퓨팅 환경에서 이산대수 문제와 인수 분해 문제가 풀리게 되면서 래티스와 코드 기반 등 다른 문제에 기반하는 많은 암호 시스템이 연구되고 있는데 퍼지 추출기 또한 양자 알고리즘에 안전하게 설계되고 있다. 본 논문에서는 양자 컴퓨팅 환경에 안전한 LWE 문제 기반의 퍼지 추출기들을 비교 분석하며 향후 연구 방향을 제시한다.

I. 서론

생체 정보는 사용자가 따로 어떤 정보를 기억할 필요가 없고 분실의 위험이 없어 접근이 쉽고 편리하다. 또한, 고유성과 불변성이 있기 때문에 사용자 인증 수단으로의 이용이 늘고 있다. 하지만 기존의 암호 시스템에서 사용되는 문자열처럼 균일분포상에 존재하지 않고 측정할 때마다 노이즈가 존재해 비슷하지만 항상 동일하게 측정하기 어렵다. 때문에 생체 정보를 기존 암호 시스템의 인증 수단으로 사용하기에는 안전을 장담할 수 없다.

퍼지 추출기는 위와 같은 문제를 해결하여 노이즈가 섞인 소스로부터 믿을 수 있는 키 생성을 가능하게 한다. Dobis 등은 Gen(생성, 처음 사용)과 Rep(재현, 이후 사용)의 두 가지의 알고리즘으로 구성된 최초의 퍼지 추출기 기법을 제안하였다[1]. Gen 알고리즘을 통해 재현 가능한 키 R과 키의 재현을 도와줄 공개 헬퍼 문자열 P를 생성하고, Rep 알고리즘을 통해 키 R을 재현한다.

지문, 홍채와 같은 생체 정보의 경우에는 위에 나온 것처럼 고유성과 불변성을 갖고 있기 때문에 타인에게 노출되는 경우 영구적으로 사용할 수 없게 된다. 하지만 최초에 나온 기법들은 이러한 문제가 해결되지 않아 동일한 사람의 생체 정보 W로부터 Gen 알고리즘을 통해 다수의 키와 헬퍼 문자열을 추출을 하게 되면 다른 암호 시스템에 사용될 지라도 각 키들과 헬퍼 문자열은 W와 관련이 있게 된다. 따라서 정보 이론적으로, 다수의 키로 인해 하나의 키 R의 엔트로피가 손실되어 남지 않게 된다. 그러므로 기존 다수의 퍼지 추출기는 같은 소스

* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

† 주저자, money683@naver.com

‡ 교신저자, donghlee@korea.ac.kr

로부터의 다중 추출을 지원하지 않았다[2-5].

위와 같은 문제를 해결하기 위해 Boyen은 최초로 재사용이 가능한 퍼지 추출기를 제안했다[4]. Boyen은 랜덤 오라클 모델을 적용해 퍼지 추출기 기법을 설계한 반면, Canetti 등은 디지털 잠금이라는 강력한 도구를 사용하여 기법을 설계하여 제안하였다[6]. 재사용 가능한 퍼지 추출기에 대한 디지털 잠금이라는 패러다임에 따라 Alamelou 등은 에러를 선형적으로 허용 가능하도록 퍼지 추출기 기법을 설계하였다[7].

그러나 기존 암호 시스템에서 사용하는 인수분해, 이산대수 문제 등은 1994년 Shor가 제안한 양자 알고리즘[8]에 의해 다항 시간 내에 풀리는 문제가 있다. 따라서 양자 컴퓨팅 환경에서도 안전한 문제나 가정이 필요하고 대표적으로 LWE 문제가 주목받고 있다. 이에 따라 Fuller 등이 최초로 LWE 문제를 기반으로 한 퍼지 추출기 기법도 제안하였다[9]. 하지만 재사용 가능성이 충족되지 않기 때문에 이 기법을 수정하여 Apon 등이 재사용이 가능한 LWE 문제 기반의 퍼지 추출기를 제안하였다[10]. 이후 Wen 등은 사용자의 생체 정보 입력에 제한이 없는 LWE 문제 기반의 퍼지 추출기를 제안하였다[11].

본 논문에서는 LWE 문제에 기반한 퍼지 추출기를 살펴볼 것이며, 논문의 구성은 다음과 같다. Ⅱ장에서는 LWE 문제에 기반한 퍼지 추출기의 이해에 필요한 배경지식을 설명하고, Ⅲ장에서는 최근 LWE 문제에 기반한 퍼지 추출기 기법들을 소개하고 비교 분석한다. 마지막으로 Ⅳ장에서 결론을 맺는다.

II. 배경지식

본 장에서는 LWE 문제를 기반으로 한 퍼지 추출기 기법의 이해에 필요한 LWE 문제와 퍼지 추출기, 그리고 재사용 가능성에 대해 정의한다.

2.1 LWE(Learning with Errors) 문제 정의

LWE 문제는 LWE 분포에서 $\mathbf{s}\in\mathbb{Z}_q^n$를 찾는 탐색 문제, LWE 분포와 균일확률분포(uniform distribution)를 구분하는 결정 문제로 나뉜다.

- LWE 분포

:정수 $q\geq 2$, 정수 $n\in\mathbb{Z}^+$, 벡터 $\mathbf{s}\in\mathbb{Z}_q^n$, $\mathbb{Z}_q$ 상에서 정의되는 확률분포 χ에 대하여, $\mathbb{Z}_q^n\times\mathbb{Z}_q$상에서 정의되는 LWE 분포 $A_{\mathbf{s},\chi}$는 다음과 같이 정의된다.

$$(\mathbf{a},\langle\mathbf{a},\mathbf{s}\rangle+x),\ \text{where}\ \mathbf{a}\leftarrow\mathbb{Z}_q^n, x\leftarrow\chi$$

$m\in\mathbb{Z}^+$개의 LWE 분포 $A_{\mathbf{s},\chi}$가 모일 경우, $(\mathbf{A},\mathbf{A}^{\mathrm{T}}\mathbf{s}+\mathbf{x})$, where $\mathbf{A}\leftarrow\mathbb{Z}_q^{n\times m}$,$\mathbf{x}\leftarrow\chi^m$와 같이 표현할 수 있다.

- 결정 문제

:결정 LWE 문제(decisional LWE problem)는 균일확률분포에서 선택한 $\mathbf{s}\leftarrow\mathbb{Z}_q^n$를 이용하여 생성한 LWE 분포 $A_{\mathbf{s},\chi}$와 균일확률분포 $U(\mathbb{Z}_q^n\times\mathbb{Z}_q)$를 구분하는 문제로 정의된다.

- 탐색 문제

:탐색 LWE 문제(search LWE problem)는 균일확률분포에서 선택한 $\mathbf{s}\leftarrow\mathbb{Z}_q^n$를 이용하여 생성한 LWE 분포 $A_{\mathbf{s},\chi}$로부터 $\mathbf{s}$를 찾아내는 문제로 정의된다.

2.2 퍼지 추출기 기법 정의

퍼지 추출기 기법은 다음과 같이 2개의 알고리즘으로 이루어져 있다.

- $\mathrm{Gen}(w)$

:생체 정보 w를 입력받고 추출키 R과 헬퍼 문자열 P를 출력한다

- $\mathrm{Rep}(P,w')$

:다시 측정된 생체 정보 w'와 저장된 헬퍼 문자열 P를 입력받아 R을 출력한다. w와 w'의 오차가 초기에 설정된 임계치 t 내에 있을 때에만 재현된다.

2.3 재사용 가능성 정의

퍼지 추출기 기법의 재사용 가능성은 다음 게임에서 공격자 A가 최대 $1/2+\epsilon$의 이점만을 가질 때를 말한다.

집합 W로부터 소스 w^*를 추출해 Gen(w^*)를 계산해 P^*와 R^*를 얻어 P^*는 공격자 A에게 전달한다. 이때 A는 임계치 t에 대해 해밍 거리 $d(0,\delta) \leqq t$를 만족하는 벡터 δ를 임의로 선택해 Gen($w^*+\delta$)로부터 출력된 헬퍼 문자열과 추출키를 모두 얻어볼 수 있다. 챌린저는 $b \in \{0,1\}$을 랜덤으로 뽑아 0이면 R^*을, 1이면 랜덤으로 뽑은 $u \in \{0,1\}^l$를 A에게 보내면 $b' \in \{0,1\}$를 출력한다.

III. LWE 기반 퍼지 추출기 연구 동향

본 장에서는 LWE 문제를 기반으로 설계된 퍼지 추출기 기법을 소개한다.

3.1 Fuller 등의 연구[9]

Fuller 등은 LWE 문제를 기반으로 한 퍼지 추출기 기법을 2개의 알고리즘으로 제안하였다.

- Gen(w)

  :생체 정보 w를 입력으로 받은 뒤 행렬 $A \in Z_q^{m \times n}$와 벡터 $x \in Z_q^n$를 균일분포에서 뽑는다. 이후 헬퍼 문자열 $P=(A, Ax+w)$, 추출키 $R=x_{1,\ldots,n/2}$로 계산하여 설정하고 (R,P)를 출력한다.

- Rep(P,w')

  :Gen 알고리즘에서 생성한 헬퍼 문자열 P와 다시 측정된 생체 정보 w'를 입력으로 받는다. 이후 헬퍼 문자열 $P=(A,c)$를 통해 $b=c-w'$를 계산해 미리 설정한 임계치 t에 대해 (A,b)를 디코딩해 x를 계산한다. 마지막으로 추출키를 $R=x_{1,\ldots,n/2}$로 하여 출력한다.

해당 기법은 최초로 LWE 문제를 기반으로 한 퍼지 추출기 기법으로, Rep 단계에서 추가로 들어온 생체 정보 w'에 대해서 w와 비교해 최대 $O(\log n)$의 에러를 용인할 수 있다. 그러나 Apon 등이 분석한 바에 의하면 $m \geqq 6n$이면 두 번의 Gen을 통해 생체 정보를 복원할 수 있는 문제가 있다. 즉 재사용 가능성이 충족이 되지 않는다[10].

3.2 Apon 등의 연구[10]

Apon 등은 암, 복호화 과정을 추가해 재사용 가능성을 충족시킨 LWE 기반의 퍼지 추출기 기법을 2개의 알고리즘으로 제안하였다.

- Gen(w)

  :생체 정보 w를 입력으로 받은 뒤 행렬 $A,B \in Z_q^{m \times n}$와 벡터 $x \in Z_q^n$를 균일분포에서 뽑는다. 이후 균일분포에서 추출키로 쓰일 랜덤 문자열 $R \leftarrow \{0,1\}^m$와 랜덤 에러 벡터 $e \in Z_q^m$을 선택해 $h=Bx+e+\frac{q}{2}R$을 계산하다. 헬퍼 문자열을 $P=(c,B,h)$로 하여 (P, R)을 출력한다.

- Rep(P,w')

  :Gen 알고리즘에서 생성한 헬퍼 문자열 P와 다시 측정된 생체 정보 w'를 입력으로 받는다. 헬퍼 문자열 $P=(c,B,h)$를 통해 $b=c-w'$를 계산하여 미리 설정한 임계오차 t에 대해 (A,b)를 디코딩해 x를 계산한다. 이후 $h-Bx$를 계산해 각 요소에 대해 $\left[\frac{3q}{8}, \frac{5q}{8}\right]$내에 있으면 1, 아니면 0로 하여 R을 출력한다.

해당 기법은 Fuller 등이 제안한 기법[9]을 분석하고 수정하여 재사용 가능성을 충족시키는 퍼지 추출기 기법을 제안했다. 따라서 생체 정보 복원 시 역시 최대 $O(\log n)$의 에러를 허용

한다. 하지만 헬퍼 문자열의 크기가 w의 길이에 따라 커지기 때문에 용량이나 계산 속도에 영향을 미친다.

3.3 Wen 등의 연구[11]

Wen 등은 에러 정정 코드와 해시 함수를 적용한 LWE 기반의 퍼지 추출기 기법을 3개의 알고리즘으로 제안하였다.

해당 기법에서 생체 정보를 복원하는 데에 동형 보안 스케치를 이용한다. 보안 스케치에서는 Dobis 등이 제안한 $[k,n,2t-1]_F$-선형 에러 정정 코드를 사용한다[1]. 스케치를 생성하는 SS.Gen에 생체 정보 w를 입력하면 해당 신드롬 $syn(w)$를 스케치 s로 출력하고, 복원하는 SS.Rec에선 s와 w'을 입력받아 $\widetilde{w}=w'-e$를 출력한다.

- Init(1^λ)

  :유니버설 해싱을 통해 기법에 쓰일 해시함수 H_i를 선택하여 pp로 출력한다.

- Gen(pp, w)

  :생체 정보 w를 입력으로 받아 SS.Gen(w)을 통해 s를 생성하고 $S=H_i(w)\in Z_q^{n\times l}$를 계산한다. 이후 행렬 $A\in Z_q^{m\times n}$와 에러 $E\in\chi^{m\times l}$를 뽑아 $B=(A, AS+E)$를 계산한다. 다음 벡터 $x\in\{0,1\}^m$, $R\in\{0,1\}^l$을 뽑아 $c^\top=x^\top B+(0^\top, R^\top \cdot \lfloor \frac{q}{2} \rceil)$를 계산해 헬퍼 문자열 $P=(s,c)$와 추출키 R을 출력한다.

- Rep(pp, P, w')

  :다시 측정한 생체 정보 w'와 헬퍼 문자열 P을 입력으로 받아 SS.Rec(s,w')을 통해 $\widetilde{w}$를 생성하고 해시값 $S=H_i(\widetilde{w})$를 계산한다. 이후 $d=c^\top \cdot \begin{pmatrix}-S\\ I_l\end{pmatrix}\in Z_q^l$을 계산하여 d의 모든 요소에 대해 $\left[\frac{1}{4}q, \frac{3}{4}q\right]$내에 있으면 1을, 아니면 0으로 하여 R을 출력한다.

해당 기법은 재사용 가능성을 유지하면서 선형 에러 정정 코드를 이용하여 생체 정보 복원 시 최대 $O(n)$의 에러까지 허용할 수 있게 되었다. 하지만 역시 헬퍼 문자열의 크기가 w의 길이에 따라 커지는 문제가 있다.

IV. 결론

본 논문에서는 퍼지 추출기 기법 중 LWE 문제를 기반으로 양자 컴퓨팅 환경에 안전하게 설계된 기법들에 대해 살펴보았다. 처음 Fuller 등이 제안한 LWE 기반 퍼지 추출기 기법을 시작으로 이를 수정해 재사용 가능한 Apon 등의 기법과 에러 정정 코드를 사용해 생체 정보의 허용 에러를 늘린 Wen 등의 기법이 있었다. 하지만 재사용 가능한 두 기법 모두 헬퍼 문자열의 크기가 생체 정보 길이에 비례하여 늘어나는 문제점이 있다. 필요한 안전성을 확보하기 위해 생체 정보 길이도 충분히 길어야 하기 때문에 용량이 커진 헬퍼 문자열을 전송하는 유저와 이를 저장하는 서버에 부담이 된다. 이산 대수 문제를 기반으로 한 퍼지 추출기에서는 Seo 등이 제안한 기법처럼 이러한 문제를 해결한 결과가 나오기 시작했다[12]. 따라서 LWE 문제를 기반으로 한 퍼지 추출기에서도 이러한 문제를 해결할 수 있다면 의미 있는 연구 결과가 될 것이다.

[참고문헌]

[1] Y.Dodis, L.Reyzin, A.Smith, Fuzzy extractors: how to generate strong keysfrom biometrics and other noisy data, EUROCRYPT 2004, LNCS 3027, pp. 523-540, 2004.

[2] M.Blanton and M.Aliasgari, On the (non-)reusability of fuzzy sketches and extractors and security in the

computational setting, SECRYPT 2011, SciTePress, pp. 68-77. 2011

[3] M.Blanton and M.Aliasgari, Analysis of reusability of secure sketches and fuzzy extractors, IEEE Trans. Inf. Forensics Secur. 8(9), pp. 1433-1445, 2013.

[4] X.Boyen, Reusable cryptographic fuzzy extractors, CCS 2004, pp. 82-91, 2004.

[5] K.Simoens, P.Tuyls, B.Preneel, Privacy weaknesses in biometric sketches, 30th IEEE Symposium on Security and Privacy, IEEE Computer Society, pp. 188-203, 2009.

[6] R.Canetti, B.Fuller, O.Paneth, L.Reyzin, A.Smith, Reusable fuzzy extractors for low-entropy distributions, EUROCRYPT 2016. LNCS, vol. 9665, pp. 117 - 146, 2016.

[7] Q.Alam´elou, P.E.Berthier, C.Cachet, S.Cauchie, B.Fuller, P.Gaborit, S.Simhadri, Pseudoentropic isometries: a new framework for fuzzy extractor reusability, ASIACCS 18, 2018.

[8] P.Shor, Algorithms for Quantum Computation: Discrete Logarithms and Factoring, Foundations of Computer Science, 1994.

[9] B.Fuller, X.Meng, L.Reyzin, Computational fuzzy extractors, ASIACRYPT 2013. LNCS, vol. 8269, pp. 174-193 2013.

[10] D.Apon, C.Cho, K.Eldefrawy, J.Katz, Efficient, reusable fuzzy extractors from LWE, CSCML 2017. LNCS, vol. 10332, pp. 1-18, 2017.

[11] Y.Wen, S.Lin, Reusable Fuzzy Extractor from LWE, ACISP 2018, LNCS 10946, pp. 13 - 27, 2018.

[12] M. Seo, J. Hwang, D. Lee, S. Kim, J. Park, Fuzzy vector signature and its application to privacy-preserving authentication, IEEE Access Volume 7, pp. 69892-69906, 2019

최신 경량 블록 암호 동향†

이종혁*, 김원일*, 한정민*, 전용진**, 김종성*,**

국민대학교 정보보안암호수학과*, 국민대학교 금융정보보안학과**

Trends of Lightweight Block Ciphers

Jonghyeok Lee*, Wonil Kim*, Jeongmin Han*, Yongjin Jeon**, Jongsung Kim*,**

*Dept. of Information Security, Cryptology and Mathematics, Kookmin University.

**Dept. of Financial Information Security, Kookmin University

요 약

IoT 기기의 보급률이 증가하면서 IoT 환경에서 발생하는 데이터의 양이 증가하고 있다. 하지만 대다수의 기기가 데이터를 평문 형태로 전달하여 기밀성을 보장하지 않는다. 데이터의 기밀성이 보장되려면 암호 알고리즘을 사용해야 한다. 따라서 IoT 환경을 포함한 제한된 환경을 고려한 경량 블록 암호가 연구되고 있다. 본 논문에서는 경량 블록 암호의 표준 동향 조사 결과와 경량 기기에 적합하게 설계된 최신 국내·외 경량 블록 암호 알고리즘에 관하여 서술하였다.

I. 서론

현대 사회는 냉장고, 인덕션, CCTV 등의 가전제품들이 인터넷에 연결된 형태인 IoT(Internet of Things) 기기들이 보급되고 있다. IoT 기기는 사용자가 네트워크 통신으로 관리할 수 있다는 이점이 존재한다. 하지만 데이터 암호화를 수행하지 않는다면 데이터 절취나 위·변조 공격 등에 취약할 수 있다. 실제로 2019년 3월, ZScaler사의 조사 결과 IoT 기기 중 91.5%가 평문 형태로 데이터를 송수신하면서 해당 공격들이 충분히 가능하다는 것이 알려졌다[1]. 이를 예방하기 위해 개인정보 보호를 위해 IoT 기기들 또한 데이터 암호화 기능이 탑재되어야 한다.

일반 가정에서 사용되는 IoT 기기들의 하드웨어 성능은 데스크탑에 비해 낮은 편이다. 즉, 데스크탑 환경에서의 Trade-Off는 IoT 환경에 부적합할 수 있다. 따라서 IoT 기기의 성능을 고려하여 설계된 경량 블록암호가 필요하다. 국내의 경우 HIGHT, LEA(Lightweight Encryption Algorithm), CHAM 등의 경량 블록 암호가 개발되었으며, 국외의 경우 PRESENT, PRINCE, SIMON, SPECK 등이 개발되었다.

본 논문의 구성은 다음과 같다. 2장에서는 국외 경량 블록 암호 표준에 대해 서술하며, 3장에서는 최근 3년 내에 제안된 경량 블록 암호 알고리즘에 대해 다룬다. 그리고 4장에서 결론과 함께 마무리한다.

† 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

II. 국외 경량암호 표준

2.2 ISO/IEC

ISO/IEC 표준 중 ISO/IEC-29192-2는 경량 블록 암호에 관한 표준으로, 2012년에 PRESENT와 CLEFIA가 표준화 되었다[2]. 이후 미국의 SIMON/SPECK과 한국의 LEA의 표준화가 진행 중이다. 미국의 SIMON/SPECK은 짧은 평문 길이와 키 길이에 대한 반론이 제기되었으며, 설계 사상의 공개를 요구받았다[3]. 그 결과, 2017년 기준 표준화 진행이 지연되었으며, 2018년 초에는 표준화 작업이 중단되었다[4]. 반면에 LEA는 표준화가 순조롭게 진행되어 2019년 10월에 표준 제정이 완료되었다. 그리고 LEA를 기술한 표준 문서가 11월에 출판될 예정이다.

2.2. NIST 경량암호 표준화 프로젝트

NIST(National Institute of Standards and Technology)는 지난 2015년 7월에 경량암호 표준화 작업을 시작하였다[5]. 2017년 4월에 경량 AEAD(Authenticated Encryption with Associative Data)[1]와 경량 해시함수 최소 요구사항을 발표하였다.

그중 경량 AEAD는 128비트 이상의 키 길이를 지원해야 하며, 128비트 키는 112비트, 256비트 키는 224비트 이상의 안전성이 보장되어야 한다[6]. 그리고 설계 시 8비트, 16비트, 32비트 환경을 고려해야 하며, 전력 분석과 전자파 분석 대응 방안이 마련되어 있어야 한다.

2019년 4월 18일에 제1라운드가 발표되었으며, 2019년 8월 30일에 제1라운드 종료 및 제2라운드가 발표되어 진행 중이다.

III.경량 블록 암호

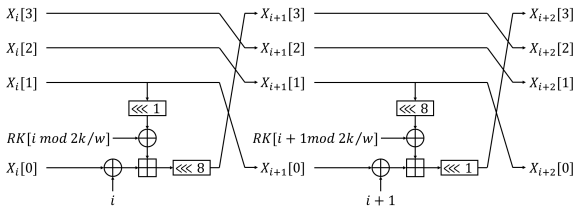
3.1. CHAM

CHAM은 2017년에 국가보안기술연구소에서 제안된 경량 블록 암호 패밀리로, 자원이 제한된 환경을 고려하여 설계되었다[7]. CHAM 패밀리에는 블록 비트 길이(n)와 키 비트 길이(k)에 따라 워드 비트 길이(w)와 라운드 수(r)가 다른데, [표 1]은 이를 정리한 것이다.

[표 1] CHAM 패밀리 구성

n/k	64/128	128/128	128/256
w	16	32	32
r	80	80	96
k/w	8	4	8

전체 구조는 ARX(Addition-Rotation-XOR) 연산 기반 4-Branch GFN(Generalized Feistel Network) 구조이다[그림 1].



[그림 1] CHAM의 짝수 라운드와 홀수 라운드

하드웨어 구현 시, 키 스케쥴은 stateless-on-the-fly 방식으로 구현할 수 있다.

3.2. GIFT

GIFT는 2017년 8월에 제안된 경량 블록 암호로, PRESENT의 선형 공격에 대한 취약성을 보완한 암호이다[8]. 전체 구조는 SPN(Substitution-Permutation-Network)이며, 64비트 또는 128비트 평문을 128비트 키로 암호화 하는데, 라운드 수는 [표 2]와 같다.

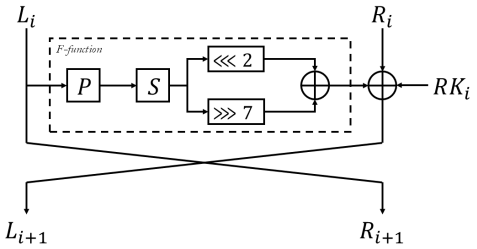
[표 2] GIFT 라운드 수

블록 비트 길이	64	128
라운드 수	28	40

1) 인증 암호화 방식에 관련 데이터를 추가하여 Cut-and-Paste 공격을 방지하는 운영모드

3.3. GRANULE

GRANULE은 2018년 6월에 제안된 초경량 블록 암호로, 임베디드 IoT 환경을 고려하여 설계되었다[9]. 전체 구조는 Feistel Network이며, 64비트 평문을 80비트 또는 128비트 키로 32라운드에 걸쳐 암호화한다[그림 2].

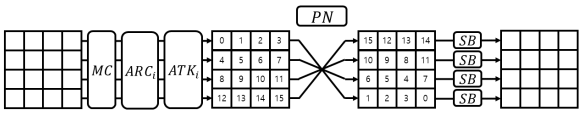


[그림 2] GRANULE의 i번째 라운드 함수

3.4. CRAFT

CRAFT(effiCient pRotection Against different Fault analysis aTacks)는 2019년 2월에 제안된 경량 Tweakable 블록 암호이다[10]. Threshold 구현이 가능하여 DFA(Different Fault Analysis) 공격에 강한 내성을 갖도록 설계 되었다.

CRAFT의 전체 구조는 SPN이며[그림 3], 64비트 평문을 128비트 키와 64비트 Tweak으로 31라운드에 걸쳐 암호화한다.



[그림 3] CRAFT의 i번째 라운드 함수

모든 내부 함수가 Involution이라는 점을 이용하여 하드웨어 구현 시 복호화 구현 비용을 감소시킬 수 있다.

3.5. TweGIFT

TweGIFT는 2019년에 제안된 경량 Tweakable 블록 암호로, 기존 경량 블록 암호인 GIFT를 재구성하였다[11]. TweGIFT는 짧은 길이의 Tweak t를 입력받아, 반복해서 연접하여 확장한 후 사용한다.

식 (1)을 확장된 Tweak의 비트 표현이라고 하자. 이때 각 T_i는 1비트이다.

$$T = T_{l-1}T_{l-2}\cdots T_1T_0 = t\|t\|\cdots\|t \quad (1)$$

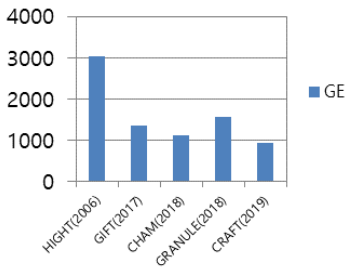
확장된 Tweak은 GIFT의 3, 7, 11, 15, 19, 23번째 라운드가 종료된 후 XOR 하는데, T_i는 i번째 Nibble의 MSB(Most Significant Bit)와 XOR된다. [표 3]은 블록 비트 길이에 따라 입력 받는 Tweak의 비트 길이와 그것의 확장된 비트 길이를 정리한 것이다.

[표 3] TweGIFT의 Tweak 비트 길이

블록 비트 길이	64	128
Tweak 비트 길이	4	4, 16
확장 후	16	32

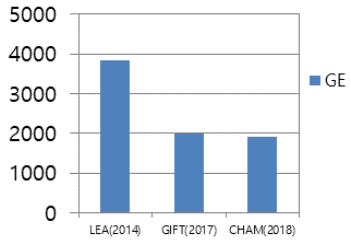
IV. 결론

[그림 4]는 국내 64비트 경량 블록 암호인 HIGHT와 GE(Gate Equivalent)를 비교한 그래프이다[12].



[그림 4] 64비트 경량 블록 암호 GE 비교

그리고 [그림 5]는 국내 128비트 경량 블록 암호인 LEA-128과 GE를 비교한 그래프이다[13]. 이때 키 길이는 모두 128비트이며, Round-based 구현이다.



[그림 5] 128비트 경량 블록 암호 GE 비교

최신 64비트 경량 블록 암호의 GE는 HIGHT의 절반 이하, 128비트의 경우 LEA-128의 절반 이하이다. 또한 그래프 추이를 보면 GE는 점점 감소하고 있으며, 블록 암호의 경량화가 활발하게 이루어지고 있다는 것을 알 수 있다. 블록 암호의 경량화가 더 활발하게 이루어지고 있다. 그러나 GE에는 반드시 하한선이 존재할 것이며, 기밀성이 필요한 기기는 일정 이상의 크기가 요구될 것이다.

[참고문헌]

[1] Tim Greene, “Study: Most enterprise IoT transactions are unencrypted”, ITworld, May.23.2019

[2] ISO/IEC 29192-2, “Information technology – Security techniques – Lightweight cryptography – Part 2: Block ciphers”, January.2012

[3] 김경애, “한국 개발 경량 블록 암호 LEA, 국제표준 선정 예정”, 보안뉴스, 2018.01.02

[4] 권준, “국가보안기술연구소의 경량 블록암호 ‘LEA’, ISO/IEC 국제표준 제정 쾌거”, 보안뉴스

[5] Meltem Sönmez Turan, Kerry McKay, Çağdaş Çalık, Donghoon Chang, Lawrence Bassham, “Status Report on the First Round of the NIST Lightweight Cryptography Standardization Process”, October. 2019

[6] NIST, “Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process”, August. 2018

[7] Koo B., Roh D., Kim H., Jung Y., Lee D G. and Kwon D. (2018) CHAM: A Family of Lightweight Block Ciphers for Resource-Constrained Devices. In: Kim H., Kim DC. (eds) Information Security and Cryptology – ICISC 2017. ICISC 2017. Lecture Notes in Computer Science, vol 10779. Springer, Cham

[8] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim and Yosuke Todo, “GIFT: A Small Present (Full version)”, IACR Cryptology ePrint Archive, 760, August. 2017

[9] Gaurav Bansod, Narayan Pisharoty, and Abhijit Patil, “GRANULE: An Ultra lightweight cipher design for embedded security”, IACR, Cryptology ePrint Archive, 600, June. 2018

[10] Christof Beierle, Gregor Leander, Amir Moradi and Shahram Rasoolzadeh, “CRAFT: Lightweight Tweakable Block Cipher with Efficient Protection Against DFA Attacks”, IACR Cryptology ePrint Archive, 210, March. 2019

[11] Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas Lopez, Mridul Nandi and Yu Sasaki, “Elastic-Tweak: A Framework for Short Tweak Tweakable Block Cipher”, IACR Cryptology ePrint Archive, 440, April.2019

[12] Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee, Donghoon Chang, Jesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim and Seongtaek Chee, “HIGHT: A New Block Cipher Suitable for Low-Resource Device”, CHES 2006, LNCS 4249, pp. 46-59, 2006.

[13] Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu and Dong-Geon Lee, “LEA: A 128-Bit Block Cipher for Fast Encryption on Common Processors”, WISA 2013, LNCS 8267, pp. 3-27, 2014

Lattice의 Determinant를 활용한 효율적인 GNFS 다항식 선택기법[1)]

전찬호*, 김수리*, 윤기순**, 장남수***, 박영호***, 김창한****, 홍석희*

*고려대학교 정보보호학과

**NSHC

***세종사이버대학교 정보보호학과

****세명대학교 정보통신학과

Enhanced Polynomial Selection for GNFS using Determinant

Chanho Jeon*, Suhri Kim*, Kisoon Yoon**, Nam Su Chang***,

Young-Ho Park***, Chang Han Kim****, Seokhie Hong*

*Graduate School of Information Security, Korea University.

**NSHC

***Sejong Cyber University

****Semyung University

요 약

본 논문에서는 인수분해를 위한 GNFS 알고리즘에서 효율적인 다항식 선택 방법을 제안한다. [2]에서 제안된 다항식 선택 기법의 경우 LLL을 이용해 다항식을 최적화할 경우, 생성된 다항식의 크기와 최적화 이후의 다항식과의 연관성이 사라져 더 이상 사용할 수 없다. 본 논문에서는 CRT로 생성된 다항식으로 형성된 래티스의 행렬식의 값(determinant)을 이용한 효율적인 다항식 선택 방법을 제안한다. RSA-220을 이용해 실험한 결과, 생성된 다항식의 행렬식과 최적화 이후 다항식은 높은 상관관계가 있음을 확인하였다. 본 논문의 결과를 GNFS의 다항식 선택에 활용한다면, 효율적인 다항식 선택을 수행해 전반적인 GNFS 속도가 향상될 것으로 기대할 수 있다.

I. 서론

1977년에 제안된 RSA 암호 시스템은 현재 공인인증서 등에 널리 사용되고 있는 공개키 암호 시스템 중 하나이다. RSA 암호는 큰 수를 인수분해하는 어려움에 기반을 두고 있으며, 효율적인 인수분해는 RSA 암호의 안전성과도 연결되기 때문에 인수분해 알고리즘 복잡도에 관한 연구가 활발히 진행되고 있다. 현재 110 digit 이상의 큰 수를 인수분해하는데 가장 효율적인 알고리즘은 General Number Field Sieve(GNFS)로, 2009년 768bit, 2012년 704bit, 2016년 729bit 인수분해에 사용되었다. GNFS는 Dixon의 difference of squares 방법에 기반을 두고 있으며, 크게 다항식 선택, 시빙, 행렬연산, 제곱근 연산의 4단계로 구성되어있다. 먼저 다항식 선택에서는 2개의 irreducible coprime 다항식 f, g가 $f(m) \equiv g(m) \equiv 0 \pmod{n}$ $m \in \mathbb{Z}/n\mathbb{Z}$를 만족하도록 f, g 를 선택한다. 이후 시빙(Sieving), 단계에서는 미리 지정한 소수들로 나누어지는 함수값을 검색하고 이를 relation으로 하여 저장한다. 행렬연산에서는 이 relation 들 중 제곱수를 만족하는 relation 집합을 선택하고 제곱근 단계에서 인수분해를 진행한다

한편, CADO-NFS는 프랑스 Inria 연구소에서 개발한 GNFS가 구현된 인수분해 및 이산대수 문제를 해결하는 공개 소프트웨어로, 2010년에 처음 공개된 이후 현재 2.3 버전까지 공개되었다. CADO-NFS를 실제 RSA-704, RSA-220 (729-bit) 인수분해에 사용되는 등 그 효율성(실효성)을 검증받고 있다 [4,5].

GNFS의 각 단계 중 다항식 선택과정이 가장 중요하다. 좋은 다항식을 선택해야 좋은 다항식을 선택해야 주어진 구간에서 relation을 많이 모을 수 있어, 시빙 단계의 시간이 줄어든다. 시빙 시간은 전체 GNFS의 수행시간에 가장 많은 부분을 차지하고 있어, 이는 더 효율적인 인수분해를 수행할 수 있을 의미한다. 여기에서 좋은 다

1) "본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다"

항식은 다항식의 Murphy E값이 높은 다항식을 의미한다.

현재 CADO-NFS에 구현된 다항식 선택은 collision을 이용해 특정 방정식의 근을 구해 다항식을 생성한 뒤 최적화 단계를 거치고, 이 중 가장 좋은 다항식을 선택해 시빙 단계로 넘어간다. 한편, [2] 에서는 기존 collision을 이용한 방정식의 근들은, 실제 Chinese Remainder Theorem(CRT)을 이용한 방정식의 근들의 부분집합으로, 더 좋은 다항식을 생성할 수 있는 근을 놓치는 경우가 발생한다는 사실을 발견하였다. 하지만 CRT를 이용해 다항식을 생성할 경우 연산 부하가 기하급수적으로 증가하여, [2]에서는 최적화 이전 다항식의 크기가 작을수록 Murphy E의 값이 크다는 사실을 실험적으로 증명하여 이를 CADO-NFS 버전 2.1.1에 적용하였다.

그러나, 가장 최신 버전인 CADO-NFS 2.3.0은 2.1.1과 달리 LLL을 이용해 다항식을 최적화하고 있으며, 이 경우 [2]에서 제안한 방법을 사용할 수 없어, CRT를 적용할 때 새로운 다항식 선택기법이 필요하다.

본 논문에서는 CRT와 래티스의 행렬식을 활용한 CRT-D라는 효율적인 GNFS 다항식 선택 방법을 제안한다. 본 논문에서는 CRT를 이용해 생성된 다항식으로 만들어진 래티스의 행렬식과 Murphy E와 연관성이 있다는 것을 확인하였다. 이를 CADO-NFS 2.3.0에 적용하면 더 효율적인 다항식 선택이 가능할 것으로 기대할 수 있다.

본 논문의 구성은 다음과 같다. 먼저, 2장에서는 본 논문을 이해하는데 필요한 배경지식에 관해 설명한다. 3장에서는 [2]에서 제안된 다항식 선택 기법과 본 논문에서 제안한 다항식 선택 기법 및 실험 결과를 제시한다.

II. 배경지식

GNFS의 다항식 선택은 크게 non-linear 방식과 linear 방식으로 구분할 수 있다. Non-linear 방식은 고차 다항식과 일차 다항식을 선택하는 방법이고, linear 방식은 두 개의 차수가 같은 다항식을 선택하는 방법이다. 본 논문에서는 현재 큰 수를 인수분해하는데 널리 사용된 Kleinjung의 non-linear 방법에 관해 설명한다.

2.1 다항식 선택

GNFS에서는 인수분해하는 수 n에 대해 다음과 같이 $\bmod n$에서 공통 근을 가지는 서로 다른 다항식 $F(x,y)$와 $G(x,y)$를 선택한다.

$$F(x,y)=\sum_{i=0}^{6}c_i x^i y^{6-i},\ G(x,y)=m_2 x-m_1 y$$

GNFS에서 Kleinjung의 다항식 선택 방법은 다음과 같다 [6]. n을 $\sum_{i=0}^{d}c_i m_i^i m_2^{d-i}$라 하자. 이 식을 전개하면 다음과 같이 나타낼 수 있다.

$$n=c_d\left(m_1^d+\frac{c_{d-1}}{c_d}m_1^{d-1}m_2\right)+\sum_{i=0}^{d-2}c_i m_1^i m_2^{d-i}$$

위 식의 양변에 $d^d c_d^{d-1}$를 곱하고 정리하면, m_1, m_2로 나타내어지는 R_0에 대해

$$d^d c_d^{d-1}n=(c_d d m_1+c_{d-1}m_2)^d-d^d c_d^{d-1}m_2^2R_0+d^d c_d^{d-1}\sum_{i=0}^{d-2}c_i m_1^i m_2^{d-i}$$

로 나타낼 수 있다. $\tilde{n}=d^d c_d^{d-1}n$이라 하고, 양변을 $\bmod m_2^2$ 으로 정리하면

$$\tilde{n}=(c_d d m_1+c_{d-1}m_2)^d \bmod m_2^2$$

가 된다. GNFS에서 m_2는 서로 다른 ℓ개의 소수들의 곱으로 이루어져 있으며, c_d는 초기 입력값이다. 따라서 위 식을 $\bmod m_2^2$에 대해 근을 구하여 $\tilde{m}=c_d d m_1+c_{d-1}m_2$를 구할 경우, $\tilde{m} \bmod d c_d$를 계산하여 c_{d-1}를 복원할 수 있으며, 이를 이용해 m_1을 복원할 수 있다. 그 후 $n=\sum_{i=0}^{d}c_i m_i^i m_2^{d-i}$에 대입하여 $f(x)$의 나머지 계수를 복원할 수 있다.

한편, 다항식 선택은 시빙 단계의 연산시간에 큰 영향을 미치기 때문에, 선택한 다항식에 대한 적절한 점수를 부여해 어느 다항식이 좋은지 판별해야 한다 [1]. 좋은 다항식을 판별하는 특성은 크게 size property와 root property의 2가지이며, 이는 다음과 같다.

(1) Size property

(a,b)를 시빙 영역 Ω에서 서로소인 한 쌍의 정수라고 할 때, 1차식 G는 영역 내에서 함수값 크기의 범위가 넓지 않지만, 비선형 다항식인 F는 범위가 넓을 수 있다. 따라서 비선형 다항식 F의 크기는 다음식과 같이 나타낼 수 있다.

$$\frac{1}{2}\log\left(s^{-d}\int_0^{2\pi}\int_0^1 F^2(s\cos\theta,\sin\theta)r^{2d+1}dr d\theta\right) \quad (1)$$

s : 편향성

다항식 F에 대한 식 (1)의 값을 lognorm이라 하며, lognorm값이 작을수록 smoothness를 기대할 수 있어, 좋은 다항식으로 분류한다.

(2) Root property

다항식 $f(x)$가 작은 소인수에 관한 합동식의 근을 많이 가지고 있다면, 함수값은 같은 크기를 가진 임의의 숫자보다는 좀 더 smooth 할 가능성이 있다. n_{p^k}를 $f(\bmod p^k)$가 가지는 근의 수라고 정의하자. n_{p^k}의 개념을 통해 p-valuation의 기댓값, $\nu_p(f)$은 다음처럼 정의할 수 있다.

$$\nu_p(f)=\sum_{k=1}^{\infty}n_{p^k}/(p^{k-1}(p-1)) \quad (2)$$

이때, 해 특성은 (2)의 식을 이용해서 다음과 같이 정량적으로 정의할 수 있다.

$$\alpha(F)=\sum_{p\le B}(\frac{1}{p-1}-\nu_p(F))\log p \qquad (3)$$

(3)의 식을 통해 다항식 F가 bound B 밑의 모든 소인수들에 대해서 얼마나 smooth한 성질을 지니고 있는지를 확인할 수 있다.

(3) Murphy E

Murphy E값은 size property와 root property를 결합하여 다항식의 성능을 하나의 점수로 나타낸 값이다. 이 방법은 기존 함수 F의 값에, $e^{\alpha(F)}$를 곱하여 두 가지 특성이 점수에 관여할 수 있도록 만든다. Murphy E 점수는 테스트 시빙을 하지 않아도 미리 다항식의 품질을 확인할 수 있으며, 이를 통해 특별히 큰 시간을 들이지 않아도 좋은 다항식들을 선별해낼 수 있다. 선택한 다항식 F,G에 대해 Murphy E의 값은 다음과 같이 계산하며, 이는 주어진 영역에서 나올 수 있는 relation의 기댓값을 의미한다. 따라서, Murphy E 값이 클수록 주어진 영역에서 더 많은 relation을 모을 수 있어, 시빙 시간이 단축될 수 있음을 의미한다.

$$\frac{6}{\pi^2}\int_{\Omega}\rho(\frac{\log|F(x,y)|+\alpha(F)}{\log B_1})\rho(\frac{\log|G(x,y)|+\alpha(G)}{\log B_2})dxdy$$

위의 식에서 적분값을 추정하기 위해서 아래와 같이 K set에 속해있는 점 (x_i,y_i)들과 관련된 합으로 표현할 수 있다.

$$E(F,G)=\sum_{i=1}^{K}\rho(\frac{\log|F(x_i,y_i)|+\alpha(F)}{\log B_1})\rho(\frac{\log|G(x_i,y_i)|+\alpha(G)}{\log B_2})$$

다항식에 대해 Murphy E의 값이 클수록 좋은 다항식으로 분류한다.

2.2 다항식의 최적화

앞에서 언급한 바와 같이, 시빙단계의 연산시간은 다항식의 성능에 의존하기 때문에, 2.1의 방법으로 다항식을 생성한 다음, 최적화 단계를 거쳐서 더 좋은 다항식을 얻도록 한다. 다항식의 최적화 방법으로는 size optimization과 root optimization 두 종류가 있다.

(1) Size optimization

다항식을 생성한 직후에는 $|c_d|,|c_{d-1}|,|c_{d-2}|$의 크기는 $(n/c_d)^{1/d}$정도의 크기를 가지는 다른 계수에 비해 매우 작다. Size optimization에서는 translation과 rotation을 이용해 계수들의 크기를 비슷하게 조정하고, 다항식 함수값의 크기를 줄이는 단계이다. 먼저 적정 k를 이용해 $f(x+k)$, $g(x+k)$로 변형시켜 (1)식에 대입한다. $f(x),g(x)$보다 $f(x+k),g(x+k)$의 lognorm값이 작으면 $f(x)$, $g(x)$를 각각 $f(x+k)$, $g(x+k)$로 치환한다. 그 후 $f(x)$를 임의의 다항식 $h(x)$를 이용해 $f(x)+h(x)g(x)$로 변형시킨 다음 lognorm값을 확인한다. 만약 $f(x)+h(x)g(x)$의 lognorm값이 더 작다면 $f(x)$를 $f(x)+h(x)g(x)$로 치환한다.

(2) Root optimization

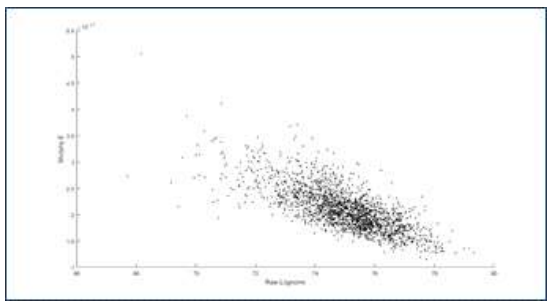
Size optimization으로 다항식의 크기를 줄인 이후, root optimization 단계를 통해 주어진 다항식의 smoothness가 크도록 변형한다. 먼저 $f(x)$를 임의의 u,v에 대해 $f_{u,v}(x)=f(x)+(ux+v)g(x)$로 변화시킨다. 그 후, u,v를 변경해가며 좋은 α값을 가지는 $f_{u,v}(x)$를 찾는다. 이 과정은 정해진 소수들의 집합에 대한 근의 개수를 확인하는 단계로, 다항식 최적화 과정 중 가장 많은 시간이 걸린다.

3. CRT-D를 이용한 다항식 선택 기법

본 장에서는 다항식으로 생성된 래티스의 행렬식을 이용한 효율적인 다항식 선택 방법에 관해 서술한다. 먼저 기존 [2]에서 제안된 CRT 및 lognorm을 이용한 다항식 선택 방법에 관해 서술한 뒤, 제안하는 방법을 소개한다.

3.1 기존 제안된 방법

$m_2=p_1p_2$라 하자. 일반적으로 $\tilde{n}=\tilde{m}^d \bmod p_1^2p_2^2$ 에서 $\tilde{m}$를 구하기 위해서는 CRT를 이용하지만, Kleinjung은 효율적인 계산을 위해서 $\tilde{n}^{1/d}$의 정수부분 m_0는 고정시켜 $\tilde{n}=(m_0+r)^d \bmod m_2^2$으로 식을 변형한다 [3]. 그 후, 소수 p_1에 대해서 $\tilde{n}=(m_0+r)^d \bmod p_1^2$를 만족하는 r을 구하고, 소수 p_2에 대해서 $\tilde{n}=(m_0+r')^d \bmod p_2^2$를 만족하는 r'을 구한다. $r=r'$일 경우 $\tilde{n}=(m_0+r)^d \bmod p_1^2p_2^2$ 로 공통근을 계산한다. 하나의 $\tilde{m}$에 대해서 하나의 다항식이 생성되고, 각각에 대해 최적화를 수행하므로 Kleinjung의 방법은 효율적으로 적은 근을 찾으며 다항식 선택도 빠르게 진행된다. 하지만 [2]에서는 이 경우 더 좋은 다항식을 생성하는 근을 놓치는 경우가 발생하는 것을 확인하였다. 한편, [2]에서는 생성한 다항식의 lognorm (raw-lognorm)과 최적화 과정 이후 Murphy E값은 다음과 같은 음의 상관관계를 확인하였다.

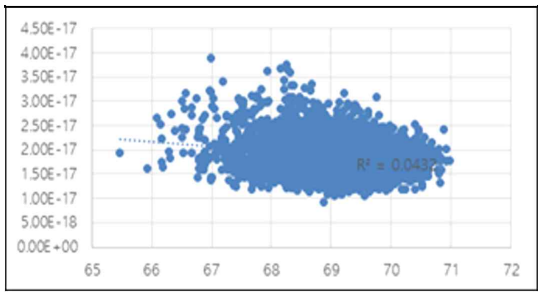


<그림 1> Raw-lognorm 과 Murphy E와의 상관관계

위의 그림에서 확인할 수 있듯이, raw-lognorm값이 작을수록 최적화 이후 Murphy E값이 크다. 따라서, [2] 에서는 CRT를 이용해 모든 다항식을 생성한 후 raw-lognorm이 작은 특정 개수의 다항식만 최적화하는 것을 제안하여, 기존 CADO-NFS보다 더 좋은 다항식을 빠르게 검색할 수 있었다.

3.2 제안하는 다항식 선택 방법

CADO-NFS 2.3.0의 경우 size optimization 과정 대신 LLL을 이용하였다. CADO-NFS 2.3.0은 생성된 다항식의 계수를 이용해 래티스를 형성하고, 해당 래티스를 LLL 알고리즘을 이용해 크기를 줄여 그 결과를 다시 다항식의 형태로 나타내는 방법을 이용하였다. 이로 인해 아래 그림과 같이 기존 [2]에 제안된 raw-lognorm과 Murphy E의 관계가 나타나지 않음을 확인하였다.



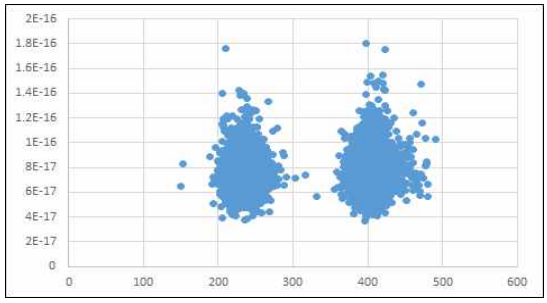
<그림 2> CADO-NFS 2.3.0에서 raw-lognorm (가로축)과 Murphy E(세로축)의 관계

따라서 본 논문에서는 생성된 다항식을 이용해 형성된 래티스의 행렬식과 Murphy E의 값의 연관성을 이용하는 방법을 제안한다. 제안하는 방법은 CRT를 이용해 모든 다항식을 생성하지만, 생성된 다항식으로 형성한 래티스의 행렬식이 작은 일부 다항식만 최적화하여 더 효율적인 다항식 선택을 제안한다.

3.3 실험 결과

실험은 CADO-NFS 2.3.0과 RSA-220을 이용해 진행하였다. 먼저 RSA-220을 이용해 CADO-NFS 2.3.0으로 다항식을 생성하고 최적화를 진행하였다.

아래의 그림에서 확인할 수 있듯이, 생성된 다항식의 행렬식의 값은 두 개의 그룹으로 나뉜다. 두 개의 그룹은 기본적으로 분포는 비슷하지만, 상대적으로 높은 Muphy E값을(1.4×10^{-16}이상) 가지는 다항식들은 많은 수가 높은 행렬식 값을 가지는 그룹에 속해있음을 알 수 있다. 따라서 행렬식의 값이 클수록 최적화 이후의 Murphy E가 높다는 것을 확인할 수 있다. 따라서, CRT로 모든 근을 생성하여도, 이 중 행렬식이 큰 일부 다항식만 최적화를 진행한다면, 더 효율적인 다항식 선택이 가능할 것이다.



<그림 3> 다항식의 행렬식(가로축)과 Murphy E(세로축)와의 관계

실험에 사용된 고차 다항식은 6차식이고, 6차항의 계수를 10^6에서 2.6×10^{14}까지 증가시키면서 행렬식의 값과 Murphy E의 값을 계산하는 실험을 진행하였다. 다항식의 생성 및 최적화를 이용하여 Murphy E값의 계산은 Intel Core i7-6700(6threads)을 이용하였고, 행렬식 값의 계산은 Intel Core i7-9700K을 이용하였다.

4. 결론

본 논문에서는 CRT와 행렬식을 활용한 효율적인 GNFS 다항식 선택 방법을 제안하였다. 행렬식의 값이 높을수록 높은 Murphy E값을 찾을 확률이 높으므로, 이를 이용하는 제안을 CADO-NFS 2.3.0에 더 적용한다면 더 효율적인 다항식 선택이 가능할 것으로 예상한다.

참고문헌

[1] S. Bai et al. “Better polynomials for GNFS,” Mathematics of Computation, 85, pp. 12, 2016

[2] S. Kim et al. “Enhanced polynomial selection method for GNFS,” 정보보호학회 논문지, 26(5), 2016

[3] T. Kleinjung et al. “Factorization of a 768-bit RSA modulus,” Annual Cryptology Conference, 2010

[4] S. Bai et al. “Factorization of RSA-704 with CADO-NFS,” 2012

[5] S. Bai et al. “Factorization of RSA-220 with CADO-NFS,” 2016

[6] T. Kleinjung, “On polynomial selection for the general number field sieve,” Mathematics of Computation, 2006

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 D

NIST Round2 양자내성암호 공모전 코드기반암호 성능 비교 분석

최승주* 장경배* 김현지* 서화정*†

*한성대학교 IT융합

Performance Analysis of Code-Based Cryptography of NIST Round2 Post Quantum Cryptography Standardization

Seung-Ju Choi* Kyoung-Bae Jang* Hyun-Ji Kim* Hwa-Jeong Seo*†

*Division of IT Fusion, Hansung University.

요 약

양자 컴퓨터의 시대가 현실로 다가옴에 따라 미국 국립표준기술연구소(National Institute of Standards and Technology, 이하 NIST)에서 양자내성암호 표준화 공모전을 개최하였다. 현재, Round1 평가를 거쳐 남은 26개의 암호들이 Round2 평가에 돌입하였다. 본 논문에서는 후보 암호 알고리즘 중 코드기반 암호들에 대한 성능 평가를 2가지 대표적인 컴퓨팅 환경에서 진행한다. 해당 Round2 코드기반암호에 대한 성능 분석은 본 논문이 최초이며 이를 통해 Round3에 진출하게 될 알고리즘을 예상해 보며 현재 코드기반암호들이 개선해야 할 부분에 대해 분석을 진행하도록 한다.

I. 서론

미국 NIST에서는 다가오는 양자 컴퓨터 시대에 대비하기 위해 2016년 양자내성암호(Post-Quantum Cryptography) 표준화 계획안을 발표하였고 2017년 말 개최한 공모전에 세계 여러 각국에서 양자내성암호 알고리즘을 후보로 제출하였다. 제출된 후보들에 대해서 Round1에서는 보안성을 중점을 두고 평가를 하였고, Round2에서는 남은 후보군들에 대한 성능 등에 대한 기준을 토대로 평가가 진행되고 있다. 해당 표준화 후보들은 전부 NIST의 보안 테스트를 통과하였기 때문에 양자에 대한 내성의 안정성이 검증되었다고 볼 수 있다. 하지만 암호에 있어 보안성뿐만 아니라 암호의 성능 또한 중요하다. 이에 본 논문에서는 제출된 후보들에 대한 성능 평가를 진행하였고 대상은 코드기반암호이다[1-7]. 현재 Round2를 진행 중인 총 26개 중 17개가 공개키 암호 후보이며, 이중에 코드 기반 암호가 7개로서 많은 수를 차지하고 있다. 성능 평가 환경은 고성능 데스크톱 프로세서 Intel과 저전력 모바일 프로세서 ARM에서 각각 진행하였다. 각각의 플랫폼에서 코드기반암호 7개의 키 생성, 암호화, 복호화에 대한 속도를 측정하였고 비교 분석해본다. 분석 결과를 통해 양자내성암호가 공통적으로 갖고 있는 문제점인 큰 키 크기, 연산 속도 등의 요소들이 고성능 및 저전력 프로세서 환경에서 어떠한 성능으로 작용되는지 알아본다. 마지막으로 다가오는 양자 시대에 암호들이 어떤 분야에 사용되는 것이 적합할 것인지 평가해보고자 한다.

II. 양자 컴퓨터와 큐비트

양자 컴퓨터는 1980년대 컴퓨터에 양자 물리학을 적용하자는 아이디어를 제안하면서 등장하였다. 이와 같은 아이디어가 나온 당시에는 사람들의 큰 관심을 끌지는 못하였지만 1990년 대소인수 분해 양자 알고리즘이 제안되면서 이를 구현하려는 많은 연구가 진행되었다.

양자 컴퓨터와 기존 컴퓨터의 가장 큰 차이점은 연산을 진행하는 단위가 달라졌다는 점이다. 기존 컴퓨터는 0과 1중 하나로만 비트 단위

의 연산을 진행하였다면 양자 컴퓨터는 큐비트라는 단위를 사용한다. 큐비트는 기존 비트와는 다르게 0과 1이라는 상태를 동시에 가질 수 있으며 이러한 상태를 중첩이라고 부른다. 이러한 상태를 가질 수 있는 큐비트로 인해 표현할 수 있는 정보의 수가 기하급수적으로 늘어났고 많은 수의 큐비트를 활용하면 기존 컴퓨터로는 처리하지 못하는 정보의 양을 표현할 수 있게 될 것으로 예상된다. 또한 큐비트는 얽힘이라는 상태를 가질 수 있는데, 이는 하나의 큐비트의 상태를 측정하였을 때 해당 큐비트와 얽혀있는 상태로 연결되어 있는 다른 큐비트의 상태가 모두 단번에 결정되는 성질이다.

이와 같은 특성들을 활용해 기존 컴퓨터로는 수년의 시간이 걸려야 하는 연산을 중첩 상태의 병렬적 연산을 통해 단 몇 분 안에 처리할 수 있게 되었다. 대표적인 양자 연산으로는 소인수분해의 주기성을 찾는 문제로써 수학자 Shor가 제안한 Shor 알고리즘 등이 있다. 해당 알고리즘은 단 한 번의 연산만으로도 소인수분해에 필요한 지수 계산의 주기를 알아낼 수 있게 하는 양자 알고리즘이다.

이와 같은 연산 능력을 가진 양자 컴퓨터가 양산화가 된다면 현재 많이 쓰이고 있는 소인수분해 문제를 기반으로 한 공개키 암호화 알고리즘들은 사용할 수 없게 된다. 이미 양자 컴퓨터에 대한 개발은 많은 기업들이 개발 중에 있으며 구글에서는 53개 큐비트로 이뤄진 양자 컴퓨터를 개발했다 발표하였고 IBM에서는 20 큐비트 프소세서를 탑재한 양자 컴퓨터를 선보였다. 이와 같이 양자 시대에 양자 우위를 선점하기 위한 다양한 발전들이 이뤄지고 있는 상황이며 이와 동시에 양자 컴퓨터 시대에 대비한 움직임 또한 활발하게 진행되고 있다.

III. NIST 양자내성암호 공모전 코드기반암호 성능분석

양자 컴퓨터가 상용화 되는 양자 시대가 오면 많은 곳에서 사용되는 대표적인 암호 알고리즘인 RSA나 AES와 같은 암호 알고리즘들이 깨지게 된다. 이러한 상황은 해당 암호가 적용되는 분야 모든 곳에 영향을 미치게 된다. 이와 같은 상황에 대비하기 위해서는 고성능 환경뿐만 아니라 IoT 디바이스와 같은 저전력 환경에서도 반드시 양자내성암호가 적용되어야 한다. 특히 IoT 플랫폼과 같은 환경에서 암호의 성능은 매우 중요하다. 한정된 자원을 갖고 암호 연산을 진행해야 하는 플랫폼이기 때문에 암호 알고리즘에 대한 최적화가 필수로 필요해진다.

현재 제출되어 있는 NIST 양자내성암호 알고리즘 중 코드기반암호 7가지에 대한 현실적인 적용 가능성을 평가하기 위해 고성능 환경과 저전력 환경 두 가지 환경 모두에서 실행해 본다. 각 암호 알고리즘의 키 생성, 암호화, 복호화 속도를 비교해보고 각 암호가 어느 분야에 적용되면 적합할 것인지 분석해보고자 한다. 성능 평가 환경은 고성능 데스크톱 프로세서인 Intel 프로세서와 저전력 모바일 프로세서인 ARM에서 진행하였고 100번의 Reference code 실행 후 평균 속도를 측정 하여 온라인에 결과물[1]을 게시하였다. 7가지 코드기반 암호 후보들의 특성과 성능을 분석해보면 아래 Table 1과 같다.

	Code	Security history	Structure	Security level	Performance
Classic McEliece	Goppa	long	KEM	IND-CCA2	low
BIKE	QC-MDPC	short	KEM	IND-CCA	high
HQC	QC	short	KEM	IND-CCA2	high
RQC	QC	short	KEM	IND-CCA2	high
ROLLO	Rank Metric	short	PKE/KEM	IND-CPA (KEM) IND-CCA2 (PKE)	high
LEDAcrypt	QC-LDPC	short	PKE/KEM	IND-CCA2	low
NTS-KEM	Goppa	long	KEM	IND-CCA	low

Table 1. Code-based Candidates Analysis

먼저 코드기반암호 중 가장 대표적인 암호 알고리즘인 McEliece와 같이 기존 Goppa 코드를 사용하는 암호와 QC, MDPC[1] 등 새로운 코드를 사용하는 암호로 크게 분류가 된다. Goppa

1. https://docs.google.com/document/d/1075XjELMU_EsOFq_3fzvjEL-eu5Ud11ORZAI5mblYmM/edit

코드는 오랜 기간 사용된 코드로서 해당 코드를 사용함으로서 얻는 보안성에 대한 안전성을 장점으로 취할 수 있게 되나 효율성 부분에서는 좋지 못한 모습을 보여준다. Goppa 코드가 아닌 QC 시리즈의 코드를 사용하는 코드들은 비교적 성능이 Goppa 코드에 비해 높다. 하지만 해당 코드는 Goppa 코드에 비해 검증된 기간이 길지 않기 때문에 지속적인 보안성에 대한 검증과 증명이 필요하다. 코드기반암호 7가지 모두 KEM 버전을 제공하며 IND-CCA의 보안레벨을 달성한다. 더 높은 보안레벨인 IND-CCA2 정도의 보안레벨을 달성하기 위해서는 암호문에 해시 값을 연산하여 추가하는 매우 비효율적인 과정이 필요하다. 실제로도 성능 평가 과정 중에서 몇몇 IND-CCA2를 달성한 암호 알고리즘을 돌리는 도중 엄청난 성능 저하가 발생하는 것을 확인할 수 있었다.

Intel 프로세서와 같은 고사양의 환경에서는 디바이스의 메모리 공간 및 자원 등에 여유가 있었기 때문에 코드기반암호의 단점인 대규모의 큰 키에 많은 영향을 받지 않고 연산을 진행할 수 있다. 해당 환경에서는 키 생성, 배포에 자원이 더 할당되는 것이 좋을 것으로 보인다. 그러나 키 교환을 한번 수행하고 나면 개인키, 공개키 쌍은 오랜 기간 사용될 수 있어야 한다. 이에 코드기반암호 중 가장 오랜 역사를 갖고 있는 Goppa 코드를 활용하여 키 생성에는 자원이 많이 들지만 다른 암호 알고리즘에 비해 준수한 암호화 및 복호화 연산 성능을 보여주는 Classic McEliece와 NTS-KEM이 적합할 것으로 보인다. 높은 보안성을 자랑하기 때문에 해당 알고리즘들을 이용하면 양자 후 시대에도 안전하고 효율적인 통신이 가능할 것으로 보인다.

저성능 환경 같은 경우 성능평가 결과, ARM 프로세서 상에서 키 생성, 암호화/복호화 속도가 고성능 프로세서 환경보다 약 10배 느리다. 그리고 암호의 연산상의 효율성이 떨어질수록 해당 수치는 더 크게 영향을 끼치게 된다. 때문에 저사양 환경의 디바이스에 Classic McEliece와 같이 느린 속도의 암호를 탑재하기에는 적합하지 않을 것으로 보인다. 이와 같이 제한된 상황에서는 암호문을 얼마나 자주 교환하는지가 매우 중요할 것으로 보인다. 잦은 통신이 필요한 경우 느린 암호화, 복호화 속도를 가진 후보는 고려할 수 없다. 메모리 자원이 허용하는 한에서 안전하고 장기간 사용 가능한 키를 사용한 통신이 가장 적절할 것이다. 만약 이와 같은 조건이 만족되지 않는다면 ROLLO와 RQC와 같이 효율성을 중시한 코드기반암호가 현재로서는 최선의 선택일 것으로 보인다. 코드기반암호를 이와 같은 저전력 환경에서도 사용하기 위해서는 Goppa 보다 효율적인 QC시리즈 코드가 연구되었듯 앞으로 최적화에 대한 더 많은 연구가 필요할 것이다.

IV. 결론

본 논문에서는 고성능 환경 및 저전력 환경에서 NIST 표준화 공모전의 코드기반암호 7가지에 대한 성능평가를 진행하였다. 각 암호의 특성과 성능에 대하여 분석해 고성능 및 저전력 환경에서 사용되기 적합한 암호 알고리즘들 각각 구분하였다. 현실에는 더 많은 다양한 환경의 디바이스가 존재하기 때문에 각각의 환경에 맞는 양자내성암호가 필요할 것으로 예상된다. 그러나 NIST에서는 이번 공모전을 통해 하나의 표준만을 선출하는 것이 아닌 여러 가지의 양자내성암호가 다양한 분야에서 각자의 특성에 맞게 선출되어 활용될 것이라 밝힌바 있다. 해당 NIST의 표준화 공모전에 지속적인 관심을 갖고 지켜봐야 하며 양자내성암호에 대한 꾸준한 연구 또한 필요할 것으로 보인다.

[참고문헌]

[1] Daniel J, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Wen Wang, Classic McEliece: conservative code-based cryptography, 2019.

[2] Nicolas Aragon, Paulo S.L.M. Barreto, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, BIKE: Bit Flipping Key Encapsulation, 2019.

[3] Carlos Aguilar Melchor, Jean-Christophe

Deneuville, Nicolas Aragon, Philippe Gaborit, Slim Bettaieb, Edorado Persichetti, Hamming Quasi-Cyclic(HQC), 2019.

[4] Marco Baldi, Franco Chiaraluce, Gerardo Pelosi, LEDAkem: a post-quantum key encapsulation mechanism based on QC-LDPC codes, 2019.

[5] Carlos Aguilar Melchor, Philippe Gaborit, Nicolas Aragon, Adrien Hauteville, Magali Bardet, Ayoub Otmani, slim Bettalieb, Olieir Ruatta, ROLLO - Rank-Ouroboros, LAKE & LOCKER, 2019.

[6] Martin Albrecht, Carolos Cid, Kenneth G. Paterson, Cen Jung Tjhai, Martin Tomlinson, NTS-KEM, 2019.

[7] Carlos Aguilar Melchor, Alain Couvreur, Nicolas Aragon, Jean-Christophe Deneubille, Slim Bettaieb, Philippe Gaborit, Loic Bidoux, Adrien Hauteville, Oliveir Blazy, Gilles Zemor, Rank Quasi-Cyclic(RQC), 2019.

1. https://docs.google.com/document/d/1075XjELMU_EsOFq_3fzvjEL-eu5Ud11ORZAI5mblYmM/edit

Fixed-Base Comb 스칼라 곱 알고리즘에 대한 단일파형 부채널 공격 향상 기법*

박동준†, 이상엽†, 조성민†, 김희석‡, 홍석희†

†고려대학교 정보보호대학원, ‡고려대학교 사이버보안학과

A Method for Improving Single-Trace Side Channel Attack on Fixed-Base Comb Scalar Multiplication Algorithm*

Dongjun Park†, Sangyub Lee†, Sungmin Cho†, HeeSeok Kim‡, Seokhie Hong†

†Graduate School of Information Security, Korea University,
‡Department of Cyber Security, Korea University

요 약

Fixed-Base Comb 방법은 타원곡선암호의 핵심 연산인 스칼라 곱을 효율적으로 계산하는 방법이다. 그러나 일부 암호 라이브러리에서는 이 방법을 부채널 대응 기법 없이 사용하고 있다. 본 논문에서는 이 방법을 대상으로 하는 향상된 공격 기법을 제안하여, 부채널 대응 기법의 필요성을 강조한다.

I. 서론

타원곡선 전자서명 알고리즘은(Elliptic Curve Digital Signature Algorithm, ECDSA)은 미국의 표준으로 등록된 암호이다[1]. ECDSA는 기존의 RSA보다 짧은 키를 사용해도 같은 수준의 보안 강도를 보인다. 따라서 ECDSA는 자원이 한정적인 소형기기, IoT기기 등에 적합한 암호로 주목받고 있다.

ECDSA에서 가장 많은 시간이 소요되는 연산은 스칼라 곱(Scalar Multiplication, SM)이다. 효율적인 SM 구현하는 방법으로는 대표적으로 Fixed-Base Comb(FBC) 방법이 있다[2]. 이 방법은 고정된 점(Fixed-Base Point)과 관련된 몇 개의 점을 참조 테이블에 저장해둠으로써 SM을 효율적으로 계산한다. 암호 라이브러리인 OpenSSL[3], Bitcoin Core[4] 등에서도 FBC 방법으로 SM을 구현하였다.

암호를 구현할 때 연산 효율성만큼이나 중요한 요소는 부채널 공격 저항성이다. 부채널 공격은 암호의 수학적 안전성과는 별개로, 기기에서 발생하는 부채널 정보로 비밀정보를 획득하는 공격이다[5,6,7]. 주로 이용되는 정보는 전력 소비량과 전자파 방출량이다. 부채널 공격이 등장함에 따라 암호 기능이 탑재된 기기의 부채널 공격 저항성이 중요해졌다.

FBC 방법으로 SM을 구현한 암호 라이브러리 중 일부는 별도의 부채널 대응 기법을 적용하지 않고 있다[4]. 본 논문에서는 FBC 방법의 부채널 공격 저항성을 분석한다. 나아가 FBC 방법을 대상으로 하는 향상된 공격 기법을 제안하여, 부채널 대응 기법의 필요성을 강조한다.

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.1711051355, 단일파형 기반 충돌 부채널 공격에 안전한 공개키 암호 대응기술 연구)

II. 배경 지식

2.1 ECDSA 서명 생성에 대한 부채널 공격

알고리즘 1은 ECDSA의 서명 생성 과정이다.

알고리즘 1. ECDSA 서명 생성[1,2]
입력: 파라미터 (p,n,G,H), 개인키 d, 메시지 m.
출력: 메시지 m에 대한 서명 결과 (γ,σ).

1. Select $k \in [1,n)$ uniformly at random.
2. $(x,y) \leftarrow kG$. ◁ *Scalar Multiplication*
3. $\gamma \leftarrow x \bmod n$.
4. If $\gamma = 0$: go to step 1.
5. $\sigma \leftarrow k^{-1}(H(m)+d\gamma) \bmod n$.
6. If $\sigma = 0$: go to step 1.
7. Return (γ,σ).

공격자의 목표는 서명에 쓰인 개인키 d를 획득하는 것이다. 스칼라 k를 알 수 있다면 다음과 같이 d를 계산할 수 있다: $d \equiv \gamma^{-1}(k\sigma - H(m)) \bmod n$. 따라서 공격자의 목표를 이루는 한 방법은, 부채널 공격을 통해 2번째 줄에 있는 SM으로부터 k를 획득하는 것이다. 단, k는 서명 생성 시마다 난수가 선택되므로 한 번의 서명 생성에 대한 부채널 정보만으로도 수행 가능한 단일파형 부채널 공격 기법이 요구된다.

2.2 FBC 방법에 대한 부채널 공격

알고리즘 2는 FBC 방법의 계산 과정이다.

알고리즘 2. Fixed-Base Comb 방법의 SM[2]
입력: 점 G, 스칼라 $k = (k_{l-1} \cdots k_1 k_0)_2$, 윈도우 너비 w.
출력: $P = kG$.

1. Divide k into w bit strings each of the same length $t = l/w$ so that $k = K^{w-1} \| \cdots \| K^1 \| K^0$. Let K_i^j denote ith bit of K^j and $K_i = (K_i^{w-1} \cdots K_i^1 K_i^0)_2$.
2. For all bit strings $(b_{w-1} \cdots b_1 b_0)_2$ of length w, compute $G[(b_{w-1} \cdots b_1 b_0)_2] = b_{w-1} 2^{(w-1)t} G + \cdots + b_1 2^t G + b_0 G$.
3. $P \leftarrow G[K_{t-1}]$.
4. For i from $t-2$ down to 0:
   4.1. $P \leftarrow 2P$. ◁ *Doubling*
   4.2. $P \leftarrow P + G[K_i]$. ◁ *Addition*
5. Return P.

2번째 줄의 참조 테이블을 생성하는 과정에서 점 G는 공개된 값이므로, 참조 테이블에 저장될 값은 누구나 알 수 있다. FBC 방법에 대한 전력 파형을 가진 공격자는 덧셈의 입력 $G[K_i]$를 추측하여 얻은 이론적 전력 소비량과, 실제 전력 파형 사이의 상관관계를 분석함으로써 K_i를 획득할 수 있다[6].

III. FBC 방법의 부채널 공격 저항성

이 장에서는 대표적인 단일파형 부채널 공격 기법인 단순 전력 분석(Single Power Analysis, SPA)[5], 수평 상관 분석(Horizontal Correlation Analysis, HCA)[6], 단일파형 충돌 공격(Single-Trace Collision Attack, STCA)[7]을 소개한다. 또한, 각 기법에 대한 FBC 방법의 부채널 공격 저항성을 분석한다.

3.1 단순 전력 분석(SPA)

SPA[5]는 비밀정보에 의존하여 수행되는 연산의 전력 패턴을 식별하여 비밀정보를 획득하는 기법이다. 예를 들어, 이진 SM은 $k_i=0$일 때 덧셈을 수행하지 않고 넘어간다. 이진 SM의 전력 파형으로부터 i번째 두배 후에 덧셈이 수행되지 않았다는 사실을 알아낼 수 있다면 $k_i=0$으로 확정할 수 있다(그림 1).

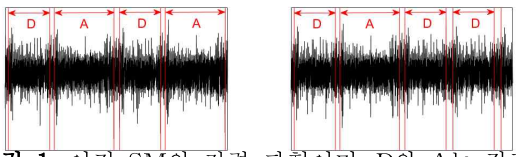


그림 1. 이진 SM의 전력 파형이며, D와 A는 각각 두배와 덧셈을 의미. 입력은 0b11(좌), 0b100(우).

FBC 방법에서는 2^w개의 점 중 하나가 더해진다. 공격자는 덧셈이 수행되었다는 사실을 알아내더라도 어떤 점과의 덧셈인지 알 수 없으므로 K_i를 확정할 수 없다. 따라서 FBC 방법은 SPA로부터 안전하다.

3.2 수평 상관 분석(HCA)

HCA[6]는 소비하는 전력이 연산 입력값의 함수인 전력 모델과 상관관계가 있다는 점을 이용한다. 하나의 덧셈은 여러 개의 체 곱셈으로 구성되어 있고, 소프트웨어 환경에서 체 곱셈은 여러 번의 단위 곱셈으로 처리된다. HCA는 단위곱셈 입력값을 추측하여 얻은 이론적 전력 소비량과 실제 전력 소비량 사이의 상관관계를 분석하여, 추측한 입력값이 실제로 연산에 관여했음을 확인하는 통계 분석 기법이다.

HCA는 이용 가능한 단위곱셈의 파형이 많을수록 성공률이 높아진다. 알고리즘 2에서 덧셈은 일반적으로 11개의 체 곱셈으로 구성되는데, 이 중 점 $G[K_i]$와 직접 관련이 있는 2개만이 HCA에 이용된다[2]. 기기의 데이터 처리 단위를 W비트라 할 때, l비트의 체 곱셈은 $(l/W)^2$번의 단위곱셈으로 처리되므로 가용 단위곱셈 파형의 총 개수는 $2(l/W)^2$이다.

FBC 방법에 대하여 HCA를 수행할 때, 스칼라의 워드 K_i를 올바르게 추측하지 못하는 상황이 발생한다(그림 2). 이는 분석에 이용된 단위곱셈 파형의 개수가 불충분하여 발생하는 현상으로 해석할 수 있다. 이러한 상황에서 공격자는 HCA를 통해 올바른 스칼라 K를 획득할 수 없다.

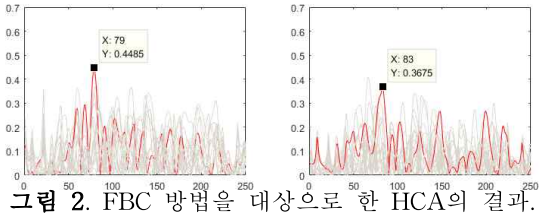


그림 2. FBC 방법을 대상으로 한 HCA의 결과. 올바른 중간값과의 상관계수(적색)가 그 외의 상관계수(회색)보다 높게 나와야 성공이지만(좌), 일부 덧셈에서 역전되는 상황이 발생(우).

3.3 단일파형 충돌 공격(STCA)

STCA[7]는 같은 입력으로 같은 연산을 처리하는, 이른바 충돌하는 두 연산의 전력 소비량이 서로 상관관계가 존재한다는 가정하에 이루어지는 공격 기법이다. 반복되는 특정 연산에 대한 전력 파형들 중에서, 상관관계가 있는 두 파형을 식별하여 두 연산의 입력이 같다고 판단하는 것이다.

두 파형 간의 상관관계를 나타내는 지표는 다양하게 연구되었다. 두 파형이 상관관계가 있을 때 큰 값을 갖는 피어슨 상관계수와, 반대로 두 파형이 상관관계가 있을 때 작은 값을 갖는 DoM(Difference of Mean), ED(Euclidean Distance), ST(Squared Pair wise T-difference) 등이 알려져 있다.

STCA로는 FBC 방법을 공격하기 어렵다. 공격자는 STCA를 통해 $t-1$개의 모든 덧셈을 같은 참조 테이블을 호출하는 덧셈끼리 묶어 2^w개의 집단으로 군집화할 수 있다. 그러나 공격자는 특정 집단이 어떤 점을 호출하는지 알 수 없으므로, $2^w!$번의 무리한 추측을 시도하는 것이 최선이기 때문이다.

IV. 제안하는 공격 기법

제안하는 공격 기법의 목표는 STCA를 이용하여, HCA 수행 시 단위곱셈 파형의 개수가 불충분해 발생하는 현상을 해소하는 것이다. 본 기법은 STCA를 수행하는 충돌 분석 단계와, HCA를 수행하는 상관분석 단계로 구성된다.

4.1 충돌 분석 단계

충돌 분석 단계에서는 STCA를 통해 서로 같은 입력을 호출한 덧셈을 식별하여, 모든 단위곱셈 파형을 2^w개의 집단으로 군집화한다.

알고리즘 2의 i번째 덧셈은 11개의 체 곱셈으로 구성되어 있다. 이 중 $G[K_i]$와 직접 관련이 있는 2개의 체 곱셈을 구성하는 총 $2(l/W)^2$개의 단위곱셈 파형을 차례로 $\epsilon_i[0],\cdots,\epsilon_i[2(l/W)^2-1]$라 하자. 단위곱셈 파형이 m개의 샘플로 이루어진 데이터라면, ϵ_i는 $2(l/W)^2\times m$ 크기의 배열로 나타낼 수 있다(그림 3).

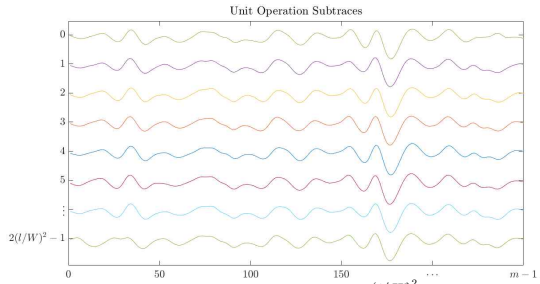


그림 3. 단위곱셈 파형 $\epsilon_i \in \mathbb{R}^{2(l/W)^2 \times m}$의 모습.

i번째와 j번째 덧셈의 입력 $G[K_i]$와 $G[K_j]$가 같은지 판단하는 지표로는 ST(이하 $T^2_{i,j}$)를 사용하였다. $T^2_{i,j}$값은 두 집단의 분산과 개수를 고려한 평균적 차이를 나타낸다. ϵ_i의 평균 파형을 μ_i, 분산 파형을 σ_i, 파형의 개수를 n_i라 할 때, ϵ_i와 ϵ_j에 대한 $T^2_{i,j}$값은 아래와 같이 정의된다.

$$T^2_{i,j} = \left(\frac{\mu_i - \mu_j}{\sqrt{\frac{\sigma_i}{n_i} + \frac{\sigma_j}{n_j}}} \right)^2$$

모든 시점에 대해서 계산한 $T^2_{i,j}$값의 합을 SOST (Sum of Squared Pairwise T-differences)라 한다. SOST가 문턱 값 이하일 때 ϵ_i와 ϵ_j는 충돌이 발생한 것으로 판단한다. 문턱 값은 충돌 확률$(1/2^w)$을 고려하여 평균 T^2값의 일정 비율로 설정한다.

알고리즘 2의 반복자 i에 대하여, 덧셈에서 충돌이 발생한 i끼리 집단화한 집합을 $\mathbb{I}_0, \cdots, \mathbb{I}_{2^w-1}$로 정의하자. 각 집합에 포함되는 단위곱셈 파형은 평균적으로 $2(l/W)^2(t-1)/2^w$개다.

4.2 상관 분석 단계

상관 분석 단계에서는, 충돌 분석 단계에서 군집화된 파형들이 참조 테이블의 어떤 점을 호출하는지 HCA를 통해 알아낸다.

집합 $\mathbb{I}_*$가 $G[guess]$를 호출한다고 추측했을 때, 전력 모델 L을 가정하여 얻는 이론적 전력 소비량은 $X = L(G[guess])$이다. $\mathbb{I}_*$에 포함된 단위곱셈들의 실제 전력 파형의 평균 파형을 $Y = \left(\sum_{\forall i \in \mathbb{I}_*} \epsilon_i \right) / \left(\sum_{\forall i \in \mathbb{I}_*} 1 \right)$라 할 때, X와 Y의 상관계수 $\rho_{guess,*}$는 아래와 같다.

$$\rho_{guess,*} = \frac{Cov(X,Y)}{\sqrt{Var(X)}\sqrt{Var(Y)}}$$

모든 $0 \le guess < 2^w$에 대하여 상관계수를 계산한다. 이중 상관계수의 값이 가장 큰 $guess$를 $\mathbb{I}_*$가 호출한 참조 테이블의 인덱스 K_*라고 추측할 수 있다.

V. 실험 결과

본 논문에서는 스칼라의 비트 길이가 $l = 256$인 ECDSA-p256r1[1]의 파라미터로 실험을 진행하였다. 윈도우 너비는 $w = 4$를 사용하여 참조 테이블을 구성하는 점은 $2^w = 16$개다. 실험에 사용된 프로세서는 ARM Cortex-M4로, 데이터 처리 단위는 $W = 32$비트이고, 동작 주파수는 7.37 MHz이다.

그림 4는 50 MS/s의 샘플링레이트로 측정한 프로세서의 전력 파형이다. 그림 4의 우측은 파형의 체곱셈 부분을 확대한 모습으로, 64번의 단위곱셈 파형이 각 250개의 샘플로 이루어져 있는 것을 보여준다.

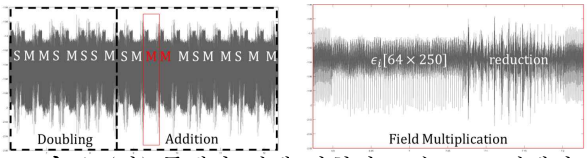


그림 4. (좌) 두배와 덧셈 파형의 모습으로, 덧셈의 3, 4번째 체 곱셈(적색)만이 $G[K_i]$와 관련 있음. (우) 덧셈 파형의 3번째 체 곱셈을 확대한 모습.

5.1 충돌 분석 단계

알고리즘 2의 반복문(4번째 줄)은 $t-1 = 63$번 반복되므로 덧셈 파형을 63개 얻을 수 있다. 각 덧셈에서 $G[K_i]$와 관련 있는 3, 4번째 체 곱셈으로부터 총 $2(l/W)^2 = 128$개의 단위곱셈 파형을 얻을 수 있다. 충돌은 평균적으로 $(t-1)/2^w \approx 4$번 발생하므로, 가용 단위곱셈 파형은 $2(l/W)^2(t-1)/2^w \approx 512$개다.

그림 5는 좌측부터 차례로 $T^2_{i,i'}$, $T^2_{i,i''}$, $T^2_{i,j}$값을 시점별로 나타낸 것이다. $T_{i,i'}$, $T_{i,i''}$값은 전체적으로 작지만, $T^2_{i,j}$값은 82번째 시점에서 큰 정점을 보인다. 따라서 i번째와 i', i''번째 덧셈은 서로 같은 참조 테이블을 호출한다고 추측할 수 있다. 반면 i번째와 j번째 덧셈은 서로 다른 참조 테이블을 호출한다고 추측할 수 있다.

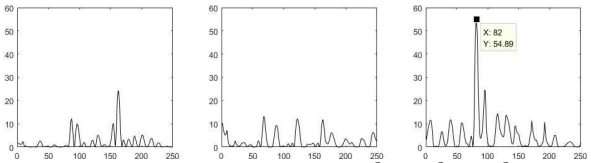


그림 5. 좌측부터 차례로 $T^2_{i,i'}$, $T^2_{i,i''}$, $T^2_{i,j}$.

$T^2_{i,i'}$값 또한 특정 시점에서 작은 정점이 발생한 것을 확인할 수 있다. 그러나 모든 시점에 대하여 계산한 $T^2_{i,j}$값의 합인 SOST가 문턱 값보다 작으므로 충돌이 발생한 것으로 판단할 수 있다.

5.2 상관 분석 단계

그림 6은 모든 $0 \le guess < 2^w$에 대하여 $\rho_{guess,*}$를 계산한 결과이다. 올바른 추측으로 계산한 상관계수는 적색으로 나타냈다. 그림 6의 좌측은 충돌 분석 단계 없이 하나의 ϵ_i만 이용한 것이고(그림 2의 우측과 동일), 그림 6의 우측은 충돌 분석 단계로 찾아낸 $(t-1)/2^w \approx 4$개의 $\epsilon_{\forall i \in \mathbb{I}_*}$를 이용한 것이다. HCA만 단독으로 수행했을 때는 덧셈이 호출하는 점을 잘못 추측했지만, 제안하는 기법은 올바르게 추측하였다.

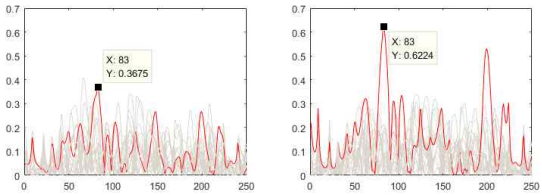


그림 6. (좌) HCA 단독. (우) STCA 병행한 HCA.

제안하는 공격 기법으로는 스칼라 K_i의 최상위 워드*를 제외한 $t-1=63$개 워드를 전부 획득하는데 성공하였다.

VI. 대응 기법

본 논문에서 제안하는 기법을 시도하기 위해서는 이론적 전력 소비량 $X=L(G[guess])$를 계산할 수 있어야 한다. 이는 암호의 파라미터 중 하나인 점 G가 공개되어 있어 가능한 일이다. 점 G를 난수화 시켜 그 값을 알 수 없게 함으로써 대응할 수 있으며, 이러한 부류의 대응 기법을 블라인딩이라 한다[8].

첫 번째 블라인딩 기법은 임의의 점 R를 선택해 $P=kG$ 대신 $P=k(G+R)-S$를 계산하는 것이다 (단, $S=kR$). 두 번째 기법은 동차좌표계를 도입하는 것이다. 동차좌표계에서는 임의의 r에 대하여 점 $G(x,y)$를 $G(rx:ry:r)$로 나타낸다. 마지막 기법은 기존 곡선과 동형인 다른 곡선상의 점 $G^*=\psi(G)$에 대하여 $P=\psi^{-1}(kG^*)$를 계산하는 것이다. 세 가지 기법 모두 추가 연산을 요구하므로 부채널 분석 저항성과 효율성 사이에서 적절히 조율하여야 한다.

단위곱셈의 순서를 섞음으로써 분석을 어렵게 할 수도 있는데, 이러한 부류의 대응 기법을 셔플링이라 한다[6]. 셔플링과 블라인딩을 함께 적용하면 공격자의 계산 복잡도를 높일 수 있다.

VII. 결론

본 논문에서는 FBC 방법의 단일파형 부채널 공격 저항성을 분석하였다. FBC 방법은 SPA, STCA로부터 안전하지만, HCA에 의해 스칼라 일부가 노출될 수 있음을 보였다. 본 논문에서 제안하는 기법으로는 스칼라 전부를 획득할 수 있었다. 따라서 FBC 방법을 사용할 때에는 반드시 부채널 대응 기법을 적용해야 한다.

제안하는 후속 연구 방향은 두 가지이다. 첫째는 충돌이 발생하는 시점인 PoI(Point of Interest)를 정밀하게 추출하는 것이다. 이 경우 충돌 분석 단계에서 덧셈을 잘못 분류할 확률을 낮추며, 그림 5에서 언급한 해석의 오류도 예방할 수 있다.

둘째는 상용 장비에 대해서 동일한 실험을 재현하는 것이다. 본 논문에서는 저사양의 프로세서에서 실험을 진행하여 양질의 부채널 정보를 획득할 수 있었지만, 실제 운용되고 있는 고사양의 상용 장비로부터 양질의 부채널 정보를 획득하기는 어렵다.

* 알고리즘 2에서 최상위 워드는 덧셈 대신 대입으로 처리.

[참고문헌]

[1] NIST, Digital signature standard, https://doi.org/10.6028/NIST.FIPS.186-4, 2013.
[2] Darrel Hankerson, Alfred Menezes, Scott Vanstone. Guide to elliptic curve cryptography. Springer US. 2011.
[3] OpenSSL, https://github.com/openssl/openssl.
[4] Bitcoin Core, https://github.com/bitcoin/bitcoin.
[5] Paul Kocher, Joshua Jaffe, Benjamin Jun. Differential power analysis. Annual International Cryptology Conference. 1999.
[6] Christophe Clavier, Benoit Feix, Georges Gagnerot, Mylene Roussellet, V. Verneuil. Horizontal correlation analysis on exponentiation. International Conference on Information and Communications Security. 2010.
[7] Neil Hanley, HeeSeok Kim, Michael Tunstall. Exploiting collisions in addition chain-based exponentiation algorithms using a single trace. Cryptographers' Track at the RSA Conference. 2015.
[8] Michael Tunstall, Marc Joye. Coordinate blinding over large prime fields. International Workshop on Cryptographic Hardware and Embedded Systems. 2010.

mbedTLS 상에서의 양자내성암호 적용 가능성 연구

윤영여*, 박찬희*, 손수민*, 이지은**, 김호원*

*부산대학교 전기컴퓨터공학부 **한국인터넷진흥원

A Study on Application Possibility of Post Quqntum Cryptograpy with mbedTLS

Youngyeo Yun*, Chanhui Park*, Sumin Son*, Jieun Lee**, Howon Kim*

* Department of Electrical and Computer Engineering,

Pusan National University

** Korea Internet & Security Agency

요 약

기존 TLS에서 사용되는 키 교환 알고리즘은 이산대수문제를 기반으로 안전도를 보장하지만 양자컴퓨터 환경에서는 이산대수문제를 쉽게 해결할 수 있다. 미국 국립표준기술연구소에서는 양자컴퓨터 환경에서도 안전한 양자내성암호의 표준 제정을 위해 양자내성암호 표준 공모전을 개최하였다. 공모전에 기제출된 격자기반 양자내성암호 Lizard와 NewHope 기반의 Ciphersuite 구현 결과를 제시하고 handshake 성능 측정 결과를 제시하였다. 이를 통해 mbedTLS 상에서의 양자내성암호 적용 가능성을 알아본다.

I. 서론

현재 보안 프로토콜로 많이 사용되는 TLS(Transport Layer Security)의 handshake 과정에서 보안통신에 사용될 키를 공유하는 키 교환 과정이 있다. 이 때, 키 교환 알고리즘으로 DH(Diffie Hellman), DHE(Diffie Hellman Ephermeral), ECDH(Elliptic Curve Diffie Hellman), ECDHE(Elliptic Curve Diffie Hellman Ephermeral) 등을 사용한다.

기존 키 교환 알고리즘은 이산대수문제를 기반으로 안전도를 보장하는데 양자컴퓨터 환경에서는 쇼어 알고리즘[1]에 의해 이산대수문제가 쉽게 해결된다. 최근 구글에서 특정 문제에 대해 슈퍼컴퓨터를 압도하는 양자우위를 달성한 양자 칩[2]을 발표하는 등 양자컴퓨터의 위협이 다가오고 있다.

미국 국립표준기술연구소(NIST)의 Report on Post-Quantum Cryptography[3]에서 양자내성암호의 중요성에 대해 강조하고 있다. 또한, NIST에서는 양자내성암호 표준 공모전을 개최하였으며, 69종의 1차 후보군이 선택되었다. 현재 2차 후보군 17종에 대한 선정이 완료되었으며, PQC forum을 통해 해당 후보군에 대한 안전성 검증이 진행되고 있다.

NIST 양자내성암호 표준 공모전에 제출된 격자기반 양자내성암호 NewHope[4]와 Lizard[5]를 mbedTLS Ciphersuite에 추가된 결과물을 통해 성능 평가 결과를 제시하고자 한다. mbedTLS 상에서의 NewHope은 기존에 구현된 코드를 활용하였고 Lizard 기반의 알고리즘인 RLizard.KEM에 대해서는 직접 구현하였다.

II. 본론

본 논문에서는 양자컴퓨터 환경에서도 안전한 양자내성암호를 mbedTLS에 적용한다. 또한, 기존에 사용되고 있는 공개키 기반의 키 교

환 알고리즘과 성능 비교 결과를 제시하고 mbedTLS 상에서의 양자내성암호 적용 가능성에 대해 알아보고자 한다.

2.1. TLS Ciphersuite

TLS 표준에서는 보안통신을 위해 사용되는 알고리즘을 Ciphersuite이라는 형태로 정의한다. TLS에서 사용되는 Ciphersuite은 그림 1과 같은 구조를 가진다.



그림 1. TLS Ciphersuite 구조

TLS Ciphersuite은 앞에서부터 순서대로 통신에 사용하는 프로토콜의 이름(TLS), 통신대상 간에 키를 공유하기 위한 키 교환 방식(RLIZARD), 인증서 검증을 위한 서명 알고리즘(ECDSA), 데이터 암복호화를 위한 블록 암호 알고리즘(ARIA_128) 및 운용 모드(CBC), 데이터 무결성을 위한 메시지 인증 알고리즘(SHA256)을 정의한다.

2.2. KEM 기반 TLS Handshake

양자내성암호 Lizard와 NewHope 기반의 키를 교환하는 방법들 중 하나로 KEM(Key Encapsulation Mechanism) 방식이 있다. 그림 2는 KEM 기반의 TLS handshake 동작 과정이다. 그림 2의 Server Key Exchange와 Client Key Exchange 메시지가 양자내성암호가 적용되는 부분이다. 그림 2의 (1)에서 공개키(PK), 비밀키(SK)를 생성하고 Server Key Exchange 패킷을 통해 클라이언트로 공개키를 전송한다. 그림 2의 (2)에서 클라이언트는 받은 공개키로 공유키(Shared Secret)와 암호문(Ciphertext)을 만들어 Client Key Exchange 패킷을 통해 암호문을 서버로 전송한다. 그림 2의 (3)에서는 서버가 받은 암호문으로 클라이언트와 동일한 공유키를 생성한다.

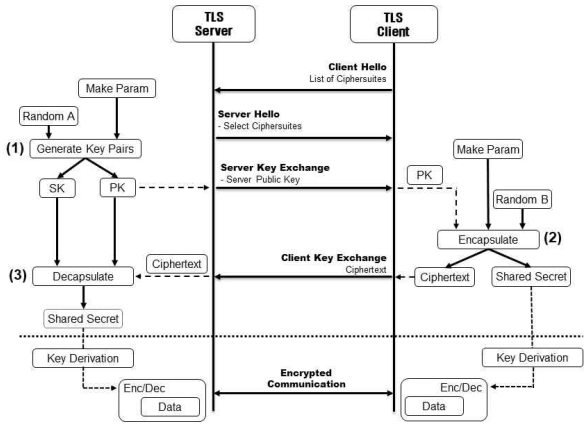


그림 2. KEM 기반의 TLS handshake

2.3. API

키 교환 알고리즘으로 Lizard 기반의 RLizard.KEM 또는 NewHope를 선택하여 KEM 방식으로 키 교환을 한다. 선택한 알고리즘을 mbedTLS에 적용하기 위한 API는 표 1, 2와 같다. RLizard.KEM과 NewHope의 암호화 방식은 다르지만 mbedTLS 상에서의 API 구조는 비슷하다. 표 1, 2를 참조하면 서버에서 키를 생성하는 API (표 1-1, 2-1), 공개키를 전송하는 API(표 1-2, 2-2), 클라이언트에서 공개키를 받는 API(표 1-3, 2-3), 암호문을 생성하는 API(표 1-4, 2-4), 클라이언트에서 공유키를 생성하는 API(표 1-4, 2-6), 암호문을 전송하는 API(표 1-5, 2-5), 서버에서 암호문을 받는 API(표 1-6, 2-7), 서버에서 공유키를 생성하는 API(표 1-7, 2-8)이 있다.

표 1. RLizard.KEM API

RLizard.KEM API
1. mbedtls_rlizard_gen_key_pair_server
2. mbedtls_rlizard_make_params_server
3. mbedtls_rlizard_parse_public_key_from_serve
4. mbedtls_rlizard_encapsulation_client
5. mbedtls_rlizard_make_params_client
6. mbedtls_rlizard_parse_cipher_text_from_client
7. mbedtls_rlizard_decapsulation_from_client

표 2. NewHope API

NewHope API
1. mbedtls_newhope_keygen_server
2. mbedtls_newhope_make_params_server
3. mbedtls_newhope_parse_public_value_from_server
4. mbedtls_newhope_gen_public_client
5. mbedtls_newhope_make_params_client
6. mbedtls_newhope_calc_secret
7. mbedtls_newhope_read_public_from_client
8. mbedtls_newhope_create_server_shared_value_n1024

2.4. 성능평가

mbedTLS 라이브러리는 기존 키 교환 알고리즘에 대한 성능 측정 코드를 제공한다. 네트워크 환경을 고려하지 않고 일정 시간 동안 키 교환에 사용되는 모든 API를 수행한 횟수로 측정하여 handshake/s 단위로 결과가 나온다. Ciphersuite에 새로 추가한 RLizard.KEM, NewHope에 대하여 mbedTLS에서 제공하는 측정 방법과 같이 키 교환에 사용되는 모든 API를 몇 번 수행하는지 확인한다. 결과는 표 3과 같다. RLizard.KEM은 기존 알고리즘 성능 평균에 약 2배 향상된 성능을 보이고 NewHope은 12배 향상된 성능을 보인다.

표 3. 성능평가 결과

※환경: Ubuntu 18.04, Intel(R) Core(TM) i7-8700k CPPU @ 3.70GHz

키 교환 알고리즘	성능
DH-2048	213 handshake/s
DHE-2048	109 handshake/s
ECDH-Curve25519	1281 handshake/s
ECDHE-Curve25519	638 handshake/s
RLIZARD-CAT1	1179 handshake/s
NewHope-1024	6729 handshake/s

2.5. 알고리즘 별 전송 데이터 크기

TLS 1.2 표준에서는 1개의 handshake 패킷에 대해 최대 크기를 16KB(16384byte)로 제한한다. 키 교환 과정에서 기존 공개키 기반의 키 교환 알고리즘과 양자내성암호의 최대 전송 데이터의 크기는 표 4와 같다.

표 4. 전송 데이터 최대 크기 비교

키 교환 알고리즘	최대 전송 데이터 크기
DH-2048	256 byte
DHE-2048	256 byte
ECDH-Curve25519	32 byte
ECDHE-Curve25519	32 byte
RLIZARD-CAT1	8256 byte
NewHope-1024	3680 byte

양자내성암호를 사용할 경우 더 많은 양의 데이터 송수신이 발생한다. 현재 RLizard와 NewHope의 경우 Handshake 단계에서 데이터의 크기가 16KB를 초과하지 않는다.

그러나 FrodoKEM, NTS-KEM, 그리고 Lizard.KEM 등 일부 양자내성암호에서 16KB 이상의 공개키 및 암호문이 생성되므로 Handshake 단계에서 Message Fragment 기능이 필요할 것으로 예측된다.

III. 결론

본 논문에서는 범용적으로 사용되는 mbedTLS에 양자내성암호를 적용한 Ciphersuite 구현 결과물에 대한 분석 결과를 제시하였다. 양자내성암호를 적용한 Ciphersuite의 성능을 비교해본 결과 비교적 빠른 연산 결과를 확인할 수 있었다. 향후 다수의 양자내성암호를 mbedTLS상에 적용할 경우 메시지를 나누어 보내기 위한 handshake message fragment 기능이 구현될 경우 다양한 양자내성암호를 적용하는 것이 가능할 것으로 예측된다.

ACKNOWLEDGMENT

이 논문은 2019년 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2017-0-00616, 격자기반 양자내성 공개키암호 스킴 개발)

[참고문헌]

[1] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput., 26 (5), 1997.
[2] Arute, F., Arya, K., Babbush, R. et al, Quantum supremacy using a programmable superconducting processor. Nature 574, 505 - 510, 2019.
[3] Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Liu, Y. K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., and Smith-Tone, D., NISTIR 8240 Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process, 2016.
[4] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - a new hope, IACR Cryptology ePrint Archive report 2015/1092, 2015.
[5] Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song, Lizard: Cut off the tail! Practical post-quantum public-key encryption from LWE and LWR, Cryptology ePrint Archive. 2016.

비트 퍼뮤테이션 기반 블록 암호에 대한 신규 부채널 분석 동향

임성혁*, 문혜원*, 박소연*, 우지은*, 이태호*, 한동국*,**

*국민대학교 정보보안암호수학과

**국민대학교 금융정보보안학과

New Side-Channel Analysis Trend for Block Ciphers based Bit Permutation

Seong-Hyuck Lim*, Hye-Won Mun*, So-Yeon Park*, Ji-Eun Woo*, Tae-Ho Lee*, Dong-Guk Han*,**

*Dept. of Information Security, Cryptology, and Mathematics, Kookmin University,

**Dept. of Financial Information Security, Kookmin University.

요 약

경량 암호는 메모리, 성능이 제한된 스마트 기기 및 IoT 장비 등의 환경에서 구현하기 위해서 만들어진 알고리즘이다. 이러한 환경에서는 확산함수로 비트 퍼뮤테이션이 종종 사용된다. 그러나 안전성 보장을 위해 사용한 비트 퍼뮤테이션의 특징이 부채널 정보를 이용하면 키를 비교적 쉽게 분석할 수 있는 요인으로 작용한다. 본 논문에서는 기존 DPA, CPA보다 시간과 메모리 측면에서 더 효율적으로 분석하는 방법에 대한 기술 동향과 실제 적용 논리를 소개한다.

I. 서론

최근 IoT 기기 및 스마트 기기의 기술 발전에 따라 이용자 수가 증가하면서 보안에 대한 문제가 대두되고 있다. 이 기기들은 기존 암호 알고리즘들을 사용할 경우 메모리나 속도 측면에서 부적합하므로 경량 암호를 사용한다. 특히 몇몇 경량 암호는 메모리를 적게 사용하면서 기존의 안전성을 가지기 위해 비선형 함수와 함께 선형 함수로 비트 퍼뮤테이션을 사용한다. 그러나 비트 퍼뮤테이션의 성질과 부채널 정보를 이용하면 키 정보가 노출될 수 있는 취약점을 가지고 있다.

부채널 분석(Side-Channel Analysis)은 암호화가 진행되는 동안 발생하는 부채널 정보(전력, 전자파 등)를 사용하여 키에 대한 정보를 얻는 공격방법이다. 부채널 분석의 한 방법인 차분전력분석(Differential Power Analysis, DPA)과 상관전력분석(Correlation Power Analysis, CPA)은 암호 알고리즘이 동작하는 동안 발생한 소비 전력들을 수집하여 그 변화를 관찰한다. DPA와 CPA는 선형 연산보다 비선형 연산을 타겟으로 하는 것이 더 효율적이라고 알려져 있다. 그러나 이 방법은 많은 수의 파형이 필요하므로 많은 시간과 메모리가 요구된다. 그래서 이 단점을 보완할 수 있는 부채널 분석인 SCADPA[1]가 제시되었다. 이 방법은 비트 퍼뮤테이션의 성질을 이용해 적은 수의 파형으로

도 키를 복구할 수 있다. 본 논문에서는 SCADPA 소개와 PRESENT-80에 대한 실험 결과를 제시한다.

II. 관련 연구

2.1 PRESENT-80 암호 소개

PRESENT-80은 SPN(Substitution Permutation Network)구조의 암호 알고리즘으로 블록과 비밀키가 각각 64비트, 80비트이고 총 31라운드로 구성되어 있다.[2]

2.1.1 기호와 표기

[표 1]은 알고리즘 설명에 필요한 기호들을 정리한 것이다.

[표 1] PRESENT 암호 알고리즘에 대한 기호

기호	설명
MK	비밀키 (80 비트)
P	평문 (64 비트)
C	암호문 (64 비트)
K_i	라운드 키 (64 비트) $(1 \le i \le 32)$
k_i	비밀키의 $i(0 \le i \le 79)$번째 비트
b_i	평문의 $i(0 \le i \le 63)$번째 비트
$\oplus$	XOR(eXclusive OR) 연산
RC	Round Counter

2.1.2 PRESENT-80 키 스케줄

각 라운드에 사용할 라운드 키를 비밀키로부터 생성한다.

$$MK = k_{79}k_{78}\cdots k_0$$

$$K_i = k_{79}k_{78}\cdots k_{16}$$

라운드 키를 생성한 후 다음과 같이 키 업데이트를 수행한다.

(1) $[k_{79}k_{78}\cdots k_1k_0] = [k_{18}k_{17}\cdots k_{20}k_{19}]$

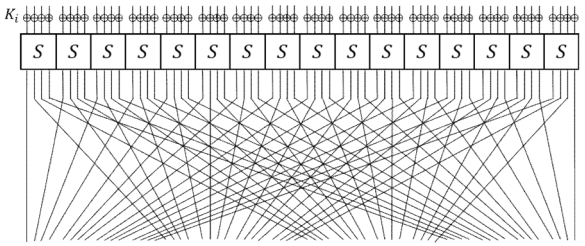
(2) $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$

(3) $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus RC$

2.1.3 PRESENT-80 암호화 과정

한 라운드는 State와 라운드 키를 XOR하는 AddRoundKey, 비선형 연산인 sBoxLayer, 비트를 확산시키는 pLayer로 이루어져 있다. 마지막 라운드 이후 AddRoundKey를 한 번 더 수행한다.

[그림 1]은 PRESENT-80의 한 라운드를 나타낸 것이다.



[그림 1] PRESENT-80 한 라운드

sBoxLayer의 SBox는 [표 2]와 같다.

[표 2] PRESENT-80 SBox

x	0	1	2	3	4	5	6	7
$S(x)$	C	5	6	B	9	0	A	D
x	8	9	A	B	C	D	E	F
$S(x)$	3	E	F	8	4	7	1	2

III. SCADPA

본 절에서는 SCADPA와 PRESENT-80에 대한 적용 논리와 실험 결과를 소개한다.

3.1.1 SCADPA

Side-Channel Assisted Differential Plaintext Attack(SCADPA)[1]은 한 니블이 다른 두 선택 평문 쌍을 암호화할 때 발생하는 전자파를 이용한 공격이다. 앞서 언급했던 통계적인 분석법인 DPA, CPA보다 비교적 적은 수의 평문을 사용하기 때문에 시간과 메모리 공간을 절약할 수 있다. 또한 SCADPA는 비선형 연산뿐만 아니라 선형연산인 비트 퍼뮤테이션의 성질도 이용한다.

비트 퍼뮤테이션은 SBox의 출력값이 다른 SBox들의 입력값에 영향을 주는 연산이다. 따라서 영향을 받은 SBox들의 입력값의 변화를 관찰하면 비트 퍼뮤테이션을 역연산하여 이전

라운드의 SBox의 출력값을 추측할 수 있다. 이 성질을 이용하면 선택 평문들을 암호화할 때의 전자파 차분으로 SBox 출력값의 차분을 알아낼 수 있다.

SCADPA 과정은 다음과 같다:

(1) 선택 평문 p를 키 k로 암호화한다. ($E_k(p)$).

(2) 2라운드 암호화가 진행하는 동안의 전자파를 수집하여 trace t를 얻는다.

(3) i번째 니블이 다른 평문 $p'(\neq p)$을 선택한다. 평문 p, p'의 i번째 니블을 각각 p_i, p_i', 키 k의 i번째 니블을 k_i라고 했을 때, $x_i = p_i \oplus k_i$, $x_i' = p_i' \oplus k_i$이다.

(4) (2)와 같은 방법으로 $E_k(p')$에 대한 trace t'을 얻는다.

(5) t와 t'의 전자파 차분인 Δt를 계산한다.

(6) $S(x_i')$의 출력값이 비트 퍼뮤테이션을 거치면서 영향을 주는 니블의 위치를 파악한다.

(7) 그 위치의 Δt를 이용해 SBox 출력값의 변화 $\Delta S_i = S(x_i) \oplus S(x_i')$를 추측한다. 이 때, t와 t'은 한 니블이 다른 평문들이 동일한 암호화를 통해 얻어지므로 Δt는 해당 위치에서 변화가 있다면 의미있는 차분을 가질 것으로 기대된다.

(8) $\Delta S_i = S(p_i \oplus k_i) \oplus S(p_i' \oplus k_i)$를 만족하는 k_i를 계산해서 k_i의 후보들을 찾는다.

(9) 또 다른 p_i'에 대해서 (3)-(7)을 반복하여 k_i 후보를 줄이고 올바른 k_i의 값을 구한다.

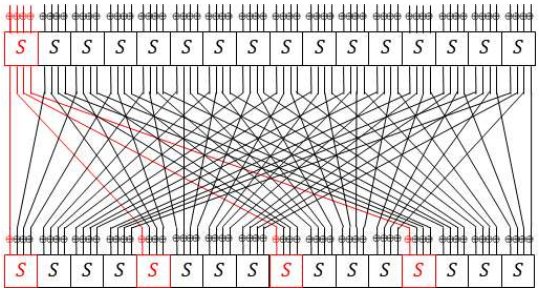
(10) 평문의 모든 니블에 대해서 (3)-(8)을 반복하여, 1라운드 키를 찾는다.

(11) 비밀키를 찾기 위해, 다음 라운드에서 전수조사를 하거나 SCADPA를 반복한다.

최선의 경우, 3개의 평문으로 라운드 키의 한 니블을 찾을 수 있다. 따라서 하나의 평문을 고정했을 때 병렬적으로 SCADPA를 하면 9번의 암호화만으로도 한 라운드 키를 찾을 수 있다.

3.1.2 PRESENT-80 SCADPA 논리

PRESENT-80은 평문의 특정 니블에 변화를 주면 비선형연산과 비트 퍼뮤테이션을 거쳐 [그림 2]와 같이 변화가 4개의 니블로 확산된다.



[그림 2] 한 니블 변화로 인한 다음 라운드의 변화

따라서 2라운드 AddRoundKey에서의 전자파 차분을 관찰하면 ΔS_i를 추정하여 키를 복구할 수 있다.

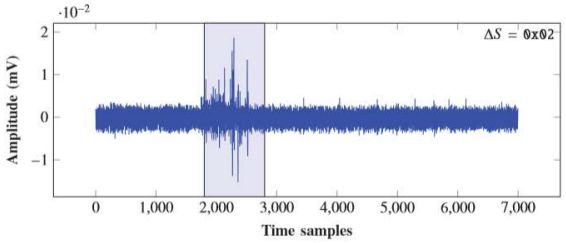
3.2 SCADPA 실험

기존 논문의 실험 환경은 [표 3]과 같다.

[표 3] 실험 환경

부채널 정보	전자파
보드	Arduino UNO 개발 보드
분석 대상 칩	Atmel, ATmega328P
전자파 수집	Langer RF-U-5-2 Probe
신호 캡처	LeCroy WaveRunner 610Zi oscilloscope
샘플링 레이트	500 MS/s
분석 위치	2라운드 AddRoundKey

평문의 하나의 니블($i = 0$)을 p_i=0x0로 고정하고 다른 평문의 니블을 p_i'=0xA로 선택한다. 선택한 두 평문에 대하여 전자파를 수집한 뒤 전자파 차분을 통해 ΔS_i를 추측한다. [그림 3]은 Δt를 나타낸 것이다.



[그림 3] p_i=0x0, p_i'=0xA에 대한 Δt

Δt를 통해 ΔS_i=0x2라고 추측하면, 아래의 식을 통해 키 후보는 0x5와 0xF이다.

$$\Delta S_i = S(p_i \oplus k_i) \oplus S(p_i' \oplus k_i),\ 0 \le i \le 15$$

또 다른 p_i'=0x3을 선택하여, 위와 같은 방법을 반복하면 ΔS_i=0x6이고, 키 후보는 0xC, 0xD, 0xE 그리고 0xF이다.

따라서 해당 니블의 키가 0xF임을 알 수 있고 위 과정을 모든 니블에 적용하여 1라운드 키 64비트를 얻어낸다.

IV. 결론 및 향후 계획

비트 퍼뮤테이션 기반의 블록 암호들은 특정 니블 값이 다른 선택 평문들에 대한 부채널 정보의 차분을 이용하면 적은 수의 암호화로 키를 추측할 수 있다. 기존 논문에서는 이 취약점을 이용해 PRESENT-80에 대한 SCADPA 실험을 통해 공격이 유효함을 보였다.

이 논리를 소비전력을 이용해 직접 재연하는 것을 시도하였지만 노이즈와 정렬의 문제로 인한 한계가 있었다. 추후 한계점에 대해 보완하여 PRESENT-80에 대한 SCADPA 재연을 할 것이다. 그리고 기존에 제안되지 않았던 다른 비트 퍼뮤테이션 기반 블록 암호에 대해 SCADPA 논리를 적용하여 실험을 진행할 예정이다.

[사사] "이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00520,SCR-Friendly 대칭키 암호 및 응용모드 개발)"

[참고문헌]

[1] Jakub Breier, Dirmanto Jap and Shivam Bhasin, "SCADPA: Side-Channel Assisted Differential-Plaintext Attack on Bit Permutation Based Ciphers", In 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2018.

[2] JA. Bogdanov, L.R Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An Ultra-LightWeight Block Cipher", CHES 2007.

[3] Jakub Breier. Dirmanto Jap, Xiaolu Hou and Shivam Bhasin, "On Side-Channel Vulnerabilities of Bit Permutations: Key Recovery and Reverse Engineering", IACR Cryptology ePrint Archive, 2018.

[4] Jakub Breier, Dirmanto Jap, Xiaolu Hou and Shivam Bhasin, "On Side Channel Vulnerabilities of BIt Permutations in Cryptographic Algorithms", IEEE Transactions on Information Forensics and Security, 2019.

Rowhammer 취약점을 이용한 공격 분석

양희동*, 이만희*
*한남대학교 컴퓨터공학과

A Survey on Rowhammer Exploit

Hee-Dong Yang*, Man-Hee Lee*
*Department of Computer Engineering, Hannam University.

요 약

현대 컴퓨팅 환경에서 중요한 역할을 하고 있는 DRAM은 제조기술의 발달함에 따라 메모리의 용량과 효율성이 증가되었다. 하지만 DRAM의 용량이 커짐에 따라 안정성에는 부정적인 영향이 드러났다. 대표적으로 하나의 메모리 Row를 반복적으로 접근할 때 인접한 메모리 Row에서 비트플립 현상이 발생하는 Rowhammer 취약점이 존재한다. Rowhammer 공격은 하드웨어 취약점을 이용한 공격으로 소프트웨어 패치로는 완벽하게 방어가 불가능하다. 이에 따라 본 논문에서는 Rawhammer 취약점에 대해 소개하고 이를 이용한 다양한 공격들을 소개한다.

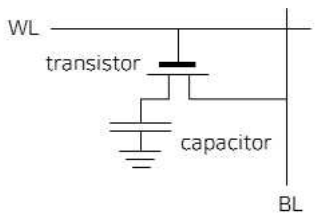
I. 서론

DRAM(Dynamic Random Access Memory)은 RAM의 한 종류로 현대 컴퓨터의 주기억장치로 컴퓨터와 모바일에서 중요한 요소이다. 최근에는 DRAM의 제조기술이 발달함에 따라 메모리의 용량과 효율성이 증가하고 있다. 메모리 셀이 작아지고 셀과 셀 사이의 간격도 줄어들어 DRAM의 용량은 커질 수 있었지만 부정적인 영향도 드러났다. 작아진 셀은 Noise Margin(잡음여유)을 감소시켜 데이터 손실에 취약하게 만들고 셀 간의 간격이 줄어들게 되면서 전자기적 결합효과를 가져오는 결과를 만들었다. 이를 이용한 Rowhammer 취약점은 하나의 메모리 Row에 반복적으로 접근하여 인접한 메모리 Row에서 비트플립(Bit-flip)을 발생시키는 공격이다. 비트플립 현상을 이용하면 같은 메모리 Bank 내의 다른 Row의 주소를 유추 및 접근할 수 있다[1].

최근에는 Rowhammer를 이용하여 커널 메모리 권한을 상승시키거나 시스템을 손상시키는 등의 다양한 공격이 등장하였다[2][3][4][5][6]. 또한, DDR3 RAM뿐만 아니라 Rowhammer 취약점을 보완했다고 알려진 DDR4 RAM에서도 여전히 취약한 것으로 밝혀졌다[7]. 본 논문에서는 Rowhammer 취약점에 대해 알아보고 Rowhammer 취약점을 이용한 다양한 공격들을 소개한다.

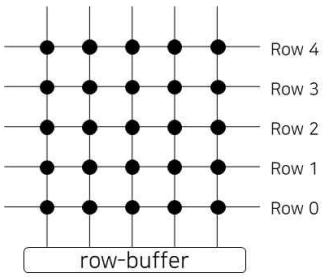
II. 배경지식

2.1 DRAM



[그림 1] 하나의 셀(Cell) 구조

DRAM의 하나의 셀(Cell)은 한 개의 커패시터와 트랜지스터 스위치로 구성되어 있으며 [그림 1]과 같이 이루어져 있다. 또한, 커패시터의 충전 유무에 따라 1bit의 값을 나타내며 각 셀은 BL(Bit Line)과 WL(Word Line)이라는 선의 교차점에 놓여 있다.



[그림 2] Row of cells

각각의 셀들은 [그림 2]와 같이 WL은 수평 방향의 row의 모든 셀에 연결되고 BL은 수직 방향의 모든 셀에 연결된다. 이때 row-buffer에 의해 연결된 이 셀 그룹을 메모리 Bank라고 한다. 메모리 Bank의 데이터에 접근하기 위해서는 먼저 대응되는 WL을 고전압으로 상승시켜 원하는 row를 활성화한다. 해당 Row의 모든 커패시터가 BL에 연결되면 데이터는 전하 형태로 row-buffer로 전송되게 되고, 이후 row-buffer의 데이터 액세스 및 수정이 가능하게 된다. 만약 같은 메모리 Bank의 다른 Row 데이터에 접근이 필요한 경우, 현재 활성화된 Row의 WL 전압을 낮추어 비활성 상태로 변경한 후 row-buffer를 Clear 시킨다. Row에 더 이상 접근이 없을 경우에는 WL을 저전압으로 낮추어 BL로부터 커패시터를 분리한다. 결과적으로 Row에 대한 모든 접근은 Row 대신 row-buffer를 통해 제공된다[8].

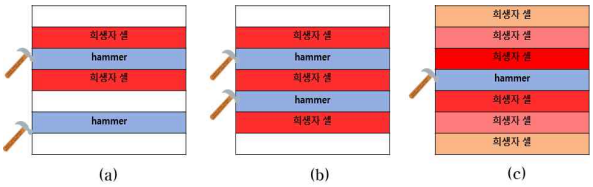
2.2 Rowhammer

Rowhammer 공격으로 알려진 메모리 교란 오류는 [그림 3]의 CodeA와 같은 어셈블리 코드를 실행시켜 발생시킬 수 있다.

```
1  CodeA:
2      mov (X),  %eax      // Read from address X
3      mov (Y),  %ebx      // Read from address Y
4      clflush (X)         // Flush cache for address X
5      clflush (Y)         // Flush cache for address Y
6      mfence
7      jmp CodeA
```

[그림 3] Rowhammer 공격을 위한 어셈블리 코드

먼저 두 개의 *mov* 명령을 통하여 DRAM에서 주소 X, Y를 읽고 설치한다. 하지만 CPU 캐시가 존재하면 메모리에 접근하지 않는다. 그러므로 CPU 캐시를 플러시하기 위해 *clflush* 명령을 수행한다. 마지막으로, *mfence*명령을 통해 다음 메모리 명령어가 실행되기 전에 데이터가 완전히 플러시 되도록 하고 계속 반복하기 위해서 *jmp* 명령어를 통해 처음으로 점프하게 만든다. 이러한 메모리 교란 오류를 이용하면 반복적으로 열리는 Row가 아닌 다른 Row에서 비트플립이 발생하여 0은 1로, 1은 0으로 비트가 뒤집히게 할 수 있다[1].



[그림 4] hammering 패턴

비트 플립을 발생시키는 Rawhammer 공격을 위한 hammering 방법으로는 Single-sided, Double-sided, One-location 패턴이 존재한다. 먼저 Single-sided Hammering 패턴은 [그림 4](a)와 같이 임의의 두 개의 메모리의 Row를 공격자 Row로 사용하고 교대로 접근하여 인접한 행에서 비트 플립을 유도하는 방법이다. 이 방법은 희생자 Row를 한쪽 면으로만 hammering하여 유연하지만 느리다. Double-sided Hammering은 [그림 4](b)와 같이 희생자 Row에 인접한 두 개의 Row를 공격자 Row로 사용하는 방법이다. 이 방법은 Single-sided Hammering 보다 효율적이지만 복잡하다. One-location Hammering은 [그림 4](c)와 같이 동일한 Row에만 hammering 하는 방법이다. close page policy와 adaptive page policy를 사용하는 시스템의 경우 row-buffer에서 캐시된 Row가 OS에 의해 자동으로 제거되므로 동일한 Row를 계속 hammering 하면 비트플립을 유발할 수 있다. 3가지의 hammering 패턴들은 비트를 뒤집는 효율에 큰 영향을 미친다[6].

III. Rowhammer 취약점 악용 공격

3.1 Drammer[3]

Drammer 공격은 공격대상이 x86 기반이었

던 이전연구들과 달리 Android 기반의 공격을 제시하였다. 기존의 연구에서는 ARM 프로세서의 성능이 좋지 않아 메모리 컨트롤러가 Rowhammer 공격을 수행하기에 적합하지 않다고 했으나 해당 연구에서는 LG Nexus 5 기기를 비롯한 ARM 프로세서를 사용하는 실제 모바일 기기에서 Double-sided Hammering 방법으로 비트플립을 발생시켰다. 이를 통해 모바일 기기에서의 루트 권한 상승을 수행시키는 공격을 수행하였다.

3.2 Nethammer[4]

Nethammer 공격은 네트워크 기반의 원격 Rowhammer 공격으로 기존의 Rowhammer 공격과 달리 공격자가 피해자의 시스템에 직접 접근하지 않아도 수행 할 수 있다. 해당 공격은 네트워크 요청을 처리하는 동안 캐시되지 않은 메모리 또는 플러시 명령을 사용하여 시스템을 공격한다. 또한, QoS(Quality of Service)기술을 악용하여 One-location 또는 Single-sided 패턴의 hammering을 수행하기 위해 네트워크 패킷들의 가공된 stream을 공격대상 디바이스에 전송한다. 공격대상 디바이스는 수신된 각 패킷에 대해 커널 드라이버 또는 사용자 응용프로그램에서 set of addresses에 접근하게 되는데 이러한 패킷을 공격자가 계속 전송함으로서 set of addresses가 hammering 되고 이에 따라 비트플립이 발생하게 된다. 이와 같은 공격을 통해 실제 시스템에서 시간당 다중 비트 플립을 유도하여 시스템에 일시적이거나 영구적인 손상을 초래할 수 있다.

3.3 GLitch[5]

GLitch는 CPU 대신 GPU를 이용한 최초의 GPU 원격 Rowhammer 공격이다. 일반적으로 Rowhammer는 CPU의 캐시되지 않은 메모리에 접근이 필요하다. 하지만 GLitch는 GPU 캐시를 이용하기 때문에 CPU의 캐싱여부와 상관없이 Rowhammer 공격을 수행 할 수 있다. 또한 캐시 Flush를 수행하지 않아도 되므로 JavaScript를 통한 원격 Rowhammer 공격이 가능하다. 또한, GLitch는 WebGL을 이용하여 브라우저에서 비트플립을 발생시킨다. WebGL은 그래픽 집약적인 애플리케이션을 위해 GPU를 가속화할 목적으로 설계된 그래픽 API를 말한다. 해당 공격은 WebGL을 이용하여 JavaScript로 Rowhammer 공격을 수행한다. 공격자는 웹브라우저로 비트 플립을 트리거할 수 있도록 만들어진 JavaScript를 통해 공격을 수행한다. 그 결과 메모리에 임의로 읽기, 쓰기가 가능해지게 되며 이를 통해 PC를 제어하고 추가적인 원격 코드를 실행할 수 있다.

3.4 RAMBleed[6]

RAMBleed 공격은 무결성을 손상시키는 기존의 Rowhammer 공격과는 달리 기밀성을 손상시켜 정보를 유출시키는 공격이다. 데이터에 접근하지 않고 데이터를 읽는 RAMBleed 공격을 위한 메모리 레이아웃은 [그림 5], [그림 6]과 같다.



[그림 5] Double-sided RAMBleed 레이아웃

[그림 5]와 같이 두 개의 4KiB 페이지가 있는 Page 레이아웃에서 A0, A1 및 A2는 공격자 데이터가 포함된 공격자 제어 Page이고 S는 피해자의 비밀이 있는 Page이며 R0은 임의의 Page를 의미한다. 공격을 위해 먼저 공격자는 A1을 하나의 비트로 채운다. 두 번째로 공격자는 자신의 페이지인 A0와 A2를 반복해서 읽으면서 A1에 대해 Double-sided Hammering을 수행한 후 공격자의 Page인 A1[i] 비트 값을 읽는다. 이때, 비트플립 현상으로 인하여 A1[i] 값은 S[i]값과 동일하다. 따라서 A1[i]를 읽으면서 S[i]값을 밝히는 방식으로 원하는 데이터를 유출시킬 수 있다. 이와 같은 방법을 Double-sided RAMBleed라고 부르며 동일한 Secret 데이터 사본 2개가 필요하므로 실제로 수행하기 어렵다. 그럼에도 불구하고 RAMBleed 연구에서는 OpenSSH 7.9 Server에

서 RSA-2048의 Key를 추출하는 공격을 수행하여 공격을 증명하였다.



[그림 6] Single-sided RAMBleed

[그림 6]은 Single-sided RAMBleed의 레이아웃으로 Double-sided RAMBleed와 동일한 단계를 수행하여 S값을 유출하는 방법이다. Single-sided RAMBleed는 하나의 Secret 데이터 사본을 공격하므로 해당 레이아웃의 A1[i]값이 S[i]값과 다를 수 있으므로 정확도는 감소되지만 절반 이상의 S값을 유출 할 수 있다.

또한, RAMBleed 공격은 비트플립을 유발하였을 때 되돌려주는 Rawhammer 방어기술인 ECC 보호기능을 사용한 최신 RAM에서도 공격 수행이 가능하다.

IV. 결론 및 향후연구

본 논문에서는 DRAM의 하드웨어 취약점인 Rawhammer 취약점과 이를 이용한 다양한 공격들을 살펴보았다. Rawhammer 취약점을 막기 위하여 GuardION, ANVIL와 같은 여러 방어 기법이 제시되었으나 새로운 공격들이 계속 등장하므로 보안 연구가 필요하다. 향후연구로는 Intel SGX와 TrustZone 기술을 활용하여 메모리에서 비트플립 현상을 악용한 내용 유출 및 원격 코드 수행을 방어하는 연구를 수행하고자 한다.

[참고문헌]

[1] Y. Kim, R. Daly, J. Kim, C. Fallin, J. Lee, D Lee, C. Wilkerson, K. Lai and O. Mutlu, "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors", ACM SIGARCH Computer Architecture News. Vol. 42. No. 3. IEEE Press, 2014.

[2] M. Seaborn and T. Dullien, "Exploiting the DRAM rowhammer bug to gain kernel privileges", http://googleprojectzero.blogspot.com/2015/03/exploiting-dram-rowhammer-bug-to-gain.html, March, 2015.

[3] V. Van der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi and C. Giuffrida, "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms", 2016 ACM SIGSAC conference on computer and communications security, 2016.

[4] M. Lipp, M. T. Aga, M. Schwarz, D. Gruss, C. Maurice, L. Raab, L. Lamster ,"Nethammer: Inducing rowhammer faults through network requests", 2018.

[5] P. Frigo, C. Giuffrida, H bos and K. Razavi, "Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU", 2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018

[6] A. Kwong, D. Genkin, D. Gruss and Y. Yarom, "RAMBleed: Reading Bits in Memory Without Accessing Them", the 41st Annual IEEE Symposium on Security & Privacy, May, 2020

[7] M. Lanteigne, "How Rowhammer Could Be Used to Exploit Weaknesses in Computer Hardware," http://www.thirdio.com/rowhammer.pdf, March, 2016

[8] D. Lee, Y. Kim, V. Seshadri, J. Liu, L Subramanian and O. Mutlu, "Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture", IEEE 19th International Symposium on High Performance Computer Architecture (HPCA), 2013.

안드로이드 스마트폰 전자파 오류 주입 공격 자동화 시스템 제안

이종혁*, 임한섭*, 한동국*,**

*국민대학교 금융정보보안학과, **국민대학교 정보보안암호수학과

Suggestion of Automation System for Electromagnetic Fault Injection Attack on Android Smartphones

JongHyeok Lee*, Han Sup Lim*, Dong-Guk Han*,**

*Dept. of Financial Information Security, Kookmin University,
**Dept. of Information Security, Cryptology, and Mathematics, Kookmin University

요 약

최근 고성능의 스마트폰에서 삼성 덱스(Dex)와 같은 기능을 이용하여 모바일 뱅킹 이외의 다양한 업무를 수행할 수 있어졌다. 스마트폰을 데스크톱과 같이 사용할 수 있어짐에 따라 보안을 요구하는 작업들이 스마트폰에서 수행되어지고 있다. 이에 따라 스마트폰 대상 부채널 공격에 대한 안전성이 요구되고 있다.

본 논문에서는 안드로이드 스마트폰을 대상으로 전자파 오류 주입 공격 실험 환경을 구성하고 이를 자동화하는 방안을 제시한다. 또한, 스마트폰 상에서 RSA-CRT 연산을 수행하는 동안에 전자파 오류 주입 공격을 수행해 잘못된 서명 값을 획득하는 실험 결과를 소개한다.

I. 서론

부채널 분석(Side Channel Attack, SCA)은 암호화/복호화 연산이 실제 장비에서 수행되는 중에 발생되는 부채널 정보를 이용하여 비밀 정보를 찾아내는 분석 방법이다. 주로 사용되는 부채널 정보로는 연산 시간, 소비 전력, 전자파 방출, 소리 등이 있다.

부채널 분석은 외부로부터의 인위적인 변경 여부를 기준으로 능동적 공격(Active Attack)과 수동적 공격(Passive Attack)으로 나뉜다. 수동적 공격은 부채널 정보에 대한 측정만으로 비밀 정보를 분석하는 시간 분석(Timing Analysis)과 전력 분석(Power Analysis) 등이 있다[3, 4]. 능동적 공격의 대표적인 예는 공격 대상에 직접적인 물리적 조작을 가하여 비밀 정보를 분석하는 오류 주입 공격(Fault Injection Attack)이 대표적이다.

오류 주입 공격은 주입하는 오류원에 따라 클락 글리치 공격(Clock Glitch Attack), 레이저 오류 주입 공격(Laser Fault Injection Attack), 전자파 오류 주입 공격(Electromagnetic Fault Injection Attack) 등이 있다[2, 6].

최근 삼성 덱스(DeX)와 같은 기능을 통해 고사양의 스마트폰을 데스크톱처럼 업무에 활용할 수 있어졌다. 이제는 모바일 뱅킹 등의 간단한 업무에 그치지 않고 다양한 업무를 스마트폰에서 수행함에 따라 스마트폰을 대상으로 하는 부채널 공격을 고려해야 한다.

본 논문에서는 안드로이드 스마트폰을 대상으로 전자파 오류 주입 공격을 자동화하는 방안을 제시한다. 또한 스마트폰에서 동작하는 RSA-CRT(Chinese Remainder Theorem)를 대상으로 오류가 주입된 서명 값을 획득하는 실험을 수행하고 그 결과를 소개한다.

II. 전자파 오류 주입 공격

클락 글리치 공격은 인위적인 클락을 주입하기 위한 보드의 조작이 요구되고 레이저 오류

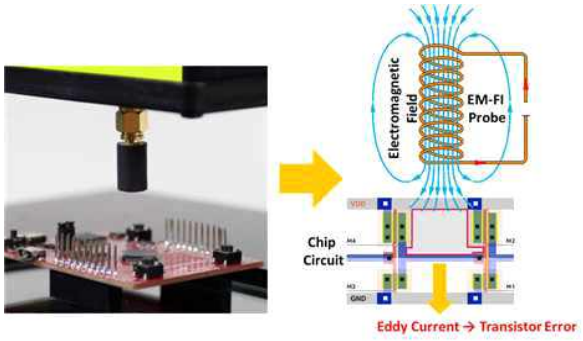


Fig. 1. Principle of an electromagnetic fault injection attack

주입 공격은 공격 대상 칩의 디캡슐레이션(Decapsulation)이 요구된다. 반면에 전자파 오류 주입 공격은 위와 같은 물리적인 조작이 필요 없음에도 불구하고 로직 타이밍 위반(Logic Timing Violation)이나 트랜지스터(Transistor)의 오동작을 일으킬 수 있다.

Fig. 1은 전자파 오류 주입 공격의 원리를 보여준다. 전자파 오류 주입 공격은 전자파 프로브 팁(Electromagnetic Probe Tip)에 순간적인 전류를 인가하여 전자기 유도 현상을 발생시킨다. 프로브 팁은 페라이트(Ferrite)에 코일(Coil)을 감은 형태로 코일에 순간적인 전류가 흐르면 코일 주변에 전자기장이 발생하고, 자속(Magnetic Flux)에 의해 생성된 칩 회로 내부의 와전류(Eddy Current)가 트랜지스터에 영향을 끼친다. 이러한 과정으로 인해 칩의 오동작이 유발된다.

III. 실험 환경

본 실험은 LG L15G Sunrise 스마트폰[5]을 대상으로 하고 있다. 1.2GHz의 동작 주파수를 갖는 Qualcomm Snapdragon 200 MSM8210 프로세서와 SKhynix H9TP32A4GDDC eMMC를 사용하며 Android 4.4 KitKat 플랫폼이 탑재되어 있다. LG L15G Sunrise 스마트폰은 프로세서 위에 금속 실드가 부착되어 있으며 그 위에는 베터리가 위치해 있다. 따라서 전자파 오류 주입 공격을 수행하기 위해 베터리와 금속 실드를 제거해야 한다. 위 작업을 수행하였을 때



Fig. 2. LG L15G Sunrise smartphone

의 스마트폰 모습은 Fig. 2와 같다.

실험 환경 구성은 Fig. 3과 같다. 전자파 오류 주입 장비로 Riscure 사의 Spider와 EM-FI Transident Probe를 사용하였으며 PC와 스마트폰 간의 원활한 통신과 트리거 생성을 위해 Arduino MEGA Adk 보드[1]를 사용하였다. 스마트폰에는 RSA-CRT 동작을 하도록 앱을 구성하였다.

통신 프로토콜은 다음과 같다.

① PC에서 Arduino MEGA Adk로 'S' 전송
② Arduino MEGA Adk에서 PC로 'start arduino' 전송, LG L15G Sunrise로 'S' 전송
③ LG L15G Sunrise에서 Arduino MEGA Adk로 'start phone' 전송, RSA-CRT 연산 수행
④ Arduino MEGA Adk에서 PC로 'start phone' 전송, 트리거 발생
⑤ LG L15G Sunrise에서 Arduino MEGA Adk로 서명 값 S 전송
⑥ Arduino MEGA Adk에서 PC로 서명 값 S 전송

Arduino MEGA Adk가 트리거를 생성하면 일정 시간 이후에 Spider가 EM-FI Transident

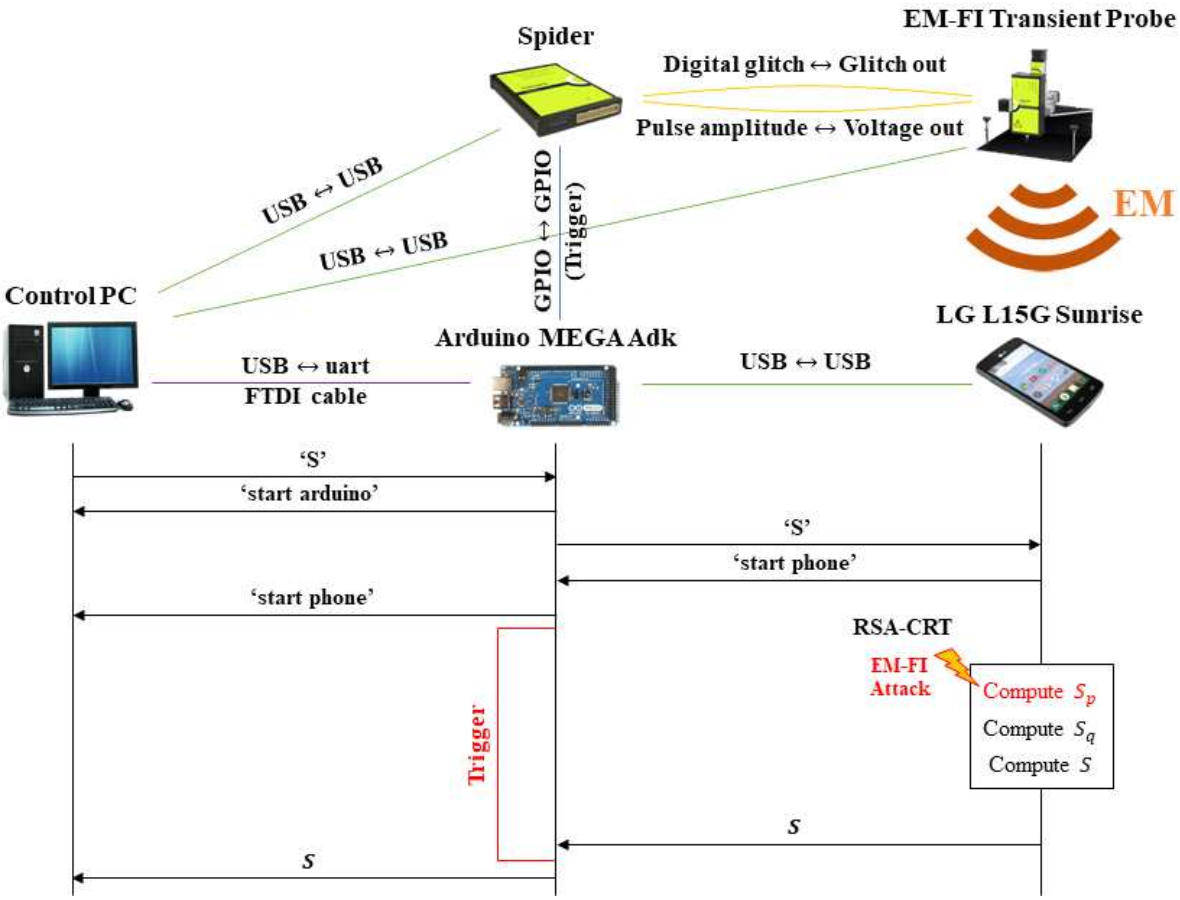


Fig. 3. Experiment diagram and protocol

Probe를 통해 전자파 오류를 LG L15G Sunrise 스마트폰의 프로세서 상에 주입한다. 딜레이는 트리거 발생으로부터 60-110ms로 설정하여 S_p 계산중에 오류를 주입해 잘못된 서명 값 이 나오도록 유도하였다.

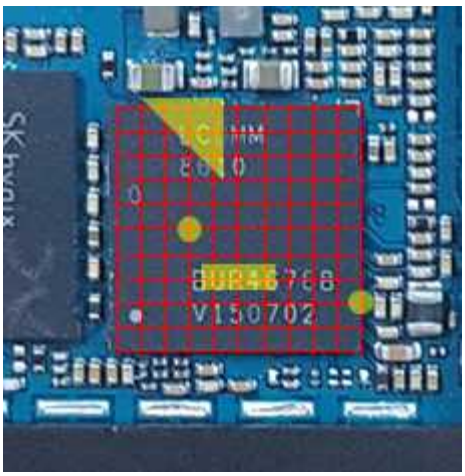


Fig. 4. Experiment result on Qualcomm Snapdragon 200 MSM 8210 processor

IV. 실험 결과

프로세서를 가로 11등분, 세로 11등분하여 총 프로세서 표면의 121개의 위치에 전자파 오류를 주입하였다. 그 결과, Fig. 4의 노란색으로 표시한 영역에서는 스마트폰이 무한 재부팅을 하거나 화면이 멈추는 현상을 확인할 수 있었다. 그 외의 영역에서는 Fig. 5와 같은 결과를 얻을 수 있었다. 통신 프로토콜의 ② 단계와 ④ 단계에서 PC가 수신한 'start arduino'와 'start phone'을 수신한 서명 값과 같이 결과 창에 출력하였다. 전자파 오류 주입 공격 결과는 다음과 같이 분류하였다.

- 초록색: 'start arduino' 정상 수신, 'start phone' 정상 수신, 옳은 서명 값 수신

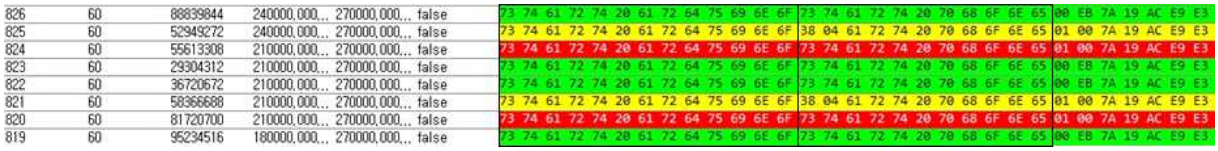


Fig. 5. Experiment

- 노란색: ‘start arduino’ 정상 수신, ‘start phone’ 비정상 수신, 옳은/잘못된 서명 값 수신
- 빨간색: ‘start arduino’ 정상 수신, ‘start phone’ 정상 수신, 잘못된 서명 값 수신

노란색은 ‘start phone’이 정상적으로 수신되지 않아 스마트폰과의 통신 상 오류가 발생하거나 스마트폰이 재부팅되는 등의 오동작이 발생하였다고 판단한 결과이다. 빨간색은 통신 프로토콜이 정상적이고 서명 값이 잘못되어 주입된 오류가 잘못된 서명 값을 계산하도록 하였다고 판단한 결과이다.

실험 결과로 얻은 잘못된 서명 값을 사용하여 비밀 키 p또는 q를 복구하는 과정을 수행하였을 때, 비밀 키를 획득할 수 없었다. 따라서 주입된 오류가 S_p 혹은 S_q를 생성하는 과정 외에 다른 동작 중에 영향을 끼친 것으로 판단된다.

V. 결론

본 논문은 안드로이드 스마트폰을 대상으로 전자파 오류 주입 공격을 자동화하는 방안을 제시하였다. 또한 스마트폰에서 동작하는 RSA-CRT를 대상으로 전자파 오류 주입 공격을 수행하여 잘못된 서명 값을 유도하는 실험을 진행하였다. 획득한 잘못된 서명으로부터 비밀 값 p 또는 q를 복구해낼 수는 없었지만 전자파 오류 주입을 통해 스마트폰의 오동작을 유도해낼 수 있었다.

따라서 전자파 오류 주입을 통해 스마트폰에 오동작을 유도할 수 있음을 확인하였다. 이를 기반으로 스마트폰을 대상으로 암호 알고리즘의 비밀 키를 획득하는 연구와 인증 알고리즘을 우회하는 연구 등을 진행할 예정이다.

[사사] 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

[참고문헌]

[1] Arduino, Arduino Mega ADK Rev3, https://store.arduino.cc/usa/arduino-mega-adk-rev3.

[2] Balasch, J., Gierlichs, B., & Verbauwhede, I. (2011). An In-depth and Black-box Characterization of the Effects of Clock Glitches on 8-bit MCUs. In 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography (pp. 105-114). IEEE.

[3] Kocher, P. C. (1996). Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Annual International Cryptology Conference (pp. 104-113). Springer, Berlin, Heidelberg.

[5] Kocher, P., Jaffe, J., & Jun, B. (1999). Differential power analysis. In Annual International Cryptology Conference (pp. 388-397). Springer, Berlin, Heidelberg.

[6] LG USA, LG Sunrise (GSM) TracFone Smartphone (L15G), https://www.lg.com/us/cell-phones/lg-L15G-sunrise.

[7] Quisquater, J. J. (2002). Eddy current for magnetic analysis with active sensor. Proceedings of Esmart, 2002, 185-194.

GPU 병렬 환경에서의 AES 암호 최적화 연구

안상우, 서석충*

국민대학교 정보보안암호수학과

Study on Optimization of AES in a Parallel GPU Environment

Sang-Woo An, SeogChung Seo

Department of Information Security, Cryptology, and Mathematics, Kookmin University.

요 약

본 논문에서는 암호의 중요성이 증대되면서 암호 알고리즘에 대한 최적화 연구가 요구됨에 따라 대표적인 국제 표준 블록 암호 알고리즘인 AES의 다양한 최적화 구현 방법을 설명하고 T-table, Bitslicing의 두 가지 관점을 통해 GPU 환경에서 최적화할 수 있는 방안을 연구한다. 이러한 최적화 기법의 적용 결과, AMD RYZEN 5 2600 CPU, NVIDIA GTX 1060 3GB GPU 상에서 64비트 운영체제를 이용할 경우 T-table 최적화 구현은 34.95Gbps, Bitslicing 최적화 구현은 27.67Gbps의 성능 측정 결과를 나타냄으로써 최적화가 적용되지 않은 기존 구현보다 최대 1690배 빠른 성능을 확인할 수 있었다.

I. 서론

4차 산업이 중심이 된 정보화 사회의 발달로 인해 정보의 가치가 증가함에 따라 보안 침해 사고의 위험성도 덩달아 커지고 있다. 특히 4차 산업의 대표주자인 IoT(Internet of Things) 환경에서 무선 통신을 하는 기기들은 도청하기가 쉬우므로 암호화가 필수적으로 요구되고 있다. 그뿐만 아니라, 클라우드 환경에서도 서버는 대량의 사용자 서비스에 대해서 개인정보의 유출 위협을 방지하기 위해 암호화가 필수 불가결하다. 이로 인해 암호화 알고리즘에 관한 연구가 활발해짐에 따라, 암호화 알고리즘에 대한 최적화 연구 또한 필요성이 증대되었다. 한편에는, 통신 기술의 발달로 통신 속도가 매우 빨라져 암호에서도 고속의 연산이 필요해졌다.

수차례 최적화가 시행된 대표적인 암호화 알고리즘으로는 국제 표준 블록 암호 알고리즘인 AES가 있다[1]. 기존에는 CPU 환경에서 8비트 기준으로 구현된 AES를 32비트 버전으로 최적화하는 방법을 연구하였다[2]. 최근에는 SIMD를 이용하여 AES를 병렬 암호화하는 최적화 구현도 있었고, CPU의 AES-NI를 이용한 최적화 연구도 이루어졌다. 또한, 클라우드 환경을 고려해서 GPU를 이용한 최적화 연구도 다수 수행되었다.

GPU를 이용한 AES 알고리즘 최적화 연구는 크게 두 가지로 나뉜다. 첫 번째는 T-table을 이용한 구현 방법이며, 두 번째는 Bitslicing 기법을 이용한 방법이다. 최근, S-box를 참조하는 AES에 대한 cache timing 공격이 가능한 것이 알려짐에 따라, 본 논문에서는 cache timing 공격에 면역이 가능한 Bitslicing을 이용한 최적화 기법에 주안점을 둔다.

본 논문에서는 두 가지 관점에서의 최적화 구현을 수행하고, 각 구현 방법을 GPU에서 최

* 이 논문은 2019년도 정부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2019R1F1A1058494)

적화할 때 고려해야 할 사항들을 제시한다. 최종적으로 수행된 최적화 결과를 제공한다.

II. 관련 연구

2.1. T-table 최적화 기법

AES 블록 암호 알고리즘은 SPN 구조를 바탕으로 하고 있으며, 128비트의 블록 길이와 128/196/256비트의 키 길이를 가진다. 라운드 수는 키 길이에 따라 10/12/14로 구성되며, 각 라운드는 SubBytes, ShiftRows, MixColumns, AddRoundKey의 변환으로 구성된다.

SubBytes 변환은 128비트 블록을 4×4바이트의 배열로 만든 State에 대하여 각각의 바이트에 대해 서로 독립적인 비선형 치환을 수행한다. 이것은 유한체 GF(2^8)상에서 곱셈의 역원을 취하는 과정과 affine 변환으로 구성된다. 이 계산 결과를 모든 Input에서 나올 수 있는 모든 Output에 대하여 Lookup Table로 정리한 것을 S-box라고 한다[1].

ShiftRows 변환은 State를 구성하는 각 행을 행의 위치에 따라 왼쪽으로 순환 이동(cyclic shifting)시킨다.

MixColumns 변환은 State의 열을 유한체 GF(2^8) 상의 다항식으로 생각하여 다항식의 곱셈을 연산한다.

기존의 AES를 최적화하기 위하여 라운드 내에서 8비트를 기준으로 수행되는 SubBytes, ShiftRows, MixColumns을 하나로 통합할 수 있다. 상기의 연산 결과를 미리 계산하여 네 개의 Lookup Table로 구성할 수 있으며 이 테이블을 T-table이라고 한다. T-table 참조는 32비트를 단위로 수행되며 기존에 참조를 수행했던 S-box 대신 T-table을 사용하여 암호화를 진행하게 된다[3].

2.2. Bitslicing 최적화 기법

AES Bitslicing 기법은 AES 암호화 알고리즘을 기존의 바이트 단위 연산을 통해 수행하는 것이 아닌 비트 단위에서의 XOR, AND, OR을 사용하여 논리 비트 연산을 수행하는 기법을 의미한다. Bitslicing의 가장 큰 장점은 Bitslicing을 이용하여 각 비트를 병렬로 암호화 과정을 수행함으로써 처리 속도를 크게 향상할 수 있다는 것이다[4].

Bitslicing의 또 다른 장점은 cache timing 공격에 면역이 된다는 것이다. 테이블을 사용하는 기존 AES에서 메모리 액세스 패턴은 암호 분석에 취약하지만, Bitslicing 구현은 논리 연산에만 기초하고 관련 테이블을 가지지 않으므로 cache timing 공격에 면역이 된다.

AES Bitslicing의 과정은 기존의 AES 암호화 과정과 크게 다르지 않지만, 병렬 연산을 수행하기 위해 비트를 재배치하는 Transpose 변환을 추가로 거치게 된다. 이 변환을 통해 블록에 담기는 데이터는 N개의 Input에 대하여 각 Input의 동일한 위치의 비트를 담게 되고 각자의 암호화 과정을 N-병렬로 동시 처리가 가능해진다.

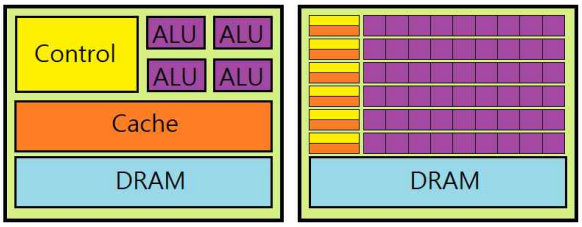
III. 실험 설계

본 장에서는 GPU 환경에서의 AES 최적화 구현에 대해 다룬다. NVIDIA의 CUDA 라이브러리를 이용하여 앞 장에서 다뤘던 T-table 최적화 구현과 Bitslicing 구현을 실험하였다.

3.1. GPU 구조

GPU는 다수의 스트리밍 멀티프로세서(Streaming Multiprocessor, SM)로 이루어져 있다. 하나의 멀티프로세서에서는 수십~수백 개 이상의 코어가 동시에 연산을 수행할 수 있다. CPU와 GPU 아키텍처의 차이점은 [그림 1]에서 확인할 수 있다.

호스트 PC에서 호출되는 CUDA GPU의 서브 프로그램을 커널(kernel)이라고 부르며, 커널은 블록으로 구성된 그리드이다. 각 블록은 여러 개의 스레드로 구성되어 스트리밍 멀티프로세서에 할당된다. 각 스레드는 자신에게 할당된 레지스터에 접근하여 각자 연산을 수행한다[5].



[그림 1] CPU와 GPU 아키텍처의 차이점[5]

3.2. GPU T-table 구현

GPU 상의 T-table 최적화 구현은 기존 CPU로 구현된 AES T-table 최적화 구현 코드를 CUDA 라이브러리를 이용하여 GPU 프로그램으로 포팅하는 방식으로 구현하였다. 암호화 연산을 수행하면서 자주 접근하는 T-table을 빠른 접근이 가능한 공유 메모리에 복사하였다. 테이블을 공유 메모리에 복사할 때는 256개 이상의 스레드를 선언하여 각 스레드가 테이블 상의 한 개의 데이터를 복사하도록 수행하였다.

3.3. CUDA Bitslicing 구현

GPU 상의 Bitslicing 최적화 구현은 [6]에서 제시된 알고리즘을 따라 수행하였다. 암호화 알고리즘을 수행하기 전 Transpose 변환을 통해 블록을 재배열하였다[4]. Transpose 알고리즘은 아래의 [그림 2]에서 확인할 수 있다.

SubBytes는 Bitslicing AES의 성능에 영향을 미치는 중요한 요인이다. Transpose 변환을 거친 블록을 병렬적으로 비트 단위의 논리 연산을 통해 계산되도록 구현하였다. 실험에 사용된 Bitslicing SubBytes 함수의 아이디어는 [7]에서 확인할 수 있다.

Bitslicing에서 MixColumns는 워드를 이동시켜서 다항식의 곱셈, 즉 시프트 연산을 수행한다. 만약 가장 상위 비트가 1이라면, 오버플로가 발생하기 때문에 상수 0x1b의 각 비트에 해당하는 자리인 0, 1, 3, 4 워드에 오버플로 비트를 추가하도록 한다.

Transpose function

Input : Bundle B of dimension n × n
Output : Transposed Bundle B

1 $m_0 \leftarrow (5555555555555555)_{16}$
2 $m_1 \leftarrow (3333333333333333)_{16}$
3 $m_2 \leftarrow (0F0F0F0F0F0F0F0F)_{16}$
4 …
5 $m_j \leftarrow \left(\{0\}^{2^j}\{1\}^{2^j}\right)_2$
6 for $j = 0$ *to* $\log_2 n$ do
7 k ← 1<< j
8 k2 ← k * 2
9 r ← k - 1
10 for $i = 0$ *to* $n/2$ do
11 $l \leftarrow 2(i - (i \wedge r)) + (i \wedge r)$
12 $temp \leftarrow (B_l \wedge m_j) \oplus ((B_{l+k2} \wedge m_j) << k)$
13 $B_{l+k2} \leftarrow (B_{l+k2} \wedge (\overline{m_j})) \oplus ((B_l \wedge (\overline{m_j})) >> k)$
14 $B_l \leftarrow temp$
15 end
16 end

Notations	Meaning
m_j	64비트 상수
$A \ll i$	A를 i비트만큼 왼쪽으로 시프트
$A \gg i$	A를 i비트만큼 오른쪽으로 시프트
$A * B$	곱하기
$A \wedge B$	AND 비트연산
$A \oplus B$	XOR 비트연산
$\bar{A}$	보수

[그림 2] Transpose 알고리즘[4]

IV. 실험 결과

실험에 사용된 환경과 GPU 사양은 [그림 3], [그림 4]과 같다.

CPU	AMD RYZEN 5 2600 3.4GHz
OS	Windows 64-bit
Memory	DDR4 32GB
GPU	NVIDIA GeForce GTX 1060 3GB
CUDA version	10.1

[그림 3] 실험 환경

CUDA Cores	1280
Graphics Clock (MHz)	1506
Processor Clock (MHz)	1708
Memory Clock (MHz)	8008
Memory Interface	GDDR5, 192bit

[그림 4] NVIDIA GeForce GTX 1060 특성

수행한 결과는 키 스케줄링을 포함하지 않는 암호화 과정에 대한 속도이다. AES 암호화 운용 방식은 ECB 모드로 설정하여 진행하였으며, AES의 키 길이는 128비트를 기준으로 측정하였다.

CPU 환경에서의 실험 결과는 [표 1]과 같다. 8비트 기반에서 SubBytes, ShiftRows, MixColumns를 모두 거치게 되는 기존 구현보다 32비트에서 일련의 과정을 T-table로 통합해서 수행하는 최적화 구현의 성능이 약 40배로 월등하게 좋아진 것을 확인할 수 있다.

구현 방법	Throughput
8비트 기반 구현	21.18Mbps
32비트 T-table 기반 구현	854.06Mbps

[표 1] CPU 환경에서의 실험 결과

GPU 환경에서의 실험 결과는 [표 2]와 같다. GPU 환경에서는 T-table을 이용한 구현이 CPU에서의 32비트 T-table 기반 구현보다 약 41배의 성능 우위을 보여주는 것을 확인할 수 있으며, 같은 GPU 상에서 Bitslicing 기법을 통해 최적화한 구현의 경우보다도 약 26%의 향상된 성능을 얻을 수 있었다.

구현 방법	Throughput
32비트 T-table 기반 구현	34.95Gbps
Bitslicing 기반 구현	27.67Gbps

[표 2] GPU 환경에서의 실험 결과

V. 결론 및 향후 연구

본 논문에서는 T-table과 Bitslicing 최적화 기법을 사용하여 CPU와 GPU 환경에서 각각 AES 암호화 알고리즘을 최적화 구현하였다. T-table 최적화 방법을 GPU에서 구현한 결과 기존 구현보다 최대 1690배의 성능을 얻을 수 있었으며, GPU에서 Bitslicing 기법으로 최적화 구현한 결과 기존 구현보다 최대 1338배의 성능을 얻을 수 있었고, GPU에서 T-table을 구현한 결과는 CPU에서의 구현보다 약 41배 향상된 성능을 확인할 수 있었다. 본 연구 결과는 IoT, 클라우드 컴퓨팅 등에서 사용자 데이터를 고속으로 암호화 처리하는 데 활용할 수 있다.

본 논문에서는 AMD RYZEN 5 2600 CPU와 NVIDIA GeForce GTX 1060 3GB GPU 환경에서만 성능 측정을 수행하였지만, 추후 다른 환경에서의 성능 측정을 통하여 상관관계를 연구할 계획이다. 또한, 본 논문에서는 AES 암호화 알고리즘에서의 최적화 구현만을 다루었지만, LEA, CHARM 등과 같은 국내 암호화 알고리즘에 대하여 Bitslicing 기법을 이용한 최적화 구현을 GPU 상에서 연구할 예정이다.

[참고문헌]

[1] NIST: Specification for the ADVANCED ENCRYPTION STANDARD (AES). Technical Report FIPS PUB 197, National Institute of Standards and Technology (NIST) (2001)

[2] EnSilica. A study of AES and its efficient implementation on eSi-RISC, (2010), http://www.ensilica.com/ip/esi-crypto/aes/

[3] Dr. Brian Gladman, A specification for Rijndael, the AES Algorithm, v3.6, (2003), https://www.gladman.me.uk/

[4] C. Rebeiro, D. Selvakumar, A.S.L. Devi, Bitslice Impementation of AES, CANS 2006, pp 203-212, International Conference on Cryptology and Network Security, (CANS), (2006)

[5] “CUDA C Programming Guide,” Cuda Toolkit v.10.1, NVIDIA, (2019)

[6] O. Hajihassani, S.K. Monfared, S.H. Kha-

steh, S. Gorgin, Fast AES Implementation : A High-Throughput Bitsliced Approach, IEEE Transactions on parallel and distrub uted systems, vol..30, No.10, (2019)

[7] D. Canright, A Very Compact S-box for AES, CHES 2005, pp 441-155, Cryptograp hic Hardware and Embedded Systems (CHES), (2005)

소수 특성을 활용한 RLWE 최적화 구현[1)]

이주엽*, 김수리*, 홍석희*

*고려대학교 정보보호학과

Optimized implementation of RLWE using Characteristic of Prime

Juyeop Lee*, Suhri Kim*, Seokhie Hong*,

*Graduate School of Information Security, Korea University

요 약

본 논문에서는 NIST(National Institute of Standards and Technology)의 PQC 표준화 공모전 Round 2 후보인 LAC의 polynomial ring을 변경하여 효율적으로 구현하는 방법에 대하여 논의한다. 기존의 LAC는 소수 251을 사용함에 따라 NTT를 적용하지 못하여 다항식 곱셈에서 많은 연산이 요구된다[1]. 본 논문에서는 소수를 변경하여 LAC의 연산 중 가장 큰 부분을 차지하는 다항식 곱셈을 4차식 모듈러스 NTT를 사용하여 효율적으로 처리하는 방법을 제안한다. 제안하는 소수를 사용한 결과 다항식 곱 연산에서 기존 대비 약 5배의 성능 향상을 얻을 수 있었으며, 추가적으로 소수의 형태에 따른 효율적인 reduction 방법을 적용하면 약 12배의 성능 향상을 얻을 수 있다. 본 논문에서 제안하는 방법을 사용한 LAC의 키 생성 과정은 기존 대비 약 12.4배의 성능 향상을 이룰 수 있었다.

I. 서론

Lattice 기반 암호 시스템은 NIST에서 진행 중인 PQC Standardization Round 2에 올라온 26개의 스킴중 12개를 차지하는 가장 주목받고 있는 암호 시스템이다. Lattice 기반 암호는 그 변형이 다양하여 LWE, RLWE, NTRU등 여러 가지 기반 문제들이 존재한다. 그중 RLWE는 RSA나 ECDH보다 빠른 속도를 가지고 있어, 현재 사용 중인 공개키 암호 시스템의 새로운 대안으로 떠오르고 있다.

RLWE는 LWE(Learning With Errors)의 큰 키 사이즈와 느린 연산 속도를 해결하기 위해서 Lyubashevsky 등에 의하여 제안되었다[2]. 해당 알고리즘은 LWE 기반 암호를 ring $R_q = \mathbb{Z}_q/(x^n+1)$ 위에서 정의한 것이다. 기존의 LWE 기반 암호는 행렬을 공개키로 사용하기 때문에 연산 역시 행렬-벡터 곱 연산이 필요하다. 반면, RLWE는 다항식을 공개키로 사용하여 연산이 다항식 곱셈으로 처리 가능하며 정의되는 소수와 차원에 따라 다항식 곱에 최적화된 연산 방법인 NTT를 적용하여 빠르게 처리할 수 있다.

한편, NIST Round 2의 후보 중 LAC는 RLWE의 안전성이 error rate에 의하여 결정되는 것을 이용함으로써, q의 크기를 줄여 error rate를 증가시켰다. LAC는 소수 $q=251$을 사용하여 Lattice 기반 암호의 가장 큰 단점인 키 사이즈를 줄일 수 있었지만, 높아진 error rate에 따라 복호화 실패 확률 역시 높아지게 되어 이를 해결하기 위해 오류 정정 코드를 적용했다. 하지만, 해당 소수의 사용은 다항식 곱에 적용할 수 있는 NTT를 적용하지 못하게 되어 RLWE의 장점인 빠른 연산 속도를 잃게 된다. 이는 암, 복호화 속도가 중요한 공개키 암호 분야에서 큰 단점으로 작용한다.

본 논문에서는 LAC에 사용되는 소수를 $q=257$로 사용하는 방법과 제안하는 소수를 사용하여 NTT를 이용한 효율적인 다항식 곱 연산을 구현하는 방안에 대하여 제시한다. 또한, $257=2^8+1$인 특성을 활용하여 reduction을 효율적으로 구성하는 방법을 제시한다. 본 논문에서 제안하는 4차식 모듈러스 NTT를 사용할 경우 기존 대비 약 5배, reduction 방법을 함께 적용한 결과 약 12.4배의 성능 향상을 얻을 수 있었다. 이는 q의 증가에 따라 암호문의 크기가 커지는 것을 감수할 수 있는 효율성이다.

1) 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

II. 배경 지식

2.1. RLWE

2010년 제안된 RLWE는 LWE보다 작은 키 사이즈와 빠른 연산 속도로 인하여 Lattice 기반 암호 시스템 중 가장 많이 이용되는 방식이다. RLWE 기반 암호는 RLWE problem의 어려움에 기반하고 있는데, 이는 LWE problem을 ring 위에서 정의한 것이다. LWE problem은 비밀키 s에 대하여 $b=as+e$로 이루어진 공개키 (a,b)를 알고 있을 때, 작은 오류 e에 의하여 비밀키 s를 알기 어렵다는 것에 어려움을 둔다. 기본적으로 RLWE의 암호 강도는 정의된 ring의 차수와 식(1)의 error rate로 인하여 결정된다.

$$\alpha = \frac{\sigma}{q}\sqrt{2\pi}\,,\sigma : \text{분포의 표준편차} \qquad (1)$$

위 식에서 알 수 있듯이 소수와 표준편차의 비에 따라 error rate가 결정된다. 즉, DSA에서 안전한 소수를 사용해야 하는 것과 달리 RLWE 기반 암호에서는 소수의 크기만 안전성에 영향을 미친다. 여러 RLWE 기반 스킴들은 효율성을 위하여 14-bit 소수 12289와 13-bit 소수 7681을 주로 사용하면서 표준편차를 조절하여 적당한 error rate를 설정하여 암호 강도를 결정한다.

2.2. NTT

NTT는 다항식 곱셈을 효율적으로 처리하는 Fast Fourier Transform(FFT)을 유한체 $\mathbb{Z}_q$에서 정의한 것이다. FFT는 $n-1$차 다항식은 n개의 점에서의 함숫값이 정해지면 유일하게 결정된다는 Lagrange's interpolation을 이용한 것으로, $n-1$차 다항식 f는 primitive $n-th$ root of unity ω에 대하여 $\omega^0,\cdots,\omega^{n-1}$에서 함숫값 $f(\omega^0),\cdots,f(\omega^{n-1})$으로 유일하게 결정된다. 이처럼 $n-1$차 다항식이 주어졌을 때, n개의 함숫값을 구하여 다항식을 표현하는 과정을 FFT라고 한다. FFT를 이용한 다항식 곱셈은 두 다항식 각각에 대하여 FFT를 적용하여 얻은 함숫값을 point-wise multiplication을 통해 새로운 다항식을 만드는 과정으로 정의된다.

Lattice 기반 암호에서 이용되는 NTT는 $\tau\in\mathbb{Z}$를 primitive 4-th root of unity라 할 때, 다음과 같은 성질을 이용한다.

$$\mathbb{Z}_q[x]/(x^{2^k}+1) \qquad (2)$$
$$\cong \mathbb{Z}_q[x]/(x^{2^{k-1}}-\tau)\times\mathbb{Z}_q[x]/(x^{2^{k-1}}+\tau)$$

위와 같은 isomorphism을 반복하면 다음의 최종 단계에 도달한다.

$$\mathbb{Z}_q[x]/(x^{2^k}+1)\cong\prod_{i=1}^{2^k}\mathbb{Z}_q[x]/(x-\tau_i)$$
$$,\tau_i : \text{primitive } 2n-th \text{ root of unity} \qquad (3)$$

위의 isomorphic ring을 이용하면 기존의 ring 위에서의 다항식 곱셈은 point-wise 곱셈으로 처리할 수 있어, 총 $O(nlogn)$의 복잡도를 가진다.

2.2.1. Forward NTT

일반적으로 $n=2^k$의 차수를 가지는 polynomial ring을 (3)과 같은 형태로 분해하는 과정을 Forward NTT라고 정의하며, 그 과정은 (2)를 반복하여 얻어진다. (2)의 mapping을 수행하기 위해서는 2^{k-1}회의 butterfly operation이 요구된다. 한 개의 butterfly operation은 한 번의 곱셈, 덧셈, 뺄셈으로 구성된다. 해당 과정을 거쳐 최종적으로 얻어진 isomorphic ring에서 정의되는 domain을 NTT domain이라 한다. 일반적으로 기존의 ring위에서 n차 다항식은 Forward NTT를 거쳐 모든 τ_i에 대하여 $x+\tau_i$로 reduction 되어 n개의 상수항에 대한 direct product로 변환된다.

2.2.2. Inverse NTT

Inverse NTT는 mapping (2)의 역과정으로 NTT domain에서 Forward NTT가 적용되기 이전의 domain으로 복원하는 작업을 진행한다. 이 과정 역시 Forward NTT와 유사한 연산량이 필요하다. Forward NTT와 Inverse NTT는 isomorphism을 이용하는 과정이기 때문에, $NTT^{-1}(NTT(f))=f$가 성립한다.

2.2.3. Base multiplication

R_q에서 정의된 다항식 f와g에 대하여 $h=fg$를 만족하는 다항식 $h\in R_q$는 다음을 통하여 효율적으로 연산할 수 있다.

$$h = NTT^{-1}(NTT(f)\circ NTT(g))$$

여기서 $NTT(f)\circ NTT(g)$를 수행하는 과정을 base multiplication이라 정의한다.

III. 제안하는 방법

본 장에서는 $q=257$에 대하여 제안하는 4차식 모듈러스 NTT를 사용하는 방법과 소수 특성에 따른 specialized reduction을 적용하는 방식에 대하여 설명한다.

3.1. 4차식 모듈러스 NTT

본 장에서는 LAC의 소수 q를 257로 변경하여 NTT를 구성하는 방법에 대하여 설명한다. 기존 LAC에서 제안된 ring은 $q=251$, $n=512$에 대하여 $R_q=\mathbb{Z}_q[x]/(x^n+1)$로 정의된다. $x^{512}+1\in\mathbb{Z}_{251}[x]$는 다음과 같이 분해된다.

$$x^{512}+1 =$$
$$(x^{256}+91x^{128}+250)(x^{256}+160x^{128}+250)$$

위 꼴은 일반적인 다항식 곱셈 방법만 적용할 수 있다.

반면, 제안된 소수를 $q=257$로 변경하여 새로운 ring을 구성하면 $x^{512}+1\in\mathbb{Z}_{257}[x]$는 다음과 같이 분해된다.

$$x^{512}+1=\prod_{i=1}^{128}(x^4-\tau_i)$$

이때, τ_i는 $x^{256}\equiv 1 \pmod{256}$의 primitive root로, $\tau=3$이 처음 등장하는 primitive 256-th root of unity modulo 256이 된다. 이 경우 일반적인 NTT처럼 1차 식으로 완벽하게 분해되지 않지만, 충분히 효율적인 형태로 분해됨을 알 수 있다. 위 식의 형태로 분해하여 NTT를 적용하는 것을 4차식 모듈러스 NTT라고 정의하겠다.

$\mathbb{Z}_{257}[x]/(x^{512}+1)$에서 정의되는 다항식에 대한 Forward NTT는 총 7단계($i=0,1,...,6$)로 구성된다. i 번째 단계에서 $512/2^i$보다 작은 차수를 가지는 2^i개의 다항식은 $512/2^{i+1}$보다 작은 차수의 다항식 두 개로 분해된다. 각 단계에서는 $\mathbb{Z}_{257}$위에서의 256회의 butterfly operation이 요구된다.

Inverse NTT는 forward NTT처럼 7단계($i=0,1,...,6$)로 구성된다. i 번째 단계에서 2^{6-i}쌍의 다항식들이 쌍별로 합쳐지고 그 과정에서 256회의 butterfly operation이 발생한다. 낮은 차수의 다항식들이 합쳐져 inverse NTT의 마지막 단계가 끝나고 나면 최종적으로 511차의 다항식의 각 계수를 128로 나눠주는 총 512번의 나눗셈이 필요하다. 하지만, 나눗셈 연산은 구현 관점에서 많은 시간을 소요하기 때문에, 128의 역을 사전연산을 통해 구해두고 512번의 곱셈 연산으로 대체한다.

Forward NTT를 통해 511차 다항식은 차수로 3을 가지는 128개의 다항식으로 분해되며, NTT domain에서 두 3차 다항식의 곱셈(base multiplication)에는 19**M**+12**a**의 연산이 필요하다. 따라서 두 개의 511차 다항식을 NTT domain에서 곱하는 과정에는 총 128(19**M**+12**a**)의 연산량이 요구된다.

다음 표는 벡터 곱 방식을 이용한 다항식 곱셈, NTT, Inverse NTT, 그리고 base multiplication에 필요한 연산량을 정리한 표이다.

[표 1] 연산량 비교

Operation	Mul	Add
Vector mul	**262,144**	**262,144**
NTT	**1,792**	**3,548**
Inverse NTT	**2,304**	**3,584**
Base mul	**2,432**	**1,536**

3.2. Modular reduction

본 장에서는 소수 $q=257$에 대한 specialized modular reduction 방법에 대하여 설명한다. NTT에서 modular reduction은 butterfly 연산을 진행할 때 발생하며, NTT의 전 과정 중 가장 많은 연산 부하를 가져온다. 실제로 $q\equiv 1 \pmod{2n}$이며 $n=512$인 경우, 적어도 12,375,700번 이상의 modular reduction이 발생한다. 따라서 효율적인 modular reduction의 적용은 연산 속도에 큰 영향을 미친다.

제안하는 소수는 $257=1\cdot 2^8+1$을 만족하는 수로, [3]에서 제안하는 방법을 응용하여 modular reduction을 수행하는 방법을 제안한다. 먼저 $\mathbb{Z}_{257}$에서 modular reduction 알고리즘을 제시하고, 알고리즘의 타당성을 설명한다.

Algorithm 1. Reduce257
Input : Any integer C Output : Integer D satisfies $0 \le D \le 256$
Step 1. $C_0 \leftarrow C \pmod{2^8}$ Step 2. $C_1 \leftarrow C/2^8$ Step 3. Return $D \leftarrow C_0 - C_1$

일반적인 modular 연산과 다르게 Algorithm 1의 Step 1은 8bit 마스킹으로 처리 가능하며, Step 2는 bit-shift로 처리할 수 있어서 효율적이다.

Algorithm 1을 실제로 구현할 경우 input인 C는 $C=a\cdot b,\ 0\le a,b<257$의 형태로 알고리즘에 들어오게 된다. 그러면 $0\le C<257^2=2^{16}+2^9+1$을 만족한다. C에 대하여 $C=C_0+2^8C_1,\ 0\le C_0<2^8$로 나타내고, $2^8\equiv -1 \pmod{257}$을 이용하면 다음의 합동 식이 성립한다.

$$C\equiv C_0-C_1 \pmod{257}$$

Algorithm 1의 결과인 C_0-C_1은 식 (4), (5)에 의하여 식(6)이 성립하게 된다.

$$0\le C_0\le 2^8 \tag{4}$$

$$0\le C_1=(C-C_0)/2^8<2^8+2+1/2^8 \tag{5}$$

$$|C_0-C_1|<(1+1/2^8)257 \tag{6}$$

따라서 위 알고리즘의 output은 최대 1번의 257을 빼줌으로써 $[0,257)$범위에 속하게 된다.

IV. 실험 결과

본 장에서는 제안하는 방법을 사용한 다항식 곱 연산 속도 비교 및 LAC의 key generation에 적용한 속도 비교를 제시한다. 실험 환경은 Intel Core i7-9700(Coffee Lake)에서 진행되었다.

4.1. 소수 교체 시 구현 결과

실제 구현에서 $q=251$에 대한 다항식 곱셈 연산은 School book multiplication 방식을 사용했고, $q=257$에 대해서는 NTT domain에서의 base multiplication을 사용했다. $\mathbb{Z}_{257}[x]/(x^{512}+1)$에서 정의된 두 개의 다항식 곱셈을 NTT domain 위에서의 base multiplication으로 대체하기 위해서는 두 다항식을 각각 NTT domain에서 정의된 다항식으로 변환하는 2회의 Forward NTT연산, NTT domain에서 1회의 base multiplication, 마지막으로 1회의 Inverse NTT 연산이 필요하다. 두 소수에 대한 다항식 곱셈의 이론적인 총연산량은 [표 2]와 같다.

[표 2]는 기존 $q=251$을 사용했을 때와 $q=257$을 사용했을 때 다항식 곱에서의 연산량을 비교한 표다.

[표 2] 총 연산량

	Mul	Add
$q=251$	262,144	262,144
$q=257$	8,320	12,180

[표 3]은 LAC의 다항식 곱과 본 논문이 제안하는 다항식 곱을 구현하여 비교한 표이다. $q=251$에 사용된 다항식 곱은 LAC의 reference code를 사용했다[1].

[표 3] 연산 속도

	다항식 곱셈	Cpu Cycles
$q=251$	98.281 μs	294,891
$q=257$	16.407 μs	49,070

[표 3]에서 확인할 수 있듯이, 실험 결과 $q=257$을 사용했을 때 기존 $q=251$대비 약 5배의 성능 향상을 보였다.

4.1. Reduce257 적용 시 속도 측정

4차식 모듈러스 NTT를 다항식 곱셈에 실제 적용할 경우 base multiplication과 NTT, NTT^{-1}과정에서 많은 modular reduction이 요구된다. Reduce257을 적용할 경우의 성능 향상을 확인하기 위한 실험 조건은 다음 [표 4]와 같다.

[표 4] Test options

	q	NTT	Reduce257
Option 1	251	x	x
Option 2	257	x	o
Option 3	257	o	x
Option 4	257	o	o

Option 1은 기존 LAC의 구현물을 이용하였으며, Option 2 ~ Option 4는 필요한 부분만 변경하여 구현하였다. 테스트 코드는 기존의 구현물을 이용했다. [표 4]를 구현하여 실험한 결과 성능은 다음 표와 같다.

[표 5] 속도 비교

	다항식 곱셈	Cpu Cycles
Option 1	98.281 μs	294,891
Option 2	80.937 μs	246,426
Option 3	16.407 μs	49,070
Option 4	6.875 μs	21,817

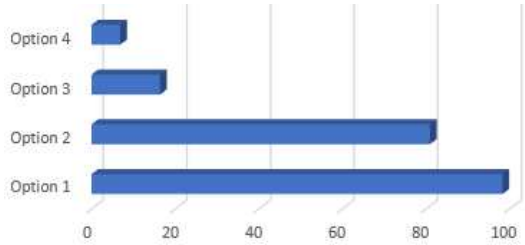


그림 1 다항식 곱셈 연산 속도

실험 결과 기존 $q=251$을 $q=257$로 바꾸어 NTT만 적용할 경우(Option 3) 약 5배의 속도 향상이, NTT와 Reduce257을 함께 적용할 경우(Option 4) 약 12배의 속도 향상을 얻을 수 있었다. 마지막으로 LAC와 본 논문의 Option 4를 사용한 경우 키 생성 단계의 속도를 비교하였다. 키 생성 단계는 $b=as+e$를 연산하는 과정으로, e를 더하는 과정은 다항식 곱셈 연산에 비해 연산 부하가 크지 않아 무시할 수 있다. 따라서, 다항식 곱셈과 큰 차이가 나지 않는다는 것을 알 수 있다.

[표 6] Key generation 비교

	Key gen	Cpu Cycles
Option1	99.063 μs	297,607
Option4	7.187 μs	22,067

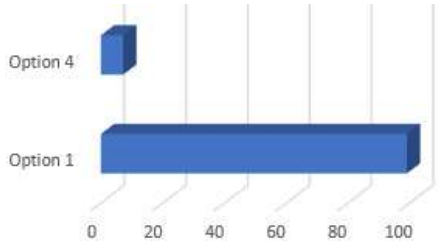


그림 2 Key generation 속도 비교

V. 결론

본 논문에서는 RLWE 기반 알고리즘에서 가장 큰 연산 부하를 차지하는 다항식 곱셈 부분을 효율적으로 연산하는 방법을 제안하였다. 본 논문에서 제안하는 소수를 이용한 NTT와 reduction 방법을 적용한 결과 키 생성 단계에서 기존보다 약 12.4배의 성능 향상을 얻을 수 있었다. 이는 q의 증가에 따른 암호문의 크기가 커지는 것을 감수할 수 있는 효율성이다.

[참고문헌]

[1] X. Lu et al. "LAC: Practical ring-lwe based public-key encryption with byte-level modulus," submission to the NIST post-quantum standardization project, 2017

[2] V. Lyubashevsky et al. On ideal lattices and learning with errors over rings. In: EUROCRYPT 2010, vol. 6110, pp. 1 - 23. 2010

[3] P. Longa et al. Speeding up the Number Theoretic Transform for Faster Ideal Lattice-Based Cryptography. CANS, Vol. 10052. pp. 124 - 139, 2016

ARIA 암호 알고리즘 TVLA 통계 검증 기법 제안+

박한별*, 김주환**, 심보연**, 서석충*,***, 장상운**** ,한동국*,***

*국민대학교 금융정보보안학과, **국민대학교 수학과, ***국민대학교 정보보안암호수학과,
****한국전자통신연구원 부설연구소

Proposal of the TVLA Statistical Verification Method for ARIA Cryptographic Algorithm

Han-Byeol Park*, Ju-Hwan Kim**, Bo-Yeon Sim**, Seog Chung Seo*,***, Sang-Woon Jang****, Dong-Guk Han*,***

*Dept. of Financial Information Security, Kookmin University,
**Dept. of Mathematics, Kookmin University,
***Dept. of Information Security, Cryptology, and Mathematics, Kookmin University,
****The Attached Institute of ETRI.

요 약

부채널 공격 저항성 시험 평가 방법론으로 제안된 TVLA(Test Vector Leakage Assessment) 통계 검증 기법은 ISO/IEC 17825 표준 문서에 대칭키 암호 알고리즘 AES와 공개키 암호 알고리즘 RSA에 대해서만 그 방법론이 제안되어 있다. 또한 CRI(Crypto Research Institute)사와 Riscure사에서 판매하여 상용되고 있는 부채널 분석 취약성 검증 장비의 TVLA 통계 검증 모듈은 AES, DES에 대해서만 그 기능을 제공하고 있다. 이에 본 논문에서는 국내 암호 알고리즘인 ARIA를 대상으로 TVLA 통계 검증을 수행하는 방법론에 대해 제안하고, 제안한 방법론에 대해 실험적으로 검증한다.

I. 서론

이론적으로 안전하게 설계된 암호 알고리즘이더라도 실제 동작하는 과정에서 발생하는 소비 전력, 전자파, 빛, 소리 등의 부가적인 정보들에 의해 비밀 정보 도출이 가능한 부채널 분석(Side Channel Analysis, SCA)에 취약하다. 이에 따라 부채널 분석 취약성에 관한 암호 모듈의 안전성 검증 필요성이 대두되었으며, 부채널 분석에 대한 안전성 시험 방법 및 기준을 제시한 국제 표준 ISO/IEC 17825가 제정되었다. ISO/IEC 17825에서는 1차 DPA(Differential Power Analysis)/DEMA(Differential Electromagnetic Analysis)에 대한 저항성 평가 방법론으로 테스트벡터를 기반으로 부채널 정보 누출 여부를 판단하는 TVLA(Test Vector Leakage Assessment) 통계 검증 기법을 제시하였다.

TVLA 통계 검증 기법은 명확한 기준에 의거하여 정해진 방식으로 진행되기 때문에 시험 결과의 객관성 및 일관성을 제공할 수 있으며, 주로 전문가에 의해 이루어질 수 있는 부채널 분석과 다르게 비전문가도 정해진 절차만 따르면 쉽게 수행할 수 있다. 그러므로 TVLA 통계 검증 기법을 이용하여 부채널 정보의 누출 여부를 사전에 판단할 수 있다면, 보다 쉽게 안전한 모듈 설계에 기여할 수 있을 것이다.

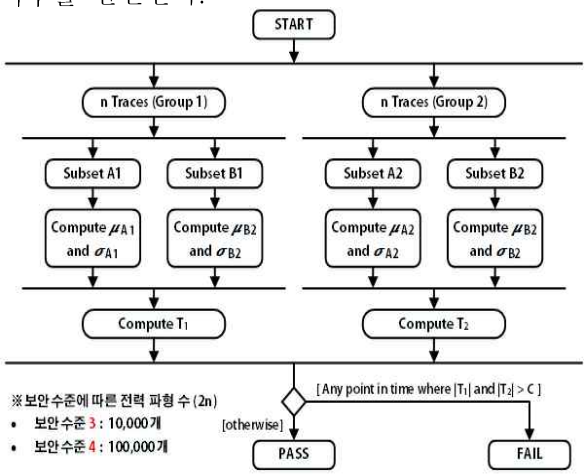
그러나 박한별 등은 현재 CRI사와 Riscure사에서 판매하여 상용되는 부채널 분석 취약성 검증 장비의 TVLA 통계 검증 모듈이 대칭키 알고리즘 AES, DES에 대해서만 일부 기능을 제공하고 있음을 보였고, 국내 암호 알고리즘에 대해 TVLA 통계 검증을 수행하기 위한 방법론이 제안되어 있지 않음을 보였다[1]. 이에 본

+ “이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2017-0-00520,SCR-Friendly 대칭키 암호 및 응용모드 개발)”

논문에서는 국내 대칭키 암호 알고리즘 ARIA에 대한 TVLA 통계 검증 방법론을 제안하고 실제 검증한 결과를 통해 제안한 방법론의 타당성을 보인다.

II. 관련 연구

본 장에서는 ISO/IEC 17825[2]에서 명시된 대칭키 암호 알고리즘 대상 TVLA 통계 검증 수행 절차를 서술한다. TVLA 통계 검증은 Welch의 T-Test를 기반으로 부채널 정보의 누출 여부를 판단한다.



[그림 1] ISO/IEC 17825에 명시된 T-Test 수행 절차

Step1. T-Test 0의 경우, [표 1]의 Data Set 1과 2에서 각각 $n/2$개의 파형을 선택하여 Group 1을 구성하고, Group 2도 동일한 방법으로 구성한다. 이때, Group 1과 2의 파형은 서로 다르다. T-Test 1~5의 경우, Data Set 1에서 n개를 선택하여 Group 1을 구성하고 나머지 n개로 Group 2를 구성한다. 이후 [표 3]의 중간 값을 기준으로 각 Group의 n개의 파형을 분류하여 집합 $A_i, B_i (i=1,2)$를 구성한다.

Step2. 두 집합 $A_i, B_i\ (i=1,2)$에 대한 T값은 다음과 같이 계산한다.

$$T = \frac{(\mu_{A_i} - \mu_{B_i})}{\sqrt{\frac{\sigma_{A_i}^2}{N_{A_i}} + \frac{\sigma_{B_i}^2}{N_{B_i}}}}$$

$N_{A_i}, N_{B_i} : A_i, B_i$집합의원소의수

$\mu_{A_i}, \mu_{B_i} : A_i, B_i$의평균, $\sigma_{A_i}\sigma_{B_i} : A_i, B_i$의표준편차

Step3. T의 절댓값이 4.5보다 크다면, 99.99%의 신뢰도로 두 집합의 평균이 다르다고 판정하고 이것은 부채널 정보가 노출되었다는 것을 의미한다. 이때 실제로 부채널 정보가 누출되지 않았으나 누출되었다고 판정하는 제1종 오류를 막기 위해 [그림 1]과 같이 T-Test를 두 번 수행하여 두 번 모두 T의 절댓값이 $C(=4.5)$를 초과하면 최종적으로 부채널 정보가 누출되었다고 판정한다.

[표 1] 대칭키 암호 알고리즘에 TVLA 테스트 벡터 (ISO/IEC 17825)

	Test Vector 성질
Data Set 1	고정 키, 랜덤 평문
Data Set 2	고정 키, 특정 조건을 만족하는 고정 평문

[표 2] Data Set 2 고정 평문 조건 (ISO/IEC 17825)

조건 1	라운드의 입력과 출력이 적어도 1바이트는 같다.
조건 2	치환 계층의 출력이 적어도 1바이트는 0이다.
조건 3	Key⊕Data함수의 출력이 적어도 1바이트는 0이다.
조건 4	적어도 1바이트의 라운드 입력이 0이다.

III. ARIA 암호 알고리즘대상 TVLA 통계 검증 방법론

본 장에서는 2장에서 서술한 ISO/IEC 17825 문서의 대칭키 암호 알고리즘 대상 TVLA 통계 검증 기법을 확장하여 ARIA 암호 알고리즘에 적용하기 위한 방법론을 제안한다.

3.1 ARIA Data Set 구성 방안

먼저 ARIA 알고리즘에 TVLA 통계 검증 기법을 적용하기 위해서는 고정 키 값에 대해 [표 2]의 4가지 조건을 동시에 만족시키는 고정 평문을 생성해야 한다. 본 논문에서는 고정 키 값이 0x0123456789ABCDEF123456789ABCDEF0일 때 [표 2]의 4가지 조건을 만족하는 방법을 서술한다. 예를 들기 위해 기준 라운드를 10라

[표 3] 대칭키 암호 알고리즘 Test Set (ISO/IEC 17825)

T-Test	중간 값
0	Data Set 1 / Data Set 2
1	라운드 입출력 해밍디스턴스
2	S-box 출력의 해밍웨이트
3	라운드 출력의 해밍웨이트
4	라운드 출력 0번째 바이트 값
5	라운드 출력 1번째 바이트 값

운드로 설정하였다고 가정하자.

(1) 조건 3 & 조건 4

Key⊕Data함수 출력의 적어도 한 바이트가 0이 되기 위해서는 9라운드의 Diffusion Layer의 출력 값, 즉, 10라운드의 입력 값과 10라운드 키 값의 적어도 한 바이트가 같게 구성이 되면 된다. 이 경우에서는 10라운드 키 값이 0x38295FB2009ED425D6B033D664FE1A2D이므로 10라운드의 입력 값 4번째 바이트를 0x00으로 설정해주면 [표 2]의 조건 3이 만족된다.

추가적으로, 이 경우에서는 10라운드 키의 4번째 바이트가 0x00이므로 위와 같은 논리로 조건 3을 만족시키면 [표 2]의 조건 4도 함께 만족시키게 된다.

(2) 조건 2

ARIA의 치환 계층 Substitution Layer의 출력이 적어도 한 바이트가 0이 되기 위해서 기준 라운드의 이전 라운드인 9라운드의 Substitution Layer 출력 값 8번째 바이트를 0x00으로 설정을 한다. Diffusion Layer의 16×16행렬 특성에 의해 10라운드 입력 값의 0, 1, 4, 7, 10, 13, 15번째 바이트의 XOR 값이 9라운드 Substitution Layer 출력의 8번째 바이트 값이 된다. 즉, [표 2]의 조건 2를 만족하기 위해서는 10라운드 입력 값의 0, 1, 4, 7, 10, 13, 15번째 바이트의 XOR 값이 0이 되도록 설정해준다.

(3) 조건 1

라운드 입력과 라운드 출력의 적어도 한 바이트를 같게 만들기 위해서, 10라운드 입력과 출력의 10번째 바이트 값을 0xAB로 설정하였다. 10라운드 출력의 10번째 바이트의 값은 10라운드 Diffusion Layer 출력의 11번째 바이트 값이다. 이는 Diffusion Layer의 16×16행렬 특성에 의해 10라운드 Substitution Layer 출력 값의 2, 3, 5, 6, 8, 13, 15번째 바이트의 XOR 값이므로 이들의 XOR 값이 0xAB가 되도록 임의로 값을 설정해준다. 이 값들을 Substitution Layer의 역 연산과 더불어 10라운드 키 값을 이용해 AddRoundKey 단계의 역 연산을 수행하면 10라운드 입력 값의 일부를 알 수 있다.

[표 4] ARIA 대상 TVLA 통계 검증을 위한 Data Set 구성

	Test Vector 성질	
Data Set 1	키	0x0123456789ABCDEF123456789ABCDEF0
	평문	랜덤한 평문
Data Set 2	키	0x0123456789ABCDEF123456789ABCDEF0
	평문	0xE6FCDF005D9E0884395BCF0FB90BC549

추가적으로 [표 2]의 조건을 만족시키는 과정에서 영향을 주지 않았던 10라운드 입력의 9, 11, 12, 14번째 바이트는 랜덤한 값들로 설정을 해주면 [표 2]의 조건을 모두 만족하는 10라운드 입력 값 전체를 알 수 있게 된다. 복호화 과정을 통해 **0xE6FCDF005D9E0884395BCF0FB90BC549**라는 평문 값을 도출할 수 있다.

3.2 ARIA Test Set 구성 방안

[표 5]의 Test Set은 ARIA 암호 알고리즘 대상 TVLA 통계 검증의 기준이 되는 중간 값이다. 이는 ISO/IEC 17825에서 제시하는 Test Set([표 3])과 [1]에서 확인했듯이 CRI의 상용장비에서 지원하는 Test Set을 ARIA의 S-Box가 4개라는 특성과 결합하여 구성한 값들이다.

[표 5] ARIA 대상 TVLA 통계 검증을 위한 Test Set 구성

T-Test	중간 값
0	Data Set 1 / Data Set 2
1	라운드 입출력 해밍디스턴스
2	S-box 출력의 해밍웨이트
3	라운드 출력의 해밍웨이트
4	라운드 출력 0번째 바이트 값
5	라운드 출력 1번째 바이트 값
6	S-Box 출력의 0번째 바이트 값
7	S-Box 출력의 1번째 바이트 값
8	S-Box 출력의 2번째 바이트 값
9	S-Box 출력의 3번째 바이트 값
10	라운드 입출력 0번째 바이트 XOR 값
11	라운드 입출력 1번째 바이트 XOR 값
12	S-Box 입출력 해밍디스턴스

IV. ARIA 암호 알고리즘대상 TVLA 통계 검증 결과

본 장에서는 3장에서 구성한 ARIA 암호 알고리즘 대상 Data Set([표 4])과 Test Set([표 5])을 이용하여 TVLA 통계 검증 수행 결과에 대해 서술한다. 또한 실제 부채널 정보의 누출 여부를 확인하기 위해 TVLA 통계 검증 수행 중간 값으로 1차 CPA(Correlation Power Anal

기호	의미
Secure	T-Test 수행 결과에서 $\|T\| > 4.5$인 시간 위치가 존재하지 않는다.
Leakage	'Fixed vs Random Test' 수행 결과에서 $\|T\| > 4.5$인 시간 위치가 존재한다.
Leakage	'Specific Test' 수행 결과에서 $\|T\| > 4.5$인 시간 위치가 존재하고, T-Test의 기준 중간 값의 위치와 $\|T\| > 4.5$ 인 시간 위치가 동일한 위치에서 발생한다.
~~Leakage~~	'Specific Test' 수행 결과에서 $\|T\| > 4.5$인 시간 위치가 존재하고, T-Test의 기준 중간 값의 위치와 $\|T\| > 4.5$ 인 시간 위치가 다른 위치에서 발생한다.

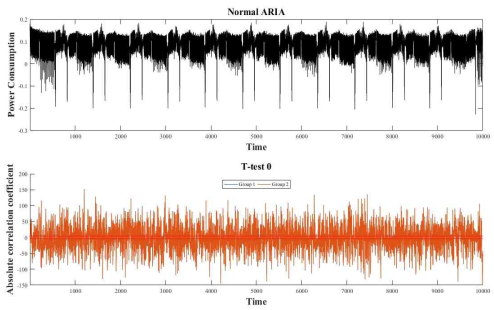
[표 6] 기호 정의

ysis)를 수행한 결과도 함께 서술한다. 실험을 위해 8비트 단위 소프트웨어로 구현된 ARIA-128 알고리즘이 ChipWhisperer-Lite 전력 수집 보드에서 보안 수준 3에 해당하는 10,000번 동작할 때의 소비 전력을 수집하였다. 이때 ARIA-128의 구현 코드로 KISA에서 제공하고 있는 공개 코드를 이용하였다.

4.1 실험 결과

[그림 4]는 ARIA-128 암호 알고리즘을 대상으로 TVLA 통계 검증과 1차 CPA를 수행한 결과이다. [표 6]은 [그림 4]를 해석하기 위해 사용되는 용어들에 대한 정의이다.

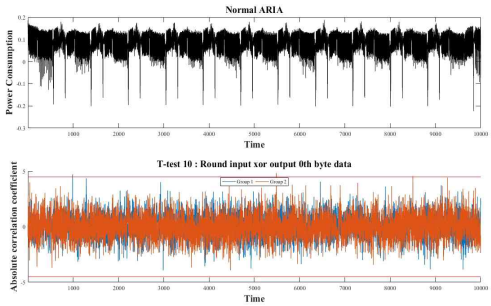
(1) Example 1 : T-Test 0



[그림 2] ARIA-128 대상 T-Test 0 결과

[그림 2]는 ARIA-128 대상 T-Test 0의 수행 결과이다. 12라운드 암호화 동작 과정 전반적으로 $|T| > 4.5$인 포인트들이 관측되었다. 이는 1차 DPA에 대한 대응 기법이 적용되지 않아 중간 값에 따른 전력 소비가 그대로 발생하므로 고정 평문을 사용한 데이터 집합의 경우의 소비 전력은 중간 값의 해밍웨이트 값으로 수렴하게 된다. 반면, 랜덤 평문을 테스트벡터로 하는 데이터 집합의 소비 전력 평균은 해밍웨이트의 평균 값인 4로 수렴하는 경향이 있다. 즉, T-Test 수행 시 고정 평문을 사용한 데이터 집합과 랜덤 평문을 사용한 집합 사이에 유의미한 통계적 차이가 발생하므로 $|T| > 4.5$인 포인트들이 관측된 것으로 해석할 수 있다.

(2) Example 2 : T-Test 12



[그림 3] ARIA-128 대상 1라운드 입출력 0번째 바이트의 XOR 값이 0인 경우 기준 T-Test 10 결과

[그림 3]은 1라운드 입출력 0번째 바이트의 XOR 값이 0인 경우를 기준으로 T-Test 10을 수행한 결과이다. 그 결과 $|T| > 4.5$인 포인트들이 일부 발생하긴 하였으나, 1라운드 위치를 기준으로 발생하지는 않았다. 이에 대해 동일한 중간 값으로 1차 CPA를 수행한 결과 [그림 4]를 참고하면, 모든 바이트에서 해당 중간 값이 노출되지 않는 것을 확인할 수 있다. 즉, [그림 3]에서 발생한 $|T| > 4.5$인 포인트들은 1라운드 입출력 0번째 바이트의 XOR 값에 대한 노출로 인해 발생한 것은 아니라고 추측할 수 있다.

(3) 전체 결과에 대한 해석

[그림 4]의 결과를 보면 전반적인 시험에서 $|T| > 4.5$인 포인트들이 관측되는 것을 확인할 수 있다. 이때 대부분의 시험에서는 시험의 기준이 되는 중간 값으로 1차 CPA를 수행한 결과 해당 중간 값에 대한 정보들의 누출이 발생하는 것을 확인할 수 있었다. 실험 대상이 1차 DPA에 대한 대응 기법이 적용되지 않았기 때문에 ARIA 알고리즘이 동작하는 동안의 소비 전력이 그대로 누출되게 된다. 이러한 원인으로

하나의 중간 값을 기준으로 T-Test를 하였을 때, 두 집합 사이의 통계적으로 유의미한 차이가 생기게 되어 $|T| > 4.5$인 포인트들이 관측되는 것으로 판단된다.

		Normal ARIA			
		Security Lv. 3			
T-Test 0	Fixed v.s. Random	Leakage			
T-Test 1	Round In ⊕ Out 1 bit HD	Secure			
T-Test 2	S-Box Out 1bit HW	S_1	S_2	S_1^{-1}	S_2^{-1}
		L	L	L	L
T-Test 3	Round Out 1 bit HW	Leakage			
T-Test 4	Round Out 0th byte data	Leakage			
T-Test 5	Round Out 1st byte data	Leakage			
T-Test 6	S-Box Out 0th byte data	Leakage			
T-Test 7	S-Box Out 1st byte data	Leakage			
T-Test 8	S-Box Out 2nd byte data	Leakage			
T-Test 9	S-Box Out 3rd byte data	Leakage			
T-Test 10	Round In ⊕ Out 0thdata	~~Leakage~~			
T-Test 11	Round In ⊕ Out 1st data	~~Leakage~~			
T-Test 12	S-Box In ⊕ Out 1 bit HD	S_1	S_2	S_1^{-1}	S_2^{-1}
		L	L	L	L
FOCPA : S-Box Output 1 bit		Leakage (All bytes)			
FOCPA : S-Box Output 8 bit		Leakage (All bytes)			
FOCPA : S-Box Input ⊕ Output 1 bit		S_1	S_1	S_1	S_1
		S_2	S_2	S_2	S_2
		S_1^{-1}	S_1^{-1}	S_1^{-1}	S_1^{-1}
		S_2^{-1}	S_2^{-1}	S_2^{-1}	S_2^{-1}
FOCPA : Round Output 1 bit		Leakage (All bytes)			
FOCPA : Round Output 8 bit (0th ,1st byte)		Leakage (All bytes)			
FOCPA : Round Input ⊕ Output (1bit & 8bit)		Secure (All bytes)			

[그림 4] ARIA-128 대상 TVLA 통계 검증과 1차 CPA 수행 결과

V.결론

TVLA 통계 검증 기법은 명확한 기준과 정해진 분석 방식으로 진행되기 때문에 시험 결과의 객관성 및 일관성을 제공할 수 있으며, 비전문가도 수행 가능하다. TVLA 통계 검증 기법을 적극적으로 활용한다면 부채널 정보의 누출 여부를 판단하여 보다 쉽게 안전한 모듈 설계에 기여할 수 있으므로 부채널 저항성 시험 평가 방법으로 반드시 필요하다. 따라서 본 논문에서 제안한 ARIA 암호 알고리즘에 대한 TVLA 통계 검증 방법론을 기반으로 현재 TVLA 통계 검증 방법론이 제시되지 않은 LEA, SEED, HIGHT와 같은 국내 암호 알고리즘들에 대해서도 점차 확대되는 계기가 되어야 한다. 이러한 연구를 바탕으로 이론적으로 안전하더라도 실제 구현 과정에서 생긴 정보 누수의 원인 파악을 할 수 있다면 안전한 암호 모듈 설계에 기여할 수 있을 것이다.

[참고문헌]

[1] 박한별, 김주환, 심보연, 서석충, 한동국 “TVLA 통계테스트 동향 분석.”, 정보보호학회 하계학술대회, 2019.

[2] Easter, R.; Quemard, J. P.; Sakurai, G. ISO/IEC DIS 17825: Information technology-Security technique-Testing methods for the mitigation of non-invasive attack classes against cryptographic modules [Z]. Berlin: DIN. 2014.

동적 분석 기반 공개키 암호알고리즘의 부채널 취약 구현 평가 연구

이예찬*, 박태환***, 진성현*, 김한빛*, 김희석**

*고려대학교 정보보호대학원 정보보호학과

**고려대학교 사이버보안학과

***한국전자통신연구원 부설연구소

Research of Searching Side-channel weakness implementation of public key cryptalgorithm based on Dynamic Analysis

Yechan Lee*, Taehwan Park***, Sunghyun Jin*, Hanbit Kim*, HeeSeok Kim**

*Graduate School of Information Security and Institute of Cyber Security & Privacy (ICSP), Korea University

**Department of Cyber Security, College of Science and Technology, Korea University

***The Attached Institute of ETRI.

요 약

동적 분석(DBI, Dynamic Binary Instrumentation) 도구는 암호 소프트웨어의 동작으로부터 이진 명령 정보를 수집할 수 있으며, 수집 대상 정보는 연산에서 참조하는 메모리의 주소, 데이터 값과 연산 명령어의 주소 등의 값이다. 기존에 동적 분석 도구는 차분 계산 분석(DCA, Differnetial Computation Analysis) 등 블록암호를 대상으로 한 부채널 분석에서만 사용되었다. 본 연구에서는 동적 분석 도구를 이용하여 부채널 대응이 적용되지 않은 공개키 암호를 분석했다. 분석 대상 암호는 몽고메리 곱셈으로 제곱-곱 연산을 수행하는 RSA 암호와 제곱 연산을 곱셈 연산으로 대체한 Atomicity RSA이며, 동적 분석 도구는 Valgrind와 PIN을 사용하였다. 독립적으로 환경을 구축한 두 팀이 서로 다른 방법으로 실험을 수행하여 동일한 결과가 도출되었다. 이를 통해 동적 분석 도구가 RSA 공개키 암호로부터 수집한 메모리 및 명령어의 주소 정보 패턴이 암호의 개인키 정보를 누출함을 확인하였다.

I. 서론

부채널 분석은 암호를 분석하여 비밀 정보를 추출할 수 있는 현실적인 공격 기법이다. 대표적으로 전력 분석 공격인 SPA, DPA[1]와 CPA [4]는 현대 암호 체계에 현실적인 위협 요소이다. 공개키 암호의 경우 키 비트에 따라 연산 과정이 달라지므로, 부채널 분석을 고려하지 않고 구현할 경우 정보의 패턴만으로도 비밀 정보를 알 수 있다.

동적 분석 도구는 소프트웨어 동작 내부에서 참조하는 메모리 정보를 수집할 수 있으며, 이를 통한 부채널 분석 공격은 화이트 박스 암호 체계에서도 유효한 공격 성능을 보였다. Bos는 이를 바탕으로 DCA 공격을 제안했고[3], 이에 따라 화이트 박스 암호에서는 동적 분석 도구를 통한 부채널 공격 및 그에 대한 안전성을 분석하는 방향의 연구가 수행되었다.

본 연구에서는 동적 분석 도구로 수집한 메모리 정보를 통해 RSA 공개키 암호 알고리즘을 최초로 분석했다. 공개키 암호 RSA의 키 비트에 따른 연산 분기는 이 메모리 정보에서도 확연하게 드러난다. 따라서 본 연구에서는 동적 분석 도구를 통해 RSA를 동작하여 메모리 정보를 수집하고, 이를

기반으로 소스 코드의 형태로 구현된 공개키 암호에 대한 부채널 분석 공격의 위험성을 확인할 수 있음을 보인다.

II. 관련 연구

2.1 동적 분석 도구를 통한 부채널 공격

동적 분석 도구는 소프트웨어의 동작으로부터 내부 연산 정보를 수집할 수 있으며, 화이트박스 블록암호에 대한 부채널 분석에 사용되었다. Chow 등의 CEJO framework[2]는 화이트박스 환경에서도 안전한 암호 구현을 제시했으나, 동적 분석 도구를 통해 수집된 정보를 바탕으로 수행한 Bos의 DCA 공격[3]으로 비밀 정보를 드러냈다. Bos는 동적 분석 도구인 Valgrind, Intel의 PIN 등을 사용하여 화이트 박스 암호로부터 메모리 정보를 수집했고, 이 정보를 바탕으로 부채널 분석인 DPA, CPA와 같은 통계적 분석을 적용하여 비밀 정보를 드러낼 수 있었다.

2.2 공개키 암호의 부채널 분석

공개키 암호는 트랩도어라 불리는 특정 정보 없이 풀기 어려운 수학적 문제를 바탕으로 구성되어, 암호화와 복호화를 서로 다른 키로 수행할 수 있는 암호 시스템이다. 따라서 현대의 공개키 암호는 모두 수학적 난제를 기반으로 안전성을 보장한다.

대표적인 공개키 암호 RSA는 큰 두 소수의 곱으로 구성된 모듈러스 안에서의 큰 정수 지수승으로 암호 연산을 수행한다. 모듈러스의 크기로 인한 소인수분해의 어려움을 통해 수학적 안전성이 보장되며, 큰 정수의 지수승 연산을 효율적으로 처리하기 위하여 제곱-곱(Square-and-Multiply) 알고리즘으로 구현한다. 제곱-곱 알고리즘으로 구현한 RSA의 전력 파형은 각 지수 비트에 따라 발생하는 제곱과 곱셈의 프리미티브 연산 차이 때문에 단순 전력 분석(SPA, Simple Power Analysis)만으로 비트의 정보가 모두 드러난다.

III. 공개키 암호 동적 분석

동적 분석 도구는 소프트웨어 환경에서 동작하는 암호의 모든 연산에 사용되는 명령어, 메모리 정보의 주소 및 데이터 값을 수집할 수 있다. 본 장에서는 이를 RSA 암호가 구현된 소스 코드의 동작에 적용하여 부채널 분석에 대한 취약성을 드러낼 수 있음을 보인다.

2.2절의 소개와 같이, 부채널 분석에 대한 어떠한 대응도 적용되지 않은 RSA 암호는 지수 비트별로 서로 다른 형태의 연산을 수행하는 조건 분기점이 발생한다. 기존 SPA의 결과와 마찬가지로, RSA 소스 코드로부터 수집한 메모리 정보에서도 이를 확인할 수 있다.

분석 환경은 다음과 같다. 먼저 분석 대상으로는 일반적인 제곱-곱 연산으로 구현된 RSA와 제곱 연산을 곱셈으로 대체하여 부채널 대응을 적용한 Atomicity RSA[7]의 두 종류로 구성했으며, 동적 분석 도구는 Bos가 제안한 Valgrind와 PIN을 모두 사용하였다. Valgrind는 화이트박스 분석을 목적으로 구성된 GitHub 공개 도구 TracerGrind[5]를 사용하였으며, Ubuntu 18.04버전의 64비트 체제 Linux에서 실험을 수행했다. PIN은 해당 도구에서 제공하는 메모리 추적 예제를 TracerPin[5]과 같은 형태로 수정하여 사용했으며, Valgrind와 같은 환경에서 실험을 수행했다.

그림 1부터 6은 동적 분석으로 수집한 2가지 종류 RSA의 명령어와 메모리 주소를 시각화한 그림이다. 그림의 x축은 모두 연산 시간 정보이며, y축은 수집된 메모리 정보의 최하위 1바이트 값이다. 모든 그림의 상단은 제곱-곱 RSA, 하단은 Atomicity RSA이다. 세부적으로 그림 1과 2는 Valgrind로 수집한 메모리 정보이며 그림 1은 메모리의 주소값의 최하위 1바이트, 그림 2는 이진 명령어의 주소값 최하위 1바이트를 그래프로 나타낸 것이다. 그림에 나타난 것과 같이, 제곱 연산과 곱셈 연산이 명확히 다른 패턴을 나타내고 있음을 확인할 수 있으며, 이를 바탕으로 RSA의 개인키 정보를 해독할 수 있다.

그림 3과 4는 RSA 공개키 암호의 지수승 연산에서 사용되는 메모리 주소에 대한 메모리

연산 시간 차이를 활용하여 비밀키를 복구한 결과를 나타낸다. 해당 그림에서 x축은 계산된 연산 시간 차이의 개수이며, y축은 계산된 연산 시간 차이를 나타낸다.

메모리 연산 시간 차이 분석을 위해 RSA 공개키 암호의 지수승 연산 방식 2종(Left-to-Right, 몽고메리 방식)에서 사용되는 데이터 메모리 주소를 기반으로 Valgrind 동적 분석 도구 및 texttrace 도구로 수집한 정보를 활용하였으며, 이를 통해 RSA 공개키 암호에서 사용되는 개인키 정보를 해독할 수 있다.

그림 5와 6의 결과는 Intel의 PIN으로 수집한 결과로, 마찬가지로 각각 메모리 주소와 이진 명령어의 주소 최하위 1바이트이다. PIN의 경우 Valgrind와 형태는 다르게 나타나지만 마찬가지로 제곱과 곱셈의 연산이 서로 다른 패턴으로 구분됨을 확인할 수 있다. Atomicity RSA의 경우, 제곱과 곱셈 연산을 동일한 곱셈 함수로 수행하므로 연산의 패턴은 구분할 수 없었으나, 두 피연산자가 서로 다른 곱셈의 경우 기존 중간값과 다른 주소값으로부터 피연산자를 참조하게 된다. PIN의 메모리 주소값의 경우 그림 5의 하단에서 확인 가능한, 상단 부분의 패턴으로 이를 식별할 수 있었으며, 단순히 곱셈 함수로 연산을 통일하는 Atomicity RSA는 참조 주소값에 의해 개인키 정보가 드러날 수 있음을 확인했다.

IV. 결론

본 연구는 동적 분석 도구로 공개키 암호로부터 메모리 정보를 수집해, 이를 바탕으로 부채널 분석 맥락의 공격이 수행 가능함을 보였다. 이는 암호 구현물로부터 부채널 취약성 구현 구조를 탐지해낼 수 있는 가능성을 보였다.

본래 동적 분석 도구는 화이트 박스를 대상으로 하는 DCA에 유효한 정보를 수집하는 도구로 여겨졌으나, 해당 연구결과를 바탕으로 일반적인 암호 시스템에 대한 부채널 분석 취약성을 파악하는 도구로서의 활용 방향성을 확인했다. 이는 동적 분석 기반 부채널 분석으로 일반적인 블랙박스 암호 시스템에 대한 부채널 취약성을 분석하는 방법론이 고안될 수 있음을 시사하는 실험결과로 해석할 수 있다.

V. Acknowledgement

본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로, 한국연구재단의 지원을 받아 수행된 연구임. (No.1711051355, 단일파형 기반 충돌 부채널 공격에 안전한 공개키 암호 대응 기술 연구)

[참고문헌]

[1] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis", in Proc. 19th Annu. Int. Cryptol. Conf., Santa Barbara, CA, USA, Aug. 1999, pp. 388-397.

[2] S. Chow, P. Eisen, H. Johnson and P. C. V. Oorschot, "White-box cryptography and an AES implementation", in Proc. 9th Workshop Sel. Areas Cryptogr. (SAC) 2002, pp. 250-270.

[3] J. W. Bos, C. Hubain, W. Michiels, and P. Teuwen, "Differential Computation Analysis: Hiding your white-box designs is not enough", in Proc. Int. Conf. Cryptogr. Hardw. Embedded Syst.}, 2016, pp. 215-236.

[4] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model", in Proc. Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES '04), M. Joye and J.-J. Quisquater, eds, 2004, pp. 16-29.

[5] https://github.com/SideChannelMarvels/Tracer

[6] P. Sasdrich, A. Moradi, and T. G¨uneysu. White-box cryptography in the gray box - a hardware implementa- tion and its side channels (to appear). In FSE 2016, LNCS. Springer, 2016.

[7] Chevallier-Mames, B., Ciet, M., Joye, M.: Low-cost solutions for preventing simple side-channel analysis: side-channel atomicity. IEEE Transactions on Computers 53, 760 - 768 (2004)

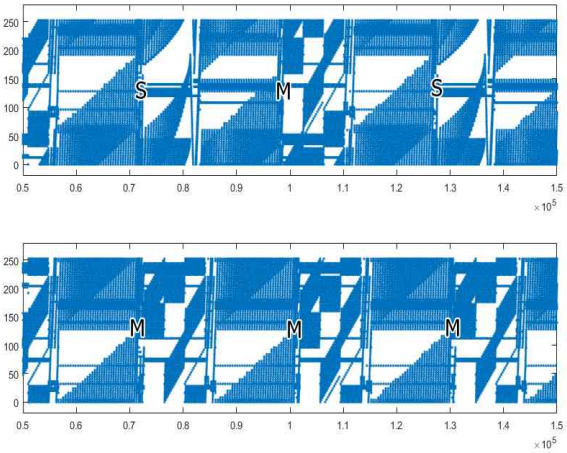


그림 1. Valgrind에서 수집한 RSA의 주소
(위 : Sq-and-mul, 아래 : Only mul)

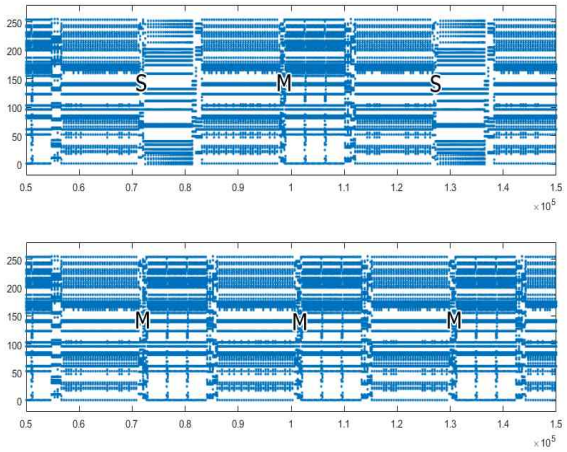


그림 2. Valgrind에서 수집한 RSA의 명령어
(위 : Sq-and-mul, 아래 : Only mul)

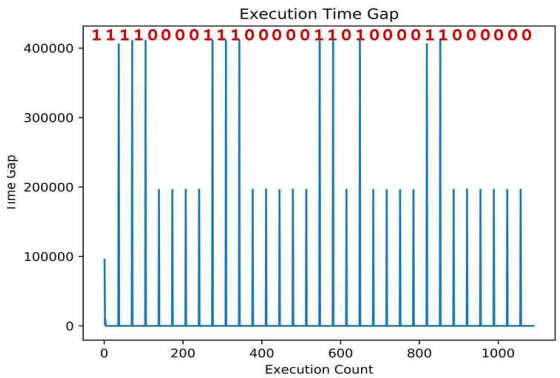


그림 3. RSA 지수승 연산(LtR) 결과 데이터 메모리 접근 연산 시간 차이 결과

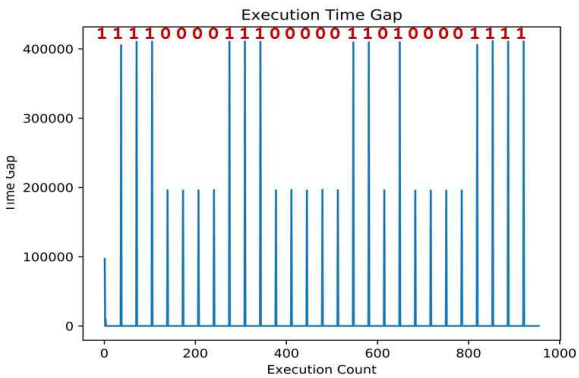


그림 4. RSA 지수승 연산(몽고메리) 결과 데이터 메모리 접근 연산 시간 차이 결과

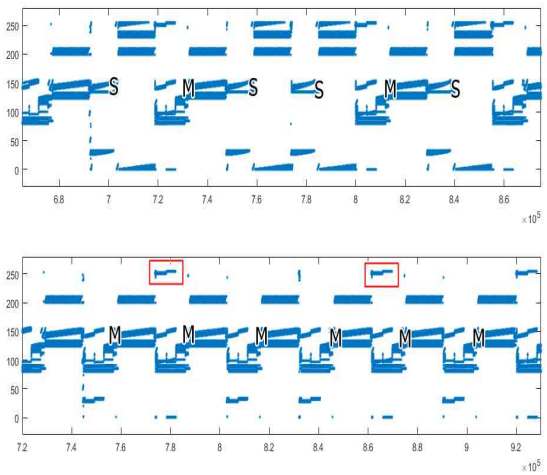


그림 5. Intel PIN에서 수집한 RSA의 주소
(위 : Sq-and-mul, 아래 : Only mul)

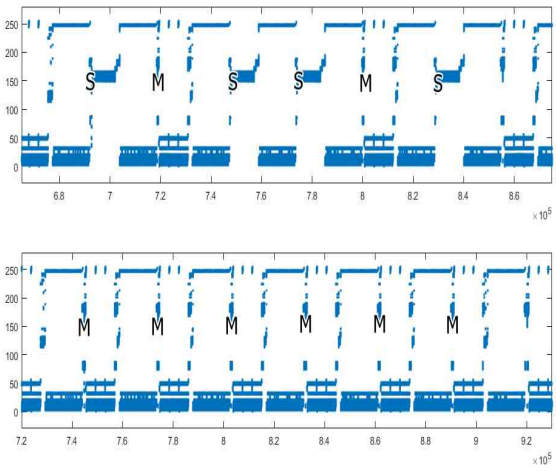


그림 6. Intel PIN에서 수집한 RSA의 명령어
(위 : Sq-and-mul, 아래 : Only mul)

가상화 환경에서 메모리 중복제거 기능을 이용한 KASLR 공격

김태현, 김태훈, 신영주

광운대학교 컴퓨터정보공학부 정보 및 사이버보안 연구실

Breaking KASLR by using Memory Deduplication in Virtualized Environments

Taehyun Kim, Taehun Kim, Youngjoo Shin

Information and Cyber Security Lab, School of Computer and Information Engineering Kwangwoon University

요 약

KASLR(Kernel Address Space Layout Randomization)은 커널이 메모리에 적재될 때마다 커널의 주소를 랜덤화 하여 ROP(Return Oriented Programming)와 같은 코드 재사용 공격에 커널을 보호하기 위한 기법으로 대부분의 최신 운영체제에 적용된다. 본 연구에서는 가상화 환경에서 널리 사용되는 메모리 중복제거 기능을 부채널로 활용하여 KASLR의 정보를 획득하는 공격을 제안한다. VMware ESXi와 KVM 하이퍼바이저 위에서 최신 패치가 적용된 리눅스 가상머신을 대상으로 실험을 진행한 결과 KASLR에 의해 랜덤화된 베이스 주소 값을 확인할 수 있었다. 본 논문에서는 제안 공격으로부터 시스템을 안전하게 보호하기 방어기법도 같이 제시한다.

I. 서론

커널은 컴퓨터의 운영체제의 핵심이 되는 프로그램 중 하나로, 시스템을 제어 및 관리하는 프로그램이다. 운영체제는 커널을 보호하기 위해 다양한 방어기법을 도입하였으며, 그중 하나가 KASLR이다. 운영체제는 KASLR을 이용하여 ROP 공격과 같은 커널의 정확한 주소를 이용하여 공격하는 코드 재사용 공격에 대한 방어기법으로 사용하고 있다.

그러나 CPU 부채널 공격에 의해 KASLR을 무력화하는 여러 가지 기법들이 발견되면서 보안에서 상당한 위협이 되어 왔다. 발견된 공격은 크게 두 가지로 나눌 수 있는데, 첫 번째는 캐시를 기반으로 한 공격으로 TLB[1] 와 캐시 최적화 기술인 프리패치[2]를 이용한 공격이 있다. 두 번째는 인텔 프로세서의 최적화 기술을 이용한 공격으로 Intel TSX(Transaction Synchronization eXtension)[3]와 비 순차 실행[4]을 이용한 공격으로 나눌 수 있다. KASLR을 CPU 부채널 공격에 방어하기 위해 KPTI(Kernel Page Table Isolation)[5]을 적용하였다. KPTI는 커널 페이지와 유저 페이지를 나누어서 유저 주소 공간에서 커널 주소 공간에 접근하는 것을 불가하게 함으로써, CPU 부채널 공격으로부터 KASLR을 무력화하는 것을 막게 하였다.

본 논문은 메모리 중복 제거 공격을 이용하여 KPTI가 적용된 상황에서도 KASLR을 무력화하는 공격을 제안한다. 메모리 중복 제거 공격은 Suzaki 등[6]에 의해 제안된 공격으로 하나의 호스트에서 여러 개의 가상머신을 운용하고 있을 때, 메모리를 절약하기 위해 같은 내용을 가진 페이지들을 하나로 합쳐서 공유하는 형태인 메모리 중복 제거 기술을 이용하여 하나의 가상머신에서 다른 가상머신이 가지고 있는 메모리 주소를 알아내는 공격이다. 본 논문에서 제안한 공격은 가상화 환경에서 사용되는

대표적인 하이퍼바이저인 KVM과 VMware ESXi 위에 동작하는 가상머신에 대해 공격을 진행하였으며, 가상머신의 운영체제로는 Ubuntu 18.04를 사용하였다. 해당 공격에 대한 방어기법으로는 메모리 중복 제거 기술을 비활성화 하는 것을 포함하여 다른 방어기법을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 제안된 공격에 대한 배경지식을 설명한다. 3장에서는 제안된 공격에 대한 공격과정을 설명하며 4장에서는 실험결과를 기술한다. 5장으로는 해당 공격에 대한 방어기법을 설명하며 6장에서 결론으로 마무리한다.

II. 배경지식

2.1 메모리 중복 제거

메모리 중복 제거는 하이퍼바이저가 가상머신을 관리하는 페이지 단위로 스캔을 하고 같은 내용을 가진 페이지들이 존재한다면, 하나의 페이지로 합치고 이 페이지는 읽기 전용(Read-only) 권한을 갖는다. 이 과정을 통해 여러 개의 가상머신이 같은 내용을 가진 페이지들을 공유해서 사용함으로써, 메모리 낭비를 줄이는 것이 가능하다. 읽기 전용으로 설정된 페이지를 쓰기 접근 (Write-access)을 하게 된다면 페이지 폴트가 발생하며, 해당 폴트를 처리한 후 하나의 페이지를 복사한 후 복사된 페이지에 대해 쓰기 접근을 진행한다. 이 과정을 COW(Copy-On-Write)라고 한다. 메모리 중복 제거를 지원하는 대표적인 하이퍼바이저는 KVM의 VMware ESXi이며, 각각 KSM(Kernel Samepage Merging)과 TPS(Transparent Page Sharing)로 메모리 중복 제거를 지원한다.

2.2 KASLR

KASLR은 운영체제 부팅 시, 운영체제를 가동하는데 필요한 커널들을 메모리에 적재할 때 적재되는 커널의 베이스 주소 값들을 랜덤화 하는 기법이다. 윈도우와 리눅스 계열에서 지원되며, ROP와 같은 커널 주소 값을 이용한 코드 재사용 공격에 대한 방어기법으로 사용된다. 랜덤화 과정은 KASLR에 적용되기 전 커널 베이스 주소에 생성된 난수만큼의 옵셋이 더해져서 커널의 베이스 주소가 결정된다.

III. 공격 기법

공격은 가상화 환경에서 메모리 중복 제거된 페이지에 쓰기 접근을 함으로써 공격자가 희생자의 커널 텍스트와 커널 모듈의 KASLR 정보를 획득할 수 있는 공격이다. 공격자와 희생자는 같은 커널 버전을 가진 환경으로 구성하고 공격과정을 세 단계로 나누어 설명한다. 앞으로 설명할 단계에서 공격자, 희생자는 각각 공격자 가상머신과 희생자 가상머신을 의미한다.

3.1 공격 대상 선정

공격하기에 앞서 공격자는 공격 대상을 선정해야 한다. 커널 텍스트는 가장 기본이 되는 커널 함수들로 이루어져 있고 커널 모듈은 다른 커널 모듈의 함수들을 참고한다. 따라서 특정 커널 모듈의 KASLR 정보를 획득하기 위해서 Fig 1. 과 같이 반드시 커널 텍스트의 KASLR 정보를 획득해야 한다. 커널 심볼에 대한 정보는 /proc/kallsyms 파일에서, 커널 모듈은 /proc/modules에서 정보를 찾을 수 있다. 커널 텍스트가 메모리에 적재될 때 배치되는 각 커널 함수의 순서는 항상 일정하므로 커널 심볼 중 하나의 심볼에 대해 KASLR의 값을 구하면 모든 커널 심볼의 주소도 알 수 있다. 반면 커널 모듈은 메모리에 적재되는 순서가 항상 일정하지 않기 때문에 특정 원하는 모듈에 대해서 각각 공격을 진행해야 한다.

3.2 대상 후보 페이지 생성

커널이 적재되는 위치의 범위는 KASLR 엔트로피에 의해 정해진다. 커널 텍스트는 9bit, 커널 모듈은 18bit 엔트로피를 갖기 때문에 커널 텍스트, 커널 모듈 공격을 위해 각각 512개, 262,144개의 후보 페이지를 준비해야 한다. 하이퍼바이저는 일반적으로 4KB 단위로 페이지

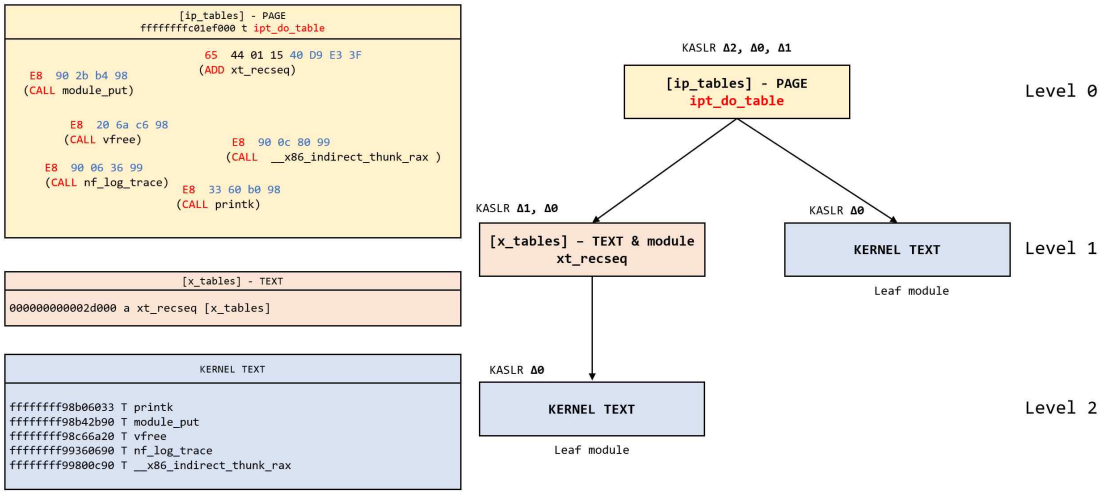


Fig 1. ipt_do_table [ip_tables]의 의존성 관계도

를 스캔하면서 메모리 중복 제거 기능을 수행하기 때문에 공격 대상의 주소에 대해서 하위 12bit를 마스킹하여 4KB의 페이지를 준비한다. 후보 페이지마다 변하는 내용은 objdump를 통해 정적 재배치에 의해 바뀐다는 것을 확인하였고, 단 2개의 타입(R_X86_64_PC32, R_X86_64_32S)에 의해 정적 재배치된다. 따라서 공격자는 희생자가 KASLR이 적용된 커널 주소에 대해 후보 페이지를 생성할 수 있다.

3.3 쓰기 접근과 시간 측정

세 번째 단계에서 공격자는 두 번째 단계에서 생성한 후보 페이지 중 희생자가 가지고 있는 페이지를 알아내야 한다. 메모리 중복 제거가 지원되는 환경에서 공격자는 mmap()을 통해 후보 페이지 모두를 메모리에 맵핑한다. 일정 시간 후에 하이퍼바이저가 중복 제거 대상 메모리를 모두 스캔하고 나면 공격자의 후보 페이지 중 하나의 페이지가 희생자와 같으므로 중복 제거된다. 공격자는 중복 제거된 페이지를 찾기 위해 맵핑한 모든 후보 페이지들에 대해 쓰기 접근을 진행하고 CPU 사이클을 측정한다. 중복 제거된 페이지는 읽기 전용 권한을 가지고 있으므로 쓰기 접근을 할 때 페이지 폴트와 COW 메커니즘이 동작하면서 10,000 ~ 100,000 사이클이 측정되고 일반적인 페이지는 100 ~ 1,000 사이클이 측정된다. 이러한 CPU 사이클 차이를 통해 공격자는 희생자가 가지고 있는 같은 페이지를 얻음으로써 희생자의 커널 텍스트, 커널 모듈의 메모리 주소를 알 수 있다.

IV. 실험

실험 환경의 구성과 공격 결과는 Table 1.과 같다. 실험은 가상화 환경에서 2개 이상의 가상 머신을 구축하고 공격을 진행하였다. 가상 머신의 운영체제는 모두 KPTI 패치가 완료된 Ubuntu 18.04.3 LTS, 최신 커널인 5.3.7 버전에서 공격하였다. 공격 대상은 ip_tables 모듈의 ipt_do_table()를 선정하여 실험을 하였다. ipt_do_table()은 x_table 모듈의 xt_recseq()와 커널 함수를 호출하기 때문에 Fig 1.처럼 Level 1에 두 개의 노드가 있다. xt_recseq()는 다른 커널 모듈의 함수들을 참고하지 않기 때문에 Level 2에 한 개의 노드만 존재한다.

Ubuntu KVM의 KSM 디폴트 환경에서 실험을 진행한 결과 커널 텍스트에 대한 KALSR 정보를 획득하는데 약 3분, 커널 모듈을 대상으로 하는데 약 12분이 소요되었다. 한편 VMware ESXI의 TPS의 디폴트 환경에서 커널 텍스트를 공격하는데 약 2시간, 커널 모듈을 공격하는데 약 8시간이 소요되었다. 이러한 차이는 KVM의 KSM과 ESXI의 TPS 기능이 얼마나 자주 메모리를 스캔하는지에 따라 발생한다.

Hypervisor	CPU	Guest OS / Kernel version	Target	Attack
Ubuntu KVM	Intel Core i5-7400	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓
	Intel Core i7-6700	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓
	Intel Core i7-8700	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓
	Intel Xeon E5-2620 v4	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓
	AMD Ryzen7 2700x	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓
VMwareESXi	Intel Core i5-7400	Ubuntu 18.04.3 / 5.3.7	Kernel Text/Module	✓

Table 1. 실험 환경 및 결과

V. 방어기법

해당 공격에 대한 몇 가지의 방어기법을 소개한다.

- 메모리 중복 제거 비활성화: 메모리 중복 제거 기술을 제거하게 된다면, 본 논문의 제안 공격을 방어하는 것이 가능하다. 그러나 이 방어기법은 서버의 성능을 크게 저하시키는 문제를 안고 있다.
- KASLR 안전성 강화: 현재 커널 모듈에 KASLR이 적용될 때 오직 처음 적재되는 커널 모듈에만 적용이 된다. 매번 커널 모듈이 적재될 때마다 해당 커널 모듈의 베이스 주소를 바꾸게 된다면, 다른 커널 모듈의 베이스 주소를 예측하는 것이 더욱 어려워지므로 공격의 난이도를 높일 수 있게 된다.

VI. 결론

본 논문에서는 가상화 환경에서 메모리 중복 제거 기능을 이용하여 리눅스 가상머신의 KALSR을 무력화할 수 있는 공격 기법을 제안하였다. 본 제안 방법은 KPTI 패치가 적용된 최신 버전의 리눅스에서도 적용이 가능함을 실험을 통해 확인하였다. 본 논문에서는 제안 공격으로부터 시스템을 안전하게 보호하기 위한 몇 가지 대응 기법들을 제시하였다.

Acknowledgement

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(2019-0-00533, 컴퓨터 프로세서의 구조적 보안 취약점 검증 및 공격 탐지 대응).

[참고문헌]

[1] Hund, R., Willems, C., Holz, T.: Practical Timing Side Channel Attacks against Kernel Space ASLR. In: S&P'13 (2013)

[2] Gruss, D., Maurice, C., Fogh, A., Lipp, M., Mangard, S.: Prefetch Side-Channel Attacks: Bypassing SMAP and Kernel ASLR. In: CCS'16 (2016)

[3] Jang, Y., Lee, S., Kim, T.: Breaking Kernel Address Space Layout Randomization with Intel TSX. In: CCS'16 (2016)

[4] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown: Reading Kernel Memory from User Space," in USENIX Security Symposium, 2018.

[5] GRUSS, D., LIPP, M., SCHWARZ, M., FELLNER, R., MAURICE, C., AND MANGARD, S. KASLR is Dead: Long Live KASLR. In International Symposium on Engineering Secure Software and Systems (2017), Springer, pp. 161 - 176.

[6] SUZAKI, K., IIJIMA, K., YAGI, T., AND ARTHO, C. Memory deduplication as a threat to the guest. In Proceedings of the 2011 European Workshop on System Security (Salzburg, Austria, 2011).

주파수 기반 스마트폰 카메라 동작 탐지 전자파 분석 연구+

이지우,* 한동국*,**

*국민대학교 금융정보보안학과, **국민대학교 정보보안암호수학과

Frequency-based Electromagnetic Analysis of Smartphone Camera Operation Detection

Ji-Woo Lee,* Dong-Guk Han*,**

*Dept. of Financial Information Security, Kookmin University,
**Dept. of Information Security, Cryptology, and Mathematics, Kookmin University.

요 약

오늘날 스마트폰에서 다양한 서비스를 제공하는 애플리케이션이 개발됨에 따라 스마트폰의 활용성은 점차 높아지는 추세이다. 하지만 최근 일부 악의적인 사용자들에 의해 카메라, 음성 녹음 등의 애플리케이션이 불법 촬영, 도청 등의 범죄 행위 수단으로 이용되면서 스마트폰을 이용한 피해 사례 또한 증가하고 있다. 본 논문에서는 스마트폰 기기를 대상으로 기종에 따라 내장된 카메라 모듈을 확인한 후, 각 스마트폰에서 카메라가 동작할 때 발생하는 전자파를 분석하여 카메라 동작 여부를 탐지할 수 있음을 보인다. 이를 통해 사진 및 동영상 촬영의 기능을 제공하는 애플리케이션의 동작 여부를 판단하여 불법 촬영과 같은 스마트폰을 이용한 범죄 행위를 탐지할 수 있다.

I. 서론

오늘날 스마트폰은 단순히 음성 통화의 기능을 넘어 메시지 송·수신, 사진 및 동영상 촬영, 음성 녹음, 인터넷 등의 서비스를 제공하고 있다. 또한, 스마트폰에 내장된 각종 센서를 활용하여 다양한 서비스를 제공하는 애플리케이션이 개발됨에 따라 스마트폰의 활용성은 점차 높아지는 추세이다. 하지만 최근 일부 악의적인 사용자들에 의해 카메라와 같은 애플리케이션이 불법 촬영 등의 범죄 행위 수단으로 이용되면서 스마트폰을 이용한 피해 사례 또한 증가하고 있다.

이에 따라 불법 촬영을 탐지하기 위한 다양한 제품들이 나오고 있지만, 현재 상용되고 있는 불법 카메라 탐지기의 경우 대부분의 불법 카메라가 사용하고 있는 1.2 GHz나 2.4 GHz 주파수 대역의 전자파만을 탐지하는 기술을 적용하고 있으며, 이는 불법 카메라의 70% 정도만 탐지할 수 있다. 또한, 스마트폰 애플리케이션 형태로 제공되는 탐지 기술은 전자파의 세기에 따라 전자기기의 존재 여부를 판단하기 때문에 정확한 범죄 행위에 대한 판단을 할 수 없다는 한계점이 존재한다.

본 논문에서는 스마트폰 기기를 대상으로 기종에 따라 내장된 카메라 모듈을 확인한 후, 각 스마트폰에서 카메라가 동작할 때 발생하는 전자파의 주파수를 분석하여 카메라 동작 여부를 탐지할 수 있음을 보인다. 이를 통해 사진 및 동영상 촬영의 기능을 제공하는 애플리케이션의 동작 여부를 판단하여 불법 촬영과 같은 스마트폰을 이용한 범죄 행위를 탐지할 수 있다.

+ 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

II. 연구 동향

2019 한국정보보호학회 하계학술대회에서 스마트폰에서 발생하는 전자파 파형을 탐지하여 카메라 및 음성 녹음의 기능을 제공하는 애플리케이션의 동작 여부를 확인하는 연구가 발표되었다[1]. 해당 연구에서는 스마트폰 CPU에서 발생하는 전자파를 측정하였으며, 애플리케이션에 따라 전자파 파형의 패턴이 다르게 발생함을 보였다. 하지만 동일한 기능을 수행하더라도 애플리케이션의 종류, 잡음 수준 등에 따라 상이한 결과가 나오기 때문에, 추후 탐지 시스템 구축을 위한 템플릿 구성에 어려움이 있다.

본 논문에서는 스마트폰의 후면 카메라가 동작할 때 발생하는 전자파의 주파수 대역을 분석하여, 카메라 모듈에 따른 템플릿 구성을 통해 불법 촬영 탐지 시스템 구축이 가능함을 실험을 통해 보인다.

III. 주파수 기반 스마트폰 카메라 동작 탐지 전자파 분석 연구

본 장에서는 스마트폰에서 발생하는 전자파를 분석하기 위한 실험 환경 및 카메라 모듈에 따른 전자파 분석 결과를 소개한다.

3.1 분석 대상

본 논문의 분석 대상 스마트폰은 SAMSUNG, LG, Apple 총 3개 제조사의 스마트폰 7종으로 선별하였으며, 이들의 카메라 모듈은 표 1과 같다.

표 1. 분석 대상 스마트폰 및 카메라 모듈

제조사	스마트폰	카메라 모듈
SAMSUNG	Galaxy S3	Sony IMX175
	Galaxy S8	Sony IMX333
	Galaxy Note 8	Sony IMX333
LG	G6	Sony IMX258
	V30	Sony IMX351
	G7 thinQ	Sony IMX351
Apple	iPhone X	Sony IMX378

3.2 분석 환경

스마트폰에서 발생하는 전자파 분석을 위한 실험 환경은 그림 1과 같이 전자파 프로브와 스펙트럼 분석기를 이용하여 구성하였다. 전자파 프로브의 경우 Langer 사의 100 kHz에서 50MHz의 주파수 대역을 탐지하는 LF R 400 프로브와 50 MHz에서 150 MHz의 주파수 대역을 탐지하는 RF R 400-1 프로브를 이용하였으며, 그림 2와 같이 각 스마트폰의 카메라 근처에 프로브를 근접시켜 전자파를 탐지하였다. 또한, 스펙트럼 분석기의 경우 100 kHz의 대역폭으로 10 ms동안 발생하는 전자파를 탐지하도록 설정하였다.



그림 1. 분석 환경



그림 2. 전자파 프로브 위치

3.3 분석 결과

본 절에서는 스마트폰 기종에 따라 카메라가 동작할 때 발생하는 주파수 기반 전자파 분석 결과를 소개한다.

전체 분석 결과는 표 2와 같으며, 결과적으로 스마트폰 기종이 아닌 카메라 모듈에 따라 동작 주파수가 결정되는 것을 확인할 수 있었다. 이는 각 스마트폰에서 제공하는 기본 카메라 애플리케이션 외에 온라인에서 제공하는 카메라 애플리케이션을 실행하였을 때도 동일한 결과를 나온다는 것을 의미한다.

본 논문에서는 대표적으로 SAMSUNG Galaxy S3에 대한 분석 결과를 소개한다.

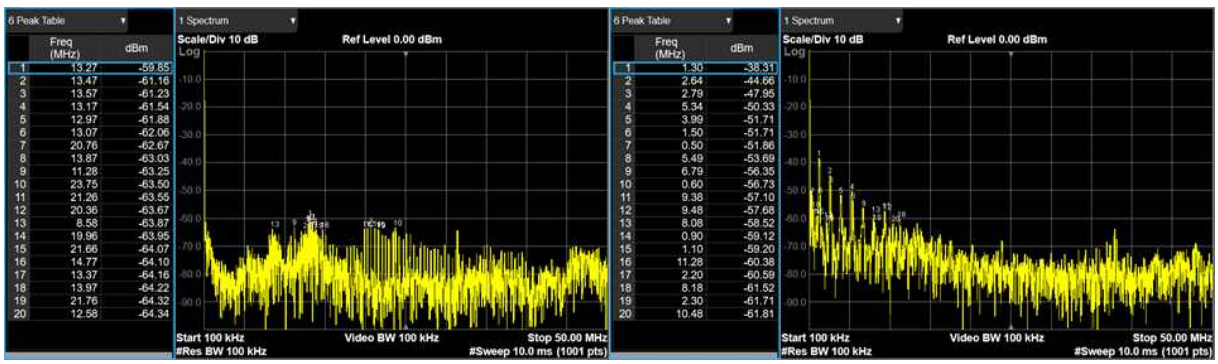
그림 3은 SAMSUNG Galaxy S3에서 카메라 동작 여부에 따라 전자파를 수집하여 주파수 도메인 상에서 바라본 파형이다. 카메라가 동작하지 않을 때는 (a)와 같이 주파수 대역이 일정하지 않지만, 카메라를 동작시키면 (b)와 같이 1.3 MHz부터 Harmonic 성질을 띄우며 피크가 발생하는 것을 확인할 수 있다. 이를 통해 해당 스마트폰의 카메라 모듈은 1.3 MHz의 동작 주파수를 가지는 것을 알 수 있다.

또한, 그림 4와 같이 카메라 동작 주파수로 주파수 필터를 적용하여 타임 도메인 상에서 바라보면 카메라 동작 여부에 따라 전자파 파형이 달라지는 것을 확인할 수 있다. (a)와 같이 카메라가 동작하지 않을 때는 잡음 신호로 추정되는 파형이 지속적으로 발생하지만, (b)와 같이 카메라가 동작하면 일정한 패턴을 가지는 파형이 발생한다.

이와 비교하기 위해 그림 5와 같이 카메라 동작 주파수가 아닌 임의의 주파수를 이용하여 필터를 적용한 후 타임 도메인 상에서 파형을 수집하였으며, 이때는 카메라 동작 여부에 따른 전자파 파형에 차이가 없음을 확인할 수 있다.

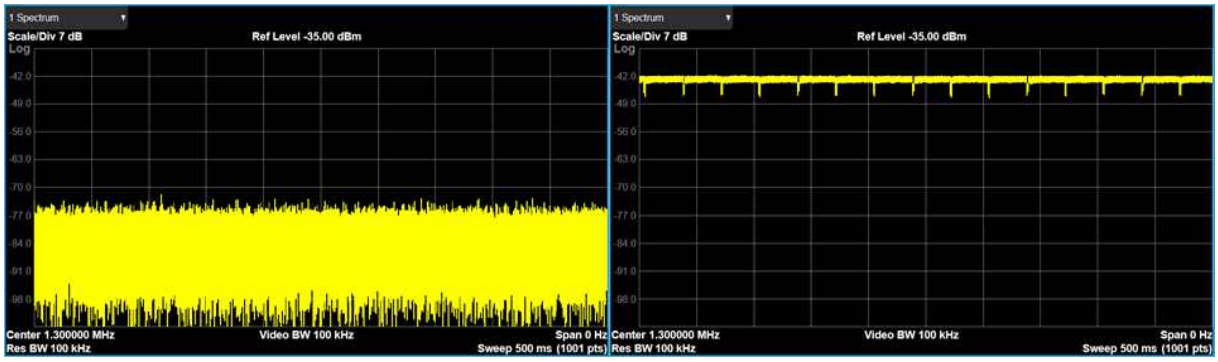
표 2. 스마트폰별 카메라 동작 시 전자파 분석 결과

제조사	스마트폰	카메라 모듈	동작 주파수 (MHz)	Harmoic 성질
SAMSUNG	Galaxy S3	Sony IMX175	1.3	O
	Galaxy S8	Sony IMX333	78	O
	Galaxy Note 8	Sony IMX333	78	O
LG	G6	Sony IMX258	1.6	O
	V30	Sony IMX351	3.89	O
	G7 thinQ	Sony IMX351	3.89	O
Apple	iPhone X	Sony IMX378	1.89	O



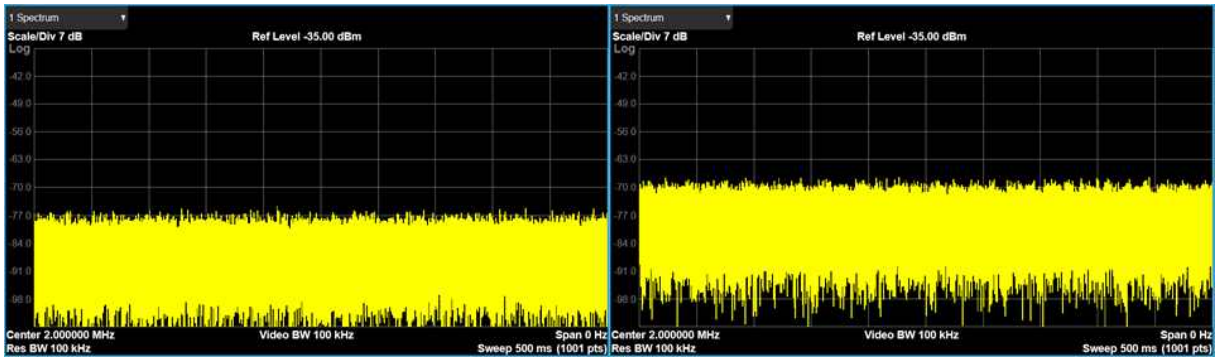
(a) (b)

그림 3. SAMSUNG Galaxy S3 동작 주파수 (a) 대기 화면 (b) 카메라 애플리케이션 실행



(a) (b)

그림 4. 카메라 동작 주파수로 필터 적용한 전자파 파형 (a) 대기 화면 (b) 카메라 애플리케이션 실행



(a) (b)

그림 5. 임의의 동작 주파수로 필터 적용한 전자파 파형 (a) 대기 화면 (b) 카메라 애플리케이션 실행

IV.결론

본 논문에서는 스마트폰에서 카메라 모듈에 따라 발생하는 전자파의 동작 주파수가 다르며, 해당 주파수로 필터 적용 시 카메라 동작 여부에 따라 전자파 파형이 달라짐을 실험을 통해 보였다. 기존에 애플리케이션에 따라 발생하는 전자파 파형을 탐지하여 카메라 및 음성 녹음을 수행하는 애플리케이션의 동작 여부를 확인하는 연구가 있었지만, 이는 동일한 기능을 수행하더라도 애플리케이션의 종류, 잡음 수준 등에 따라 상이한 결과가 나오기 때문에, 추후 탐지 시스템 구축을 위한 템플릿 구성에 어려움이 있다. 하지만 본 논문에서는 카메라 모듈에 따른 동작 주파수를 탐지하기 때문에 잡음 수준에 큰 영향을 받지 않으며, 카메라 모듈에 따른 템플릿을 용이하게 구성할 수 있다.

향후 분석 대상 스마트폰을 확대하여 분석할 예정이며, 이를 통해 상용적으로 활용 가능한 불법 카메라 탐지 기술을 구축할 수 있음이 기대된다.

[참고문헌]

[1] 이지우, 한동국, “스마트폰 노출 전자파 정보를 이용한 애플리케이션 탐지”, 2019 한국정보보호학회 하계학술대회.

CPU 환경에서 PBKDF2-HMAC-SHA256 최적화 연구

최호진, 서석충*

국민대학교 정보보안암호수학과

Study on PBKDF2-HMAC-SHA256 Optimization in CPU Environment

Ho-Jin Choi, SeogChung Seo

Department of Information Security, Cryptology, and Mathematics, Kookmin University

요 약

PBKDF2는 파일암호화, 인증시스템 구현 등 실생활에서 매우 빈번하게 사용되는 알고리즘이다. 또한, 패스워드의 낮은 엔트로피로 인하여 안전성을 높이기 위해서는 내부 이터레이션의 횟수를 높여야 한다. 하지만, 이터레이션 횟수를 올릴수록 많은 연산 시간이 소요된다. 본 논문에서는 PBKDF2의 내부 연산과정을 최적화할 수 있는 방안을 제시하고 이에 대한 성능 결과를 제시한다. 제안방법을 통하여 기본 버전 대비 총 2.2배의 성능을 개선할 수 있었다.

I. 서론

현대 사회는 IoT 환경이 많아지고 IoT 장치들이 무선 통신을 사용하고 있다. 하지만 무선 통신은 도청이 쉬워 통신 간 암호화를 적용해야 한다. 그뿐만 아니라, 클라우드 환경, 서버가 대량의 사용자 서비스에 대해서 암호화 과정이 필요하다.

PBKDF2(Password Based Key Derivation Function, 이하 PBKDF)는 사용자로부터 패스워드를 입력받아 암호 키를 도출하는 알고리즘이다. 하지만, 패스워드의 경우 일반 난수키에 비해 엔트로피가 낮기 때문에 내부적으로 많은 수의 이터레이션을 통해 키를 도출한다. 권고하고 있는 최소 이터레이션 횟수는 1,000번이나, 공격방법의 발전으로 인해 더 높은 횟수의 이터레이션 동작이 요구되고 있는 실정이다. 본 논문에서는 PBKDF2의 연산을 최적화할 수 있는 방법을 제시하여 더 빠르게 많은 수의 이터레이션을 수행할 수 있도록 한다.

본 논문에서는 PBKDF2를 최적화하기 위해 총 4가지 방법이(BR, IS, ZB, BO) 있으며 이를 최적화 구현하여 총 2.2배의 성능을 달성할 수 있었다.

* 이 논문은 2019년도 정부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2019R1F1A1058494)

II. 관련연구

2.1 PBKDF2 구조

PBKDF는 기존 키와 다른 파라미터로 파생키를 생성하는 키 파생 함수(Key Derivation Function, 이하 KDF)를 이용하는 함수이다.. PBKDF의 버전은 PBKDF1과 PBKDF2가 존재한다.[1] 하지만 PBKDF1에서 주로 사용하는 해시 함수인 MD2, MD5, SHA-1 안전성 문제 및 일부 애플리케이션에서 PBKDF1의 안전성은 크지 않다.[1] 따라서 PBKDF2 사용이 권장된다.

```
PBKDF2 (P, S, c, dkLen)

Options:        PRF         underlying pseudorandom function (hLen
                            denotes the length in octets of the
                            pseudorandom function output)

Input:          P           password, an octet string
                S           salt, an octet string
                c           iteration count, a positive integer
                dkLen       intended length in octets of the derived
                            key, a positive integer, at most
                            (2^32 - 1) * hLen

Output:         DK          derived key, a dkLen-octet string
```

[그림 1] PBKDF2 Input/Output [1]

[그림 1]에서는 PBKDF2의 Input과 Output을 제시한다. [1, 3]에 의하면 PBKDF2로 원하는 길이만큼 DK를 생성할 수 있고, PBKDF2 작동 과정에서 DK를 커버할 수 있는 Ti 블록을 여러개 생성한다. 각각의 Ti 블록은 PRF를 반복 횟수인 C만큼 반복 호출하며 블록을 생성한다. 즉 Ti의 블록길이는 PRF의 출력길이(hLen)에 의존한다.

Ti 블록과 출력 값인 DK를 생성하는 과정은 [그림 2]와 같다.[1, 2]

$$DK = T_1 || T_2 || \dots || T_{\lceil dkLen/hLen \rceil}$$

$$T_1 = Function(p, s, c, 1)$$
$$T_2 = Function(p, s, c, 2)$$
$$\vdots$$
$$T_i = Function(p, s, c, i)$$
$$\vdots$$
$$T_{\lceil dkLen/hLen \rceil} = Function(p, s, c, \lceil dkLen/hLen \rceil).$$

[그림 2] PBKDF2 Process [2]

[그림 2]에서 제시되는 Function은 [그림 1]에서 PRF를 뜻한다. DK의 길이만큼 커버하기 위해 Ti 블록을 생성하는데 총 [dkLen/hLen]만큼의 개수가 필요하다. 출력 값인 DK를 생성하기 위한 각각의 Ti 블록은 [그림 3]과 같다.

$$T_i = U_1 \oplus U_2 \oplus \dots \oplus U_c$$

$$U_1 = PRF(p, s||i)$$
$$U_2 = PRF(p, U_1)$$
$$\vdots$$
$$U_c = PRF(p, U_{c-1})$$

[그림 3] PBKDF2 Process[2]

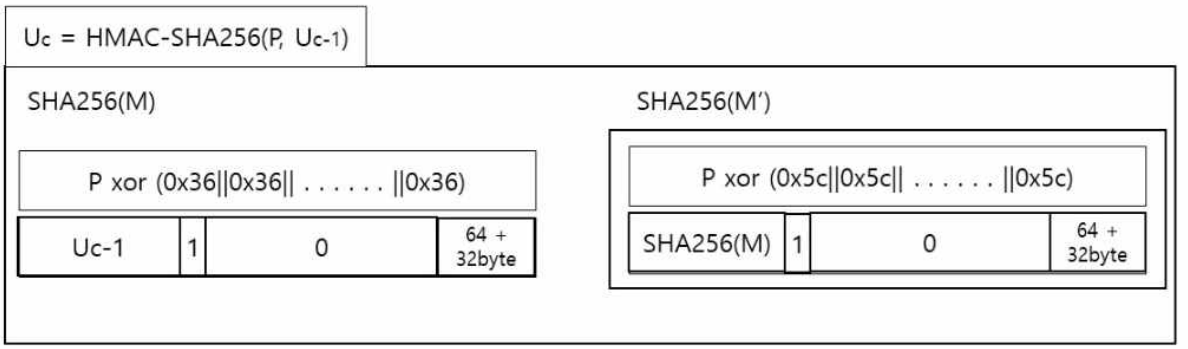
다음 절에서는 PBKDF2 과정에서 최적화 방안을 제시했다.

2.2 PBKDF2 최적화 연구

[2, 3]에서는 PRF를 HMAC-SHA160을 선택하여 PBKDF2를 구현 및 최적화 하였다. [3]에서는 PBKDF2 인증 과정에서 [OPT-01] Early exit과 HMAC optimization 과정에서의 불필요한 연산을 줄이고 Input size가 고정되는 점을 이용한 [OPT-02] Block reduction, [OPT-03] Input size가 있다. 또한 SHA-1에서 0인 값을 연산을 제외하고 연산하는 [OPT-04] Word expansion phase, [OPT-05] Zero-based Optimiation, 초기 3라운드를 고정 값으로 할당할 수 있는 [OPT-06] Three-round optimization를 제시하였다.[3]

총 6가지의 최적화 방법을 적용, OpenSSL ver 1.1.0e 와 비교하여 CPU 기준 대략 2.1배의 처리속도 향상을 보여주었다.[2 ,3]

PBKDF2-HMAC-SHA160에 대한 최적화 연구는 있었지만, SHA-1의 충돌쌍 발견으로 인해 더 높은 보안수준의 해시함수 사용이 요구된다. 본 논문에서는 [2, 3]에서 제안된 기존의 최적화 방법을 HMAC-SHA256으로 확장할 수 있는 방법과 추가적으로 성능을 향상 시킬 수 있는 방법을 제안한다.



[그림 4] PBKDF2 - HMAC- SHA256 Ui 블록 생성 과정

III. 제안하는 최적화 설계 방안

다음은 본 논문에서 고려하고 있는 최적화 방안들에 대한 설명이다.

3.1 Block-Reduction in HMAC (BR)

Block-Reduction 과정은 PBKDF2-HMAC-SHA256의 연산 과정에서 패스워드의 값은 변하지 않는다는 점과 HMAC에서 Password ⊕ IPAD 값과 password ⊕ OPAD 의 Hash값을 매번 호출하는 점을 이용한 방법이다.

최적화 방법은 Password ⊕ IPAD 값과 password ⊕ OPAD 값들의 해시 값을 미리 계산하여 Ti 블록 계산 시 필요한 Ui에서 호출한다.

즉 [그림 4]에서 Ti를 하나 생성하기 위해서는 총 4 * iteration count C 만큼 SHA256(password_Block)이 호출되지만, SHA256(password Block)값 사전 계산 시, 2 + 2* iteration count C 만큼 호출 수가 감소한다.

3.2 Input Size in HMAC (IS)

3.2절은 PBKDF2에서 Input size가 고정되는 부분을 이용하여 최적화 하는 방법이다. 즉 해시 함수의 마지막 블록의 길이 정보를 저장하는 것을 생략할 수 있는 최적화 방법이다.

PBKDF2-HMAC-SHA256에서 해시함수의 블록 길이보다 메시지가 크면 블록 크기만큼 분할하여 사용하고 password는 길이가 블록 길이보다 길어진다면 SHA256(P)값을 패스워드로 사용한다.

PBKDF2의 구조 중 Ti 값을 계산하는 부분인 [그림 3]을 살펴보자. T1을 제외한 Ti 블록은 [그림 4]의 구성과 같아진다.

즉 메시지 분할이 필요 없어지고 입력 길이인 L값이 64byte(Password와 OPAD, IPAD의 XOR 값 블록 과정) + 32byte(첫 번째 블록의 Output 길이)로 고정된다. 블록의 일정 값을 고정 값으로 할당 + 메시지 분할 과정 여부 확인 부분을 제거하여 속도를 향상할 수 있다.

3.3 Zero-Based optimization (ZB)

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

$$\sum_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$$
$$\sum_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$$
$$\sigma_0^{\{256\}}(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$$
$$\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$$

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases}$$

[그림 5] SHA256 Word-Expansion [4]

3.3절의 최적화 방법은 Word-Expansion 과정 중 W[i]값이 0이 되는 부분의 연산을 제외하는 방법이다. [그림 5]에서 W[i]가 0이 되면 연산에 영향을 주지 않는다. [그림 4]와 같이 W9, W10, W11, W12, W13, W14가 0으로 고정된다. 연산 중 Word가 0인 부분을 연산에서 제

외한다.

3.4 Block Opeartion in SHA256 (BO)

3.4절에서는 SHA256 Round 함수에서 여러 라운드의 할당과 연산을 한 라운드에 처리하는 방법이다.

For t=0 to 63:
{

$$T_1 = h + \sum_1^{\{256\}}(e) + Ch(e,f,g) + K_t^{\{256\}} + W_t$$

$$T_2 = \sum_0^{\{256\}}(a) + Maj(a,b,c)$$

$$h = g$$

$$g = f$$

$$f = e$$

$$e = d + T_1$$

$$d = c$$

$$c = b$$

$$b = a$$

$$a = T_1 + T_2$$

[그림 6] SHA256 Block Round Operation[4]

	h	g	f	e
0R	g	f	e	d + T(0,1)
1R	f	e	d + T(0,1)	c + T(1,1)
2R	e	d + T(0,1)	c + T(1,1)	b + T(2,1)
3R	d + T(0.1)	c + T(1,1)	b + T(2,1)	a + T(3,1)

[표 1] 라운드 별 파라미터 변화 값.

[표 1]은 라운드별 특정 파라미터들(h, g, f, e)의 라운드 연산 후 값을 표현한 표이며 T(N, 1)는 [그림 6]에서 N-Round temp1 값을 의미한다.

모든 파라미터의 4라운드 연산 값을 [표 1]과 같이 temp의 연산과 기존 파라미터의 연산 값으로 할당함에 따라 라운드 수를 64Round에서 적은 수의 라운드로 줄일 수 있다.

IV. 성능 측정

성능 측정 CPU는 Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 64bit 운영체제, OS는 Window10을 적용하였다.

비교 대상은 기본 구현 버전, (BR + IS)버전, (ZB + OB)버전, 모든 방법을 적용한 버전을 비교했다. 추가적으로 모든 방법을 적용한 버전과 OpenSSL 1.1.1d 버전과 속도 차이를 비교했다.

Openssl 1.1.1d	Naive	BR+IS	ZB+OB	ALL version
220	444	224	416	203

[표 2] 속도 측정 결과 표(단위 1000clock/sec)

[표 2]로 사용한 Input은 P = “password”, S = “salt”, dklen = 32, iteration count = 409600 이며 각 값은 함수를 1회 호출했을 시 걸리는 시간이다. ALL version은 대략 Naive 버전보다 약 2.2배, OpenSSL버전보다 약 8% 속도 상승이 있었다.

V. 결론 및 향후 연구

본 논문에서 제시한 4가지 방법으로 총 기본 버전 대비 2.2배 및 OpenSSL 대비 8% 성능 상향을 보였다. 본 논문은 CPU 환경에서의 적용 방안만을 제시했다. SHA256 과정에서 Mask를 이용한 GPU 메모리 최적화 및 Word-Expansion 과정에서 실질적으로 사용하는 Word만을 이용하여 최적화 하는 방법도 제시되었다. [3]

향후 연구 방향은 CPU 환경뿐만 아니라 PBKDF2 - HMAC - SHA256를 임베디드 환경 및 GPU 환경에서의 최적화 방안을 연구할 것이다.

[참고문헌]

[1] RFC2898, PKCS #5 : Password-Based Cryptography Specification Version 2.0

[2] Exploiting an HMAC-SHA-1 optimization to speed up PBKDF2, Andrea Visconti and Federico Gorla

[3] Understanding Optimizations and Measuring Performances of PBKDF2, Andrea Francesco Iuorio and Andrea Visconti

[4] FIPS PUB 180-4, FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, Secure Hash Standard (SHS)

[5] FIPS PUB 198-1, FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, The Keyed-Hash Message Authentication Code (HMAC)

Hardware Performance Counter를 이용한 Meltdown 공격 프로세스 동적 탐지 방법

김문선, 박승수, 이만희

한남대학교 컴퓨터공학과

kmoonsun95@gmail.com, parkseungsoo.kr@gmail.com, manheelee@hnu.kr

Dynamic Detecting Meltdown Attack Process by using Hardware Performance Counter

Moon-Sun Kim, Seung-Soo Park, Man-Hee Lee

Department of Computer Engineering, Hannam University

요 약

Meltdown(CVE-2017-5754)은 프로세서의 비순차적 명령어 처리의 취약점을 이용해 일반 사용자가 Kernel 메모리를 읽을 수 있는 보안 취약점이다. 해당 취약점은 하드웨어 결함이기 때문에 소프트웨어를 통한 완화가 어렵고 성능 저하를 동반한다. 본 논문은 해당 문제를 해결하기 위해 Meltdown 공격 프로세스를 동적으로 탐지할 수 있는 모델을 제안한다. 프로세서에 포함된 HPC를 이용하여 Meltdown 공격 프로세스의 하드웨어 이벤트를 측정하고, Random Forest 모델에 학습시켰다. 실험 결과, Linux 시스템에서 동작하는 Meltdown 공격 프로세스 이벤트를 100% 탐지할 수 있었으며 오탐률은 0%로 나타났다.

I. 서론

프로세서는 Out-of-Order Execution 기술을 통해 여러 가지 명령어를 실행 순서와 관계없이 병렬 처리한다. 이 기술은 오랜 기간 동안 프로세서의 성능 향상에 기여했으며 오늘날 대부분의 프로세서가 사용하고 있다. 하지만, 2018년 1월에 발표된 Meltdown 취약점을 통해 Out-of-Order Execution의 명령어 처리 방식이 Kernel 메모리를 유출할 수 있는 것이 밝혀졌다[1,2].

Meltdown 취약점은 발표되기 이전부터 완화 방법에 관한 연구가 이루어졌으며, OS 기반의 대응 패치로서 KPTI(Kernel page-table isolation)가 고안되었다[3]. 또한 프로세서 제조사들은 별도의 Microcode 업데이트를 통해 해당 취약점에 대응하였다. 하지만, 해당 패치의 적용 범위는 제한적이다. Windows는 점유율이 낮은 OS(약 4%의 점유율)들과 32bit 시스템에는 이 패치를 제공하지 않는다. 또한 더 이상 BIOS 업데이트를 지원하지 않는 구형 메인보드 기반 시스템은 Microcode 업데이트가 불가능하다. 무엇보다도, KPTI는 일반적인 환경에서 평균 13%~17%의 성능 저하를 유발한다. 따라서 Meltdown 취약점이 KPTI 패치로 완전히 해결되었다고 말하기 어렵다[4].

우리는 KPTI 없이 Meltdown 공격을 방지하기 위해 Meltdown 공격 프로세스를 탐지하는 연구를 조사하였다. 조사 결과, 대부분의 연구는 Meltdown과 함께 발표된 Spectre 공격 탐지에 집중되고 있었으며, Meltdown 공격 탐지에 관한 연구는 미미하였다. Spectre 공격 탐지마저도 이 공격에 사용되는 Cache Side-channel 공격 패턴 또는 실제 공격 상황과 차이가 있는 PoC(Proof of Concept) 프로세스를 탐지하는 방법이 주를 이루고 있다[5,6,7,8].

본 논문은 기존에 연구되었던 Meltdown과 Spectre 공격 프로세스 탐지 방법을 개선한 새로운 Meltdown 공격 프로세스 동적 탐지 모델

을 제안한다. 우리는 기존 연구에서 미비했던 탐지 시그니처를 보완하고, 실제 공격 상황에서 데이터를 수집한 후 머신러닝 알고리즘인 Random Forest를 사용하여 Meltdown 공격 프로세스 탐지 모델을 만들었다. 실험 결과, Linux 시스템에서 동작하는 Meltdown 공격 프로세스 이벤트를 100% 탐지할 수 있었으며, 오탐률은 0%로 나타났다.

II. 관련 연구

2.1 Meltdown

비순차 프로세서의 파이프라인에서 명령어는 크게 Fetch, Decode, Execute, Write, Commit 과정을 거쳐 실행된다. Fetch는 명령어의 인출 과정이며 최종 단계인 Commit은 프로세서에 명령어 실행 결과가 적용되는 단계이다. Out-of-Order Execution은 순차적으로 Fetch된 명령어에서 의존성이 없는 명령어를 찾은 뒤, 비순차적으로 실행하는 기술이다[2].

Meltdown은 Out-of-Order Execution의 진행 과정에서 Execute 단계의 취약점을 악용한다. Fig. 1은 Meltdown 공격을 간단하게 표현한 예시이다. Fig. 1의 *raise_exception*(line 1) 함수는 exception을 유발하기 때문에 후행 함수인 *access*(line 3)는 실행되지 않는다. 하지만, 프로세서 내부에서는 앞선 명령어와 의존성이 없는 *access* 함수가 비순차적으로 실행된다.

access 함수는 잘못 실행된 명령어이기 때문에 프로세서는 rollback interrupt를 통해 access 함수의 실행을 취소한다. 하지만, Intel 프로세서의 rollback interrupt는 Commit 단계 직전에 발생한다. 따라서 비순차 실행 명령어가 Execute 단계에서 참조한 오퍼랜드의 주소는 L1 cache에 적재된다. 이 값은 일반적인 방법으로 접근할 수 없지만, Cache Side-channel 공격을 수행하면 간접적으로 읽을 수 있다[1].

```
1   raise_exception();
2   // Process terminates.
3   access(array[data * page_size)]);
```

Fig. 1. Toy example of meltdown attack

2.2 Hardware Performance Counter

HPC(Hardware Performance Counter)는 프로세서에 내장된 레지스터로서, syscall과 같이 프로세스가 실행되면서 발생하는 이벤트의 횟수를 정밀하게 측정할 수 있다. 프로세스의 이벤트 정보는 악성 프로세스를 탐지하는 시그니처로 사용될 수 있다.

HPC는 하드웨어로 구현되기 때문에 시스템 오버헤드가 적다. Linux에서는 일반적으로 perf 명령어를 통해 HPC를 제어하며, 이 명령어는 모든 권한에서 사용할 수 있다.

2.3 Meltdown 및 Spectre 동적 탐지 방법

Jonas Depoix 등은 HPC를 통해 Spectre 공격이 수행하는 Cache Side-channel 공격의 프로세서 이벤트 패턴을 측정한 뒤, 인공신경망 모델에 적용하였다. 이 방법을 통해 Spectre 공격을 99.2% 정확도로 탐지하였다[6].

Taha Atahan Akyildiz 등은 Meltdown 공격을 실시간 탐지 및 격리하는 도구인 MeltdownDetector를 제안했다. 이는 Meltdown 공격이 Kernel 메모리에 접근할 때 발생하는 segmentation fault signal을 시그니처로 활용했다. MeltdownDetector는 평균 0.55%의 오버헤드로 Meltdown 공격을 탐지한다[7].

Congmiao Li 등은 HPC를 통해 일반 프로세스와 Spectre 공격 프로세스의 이벤트를 수집하였다. 이후 해당 정보를 이용하여 다양한 머신러닝 알고리즘을 학습시켰다. 이를 통해 Spectre 공격을 99.23% 정확도로 탐지했다[8].

III. 실험 설계 및 데이터 수집

Meltdown 취약점 공격을 동적으로 탐지하기 위한 연구는 꾸준히 진행되었으나, HPC를 이용한 탐지 연구는 아직 발견하지 못했다. 우리는 효율적으로 Meltdown 공격 프로세스를 탐지하기 위해 기존의 HPC를 이용한 Spectre 탐지 연구들의 미비점을 개선하고, 이를 바탕으로 Meltdown 공격 프로세스 동적 탐지 모델을 제작하였다.

Congmiao Li등과 Jonas Depoix등의 연구는 Spectre 공격 프로세스의 이벤트를 HPC를 통

해 측정하고 머신러닝을 통해 분류했다는 점이 동일하다. 또한, 두 연구는 공통적으로 Spectre PoC 코드를 통해 공격 데이터를 수집하였다. 하지만, 우리는 Spectre PoC가 자신에게 할당된 메모리를 읽는 단순한 동작을 수행하기 때문에 실제 공격과 상당한 차이를 보일 것으로 예상하고 이를 비교하는 실험을 진행했다.

실험 결과, Instructions, L1-Cache-loads, Branch-misses를 비롯한 수치들이 실제 공격과 확연한 차이를 보이는 것으로 확인했다. Table 1.은 Spectre PoC와 이를 바탕으로 제작한 실제 공격 코드의 이벤트를 측정 실험 결과이다.

우리는 이러한 한계점을 개선하기 위해 실제 Meltdown 공격을 수행하는 프로세스를 바탕으로 실험을 설계하였다. 또한, 시그니처로 사용할 프로세서 이벤트와 데이터 수집 주기를 실제 Meltdown 공격 탐지에 적합하도록 개선하였다.

Table 1. Average of 10 measurements of PoC code and real attack by HPC

Events	PoC	Real Attack
Instructions	15676678	28122147
Cache-reference	118925	200747
LLC-load-misses	78769	160156
L1-Cache-loads	3785560	7123239
L1-Cache-misses	104126	185370
Branch-instructions	1611928	3626058
Branch-misses	15848	30022

3.1 프로세스 이벤트 분석

데이터 수집을 위해 Meltdown 공격에 적합하다고 판단되는 7가지 프로세스 이벤트를 선정하였다. 해당 이벤트는 Table 1.의 Events 목록과 동일하다.

Meltdown 공격은 Out-of-Order Execution을 통해 L1 cache에 기록된 Kernel 주소를 Cache Side-channel의 일종인 Flush+Reload 공격을 통해 유출한다. 이 공격은 많은 횟수의 LLC(Last Level Cache, L3) flush 명령어(clflush)를 호출한다. 따라서 LLC-load-misses가 Meltdown 프로세스 탐지의 주요 지표로 활용될 것으로 판단된다.

또한, 이 공격은 Kernel 메모리에 접근하는 명령어를 반복적으로 비순차 실행시킨다. 이 과정에서 많은 수의 L1-Cache-loads가 발생한다. 이때 Kernel 메모리는 cache에 적재될 확률이 낮기 때문에 많은 L1-Cache-misses가 발생한다. 따라서 L1 cache 관련 이벤트는 Meltdown 공격을 식별하기 위한 좋은 지표로 판단된다.

Meltdown 공격은 Kernel 메모리 접근 함수를 포함하여 많은 횟수의 반복문을 실행하기 때문에 일반 프로세스에 비해 많은 횟수의 branch 명령어가 실행된다. 또한, Kernel 메모리 읽기 성공과 실패를 반복하는 과정에서 branch-misses가 발생한다. 따라서 branch 관련 지표를 수집할 이벤트로 선정하였다.

3.2 데이터 수집 방법 및 시나리오

우리는 Meltdown 프로세스를 동적으로 탐지할 수 있는 모델을 만들기 위해 Random Forest 알고리즘을 사용하였다. 수집한 데이터 셋은 정상 프로세스와 공격 프로세스로 부터 측정한 9,713개의 이벤트 값이다. 데이터 수집은 일반 프로세스와 공격 프로세스를 perf 명령어를 통해 각각 1초 간격으로 1000초간 측정하였다(약 16분).

학습 데이터를 획득하기 위해 총 10가지 데이터 측정 프로세스의 시나리오를 구성하였다. 시나리오 1~4는 일반 프로세스이며 5~10은 Meltdown 공격 및 유사한 명령어 패턴을 가지는 프로세스로 구성된다. 다음은 10가지 시나리오를 정리한 것이다.

1) Firefox에서 웹서핑 및 YouTube
2) htop 명령어 프로세스
3) libvirt 기반 KVM 프로세스
4) 복잡한 분기명령과 malloc(), free() 반복
5) LLC flush를 하지 않는 Meltdown 코드
6) Flush+Reload PoC
7) Meltdown 코드에 포함된 Flush+Reload
8) Meltdown PoC
9) Meltdown - 다른 프로세스 메모리 dump
10) Meltdown - Kernel Text 영역 dump

IV. 실험 결과

실험 환경은 Intel i3-4005U, 4GB, Ubuntu 16.04(4.15 Kernel)이다. 3.2에서 수집한 데이터 셋의 80%를 사용하여 Random Forest 모델을 훈련하였으며, 나머지 20% 데이터를 통해 탐지 실험을 진행하였다. 실험 결과, 일반 프로세스와 공격 프로세스를 100% 구분하였으며, 오탐률은 0%로 나타났다. Table 2.는 실험 결과를 정리한 것이다.

Table 2. Random Forest classification report

total events data	**2914**
number of normal events	1114
number of attack events	1800
total accuracy	**100%**
normal events	100%
attack events	100%
confusion matrix	
true positive rate	100%
false positive rate	0%

V. 결 론

우리는 기존에 진행된 Meltdown과 Spectre 동적 탐지 연구의 미비점을 보완하여 보다 실제적인 Meltdown 프로세스 동적 탐지 모델을 제안했다. 이 모델은 HPC를 통해 프로세스의 이벤트를 관측하고, Random Forest 분류를 통해 프로세스의 악성 여부를 판단한다. 실험 결과, 공격 프로세스 이벤트를 100% 탐지하였으며, 오탐률은 0%로 나타났다.

해당 연구 결과는 KPTI가 적용되지 않은 환경에서 효과적으로 해당 공격 프로세스를 탐지할 수 있을 것으로 기대된다. 하지만, 해당 탐지 모델은 Meltdown 프로세스만 탐지할 수 있는 미비점이 있다. 우리는 이 점을 보완하기 위해 향후 연구로 Meltdown과 Spectre를 비롯한 Cache Side-channel 공격 프로세스를 실시간으로 탐지할 수 있는 도구를 개발할 예정이다.

[참고문헌]

[1] Moritz Lipp, Michael Schwarz, Daniel Gruss1, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg, "Meltdown: Reading Kernel Memory from User Space," the 27th USENIX Security Symposium, pp.973-990, August. 2018.

[2] A. Fog, “The Microarchitecture of Intel, AMD and VIA CPUs,” May. 2017, http://www.agner.org/optimize/microarchitecture.pdf

[3] Meltdown/Spectre mitigation for 4.15 and beyond,“ https://lwn.net/Articles/744287/

[4] Brendan Gregg's Blog, http://www.brendangregg.com/blog/2018-02-09/kpti-kaiser-meltdown-performance.html

[5] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz and Yuval Yarom. "Spectre attacks: Exploiting speculative execution," arXiv preprint arXiv:1801.01203, Jan. 2018.

[6] Jonas Depoix and Philipp Altmeyer, "Detecting Spectre Attacks by identifying Cache Side-Channel Attacks using Machine Learning," WAMOS 2018, pp. 75-85, Jul. 2018.

[7] Taha Atahan Akyildiz, Can Berk Guzgeren, Cemal Yilmaz, and Erkay Savas, “MeltdownDetector: A Runtime Approach for Detecting Meltdown Attacks,” International Association for Cryptologic Research, May. 2019.

[8] Congmiao Li and Jean-Luc Gaudiot, “Online Detection of Spectre Attacks Using Microarchitectural Traces from Performance Counters,” 30th IEEE International Symposium on Computer Architecture and High Performance Computing, pp. 25-28, September. 2018.

그림 파형을 이용한 비프로파일링 기반 딥러닝 공격

원유승*, 박종연**, 한동국***

*난양기술대학교 Temasek Laboratories,
**삼성전자,
***국민대학교 수학과, 금융정보보안학과

Deep learning-based Non-Profiling attack with Picture trace

Yoo-Seung Won*, Jong-Yeon Park**, Dong-Guk Han***

*Temasek Laboratories, Nanyang Technological University,
**Samsung Electronics,
***Department of Mathematics, Department of Financial Information Security, Kookmin University.

요 약

기존에 알려진 딥러닝을 활용한 부채널 분석은 단순히 딥러닝 방법론을 적용하여 그 결과에 중점을 두었다. 하지만 부채널 분석 분야에 딥러닝의 성질을 더 효과적으로 적용하기 위한 방법론은 지금까지 고려되지 못하였다. 본 논문에서 딥러닝의 성질을 최대한 활용하기 위하여, 기존의 파형을 그림 파형으로 변환하는 방법과 그 적용 원리를 설명한다. 이에 대한 실험으로, 부채널 분석 딥러닝 database로 활용되는 ASCAD database에 대해서 최근 발표된 CHES2019 딥러닝 공격은 키 추출을 실패하였으나, 제안된 분석 기법으로 키 추출이 확연히 가능한 것을 확인하였다.

I. 서론

최근 부채널 분석에 딥러닝을 적용하여 많은 분석기법들이 활용되고 있다. 다중퍼셉트론(Multi-Layer Perceptron, MLP)과 콘볼루션 신경망(Convolutional Neural Netowork, CNN)이 부채널 분석의 프로파일링 공격에 줄곧 활용되었으며, 최근 비프로파일링 공격에서의 활용법도 CHES2019 [4]에 발표되었다. 하지만, 딥러닝을 활용한 기존의 방법들은 단순히 부채널 분석에 딥러닝 방식을 적용시키는 것에 국한되었다. 딥러닝은 초창기 연구는 MNIST database [2]를 기반으로 대부분 그 방법론들이 발전해왔다. 즉, 딥러닝을 좀 더 잘 활용하기 위해서는 MNIST database와 같은 형태로 부채널 분석 파형을 변환한다면, 딥러닝의 많은 방법론들이 적용될 것으로 기대할 수 있다. 본 논문에서는 MNIST-style로 부채널 분석 파형을 수정하여, 딥러닝을 적용한 결과를 나타낸다. 이에 대한 직관적인 원리와 통계적 기준을 새로 고안하여 딥러닝 분석에 더 적합한 형태의 새로운 공격법을 제안한다.

본 공격 결과의 성능 비교를 위하여 CHES 2019 [4]에서 제안된 방법도 수행한다. 객관성을 유지하기 위하여 딥러닝을 이용한 연구에서 부채널 분석에서의 성능 척도 기준 역할을 하는 ASCAD database [3]에 대해서 공격을 수행한다. 그 결과, [4]의 공격법으로는 키를 찾을 수 없지만, 본 논문의 제안에서는 충분히 키를 찾을 수 있음을 확인할 수 있다.

II. 사전 연구 지식

2.1 다중퍼셉트론을 이용한 부채널 공격

딥러닝의 기초라고 할 수 있는 다중퍼셉트론은 부채널에서 입력 레이어로 파형의 포인트로 설정하며, 출력 레이어로 Hamming-weight 1비트를 설정한다. 특히, 출력 레이어에서는

one-hot encoding을 적용할 수 있다 [4]. 이외에 히든레이어 개수, 학습률, 활성화 함수, 초기화 방법 등이 하이퍼 파라미터로 있을 수 있으며, overfitting이나 메모리 효율성을 위하여 batch나 dropout이 활용될 수 있다 [1].

III. 그림 파형 변환 방법과 분석

본 장에서는 그림 파형 변환 방법론과 이에 대한 분석 방법론을 설명한다.

3.1 그림 파형 인코딩 방법

인코딩 방법은 파형을 다음과 같이 그림 1에 나타낸 것처럼 y축에 대하여 Resolution을 결정하여 포인트가 존재하는 영역에는 '1'로 인코딩하며, 그렇지 않은 영역은 '0'으로 인코딩한다. 본 방식은 MNIST database에서 그림을 표현하는 방식과 동일하다.

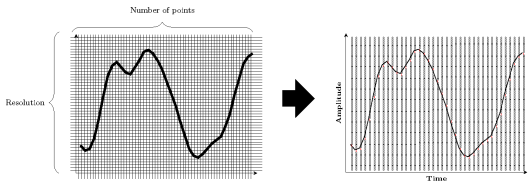


그림 1. 그림 파형 변환 방법

3.2 파형 크기 줄이는 방법

앞 절의 방법을 활용하면 파형의 크기가 상당히 증가한다. 이는 딥러닝 입력 레이어의 크기에 영향을 미치기 때문에, 입력 크기를 줄이는 방법론이 필요하다. 이는 공격하고자 하는 파형을 모두 스캔하여 인코딩이 항상 '0'인 부분을 입력으로 활용하지 않으면 된다. 그 이유는 항상 입력이 '0'이기 때문에 딥러닝을 활용한 학습에 어떠한 영향도 미치지 않기 때문이다.

3.3 옳은 키 결정 방법

[4]에서는 각 키의 학습속도를 비교하는 방법으로 training accuracy가 틀린 키에 비하여 높게 설정되면, 이를 옳은 키로 간주하였다. 하지만 본 논문의 제안 방법에서는 training에서 옳은 키와 틀린 키에 대해서 모두 accuracy가 거의 100%에 가깝게 도달하였다. 이는 딥러닝에서 흔히 관찰 할 수 있는 overfitting 현상이다. 따라서 이를 해결하기 위하여 validation accuracy에 대한 통계적 분석법을 적용하여, 분석된 키가 옳은 키일 확률을 책정한다. 본 분석법은 [4]에서와 같이 1비트 Hamming-weight를 적용한다. 즉, 비프로파일링 성질을 유지하기 위하여 예시적으로 10,000개 파형을 수집하였다면, 7,000개는 각 후보키에 대하여 training하는데, 나머지 3,000개는 각 후보키에 의해서 training된 것의 accuracy를 측정하는데 사용한다. 이를 본 논문에서 validation accuracy라고 일컫는다. Training에 사용하지 않은 validation 파형 n개에 대하여, 옳은 중간 값은 '0'과 '1'로 균등하게 분포한다고 가정한다. 베르누이 시행에 의하여 validation 파형에 대한 공격 $X_1, X_2, \cdots X_n$은 각각 독립이면, 모든 베르누이 시행의 각각 성공확률은 50%이다. 이들의 합은 다음과 같이 이항분포를 따른다.

$$\sum_{k=1}^{n} X_k \sim B\left(n, \frac{1}{2}\right)$$

n이 충분히 크다면, 다음과 같이 정규분포를 따른다.

$$\sum_{k=1}^{n} X_k \sim N\left(n \times \frac{1}{2}, \frac{1}{2} \times (1 - \frac{1}{2})\right)$$

예를 들어, n이 3,000개라면 $N(1500, 750)$을 따르며, 추측된 키에 대한 에러 확률은 누적분포 함수 $\int_{-\infty}^{x} \frac{1}{\sqrt{2\pi \times 750}} \exp\left(-\frac{1}{2} \times \left(\frac{x-1500}{750}\right)^2\right)$로 서 $P(x, 1500, \sqrt{750})$을 계산할 수 있다. 만약 추측한 키가 3,000개 중에 1,600개를 정답으로 맞췄다면, 이에 대한 확률은 $P(1600, 1500, \sqrt{750}) = 0.999869$이다. 즉, 추측된 키가 틀릴 확률은 1-0.999869=0.000131로 0.131%이다.

3.4 그림 파형이 딥러닝에 적합한 이유에 대한 직관적인 분석

그림 2 에서와 같이 그림 파형을 변환하기 전과 변환 후에 차이점을 나타낸 것이다. 그림 파형 변환 전에는 파형은 단순히 vector sequence에서 데이터 변환이 이루어졌다고 할

수 있고, 그림 파형은 위치변환이 나타났다고 할 수 있다.

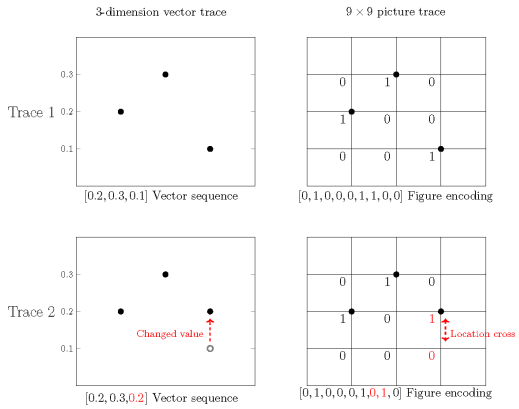


그림 2. 그림 파형 변환 전과 후 차이

이를 딥러닝과 연관시켜서 해석하자면, 그림 파형 변환 전에는 각 파형에 대하서 뉴런에 입력되는 값이 달라지는 형태이지만, 그림 파형 변환 후에는 각 파형의 데이터 값의 입력 뉴런 위치가 바뀌는 것을 일컫는다.

이를 예시적으로 딥러닝의 출력 뉴런에 미치는 직관적인 설명을 나타내면 다음과 같다. 직관적인 설명을 위하여, 예시를 그림 3, 4와 같이 간단히 작성하였다. 그림 파형으로의 변환 전과 후의 러닝 목적은 입력 데이터 1 또는 2가 들어와 1일 때는 아래 그림에서 출력 레이어의 one-hot encoding으로서 위의 값이 선정되고 2일 때는 아래 값이 선정되길 원한다고 가정한다. (w와 b는 각각 weight와 bias를 의미한다.)

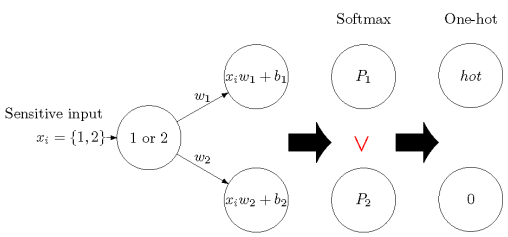


그림 3. 그림 파형 변환 전의 학습 방법 (예시)

따라서 그림 파형 변환 전(그림 3)에는 입력 값이 1일 때와 2일 때의 목적은 다음 수식으로 정리할 수 있다.

Input 1: $w_1 + b_1 > w_2 + b_2$ ($P_1 > P_2$ case)
Input 2: $2w_1 + b_1 < 2w_2 + b_2$ ($P_1 < P_2$ case)

딥러닝은 위의 수식을 풀기 위하여 학습할 것이다. 하지만, 직관적으로 단번에 w_1, w_2, b_1, b_2를 찾기란 아래의 그림 파형 변환 후와 비교해 쉽지 않다.

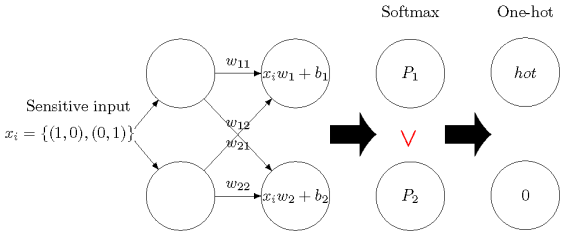


그림 4. 그림 파형 변환 후의 학습 방법 (예시)

그림 파형 변환 후(그림 4)의 식은 입력 1,2가 인코딩을 거쳐 각각 (1,0), (0,1)로 표현된다고 가정한다. 이를 학습하기 위해서는 다음과 같은 수식을 풀어내야 한다.

Input (1,0): $w_{11} + b_1 > w_{12} + b_2$ ($P_1 > P_2$ case)
Input (0,1): $w_{21} + b_1 < w_{22} + b_2$ ($P_1 < P_2$ case)

이는 직관적으로, $w_{11} > w_{21}$이고 $w_{12} < w_{22}$이면 솔루션을 얻을 수 있다. 따라서 그림 파형 변환 후에 직관적으로 딥러닝이 빠르고 정확하게 연산될 것으로 기대할 수 있다.

IV. ASCAD database에서 실험 결과

부채널 분석에서 딥러닝 공격으로 활용되는 ASCAD database에 대하여, 이차 분석을 수행한다. 비교 분석을 위하여 최근 공개된 CHES2019 [4]의 공격을 공개된 파라미터를 활용하여 수행하였다.

그림 5에서의 Figure-113과 Figure-29은 본 제안 방법의 resolution을 각각 113, 29를 사용한 것이다. 113을 사용한 이유는 ASCAD database의 최대 resolution이 113이며 29는 이에 대한 1/4 resolution을 적용한 것이다. Normal은 CHES2019 [4]에서의 공격방법론을 적용한 것이다. 각각의 자세한 파라미터는 4.1절을 활용하면 알 수 있다. 또한, 수평선으로 붉은색 2가지는 앞서 설명한 공격 기준을 설정

한 것이다. 붉은색 2가지 중 상대적으로 낮게 그려진 선은 분석된 키가 옳은 키일 확률이 99.9%이고, 높게 그려진 선은 99.99%를 의미한다. 이와 같은 기준으로 최근에 공개된 CHES2019 [4] 방법론으로 이차 분석으로 옳은 키를 찾을 수 없지만, 제안된 분석법을 활용하면 옳은 키를 명료하게 찾을 수 있음이 나타났다.

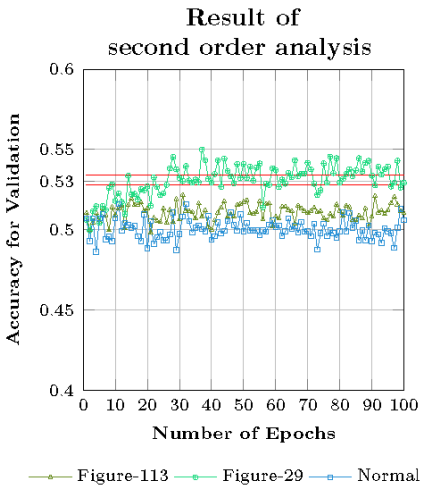


그림 5. ASCAD database에 대한 딥러닝 분석 결과

4.1 딥러닝 공격에 활용한 파라미터

분석을 위해서 Tensorflow[1]을 활용하였으며, 다중퍼셉트론 기법을 사용하였다.

표 1. ASCAD database에 대한 딥러닝 파라미터

ASCAD DB	Normal	Figure-29	Figure-113
Input size	700	3,271	10,528
Hidden layer	2		
Neuron	20 × 10	200 × 100	
Label	Hamming-weight 최하위 1비트		
Optimizer	Adam (기본 세팅)		
Dropout	N/A	각 layer마다 70%	
Activation function	ReLU, softmax, one-hot encoding		
Learning rate	0.001		
Batch	1,000	500	
#Traces	Training: 7,000 / Validation: 3,000		
Initializing	Xavier initialization		
Normalization	-1 to 1	N/A	

V. 결과 및 향후 연구

부채널 분석에 최근 많은 연구자들이 딥러닝 방법론을 활용하였음에도, 딥러닝의 원리를 활용하여 부채널 분석에 접목하는 경우는 없었다. 딥러닝의 학습 방법론을 최대한 활용하기 위하여 그림 파형 변환을 사용하였다. 최신 연구 결과인 CHES2019 [4]에서 공격법이 실패했음에도, 그림 파형을 활용한 분석법은 통계적 기준하에 옳은 키를 찾아낼 수 있었다. 이와 같은 그림 파형 분석법은 더 많은 잠재성을 지닌다. 더 많은 딥러닝 방법론을 적극 활용하거나, 그림 파형의 의미있는 포인트를 '1'이 아닌 값으로 활용하여 더 좋은 분석법을 기대할 수 있다.

[사사] "본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었음"

[참고문헌]

[1] Deep learning website. https://deeplearning.net/tutorial/tutorial.

[2] Y. Lecun, C. Cortes, and C. J.C. Burges. The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

[3] E. Prouff, R. Strullu, R. Benadjila, E. Cagli, and C. Dumans. Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database. Cryptology ePrint Archive, Report 2018/053, https://eprint.iacr.org/2018/053. (2018)

[4] B. Timon. Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019(2), pp. 17-131. (2019)

[5] Y.S. Won and J.Y. Park. Non-Profiled Side Channel Attack based on Deep learning using Picture Trace. Cryptology ePrint Archive, Report 2019/1242, https://eprint.iacr.org/2019/1242. (2019)

블록 암호에 대한 부채널 분석용 전력 파형 데이터 셋 구축*

배대현*, 하재철*, 강유성**, 김태성**, 최두호**

*호서대학교 컴퓨터정보공학부

**한국전자통신연구원

Power Trace Dataset Construction for Side Channel Analysis on Block Cipher

Daehyeon Bae*, Jaecheol Ha*, Yousung Kang**, Taesung Kim**, Dooho Choi**

*Division of Computer and Information Engineering, Hoseo University.

**Electronics and Telecommunications Research Institute.

요 약

본 논문에서는 부채널 분석 연구의 활성화를 위하여 상용 블록암호 알고리즘들에 대한 소비전력 파형 데이터 셋을 구축하며 이에 대한 정보를 제공한다. AES, SEED, 그리고 LEA 암호 알고리즘을 대상으로 부채널 분석 대응 기법이 적용되지 않은 경우와 1차 마스킹 기법을 적용한 경우의 소비전력 파형을 SCARF-AVR 보드와 SCARF-STM32 보드에서 측정하였다. 또한, 실제 배포된 파형 중 부채널 분석 대응책이 적용되지 않은 일부 파형에 대해 CPA 공격 및 MLP모델을 학습시켜 공격을 시도함으로써 분석이 가능한 파형임을 보인다.

I. 서론

암호 및 전자서명은 정보보호의 핵심 요소이며 이는 암호 알고리즘을 통해 구현 및 서비스되는데, 이때 암호 알고리즘이 이론적으로 안전할지라도 구현과정이 미흡하면 부채널 분석 공격 등의 물리적 공격에 무력화될 수 있다. 대표적인 부채널 분석 공격으로는 암호 알고리즘이 구동될 때의 소비전력을 분석해 비밀 키를 찾는 전력 분석 공격이 있으며 실제로 대부분의 블록 암호 알고리즘이 해당 공격으로 비밀 키가 노출될 수 있음이 증명되었다[1, 2]. 이러한 이유로 부채널 분석 공격은 암호 알고리즘 개발 단계에서 필수적으로 고려해야 할 사항 중 하나로 자리 잡았지만, 이에 대한 실험 환경을 구축하기에는 여러 측면에서 많은 어려움이 존재한다.

따라서 본 논문에서는 AES, SEED, 그리고 LEA 암호 알고리즘이 동작하는 과정에서 발생하는 소비전력 파형을 제공함으로써 개발자 혹은 연구자들의 파형 데이터 셋 확보의 어려움을 최소화하고자 한다. 데이터 셋 구축은 각 암호 알고리즘에 대해 공격 대응 기법이 없는 경우와 1차 마스킹을 적용한 경우로 구분하여 SCARF-AVR 보드와 SCARF-STM32 보드에서 측정하였다. 또한, 부채널 분석 대응책이 적용되지 않은 파형에 대해 CPA 및 MLP(Multi Layer Perceptron)를 학습시킨 공격을 시도함으로써 분석 가능한 파형임을 보인다.

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2018-0-00189, 고신뢰 지능정보 서비스에서 휴먼(H)-인프라(I)-서비스(S)를 연결하는 Portal Device 보안 기술 개발)

II. 파형 수집 대상 및 방법

2.1 측정 대상 암호 알고리즘

소비전력 파형 수집 대상 암호 알고리즘은 각 암호 설계 구조별 1개씩 선정하였으며, 구체적으로 SPN 구조의 AES-128, Feistel 구조의 SEED-128, ARX 구조의 LEA-128을 선택하였다[3~5]. 이때 암호 알고리즘 및 부채널 분석에 대한 대응책의 조합에 따라 총 7개 경우를 대상으로 한 블록 암호화 과정에서 발생하는 소비전력을 측정하였다. 부채널 분석 대응책으로는 마스킹(masking) 기법과 연산의 순서를 무작위로 섞는 셔플링(shuffling) 기법을 사용하였으며 이들의 7가지 조합은 Table. 1과 같다.

Table 1. Block ciphers and power analysis countermeasures

Algorithm	Countermeasure
AES-128	Unprotected
AES-128	Masking
AES-128	Masking + Shuffling
SEED-128	Unprotected
SEED-128	Masking
LEA-128	Unprotected
LEA-128	Masking

AES에 대한 셔플링 기법은 SubBytes 연산에만 적용된다. 또한, 현재 각 마스킹 기법에 대한 세부적인 알고리즘은 공개하지 않은 상태이며 마스킹을 적용한 라운드에 관한 표는 Table. 2와 같다.

Table 2. Masked round on the block ciphers.

Algorithm	Masked round
AES-128 masking	Full round (1 ~ 16)
SEED-128 masking	First and last two rounds each (1, 2, 15, 16)
LEA-128 masking	First and last two rounds each (1, 2, 23, 24)

모든 파형은 2종류의 SCARF 보드에서 각각 측정하였으며 DPA/CPA와 같은 None-Profiling 공격용 데이터와 Machine Learning 모델을 이용할 수 있는 Profiling 공격을 위한 학습용 데이터 셋을 구축하였다. 이때 DPA/CPA 분석 용도는 고정된 키를 사용하며 암호화 과정 시 측정된 2,000개의 파형을 제공한다. Machine Learning 모델 학습용은 랜덤한 키를 사용하여 암호화 과정에서 측정한 5,000개의 파형을 제공한다. 종합하면 부채널 공격 대응책 옵션을 포함한 총 7가지의 암호 알고리즘을 대상으로 2개의 보드 그리고 2개의 용도로 측정하므로 7×2×2인 28개의 파형 데이터 셋을 구축하였다.

2.2 측정 대상 보드 및 MCU

전력 분석 파형은 8bit MCU ATmega128가 탑재된 SCARF-AVR 보드, 32bit MCU STM32F411RET가 탑재된 SCARF-STM 보드를 대상으로 수집하였다. 구체적인 측정 대상으로 사용된 보드에 대한 표는 Table. 3과 같다. 모든 전력 파형은 LeCroy사의 오실로스코프 WaveRunner 104MXi-A를 사용해 측정하였으며 초당 250M 샘플의 표본화율로 수집하였다.

Table 3. Experimental board and its MCU

Board	MCU
SCARF-AVR	Atmega128 (8bit)
SCARF-STM	STM32F411RET (Cortex-M4 based, 32bit)

2.3 파형 데이터 배포 형식

배포용 파일은 압축되어 있으며, 압축된 파일의 구조는 다음과 같다.

- {identifier}-info.txt ····· 파형 정보 텍스트 파일
- {identifier}.btr····· 파형 데이터 이진파일
- {identifier}-key.txt····· 키 (16진수 문자열)
- {identifier}-plain.txt····· 평문 (16진수 문자열)
- {identifier}-cipher.txt··· 암호문 (16진수 문자열)

{identifier}는 암호 알고리즘 및 대응책에 대

한 간단한 식별자를 의미하며, btr(binary traces) 파일은 실제 파형 데이터를 포함한 이진 파일이다. 전력 파형 데이터를 포함하는 이진 파일의 구조를 나타낸 것이 Fig. 1이다.



Fig. 1. Structure of binary braces file (*.btr)

btr 파일의 각 요소는 Little-endian의 바이트 순서(byte order)를 가지며 파형의 전체 개수를 의미하는 '파형 수'와 각 파형당 샘플 수를 의미하는 '파형당 샘플 수' 그리고 각 파형의 첫 번째로 나오는 인덱스인 '파형 인덱스'는 모두 4Byte 크기의 int 형이다. 파형의 인덱스 이후에는 파형의 각 샘플 값들이 나열되며, 각 샘플, 즉 1개의 포인트는 4바이트 실수(float)형이다.

III. 파형 수집 결과 및 테스트

3.1 파형 수집 결과

총 28가지의 블록 암호 알고리즘 및 대응책의 조합에 대한 파형 측정 결과표는 Table. 4와 같다.

Fig. 2는 측정된 파형의 예시로, AVR 보드에서 동작하는 마스킹 기법이 적용된 LEA의 소비전력 파형이다. 시작과 끝부분에 각 2개의 마스킹이 적용된 라운드와, 파형 중간에 마스킹이 적용되지 않은 나머지 라운드를 확인할 수 있다.

Table 4. The result of collecting traces for each algorithms and countermeasures.

Algorithm	Countermeasure	Samples	Usage	MCU
AES-128	Unprotected	425,001	DPA/CPA	ATmega128
AES-128	Masking	625,001	DPA/CPA	ATmega128
AES-128	Masking+Shuffling	687,501	DPA/CPA	ATmega128
AES-128	Unprotected	425,001	Machine Learning	ATmega128
AES-128	Masking	625,001	Machine Learning	ATmega128
AES-128	Masking+Shuffling	687,501	Machine Learning	ATmega128
AES-128	Unprotected	500,001	DPA/CPA	STM32F411RET
AES-128	Masking	600,001	DPA/CPA	STM32F411RET
AES-128	Masking+Shuffling	630,001	DPA/CPA	STM32F411RET
AES-128	Unprotected	500,001	Machine Learning	STM32F411RET
AES-128	Masking	600,001	Machine Learning	STM32F411RET
AES-128	Masking+Shuffling	630,001	Machine Learning	STM32F411RET
SEED-128	Unprotected	812,501	DPA/CPA	ATmega128
SEED-128	Masking	2,800,000	DPA/CPA	ATmega128
SEED-128	Unprotected	785,001	Machine Learning	ATmega128
SEED-128	Masking	2,800,001	Machine Learning	ATmega128
SEED-128	Unprotected	500,001	DPA/CPA	STM32F411RET
SEED-128	Masking	665,001	DPA/CPA	STM32F411RET
SEED-128	Unprotected	500,001	Machine Learning	STM32F411RET
SEED-128	Masking	665,001	Machine Learning	STM32F411RET
LEA-128	Unprotected	1,550,001	DPA/CPA	ATmega128
LEA-128	Masking	4,850,001	DPA/CPA	ATmega128
LEA-128	Unprotected	1,550,001	Machine Learning	ATmega128
LEA-128	Masking	4,850,001	Machine Learning	ATmega128
LEA-128	Unprotected	125,001	DPA/CPA	STM32F411RET
LEA-128	Masking	285,001	DPA/CPA	STM32F411RET
LEA-128	Unprotected	125,001	Machine Learning	STM32F411RET
LEA-128	Masking	285,001	Machine Learning	STM32F411RET

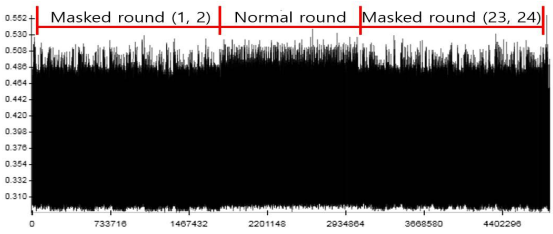


Fig. 2. Example trace of masked LEA-128

3.2 파형에 대한 공격 예시(CPA)

공개 데이터 셋 중 AVR 보드에서 DPA/CPA 용도로 측정된 대응책 없는 AES의 1라운드 첫 번째 바이트 SubBytes 연산 결과를 대상으로 CPA를 진행해 보았다. Fig. 3은 각 후보 키에 대한 최대 상관계수 그래프이며 올바른 키에서 매우 높은 상관계수를 갖는 것을 확인할 수 있다.

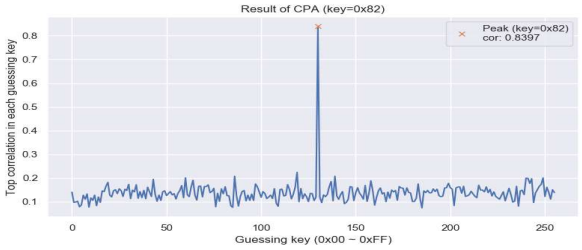


Fig. 3. Result of CPA attack on AVR-AES

3.3 파형에 대한 공격 예시(MLP)

공개 데이터 셋 중 AVR 보드에서 Machine Learning 모델 학습용으로 측정된 대응책 없는 AES를 대상으로 기본적인 MLP 모델을 학습시킨 결과이다. 이때 1라운드의 첫 번째 바이트 SubBytes 연산의 결과 값을 학습했으며, 약 99%의 정확도로 256개의 키 중 하나를 찾아낼 수 있었다.

또한, Fig. 4는 학습된 모델의 입력층과 은닉층 사이의 가중치 값들을 시각화한 결과이다. 균일한 가중치 값을 가지지 않고 특정 부분만 큰 값을 가지는 것으로 보아, 해당 부분이 실제 SubBytes 연산이 이루어지는 부분임을 추측할 수 있으므로 모델이 올바르게 학습했음을 알 수 있다.

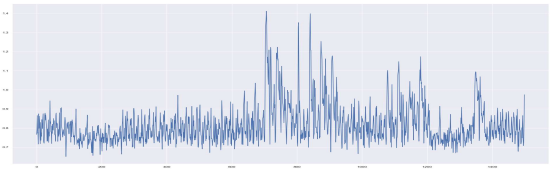


Fig. 4. Weights of input-hidden layer on MLP

구축된 파형을 대상으로 CPA 공격과 MLP 모델을 학습시켜 공격한 결과 성공적으로 비밀키를 찾을 수 있음을 확인했다. 이를 통해 부채널 분석 대응책이 설계되지 않은 암호 알고리즘은 충분히 공격 가능하다는 것을 알 수 있다.

IV. 결론

본 논문에서는 AES, SEED 그리고 LEA 블록 암호 알고리즘에 대한 전력 분석 연구용 데이터 셋을 구축하고 이를 활용할 수 있는 방법을 제시하였다. 데이터 셋은 부채널 분석 공격 방어 대책을 적용한 경우와 그렇지 않은 경우를 가정하여 다양하게 구성하였다. 구축된 파형 데이터 셋은 한국전자통신연구원이 관리하는 TrusThingz(http://www.trusthingz.org)에 공개되어 있으며 회원가입 및 로그인 후에 활용할 수 있다.

[참고문헌]

[1] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In CRYPTO '99, LNCS, 1666, pp. 388 - 397, 1999.

[2] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," CHES'04, LNCS 3156, pp. 16-29, 2004.

[3] National Institute of Standards and Technology, "Advanced Encryption Standards," NIST FIPS PUB 197, 2001.

[4] K. Jeong, J. Choi, Y. Lee, C. Lee, J. Sung, H. Park, Y. Kang, Update on SEED: SEED-192/256, ISA'09, LNCS 5576, pp. 1 - 10, 2009.

[5] D. Hong, J. Lee, D. Kim, D. Kwon, K. Ryu, and D. Lee, "LEA, A 128-bit block cipher for fast encryption on common processors," WISA'13, LNCS 8267, pp. 3-27, 2014.

비트슬라이스 구현 블록암호에 대한 향상된 부채널 분석 기법 - Case Study : Robin

김연재*, 김수진*, 한재승*, 한동국*,**

*국민대학교 정보보안암호수학과, **국민대학교 금융정보보안학과

Improved Side-Channel Analysis Techniques for Bitslice Implementation Block Ciphers - Case Study : Robin

Yeon-Jae Kim*, Soo-Jin Kim*, Jae-Seung Han*, Dong-Guk Han*,**

*Dept. of Information Security, Cryptology, and Mathematics, Kookmin University,

**Dept. of Financial Information Security, Kookmin University.

요 약

부채널 분석의 취약점에 대응하기 위해 비트슬라이스 S-Box를 기반으로 하는 암호 알고리즘에 대한 관심이 높아지고 있다. 이러한 암호는 상관전력분석에서 비선형 연산인 S-Box를 분석 지점으로 할 때 중간값을 단일 비트로 선택한다. 이때 비트슬라이스 기반 암호 Robin의 1차 상관전력분석 결과를 통해 비트별 분석 성능의 차이가 있음을 보인다. 본 논문에서는 비트별 분석 성능 차이를 예측하는 일반적 방법을 제시하고 Robin의 1차 상관전력분석 결과와 제시한 비트별 분석 성능 예측값을 비교하는 실험 결과를 제시하여 방법의 실효성을 입증한다.

I. 서론

부채널 분석(Side Channel Attack)은 부채널 정보(전력, 전자파, 소리 등)를 이용하여 암호 키 등의 비밀 정보를 분석해내는 기법이다. 그 중에서 상관전력분석(Correlation Power Analysis, CPA)은 다량의 전력정보를 이용해 소비전력과 알고리즘이 동작할 때 중간값의 상관계수를 계산하여 키를 분석하는 방법이다. 이러한 분석은 수학적으로 안전한 암호 알고리즘에 대해서도 적용되기 때문에 효율적인 부채널 분석 대응 기법을 염두에 두고 암호를 설계하는 경우가 많아지고 있다.

최근에 부채널 공격 대응 방안을 고려해 비트슬라이스 S-Box를 사용하는 암호 알고리즘들이 많이 연구되고 있다. 비트슬라이스 S-Box는 비트 간 논리연산을 이용해서 S-Box를 구현하는 방법이다. 논리연산의 특성을 이용하면 고차 마스킹을 부울린 레벨에서 적용할 수 있으므로 비교적 적은 연산량으로 마스킹을 할 수 있는 장점이 있다. 이러한 비트슬라이스 S-Box 기반의 암호 알고리즘을 상관전력분석의 관점에서 보면 일반적인 테이블 구현 S-Box 암호 알고리즘과 중간값 선택에 있어서 차이가 생긴다.

본 논문에서는 비트슬라이스 기반 암호 알고리즘의 1차 상관전력분석에서 S-Box 출력의 단일 비트를 중간값으로 선택해야 하는 이유와 그중 몇 번째 비트를 선택하는 것이 효과적인 분석 성능을 가져올지 예측하는 계산적 방법에 대해 제시하고, 비트슬라이스 S-Box 기반 블록암호 중 Robin 암호 알고리즘을 이용해 방법의 실효성에 대해 실험적으로 증명하고자 한다.

II. 관련 연구

2.1 비트슬라이스 기반 S-Box

비트슬라이스 기반 S-Box는 알고리즘을 AND(&), OR(|), XOR(⊕)의 비트 논리 연산으

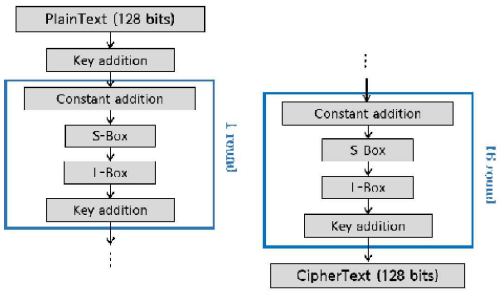
로 구현한다. 이 경우 S-Box 테이블 참조 연산이 필요하지 않으므로 메모리를 거의 사용하지 않는다. 또한, 비트 간 논리연산의 특성을 이용해 병렬적으로 S-Box 연산을 처리할 수 있으므로 연산 시간을 줄일 수 있다.

2.2 비트슬라이스 기반 암호에 대한 상관전력분석

비트슬라이스 기반 S-Box는 입출력 단위로 연산하지 않기 때문에 S-Box 출력값 전체를 중간값으로 선택하여 분석하는 것보다 출력값의 특정 비트를 중간값으로 선택하여 분석하는 것이 효율적이다.

2.3 Robin 알고리즘 구조

Robin은 128비트 데이터 입출력, 128비트 키 길이를 사용하는 SPN(Substitution Permutation Network) 구조이며 비트슬라이스 기반 암호 알고리즘이다. [그림 1]와 같이 총 16라운드로 구성되며 키 스케줄은 따로 사용하지 않는다. 한 라운드는 선형연산인 Key addition, Constant addition과 16비트 입출력 L-Box, 비선형 연산인 8비트 입출력 S-Box로 이루어진다.



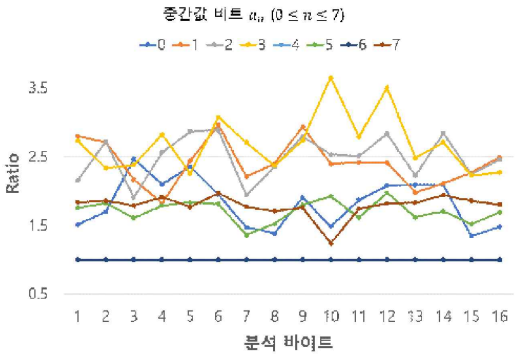
[그림 1] Robin 암호화 구조

2.4 Robin 1차 상관전력분석 결과

2.2절과 같은 이유로 Robin에 대해 1차 상관전력분석을 할 때 중간값을 단일 비트로 선택해야 한다. [그림 2]는 [표 1]의 실험환경을 기반으로 Robin S-Box 출력 비트별로 1차 상관전력분석을 적용했을 때 Ratio들을 나타낸 그래프이다. 이때 Ratio는 1등 키의 상관계수를 2등 키의 상관계수로 나눈 값이다.

[표 1] 실험환경

전력 수집 보드	Chipwisperer-Lite
분석 대상 칩	Atmel XMEGA128 (8 비트 기반 프로세서)
파형 수	2,000
샘플링 레이트	29.538 MS/s
분석 위치	1라운드 S-Box
암호 구현	8비트 기반 구현 코드
전력 모델	단일 비트



[그림 2] Robin 1라운드 키 분석 결과

[그림 2]를 보면 1차 상관전력분석에서 중간값으로 선택한 비트에 따라 분석 성능이 차이가 발생하게 된다. 본 논문에서는 효율적인 1차 상관전력분석을 위한 중간값 비트 선택방안에 대해 논의하고자 한다.

Ⅲ. 본론

먼저 상관전력분석의 성능 지표로 Ratio를 사용하며 Ratio가 높을수록 분석 성능이 우월한 것으로 판단한다.

본 절에서는 비트슬라이스 S-Box를 사용하는 암호에서 분석 성능이 좋은 중간값 비트를 판별하는 방법에 대해 제시한다. [표 2]는 사용할 기호들에 대해 간단히 정의한 것이다.

[표 2] 기호 정의

기호	설명
$\oplus$	XOR 연산
$S(x)$	n비트 입출력 S-Box의 출력값
$S_j(x)$	$S(x)$의 j번째 비트
$\max(a,b)$	a,b중 최댓값

입력의 i번째 비트에 따른 S-Box 출력의 j번째 비트의 선형성을 계산하는 방법을 다음과 같이 정의한다.

$$b_{ij} = \left| \sum_{X=0}^{2^n-1} \{ S_j(X) \oplus S_j(X \oplus 2^i) \} - 2^{n-1} \right|$$

b_{ij}의 값이 0에 가까워질수록 입력 i번째 비트에 대해 출력 j번째 비트가 선형성이 낮아지고, 반대로 b_{ij}의 값이 2^{n-1}에 가까워질수록 입력 i번째 비트에 따른 출력 j번째 비트의 선형성이 높아진다.

이때 입력 X와 동일한 값에 i번째 비트만 다른 키가 XOR된 값을 $X \oplus 2^i$이라고 보면, b_{ij}는 i번째 비트가 다른 두 키에 대한 중간값(S-Box 출력값 j번째 비트)의 선형성을 나타낸다고 할 수 있다.

상관전력분석에 있어서 옳은 키의 중간값이 2등 키의 중간값과 선형성이 낮을수록 Ratio가 높은 것을 기대할 수 있다. 그러므로 동일한 j에 대해 b_{ij}의 최댓값이 Ratio에 직접적인 영향을 준다. 특히 b_{ij}의 값이 2^{n-1}이면 차분이 2^i인 입력들이 모두 같은 j번째 비트 차분값을 가진다. 즉 상관전력분석에 있어서 차분이 2^i인 키들이 절댓값이 같은 상관계수를 가지므로 Ratio가 1이 된다.

$$B_j = \max(b_{0j}, b_{1j}, \dots, b_{(n-1)j})$$

위와 같이 B_j를 정의하고 이것을 출력의 j번째 비트에 대한 상관전력분석의 예측 성능 지표로 사용한다.

Ⅳ. 실험 결과

본 절에서는 3절에서 제시한 방법을 Robin 암호에 적용하여 입력 비트와 출력 비트 간의 선형성을 계산한다. 그 결과와 [표 1]의 실험환경을 기반으로 키 첫 번째 바이트(0x2B)에 대해 1차 상관전력분석을 수행한 결과를 비교한다.

4.1 Robin에 예측 방법을 적용

Robin 암호 알고리즘의 8비트 S-Box 입력을 $X = x_7x_6x_5x_4x_3x_2x_1x_0$, 출력을 $Y = y_7y_6y_5y_4y_3y_2y_1y_0$ 라고 하자. 이때 [표 3], [표 4]는 3절의 방법을 이용하여 선형성을 계산한 표이다.

[표 3] x_i와 y_j에 따른 b_{ij}

	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7
y_0	32	0	0	64	8	8	0	16
y_1	16	32	0	0	8	16	4	0
y_2	0	0	0	0	0	16	8	16
y_3	16	16	32	16	12	8	12	0
y_4	0	0	128	128	32	0	0	64
y_5	0	64	64	0	16	32	0	0
y_6	128	0	0	128	0	0	0	0
y_7	64	64	0	0	16	16	32	16

[표 4] y_j에 따른 B_j

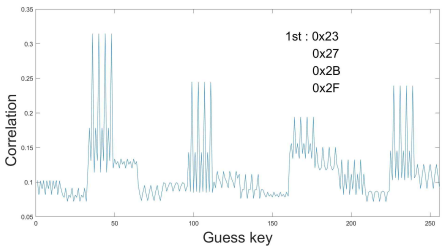
	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
B_j	64	32	16	32	128	64	128	64

4.2 이론 예측 결과와 실험 결과 비교

[표 4]의 내용을 바탕으로 B_j의 크기에 따라 출력 비트들을 3개의 집합으로 나눌 수 있다. 이때 3개의 집합은 $C_1 = \{y_4, y_6\}$, $C_2 = \{y_0, y_5, y_7\}$, $C_3 = \{y_1, y_2, y_3\}$이다.

(1) C_1에서 중간값으로 선택한 경우

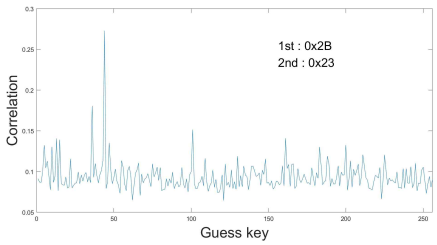
[표 3]에서 입력 x_2, x_3의 비트 변화가 출력 y_4의 비트 변화에 주는 선형성이 최대치인 128이다. 따라서 Ratio가 1이 되어 분석 성능이 좋지 않을 것이라 예측하였다. 실제 y_4을 중간값으로 선택해 1차 상관전력분석을 진행한 결과 [그림 3]을 보면 2번째, 3번째 비트만 다른 후보키들(0x23, 0x27, 0x2B, 0x2F)은 모두 같은 상관계수를 가진다. 따라서 예측과 같은 결과가 나타나는 것을 알 수 있다.



[그림 3] y_4분석 결과 추측키 별 상관계수

(2) C_2에서 중간값으로 선택한 경우

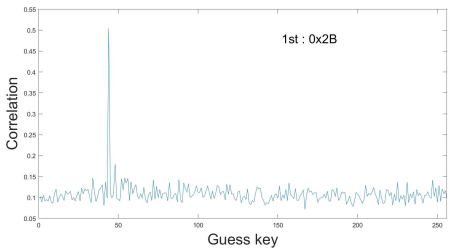
[표 3]을 보면 다른 입력 비트들에 비해서 x_3와 y_0가 선형성이 높다. 따라서 x_3비트가 다른 키 후보끼리 Ratio가 낮을 것으로 예측했다. 실제로 y_0을 중간값으로 선택하면 [그림 4]와 같이 옳은 키(0x2B)와 3번째 비트만 다른 후보 키(0x23)가 다른 후보 키들보다 비교적 더 높은 상관계수를 갖는 것을 볼 수 있다.



[그림 4] y_0분석 결과 추측키 별 상관계수

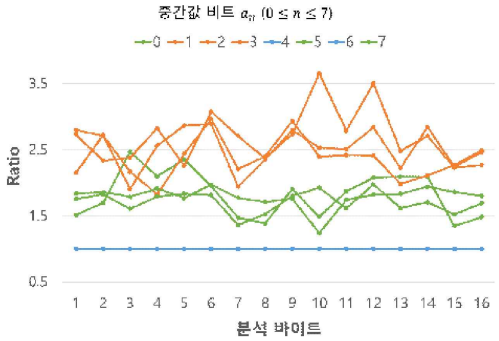
(3) C_3에서 중간값으로 선택한 경우

[표 4]에서 y_1, y_2, y_3은 다른 비트들에 비해서 입력 비트에 따른 선형성이 낮다. 따라서 C_3에서 중간값을 선택한다면 Ratio가 높을 것이라고 기대한다. [그림 5]과 같이 실제 y_1을 중간값으로 선택했을 때 옳은 키(0x2B)의 상관계수가 다른 후보 키들에 비해 월등하게 높아 분석 성능이 좋다.



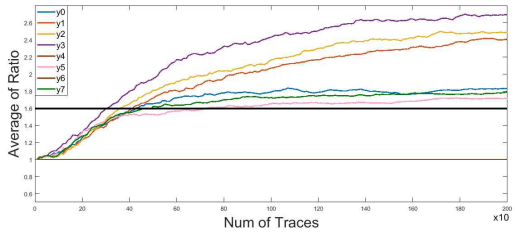
[그림 5] y_1분석 결과 추측키 별 상관계수

[그림 6]은 각 비트를 중간값으로 선택했을 때 3개의 집합별로 Ratio를 나타낸 그래프이다. 따라서 Robin에 대한 실험 결과, 선형성이 비교적 낮은 C_3의 원소(y_1, y_2, y_3)중에서 중간값을 선택하면 분석 성능이 높아짐을 알 수 있다.



[그림 6] Robin 1라운드 키 분석 결과

[그림 7]은 모든 바이트를 분석했을 때 파형 수에 따른 비트별 평균 Ratio 그래프이다. Ratio가 1.6을 넘는 최소분석 파형 수는 C_3이 약 400개이고, C_2이 약 600개로 약 1.5배 차이가 난다.



[그림 7] 파형 수에 따른 비트별 평균 Ratio

Ⅴ. 결론

본 논문에서는 중간값 비트 선택에 따른 상관전력분석의 성능에 대한 예측 방법을 제시했고, Robin 암호 알고리즘의 1차 상관전력분석에 적용하여 실제 분석 결과를 통해 제시한 방법의 실효성을 증명했다. 따라서 비트슬라이스 구조 S-Box 기반 블록 암호에 상관전력분석을 적용할 때 제시한 방법을 이용해 중간값 비트를 선택하면 더 효율적인 상관전력분석이 가능하다.

[사사] "본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다."

[참고문헌]

[1] Grosso, V., Leurent, G., Standaert, F. X., & Varıcı, K., " LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations", FSE'14, 2014

FPGA 부채널분석 시험 보드를 활용한 8비트 MCU 기반 소프트웨어 부채널분석 플랫폼 구현

강준기*, 이봉수*

*ETRI부설 국가보안기술연구소

8-Bit MCU based Software Side-Channel Analysis Platform Implementation using FPGA Side-Channel Analysis Evaluation Board

Junki Kang*, Bong Soo Lee*

*The Affiliated Institute of ETRI

요 약

본 논문에서는 FPGA 기반 부채널분석 플랫폼인 SASEBO-X를 활용하여 8비트 마이크로컨트롤러 AVR을 구현하였고, 구현된 AVR 프로세서에 AES 소프트웨어 연산을 할 수 있도록 프로그램 하였다. 본 논문의 결과물은 하드웨어·소프트웨어 혼합형 부채널분석 플랫폼으로 활용될 수 있을 것으로 기대한다.

I. 서론

20년 전 P.Kocher에 의해 부채널분석공격이 소개된 이래로 다양한 국가, 다양한 학교·연구기관에서 부채널분석공격에 대한 수많은 연구가 진행되어왔다[1]. 지금은 논리적으로 안전성을 보장할 수 있는 암호 알고리즘을 다양한 플랫폼에 구현하여 적용하는 과정에서 플랫폼이 사용하는 전력이나 방사되는 전자파 신호 등을 통해 암호 키가 노출될 수 있다는 사실은 널리 알려진 상식이 되었다. 또한 제한적인 규모를 가진 소형 임베디드 시스템과 스마트카드를 넘어 대중적으로 사용하는 스마트폰과 개인용 컴퓨터를 대상으로 부채널분석공격이 가능한 수준에 이르렀다[2].

부채널분석공격이 대중화된 연구 주제로 발전하는 과정과 동시에 부채널분석공격 실험을 위한 플랫폼 또한 함께 발전이 이루어졌다. 일본은 2007년부터 SASEBO 프로젝트를 통해 FPGA 기반의 부채널분석 시험용 플랫폼을 제공하였다[3]. 국내에서는 ETRI에서 2009년부터 개발한 Scarf 플랫폼 통해 소프트웨어 및 하드웨어 기반의 부채널분석을 쉽게 시험할 수 있게 되었으며[4], NewAE社에서 판매하는 Chip Whisperer와 같은 저가형 부채널분석 실험용 보드와 Arduino와 같은 단일 보드 마이크로컨트롤러의 대중화는 부채널분석 실험 환경을 더욱 확장시킬 수 있었다.

본 논문에서는 기존 하드웨어 기반 플랫폼과 소프트웨어 기반 부채널분석 플랫폼을 혼합한 형태로 FPGA 기반 하드웨어 부채널분석 플랫폼인 SASEBO-X 보드를 활용하여 Atmel의 8비트 마이크로컨트롤러인 AVR 코어를 구현하였다. FPGA에 구현된 AVR 코어에서 소프트웨어로 개발된 암호 알고리즘 연산을 수행할 수 있으며 부채널분석 실험이 가능할 수 있도록 하였다.

II. 배경

CC 평가기관 등에서 사용되는 특수 목적의 부채널분석 시험 플랫폼 등을 제외하고 일반적으로 사용되는 부채널분석 실험을 위한 플랫폼은 크게 하드웨어 기반의 플랫폼과 소프트웨어 기반의 플랫폼으로 나눌 수 있다. 하드웨어 기반 플랫폼은 일반적으로 FPGA를 탑재하여 만들며 SASEBO 프로젝트의 시험 보드는 대표적인 하드웨어 기반의 플랫폼으로 볼 수 있다. 소프트웨어 플랫폼은 8비트 마이크로컨트롤러 또는 32비트 프로세서를 탑재하며 ETRI의 Scarf 보드와 NewAE社의 ChipWhisperer는 소프트웨어 플랫폼의 예로 볼 수 있다.

하드웨어 플랫폼은 HDL 언어로 구현된 암호 알고리즘을 합성 후 FPGA에서 동작시키며 하드웨어로 구현된 알고리즘이 동작하는 동안 소모되는 전력 또는 방사되는 전자파를 측정하여 부채널분석 시험을 수행하게 된다.

소프트웨어 플랫폼은 프로그램 언어로 구현된 암호 알고리즘을 컴파일 후 내장형 시스템의 프로세서에서 동작시키며, 소프트웨어로 구현된 알고리즘이 동작하는 동안 소모되는 전력 또는 방사되는 전자파를 측정하여 부채널분석 시험을 수행하게 된다. 소프트웨어 플랫폼은 일반적으로 내장형 시스템에 널리 사용되는 32비트 프로세서인 ARM과 8비트 마이크로컨트롤러인 AVR이 주로 대상이 된다.

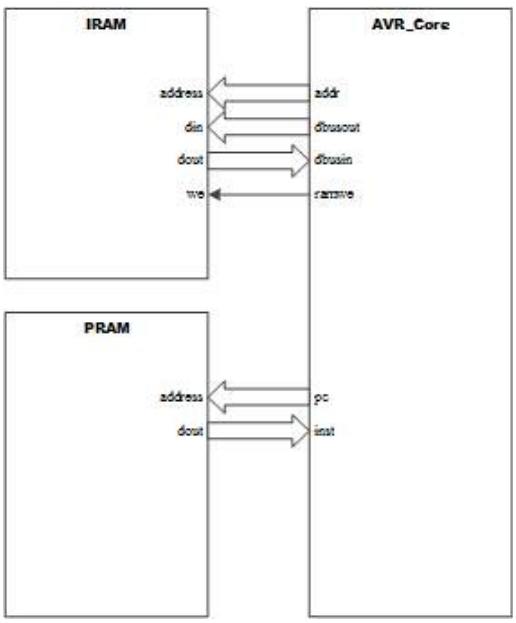
하드웨어로 구현된 알고리즘은 소프트웨어로 구현된 알고리즘보다 빠르다는 장점이 있다. 그러나 하드웨어로 구현된 알고리즘을 제품화 하는데 상대적으로 많은 비용이 들고 수정이 어렵다는 단점 때문에 스마트카드 또는 일부 마이크로컨트롤러 등에 구현된 경우를 제외하고 대중적으로 사용되는 제품군에서는 소프트웨어로 암호 알고리즘을 구현하여 탑재한다.

부채널분석과 관련된 연구의 많은 부분은 소프트웨어 플랫폼을 기반으로 구현과 실험이 진행된다. 그러나 구현된 암호 알고리즘이 예상하지 못한 부분에서 부채널분석 공격에 관련된 정보 누설이 발견되는 경우가 있거나 또는 부채널분석 시험을 위해 플랫폼 자체 또는 프로세서 자체의 수정이 필요한 경우가 있다.

III. 8비트 마이크로컨트롤러 구현과 FPGA 부채널분석 플랫폼

부채널분석을 위한 소프트웨어 플랫폼은 플랫폼에 고정된 프로세서를 수정할 수 없기에 예상하지 못한 정보누설을 파악하기 힘들뿐만 아니라 임의의 프로세서 수정이 필요할 때 수정이 힘들다는 단점이 있다. 따라서 본 논문에서는 8비트 마이크로컨트롤러인 AVR을 직접 구현하고 이를 FPGA 기반 부채널분석 하드웨어 플랫폼에 탑재하여 부채널분석 시험을 수행할 수 있는 혼합형 플랫폼을 구현하였다.

AVR 프로세서는 기본적으로 하나의 코어에 다양한 주변장치를 탑재하고 있다. AVR의 코어는 ALU와 32개의 General Purpose Register, Instruction Register/Decoder, Stack pointer등이 포함되며 Instruction을 Fetch하고 실행시키는 기능을 수행한다. 또한 AVR은 하버드 구조의 프로세서로 프로그램 메모리와 데이터 메모리가 분리되어 동시에 접근 가능한 구조를 가지고 있다[5].



[그림 1] 구현된 AVR 코어 블록도

본 논문에서 구현한 AVR 프로세서는 오픈소스를 기반하고 있다[6]. 구현된 AVR 프로세서는 부채널분석과 관련 없는 UART, SPI, GPIO등 인터페이스와 Timer와 Interrupt 제어

기 등 주변장치 회로를 모두 제거하고 기능상의 오류를 수정하여 [그림 1]과 같이 단순화 하여 구현할 수 있었다.

구현한 AVR 코어가 동작하기 위해서 프로그램 메모리에 프로그램이 필요하다. 다시 말해 AVR 코어가 AES 암호 연산을 하려면 프로그램 메모리에 AES 암호 연산을 위한 기계어(명령어)가 있어야 한다. 일반적인 AVR 프로세서는 ISP를 통해 프로그램을 내장된 플래시메모리에 다운로드 할 수 있지만 FPGA에 구현한 AVR 코어는 플래시메모리가 없다. 더욱이 최종적으로 합성된 하드웨어는 SASEBO-X 하드웨어 부채널분석 플랫폼을 대상으로 하고 있기에 가용한 자원에 한계가 있었다.

본 논문에서는 구현한 AVR 코어를 동작시키기 위해 어셈블리언어로 작성한 AES 소스코드를 컴파일한 후, 기계어로 변환된 바이너리 파일을 FPGA의 ROM으로 변환하여 사용하였다. [그림 1]의 블록도에서 프로그램 메모리에 해당하는 PRAM에 Address와 Dout 신호만 있는 이유가 PRAM을 ROM으로 변환하였기 때문이다. 본 논문에서 구현한 AVR 코어는 SASEBO-X 보드에 탑재하였으며 FPGA 합성 결과 ROM을 제외하고 1,258개의 Slice LUT, 466개의 Slice Register를 사용했다.

IV. 결론

본 논문에서는 부채널분석을 위한 하드웨어 플랫폼과 소프트웨어 플랫폼의 장단점을 비교하고 소프트웨어 플랫폼의 한계를 보완하기 위한 하드웨어·소프트웨어 혼합형 플랫폼을 구현하였다. 본 논문에서 제안한 FPGA 하드웨어 보드에 구현한 AVR 마이크로컨트롤러가 실제 AVR 프로세서와 완벽하게 일치할 수는 없다. 그러나 소프트웨어 플랫폼을 대상으로 부채널분석 시험을 하는 과정에 예상하지 못한 정보누설과 하드웨어적인 수정과 같은 부분을 해결할 수 있는 대안으로 생각된다. 향후 프로세서 자체의 부채널분석 대응기법 연구와 새로운 형태의 플랫폼 개발에 도움이 될 수 있을 것으로 기대한다.

[참고문헌]

[1] P.Kocher, J.Jaffe, and B.Jun, “Differential Power analysis”, Proc. CRYPTO, 104-113, 1999.

[2] D.Genkin, L.Pachmanov, I.Pipman, and E.Tromer, “Stealing Keys from PCs using Radio”, IARC ePrint 2015/170, 2015.

[3] http://satoh.cs.uec.ac.jp/SASEBO/en/board/

[4] 강준기, 최두호, 강유성, 김주헌, 김태성, 오경희, 최용제, “SCARF: Side-Channel Analysis Resistance Framework”, 대한전자공학회 학술대회, 2010.

[5] https://www.microchips.com

[6] http://opencores.org

차분 딥러닝 분석 성능 향상을 위한 새로운 구조 제안

권동근*, 진성현*, 김한빛*, 홍석희*, 김희석**

*고려대학교 정보보호대학원 **사이버보안전공

Improving Differential Deep Learning Analysis using Novel Neural Network Architecture

Donggeun Kwon*, Sunghyun Jin*, Hanbit Kim*, Seokhie Hong*, HeeSeok Kim**

*School of Cybersecurity, Korea University.

*Department of Cyber Security, College of Science and Technology, Korea University.

요 약

차분 딥러닝 분석은 Timon에 의해 제안된 최초의 딥러닝을 활용한 논프로파일링 부채널 분석 기법으로, 추측 키의 라벨로 분류하는 데 필요한 학습량을 구별자로 이용하여 비밀키를 복구한다. 하지만, 제안된 기법은 동일한 신경망을 반복적으로 학습해야 하여 학습에 필요한 비용이 키 추측의 수만큼 증가하면서도 중간 과정에서 결과를 관찰하기에 매우 어려운 구조였다. 본 논문은 기존 기법의 알고리즘을 변경하여 동일한 모델을 반복적으로 학습하는 구조를 피하고, 한 번의 학습으로 수행할 수 있으면서도 유사한 분석 성능을 가지는 새로운 알고리즘과 신경망 구조를 제안한다. 또한, ChipWhisperer-Lite로부터 수집한 전력 파형과 ANSSI-FR에서 공개한 데이터셋 ASCAD에 대한 분석 결과를 보임으로써 제안 기법의 성능을 검증한다.

I. 서론

공격자가 물리적 장비를 접근할 수 있는 환경에 도래함에 따라 장비에 탑재된 암호알고리즘이 수학적인 안전성뿐만 아니라 물리적 공격에 대한 안전성 또한 요구받게 되었다. 암호 장비에 대한 대표적인 물리적 공격으로는 부채널 분석이 있으며, 부채널 분석을 통하여 실제 운용되는 장비에 대한 공격 사례가 등장하고 있다[1].

실제 장비에 대한 강력한 공격인 부채널 분석과 실제 데이터 분석에서 높은 성능을 보여주는 딥러닝을 결합한 연구가 최근 이루어지고 있다. 딥러닝 기반의 부채널 분석은 주로 프로파일링 환경을 가정한다. 프로파일링 장비를 통해 획득한 부채널 정보를 이용하여 신경망을 학습한 후, 학습된 신경망을 이용하여 공격 장비의 부채널 정보를 분류하여 비밀정보를 복원한다. 딥러닝을 활용한 부채널 분석을 이용한다면 전처리 과정없이 마스킹, 하이딩 대응기법을 무력화 가능하다는 것이 알려졌다[2].

프로파일링 환경에서의 연구로 치중된 상황에서 2019년 CHES에서 논프로파일링 환경에서 딥러닝을 활용한 부채널 분석 기법이 Timon에 의해 제안되었다[3]. Timon이 제안한 기법은 기존 다중 파형 분석에서 주로 사용되던 소비 전력 파형과 라벨이라 하는 중간값에 대한 전력 모델값(예상 소비 전력) 사이의 상관관계를 구별자로 이용하는 대신에 학습의 경향성을 구별자로 이용한다. 딥러닝을 이용하여 학습 데이터를 분류하는 모델이 특정 손실(Loss)이나 정확도(Accuracy)에 도달하는 에폭(Epoch) 수 또는 해당 모니터링되는 측정 지표 등 학습의 경향성을 구별자로 사용하여, 올바른 키와 틀린 키를 구분할 수 있다는 특징을 이용한다. Timon의 연구는 논프로파일링 환경에서도 딥러닝을 활용한 분석이 가능하다는 것을 보여주는 첫 번째 사례이다.

하지만, 제안된 기법을 수행하기 위해서는 키 추측의 가짓수(k)만큼 동일한 모델을 에폭(e)만큼 신경망 학습을 수행해야 하여 적어도 $k \times e$번의 에폭으로 학습 연산에 대한 비용이 기존 연구들과 비교적 높으며, 256개의 같은 신경망을 동일한 값으로 초기화 작업이 필요하여 데이터 입출력 작업에 대한 비용도 무시할 수 없다. 또한, 몇 번의 에폭이 수행되어야 키를 구분할 수 있을지 모르기 때문에, 각 키 추측에 따른 모델들을 각 1 에폭씩 학습하며 학습의 경향성을 관찰하게 되면 이에 대한 데이터 입출력 작업 비용은 무시할 수 없을 정도로 커지게 된다.

본 논문에서는 기존에 제안된 기법의 문제점들을 개선하기 위해 새로운 신경망 구조를 제안한다. 제안하는 신경망 구조는 동일한 신경망을 키 추측의 수만큼 병렬적으로 연결시켜 서로 다른 그레이디언트(Gradient)를 통한 학습이 이루어지도록 하여 256개

를 각각 학습시키는 것과 동일한 효과를 내도록 한다. 제안하는 신경망을 이용할 경우, 기존 제안된 기법에서 키 추측만큼 동일한 모델을 학습할 필요 없이 1번의 학습으로 수행 가능하며, 데이터 입출력에 소모되는 비용도 감소한다. 또한, 에폭마다 모든 추측에 대한 학습 경향성을 모니터링할 수 있어, 공격 수행 중에 학습량을 조절할 수 있다는 장점이 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구, 논프로파일링 딥러닝 기반 부채널 분석 기법인 차분 딥러닝 분석을 소개한다. 3장에서는 본 논문에서 제안하는 기법에 관해 설명하며, 이어지는 다음 장에서 제안 기법을 실제 전력 파형에 대해 실험하여 검증한다. 마지막 5장에서는 결론과 추후 연구의 방향성에 대해 언급하며 마친다.

II. 관련 연구 - 차분 딥러닝 분석

본 장에서는 기존에 제안된 딥러닝 기반 논프로파일링 부채널 분석 기법인 차분 딥러닝 분석(Differential Deep Learning Analysis)에 대하여 소개한다.

차분 딥러닝 분석은 딥러닝을 활용한 논프로파일링 환경에서의 최초의 분석 기법으로, 공격자 가정은 기존 비프로파일링 부채널 분석에서의 공격자 가정과 동일하다. 공격자는 부채널 정보(소비 전력)와 해당 부채널 정보에 대한 평문을 수집할 수 있는 환경으로, 프로파일링 공격과 다르게 사전 정보를 통해 신경망을 학습할 수 없는 환경이다. 따라서, 공격자는 차분 전력 분석이나 상관 전력 분석과 같이 부채널 정보와 평문을 통해서 비밀키를 추출해야 한다. [Alg 1]은 차분 딥러닝 분석에 대한 명세이다.

Alg 1. 차분 딥러닝 분석 (Case-AES)

Input : N 개의 부채널 정보 $(T_i)_{1 \le i \le N}$와 그에 대응되는 평문 $(P_i)_{1 \le i \le N}$, 신경망 Net.

1. $X = (T_i)_{1 \le i \le N}$ // 학습 데이터 설정
2. **for** $k = 0 \ldots 255$ // 추측 비밀키
3. $init(Net)$ // 가중치 초기화
4. $(H_{i,k})_{1 \le i \le N} \leftarrow M((P_i)_{1 \le i \le N}, k)$
5. $Y_k = (H_{i,k})_{1 \le i \le N}$ // 라벨 설정
6. $DL_k(Net, X, Y_k, e)$ // 신경망 학습 e회
7. **end for**
8. **return** $(DL_i)_{0 \le i \le 255}$ 결과 분석

공격자는 우선 수집한 파형을 학습 데이터로 설정한다. 다음으로 각 키 추측에 따라서 아래 과정을 반복적으로 수행한다. 첫 번째로 신경망 Net의 파라미터인 가중치와 바이어스 등을 초기화한다. 초기화는 모든 키 추측에 대한 신경망이 동일한 파라미터를 공유하기 위해서 수행한다. 두 번째로 추측한 비밀키 값 k을 통해 중간값을 추측하고, 추측한 중간값과 소비 전력 모델을 통해 예상 소비 전력값 $H_{i,k}$을 계산한다. (M은 평문과 키를 조합한 중간값의 소비 전력 모델을 계산하는 함수이다.) 세 번째로는 앞서 계산한 예상 소비 전력값을 학습 데이터의 라벨로 설정한다. 마지막으로 학습 데이터 X와 그에 대한 라벨 Y_k을 이용하여 Net_k을 e 에폭만큼 학습시킨다. 모든 키 추측에 대한 Net의 학습을 완료하였다면, DL 결과들을 통해 비밀정보를 획득할 수 있다. [Alg 1]의 8번째 줄에서 "$(DL_i)_{0 \le i \le 255}$ 결과 분석"이라고 명시된 내용에 대한 구체적인 방법으로는 3가지가 제안되었으며, 첫 번째는 다층 퍼셉트론의 첫 번째 은닉층의 가중치 기반의 민감도 분석(Sensitivity Analysis based on MLP First Layer Weights)이며, 두 번째로는 신경망의 입력 기반의 민감도 분석(Sensitivity Analysis based on Network Inputs)이다. 마지막으로 딥러닝에서 주로 이용하는 손실과 정확도 지표(Loss and Accuracy Metrics)가 있다.

III. 제안 기법

본 장에서는 기존 차분 딥러닝 분석 기법의 성능을 향상할 수 있는 알고리즘과 새로운 신경망 구조를 제안한다. 제안하는 신경망의 구조는 기존 기법에서 이용되는 신경망 하나를 키 추측의 수만큼 복제하여 입력층과 출력층만 공유하고 그 이외의 파라미터는 서로 연결되지 않도록 local-connected 하게 설계한다. 제안하는 기법에 대한 신경망 구조의 부분은 [Fig 1]에서 확인할 수 있다. 제안하는 신경망 구조를 이용할 경우에는 제안하는 [Alg 2]을 통한 논프로파일링 딥러닝 기반 부채널 분석을 수행할 수 있다.

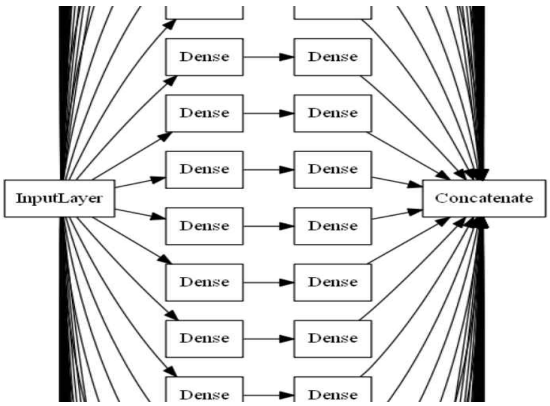
Alg 2. 제안 기법 (Case-AES)

Input : N 개의 부채널 정보 $(T_i)_{1 \le i \le N}$와 그에 대응되는 평문 $(P_i)_{1 \le i \le N}$, 신경망 Net.

1. $X = (T_i)_{1 \le i \le N}$ // 학습 데이터 설정
2. **for** $k = 0 \ldots 255$ // 예상 소비 전력 계산
3. $(H_{i,k})_{1 \le i \le N} \leftarrow M((P_i)_{1 \le i \le N}, k)$
4. **end for**
5. $Y = (H_{i,k})_{1 \le i \le N, 0 \le k \le 255}$ // 라벨 설정
6. **for** $i = 0 \ldots e$
7. $DL(Net, X, Y, 1)$ // 신경망 학습 1회
8. DL 결과 분석 // 중간 모니터링
9. **if** find secret key K
10. **break**
11. **end for**
12. **return** secret key K

[Alg 2]에서 볼 수 있는 것과 같이, 제안 기법을 이용한다면 e번의 신경망 학습으로 논프로파일링 공격을 수행할 수 있음을 알 수 있다. 4번째 줄까지는 고전적인 부채널 분석 기법, 차분 전력 분석을 포함

한 차분 딥러닝 분석 모두 수행하는 과정이며, 5번째 줄에서 라벨 Y는 차분 딥러닝 분석과 다르게 $(N \times 2)$ 크기의 행렬이 아닌 $(N \times 256)$ 크기의 행렬이다. 출력층의 각 노드는 각 키 추측에 대한 결과를 대변하므로 $(N \times 256)$ 크기의 라벨을 설정하게 된다. [Alg 2]의 6부터 11번째 줄에서 확인할 수 있는 것처럼 1 에폭씩 신경망을 학습하면서 나오는 학습 결과(손실, 정확도, 그레이디언트)의 경향성을 판단하며 키를 판별한다. 8번째 줄의 DL 결과 분석은 기존 제안된 기법과 동일한 3가지 방법을 이용할 수 있다. [Alg 1]에서는 $k \times e$ 번의 학습을 수행하지만, [Alg 2]에서는 e번의 학습만을 수행한다.



[Fig 1] Part of proposal neural network architecture with 256 small neural networks

IV. 실험 결과

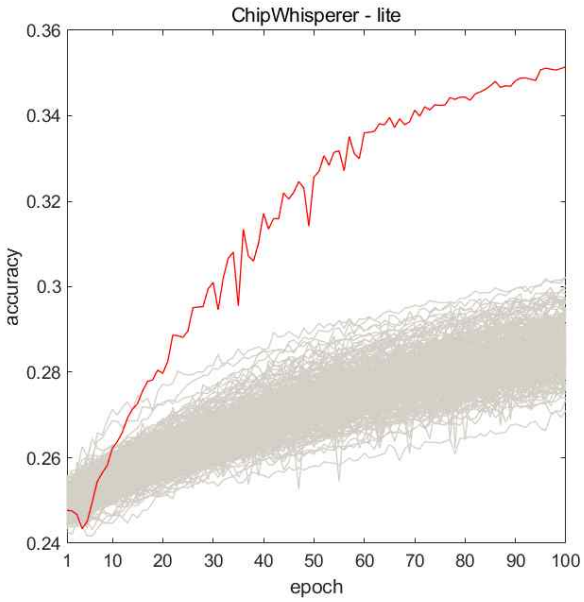
제안 기법에서 이용한 신경망의 구조는 Timon이 제안한 구조 MLP_{exp}를 기본으로 생성하였다. 본 논문에서 사용한 신경망의 하이퍼 파라미터는 첫 번째 은닉층은 20개의 노드에 ReLU 활성 함수, 두 번째 은닉층은 10개의 노드에 ReLU 활성 함수, 마지막 층은 1개의 노드에 Sigmoid 활성 함수로 구성하였고, 이를 256개를 결합하여 사용하였다. 손실 함수로는 Binary Cross-Entropy 함수, 키 추측을 위한 지표로는 합 제곱 오차(Sum squared error)를 이용하였다. [Table 1]은 기존 기법과 제안 기법에 대한 수행 시간 및 메모리를 측정하여 비교한 결과이다. Total training time은 신경망 설정 및 학습, 데이터 입출력 작업을 포함한 수행 시간이다. 하지만, 공격 수행 시의 프로그램 메모리 사용량을 측정한 결과는 제안 기법이 3.25배 정도 높은 것을 확인하였다.

Table 1. Comparison of training time on ChipWhisperer-Lite when $k = 256$, $e = 100$

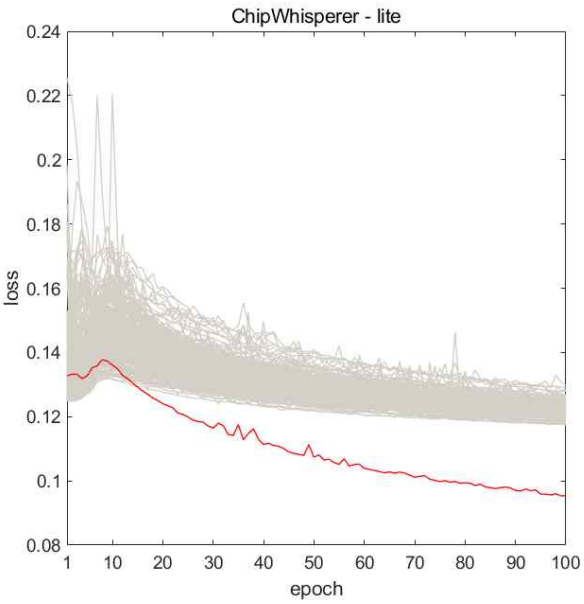
	Single epoch time	Total training time	Memory cost
DDLA	0.34 s	8,502.5 s	2,535 MB
Proposal	42 s	4,946.1 s	8,247 MB

4.1 ChipWhisperer-Lite

본 논문의 실험에 이용한 첫 번째 데이터셋은 동작 주파수 7.37MHz의 Atmel XMEGA128 (8-bit 프로세서) 보드에서 임의의 평문에 대한 대응기법이 적용되지 않은 AES-128 1 Round SubBytes 연산을 수행할 때, ChipWhisperer -Lite를 이용하여 클락당 샘플수 4(point/clock)로 총 800포인트의 10,000개 전력 파형을 수집한 데이터이다. 그중 9,000개만을 학습 데이터로 활용하였다. 수집한 데이터에 대한 제안 기법을 적용한 실험 결과는 [Fig 2, 3]에서 확인할 수 있으며, 정확도, 손실 지표 모두에서 성공적으로 제안 기법을 이용하여 비밀키를 복원 가능하다는 것을 확인할 수 있다.



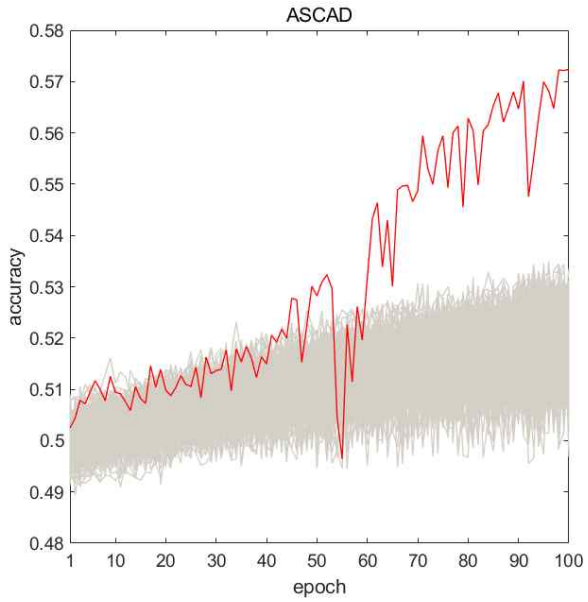
[Fig 2] Attack on ChipWhisperer-Lite (Accuracy)



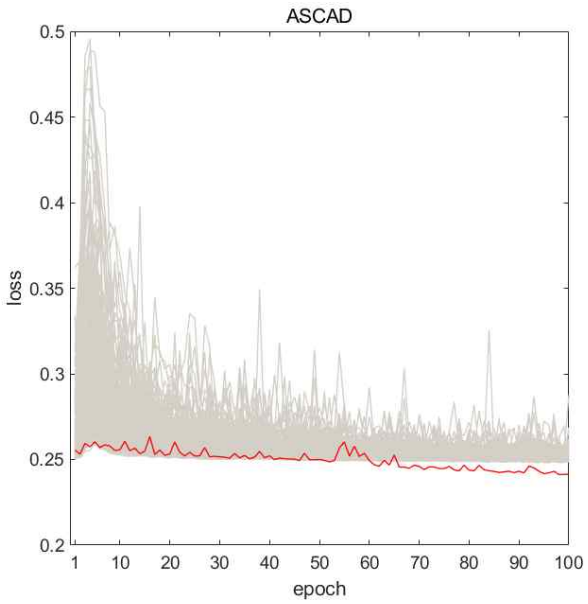
[Fig 3] Attack on ChipWhisperer-Lite (Loss)

4.2 ANSSI SCA Database (ASCAD)

두 번째 데이터셋은 ANSSI-FR에서 부채널 분석 성능 벤치마킹을 위해 공개한 데이터셋 ASCAD[4]으로 ATmega8515에서 동작하는 마스킹 대응기법이 적용된 AES 구현에 대한 전자기파 파형을 2Gs/s로 수집한 데이터이며 총 6만 개의 파형이 존재한다. 본 실험에서는 *ASCAD.h5* 파일에 존재하는 3번째 바이트의 SubBytes 연산에 대한 700포인트 길이의 전자기파 파형 2만 개를 이용하였다. 오픈 데이터셋에 제안 기법을 적용한 실험 결과는 [Fig 4, 5]에서 관찰할 수 있으며, 정확도, 손실 지표를 통해 제안 기법을 이용한 비밀키 복구가 모두에서 성공적으로 수행 가능하다는 것을 확인할 수 있다.



[Fig 4] Attack on ASCAD (Accuracy)



[Fig 5] Attack on ASCAD (Accuracy)

V. 결론

본 논문에서는 기존에 제안된 차분 딥러닝 분석을 실제 수행함에 있어서 발생하는 문제점들을 극복하고자 개선 방안들을 제시하였다. 그 결과로 많은 데이터 입출력 비용 및 학습 시간이 감소하였고, 전체적인 공격 수행 시간을 감소시켰다. 하지만, 본 논문의 제안 기법은 신경망의 파라미터가 256배 증가하여 메모리에 대한 비용이 증가한다는 문제점이 존재한다. 공격자는 메모리 자원에 따라 제안 기법을 선택하여 수행 시간을 단축할 수 있다. 학습의 불안정성은 하이퍼 파라미터 탐색으로 해소할 수 있으며, 높은 메모리 비용 문제는 최적화 구현 및 알고리즘 개선과 같은 추후 연구로 진행될 것으로 예상한다.

VI. Acknowledgement

본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

[참고문헌]

[1] Kim, Tae Won, Tae Hyun Kim, and Seokhie Hong. "Breaking korea tansit card with side-channel analysis attack-unauthorized recharging." 18th Annual BlackHat Asia Conference, Singapore. 2017.

[2] Cagli, Eleonora, Cécile Dumas, and Emmanuel Prouff. "Convolutional neural networks with data augmentation against jitter-based countermeasures." International Conference on Cryptographic Hardware and Embedded Systems. Springer, Cham, 2017.

[3] Timon, Benjamin. "Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis." IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019.

[4] Emmanuel Prouff and Remi Strullu and Ryad Benadjila and Eleonora Cagli and Cecile Dumas. "Study of Deep Learning Techniques for Side-Channel Analysis and Introduction to ASCAD Database", Cryptology ePrint Archive: Report 2018/053, 2018.

국내 이통사 USIM 인증 알고리즘 부채널분석 취약성

이태호*, 한동국*,**

*국민대학교 정보보안암호수학과, **국민대학교 금융정보보안학과

Side-Channel Analysis Vulnerability of Domestic Mobile Carrier USIM Authentication Algorithm

Tae-Ho Lee*, Dong-Guk Han*,**

*Dept. of Information Security, Cryptology, and Mathematics, Kookmin University,

**Dept. of Financial Information Security, Kookmin University.

요 약

국내에서 사용하는 모든 휴대전화는 전화를 주고받기 위해 고유한 개인정보를 보관하는 USIM을 장착하고 있다. USIM이 보관하는 개인정보가 고유하므로 어느 휴대전화에나 같은 USIM을 장착하면 같은 번호로 이용할 수 있다. 이는 개인정보를 도용당하면 도용한 사람이 어느 휴대전화를 사용해도 쉽게 내 번호를 이용할 수 있음을 의미한다. 이것을 방지하기 위해 USIM은 MILENAGE 알고리즘을 탑재하여 인증 및 키 일치, 암호화, 무결성 확인 등을 수행한다. 안전하게 사용하기 위해 제시한 알고리즘이지만 본 논문에서는 이 알고리즘을 이용한 국내 이통사 USIM 부채널 분석을 통한 마스터키 복원 방법 및 실험 결과를 소개한다.

I. 서론

SIM(Subscriber Identify Module)은 이동 전화 기술 규격에 제한 없이 어느 지역에서나 SIM이 장착된 휴대전화로 기존의 번호를 자유롭게 사용할 수 있게 하는 모듈이다. 국내에서는 이동 전화 기술 규격 중 하나인 GSM(Global System for Mobile Communication)을 발전시킨 WCDMA(Wideband Code Division Multiple Access)를 3세대 이동통신 UMTS(Universal Mobile Telecommunication System)의 규격으로 도입했다. 따라서 UMTS에서도 사용할 수 있는 SIM 카드인 USIM(Universal SIM) 카드가 만들어졌다.

최근에 만들어진 모든 휴대전화는 USIM 카드를 사용하는데 국내 USIM 카드는 개인정보를 가지고 있어 하나의 카드를 어느 휴대전화에 장착해도 기존에 사용하던 것과 같은 번호로 사용할 수 있다.

따라서 USIM 카드를 복제할 수 있다면 쉽게 개인정보를 도용할 수 있어 복제되지 않도록 안전하게 사용하는 것이 중요하다.

본 논문에서는 USIM 카드의 인증 과정에서 발생하는 전력 정보를 이용해 인증 단계에서 사용되는 마스터키를 찾는 방법을 제시하고, 실제 USIM 카드 분석 결과를 소개한다.

II. 관련 연구

[표 1]은 본 논문에서 사용하는 개념과 그에 대한 설명이다.

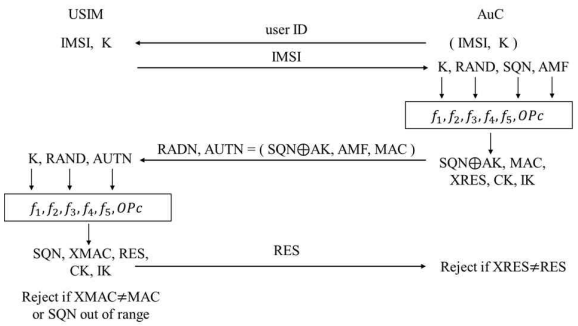
[표 1] 개념들과 각각의 설명

개념	설명
AuC	Authentication Center
K	USIM의 고유한 마스터키
RAND	128-bit 난수
SQN	48-bit Sequence Number

AMF	16-bit Authenticated and key Management Field
MAC	64-bit Message Authentication Code
XMAC	64-bit Expected MAC
RES	64-bit Response
XRES	64-bit Expected RES
AK	48-bit Anonymity Key
CK	128-bit Cipher Key
IK	128-bit Integrity Key
r_i	Integers $\in [0,127]$
c_i	128-bit Constants
OPc	128-bit Operand Code
E_K	AES-128 (마스터키 : K)

2. 1 UMTS Infrastructure

다음 [그림 1]은 USIM과 인증 센터의 양방향 인증 과정이다.



[그림 1] UMTS Infrastructure

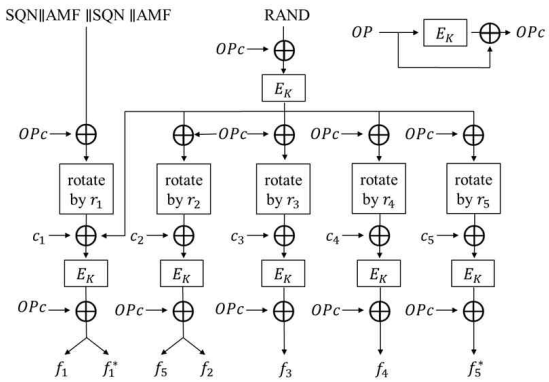
인증 센터는 USIM을 발급할 때 해당 USIM에 고유한 IMSI(International Mobile Subscriber Identity) 값과 고유한 마스터키 K를 할당하고 서버에 저장해둔다. 이후 USIM이 처음 동작하면 인증 센터와의 인증을 수행하기 위해 IMSI 값을 인증 센터에 보낸다. 인증 센터는 IMSI 값에 대응되는 K와 RAND, SQN, AMF로 MILENAGE 알고리즘을 수행해 AK, MAC, XRES, CK, IK를 생성한다. 이 중 AK, SQN, AMF, MAC으로 AUTN(128-bit) 값을 구성하고 RAND(128-bit) 값과 함께 USIM에 전송한다.

USIM은 K와 RAND, AUTN으로 MILENAGE 알고리즘을 수행해 SQN, XMAC, RES, CK, IK를 차례로 만들어낸다. 이때 XMAC과 전달받은 MAC 값이 다르거나 SQN 순서번호가 지정 범위 내에 존재하지 않다면 더 이상의 알고리즘 진행을 그만두고 인증 실패를 반환한다. 반면에 XMAC과 MAC 값이 같고 SQN 순서번호가 지정 범위 내에 존재하면 RES를 인증 센터에 전송한다.

인증 센터는 해당 RES와 기존에 생성해둔 XRES를 비교하여 같다면 인증을 성공적으로 마치고, 같지 않다면 인증 실패를 반환한다.

2. 2 MILENAGE 알고리즘

[그림 2]는 MILENAGE 알고리즘의 인증 및 키 일치 절차에서 사용되는 7가지 함수를 나타낸 그림이다.



[그림 2] MILENAGE 알고리즘 - 인증, 키 일치

[표 2]는 [그림 2]의 함수들의 생성 값에 대한 설명이다.

[표 2] 각 함수별 생성 값

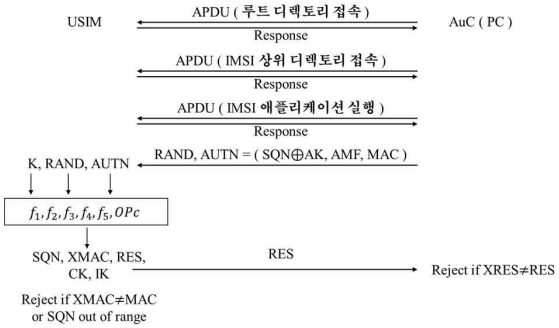
f1	인증 시 사용하는 MAC-A
f2	인증 센터에 인증받기 위해 사용하는 RES
f3	암호화 키 CK
f4	무결성 키 IK
f5	익명 키 AK
f1*	재동기화 시 사용하는 MAC-S
f5*	재동기화 시 사용하는 익명 키 AK

본 논문에서 인증 과정 중에 사용하는 함수는 f1과 f5이다.

III. MILENAGE 알고리즘에 대한 부채널 분석

3. 1 실험 환경 구축

인증 과정의 전력 파형을 수집하기 위해 기존의 UMTS 구조를 조작해 [그림 3]과 같이 동작하는 스크립트를 구성한다.



[그림 3] Modified UMTS Infrastructure

USIM은 루트 디렉토리, IMSI 애플리케이션의 상위 디렉토리, IMSI 애플리케이션을 순서대로 진입해야 올바르게 인증을 수행할 수 있다.

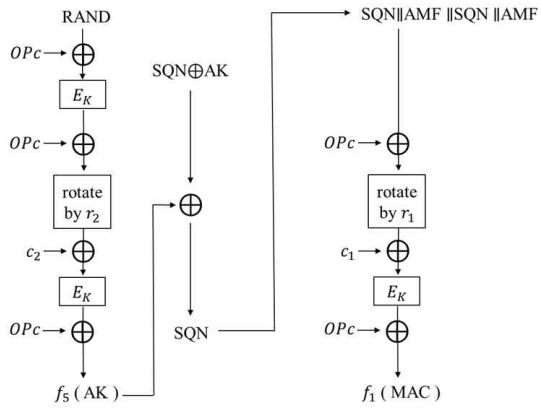
이때 진입을 위한 통신 APDU(Application Protocol Data Unit)는 [그림 4]와 같이 구성한다.

```
00 A4 08 04 02 2F 00   select file with 02(0x02)-byte argument 2F 00
00 C0 00 00 **         get response of **-byte
00 B2 01 04 **         read records of **-byte
00 A4 04 04 10 A0 ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** FF
                       select file with 16(0x10)-byte argument A0**FF
00 C0 00 00 35         get response of 53(0x35)-byte
00 A4 00 04 02 6F 07   select file with 02(0x02)-byte argument 6F 07
00 C0 00 00 **         get response of **-byte
00 88 00 81 22 10 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
                  10 BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB
                       run authentication on 16(0x10)-byte RAND=AA**AA, AUTN=BB**BB
```

[그림 4] 통신 APDU

[그림 4]에서 빨간색으로 표시된 부분은 USIM에 따라 변할 수 있는 값을 나타낸다. 즉, IMSI 애플리케이션의 상위 디렉토리 진입 APDU는 USIM마다 다르게 나타날 수 있다. 올바른 진입을 위해 3행의 명령문을 통해 레코드를 읽어 'A0 ** FF' 부분을 진입 APDU 값으로 사용한다.

3. 2 분석 설계



[그림 5] 인증 센터 인증 과정

[그림 5]는 USIM이 인증하고자 하는 인증 센터가 올바른 인증 센터인지 확인하기 위해 AK, XMAC 값을 계산하는 과정이다.

이 과정에서 발생하는 전력을 이용해 다음과 같은 방법으로 K, OPc, r, c를 복구할 수 있다.

(1) OPc ⊕K 복구

알고 있는 RAND를 평문처럼 취급해 첫 번째 E_K의 1라운드 SubBytes를 공격하면 $OPc \oplus K$를 복구할 수 있다.

(2) K 복구 (OPc 복구)

(1)에서 복구한 값을 이용하면 1라운드 출력 값을 알 수 있다. 따라서 E_K의 2라운드 SubBytes를 공격해 두 번째 라운드키를 복구할 수 있고, 이를 역연산하면 마스터키 K를 복구할 수 있다. OPc는 (1)에서 복구한 값에 K와 ⊕를 하는 것으로 복구할 수 있다.

(3) c 복구

마스터키 K를 알기 때문에 c를 두 번째 E_K의 키로 취급해 r을 고려해 1라운드 SubBytes를 공격하면 복구할 수 있다. 이때 $r \in [0,127]$은 실제 공격이 SubBytes에서 바이트 단위로 이루어지므로 $r=1$일 때와 $r=9$일 때가 동일하게 분석된다. 따라서 c는 $r \in [0,7]$에 대해서 반복하면 복구할 수 있다.

(4) r 복구

마지막으로 정확한 r은 두 번째 E_K의 2라운드 SubBytes를 공격해 복구할 수 있다. ((3)에서 분석한 r이 1이었다면 1, 9, 17, ..., 121. 즉, 16번만 추측하면 된다)

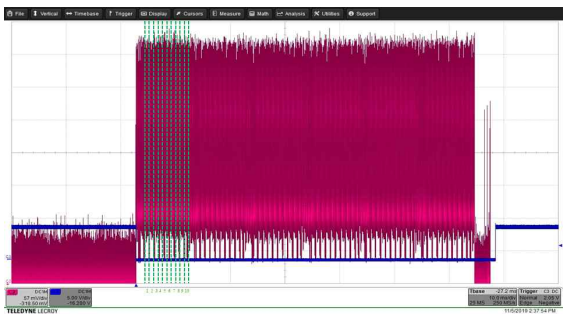
(3)-(4)를 다른 함수들에 대해 반복하면 모든 r, c들을 찾을 수 있다.

3. 3 파형 수집 및 전력 분석

실험 환경은 [표 3]과 같이 구성했다.

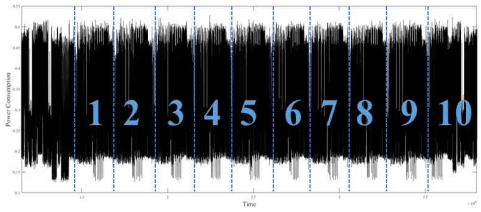
[표 3] 실험 환경

타겟 보드	SCARF-CEBv1.5
CPU	ATmega128
메모리	Flash Memory 128K EEPROM 4K SRAM 4KB
동작 주파수	14.745 MHz(외부 clock)
카드 제어 프로세서	NXP TDA8007
카드 슬롯	카드 타입
전력수집 보드	Oscilloscope HDO6104a
샘플링 레이트	500MS
파형 수	100,000
포인트 수	50,000



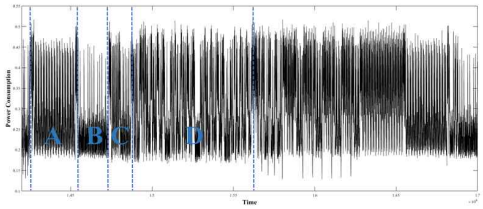
[그림 6] 인증 과정 중에 발생한 전력 파형

수집된 파형은 [그림 6]과 같이 6개의 큰 반복으로 이루어져 있고, 그 내부는 10번의 반복 구조를 보인다. 6개 중 하나를 자세히 확대하면 [그림 7]과 같이 나타난다.



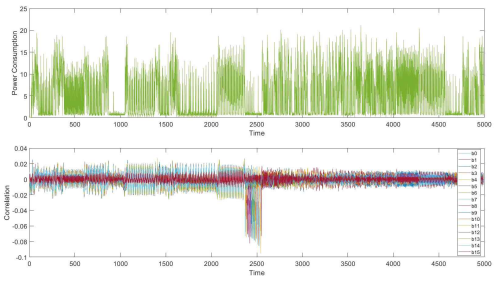
[그림 7] AES-128

[그림 7]의 1을 다시 한 번 확대하면 [그림 8]과 같이 나타난다.



[그림 8] AES-128의 첫 번째 라운드

[그림 8]은 A, B, C, D 4개의 부분 함수로 이루어졌다고 볼 수 있다. A, B는 16번의 반복구조를 가져 각각 AddRoundKey, SubBytes로 추측할 수 있다. 또한 10라운드에서 발생하지 않는 D연산을 MixColumns로 추측했다. 그리고 그 사이에 있는 C를 ShiftRows로 추측했다.



[그림 9] 1 Round SubBytes FOCPA 결과

[그림 9]는 첫 번째 라운드의 SubBytes 부분을 압축과 정렬로 전처리한 파형과 그 파형을 SubBytes 출력 값을 이용해 분석한 옳은 키의 상관계수이다. 해당 분석을 통해 (1)에 해당하는 $OPc \oplus K$ 를 복구했다.

IV. 결론

본 논문에서는 안전성을 위해 도입한 MILENAGE 알고리즘을 동작시킬 때 발생하는 전력 정보가 마스터키를 복구하는 데 사용될 수 있다는 것을 보였다.

현재 실험을 진행한 USIM들은 남은 재고만 팔고 있는 제품들이다. 따라서 추후 시중에 신형으로 팔고 있는 USIM이 동일한 취약성을 갖는지에 대해 분석하고자 한다.

[사사] “본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.”

[참고문헌]

[1] Junrong Liu, et al. “Small Tweaks do Not Help: Differential Power Analysis of MILENAGE Implementations in 3G / 4G USIM Cards”, 2015.

[2] ETSI, “3GPP TS 35.206 version 15.0.0 Release 15”, 2018.

[3] ETSI, “3GPP TS 31.102 version 15.2.0 Release 15”, 2018.

키워드 검색 기반의 이미지 증거 분석*

유동민†, 오희국‡
한양대학교 컴퓨터공학과

Evidence analysis based on keyword search

DongMin Yoo, Heekuck Oh
Department of computer science and engineering, Hanyang University.

요 약

최근 다량의 디지털 정보로 인해 디지털 증거를 분석하기 위해서는 많은 작업이 필요하다. 검색이나 파서를 이용하여 찾을 수 있는 문서파일과 달리 이미지파일의 경우 컴퓨터가 글자로 인식할 수 없는 바이너리 데이터로 구성되어 있어 조사관은 이미지를 수동으로 확인해야 한다. 이 논문에서는 원하는 분류의 이미지를 빠르게 찾기 위한 키워드 검색 기반의 이미지 증거분석 모델을 제시한다. 이를 위해선 이미지의 사전분석이 필요한데, 현재 머신러닝과 이미지분석 기술이 발달하면서 이미지 안 글자를 인식할 수 있고 무슨 사물인지 또는 사람의 얼굴까지 분류해낸다. 이러한 이미지분석 방법을 통해 이미지의 정보를 텍스트로 변환시키며, 문서화된 정보는 이미지를 수동을 확인하지 않고도 조사관이 관련 키워드를 가지는 이미지를 빠르게 검색할 수 있게 도움을 준다.

I. 서론

최근 전국 곳곳에 CCTV 또는 블랙박스가 설치되어 있고 사람들은 스마트폰과 디지털카메라를 이용해 하루에도 수많은 이미지와 영상을 만들어낸다. 이 같은 이미지(JPG, PNG)파일은 모바일이나 컴퓨터에 저장되며, 최근에는 모바일에서 찍은 데이터가 클라우드를 통해 자동으로 서버에 저장된다.

이러한 추세에서 수사기관에서는 수색된 디지털매체를 분석 할 때 증거의 단서를 찾기 위해 상당한 양의 파일을 확인해야 하며, 수많은 이미지파일을 확인해야 하는 조사관은 큰 난관에 봉착할 수 있다[1].

디지털 증거를 찾을 때 문서화된 파일들은 검색과 파서를 이용하여 원하는 데이터를 빠르게 접근할 수 있는 반면에 이미지파일이나 동영상 파일은 컴퓨터가 인식할 수 없는 바이너리 데이터로 구성되어 있어 검색이나 파서를 통해 이미지를 찾을 수 없기 때문에 조사관은 이미지파일을 하나씩 수동으로 확인해야한다. 이는 조사관의 업무강도를 높이는 것이고 작업이 비효율적으로 처리될 수 있다. 조사관들이 이미지 증거에 빠르게 접근하기 위해서는 이미지 파일을 검색기반으로 찾아야 한다. 이를 위해서는 바이너리 데이터를 분석하여 컴퓨터가 인식할 수 있는 데이터로 변환하는 작업이 필요하다. 이미지증거를 검색기반으로 빠르게 검색할 수 있다면, 조사관은 기존보다 더욱 효율적인 분석이 가능할 것이다.

예를 들어 만약 당신이 조사관이고 어떤 용의자가 DUBAI에 있었다는 증거를 찾아야한다. 보통 이러한 증거는 이미지를 통해 찾을 수 있는데, 이는 DUBAI의 랜드마크를 찾거나 고유

* “이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A2C2003045).”
† 주 저자 : ehdals614@naver.com
‡ 교신 저자 : hkoh@hanyang.ac.kr

한 차량의 번호판을 식별하여 찾을 수도 있다. 현재의 도구로는 이미지를 조사관이 일일이 클릭해가며 확인해야 하지만 검색기반으로 이미지를 서칭할 수 있다면, DUBAI, 랜드마크를 검색하는 것으로 해당 이미지를 빠르게 찾을 수 있다.

이 작업은 머신러닝과 컴퓨터 비전의 기술이 발달하면서 충분히 사용 가능한 단계까지 와있다. 실제로 OCR, Face Detect, Object Detect, 같은 기술은 이미지에 대한 머신러닝과 비전 분석을 이용하여 글자를 인식하고, 사람의 얼굴과 어떤 사물인지 분류할 수 있으며, 더 나아가서 장면에 대해서 어떤 상황인지에 대한 해석을 제공하기도 한다. 이러한 방법들을 이용해 이미지에 대한 시각적 분석을 바탕으로 이미지를 문서화 할 수 있으며, 조사관은 문서화된 정보를 이용하여 키워드 검색을 기반으로 이전과는 다르게 이미지 증거들을 빠르게 찾을 수 있다. 더 나아가서 이미지 정보를 데이터베이스화 시켰기 때문에 이미지들 간에 공통된 키워드가 존재하는지 확인할 수 있고 이를 통해 연관된 이미지를 나열할 수 있다.

II. 관련연구

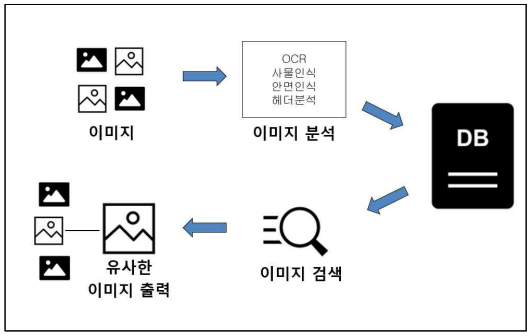
본 논문에서는 이미지의 분석과 이미지를 어떻게 문서화 하고 그 효과에 대해서 설명한다. 기존에 연구된 이미지의 분석방법은 다양하게 존재한다. 첫째로는 EXIF 데이터(교환 가능한 이미지 파일 포맷)를 분석하는 방법이 있다. EXIF 데이터를 포함하는 파일 구조는 이미지 자체에 대해 잠재적으로 가치 있는 정보를 가지고 있다. 이것은 사진을 찍기 위해 사용된 카메라 유형, 사진이 찍힌 날짜 및 시간, 그리고 말 그대로 사진과 관련된 수백 개의 데이터 요소를 포함할 수 있다. 둘째로는 컴퓨터 비전 분석방법을 이용할 수 있다. 대표적인 컴퓨터 비전기술로는 안면인식, 사물인식, OCR등이 존재한다. OCR은 사람이 쓰거나 기계로 인쇄한 문자의 영상을 이미지 스캐너로 획득하여 기계가 읽을 수 있는 문자로 변환하는 것이다. 사물인식(Object Recognition)은 이미지 또는 비디오 상의 객체를 식별하는 컴퓨터 비전 기술이다. 안면인식(Face Recognition)은 이미지파일들 속에 탐지되는 인물들의 얼굴을 특정할 수 있는 기술이다. 기존에 디지털포렌식 도구인 Encase, Autopsy 등은 이미지의 비전분석을 기능을 제공하지 않는다.[2] 그러나 여기서 사용할 컴퓨터 비전의 핵심기술인 OCR, FACE Recognition, Object Recognition의 대해서는 개발된 도구가 상당히 있다.

개별 기능을 지원하는 도구를 살펴보면, OCR 프로그램으로 tesseract, ABBYY, Adobe, Nuance, Readiris 가 있고, Object Recognition 프로그램으로는 구글의 Python Object Detection with Tensorflow가 유명하다. FACE Recognition 또한 구글의 Tensorflow를 통해 많은 머신러닝을 통한 기법들이 나열되어 있다. Microsoft Azure의 Face API도 안면인식을 제공한다. 위 방법을 통합하는 도구 또한 존재하는데, AMAZON의 Amazon rekognition 도구는 위 모든 기능을 통합한 API 형태로 제공하여 준다.[3]

III. 구현

기존의 디지털포렌식 도구로는 이미지의 시각적 분석을 할 수 없으며, 기존에 존재하는 시각적 분석 도구인 Amazon Rekognition 또한 입력 이미지에 대한 결과만을 출력해줄 뿐 이미지에 대한 통합분석을 하여, 데이터베이스를 구축하지 않는다. 그리고 더욱 효율적으로 분석하기 위한 이미지 증거 간에 상관분석을 실시하는 기능 또한 존재하지 않는다. 또한 현재의 존재하는 도구는 이미지의 시각적 분석을 통한 문서화(데이터베이스)와 이미지 간 상관분석을 해주는 도구가 없다.

우리는 이를 해결하기 위해 이미지 통합 분석 도구인 Amazon Rekognition API를 사용하여 이미지의 시각적 분석을 실시하고 각 이미지에 대한 정보를 문서화 하며, 각 이미지 간 정보를 바탕으로 키워드 검색 기반의 이미지 검색 도구를 개발하고 상관분석을 통하여 찾은 이미지와 비슷한 키워드를 가지는 이미지를 출력하여 보여준다.



[그림 1] 검색기반의 이미지 분석 모델

IV. 연구 사례

이미지 분석을 통해 이미지를 잘 문서화하는지 실험하였다.



[그림 2] 테스트에 사용된 이미지

4.1. EXIF 분석

Python Open API를 활용하여 이미지파일의 대한 EXIF값을 파싱하여 저장한다. 이 정보는 다른 이미지와 비교 분석할 때 사용된다.

```
test.jpg
('ExifVersion', '0221')
('ComponentsConfiguration', '\x01\x02\x03\x00')
('CompressedBitsPerPixel', (471040, 1228800))
('DateTimeOriginal', u'2003:11:20 11:39:33')
('DateTimeDigitized', u'2003:11:20 11:39:33')
('MaxApertureValue', (28, 10))
('FlashPixVersion', '0100')
('MeteringMode', 5)
('LightSource', 0)
('Flash', 24)
('FocalLength', (1460, 100))
('ExifImageWidth', 1280)
('Make', u'CASIO COMPUTER CO.,LTD ')
('Model', u'EX-Z4  ')
('Orientation', 1)
('ExifImageHeight', 960)
('Contrast', 0)
('ExposureBiasValue', (0, 3))
('XResolution', (72, 1))
('YResolution', (72, 1))
('ExposureTime', (1, 320))
('FileSource', '\x03')
```

[그림 3] EXIF분석의 결과

4.2. 글자인식

Amazon Rekognition API를 사용하여 이미지 속 텍스트를 식별하였다. [그림2]이미지에 대해서 약 42개의 글자를 출력했으며, 핵심적인 단어인, TAXI, DUBAI, VILLA, 48270 등이 포함되어 있다.

```
Detected text:Villa
Confidence: 94.02%
Id: 0
Type:LINE
()
Detected text:TAXI
Confidence: 98.17%
Id: 1
Type:LINE
()
Detected text:RIR
Confidence: 71.07%
Id: 2
Type:LINE
()
Detected text:48270
Confidence: 99.70%
Id: 40
Parent Id: 16
Type:WORD
()
Detected text:DUBAIOLN
Confidence: 90.88%
Id: 41
Parent Id: 17
Type:WORD
()
Text detected: 42
```

[그림 4] 텍스트 검색의 결과

4.3. 사물인식

Amazon Rekogntion API를 사용하여 사물인식 실시하였다. [그림2]이미지에 대해서 약 5개의 결과를 출력했다.(CAR, Vehicle, License Plate, Transportation, Automobile)

```
Detected labels in test.jpg
Transportation : 99.9327011108
Car : 99.9327011108
Vehicle : 99.9327011108
Automobile : 99.9327011108
License Plate : 93.483757019
Labels detected: 5
```

[그림 5] 사물인식의 결과

4.4. 안면인식

Amazon Rekogntion API를 사용하여 안면인식을 실시하였다. 이미지에 나타나는 안면을 인식하여 각 얼굴에 고유한 Face ID를 생성하고 Index에 집어넣는다. 만약 다음 이미지에서 유사한 얼굴이 발견되면 결과에 출력된다.

```
Faces indexed:
  Face ID: d45b88a9-c2e4-4f74-b09f-5f9e68e9928e
  Location: {u'Width': 0.3477552831172943, u'Top
 0.38098618388175964, u'Height': 0.3400074541568
Faces not indexed:
Faces indexed count: 1
```

[그림 6] Face 분석의 결과 [그림2]에서는 얼굴이 검출이 되지 않아 임의의 사진을 사용하였다.

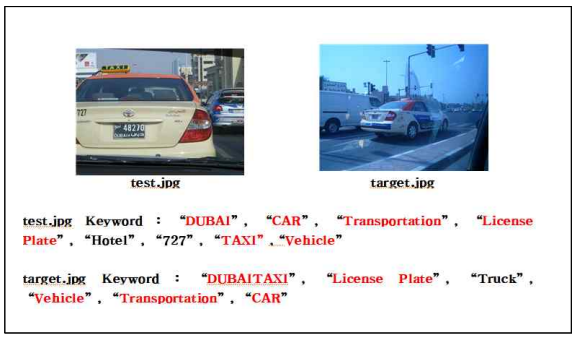
4.5 유사도 검사

위 4가지 방법을 이용하여 각 이미지에 대한 정보를 데이터베이스로 만든다.


```
('SceneCaptureType', 0)
('ResolutionUnit', 2)
('WhiteBalance', 0)
('GainControl', 0)
('FNumber', (70, 10))
('Software', u'1.00                ')
('DateTime', u'2003:11:20 11:39:33')
('Saturation', 0)
('FocalLengthIn35mmFilm', 88)
('Sharpness', 0)
('DigitalZoomRatio', (0, 0))
('CustomRendered', 0)
('PrintImageMatching', 'PrintIM\x000250\x00\x00\x00\x04\x00\x01\x00\x16\x00\x16\x00\x02\x
\x00\x00\x01\x00\x05\x00\x00\x00\x01\x01\x01\x00\x00\x00')
('ExposureMode', 0)
('ExifOffset', 276)
('YCbCrPositioning', 1)
{keyWord,"Car","Vehicle","Transporation","License
Plate","Villa","TAXI","RIR","Hotel","Galleria","KARANOA","00081470270","727","DUBAIOLN")
```


[그림 7] 이미지분석 정보 텍스트로 변환

데이터베이스화된 정보를 이용하면 검색을 통해 입력키워드와 연관된 이미지를 검색할 수 있고, 데이터베이스의 각 이미지를 비교하여 동일한 키워드가 얼마큼 중복되는지를 확인한다면, 검색된 이미지와 다른 이미지 사이의 유사도를 검사할 수 있다.



[그림 8] 두 이미지는 다른 이미지와 비교해서 비교적 비슷한 키워드를 가지고 있다.

[그림 8]에서 보이는 두이미지의 결과를 비교한 결과, EXIF는 42개 항목 중 시간항목을 제외한 35개 항목이 일치했으며, 비전분석의 결과로는 정확도가 90프로가 넘는 항목들에 대해서 25개의 키워드 중 11개가 동일한 키워드를 가지고 있었다.

V. 결론

본 논문에서는 수동으로 진행해야 하는 이미지 분석과정에 대해서 어떻게 자동으로 이미지를 탐색할 수 있는지에 대한 방법을 제시한다. 수색된 이미지 파일들에 대해서 이 모델을 사용함으로써 수많은 이미지데이터를 텍스트형식으로 데이터베이스화하고 조사관은 키워드 검색 기반으로 데이터베이스에서 관련 이미지를 빠르게 검색할 수 있으며, 검색된 이미지에 상관분석을 통해 유사한 유형의 다른 파일을 더욱 쉽게 찾을 수 있다.

[참고문헌]

[1] 이유지,"디지털포렌식 '이미지·동영상' 자동분석 수요 확대…"범죄수사 효율성 향상" https://byline.network/2017/05/1-731/ ,(2017.05).

[2]Opentext,"guidancesoftware.com/document/product-brief/encase-forensic-product-overview"

[3]Amazon,"Rekognition,aws.amazon.com/rekognition/?nc1=h_ls"

디지털 포렌식을 위한 Android 및 Windows 환경에서 카카오톡 아티팩트 분석(Ⅱ)

이나비*, 김광조*

*KAIST 정보보호대학원

Analysis of Artifacts in KakaoTalk under Android and Windows Environments for Digital Forensics (Ⅱ)

Nabi Lee*, Kwangjo Kim*

*Graduate School of Information Security, KAIST.

요 약

카카오톡은 국내에서 압도적인 점유율을 가진 채팅 애플리케이션으로 사용자의 스마트폰 또는 PC에 저장된 사용자의 송·수신 메시지는 사용자의 생활패턴, 심리상태 등을 확인할 수 있어 포렌식 분야에 적극 활용되고 있다. 하지만 카카오톡 메시지에 대한 심층학습 기반 감성분석은 극성 (긍정·부정)을 포함한 데이터 수집이 어렵다는 이유 등으로 지금까지 한 번도 시도된 적이 없었다. 이에 본 논문에서는 독자적으로 수집 및 전처리한 카카오톡 메시지 데이터셋을 Input, Hidden, Output 레이어로 구성된 기본적인 RNN (Recurrent Neural Network)으로 학습시켜, 카카오톡 메시지에 담긴 사용자의 주관 (최근 관심사, 심리적 상태, 채팅 참여자와의 밀접도 등)을 파악하고 이를 수사에 효과적으로 활용하는 방안을 살펴보았다.

I. 서론

카카오톡은 국내 스마트폰 사용자 중 97%가 애용하는 채팅앱으로 텍스트, 사진, 음성, 동영상, 화상통화 등 다양한 콘텐츠가 소통에 활용된다. 모든 송·수신 메시지는 사용자의 스마트폰과 PC에 저장되기 때문에 포렌식 연구에서 사용자의 스마트폰 또는 PC에 잔존하는 카카오톡 메시지 흔적(아티팩트) 추출 및 분석은 대단히 중요하다.

선행연구 [1]에서 저자는 Android 6(마시멜로) 환경의 모바일용 카카오톡과 Windows 10 환경의 PC용 카카오톡 아티팩트 유형과 속성정보의 공통점·차이점을 분석하여 이를 효과적으로 포렌식에 활용하는 방안을 제시하였다. 또한 추출 가능한 데이터의 양이 방대하고 수사관의 재량에 따라 획득 가능한 정보의 수준에 격차가 발생하는 만큼 다양한 수사기법의 융합이 필요하다는데 공감대를 형성하였다.

본 논문에서는 연구 [1]에서 확인한 아티팩트 중 메시지 감성분석에 포커스를 맞추었다. 감성분석은 텍스트 마이닝 기법 중 하나로 opinion mining이라고도 불린다. 사용자가 생성한 온라인 텍스트 속에 담긴 감성 (sentiment), 정서 (affect), 주관 (subjectivity), 또는 감정 (emotion)을 식별하기 위해 사용된다 [2]. 카카오톡 사용자는 자의든 타의든 일상생활의 많은 부분을 카카오톡을 통해 공유한다. 메시지를 송·수신하는 대상과 상황에 따라 사용자는 긍정·부정 등 감정의 흐름을 경험한다. 따라서 카카오톡 메시지 감성분석은 메시지에 담긴 사용자의 주관 - 최근 관심사, 심리적 상태, 채팅 참여자와의 밀접도 -을 파악하고 이를 프로파일링 (profiling)에 활용할 수 있다.

II. 관련연구

지금까지 특정 대상에 대한 의견을 수집하고 분류하는 과정인 감성분석은 주로 마케팅 영역에서 활용되었다. 설문, 포럼, 블로그, 웹사이트 등에서 여론을 수집하고 그 흐름을 분석하여 시장의 요구에 기민하게 대응하기 위하여 감성분석을 활용하였다. 최근 한국어 대상 연구도 활발하다. 하지만 상업적인 목적 외에는 연구가 더딘 실정이며, 특히 카카오톡 등 SNS에 대한 감성분석은 연구가 희박하다. 데이터셋 구성에

어려움이 있기 때문이다.

상품평, 영화평점/리뷰는 사용자가 평가할 대상 (상품, 영화 등)이 존재하기 때문에 긍정·부정 등 사용자의 주관이 담긴 데이터를 수집하는데 어려움이 없다. 반면, 카카오톡, 트위터 등의 SNS는 다양한 주제로 개인의 의견을 표현할 뿐 그 의견의 극성 (긍정·부정 등)이 무엇인지, 무엇으로부터 기인했는지 우리는 주목하지 않는다. 하지만 포렌식 관점에서는 SNS상 문구 하나하나가 사용자의 상태를 유추할 수 있는 실마리가 된다.

연구 [3]에서는 트위터에서 한국어로 작성된 트윗의 긍정·부정 여부를 통계적 기반 기계학습을 통해 분류하였다. 감정분류 대상인 텍스트에서 단어를 추출하고 그 단어에 해당하는 사전에 정의된 감정값을 추출하여 문장의 감정 정도를 계산한 뒤 그 결과값으로 긍정·부정을 분류하였다. 실험결과 약 76%의 분류 정밀도 [Precision, 긍(부)정이라고 판정한 것 중 실제 긍(부)정 수]를 보이는 반면 재현율 [Recall, 전체 긍(부)정 수 중 판정된 긍(부)정 수]이 낮아 학습 방법의 업데이트가 요구되었다.

연구 [4]은 문맥 표현 중 하나인 ELMo (Embedding from Language Models) 기법을 소개하고 한국어 영화평 감성 분석을 위하여 NSMC (Naver Sentiment Movie Corpus)와 다음카카오 (DaumKakao) 데이터를 셀프 어텐션과 ELMo를 이용한 RNN (Recurrent Neural Network)으로 학습시키는 모델을 제안하였다. 기존 모델에 비해 성능이 좋음을 입증하였다.

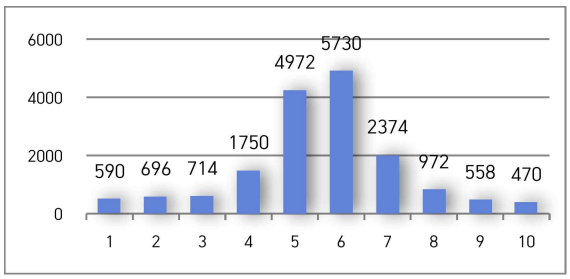
본 연구에서는 카카오톡 메시지 감성 분석을 수행하기 위하여 독자적으로 수집 및 구축한 카카오톡 메시지 데이터를 말뭉치로 사용하였다. 긍정 또는 부정의 상태를 도출하는 분류문제이므로 many-to-one 문제로 정의하고 Input, Hidden, Output Layer로 구성된 기본적인 RNN 모델을 적용하였다. 학습모델은 새로운 말뭉치에 대한 감성 분석 가능성을 확인한 뒤 업데이트 예정이다.

III. 연구설계

1.1 데이터 수집

카카오톡 메시지를 감성분석하려면 각각의 메시지는 긍정 또는 부정에 대한 정보를 포함하고 있어야 한다. 우선 우리는 2019. 7. 1. ~ 2019. 9. 30.까지의 카카오톡 메시지 18,826건을 수집한 뒤 메시지 작성자에게 메시지 작성 당시의 감정 상태를 수렴하여 1점~10점까지 (1에 가까우면 부정적인 상태, 10에 가까우면 긍정적인 상태) 점수화하였다. 본 연구에서 필요한 것은 긍정 또는 부정에 대한 분류이므로 1점~10점까지 점수화된 메시지 중 4점~7점 구간은 긍정 또는 부정의 극성이 약하거나 정보전달 목적의 메시지들로 극성을 전혀 띄고 있지 않다고 판단하여 제외하고 1점~3점 구간을 부정의 상태인 '0', 8점~10점 구간을 긍정의 상태인 '1'로 재분류하였다. 이렇게 분류된 메시지는 총 4,000건으로 긍정 2,000건 부정 2,000건 동일한 비율이다.

Figure 1. 카카오톡 메시지 감정분포



1.2 데이터 점검 및 탐색

우리가 수집한 카카오톡 메시지 18,826건 중 약 80% (15,060건)가 연구 [1]에서 Axion Process를 통해 수집한 csv 파일이고, 나머지는 카카오톡 메시지 내보내기 기능을 통해 수집한 txt 파일이다. 이를 Tab으로 구분한 txt 파일 형태로 통일하였다. 또한 데이터는 user, date, contents, label 네 개의 열로 이루어져 있는데 user는 메시지 작성자, date는 메시지 작성 일시, contents는 메시지 내용, label은 긍정(1), 부정(0)인지를 나타낸다.

1.3 데이터 전처리 및 정제

카카오톡은 일상생활 대화부터 업무, 동호회, 관심사별 오픈채팅 등 사용자에 따라 그 내용과 유형이 다양하고 외부에 유출될 경우 사생활이 침해받을 수 있는 내용도 다수 있기 때문에 특정 개인·집단을 직·간접적으로 지칭 및 유추 가능할 수 있는 내용에 대한 정제가 필요하다. facebook 메신저, 인스타그램 등 여느 SNS와 같이 이모티콘, 띄어쓰기, 'ㅋㅋ' 등 분리된 자음·모음 등에 대한 선별적인 정제도 요구된다.

1.3.1 형태소 분석 및 품사 태깅

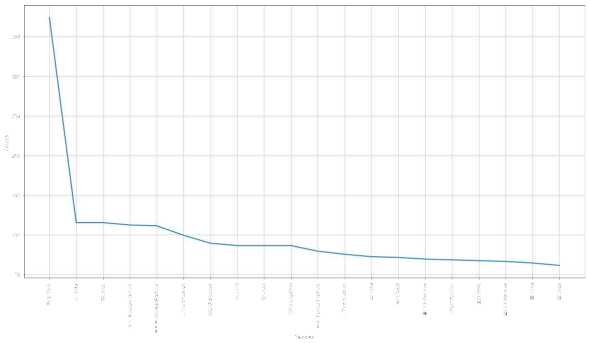
데이터는 KoNLPy 라이브러리의 okt 클래스를 이용하여 형태소 분석 및 품사 태깅을 하였다 [5]. 카카오톡 메시지는 1.3.1에서 설명한대로 특성상 맞춤법이나 띄어쓰기 등이 제대로 되어

있지 않은 경우가 많기 때문에 정확한 분류를 위하여 띄어쓰기, 맞춤법 등을 정제해주는 KoNLPy를 이용하였다. KoNLPy 라이브러리를 통해 분석된 형태소 하나하나를 데이터의 토큰(token)이라 한다.

1.3.2 중복토큰 분석

분석한 데이터의 토큰 중 일부는 중복된 경우가 있을 수 있으므로 nltk 라이브러리를 통해서 전처리를 하였다. 가령 카카오톡 메시지 4,000건을 okt로 형태소를 분석해보면 전체 토큰의 개수는 24,496개이지만 중복을 제외한 토큰의 개수는 4,910건이다. 출현 빈도가 높은 상위 토큰 20개를 보면 주로 '이', '에' 같은 조사 또는 'ㅋ' 였다.

Figure 2. 출현빈도 높은 상위 토큰 20개



1.3.3 데이터 벡터화

분석된 토큰으로 데이터를 벡터화한다. 원 핫 인코딩 대신 CountVectorization을 사용하였다. CountVectorization은 Scikit-Learn 라이브러리의 문서 전처리 클래스 중 하나이다. 문서 집합에서 단어 토큰을 생성하고 각 단어의 수를 세어 각 문서를 BOW (Bag of Words) 인코딩 벡터로 변환한다.

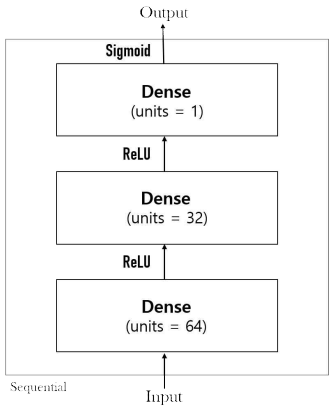
1.4 데이터 모델링 및 학습

인공신경망 (Artificial Neural Network)을 구현하는 여러 가지 Neural Network 생성용 library 중 본 연구에서는 Tensorflow와 Keras를 사용하였다. 네트워크 내부 레이어 (layer) 구성은 Keras 내부 모듈 중 Sequential과 Dense를 사용하였다. 데이터는 카카오톡 메시지 4,000건 중 training dataset과 validation dataset을 8:2 비율로 각각 3200건, 800건으로 나누어 활용하였다.

1.4.1 모델 정의

Fig. 3과 같이 Sequential 모듈을 통해 신경망 Template를 만들고 training dataset을 Input 레이어에 넣은 뒤 각각의 뉴런을 활성화시킨다. activaiton 함수로는 'ReLU (Rectifier Function)'를 사용하였다. 이를 통해 얻은 우선 예측값을 실제 결과값과 비교하여 그 차이를 계산한다 (forward propagation). 당연히 오차가 발생하기 때문에 우리는 이 차이를 최소화하기 위하여 예측값을 다시 구하는 과정을 반복하게 되는데 이때 MSProp 옵티마이저를 통해서 경사하강법으로 오차를 최소화하는 과정을 거친다. Output 레이어에서는 이진 분류에 적합한 sigmoid 함수와 validation dataset을 사용하여 학습한 모델의 성능을 검증한다. 손실함수로는 binary_crossentropy를 사용하였다.

Figure 3. 카카오톡 메시지 감성분석 모델



1.4.2 모델 구현 (학습) 결과

카카오톡 메시지 4,000건에 대해 Fig. 3 모델을 기반으로 학습할 경우 (batch size: 256, epoch: 10) 정확도는 최초 60%에서 점차 상승하여 약 94%에 도달하였으며 손실률은 최초 70%에서 7%대까지 감소하는 등 카카오톡 메시지 감성분석에 해당 모델이 유용하게 활용될 수 있음을 보였다.

Figure 4. 학습모델의 정확도

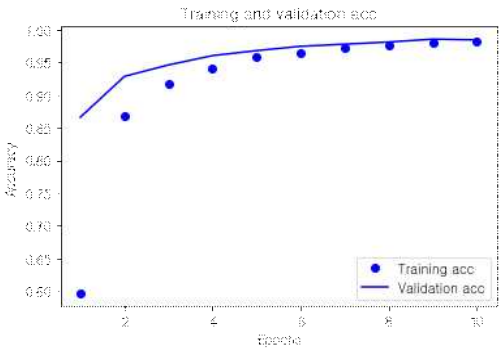
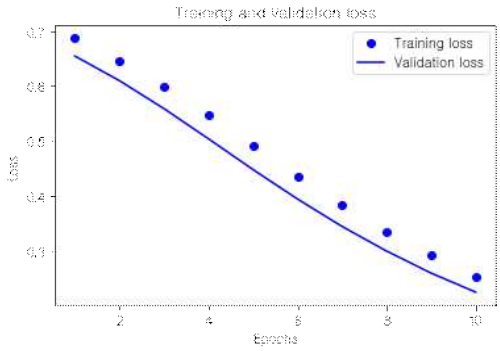


Figure 5. 학습모델의 손실률



IV. 평가

1.4.1의 모델의 성능평가를 위하여 우리는 train dataset 및 validation dataset과 중복되지 않는 카카오톡 메시지 200건 (긍정, 부정 각각 100건)을 추가로 준비하여 평가 (정확도, 정밀도, 민감도, 특이도)를 진행하였다 [6].

Table 1. 카카오톡 메시지 감성분석모델 성능평가

구 분		실제 정답	
		긍정 (Positive)	부정 (Negative)
실험 결과	긍정 (Positive)	69	17
	부정 (Negative)	31	83

1.1 정확도 (Accuracy)

카카오톡 메시지 200건 중 152건의 메시지 극성을 제대로 판별하여 정확도는 76%이다.

1.2 정밀도 (Precision)

긍정으로 분류된 메시지 86건 중 실제 긍정 메시지인 경우는 69건으로 긍정에 대한 분류 정밀도는 약 80.2%이며, 부정으로 분류된 메시지 114건 중 실제 부정 메시지인 경우는 83건으로 부정에 대한 분류 정밀도는 약 72.8%인 것으로 확인되었다.

1.3 민감도 (Sensitivity)

재현율 (Recall)로도 불린다. 실제 긍정 메시지 100건 중에서 긍정으로 제대로 분류된 메시지는 69건으로 69%의 민감도를 보였다.

1.4 특이도 (Specificity, True Negative Rate)

실제 부정 메시지 100건 중에서 부정으로 제대로 분류된 메시지는 83건으로 83%의 특이도를 보였다.

카카오톡 메시지 감성분석모델에 대한 평가 결과, 학습단계에서의 정확도 대비 약 18% 감소한 76%를 기록하였다. 분류 정밀도, 민감도, 특이도 역시 70%~80% 수준을 보였으며, 긍정보다는 부정의 감정이 담긴 메시지 분류에 상대적으로 높은 성능을 보임을 확인하였다.

V. 결론

본 연구에서는 지금까지 카카오톡 메시지 감성분석을 위하여 18,826건의 카카오톡 메시지의 극성을 분류하고 극성이 뚜렷한 메시지 4,000건을 선별하여 이를 Input으로 하는 RNN 모델을 설계 및 구현하였다. 또한, 학습이 완료된 모델에 대해 카카오톡 메시지 200건을 추가 수집하여 성능평가를 진행한 결과 76%의 정확도를 보였다. 학습단계에서의 정확도 94% 대비 정확도가 감소한 이유는 감성분석모델의 한계 때문인 것으로 판단된다. 기본적인 RNN 구조를 기반으로 모델링하였기 때문에 메시지 길이가 길거나 판별이 어려운 것이 나오면 생략해버리는 경우가 발생할 수 있다. 이에 향후에는 LSTM (Long Short Term Memory)을 활용한 카카오톡 메시지 감성분석모델을 설계 및 구현 후, 본 연구의 모델과 성능을 비교할 필요가 있다.

[참고문헌]

[1] 이나비, and 김광조. “디지털 포렌식을 위한 Android 및 Windows 환경에서 카카오톡 메시지의 아티팩트 분석(Ⅰ).”

[2] 김유영, and 송민. "영화 리뷰 감성분석을 위한 텍스트 마이닝 기반 감성 분류기 구축." 지능정보연구 22.3 (2016): 71-89.

[3] 임좌상, and 김진만. "한국어 트위터의 감정 분류를 위한 기계학습의 실증적 비교." 멀티미디어학회논문지 17.2 (2014): 232-239.

[4] 박천음, et al. "문맥 표현과 셀프 어텐션을 이용한 한국어 영화평 감성 분석." 정보과학회논문지 46.9 (2019): 901-908.

[5] 박은정, 조성준, “KoNLPy: 쉽고 간결한 한국어 정보처리 파이썬 패키지”, 제 26회 한글 및 한국어 정보처리 학술대회 논문집, 2014.

[6] Olson, David L.; and Delen, Dursun (2008); Advanced Data Mining Techniques, Springer, 1st edition (February 1, 2008), page 138, ISBN 3-540-76916-1

드롭박스 포렌식 데이터 획득 기법

김은진*, 황현욱*, 손기욱*
*ETRI 부설연구소

How To Acquire Forensic Data From DropBox Storage

Eunjin Kim*, Hyunuk Hwang*, Kiwook Sohn*
*The Affiliated Institute of ETRI

요 약

오늘날 클라우드 서비스는 보편화 되었으며, 사용자의 데이터 또한 클라우드 서버에 저장되는 추세이다. 이에 디지털 수사에 있어 클라우드 서비스에 저장된 사용자의 데이터 획득에 대한 중요성이 커지고 있다. 본 논문에서는 포렌식 도구에서 사용자의 드롭박스 계정을 이용하여 드롭박스 데이터의 접근을 허가받는 방법에 대하여 설명한다. 또 드롭박스의 API와 인증 토큰을 이용하여 드롭박스에 저장된 파일의 목록을 획득하고 획득한 파일의 목록에서 내려받기할 파일을 선택할 수 있다.

I. 서론

오늘날 클라우드 서비스 환경은 대중화되어 일반적인 사용자를 포함하여 기업에 이르기까지 다양한 분야에서 사용되고 있다. 클라우드 서비스의 확산에 따라 사용자의 데이터는 사용자의 기기가 아닌 클라우드에 저장되는 추세로 변하고 있다. 이러한 환경 변화에 따라 클라우드 서버에 저장된 사용자 데이터를 조사할 필요성이 높아지고 있다.

드롭박스는 전 세계적으로 많은 사람이 사용중인 클라우드 스토리지 서비스이다. 이에 디지털 수사 시에 드롭박스의 데이터를 획득하여 조사할 필요성이 있다. 드롭박스는 서드파티 응용프로그램에서 파일을 다운로드 혹은 올릴 수 있도록 REST API를 제공하고 있다[1]. REST API는 2000년에 발간된 Roy Thomas Fielding의 논문에서 처음 소개되었으며, 기존의 HTTP를 이용하여 리소스에 대한 접근을 수행할 수 있다[2]. 본 논문에서는 REST API를 이용하여 드롭박스의 포렌식 데이터를 획득하는 방법을 소개한다.

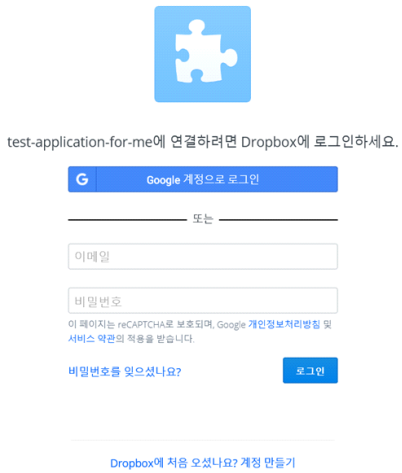
II. 드롭박스 데이터 획득

2장에서는 드롭박스의 REST API를 이용하여 사용자의 드롭박스 데이터를 획득하는 방법에 대하여 설명한다. 본 논문에서는 드롭박스 계정 수사에 앞서, 이미 조사할 사용자의 ID, 비밀번호 등의 로그인 정보를 알고 있다고 가정한다. 이를 이용하여 드롭박스 계정 데이터에 접근할 수 있도록 OAuth 인증 토큰을 발급받는다. OAuth는 서드파티 응용프로그램이 자원의 소유자를 대신하여 파일 등의 자원을 요청할 수 있도록 접근 권한을 위임하는 방법이다[3]. 인증 토큰을 발급받은 뒤, 사용자 계정에 저장된 파일의 목록을 획득한 뒤, 최종적으로 파일 데이터를 획득한다. 드롭박스의 REST API를 이용하여 데이터 획득을 실험하기 위하여 우분투 16.04에서 curl 명령어를 이용하여 REST API를 실행한다[4].

2.1 드롭박스 접근 권한 획득

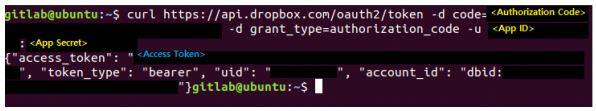
드롭박스 계정의 인증 토큰을 받기 위해서는 드롭박스 데이터 획득을 수행할 도구의 'App ID'와 'App Secret' 정보가 필요하다. 이 정보를 얻기 위해서. 드롭박스의 앱 콘솔에 도구를 등록해야 한다. 드롭박스의 앱 콘솔에 도구를 등록하면 해당 도구의 'App ID'와 'App Secret' 정보가 생성되며, 이 정보와 드롭박스의 API를 사용하여 인증 토큰을 발급받을 수 있다.

드롭박스의 인증 토큰을 획득하기 위해서는 앞서 등록한 도구의 로그인 페이지에서 데이터를 획득할 사용자 계정으로 로그인해야 한다. [그림 1]은 로그인 요청 URL 'https://www.dropbox.com/oauth2/authorize?client_id=<*AppID*>&response_type=code'에 웹브라우저를 통해 접속한 화면이다. 이 페이지에서 드롭박스 데이터를 획득할 사용자 계정으로 로그인할 경우, 사용자의 인증 토큰을 요청하는데 필요한 Authorization Code가 웹브라우저에 표시된다. 하나의 Authorization Code로 단 한 번만 인증이 가능하다.



[그림 1] 드롭박스 앱 로그인 화면

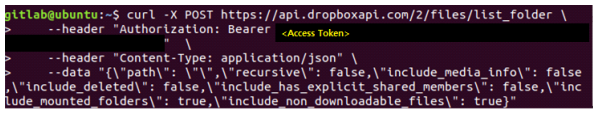
Authorization Code를 이용하여 드롭박스의 OAuth2.0 인증 서버로 인증 토큰 발급 요청을 보낼 수 있다. 인증 토큰 발급 요청에는 Authorization Code와 인증을 요청한 앱의 App ID, App Secret이 필요하다. [그림 2]는 curl을 이용하여 인증 토큰을 발급 요청하는 화면이다. 'https://api.dropbox.com/oauth2/token' URL에 Authorization Code, grant_type, App ID와 App Secret을 POST 데이터로 전송한다. 요청에 대한 응답은 JSON 형식으로 반환된다. 인증 토큰은 'access_token' 값에 기록된 것을 확인할 수 있다.



[그림 2] 인증 토큰 요청 및 응답

2.2 드롭박스 파일 목록 획득

드롭박스는 지정한 디렉터리에 속한 파일의 리스트를 반환하는 List Folder API를 제공한다. List Folder API는 각 파일의 이름, 크기, 시간 정보 등의 파일 정보를 제공한다. List Folder API를 이용하여 사용자의 폴더 구조 및 파일 목록을 얻기 위해서는 인증 토큰이 필요하다. 인증 토큰 정보는 요청의 헤더에 입력이 되며, API의 인자는 JSON 데이터형식으로 POST 방식을 이용하여 전송한다. [그림 3]은 curl 명령을 이용하여 파일의 목록을 요청한 화면이다.



[그림 3] 파일 목록 요청 응답

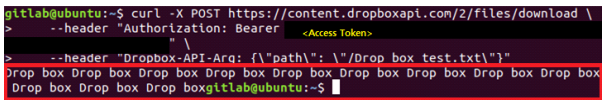
List Folder API는 인자 "path"에 지정된 경로에 저장된 파일 혹은 폴더의 정보를 반환한다. 반환 형식은 JSON이며, ".tag"에 file 혹은 folder, deleted 정보 등 파일의 종류 혹은 상태를 나타낸다. "name" 변수는 파일 혹은 폴더의 이름을 나타낸다. "client_modified", "server_modified"에는 각각 클라이언트 수정시각과 서버 수정시각 값을 저장하며, "size"에는 파일의 크기 정보가 저장된다. [표 1]의 JSON 데이터 예시는 [그림 3]의 요청에 대한 응답 중 "Drop box test.txt" 파일 항목의 정보를 옮긴 것이다.

[표 1] 파일 목록에 저장된 파일 항목 예시

{".tag": "file",
"name": "Drop box test.txt",
"path_lower": "/drop box test.txt",
"path_display": "/Drop box test.txt",
"id":"id:vqCs12rNTdYAAAAAAAAAGA",
"client_modified": "2018-10-26T01:26:43Z",
"server_modified": "2018-10-26T01:27:00Z",
"rev": "190e474183",
"size": 107,
"is_downloadable": true,
"content_hash":
"4ab7bc4c5270526ac105df556afc564dc24da4179e051b5ff0
19cfa764e5da8b"}

2.3 드롭박스 파일 데이터 획득

드롭박스의 File Download API는 파일 이름이나 수정(Revision) 번호를 통해 지정된 파일을 내려받을 수 있도록 제공한다. 앞서 얻은 파일 목록과 파일의 정보를 이용하여 획득할 파일을 선택하여 개별 파일에 대하여 내려받기 요청을 보낼 수 있다. 요청에 대한 응답으로 파일의 데이터가 반환된다. [그림 4]는 curl 명령어로 드롭박스 File Download API 호출하여 파일 데이터를 획득 요청하는 화면이다. 헤더에 사용자의 인증 토큰 정보와 JSON 데이터형식으로, path 혹은 파일의 revision 정보를 문자열 형식으로 입력하여 다운로드 대상 파일을 지정한다. 응답으로 “Drop box Test.txt”의 데이터가 출력된 것을 확인할 수 있다.



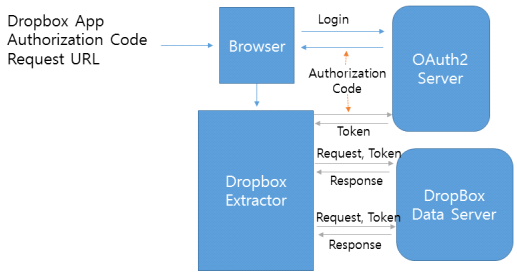
[그림 4] 파일 다운로드 요청 및 응답

III. 구현 및 평가

2장에서 설명한 방법을 이용하여 드롭박스의 인증 토큰을 얻고, 계정에 저장된 파일의 전체 목록과 파일 데이터를 얻는 도구를 구현하였다. 인증 토큰을 얻기 위하여 먼저 로그인 요청 URL에 접속하여 사용자 계정의 인증 정보를 입력한 뒤 Authorization Code를 도구에 입력한다. 인증 토큰을 얻어 오는 데 성공하면, 사용자 계정의 최상위 경로로부터 순회하며 전체 파일 목록을 획득한다. 마지막으로 획득한 파일 목을 기반으로 전체 파일을 내려받는다.

3.1 드롭박스 인증 토큰 획득

인증 토큰을 얻기 위하여 먼저 로그인 요청 URL에 접속하여 사용자 계정의 인증 정보를 입력한 뒤 Authorization Code를 도구에 입력한다. 입력된 Authorization Code를 이용하여 인증 토큰을 드롭박스 인증 서버에 요청한 후, 이를 내부적으로 저장하여 드롭박스의 데이터를 획득하는 데 활용한다. [그림 5]는 도구의 드롭박스 인증 과정을 나타냈다.



[그림 5] 도구의 드롭박스 인증 과정

3.1 파일 목록 획득

획득한 인증 토큰을 이용하여 그림 5와 같이 전체 파일 목록을 획득하였다. 파일의 경로, 클라이언트 수정시각, 서버 수정시각, 파일의 크기 정보를 출력하였다. [그림 6]은 드롭박스의 사용자 계정에 저장된 전체 파일 목록을 출력한 화면이다.

3.2 파일 데이터 획득

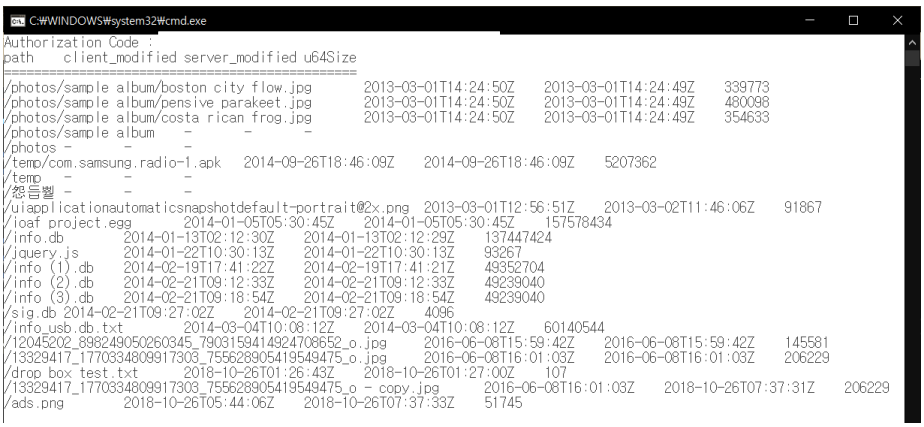
파일 데이터를 획득하기 위하여 3.1에서 획득한 파일의 목록 및 정보를 이용하여 사용자 계정의 전체 파일 데이터를 내려받기하였다. [그림 7]은 드롭박스 계정에 저장된 모든 파일을 내려받기 한 결과이다. 3.1에서 획득한 파일 목록에 있는 모든 파일이 정상적으로 내려받기 되었다.

IV. 결론

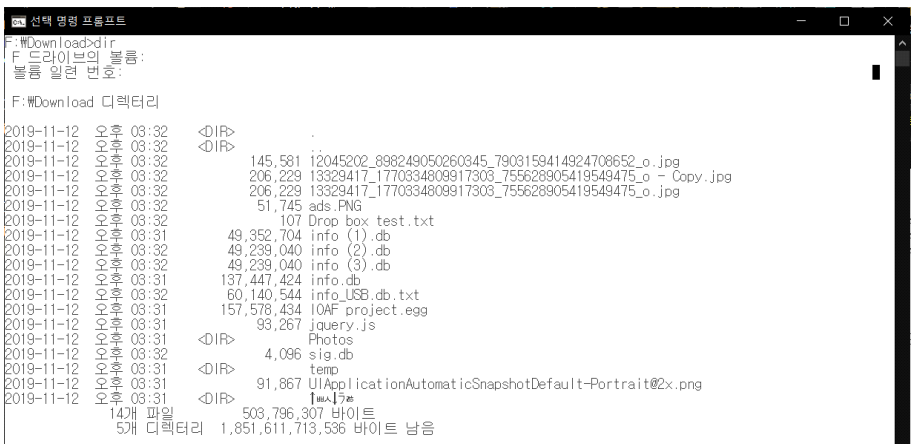
지금까지 수사 환경에서 사용자의 계정을 이용하여 드롭박스의 데이터를 획득하는 방법에 대하여 설명하였다. 이 논문에서 제안한 방법을 포렌식 도구에 구현할 경우, 사용자 계정정보를 이용하여 인증 토큰을 생성하여 포렌식 도구가 드롭박스의 클라우드 데이터에 접근할 수 있도

록 할 수 있다. 또 사용자 계정에 저장된 파일의 목록을 먼저 획득하여, 사건에 관련이 있는 데이터만을 선택적으로 내려받기할 수 있다.

클라우드 서비스는 스토리지뿐만 아니라 다양한 서비스에 이용되고 있다. 향후 드롭박스와 같은 클라우드 스토리지뿐만 아니라 다른 주요한 클라우드 서비스로 연구를 확대하여 다양한 클라우드 데이터 획득 기술에 관한 연구가 필요할 것이다.



[그림 6] 구현한 도구를 이용하여 획득한 전체 파일 목록



[그림 7] 구현한 도구를 이용하여 내려받기한 전체 파일

[참고문헌]

[1] Dropbox for HTTP Developer, https://www.dropbox.com/developers/documentation/http/documentation

[2] Roy Thomas Fielding, Architectural Styles and the Design of Network-based Software Architectures (Doctoral dissertation), University of California, Irvine, 2000

[3] 금융보안원, OAuth 2.0 개요 및 보안 고려사항, 2015

[4] curl, https://curl.haxx.se/

윈도우 아티팩트 생성 추적 방안

고신우*, 이건우**, 이세영***, 이충호****, 정희태*****

서울여자대학교 정보보호학과, 고려대학교 컴퓨터학과, 경북대학교 컴퓨터학부,
한세사이버보안고등학교 정보보안과, 조선대학교 컴퓨터공학과

Windows Artifact Generation Tracing Method

Shin-Woo Ko* Geon-Woo Lee**, Se-Yeong Lee***, Chung-Ho Lee****,

*Department of Information Security, Seoul Women's University

**Department of Computer Science and Engineering, Korea University.

***Department of Computer Science and Engineering, Kyungpook National University.

****Hansei Cyber Security Highschool

*****Department of Computer Engineering, Chosun University

요 약

디지털 포렌식 또는 침해사고 대응을 수행할 때 응용 프로그램에서 발생하는 아티팩트를 활용하면 프로그램의 동작을 추측할 수 있다. 다만, 해당 프로그램에서 사용하는 아티팩트가 어떠한 시점에 어떤 경로에서 발생하는지에 대한 정보를 사전에 파악해야 하는 문제가 존재한다. 어떤 아티팩트를 생성하는지 공개되지 않은 응용 프로그램을 분석하는 경우와 버전 업데이트 등의 사유로 사용하는 아티팩트의 경로가 변경된 경우에 아티팩트 발생 경로를 파악하기 어렵다. 따라서, 본 논문에서는 응용 프로그램에서 생성하는 아티팩트를 추적하는 기존의 방법을 정리하고, 디버깅 개념을 도입해 앞서 기술한 2가지 경우에서 발생할 수 있는 문제들을 해결하고 결론적으로 더 효율적인 아티팩트 생성 추적방안을 제시하고자 한다.

I. 서론

윈도우 시스템의 응용 프로그램에서 발생하는 아티팩트는 프로그램 동작 분석 또는 사용 흔적을 분석하는데 사용자에게 편의를 제공한다. 예를 들어, Microsoft Edge와 같은 웹 브라우저는 웹사이트에 접속할 때 접속 기록을 "WebCacheVxx.dat"와 같은 특정 경로에 저장하고, KakaoTalk과 같은 온라인 메신저는 채팅 기록을 "KakaoTalk.db" 내부에 저장한다[1]. 위와 같이 특정 프로그램에서 생성하는 아티팩트가 어떤 시점에 어떤 경로에 생성되거나 생성하는 지에 대한 정보가 이미 밝혀진 경우에는 해당 아티팩트를 확보해 분석을 진행할 수 있다.

따라서 아티팩트 분석을 위해 아티팩트가 생성되는 경로를 미리 파악된 경우에만 분석을 진행할 수 있으므로 아티팩트 생성을 추적하는 것은 중요하게 다루어져야 할 문제다. 응용 프로그램에서 생성하는 아티팩트가 무엇인지 알

려지지 않거나, 버전 업데이트 또는 패치로 인하여 사용하는 아티팩트가 변한 경우에는 아티팩트를 확보하기 어렵다. 이러면 아티팩트의 생성 시점과 경로를 재분석해야 하는 문제가 발생한다. 기존의 아티팩트 생성 추적 방법으로는 수동으로 프로그램을 분석하는 역공학(리버스 엔지니어링) 기법을 사용해 분석을 진행해 사용하는 아티팩트를 추적하는 방법, 그리고 Process Monitor(procmon) 도구와 같이 윈도우 시스템의 API 호출을 추적하는 방법을 제공하는 프로그램을 통해 아티팩트 생성을 추적할 수 있다. 이에 본 논문은 두 가지 방법을 모두 적용해 아티팩트 생성을 좀 더 효율적으로 추적하는 방법을 제시하고자 한다.

II. 관련 연구 및 기술

아티팩트와 관련이 있는 연구는 크게 2가지 종류로 나뉜다. 특정 아티팩트에서 획득할 수 있는 정보를 통해 사용 흔적을 추적하는 방법에 관한 연구, 특정 프로그램에서 발생하는 아티팩트를 추적하는 것에 관한 연구로 총 2가지로 구분할 수 있다. 아티팩트의 생성 원인과 경로가 알려진 상태에서는 생성된 아티팩트를 추적하거나 상세하게 분석하는 위주의 연구가 진행되었다[1][2]. 다만 해당 연구들의 결과는 아티팩트의 위치가 알려지지 않으면 아티팩트를 획득하는 것이 어려우므로 분석에 많은 제한이 있다. 따라서 아티팩트의 생성 원인과 경로가 알려지지 않을 때는 아티팩트의 생성을 추적하는 주제의 연구가 주로 진행되었고, 샌드박스(Sandbox)를 이용하거나 모니터를 이용한 캡처링(Capturing) 위주의 기술을 사용되었다[3][4].

다만 샌드박스 기술을 기반을 둔 연구들의 결과는 실제 분석환경을 설정할 수 없거나, 분석환경 설정에 많은 시간을 요구하는 경우가 많다. 그뿐만 아니라, 캡처링을 사용해 분석을 진행한다면 사용자의 구체적인 행위를 파악하기가 어렵거나, 안티 디버깅 기법이 적용된 프로그램의 분석은 제한되는 경우가 발생할 수 있다. 따라서, 기존의 연구 및 기술이 가진 한계점을 해결하기 위해서는 따로 분석환경 구축 없이 실제 분석환경에서 응용 프로그램의 아티팩트 생성 추적을 효율적으로 진행할 수 있는 도구가 필요하다.

III. 제안

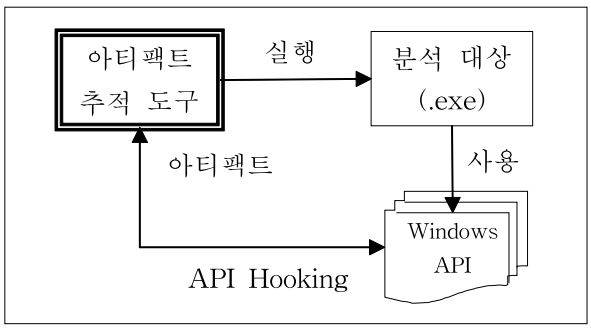
본 논문은 디버깅 개념을 도입하여 아티팩트의 생성을 효율적으로 추적 및 분석하는 방안을 제시하고자 한다.

3.1 디버깅 개념 도입

본 논문에서 제안하는 디버깅 개념은 주로 사용되는 디스어셈블리 레벨에서의 중단점(breakpoint)를 설정하는 디버깅이 아닌 Windows API 호출 단에 API Hooking을 이용하여 가상의 중단점을 설정하고 프로그램의 행위를 추적하는 개념이다.

3.2 프로그램 구성

디버깅 개념을 도입한 아티팩트 생성·추적 도구의 구조는 다음과 같다.

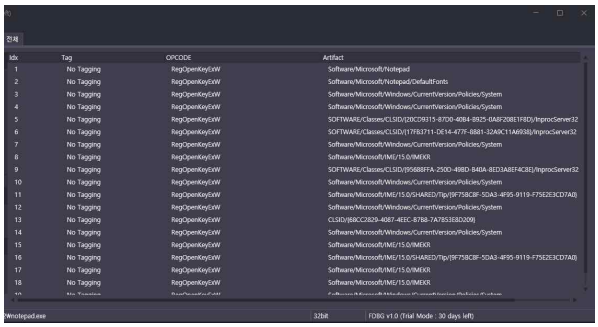


<그림 1> 프로그램 동작 구조

아티팩트 생성·추적 도구(이하 도구)를 실행시킬 때, 분석 대상의 절대 경로를 제공하면 해당 프로그램을 Suspend 상태로 실행을 시키고 안티 디버깅 혹은 안티 리버싱 기술을 우회하는 작업을 수행한다. 이후 해당 프로그램에서 실행하는 모든 Windows API에 Hooking을 하여 모든 API Call을 추적할 수 있게 된다. 이후 분석 대상 프로그램이 Windows API를 사용하

는 행위(RegQueryValueEx, ReadFile, CreateFile 등)를 수행할 때마다 사용된 함수명과 인자 값, 즉 경로와 생성 시점을 확보할 수 있다. 이렇게 확보한 정보들을 사용자가 어떤 행위를 했을 때 발생한 정보인지를 분류하기 위해 선택된 메뉴별로 각 정보를 분류한다. 그뿐만 아니라 버전 업데이트 혹은 패치가 이루어지기 전·후의 프로그램 두 가지 모두를 확보한 뒤 메뉴별로 분류된 정보들을 서로 비교하면 버전 변화에 따른 차이 또한 확인할 수 있다. 이렇게 분류된 정보는 기존 연구의 문제를 개선할 뿐만 아니라, 정확하게 사용자의 행위를 추적할 수 있다.

아래 사진은 3장에서 제안한 기술을 사용하여 만든 예시 프로그램이며, 메모장을 예시로 분석을 진행한 경우, 정상적으로 아티팩트가 추적되는 것을 확인할 수 있다.



<그림 2> 실제 구현된 프로그램의 작동예시

IV. 비교

본 논문이 2장에서 언급한 바와 같이 관련 연구 및 기술보다 가지는 장점은 다음과 같다. 만약 아티팩트의 생성 원인이 알려지었을 때 본 논문에서 제시한 방안을 적용한 도구를 통하여, 아티팩트의 정확한 생성 시점과 생성 경로와 프로그램의 동작 별 생성되는 아티팩트를 추적할 수 있는 장점이 있다. 또한, 아티팩트의 생성 원인과 경로를 모르는 경우 역시 해당 도구를 사용하여 분석환경 설정에 소요되는 시간을 줄여 보다 효율적이고 빠른 분석을 기대할 수 있다.

아티팩트 분석 도구	샌드박스	제안 방안
알려진 아티팩트에 대해 정보를 추출	가상환경에서 실행시킨 후 아티팩트 추출	디버깅 개념을 도입하여 API Hooking 후 아티팩트 추적
아티팩트 생성 위치를 모를 때 사용 불가능	아티팩트 생성 위치를 모를 때도 사용 가능	추가로 아티팩트 생성 시점 또는 기능에 대해 정밀한 분류 가능
	분석환경을 설정하기 위하여 시간 필요	분석환경에서 바로 실행 가능

<표 1> 기존 연구와의 비교

V. 결론

최소한 아티팩트의 생성 경로를 추적 또는 확인을 통해 미리 경로를 파악하고 있어야 분석을 진행할 수 있다. 알려지지 않은 응용 프로그램에서 발생하는 아티팩트를 추적하는 것은 기존의 연구 혹은 방법을 통해서는 아티팩트가 생성되는 명확한 시점을 파악하기 어렵거나 분석환경 구성에 많은 시간이 소요되는 등 여러 가지 문제가 제기되었다. 본 논문은 Windows API에 Hooking을 진행해 어떤 아티팩트에 접근하는지 이미 알려진 응용 프로그램을 분석하는 경우에는 아티팩트를 생성하는 명확한 시점을 확보할 수 있다는 장점이 있다. 메모장(“notepad.exe“)을 예시로 들어보면 저장 버튼을 눌렀을 때 저장되는 파일의 절대 경로를 확인할 수 있으므로 사용자가 저장 버튼을 눌렀다고 추정할 수 있다. 이처럼 사용자의 행위를 추적할 수 있는 용도로 사용될 수 있다. 또한, 아티팩트를 생성하는 시점이나 경로가 알려지지 않은 응용 프로그램을 분석할 경우, 해당 프로그램에서 생성하는 아티팩트의 경로와 생성 시점을 확보할 수 있다. 마지막으로, 기존의 샌드박스를 활용한 분석 방법보다 본 논문에서 제안한 도구를 활용하였을 때 분석환경 설정에 소요되는 시간을 단축할 수 있다는 이점도 있다.

본 논문에서는 기존의 윈도우 시스템의 응용 프로그램에서 생성하는 아티팩트들을 추적하는

방안을 제시하고 기존 기술이 가진 한계를 해결할 수 있는 대안을 제시하였다. 응용 프로그램을 분석할 때 본 논문에서 제시한 아티팩트 생성 추적방안이 분석 기법의 하나로 고려될 수 있을 것이다. 윈도우 시스템에서의 추적방안만 다루었기 때문에 추후에는 다른 플랫폼에서의 추적방안 또한 연구되어야 할 것이다.

[참고문헌]

[1] 윤종철, 박용석. (2016). KakaoTalk의 채팅 메시지 포렌식 분석 연구 및 WhatsApp의 Artifacts 와의 비교 분석. 한국정보통신학회논문지, 20(4), 777-785.

[2] 김진성, 은창오, 정임영. (2017). 윈도우 환경의 아티팩트를 활용한 자동화된 사용자 분석 방안. 한국통신학회 학술대회논문집, (), 1437-1438.

[3] 이상진, 강봉구, 윤종성, 이민욱. (2016). Cuckoo Sandbox를 이용한 포렌식 침해지표 자동생성 및 활용 방안. 정보처리학회논문지. 컴퓨터 및 통신시스템, 5(11), 419-426.

[4] Meshram, M.P., &Kapgate, D. (2015). Investigating the Artifacts Using Windows Registry and Log Files.

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 E

WBAN 환경에서의 안전한 1라운드 인증 프로토콜[1)]

최재현* 오상윤* 탁금지* 정익래* 변진욱**

*고려대학교 정보보호대학원 정보보호학과 **평택대학교 정보통신학과

Jae Hyun Choi* Sangyun Oh* Gum Ji Tak*

Ik Rae Jeong* Jin Wook Byun**

*Graduate School of Information Security, Korea University.

** Dep. of Information and Communication, Pyeongtaek University

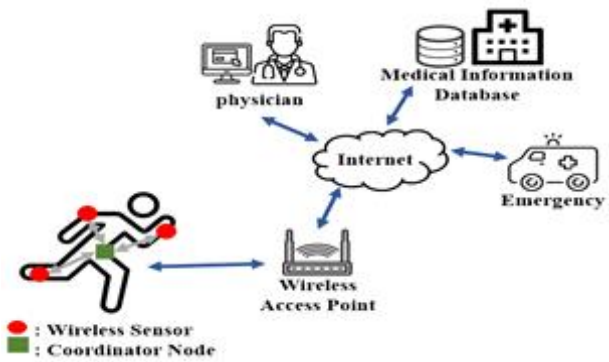
요 약

센서 기반 웨어러블 장치는 Wireless Body Area Network(WBAN)을 이용하여 사용자의 생체 정보를 서버에 전달한다. WBAN은 사용자의 민감한 정보를 다루기 때문에, 해당 분야에 대한 연구가 활발해 지고 있다. 최근 Li 기타 등은 Liu 기타 등의 프로토콜에서 사용자의 프라이버시를 개선했다. 하지만 해당 논문에서 Network Manager(*NM*)는 사용자로 위장하는 방식의 공격 가능하다. 본 논문에서, 이에 대한 해결과 실용적인 대책을 제시한다.

I. 서론

Wireless Body Area Network(WBAN)은 <그림 1>에서 처럼 임베디드 기기를 활용하여 사용자의 신체 범위 내에서 이루어지는 근거리 무선 통신기법을 의미한다. 최근 임베디드 기기의 발전으로 인해 해당 분야의 연구가 더욱 활발해지고 있다[1][2]. 특히, WBAN의 특성 때문에 주로 의료 기관에서 환자의 건강 관리에 사용되고 있다. 민감한 개인정보를 담고 있어서 보안이 잘 지켜져야 하고 임베디드 기기의 특성인 저장 공간 등의 제한을 고려해야 한다.

2016년 Liu 기타 등[3]은 WBAN에서의 효율적인 1라운드 익명 프로토콜을 제시했다. 해당 프로토콜에는 U_i, AS, NM이 사용된다. 등록 단계에서 NM은 인증 단계에서 사용될 사용자와 서버의 공개키를 안전하게 배포한다. 2017년 Li 기타 등[4]은 Liu 기타 등의 프로토콜이 공격자의 가장공격에 안전하지 않음을 보이고 그에 대한 보안 취약점을 개선했다. 그러나 Li 기타 등의 프로토콜 역시 가장공격에 취약하다. 본 논문에서는 Li 기타 등에서 가능한 가장공격을 보이고, 이에 대한 개선방안을 제안한다.



<그림 1> WBAN 활용 방안

II. 관련 연구

2장에서는 제안하는 프로토콜에서 사용되는

1) 이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. 2017R1D1A1B03032424)

ECC와 WBAN에 대한 내용을 설명한다.

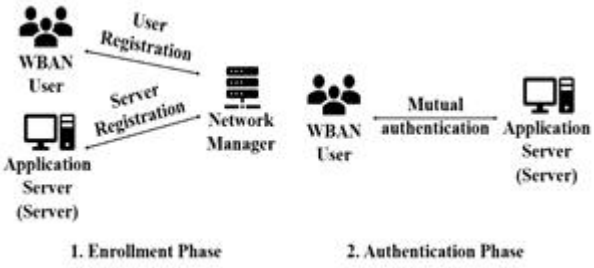
2.1 Elliptic Curve Cryptography(ECC)

ECC는 타원곡선 이론에 기반을 둔 공개키 암호 시스템이다. 타원 곡선과 관련 있는 수학적 연산을 아래와 같이 정의한다.

• 타원곡선은 유한체에서 정수로 이루어진 E 상의 점들의 집합이다. ($E: y^2 = x^3 + ax + b$). 타원곡선 상에서의 곱셈은 타원곡선 상에서 정의된 덧셈 연산을 반복하는 것이다.

2.2 WBAN의 구조

WBAN의 통신 과정은 아래의 <그림 2>와 같이 WBAN 사용자 U_i, 네트워크 매니저 NM, 어플리케이션 서버 AS로 구성되어 있다. 각 요소에 대한 정의는 Li 기타 등의 프로토콜과 같은 정의를 사용한다. 통신은 크게 다음과 같은 두 가지 단계를 거쳐 진행한다.



<그림 2> WBAN의 통신과정 구조

• 등록 단계 : U_i와 AS는 각자 신뢰할 수 있는 NM에 신원을 등록하고 통신 키를 발급받는다.

• 인증 단계 : U_i와 AS가 안전하지 않은 통신 채널 상에서 상호 인증을 수행한다.

Ⅲ. Li 기타 등의 프로토콜 분석

전체적으로 프로토콜에서 사용될 표기와 그에 대한 설명은 <표 1>에서 정리하였다.

표기	설명
U_i, AS, NM	i번째 사용자, 서버, 네트워크매니저
id_{U_i}	U_i의 아이디
$hash(.)$	암호학적으로 안전한 해시 함수
sk_{U_i}/pk_{U_i}	U_i의 개인키/공개키
S	AS, NM간의 공유 된 대칭키
k, r	임의의 수
$E_k(.)/D_k(.)$	대칭키 암/복호화 연산
MAC	메시지 인증 코드
t_c	현재의 타임 스탬프

<표 1> 표기 및 설명에 대한 소개

1. 등록 단계

• U_i는 id_{U_i}와 자신의 공개키인 pk_{U_i}를 NM에게 보낸다.

• NM은 랜덤값 k를 Z_q^*에서 선택한다. 타원곡선의 생성원 P로 $Ind_U = kP$와 $U = k \cdot pk_{U_i}$를 계산한다. AS와 사용자에게 제공될 서비스가 포함된 $Right$ 값을 대칭키 S로 암호화한 $E = E_S(Ind_U, Right)$를 계산한다.

• NM은 $\{Ind_U, E, U, k\}$을 U_i에게 보내고 U_i는 이 인자들을 안전하게 저장한다.

2. 인증 단계

• <그림 3>에서 U_i는 랜덤값 r과 AS의 공개키pk_{AS}를 이용하여 $R = r \cdot P$, $Q = r \cdot pk_{AS}$를 계산한다. NM에게 받은 Ind_U, E를 대칭키 Q로 암호화한 $E_U = E_Q(Ind_U, R, E)$를 만든다. $y = hash(t_c, R + U)$, 자신의 개인키 sk_{U_i}와 NM로부터 받은 k를 사용하여 $c = r - sk_{U_i} \cdot y \cdot k$를 계산한다. E_U, R, c, t_c를 메시지 M이라 하고 AS에게 전송한다.

• AS는 자신의 비밀키 sk_{AS}에 R을 곱하여 Q'을 계산한다. Q'을 사용하여 암호문 E_U로 부터 $\{Ind_U, R, E\}$를 복호화한다. 암호문 E는 NM과 사전에 공유하였던 대칭키 S로 복호화하여 $Ind'_U, U, Right$를 얻는다. 이때 구해진 Ind'_U와 Ind_U가 서로 같은지 확인 한다. AS는 $y' = hash(t_c, R + U)$을 계산하고 검증식 $cP + y'U \stackrel{?}{=} R$을 수행한다. 검증식을 통과하면 MAC키 값으로 사용할 세션키 $K = hash(Ind_U \| t_c \| Q', R + U)$를 생성하고, 인증할 메시지 $M' = MAC_K(y')$을 전송한다.

• U_i는 $K' = hash(Ind_U \| t_c \| Q', R + U)$를 계산하여 전송받은 M'과 생성한 K'의 MAC 값이 같은지 확인하고 검증이 될 경우 K'를 AS와 공유된 세션키로 사용한다.

U_i (Secure channel) NM

유저와 서버 등록

1. id_U를 선택한다

2. $\{id_{U_i}, pk_{U_i}\}$ →

3. $k \in_R Z_q^*$ 를 생성

$Ind_U = kP, U = k \cdot pk_{U_i}$

$E = E_S(Ind_U, U, Right)$

4. $\{Ind_U, E, U, k\}$ ←

U_i는 받은 인자들을 안전하게 저장한다

U_i (Insecure channel) AS

유저 로그인 및 인증

1. $r \in_R Z_q^*$ 을 선택

$R = rP, Q = r \cdot pk_{AS}$

$E_U = E_Q(Ind_U, R, E)$

$y = hash(t_c, R + U)$

$c = r - sk_{U_i} \cdot y \cdot k$

2. $M = \{E_U, R, c, t\}$ →

3. $Q' = sk_{AS} R$

$\{Ind_U, R, E\} = D_{Q'}(E_U)$

$\{Ind'_U, U, Right\} = D_S(E)$

$Ind'_U ? = Ind_U$ 확인

$y' = hash(t_c, R + U)$

$cP + y'U \stackrel{?}{=} R$ 검증

$K = hash(Ind_U || t_c || Q', R + U)$

4. $M' = MAC_K(y')$ ←

5. $M' ? = MAC_{K'}(y)$ 검증

세션키 공유 $K = hash(Ind_U || t_c || Q', R + U)$ ↔

<그림 3> Li 기타 등의 프로토콜

3. Li 기타 등의 프로토콜 취약점

Li 기타 등의 프로토콜에서 U_i가 생성하는 값들은 모두 NM이 생성 가능하고, AS는 통신하는 상대를 명확히 알 수 없기 때문에 NM이 U_i를 가장할 수 있다. U_i를 가장하기 위해서 랜덤값 r', 임의의 공개키, 개인키 쌍 (pk', sk')가 필요하다. <그림 4>에서처럼 NM은 U_i를 프로토콜에서 고립시키고 U_i인척 가장할 수 있다.

<그림 4> 가장공격 시나리오

(U_i를 고립) NM

1. pk', sk' 생성

2. $k \in_R Z_q^*$ 를 생성

$Ind_U = kP, U' = k \cdot pk'_U$

$E = E_S(Ind_U, U', Right)$

NM (U_i) (Insecure channel) AS

유저 로그인 및 인증

1. $r' \in_R Z_q$ 을 선택

$R' = r'P, Q' = r' \cdot pk_{AS}$

$E_U = E_Q(Ind_U, R, E)$

$y' = hash(t_c, R + U')$

$c' = r - sk'_U \cdot y' \cdot k$

가장공격 시나리오에서 AS에서 진행되는 검증과정은 Li 기타 등의 프로토콜과 동일하므로 생략한다.

Ⅵ. 해결 방안

본 논문에서 NM이 사용자로 가장하는 공격에 대하여 제안하는 해결 방안은 시간검증 신뢰기관 TA를 도입하는 것이다. TA가 사용자를 인증하고 인증서를 발급하기 때문에, 인증서를 위조할 수 없는 NM은 가장공격이 불가능하다.

1. 등록 단계

• U_i는 랜덤값 k를 Z_q^*에서 선택하고, $U = k \cdot pk_{U_i}$와 함께 사용자가 생성한 타임스탬프 t_u를 시간검증 신뢰기관(TA)에게 전송한다.

• TA는 U_i로부터 받은 값을 자신의 개인키 sk_{TA}로 서명한 후 U_i에게 보낸다. U_i는 TA로부터 받은 서명값과 함께 U, t_u값을 NM에게 전송한다.

• NM은 AS와의 대칭키 S를 사용하여 암호화한 $E = E_S(U, Right, t_u, \sigma_{TA})$값과, 자신의 개인키로 U를 서명한 인증서 $cert = sig_{sk_{NM}}(U)$를 만들어 U_i에게 전송한다.

2. 인증 단계

• U_i는 세션키에 사용될 난수 γ와 NM으로부터 받은 E값을 pk_{AS}로 암호화 한다. t_c를 $k \cdot sk_{U_i}$로 서명을 한 뒤 이 값들과 $cert$를 AS

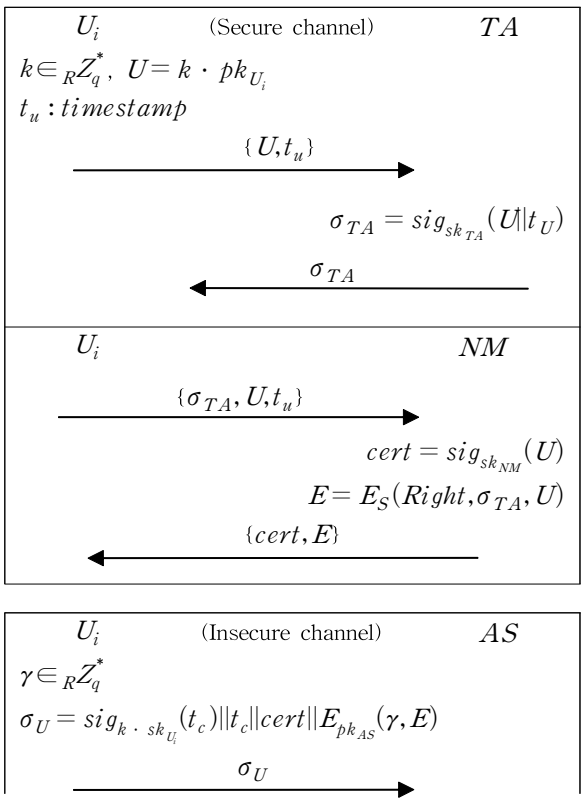
에게 전송한다.

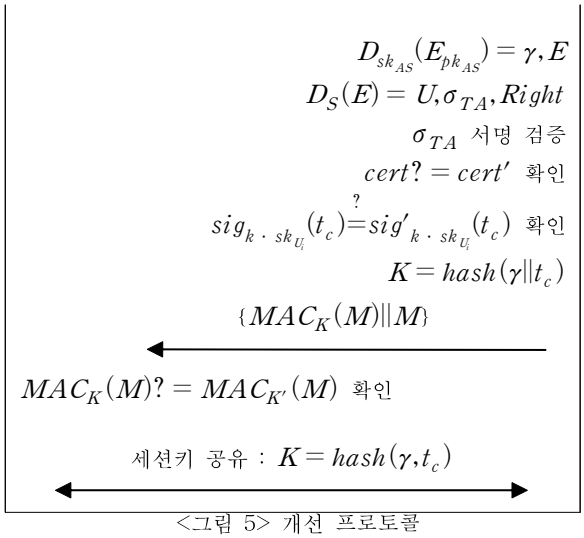
• AS는 E를 복호화 하여 $U, Right, \sigma_{TA}$를 구하고 $cert, \sigma_{TA}, U, U_i$에 대하여 검증하는 과정을 거친다. 검증식이 모두 통과될 경우, 보낼 메시지 M과 $K = hash(\gamma \| t_c)$를 MAC키로 사용하는 M의 MAC을 전송한다.

3. 안전성 분석

본 논문에서 제안한 프로토콜에서는 시간검증 신뢰기관 TA를 도입한다. TA는 U_i가 생성한 타임스탬프가 안전한 채널을 통해 전달받은 후 유효한지 검사를 하고 서명을 한다.

<그림 5>에서 U_i는 신뢰기관으로부터 받은 서명 σ_{TA}으로 NM과 AS에게 현재 통신 주체가 정당한 사용자임을 증명할 수 있다. 프로토콜에 사용되는 TA 서명은 TA의 개인키가 없으면 복제 불가능하다. 따라서 NM은 프로토콜에 사용되는 모든 매개변수 들을 생성할 수 없기 때문에 가장공격이 불가능하다. 또한 타임스탬프를 활용하여 재생 공격을 방지한다. K를 만들 때 U_i가 생성한 난수값이 활용되고 이 난수는 AS만 복호화 할 수 있어서 공격자는 K를 만들 수 없다.





<그림 5> 개선 프로토콜

[참고문헌]

[1] D. He, S. Zeadally and L. Wu, “Certificateless Public Auditing Scheme for Cloud-Assisted Wireless Body Area Networks”, IEEE Systems Journal, vol. 12, no. 1, pp. 64-73, Mar 2018.

[2] J. Shen, S. Chang, J. Shen, Q. Liu, and X. Sun,” A lightweight multi-layer authentication protocol for wireless body area networks”, Future Generation Computer Systems, vol. 78, part 3, pp. 956-963, Jan 2018.

[3] J. Liu, L. Zhang, and R. Sun, “1-Raap: an efficient 1-round anonymous authentication protocol for wireless body area networks.” Sensors. vol 16. 728, May 2016

[4] X. Li, J. Peng, S. Kumari, F. Wu, M. Karuppiah, and K. K. R. Choo, “An enhanced 1-round authentication protocol for wireless body area networks with user anonymity,” Comput. Elect. Eng, vol 61, pp. 238-249, Nov 2017

FANET 환경에서의 효율적인 Proxy Signature 기법*

오상윤, 정익래

고려대학교 정보보호학과

An Efficient Proxy Signature In FANET Network

Sangyun Oh, Ik Rae Jeong

Division of information security, Korea University.

요 약

프록시 서명(Proxy Signature)이란 자신의 서명 권한을 넘겨줘서, 권한을 전달받은 사람이 원본 서명자(Original Signer)처럼 서명이 가능한 방법을 말한다. 이와 더불어 최근 UAV를 지역 감시 및 추적의 용도로 사용하는 경우가 많아지고 있다. 이러한 환경에서, UAV의 구성 요소는 데이터 수집을 위한 센서, 통신 장치, 상호 작용을 위한 기타 장치로 구성이 된다. 통신 요소가 들어가기 때문에 보안 및 인증절차를 고려해야 하며 무선 센서 네트워크의 특성인 제한된 연산 능력도 고려해야 한다. 본 논문에서는 이와 관련해 최근 발표된 UAV상에서의 프록시 서명에 대해 다룬 Verma등의 제안에서의 개선사항을 소개한다.

I. 서론

서명 권한을 다른 사람에게 위임해주는 프록시 서명[1]은 서명자가 부재일 경우 대리자가 서명해주는 식으로 여러 환경에서 응용되어 쓰이고 있다[2]. 또한, 최근 IoT 기기의 발달로 인해 각 기기 간의 통신 기법이 세분화되고 있으며 특히 스마트 교통 시스템이 발전하면서 VANET[3](Vehicular Ad-Hoc Network)도 같이 발전해왔다. 여기서 더 나아가서, 최근 UAV(드론 등) 관련 산업이 발전함에 따라 UAV는 VANET의 중요한 요소로써 감시, 정보 전달 등의 역할로 사용됐다. 하지만 UAV의 특성상 VANET로 정의하기에는 그 범위가 너무 넓어지기에 FANET(Flying Ad-Hoc Network)가 나오게 되었다. 표 1에 나와 있듯이, FANET는 VANET에 비해서 이동성이 뛰어나 네트워크 연결과 통신 효율 부분에서 까다로운 시스템이다. FANET는 센서를 달고 있는 여러 대의 UAV로 구성이 되어 있으며, 산불 감시[4], 표적 탐지[5], 긴급상황에서의 생존자 추적 및 모니터링[6] 등에 쓰인다.

2019년에 Verma등은 UAV상에서 작동하는 프록시 서명 기법을 제안하였다[7]. Label-1에 있는 드론은 서버로부터 프록시 서명 권한을 넘겨받고, 서버 대신 서명을 해주는 방식이다. 하지만 UAV 시스템을 고려했을 때, Verma등의 프로토콜은 Label-2 단에서 단순 중간자의 역할밖에 하지 않기에 본 논문에서는 Label-2의 UAV를 활용하는 계층적 구조의 프록시 서명을 제안한다.

2장에서는 페어링 연산기법과 프록시 서명에 대해서 다룬다. 3장은 Verma등의 프로토콜에 대해서 논의하고, 4장에서는 개선된 프로토콜을 제안한다. 끝으로, 5장에서 결론을 맺는다.

* 본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

	VANET	UANET
Description	Self-organizing network with distrubuted ground vehicles	Self-organizing network with distrubuted air vehicles (drone).
Topology	Mesh	Star, Mesh
Node mobility	High speed, 0-30m/s, 2-demension space	Very high speed, 0-100m/s, 3-demension space
Node limitation	Low computational power.	Low computational power, limit of storage.

표 1. VANET과 UANET의 비교[8]

II. 관련 연구

2장에서는 논문에서 사용한 연산기법인 페어링 기법과 프록시 서명에 대해서 소개한다.

2.1. Bilinear Pairing

덧셈 순환군 G_1과 곱셈 순환군 G_2는 모두 위수를 소수 p로 갖는다고 가정한다. Bilinear Pairing은 map $e : G_1 \times G_2 \rightarrow G_2$ 로 나타낼 수 있으며 다음과 같은 특성이 있다.

1) 겹선형성(Bilinearity)

$\forall P_1, P_2 \in G_1$와 $p_1, p_2 \in Z_p^*$에 대해 $e(p_1P_1, p_2P_2) = e(P_1, P_2)^{p_1p_2} \in G_1$를 만족한다.

2) 비소실성(Non-Degenerate)

$e(P_1, P_2) \neq 1$에 대해서 $\forall P_1, P_2 \in G_1$을 만족한다.

3) 계산 가능성(Computatility)

임의의 원소 $\forall P_1, P_2 \in G_1$에 대해서 $e(P_1, P_2)$을 쉽게 계산할 수 있다.

2.2. 프록시 서명

1장에서 언급했던 프록시 서명에 대해 보안 요구사항을 설명한다.

1) 구별 가능(Verifiability) : 프록시 서명은 원본 서명자(Original Signer)의 서명과는 구분돼야 한다.

2) 강한 위조 불가능(Strong Unforgeability) : 지정된 프록시 서명자와 원본 서명자만이 유효한 서명을 만들 수 있다.

3) 강한 신원 확인(Strong Identifiability) : 검증자는 프록시 서명으로부터 원본 서명자의 서명 값을 확인할 수 있어야 한다.

4) 강한 부인 불가능(Strong Undeniability) : 프록시 서명자는 자신이 서명한 것에 대해서 부인할 수 없어야 한다.

5) 프록시 서명자의 위반 불가능(Proxy Signer's Deviation) : 대리 서명자는 서명 위임 없이는 서명할 수 없다.

III. 이전 프로토콜에 대한 분석

3장에서는 Verma등의 프로토콜에 대해 살펴본다. 이 프로토콜에는 CA, Alice(Original Signer), Bob(Designated Receiver), Charls (Proxy Signer)로 총 4개의 개체가 참여한다. 또한, Label-1의 UAV들은 통신 범위에 한정이 있기에 Label-2의 컴퓨팅 파워가 높은 드론을 통해 Base Station과 통신을 한다. 시스템 모델은 그림 1과 같으며, 다음과 같은 단계들로 진행이 된다.

1) Setup

Security parameter 1^q를 통해 system parameter $(G_1, G_2, e, Q, P, mpk, H, H_0, p, H_1)$과 CA의 키 쌍$(mpk, msk)$을 만든다.

e : Bilinear map, $Q, P \in G_1$, $H : \{0,1\}^* \times G_1 \rightarrow G_1$, $H_0 : \{0,1\}^* \times G_t \times G_t \times G_t \rightarrow G_t$, $H_1 : \{0,1\}^* \times G_1 \times G_1 \rightarrow G_1$

Choose $s \in Z_p^*$, $mpk = sp, msk = s$

2) UserKGen

모든 사용자는 비밀키인 $x_{ID} \in Z_p^*$를 랜덤하게 선택하고, 공개키 PK_{ID}^1, PK_{ID}^2를 계산한다. 사용자의 키 쌍은 다음과 같다.

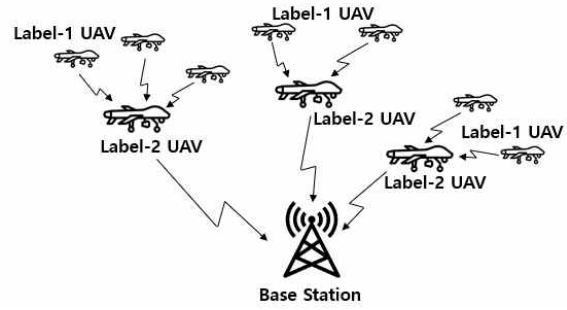


그림 1. FANET UAV System

$PK_{ID} = (PK_{ID}^1 = x_{ID}P,\ PK_{ID}^2 = x_{ID}Q), x_{ID}$

3) Certify

CA는 $Params, ID_u, PK_u^1$을 통해 모든 참여자 u의 인증서 $Cert_u = sH(ID_u, PK_u^1)$를 만들고 배포한다. 따라서 모든 참여자는 CA, Alice, Bob, Charls의 인증서를 얻을 수 있다.

4) DelGen

Alice는 warrant $w, Params, Cert_A, Cert_C, Cert_B$를 통해 서명 위임자(delegation)$\sigma_{J\to C} = x_A H_0(w, Cert_A, Cert_C, Cert_B)$를 계산한다.

5) DelVer

프록시 서명자 Charls는 $e(\sigma_{J\to C}, p)$의 유효성을 확인하고, 맞다면 서명 위임자를 받아들인다. $e(H_0(w, Cert_A, Cert_C, Cert_B), PK_A^1)$

6) CBDVProxySign

메시지 m에 대한 프록시 서명인 σ를 프록시 서명자인 Charls가 계산하고 서명 (m, σ, w)을 출력한다.

$$\sigma = (H_1(m, x_C PK_B^1, Cert_C) + x_C)^{-1}\sigma_{A\to C}$$

7) CBDVProxySVer

검증자 Bob은 $e(\sigma, H_1(m, x_B PK_C^1, Cert_C)p + PK_C^1)? = e(H_0(w, Cert_A, Cert_C, Cert_B), PK_{A)}^1$을 통해 서명을 검증한다.

IV. 프로토콜 제안

실제로 UAV를 통한 프록시 서명을 하기 위해서는 중간자 역할을 해주는 Label-2의 역할이 중요하다. 하지만, Verma의 논문에는 Base Station과 프록시 서명자에 대해서만 다루며, 중간 노드의 역할인 Bob은 단순히 값의 전달만 한다. 컴퓨팅 파워가 강한 Label-2의 UAV를 단순 중계자 역할로만 제한하는 것은 시스템적으로 봤을 때 비현실적이며 효율적이지 않다. 본 논문에서는 Label-2 UAV의 효율적인 사용을 위해 계층적 구조의 위임 네트워크를 제안한다.

4.1 Phase 1

먼저 원본 서명자와 Label-2 단에서 서명 위임에 관해 설명한다.

1) DelGen

Alice는 $w, Params, Cert_A, Cert_B$를 통해 서명 위임자 $\sigma_{A\to B} = x_A H_0(w, Cert_A, Cert_B)$를 계산한다.

2) DelVer

Bob은 $e(\sigma_{A\to B}, p)$의 유효성을 확인한다.

$e(H_0(w, Cert_A, Cert_B), PK_A^1)$

3) CBDVProxySign

Bob은 프록시 서명 (m, σ_B, w)을 만든다.

$$\sigma_B = (H_1(m, x_B PK_A^1) + x_B)^{-1}\sigma_{A\to B}$$

4) CBDVProxySVer

검증자 Alice는 $e(\sigma_B, H_1(m, x_A PK_B^1)p + PK_B^1)? = e(H_0(w, Cert_A, Cert_B), PK_{A)}^1$을 통해 서명을 검증한다.

4.2 Phase 2

그다음으로 Label-2와 Label-1의 서명 위임에 관해 설명한다.

1) DelGen

Bob은 $\sigma_{A\to B}$, $Params, Cert_B, Cert_C$로 위임 권한자 $\sigma_{B\to C} = x_B H_0(\sigma_{A\to B}, Cert_B, Cert_C)$를

만든다.

2) DelVer

Charls는 $e(\sigma_{B\to C}, p)$의 유효성을 확인한다.

$e(H_0(\sigma_{A\to B}, Cert_B, Cert_C), PK_B^1)$

3) CBDVProxySign

Charls는 프록시 서명 $(m', \sigma_C, \sigma_{A\to B})$을 만든다.

$\sigma_C = (H_1(m', x_C PK_B^1) + x_C)^{-1}\sigma_{B\to C}$

4) CBDVProxySVer

검증자 Bob은 $e(\sigma_C, H_1(m', x_B PK_C^1)p + PK_C^1)? = e(H_0(\sigma_{A\to B}, Cert_B, Cert_C), PK_{B)}^1$ 을 통해 서명을 검증한다.

4.3 서명 검증

Alice가 Charls의 서명 $(m', \sigma_C, \sigma_{A\to B})$을 검증하는 과정은 다음과 같다.

1) Bob은 Alice에게 서명과 $\sigma_{B\to A} = (H_1(m', x_B, PK_C^1)p + PK_C^1)\sigma_{A\to B}$를 전달한다.

2) Alice는 $e(\sigma_C, \sigma_{B\to A} H_0(w, Cert_A, Cert_B)^{-1}? = e(PK_B^1 H_0(\sigma_{A\to B}, Cert_B, Cert_C), x_A)$를 검증한다.

V. 결론

본 논문에서는 Verma의 제안을 검토했고, UAV의 시스템에 맞게 계층적인 구조로 위임권한을 넘겨주고, 서명을 검증하는 방식을 제안했다.

[참고문헌]

[1] M. Mambo, K. Usuda, and E. Okamoto, “Proxy signatures: Delegation of the power to sign messages”, IEICE transactions on fundamentals of electronics, communications and computer sciences, vol. 79, no. 9, pp. 1338 - 1354, Sep. 1996.

[2] N. Harini, K. Devisetty, and D. Aruna, “Secure Proxy Blind ECDS Algorithm for IoT”, International Journal of Pure and Applied Mathematics, vol. 118, no. 7, pp. 437-445, 2018

[3] N. Kumar, R. Iqbal, S. Misra, and J. J. Rodrigues, “An intelligent approach for building a secure decentralized public key infrastructure in VANET”, Journal of Computer and System Sciences, vol. 81, no. 6, pp. 1042-1058, Sep. 2015.

[4] L. Merino, F. Caballero, J. R. Martínez-de Dios, I. Maza, and A. Ollero, “An unmanned aircraft system for automatic forest fire monitoring and measurement”, Journal of Intelligent & Robotic Systems, vol. 65, no. 1-4, pp. 533-548, Aug. 2012.

[5] G. York and D. J. Pack, “Ground target detection using cooperative unmanned aerial systems”, Journal of Intelligent & Robotic Systems, vol. 65, no. 1-4, pp. 473-478, Aug. 2012.

[6] I. Maza, F. Caballero, J. Capitán, J. R. Martínez-de-Dios, and A. Ollero, “Experimental results in multi-UAV coordination for disaster management and civil security applications”, Journal of Intelligent & Robotic Systems, vol. 61, no. 1-4, pp. 563-585, Dec. 2011.

[7] G. K. Verma, B. B. Singh, N. Kumar, and D. He, “CB-PS: An Efficient Short-Certificate-Based Proxy Signature Scheme for UAVs”, IEEE Systems Journal, pp. 1-12, Jul. 2019

[8] B. Semal, K. Markantonakis, and R. N. Akram, “A Certificateless Group Authenticated Key Agreement Protocol for Secure Communication in Untrusted UAV Networks”, IEEE/AIAA 37th Digital Avionics Systems Conference, pp. 1-8, Sep. 2018

소프트웨어 정의 네트워크(SDN)를 대상으로 한 퍼즈테스팅의 평가

위성일, 김현태, 이현주, 손수엘
한국과학기술원 정보보호대학원

Evaluating Fuzz Testing on Software-Defined Network

Seongil Wi, Hyuntae Kim, Hyunjoo Lee, Sooel Son
Graduate School of Information Security, KAIST

요 약

차세대 네트워크 시스템으로서 급부상하고 있는 소프트웨어 정의 네트워크(SDN: Software-Defined Network)의 보안 점검은 매우 중요하다. 최근 들어, 효율적인 SDN의 보안 점검을 위하여, 자동화된 소프트웨어 테스팅 기법 중 하나인 퍼즈테스팅을 SDN에 적용하는 연구가 다수 제안되고 있다. 퍼즈테스팅은 무작위성을 지니고 있기 때문에, 테스팅 도구에 대한 성능을 철저하게 평가해야 하는 테스팅 기법 중 하나이다. 본 논문에서는 SDN을 대상으로 퍼징을 수행한 도구들에 대하여 어떤 방식으로 성능평가가 진행되었는지 살펴보고자 한다. 또한, 각 평가방식의 한계점을 파악하고 앞으로의 연구 및 평가의 방향성을 제시한다.

I. 서론

최근 각광을 받고 있는 소프트웨어 정의 네트워크(SDN: Software-Defined Network)는 네트워크를 유연하게 관리하기 위하여 등장한 개방형 네트워크 인프라이다. 전통적인 네트워크에서는 패킷의 전달 경로를 계산하는 제어평면(Control Plane)과 계산된 경로에 따라 패킷을 실제로 전달하는 데이터평면(Data Plane)이 한 장비에 같이 운용되어 왔다. SDN은 제어평면과 데이터평면을 분리하고 논리적으로 중앙집권화된 제어평면을 활용함으로써, 기존의 네트워크 구조가 가지고 있는 경직성 및 설정 갱신의 지연과 같은 문제를 해결하였다. 특별히, 관리자는 SDN의 인프라를 통하여 중앙집권화된 시각으로 트래픽을 유연하게 관리 및 설정 할 수 있게 되고, 소프트웨어 프로그래밍을 기반으로 한 네트워크 기능 및 경로 설정을 통해 추상화된 관리가 가능해 진다.

최근, SDN이 차세대 네트워크 시스템으로 자리매김하면서, 이에 대한 사이버 보안의 문제가 대두되고 있다 [13]. 이에 따라, SDN에 대한 보안 점검의 필요성 또한 급증하고 있는 상황이다. 특별히, SDN에 자동화된 테스팅 기법 중 하나인 퍼즈테스팅, 퍼징(Fuzzing)을 적용하는 연구가 계속해서 소개되고 있다 [6], [7], [8], [9], [10]. 퍼즈테스팅은 1990년 Miller [1] 에 의하여 제안된 테스팅 수단으로, 임의의 입력 값을 만들어 내고, 그 입력 값을 통하여 프로그램을 실행함으로써 자동적으로 보안 취약점을 탐색하는 소프트웨어 점검기법이다 [2, 3, 4, 5].

퍼징은 임의적으로 무작위 입력 값을 만들어 내는 특성으로 인하여, 성능평가(Evaluation)의 절차에서 통계적 수치에 의한 객관적 가설 검증이 요구된다 [11]. 이러한 합리적인 평가를 통하여, 제안되는 퍼징 도구의 우수성이 드러날 경우, 해당 시스템의 향상된 보안성에 대해서 검증 및 신뢰할 수 있게 된다.

본 연구에서는 SDN에 퍼징을 적용한 굴지의

<표 1> SDN 퍼징 평가의 정리

Fuzzer Name	Benchmarks	OF Version	Baseline	Open Source	Testing Time	Seed	# of Bugs	Bug Counting
DELTA [6]	Brocade Vyatta, Floodlight, ODL, ONOS	1.0 1.3	✗	✓	238s	?	7	✗
BEADS [7]	Floodlight, ONOS, POX, Ryu	1.0	✗	✓	200h	Ping & Iperf Command	831	# of Request
AIM-SDN [8]	ODL, ONOS	✗	✗	✗	✗	?	12	Manual Analysis
STS [9]	Floodlight, NOX, ONOS, POX, Pyretic	✗	✗	✓	10h	✗	7	Bug Oracle
FSF [10]	Floodlight	1.3	DELTA	✗	24h	Ping & Iperf Command, Grammar-based Packet Generation	2	Manual Analysis

연구들에 대하여, 어떤 방식으로 평가를 진행하고 있는지 살펴보고자 한다. 또한, 기존 연구에서 진행하는 성능평가의 한계점을 지적하고, 발전 방향을 모색하도록 한다.

II. SDN 퍼징 평가의 개요 및 실험 구성

본 연구에서는 SDN 시스템에 퍼징을 적용한 사례들에 대하여, 어떤 방식으로 평가를 진행했는지 알아보기 위해, 최근 5년간 네트워크 및 보안 학회에 게재된 SDN 퍼징 논문을 분석하였다. 조사의 대상이 되는 SDN 퍼저는 다음과 같다: DELTA [6], BEADS [7], AIM-SDN [8], STS [9], FSF [10].

<표 1>은 각 퍼저에 대하여 실험 구성 및 성능평가가 어떻게 이루어졌는지 보여준다. 특별히, <표 1>에서는 각 퍼저에 대하여, 어떠한 컨트롤러를 대상으로 실험을 진행하였는지(Benchmarks), 실험에 이용된 OpenFlow의 버전은 무엇인지(OF Version), 해당 도구와의 비교대상이 되는 퍼저는 무엇인지(Baseline), 제안한 퍼저의 소스코드 공개 여부(Open Source), 퍼저를 이용하여 테스팅을 수행한 시간(Testing Time), 퍼저에 사용된 초기 테스트 입력 값(Seed), 탐지한 버그의 개수(# of Bugs), 버그의 계수 방법(Bug Counting)에 대하여 나타낸다. 각 컬럼에서 ✗ 표시는 해당 요소가 퍼저의 논문에서 언급되지 않음을, ? 표시는 해당 요소가 언급은 되었으나 자세한 내용이 생략되어 있음을 의미한다. Open Source 컬럼에서 ✓ 표시는 해당 퍼저의 소스코드가 공개되어 있음을 내포한다.

<표 2>는 각 퍼저에서 생산한 테스팅 인풋에 의하여 버그가 유발됐는지의 유무를 판단해주는 "버그오라클"의 각 항목을 정리하여 나타낸다. 일반적인 퍼저의 경우, 크래쉬 유무로 버그 오라클을 제정하는 반면, SDN 퍼저의 버그오라클은 표 2에서 볼 수 있듯이 SDN 환경에 초점이 맞추어져, 조금 더 의미론적이고, 복잡한 버그를 다루고 있다 [12].

III. SDN 퍼징 평가의 분석 및 한계점

일반적으로 퍼저의 성능을 평가하기 위하여, 다음과 같은 과정을 거친다 [11]: 퍼저 A에 대한 평가를 위하여 (1) 퍼저의 **비교대상이 되는 퍼저** B를 선정; (2) 어떤 **프로그램**을 대상으로 비교할지 결정; (3) A와 B의 성능을 비교하기 위하여, **측정 지표**를 마련 (보통 버그의 개수로 성능을 비교); (4) 퍼저의 **설정 변수**(예: 초기 입력 값, 테스팅 수행 시간) 결정. 본 장에서는 상술한 평가방식이, SDN을 대상으로 하는 퍼즈 테스팅 도구의 성능평가에서 합리적으로 적용되고 있는지에 대하여 비교 분석하고, 그 한계점을 정리한다.

비교대상의 부재. <표 1>의 "Baseline" 컬럼에서 볼 수 있듯이, 대부분의 연구에서는 타 퍼저와의 비교평가를 수행하지 않고 있다(5개의 퍼저 중 4개가 비교평가를 하지 않고 있다). 유일하게 비교평가를 수행하는 FSF [10] 조차, 직접적으로 버그의 개수를 비교하는 것이 아니라, SDN 컨트롤러 코드의 커버리지가 더 높다는 관찰을 통하여, DELTA [6] 보다 성능이 우위에 있음을 간접적으로 설파하고 있다. 또한, 논문에서 제안된 퍼저가 향후 연구의 비교평가에서 사용 될 수 있도록, 소스코드를 공개를 장려

<표 2> SDN 퍼징에서 사용된 버그오라클

Fuzzer Name	Bug Oracle
DELTA [6]	(1) Application Liveness (2) Controller Liveness (3) Internal-storage Poisoning (4) Inter-host Communication Disconnection (5) OpenFlow Error Messages (6) Switch Disconnection (7) Switch-performance Downgrade
BEADS [7]	(1) Controller-performance Downgrade (2) Inter-host Communication Disconnection (3) Network State (4) OpenFlow Error Messages (5) Switch-performance Downgrade
AIM-SDN [8]	401, 404, 405, 429, 500, 503, 507 HTTP Response
STS [9]	(1) Blackhole Freeness (2) Controller Liveness (3) Inter-host Communication Disconnection (4) Loop Freeness (5) POX ACL Compliance
FSF [10]	(1) Controller Liveness (2) Controller-performance Downgrade (3) Inter-host Communication Disconnection (4) Switch Disconnection

하고 있지만 실제는 그러지 못한 현실을 확인할 수 있다.

일반적인 퍼저들이 경쟁을 통하여 발전되어왔던 것처럼, SDN 퍼징연구가 더욱 견고하게 발전되기 위해서는, 비교대상을 기준으로 하여 우위를 점하는 평가구조가 선행되어야 할 것이다. 또한, 향후 이루어질 SDN 퍼징연구에서는 도구의 소스코드를 공적으로 공개하여, 다른 연구자가 해당 퍼저를 자유롭게 이용할 수 있도록 장려함으로써, 비교평가의 구조가 정착되도록 기여해야 할 것이다.

설정 변수의 주관성. 각 연구에서 버그를 찾기 위하여 퍼저를 돌린 시간은 <표 1>의 “Testing Time” 컬럼에 볼 수 있듯이, 몇 백 초에서 몇 백 시간대로 다양하게 설정되어 있다. 각 퍼저는 이러한 실험 시간을 어떻게 정했는지에 대하여 정당성을 부여하지 않고, 주관적으로 설정하였기 때문에, 제안하는 알고리즘의 효율성을 명확하게 입증하기가 힘든 실정이다.

또한, 어떠한 초기 입력 값을 이용하여 퍼저를 실행시켰는지에 따라서, 퍼저의 성능이 매우 달라지기 때문에 [4], [11], 입력 값을 어떻게 생산했는지 명확하게 알려주지 않거나(<표 1> “Seed” 컬럼의 ✗ 와 ? 표시), BEADS [7] 와 같이 오직 몇 개의 이벤트를 통하여 초기 트래픽을 제한적으로 생산할 경우, 제안하는 알고리즘의 버그 탐지능력을 효율적으로 드러낼 수 없다.

보다 합리적인 평가를 위하여, 각 퍼저마다 동일한 단위 시간(보통 너무 길지도, 짧지도 않은 24시간으로 설정 [11])으로 타임아웃을 설정하여 그 안에서 시간에 따른 버그 탐지 추이를 평가해야 한다. 또한, 초기 입력 값을 어떻게 생산했는지를 명시하고, 비교평가를 수행할 때는 더욱 다양한 입력 값을 대상으로 평가를 진행하여 알고리즘의 우수성을 드러내야 할 것이다.

측정 지표의 주관성 및 부정확성. 현존하는 모든 SDN 퍼저는 성능을 평가하기 위하여, 탐지한 버그의 개수를 직접적으로 계수하는 과정을 거친다(<표 1>의 “# of Bugs” 컬럼). 특별히, 각 퍼저에서는, 버그오라클을 통과한 리퀘스트에 대하여, “Bug Counting” 컬럼에서 보여주는 방법으로 버그를 계수한다. 여기서, 버그를 정의하는 버그오라클과(<표 2>), 버그를 계수하는 방법(<표 1>의 “Bug Counting” 컬럼)이 퍼저마다 다르기 때문에, 퍼저의 비교평가를 정확하게 수행할 수 없다는 한계점이 존재한다.

같은 맥락에서, 버그의 계수방법 자체도 합리적이지 못한 경우가 존재한다. BEADS [7] 는 버그를 유발시키는 리퀘스트의 개수를 버그의 개수로 취급하여 계수하였기 때문에, 실제로 유발된 버그보다 더 많이 계수하여 보고하고 있다. 반면, STS [9] 는 버그오라클의 각 항목을 기반으로 버그의 개수를 셌기 때문에, 실제로 발생된 버그보다 더 적게 계수했을 가능성이 크다. AIM-SDN [8] 와 FSF [10] 는 리퀘스트를 일일이 수동적으로 분석했기 때문에, 타 퍼저 보다는 비교적 정확하게 버그를 계수하였으나, 자동적으로 버그를 찾고 결과 값을 보고해주는 퍼저의 특성에 맞지 않게, 버그의 분석에 대하여 무시할 수 없는 오버헤드가 발생하게 된다.

이에 대한 한계점을 극복하기 위해서, 측정 지표를 하나로 통합시키고, 더욱 합리적인 버그 계수 시스템을 마련해야 한다. 자세하게는, 표준화되고 공통적인 버그오라클을 제정하여, 각 퍼저가 이를 사용하게 함으로써, 유의미한 비교평가를 수행할 수 있도록 해야 한다. 또한, 더욱 정확하고 일관적인 버그 계수를 위하여, 버그마다 일일이 패치가 이루어진 Ground Truth

를 벤치마크로 활용하여, 퍼저의 실험을 진행할 필요가 있다.

IV. 결론

본 논문에서는 SDN시스템에 퍼징을 수행한 연구들에 대하여 어떤 방식으로 평가가 진행되었는지 살펴보고, 각 연구에서의 한계점들을 파악하였다. 성능에 대한 비교평가를 더욱 지향하고, 그를 위한 설정 변수 및 측정 지표 설정에 정당성 및 일관성을 부여함으로써, 이러한 한계점을 극복할 수 있을 것이다.

Acknowledgement

이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00254, SDN 보안 기술개발)

[참고문헌]

[1] Barton P Miller, Louis Fredriksen, and Bryan So. “An empirical study of the reliability of UNIX utilities,” in Communications of the ACM, 1990.

[2] Cha, S.K., Woo, M., Brumley, D.: Program-adaptive mutational fuzzing. In: Proceedings of the IEEE Symposium on Security and Privacy. pp. 725-741 (2015)

[3] Rawat, S., Jain, V., Kumar, A., Cojocar, L., Giuffrida, C., Bos, H.: VUzzer: Application-aware evolutionary fuzzing. In: Proceedings of the Network and Distributed System Security Symposium (2017)

[4] Rebert, A., Cha, S.K., Avgerinos, T., Foote, J., Warren, D., Grieco, G., Brumley, D.: Optimizing seed selection for fuzzing. In: Proceedings of the USENIX Security Symposium. pp. 861-875 (2014)

[5] Zalewski, M.: American Fuzzy Lop. http://lcamtuf.coredump.cx/afl/

[6] Lee, S., Yoon, C., Lee, C., Shin, S., Yegneswaran, V., Porras, P.A.: DELTA: A security assessment framework for software-defined networks. In: Proceedings of the Network and Distributed System Security Symposium (2017)

[7] Jero, S., Bu, X., Nita-Rotaru, C., Okhravi, H., Skowyra, R., Fahmy, S.: BEADS: Automated attack discovery in openflow-based SDN systems. In: Proceedings of the International Conference on Research in Attacks, Intrusions, and Defenses. pp.311-333 (2017)

[8] Dixit, V.H., Doup´e, A., Shoshitaishvili, Y., Zhao, Z., Ahn, G.J.: AIM-SDN: Attacking information mismanagement in SDN-datastores. In: Proceedings of the ACM Conference on Computer and Communications Security. pp. 664-676 (2018)

[9] Scott, C., Wundsam, A., Raghavan, B., Panda, A., Or, A., Lai, J., Huang, E., Liu, Z., El-Has`sany, A., Whitlock, S., et al.: Troubleshooting blackbox sdn control software with minimal causal sequences. ACM SIGCOMM Computer Communication Review 44(4), 395-406 (2015)

[10] Kim, H., Wi, S., Lee, H., Son.: FSF: Code Coverage-driven Fuzzing for Software-Defined Networking. In: Proceedings of the World Conference on Information Security Applications (2019)

[11] Klees, G., Ruef, A., Cooper, B., Wei, S., Hicks, M.: Evaluating fuzz testing. In: roceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp.2123-2138 (2018)

[12] 위성일, 손수엘: 소프트웨어 정의 네트워크(SDN)를 대상으로 한 퍼즈테스팅 관련 연구 조사, 한국정보처리학회 추계학술발표대회, 2018

[13] Yoon, C., Lee, S., Kang, H., Park, T., Shin, S., Yegneswaran, V., Porras, P., Gu, G.: Flow wars: Systemizing the attack surface and defenses in software-defined networks. IEEE/ACM Transactions on Networking 25(6), 3514-3530 (2017)

Newhope의 키 조정 메커니즘을 통한 Apon 등의 프로토콜 구체적 사례 검증[1)]

홍 동 연* 최 락 용** 김 광 조*,**

*카이스트 정보보호대학원 **카이스트 전산학부

Instantiation of Apon *et al.* Protocol using Newhope Key Reconciliation Mechanism

Dongyeon Hong* Rakyong Choi** Kwangjo Kim*,**

*Graduate School of Information Security. KAIST **School of Computing. KAIST

요 약

안전하지 않은 채널에서 통신 상대방과 안전한 소통을 하기 위해 암호시스템을 사용한다. 암호시스템을 사용하기 전 두 사용자 혹은 그룹 내 비밀키 교환이 우선 수행되어야 한다. 근래 두 명이 아닌 그룹 단위의 작업 환경이 많아짐에 따라 그룹 내 키 교환이 중요해지고 있다. Apon 등은 동일한 라운드의 격자 기반 그룹 키 교환 프로토콜을 최초로 제안하였다. 그러나 프로토콜의 키 조정 메커니즘을 구체적으로 설계하지 않고 제안하였다. 본 논문에서는 해당 부분의 구체적인 설계를 통해 프로토콜을 완성하고 정확성과 안전성을 살펴보도록 한다.

I. 서론

1.1 배경

일반적으로 카카오톡, 라인 등 메신저를 사용하여 다른 사용자와 쉽게 대화를 하거나 문서 파일을 주고받는다. [1]에서 일반적인 카카오톡의 대화를 일부 복구할 수 있는 점을 지적했듯이 일반적인 메신저는 정보 노출에 취약하다. 비밀키 암호를 사용하여 암호화 후 정보를 보낼 경우 정보 노출을 방지할 수 있다. 비밀키 암호를 사용하기 위해 사전에 키 교환 프로토콜을 통해 사용자 간 동일한 키를 공유해야 한다. 최근 페이스북의 그룹 채팅과 스카이프를 통해 둘 이상의 사용자가 동시에 화상 통화를 하거나 정보를 주고받는 사례가 많아지고 있다. 두 사용자인 경우와 마찬가지로 그룹 내 안전한 소통을 위해서는 그룹 단위의 키 교환 프로토콜이 사전에 수행되어야 한다. 이를 그룹 키 교환 프로토콜이라 한다. 그러나 대부분의 그룹 키 교환 프로토콜은 이산 대수 문제, 인수분해 문제에 기반하고 있어 양자컴퓨터를 이용한 공격에 내성을 갖추고 있지 않다. 이를 대비해 NIST(미국 국립표준기술연구소)에서 양자컴퓨터에 내성을 갖춘 키 교환 프로토콜의 표준화 작업을 진행하고 있다. 그러나 그룹 키 교환 프로토콜의 표준화 작업은 진행하고 있지 않다. 두 사용자 간 키 교환 프로토콜을 반복적으로 수행함으로 그룹 내 키 교환을 했다고 할 수 있으나 그룹 내 사용자가 N일 때 각 사용자는 $N-1$번의 키 교환 프로토콜을 수행해야 하므로 효율성이 떨어진다. 그룹 키 교환 프로토콜을 사용하여 더 효율적으로 그룹 내 동일한 키를 공유할 수 있으며 [2-4]가

1) 본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행되었습니다. (No.2017-0-00555, 양자컴퓨터 환경에서 래티스 문제를 이용한 다자간 인증키교환 프로토콜 연구)

초기 설정: R_q, $a \in R_q$, σ_1, σ_2를 설정한다.

라운드 1: 사용자 P_i는 s_i, e_i를 χ_{σ_1}에서 샘플링하고 $z_i = as_i + e_i$를 모든 사용자에게 전파한다.

라운드 2: 사용자 P_0는 χ_{σ_2}에서 e_0'를 샘플링하고 남은 사용자 P_i는 χ_{σ_1}에서 e_i'를 샘플링한다.
각 사용자 P_i는 $X_i = (z_{i+1} - z_{i-1})s_j + e_j'$을 모든 사용자에게 전파한다.

라운드 3: 사용자 P_{N-1}은 χ_{σ_1}에서 e_{N-1}''을 샘플링하고 $b_{N-1} = z_{N-2}Ns_{N-1} + (N-1)X_{N-1} + (N-2)X_0 + \cdots + X_{N-3} + e_{N-1}''$을 계산한다.
$recMsg(b_{N-1})$을 통해 (rec, k_{N-1})을 구하고 rec을 모든 사용자에게 전파하고 그룹키 sk_{N-1}을 $Hash(k_{N-1})$을 통해 계산한다.

키 계산: P_{N-1}을 제외한 사용자 P_i는 $b_i = z_{i-1}Ns_i + (N-1)X_i + (N-2)X_{i+1} + \cdots + X_{i+N-2}$를 계산하고 $recKey(b_i, rec)$을 통해 k_i를 구한다. 마지막으로 $Hash(k_i)$를 통해 그룹 키 k_i를 계산한다.

그림 1 ADGK19의 프로토콜

그룹키 교환 프로토콜을 제안하였다. 그 중 [2, 3]이 제안한 프로토콜은 Ring Learning with Errors (RLWE) 문제를 기반으로 설계되었다. 그러나 [2]의 프로토콜 중 일부분은 포괄적으로 설계되어있어 구체적인 설계가 추가적으로 필요하다. 본 논문에서는 해당 부분을 더 구체화하여 [2]를 완성하고자 하며 본 프로토콜의 정확성과 안전성을 살펴보고자 한다.

1.2 논문 구성

본 논문의 구성으로 Ⅱ장은 관련 논문을 살펴보며 Ⅲ장에서 제안 프로토콜에 필요한 배경 지식을 간략히 살펴본다. Ⅳ장은 제안 프로토콜을 다룬다. Ⅴ장에서는 Ⅳ장에서 다루었던 프로토콜의 정확성과 안전성에 대해 다룬다. Ⅵ장에서는 결론과 추후 연구에 대한 가능성을 제시한다.

II. 관련 논문

2.1 Apon 등의 프로토콜

Apon 등[2] (이하 ADGK19)은 Burmester와 Desmedt [5]이 제안한 프로토콜 중 방송형 방식에 RLWE 문제를 적용하였다. [5]와 다르게 ADGK19에선 두 개의 이산 가우시안 분포 (discrete Gaussian distribution) χ_{σ_1}과 χ_{σ_2}를 사용하며 군이 아닌 환 $R_q = Z_q[x]/(x^n+1)$에서 연산을 수행하고 $a \in R_q$과 표준편차 σ_1, σ_2를 공개 파라미터로 설정한다. 프로토콜은 그림 1과 같다. 이때 *recMsg*와 *recKey*를 키 조정 메커니즘 (key reconciliation mechanism)이라 하고 그룹 내 사용자가 동일한 키를 공유하기 위해 보조적인 역할을 한다.

2.2 키 조정 메커니즘

키 조정 메커니즘은 Ding 등[3]이 제일 먼저 제안하였으며 뒤이어 [6, 7]이 같은 아이디어를 사용하여 사용자 간 동일한 키를 공유한다는 걸 증명하였다. 이를 프로토콜의 정확성이라 한다. Ding 등[3]과 Peikert [7]는 Z_q에서 $\{0, 1\}$로 가는 함수를 키 조정 메커니즘으로 사용하며 입력값의 범위에 따라 0과 1을 반환한다. 이와 달리 NewHope[6]은 다른 방식의 키 조정 메커니즘을 설계하는데 4장에서 자세히 다룬다.

III. 배경 지식

3.1 Closest vector problem

격자 기반 암호에서 어려운 문제로 알려져 있으며 그림 2와 같이 격자 L과 벡터 w가 주어져 있을 때 w와 가장 가까운 격자점 v를 찾는 것이다.

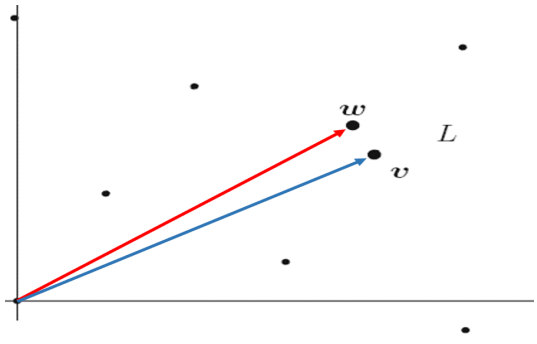


그림 2 Closest Vector Problem[8]

3.2 보로노이 셀

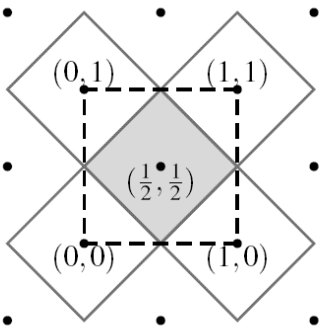


그림 3 보로노이 셀 [2]

보로노이 셀은 격자 L이 주어졌을 때 $V(L)=\{x\in R^n \mid \|x\|<\|x-v\| for\ all\ v\in L\}$로 정의하며 원점과의 거리가 다른 격자점보다 가까운 벡터들의 집합이다. 그림 3에서 원점을 중심으로 한 실선의 사각형이 보로노이 셀이다. 원점 외에 다른 격자점을 중심으로 할 수 있으며 그림 3의 회색 부분은 (1/2, 1/2)을 중심으로 한 보로노이 셀이다.

IV. 제안 알고리즘

본 장에서는 NewHope의 조정 메커니즘을 살펴보고 이를 ADGK19에 적용하여 ADGK19를 구체화한다.

4.1 NewHope의 조정 메커니즘

NewHope의 조정 메커니즘은 두 개의 함수 *HelpRec*과 *Rec*로 나뉘어진다. *HelpRec*은 벡터 $\mathbf{x}\in Z_q^4$와 $b\in\{0,1\}$를 입력을 받아 4차원 격자에서 벡터 $2^r/q\,(\mathbf{x}+b\mathbf{g})$와 가까운 격자점의 계수를 반환한다. 이때 r은 비트수를 나타내며 b는 *HelpRec* 결과값의 편향성을 없애기 위해 사용되며 벡터 $\mathbf{g}$는 $(1/2, 1/2, 1/2, 1/2)^T$이다. 즉, $HelpRec(\mathbf{x}, b)=CVP_{D_4}(2^r/q(\mathbf{x}+b\mathbf{g}))$로 정의할 수 있다. *Decode*는 입력값 벡터 $\mathbf{x_i}\in Z_q^4$와 *HelpRec*의 결과값 $\mathbf{r}\in Z_{2^r}^4$을 사용하여 $(1/2, 1/2, 1/2, 1/2)$의 보로노이 셀 포함 유무에 따라 {0, 1} 중 하나의 값을 반환한다.

4.2 제안 알고리즘

초기설정에서 NewHope과 동일하게 a를 무작위한 값으로 보이도록 하고 프로토콜이 실행될 때마다 새로 설정하여 백도어 공격과 로그잼과 같은 공격을 방지한다. 그림 1의 라운드 3과 키 계산 단계의 *recMsg*와 *recKey*는 NewHope의 *HelpRec*과 *Rec*을 적용하여 rec은 b_{N-1}과 가까운 격자점의 계수이다. 이때 HelpRec과 Rec의 입력 벡터는 4차원이고 b_{N-1}과 b_i는 R_q의 원소로 n차원 벡터이기 때문에 b_{N-1}, b_i를 $n/4$개의 벡터로 나누어 키 조정 메커니즘을 수행한다.

V. 프로토콜의 정확성 및 안전성

그러나 ADGK19과 NewHope에서 서로 다른 분포를 사용하기 때문에 프로토콜의 정확성과 안전성에 문제가 발생할 수 있다. 본 장에서 두 분포의 차이를 최대한 줄일 수 있는 표준편차를 찾아 ADGK19와 NewHope의 정확성과 안전성 증명 모두 적용하고자 한다.

ADGK19는 이산 가우시안 분포를 NewHope은 중앙화된 이항 분포(centered binomial distribution)을 사용한다. 중앙화된 이항 분포는 $\sum_{i=0}^{k-1}(b_i-b_i')$으로 정의한다. 이때 k는 샘플링 횟수로 NewHope과 같이 16으로 설정하며 b_i, b_i'은 1/2의 확률 0과 1의 값을 갖는다. 중앙화된 이항 분포를 다시 정리하면 $\sum_{i=0}^{15}b_i-\sum_{i=0}^{15}b_i'$이며 두 이항 분포의 차이로 볼 수 있다. 각 이항 분포를 기댓값과 분산이 각각 0과 4인 가우시안 분포로 근사한 후 차이를 보면 기댓값이 0이고 분산이 8인 가우시안 분포로 근사할 수 있다. 근사한 가우시안 분포와 중앙화된 이항 분포를 비교하면 그림 4와 같으며 차이가 거의 없음을 알 수 있다. 이산 가우시안 분포는 가우시안 분포를 따르므로 ADGK19와 NewHope에서 보인 정확도와 안전성 증명을 적용할 수 있다.

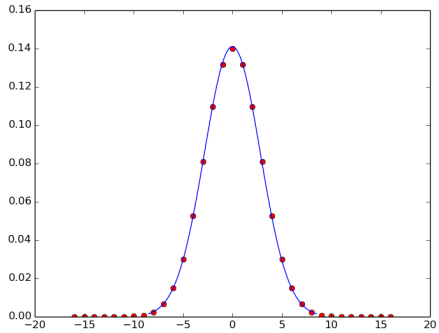


그림 4 가우시안 분포(파란색 선)과 중앙 화된 이항 분포(빨간점) 비교 그래프

VI. 결론 및 향후 과제

본 논문에서는 ADGK19에서 제안한 그룹 키 교환 프로토콜을 NewHope의 키 조정 메커니즘을 적용하여 구체화하였다. 또한 ADGK19과 NewHope은 서로 다른 분포를 사용하여 제안 프로토콜의 정확성과 안전성이 보장되지 않아 ADGK19와 NewHope에서 보인 정확성과 안전성을 동시에 만족하는 표준편차를 제안하였다.

최근 격자 기반 키 교환 프로토콜에 키 재사용 공격 [9-11]이 이루어지고 있으며 특히 [9]는 NewHope에 키 재사용 공격을 제안하였다. 향후 연구 과제로 [9]의 공격을 본 논문에 적용해보고 [13]에서 제안한 pasteurization 기법을 통해 키 재사용 공격 방지를 제안할 예정이다.

[참고문헌]

[1] 이나비, 김광조, 디지털 포렌식을 위한 Android 및 Windows 환경에서 카카오톡 메시지의 아티팩트 분석 (Ⅰ), 2019 정보보호학술발표회논문집 충청지부, 2019.

[2] D.Apon, D.Dachman-Soled, H.Gong, & J.Katz, Constant-round group key exchange from the Ring-LWE assumption, PQCRYPTO 2019, 2019.

[3] J.Ding, X.Xie, & X.Lin, A Simple Provably secure key exchange scheme based on the learning with errors Problem, IACR Cryptology ePrint Archive, 688, 2012.

[4] R. Choi & K. Kim, A novel non-interactive multi-party key exchange from homomorphic encryption, in ProvSec Workshop 2018, 2018.

[5] M.Burmester & Y.Desmedt, A secure and efficient conference key distribution system, In Workshop on the Theory and Application of Cryptographic Techniques (pp.275-286), Springer, 1994.

[6] E.Alkim, L.Ducas, T.Pöppelmann, & P.Schwabe, Post-quantum key exchange—a new hope, In 25th USENIX Security Symposium (pp. 327-343), 2016.

[7] C.Peikert, Lattice cryptography for the internet, In international workshop on post-quantum cryptography (pp. 197-219), Springer, 2014.

[8] J.Hoffstein, J.Pipher, & J.H.Silverman, An introduction to mathematical crypto graphy (Vol. 1), Springer, 2008.

[9] S.R.Fluhrer, Cryptanalysis of ring- LWE based key exchange with key share reuse, IACR Cryptology ePrint Archive, 85, 2016.

[10] J.Ding, S.Alsayigh, R.V.Saraswathy, S.Fluhrer, & X.Lin, Leakage of signal function with reused keys in RLWE key exchange, In 2017 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE, 2017

[11] C.Liu, Z.Zheng, & G.Zou, Key reuse attack on newhope key exchange protocol, In International Conference on Information Security and Cryptology (pp. 163-176). Springer, 2018.

[12] J.Ding, P.Branco, & K.Schmitt, Key exchange and authenticated key exchange with reusable keys based on RLWE assumption, IACR Cryptology ePrint Archive, 665, 2019.

랜덤 메모리 할당을 이용한 향상된 스택 메모리 보호 기법*

진홍주*, 박문찬*, 이동훈*

*고려대학교 정보보호학과

Improved Stack Memory Protection Technique Using Random Memory Allocation*

Hongjoo Jin*, Moon Chan Park*, Dong Hoon Lee*

*Graduate School of Information Security, Korea University.

요 약

메모리 손상 취약점은 서버 프로그램부터 사물 인터넷(IoT) 디바이스까지 다양한 환경에서 발생한다. 특히 임베디드 시스템을 기반으로 하는 사물 인터넷 제품의 시장 규모가 성장함에 따라 스택 메모리 취약점의 발생 또한 증가하는 추세를 보이고 있다. 스택 메모리 취약점을 이용하면 스택에 저장된 반환 주소를 변조할 수 있고, 변조된 반환 주소를 이용하여 프로그램의 제어를 조작할 수 있다. 이와 같은 공격으로부터 스택 메모리를 보호하기 위해 제안된 SafeStack 기법은 기법의 효율성과 보안성을 인정받아 상용 제품의 보안 확장으로 채택되었다. 하지만 SafeStack은 실행 시간에 효과적인 메모리 전수 조사 공격을 수행하는 할당 오라클 공격에 의해 보안성이 크게 낮아지는 것이 드러났다. 본 논문에서는 할당 오라클 공격으로부터 SafeStack을 보호하는 Satellite 기법을 제안한다. Satellite 기법은 초당 평균 1,181번의 랜덤 메모리 할당을 수행하여 할당 오라클을 효과적으로 방해한다. 또한, SPEC CPU2017 벤치마크에서 평균 1.66 % 의 성능 오버헤드를 부과하는 효율적인 기법이다.

I. 서론

메모리 손상(Memory Corruption) 취약점은 다량의 데이터를 처리하는 서버 프로그램부터 임베디드 시스템을 기반으로 하는 사물 인터넷(IoT) 제품까지 다양한 환경에서 발생한다. 특히 스택 메모리 영역에서 발생하는 스택 메모리 손상(버퍼 오버플로, 버퍼 오버라이트 등)은 공격자에게 유용한 도구로써 사용된다[1][3][4]. 최근, IoT 시장 규모가 성장함에 따라 스택 메모리 취약점의 발생 또한 증가하는 추세를 보이고 있다(Fig. 1). 따라서 효율적이고 효과적인 스택 메모리 보호 기법의 연구 및 개발이 중요하다.

힙 메모리와 달리 스택 메모리는 메타데이터를 유지하지 않고 명령어 레벨에서 프로세스 메모리를 직접 관리한다. 그렇기 때문에 메모리 관리 측면에서 효율적이지만 단순한 구조로 인해 스택 메모리에 저장된 데이터가 쉽게 변조될 수 있다. 외부 공격자는 이러한 특성을 악용해서 프로그램의 제어 흐름을 결정하는 반환 주소(Return Address)를 변조하여 프로그램의 제어를 탈취하는 제어 흐름 하이재킹(Control-flow Hijacking) 공격을 수행한다[1][2][5]. 제어 흐름 탈취에 성공한 공격자는 권한 상승, 쉘 코드 실행 등 시스템에 치명적인 행위를 수행할 수 있다.

스택 메모리에 저장되는 반환 주소를 보호하기 위해 Stack Cookies[7], Shadow Stack[8], SafeStack[9] 등 다양한 기법들이 제안되었다. Stack Cookies[7]는 프로그램 성능에 거의 영향을 주지 않는 효율적인 기법이지만 낮은 보안 강도를 제공한다. 반면에 Shadow Stack[8]은 반환 주소의 무결성을 보장하는 강력한 보호 기법이지만 높은 성능 오버헤드를 부과한다. SafeStack[9]은 앞서 설명한 두 기법의 절충으로, 효과적인 스택 메모리 보호를 제공하면서 좋은 성능을 제공하는 기법이다. SafeStack은 낮은 성능 오버헤드로 효과적인 보호를 제공함을 인정받아 상용 컴파일러 프레임워크인 LLVM[10]에서 기존의 스택

* 이 성과는 2019년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.NRF-2017R1A2B3009643).

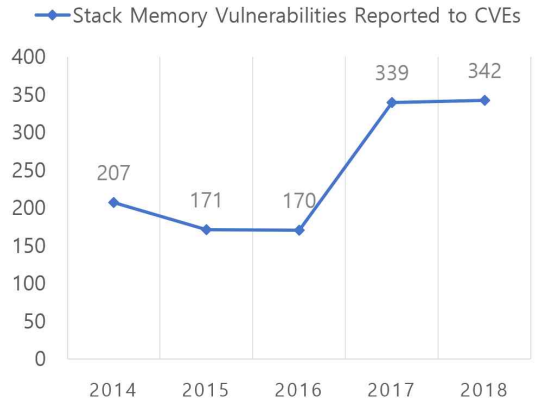


Fig. 1. CVE [6] ID가 부여된 스택 메모리 취약점의 연도별 추이.

보호 기법인 Stack Cookies[7] 대신 채택되었다.

SafeStack은 LLVM 컴파일러의 보안 확장으로 채택된 이후 큰 주목을 받았고, SafeStack에 대한 공격을 수행하는 연구와 SafeStack을 보완하는 연구들이 활발하게 진행되었다. 후속 연구 중 할당 오라클(Allocation Oracle)[11]은 실행 시간에 SafeStack의 보안 속성을 파괴하는 공격 기법이다. 주소공간의 큰 영역이 제공하는 엔트로피에 숨겨진 SafeStack의 위치를 할당 오라클을 이용하여 효과적으로 찾아냈다. 본 논문에서는 실행 시간에 할당 오라클 공격으로부터 SafeStack을 보호하는 기법을 제안하고, 기법의 보안성 분석 및 성능 평가를 수행한다.

II. 배경

2.1 SafeStack

SafeStack[9]은 메모리 격리(Memory Isolation)와 정보 은닉(Information Hiding)을 결합하여 스택 메모리의 반환 주소를 보호하는 기법이다. 일반적으로 버퍼 오버플로는 인접한 스택 객체를 덮어쓰기 때문에 문제가 되며, 버퍼 오버플로를 이용하여 반환 주소를 변조하는 공격이 빈번하게 발생한다. SafeStack은 안전한 객체(Safe Object)와 안전하지 않은 객체(Unsafe Object)를 분류하고, 각각 2개의 스택(SafeStack, UnSafeStack)에 저장하여 메모리를 격리한다. 버퍼 오버플로 가능성이 있는 안전하지 않은 객체는 UnSafeStack에 저장하고, 반환 주소와 같은 안전 객체는 SafeStack에 저장하여 반환 주소를 보호한다. 또한, 표준 시스템 방어 메커니즘인 ASLR(Address Space Layout Randomization)[12]을 이용하여 가상 주소 공간의 큰 엔트로피(x86-64의 경우 2^{28})에 SafeStack의 위치를 숨긴다. 이러한 정보 은닉은 프로그램이 메모리에 로드될 때마다 스택, 힙 등 메모리 세그먼트의 주소를 임의로 배치하여 메모리 전수 조사 공격(반환 주소가 저장된 주소 값을 추측하여 변조하는 공격)을 효과적으로 막는다. Fig. 2.에서 SafeStack이 적용된 프로그램의 메모리 레이아웃을 확인할 수 있다.

2.2 할당 오라클

할당 오라클[11]은 ASLR에 기반 하는 정보 은닉 속성이 제공하는 보호 강도(2^{28}의 엔트로피)를 효과적으로 줄이는 공격 기법이다. 이 기법은 가상 메모리 공간의 모든 Hole 크기를 식별하면 Hole이 아닌 나머지 부분의 레이아웃을 쉽게 추론할 수 있다는 관찰에 기반하는 공격이다(Fig. 2.). 서버 프로그램에서 공격자가 원하는 크기의 메모리를 자유롭게 할당할 수 있게 하는 할당 오라클을 찾고, 할당 시도를 반복하여 할당 가능한 Hole의 크기를 구한다. 수많은 메모리 할당 시도와 할당 된 메모리 크기를 이용한 이진 탐색 알고리즘으로 효율적으로 모든 Hole을 식별할 수 있다. 프로세스의 Hole을 식별한 공격자는 각각의 메모리 세그먼트의 할당 특성을 이용하여 전체 메모리 레이아웃을 얻어낸다. 일반적인 경우 평균 12분 만에 숨겨진 SafeStack의 위치를 식별하였으며[11], SafeStack의 위치(주소)가 노출되면 저장된 반환 주소를 쉽게 변조할 수 있다.

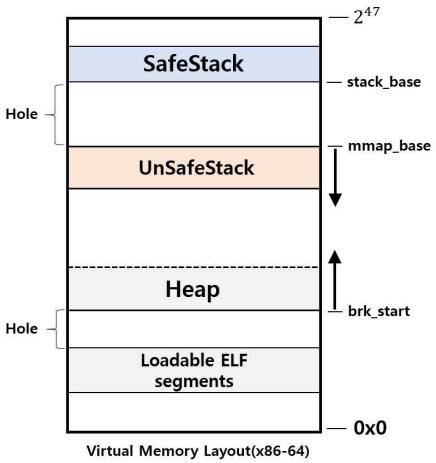


Fig. 2. SafeStack 기법이 적용된 프로세스 메모리(SafeStack, UnSafeStack, Heap 등 할당된 영역들 사이의 할당되지 않은 영역을 Hole이라 지칭함).

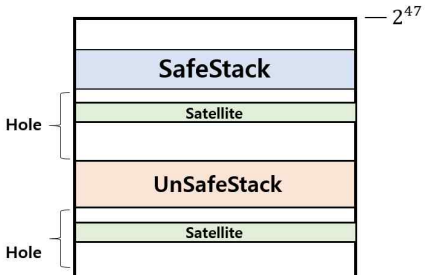


Fig. 3. Satellite 기법의 할당 예. 실행 시간에 Satellite의 위치를 변화시켜 Hole의 크기를 바꾼다.

III. 설계 및 구현

3.1 Satellite

ASLR에 기반 하는 SafeStack 기법은 프로그램 실행 시 메모리에 로드될 때마다 SafeStack과 다른 구성요소들을 임의의 주소에 배치한다. 하지만 할당 오라클은 로드 타임 이후 실행 시간에 지속적인 메모리 할당 요청을 수행하여 Hole의 크기를 알아내는 작업을 수행한다. 그렇기 때문에 ASLR에 의존한 정보 은닉 속성을 쉽게 저해하고, 큰 주소공간에 SafeStack을 숨기는 엔트로피를 크게 낮춘다. 따라서 실행 시간에 Hole 크기를 알아내는 작업을 막거나 방해할 수 있으면 SafeStack의 정보 은닉 속성을 보호할 수 있다.

본 논문에서는 SafeStack에 대한 할당 오라클 공격을 효과적으로 막는 보호 기법인 Satellite를 제안한다. Satellite는 랜덤 메모리 할당을 이용한 기법으로, 2개의 작은 메모리 조각(4KB)을 주소 공간에 배치한다. 실행 시간에 임의의 시간과 임의의 주소에 메모리를 할당 해제 및 재할당하여 Hole의 크기를 지속적으로 변화시킨다. 평균 12분 동안 메모리 할당과 해제를 반복하여 Hole의 크기를 찾는 공격자에 대하여 짧은 주기로 주소 공간에 Satellite를 랜덤 할당시킨다. Satellite는 Fig. 3.과 같이 주소 공간에 배치되며 임의의 주기로 할당되지 않은 주소 공간에 랜덤하게 재배치된다. 기법을 실험 및 평가하기 위해 LLVM 컴파일러에 Satellite 모듈을 추가하였다.

```
1 srand((unsigned int)time(NULL));
2 satA_base = rand()%0x7ffff7ffffff + 0x100000000;
3 satA_mmap = mmap(satA_base, satA_Size, PROT_READ |
  PROT_WRITE, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);
4 srand((unsigned int)time(NULL));
5 satB_base = rand()%0x7ffff7ffffff + 0x100000000;
6 satB_mmap = mmap(satB_base, satB_Size, PROT_READ |
  PROT_WRITE, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);
```

Fig. 4. 라인 1-2와 4-5는 Satellite가 임의의 주소에 할당되도록 난수를 생성하는 작업이다. 생성된 베이스 주소 *SatA_base/SatB_base*를 인자로 사용하여 라인 3과 6에서 *mmap()*을 호출한다.

3.2 Implementation

Satellite는 컴파일 시간에 LLVM 중간 표현(Intermediate Representation, IR) 코드를 다시 작성하는 코드 재작성기(Code Rewriter)와 실제 동작을 수행하는 Satellite 모듈로 구성된다. 코드 재작성기는 원본 프로그램에 Satellite 모듈을 초기화 시키며, Satellite 모듈을 호출하는 코드를 삽입한다. *call Satellite()*는 컴파일러 사용자가 호출 빈도를 지정할 수 있으며, 시스템의 성능과 필요한 보안 강도에 따라 조절할 수 있다. Satellite 모듈은 *munmap()*과 *mmap()* 시스템 콜을 사용하여 4KB 크기의 Satellite 메모리를 할당 해제 및 다시 할당한다. *mmap()*으로 Satellite를 할당할 때 *mmap_base* 주소로 난수를 사용하여 랜덤 메모리 할당을 구현한다. Fig. 4.에서 Satellite의 랜덤 메모리 할당을 지원하는 코드 부분을 확인할 수 있다.

IV. 평가

본 논문에서는 Satellite 모듈의 보안성과 성능을 평가하기 위해 SPEC CPU2017 벤치마크[13]를 사용한다. SPEC은 널리 사용되는 프로그램을 모아놓은 벤치마크 플랫폼이며, 다양한 입력 값에 대한 성능 측정을 제공한다. 우리는 SafeStack과 Satellite 모듈을 각각 14개 벤치마크 프로그램에 적용하여 비교 및 분석하였다.

4.1 Security Analysis

할당 오라클의 공격 작업은 수분동안 메모리 할당 시도를 반복한다. 메모리 할당을 시도할 때 할당 크기는 이진 탐색 알고리즘에 따라 정해지며, 크고 작은 메모리의 할당 요청을 반복한 후 Hole의 크기를 알아낸다. 따라서 랜덤 메모리 할당이 할당 오라클의 공격 시간 동안 이루어지면 공격을 방해할 수 있다. 우리의 실험에서 Satellite는 초당 평균 1,181번의 랜덤 메모리 할당을 수행하여 할당 오라클을 효과적으로 방해한다(Table 1).

Table 1. SPEC CPU2017 벤치마크에 대한 성능 및 할당 빈도 측정.

Program (SPEC CPU2017)	SafeStack (s)	SafeStack + Satellite (s)	Over head	Frequ ency
500.perlbench	309.19	311.22	0.7 %	3
502.gcc	343.46	359.70	4.7 %	8827
505.mcf	700.71	695.38	-0.8 %	963
508.namd	176.19	177.06	0.5 %	1189
511.povray	295.80	303.46	2.6 %	243
519.lbm	1340.90	1336.93	-0.3 %	5
520.omnetpp	791.97	799.26	0.9 %	116
523.xalancbmk	368.14	392.27	6.6 %	292
525.x264	192.75	196.03	1.7 %	2171
531.deepsjeng	229.03	239.67	4.7 %	2188
538.imagick	270.40	270.83	0.2 %	52
541.leela	349.49	353.85	1.3 %	175
544.nab	261.34	261.75	0.2 %	100
557.xz	412.55	413.28	0.2 %	215
Average			1.66 %	1181

4.2 Performance Analysis

SafeStack에 Satellite가 추가되었을 때의 성능 오버 헤드를 평가하기 위해 SafeStack만 적용된 벤치마크 프로그램과 비교한다. 우리는 x86-64 아키텍처인 Intel i7-9700k 3.60GHz 8-Core CPU에 32GB 메모리를 사용하는 환경에서 실험을 진행하였다. 사용된 LLVM 9.0.0 버전은 보안 확장으로 SafeStack을 지원하며, 컴파일 패스에 -*fsanitize=safe-stack* 를 추가하여 적용하였다. SPEC CPU2017 벤치마크 프로그램들을 각각 8번 실행시킨 평균값을 평가에 사용하였으며, 평균 오버헤드는 1.66 % 로 성능 영향이 미미한 것을 확인하였다. Table 1.에서 벤치마크 프로그램에 대한 성능 측정치를 확인할 수 있다.

V.결론

본 논문에서는 할당 오라클에 대한 스택 메모리 보호 기법인 Satellite를 구현 및 평가하였다. Satellite는 평균 1,181번의 초당 랜덤 할당을 제공하며, 1.66 %의 성능 오버헤드로 실행시간에 효과적으로 SafeStack을 보호할 수 있다. 하지만 Satellite 기법은 모듈 호출 횟수를 프로그램마다 지정해야하기 때문에 편의성이 떨어지는 단점이 있다. 따라서 향후 연구로 Satellite 기법을 멀티 쓰레딩 방식으로 구현할 예정이다. 일정한 시간 주기의 랜덤 메모리 할당을 지원하여 모듈 호출에 따른 성능 부하를 줄이고, 사용자 편의성을 높일 것이다.

[참고문헌]

[1] A. One, “Smashing the stack for fun and profit,” Phrack magazine, 1996.

[2] H. Shacham, “The geometry of innocent flesh on the bone: Return-into libc without function calls (on the x86),” in CCS ’07, 2007.

[3] R. Roemer, E. Buchanan, H. Shacham, and S. Savage, “Return-oriented programming: Systems, languages, and applications,” TISSEC, 2012.

[4] S. Checkoway, L. Davi, A. Dmitrienko, A.-R. Sadeghi, H. Shacham, and M. Winandy, “Return-oriented programming without returns,” in CCS ’10, 2010.

[5] Nergal, “The advanced return-to-lib(c) exploits: Pax case study,” Phrack magazine, 2001.

[6] CVE, https://cve.mitre.org/index.html.

[7] C. Cowan, C. Pu, D. Maier, J. Walpole, P. Bakke, S. Beattie, A. Grier, P. Wagle, Q. Zhang, and H. Hinton, “Stackguard: Automatic adaptive detection and prevention of buffer-overflow attacks.” in USENIX SEC ’98, 1998.

[8] J. Xu, Z. Kalbarczyk, S. Patel, and R. K. Iyer, “Architecture support for defending against buffer overflow attacks,” 2002.

[9] V. Kuznetsov, L. Szekeres, M. Payer, G. Candea, R. Sekar, and D. Song, “Code-pointer integrity,” in OSDI ’14, 2014.

[10] C. Lattner and V. Adve, “Llvm: A compilation framework for lifelong program analysis & transformation,” in CGO, 2004.

[11] A. Oikonomopoulos, E. Athanasopoulos, H. Bos, and C. Giuffrida, “Poking Holes in Information Hiding,” in USENIX SEC, 2016.

[12] PaX Team, PaX address space layout randomization (ASLR). http://pax.grsecurity.net/docs/aslr.txt.

[13] SPEC CPU2017, https://www.spec.org/cpu2017.

커널 소프트웨어에 대한 방어 기법 연구 동향*

채수민*, 진홍주*, 이동훈*
*고려대학교 정보보호학과

Research trends of Defense about Kernel Software

Sumin Chae*, Hongjoo Jin*, Dong Hoon Lee*
*Graduate School of Information Security, Korea University.

요 약

상용 소프트웨어에 대한 다양한 방어 기법들이 나오고 있으며, 많은 방어 기법들은 애플리케이션보다 낮은 레벨인 커널 소프트웨어 대해 안전하다고 가정한다. 하지만 상용 커널 소프트웨어는 보안보다는 다양한 서비스와 성능 향상을 중심으로 발전하고 있다. 범용성을 향상시키기 위해 적재 가능 커널 모듈을 지원하고, 성능 향상을 위해 멀티 코어, 물리 메모리의 공유 등 다양한 방법에 관한 연구가 진행되어왔다. 커널 소프트웨어의 특성상 모든 서비스 요청은 신뢰되지 않은 사용자에게 받고, 실제로 이에 기반을 둔 커널 소프트웨어의 취약점을 이용한 많은 악용 사례들이 발생하고 있다. 이를 위해 운영체제와 소프트웨어 보안 분야에서 많은 방어 기법을 제시하였다. 본 논문에서는 커널 소프트웨어 공격 메커니즘을 분류하고, 제어 흐름 무결성을 보호하기 위한 방어 기법[1,2,3,4,5,6,7,8,9]에 대해 살펴본다. 이후 커널 소프트웨어에서 고려해야 할 요소에 대해 논하고자 한다.

I. 서론

현대 컴퓨터 시스템에서 커널 소프트웨어는 하드웨어와 애플리케이션의 중간 다리 역할을 하며 운영체제의 핵심 부분이다. 커널 소프트웨어는 하드웨어 프로세스의 보안(e.g., 페이지에 대한 접근 통제)을 책임지며, 여러 프로세스가 유한한 자원을 효율적으로 사용하기 위해 관리를 하고, 추상화를 통해 일관성 있는 인터페이스를 하드웨어에 제공한다. 이러한 많은 기능을 효율적으로 수행하는 과정에서 보안성에 문제가 발생한다. 예를 들면 자원을 효율적으로 관리하기 위해 서로 다른 프로세스가 같은 물리 메모리 공간을 공유하여 보안성 문제가 발생한다. 또한, 장치를 제어하기 위해 커널의 일부분으로 동작하는 장치 드라이버를 이용하여 커널 함수를 사용 후 권한을 상승시키거나 페이지 테이블 엔트리를 수정하여 권한이 없는 페이지에 접근할 수 있게 만들 수 있다.

커널 소프트웨어는 많은 위협에 노출되어 있으며 이에 대한 보안성을 고려해야 한다. 특히 상위 레벨의 애플리케이션에 의해 결정된 제어 흐름을 전달받아 커널 소프트웨어에서 제어 흐름에 대한 무결성을 유지하는 방안은 소프트웨어 보안 분야에서 꾸준히 연구되는 분야이다. 본 논문에서는 커널 소프트웨어의 공격 메커니즘을 분류하고, 제어 흐름 무결성을 보호하기 위한 방어 기법에 대해 살펴본다. 현재까지 나온 연구들을 종합하여 커널 소프트웨어 보안에서 고려해야 할 네 가지 요소에 대해 다룬다.

* 이 성과는 2019년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.NRF-2017R1A2B3009643).

II. 배경지식

2.1 커널 소프트웨어

커널 소프트웨어는 프로그램과 하드웨어 사

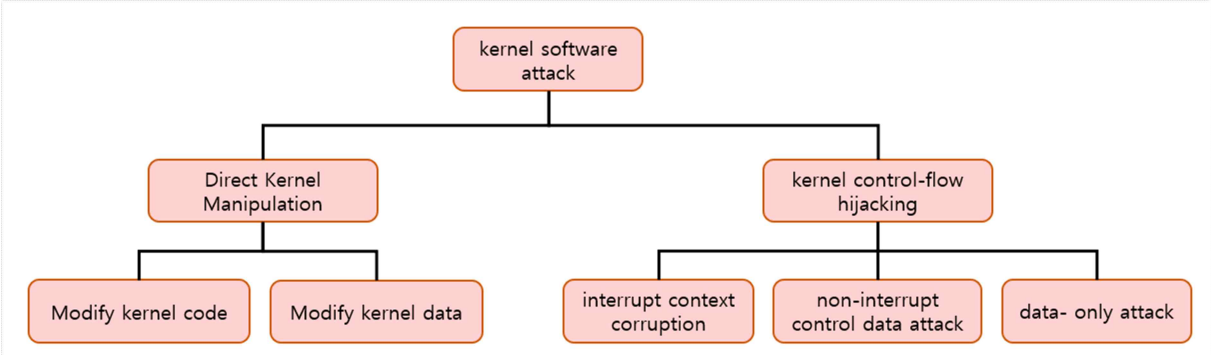


Fig. 1. 커널 소프트웨어의 공격 기법

이의 통신을 관리하는 소프트웨어이다. 다중 작업을 지원하기 위해 프로세스 스케줄링을 하며, 소프트웨어로부터 시스템 호출을 하드웨어가 처리할 수 있도록 변환하는 역할을 한다. 커널 소프트웨어는 대부분이 C언어로 작성되어 있으며, 프로세스 구조에 의존적인 부분들과 성능이 중요한 부분은 기계어 코드로 작성되어 있다. 커널 소프트웨어는 크게 단일 커널 방식과 마이크로 커널 방식이 있다. 마이크로 커널 방식은 커널 공간에서 기본적인 가상 메모리, 스레드, 스케줄링을 관리하고 프로세스 사이의 IPC (inter-process communication)를 지원한다. 이외 장치 드라이버, 파일시스템 관리 등은 외부 모듈화를 통해 커널 공간으로부터 분리하였다. 마이크로 커널과 달리 단일 커널 방식에서는 입출력 기능, 네트워크 기능, 장치 지원 등 여러 기능을 커널 공간에서 실행하는 방식이다.

2.2 VMM(Virtual Machine Monitor)

가상 머신 모니터(하이퍼바이저)는 메인 프레임 컴퓨터에서 다수의 운영체제를 동시에 실행하기 위한 플랫폼이다. 많은 연구에서 커널 소프트웨어의 보안성을 향상시키기 위해 가상 머신 모니터를 기반으로 설계한다. 가상 머신 모니터 기반의 시스템은 게스트 운영체제와 하이퍼바이저 간의 계층 구조를 형성한다. 많은 상용 커널 소프트웨어에서는 단일 커널 방식을 사용하고, 이를 사용하면 장치 드라이버와 같은 적재 가능한 커널 모듈이 커널 소프트웨어 메모리와 같은 공간에 있으므로 자체적인 제한이 불가능하다. 가상 머신 모니터 기반의 시스템의 하이퍼바이저는 더 높은 수준의 보안 권한을 갖게 되어 메모리 접근 및 수정에 대한 접근 통제를 할 수 있게 된다.

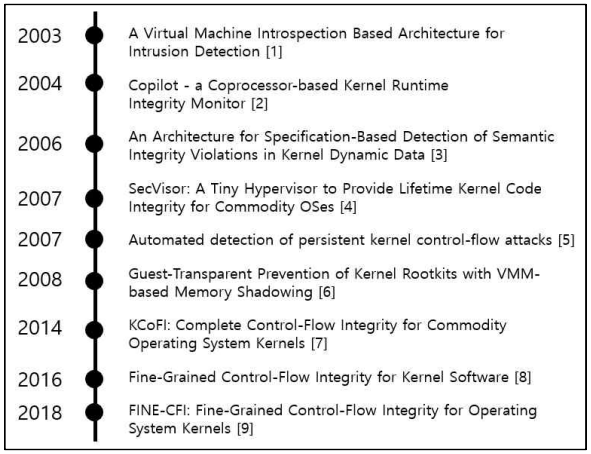
III. 커널 소프트웨어 보안

커널 소프트웨어는 크게 두 가지 공격을 통해 공격자의 목표를 달성할 수 있으며 공격과 방어 기법에 관한 많은 연구가 이뤄졌다. Fig. 1은 커널 소프트웨어의 공격 기법에 대한 그래프이다. 직접 커널 조작(DKM)은 커널 내부의 기존 코드를 수정하거나 중요한 커널 데이터 구조를 수정하는 것을 의미한다. 적재 가능 커널 모듈을 사용하여 기존 커널 코드를 수정하고 시스템에 악영향을 미칠 수 있다. 또한, 커널 영역에서 관리되는 테이블이나 리스트의 값을 수정함으로써 페이지 접근에 대한 권한 상승을 일으킬 수 있다.

이와 더불어 커널 소프트웨어의 유효하지 않은 제어 흐름으로 접근하여 정상적인 제어 흐름을 방해하는 커널 제어 흐름 하이재킹(kernel control-flow hijacking)이 있다. 제어 흐름 무결성은 프로그램의 제어 흐름이 정적으로 계산된 제어 흐름 그래프(control-flow graph)를 벗어나지 않는 것을 의미한다. 먼저 제어 흐름 무결성을 위반하는 커널 소프트웨어 공격에는 상용 소프트웨어에서 발생하는 제어 흐름 하이재킹과 같은 기법인 non-interrupt control data attack이 있다. 커널 소프트웨어의 특성상 기존 소프트웨어와는 다르게 내부/외부 인터럽트를 통해 문맥 전환을 하는 이벤트가 존재하며 이는 임의의 명령어 다음에 발생할 수 있다. 제어 흐름 그래프로 명확히 나타내지 못하기 때문에

과도 근사가 발생할 수 있으며 이러한 취약점을 사용하여 interrupt context corruption을 발생시킬 수 있다. 마지막으로 커널의 제어 흐름에 직접 영향을 미치는 값은 포인터나 반환 주소는 아니지만, 간접적으로 영향을 미치는 데이터에 대한 공격인 data-only attack 또한 커널 소프트웨어에서 발생할 수 있는 위협이다.

Table. 1. 커널 소프트웨어의 주요 방어 기법에 대한 타임라인



이러한 공격에 대응하여 여러 방어 기법이 나왔다. 초기 커널 소프트웨어는 루트킷에 대한 탐지 및 보호에 초점을 맞췄다. 루트킷은 공격자의 존재를 숨김과 동시에 허가되지 않은 컴퓨터나 소프트웨어의 영역에 접근할 수 있게 하는 공격이다. 추가적인 하드웨어 장치를 이용하는 PIC카드 모니터를 사용하는 방법[2]과 의미론적 정보를 사용하는 방법[3]을 통해 호스트 커널의 루트킷을 탐지하였다. 이후 호스트 커널이 루트킷에 공격을 받으면 바로 보호하는 방법이 연구되었다. 이에 대한 방법으로는 많은 연구에서 가상 머신 모니터를 기반으로 하는 시스템 구축하는 것이다.[1,4,5,6,7,8] 커널의 계층화를 이용한 서로 다른 권한 부여가 핵심 아이디어이다.

IV. 제어 흐름 무결성 방어 기법

커널 소프트웨어에 대한 제어 흐름 무결성은 제어 흐름 그래프의 정확도에 따라 Coarse-grained CFI와 Find-grained CFI로 구분된다. Coarse-grained CFI는 상용 프로그램의 주어진 정보의 한계나, 실제 동작 시 성능상의 이유로 인해 정확도가 낮은 제어 흐름 그래프를 통해 제어 흐름의 무결성을 탐지하는 방어 기법이다. Find-grained CFI는 Couarse-grained CFI보다 제어 흐름 그래프가 정확하기 때문에 보안성은 강하지만 추가적인 작업(e.g. 추가적인 하드웨어를 통한 제어 흐름 추적)을 수행하여 성능이 낮다.

SBCFI(state-based control-flow integrity)는 Coarse-grained CFI로써, 제어 흐름의 무결성 검사를 모든 명령어 마다가 아닌 주기적으로 검사하는 방법이다. VMM기반 시스템을 구축하여 프로그램을 실행하고 있는 호스트 커널과 다른 호스트에 모니터 가상 머신(Monitor Virtual Machine)을 독립적으로 운영한다. 이를 통해 제어 흐름 무결성 검사의 독립성을 보장받을 수 있다. 모니터 가상 머신은 두 가지 확인을 통해 제어 흐름의 무결성을 보장한다. 먼저 커널 텍스트가 수정되지 않았는지 확인한다. 부트로더가 로드하는 정적인 커널 소프트웨어와 동적으로 로드되는 적재 가능한 커널 모듈에 대해서 정적 커널 바이너리를 검사하여 결정한다. 두 번째로 도달 가능한 모든 함수 포인터를 확인한다. 함수 혹은 레이블에 대한 간접 호출, 함수 호출을 반환하는 명령어에 대한 추적을 통해 커널 소프트웨어의 제어 흐름 그래프를 만들고 유효성을 확인할 수 있다.

FINE-CFI는 Fine-grained CFI로써, 세 가지 유형(직접/간접적으로 함수를 호출하는 것과 함수를 반환하는 것)을 통해 커널 소프트웨어의 제어 흐름 그래프를 그린다. 직접 함수를 호출하는 것은 정적으로 명시된 오프셋을 통해 그 위치를 확인할 수 있고, 함수를 반환하는 것은 스택에 반환 인자가 들어있는 위치를 통해 알 수 있다. 간접적으로 함수를 호출하는 것은 정적 분석을 통해서는 알기 어려우므로 IR 코드를 사용하여 point-to-set 분석을 실행한다. 커널 소프트웨어의 코딩 특성상 역추적되는 한 함수 포인터가 할당되거나 초기화된 위치를 항상 찾을 수 있으므로 point-to-set 분석을 통해 해당 위치에 대한 유효한 대상 집합을 찾을 수 있다. 이와 더불어 인터럽트 문맥 전환에서 제

어 데이터를 보호하기 위해 가상 환경 모니터 기반의 접근 방식을 사용한다.

V. 커널 소프트웨어의 보안 고려사항

커널 소프트웨어 보안의 고려사항으로는 크게 네 가지로 나눌 수 있다. 첫 번째로 제어 흐름 그래프를 구성할 때, 함수 포인터에 대한 간접 호출을 처리하는 방법이다. 커널 소프트웨어의 함수 포인터에 대한 규칙성과 가정을 통해 유효한 대상 집합을 줄이는 방법을 사용하고 있다. 두 번째로는 내부/외부 인터럽트의 개입에도 일관성을 유지하는 방법이다. 인터럽트가 발생했을 때 커널 소프트웨어의 메커니즘을 분석하고 이를 제어 흐름 그래프에 적용하여 해결하거나, 상용 커널 소프트웨어에 범용적으로 적용하기 위해서 독립적인 호스트 커널을 사용할 수 있다. 세 번째로는 커널의 수정을 최소화하는 것이다. 커널 소프트웨어는 내부 구조가 복잡하게 구성되어있기 때문에 그 시스템을 정확히 알고 있지 않으면 조금의 수정으로도 많은 에러가 발생할 수 있다. 추가 수정을 없애기 위해 추가적인 하드웨어 장치를 설치하여 모니터링 방식을 취할 수 있다. 마지막으로 성능에 대한 중요성이다. 커널 소프트웨어는 애플리케이션과 하드웨어의 중간 매개체이다. 즉 커널 소프트웨어의 성능이 떨어진다면 병목 현상이 발생하기 때문에 성능은 커널 소프트웨어 보안에서 중요한 고려사항이다.

VI. 결론

본 논문에서는 커널 소프트웨어의 공격과 방어에 관한 연구 흐름과 보안성을 위해 어떤 고려사항이 있는지 분석하였다. 상용 커널 소프트웨어에는 다양한 종류가 있다. 많이 사용하는 데스크톱이나 노트북에 있는 커널 소프트웨어, 핸드폰에서 사용하는 커널 소프트웨어 등 용도에 따라서 보안 요구사항에 맞게 커널 소프트웨어는 발전하고 있다. 최근 연구가 활발히 진행되는 물리적인 접근이 가능한 공격자 모델에 대해서도 커널 소프트웨어에서 보안에 고려해야 할 요소로 볼 필요성이 있다.

[참고문헌]

[1] T. Garfinkel and M. Rosenblum, "A Virtual Machine Introspection Based Architecture for Intrusion Detection." In Symposium on Network and Distributed System Security (NDSS), 2003

[2] N. Petroni, NL T. Fraser and W. Arbaugh, "Copilot - a Coprocessor-based Kernel Runtime Integrity Monitor" In USENIX security Symposium, 2004

[3] N. Petroni, NL T. Fraser and W. Arbaugh, "An Architecture for Specification-Based Detection of Semantic Integrity Violations in Kernel Dynamic Data" In USENIX security Symposium, 2006

[4] A. Seshadri, M. Luk and A. Perrig, "SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity OSes" In ACM SIGOPS Operating Systems Review, 2007

[5] N. Petroni and M. Hicks, "Automated Detection of Persistent Kernel Control-Flow Attacks", In ACM CCS, 2007

[6] R. Riley, X. Jiang and D, Xu, "Guest-Transparent Prevention of Kernel Rootkits with VMM-based Memory Shadowing" In International Workshop on Recent Advances in Intrusion Detection(RAID), 2008

[7] J. Criswell, N. Dautenhahn and V. Adve, "KCOFI: Complete Control-Flow Integrity for Commodity Operating System Kernels" In IEEE Symposium on Security and Privacy(S&P), 2014

[8] X. Ge, N. Talele and T.jaeger, "Fine-Grained Control-Flow Integrity for Kernel Software" In IEEE Symposium on Security and Privacy(S&P), 2016

[9] J. Li, X. Tong and J. Ma, "FINE-CFI: Fine-Grained Control-Flow Integrity for Operating System Kernels" In IEEE Transactions on Information Forensics and Security, 2018

Chrome Extension을 활용한 로그인 평문전송 탐지 및 차단 툴 개발

강동혁, 최영근, 박수진, 이장우, 손영락

수원대학교, 가톨릭대학교, 순천향대학교

Developing login plaintext detection and blocking tool using Chrome Extension

Dong Hyuck Kang, Young Geun Choi, Su Jin Park, Jang Woo Lee, Young Rag Son

The University of Suwon, Catholic University of Korea, Soonchunhyang University

요 약

정보통신망법 제 28조(개인정보의 보호조치)에 따르면 정보통신서비스 제공자 등이 개인정보를 처리할 때에는 개인정보의 분실 도난 누출 변조 또는 훼손을 방지하고 개인정보의 안전성을 확보하기 위해 기술적 관리적 조치를 해야 한다고 명시되어 있다. 하지만 이러한 조치를 취하지 않은 곳이 현재까지 많으며, 로그인 평문전송은 최근 유행하는 크레덴셜 스터핑(credential stuffing)에 취약하다. 이에 클라이언트와 관리자가 사용가능한 로그인 평문전송 점검 툴을 개발하여 위험성을 알리고 보안의식을 고취시키고자 한다.

I. 서론

본 논문에서는 정보통신망법 제 28조(개인정보의 보호조치)에 따르면 정보통신서비스 제공자 등이 개인정보를 처리할 때에는 개인정보의 분실 도난 누출 변조 또는 훼손을 방지하고 개인정보의 안전성을 확보하기 위해 기술적 관리적 조치를 해야 한다고 명시되어 있다. 하지만 현재까지도 이러한 조치를 취하지 않은 웹 사이트들이 존재하고 있다.

로그인 평문전송은 최근 유행하는 크레덴셜 스터핑(Credential stuffing) 공격에 취약하다. Wireshark와 같은 네트워크 패킷 캡처 프로그램을 사용하면 네트워크 패킷을 볼 수 있지만, 일반인들에게 이러한 프로그램을 사용하여 패킷을 확인하는 과정에 어려움이 있으리라 판단하였다. 본 논문에서는 클라이언트가 네트워크 전송 구간에서 로그인 정보가 평문으로 전송되는지, 암호화되어 전송되는지 쉽게 확인할 수 있도록 하는 프로그램을 개발하였다. 이 프로그램은 접근성이 뛰어난 Chrome Extension을 기반으로 하고 있다.

본 논문에서는 로그인 평문전송의 위험성을 알리며 클라이언트의 보안의식을 고취시키고자 한다.

II. 본론

2.1 Credential Stuffing 공격

크레덴셜 스터핑(Credential stuffing)은 수집된 사용자의 아이디와 비밀번호를 여러 번 자동 대입하여 사용자 계정에 부정하게 접근하는 공격 방식을 말한다.

이 방법을 통해 여러 해 동안 데이터를 침해하는 사건이 발생하였고, 이를 통해 수십억 개의 로그인 정보들이 해커의 수중에 들어갔다. 해커들은 사용자의 로그인 정보를 입력하여 스팸, 피싱, 계정 탈취 등의 공격을 시도한다. 이처럼 크레덴셜 스터핑(Credential stuffing) 공격

은 사이버 범죄자가 탈취한 로그인 정보를 악용하는 대표적 방식 중 하나이다.

Top Attack Destinations

DESTINATION COUNTRY	ATO HEUR.LOGINS
United States	12,522,943,520
India	1,208,749,669
Canada	1,025,445,535
Germany	760,722,969
Australia	104,655,154
Korea	37,112,529
China	26,173,541
Gibraltar	6,559,360
Netherlands	4,991,790
Japan	3,424,334
Italy	2,601,632
France	1,864,733
Hong Kong	1,305,262

[그림 1] Akamai의 나라별 크레덴셜 스터핑 공격 통계자료

[그림 1]의 통계 자료를 통해 미국은 1위로 약 125억 건의 크레덴셜 스터핑 공격을 받았으며, 한국은 6위로 약 3천 7백만 건의 공격을 받았다.

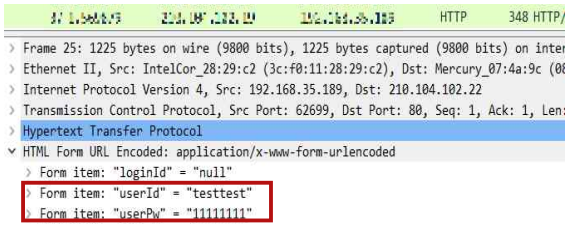
2.2 Wireshark를 이용한 네트워크 패킷 캡처

본 논문에서는 네트워크 패킷 캡처 프로그램 Wireshark를 이용하여 얼마나 많은 사이트들에서 로그인 정보가 평문으로 전송되는지 확인해 보았다.



[그림 2] 웹 사이트의 로그인 창

[그림2]의 사이트는 보안 프로그램을 설치하지만 [그림3]와 같이 Wireshark를 통해 로그인이 평문으로 전송되는 것을 확인 할 수 있다.



[그림 3] 네트워크 패킷 캡처

2.3 Chrome Extension 개발

2.3.1. 개요

HTML5 표준이 제정되기 전에는 비교적 HTML 표준이라는 개념이 모호하고 유연하여 브라우저마다 독자적인 표준이 존재하거나, Flash나 Java Applet 같은 외부 프로그램에 의존하여 동작하는 등의 문제가 많았다. 하지만 최근에는 HTML5 표준이 보편적으로 잘 사용이 되면서 비표준적인 요소들이 많이 사라지게 되었고, 자바스크립트 또한 발전을 거듭하며 단순한 스크립트 언어에서 브라우저의 Assembly Language에 가까운 위상을 얻게 되었다. 이로써 Web은 정적이라는 개념에서 벗어나 자바스크립트로 동작하는 동적인 Web 환경으로 변화하게 되었다. 이는 웹상에서 사용되는 로그인 메커니즘에도 영향을 주었다. 기존에는 정적인 웹에서 동적인 동작을 구현하기 위하여 위에 언급한 Flash, Java Applet과 같은 외부 플러그인을 많이 사용했지만, 이후에는 XMLHttpRequest 객체를 이용한 AJAX 통신이 가능해지게 되었다. 이러한 변화에는 HTML5 표준 발표 → 자바스크립트 모듈화 → jQuery 활성화 → AJAX의 보급 → 자바스크립트 프레임워크의 발전과 같은 흐름이 중심에 있었다고 추측할 수 있다.

2.3.2 프로그램 메커니즘

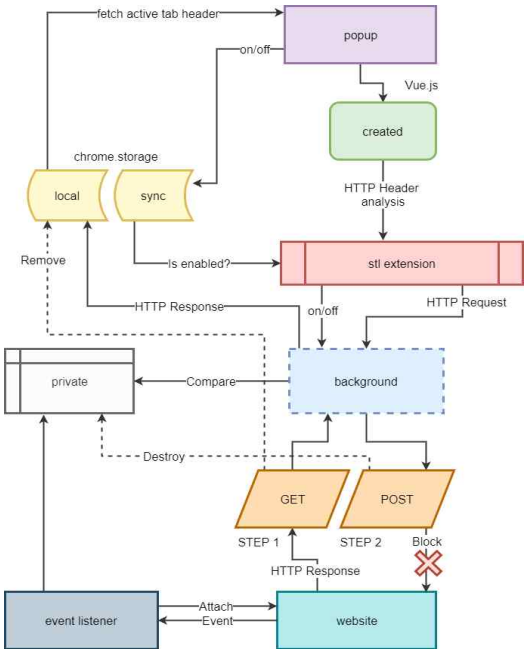
로그인 메커니즘으로 크게 두 가지를 대상으로 개발을 진행하게 된 것도 위 흐름의 영향이 있었다고 할 수 있다.

하나는 form 태그와 button, input[type=submit]을 이용한 x-www-form-urlencoded 전송 로그인이 있고, 나머지 하나로는 자바스크립트를 이용한 xhr 전송 로그인이다.

form 태그를 이용한 전송의 경우에는 내부에

포함된 input 태그의 name과 value에 따라서 name=value 형식으로 전송이 이루어지게 된다.

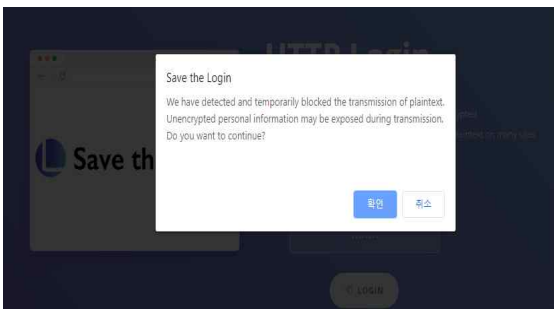
xhr 전송은 이보다 더 복잡한데, 프론트엔드의 구현에 따라서 사이트마다 다른 전송방식과 포맷을 가지고 있고, 마지막에 XMLHttpRequest 객체의 생성을 통해 전송이 이루어진다는 공통점이 있다. (여기서는 크롬 익스텐션의 구현에 관한 내용을 다루기 때문에, MS의 Internet Explorer의 독자적인 ActiveXObject("Microsoft.XMLHTTP")는 예외로 한다.)



[그림 4] flowchart

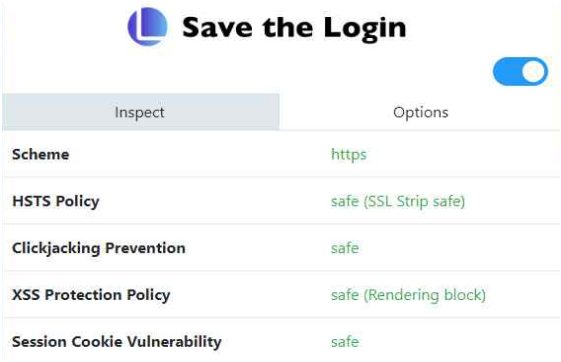
2. 4 Save the Login의 기능

먼저 익스텐션 'Save the Login'의 주 기능은 로그인 평문전송 탐지 및 차단이다.



[그림 5] 로그인 평문전송 탐지

[그림 5]와 같이 클라이언트가 네트워크 전송 전에 로그인 정보가 평문으로 전송된다면 알림을 띄워 경고한다.



[그림 6] 안전한 사이트

[그림 6]은 'Save the Login'의 부가기능이다. 이는 로그인 평문전송을 탐지할 뿐 아니라 팝업창을 통해 여러 가지 웹 설정의 미약한 부분에 대해 알려준다.

```
const url = new URL(item.url);
const scheme = url.protocol.slice(0, -1);

if (scheme === 'https') {
  wrapper.scheme.grade = 'SAFE';
}
this.scheme = scheme;
```

[그림 7] URI Scheme 코드

[그림7]은 URI Scheme은 HTTP와 HTTPS를 구분하여 알려주는 기능이다.

```
// Check HSTS header
  case 'strict-transport-security': {
    if (scheme === 'https') {
      // Get max-age value parsed from HSTS header
      let maxAge = parseInt((/max\-age=([0-9]+)/gi.exec(header.value) || [, '0'])[1]);
      wrapper.hsts_policy.description = `${chrome.i18n.getMessage('moderate')}`;
      wrapper.hsts_policy.grade = 'NORM';
      // Check HSTS header setting is safe from SSL Strip attack
      if (header.value.toLowerCase().indexOf('includesubdomains') !== -1) {
        wrapper.hsts_policy.description = `${chrome.i18n.getMessage(
          'safe'
        )} (SSL Strip safe)`;
        wrapper.hsts_policy.grade = 'SAFE';
      }
    }
    break;
  }
```

[그림 8] HSTS Policy 코드

[그림 8]은 HSTS(HTTP Strict Transport S

ecurity) 정책은 HTTPS 프로토콜만 사용하도록 한다.

```
case 'x-frame-options': {
  wrapper.clickjacking_prevention.description = `${chrome.i18n.getMessage('moderate')}`;
  wrapper.clickjacking_prevention.grade = 'NORM';

  if (
    header.value.toLowerCase() === 'deny' ||
    header.value.toLowerCase() === 'sameorigin'
  ) {
    wrapper.clickjacking_prevention.description = `${chrome.i18n.getMessage('safe')}`;
    wrapper.clickjacking_prevention.grade = 'SAFE';
  }
  break;
}
```

[그림 9] Clickjacking Prevention 코드

[그림 9]는 Clickjacking은 마우스 클릭과 하이재킹의 합성어이다. Clickjacking 공격은 공격자가 사용자가 의도 하지 않은 것을 클릭하게 한다.

```
case 'x-xss-protection': {
  if (header.value.toString() !== '0') {
    wrapper.xss_protection_policy.description = `${chrome.i18n.getMessage('moderate')}`;
    wrapper.xss_protection_policy.grade = 'NORM';
    if (header.value.indexOf('mode=block') !== -1) {
      wrapper.xss_protection_policy.description = `${chrome.i18n.getMessage(
        'safe'
      )} (Rendering block)`;
      wrapper.xss_protection_policy.grade = 'SAFE';
    }
  }
  break;
}
```

[그림 10] XSS Protection Policy 코드

[그림 10]은 XSS(CrossSiteScripting) 보호 정책은 공격자가 작성한 스크립트가 사용자에게 전송되지 않도록 한다.

```
cookiePromise: function(wrapper, url) {
  return new Promise((resolve, reject) => {
    chrome.cookies.getAll({ url: url }, cookies => {
      let flag = false;
      for (let i = 0; i < cookies.length; ++i) {
        // If cookie type is session and httpOnly flag is disabled
        // Vulnerable to session cookie hijacking
        if (cookies[i].session && !cookies[i].httpOnly) {
          // Test cookie name
          const name = cookies[i].name;
          const tests = [/^ID_/i, /_ID$/i, /SESS(ION)?/gi, /LOG(IN|GED)/gi, /ACCOUNT/gi,
            /MEMBER/gi, /AUTH(ORIZED?|ENTICATED?)?/gi, /ADM(IN)?/gi, /^TOKEN_/i, /_TOKEN$/i,];
          if (tests.some(t => t.exec(name))) {
            flag = true;
            break;
          }
        }
      }
      if (!flag) {
        wrapper.session_cookie_xss.description = chrome.i18n.getMessage('safe');
        wrapper.session_cookie_xss.grade = 'SAFE';
      }
      return resolve();
    });
  });
```

[그림 11] Session Cookie Vulnerability 코드

[그림 11]은 세션 쿠키는 일반적으로 세션 ID를 저장하거나 토큰을 액세스하는 데 사용된다. 공격자가 노출되거나 손상되면 사용자를 가장하거나 응용 프로그램에 대한 권한 상승을 시도할 수 있다.

II. 결론

본 논문에서는 로그인 평문전송에 대한 위험성을 일반인들에게 알리고자 개발한 로그인 평문전송 탐지, 차단과 웹 취약점을 알려주는 기능이 있는 Chrome Web Browser 기반의 Chrome Extension 개발에 관해 설명하였다. 이러한 개발을 통해 사용자들이 로그인 평문전송의 위험성에 관심을 가질 수 있도록 하며, 보다 안전한 웹 환경을 이용할 수 있을 것으로 기대한다.

[참고문헌]

[1] 심재문, 김현진, 김설희 외 2명 “Chrome 브라우저 상의 사용자 맞춤 북마크 프로그램 설계 및 구현”, 한국정보과학회 학술발표논문집, 2013.

[2] 조규철, 하진욱, 류성민 “Chrome Web Browser의 Chrome Extension을 활용한 웹 시스템 개발”, 한국컴퓨터정보학회, 한국컴퓨터정보학회 학술발표논문집, 2017년 한국컴퓨터정보학회 하계학술대회 논문집 제25권 제2호 2017. 07.

[3] 강석진, 최진영 “시큐어코딩 활성화를 위한 점검도구 확산 방안에 대한 연구”, 한국정보과학회한국정보과학회 학술발표논문집한국정보과학회 2016년 동계학술대회 논문집.2016.12.

[4] 2019년 인터넷 보안 현황 보고서: 크리덴셜 스터핑 - 공격과 경제, Akamai, 2019.

자체 수정 코드 및 분석 기술 동향*

류승호, 오희국†

한양대학교 컴퓨터공학과

Survey on SMC(Self-modifying Code) and SMC Analysis

Seungho Ryu, Heekuck Oh*

*Department of Computer Science and Engineering, Hanyang University.

요 약

자체 수정 코드(SMC: Self-modifying code)는 런타임에 바이너리 코드를 읽고, 생성 또는 변경하는 프로그램을 말한다. 런타임 코드 생성 및 최적화, 동적 로딩 및 링킹, OS 부트로더, 적시(JIT) 컴파일러, 이진 변환(binary translation), 패커 또는 멀웨어의 동적 코드 암·복호화 및 난독화 등에 사용되며, 세계에서 개발되는 여러 주요 소프트웨어 및 시스템에서 널리 쓰이고 있다. CPU와 같은 HW 수준에서 보면 이는 아주 자연스러운 기술이다. 그러나 문제는 SW 수준으로 올라오면, 코드의 보안성을 위해 고안된 기술인 DEP(또는 W^X)와 같은 것들로 인해 대다수의 개발자 및 기술에 '코드는 수정되지 않는다.'는 암묵적인 가정이 추가된다는 것이다. 또한 SMC는 런타임에 코드를 수정한다는 그 자체의 동작 특성 때문에 분석과 검증에 큰 어려움이 있다. 이로 인해 SMC와 해당 코드를 분석하는 기술은 연구가 부진한 상황이다. 본 논문에서는 최신 커널에 사용되는 SMC 응용 기술과 SMC 분석 기술의 동향에 대해 간략하게 설명하고자 한다.

I. 서론

자체 수정 코드(SMC)는 런타임에 바이너리 코드를 읽고, 생성 또는 변경하는 프로그램을 말한다. 런타임 코드 생성 및 최적화, 동적 로딩 및 링킹, OS 부트로더, JIT 컴파일러, 이진 변환, 패커 또는 멀웨어의 동적 코드 암·복호화 및 난독화 등에 사용되며, 세계에서 개발되는 여러 주요 소프트웨어 및 시스템에서 널리 쓰이고 있다.

예를 들어, 런타임 코드 생성 및 JIT 컴파일은 운영체제 및 기타 응용 프로그램의 성능을 향상 시킬 수 있다. 동적 코드 최적화는 시스템의 성능을 향상 시키거나 시스템 자원이 충분하지 않은 기기에서 메모리에 상주하는 코드 크기를 최소화 할 수 있다. 동적 코드 암·복호화 또는 난독화는 코드 보호 및 변조 방지 소프트웨어나 멀웨어에 사용되어 보호된 바이너리를 분석(디버깅하거나 디컴파일하는 것)을 어렵게 만들 수 있다. 또한 커널의 동적 로드 및 링킹, 부트로더와 같은 운영체제의 핵심 기능이나 이진 변환 및 가상머신(VM: Virtual Machine)과 같은 응용 프로그램에서도 사용된다.

그러나 SMC는 런타임에 코드를 수정한다는 그 자체의 동작 특성 때문에 분석과 검증에 큰 어려움이 있다. 정적분석으로는 실제 수행 시의 코드와 레지스터가 어떻게 변화하는지 알지 못하며 오탐과 미탐이 상당수 존재한다. 동적 분석은 코드의 실제 수행 결과와 변화를 알 수 있지만, 해당 수행경로를 제외한 모든 프로그램의 경로에 대한 분석을 수행하기 어렵다. 또한, 코드의 보안성을 위해 고안된 기술인 DEP(또는 W^X)와 같은 것들로 인해 대다수의 개발자 및 기술에 '코드는 수정되지 않는다.' 라는 암묵적인 가정이 추가됨으로써 인해 SMC와 해당 코드를 분석하는 기술은 연구가 부진한 상황이다. 실제

* 본 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2019R1A2C2003045).

† 교신저자: hkoh@hanyang.ac.kr

최신 역어셈블 도구에서 조차도 SMC는 분석이 어렵다는 이유로 대상에서 제외하고 있다.

본 논문에서는 후속 연구를 위해 최신 커널에 사용되는 SMC 기술과 SMC 분석 기술의 동향에 대해 간략하게 설명하고자 한다. 이후 논문의 구성은 다음과 같다. 2장에서는 커널에 적용된 최신 SMC 기술 몇 가지에 대해 간략히 설명한다. 3장에서는 역어셈블 기술 혹은 멀웨어 탐지 기술 등의 SMC를 분석하려는 연구에 관해 설명한다. 4장에서는 결론과 향후 연구계획에 대해 설명하고 마무리 짓는다.

II. SMC 응용 기술

2.1 Static Keys(또는 Static branches)

커널에는 성능을 위해 SMC를 사용하여 fast-path를 수행하는 기능이 있다. 분기문 검사는 프로세서에서 분기 예측 실패를 일으킬 수 있으며, 이는 큰 성능 오버헤드를 유발하게 된다.

Static keys는 오버헤드 없이 분기문 처리를 구현한 기법으로, 특정한 키값에 따라 다음 명령어를 JUMP 명령어 또는 NOP 명령어로 변환한다. Static keys는 nop 명령어와 두 개의 반환문만을 가지고 있는데, 첫 번째는 항상 false를 반환하고 두 번째는 항상 true를 반환하도록 되어있다.

따라서 이 함수는 기본적으로 false를 반환하고, 키값에 의해 명령어 변환이 이루어 지면 NOP 명령어는 JUMP 명령어로 대체되고 이는 두 번째 반환문의 주소를 가르키므로 결과값은 true가 나오게 된다.

2.2 Alternative

점점 더 많은 하드웨어와 명령어가 프로세서에 추가되고 있지만, 커널은 다양한 플랫폼에서 실행 되어야 하기 때문에 호환성을 위해 새로운 명령어가 추가 되더라도 바로 적용하여 코드를 수정하기 힘들다. 이에 커널에선 성능과 호환성을 조정하기 위해 하드웨어와 프로세서 기능 지원에 따라 최적화된 명령어를 런타임에 자동으로 변경해가며 사용한다.

Alternative란 컴파일 타임에 실행할 명령(old-instruction)을 준비하고, 조건에 따라 기존 명령을 새 명령(new-instruction)으로 대체시켜 동작시키는 기법으로 조건 처리를 했어도 수행 성능에 영향이 가지 않게 하는 기법이다. 예를들면 메모리 배리어를 cpu가 지원하는 명령어 세트에 따라 부트업 시 lock, mfence 등의 코드를 선택하여 사용하도록 수정한다. 이 기술은 Static Keys와 유사하지만 더 폭넓은 사용이 가능하다.

2.3 KGDB

KGDB는 커널 디버거이다. 개발자가 커널 또는 커널 모듈을 디버깅 할 때, GDB와 유사한 디버거 front-end 가 있는 다른 컴퓨터를 사용하여 직렬(serial) 포트 또는 네트워크를 통해 컴퓨터를 원격으로 연결할 수 있다. 디버거 front-end는 원격 GDB 프로토콜에 의해 디버그 된 커널에서 KGDB와 통신하며 KGDB가 실제 명령어 수정 및 작업을 수행하게된다.

프로세서는 하드웨어 디버깅 메커니즘을 제공하지만, 리소스 제한으로 인해 break point 수에 제한이 있다. 이를 해결하기 위해 일반적인 디버거에서 흔히 사용하는 방법이 소프트웨어 break point 이다. 소프트웨어 메커니즘을 사용하면 break point 지점의 명령어를 커널 실행 중단을 위한 인터럽트 코드 INT3 명령어로 수정한다. 명령어 바이트가 INT3 명령어로 대체 되면, KGDB는 해당 바이트의 주소와 원본값을 기록하고 실제 코드, 수행시 혹은 break point가 수정될 때 이를 통해 다시 원본 바이트 값을 복구한다.

2.4 Ftrace

Ftrace는 시스템 개발자와 설계자가 커널 내부에서 진행되는 작업을 찾을 수 있도록 설계된 커널 추적용 프로그램이다. 사용자 공간 이외에서 발생하는 대기 시간 및 성능 문제를 디버깅 또는 분석하는 데에도 사용할 수 있다. 일반적으로 Ftrace는 커널을 실시간 패치(e.g. 우분투의 Canonical Livepatch 등)

하는 데 사용된다.

Ftrace를 통해 컴파일러는 추적할 수 있는 함수의 진입점에 추가적인 CALL 명령어를 삽입한다. 부팅시 해당 지점은 NOP 명령어로 대체되고, trace point가 활성화되면 함수의 진입점에 있는 NOP 명령어를 다시 CALL 명령어로 수정한다. Static keys, KGDB와 마찬가지로 명령어 수정 프로세스는 INT3 명령어를 통해 처리되는 것은 유사하지만, Ftrace로 호출 된 함수는 trampoline 방식으로 처리된다는 다른 특징이 있다.

2.5 Kprobes

Kprobes를 통해 개발자는 모든 커널 루틴을 동적으로 수정할 수 있고, 커널의 중단없이 디버깅 및 성능 정보를 수집할 수 있다. 또한 break point에 도달했을 때 호출될 핸들러 루틴을 지정하여 대부분의 커널 코드 주소에서 trap 할 수도 있다. 일반적으로 Kprobes도 앞서 설명한 기술들과 유사하게 INT3 명령어를 통해 이를 처리한다.

III. SMC 분석 기술

3.1 DIOTA

이 프로그램은 정확하게는 동적 바이너리 계측(instrumentation) 프로그램이다[1]. 동적 바이너리 계측은 런타임에 코드를 수정하게 되는데, 이는 코드뿐만 아니라 데이터와 이들 사이의 오프셋마저도 변경시킬 가능성이 있기 때문에 프로그램이 수정하는 지점과 이로 인해 영향이 미쳐 추가로 수정이 발생하게 될 부분에 대한 분석이 같이 병행되어야 한다.

DIOTA에서는 이와같은 문제 해결을 위해 메모리의 코드 블록을 복사하고 trampoline 코드를 삽입한 뒤, 이를 통해 런타임시의 실제 주소를 컨트롤할 수 있도록 한다. 또한 공유 라이브러리화된 동적 링커를 통해 수정된 코드 블록을 추적, 기록하여 관리한다.

3.2 GCAP

런타임에 바이너리 SMC의 모듈식 분석·추론·검증을 수행하는 Hoare-logic 스타일 프레임워크이다[2]. 고전적인 Hoare-logic을 확장하여 명령어 코드를 정기적으로 변경 가능한 데이터 구조로 취급한다.

GTM(General Target Machine)이라 명명한 폰노이만 구조 모델을 통해 데이터 타입과 상태, 명령어 등을 일반화 하여 정의하고 각 머신 아키텍쳐(x86, ARM 등)에 따라 추가적인 시멘틱과 opcode들을 매칭시켜 실제 코드를 다루게 된다. 이를 통해 x86 바이오스의 부트로더, Parametric 피보나치 코드, 자가복제 코드, 암·복호화 코드 등의 SMC를 분석한 결과를 논문에 게재하였다.

3.3 BitBlaze기반 동적 backward 분석

해당 기법은 일반적으로 멀웨어에서 리버스 엔지니어링 방해를 위해 사용되는 SMC의 동작 방식인 restore-execute-hide 방식(일종의 암·복호화)의 코드를 동적 backward 분석을 통해 검출 및 시멘틱 복구를 하는 것이다[3].

이를 위해 코드를 한 번 이상 동적으로 수행해야 하며, 명령어 실행 시 메모리의 값과 주소를 저장하고 추적한다. 이후 메모리 쓰기 명령어(e.g. ADC, ADD, AND 등)를 검색하여 해당 명령어의 목적지 주소와 값을 앞서 저장했던 값들과 비교하며 수정된 명령어, 즉 원래 코드가 수행하고자 하는 시멘틱을 찾는다.

해당 기법을 통해 Aspack, UPX, Themida, Pecrypt 등 약 10가지 패커로 패킹된 코드의 원본을 복구해 내는 것을 성공하였다.

3.4 SM-PDS(Self-Modifying Push-Down System)

PDS는 절차 호출을 정확하게 모델링 하고 프로그램 스택을 모방할 수 있기 때문에 순차 프로그램 분석에 광범위하게 사용되는 일반적인 모델이다. SM-PDS는 이러한 PDS를 확장하여 SMC를 처리할 수 있도록 수정하고 바이너리를 SM-PDS로 변환하여 Reachability 분석을 수행한다.

본 기법을 설명한 논문에서는 SM-PDS의 증명을 위해 이를 동등한 의미를 가지는 유한

상태의 PDS로 변환할 수 있음을 보였다. 그리고 SMC가 포함된 멀웨어 분석을 위해 SM-PDS 상의 유한 오토마타를 정의한 뒤 이 SM-PDS이 도달 가능한 무한 집합을 계산하는 알고리즘을 통해 기존 PDS로는 처리할 수 없었던 SMC가 포함된 멀웨어들을 분석하였고 해당 결과를 게재하였다.

IV. 결론

본 논문에서는 후속 연구를 위해 최신 커널과 멀웨어에 사용되는 SMC와 SMC 분석 기술의 동향에 대해 간략하게 설명하였다. 지면에서는 커널과 관련된 일부분의 기술만 소개하였지만 실제로 SMC는 멀웨어나 여러 소프트웨어에서 다양하게 사용되고 있다. 따라서 해당 기술을 올바르게 이해하고 분석, 검증할 수 있는 기술의 개발은 상당히 중요하다고 할 수 있을 것이다.

그러나 당 기술과 관련된 연구 및 기술 개발 사례는 현재 미진한 실정이다. 본 논문과 더불어 지면 분량상 싣지 못한 분석 자료는 SMC를 정적 분석하여 정확하게 역어셈블 할 수 있도록 바이너리를 수정하는 것을 목표로 하는 연구에 쓰일 예정이다.

[참고문헌]

[1] J. Maebe, K. De Bosschere, "Instrumenting self-modifying code," in proceedings of the Fifth International Workshop on AADEBUG 2003

[2] Hongxu Cai, Zhong Shao, Alexander Vaynberg, "Certified Self-Modifying Code," ACM PLDI 2007

[3] Shi Dawei, Lv Delong, and Ye Zhibin, "Dynamic Self-modifying Code Detection Based on Backward Analysis," in proceedings of the 10th ICCAE 2018

[4] Tayssir Touili, Xin Ye, "Reachability Analysis of Self Modifying Code," in proceedings of the 22nd ICECCS 2017

서버리스 플랫폼에서의 정확한 요금 산정을 위한 초 저지연 리소스 미터링 모듈

최상훈*, 이병용*, 박건호*, 박기웅†

*세종대학교 시스템 보안 연구실, † 세종대학교 정보보호학과

Low Latency Resource Metering Module for Fair Pricing in Serverless Platform

Sang-Hoon Choi, Byeong-Yong Yi, Keon-Ho Park, Ki-Woong Park†

*SysCore Lab., Sejong University.

† Department of Computer and Information Security, Sejong University

요 약

최근 클라우드 컴퓨팅 기술이 발달하면서 함수 단위의 서비스 제공 플랫폼인 서버리스 플랫폼이 등장하게 되었다. 서버리스 플랫폼에선 기본적으로 리소스 모니터링 주기를 기반으로 API 사용에 따른 과금이 이루어진다. 따라서 정확한 과금을 위해선, 리소스 사용량을 측정할 수 있는 리소스 모니터링 도구의 성능이 중요하다. 본 논문에선, 기존에 서비리스 환경에서 사용하는 리소스 미터링 도구 한계점을 해결 하기위해 Linux Kernel 모듈 기반의 초 저지연 리소스 미터링 방안을 제안한다. 본 논문에서 제안하는 모듈은 기존 컨테이너 모니터링 도구인 cAdvisor에 비교하여 연산 오버헤드가 적고, 빠른 주기로 리소스 메트릭 수집이 가능하다. 본 논문에서 제안한 기술은 서버리스 서비스 사용량에 따른 정확한 요금 산정을 위한 기반 기술로 활용이 가능하다.

I. 서론

클라우드 컴퓨팅 기술의 발달로 인해 클라우드의 장점을 활용한 다양한 서비스들이 등장하고 있다. 클라우드 컴퓨팅은 기존의 레거시 시스템과 비교하여 비용 측면의 절감 효과가 있다는 장점이 있어 그에 따른 수요가 증가하고 있다. 요금 관련 트렌드에 대해 가장 잘반영한 기술은 FaaS(Function as a Service)가 있다. FaaS는 사용자가 원하는 기능(로직)을 함수로 작성하면 컨테이너를 통해 해당 함수를 수행할 수 있도록 지원하는 서비스이다. 따라서 사용자는 함수 수행 요청에 대해, 실질적으로 컨테이너에서 함수 실행된 시간에 대해서만 요금만 지불하면 된다는 특징이 있다. 하지만, 기본적으로 함수의 실행 시간 측정은 리소스 모니터링 기술이 기반이 되므로, 서버리스 플랫폼에서 세밀한 요금 계산을 위해선 리소스 모니터링 도구의 성능이 중요하다. 본 논문에서는 기존 서버리스 플랫폼에서 사용하는 모니터링 도구의 성능 개선을 위해 Linux Kernel 모듈을 활용한 초 저지연 리소스 미터링 모듈을 제안한다. 제안하는 미터링 모듈은 기존 컨테이너 환경에서 사용하는 cAdvisor[1]에 비해 낮은 오버헤드를 발생시키며, 더 빠른 주기로 리소스 미터링이 가능하다. 본 논문의 구성은 다음과 같다. 2장에서는 본 논문의 기반이 되는 배경지식에 대해서 설명한다. 3장에서는 본 논문에서 제시하는 초 저지연 리소스 미터링 모듈을 제안한다. 4장에서는 기존의 컨테이너 시스템에서 사용하는 미터링 도구인 cAdvisor와 본 논문에서 제시하는 미터링 방안에 대한 성능 비교를 수행하고 5장에서는 결론 및 향후 연구에 대해서 기술한다.

II. 배경지식

본 장에서는 상용 서버리스 플랫폼에 대해 설명하고, 미터링에 사용되는 cAdvisor 등의 컨테이너 모니터링 도구의 특징에 대해서 설명한

다. 또한, 기존 서버리스 플랫폼의 과금 체계에 대해 기술하고, 과금체계의 공통적인 한계점에 대해서 기술한다.

2.1 서버리스 플랫폼 서비스

기존의 서버리스 플랫폼에는 Amazon 사의 AWS Lambda[2] Google 사의 Google Cloud Functions[3] 이 있다.

AWS Lambda: AWS Lambda는 Amazon 사에서 제공하는 상용 서버리스 플랫폼으로서 2014년에 서비스를 공개하였다. 현재 Python, Go, Java, C# 등의 언어를 지원하고 있으며, 요청한 API의 구동을 위해 할당 가능한 최대 시간은 900초이다. 플랫폼 내부의 성능 및 이벤트 모니터링, 로깅 기능을 수행하기 위해 CloudWatch[4]를 사용한다. API 구동에 할당한 시간에 따라 100ms 단위로 요금이 부과된다는 점이 있다. 즉, 구동 시간 관련하여 부과되는 최소 요금은 100ms부터 부과된다는 것을 알 수 있다.

Cloud Functions: Cloud Funtions는 Google 사에서 제공하는 상용 서버리스 플랫폼으로서, 2018년에 서비스를 공개하였다. 현재 지원 가능한 언어는 Python, Go, Node.js 등의 언어이며, 요청한 API의 구동을 위해 할당 가능한 시간 최대 540초이다. 플랫폼 내부의 로깅 및 모니터링 도구로서 Stackdriver[5]를 사용한다. Cloud Funtions 또한, API 구동에 할당한 시간에 따라서 100ms 단위로 요금을 부과한다.

2.2 서버리스 플랫폼의 과금 정책에서 리소스 미터링 도구의 역할

cAdvisor는 Google 사에서 오픈소스로 배포하고 있는 도커 컨테이너 리소스 사용량 도구이다. cAdvisor는 리눅스 커널에서 프로세스의 리소스 사용의 관리를 수행하는 cgroups 기반으로 개발되었으며, 컨테이너 워크로드 관리를 위한 플랫폼인 쿠버네티스 환경에서 리소스 모니터링 도구로서 사용되고 있다.

서버리스 플랫폼에서 기본적인 과금관련 정책은 리소스 사용량에 따른 과금이다. cAdvisor 등의 모니터링 도구를 통해 리소스 사용량 측정을 수행한 뒤, 리소스 사용량과 사용 시간에 비례하여 과금을 수행하므로 서버리스 플랫폼에서 과금 서비스의 기반 역할을 수행한다. 따라서, 최대 측정 가능한 주기 및 모니터링 도구 자체의 연산 오버헤드는 과금 서비스의 기본적인 요소가 된다.

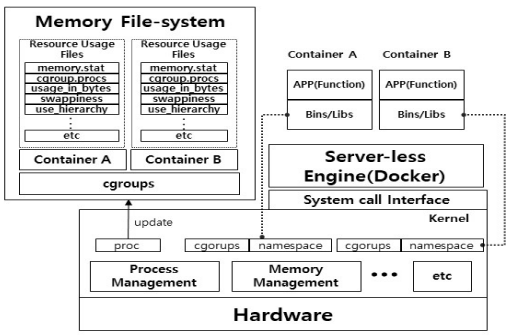
2.3 기존 서버리스 플랫폼의 과금 관련 모니터링 도구와의 연관성 및 한계점

AWS Lambda, Google Clound Funtions 등의 서버리스 플랫폼은 기본 과금 단위가 100ms 단위로 반올림하여 정산된다는 특징이 있다. 이처럼 100ms 단위의 요금 정책을 선정한 이유는 미시적 관점에서 100ms 보다 낮은 단위의 리소스 사용량 측정에 한계가 있기 때문이다. 서버리스 플랫폼에서 리소스 사용량 미터링을 위해 사용하는 cAdvisor등의 리소스 모니터링 도구의 최대 수집주기가 곧 서버리스 플랫폼 모니터링 도구의 성능이 이어짐을 의미한다. 따라서, 악의적인 사용자는 Public API를 Function Container를 통해 사용자에게 제공할 경우 발생할 수 있는 과요금 문제를 해결하기 위해선, 기존 모니터링 도구의 리소스 수집 주기에서 개선이 필요하다.

III. 초 저지연 리소스 미터링

본 장에서는 기존 리소스 사용량 수집 방법의 성능적인 문제점을 분석하고, 이를 해결 한 초 저지연 리소스 미터링 모듈을 설명한다.

3.1 cAdvisor 분석을 통한 문제점 도출



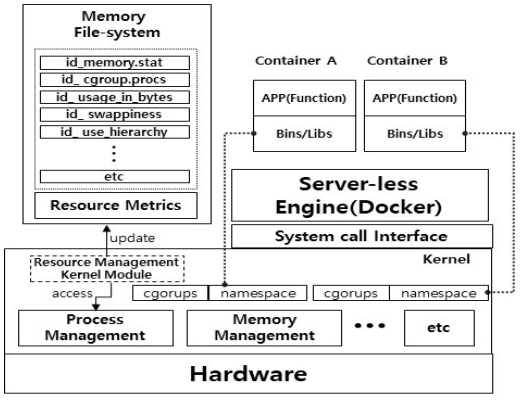
[그림 1] 기존 리소스 사용량 수집 방법

본 연구에서는 기존 컨테이너 플랫폼에서 리

소스 사용량 측정을 위해 사용되는 cAdvisor가 리소스 메트릭 정보를 빠른 주기로 수집하지 못하는 원인을 도출하기 위해서 소스코드 레벨에서 분석을 진행하였다.

분석을 통해 도출 된 원인을 다음과 같이 도출 할 수 있었다. 기존의 cAdvisor의 경우 proc 파일 시스템 내부에 생성되는 프로세스 리소스 정보를 파일로 접근하여 읽기를 통해 사용량 정보를 수집한다. 하지만, 리소스 사용량에 대한 정보는 컨테이너(프로세스) 마다 proc 파일 시스템 내부 별도의 영역에 존재하며, 리소스 종류에 따라 별도의 파일(memory.stat, memory.usage_in_bytes 등)로 관리 된다. 이와 같은 과정에서 프로세스 리소스 정보의 파일 위치를 추적하기 위한 연산과 다양한 파일에 대한 반복적인 읽기 연산으로 인해 Context Switch가 빈번하게 발생한다.

3.2 초 저지연 리소스 미터링 설계 및 구현



[그림 2] 리소스 미터링 커널 모듈 구조

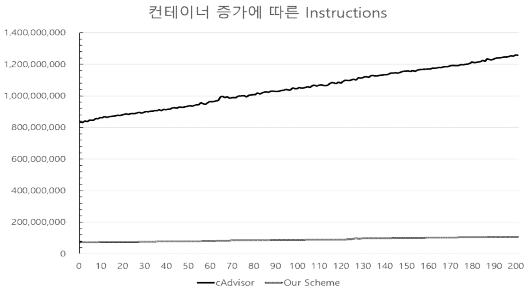
cAdvisor 분석을 통해 컨테이너 리소스 사용량 측정에서 발생하는 불필요한 연산 오버헤드의 문제점을 파악하였다. 이와 같은 문제를 해결하기 위해 [그림 2]과 같이 커널 모듈을 통해 컨테이너의 리소스 사용량 정보를 효율적으로 수집 할 수 있도록 리소스 미터링 커널 모듈을 설계 및 구현 하였다.

본 논문에서 제안한 리소스 미터링 커널 모듈은 도커 엔진을 통해 구동중인 컨테이너 pid 추적을 통해 프로세스 구조체에 직접 접근하여, 프로세스 구조체 정보(리소스 관련 정보)가 업데이트 될 때마다 리소스 사용량 정보를 ReousrceMetric 이라는 proc 파일시스템 영역에 리소스 메트릭 정보를 업데이트 한다. 따라서, 단 한번의 proc 파일 시스템 영역 읽기연산을 수행함으로써 시스템 내 구동중인 모든 컨테이너들의 리소스 사용량 메트릭 정보를 수집할 수 있다.

IV. 성능평가

4.1 연산 오버헤드 성능평가

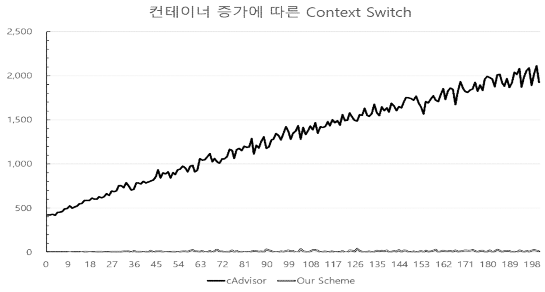
본 연구에서 제안한 초 저지연 리소스 미터링 커널 모듈의 오버헤드 성능 측정하기 위해, 컨테이너 리소스 자원 수집에 활용되는 Google 사의 cAdvisor와 성능을 비교하였다. 실험 방법은 Linux에서 제공하는 벤치마크 도구인 Perf를 사용하여 수행하였으며, Perf가 지원하는 성능관련 지표 중 Context Switch와 Instrcutions를 기준으로 성능평가를 수행하였다.



[그림 3] 컨테이너 증가에 따른 Instructions 변화

cAdvisor가 컨테이너 리소스 자원 수집할 때 컨테이너의 cgroups 파일 추적, 파일 읽기, 리소스 관련 정보가 포함된 구조체 접근 및 읽기가 발생한다. 제안한 초 저지연 리소스 미터링 커널 모듈의 경우 리소스 관련 정보가 포함된 구조체에 직접 접근하기 때문에 컨테이너의 cgroups 파일 추적 과정이 생략되고, 파일 읽기가 한번만 수행된다. Context Switch는 Task와 비례하여 발생한다. 따라서 컨테이너의 수만큼 파일 접근 및 읽기가 발생하는 cAdvisor에 비해, 컨테이너 수와 관계없이 한번의 파일 읽기가 발생하는 초 저지연 리소스 미터링 커널 모

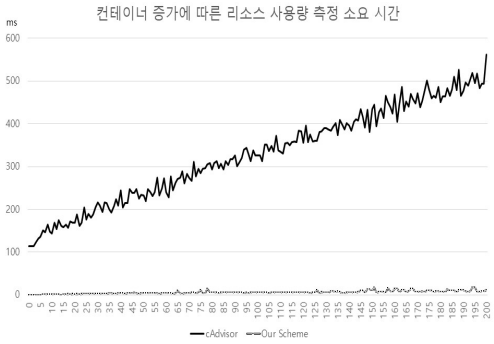
듈이 적은 Context Switch를 유발한다.



[그림 4] 컨테이너 증가에 따른 Context Switch 변화

Instructions에 대한 성능 평가는 [그림 4]와 같다. Instructions는 리소스 자원 수집을 위해 수행되는 과정에 비례하며, 파일 접근 및 읽기, 구조체 접근 및 읽기 등 여러 단계를 거쳐 정보를 수집하는 cAdvisor와 비교하여 본 논문에서 제안하는 모듈은 구조체에 직접 접근하기 때문에 최소의 명령어로 자원 사용량 수집이 가능하다.

4.2 리소스 수집 소요시간 성능평가



[그림 5] 리소스 수집 소요 시간 비교

리소스 수집 소요 시간은 서버리스 환경에서 정확한 리소스 미터링(요금산정)을 미세하게 측정하기 위한 가장 중요한 요소이다. 본 장에서는 제안한 초 저지연 리소스 미터링 모듈과 CAdvisor의 리소스 수집 소요 시간에 대한 성능을 비교한다. 결과는 [그림 5]와 같다. 동일한 환경에서 리소스 수집를 수행할 경우, 리소스 수집 소요 시간은 한 번의 리소스 수집 연산에 비례한다. 따라서 연산 오버헤드 성능 평가를 통해 적은 연산 오버헤드를 발생시키는 초 저지연 리소스 미터링 모듈이 적은 수집 소요 시간을 보이며, 이에 따라 더 빠른 주기로 수집가능하다는 것을 볼 수 있다.

V. 결론

본 논문에서 수행한 실험 결과에서는 제안한 초 저지연 리소스 미터링 모듈이 cAdvisor에 비해 낮은 연산 오버헤드와 빠른 리소스 수집 주기를 보였다. 연산 오버헤드 측면에서는 Context Switch와 Instructions을 기준으로 연산 오버헤드를 측정하였다. Context Switch의 경우, Task와 비례하여 발생하기 때문에, 파일 읽기가 컨테이너의 수만큼 발생하는 cAdviosr와 비교하여 한 번의 파일 읽기가 수행되는 초 지연 리소스 미터링 모듈이 더 적은 Context Switch를 유발하였다. 또한 Instructions은 cAdvisor는 리소스 사용량 측정을 위해 컨테이너의 cgroups 파일 추적과 파일 읽기, 리소스 사용량 측정을 위한 구조체 접근 등 여러 단계를 거치며 이에 대한 명령어가 수행되는 반면, 제안한 초지연 리소스 미터링 모듈은 cgroups 파일 추적 과정이 생략되고, 파일 읽기가 한번만 수행되며, 리소스 사용량 측정을 위한 구조체에 직접 접근하기 때문에 연산에 필요한 명령어 수가 최소화 된다. 또한 하나의 메모리 영역(파일)에 대해 접근하기 때문에 Context Switch가 최소화 된다. 본 논문에서 제안한 기술은 서버리스 서비스 사용량에 따른 정확한 요금 산정을 위한 기반 기술로 활용할 수 있다.

[참고문헌]

[1] Casalicchio, Emiliano, and Vanessa Perciballi. "Measuring docker performance: What a mess!!!." Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion. ACM, 2017.

[2] Villamizar, Mario, et al. "Infrastructure cost comparison of running web applications in the cloud using AWS

lambda and monolithic and microservice architectures." 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE, 2016.

[3] Google Cloud Function,

https://cloud.google.com/functions

[4] Cloud watch,

https://aws.amazon.com/cloudwatch/

[5] Statck Drvier,

https://cloud.google.com/stackdriver

Intel Processor Trace를 활용한 가상머신 수행 흐름 무결성에 관한 연구

서지원*, 방인영*, 백윤흥*

*서울대학교 전기정보공학부, 반도체공동연구소

A Study on Control-Flow Integrity for Virtual Machines Using Intel Processor Trace

Ji-Won Seo*, In-Young Bang*, Yun-Heung Paek*

*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC), Seoul National University

요 약

하이퍼바이저가 각각의 게스트 커널을 분리하고 있기 때문에 가상환경에서 게스트 커널의 무결성 여부는 전체 시스템의 보안에 영향을 미치지 못한다고 여겨진다. 그러나, 최근 연구는 악의적인 사용자가 사용자 공간(User Space) 에서 게스트 커널을 공격하여 궁극적으로 VM Escape Attack을 통해 전체 시스템의 보안을 위협할 수 있게 만드는 위험성을 제기하고 있다. Virtual Machine Introspection (VMI) 과 같은 기법을 이용하여 해결하고자 하였으나, 이는 일부 행위만 알 수 있을뿐더러 성능 오버헤드 또한 크다는 단점을 갖고 있다. 본 연구에서는 기존 연구들에 대한 소개를 한 후, 본 연구에서는, VM 에 대한 더 세부적인 정보를 얻을뿐더러 더 효율적으로 정보를 얻을 수 있는 Intel Processor Trace를 활용할 수 있는 방안에 대해 제시할 것이다.

I. 서론

하나의 물리 머신에서 여러 개의 게스트 VM을 구동하는 것을 가상 환경이라 한다. 이러한 가상화는 클라우드 컴퓨팅과 같이 고효율 및 저효율의 최신 컴퓨팅 인프라를 구축하고 관리할 수 있는 기술로 널리 사용되고 있다.

이러한 환경에서, 통념상, 하이퍼바이저가 각각의 게스트 커널을 분리하고 있기 때문에 가상환경에서 게스트 커널의 무결성 여부는 전체 시스템의 보안에 영향을 미치지 못한다고 여겨진다. 그러나, 최근 연구[1,2] 는 악의적인 사용자가 사용자 공간(User Space) 에서 게스트 커널을 공격하여 궁극적으로 VM Escape Attack을 통해 전체 시스템의 보안을 위협할 수 있게 만드는 위험성을 제기하고 있다. 커널은 모든 코드와 데이터를 단일 주소 공간에 배치하는 디자인이므로, 악의정인 사용자, 즉 공격자가 몇 가지 보안 허점을 이용하여 커널의 일부를 손상 시킨다면, 전체 커널을 금방 장악 당하기 쉽다는 특성을 가지고 있다. 또한, 악의적인 사용자가 커널을 제어할 수 있는 능력이 있다면 내부 상태를 조작하여 공격 징후를 숨기고자 한다. 이러한 상황에서 공격을 정확하게 감지하기 위해서는, 하이퍼바이저에서 게스트 커널의 전체 행위를 모니터링 해야 할 필요가 있다. 그러나, 하이퍼바이저에서 게스트 커널의 행위를 알아 내는 것은 의미상의 차이 (Semantic Gap) 문제로 인하여 어려움을 갖고 있다. 많은 연구들은 Virtual Machine Introspection (VMI) 과 같은 기법을 이용하여 해결하고자 하였으나, 이는 일부 행위만 알 수 있을뿐더러 성능 오버헤드 또한 크다는 단점을 갖고 있다. 이러한 VMI 연구들은 클라우드 환경에서 가상머신 각각의 상태 정보를 추출하여 보안 검증을 하게 된다. 이때 기본적으로 바이너리를 수정하여 가상머신이 동작하며 발생하는 다양한 이벤트들에 대한 정보를 추출하여 검증한다. 그러나 이러한 기법은 실시간으로 모니터링을 하기에는 성능 오버헤드가 크므로 개선이 필요하다.

우리는 이를 해결하기 위해, 하이퍼바이저에

서 게스트 커널의 수행 흐름 정보를 신속하게 얻기 위해 Intel PT를 사용하였다. Intel PT 는 인텔에서 제공하는 하드웨어로 적은 성능 오버헤드로 수행 흐름 정보를 추출할 수 있는 하드웨어 기능이다. 종래에 흐름 수행 정보 무결성을 위해 Intel PT를 사용한 연구를 클라우드 컴퓨팅 상황인 VM 에 그대로 적용하는 데에는 어려움이 있다. 본래 PT 는 디버깅 용도 이므로 런타임에 그대로 적용하는 것이 어렵다. 또한, VM 커널의 context 정보 (e.g. interrupt handle, pid, tid, vmid)를 상세하게 반영하고 있지 않다. 따라서, 각각의 문제를 해결하기 위해 PT 패킷 디코더와 커널 수정이 필요하다. 본 연구에서는, 해당 프레임워크의 효용성을 증명하기 위해 시스템 콜이 호출 되거나, 인터럽트가 불리었을 때 각각의 로그를 VM별, Process 별로 추출할 수 있도록 수정하였다. VM 에 대한 더 세부적인 정보는 향후 연구를 통해 진행할 예정이고, 기존 연구에서 제시한 방어법들을 소개 한 뒤 결론을 제시하도록 하겠다.

II. 배경 이론 및 기존 연구에 대한 소개

2.1 Intel PT

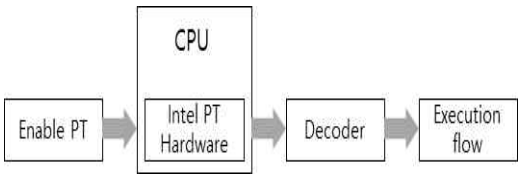


그림 1 Intel PT 동작 과정

최근 Intel에서 소개한 하드웨어인 Intel Processor Trace 는 적은 오버헤드로 프로그램의 수행 흐름을 추출할 수 있도록 제공된 새로운 하드웨어 기능이다. 이를 이용하면 수행 흐름들을 완벽하게 기록할 수 있을뿐더러, 수행 흐름 정보를 여러 형태의 패킷 정보로 메모리 버퍼에 저장하게 된다. 분석하는데 필요한 주요 패킷으로는 TNT, TIP, PSB, FUP, PIP 패킷이 있다. TNT 패킷은 직접 분기문이 Taken 인지 not-Taken 인지에 대해 기록하는 패킷이다. 해당 패킷은 각각 프로세스의 제어 흐름의 모든 변경을 나타내는 트레이스로 대부분의 패킷을 차지한다. TIP 패킷 역시 트레이스에 대부분에서 나타나는데, 이 패킷은 간접 분기문, 예외처리, 인터럽트가 발생할 경우의 목적지 주소를 기록하는 패킷이다. PSB 패킷은 메모리 영역에 일정량의 패킷들이 쌓이면 발생하는 패킷으로 분석하는 시점의 기준을 정리하는데 쓰이는 패킷이다. FUP 패킷은 인터럽트와 예외 처리가 발생할 경우 출발지 주소를 기록하는 패킷으로 CR3 레지스터가 업데이트 되고 CPU에 의해 인터럽트와 같은 비동기 이벤트가 발생할 경우 생성된다. PIP 패킷은 현재 사용하고 있는 페이징 정보를 담고 있는 CR3 레지스터가 변화가 있을 때마다 CPU 에 의해 발생하는 패킷이다. 본 논문의 경우 가상 머신 커널의 수행과 호스트 커널의 수행을 구분하기 위해 해당 패킷 내 NR(non-root) bit을 통해 둘을 구분하였다.

2.2 Intel PT를 활용한 흐름 정보 무결성

2.1에서 언급한 Intel PT를 활용하여 프로그램의 수행 흐름 정보를 효율적으로 추출 하여, 수행 흐름 무결성을 위한 연구들이 있다. [3,4]

위 연구들은 사용자 응용 프로그램에 대한 수행 흐름 무결성을 검증한 연구들로, 본 연구에서 주장하는 클라우드 환경의 가상 머신들에 대한 수행 흐름 무결성을 검증하기 위해 기존 기법들을 그대로 사용하기 어렵다. PT-CFI[3]는 후방 간선에 대한 수행 흐름 검증을 목적으로 한 연구로, 적은 오버헤드로 추출하였다. 본 연구에서는 기존 오프라인 디버깅 목적의 PT를 온라인에서 사용하도록 하였고 2.1에서 말한 TIP 패킷을 이용하여 수행 흐름 무결성 검증을 하였다. GRIFFIN[4] 은 PT-CFI를 보완하여, 후방 간선 뿐 아니라 전방 간선에 대해서도 수행 흐름 무결성 정책을 지원한다. 해당 연구에서는 Intel PT 가 기록해 주지 않는 call site 정보를 알아내기 위해 추가적으로 바이너리 분

석 기술을 추가하였고 수행 흐름 정책을 코어스 그레인, 파인 그레인, 스테이플로 나눠 좀 더 심층적인 분석이 가능토록 하였다.

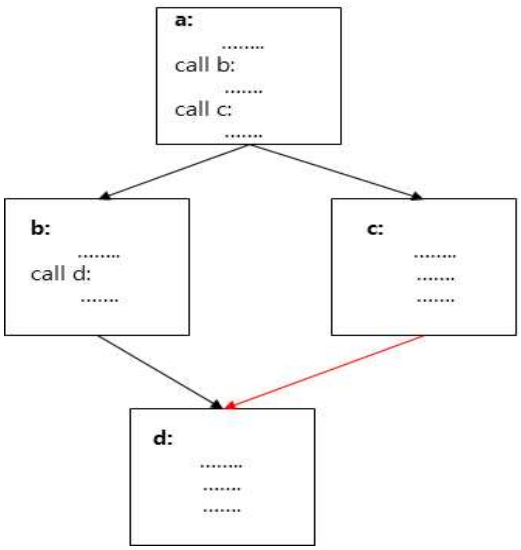


그림 2 수행 흐름 무결성 검증 과정 예시

2.3 VMI (Virtual Machine Introspection)

최근 가상화가 증가함에 따라 VMI (Virtual Machine Introspection) 는 활발한 연구 분야로 게스트 운영체제의 런타임 상태를 모니터링 할 수 있는 방법으로 부상하고 있다. VMI[5,6,7]에 대한 대부분 연구는 관리자에게 시스템 콜과 같은 게스트 가상머신 커널의 이벤트를 넘겨 관리자가 있는 하이퍼바이저 계층으로 전달하고 커널 동작을 검사하는 기능을 관리자에게 제공한다. [5] 에서는 하이퍼바이저 계층에서 V 내부 시스템 호출 이벤트를 해석한다. 즉 허니팟 내에서 실행되는 모든 프로세스에 의해 호출된 시스템 콜을 수집하는 VMI 기법을 제안한다. 외부에서 허니팟 VM 내부 이벤트를 확인하기 위해 매개 변수를 검사하여 각 시스템 콜을 검사한다. 이 솔루션은 가상화 플랫폼을 위한 세분화된 모니터링 기법을 제공하지만, 이 모니터링 기법은 개별 대상 가상머신에서만 실행될 수 있으므로 대규모 가상화 환경에는 적용하기가 어렵다. 또한 이 솔루션은 게스트 가상머신이 시작되어 실행중인 동시에 시스템 콜을 호출하는 공격을 감지 할 수 없다는 단점이 있어 온라인에 이상 징후 판단을 하기 어렵다.

III. 가상 머신에 대한 수행 흐름 무결성 검증을 위한 Intel PT 수정

여러 게스트 가상머신이 동시에 실행되는 가상화된 시스템에서는 모든 게스트 가상머신의 추적 패킷들은 CPU 에 의해 생성된 순서대로 혼합된다. 따라서, 여러 가상머신이 동시에 수행되고 있는 상황에서 각각의 가상머신의 행위를 기록하고 있는 패킷들이 섞이지 않고 분리하는 기술이 필요하다. 그림 3에서 게스트 가상머신이 VCPU1, VCPU2, 두 개의 코어가 도는 것 처럼 각 게스트 가상머신은 VCPU 의 형태로 CPU 리소스가 할당받게 된다. 본 연구는 이 점을 착안하여 모든 가상머신 명령어는 VCPU 에 의해 실행되므로 가상머신을 실행하는 특정 VCPU 에 의해 일련의 PT 패킷이 생성된다. 본 연구에서는 게스트 가상머신이 생성되고 VCPU 가 할당될 때, VCPU 당 추적 버퍼를 준비하여 각 VCPU 에 해당하는 패킷 리스트를 구성한다. 즉 VM-enter 가 발생할 때마다 Intel PT를 동작시켜 VCPU에서 발생하는 패킷들을 해당 트레이스 버퍼에 기록하여, 동시에 동작하는 모든 가상머신들에 대하여 각각 패킷들을 처리할 수 있도록 Intel PT를 수정하였다.

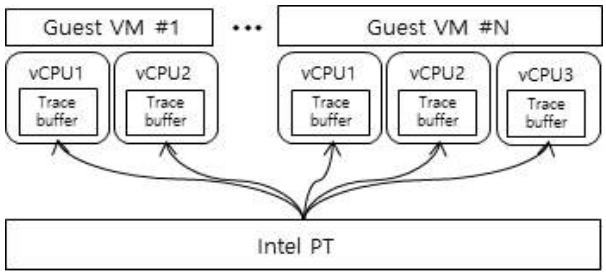


그림 3 VCPU 당 Trace Buffer를 통한 로그 수집

IV. 결론

오늘날 클라우드 컴퓨팅 환경을 통해 많은 사용자들이 클라우드 환경에서 많은 서비스를 이용하고 있고 이에 따라 클라우드 컴퓨팅 환경에서의 보안 관점의 문제에 대한 제기가 떠

오르고 있다. 본 연구는 Intel에서 제안한 Processor Trace 라는 하드웨어를 활용하여 각 가상머신에서 실행하는 사용자 프로그램의 수행 흐름 무결성을 적용시킬 수 있도록 기존 논문들에 살펴보았다.

여러 개의 가상머신이 동시에 실행되는 시스템에서 실시간으로 가상머신에 대한 수행 흐름 무결성을 검증하기는 어렵다. 본 연구는 이에 대한 선행 연구로써 실시간으로 VM 별 데이터를 추출할 수 있는 방안에 대해 기존 연구를 조사하였고 이에 대해 발전할 수 있는 방향성을 제시하였다. 향후 연구에서는 이와 같이 실시간으로 추출할 수 있는 데이터 기법을 이용하여 머신러닝에 적용하여 자동화된 수행 흐름 무결성 검증을 할 수 있도록 발전해 나가고자 한다. 이를 위해서 유실되는 데이터 없이 실시간으로 모니터링 할 수 있는 방법도 고려할 계획이다.

ACKNOWLEDGMENT

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구와 (No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발) 2019년도 두뇌한국21플러스사업에 의하여 지원되었음.

[참고문헌]

[1] Shengmei Luo, Zhaoji Lin, Xiaohua Chen, Zhuolin Yang, and Jianyong Chen. Virtualization security for cloud computing service. In Cloud and Service Computing (CSC), 2011 International Conference on, pages 174 - 179. IEEE, 2011.

[2] Jyotiprakash Sahoo, Subasish Mohapatra, and Radha Lath. Virtualization: A survey on concepts, taxonomy and associated security issues. In Computer and Network Technology (ICCNT), 2010 Second International Conference on, pages 222 - 226. IEEE, 2010.

[3] Gu, Yufei, et al. "PT-CFI: Transparent backward-edge control flow violation detection using intel processor trace." Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy. ACM, 2017

[4] Ge, Xinyang, Weidong Cui, and Trent Jaeger. "Griffin: Guarding control flows using intel processor trace." ACM SIGOPS Operating Systems Review 51.2 (2017): 585-598.

[5] Yonghwi Kwon, Brendan Saltaformaggio, I Luk Kim, Kyu Hyung Lee, Xiangyu Zhang, and Dongyan Xu. A2c:Self destructing exploit executions via input perturbation.In Proceedings of NDSS, volume 17, 2017.

[6] Zhiqiang Lin and Yangchun Fu. Space traveling across vm: Automatically bridging the semantic gap in virtual machine introspection via online kernel data redirection. In 2012 IEEE Symposium on Security and Privacy, pages 586 - 600. IEEE, 2012.

[7] Rui Wu, Ping Chen, Peng Liu, and Bing Mao. System call redirection: A practical approach to meeting realworld virtual machine introspection needs. In Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on, pages 574 - 585. IEEE, 2014..

Binary Diffing: An Empirical Analysis and Enhancements

Sami Ullah, and Heekuck Oh

Department of Computer Science and Engineering, Hanyang University.

Abstract

Vulnerabilities are continuously discovered and common in all software platforms. In the proprietary softwares, vulnerabilities patch is only released in a binary update. To analyse the cause of patch in binaries, binary diffing technique has been developed over the period of time. Such an industry standard and state-of-the-art binary diffing tool is *Diaphora,* often used to compare two binaries and extract the changes, but its not reliable for small changes in functions. In this research paper, we have empirically analysed the efficiency of *Diaphora* in terms of its classification accuracy, improved the diffing algorithm and proposed some new heuristics. The proposed combination of diffing algorithm and new heuristics have significantly reduced the false positives and the false negatives. The proposed improvements enable *Diaphora* to detect even small changes with high accuracy. For the test binaries, we have achieved 98% classification accuracy.

I. Introduction

Knowing the cause of vulnerability is influential to eliminate similar vulnerable code from other software products [3][4]. In contrary, proprietary softwares practice an opposite policy and never disclose the cause of vulnerability. To address this issue, reverse engineering techniques have been developed, which can effectively detect the changed functions after patch and highlight changed part in assembly code of a changed function.

Reverse engineering techniques consider the two major properties of assembly code. 1) Assembly as sequence of words 2) Structural features like control flow graph (CFG) [2]. TD-IDF (Term Frequency Inverse Document Frequency) based algorithms exploits the text property of assembly code and calculate the Jaccard similarity to compare assembly code of two functions. BinDiff [5] exploits the structural properties and focuses on graph centric analysis, neglecting as much as assembly as possible. To cope with graph isomorphic nature, a variety of heuristic are used to select the functions before the graph (CFG) comparison [1].

Assembly code of a function from different versions can be different due to compiler optimization e.g. *xor rax, rax"* and *"xor rbx, rbx"* instructions are semantically same but different for text based diffing algorithm. Similarly due to compiler optimizations, exactly similar functions can have different CFG e.g. *"if", "else"* basic block inversion. Independently, both of these techniques have limitations and can't achieve high classification accuracy. The current industry standard binary diffing tools use a hybrid approach and try to compensate the limitations of individual techniques. One such popular tool is *Diaphora*, which is an IDA Pro plugin and implemented in python.

* This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science and ICT (NRF-2019R1A2C2003045).

Author: sami.ullah2316@gmail.com

Diaphora algorithm uses hybrid approach and achieves a better accuracy compared to the individual techniques. In this paper, we have empirically analyzed *Diaphora* and found some special cases where *Diaphora* misclassifies and proposed improvements in diffing algorithm and heuristics. The proposed improvements achieves ~98% classification accuracy.

The rest of the paper is structured as follows. section 2) Diaphora analysis section 3) Proposed Improvements section 4) Evaluation results section 5) Conclusion and future work.

II. Diaphora Analysis

The released patched binary is mostly similar to previous version and only few functions are modified. The diffing objective is to classify the functions and focus on finding changed functions. As it is matching problem, one vs all, to reduce the function search space, *Diaphora* diffing algorithm first finds best matches and later finds partial matches. *Diaphora* has 53 different heuristics to select probably similar functions in two different binaries and then applies sequence matching or structural matching on assembly code to classify them as similar or changed functions based on similarity ratio. If the similarity ratio is 1.0, its classified as best match otherwise it's a partial match. The detailed diffing algorithm is shown in figure 1.

The classification accuracy of diffing algorithm depends on the choice of heuristics and their usages priority. The heuristic categories are discussed below in their priority order.

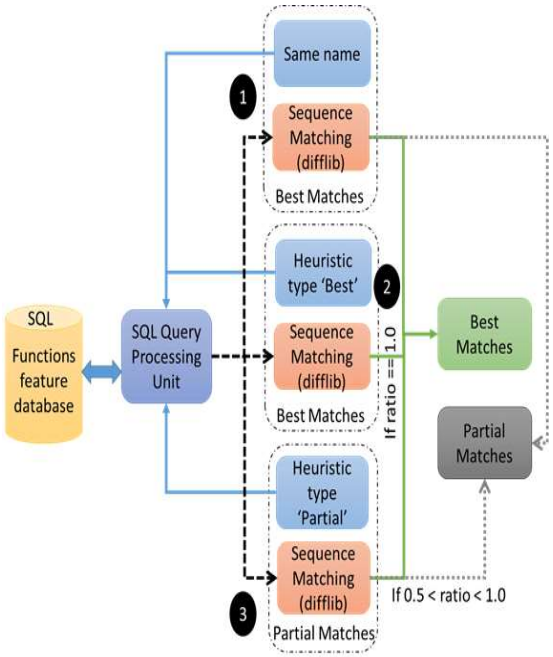


Figure 1 Diaphora algorithm

2.1. Same name heuristic

The top priority heuristic is "*same name*" and its selects the functions from two binaries, which have similar name.

2.2. Best heuristics

The heuristics which give no false positive are categorized as best heuristics. *Diaphora* has 7 such heuristics like same function hash, bytes hash, bytes sum, equal assembly etc. Best heuristics are used to swiftly classify best matches. Only those functions are classified as best match, for which similarity ratio is 1.0.

2.3. Partial heuristics

The heuristics which have threshold value as matching ratio are categorized as partial heuristics. If the matching ratio is exactly 1.0 then its classified as best match. If the ratio is above threshold value and less than 1.0 then function is classified as partial match.

Partially matched functions are the changed functions, which have been modified after binary patch.

1) Limitations of Diaphora

We have conducted an extensive empirical analysis of *Diaphora* to check the accuracy of each individual heuristic. We have tested *Diaphora* on XNU kernel (high sierra) and found out that "*same name*" and best heuristics (7) classifies almost 90% of functions and rest of functions are processed by partial heuristics (45). But to our surprise, 5 out of 7 best heuristics misclassify the functions and this pre-classification results in enormous false negatives. We have found special cases of false negatives, which are

- Variable initialization
- different function calls or conditions
- Additional arguments in function call or additional condition in conditional blocks.
- Number of arguments are same but value to argument is changed
- Single variable addition or deletion
- Big functions ignores small changes.

3. Proposed Improvements

The *Diaphora* limitations are due to unreliable best heuristics and unfortunately almost 90% classification is done by these unreliable best heuristics; resulting in many false negatives. Our proposed improvements tries to compensate these limitation and given below with detailed explanation.

3.1. *optimization of heuristics*

Only 2 best heuristics give no false positive. Function hash, bytes hash, bytes sum is calculated manually and *Diaphora* implementation is not accurate in some cases. So we have removed 5 such heuristics from best heuristics category.

3.2. *Assembly stripping*

An assembly instruction from two similar functions can have different offset value after compilation. Such instructions are semantically same but different for text based sequence matching. For example, "*jmp 0xFFFF*" and "*jmp 0xFFFA*" are semantically same, they are jumping to same basic block but address is different in two version; which makes them different for sequence matching. We have applied assembly stripping to make instruction similar in these cases.

3.3. *New heuristics*

Although, *Diaphora* has a big set (53) of heuristics and usually sufficient. We have found a new heuristic "*number of assembly instructions*", which is very accurate to detect minor changes like special cases.

3.4. *Optimization of algorithm*

Diaphora has preferred "*same name*" heuristic and almost 80-85% functions are classified by this heuristic. Using sequence matching under "*same name*" heuristic is not optimal and produces many false negatives. The reason is assembly is not simple text, instructions have structural and semantic meaning. To address this, we have given 2 best heuristics top priority to classify best matches and second priority to "*same name*" heuristic. The 3rd priority is similar to original *Diaphora* but sequence ratio precision is modified to 3 (from 2). The optimization objective is to priorities those heuristics which do accurate classification. So the best 2 heuristics classifies the most of best matches and reduces the function search space. Consequently, the unreliable heuristics are applied on less functions and ultimately we have achieved better performance.

XNU version	Changed	Original Diaphora			Modified Diaphora		
		Detected	False positive	False Negative	Detected	False positive	False negative
xnu-4570.31.3 ... xnu-4570.41.2	58	37	14	35	59	1	0
xnu-4570.51.1 ... xnu-4570.61.1	33	45	16	4	33	0	0
xnu-4570.61.1 ... xnu-4570.71.2	132	155	37	14	135	3	0

Figure 2 Evaluation results for original *Diaphora* and modified *Diaphora*

4. Evaluation Result

We have evaluated *Diaphora* using 5 version of XNU kernel binaries (high sierra) on a mac machine. The performance is measured in terms of false positives and false negatives. False negatives are the cases, when a function is changed after patch but it is not detected by diffing algorithm. False positives are functions which are not changed but diffing algorithm classifies them changed. False negatives are very critical to the classification accuracy as compared to false positives.

As shown in figure 2, in the original *Diaphora* there are large number of false positives and false negatives. After applying the improvements, we have achieved remarkable classification accuracy. The false negatives are critical and in the modified *Diaphora* there is no false negative in each version. False positives are also significantly reduced but there are few false positives cases. Overall modified *Diaphora* performs much better than original and gives better classification accuracy.

5. Conclusion and Future Work

When source-code is not available binary diffing is a viable solution, which can aid in extracting the changes in patched version. Semantic and structural features of assembly code are very important for efficient matching in binary diffing algorithm. To reduce the function search space, *Diaphora* is relying on heuristics. So the choice and priority of heuristics is very important for better classification.

In future, we further aim to reduce the false positives and extend our test set to all XNU kernel versions.

References

[1] Flake, H., 2004, July. Structural comparison of executable objects. In DIMVA (Vol. 46, pp. 161-173).

[2] Karamitas, Chariton, and Athanasios Kehagias. "Efficient features for function matching between binary executables." 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2018.

[3] Pewny, Jannik, et al. "Cross-architecture bug search in binary executables." 2015 IEEE Symposium on Security and Privacy. IEEE, 2015.

[4] Oh, Jeongwook. "Fight against 1-day exploits: Diffing binaries vs anti-diffing binaries." Black Hat. Black Hat (2009).

[5] Halvar Flake. Graph-based binary analysis. Black Hat Europe, 2003. www.blackhat.com/presentations/bh-europe-03/bh-europe-03-halvarflake.pdf

가상 환경에서의 효과적인 트레이스 추출 방안에 대한 연구

방인영*, 서지원*, 백윤흥*

*서울대학교 전기정보공학부, 반도체공동연구소

A Study on Effective Trace Extraction in Virtual Environment

In-Young Bang*, Ji-Won Seo*, and Yun-Heung Paek*

*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC), Seoul National University.

요 약

클라우드 컴퓨팅 환경에서 커널을 보호하기 위해 수행 흐름 무결성을 검증하는 것은 보안 적으로 중요한 일이다. 그러나 이를 위해서는 효과적인 수행 흐름 트레이스를 추출하는 것이 필요한데, 커널의 동적이고 선제적(preemptive)인 특성으로 인해 커널에 대한 수행 흐름 모니터링의 난이도는 크게 높아진다. 또한 게스트 커널을 대상으로 할 경우 가상 게스트 머신의 개수만큼 커널이 존재하므로 그 어려움이 배가 된다. 본 논문에서는 시스템적인 관점에서 Intel PT를 통한 게스트 커널의 프로세서 트레이스를 추출의 어려움에 대해 분석하고, 이를 공학적으로 해결하는 방안을 제시하고자 한다

I. 서론

클라우드 컴퓨팅 환경을 통해 낮은 비용으로 충분한 성능의 컴퓨팅 제반을 구축할 수 있으며, 가상화를 통해 하나의 물리 머신에서 다수의 게스트 가상 머신의 구동을 가능케함으로써 가상환경은 현실화 되었다. 오늘날 많은 사람들이 음성인식 기술과 같은 인공지능 서비스를 비롯하여 스마트 홈과 같은 사물 인터넷 서비스 등을 이용하며 가상 환경의 혜택을 누리고 있지만, 동시에 클라우드 컴퓨팅 환경의 보안성에 대한 우려도 커지고 있는 실정이다.

하나의 물리 머신을 다수의 사용자가 공유하는 클라우드 환경의 특성 상, 사용자는 자신들의 데이터에 대한 보안 우려를 떨칠 수 없으며, 따라서 클라우드 관리자는 각각의 사용자에게 제공하는 실행 환경이 철저히 격리되도록 함으로써 높은 보안성을 보장할 책임이 있다.

안전한 실행 환경 격리를 보장하기 위해서는 기저에 있는 게스트 커널의 무결성과 깊은 연관이 있으며, 클라우드 공급자는 개별 클라우드 사용자가 자신의 어플리케이션을 수행하는 가상머신의 커널 무결성을 보장해주어야 한다.

커널에 대한 전통적인 공격으로는 공격자가 실행하고자 하는 바이너리를 버퍼 오브플로우를 통해 코드영역에 쓰고, 해당 페이로드가 수행되게 하여 목적을 달성하는 공격이 있다. 이 공격은 W xor X라는 원칙을 통해 메모리의 주소에 대해 읽기와 수행 권한이 동시에 주어지지 않도록 함으로써 막을 수 있다. Aarch64의 경우 HCR 레지스터의 WXN 비트를 이용하면 쓰기 권한이 부여된 메모리 주소에 대한 수행 자체가 불가능하며, 이를 통해 쉽게 방어가 가능하다.

이후 수행 가능한 메모리 영역에 대한 쓰기가 불가능해 지면서, 수행흐름을 변조하여 코드 영역의 공격자가 원하는 특정 명령어들을 원하는 순서대로 수행시킬 수 있게 하는 다양한 코드 재사용 공격 문제가 제시되었고[90년대 논

문], 이를 막기 위해 수행 흐름 변조를 차단하거나 감시하기 위한 수행 흐름 무결성에 대한 연구가 활발히 진행되었다. 최초로 코드 재사용 공격이 제시된 이후 많은 시간이 지났지만 여전히 코드 재사용 공격은 유효함이 입증되었으며, 수행 흐름 무결성에 대한 감시는 여전히 중요한 커널 보호 기법으로 여겨진다.

대상 시스템의 수행 흐름을 효과적으로 모니터하는 방법으로 Intel Processor Trace(이하 Intel PT)라는 하드웨어를 이용하는 방법이 제시되었는데[2], Intel PT를 이용하면 5% 미만의 성능 오버헤드로 코드 수정 없이 CPU의 수행 흐름을 추출하여 커널의 무결성을 검증하거나 감시하는데 사용될 수 있다.

하지만 커널의 동적이고 선제적(preemptive)인 특성으로 인해 커널에 대한 수행 흐름 모니터링의 난이도는 크게 높아진다. 또한 게스트 커널을 대상으로 할 경우 가상 게스트 머신의 개수만큼 커널이 존재하므로 그 어려움이 배가 된다. 본 논문에서는 시스템적인 관점에서 Intel PT를 통한 게스트 커널의 프로세서 트레이스를 추출의 어려움에 대해 분석하고, 이를 공학적으로 해결하는 방안을 제시하고자 한다.

II. 배경 이론

2.1 Intel PT

Intel PT는 브로드웰(Broadwell) 이후 x86/x64 아키텍처에서 프로세서의 정보를 실시간으로 추출하기 위해 지원되는 하드웨어 기능이다. 실행 트레이스는 데이터 패킷을 순차적으로 연결하여 형성되며, 표 1은 Intel PT의 수 많은 트레이스 패킷 중 본 연구에서 게스트 커널의 수행 흐름 분석에 필요한 대표적인 패킷에 대해 정리한 것이다[3].

Packet Name	Packet Description
Paging Information Packet (PIP)	CR3 레지스터의 변화가 있을 때 발생하는 패킷이다. CR3 레지스터는 현재 태스크가 사용하는 pgd(page global directory)의 주소를 가리키며 시스템 설정에 따라 pcid (process-context identifier)를 포함 수 있다.
Target IP Packet (TIP)	모든 indirect branch가 발생할 때 목적 주소(target address)를 담고 있는 패킷이다. Exception이나 interrupt가 발생할 경우 vector table의 base로 점프한 뒤 해당 핸들러로 간접 분기를 하기 때문에 이 패킷을 통해 예외와 인터럽트의 주소 또한 추적하는 것이 가능하다.
Flow Update Packets (FUP)	인터럽트와 예외와 같은 비동기적 이벤트가 발생했을 때 원점 주소(source address)를 나타내는 패킷이다.
Packet Stream Boundary (PSB)	패킷을 저장하는 버퍼에 일정하게 나타나는 패킷으로 여러 CPU를 트레이싱하고 있을 경우 이 패킷을 이용하여 동기화하는 것이 가능하다.
TNT Packet	Direct conditional branch의 taken/not-taken 여부를 나타낸다.

표 1. Intel PT의 트레이스 패킷과 그 용도

2.2 공격 모델

공격자는 victim 가상 머신에 네트워크로 접속하여 가상 머신의 커널을 공격한다. 공격자는 커널의 취약점을 이용하여 수행 흐름 탈취(control flow hijack)공격을 행할 수 있지만, DOP(Data Oriented Programming) 공격은 제외한다.

또한 부트 시퀀스에는 공격자가 개입할 수 없

으며, Intel PT의 초기화에도 따라서 영향을 미칠 수 없다.

III.본론

3.1 커널의 선제적 특성으로 인한 어려움

커널의 수행 흐름 분석이 어려운 이유 중 하나는 커널 프로세스 간에 문맥 전환이 빈번하게 일어나고, 인터럽트나 예외와 같은 비동기적 이벤트로 인해 프로세스 별 트레이싱이 어렵다는 데 있다. 이를 해결하기 위해 그림 1.의 휴리스틱 알고리즘을 제안한다. record라는 구조체에 프로세스의 정보를 담고, 패킷에 따라 이를 처리하고 record 구조체를 담는 buffer를 두어 필요시 구조체를 저장하고 복원한다.

해당 알고리즘을 이용하면 패킷을 디코드할 때마다 프로세스의 시작과 종료 단위로 연속된 트레이스를 추출할 수 있다. 많은 부분이 생략되었지만, 프로세스가 종료됐는지 문맥전환을 위해 커널의 큐에 진입했는 지를 페이지 테이블의 valid 비트를 통해 커널 자료구조의 접근 없이 처리할 수 있다.

3.2 가상 머신에 따른 패킷 구분

실제 가상 환경에서는 다수의 가상머신이 동시에 구동되고 있기때문에 가상머신을 구분하여 패킷을 분류하는 과정이 반드시 필요하다. 출력되는 패킷 자체로는 가상머신을 식별할 수 있는 단서가 없기 때문에, Intel PT의 버퍼를 CPU 가상화 단위인 VCPU 마다 할당하고 기록함으로써 이를 해결한다.

x86/x64 virtualization extension에서는 VM-enter마다 호스트 머신이 VCPU로 실행되는 게스트 머신의 제어를 전달하기 때문에 VM-enter 코드를 수정하여 게스트 머신의 트레이스 패킷의 기록을 해당하는 VCPU의 Intel PT 버퍼에 저장하게 함으로써 구현이 가능하다. 또한 낮은 확률로 VCPU 간 프로세스 migration이 발생하더라도 VM-exit에서 이를 처리하는 것이 가능하다. 이와 같은 구현을 통해 동시적으로 구동하는 모든 가상머신을 구분하여 패킷을 처리하는 것이 가능하다.

Algorithm 2 Context sensitive packet decoding

```
1: current_record
2: buffer
3: Struct record
4:      trace
5:      pgd_base
6:      pcid
7:      state
8:: function search_record(PIP)
9::             search record from buffer
10:             if found then
11:                     get and remove record from buffer
12:                     current_record = found
13:                     current_record.state = none
14:             else
15:                     current_record = {0, PIP.pgd_base,
PIP.pcid, none}
16:             endif
17: function context_sensitive_decoder( packet )
18:     if packet is PIP & state is async then
19:             search_record(PIP)
20:     if packet is PIP & state is none then
21:             if current_record.pgd_base is valid then
22:                     push to buffer
23:             else
24:                     flush current_record
25:             current_record = {0, PIP.pgd_base, PIP.pcid,
none}
26:     else if packet is FUP & state is none then
27:             current_record. state = async
28:     else if current_record.state is none then
29:             current_record.trace += packet
30:     endif
```

그림 1. 비동기적 이벤트와 문맥 전환을 고려한 프로세스 별 패킷 출력 알고리즘의 수도 코드

IV.결론

지금까지 Intel PT를 이용하여 게스트 커널의 프로세서 트레이스를 추출하는 것이 왜 어려운지를 분석하고 이에 대한 해결 방안에 대해 간략히 논의해보았다. Intel PT에서 추출한 트레이스를 기반으로 수행 흐름 무결성을 검증하거나 감시하는 연구는 지금까지 진행된 바 있으므로 본 연구를 통해 게스트 커널의 프로세서 트레이스를 효과적으로 추출하고 이를 바탕으로 가상 환경에서의 게스트 커널에 대한 수행 흐름 무결성을 검증할 수 있다.

V.ACKNOWLEDGEMENT

이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발). 또한, 이 논문은 2019년도 두뇌한국21플러스 사업에 의하여 지원되었음.

[참고문헌]

[1] Hovav Shacham. 2007. The Geometry of Innocent Flesh on the Bone: Return-into-libc Without Function Calls (on the x86)., CCS.2007.

[2] Victor van der Veen, Dennis Andriesse, Manolis Stamatogiannakis, Xi Chen, Herbert Bos, and Cristiano Giuffrdia, The Dynamics of Innocent Flesh on the Bone: Code Reuse Ten Years Later, CCS, November, 2017

[3] Guide, Part. “Intel® 64 and IA-32 Architectures Software Developer’s Manual.” Volume 3B: System programming Guide, Part2 (2011).

하드웨어 암호 IP 테스트를 위한 SoC보드상에서의 AMBA AXI환경 구축

이진재*, 지장현*, 박종욱*, 김호원*

*부산대학교 전기컴퓨터공학부

Building an AMBA AXI Environment on a SoC Board for Hardware Cryptographic IP Testing

Jin-Jae Lee*, Jang-Hyun Ji*, Jong-Uk Park*, Ho-Won Kim*

*Division of computer science & engineering, Pusan National University.

요 약

암호의 하드웨어적 구현에 있어서 구현된 IP의 테스트 환경 구축은 필수적인 사항이다. 하드웨어 암호 IP가 수정을 거치더라도 테스트가 원활하게 진행되기 위해서는 수정사항을 빠르게 반영할 수 있는 테스트 환경이 요구된다. 본 연구에서는 SoC보드상에서의 AMBA AXI 인터페이스를 이용한 테스트 환경 구축으로 암호 IP와 임베디드 OS간의 빠른 데이터 전송과 암호 IP의 암호화 방식에 따른 범용 API를 구현하였다.

I. 서론

기존의 하드웨어로 구현된 암호 IP(Intellectual Property)의 테스트는 외부 통신 인터페이스를 통한 FPGA와 PC의 연결로 이루어지는 것이 일반적이다. 이와 같은 환경에서 테스트를 실행할 경우 FPGA 외부 인터페이스의 통신 속도 때문에 전체 테스트 속도가 늦어지게 된다. 더욱이 통신 데이터의 양이 많을 경우 전체 테스트의 속도는 현저하게 저하된다. 이에 SoC(System on Chip) 보드 상에서 AMBA AXI 인터페이스를 이용한 하드웨어 암호 IP의 범용적인 테스트 환경을 구축하고자 한다.

암호 IP 테스트 환경 구축과 더불어 실질적인 암호 IP 테스트를 실시하기 위해 블록암호 AES(Advanced Encryption Standard)와 해시암호 SHA(Secure Hash Algorithm)의 하드웨어 구현 또한 함께 진행하였다.

1.1 AES

고급 암호화 표준이라고 불리는 AES 암호 알고리즘은 DES 를 대체한 암호 알고리즘이며 암호화와 복호화 과정에서 동일한 키를 사용하는 대칭 키 알고리즘이다. 가변 길이의 블록과 128/192/256bit 의 가변 길이의 키 사용이 가능하다.

AES 의 암/복호화 과정은 먼저, 128, 192 또는 256 비트 길이인 하나의 주 암호화 키를 받아서 매 라운드에서 사용할 여러 개의 128 비트 라운드 키를 생성한다. 그리고 위의 단계에서 생성한 라운드 키 중 첫번째 키를 사용, AddRoundKey 를 한 번 실행한다. 1~(9, 11, 13) 라운드에서는 SubBytes, ShiftRows, MixColumns, AddRoundKey 를 순서대로 실행한다. 이것을 AES-128, 192, 256 에 따라 각각 9 번, 11 번, 13 번 반복한다. 마지막 (10, 12, 14)번째 라운드에서는 MixColumns 를 제외한 SubBytes, ShiftRows, AddRoundKey 를 순서대로 실행한다.

본 연구에서 구현한 AES 는 주 암호화 키로 128, 192, 256 비트 길이를 모두 지원하며 그

림 1 과 같은 암/복호화 흐름을 가진다. 최초의 IDLE 상태에서 키 길이, 암/복호화 키, 메시지가 입력되면 암/복호화 상태에 진입하게 되고 마지막 블록의 암/복호화 전 까지 현재 상태가 반복된다. 마지막 블록의 암/복호화의 경우 FIN 신호가 입력되면서 상태에 진입하게 된다. 마지막 암/복호화가 종료되면 DONE 상태로 진입하고 다음 클럭에 다시 IDLE 상태로 되돌아가게 된다. 암/복호화 흐름은 테스트 환경 구축 후에 암호 IP 의 테스트 API 생성 시에 사용된다.

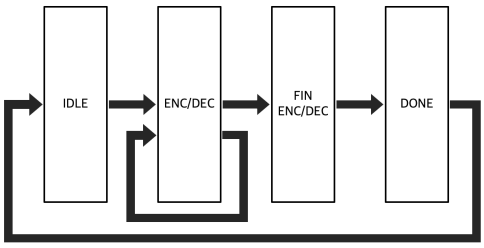


그림 1 AES 모듈 암/복호화 흐름

1.2 SHA

SHA(Secure Hash Algorithm, 안전한 해시 알고리즘) 함수들은 서로 관련된 암호학적 해시 함수들의 모음이다. 이들 함수는 미국 국가안보국(NSA)이 처음으로 설계했으며 미국 국가 표준으로 지정되었다.

SHA256/384 는 해시 암호화를 시작하기 전에 메시지의 길이가 512/1024 비트의 배수가 되도록 padding 을 추가하는 전처리 단계를 거치고 512/1024 비트 별로 해시 암호화를 진행한다. 해시 라운드에서 이전 라운드의 결과는 다음 라운드의 해시 암호화에 영향을 주고 마지막으로 생성되는 해시 코드는 256/384 비트의 길이를 가진다.

본 연구에서 SHA 는 SHA256 과 SHA384 를 하나의 모듈로 통합해서 구현하였으며 그림 2 와 같은 암호화 흐름을 가진다. 최초의 IDLE 상태에서 암호화 모드(SHA256 or SHA384), 메시지 길이, 메시지가 입력되면 해시 암호화 상태에 진입하게 된다. 해시 암호화가 종료되면 DONE 상태로 진입하고 다음 클럭에 다시 IDLE 상태로 되돌아가게 된다. 암호화 흐름은 테스트 환경 구축 후에 암호 IP 의 테스트 API 생성 시에 사용된다.

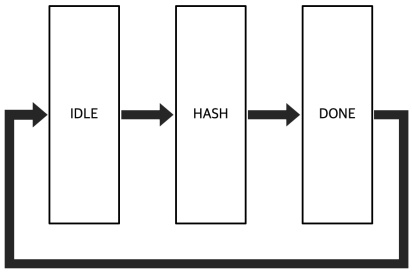


그림 2 SHA 모듈 암호화 흐름

1.3 AXI bus Interface

SoC 내부에서 IP 들을 연결하기 위해서는 규칙이 필요하다. 그 중에서 ARM 이 주도하여 bus 들 끼리 어떻게 연결하고, IP 끼리 서로 어떻게 통신 할 것이냐를 약속한 것(protocol)이 AMBA (Advanced Microcontroller Bus Architecture)이다. 그리고 그 내부에서 bus 의 통신방식을 정해놓은 규격을 bus Interface 라고 한다. AXI 는 AMBA 의 bus Interface 중 하나이며 여러 IP 를 연결하기 위해 채널 개념을 도입하여 칩 상의 bus 네트워크를 불필요하게 점유하는 상황을 피할 수 있도록 하였다. 총 5 개의 채널을 가지며 주소 채널과 데이터 채널을 독립적으로 분리하여 사용한다.

AXI 는 첫 주소를 보내고 다음 주소부터는 첫 주소부터 차례대로 증가시켜 이후부터는 데이터의 전송만 이루어지는 burst 모드라는 전송 방법을 가지는데 본 연구에서는 burst 모드를 사용하지 않는 AXI4 Lite sub Interface 를 사용한다.

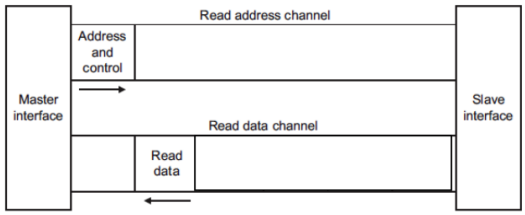


그림 3 AXI4 Lite Read 채널

AXI4 Lite 의 Read 는 Master 에서 Read address 채널을 통해 데이터를 읽을 주소와 컨트롤 신호를 Slave 로 보내고, Slave 에서 Read data 채널을 통해 Master 로부터 전송받은 주소의 Read 데이터를 Master 에 전송하는 방식으로 동작한다.

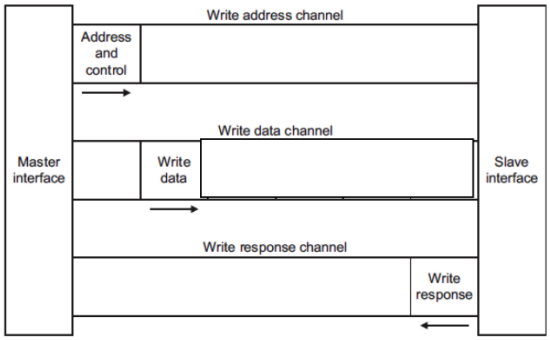


그림 4 AXI4 Lite Write 채널

AXI4 Lite의 Write는 Master에서 Write address 채널을 통해 데이터를 쓸 주소와 컨트롤 신호를 Slave로 보낸 뒤에 Write data 채널을 통해 Write 데이터를 Slave로 전송하고, Slave에서 Write response 채널을 통해 Master에게 쓰기 동작의 결과(성공/실패)를 전송하는 방식으로 동작한다.

1.4 구현

본 연구에서는 XILINX社의 Zynq UltraScale+ MPSoC ZU3EG A484 프로세서를 사용한 AVNET社의 Ultra96 보드를 사용하여 구현을 진행했다. Zynq UltraScale+ MPSoC 시리즈의 프로세서는 임베디드 보드와 유사하게 동작하는 Processing System(PS) 영역과 FPGA 보드와 유사하게 동작하는 Programmable Logic(PL) 영역으로 구분된다. 앞서 구현된 AES와 SHA 암호 IP는 PL 영역에 들어가게 되고 PS 영역에는 임베디드 리눅스 OS가 들어가게 된다.

먼저, 구현된 AES와 SHA 암호 IP는 AXI4 Lite의 Slave로 동작하기 위해 표1의 AXI4 Lite Slave의 인터페이스에 맞게 wrapping 하였다.

Slave Interface Write Address
S_AXI_AWADDR
S_AXI_AWVALID
S_AXI_AWREADY
Slave Interface Write Data
S_AXI_WDATA
S_AXI_WSTRB
S_AXI_WVALID
S_AXI_WREADY
Slave Interface Write Response
S_AXI_BRESP
S_AXI_BVALID
S_AXI_BREADY
Slave Interface Read Address
S_AXI_ARADDR
S_AXI_ARVALID
S_AXI_ARREADY
Slave Interface Read Data
S_AXI_RDATA
S_AXI_RRESP
S_AXI_RVALID
S_AXI_RREADY

표 1 AXI4 Lite Slave Interface 신호

암호 IP가 AXI4 Lite Slave Interface에 맞게 wrapping이 되면 AXI bus에 연결될 준비가 된다. IP와 bus 및 PS 영역 간의 연결은 XILINX社의 VIVADO라는 프로그램을 사용하였고 AXI bus의 IP는 프로그램에서 제공된다. wrapping 된 AES와 SHA 두 IP는 AXI bus와 AXI4 Lite 프로토콜로 연결되며 이때 AXI bus는 Master로, 암호 IP들은 Slave로써 동작하게 된다. PS 영역 또한 AXI bus와 연결되며 이때는 PS 영역이 Master로, AXI bus는 Slave로써 동작한다.

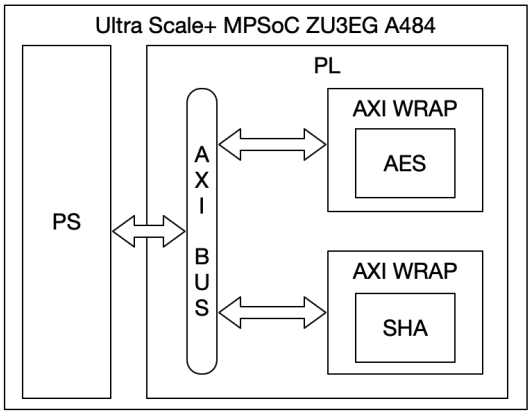


그림 5 AXI bus 연결 구성

연결이 완료되면 암호 IP들을 제어할 임베디드 리눅스 OS가 필요하므로 본 연구에서는 XILINX社의 Petalinux tools를 이용하여 OS 이미지를 만들었다. 생성된 OS를 이용하여 정상적으로 SoC 보드를 부팅 하면 암호 IP들에 할당된 주소 영역에 접근하기 위한 API를 만들어 주어야 하며 이는 메모리 매핑 방식을 이용하였다. API 생성시에는 블록암호 AES의 경우, 입력 값인 암/복호화 키 길이, 암/복호화 여부, 암/복호

화 키, 메시지 등을 API의 입력 인자로 만들어 주었고 해시 암호 SHA의 경우, 암호화 모드, 메시지 길이, 메시지를 API의 입력 인자로 만들어 주었다. 하드웨어 암호 IP의 테스트는 구현된 API를 이용하여 암호 IP의 암/복호화 흐름에 따라 시나리오를 구성하였다.

1.5 결과

구성된 테스트 환경에서의 암호 IP 테스트는 표2와 같은 결과를 보였다. 기존의 SPI 인터페이스를 이용한 테스트 환경보다 AES에서는 약 12배, SHA에서는 약 9배의 성능 향상을 얻을 수 있었다.

	AES-CBC-128		SHA256	
	AXI	SPI	AXI	SPI
소모 시간	0.053 ms	0.632 ms	0.134 ms	1.141 ms

표 2 암호 IP 테스트 속도 비교

또한, 유사한 종류의 암호화 알고리즘을 사용하여 IP를 테스트할 경우, 미리 구성한 테스트 환경을 다시 이용하여 실행할 수 있기 때문에 블록암호 ARIA에 적용한 경우는 AES의 테스트 환경을 그대로 활용하여 하드웨어 암호 IP의 전체적인 개발 속도를 향상 시킬 수 있었다.

II. 결론

본 연구에서는 SoC보드 상에서 AMBA AXI 인터페이스를 이용하여 하드웨어 암호 IP의 테스트 환경을 구축하였고 한번 구성한 테스트 환경을 활용하여 다양한 하드웨어 암호 IP를 고속으로 테스트할 수 있을 것으로 보인다.

ACKNOWLEDGEMENT

본 연구는 IDEC에서 EDA Tool를 지원받아 수행하였습니다.

[참고문헌]

[1] "Advanced Encryption Standard (AES)", National Institute of Standards and Technology, November 26 2001, [online] Available: http://csrc.nist.gov/publications/fips/fips197/fips197.pdf.

[2] "Secure Hash Signature Standard (SHS)", National Institute of Standards and Technology, August 1 2002, [online] Available: https://csrc.nist.gov/csrc/media/publications/fips/180/2/archive/2002-08-01/documents/fips180-2.pdf

[3] Damjan Rakanović ; Rastislav Struharik, IP core for AES256 and TDES algorithms with AXI interface, 24th Telecommunications Forum (TELFOR), 2016.

[4] "Vivado AXI Reference Guide", National Institute of Standards and Technology, July 15 2017, [online] Available: https://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/latest/ug1037-vivado-axi-reference-guide.pdf

TEE 기반의 Intel SGX 취약점 연구 동향

김경호* 권용빈* 김현지* 김현준* 서화정*
*한성대학교 IT융합공학과

TEE-based Intel SGX Vulnerability Research Trend

Kyung-Ho Kim* Yong-Bin Kwon* Hyun-Ji Kim*
Hyun-Jun Kim* Hwa-Jeong Seo*
*Division of IT Convergence Engineering, Hansung University.

요 약

Intel SGX(Software Guard Extension)는 하드웨어 기반의 독립된 실행환경을 제공하여 데이터의 무결성 및 기밀성을 유지하는 대표적인 TEE(Trusted Execution Environment) 보안 플랫폼이다. 하지만 SGX의 독립된 실행환경은 CPU 코어, Cache, 메모리 등 프로세서의 실행 환경을 그대로 사용한다. 따라서 기존 프로세서의 취약점을 이용해 Intel SGX에서 동작하는 데이터의 무결성 및 기밀성을 훼손하는 다양한 취약점 연구가 진행 중이다. 본 논문에서는 Intel SGX에 대한 취약점 연구 동향에 대해서 살펴보고, 이에 따른 향후 연구 전망을 제시하고자 한다.

I. 서론

현대의 컴퓨터 시스템은 운영체제와 같은 시스템 소프트웨어를 기반으로 사용자의 편의성을 극대화한다. 이러한 시스템 소프트웨어는 시스템의 전반적인 정보를 관리하기 위해 관리자 권한을 설정하여 시스템을 관리한다. 따라서 공격자가 비정상적인 방법으로 관리자 권한을 빼앗게 된다면 해당 시스템에서 동작하는 프로세스들의 데이터 무결성 및 기밀성이 훼손될 수 있다.

Intel SGX는 기존 시스템의 단점을 극복하기 위해 하드웨어 기반의 독립된 실행환경을 제공한다. 독립된 실행환경에서 동작하는 응용 프로그램의 데이터는 시스템 소프트웨어나 다른 응용 프로그램이 비정상적으로는 절대 접근할 수 없다. 따라서 공격자가 시스템의 관리자 권한을 탈취하더라도 SGX 내부에서 동작하는 데이터는 무결성과 기밀성을 유지할 수 있다.[1,2]

하지만 Intel SGX는 하드웨어적으로 완벽히 분리된 것이 아니라 CPU 코어, Cache, 메모리 등 기존 프로세스의 실행환경을 공유한다. 따라서 Meltdown 및 Spectre, Cache 접근 시간을 이용한 부채널 공격, SGX 내부에서 동작하는 코드에 사용된 취약점 함수를 이용한 ROP (Return-Oriented Programming) 공격 등 SGX의 독립된 실행환경에 기존 프로세서의 취약점을 적용시킨 다양한 연구 사례가 지속적으로 발표되고 있다.

본 논문에서는 Intel SGX에 대한 대표적인 취약점 연구 동향에 대해서 살펴보고자 한다. 본 논문의 구성은 다음과 같다. 2장에서 Intel SGX에 대한 설명과 기존 프로세서 및 프로그램의 취약점에 대해 기술한다. 3장에서는 Intel SGX의 최신 취약점 연구 동향을 알아보고 4장에서는 결론 및 향후 연구 방향에 대해 기술한다.

II. 관련 연구

2.1 Intel SGX

Intel SGX는 대표적인 TEE 기반의 보안 플랫폼으로 Enclave라는 독립된 실행환경을 제공한다. Enclave는 PRM(Processor Reserved Memory)이라는 Enclave 전용 메모리를 따로 분리하여 다른 소프트웨어의 접근을 차단한다. 또한 Enclave가 사용하는 분리된 메모리는 MEE (Memory Encryption Engine) 기술을 이용하여 데이터가 암호화된 뒤 저장되기 때문에 공격자의 직접 물리 메모리 공격에 대한 대응성을 가진다.

또한 Enclave는 확장성을 위해 기존 프로세스의 주소변환 체계를 그대로 사용한다. 따라서 비정상적인 분기문을 이용하여 Enclave에 대한 접근을 할 수 있는데 이런 경우 비정상적인 방법으로 Enclave 메모리에 접근한 경우 Abort page semantic을 발생시켜 모든 값을 -1로 리턴하고 모든 연산을 무시하여 비정상적인 접근을 차단한다.

Intel SGX는 ECALL과 OCALL을 이용하여 Enclave를 동작한다. ECALL 함수는 Enclave 내부에서 동작하는 함수를 호출하는 것으로 ECALL 함수 호출된 이후로는 Enclave에서 동작하기 때문에 안전한 실행환경을 보장한다. 그리고 OCALL 함수는 Enclave 내부에서 동작하는 함수에서 신뢰할 수 없는 영역의 함수를 호출하는 것으로 신뢰할 수 있는 영역과 신뢰할 수 없는 영역 사이에 안전한 방식으로 동작할 수 있도록 지원한다.

또한 Intel SGX는 Key Derivation Material을 이용해 다양한 종류의 암호화 키를 제공한다.[1] Intel SGX에서 사용되는 다양한 종류의 키 값은 Enclave 생성 중에 생성되는 매개 변수의 값에 따라 결정되기 때문에 공격자가 키 값을 알아내는 것이 불가능하다.

2.2 Meltdown & Spectre

2018년에 새롭게 발표된 취약점인 Meltdown과 Spectre는 프로세서의 연산시간 단축을 위한 최적화 기법의 취약점을 이용한 공격이다.[3,4]

Meltdown은 Intel CPU에서 발생된 취약점으로 최적화 기술 중 하나인 비순차 실행으로 인해 발생하는 취약점을 이용한 공격이다. 최신 프로세서는 다수의 연산을 한 번에 처리할 수 있기 때문에 서로 의존성이 없는 연산은 순서에 상관없이 실행한다. 이 과정에서 사전에 연산된 결과를 Cache에 저장하는데 이러한 특징을 이용하여 순차적인 실행에서는 실행되지 않을 비밀 데이터 접근 명령이 비순차 실행으로 인하여 Cache에 저장된다. 그리고 Cache Timing Attack을 통해 Cache에 저장된 비밀 정보를 탈취한다.

Spectre는 대부분의 프로세서가 사용하는 예측 실행으로 인해 발생하는 취약점을 이용한 공격이다. 대표적인 예측 실행으로 분기 예측이 있는데 특정 조건문의 결과가 지속적으로 동일하면 해당 명령을 사전에 연산한 뒤 Cache에 저장한다. 이 때 조건문 결과가 다르면 Cache에 저장된 데이터는 Flush하지만 Cache에 로드되고 Flush되는 사이에 Cache Timing Attack을 통해 비밀 정보를 탈취한다.[5]

2.3 ROP Attack

프로세스가 실행되면 운영체제는 가상 메모리를 할당하여 명령어 및 라이브러리 함수를 할당한다. 호출된 함수는 실행된 이후에 이전에 호출한 함수로 돌아가야 하기 때문에 스택 메모리에 이전 함수의 주소를 저장한다. 따라서 모든 함수에는 실행이 끝난 후 스택 메모리에 저장된 주소로 분기하는 명령어가 포함된다.

ROP Attack은 이러한 특징을 이용하여 취약점이 존재하는 라이브러리 함수를 사용한 경우 Return 주소를 공격자가 임의로 변경하여 원하는 연산을 수행할 수 있다.

ROP Attack을 막기 위해 ASLR(Address Space Layout Randomization), DEP(Data Execution Prevention) 등 대응 기법이 존재하지만 PLT(Procedure Linkage Table) 및 GOT(Global Offset Table)를 이용한 우회 기법 또한 존재한다. 이러한 공격을 막기 위한 방법은 취약점이 존재하는 함수를 사용을 자제하는 것이다.[6]

III. 취약점 동향

3.1 Foreshadow

2018년에 발표된 Foreshadow Attack은 Intel CPU의 취약점인 Meltdown을 Intel SGX에 적용한 공격이다. Enclave는 어떠한 권한의 소프트웨어도 접근 할 수 없어야 하지만 비순차 실행을 이용하여 비정상적인 접근을 통해 Cache에 비밀 데이터 저장이 가능하다.[7]

Foreshadow의 공격 과정은 Meltdown과 동일하다. 우선 Enclave의 비밀 데이터에 접근하는 명령어를 이용한다. 정상적인 실행 흐름이라면 Abort page semantic이 발생하여 접근을 차단하지만 비순차 실행에 의하여 모두 비워진 L1 Cache에 비밀 데이터를 저장한 뒤 데이터 접근 속도 차이를 이용한 Cache Timing Attack인 Flush + Reload 방식을 이용하여 비밀 데이터를 탈취한다.

기존의 Meltdown과 같이 관리자 권한이 없어도 공격이 가능하지만 관리자 권한을 이용하면 데이터를 L1 Cache에 정확하게 저장할 수 있고 공격의 성공률을 높일 수 있다.

Intel은 Foreshadow 공격에 대응하기 위해 소프트웨어 보안 패치 및 하드웨어 마이크로코드 수정을 진행하였고 Intel TSX(Transactio-nal Synchronization Extensions)을 이용한 예외 탐지 연구 또한 활발히 진행되고 있다.

3.2 Cache Attack

Intel SGX는 부채널 공격에 내성을 가지지 않는다. 또한 일반적인 프로그램과 똑같이 Cache 메모리를 사용하기 때문에 Enclave 내부에서 동작하는 응용 프로그램이 Cache를 사용하는 경우 접근 속도 차이를 이용한 부채널 공격 연구 사례가 발표되었다.[8,9]

대부분의 Cache Attack은 동일한 코어에서 하이퍼쓰레딩을 이용하여 공격자 쓰레드와 희생자 쓰레드를 동시에 실행시킨다. 하이퍼쓰레딩의 경우 각 쓰레드가 L1 Cache를 공유하게 되는데 이 때 Intel PMC(Performance Monitori-ng Counter)를 이용하여 Cache Hit, Miss를 판단하여 Cache에 저장된 데이터를 탈취 할 수 있다.

대표적인 Cache Attack으로 희생자 쓰레드에서 동작하는 T-Table을 이용한 AES 암호화 알고리즘을 마지막 라운드에서 PMC를 이용하여 Prime + Probe 방식의 Cache Timing Attack을 이용하여 T-Table을 알아내고 Neve&Seifert의 제거(Elimination) 방법을 구현하여 AES의 마지막 라운드의 비밀 키를 추출할 수 있다.

3.2 ROP Attack

보편적인 해킹 기법인 ROP 공격은 동일한 방식의 취약점으로 Enclave 내부에서도 실행될 수 있다. 우선 공격자가 Enclave 내부의 코드에 대해서 역공학(Reversing)을 통해 취약점이 존재하는 코드를 알고 있다는 전제 하에 공격을 진행한다.[10]

Enclave에서 동작하는 응용프로그램의 경우 공격자가 정적 분석을 통해서 Gadget을 얻을 수 없기 때문에 Enclave의 AEX(Asynchronou-s Enclave Exit)의 특징을 이용하여 찾아낸 Gadget의 POP 명령어 수를 알아낸다. 그 이후 찾아낸 Gadget들 중 공격에 필요한 Gadget를 찾기 위해 Enclave에서 사용하는 명령어인 ENCLU 명령어를 사용하여 Leaf Function으로 지정된 함수들 중 EEXIT 함수 실행을 확인하여 공격에 필요한 Gadget을 찾아낸 후 Gadget Chaining을 이용하여 공격을 수행한다.

이러한 ROP Attack을 막기 위해 실행 시 Enclave 메모리를 무작위화하는 대응방안인 SGX-shield와 같은 다양한 연구가 진행 중이다.

3.3 SGX-Bomb

대부분의 취약점은 기존의 운영체제 및 시스템에 존재하는 취약점을 Intel SGX에 적용시켜서 공격한다. 하지만 SGX-Bomb의 경우 하드웨어적 결함인 Rowhammer Attack을 이용하여 Enclave의 무결성 검사를 실패하게 만들어 서비스 거부 공격(Denial-of-Service At-tack)을 수행한다.[11]

SGX-Bomb은 DRAM의 Cell 밀집도가 높아짐에 따라 동일한 메모리 뱅크에 반복적으로 접근하여 하드웨어적 결함으로 인한 Bit flip을

발생시키는 Rowhammer Attack을 수행하여 Enclave 내부의 메모리의 값의 변동을 일으킨다. Enclave는 비정상적인 데이터의 변화를 막기 위해 주기적으로 무결성 검사를 수행하는데 무결성 검사에 실패하게 되면 소프트웨어 방어를 목적으로 프로세스를 중지한다. 그 결과 Intel SGX상에서 원격 서비스를 제공하는 서버의 경우 프로세스가 중지되는 서비스 거부 공격을 당하게 된다.

IV. 결론

본 논문에서는 TEE 기반의 Intel SGX에 대한 최신 취약점 연구 동향에 대해서 알아보았다. Intel SGX에서 발생하는 대부분의 취약점은 기존 시스템 취약점을 적용시킨 공격이 대부분이다. Intel은 이러한 취약점을 막기 위해 소프트웨어 보안 패치 및 마이크로코드 업데이트를 진행하고 있다. 하지만 여전히 Cache를 이용한 부채널 공격에 대한 대응 방안을 제공하지 않기 때문에 사용자가 부채널 공격에 강인한 응용 프로그램을 제작해야 한다.

본 논문에서 알아본 취약점 연구 동향을 토대로 앞으로의 연구 방향은 Intel SGX 상에서 부채널 공격에 안전한 암호 구현 및 새로운 취약점 발견과 그에 맞는 대응 방안을 제시하는데 있다.

[참고문헌]

[1] Costan, Victor, and Srinivas Devadas. "Intel SGX Explained." IACR Cryptology ePrint Archive 2016.086 (2016): 1-118.

[2] Payne, Ray, et al."Integrated security suite architecture and system software/hardware." U.S. Patent Application No. 10/843,180.

[3] Lipp, Moritz, et al. "Meltdown." arXiv preprint arXiv:1801.01207 (2018).

[4] Kocher, Paul, et al. "Spectre attacks: Exploiting speculative execution." arXiv preprint arXiv:1801.01203 (2018).

[5] Gruss, Daniel, et al. "Flush+ Flush: a fast and stealthy cache attack." International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, Cham, 2016.

[6] Hund, Ralf, Carsten Willems, and Thorsten Holz. "Practical timing side channel attacks against kernel space ASLR." 2013 IEEE Symposium on Security and Privacy. IEEE, 2013.

[7] Van Bulck, Jo, et al. "Foreshadow: Extracting the keys to the intel {SGX} kingdom with transient out-of-order execution."27th {USENIX} Security Symposium ({USENIX} Security 18). 2018.

[8] Brasser, Ferdinand, et al. "Software grand exposure:{SGX} cache attacks are practical." 11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17). 2017.

[9] Götzfried, Johannes, et al. "Cache attacks on Intel SGX." Proceedings of the 10th European Workshop on Systems Security. ACM, 2017.

[10] Lee, Jaehyuk, et al. "Hacking in darkness: Return-oriented programming against secure enclaves." 26th {USENIX} Security Symposium ({USENIX} Security 17). 2017.

[11] Jang, Yeongjin, et al. "SGX-Bomb: Locking down the processor via Rowhammer attack." Proceedings of the 2nd Workshop on System Software for Trusted Execution. ACM, 2017.

공인인증서 유출 방지를 위한 Intel SGX 기반 관리 방안 연구*

배재건, 공성현, 이창훈†

서울과학기술대학교 컴퓨터공학과

A Study on Intel SGX-based Management Scheme to Prevent Certificate Leakage Accident

Dept. of Computer Science and Engineering, Seoul National University of Science and Technology

요 약

개인의 전자 신분증 역할을 하는 공인 인증서는 은행 서비스 및 정부 민원 서비스 등 다양한 서비스에서 이용자의 신원을 증명하는데 사용된다. 그러나 현재 공인인증서는 단순히 PC 및 모바일 기기에 저장되는 형태로 파일 복사를 통해 간단히 복제가 가능하다. 공격자가 공인인증서의 비밀번호만 획득한다면 공인인증서를 복사하여 범죄에 악용할 수 있다. 이를 방지하기 위해 본 논문에서는 Intel SGX를 활용하여 공인인증서를 안전하게 관리할 수 있는 방안을 제안한다. 사용자는 Intel SGX의 Sealing 기술을 이용하여 발급받은 공인인증서를 암호화하여 사용자의 PC 또는 SGX가 적용된 클라우드에 저장함으로써 이를 통해 공격자가 공인인증서를 탈취하는 것을 방지할 수 있다.

I. 서론

공인인증서는 1999년 시행된 전자 서명법에 의해 도입되었으며 공개키 암호를 기반으로 인터넷에서 개인의 신원을 증명하는데 이용된다[1]. 공인인증서는 인증기관에서 인증하는 증명서로 서비스 제공자는 사용자의 공개키를 획득하여 통신에 필요한 대칭키를 암호화 하여 사용자에게 전송한다. 사용자는 개인키를 이용해 대칭키를 복호화하여 안전하게 전달받을 수 있으며 공개키 소유자의 공개키를 통해 전자 서명을 검증 할 수 있다. 인증기관으로부터 발급받은 공인인증서는 PC, USB, 스마트폰 등 사용자가 지정한 저장장치에 저장된다. 사용자가 저장한 공인 인증서는 단순히 인증서 파일을 복사하는 방법을 통해 다른 저장장치로 옮길 수 있다. 공격자가 사용자의 PC를 감염시키는데 성공하였을 경우 공인인증서를 단순히 복사하기만 하면 손쉽게 탈취가 가능하며 공인인증서의 비밀번호만 획득하면 민원 신청, 인터넷 뱅킹 등 공인인증서를 사용하는 모든 시스템을 이용할 수 있다. 공인인증서는 2019년 발급 건수는 사상 최대를 기록하였으며 정부 민원 시스템, 인터넷 뱅킹 등 여전히 많은 시스템에서 사용되고 있다[2].

Intel SGX(Software Guard eXtensions)는 Intel 스카이레이크 아키텍처부터 도입되었으며, 애플리케이션의 코드와 데이터를 보호하기 위해 추가된 명령어 집합이다. SGX는 Enclave라는 고립된 영역을 통해 고립된 영역의 메모리 접근을 차단하는 기술이다. Enclave 밖으로 데이터를 내보내려면 Sealing을 통해 암호화하여 내보내게 된다. 공격자가 OS까지 감염시켜도 Enclave에서 암호화된 데이터를 복호화 할 수

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구 (No.2017-0-00344, 최신 모바일 기기에 대한 암호해독 및 포렌식 분석)

† 교신저자, chlee@seoultech.ac.kr (Corresponding author)

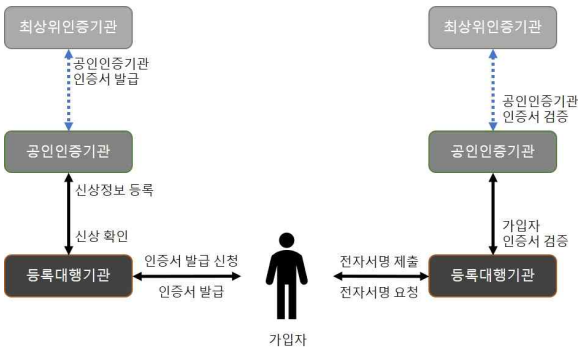
없기 때문에 안전하게 데이터를 보호할 수 있다. 공인인증서를 Sealing을 통해 암호화하여 저장한다면 악성코드나 감염된 OS 등으로부터 안전하게 보관할 수 있다. 따라서 본 논문에서는 공인인증서를 안전하게 보관하기 위하여 Intel SGX를 이용한 공인인증서 관리 방안을 제안하고자 한다.

본 논문의 구성은 2장에서 관련 기술을 소개하고 3장에서 Intel SGX를 이용한 공인인증서 관리 방안을 제안한다. 마지막으로 4장에서 결론짓는다.

II. 관련 기술

2.1 공인인증서

공인인증서는 1999년 7월 전자서명법을 통해 도입되었다. 전자서명법 제4조의 규정의 의해 2019년 현재 6개의 공인인증기관이 존재하며 이러한 인증기관으로부터 발급받은 인증서를 의미한다[3]. 공인인증서에는 가입자의 이름, 전자서명 검증정보, 가입자 및 공인인증기관이 이용하는 전자서명 방식 그리고 공인인증서의 일



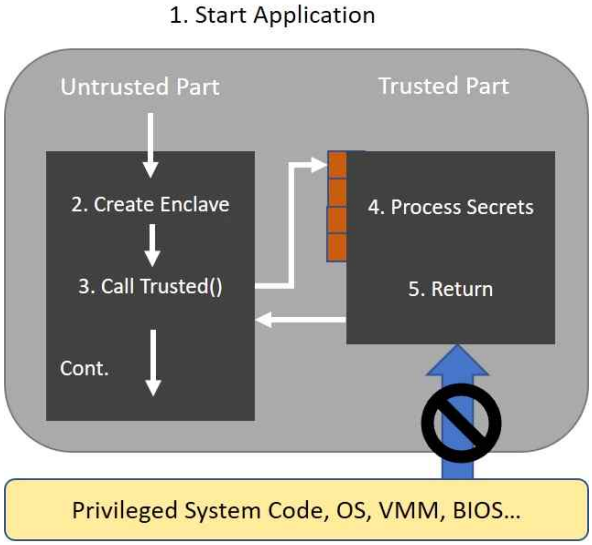
(그림 1) Procedure for Issuing Certificates

련번호 등 다수의 정보가 포함되어있다.

공인인증서 발급 및 사용과정은 그림 1[3]과 같다. 사용자는 공인인증서 발급을 위해 등록대행기관인 은행 등의 홈페이지에서 인증서 발급을 신청한다. 등록대행기관은 사용자의 신원을 식별 및 확인하고 공인인증기관에 신청 정보를 등록한다. 이후 등록대행기관을 거쳐 사용자는 공인인증서를 발급받게 된다. 사용자가 공인인증서를 필요로 하는 서비스를 사용할 경우 해당 서비스 제공자는 전자서명을 요청한다. 사용자는 전자서명을 제출하고 서비스 제공자는 이를 공인인증기관을 통해 검증한다. 사용자는 공인인증기관으로부터 인증서 검증에 성공한 경우 서비스를 제공받을 수 있다.

2.2 Intel SGX

Intel SGX는 Intel 스카이레이크 아키텍처부터 적용된 새로운 명령어 집합이다[4]. 개발자는 SGX를 통해 보호할 코드와 데이터를 선택할 수 있다. SGX를 사용하면 메모리상에 Enclave라는 고립된 영역을 생성하며 모든 접근은 CPU에 의해 보호된다. Enclave는 안전하지 않은 응용프로그램에 안전한 영역인 샌드박스를 구현한 것으로 해당 영역은 관리자 권한으로 실행되는 프로세스, 운영체제, 하이퍼바이저 등의 접근이 불가능하다. 따라서 공격자의 공격 벡터를 최소화할 수 있으며 데이터를 보다 안전하게 보호할 수 있다.



(그림 2) SGX Application Execution Flow

SGX를 사용하는 응용프로그램은 Untrusted Part와 Trusted Part로 구성된다. Untrusted Part는 Enclave를 제외한 영역을 의미하며 Trusted Part는 Enclave를 통해 보호받는 영역을 의미한다. Untrusted Part는 ECall(Enclave Interface Function)을 통해 Enclave에 접근이 가능하며 Enclave에서는 OCall(Calls outside the Enclave)을 통해 Untrusted Part에 접근이 가능하다. SGX를 이용하는 응용프로그램의 실행 과정은 그림 2[4]와 같다. 응용프로그램이 실행되면 Untrusted Part와 Trusted Part를 구분한 뒤 Enclave를 생성한다. Trusted Function을

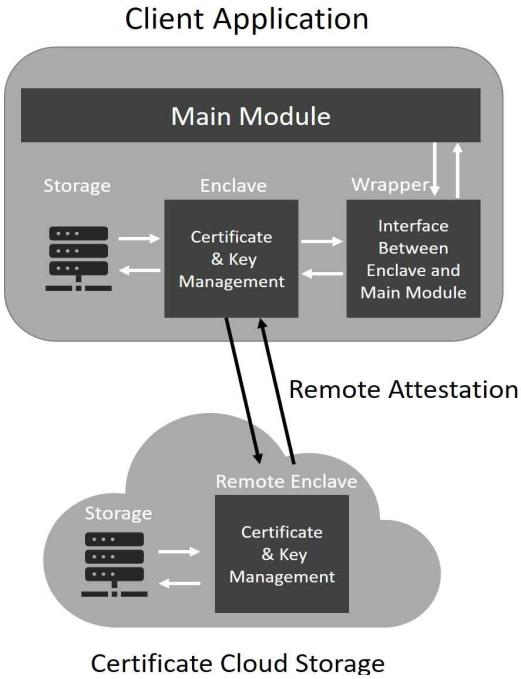
호출하게 되면 실행 흐름이 Enclave 내부로 전환된다. Enclave에서 메모리에 대해 외부 접근을 제한하며 Trusted Function을 수행한다. Trusted Function이 수행된 뒤 Enclave에 데이터가 남아있어도 메모리 영역에 암호화 되어 있기 때문에 데이터는 안전하게 보호된다. 이후 응용프로그램은 Untrusted 영역으로 전환되어 나머지 코드 영역을 실행하게 된다.

SGX에서는 Enclave에서 사용한 데이터를 파일 시스템 등에 저장해야 할 경우를 위해 Sealing 기능을 제공한다[5]. Sealing 기능을 제공하기 위해 SGX는 Enclave에 의해 생성된 키를 이용해 데이터를 암호화하여 저장한다. 암호화된 데이터는 Enclave에서 Unsealing을 통해 다시 복호화 할 수 있다. 키 정책에 따라 완전히 동일한 Enclave만 Unsealing 할 수 있을지, 같은 시스템, 같은 소프트웨어 벤더의 응용프로그램이 생성한 다른 Enclave 또한 Unsealing 할 수 있을지 결정할 수 있다.

서로 다른 Enclave 간 안전한 통신을 위해 SGX는 Remote Attestation 기술을 제공한다[6]. 해당 기술을 이용하면 SGX를 이용하는 두 클라이언트 간 신뢰할 수 있는 통신 채널을 생성할 수 있으며 SGX가 실행중이고 훼손되지 않았음을 증명할 수 있다.

III. Intel SGX를 이용한 공인인증서 관리 방안

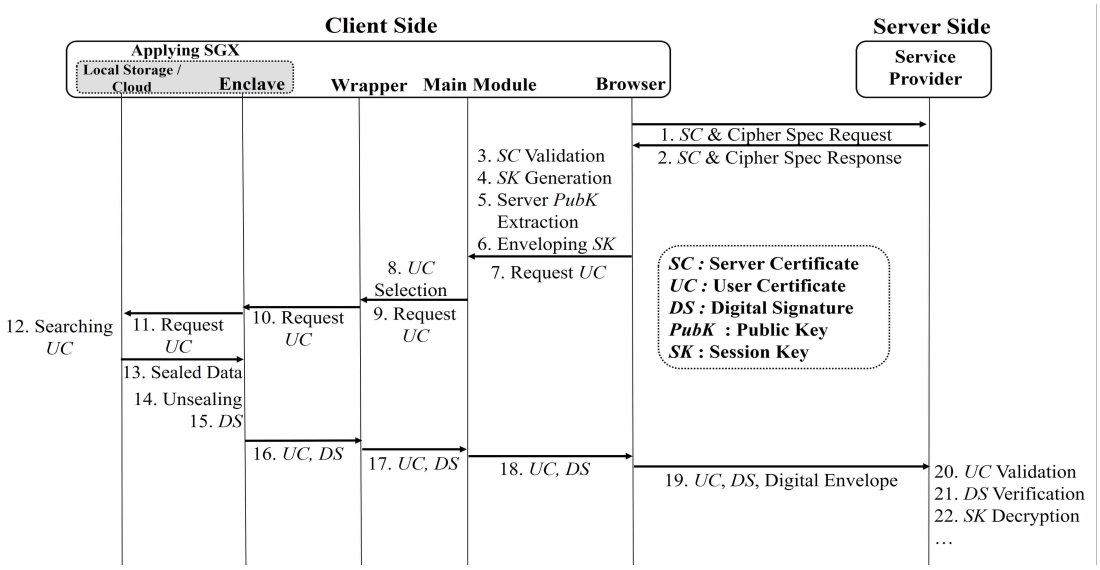
본 절에서는 Intel SGX의 Sealing 기술을 이용하여 공인인증서를 안전하게 관리할 수 있는 방안에 대해 제안한다. 공인인증서를 유출로부터 안전하게 관리하기 위하여 공인인증서 소프트웨어에 Intel SGX를 적용한다. 제안하는 Intel SGX를 이용한 공인인증서 관리 소프트웨어의 구조는 그림 3과 같다. 기존 공인인증서 관리 소프트웨어의 기능을 담당하는 Main Module, 공인인증서 파일 및 키 관리를 담당하는 Enclave 영역, Enclave 영역과 Main Module간의 연동 및 구현상의 용이함을 위한 Wrapper로 구성된다. 또한 사용자는 공인인증서 보관 장소를 SGX를 사용하는 클라우드 또는 자신의 PC로 선택할 수 있다. 공인인증서를 관리할 클라우드 또한 SGX를 통해 공인인증서를 관리한다. SGX를 적용하여 공인인증서를 보관 할 경우 클라우드 서버의 데이터가 유출되어도 Enclave를 거치지 않고 저장된 공인인증서를 복호화 할 수 없기 때문에 안전하게 보관할 수 있다. 사용자의 Enclave 내의 인증서를 안전하게 클라우드로 전송하기 위하여 SGX의 Remote Attestation 기술을 사용한다. 이를 통해 신뢰할 수 있는 통신 채널을 형성하여 공인인증서를 전송한다.



(그림 3) Software Structure for Certificate Management

Enclave를 통해 Sealing된 데이터는 Enclave 내에서 생성된 키를 통해 암호화되기 때문에 소프트웨어 동적 분석, 패킷 스니핑 및 메모리 분석 등을 통해서 키를 추출할 수 없다. 따라서 공인인증서를 개인 PC에 보관하여도 추가적인 보안 장치를 설치하지 않고 안전하게 공인인증서를 관리할 수 있다.

그림 4는 공인인증서를 사용하는 전형적인 Secure Transaction의 과정의 일부를 나타낸다[2]. 실제 Secure Transaction 과정에서 제안한 모델을 적용하였을 경우 SGX를 통해 보호되는 영역은 음영으로 표시하였다. User Certificate Selection 이후 해당하는 공인인증서를 저장소에서 추출하고 Digital Signature 과정에서 Intel



(그림 4) Typical Example of NPKI-Based Secure Internet Transaction with Intel SGX

SGX를 통해 공인인증서를 Enclave 외부에 노출시키지 않고 안전하게 인증이 가능하다.

IV. 결론

공인인증서를 폐지하려는 움직임에도 불구하고 2019년 공인인증서 발급 횟수는 역대 최대를 기록하였다. 공인인증서는 복사가 쉽기 때문에 사용자는 공인인증서 유출에 주의해야 한다. 유출된 공인인증서의 패스워드만 알면 뱅킹, 전자정부 민원 서비스 등 수많은 서비스에 악용이 가능하다. 이에 따라 본 논문에서는 공인인증서를 보다 안전하게 보관하기 위하여 Intel SGX를 이용한 공인인증서 관리 방안을 제안하였다. 본 논문에서 제안한 방식은 Intel 스카이레이크 이상의 아키텍처가 적용된 CPU에 모두 적용 가능하다.

최근 공인인증서 외에도 블록체인 방식 등 새로운 전자 서명 방식이 도입되고 있다. 향후 연구에서는 이러한 새로운 전자 서명 방식에서도 SGX를 통해 서명 정보를 안전하게 보관할 수 있는 방안에 대한 연구를 수행할 예정이다.

[참고문헌]

[1] 박승철. 국내 공인인증서(NPKI)와 국제 표준(SSL/TLS) 기반의 안전 인터넷 거래 비교 분석. 한국정보통신학회논문지, 20(2), 289-298, 2016.

[2] 한국경제, [숫자로 읽는 세상] 대통령도 폐지 지시한 '공인인증' 발급 오히려 크게 늘어 [Online]. Available at: https://www.hankyung.com/economy/article/2019101836311, (Accessed : 2019.11.06.).

[3] KISA 전자서명인증관리센터, 공인전자서명인증관리체계 공인인증서 SW[Online]. Available at : https://www.rootca.or.kr/kor/accredited/accredited06_01.jsp, (Accessed : 2019.10.27.).

[4] Intel, Enhanced Security Features for Applications and Data In-use, Available at : https://software.intel.com/sites/default/files/managed/6a/85/intel-sgx-product-brief-2019.pdf [Online], (Accessed : 2019.10.28.).

[5] Intel, Intel® Software Guard Extensions (Intel® SGX) Developer Guide. Available at : https://software.intel.com/en-us/sgx/sdk, (Accessed : 2019.10.28.).

[6] Intel, Code Sample: Intel® Software Guard Extensions Remote Attestation End-to-End Example[Online], Available at : https://software.intel.com/en-us/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example, (Accessed : 2019.10.30.).

Intel SGX에서의 Spectre 공격 분석

한찬희, 이만희
한남대학교 컴퓨터공학과

Survey of Spectre Attacks on Intel SGX

Chan-Hee Han, Man-Hee Lee
Department of Computer Engineering, Hannam University.

요 약

현대 사회에서는 컴퓨터로 대부분의 업무를 처리하므로 더 높은 보안이 요구된다. 이를 위해 Intel은 안전한 실행환경을 제공하는 Intel SGX를 출시하였다. Intel SGX에서는 사용자의 데이터를 안전하게 보관할 수 있는 컨테이너인 Enclave를 제공한다. 하지만 Intel SGX의 런타임 라이브러리에 부채널 공격에 취약한 가젯 코드가 발견되었고 이를 악용한 공격들이 등장하였다. 본 논문에서는 Intel SGX와 Spectre 공격에 필요한 Speculative Execution과 캐시 부 채널 공격에 관해 소개하고 SGX 환경에서의 Spectre 공격을 소개한다.

I. 서론

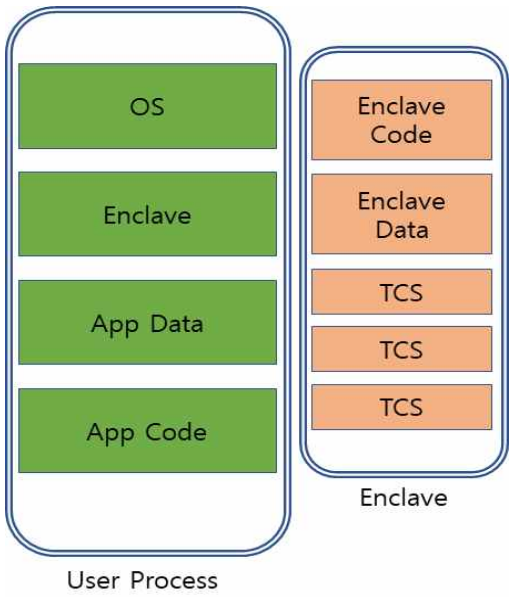
현대 사회는 모든 업무처리가 전산화되면서 컴퓨터 보안에 대한 중요도가 높아졌다. 예를 들면 카드번호, 국방, 생채 인식정보, 재무 관련 정보, 비밀번호 등 민감한 정보들이 컴퓨터 내 저장공간에 저장된다. 민감 정보들이 유출되면 재산상에 막대한 손해가 올 뿐 아니라 개인, 기업, 국가적 혼란을 초래할 수 있으므로 이에 대한 보안이 더욱 필요하다. 이에 대해 Intel 사에서 민감 정보에 대한 보안 솔루션을 제공했다. 2015년, 6세대 Intel 코어 프로세서(코드명 : 스카이레이크)부터 응용프로그램 계층에서 하드웨어 보안을 강화하는 x86 명령어 세트 아키텍처인 SGX(Software Guard eXtensions)를 적용했다[1]. SGX는 격리된 메모리 공간, 메모리 암호화, 밀폐된 저장공간을 제공함으로써 프로세서 안에 응용프로그램 보안을 더욱 강화한다. 그러나 SGX에도 취약한 코드 패턴인 가젯(Gadget) 코드가 존재한다. Intel SDK, Rust-SGX와 같은 런타임 라이브러리에 있는 가젯은 응용프로그램을 실행하는 데 필요한 코드 중 일부이기 때문에 제거할 수 없다. 한편, 2018년 1월 Google project zero 팀에서 부 채널 공격 중 하나인 Meltdown&Spectre가 발표됐다. 이 공격은 Flush+Reload, Prime+Probe와 같이 Cache hit와 Cache miss의 타이밍 차이를 활용해 희생자의 메모리 접근 패턴을 유추한다. 이 공격은 가젯 코드를 악용하여 SGX의 격리된 메모리 공간인 Enclave 내에 있는 데이터를 탈취한다. 이로 인해 Enclave 내에 있는 데이터들은 더는 안전하지 않다. 본 논문에서는 Intel SGX와 Spectre 공격에 필요한 Speculative Execution과 캐시 부 채널 공격에 관해 소개하고 Spectre v1 공격을 활용한 예제 코드로 직접 SGX 환경에서 공격을 수행한다.

II. 관련 연구

2.1 Intel SGX

Intel SGX는 x86 명령어 세트 아키텍처를 확장한 것이다. 메모리를 격리하고 메모리를 암호화, 봉인된 저장공간과 원격 증명 등을 제공하여 응용프로그램 보안을 제공하는 Intel 프로세서의 명령어 세트 아키텍처이다[1, 2].

Enclave User Process 내부에 보호된 컨테이너를 생성하고, 응용프로그램의 코드와 데이터를 인스턴스화 한 메모리 공간이다. Enclave 내부의 메모리는 기밀성과 무결성을 보장하여 신뢰할 수 없는 시스템 소프트웨어로부터 보호한다. [그림 1]은 SGX가 User Process 공간에 Enclave라는 공간을 인스턴스하여 암호화한 메모리 공간을 만든 것이다.



[그림 1] User Process 내부에 있는 Enclave

2.2 Speculative Execution

프로세서는 프로그램의 다음 명령의 흐름을 알지 못한다. 예를 들어 선형 명령에 따라 비순차적으로 실행하는 동안 조건문의 참과 거짓을 판별하기 전까지 다음 명령어를 수행하지 못한다. 하지만 Speculative Execution, 추측실행은 현재 명령어를 실행하는 레지스터 상태를 저장한다. 이후 프로그램이 실행할 경로를 예측하고 다음 명령어를 미리 수행할 수 있다. 이때 예측한 경로가 올바른 것으로 확인이 되면 전에 저장했던 레지스터가 필요가 없게 되고 레지스터의 상태에 해당하는 명령어가 폐기된다. 프로세서가 예측한 경로가 올바르지 않다고 판단할 때 레지스터 상태를 다시 로딩함으로써 잘못된 경로를 따라, 보류되었던 모든 명령어를 버리고 올바른 경로를 따라 실행을 다시 시작한다. 이러한 이유로 추측실행은 마치 실행이 올바른 경로로 가는 것처럼 프로그램의 논리적인 상태를 유지한다[3, 4].

2.3 캐시 부 채널 공격

캐시 부 채널 공격은 L3와 같은 공유 캐시메모리 공간의 Cache hit 와 Cache miss의 시간 차이를 이용하여 희생자의 메모리에 있는 데이터를 유추한다[3, 5, 6]. 대표적으로 Prime+Probe, Flush+Reload 공격이 있다. 공격자는 x86의 clflush 명령어를 사용하여 전체 캐시 계층에서 희생자와 공유하고 있는 캐시 라인을 제거한다. 이후 희생자가 공유캐시 라인에 접근할 때 메모리에 있던 데이터가 공유캐시에 적재가 되고 같은 캐시 라인에 접근 시 캐시에서 바로 데이터를 얻을 수 있어 접근 속도가 대폭 향상된다. 반대의 경우, 데이터를 메모리에서 불러와야 하므로 접근시간이 길어진다. 따라서 공격자는 캐시 라인의 접근시간을 측정하여 희생자의 데이터를 유출할 수 있다.

2.4 Spectre v1 공격

Spectre 공격은 앞서 설명한 Speculative Execution과 캐시 부채널 공격을 활용한다[3]. Spectre 공격 중 대표적인 BCB(Bound Check Bypass on Load), 경계 검사 우회를 하는 공격인 Spectre v1에 관해 설명한다.

```
void victim_function(size_t x) {
    if (x < array1_size) {
        temp &= array2[array1[x] * 512];
    }
}
```

[그림 2] BCB 공격 코드 예제

[그림 2]는 추측실행을 통해 분기 문을 거치지 않고 내부의 명령어를 수행하는 코드 예제이다. 악의적인 변수 x 와 array1_size 값을 비교한다. clflush 명령어에 의해 array1_size 값을 읽어 올 때 Cache miss 나게 되어 메인 메모리에서 값을 가져올 때까지 많은 시간이 걸린다. 대기 시간 동안, 조건문을 예측하는 분기 예측기는 if 문이 참이라고 가정하고 다음 명령어를 수행한다. 이때 x는 조건을 넘는 값이 되어 array1[x]*512의 결과값은 다른 영역의 메모리에 침범한다. 이때 메모리에 array1_size 값을 가져와 x값과 비교할 때 맞지 않음으로 레지스터 상태를 처음으로 되감는다. 하지만 프로세서는 이미 연산을 수행하여 캐시의 주소에 array 2 값이 로드된다. 이후 공격자는 캐시 부 채널 공격을 이용하여 반복문을 통해 x에 0~255를 입력하여 캐시에 접근하는 시간을 측정한다. 접근시간이 짧은 경우 해당 캐시 라인에 희생자의 데이터가 적재됐다는 것을 유추할 수 있다.

III. Intel SGX에서의 Spectre 공격

본 앞서 설명한 Spectre 공격을 이용하면 Intel SGX 환경에서도 비밀 값 유출이 가능하다[5]. 먼저 실험을 위해 sgx_create_enclave() 함수를 호출하여 희생자의 Enclave를 생성 및 초기화하고 그 안에 비밀 데이터를 저장한다. 이후 Spectre 공격은 다음과 같은 순서로 진행된다.

1) [그림 3]의 ecall_get_offset() 함수를 통해 희생자의 비밀 데이터 secret을 캐시에 적재한다. ECall은 Enclave 내에 프로그래밍을 위해 인자값으로 받을 수 있는 모든 함수 집합이다.

```
size_t ecall_get_offset(){
    temp = secret[0];
    return (size_t)(secret-(char*)array1);
}
```

[그림 3] ecall_get_offset() 함수

2) readMemoryByte() 함수를 통해 조건문을 반복 수행하여 분기 예측기를 학습한다. [그림 4] ecall_victim_function() 함수를 이용하여 위 희생자의 데이터를 탈취한다. 따라서 변수 x가 조건문에 벗어나는 값이어도 참이라고 판단하고 다음 명령어인 temp &= array2[array1[x] * 512]를 추측 실행한다. x는 outside_array1_size를 초과한 값이기 때문에 array1[x]은 다른 영역의 메모리 값에 침범하게 되고 array2 배열 또한 다른 영역에 값을 저장한다.

```
void ecall_victim_function(size_t x, uint8_t * array2,
                           unsigned int * outside_array1_size){
    if (x < *outside_array1_size){
        temp &= array2[array1[x] * 512];
    }
}
```

[그림 4] ecall_victim_function() 함수

3) 캐시 부 채널 공격을 통해 x값을 입력할 때 메모리 접근시간 차이를 측정한다. 기준이 되는 클럭보다 길게 나오면 Cache miss로 희생자가 불러오지 않은 데이터이고, 짧게 나오면 Cache hit가 되어 희생자가 불러온 데이터를 유추할 수 있다.

해당 공격을 수행하면 악의적인 x값이 추측실행으로 다른 영역의 메모리 공간으로 침범하게 되고 공격자는 캐시 부 채널 공격을 이용하여 문자열을 읽어올 수 있다. 추가적으로, 우리는 예제코드를 이용하여 실제 Intel SGX환경의 Enclave 내부에 있는 비밀 데이터 유출 공격을 수행하였으며 결과는 [그림 5]와 같다[7].



[그림 5] SGX에서 Spectre 공격 실험 결과

Ⅳ. 결론

본 논문에서는 안전하게 데이터를 저장하기 위해 만들어진 Intel SGX와 SGX에서의

Spectre 공격을 소개하고 직접 실험을 수행하였다. 이를 통해 안전한 실행 환경을 위해 만들어진 Intel SGX의 Enclave 내부의 데이터도 부 채널 공격을 사용하여 유출이 가능하다는 것을 확인하였다. 하지만 해당 공격은 로컬에서 진행하였으며 공격자가 같은 프로세서나 다른 프로세서 내의 Enclave를 통제 할 수 있어야 된다는 한계점이 존재한다. 향후 연구로는 부 채널 공격으로부터 Enclave 내에 있는 데이터를 안전하게 보호할 수 있는 기법에 관해 연구하고 실제 공격 환경에서 Spectre 공격을 수행하는 모델링 연구를 진행할 예정이다.

[참고문헌]

[1] Prerit Jain, Soham Desai, Seongmin Kim, Ming-Wei Shih, JaeHyuk Lee, Changho Choi, Youjung Shin, Taesoo Kim, Brent Byunghoon Kang, Dongsu Han, “Open Platform for SGX Research”, The Network and Distributed System Security Symposium 2016, February, 2016.

[2] Victor Costan, Srinivas Devadas, “Intel SGX Explained”, IACR Cryptology ePrint Archive, 2016.

[3] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, Yuval Yarom. "Spectre Attacks: Exploiting speculative execution", arXiv preprint arXiv:1801.01203, January, 2018.

[4] Fay Chang, Garth A. Gibson, “Automatic I/O Hint Generation Through Speculative Execution”, 3rd Symposium on Operating Systems Design and Implementation, February, 1999.

[5] Guoxing Chen, Sanchuan Chen, Yuan Xiao, Yinqian Zhang, Zhiqiang Lin, Ten H. Lai, “SgxPectre Attacks: Stealing Intel Secrets from SGX Enclaves via Speculative Execution”, 2019 IEEE European Symposium on Security and Privacy (EuroS&P), June, 2019.

[6] Yuval Yarom, Katrina Falkner, “FLUSH+RELOAD: a High Resolution, Low Noise, L3 Cache Side-Channel Attack”, 23rd USENIX Security Symposium, August, 2014.

[7] https://github.com/lsds/spectre-attack-sgx, "spectre-attack-sgx", github, January, 2018.

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 F

정보시스템 구축.보안지침 대한 47개 비교분석

류호경* 명성식** 이호준***
고려대학교 컴퓨터정보통신대학원

47 ways to improve information system construction and security guidelines

Ho-Kyung Ryu* sung-sik myung** Hojun Lee***

요 약

4차 산업 혁명시대 속 소프트웨어의 대중화로 인해 우리 사회는 컴퓨터 통신이 없이는 생활할 수 없는 시대에 살고 있으며, 많은 소프트웨어 시스템 등을 통해 일상생활 속에서 유용하게 사용하여 살고 있다. 하지만 다양한 소프트웨어의 증가로 인해 많은 보안 약점이 증가하고 있다. 기업들은 이러한 보안 약점의 위험성에 관하여 중요성을 인식하고 안정적인 프로그램을 개발할 수 있도록 노력하고 있다. 국내의 경우 행정안전부에서 배포한 행정안전부 정보시스템 구축 지침과 참고 자료인 소프트웨어 개발 보안 가이드를 통해 소프트웨어 보안 약점 제거 활동을 의무화하여 시행하고 있다. 본 연구에서는 2019년 새롭게 발표한 CWE Top 25에 대하여 기존에 발표된 보안 약점 목록과 비교 분석한 후 행정안전부의 정보시스템 구축 운영지침을 개선하는 방향을 제안하였다.

I. 서론

4차 산업 혁명시대 중심 속에 있는 현대사회에서 다양한 소프트웨어의 개발과 활성화로 인해 소프트웨어 보안 역시 중요도가 높아지고 있다. 대부분의 생활이 소프트웨어를 사용하고 활용 범위가 넓어진 만큼 새로운 보안 약점들이 드러나고 있다. 4차 산업 혁명의 가속화로 사물인터넷(IoT)·인공지능(AI) 등 신기술을 적용한 소프트웨어 제품이 늘면서 이에 따른 보안사고도 함께 급증하고 있다.

과학기술정보통신부에 따르면 사업체의 침해사고 경험률은 2.3%로 전년 조사 결과(2.2%)와 유사하며, 대부분 침해사고가 경미(69.2%)한 수준으로 발생했으나, 경험 유형 중 랜섬웨어로 인한 피해(56.3%, 30.8%p↑)가 큰 폭으로 증가하였고, 이런 제품들이 인터넷에 연결돼 있어 네트워크를 통한 보안사고 위험이 잠재돼 있음을 예상하고 있다.[1] 정보유출, 자원공유·집중화로 서비스 장애 시 대규모 피해 발생에 대한 대비가 필요할 것으로 보인다. 취약점 관리체계 운영, 랜섬웨어 피해 예방 수칙 이용자의 보안의식 제고 등 정보보호 강화가 필요하다. 그러나, 국내의 경우 소프트웨어의 구축단계에서부터 보안에 대한 관리체계가 구축되어 있지 않아 보안 위협 및 침해사고에 따른 큰 피해가 발생할 수 있다.

본 논문에서는 MITRE에서 제공된 2019 CWE Top 25를 분석하여 행정안전부에서 발표한 정보시스템 구축 운영지침를 개선하는 방향을 제안하였다.

본 논문은 다음과 같이 구성된다. 2장에서는 새롭게 개편된 CVSS 평가 방식을 소개한 후, 기존의 CWE 2011/CWE 2019 Top25 및 OWASP Top10에 대한 비교분석 한다. 3장에서는 행안부에서 제시한 정보시스템 구축 지침과 2019 CWE Top의 정보 활용방안에 대해 비교 분석하고 마지막으로 결론 및 추후 연구 방향에 관해 기술한다.

II. 2019 CWE Top 25 구성

소프트웨어 약점에 대해 표준화되어 분류법으로 지정된 2019 CWE Top 25에 대해 기존의 2011 CWE Top 25와 OWASP Top 10(2017)을 비교 분석한다. 또한 산정된 약점(weakness)의 순위를 계산하기 위한 방식인 신규 취약점의 위험도 스코어링 시스템에 관련한 연구를 살펴 본다.

2.1 CVSS(Common Vulnerability Scoring) 산정된 스코어링 시스템 방법론

기존의 제공된 2011년에는 개발자, 최고 보안 분석가, 연구원 및 공급 업체와의 설문조사 및 개인 인터뷰를 통해 데이터가 모집되어 정보를 제공하고 있다. 이러한 방법론은 먼저 각 전문가의 평가에 따라 중요하거나 가장 넓게 확산해 있다고 판단되는 취약점을 설문을 통해서 선정한다. 그다음 CVSS 방법론에 따라 순위가 결정될 수 있도록 정규화되어 리스트들이 분류된다. 이런 방법을 통해 측정할 경우 설문조사에 투자되는 노동력과 선택되는 기준에 대한 객관성에 대한 의구심을 가질 수 있다. 따라서 2019 CWE TOP 25에서의 목록은 실제 일상생활에서 발생 되는 취약점을 기반으로 NVD의 데이터에서 분류하여 구성되어 있다. 이에 설문조사 및 의견에 따른 취약점 위험도 측정방식을 객관적이고 통계학적인 접근법 점수를 제공하는 CVSS 점수를 참고하여 순위를 측정한다. 공식은 CWE가 취약점의 근본 원인인 빈도와 악용의 예상 심각도를 결합한다. 두 경우 모두 빈도와 심각도는 최소값과 최대값을 기준으로 정규화하여 산정하였다.[2]

Freq = {count(CWE_X' ∈ NVD) for each CWE_X' in NVD}

Fr(CWE_X) = (count(CWE_X ∈ NVD) - min(Freq)) / (max(Freq) - min(Freq)) (1)

CWE의 빈도를 결정하기 위한 공식은 (1)과 같이 NVD 내에서 CVE에 매핑 된 횟수를 계산하였다. 또한 NVD 내에서 전체 CVE 세트를 사용하면 주파수 비율이 매우 낮아지고 서로 다른 취약점 유형 간에 차이가 거의 없기 때문에 관련 취약점이 있는 CVE 만이 계산에 사용되었다.

Sv (CWE_X) =(average_CVSS_for_CWE_X - min(CVSS)) / (max(CVSS) - min(CVSS)) (2)

취약점의 심각도는 (2)와 같이 특정 CWE에 매핑되는 모든 CVE의 평균 CVSS 점수로 구성하였다.

Score(CWE_X) = Fr(CWE_X) * Sv(CWE_X) * 100 (3)

특정 CWE에 의해 제시된 위험 수준은 (3)와 같이 심각도 점수에 빈도 점수를 곱하여 가중치를 부여하였다.

추가적으로 점수 산정 방법에서 거의 악용되지 않는 취약점과 관련해서는 심각성과 관계없이 높은 점수를 부여하지 않았다. 또한 영향도에 따라 비례적으로 점수를 할당한다.[2]

2.2 2019/2011 CWE Top 25 비교 분석 및 주요 보안 약점 목록

본 논문에서 수집된 2019 CWE Top 25은 최근 보고되지 않은 대상 리스트가 아닌 기존부터 누적되어 알려진 대상 취약점 리스트들이 대부분이다. 2011년과 다른 점은 산정 방식인 스코어링 시스템을 통계 방식을 접근하였기에 더욱 객관적인 점수 체계로 평가되었다. 마이터의 소프트웨어 보장 부문 수석인 드류 버트너(Drew Buttner)는 "실제 현장에서 벌어지는 일들을 최대한 있는 그대로 반영한 목록을 만들고 싶었다"고 설명한다.[3] 또한 2019 CWE에서부터 상위권에 차지한 리스트들은 변동이 미비하다.

[그림 1]은 산출된 24개 항목의 순위와 중요도 점수를 보여준다.

[표1] 2019 CWE Top 25 신규 리스트

순위	CWE -ID	보안 약점 이름	점수
1	CWE-119	메모리 버퍼의 경계 내에서 부적절한 동작 제한	75.56
2	CWE-79	웹 페이지 생성 중 입력의 부적절한 중립화 ('cross-site scripting')	45.69
3	CWE-20	부적절한 입력 검증	43.61
4	CWE-200	정보 노출	32.12
5	CWE-125	범위를 벗어난 읽기	26.53
6	CWE-89	SQL 명령에 사용 된 특수 요소의 부적절한 중립화 ('SQL Injection')	24.54
7	CWE-416	무료 후 사용	17.94
8	CWE-190	정수 오버플로 또는 랩 어라운드	17.35
9	CWE-352	CSRF (Cross-Site Request Forgery)	15.54
10	CWE-22	제한된 디렉토리에 대한 경로 이름의 부적절한 제한 ('경로 탐색')	14.1
11	CWE-78	OS 명령 ('OS 명령 주입')에 사용 된 특수 요소의 부적절한 중화	11.47
12	CWE-787	범위를 벗어난 쓰기	11.08
13	CWE-287	부적절한 인증	10.78
14	CWE-476	NULL 포인터 역 참조	9.74
15	CWE-732	중요한 리소스에 대한 잘못된 권한 할당	6.33
16	CWE-434	위험한 유형의 파일 무제한 업로드	5.5
17	CWE-611	XML 외부 엔터티 참조의 부적절한 제한	5.48
18	CWE-94	부적절한 코드 생성 제어 ('코드 삽입')	5.36
19	CWE-798	하드 코딩 된 자격 증명 사용	5.12
20	CWE-400	통제되지 않은 자원 소비	5.04
21	CWE-772	유효 수명 후 자원의 누락	5.04
22	CWE-426	신뢰할 수없는 검색 경로	4.4
23	CWE-502	신뢰할 수없는 데이터의 역 직렬화	4.3
24	CWE-269	부적절한 권한 관리	4.23
25	CWE-295	부적절한 인증서 검증	4.06

상위 3개의 기존에 발표된 약점이며 이미 언급이 많이 된 내용이다.

• CWE-119 : 버퍼오버플로우(buffer overflow)관련이다. 메모리 버퍼의 한계 내에서 이뤄지는 운영에 제대로 된 제한을 걸지 못해서 생기는 오류로 구성되어 있다.

• CWE-79 : (Cross-site Scripting)이다. 웹 페이지 생성 시 입력 값을 제대로 확인하지 못해서 생기는 오류로 산정되었다.

• CWE-20 : 부적절한 입력 검증이다. 적절한 입력값을 제대로 확인하지 않아서 생기는 문제다.

전반적으로 2011년 대비하여 메모리 버퍼에 대한 임계치 검증 부재로 인한 버퍼 오버플로우 공격과 입력값 검증 부재로 인해 발생하는 XSS등 관련한 공격들이 상위권에 분류되었다. 이는 이전의 분류법들을 참조한 것으로 이전과 특별한 점이 없다는 점을 강조한다. OWASP Top 10은 2010년부터 웹 관련 위험에 대해 주로 제공하고 있으며, 2003년 처음으로 10개 항목으로 이루어진 목록이 발표되었다. OWASP Top 10- 2017과 비교하였을 때 버퍼 오버플로우 관련 약점 이외에는 웹 관련 위험요소를 중점적으로 다루고 있다.[4]

III. 국내 보안 약점 가이드 구성 및 CWE 비교 분석

3.1. 행정안전부 정보시스템 구축 운영지침

소프트웨어의 개발 보안 의무화는 중요한 역할을 담당하고 있다. 국내의 경우 행정안전부에서 배포한, 정보시스템 구축 운영 지침은 실무에서 보안 부분의 세부적인 방향을 가이드하기 위한 중요한 기준이 된다. 보안설계의 약점 기준은 7개의 보안 약점 항목으로 구성되어 있으며 공공 정보시템의 특성을 반영하여 웹 기반의 서비스에 중점을 둔 보안 약점 목록이다.[5] 표[2]는 47개 보안 약점의 구성을 보여주고 있다.[6]

[표2] 정보시스템 구축 운영 지침 및 CWE 2019비교

유형	주요 취약점 내용	영역별 보안약점 번호
입력 데이터 검증 및 표현	SQL 삽입	CWE-89
	경로 조작 및 자원 삽입	CWE-22
	크로스사이트 스크립트	CWE-79
	운영체제 명령어 삽입	CWE-78
	위험한 형식 파일 업로드	CWE-434
	신뢰되지 않는 URL 주소로 자동접속 연결	CWE-426
	XQuery 삽입	
	XPath 삽입	
	LDAP 삽입	
	크로스사이트 요청 위조	CWE-352
	HTTP 응답분할	
	정수형 오버플로우	CWE-190
	보안기능 결정에 사용 되는 부적절한 입력값	CWE-94 CWE-20
	메모리 버퍼 오버플로우	CWE-119 CWE-125 CWE-787
	포맷 스트링 삽입	
보안기능	적절한 인증 없는 중요기능 허용	CWE-287
	부적절한 인가	
	중요한 자원에 대한 잘못된 권한 설정	CWE-732
	취약한 암호화 알고리즘 사용	
	중요정보 평문저장	
	중요정보 평문전송	
	하드코드된 비밀번호	CWE-798
	충분하지 않은 키 길이 사용	
	적절하지 않은 난수 값 사용	
	하드코드된 암호화 키	
	취약한 비밀번호 허용	
	사용자 하드디스크에 저장되는 쿠키를 통한 정보 노출	
	주석문 안에 포함된 시스템 주요정보	
	솔트 없이 일방향 해쉬함수 사용	
	무결성 검사없는 코드 다운로드	
	반복된 인증시도 제한 기능 부재	
시간 및 상태	경쟁조건: 검사 시점과 사용 시점(TOCTOU)	
	종료되지 않는 반복문 또는 재귀 함수	
에러처리	오류 메시지를 통한 정보 노출	
	오류 상황 대응 부재	
	부적절한 예외 처리	CWE-269
코드오류	Null Pointer 역참조	CWE-476
	부적절한 자원 해제	CWE-400 CWE-772
	해제된 자원 사용	CWE-416
	초기화되지 않은 변수 사용	
캡슐화	잘못된 세션에 의한 데이터 정보 노출	
	제거되지 않고 남은 디버그 코드	
	시스템 데이터 정보노출	CWE-200
	Public 메소드부터 반환된 Private 배열	
	Private 배열에 Public 데이터 할당	
API 오용	DNS lookup에 의존한 보안결정	
	취약한 API 사용	

3.2. 소프트웨어 개발보안 가이드

행정자치부는 '행정기관 및 공공기관 정보시스템 구축·운영 지침'의 소프트웨어 개발 보안 부분의 세부적인 부분을 가이드하기 위해 코딩 관점으로 구성된 소프트웨어 개발 보안 가이드를 배포하였다. 분야별 보안상의 취약점을 기준으로 각 보안 약점에 대해 간단한 설명과 같이 보안 문제점 목록과 보안 대책과, 코딩 예시, 용어정리로 구성되어 있다. 참고 자료로는 CWE, OWASP 등 관련 링크로 이루어져 있다.[7] 그러나 운영체제나 그 외의 환경설정으로 인해 바로 적용하기에 어려움을 느낄 수도 있다.

3.3 정보시스템 구축 지침과 2019 CWE 비교

본 연구에서 선정한 정보시스템 구축 운영지침은 2019년 발표된 CWE Top 25 리스트와 비교하였을 때 [표]2와 같이 전반적으로 포괄하고 있다. 위험도 산정 방식에 따른 CWE의 보안약점 분류 방식에 비추어 볼 때 '입력 데이터 검증 및 표현영역의 부분의 메모리 버퍼오버플로우에 관한 운영지침은 하나만 존재하는 반면, CWE 보안 약점 기준에는 CWE-119(Improper Restriction of Operations within the Bounds of a Memory Buffer) /CWE-125(Out-of-bounds Read) /CWE-190(Integer Overflow or Wraparound) /CWE-787(Out-of-bounds Write)약점 등 항목들의 개수가 다양하고 약점 간 중복성이 포함되어 상위 등급에 분류되어 있다. 또한 소프트웨어 약점에 대한 보안사고가 증가함에 따라 CWE-611 (Improper Restriction of XML External Entity Reference) /CWE-502 (Deserialization of Untrusted Data) /CWE-295 (Improper Certificate Validation)에 대한 약점들이 강조되어 분류되었다. 간단한 예시로 CWE-611 경우 XML 외부 엔터티 참조의 부적절한 제한으로 인해 XML의 파일업로드 과정에서 <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>와 같이 공격자가 서버에서 데이터를 가져오려는 행위를 할 때 노출될 수 있다는 점을 보여주고 있다. 해당 약점은 2017 OWSAP Top-10의 XML외부개체(XXE)에 새롭게 추가되어 분류되고 있다. 또한 데이터를 역 직렬화하는 과정에서 발생하는 CWE-502 같은 경우 상대적으로 공격자가 코드를 수정하기에 공격 시도는 어려울 수 있다. 그러나 공격자가 애플리케이션 로직을 수정할 수 있거나 사용 가능한 클래스가 존재할 경우 임의의 원격 코드로 실행하여 악용할 수 있다. 해당 약점은 2017 OWSAP Top-10의 안전하지 않은 역 직렬화에 분류되고 있다.[4] 보고된 약점에 대해서는 정보시스템 구축 지침에 세분화되어 반영될 필요가 있다.

이는 전체적으로 정보시스템 구축 지침이 기존 CWE의 목록과 비슷한 양상을 확인 할 수 있다. 즉 기존 발표된 소프트웨어 약점들이 향후 몇 년간 상위권 리스트에 지속해서 남아 있을 가능성을 연구 결과를 통해 알 수 있다.

IV. 데이터 중심 방법론의 한계

본 연구에서 분석한 MITRE의 2019 CWE (Common

Weakness Enumeration) Top 25 가장 위험한 소프트웨어 약점 목록을 발표된 리스트들은 한계점을 지니고 있다. 이번 산정 기준은 NVD에서 공개적으로 보고되고 취합된 데이터 이외에 CVE ID가 없는 수 많은 보안 약점을 확인할 수 있다. 따라서 NVD에 포함되지 않은 취약점에 관련된 목록들은 CVSS 분류 방식에 점수가 산정되지 않아 약점 리스트에서 제외되어 분류되고 있다. 또한, 취약점을 보고하는 공급 업체에서 사용하는 언어로 인해 CVE / NVD 데이터는 특정 영역에 평준되어 있다. 추가적으로 기존에 연구원과 정적 도구가 찾아내던 유형의 취약점에 관하여도 산정 방식 변경으로 인해 반영되는데 한계점이 있다.[4]

V. 결론

본 논문에서는 2019 CWE (Common Weakness Enumeration) Top 25를 주제로 새롭게 발표된 구성을 소개하였다. 또한 CWE의 새로운 스코어링 산정 방식에 대해 분석한 후 국내 보안 기준이 되는 정보시스템 구축 운영지침에 관하여 비교 분석 및 개선 방향을 제안하였다. 새롭게 발표된 CWE 표준목록과의 분석을 통해 전반적으로 구축 운영지침에 관하여 변경될 점은 없었으나, 익스플로잇 보안 사고 증가에 따라 강조되어야 할 보안 약점 및 방향을 도출하였다. 향후 연구로는 본 논문에서 자세하게 다루지 못한 CWE 한계점에 관하여 사용 프로그래밍 언어별로 구체화하여 주요 문제점과 해결책을 제시하고, CWSS에 대한 산정방식 값에 대한 결합과 매년 발생하는 빈도수가 높은 취약점 목록에 접목해 CWE의 약점 목록의 객관화 지표 분류에 향상할 추가적인 연구가 필요하다.

참고문헌

[1] 과학기술정보통신부, [과기정통부,2018년 정보보호 실태조사 결과발표] ,2019

[2] CWE, https://cwe.mitre.org/

[3] MITRE Corporation “마이터 코퍼레이션, 가장 위험한 취약점 25개 선정”,보안뉴스. 2019

[4] OWSAP Top 10, https://www.owasp.org/

[5] 박양환, 김민경 “전자정부 소프트웨어의 보안성 강화를 위한 개발보안 제도 연구”정보보호학회논문지,26(1), pp. 20~26

[6] 행정안전부, 정보시스템 구축 운영 지침(행정안전부고시 제2019-69호), 행정안전부, 2019. 8

[7] 행정자치부,소프트웨어개발보안가이드,pp.301~304,2017.01

ICT 공급망 보안관리 통제 개발을 위한 국내외 관련 기준 비교분석

고영현*, 김정덕**

*중앙대학교 융합보안학과, 중앙대학교 산업보안학과**

A Study on Comparative Analysis of the ICT Supply Chain Security Management Controls

Young Hyun Ko*, Jungduk Kim**

*Dept. of Convergence Security, Graduate School of Chung-Ang University.

**Dept. Industrial Security, Chung-Ang University.

요 약

4차 산업혁명에 따라 융합 ICT 서비스가 현실화되고 있으며, 산업, 행정, 국방 등 국가 전반에서 공급자와 제3자로부터의 ICT 공급망 보안 위험이 증가하고 있다. 본 연구에서는 해외기준 분석을 통해 ICT 공급망 보안 위험관리를 위한 통제항목의 기반이 될 지표를 고안하였으며 국내에서 운용 중인 보안관리체계 ISMS-P와 비교 분석하여 차이점을 도출하였다. ISMS-P의 보호대책 요구사항과 매핑한 결과, ISMS-P는 ICT 공급망의 개발, 공급 및 유지보수, 관리 단계 전반적으로 ICT 공급망 위험관리에 미흡하며 개선되어야 함이 도출되었다. 향후 ICT 공급망 보안 관리체계 또는 평가체계 마련 시 이를 활용하여 국내 디지털 비즈니스의 신뢰성과 안정성을 높이고 ICT 산업 활성화에 기여할 수 있을 것으로 기대된다.

I. 서론

4차 산업혁명이 도래하면서 스마트 팩토리, 사물인터넷 등 융합 ICT 서비스가 현실화되고 있다. 산업, 행정, 국방 등 국가 전반에서 디지털화가 확산됨에 따라 개인, 기업, 정부 등 모든 주체의 활동 기반이 사이버공간으로 확대되고 있다. 따라서, 안전성이 확보되지 않은 채 도입되는 ICT 제품 및 서비스는 국민과 국가 운영 전반의 신뢰성 확보 및 디지털 비즈니스 성장에 큰 장애물로 작용한다[1].

미국 국립표준기술연구소에 따르면 ICT 기술은 복잡하고, 아웃소싱의 여러 계층으로 구성된 전 세계적으로 분산된, 상호 연결된 공급망 생태계에 의존한다[2]. 따라서 ICT 기술을 이용한 제품 및 서비스를 공급 및 이용하는 기업들은 ICT 공급망에 대한 의존도와 위험이 상승하게 되었다. IBM에 따르면 ICT 공급망에 침투하여 최종 사용자에게 전달되는 소프트웨어나 하드웨어를 변조하는 ICT 공급망 공격은 지속적으로 증가하고 있으며 2018년에는 전년도 대비 78% 증가하였다[3]. 증가하는 ICT 공급망 위험에도 불구하고, Marsh와 Microsoft의 조사 결과에 따르면, 조직의 43%가 제3자로부터의 공급망 위협에 대응할 능력이 전혀 없다고 답하였다[4].

과학기술정보통신부는 2019년 1월 8일 민간부문 정보보호 종합계획을 발표하였으며 주요정보통신기반시설의 잠재적 위협에 대한 선제적 대응을 위한 ICT 공급망의 보안관리 강화를 강조하였다[5]. 특히, 미국 등 주요 선진국의 공급망 위험관리 방법을 토대로 한국형 공급망 보안관리 체계(방법론, 절차, 보안점검 리스트 등) 마련을 촉구했다. 청와대 국가안보실은 2019년 4월 3일 국가 사이버 안보 전략을 발표하였으며, 주요정보통신기반시설에 공급되는 주요 ICT 장비의 공급망 보안을 위한 관리체계 구축 및 관계기관 이행점검을 위한 제도 기반 마련을 촉구하였다[6].

II. 관련 연구

2.1 ICT 공급망의 개념 및 특성

ICT 공급망 보안 국제표준인 ISO/IEC 27036-1은 ICT 공급망을 ICT 제품 및 서비스의 설계, 개발, 생산, 전달의 프로세스가 연속적으로 일어나는 흐름에서 공급망 상의 이해관계자들이 계약과 같은 일련의 방법을 통해 관계를 형성하는 과정으로 정의하였다[7]. ISO/IEC 27036-2는 ICT 공급망 프로세스를 크게 프로젝트 프로세스와 프로젝트 계약 프로세스로 구분하였으며, 프로젝트 프로세스는 세부 15단계로 나뉘며, 계약 프로젝트 프로세스는 세부 5단계로 나뉜다. 세부 내용은 아래 표1과 같다[8].

표 1 - ICT 공급망 프로세스

ICT 공급망 프로세스		
프로젝트 프로세스	합의	획득
		공급
	조직적 프로젝트 지원	라이프사이클 모델관리
		기반시설 관리
		포트폴리오 관리
		인적자원 관리
		품질 관리
	프로젝트	프로젝트 계획
		프로젝트 평가 및 통제
		의사결정 관리
		위험관리
		구성 관리
		정보 관리
		측정
	기술적	아키텍쳐 구성
프로젝트 계약 프로세스	공급자 관계	관계 계획
		공급자 선택
		관계 합의
		관계 관리
		관계 종료

ISO/IEC 27036-3은 ICT 공급망의 유형을 5가지로 정의하였으며, 그 유형은 아래와 같다[9]:

- ICT 제품의 설계, 개발, 납품, 구축

- 상용 소프트웨어(COTS: Commercial-off-the-shelf) 개발

- 오픈소스 제품 개발 및 유통

- 획득자의 정보 및 시스템을 공급자가 관리 및 운영(예:아웃소싱)

- 공급자의 시스템 및 자원을 관리기관이 이용(예: 클라우드 컴퓨팅)

MITRE가 41가지 ICT 공급망 공격을 분석한 결과, 전체 공격의 2/3 이상이 엔지니어링 및 제조 개발 단계를 대상으로 했다. 또한, 절반 이상의 공격이 프로그램 오피스 및 주 계약자를 대상으로 했으며, 하도급 업자를 대상으로 성공한 공격 중 대부분은 주계약자와 통합업체에도 연쇄적으로 공격이 성공한 것을 확인할 수 있다[10].

2.2 ICT 공급망 보안 관련 해외기준 분석

미국 국립기술표준원은 2017년도 트럼프 대통령의 행정명령을 반영하여, 2018년 4월 16일 'Cybersecurity Framework 1.1'을 발표하였다. 이는 기존의 1.0버전을 개선한 것이며, 사이버 위험관리 프레임워크 항목에 공급망 위험관리를 추가하였고, 5가지 세부항목은 아래와 같다[11]:

-ICT 공급망 관리 프로세스가 식별, 설계, 평가, 관리되고 있는가?

-정보시스템 및 서비스의 공급자와 제3자 협력사들이 식별, 우선 순위화되었으며, ICT 공급망 위험평가 프로세스에 의해 평가되었는가?

-공급자와 제3자와의 계약이 조직의 사이버보안 프로그램 및 사이버공급망 위험관리 계획 및 목적에 부합하도록 수행되었는가?

-공급자와 제3자 협력사를 정기적으로 감사 및 평가를 하는가?

-사고 대응 및 복구 계획 및 테스트가 공급자와 제3자 협력사와 같이 수행되는가?

사이버 위험관리 프레임워크는 인식, 보호, 탐지, 대응, 복구의 5가지 단계로 구성되어 있으며, 그 중 인식 단계에는 공급망 위험관리 항목이 포함된다. 주요 활동은 아래와 같다:

- 공급업체에 대한 사이버보안 요구사항 결정

- 사이버보안 요구사항을 계약 단계에 적용

- 공급자에게 사이버보안 요구사항의 검증 방법 전달

- 사이버보안 요구사항이 다양한 평가 방법론을 통해 충족되는지 검증

사이버보안 프레임워크를 활용한 사이버보안 위험관리 자가평가는 크게 5가지 항목으로 구성되어 있고, 그 내용은 아래와 같다:

- 사이버보안 위험관리 프로세스를 조직 이해당사자들이 시별, 수립, 평가, 관리 및 동의하는가?

- 정보시스템 구성요소, 서비스 공급업체 및 제3자가 사이버보안 위험관리 평가 프로세스를 사용하여 식별되고, 우선순위가 지정되며 평가되는가?

- 협력사와의 계약이 조직의 사이버보안 목표를 달성하기 위해 고안된 조치를 반영하는가?

- 공급자와 제3자가 계약상 의무를 준수하는지 주기적으로 감사 및 평가하는가?

- 사고 발생 시 공급업체 및 제3자가 포함된 대응계획이 구축되어 있는가?

화웨이는 공급망의 사이버보안 수준에 영향을 줄 수 있는 제품 및 서비스 공급자들에 대해 사이버보안 시스템 적격성 평가 표준을 운영하고 있으며, 감사를 통해 평가하여 준수율 70% 이하의 경우 Fail(High risk)로 판정한다. 화웨이가 발표한 백서에 따르면 10개 영역, 49개 항목으로 구성되어 있다[12]. 특히, 계약 보안, 제3자 SW 보안, 딜리버리 보안 및 추적 가능성에 대한 보안에 중점을 두고 있다.

III. 국내외 관련 기준 비교분석

ICT 공급망 보안 관련 해외기준 및 표준을 분석한 결과 공급자, 즉 주계약자에 대한 통제를 통해 제3자

(협력사 및 하도급 업체)로부터의 위험을 관리할 필요성이 도출되었다. 또한, 이를 위해서는 ICT 제품 및 서비스의 디자인부터 폐기까지 이르는 기술 개발과 공급 프로세스 전체를 고려할 필요가 있다. 따라서, 관련 해외 표준 및 보고서를 바탕으로, ICT 공급망 위험관리를 위한 보안 통제항목의 기반이 될 지표를 고안하였다. ICT 공급망 라이프사이클을 개발, 공급 및 유지보수, 관리 3단계로 구분 짓고, 11가지의 영역으로 세분화하여 35가지의 지표를 도출하였다. 이 지표를 국내에서 운용되는 보안관리체계인 ISMS-P의 보호대책 요구사항과 매핑하여 차이점을 도출하였다, 그 결과는 아래 그림1과 같다.

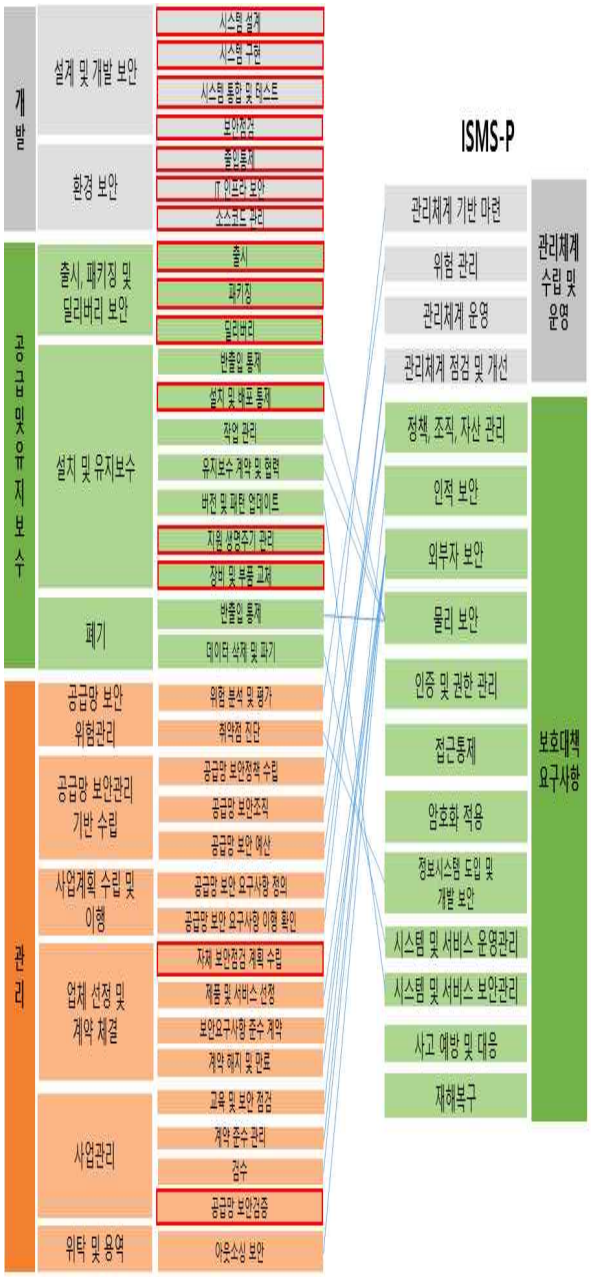


그림 1 - ICT 공급망 보안 지표와 ISMS-P 보호대책 요구사항 매핑

비교분석 결과, 첫째, ISMS-P에서는 ICT 공급망의 개발 단계를 고려하고 있지 않은 것으로 나타났다[13]. 시스템의 설계, 구현, 시험 단계에서 정보보호를 고려한 사전점검을 시행하는 정보보호 사전점검 제도가 있지만, 해당 제도가 ISMS-P와 연계되지는 않는 것으로 나타났다[14]. 둘째, ICT 제품 및 서비스의 공급 및 유지보수 단계에 대한 보호대책이 미흡한 것으로 드러났다. 특히, ICT 제품 및 서비스의 출시, 패키징, 딜리버리, 지원 생명주기 관리, 장비 및 부품교체 시 발생할 수 있는 보안 위험을 고려하지 않고 있는 것으로 나타났다[13]. 셋째, 관리체계 수립 및 운영 부분은 ICT 공급망 위험을 고려하여 개선할 필요가 있다. 공급망 보안 위험을 고려하여 정책 및 조직 및 사업계획을 수립해야 한다. 이를 점검할 수 있는 자체 공급망 보안점검 계획을 수립해야 하며 공급망 보안검증을 주기적으로 실시하도록 통제해야 한다. 넷째, 업체선정 및 계약 체결, 사업관리, 위탁 및 용역의 통제항목이 ISMS-P의 외부자 보안과 매핑되지만 더욱 구체화 및 세분화할 필요가 있다. ISMS-P의 외부자 보안 분야의 세부 점검항목을 보면 외부자 현황을 식별하고 있는가? 또는 정보보호 역량을 고려하도록 외부 서비스 위탁업체를 선정하는가? 등으로 서술되어 있어 다소 추상적이고 구체적이지 못한 면이 있다[13].

IV. 시사점 및 향후 연구과제

4.1 시사점

본 논문에서는 ICT 공급망 보안통제 항목 개발을 위해 ICT 공급망 보안 관련 해외 기준을 분석하였다. ICT 공급망의 특성상 엔지니어링 및 제조 개발 단계에서 공급업체와 이에 걸친 제3자 (주계약자 및 하도급 업체)를 중심으로 공격이 발생하고 있다. 미국은 연방 기관들을 중심으로 공급자와 그에 연계된 제3자 및 협력사에 대한 위험관리에 집중하고 있다. Huawei는 공급업체 및 제3자의 사이버보안 시스템 적격성을 평가하는 자체적인 공급자 평가체계를 운영하고 있다. 관련 해외 기준을 바탕으로 ICT 공급망 위험관리를 위한 보안 관련 지표를 고안하였으며, 국내에서 운용되는 보안관리체계인 ISMS-P와 비교분석하였다. 그 결과 ISMS-P는 ICT 공급망의 개발, 공급 및 유지보수, 관리 단계 전반적으로 ICT 공급망 위험관리를 위해 개선되어야 함이 도출되었다.

4.2 향후 연구과제 및 한계점

ICT 공급망 위험관리를 위해서는 ICT 공급망 통제항목이 개발되어야 한다. 이를 위해서 첫째, 기존의 ISMS-P의 보호대책 요구사항을 개선하거나, 또는 새로운 ICT 공급망 보안에 특화된 보안관리 체계 또는 평가체계를 마련할 필요가 있다. 둘째, 해당 체계를 강제화할 법의 제도화가 필요하다. ISO27001과 같이 조직의 자체 평가로 시작하여 점차적으로 강제화하는 방안도 고려할 수 있다. 셋째, 고안된 ICT 공급망 보안 지표를 기준으로 통제항목과 세부 점검항목을 개발하여 이의 정확성 및 실효성을 검증하는 연구가 필요하다.

본 논문의 한계점은 다음과 같다. 첫째, 본 연구는 일부 기관의 분석 및 동향 보고서를 대상으로 진행되었으므로, 더 정교한 ICT 공급망 통제항목 개발을 위해 더 많은 해외 관련 기준을 분석할 필요가 있다. 둘째, 본 연구는 고안된 ICT 공급망 통제항목을 ISMS-P와 비교 분석하였다. ISO 27001과 같은 다른 보안관리체계와도 비교 분석할 필요가 있다.

[참고문헌]

[1] 국가 사이버 안보 기본계획, 2019년 9월.
[2] NIST, Supply Chain Risk Management Practices for Federal Information Systems and Organizations, April 2015.
[3] IBM, Internet Security Threat Report, Volume. 24, February 2019.
[4] Marsh-Microsoft, 2019 Global Risk Perception Survey, September, 2019.
[5] 과학기술정보통신부, 민간부문 정보보호 종합계획, 2019년 1월.
[6] 청와대 국가안보실, 국가 사이버 안보 전략, 2019년 4월.
[7] 진철구, 김건우, 김정덕, "ICT 공급망내 이해관계자간 협업을 위한 핵심성공요인에 관한 연구", 한국경영정보학회 추계학술대회발표집, pp.16-20, 2016년 11월.
[8] ISO/IEC, "ISO/IEC 27036-2, Information technology-Security techniques-Information security in supplier relationships-Part2: Requirements", March, 2014.
[9] ISO/IEC, "ISO/IEC 27036-3, Information technology-Security techniques-Information security in supplier relationships-Part3: Guidelines for ICT Supply Chain Security", November, 2013.
[10] MITRE, Supply Chain Attack Framework and Attack Patterns, December 2013.
[11] NIST, Framework for Improving Critical Infrastructure Cybersecurity, April, 2018.
[12] Huawei, The Global Cyber Security Challenge, June, 2016.
[13] 한국인터넷진흥원, 정보보호 및 개인정보보호 관리체계 인증제도 안내서-인증기준편, 2019년 1월.
[14] 한국인터넷진흥원, 정보보호 사전점검 안내서, 2017년 11월.

IT 공급망 해킹 동향 분석

김광준* 손효현* 이만희*
*한남대학교 컴퓨터공학과

Analysis of IT Supply Chain Hacking Trends

Kwang-jun Kim*, Hyo-Hyun Son*, Man-Hee Lee*
*Department of Computer Engineering, Hannam University.

요 약

2019년, 글로벌 매거진인 블룸버그 언론사는 중국 화웨이 통신장비에 백도어가 발견되었다고 보도했다. 그 이후 미국은 정보통신 기술과 서비스 보호를 위한 국가비상사태를 선포하며 중국과의 무역 전쟁을 시작하였다. 이 사건들을 통해 전 세계적으로 공급망 보안의 중요성이 부각되었으나, 아직 뚜렷한 대책은 부재인 상태이다. 또한, 2019년 사이버 전망 중 하나로 소프트웨어 공급망 대상 사이버 공격 증가가 선정된 만큼, 지속적으로 국내외에서 발생한 IT 공급망 사고 사례에 대한 기술적인 요소 식별 및 분석이 필요한 실정이다. 그에 따라 본 논문에서는 전 세계적으로 발생한 IT 공급망 해킹 이슈 및 사고 사례들을 분석하고, 공격 특징별로 나열하여 소개한다.

I. 서론

2019년, 미국과 중국의 대규모 무역협상 중단 사건이 발생했다. 이 배경에는 중국 통신장비 제조사인 화웨이 업체가 미국의 정보통신 기술을 빼돌린다는 혐의에 있다. 이 사건 이후 미국 도널드 트럼프 대통령은 미국 정보 통신 기술과 서비스 보호를 위한 국가비상사태를 선포하였다.

그뿐만 아니라 같은 해 4월, 미국 글로벌 비즈니스 매거진인 블룸버그 언론사는 보다폰에서 운영 중인 화웨이 통신장비에 백도어가 발견되었다고 보도했다. 보다폰은 영국의 다국적 이동 통신 업체로서 2009년과 2011년 보다폰 보안 브리핑 문서에 백도어 발견 사실이 기록되어 있는 것으로 확인되었다. 백도어가 발견된 장비는 이탈리아의 유선 네트워크를 연결한 라우터 장비로써, 연결된 단말기에 무단 액세스할 수 있는 숨겨진 백도어가 포함된 장비를 기업과 가정에 제공한 것으로 확인되었다.

이와 같은 사건들은 전 세계적으로 공급망 보안의 경각심을 일깨워주는 계기가 되었다. 기업 자체의 보안수준이 향상됨에 따라 공격자는 기업에 직접인 해킹이 어려워지자 우회경로를 찾으면서 공급망 위협이 발생하고 있다. 즉, 공격대상 기관의 높은 보안시스템을 해킹하기보다 납품업체 및 하청업체와 같은 상대적으로 보안이 허술한 공급망의 약점을 이용하기 시작하였다.

이처럼 전 세계적으로 공급망 위협이 증가함에 따라, 2018년 KISA가 발행한 사이버 위협 동향 보고서에는 공급망을 이용한 사이버 공격이 꾸준히 높아지고 있어 그 대책이 필요함을 강조하였다[1]. 이후 2019년에 발행한 보고서에는 7대 사이버 전망 중 하나로 소프트웨어 공급망 대상 사이버 공격 증가를 선정하였다[2]. 또한, 미국의 보안 업체 중 하나인 Symantec에서 발행한 ISTR(Internet Security Threat Report)에는 2018년 공급망 공격은 전년 대비

78%가 증가하였으며, 공급망 공격이 꾸준히 증가하고 있음을 강조하였다. 이처럼 IT 공급망 공격과 피해는 국내외 모두 그 위험성 및 중요성이 주목받고 있는 현황이다[3].

따라서 지속적으로 국내외에서 발생한 IT 공급망 해킹 이슈 및 사고 사례를 모니터링하며 해킹에 대한 기술적인 요소 식별 및 분석이 필요하다. 본 논문에서는 국내외에서 발생한 IT 공급망 해킹 사건들을 분석하고, 공격 특징별로 나열하여 소개한다.

본 논문은 다음과 같이 구성된다. 2장에서는 IT 공급망 해킹 이슈 및 사고 사례를 특징에 따라 나누어 기술하고, 3장에서는 지속적인 IT 공급망 동향에 대한 모니터링의 필요성과 공급망 사고의 위험성을 강조하며 결론을 맺는다.

II. 공급망 공격 사례

공급망이란 제품이나 서비스가 공급자로부터 소비자에게 전달되는 과정 중 사람, 정부, 자원, 조직 등에 대한 전반적인 시스템을 의미한다[1]. 그중 공급망 공격이란 공급망에 침투하여 사용자에게 전달되는 소프트웨어나 하드웨어를 변조하는 형태의 공격을 의미한다. 이 장에서는 최근 발생한 공급망 공격 사례를 공격 특징에 따라 개발환경 침투와 업데이트 서버 침투로 나누어 소개한다.

2.1 개발환경 침투

개발환경 침투란 공격자가 개발자의 소프트웨어 개발환경에 접근하여 공격을 시도하는 방법을 말한다. Fig. 1과 같이 개발 또는 관리자 PC에 침투하거나, 제품빌드 서버에 침투하여 정상 프로그램에 악성코드를 삽입한다. 악성코드에 감염된 제품빌드 서버를 통해 빌드된 제품은 악성코드가 삽입된 채로 배포 서버로 업로드 된다. 이후 악성코드가 포함된 제품을 설치한 이용자는 악성코드에 감염되고, 감염된 PC 정보를 공격자의 C&C서버로 전송하게 된다.

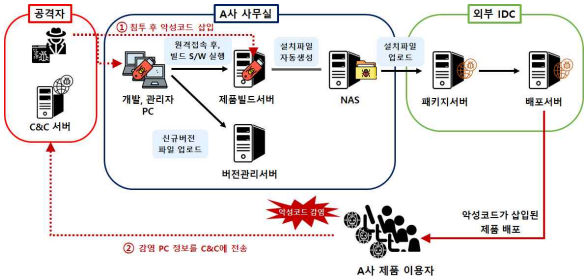


Fig. 1. 개발환경 침투 도식화

2.1.1 Copay사 Bitcoin Wallet

2018년, 깃허브 공식 게시판에 보고된 코드 검증을 통하여 Copay 사의 앱에 사용된 오픈소스 라이브러리에 악성코드가 심어져 있음이 확인되었다. 검증 결과 오픈 소스 코드인 event-stream 3.3.6버전 패키지에 포함된 flatmap-stream 라이브러리에 백도어 악성코드가 삽입된 것으로 확인되었다. 해당 악성코드는 개발자가 앱을 출시하기 위해 앱을 패키징할 때 발현되므로, 개발자는 사전에 악성코드를 발견하지 못하였다. 악성코드가 삽입된 앱을 설치한 이용자는 개인 정보를 탈취당하였고, 공격자는 탈취한 개인 정보 및 설치된 백도어를 이용하여 비트코인 탈취를 시도하였다[4].

2.1.2 ShadowPad 악성코드 삽입

2017년, 카스퍼스키랩은 전 세계적인 이용자층을 보유한 서버 관리 소프트웨어 제품인 넷사랑 프로그램에 백도어가 설치되어 있음을 발견했다. 조사 결과, 공격자는 넷사랑 제품빌드 서버에 침투 후 서버 관리 프로그램에 Shadowpad 백도어를 삽입하였다. 해당 버전을 통해 공격자는 이용자들에게 추가적인 악성 모듈을 다운로드하게 하거나 제품 이용자의 데이터 탈취를 시도한 것으로 확인되었다[5].

2.1.3 XcodeGhost

2015년, 백도어가 설치된 iOS 앱이 공식 앱스토어에 등록되는 사고가 발생하였다. 공격자는 애플 통합 개발 프레임워크인 엑스코드(Xcode)에 악성코드를 삽입한 후 각종 인터넷 게시판에 업로드하였다. 비공식 다운로드 경로를 통해 엑스코드를 다운받은 중국 개발자들은

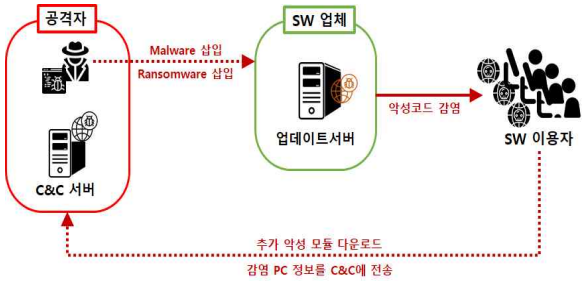


Fig. 3. 업데이트 서버 침투 도식화

감염된 프레임워크를 사용하여 앱을 개발하였고, 자동으로 악성코드가 포함된 앱을 의심 없이 앱스토어에 등록하였다. 그 결과 공식 앱스토어를 통해 많은 사용자들에게 감염된 앱이 배포되었고, 공격자는 감염된 앱을 통해 C&C서버로 사용자 정보를 탈취 및 추가 명령을 하달하였다.

2.2 업데이트 서버 침투

업데이트 서버 침투란 공격자가 제품의 업데이트 서버에 침투하여 공격을 시도하는 방법이다. Fig. 3와 같이 업데이트 서버 또는 배포 서버를 해킹하여 악성코드를 정상 프로그램으로 위장하여 배포한다. 업데이트 서버 침투는 개발환경 침투에 비해 코드 변조 등의 한계가 있어 공격이 발각될 확률이 높다. 하지만 공격을 위해 투자한 시간 대비 상당한 효과를 얻을 수 있는 전략으로, 공격자들이 주로 사용하는 공급망 공격 방법이다.

2.2.1 ASUS 업데이트 서버 침투

2019년, 카스퍼스키랩은 컴퓨터, 네트워크 장비 등을 제조하는 ASUS 사의 업데이트 서버가 침해당했음을 발표하였다. 공격자는 ASUS 업데이트 서버에 침투하여 업데이트 파일에 악성코드를 삽입하였고, 해당 버전으로 업데이트한 사용자는 모두 악성코드에 감염되었다. 카스퍼스키 솔루션에 탐지된 감염 PC는 대략 5만 7천대로 통계적 방법에 기반하면 전 세계적으로 100만대 이상의 PC가 악성코드에 감염되었을 것으로 추정된다[6].

2.2.2 C클리너 다운로드 서버 침투

2017년, 백신 업체 어베스트에서 인수한 C클리너에 악성코드가 삽입되는 사고가 발생하였다. 공격자는 어베스트 다운로드 서버를 침투하여 정상적인 C클리너 파일을 악성코드가 심어진 C클리너(5.33버전)로 교체하였고, 그 결과 한 달간 230만 명 이상이 감염된 C클리너를 다운받았다. 감염된 C클리너를 설치한 이용자들은 시스템 정보를 공격자의 C&C 서버로 전송하고, C&C 서버로부터 추가적인 명령을 하달받게 되었다.

2.2.3 MeDoc 업데이트 서버 침투

2017년, 우크라이나의 세무 회계 소프트웨어 업체인 MeDoc 사의 업데이트 서버가 침투당하는 사고가 발생하였다. 공격자는 MeDoc 업데이트 서버를 침투한 후 업데이트 파일에 랜섬웨어 감염을 목적으로 한 악성코드를 삽입하였다. 이후 소프트웨어의 자동 업데이트를 통해 이 제품을 사용하는 모든 기기가 랜섬웨어에 감염되는 사고가 발생하였다. 이 사건은 플랫폼의 정상적인 업데이트 과정에서 악성코드가 유입되어 해당 제품을 사용하는 모든 기업이 랜섬웨어에 감염된 사건으로, 대표적인 IT 공급망 공격 사고라 할 수 있다.

2.3 정리 및 시사점

최근 발생한 국내외 공급망 침해사고 중 대표적인 6개의 사건을 공격 특징별로 3개씩 구분하여 나열하였다. Fig. 2은 앞서 소개한 공급망 공격을 사고발생 순으로 나열한 타임라인이

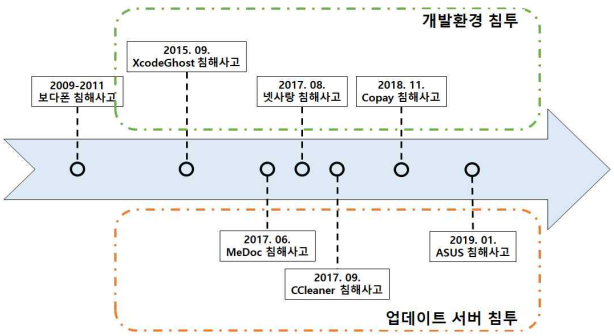


Fig. 2. IT 공급망 공격 타임라인

다. 이 밖에도 공장 출하 시 악성코드가 삽입되는 하드웨어 공급망 공격 사건과 생략한 IT 공급망 공격 사례를 생각하면, 꾸준히 공급망 공격을 통한 기업 침투 및 기밀자료 유출이 발생하고 있음을 알 수 있다.

이렇듯 기존에 사용 중인 소프트웨어가 정상적인 방법으로 업데이트 서버를 통해 악성코드가 유입되므로, 일반적인 바이러스 침투와 비교하면 공급망 공격은 탐지가 매우 어려운 문제가 있다. 또한, 개발환경에 침투하여 정상 파일에 악성 모듈을 삽입할 경우, 전자서명까지 속일 수 있으므로 일반 고객으로서는 해당 프로그램이 악성코드에 감염되었을 거라는 의심을 하기가 사실상 불가능하다. 따라서 IT 공급망의 해킹 및 사고 사례에 대해 지속적으로 모니터링하며 해킹에 대한 기술적 요소 식별 및 분석을 통해 증가하는 공급망 침해 시도에 대응하는 자세가 필요하다.

III. 결론

최근 공급망 공격은 높은 보안시스템이 구축된 기관에 직접 침투하기보다, 상대적으로 보안 인프라가 미흡한 제조·납품 업체를 통해 우회 침투하기 시작하였다. 공급망 공격은 기관 내 사용 중인 소프트웨어가 정상적인 방법으로 업데이트 서버를 통해 악성코드가 유입되므로 사전에 탐지 및 방어가 매우 어렵다.

IT 공급망 공격은 전 세계적으로 꾸준히 증가하고 있음에 따라 지속적으로 해킹 이슈 및 사고 사례를 모니터링하며 해킹에 대한 기술적인 요소 식별 및 분석이 매우 필요한 실정이다. 이에 따라 본 논문에서는 그간 알려진 다양한 공급망 공격을 개발환경 침투와 업데이트 서버 침투로 분류하여 소개함으로써 공급망 공격의 위험성을 강조하였다.

향후 연구로는 국내 환경에 적합한 IT 공급망 보안 체계 수립을 위해 전 세계적으로 시행중인 공급망 위험관리 체계를 조사·분석할 예정이다. 최종적으로 국가 차원에서 IT 공급망을 관리할 수 있는 시스템 및 체계 구축에 관한 연구를 수행할 예정이다.

[참고문헌]

[1] KISA, “2018년 2분기 사이버 위협 동향 보고서,” Jul. 2018.

[2] KISA, “20118년 4분기 사이버 위협 동향 보고서,” Jan. 2019.

[3] Symantec, “Internet Security Threat Report,” Feb. 2019.

[4] 테크레시피, “비트코인 지갑 노린 오픈소스 공급망 공격,” Dec. 2018. (https://techrecipe.co.kr/posts/4946)

[5] ITDaily, “카스퍼스키랩, 대규모 섀도우패드 공격 징후 확인,” Aug. 2017. (http://www.itdaily.kr/news/articleView.html?idxno=84647)

[6] KISA, “「공급망 공격 사례 분석 및 대응방안」 기술보고서,” Jul. 2019.

화이트해커와 기업 분석을 통한 국내 버그바운티 활성화 방안 제시

하희헌, 정현경, 강민주, 지용빈, 이광진, 신예준

KITRI 차세대 보안리더 양성 프로그램 BoB(Best of the Best)

A Proposal for Revitalizing BugBounty in Korea through White Hacker and Enterprise Analysis

HeeHeon Ha, HyunKyung Jung, MinJu Kang, YongBin Ji, GwangJin Lee, YeJun Shin

KITRI 차세대 보안리더 양성 프로그램 BoB(Best of the Best)

요 약

버그바운티란 기업의 제품이나 서비스의 취약점을 발견한 화이트해커에게 포상금을 지급하는 제도이다. 해외에서는 기업 자체 제품 및 서비스의 보안을 강화하려는 수단으로써 버그바운티 프로그램을 적극적으로 활용하고 있다. 반면 국내에는 버그바운티 프로그램이 아직 활성화되고 있지 않은 실정이다. 본 논문에서는 국내 버그바운티 활성화를 위해 화이트해커와 버그바운티를 운영하는 기업 보안 담당자 사이에 발생할 수 있는 의견 차이를 분석하여, 국내 버그바운티 활성화를 위한 방안을 제시한다.

I. 서론

제 4차 산업 혁명의 시대가 본격화되고 소프트웨어 및 네트워크 기술이 비약적으로 발전하면서 취약점 공격 및 위협은 증대되고 있다. 이에 따라, 취약점 정보는 상품으로서 가치를 지니게 되었다[1][2].

이러한 취약점 정보는 이전부터 블랙마켓에서 거래해 왔기 때문에 현재 시행되고 있는 버그바운티는 새로운 프로그램이 아니다[3].

기존 연구에 따르면 버그바운티는 악용 가능한 취약점을 줄여 블랙마켓에 참여할 가능성을 감소시킬 수 있는 인센티브를 제공하고, 조직이 버그바운티 프로그램을 시행하는 것은 보안 전문가를 직접 고용하는 것보다 보안 연구비를 절감할 수 있고, 대량의 취약점을 발견하는데 효율적이라고 설명한다[4][5].

구글(Google), 페이스북(Facebook), 마이크로소프트(Microsoft) 등 글로벌 ICT 기업들은 이미 기업 자체 제품 및 서비스의 보안을 강화하려는 수단으로써 버그바운티 프로그램을 적극적으로 활용하고 있다.

반면 국내에는 버그바운티 프로그램이 아직 활성화되고 있지 않은 실정이다. 한국인터넷진흥원(Korea Internet and Security Agency, KISA)이 주도하여 'S/W 신규 취약점 신고포상제'를 기업과 공동 운영하고 있지만, 기업 자체적으로 버그바운티를 시행하는 기업은 극히 일부에 불과하다[6].

본 연구에서는 이러한 문제점에 초점을 맞춰 화이트해커와 버그바운티를 운영하는 기업 보안 담당자의 의견을 각각 설문조사와 인터뷰를 통하여 수집한 후 비교 분석함으로써 취약점 제보자와 평가자의 의견을 반영한 버그바운티 활성화 방안 제시를 목적으로 한다.

II. 배경 및 관련 연구

2.1 용어 정의

버그바운티는 보안취약점 신고포상제로 기업의 서비스 및 제품의 취약점을 신고한 해커에게 포상을 주는 제도이다.

버그바운티 프로그램은 자사 서비스와 제품을 해킹해 취약점을 신고한 해커에게 포상금을 지급하는 것을 뜻한다.

버그바운티 플랫폼은 기업들이 해커들과 쉽게 상호작용하고 보고서를 분류하고 보상해주는 것을 뜻한다.

2.2 관련 연구 동향

버그바운티 프로그램에 대한 연구는 다양한 방향으로 진행되고 있다.

기업들이 화이트해커 커뮤니티와 원활한 커뮤니케이션 관계를 구축하기 위한 노력이 필요함과 동시에 동기부여 식 맞춤형 보상 시스템이 필요하다[7].

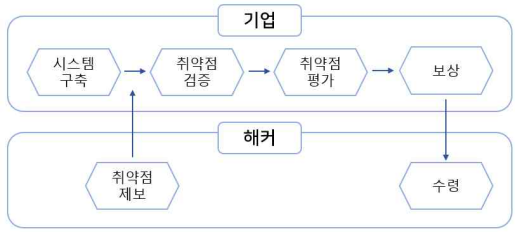
취약점을 발견하는 화이트해커와 화이트해커 커뮤니티 내부의 다양성에 초점을 맞춰서 탐색, 연구하여 더 많은 참여와 다양성이 취약점 발견 프로세스의 생산성을 높이는데 기여한다는 것을 설명한다[8].

III. 현황 파악

화이트해커와 기업 보안 담당자의 버그바운티 프로세스에 대한 각각의 의견을 수집하기 위해 설문조사와 인터뷰를 설계하여 2019년 10월 ~ 11월 동안 온, 오프라인으로 시행했다.

3.1 현황 분석 방법

설문조사는 KISA 명예의전당 전체 리스트 중 일부와 버그바운티 참여 경험이 있는 화이트해커들 중 101명을 선정하였고, 인터뷰는 국내에서 버그바운티를 운영하고 있는 5개 기업의 보안 담당자와 인터뷰를 진행하였다.



[그림 1] 버그바운티 프로세스

[그림 1]은 일반적인 버그바운티 프로세스를 보여준다. 본 연구에서는 크게 6단계로 버그바운티 프로세스를 구분하였다.

설문조사와 인터뷰 질문 리스트의 경우 각 프로세스를 중심으로 버그바운티 관련 연구 및 취약점 발견 프로세스 연구[6]를 참고하여 화이트해커와 기업 보안 담당자에게 해당하는 질문을 작성하였다.

3.2 현황 분석 결과

화이트해커 설문조사와 기업 보안 담당자의 인터뷰 결과는 [표 1], [표 2]에 정리된 바와 같다. 설문조사는 참여 경험, 버그바운티 제보, 기

[표 1] 화이트해커 설문조사 결과 요약

구분	질문 / 답변	답변 1	답변 2	답변 3	답변 4
참여 경험	접근 경로	지인 추천 (39.7%)	보안 컨퍼런스 (19.8%)	공식 홈페이지 (19.8%)	SNS (19%)
	참여 동기	금전 (37.4%)	명예 (25.2%)	흥미 (19.8%)	학습 (17.6%)
버그 바운티 제보	보고서 항목 중요도	PoC (21%)	영향 및 시나리오 (16%)	재현 방법 (14%)	취약점 개요 (14%)
	취약점 공개 허용 범위	취약점유형과 버그바운티 금액 (34.2%)	민감정보 제외 보고서 전문 (31.6%)	취약점 요약 (31.6%)	기타 (3.6%)
	추가 요청항목	재현 방법 (54.9%)	PoC (19.6%)	영향 및 시나리오 (15.8%)	취약점 개요 (6.1%)
기타	개선이 필요한 항목	포상 인정범위 안내 (34%)	보고서 양식 (23.4%)	제한된 행위 안내 (10.6%)	취약점 카테고리 분류 (8.5%)

[표 2] 기업 보안 담당자 인터뷰 결과 요약

구분	질문/답변	플랫폼 A	플랫폼 B	프로그램 a	프로그램 b	프로그램 c
버그 바운티 도입	참고자료 확보	부족했음	충분하지 않았음	국내 자료 부재	어려웠음	-
	기대효과	침해사고 예방 인력 양성	보안 향상 학습 욕구	비용 절감 새로운 관점 획득	-	보안사고 예방
버그 바운티 업무	홍보 방식	SNS, 언론보도	SNS, 언론보도	SNS, Web	SNS	컨퍼런스, Web
	보고서 항목 중요도	영향도, 보급도, 발굴난이도, 재현방법	영향도, reliability, PoC	영향도, PoC	PoC	재현방법
	추가 작성을 요청 받은 보고서 항목	재현방법	재현방법, 동의서	재현방법, 지급관련서류	재현방법, PoC	재현방법
	업무의 고충	구축, 검증	구축	구축	구축	평가

타 내용 등을 조사하여 종합하였고, 인터뷰의 경우 버그바운티 도입과 업무 등을 조사하여 정리하였다.

3.2.1 홍보 방식

화이트해커의 접근 경로는 지인 추천(39.7%), 보안 컨퍼런스(19.8%)와 같이 커뮤니티 활동을 통하여 버그바운티에 참여함을 나타낸다[표 1].

반면에 기업 보안 담당자의 홍보 방법은 대부분 SNS, Web, 언론 보도를 통해 홍보한다는 사실을 나타낸다[표 2].

이를 통해 기존 기업의 홍보 방식은 화이트해커들의 참여 유도에 큰 영향을 끼치지 못한다는 것을 시사한다.

3.2.2 보고서 양식 및 항목 중요도

[표 1]의 개선이 필요한 항목은 화이트해커가 가장 많은 참여 경험이 있는 버그바운티 플랫폼을 대상으로 한 결과를 기반으로 작성하였고, 버그바운티 플랫폼의 안내와 보고서 양식 항목에서 개선이 필요함을 나타냈다.

[표 1]과 [표 2]의 보고서 항목 중요도는 화이트해커의 경우 PoC(Proof of Concept, 21%), 영향 및 시나리오(16%)와 같이 취약점을 증명하는 결과와 취약점이 끼치는 영향을 중요시하지만, 기업 보안 담당자의 경우 PoC와 영향도는 물론 재현방법과 같은 취약점이 발생하는 원인에 대해서 중요하게 생각한다는 것을 나타낸다.

현황을 분석한 결과 화이트해커와 기업 보안 담당자 간 홍보 방식과 보고서 항목의 중요도에서 문제점이 도출되었고, 이에 대한 인식차를 해결할 수 있는 방안을 4장에서 제안한다.

IV. 활성화 방안 제시

4장에서는 위에서 비교 분석한 문제점들에 대한 국내 버그바운티 활성화 방안을 기술하고자 한다.

4.1 버그바운티 참여 유발을 위한 커뮤니티 마련

해외의 버그바운티는 버그바운티 커뮤니티를 발판 삼아 활성화되어 있다. 해당 커뮤니티로는 취약점 연구&대회의 성격을 가진 폰투온(Pwn2Own)이 있다.

폰투온의 시행 배경은 자주 사용하는 S/W와 H/W의 취약점을 발견하여 더 높은 보안수준을 만드는 것을 목표로 한다. 특히 발표 90일 후 취약점이 발견된 기기의 익스플로잇이 공개되는 만큼 기업은 해당 취약점을 패치하기 위해 보안 이슈에 적극적으로 대처하도록 유도할 수 있다.

반면에 국내의 경우 버그바운티 커뮤니티는 거의 전무하다. 국내 커뮤니티로는 2018년 KISA에서 진행하였던 버그바운티 대회 'Hack the KISA'가 있다. 국내 공공기관 최초로 진행되었던 버그바운티 대회이며 총 485명의 민간 보안 전문가들이 참여하였으며, 이벤트성 대회임에도 큰 의미를 남겼다.

이처럼 화이트해커들의 버그바운티 참여를

지속적으로 유발할 수 있는 커뮤니티가 필요하다. 다양한 이벤트와 보안 컨퍼런스를 통하여 버그바운티 참여율을 향상시키면 블랙마켓의 참여율을 저조시키고, 취약점 발견 프로세스의 생산성을 높일 수 있다[5][8].

4.2 보고서 양식 개선 및 구체적인 요구 명시

[표 1]의 개선이 필요한 항목에서 보고서 양식의 문제점이 도출되어 이를 개선하기 위해 해외 플랫폼의 보고서 양식과 비교하였다.

[표 4] 국내, 해외 플랫폼 보고서 양식 비교

국내 플랫폼 A	해외 플랫폼 A'
제목	요약
요약	기술
S/W버전	영향 받은 제품, 서비스
발생환경	인증
취약점 발견방법	재현단계
취약점 발생원인	영향
취약점 증명/검증	-
취약점 악용시나리오	-
조치방안	-
기타	-

해외 플랫폼의 보고서 항목과 국내 플랫폼 보고서 항목을 비교했을 때, 해외 플랫폼 보고서 항목이 국내 플랫폼 보고서 양식보다 간소하다는 사실을 알 수 있다[표 4].

따라서 국내의 보고서 양식도 해외처럼 간소화되는 것이 필요하다. 추가로 사람마다 관점과 기준의 차이로 인해 이해하는 바가 달라질 수 있기 때문에 보고서 항목 부분에서 구체적인 요구 명시가 필요하다. 특정 수준까지의 설명과 적절한 예시의 제시는 구체적인 요구 명시의 좋은 방법이다.

V. 결론 및 향후 계획

본 연구에서는 설문조사와 인터뷰를 통해 화이트해커와 버그바운티를 운영하는 기업 보안 담당자 사이에 발생할 수 있는 의견 차이를 분석하여, 국내 버그바운티 활성화를 위한 방안을 제시하였다.

화이트해커와 기업 보안 담당자 사이에서 홍보 방식과 보고서 양식 및 항목 중요도에서 의견 차이를 도출하였고, 버그바운티의 지속적인 참여를 유발시키기 위해서 보안 컨퍼런스와 버그바운티 대회와 같은 커뮤니티의 필요성을 나타내었다. 또한 보고서 양식 및 항목 중요도의 의견 차이를 좁히기 위해 보고서 양식의 개선과 요구사항을 구체적으로 명시하는 방안을 제시하였다.

향후 계획으로 기업들의 버그바운티 접근과 적용의 용이성을 위해 기업 보안 담당자들의 의견을 반영하여 버그바운티 프로그램 도입 가이드라인을 제시하고, 국내의 버그바운티 활성화를 위하여 화이트해커들의 의견과 기업 보안 담당자들의 의견을 반영하여 버그바운티 플랫폼을 구축할 것이다.

[참고문헌]

[1] Lillian Ablon, M. C. Libicki, A. A. Golay, Markets for Cybercrime Tools and Stolen Data: Hackers Bazaar, RAND, 2014.

[2] 홍준호, 유현우, "화이트해커 양성 및 활성화 방안에 대한 연구", 한국법학회, pp. 463-515, 2017년 12월.

[3] Charlie Miller, "The Legitimate Vulnerability Market: Inside the secretive world of 0-day exploit sales", Workshop on the Economics of Information Security, 2007.

[4] M. Finifter, D. Akhawe, and D. Wagner, "An empirical study of vulnerability rewards programs", USENIX Security Symposium, pp. 273 - 288, 2013.

[5] Jukka Ruohonen, Luca Allodi, "A Bug Bounty Perspective on the Disclosure of Web Vulnerabilities", Workshop on the Economics of Information Security, 2018.

[6] 한국인터넷진흥원, S/W 신규 취약점 신고포상제 운영 안내서.

[7] D. Votipka, R. Stevens, E. M. Redmiles, J Hu, and M. L. Mazurek, "Hackers vs. Testers: Acomparison of Software Vulnerability Discovery Processes". IEEE Symposium on Security, 2018.

[8] M. Zhao, J. Grossklags, and K. Chen. "An Exploratory Study of White Hat Behaviors in a Web Vulnerability Disclosure Program", ACM workshop on security information workers, pp. 51-58, 2014.

블록체인 노드의 네트워크 고립을 방지하는 전파 프로토콜 제안*

김미연** 염흥열***

**(주)엔에스에이치씨 ***순천향대학교 정보보호학과

Proposal of Broadcast Protocol Preventing Blockchain Nodes from Network Isolation

Mee Yeon Kim** Heung Yeol Youm***

**NSHC ***Department of Information Security Engineering, SoonChunHyang University.

요 약

블록체인은 P2P 네트워크를 기반으로 구성된다. 노드가 처음 블록체인 네트워크에 합류하면 다른 노드와 연결되고 노드끼리 메시지를 주고받으며, 블록체인 네트워크에 합류하면서부터 메시지를 주고받는 동안 노드는 sybil attack, eclipse attack, selfish mining 등 다양한 위협에 노출된다. 특히 처음 합류하게 되는 노드는 악의적인 노드에 연결될 시 조작된 블록체인을 전달받거나 악의적인 노드 하위의 마이닝 봇처럼 이용될 수 있는 위험이 있다. 본 논문에서는 이러한 공격에 대한 대안으로서 상위 지분율을 가진 노드를 대표로 선정하여 처음 네트워크에 합류하게 되는 노드에게 블록체인을 동기화할 수 있는 권한을 부여하는 위임 전파 프로토콜(Delegated Broadcast Protocol)을 제안한다. 제안 프로토콜을 통해 네트워크 고립을 방지하고, 새로 전파되는 다수의 블록에 대한 신뢰성을 증대시키는 방안을 제시한다.

I. 서론

블록체인은 P2P 네트워크를 기반으로 네트워크를 구성한다. 노드가 처음 블록체인 네트워크에 합류하면 다른 노드와 연결되고 노드끼리 메시지를 주고받는다. 주고받는 메시지에는 버전 혹은 ping, 트랜잭션, 블록에 대한 메시지 등이 포함된다. 또한 노드는 블록체인 네트워크에 합류하면서부터 메시지를 주고받는 동안 다양한 위협에 노출된다.

관련된 공격으로는 네트워크 및 마이닝에 관한 공격인 sybil attack, eclipse attack, selfish mining 등이 있다. 공격자는 이러한 비정상적인 행위로 이득을 취하거나 다른 노드를 악의적으로 이용하기도 한다. 특히 처음 합류하게 되는 노드는 악의적인 노드에 연결될 시 조작된 블록체인을 전달받거나 악의적인 노드 하위의 마이닝 봇처럼 이용될 수 있는 위험이 있다.

본 논문에서는 새로 합류하는 노드에 대한 네트워크 고립을 방지하고, selfish mining을 방지하는 대안으로서 위임 전파 프로토콜(Delegated Broadcast Protocol)을 제안한다.

위임 전파 프로토콜(Delegated Broadcast Protocol)이란 상위 지분율을 가진 노드를 대표로 선정하여 처음 네트워크에 합류하게 되는 노드에게 블록체인을 동기화할 수 있는 권한을 부여하는 전파 프로토콜을 의미한다.

본 논문은 2장에서 제안 프로토콜과 관련된 네트워크 및 마이닝 공격에 대해 소개하고, 3장에서 제안 프로토콜에 대한 설명, 상세 설명과 이에 대한 검토를 통해 각 공격의 대책이 될

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (2019-0-00660, 차세대 ICT 환경에서의 보안 및 개인정보 보호 기술 국제 표준화 추진)

수 있는지에 대한 여부를 검토한다. 마지막으로 4장 결론에서 본 프로토콜의 의의를 밝힌다.

II. 관련 공격

본 장에서는 제안 프로토콜로 방지할 수 있는 블록체인의 네트워크 및 마이닝 관련 공격에 대해 기술한다.

2.1 Sybil attack

한 공격자가 다수의 신원을 생성하여 마치 개별적인 노드인 것처럼 가장하는 공격이다. 공격자는 하나의 노드에서 다수의 신원을 관리하거나, 다수의 노드를 이용하여 실제보다 더 큰 영향력을 행사할 수도 있다. 공격자는 다수의 가짜 신원을 이용해 피해자와 연결되는 모든 피어 노드를 장악한다. 이를 통해 피해 노드를 네트워크 상 고립시켜 피해 노드가 생성한 트랜잭션을 이중 지불 공격에 사용하는 등의 조작을 할 수 있다[1].

2.2 Eclipse attack

공격자가 다수의 IP를 이용해 피해 노드를 고립시키는 공격을 의미한다. 공격자는 먼저 피해 노드와 연결되는 통신을 반복적으로 수립하여, 피해 노드의 피어 테이블에 공격자의 IP 혹은 블록체인 네트워크와 관련없는 IP로 채운다. 피해 노드가 버전 업데이트, 네트워크 연결 실패 혹은 파워에 문제가 발생하는 등의 이유로 시스템을 재구동하는 경우, 기존의 연결을 손실하고 공격자의 IP로만 통신을 구성하게 될 확률이 매우 높아진다. 공격이 성공하여 피해 노드의 통신을 장악하면 공격자는 피해 노드를 악의적인 마이닝, 합의 공격, 이중 지불 등에 이용할 수 있다.[1]

2.3 Selfish mining

Selfish mining은 악의적인 마이너가 블록을 생성하여 일부 블록만 전파하거나, 다수의 블록을 한꺼번에 전파하는 공격이다. 이를 통해 블록체인을 탈취하고 기존에 있던 블록을 폐기하여, 일반 마이너는 그로 취득한 이익과 사용한 자원을 잃는다[1].

악의적인 마이너는 기존 블록체인과 동일한 자신만의 체인을 두고, 블록을 생성한다. 블록을 생성하여도 바로 전파하지 않고 다른 노드가 블록을 생성하면 그때 자신의 블록을 전파한다. 만약 개인이 가진 체인이 기존 체인보다 1 블록 이상 길어지는 경우 제일 끝단의 블록만 전파한다. 그러면 기존의 블록체인이 악의적인 마이너의 블록체인으로 변경되고, 새로 생성된 블록들이 폐기되면서 블록 보상을 잃게 된다[1].

III. 제안

3.1 위임 전파 프로토콜

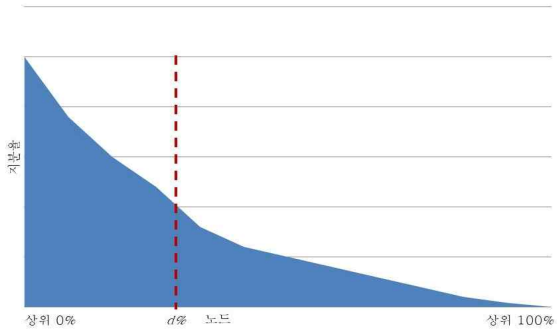
위임 전파 프로토콜은 네트워크 불균형 및 고립 등으로 생길 수 있는 위협을 방지하기 위해 고안된 전파 프로토콜이다. 특정 개수 이상의 다수의 블록을 업데이트해야 하는 경우, 상위 지분율을 가진 노드만이 블록을 전파할 수 있게 허가하며, 이를 통해 업데이트된 블록에 대한 신뢰성을 증대시킨다.

예를 들어 새로 네트워크에 합류하게 된 노드는 피어 노드로부터 블록체인을 동기화 받아야한다. 그러나 조작된 블록체인을 받는 경우, 악의적인 공격자에게 자산이 탈취되거나, 공격자 하위의 마이닝 봇처럼 이용될 수 있다. 이를 방지하기 위해 대표 노드만을 통해 전체 블록체인을 동기화하며, 검증을 거쳐 블록체인을 받아들인다. 이때 대표 노드가 아닌 노드의 전파 메시지는 받아들이지 않는다.

제안하는 프로토콜은 지분을 많이 가진 노드일수록 네트워크를 깨끗하게 유지할 것이라는 동인에 기인한다. PoS 등 일부 블록체인 합의 알고리즘은 블록체인 시스템의 노드가 가진 암호화폐 지분이 많을수록 네트워크를 깨끗하게 유지하여 화폐의 가치를 떨어뜨리지 않을 것이라는 동인에 기반한다. 이와 마찬가지로 제안 프로토콜도 상위 지분을 가진 노드일수록 악의적이지 않을 확률이 높다는 전제를 기반으로 한다. 그러나 PoS, 혹은 DPoS와 달리 위임 전파 프로토콜은 다수의 블록을 한꺼번에 전파하는 경우에만 적용되며 블록 생성과는 무관하다.

이는 새로 합류하는 신규 노드, 특정 개수 이상의 블록을 업데이트해야 하는 일부 노드에 대해 네트워크 고립 및 악용을 방지하고자 하는 대안으로 블록 생성과는 무관하게 작동한다.

전체 노드의 지분율을 나타내는 그래프 [그림 1]이 있다고 하자. x축은 상위 0% 노드서부터 100% 노드까지 나열하였고, y축은 각 노드가 가진 지분율을 표시한다. 빨간 점선은 대표 노드로 위임되는 노드를 선정하는 기준이 되는 상위 지분율 $d\%$ 지점을 가리킨다. 즉, $d\%$를 기점으로 $d\%$와 같거나 많은 지분율을 가진 노드는 대표 노드가 되고, 그 외의 노드는 일반적인 노드가 된다.



[그림 1] 전체 노드의 지분율 그래프

이때 전체 노드 중 상위 $d\%$에 위치한 노드의 지분율 $stake_d$ 는 [표 1]과 같은 예시 방식으로 구할 수 있다.

[표 1] 지분율 $stake_d$를 구하는 예시 방식[2]

상위 $d\%$에 위치한 노드의 지분율 $stake_d$를 구하는 예시식	
$a = \frac{d}{100}(N+1),$ $a-th\ value = stake_d$	(1)

이때 N 은 전체 노드수를 의미하며 a 는 상위 $d\%$에 해당하는 실제 서수값을 나타낸다. 따라서 a 번째에 해당되는 실제값이 $stake_d$ 가 된다.

[표 2]의 알고리즘은 위임 전파 프로토콜이 $Node_i$ 의 지분율을 계산하여 대표(*delegation*)인지 아닌지를 판단하고 그에 따라 메시지를 처리하는 프로세스를 알고리즘으로 나타낸 것이다. (1)전체 암호 자산의 총량 중에서 $Node_i$ 가 가진 암호 자산량을 나눠 $Node_i$ 의 지분률 $stake_i$ 를 구한다. (2)$Node_i$ 의 지분율 $stake_i$ 가 $d\%$의 지분율 $stake_d$ 보다 큰 경우 *delegation*으로 지정하고 (3)아닌 경우 *None*으로 지정한다. (4)$Node_i$ 가 *delegation*인 경우 메시지를 처리하는 함수 MSG()를 호출하고 아닌 경우 *False*값을 반환하여 프로토콜을 중단한다.

[표 2] 위임 전파 프로토콜 알고리즘

Delegated Broadcast Protocol
1: $stake_i \leftarrow \frac{amount_i}{amount_{al}}$
2: IF $stake_i \geq stake_d$ THEN $Node_i$ = delegation
3: ELSE $Node_i$ = None
4: IF $Node_i$ == delegation THEN Call MSG()
5: ELSE Return False

상위 $d\%$의 노드를 결정하는 기준인 d 값은 적용하는 블록체인 플랫폼의 특성(e.g. 전체 노드 수, 활동성, 블록체인 종류, 시간당 트랜잭션 수, 블록 생성 수 등)에 따라 조정할 수 있다. 또한 지분율은 변동될 수 있기 때문에 $d\%$의 지분율을 측정하는 주기를 결정해야 하며 이 또한 플랫폼에 적절하게 설정할 수 있다. 노드별로 inbound/outbound로 연결되어야 하는 최소 대표 노드의 수를 조절할 수 있다. 대표 노드 선출 방식을 지분 기반 선정 방법에 더해 Round Robin, PoA(평판 기반) 등을 추가적으로 결합하여, 대표 노드를 보다 유동적으로 선정할 수 있다. 위와 같이 위임 전파 프로토콜 관련 설정을 커스터마이즈하여 적용할 수 있다.

3.2 위임 전파 프로토콜에 대한 검토

위임 전파 프로토콜로 조작된 블록을 전파하여 피해 노드를 장악하기 위해서는 피해 노드와 연결되는 대표자의 노드가 모두 공격자의 노드여야 한다. 즉, m 개의 악의적인 노드의 집합 $Node_m$ 이 공격을 성공시키기 위해서는 각 노드가 정한 연결되어야 하는 최소 대표 노드 수 b 에 대해 다음 두 조건을 만족시켜야 한다.

[표 3] 위임 전파 프로토콜 공격 성공 조건

공격 성공 조건	
$m \geq b$	(1)
$where\ i \in [1,m]$	(2)
$\sum_{i=1}^{m} stake_i \geq b \bullet stake_d$	

악의적인 노드의 집합이 대표로 위임받아 피해 노드와의 통신을 장악해야 하므로, 공격자의 노드는 대표 노드 최소 연결 수 b 이상이 되어야 하고, 따라서 조건 (1)을 만족하여야 한다. 또한 악의적인 노드는 지분율을 $stake_d$ 이상이 가져야 대표로 위임될 수 있기 때문에, 조건 (2)를 만족해야 한다.

그렇다면 위임 전파 프로토콜이 적용된 블록체인에서 공격자가 sybil attack, eclipse attack 등 피해 노드의 네트워크를 고립시키는 공격을 수행하는 경우를 검토해보도록 한다. 이 경우 공격자가 대표 노드 최소 연결 수 b 를 채우지 못하면, 피해 노드는 대표 노드에게 네트워크 통신 수립을 요청할 것이고 네트워크 고립을 탈출할 수 있다. 따라서 공격자가 피해 노드의 IP 테이블 혹은 통신을 장악하기 위해서는 b개 이상의 대표 노드를 보유해야 하며, $b \bullet stake_d$ 이상의 지분을 최소로 소유해야 한다. 즉 조건 (1),(2)를 추가적으로 충족해야 공격이 성공가능하다.

두 번째로 위임 전파 프로토콜 상 selfish mining을 시도하는 경우를 고려해보도록 하자. 위임 전파 프로토콜은 상위 지분율을 가진 노드만이 다수 개의 블록을 전파할 수 있게 허용한다. 그렇다면 c 개 이상 블록을 전파하는 경우 위임 전파 프로토콜을 적용해야 한다고 가정해보자. Selfish mining으로 생성한 블록이 c 개 미만인 경우는 위임 전파 프로토콜로 방지할 수 없으나, 생성한 블록이 $n \geq c$ 를 만족하는 n 개의 경우 selfish mining을 수행한 노드가 *delegation*이 아니라면 중간 블록을 전파할 수 없다. 즉 블록체인 끝단서부터 n번째의 블록과 c번째의 블록을 전파할 수 없게 된다. 따라서 selfish mining을 수행하는 마이너가 대표 노드를 갖고 있지 않으면 공격이 불가능하다.

IV. 결론

본 논문에서는 네트워크 고립을 방지하고, 전파되는 다수의 블록에 대한 신뢰성을 증대시키는 방안으로 위임 전파 프로토콜을 제안하였다. 우선 제안 프로토콜과 관련될 수 있는 sybil attack, eclipse attack, selfish mining 등 네트워크 및 마이닝 공격을 살펴보았다. 또한 위임 전파 프로토콜에 대한 설명, 전제에 대해 설명하였고, $stake_d$를 구하는 예시, 프로토콜에 대한 상세 설명을 기술하였다. 마지막으로 제안 프로토콜에 대한 공격이 성공하기 위한 조건과 제안 프로토콜이 어떻게 관련 공격을 방지할 수 있는지를 검토하였다.

위임 전파 프로토콜은 상위 지분율을 가진 노드만이 특정 수 이상의 다수의 블록을 전파할 수 있게 허가해주는 프로토콜로, 네트워크에 합류하는 신규 노드의 블록체인 동기화나 네트워크에 전파되는 다수의 블록에 대한 신뢰성을 증대시켜 줄 수 있다. 또한 네트워크 고립, selfish mining등으로 비롯되는 공격의 충족 조건을 더욱 어렵게 하는 의의를 가진다. 따라서 네트워크에 전파되는 다수의 블록에 대한 신뢰성을 증대시키고, 조작된 블록체인을 받거나, 악의적인 공격자에게 자산이 탈취, 공격자 하위의 마이닝 봇처럼 이용되는 것을 일정 부분 방지할 수 있을 것으로 기대된다.

[참고문헌]

[1] J.H.Mosakheil, “Security Threats Classification in Blockchains,” *Culminating Projects in Information Assurance,* May, 2018. Available: https://repository.stcloudstate.edu/msia_etds/48/, [Accessed July 31, 2019]

[2] D.Lane, “Percentiles,” *Openstax CNX,* 20, Sep., 2010. Available: https://cnx.org/contents/223y7Xzw@12/Percentiles, [Accessed July 31, 2019]

블록체인에 특화된 개인정보영향평가 방안에 관한 연구 - 국내 개인정보보호법 및 허가형 블록체인을 중심으로 -

김경하*, 장남수*

*세종사이버대학교 정보보호대학원

A Study on Privacy Impact Assessment for Permissioned Blockchain

Kyung Ha Kim*, Nam Su Chang**

*Graduate school of Information Security, Sejong Cyber University.

요 약

최근 블록체인에 대한 관심이 집중되면서 정부 차원의 블록체인 시범서비스 등 블록체인 기반 서비스가 크게 증가하고 있다. 블록체인은 탈중개성, 무결성, 비가역성 등의 특성으로 인해 기존의 중앙 집중화된 시스템 및 서비스에 비해 보안성·신뢰성 측면에서 장점이 있는 것으로 알려져 있으나 개인키 유출, 스마트계약 프로그램의 취약점 악용 등 다양한 보안 이슈가 발생하고 있으며 개인정보의 삭제권·정정권 등 개인정보보호 관련 법규 준수가 어려운 문제점을 보유하고 있다. 본 연구에서는 국내 개인정보보호법 및 허가형 블록체인을 중심으로 개인정보보호 이슈를 도출하고 이를 사전에 예방할 수 있도록 블록체인에 특화된 개인정보 영향평가 방안을 제시하고자 한다.

I. 서론

블록체인(Blockchain)은 4차 산업혁명 시대에 있어 사물인터넷, 클라우드, 빅데이터, AI 등과 더불어 핵심 IT 기술로 거론되고 있으며, 아직은 성숙되지 않은 기술로서 실제 활용 사례는 많지 않으나 비트코인, 이더리움 등 암호화폐와 더불어 금융, 의료 등 다양한 산업 분야에서 널리 활용되어 기존에는 존재하지 않던 새로운 서비스를 가능하게 할 것으로 기대되고 있다. 이에 따라 국내에서도 많은 기업들이 블록체인 기술의 개발 및 도입을 적극 검토하고 있으며 과기정통부, 금융위원회 등 정부 부처에서도 블록체인 관련 다양한 시범사업을 추진하고 있다. 하지만, 블록체인은 그 기술의 특성상 개인정보보호 관련 법규와 충돌이 발생할 수 있으며 구성방식에 따라 안전성 확보가 어려울 수 있음에도 불구하고 블록체인 기술 및 블록체인 기반 서비스 개발시 개인정보 관련 침해요인에 대한 사전 고려가 미흡한 상황이며, 이러한 문제들을 사전에 예방하기 위해 개인정보 영향평가(PIA; Privacy Impact Assessment)를 수행하고자 하더라도 기존 개인정보 영향평가 수행안내서[1]에서 제시된 평가지표는 일반적인 형태의 개인정보처리시스템을 대상으로 작성된 것으로서 블록체인과 같은 신기술의 특성을 반영하지 못하고 있어 활용에 어려움이 존재한다. 본 논문에서는 블록체인의 개인정보보호 이슈에 대한 분석을 기반으로 블록체인 환경에 특화된 개인정보 영향평가 방안을 제안하고자 한다.

II. 블록체인의 개인정보보호 이슈

블록체인은 유형이 매우 다양하고 아직은 미성숙된 기술로서 새로운 기술이 계속 나오고 있어 단정적으로 일반화하기는 어려우며, 블록체인에서 처리되는 데이터, 특히 해시값에 대한 개인정보 여부 및 공개형 블록체인에서 개인정보처리자 지정 가능 여부 등 개인정보보호 관련 법규를 적용함에 있어 명확하게 정의하기 어려운 부분이 다수 존재한다[2,3,4]. 다만 공개형 블록체인과는 달리 허가형 블록체인의 경우

개인정보처리자가 특정되고 다른 데이터베이스 등과 결합하여 개인이 식별될 가능성이 높아 개인정보보호 관련 법규의 직접적인 적용 대상이 될 수 있다. 따라서 허가형 블록체인을 구현하고자 하는 경우에는 개인정보 영향평가를 통해 설계 단계에서부터 개인정보보호 관련 컴플라이언스 요건을 준수할 수 있도록 할 필요가 있다. 국내 개인정보보호법을 기준으로 주요 쟁점사항들을 아래 [표1]과 같이 정리할 수 있다.

[표 1] 허가형 블록체인 개인정보보호 주요 이슈

No	주요 이슈	설명
1	블록체인 데이터(해시값)의 개인정보 여부	- 블록체인 외부에 저장된 개인정보와 용이하게 결합 가능하다면 개인정보에 해당
2	개인정보처리자 지정 여부	- 컨소시엄 블록체인의 경우 컴소시엄 참여자가 개인정보처리자가 될 수 있음 - 프라이빗 블록체인의 경우 운영자가 개인정보처리자가 됨
3	개인정보처리 시스템 식별 여부	- 블록체인 서비스를 구성하는 요소(블록체인 노드, DApp. 데이터베이스 등)는 모두 해당 - 개인정보의 안전성 확보조치 기준 준수 및 블록체인에 특화된 보안위협 대응 필요
4	개인정보 파기	- 블록체인 특성상 블록체인내 데이터 파기 어려움 - 오프체인 스토리지, 블랙리스팅, 하드포크 등 방안 고려 필요
5	정정권·삭제권 등 정보주체 권리보장	- 블록체인 특성상 정정권 및 삭제권 보장 어려움 - 정정권·삭제권 요구대상 정보는 블록체인에 직접적으로 기록하지 않도록 설계 필요
6	개인정보 제3자 제공	- 컨소시엄 블록체인의 경우 블록체인 동기화 과정에서 제3자 제공 발생 - 신규 컨소시엄 참여자 발생 시 기존 정보주체에 대한 재동의 이슈 대응방안 고려 필요
7	기타	- 정보주체 동의 시 블록체인 관련 사항 고지 - 국외 이전, 민감정보 처리 등 블록체인 구성 및 데이터 유형별 조치사항 검토 필요

III. 허가형 블록체인에 대한 개인정보 영향평가 방안

3.1 개인정보 영향평가 개요

개인정보 영향평가(PIA)란 개인정보 처리로 인한 잠재적인 개인정보 침해 발생 가능성 및 그 영향을 사전에 조사·예측·검토하여 개선방안을 마련하는 체계적인 절차를 말한다. 국내의 경우 개인정보보호법 제33조 및 동법 시행령 제35조에 따라 공공기관이 개인정보파일을 구축·운용함에 있어 민감정보 또는 고유식별정보를 5만명 이상 처리하거나, 내·외부시스템과 연계를 통해 50만명 이상을 처리하거나, 100만명 이상 정보주체의 개인정보를 처리하려는 경우에는 해당 개인정보처리시스템의 설계 완료 전에 행정안전부장관이 지정한 영향평가기관을 통해 영향평가를 수행하고 그 결과를 행정안전부장관에게 제출하도록 규정하고 있다. 이러한 개인정보 영향평가는 사전 준비, 영향평가 수행, 이행의 총 3단계로 구성된다. 사전 준비단계에서는 영향평가 사업계획을 수립하여 예산을 확보하여 평가기관을 선정하고, 영향평가 수행 단계에서는 평가기관이 개인정보 침해요인을 분석하고 개선계획을 수립하여 영향평가서를 작성하며, 이행 단계에서는 영향평가서의 침해요인에 대한 개선계획이 반영되는가를 점검한다.

3.2 블록체인에 대한 개인정보 영향평가 절차별 고려사항

①개인정보 영향평가 필요성 검토

블록체인의 경우 우선 블록체인에서 처리되는 데이터가 개인정보에 해당되는지를 우선 검토하여야 한다. 만약 개인정보를 처리하지 않는다면 개인정보 영향평가가 불필요할 수 있다. 반대로 개인정보를 처리하는 것으로 판단된다면 일반적인 개인정보처리시스템과 동일한 기준이 적용되므로 공공기관의 경우 블록체인 상에서 처리되는 개인정보의 건수를 예측하여 법적 의무조건에 해당하는지 여부를 우선 판단하여야 한다. 다만 법적 의무조건에 해당되지 않더라도 블록체인 기술의 특성상 개인정보보호

관련 법규 위반 및 침해 리스크가 높기 때문에 개인정보보호 내재화(Privacy by design) 관점에서 개인정보 영향평가의 수행을 적극 고려할 필요가 있다.

②개인정보 처리현황 및 흐름분석

블록체인을 기반으로 처리되는 개인정보 항목을 식별하고 관련 업무 흐름을 파악하여 개인정보 취급업무표 및 업무흐름도를 작성한다. 이후 이를 바탕으로 개인정보의 수집, 보유, 이용·제공, 파기의 개인정보 처리단계별 흐름을 파악하여 개인정보흐름표 및 개인정보흐름도를 작성한다. 마지막으로 블록체인 노드, 스마트계약 등을 포함하여 블록체인 서비스를 구성하는 개인정보처리시스템, 정보보호시스템, 연계시스템 등의 인프라 구성을 파악하여 정보시스템 구조도 및 정보보호 현황표를 작성한다. 블록체인 기술이 활용되는 경우 블록체인 처리환경 및 데이터 구조 분석을 통해 블록체인에 저장되는 개인정보 항목을 식별하고 개인정보 처리업무표 및 개인정보 흐름도 작성시 해당 사항을 포함한다. 또한, 블록체인에 적용된 보안기술 및 블록체인 노드 등 관련 구성요소를 모두 포함하여 정보시스템 구조도를 작성한다.

[표 2] 블록체인 처리환경 및 흐름 분석시 확인사항

구분	주요 확인 사항
블록체인 기술현황	- 블록체인 유형 및 거버넌스 : 공개형, 허가형(컨소시엄 또는 프라이빗) 등 - 플랫폼 기술 : 하이퍼렛저 패브릭 등 - 합의 알고리즘: PBFT, PoA 등
블록체인 데이터구조	- 블록체인 헤더 및 바디 구조 - 블록체인 주소값 생성 매커니즘 - 상태DB 등 데이터베이스 구조 - 블록체인 내 개인정보 항목 : 식별자, 속성자, 가명정보(해시값) 등
블록체인 아키텍처	- 블록체인 노드 구성 : 노드수 및 노드별 운영주체, 논리적·물리적 구성 - 블록체인 네트워크 구성 - 블록체인 보안 대책 : 암호화, 인증, 접근권한, 접근통제 등
블록체인 서비스 및 개인정보 흐름	- 블록체인 서비스 현황 : 목적, 기능, 메뉴 구성, 정보주체 및 취급자 유형 등 - 블록체인 업무흐름 : 가입, 이용, 탈퇴 등 - 블록체인 데이터 흐름 : 수집, 저장, 이용, 제공·동기화, 파기 등

③개인정보 침해요인 분석

개인정보 침해요인 분석단계에서는 개인정보 흐름분석 결과를 바탕으로 평가항목 작성, 개인정보 보호조치 현황파악, 개인정보 침해요인 도출 및 개인정보 위험도 산정의 순으로 진행된다. 평가항목은 '개인정보 영향평가에 관한 고시' 별표4 및 '개인정보 영향평가 수행안내서[1]'에 제시된 85개 평가지표를 기반으로 침해사고 사례, 법제도의 변화, 대상 기관 및 대상 시스템의 특성 등에 따라 추가·삭제·변경 등 탄력적으로 구성하여 사용하며, 특히 블록체인과 같이 특정 IT 기술을 사용할 경우에는 특화된 평가항목을 개발하여 평가를 수행하여야 한다. 블록체인 기술 및 구조 등에 따라 상이할 수 있으나, 일반적으로 활용할 수 있는 블록체인 특화 평가항목은 아래의 [표3]과 같다.

[표 3] 블록체인 특화 평가항목 예시

평가영역	블록체인 특화 평가항목
대상시스템 개인정보보호 관리체계	- 개인정보 처리방침에서 블록체인 서비스의 구조 및 정보주체가 반드시 알아야 할 사항에 대해 이해하기 쉽게 공개하도록 계획하고 있는가?
개인정보 처리단계별 보호조치	- 블록체인(헤더 및 바디) 상에 해시값 외에 직접적으로 개인을 식별할 수 있는 개인정보(이름, 이메일 등)를 저장하지 않도록 설계하고 있는가? - 개인정보 수집 및 제공 동의 시 블록체인에서 처리되는 개인정보(해시값 등) 항목에 대해 명확히 고지하고 동의를 받도록 계획하고 있는가? - 컨소시엄 블록체인 참여자간의 개인정보보호 관련 역할 및 책임을 정의하고 이를 계약 등에 반영하는가? - 개인정보 파기 및 정정권·삭제권 관련 리스크를 최소화 할 수 있도록 설계하고 있는가?
대상시스템의 기술적 보호조치	- 개인키(정보주체, 노드)의 분실, 도난, 훼손 등에 대한 적절한 대책이 계획되어 있는가? - 블록체인 S/W 및 스마트계약 코드에 대한 취약점 점검 및 보안패치 방안이 계획되어 있는가?
특정IT기술 개인정보 보호	- 블록체인 데이터에 대한 프라이버시 보호 및 기밀성 확보방안은 적절한가? - 블록체인 데이터에 대한 무결성 확보 방안(노드 수 및 구성 등)은 적절한가?

④개선계획 수립

개선계획 수립 단계에서는 개인정보 침해요인별 위험도 분석에 기반하여 위험요소를 최소화하기 위한 개선방안 및 개선계획을 수립한다. 블록체인은 아직 미성숙한 기술로서 그 한계를 극복하기 위한 신기술들이 계속적으로 나타나고 있으므로 개선계획 수립 시점의 최신 기술 동향을 파악하여 적절한 대안을 마련한다.

IV. 결론

지금까지 본 연구에서는 블록체인과 관련된 개인정보보호 이슈 사항을 종합적으로 도출하고, 국내 개인정보 보호법을 중심으로 허가형 블록체인 개발 시 개인정보 침해를 사전에 예측·예방할 수 있도록 블록체인에 특화된 개인정보 영향평가 방안을 제안하였다. 추가 연구를 통해서는 개인정보 보호법 외에 정보통신망법을 함께 고려하여 블록체인 시범사업 등 실제 현장에 적용할 수 있도록 상세한 수준의 블록체인 특화 개인정보영향평가 방안 및 평가항목을 제시하고자 한다. 또한 EU GDPR 등 글로벌 컴플라이언스 환경에서 적용 가능한 개인정보영향평가 방안에 대해 연구를 진행할 예정이다.

[참고문헌]

[1] 행정안전부·한국인터넷진흥원, “개인정보 영향평가 수행안내서”, 2018.04

[2] 박민정, 채상미, 이명준, “개인정보보호법제 관점에서 본 블록체인의 법적 쟁점 GDPR 및 국내 개인정보보호법을 바탕으로”, J. Inf. Technol. Appl. Manag. 25(2), pp.133-146, June 2018

[3] 법무법인(유한) 광장, “블록체인 기술이 개인정보보호에 미치는 영향에 관한 연구″, 개인정보보호위원회, 2018.10

[4] 박진아, “블록체인과 개인정보보호 - 블록체인 매직(Magic)과 법적 도전”, 영남법학 제47, pp.199-237, 2018.12

[5] The European Union Blockchain Observatory and Forum, “Blockchain and the GDPR”, 2018.10

[6] 성신여대산학협력단, “블록체인기술 금융분야 도입방안을 위한 연구”, 금융위원회, 2016.06

[7] 상명여대산학협력단, “정보통신서비스 분야에서 블록체인 활용에 따른 개인정보보호 이슈 연구”, 한국인터넷진흥원, 2018.12

[8] 이대희, “블록체인 기술과 개인정보 쟁점”, 정보법학 제22권 제3호, 2018.12

[9] 최용혁, “블록체인 기술 적용과 개인정보 삭제 및 제3자 제공의 법적 문제에 관한 연구”, 고려대학교 정보보호대학원 석사학위논문, 2018.12

[10] 조수영, “개인정보보호법과 EU의 GDPR에서의 프라이버시 보호에 관한 연구”, 법학논고 61, pp. 117-146, 2018.04

[11] 장시영, “개인정보 영향평가 제도의 국·내외 현황 비교 및 시사점 분석”, 정보통신방송정책 제30권 14호 통권 647호, 2018.08

[12] WP29, “Guidelines on Data Protection Impact Assessment (DPIA) and determining whether processing is ‘likely to result in a high risk’ for the purposes of Regulation“, 2016.

zk-SNARKs기반 블록체인 익명 인증 아키텍처

강원태*, 김유나, 김호원*

*부산대학교 전기전자컴퓨터공학부, 부산대학교 사물인터넷 연구센터

wontae@islab.re.kr, yuna@islab.re.kr, howonkim@pusan.ac.kr

zk-SNARK based Anonymous Authentication Architecture on Blockchain

Won-Tae Kang*, Yuna Kim*, Ho-Won Kim*

*Department of Electrical and Computer Engineering,
Pusan National University.

요 약

4차 산업혁명의 핵심기술인 IoT, 클라우드, 빅 데이터, AI 기술 개발에 있어서 데이터는 필수 요소로 취급되는데, 이는 데이터의 품질 및 양이 기술완성도를 결정하기 때문이다. 이로 인하여 사회 전반적으로 무분별한 데이터 수집 및 공유가 일어나고 이로 인해 데이터 프라이버시를 침해되는 다양한 문제들이 발생하고 있다. 본 연구에서는 영지식 증명 체계 중 하나인 zk-SNARK와 블록체인을 사용하여 익명 인증 및 개인 정보 보호 아키텍처를 제안한다.

I. 서론

현대 사회에서 데이터는 필수 불가결 한 요소가 되었다. 여행, 음식, 문화 및 비즈니스를 포함한 모든 분야에서 데이터를 수집하고 활용하며, 개인 정보를 제공하고 맞춤형 서비스를 받는 것이 당연한 사회에 살고 있다. 이러한 맞춤형 서비스 제공업체가 늘어나게 되면서 발생하는 문제점이 있는데 그것은 해당 서비스이용자가 개인정보 유출의 위험에 처하게 된다는 것이다.

앞에서 언급했듯이 데이터는 중요한 자산이 되었고, 이로 인해 개인정보를 탈취하기 위한 시도가 늘어나고 있는 추세이다. 또한 소규모 서비스 제공업체부터 페이스북, 구글 등 대형 IT업체까지 개인정보를 목적으로한 사이버 공격으로 인한 대량의 개인정보 유출 사례역시 늘어나고 있다. 여기서 주목해야 할 점은 사이버 공격을 통해서만 개인 정보를 유출되는 것이 아니라는 점이다. 이용자의 개인정보를 수집하는 서비스 제공업체도 외부로 개인정보를 유출하거나, 본인 동의 없이 사업을 위한 마케팅 요소로 활용하는 등 이용자의 프라이버시를 침해하는 위협 요소가 되고 있다.[1] 본 연구에서는 이 문제를 해결하기 위한 블록체인 기반의 익명 인증 체계를 제안한다. 블록체인은 외부 공격에 의한 피해를 최소화하기 위해 개인정보에 대한 관리 주체를 서비스 제공자에서 개인으로 변경하고, 영지식 프로토콜인 zk-SNARK를 통해 노출되는 개인정보를 최소화하여 사용자에 대해 안전한 데이터 프라이버시를 보장한다.

II. 배경지식

2.1 zk-SNARKs

zk-SNARK (zero-knowledge Succinct Non-interactive ARguments of Knowledge)는 증명자가 자체 정보를 공개하지 않으며, 증명

자와 검증 자 간의 온라인 상호 작용없이 증명자가 정보의 소유자임을 증명하는 ZKP (Zero-Knowledge Proof)의 한 유형이다.

zk-SNARK에는 ZKP (Completeness, Soundness 및 zero-knowledge)의 세 가지 속성과 간결함과 비대화 속성이 있다. '간결함'은 검증자가 증명자가 제시 한 증거를 간략하게 확인할 수 있음을 의미하며 '비 대화식'은 검증자가 증명자가 제시한 증거를 상호 작용없이 검증을 진행함을 의미한다.

zk-SNARK는 이러한 특성을 만족시키기 위해 Homomorphic Hiding 및 이차 산술 프로그램 (QAP)을 사용하여 피노키오 프로토콜을 도입했습니다. [4][6]

2.2 Homomorphic Hiding

Homomorphic Hiding은 zk-SNARK에서 사용하는 정보은닉 기법으로 다음과 같은 특징이 있다[2]:

- Homomorphic Hiding E(x)에서 x를 유추하기 어렵다
- $x \neq y$ then $E(x) \neq E(y)$
- E(x), E(y)를 이용해 E(x + y)를 계산할 수 있다

위의 특성을 구현하기 위해 Homomorphic Hiding은 동형암호화 및 타원곡선 페어링 (Elliptic curve paring)을 사용했다.

첫 번째로 동형암호에는 숨겨진 비밀 정보로 계산할 수 있는 특징이 있는데 이 특징을 사용하면 암호화 된 값을 계산하여 프로토콜을 수행 할 수 있다. [2] [3]

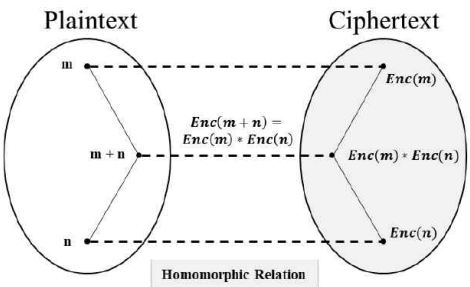


그림 1 동형암호 예시

두 번째는 비밀정보(x)를 숨기기 위해 사용되는 타원곡선 페어링이다. 타원곡선 페어링은 다음 특성을 사용하여 비밀 정보를 숨긴다.

- 점 p와 G가 타원곡선 상에 있는 점이고 G를 알려진 점이라고 가정할 때 점 P = p * G(p는 0이 아닌 상수)에서 p를 찾기 어렵다 (이산로그 문제).
- 타원곡선 페어링의 Bilinearity 특성은 다음과 같이 표현할 수 있다.
  - $e(A, B+C)=e(A,B) \cdot e(A,C)$
  - $e(A+B, C)=e(A,C) \cdot e(B,C)$
  - $e(nA, B)=e(A,nB)=e(A,B)^n$

위의 속성을 사용하면 타원 곡선에서 비밀 정보를 숨길 수 있다.[5]

2.3 Pinocchio Protocol

피노키오 프로토콜은 zk-SNARK의 실용적인 프로토콜으로 숨기고자 하는 값을 암호화하고 암호화된 값을 계산하여 검증을 수행 할 수 있다. 이 프로토콜은 검증을 위한 키 페어를 생성하는 KeyGen단계(Proving key, Verification key), Proving 키를 통해 증명을 생성하는 Prove 단계 및 Verification 키를 통해 증명을 검증하는 Verify 단계로 구성된다. 피노키오 프로토콜은 zk-SNARK에서 증명자가 검증자에게 비밀정보를 공유하지 않고 증명자가 비밀 정보를 알고 있는지 검증자가 확인할 수 있도록 한다.[7]

이 프로토콜을 수행하기 위해서는 공개 입력 값과 비밀입력 값을 지정해야 된다. 공개 입력 값은 모든 사람에게 공개되며 비밀입력 값은 Prover만 아는 비밀정보(증인 가치)이고 이를 통해서 상기 언급되었던 3가지 단계를 거쳐 프로토콜이 수행된다.[4][8]

2.4 Hyperledger Fabric

Hyperledger는 2015년 12월 Linux Foundation에서 시작한 오픈소스 블록체인 프로젝트이다. Fabric은 Hyperledger 프로젝트 중 기업을 위한 범용 블록체인 프레임 워크이며 멤버십 서비스를 기반으로 허가/프라이빗 구조

를 가지고 있다. 블록체인 네트워크를 구성하는 노드는 Peer(Endorser, Committer), Orderer, Fabric-CA 등으로 나뉘며 블록체인 네트워크에서 Golang, Nodejs 등의 프로그램언어를 이용해 비즈니스 로직을 구현할 수 있는 체인 코드가 있으며 스마트컨트랙트와 유사한 역할을 수행한다.

체인코드와 원장은 채널에서 공유되며 한 피어가 다른 채널에 참여하더라도 채널은 서로 의존하지 않고 독립적으로 작동한다.[4]

본 연구에서는 3장에서 Hyperledger 패브릭과 zk-SNARK, Pinocchio Protocol을 사용하여 익명 인증을 제시한다.

III. 본론

3.1 zk-SNARK기반 블록체인 아키텍처

본 항목에서는 컨소시엄 블록체인에서 zk-SNARK를 이용한 개인정보보호 인증 아키텍처를 제안하며 시스템의 개요는 아래와 같다.

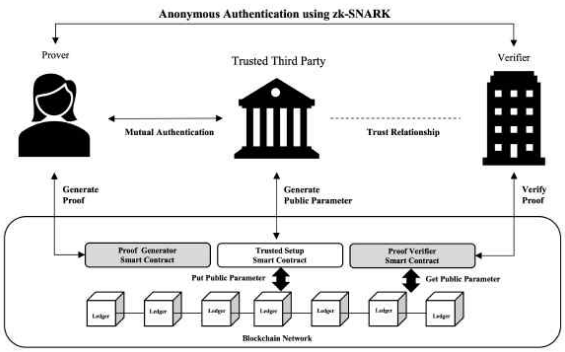


그림 2 익명인증을 위한 시스템 아키텍처

그림 2에 나와있는 시스템 아키텍처는 TTP(Trusted Third Party)를 통해 증명자(Prover)가 블록체인 네트워크의 검증자(Verifier)에 입증하고자하는 정보를 기록하는데 이 때 블록체인 네트워크의 분산 원장에 기록 된 정보에는 사용자를 식별 할 수 있는 정보가 포함되어 있지 않다.

오직 증명자만이 분산원장에 기록 된 정보의 소유권을 증명할 수 있는데, 이는 증명자만이 정보의 증거를 생성 할 수 있음을 의미한다.

3.2 시스템 프로세스

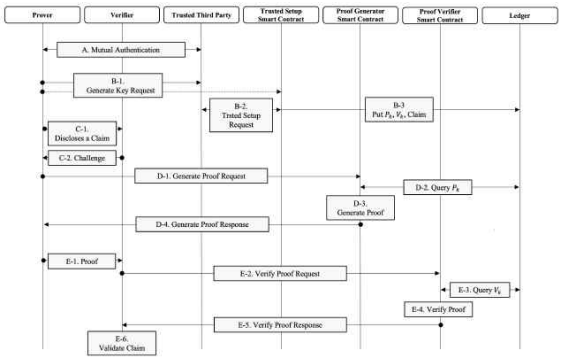


그림 3 익명인증 시퀀스 다이어그램

- Mutual Authentication

증명자(Prover)가 개인 또는 특정정보의 소유자임을 증명하는 프로세스로 TTP(Trusted Third Party)가 증명자의 정보를 확인할 수 있다고 가정하고 TTP가 제시한 상호인증 방법을 통해 상호인증을 진행한다.

- Trusted Setup

TTP는 증명자 정보에 대한 증명과 키 페어를 생성한다. (B-1)은 키 페어를 생성하기 위해 다항식의 제약조건을 TTP에 전송하고 TTP를 통해 증명자의 클레임을 전송하는 단계이며 (B-3)는 생성된 키 페어와 클레임을 분산원장에 저장하는 단계이다.

- Claim to Verifier

검증자(Verifier)는 서비스를 제공하거나 증명자의 특정 정보를 확인하는 엔티티(Entity)로 수행하는 프로세스를 살펴보면, (C-1) 단계에서 증명자는 먼저 분산원장에 저장된 클레임의 소유권을 검증자에게 명시하고 (C-2) 단계에서 검증자는 임의의 값(Challenge)를 생성하여 증명자에게 보내고 검증자가 소유한 다항식의 입력 값에 할당된다.

- Generate Proof

증명자는 자신이 클레임의 소유자임을 증명하는 증거(Proof)를 생성하는데 세부 프로세스를 살펴보면, (D-1)에서 PGSC(Proof Generator Smart Contract)를 호출하여 증명을 생성하는데 이때 증명자가 소유한 다항식의 제약사항과

검증자가 전달한 임의의 값이 입력으로 사용된다. (D-2)에서 PGSC는 분산원장에 저장된 Proving Key를 가져오고 (D-3)에서는 (D-1)에서 입력받은 값을 통해 증명을 생성한다. 만들어진 증명은 (D-4) 단계를 통해 증명자에게 전달된다.

- Verify Proof

증명자는 생성된 증명과 다항식의 결과 값을 검증자에게 보내는 (E-1)단계를 수행하고 (E-2)단계에서 검증자는 증명자에게 전달받은 값과 임의의 값을 PVSC(Proof Verifier Smart Contract)에 전달한다. (E-3)에서 분산원장에 저장되어 있는 Verification Key를 가져오고 (E-4)에서 Verification Key와 증명, 임의의 값 및 다항식 출력 값을 통해서 zk-SNARK 기반 검증 로직을 통해 증명자가 임의의 값을 이용하여 출력 값을 산출했는지 검증하고 (E-5)에서 그 결과를 검증자에게 전송하고 최종적으로 (E-6)에서 검증자는 검증결과를 기반으로 증명자에 대한 클레임의 진위여부를 판단한다.

IV. 결론 및 향후연구

본 논문에서는 블록체인과 zk-SNARK를 사용하여 개인정보 보호 아키텍처를 제안했다. 해당 아키텍처를 통해 사용자는 자신의 식별정보는 숨기고 최소한의 정보만을 노출하여 특정 정보를 증명할 수 있으며, 서비스 제공자의 경우 본 아키텍처의 기반이 엔터프라이즈 형 블록체인인 하이퍼레저 패브릭이기 때문에 운영적 측면에서 타 플랫폼 대비 좋은 효율을 기대할 수 있다. 하지만 기술상용화 관점에서 봤을 때 보다 면밀한 성능테스트와 그에 따른 개선사항이 필요하기 때문에 이와 관련된 연구를 지속적으로 수행할 것이다.

Acknowledgement

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구)

[참고문헌]

[1] Microsoft Homepage, https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE2DjfY, last accessed 2019.

[2] Maksym Petkus, Why and How zk-SNARK Works: Definitive Explanation, arXiv:1906.07221 [cs.CR], 2019.

[3] ELETRIC COIN, Homepage, https://electriccoin.co/blog/snark-explain/, last accessed 2019.

[4] Hyperledger Fabric Homepage, https://hyperledger-fabric.readthedocs.io/en/release-1.4/, last accessed 2019.

[5] Jan Camenisch, Anna Lysyanskaya, Signature Schemes and Anonymous Credentials from Bilinear Maps, Advances in Cryptology - CRYPTO 2004 pp 56-72, 2004.

[6] Jan Camenisch, Manu Drijvers and Anja Lehmann, Anonymous Attestation Using the Strong Diffie Hellman Assumption Revisited, Trust 2016: Trust and Trustworthy Computing pp 1-20, 2016.

[7] Okamoto, T.: Efficient Blind and Partially Blind Signatures Without Random Oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 80 - 99. Springer, Heidelberg, 2006.

[8] Bryan Parno, Jon Howell, Craig Gentry and Mariana Raykova, Pinocchio: Nearly Practical Verifiable Computation, 2013 IEEE Symposium on Security and Privacy, 2013.

Permissioned 블록체인 환경에서의 안전한 키 생성에 대한 연구+

김태훈*, 이임영**

순천향대학교 컴퓨터학과

A study on Secure Key Generation in Permissioned Blockchain Environment

Tae-Hoon Kim*, Im-Yeong Lee**

Dept of Computer Science and Engineering, Soonchunhyang University*,**

요 약

일반적인 블록체인 키 생성 방법은 비결정적 지갑 생성, 결정적 지갑 생성, 계층 결정적 지갑 생성으로 나뉜다. 이 중 비결정적 지갑은 모든 키의 복사본을 주기적으로 백업을 해야 한다는 단점을 가진다. 이를 해결하기 위하여 계층 결정적 지갑이 등장했다. 계층 결정적 지갑은 비트코인의 BIP32 표준으로 정의되어 현재 가장 많이 사용되는 기술이며, 공격자는 부모 공개키와 자식 개인키를 획득하면 해당 부모에서 파생된 키들을 쉽게 유출할 수 있으므로, 부모 공개키에서 자식 공개키를 파생하는 기능을 제거했다. 또한, 블록체인 사용자들끼리 거래를 이용하기 위하여 트랜잭션을 빈번하게 사용할 때마다 동일한 주소가 블록에 계속 저장된다. 이로 인해 주소의 프라이버시를 유추할 수 있고, 침해될 가능성 문제를 제시한다.

본 논문에서는 이를 해결하기 위하여 BIP32에서 제거된 기능을 유지하며, 권한 상승 공격을 방지한다. 또한, 블록체인 사용자의 프라이버시를 위해 일회용 주소를 사용함으로써 사용자 프라이버시에 대한 익명성 보안이 강화된 Permissioned 블록체인 환경에서의 안전한 키 생성에 대한 연구를 제안한다.

I. 서론

2009년 블록체인을 최초로 이용한 암호 화폐 시스템인 비트코인이 출시된 이후로, 블록체인에 대한 관심이 높아졌다. 블록체인은 암호 화폐에서 많은 분야로 확장되어 기술을 적용되고 있다. 대부분의 암호 화폐는 Public 블록체인 환경과 기업과 같은 환경에서 사용되는 Permissioned 블록체인 환경으로 나눌 수 있다. 이러한 블록체인 사용자는 개인키 및 개인 정보 유출을 최소화하기 위해 모든 개인키, 공개키 키 쌍을 자주 갱신하여 사용해야 하며, 개인키 관리는 매우 중요하다. 따라서 각 트랜잭션마다 새로운 키 쌍을 생성하는 것이 일반적이며, 이러한 키 쌍을 생성하고 관리하는 소프트웨어 프로그램을 '지갑'이라고 일컫는다.

블록체인 키 생성에는 비결정적 지갑 생성, 결정적 지갑 생성, 계층 결정적 지갑 생성으로 나뉜다. 비결정적 지갑은 개인키을 무작위로 생성한 후 거래가 필요할 때마다 개인키에 대응되는 공개키를 생성하고 주소로 변환하여 사용한다. 그러나, 비결정적 지갑은 모든 키의 복사본을 주기적으로 백업을 해야 하는 단점을 가진다. 이를 해결하기 위하여 계층 결정적 지갑이 등장했다. 계층 결정적 지갑은 비트코인의 BIP32 표준으로 정의되어 현재 가장 많이 사용되는 지갑이다[1]. 하지만, 계층 결정적 지갑에도 문제가 발견되었다. 기존의 모든 계층 결정적 지갑에서 공격자는 부모 공개키와 자식 개인키를 획득하면 해당 부모에서 파생된 키들을 쉽게 유출할 수 있는 권한 상승 공격 문제를 발견하였다[2]. 이를 해결하기 위해 개선된 계층

+ 이 논문은 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(No. NRF-2019R1A2C1085718) 및 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2019-2015-0-00403)

결정적 지갑의 큰 장점인 부모 공개키에서 자식 공개키를 파생하는 기능을 제거했다. 하지만 트랜잭션들을 발생시킬 때마다 이용되는 주소가 많은 거래로 인하여 대상을 명확하게 특정할 수 없지만, 그 주소를 사용하는 사람이 무엇을 했었는지에 대한 트랜잭션을 수집하여 프라이버시 침해 문제를 발생시킬 수 있다.

본 논문에서는 이를 해결하기 위하여 BIP32에서 제거된 기능을 살리며, 권한 상승 공격을 방지한다. 또한, 블록체인 사용자의 프라이버시 보호를 위해 일회용 주소를 사용함으로써 사용자 프라이버시에 대한 익명성 보안을 강화하는 Permissioned 블록체인 환경에서의 안전한 키 생성 제안 방식을 제안한다.

II. 관련 연구

2.1 Permissioned 블록체인

블록체인은 크게 Public, Private, Consortium으로 나뉜다. 그리고 세부적으로 Permissioned, Permissionless로 다시 나뉜다. Permissioned 블록체인은 누구나 접근할 수 없으며, 허가를 받은 사람만 접근할 수 있다. 이러한 특징으로 Permissioned 블록체인은 회사 측면에서 주로 사용하는 블록체인 모든 정보를 블록으로 처리하지 않으며, 이로 인해 블록 생성의 보상으로 코인을 지급하는 Public 블록체인과 차별성을 갖는다. 결과적으로 Permissionless 블록체인 보다 탈중앙의 성격이 약한 대신에, 보안과 속도에는 우수한 특징을 가진다.

2.2 BIP32

BIP(Bitcoin Improvement Proposal)는 비트코인의 기술을 논의하기 위해 작성하거나 공개된 제안 양식이며, BIP32는 계층 결정적 지갑(Hierarchical Deterministic Wallet, HD)의 일반적인 형식과 계층 결정적 지갑을 구축하는 방법을 설명한 문서이다. BIP32는 하나의 비트코인 주소보다 여러개의 비트코인 주소를 가지고 입출금을 하면 복수의 주소를 사용하는 사용자는 하나의 주소만을 사용하는 사람보다 블록체인을 감시하는 것을 더 어렵게하기 때문에 익명성에 도움이 된다. BIP32는 처음 계층 결정적 지갑에 대한 구조를 제안했는데, 계층 결정적 지갑은 BIP32 문서를 코드로 프로그래밍화한 것을 말한다. 계층 결정적 지갑의 특징은 트리 구조에서 생성된 키를 저장하는 방식이며, 부모 키가 자식키를 파생시킬 수 있는 구조이다. 결정적 키 생성 기법과는 다르게 여러 개의 파생 구조를 가지고 있어, 중간 계층의 개인키가 유출되더라도 다른 가지의 파생된 키에는 별다른 영향을 끼치지 않아 보안에 유리하다.

2.3 Cryptonote

크립토노트(Cryptonote)란 송신자를 식별하지 못하도록 개인 정보 지향 디지털 통화에 사용되는 프로토콜이다[3]. 크립토노트의 첫 번째 버전은 2012년에 발표되었다. 백서 원본 작성자는 알 수 없으며, Nicolas van Saberhagen이라는 익명의 이름을 사용했다. 크립토노트의 특징으로는 추적 불가능한 트랜잭션 크립토노트는 링 서명이라는 방식을 사용하여 보내는 사람이 누군지 알 수 없도록 숨기며, 연결되지 않은 트랜잭션 보내는 사람이 누구인지 몰라도 받는 사람이 누구인지는 알 수 있다는 점이다. 즉, .코인 거래의 상황에서 공개키는 수신되기 전에 그룹으로 묶여 있기 때문에 송신자를 식별하는 것을 불가능하게 만든다. 또한, 크립토노트는 트랜잭션마다 한 번만 사용하는 일회용 주소를 만들어 낸다. 이 주소는 받는 사람의 공개키로부터 생성된 것으로, 오직 받는 사람만이 자신의 것인지 알 수 있다[4].

III. 보안 요구 사항

블록체인의 키 생성 방식에서는 다음과 같은 요구 사항을 만족해야 한다.

- 권한 상승 공격 방지 : 부모의 공개키와 파생된 자식 개인키가 유출되면 해당 부모에서 파생된 모든 자식 개인키를 알 수 있기 때문에 악의적인 공격자는 해당 부모에서 파생된 모든 자식 개인키를 유추할 수 없어야 한다.
- 서명 위조 : 부모가 자식 개인키를 이용하여

서명을 대신 생성하거나 위조할 수 없어야 한다.

- 익명성 : 어떤 행위를 한 사람이 누구인지 알 수 없어야 한다.

IV. 제안 방식

본 제안 방식에서는 Permissioned 블록체인 환경에서의 안전한 키 생성에 대하여 앞서 분석한 문제점을 개선하는 기법을 제안한다.

4.1 설정 단계

이 단계는 제안 방식에 필요한 시스템 계수들을 Master가 구성하는 단계로써 다음과 같이 진행한다.

- ***Step 1*** 64-Byte 크기의 seed를 생성한다.
- ***Step 2*** 생성한 seed를 이용하여 개인키와 공개키를 생성한다.
- ***Step 3*** 랜덤 값 c, d, k를 선택한다.
- ***Step 4*** 트랩도어 해시 키 쌍(트랩도어 키, 해시 함수 키)를 생성한다.
- ***Step 5*** 트랩 도어 함수와 난수, 메시지를 이용하여 e_0를 생성한다.
- ***Step 6*** $n = (c, d, k)$로 설정한다.

4.2 자식 공개키/개인키 생성 단계

이 단계는 상위의 존재인 Master 또는 Semi-Master가 자식 공개키, 개인키를 생성하는 단계이며, 다음과 같이 진행한다.

- ***Step 1*** 상위의 존재가 Master일 경우 Master 공개키와 체인코드, 인덱스를 이용하여 자식 또는 손자 공개키를 파생한다.
- ***Step 2*** 상위의 존재가 Semi-Master일 경우 부모 공개키와 체인코드, 인덱스를 이용하여 자식 공개키를 파생한다.
- ***Step 3*** 상위의 존재가 Master일 경우 Master 개인키와 체인코드, 인덱스를 이용하여 자식 공개키를 파생한다. (Semi-Master는 자식 개인키를 생성할 수 없다.)

4.3 일회용 주소 생성 단계

이 단계는 사용자가 지갑 외부 사용자와 트랜잭션으로 데이터를 주고 받을 때 필요한 일회용 주소를 생성하는 단계이며, 다음과 같이 진행한다.

- ***Step 1*** 송신자는 랜덤 값 r을 선택하여 임시 값 $R_i = g^r$을 생성한다.
- ***Step 2*** 송신자는 수신자의 공개키를 획득하면서 임시 값을 공유한다.
- ***Step 3*** 수신자의 공개키와 임시 값을 이용하여 임시 키 RK를 생성한다.
- ***Step 4*** 임시 키를 이용하여 수신자의 일회용 주소를 생성한다.
- ***Step 5*** 수신자는 임시 값과 개인키를 이용하여 일회용 주소 대응 키 값을 획득한다.
- ***Step 6*** 송신자는 데이터를 일회용 주소로 전송한다.

4.4 서명 생성 및 검증 단계

이 단계는 서명을 생성하는 단계로써, Master 또는 해당 사용자만 서명을 생성할 수 있으며, 다음과 같이 진행한다.

- ***Step 1*** Master는 트랩도어 해시 함수를 이용하여 $i-th$ 자식 트랩도어 해시 키를 생성한다.
- ***Step 2*** 서명 값과 출동하는 값을 연산하여 생성한다.
- ***Step 3*** Master는 하위의 존재가 Semi-Master일 경우 Master 트랩 키, 일부 매개변수, SemiMaster에 대한 서명 값을 전송한다.
- ***Step 4*** Master는 하위의 존재가 일반 사용자일 경우 $i-th$ 자식 트랩 키, 일부 매개변수, $i-th$ 자식에 대한 서명 값을 전송한다.
- ***Step 5*** 일반 사용자는 Master에게 서명 값을 받지 않고, 스스로 자신의 개인키로 서명을 생성할 수 있다.
- ***Step 6*** Master로부터 받은 서명 값은 트랩 키, 일부 매개변수, 서명 값을 이용하여 서명 검증이 가능하다.
- ***Step 7*** 사용자 스스로 생성한 서명 값은 대응되는 공개키를 이용하여 서명 검증이 가능하다.

4.5 추적 단계

이 단계는 신뢰받는 제 3자인 검증자가 일회용 주소를 사용하는 사용자들을 확인할 필요가 있을 때 사용되는 단계이며, 다음과 같이 진행한다.

- ***Step 1*** 검증자는 일회용 주소 사용자로부터 공개키, 임시 키를 요청한다.
- ***Step 2*** 일회용 주소 사용자는 검증자에게 자신의 공개키, 임시 키를 전송한다.
- ***Step 3*** 사용자의 공개키, 일회용 주소, 임시 키, 추적 키를 이용하여 새로운 일회용 주소를 생성한다,
- ***Step 4*** 기존의 일회용 주소와 새로 생성한 일회용 주소를 비교하여 일치하는지 비교한다.
- ***Step 5*** 일치할 경우 1을 반환하며, 일치하지 않을 경우 0을 반환한다.

V. 제안 방식 분석

- 권한 상승 공격 방지 : 부모의 공개키와 파생된 자식 개인키가 유출되면 해당 부모에서 파생된 모든 자식 개인키를 알 수 있다. 그렇기 때문에 Master를 제외한 Semi-Master는 자식 개인키를 생성할 수 없게 하여 부모 공개키와 자식 개인키 연결성을 끊어 권한 상승 공격을 방지한다.
- 서명 위조 : 부모는 자식의 개인키를 모르기 때문에 자식 개인키를 이용하여 서명을 대신 생성할 수 없다.
- 익명성 : 블록체인 사용자의 프라이버시를 위해 지갑 외부 사용자와 트랜잭션으로 데이터를 주고 받을 때에는 일회용 주소를 사용하여 송수신하며, 사용자 프라이버시에 대한 익명성 보안을 더욱 강화시킨다..

VI. 결론

초기 비결정적 지갑은 개인키을 무작위로 생성한 후 거래가 필요할 때마다 개인키에 대응되는 공개키를 생성하고 주소로 변환하여 사용하였다. 그러나, 비결정적 지갑은 모든 키의 복사본을 주기적으로 저장 및 백업을 해야 한다는 단점을 가진다.

이를 해결하기 위하여 계층 결정적 지갑이 등장했지만 계층 결정적 지갑에도 문제가 발견되었다.. 기존의 모든 계층 결정적 지갑에서 공격자는 부모 공개키와 자식 개인키가 있으면 해당 부모에서 파생된 키들을 쉽게 유출할 수 있는 권한 상승 공격 문제를 발견하였다. 개선된 계층 결정적 지갑에서 부모 공개키에서 자식 공개키를 파생하는 기능을 제거했다.

또한, 많은 거래로 인하여 트랜잭션들을 발생시킬 때마다 이용되는 주소의 프라이버시 문제가 발생되었다. 본 논문에서는 이를 해결하기 위하여 BIP32에서 제거된 기능을 유지하며, 권한 상승 공격을 방지하기 위하여 Master를 제외한 Semi-Master는 자식 개인키를 생성할 수 없게 함으로써 부모 공개키와 자식 개인키 연결성을 끊어 권한 상승 공격을 방지한다. 또한, 블록체인 사용자의 프라이버시 보호를 위해 지갑 외부 사용자와 트랜잭션으로 데이터를 주고 받을 때에는 일회용 주소를 사용하여 송수신하며, 사용자 프라이버시에 대한 익명성 보안을 더욱 강화하는 Permissioned 블록체인 환경에서의 안전한 키 생성 제안 방식을 제안한다.

[참고문헌]

[1] Wuille, Pieter. "BIP32: Hierarchical Deterministic Wallets." h ttps://github. com/genjix/bips/blob/Master/bip-0032. md (2012).

[2] Fan, Chun-I., et al. "Secure hierarchical bitcoin wallet scheme against privilege escalation attacks." 2018 IEEE Conference on Dependable and Secure Computing (DSC). IEEE, 2018.

[3] Van Saberhagen, Nicolas. "CryptoNote v 2.0." (2013).

[4] Luo, Yili, et al. "A New One-time Address Scheme for Blockchain." 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC). IEEE, 2019.

블록체인 DID 동향 및 분석

이상현*, 김용수*, 김호원*

*부산대학교 전기전자컴퓨터공학과

jdsd2233@gmail.com, dkgoggog0329@gmail.com, howonkim@pusan.ac.kr

DID with Blockchain Trends and Analysis

Sang-Hyun Lee*, Yong-Su Kim*, Ho-Won Kim*

*Department of Electrical and Computer Engineering,
Pusan National University.

요 약

DID는 개인의 신원 정보를 개인이 통제할 수 있는 기술로 프라이버시 보호에 대한 관심이 높아지면서 주목 받고 있다. 탈중앙화 특성을 가진 블록체인이 떠오르면서 DID와 블록체인이 결합된 기술이 활발하게 연구되고 있는데, DID의 특성인 지속성, 탈중앙성, 휴대성, 프라이버시 보호는 블록체인과 결합되었을 때 긍정적인 요소가 향상된다. 이에 따라 다양한 방식의 블록체인 기반 DID 기술이 등장하고 있다. 본 논문에서는 DID에 대해 알아보고 블록체인과 결합된 DID 기술의 동향을 분석한다.

I. 서론

DID(Decentralized Identifier)는 신원 정보가 기존의 방식처럼 중앙집중 시스템에 의해 통제되지 않고 개인이 자신의 정보에 대한 통제권을 가지는 기술이다. DID를 사용하는 개인은 스스로 신원에 대한 관리, 신원 정보 제출 범위, 제출 대상 통제 등을 수행할 수 있다. DID에 블록체인을 도입하게 되면 블록체인의 장점인 데이터에 대한 투명성 및 무결성을 확보할 수 있게 된다. 따라서, 신원 정보에 대한 위변조가 불가능해진다.[1][2]

DID는 개인의 프라이버시 보호에 대한 관심이 많아짐과 더불어 서비스 제공자가 불필요하게 사용자 인증정보와 개인정보를 수집하는 사례가 증가하면서 관심이 높아졌다. 기존 시스템에서 서비스 제공자는 모든 정보를 중앙화하여 관리하기 때문에 공격으로 인한 개인정보 유출의 우려가 높다. 실제로 많은 시스템들에서 해킹으로 인한 개인정보 유출이 이루어지고 있는데, 대규모 시스템에서 개인정보 유출이 일어날 경우 피해 규모가 심각하게 커진다.

이러한 문제점을 해결하기 위해 국내외에서 블록체인과 DID를 결합한 기술이 활발하게 연구되고 있다. 본 논문에서는 국내외에서 진행중인 DID의 특징 및 동향에 대해 기술한다.

II. DID(Decentralized Identifier) 개요

DID는 중앙집중 시스템에서 가지지 않는 다양한 특징을 가지게 된다. 대표적으로 지속성, 탈중앙성, 휴대성, 프라이버시 보호가 있다.[3]

2.1 지속성

DID의 사용자는 외부 환경이 변하더라도 자신의 신원 정보를 계속해서 사용할 수 있다. 서비스 제공자에 의해 신원 정보가 관리되던 중앙형 시스템의 경우 서비스 제공자가 서비스를 중단할 경우 신원 정보가 유효하지 않게 되지

만 DID에서는 외부 환경과 무관하게 신원 정보의 유효성이 유지된다.

2.2 탈중앙성

신원 정보의 발행 및 검증이 서비스 제공자에게 종속되지 않고, 블록체인 참여 노드 기반으로 독립적으로 운영된다. 누구나 신원 정보를 생성하고 이용할 수 있으며 관리 또한 스스로 하게 된다. 탈중앙성을 띄고 있지만 최초에 사용자 신원 정보를 생성할 때 이를 검증할 수 있는 신뢰 기관이 필요하다.

2.3 휴대성

DID의 사용자는 스마트폰, 칩이 내장된 카드 등을 통해 스스로 신원 정보를 선택 후 제공할 수 있다. 이를 통해 필요한 경우, 신원 정보를 선택하여 서비스 제공자에게 제공할 수 있다.

2.4 프라이버시 보호

서비스 사용자는 DID 사용자의 동의 없이 신원 정보를 활용할 수 없다. 사용자는 신원 정보의 관리를 스스로 하기 때문에 서비스 제공자는 필요한 만큼의 신원 정보만 확인할 수 있다. 블록체인 기반의 DID의 경우 신원 정보를 블록체인 내에 저장할 경우 누구나 신원 정보를 확인할 수 있게 되기 때문에 블록체인 내에는 암호화하여 저장하거나, 블록체인 외부의 DB에 정보를 저장한다.

III. 국내외 DID 동향

2.1 국내 동향

2.1.1 메타디움(Metadium)[4]

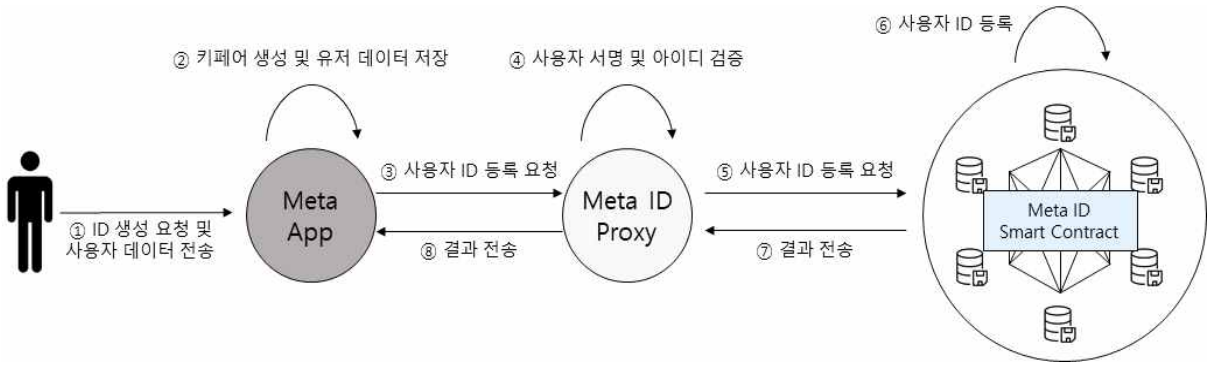
메타디움은 블록체인 기반의 신원 인증을 위한 프로토콜을 제공한다. 메타디움은 인터넷의 활성화에 따른 신원 정보의 과도한 사용으로 인한 해킹, 유출 등의 문제점의 해결책을 제시한다. 사용자는 메타디움을 이용해 아이디를 생성하고, 다양한 서비스에 키를 등록함으로써 하나의 메타 아이디(Meta ID)를 통해 서비스를 이용할 수 있다. 메타 ID를 통해 사용자가 자신의 ID를 관리하며 사용자의 동의가 있어야만 신원 정보를 제공한다.

메타 아이디는 사용자들에게 자신의 정보에 대한 소유권을 가질 수 있게해, 자기 주권(self-sovereign) 신원이 가능하게 한다. 메타 ID에 대한 생성 예시는 그림 1과 같다. 메타 아이디에는 생성 후 변경이 불가능한 고유한 값으로 사용자의 키, 메타프린트, 신원 검증을 위한 정보가 들어있다. 등록된 키를 가진 사용자는 신원을 증명할 수 있으며 정보를 추가, 변경, 삭제할 수 있다.

2.2 국외 동향

2.2.1 Sovrin[5]

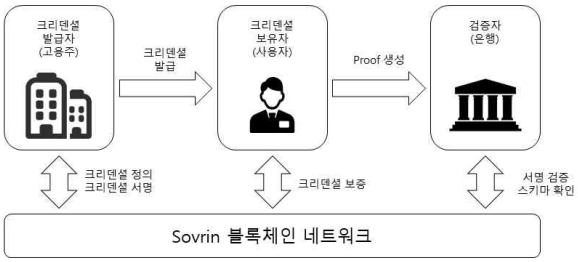
비영리 재단인 Sovrin에서 개발하는 플랫폼으로 디지털 신원 정보에 대해 사용자의 주권을 실현할 수 있도록 하는 퍼블릭 서비스 유틸리티이다. 재단은 Sovrin Trust Framework를 제정하고 이를 구현한다. 사용자는 신원 정보를 신원 정보 발행자에게 요청하고, 공개하고 싶은 만큼의 신원 정보를 선택해 이를 서비스 제공자에게 제시한다. 사용자의 신원 정보는 사용자



[그림 1] 메타디움 ID 생성 예시

의 모바일 단말에 저장되며 신원 증명이 필요한 경우 사용자의 개인키로 서명하여 제출한다.

Sovrin은 허가형 블록체인을 사용하기 때문에 신뢰할 수 있는 기관들에게만 블록체인에 데이터를 기록할 수 있는 권한을 부여하고 일반 사용자는 데이터 조회만 가능하다. Sovrin의 서비스 개요는 그림 2와과 같다. Sovrin에서의 인증서인 크리덴셜의 발급, 등록, 검증 등이 Sovrin 블록체인 네트워크를 통해서 이루어진다. 사용자는 이를 통해 자신의 신원을 인증할 수 있다.



[그림 2] Sovrin 서비스 개요

2.2.2 uPort[6]

이더리움 기반 소프트웨어 개발사인 컨센시스(ConsenSys)에서 개발한 플랫폼이다. 이더리움 네트워크 기반의 신원 증명 기술을 구현하였다. 플랫폼 내에는 사용자의 신원 확인을 위한 메시지 프로토콜, 증명서 발급, 어플리케이션, 전자지갑 등이 포함되어있다. uPort 플랫폼의 전체적인 구조는 그림 3과 같다.

uPort는 블록체인의 경량화와 공개키 갱신을 지원하는데, 경량화를 위해 블록체인에는 공개키의 해시 값을 저장하고, 공개 키는 별도의 분산형 파일 시스템(Inter Planetary File System, IPFS)에 저장한다. 그리고, 분산 아이디와 공개키를 별도 관리해, 키 누출로 인해 키 갱신이 필요한 경우 분산 아이디 전체를 수정할 필요가 없다.

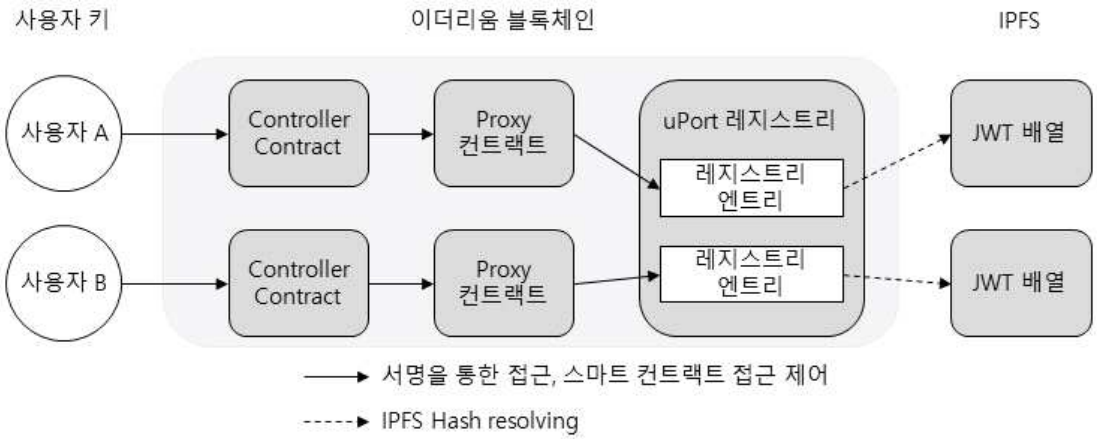
uPort 서비스의 경우 사용자 요청에 따라 신원 정보를 생성하고 이에 대한 공개 키의 해시 값은 블록체인에, 공개 키는 분산형 파일 시스템에 저장한다. 사용자는 신원 정보를 활용할 경우 개인키로 서명한 후 이를 서비스 제공자에게 제시한다. 서비스 제공자는 사용자의 신원 정보를 블록체인에 등록된 공개 키의 해시 값과 분산형 파일 시스템에 등록된 실제 공개 키를 통해 검증한다.

2.2.3 ShoCard[7]

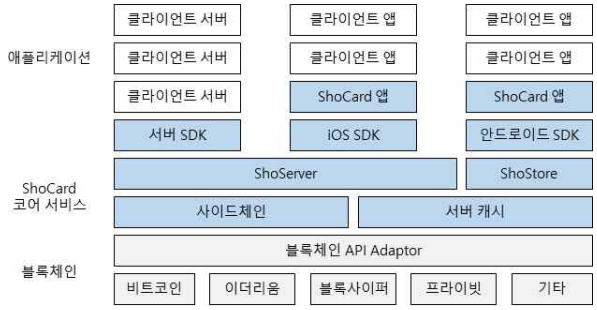
블록체인 기반의 모바일 신원 관리 플랫폼으로 모바일 사용자의 인증을 위한 애플리케이션이 구현되어있다. ShoCard의 구조는 그림 4와 같다. 사용자들은 발급된 신원 정보를 사용해 서비스 제공자에게 필요한 신원 정보를 선택하여 제공할 수 있다.

퍼블릭 블록체인을 기반으로 기존의 증명서를 사용하며 신원 정보의 위·변조를 검증한다. 사용자는 기존에 발급받은 신분증을 스캔하여 신원 정보를 생성하고, 추후 모바일 단말에서 사용할 수 있다.

사용자는 신분증 정보를 해시 및 개인키로



[그림 3] uPort 플랫폼 구조



[그림 4] ShoCard 구조

서명해 생성한 신원 정보를 블록체인에 저장하게 되는데, 서비스 제공자는 신원 정보 확인 후 사용자가 제공하는 정보에 대해 위·변조 여부를 블록체인 상의 정보를 통해 검증할 수 있다.

IV. 결론

DID는 기존에 사용자 신원 증명을 위해 서비스 제공자에게 과도하게 많은 정보를 제공하는 문제를 해결할 수 있는 기술이다. 본 논문에서 보았듯이 이러한 문제점을 해결하기 위해 다양한 방식의 접근이 이루어지고 있다. 그 중 공통적인 요소는 사용자가 원하는 만큼의 정보를 제공할 수 있다는 것이다. DID 기술을 통해 사용자는 개인 정보 유출에 대한 걱정 없이 다양한 서비스를 이용할 수 있을 것이다.

현재 DID의 기술적인 요소는 활발하게 연구중이나 여러 법적 제약에 따라 상용화는 활발하게 이루어지고 있지 않다. 향후에 DID 기술이 발전함에 따라 도입의 필요성을 말하는 목소리가 높아지고 개인 정보를 보호하는 인증을 위한 핵심적인 역할을 할 것이다.

Acknowledgement

이 논문은 국토교통부의 스마트시티 혁신인재 육성사업으로 지원되었습니다.

[참고문헌]

[1] Zachary Diebold, “Self-Sovereign Identity using Smart Contracts on the Ethereum Blockchain”, Master Degree, University of Dublin, Trinity College, May, 2017

[2] Hans Moonen, Marten Van Sinderen, Roel Steenbergen, “Towards Self-Sovereign Identity using Blockchain Technology“, 2016

[3] 전자금융과 금융보안, 금융보안원, Apr, 2019

[3] Metadium Whitepaper v1.0.4(2019). [online] Available at https://www.metadium.com/kr/wwhitepaper [Accessed Nov 2, 2019]

[4] Sovrin : A Protocol and Token for Self-Sovereign Identity and Decentralized Trust(2018). [online] Available at :https://sovrin.org/wp-content/uploads/2018/03/Sovrin-Protocol-and-Token-White-Paper.pdf [Accessed 1 Nov, 2019]

[5] UPORT : A PLATFORM FOR SELF-SOVEREIGN IDENTITY(2016). [online] Available at: http://blockchainlab.com/pdf/uPort_whitepaper_DRAFT20161020.pdf [Accessed 2 Nov, 2019]

[6] ShoCard-Whitepaper-2019-b with recovery(2019). [online] Available at:https://shocard.com/wp-content/uploads/2019/02/ShoCard-Whitepaper-2019.pdf [Accessed 1 Nov, 2019]

로컬 원데이 메시지 블록체인을 활용한 차량 평판 시스템 설계

이수진*, 서승현*

*한양대학교 전자공학과

A Design of Vehicle Reputation System Using Local One-day Message Blockchain

Soojin Lee*, Seung-Hyun Seo*

*Division of Electronic Engineering, Hanyang University.

요 약

가까운 미래의 자율주행 차량이 운행하는 지능형 교통시스템에서 차량들은 실시간 교통 정보를 활발하게 공유한다. 이때 서로의 신원 정보가 없는 차량 간의 데이터 공유하므로 송신 차량과 송신 차량의 메시지를 신뢰하기 위한 판단 척도가 필요하다. 최근 연구들은 송신 차량의 과거 행실에 따른 평판 점수와 메시지에 대한 주변 차량들의 투표 프로토콜을 통해 이를 해결했으며 위조되지 않고 투명하게 공개되어야 하는 평판 점수와 투표 기록을 위해 블록체인 기술을 적용했다. 그러나 차량의 이동성, 계산·저장 능력의 한계를 고려하지 않거나 RSU에 대한 의존도가 높아서 실효성이 부족하다. 또한 안전하게 제공하지 않는 차량의 익명성 문제가 있다. 따라서 본 논문은 원데이 블록체인을 이용하여 이 계층 구조의 차량 평판 블록체인 시스템을 설계했다. 특정 지역에 있는 차량과 RSU가 참여하는 원 데이 블록체인은 하루 동안 발생한 메시지와 투표 정보를 기록하고 RSU가 참여하는 글로벌 평판 블록체인에는 영구적으로 저장되는 차량 평판 점수를 저장한다. 차량은 이전에 발생한 모든 기록을 저장해야 한다는 부담을 줄이면서 직접 블록 정보에 접근한다. 또한, 지역, 날짜에 따라 차량이 다른 개인키-공개키 쌍을 사용하여 차량의 신원과 네트워크에서의 활동 간의 매핑을 방지하고 차량의 프라이버시를 보호한다.

I. 서론

자율주행 차량 기술, 차량 통신 네트워크와 IoT 센서의 발전으로 차량의 센싱 능력과 통신 능력은 빠른 속도로 향상되어 왔다. 이 기술들을 기반으로 지능형 교통시스템에서 자동차들은 교통 데이터를 생성하고 이를 다른 자동차와 RSU와 활발하게 공유한다. 교통 네트워크에서의 활발한 데이터 공유는 교통사고에 대한 빠른 대응과 안전하고 효율적인 교통 관리 체제 구축에 기여를 한다.

그런데 차량 간의 데이터 공유 과정에서 악의적인 차량이 존재할 시, 실제 교통 상황과 일치하지 않는 거짓 메시지를 전달하여 교통 정보 시스템에 혼란을 유발할 가능성이 높다. 따라서 차량은 수신 받은 메시지와 메시지를 송신한 차량의 신뢰여부를 판단해야한다. 차량과 메시지의 신뢰 정보를 습득하기 위해 차량 평판을 이용할 수 있다. 차량의 평판은 교통 네트워크에서의 차량의 과거 행적을 기반으로 산정된다.

이 성과는 2018년도 미래창조과학부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2018R1A2B6006903).

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2018-0-01417)

각 차량은 송신 차량의 평판 점수를 토대로 메시지의 신뢰성을 계산한다. 또는 사고 발생 지역 근처에 위치한 차량들이 사고를 목격했을 시, 해당 사고 메시지에 투표하여 메시지의 신빙성을 높인다.

RSU들이 분산화 되어있고 넓은 범위로 운영되는 교통 네트워크에서 차량의 평판과 메시지 투표 기록은 안전하게 저장하고 공유되어야 한다. 이를 위해 교통 메시지 및 차량 평판모델에서 원장의 무결성을 보장하는 블록체인 기술을 적용하는 연구들이 활발하다[2-6]. 그러나 여전히 이동성, 한정된 연산 능력와 같은 차량의 특성을 고려하지 않아 실제 차량 네트워크에 적합하지 않거나 차량의 RSU에 대한 의존도가 높고 차량의 프라이버시를 고려하지 않았다.

이를 해결하기 위해 본 논문은 로컬 원데이 메시지 블록체인을 사용하여 이 계층 구조의 차량 평판 시스템을 제안한다. 로컬 원데이 메시지 블록체인은 차량의 이동성을 고려해 지역별로 원장을 가지며 하루 동안 생성된 메시지, 투표 기록을 저장하고 다음날에는 소멸되는 임시 블록체인이다. 차량은 현재 위치한 지역에서 그날 생성된 블록만 저장하므로 데이터 저장의 부담을 줄이고 현재 위치에서 발생한 최근 메시지 정보를 직접적으로 접근 가능하다. 영구적으로 기록되어야 하는 차량 평판 정보는 계산 능력이 높은 RSU가 참여하는 글로벌 평판 블록체인에 저장한다. 본 시스템에서 차량은 지역과 날짜에 따라 계좌주소를 새로 생성하도록 하여 차량의 신원과 계좌주소의 링크는 분리되고 익명성을 보장받는다.

II. 관련 연구

[2]는 처음으로 블록체인을 차량 평판 시스템에 적용했다. 차량들이 블록체인 노드로 참여하며 메시지의 투표 내역을 안전하게 저장한다. 그러나 차량이 작업증명(Proof of Work: PoW)을 이용하여 블록을 채굴하기 때문에 각 차량의 전력 소모가 크다. [3]는 RSU들이 블록체인 데이터를 저장하고 채굴 노드로 블록체인 네트워크에 참여한다. 스마트 컨트랙트를 통해 안전한 데이터 공유를 한다. [4]의 저자는 작업증명과 지분증명(Proof of Stake: PoS)를 결합하여 RSU가 블록에 많은 트랜젝션을 포함하도록 유도하는 합의 알고리즘을 제안했다. [3,4]는 RSU가 고정적인 위치에서 블록을 생성하므로 차량이 채굴자인 모델보다 블록의 전달과 원장의 동기화가 더 효율적이다. 그러나 RSU에 대한 의존도가 높아 완전한 분산화를 이룬 시스템으로 보기 어렵다. [5]에서는 멀티-계층 블록체인 구조의 global trustworthy system을 제안했다. 차량들이 노드로 참여하는 platoon 블록체인은 차량의 신뢰 장소를 저장하고 RSU가 참여하는 global 블록체인은 platoon 블록체인을 주기적으로 저장한다. 그러나 구체적인 메시지의 신뢰성 결정 방법이 제시되지 않았다.

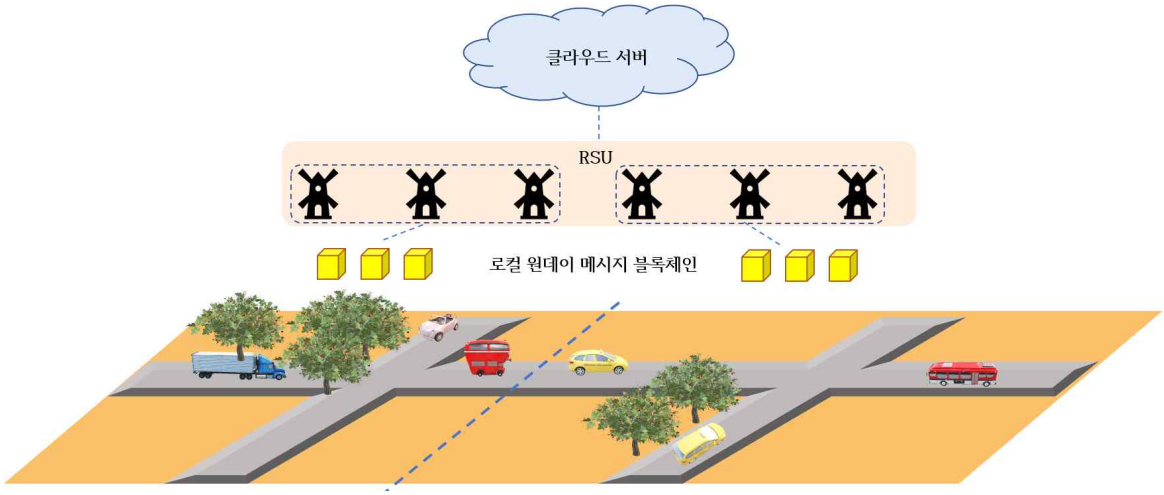
블록체인을 차량 평판 시스템에 적용하면서 모든 차량의 평판 점수와 메시지 투표 기록이 노출되는 문제가 있다. [6]은 threshold ring signature을 적용하여 차량들의 메시지 투표 내역을 가리는 Creditcoin을 제안했다. 그러나 ring signature를 생성하는 과정에서 서명 위조 가능성이 있다.

III. 제안 모델

본 장에서는 모델의 주요 구성요소인 로컬 원데이 메시지 블록체인과 글로벌 차량 평판 블록체인의 특징에 대해 서술한다. 전체적인 모델의 구조는 [그림 1]에서 보인다.

3.1 로컬 원데이 메시지 블록체인

로컬 원데이 메시지 블록체인은 지역별로 독립적으로 운영되며 차량이 하루 동안 생성된 메시지와 차량의 메시지 투표내용을 저장한다. 특정 지역에 위치한 차량과 RSU가 블록체인 노드로 참여하고 이때 RSU가 채굴자가 된다. 실시간 정보 공유를 위해 PBFT와 같은 빠른 속도의 합의 알고리즘을 적용한다. 정해진 시간으로부터 24시간이 지나면 블록체인 정보는 소멸되고 RSU는 다시 첫 번째 블록을 만들어서 새로운 블록체인 원장을 생성한다. 차량들은 여러 로컬 원데이 메시지 블록체인마다 일회용 개인키와 공개키 쌍을 새로 생성한다. 차량들의 신원은 지역별, 시간별로 다르게 존재하여 차량



[그림 1] 전체 모델

의 모든 활동내역을 매핑할 수 없기 때문에 차량의 프라이버시가 보호된다. 차량은 현재 위치한 지역의 로컬 원데이 베시지 블록체인 정보만 저장해도 네트워크 활동에 참여가능하다.

3.2 글로벌 차량 평판 블록체인

글로벌 차량 평판 블록체인은 모든 RSU가 풀 노드로 참여하고 차량의 평판 점수를 기록한다. 로컬 원데이 메시지 블록체인에서 기록된 메시지 정보에 따라 차량의 활동 점수를 매겨 평판 점수를 업데이트한다. 이때 RSU는 차량의 신원과 평판 점수의 매핑을 할 수 있다. 차량은 필요에 따라 본인 혹은 다른 차량의 평판 점수를 알기 원할 때, RSU에 요청하여 알 수 있다.

Ⅳ. 운영 시나리오

전체 모델의 운영 시나리오는 다음과 같다.

4.1 차량 등록

운전자는 새로 구매한 차량을 TA를 통해 등록한다. TA는 차량이 생성한 공개키에 대해 공인 인증서를 발급해하고 공개키 정보는 RSU들에게 전달한다. 해당 공개키는 차량의 신원 ID이자 글로벌 차량 평판 블록체인의 계정이다.

4.2 차량의 로컬 네트워크 참여

차량은 로컬 원데이 메시지 블록체인에 참여하기 위해 먼저 일회용으로 사용할 개인키와 공개키를 생성한다. 그리고 자신의 위치 정보와 일회용 공개키에 신원 ID로 서명하여 주변 RSU에게 보낸다. RSU는 차량이 해당 위치에 있는지 확인하고 서명을 검증 한 후, 현재 시간까지 생성된 로컬 원데이 메시지 블록체인을 전송한다. 차량의 일회용 공개키는 원데이 블록체인의 계정으로 사용한다. RSU는 해당 계정으로 차량이 활동하면서 사용할 코인을 지급한다. 이때 금액은 차량의 평판에 비례하여 결정된다.

4.3 메시지 트랜잭션 생성 및 배포

차량은 사고를 목격할 시. 먼저 사고 메시지를 생성하여 주변 차량에게 보낸다. 해당 메시지 내용에 동의하는 수신 차량들은 자신의 코인을 이용하여 투표 메시지를 생성하여 다시 송신 차량에게 전달한다. 특정 수 이상의 투표 메시지가 모이면 차량은 자신의 코인을 더해 메시지 트랜잭션을 생성하고 주변 RSU들과 차량들에게 배포한다.

4.4 로컬 원데이 메시지 블록체인의 블록 생성 및 배포

RSU는 받은 메시지 트랜잭션의 서명 값들을 검증하고 메시지 내용의 진위여부를 다시 확인한다. 유효한 트랜잭션이라고 판단되면 RSU들은 합의 알고리즘을 통해 채굴자를 선정하고 채굴 RSU는 검증된 트랜잭션을 모아 블록을 생성하고 로컬 네트워크에 참여하고 있는 차량들과 RSU들에게 전달한다..

4.4 글로벌 차량 평판 블록체인의 평판 점수 업데이트

로컬 원데이 메시지 블록체인의 기록된 각 차량의 활동 내역을 토대로 평판 점수 트랜잭션

을 생성한다. 메시지 생성과 투표를 적극적으로 한 차량에 대해서는 높은 점수를 주고 소극적으로 활동했거나 악의적인 메시지를 생성한 것이 발견되었던 차량은 낮은 점수를 주거나 기존 평판 점수를 차감하도록 한다. 평판 점수 트랜잭션은 모여서 글로벌 차량 평판 블록체인에 영구적으로 기록된다.

4.5 차량의 로컬 네트워크 이탈

특정 지역을 떠나는 차량은 자신이 소유하고 있던 로컬 원데이 메시지 블록체인 데이터를 지운다. 지역 가장자리에 있는 RSU는 차량이 지역을 떠나는 것을 감지한 후, 이를 다른 RSU에게 알린다. 다른 지역에 도착한 차량은 다시 로클 네트워크에 참여하기 위해 4.2의 과정을 시작한다.

Ⅴ. 결론

본 논문은 두 개의 블록체인을 계층 구조로 연결하여 차량 네트워크에서 차량의 데이터 공유와 평판 측정을 효율적으로 할 수 있다. 차량은 지역별로 하루 동안 운행되는 로컬 원데이 메시지 블록체인을 통해 실시간으로 필요한 메시지를 직접적으로 읽을 수 있으며 불필요한 과거 메시지 정보를 모두 저장하지 않아도 되어 메모리 낭비를 하지 않는다. 또한 지속적으로 기록되고 저장되어야 하는 차량 평판 정보는 연산 능력이 높은 RSU가 글로벌 차량 평판 블록체인에 저장하여 변조 위험 없이 평판 정보의 무결성을 보장한다. 또한 일회용 공개키 사용으로 차량들은 다른 차량의 신원을 이전 활동과 연결질 수 없어서 차량의 프라이버시를 보호한다. 제안하는 모델을 통해 차량들과 RSU는 차량 네트워크에서 안전하고 효율적인 교통 시스템 운영에 기여할 것으로 기대된다.

[참고문헌]

[1] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

[2] Yang, Z., Zheng, K., Yang, K., & Leung, V. C. (2017, October). A blockchain-based reputation system for data credibility assessment in vehicular networks. In 2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC) (pp. 1-5). IEEE.

[3] Kang, J., Yu, R., Huang, X., Wu, M., Maharjan, S., Xie, S., & Zhang, Y. (2018). Blockchain for secure and efficient data sharing in vehicular edge computing and networks. IEEE Internet of Things Journal.

[4] Yang, Z., Yang, K., Lei, L., Zheng, K., & Leung, V. C. (2018). Blockchain-based decentralized trust management in vehicular networks. IEEE Internet of Things Journal, 6(2), 1495-1505.

[5] Kandah, F., Huber, B., Skjellum, A., & Altarawneh, A. (2019, January). A Blockchain-based Trust Management Approach for Connected Autonomous Vehicles in Smart Cities. In 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0544-0549). IEEE.

[6] Li, L., Liu, J., Cheng, L., Qiu, S., Wang, W., Zhang, X., & Zhang, Z. (2018). Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles. IEEE Transactions on Intelligent Transportation Systems, 19(7), 2204-2220.

블록체인 기반 DO-ABE를 이용한 커넥티드 카 데이터 공유 기법

정병규*, 김혜빈*, 신상욱**

*부경대학교 대학원 정보보호학협동과정

**부경대학교 IT융합응용공학과

Data Sharing Technique for Connected Car Using Blockchain-Based DO-ABE

Byeong-Gyu Jeong*, Hye-Bin Kim*, Sang Uk Shin**

*Interdisciplinary Program of Information Security, Graduated School, Pukyong National University

**Dept. of IT Convergence and Application Eng., Pukyong National University

요 약

차량 네트워크에서의 안전하고 신속한 데이터 공유 및 거래에 대한 구제적인 프로세스가 요구되고 자율 주행 자동차의 사고에 대한 책임 추적의 문제가 대두되고 있다. 이에 따라 본 논문에서는 블록체인 기반의 DO-ABE를 이용한 데이터 공유 기법과 함께 블록체인 기반의 차량 데이터 마켓플레이스 플랫폼 모델을 제안한다. 해당 모델을 통해 차량 사고 수사관은 효율적이고 신속한 사고 수사를 할 수 있고 차량 정비소와 보험사의 경우 정확한 진단과 가격 책정을 할 수 있다.

I. 서론

최근 5G 네트워크의 발전에 따라 차량 내·외부로 네트워크 연결이 가능한 커넥티드 카, 자율주행 자동차의 연구가 활발해지고 있다. 커넥티드 카란 무선랜이 장착되어 인터넷 접속이 가능한 차량을 의미한다[3]. 최근에는 Vehicle to Everything(V2X)로 기술이 발전하면서 고속도로, 선박, 기차, 비행기 등의 모든 유형의 차량뿐만 아니라 신호등, 기지국, 각종 무선 기기 등 모든 인프라 시스템을 연결한다. 이에 따라 차량 네트워크에서의 안전하고 신속한 데이터 공유 및 거래에 대한 구체적인 프로세스가 요구되고 있다[1]. 또한 자율 주행 자동차의 사고에 대한 책임 추적의 문제도 해결과제로 남아있다[2].

본 논문에서는 블록체인을 이용하여 차량 네트워크에서의 데이터를 안전하게 저장하고 컨소시엄으로 연결된 집단끼리 차량 데이터를 투명하게 공개하여 데이터를 보다 효율적으로 활용할 수 있는 모델을 제안한다. 또한 데이터 소유주 기반의 속성 기반 암호화를 통해 데이터 소유주가 직접 데이터를 제어할 수 있는 데이터 공유·거래 프로세스를 제안한다.

II. 관련 연구 : DO-ABE

2015년 Jindan Zhang은 ABE 기반의 DO-ABE(Data Owner based Attribute Based Encryption)를 제안했다[5]. 기존 ABE 체계는 사용자를 위한 비밀키를 생성하기 위해 PKG의 존재를 요구한다. 그에 반해 DO-ABE는 암호화에 사용되는 임의성을 이용해 사용자를 위한 비밀키를 생성한다. 기존 ABE와 비교하여 DO-ABE는 암호화된 콘텐츠에 대한 메시지 수준 기반 세부적인 제어를 쉽게 달성할 수 있다. DO-ABE는 키 정책 DO-ABE(CPDO-ABE)와

암호문 정책 DO-ABE(CPDO-ABE)로 구분한다. 암호화 알고리즘은 LSSS 접근 행렬 M에 따라 랜덤 지수 $s \in Z_p$를 분배한다. 개인키는 충돌 공격(collusion attack)을 피하기 위해 무작위로 지정된다. CPDO-ABE에서는 4개의 주요 함수를 설명한다.

1) Setup(U) : setup 알고리즘은 시스템에서 속성들의 수 U를 입력으로 받는다. 소수 위수 p의 군 G, 생성자 g, 시스템에서 U개의 속성에 연관된 U개 랜덤한 군의 원소 $h_1, h_2, ..., h_U \in G$를 선택한다. 이 때 공개 파라미터는 $(g, h_1, h_2, ..., h_U)$이다.

2) KeyGen(MSK, S) : 암호화 개체(데이터 소유자)는 랜덤 지수 $\beta, \alpha \in Z_p$를 선택하고 공개키 $PK = (e(g,g)^{\beta}, g^{\alpha})$를 계산한다. 암호화 개체는 마스터 비밀키로 $MSK = g^{\beta}$를 설정하고 속성들의 집합 S와 함께 입력으로 받는다. 그리고 $K = g^{\beta} g^{\alpha t}$, $L = g^{t}$, $\forall x \in S$x, $K_x = h_x^t$를 이용해 사용자 비밀키 SK를 생성한다.

3) Encrypt(PK, ST, m) : 암호화 알고리즘은 공개키 PK와 암호화할 메시지 m을 입력으로 받는다. 추가적으로 LSSS 접근 구조 $ST = (AM, p)$을 입력으로 받는다. 이 때, AM은 $l \times n$ 행렬이다. 알고리즘은 랜덤 벡터 $\vec{v} = (s, y_1, y_2, ..., y_n) \in Z_p^n$과 임의의 $r_1, r_2, ..., r_l \in Z_p$를 선택한다. 암호문은 ST와 함께 다음과 같이 CT로 공개된다. $C = m \cdot e(g,g)^{\beta s}$, $C' = g^{s}$.

4) Decrypt(CT, SK) : 복호화 알고리즘은 접근 구조 $ST = (AM, p)$에 대한 암호문 CT와 집합 S에 대한 비밀키 SK를 입력으로 받는다. 복호화 알고리즘은 먼저 다음을 계산한다.

$$\frac{e(C', K)}{\Pi_{i \in I}((e(C_i, L) e(D_i, K_{p(i)}))^{\omega i})} =$$

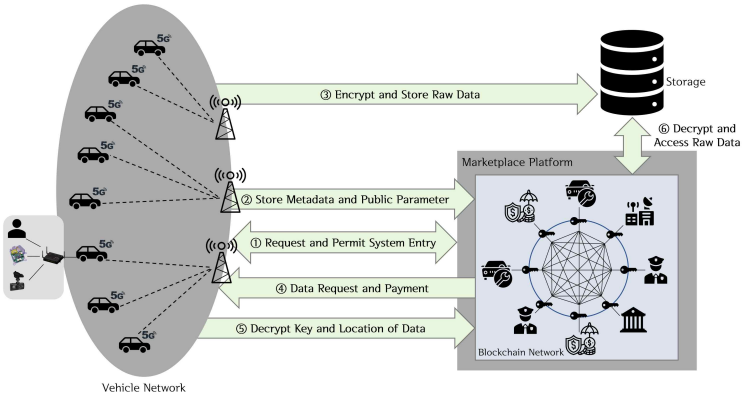
$$\frac{e(g,g)^{\beta s} e(g,g)^{\alpha s t}}{\Pi_{i \in I} e(g,g)^{t a \omega_i \lambda_i}} = e(g,g)^{\beta s}$$

복호화 알고리즘은 이 값을 C로 나누어 메시지 m을 얻을 수 있다.

데이터 소비자는 동일한 속성값을 공유하지만 데이터 소유주가 임의로 α, β를 선택할 수 있으므로 결과적으로 다른 데이터를 얻게 된다. 따라서 DO-ABE는 기존 전통적인 ABE보다 접근 제어에 더 유연하다.

III. 블록체인 기반 커넥티드 카 데이터 공유 모델

이 절에서는 블록체인 기반의 커넥티드 카 데이터 공유 모델의 구조와 특징에 대해 논의한다. [그림 1]은 블록체인 기반의 커넥티드 카 데이터 공유 모델 구조를 나타낸다. 참여자는 데이터 소유주, 데이터 소비자, 서비스 제공자로 크게 세 유형으로 나눌 수 있다. 데이터 소유주는 블록체인 네트워크와 상호 연결하는 5G 통신이 가능한 커넥티드 카 또는 블랙박스로 가정한다. 데이터 소유주는 이동 경로와 시간 정보가 담긴 차량 데이터와 블랙박스 데이터를 5G 기지국을 통해 블록체인 네트워크에 제출한다. 이러한 데이터들은 이후 트랜잭션화 되어 블록체인에 저장된다. 데이터 소유주들은 자신의 블랙박스 데이터를 수집하여 차량 데이터 마켓플레이스 플랫폼에 제공하고 이에 대한 국가적인 금



[그림 1] 블록체인 기반의 커넥티드 카 데이터 공유 모델 구조

전적 보상을 받을 수 있다. 데이터 소비자는 차량 블랙박스 데이터를 필요로 하는 차량사고 수사관, 보험 회사, 차량 정비소, 법원, 5G 네트워크를 지원하는 통신사 등이 해당된다. 마켓플레이스 플랫폼에서 필요한 데이터를 찾은 뒤 적절한 접근 수단을 지불해야 한다. 접근 수단에는 정당한 사용자라는 인증서, 차량사고 수사관의 경우 영장이 여기에 해당된다. 거래가 성사되면 데이터 소유주로부터 원본 데이터에 접근할 수 있는 위치와 키를 부여받는다. 서비스 제공자는 원하는 블랙박스 데이터를 검색하고 거래할 수 있는 마켓플레이스 플랫폼이다.

해당 모델은 거래되는 데이터가 민감한 정보가 있는 블랙박스 영상이기 때문에 참가하는 개체들을 인증하고 검증하여 제한할 필요가 있다. 따라서 블록체인은 컨소시엄 허가형 블록체인(Consortium Permissioned Blockchain)의 형태가 된다. 원본 데이터와 메타데이터를 외부 스토리지와 블록체인에 분리하여 저장함으로써 민감한 정보를 비밀로 유지할 수 있다.

블록체인 트랜잭션의 구성은 타임스탬프값, 위치 정보, 익명으로 된 소유주 정보, 원본 데이터의 위치, 원본 데이터의 무결성 검증을 위한 해시값으로 구성되어 있으며 위치 정보는 블랙박스 데이터에 해당하는 출발지, 목적지, 이동시간 및 거리가 포함된다. 익명으로 된 소유주 정보는 차량 네트워크에 사용되는 게이트웨이의 SIM(Subscriber Identification Module) 카드에 포함된 사용자 식별정보를 이용하여 부여된 UID이다. 블록체인에는 확장성과 효율성을 위해 데이터에 대한 참조만 온체인에 저장되기 때문에 외부 저장소에 있는 원본의 위치도 트랜잭션에 포함되어야 한다.

해당 모델의 합의 매커니즘은 PBFT를 채택한다. PBFT 합의 알고리즘은 비동기 네트워크에서 배신자 노드가 f개 있을 때, 총 노드의 개수가 3f+1개 이상이면 해당 네트워크에서 이뤄지는 합의는 신뢰할 수 있다. 제안된 시스템 모델에 참여하는 집단이 적어도 5개이기 때문에 한 집단에서 모의하고 분기를 위한 악의적인 행동을 해도 나머지 4개의 집단이 정직하다면 분기를 방지하고 합의에 이를 수 있다.

마켓플레이스 플랫폼에 참여하는 개체의 인증과 키 생성 및 교환, 데이터를 등록하고 저장하는 등 대부분의 작업들은 컨트랙트로 이루어진다. 정보의 민감한 정도와 계산 능력 정도에 따라 온체인(on-chain)에서 이루어지는 스마트 컨트랙트와 오프체인(off-chain)에서 이루어지는 프라이빗 컨트랙트로 나누어 수행한다.

IV. 블록체인 기반 DO-ABE를 이용한 데이터 공유 기법

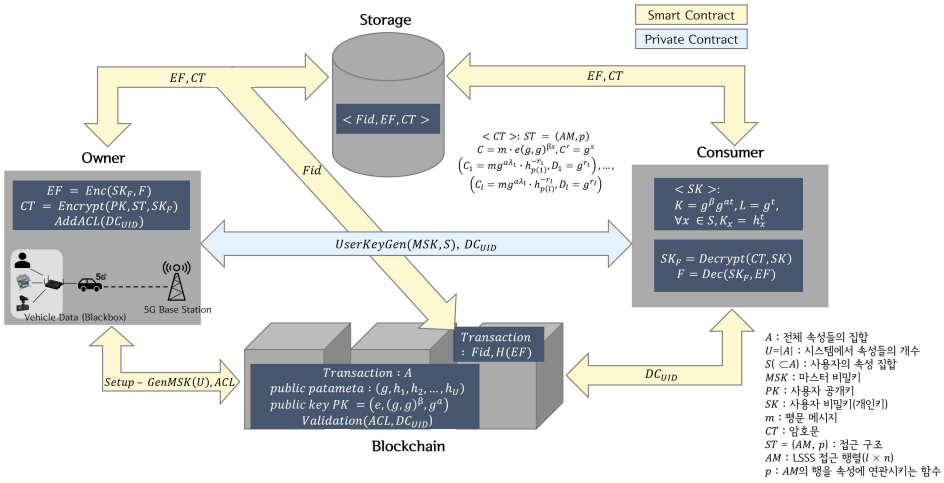
이 절에서는 클라우드 기반이였던 DO-ABE를 블록체인 기반으로 변형한 형태로 제시하고 이를 이용한 해당 모델에서의 데이터 공유 및 거래 프로세스를 제시한다.

[그림 2]은 데이터를 등록하고 접근하는 데이터 공유 프로세스를 나타낸 흐름도이다.

데이터 소유자는 스마트 컨트랙트를 통해 거래가 완료된 소비자의 UID를 ACL(Access Control List)에 정의하고, 이를 암호화한 뒤에 블록체인 트랜잭션에 게시한다. 데이터 소비자는 스마트 컨트랙트에 의해 ACL이 허용하는 경우에만 데이터에 접근하고 처리할 수 있다.

데이터 소유자는 마스터 비밀키 MSK를 생성한다. EDR 데이터, GPS를 기반으로 한 이동 거리 및 시간, 소유주의 UID를 포함한 정보들을 차량 게이트웨이가 수집 후 사용자 공개키 PK와 함께 블록체인에 게시한다. 소유자는 공유할 원본 블랙박스 데이터 F 를 대칭키 sk_F로 암호화하여 EF를 생성하고, 해당 대칭키 sk_F는 CP-ABE로 암호화한다. 이를 위해 적절한 접근 구조 ST를 정의하여 $Encrypt(PK, ST; sk_F)$를 수행하여 암호문 CT를 계산한다. 블랙박스 데이터 F의 식별자 Fid, 해시값 $H(EF)$ 등을 포함한 메타데이터는 블록체인에 공개하고, (Fid, EF, CT)는 외부 스토리지에 저장한다. 이때 저장소의 주소는 $H(F)$로 사용한다.

소비자는 마켓플레이스 플랫폼을 통해 원하는 자료를 검색하여 해당 자료의 메타데이터 정보를 얻는다. 거래가 성공적으로 이루어지면, 오프체인으로 소유자에게 해당 블랙박스 영상의 속성 키 생성을 요청한다. 소유자는 소비자의 속성 비밀키



[그림 2] 데이터 공유 프로세스

SK=$UserKeyGen(MSK, S)$를 생성하여 저장 위치 $H(F)$와 함께 오프체인 프라이빗 컨트랙트를 통해 안전하게 분배한다. 소비자는 저장 위치 $H(F)$에 저장된 블랙박스 데이터 (Fid, EF, CT)를 획득한 후, sk_F = $Decrypt(CT, SK)$를 계산한다. 소비자는 sk_F로 EF를 복호화하여 원 데이터 F를 얻을 수 있다.

V.결론

본 논문에서는 블록체인 기반 DO-ABE를 이용한 커넥티드 카 차량 데이터 공유 및 관리 기법에 대해 제안하였다. 해당 모델을 통해 차량 사고 수사관은 효율적이고 신속한 수사를 할 수 있으며 보험사나 차량 정비소는 정확한 진단과 가격 책정이 가능해진다. DO-ABE를 통해 데이터 소유주가 본인의 데이터에 대한 세부적인 제어가 가능하며 off-chain 프라이빗 컨트랙트를 통한 키 생성과 거래를 통해 블록체인 상에서의 효율적이고 신속한 계산을 보장할 수 있다.

향후 스마트 컨트랙트의 설계와 구현, 데이터 소유주에 대한 보상 문제와 구체적인 ACL 정책 설계가 연구 과제로 남아있다.

Acknowledgement

본 연구 논문은 한국전자통신연구원 연구운영비지원사업(19ZH1200, 데이터 안심사회를 위한 트러스트 데이터 커넥톰 핵심 원천 기술 개발)의 위탁연구과제의 연구결과입니다.

[참고문헌]

[1] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher and Valerie Young, “Mobile Edge Computing A key technology towards 5G”, ISBN No. 979-10-92620-08-5, European Telecommunications Standards Institute, 2015.

[2] Hevelke, Alexander, and Julian Nida-Rümelin. “Responsibility for crashes of autonomous vehicles: an ethical analysis”, Science and engineering ethics, vol. 21, no. 3 pp. 619-630, Jun 2015.

[3] Coppola, Riccardo, and Maurizio Morisio, “Connected car: technologies, issues, future trends.”, ACM Computing Surveys (CSUR), vol. 49.3, no. 46, pp 1-36, Oct 2016.

[4] Castro, Miguel, and Barbara Liskov, “Practical Byzantine fault tolerance.”, OSDI, vol. 99, no. 1999, 1999.

[5] Jindan Zhang, Xu An Wang, Jianfeng Ma, “Data Owner Based Attribute Based Encryption”, 2015 International Conference on Intelligent Networking and Collaborative Systems. IEEE, Nov 2015.

[6] 김승현, 김수형, “블록체인 기반 접근제어 기술 동향”, 34(4), Electronics and Telecommunications Research Institute, Aug 2019.

블록체인을 활용한 종량제 봉투 실명제 제안

심민주*, 권용빈*, 엄시우**, 윤재웅**, 임현빈**, 서화정*†

*한성대학교 IT융합공학부

**한성대학교 컴퓨터공학부

Real name system for garbage bags using blockchain

Min-Joo Sim*, Yong-Bin Kwon*,

Si-Woo Eum**, Jae-Woong Yun**,Hyun-Bin Im**,

Hwa-Jeong Seo*†

*Division of IT Convergence Engineering, Hansung University

**Division of Computer Engineering, Hansung University

요 약

2015년 처음 시범 운영된 종량제 봉투 실명제는 시범 운영 지역의 재활용품 분리수거 비율이 높아져 생활쓰레기 배출량이 이전의 1/3이나 줄어든 효과를 거뒀다. 하지만 종량제 봉투 실명제는 사생활 침해 문제로 인해 시행에 대한 어려움을 겪고 있다. 이와 같은 사생활 침해 문제를 보완하기 위해 본 논문에서는 사용자의 정보가 담긴 RFID 카드와 종량제 봉투의 바코드를 이용한 종량제 봉투 수거 기계를 제안한다. 기계 설치가 힘든 장소에는 종량제 봉투 구매 시 사용자의 정보가 들어있는 RFID 카드와 종량제 바코드를 찍으면 그 정보를 암호화한 값이 적힌 스티커를 발급받아 종량제 봉투 배출 시 암호화된 스티커를 부착하는 방법을 제안한다.

I. 서론

쓰레기를 제대로 분리수거를 하지 않은 채 비양심적으로 버리는 쓰레기를 줄이기 위해 종량제 봉투에 주소를 적게 하는 방법인 '종량제 봉투 실명제'가 언급되고, 현재 시험 운영 중인 곳도 있다. 실명제를 도입하면 종량제 봉투에 분리수거를 해야 하는 쓰레기나 음식물 쓰레기를 섞어서 버리는 일이 줄어들고 분리배출이 활성화되어 배출되는 쓰레기의 양이 감소하는 장점이 있다. 하지만 종량제 봉투 실명제는 '사생활 침해'라는 큰 단점을 지니고 있다.

본 논문에서는 현재 시행되고 종량제 봉투 실명제의 문제점을 제시하고 이 문제의 해결책으로 블록체인을 이용한 새로운 종량제 봉투 수거 기계와 기계 설치가 힘든 장소에는 암호화된 스티커를 종량제 봉투에 부착하여 종량제 봉투 실명제로 인해 발생할 수 있는 사생활 침해 문제를 해소 방안을 제안한다.

II. 기존 종량제 봉투 실명제

종량제는 폐기물 발생량을 줄일 목적으로 설계된 제도이다. 배출되는 폐기물량에 따라 요금을 부과하는 제도로 한국은 1995년부터 생활폐기물 발생량 저감을 위해 전국적으로 생활폐기물 종량제를 도입하고 시행하고 있다.

종량제 봉투 실명제는 2015년 강원도 평창군에서 주민들의 제안으로 처음 시범 운영했다. 실명제 스티커를 붙이지 않은 종량제 봉투는 5일간 수거하지 않고, 음식물 쓰레기나 재활용품을 함께 배출한 경우 10만 원의 과태료를 부과하는 방법을 시행했다. 이 결과로 생활쓰레기

배출량이 하루 평균 7~8톤가량이 줄어들고, 주민들의 재활용품 분리수거 비율이 높아져 생활쓰레기 배출량이 이전의 1/3이나 줄어든 효과를 거뒀다. 실명제가 이렇게 큰 성과를 거뒀음에도 불구하고 2016년 경기도 수원시 영통구에서 종량제 봉투 실명제 시범 운영이 결정된 후, 다음과 같은 인권침해 문제로 많은 논란이 있었다. 예를 들면, 개인 쓰레기의 상세주소를 이용하여 범죄에 악용이 가능한 점과 쓰레기를 보고 여성 혼자 사는 집을 알아차릴 가능성 등의 여러 논란으로 시행되지 않았다. 종량제 봉투 실명제에 대한 논란은 현재까지 많은 의견 충돌이 발생하고 있다.

하지만, 쓰레기로 인한 환경오염과 수도권에서 발생하는 쓰레기를 점점 감당하기 힘든 수도권 매립지 규모 등의 한계를 극복하기 위해서 철저한 분리배출을 통한 생활쓰레기 감량과 생활습관의 변화가 필요하다. 최근에는 이와 같은 이점으로 인해 시청이나 구청 등 공공기관은 부서 실명제를 도입하여 실천하며 있다. 이와 같은 실명제를 공공기관의 부서가 아닌 전 지역으로 확산시키기 위해서는 사생활 침해 문제가 해결되어야 한다. 본 논문에서 제안하는 방법은 현재 논란이 되는 종량제 봉투 실명제를 시행했을 때의 사생활 침해 문제를 해결할 수 있다.

III. 블록체인

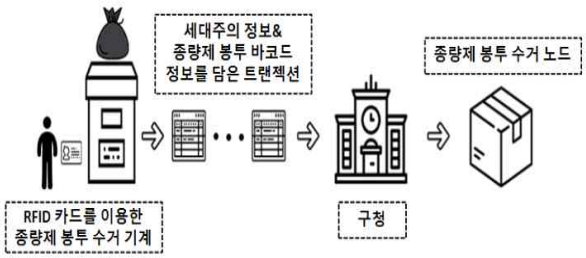
블록체인은 완벽한 보안이 확보되었다고 할 수 없는 인터넷망에서 어떠한 중재자도 없이 가치가 있는 대상을 안전하고 자유롭게 교환할 수 있는 분산 컴퓨팅 기술 기반의 원장 관리 기술이다. 블록체인 네트워크 참여자들은 모두 동등한 계층이기 때문에 같은 정보 확인, 관리가 가능할 뿐 아니라 분산 장부로 인해 복제, 공유, 동기화된 정보는 모두 기록으로 남아 불변성이 확보된다. 블록체인 암호화 기술로는 데이터의 무결성 검증과 거래의 부인 방지를 위한 공개키 기반의 디지털 서명 기법이 사용되고 있다.

IV. 영지식 증명 (zero-knowledge Succinct Non-interactive Argument of Knowledge)

영지식 증명은 1985년 Shafi Goldwasser, Silvio Micali, Charles Rackoff의 논문 “The Knowledge Complexity of Interactive Proof-Systems”에서 처음 소개된 개념으로 현재 암호학에서는 다자간의 비대면 통신 프로토콜에서 정보보호 기능을 제공하기 위해 암호기술을 적용하는 암호 프로토콜 중 매우 중요한 고급 암호 프로토콜의 한 종류로 영지식 증명을 취급한다. 영지식 증명은 쉽게 말해 상대방에게 어떤 사항이 참이라는 것을 증명할 때, 그 문장의 참 거짓 여부를 제외한 어떠한 것도 노출되지 않은 상호적인 절차를 의미한다. 2016년도에 개발된 암호 화폐인 Zcash에 영지식 증명이 사용되어 Zcash를 소유한 사람들끼리 일체의 개인 정보를 노출하지 않고, 상호 간의 코인 전송이 가능하다.

V.시스템 제안

본 논문에서 블록체인 기반의 종량제 봉투 실명제 시스템을 제안한다.



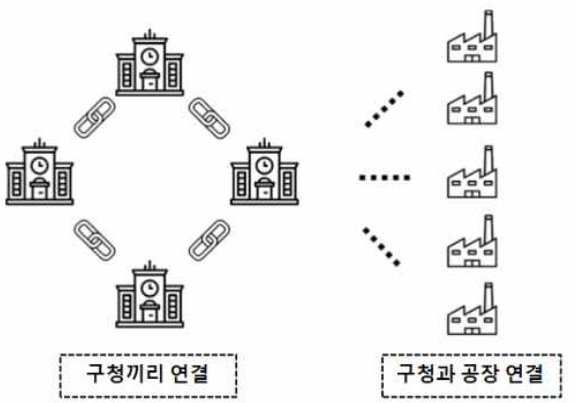
(그림 1. 종량제 봉투를 버리는 과정)

요즘 아파트에서 흔히 볼 수 있는 RFID 카드를 이용한 음식물 쓰레기 수거 기계와 같이 아파트 혹은 주택가에 지정된 종량제 봉투를 버리는 장소에 RFID 종량제 봉투 수거 기계를 설치한다. 각 가정에서는 이 기계에 종량제 봉투를 버리기 위해서는 종량제 봉투 수거 기계에 각 가정에 부여받은 RFID 카드를 찍고 종량제 봉투에 새겨져 있는 바코드를 찍으면 수거 기계

가 열리게 되어 버릴 수 있다.

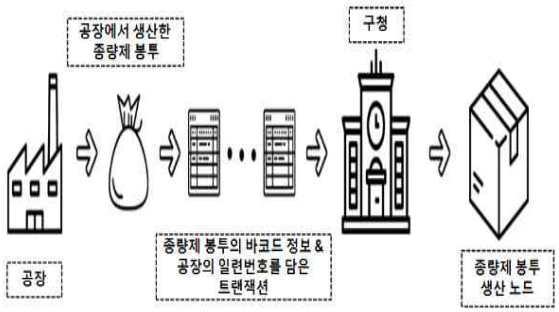
이 RFID 카드에는 주민들의 개인 정보는 영지식 증명을 이용해 저장된다. 여기서 증명자는 세대주를 의미하며 검증자는 구청을 의미한다. 세대주는 구청을 제외한 다른 이들에게는 세대주의 어떠한 정보도 주고받지 않으며 이 정보는 암호화되어 저장되어 세대주의 정보를 숨겨서 공개함으로써 실명제의 부작용 보안이 가능하다. 종량제 수거 기계에 종량제 봉투를 버리고 나면 트랜잭션이 생성되고, 그 트랜잭션은 세대주와 구청이 연결된 노드를 통해 구청으로 전송된다. 구청에서는 전송받은 트랜잭션을 모아 블록을 형성한다. 세대주 트랜잭션에는 종량제 봉투의 일련번호, 배출 날짜, 세대주 정보가 기록된다. 위의 그림 1과 같이 종량제 봉투 기계가 설치가 가능한 지역의 버리는 과정을 나타낼 수 있다.

종량제 봉투 수거 기계 설치가 힘든 지역의 경우에는 스티커를 부착하여 종량제 봉투를 버린다. 대형 마트나 편의점과 같은 종량제 봉투 판매소에 스티커 뽑는 기계가 배치되어 가정마다 받은 RFID 카드를 이 기계에 인식시키게 되면 카드에 있는 개인 정보와 봉투의 바코드 값을 해시를 통해 암호화한 값을 스티커에 적어 인쇄해준다. 이 스티커는 종량제 봉투 구매 전 뽑아서 직원에게 보여주어야 스티커를 발급받은 양만큼의 종량제 봉투 구매가 가능하다. 이는 카드와 봉투를 종량제 봉투 수거 기계에 찍는 행위를 대신해준다. 이 스티커가 부착되어 있지 않은 경우, 종량제 봉투를 수거해가지 않는다.



(그림2. 종량제 봉투 생산 블록체인 네트워크)

종량제 봉투 위조와 다른 지역에서 구매한 종량제 봉투를 우리 지역에 버리는 행위를 막기 위해 생산 공장에서는 종량제 봉투를 생산할 때 생산 공장이 부여받은 일련번호 정보와 생산량이 저장된 트랜잭션이 생성되며 구청에 모여 블록을 형성한다. 위의 그림 2와 같이 종량제 봉투 생산 네트워크를 나타낼 수 있다. 구청에서는 전송받은 트랜잭션을 모아 블록을 형성한다. 구청에서는 생산 공장에서 보낸 정보와 세대주가 버린 종량제 봉투의 정보를 비교하여 위조 여부의 판별이 가능하다. 아래의 그림 3과 같이 종량제 봉투를 생산하는 과정을 나타낼 수 있다.



(그림 3. 종량제 봉투를 생산하는 과정)

'서울 열린 데이터 광장' 사이트에서 제공하는 공공데이터 '서울특별시 쓰레기종량제 봉투 판매업소 정보' Open API로 구청과 생산 공장을 연결하고, '서울시 쓰레기 수거 현황 통계' Open API로 다른 구의 종량제 봉투를 구매하여 우리 구에 버리는 경우 확인이 가능하다.



(그림 4. 종량제 봉투 생산량과 수거량 비교)

다른 구의 서울시의 종량제 봉투를 통일하여 각 구청 종량제 봉투 재고량을 정보를 확인하여 재고가 부족한 구청은 재고가 남는 구청에서 종량제 봉투 구매가 가능하여 종량제 봉투 재고관리가 가능하다.

VI. 한계

본 논문에서 제안하는 블록체인 기반의 종량제 봉투 실명제는 몇몇 한계점이 있다. Open API를 이용하여 다른 구의 종량제 봉투를 구매하여 배출하는 경우를 단속하는 것은 Open API를 제공하는 공공기관이 '서울 열린 데이터 광장'이라는 점에서 전국이 아닌 서울로 한정하였다. 또한. 각 구청끼리 종량제 봉투의 남는 재고를 거래하는 과정을 시행하기 위해서는 종량제 봉투가 통일되었다는 가정이 필요하다.

VII. 결론

본 논문에서는 종량제 봉투 실명제 시행에 논란이 된 사생활 침해 문제를 블록체인을 기반한 각 가정의 개인 정보를 의미하는 일련번호를 담고 있는 RFID 카드를 이용한 종량제 봉투 수거 기계와 기계 설치가 불가능한 지역에는 종량제 봉투 구매 시 각 가정의 개인 정보가 들어있는 RFID 카드와 구매하는 종량제 봉투의 바코드를 찍으면 두 값의 해시를 통한 암호화된 값을 지닌 스티커를 발급하는 방법을 제안하였다.

[참고문헌]

[1] Jeong-hyuk Lee, Jung Yeon Hwang, Hyun-ok Oh, Ji-hye Kim, "Personal Information Management System with Blockchain Using zk-SNARK" in Journal of the Korea Institute of Information Security & Cryptology 29(2), 2019.4

[2] Dong-Young Lee, Ji-Woo Park, Jun-Ha Lee, Sang-Rok Lee, Soo-Yong Park, "Blockchain Core Technology and Domestic and Foreign Trends" in Communications of the Korean Institute of Information Scientists and Engineers 35(6), 2017.6

[3] 양진우, "RFID기반 음식물쓰레기 종량제 성과분석 및 추진방안", BUSAN DEVELOPMENT FORUM (160), 2016.08

안전한 전자상거래를 위한 보안 기법 연구: 온라인 현금자산 보안을 중심으로

박용현[1], 조민주[2], 김남형[3], 정재영[4], 이경민[5], 이준범[6], 김홍진[7]

[1]한국과학기술원, [2]동국대학교 경주캠퍼스, [3]순천향대학교, [4]한양대학교, [5]동국대학교 경주캠퍼스, [6]대구가톨릭대학교, [7]LG CNS

A Study on Security Technique for Secure Electronic Commerce: Security of Online Property

Yong-Hyeon Park[1], Min-Ju Jo[2], Nam-Hyeong Kim[3], Jae-Young Jeong[4], Kyung-Min Lee[5], Jun-Beom Lee[6], Hong-Jin Kim[7]

[1]KAIST, [2]Dongguk University Gyeongju, [3]Soonchunhyang University, [4]Hanyang University, [5]Dongguk University Gyeongju, [6]The Catholic University of Daegu, [7]LG CNS

요 약

2000년대 초 전자상거래가 활성화되며 이용자 수가 급격하게 증가했고, 인터넷 이용자들의 전자상거래 의존도는 점차 높아져[1] 2016년 전자상거래 규모는 약 4000억 달러에 이르렀다[2]. 하지만 전자상거래 서비스 이용자들은 보안 문제를 심각하게 여기지 않고 물품 구매에만 신경을 쓰는 모습을 보였으며[3] 여기에 전자상거래 서비스 업체의 보안 허점이 적시에 발견 및 수정되지 않아 서비스 이용자의 온라인 현금자산이 공격자에 의해 무단 출금되는 보안사고가 발생하였다[4]. 이는 전자상거래 서비스 이용자의 보안 의식 개선이 필요함을 보여줌과 동시에, 이와 무관하게 강력한 보안 수준을 가진 플랫폼이 구축 되어야함을 반증한다. 이러한 현실을 배경으로 해당 논문에서는 전자상거래의 등장과 함께 도입된 온라인 현금자산에 대한 보안을 중심으로 개발 과정에서 고려하여야 하는 보안 기법인 암호화 프로토콜 사용, 세션 관리, 무차별 대입 공격 방어, 로그인 모니터링, 추가 패스워드 활용에 대한 조사 및 연구를 수행하였다.

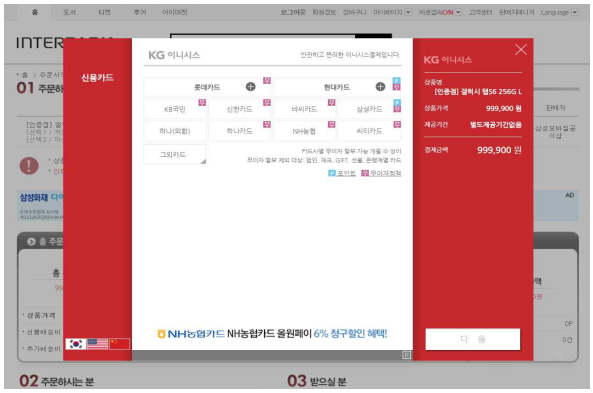
I. 서론

2000년대 전자상거래가 활성화되며 관련 업체들의 수가 크게 증가하였다[5]. 전자상거래 업체들은 규모가 커지자 전자결제 업무를 전자결제 서비스 제공자에게 위임하였고, 그 결과 안전한 전자상거래 서비스를 제공할 수 있게 되었다. 전자결제 서비스 아웃소싱은 현금자산을 보호하며 안전한 전자거래 환경을 조성하는데 기여했지만, 전자상거래 업체 단독으로 관리하는 온라인 현금자산은 여전히 위협에 노출되어 있다.

온라인 현금자산은 온라인 상에서 현금과 동일하게 혹은 유사하게 사용할 수 있는 현금자산이다. 많은 전자상거래 업체는 신용카드 수수료 절감, 사용 편의성 증진을 통한 고객 유치 등의 목적으로 자사에서 사용 가능한 전자화폐를 발행하며 사용을 유도하는데 이 전자화폐는 전자거래의 보안을 담당하는 외부 업체와 무관

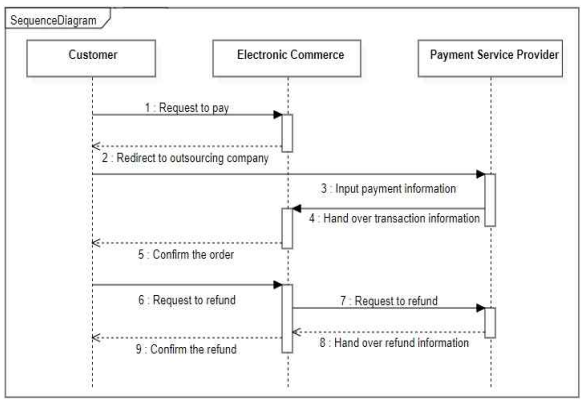
본 연구는 한국기술정보연구원(KITRI) 지원 및 차세대 보안 리더 양성 프로그램(Best of the Best)의 관리로 수행되었습니다.

[1]한국과학기술원 정보보호대학원 (yh8237@kaist.ac.kr)
[2]동국대학교 경주캠퍼스 컴퓨터공학과 (jmxx.98@gmail.com)
[3]순천향대학교 (abysswhale@gmail.com)
[4]한양대학교 원자력공학과 (wiwiwi79@hanyang.ac.kr)
[5]동국대학교 경주캠퍼스 컴퓨터공학과 (lymmok1@gmail.com)
[6]대구가톨릭대학교 정보보호학과 (necropolis2567@gmail.com)
[7]LG CNS (hjkim0892@gmail.com)

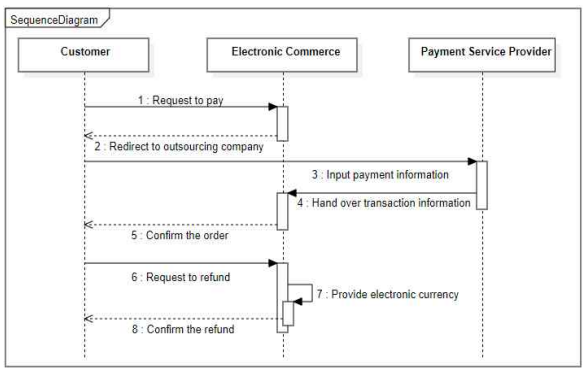


[그림 1] 국내 전자상거래 업체 인터파크에 전자결제 서비스를 제공하는 KG 이니시스

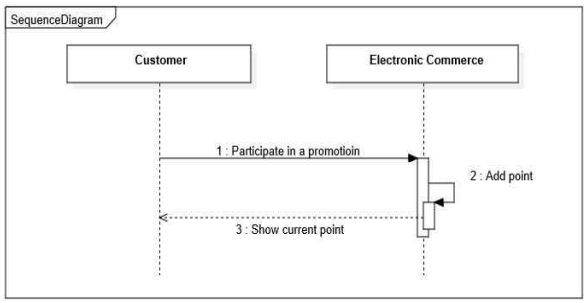
하게 발행되며 관리된다. 구매한 물건을환불하는 경우 거래 취소는 전자결제 서비스 제공자를 통해 이루어져야 하지만 전자화폐로 환불을 하는 경우 단순 취소처리와 함께 자체적으로 전자화폐를 적립한다. 전자상거래 업체에서 마케팅을 목적으로 포인트를 적립하는 경우에도 전자상거래 업체만이 관여한다.



[그림 2-1] 환불 대금을 현금으로 반환할 때의 시컨스 다이어그램



[그림 2-2] 환불 대금을 전자화폐로 반환할 때의 시컨스 다이어그램



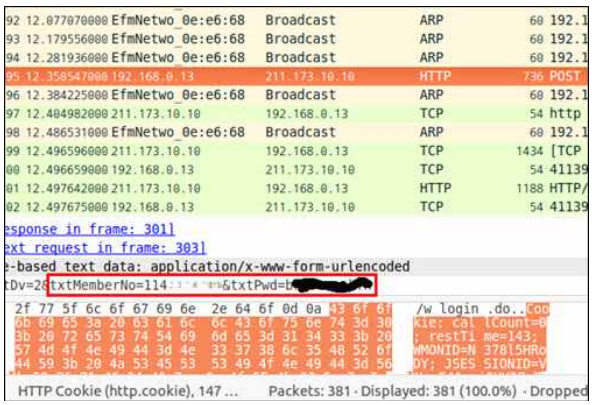
[그림 2-3] 전자상거래 서비스 제공자가 이용자에게 포인트를 적립할 때의 시컨스 다이어그램

이렇게 적립된 온라인 현금자산은 현금과 동일하게 사용될 수 있음에도 금융회사나 외부 보안 업체의 관리를 받지 않으며, 오직 전자상거래 업체만이 관리에 관여한다. 이 경우 전자금융거래법 3장 21조에 근거하여 전자상거래 업체는 선량한 관리자로서 고객의 온라인 현금자산을 보호하기 위한 자체적인 노력을 다할 의무가 있다. 본 논문의 본론에서는 서비스 이용자의 온라인 현금자산을 안전하게 보호하기 위한 보안 기법을 소개하고, 관련 연구 및 실제로 적용된 사례를 연구 및 조사하였다.

II. 적용 가능한 보안 기법

2.1 암호화 프로토콜 사용

서비스 이용자가 로그인을 시도할 때 입력한 계정 정보가 웹 서버로 전송된다. 이때 정보들이 암호화되지 않은 평문으로 전송된다면 패킷 캡처를 통해 손쉽게 계정 정보 스니핑이 가능하다. 만일 악의적 의도를 가진 사용자가 스니핑한 계정으로 로그인하여 온라인 현금자산을 무단으로 사용한다면 직접적인 금전 피해가 생길 것이다. 공격 재현이 어렵지 않고 피해가 크다는 점에서 반드시 적용되어야 할 기본적인 보안 기법이다. 해당 취약점은 2015년 코레일 예매 페이지에서 발견되어 한국인터넷진흥원(KISA)에 제보된 바 있다. 이를 막기 위해 암호화 프로토콜인 HTTPS를 사용한 통신이 권장된다.



[그림 3] 코레일 평문 전송 취약점 발견 당시 확인된 패킷[6]

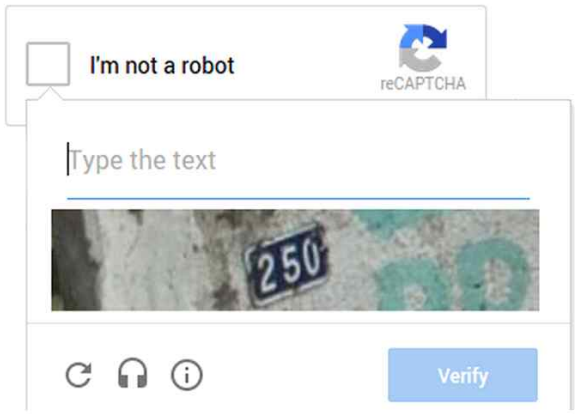
2.2 세션 관리[7]

세션은 사용자 정보를 저장하는 방법의 일종이며, 사용자가 로그인에 성공하면 웹 서버 측에서 무작위로 생성하는 값을 의미한다. 상태가 유지되지 않는 HTTP 프로토콜의 단점을 보완하기 위해 사용된다. 브라우저는 세션 값을 이용하여 로그인을 유지하는데, 이는 공격자가 세션을 한 번 탈취하면 세션이 만료될 때까지 무단 로그인이 가능함을 의미한다. 세션 하이재킹 공격을 통해 세션 탈취가 가능한데, 이러한 공격을 방지하기 위한 여러 기법을 제시하고자 한다.

먼저, 세션은 예측 불가능한 무작위 값을 가져야 하며 동시에 과도하게 짧거나 의미 있는 값을 가져서는 안 된다. 또한, 세션은 생성될 때마다 다른 값을 가져야 하고, 각 세션 값은 적절한 시간 뒤에 반드시 파기되어야 한다. 마지막으로, 2.1에서 활용한 암호화 기법을 적용하여 세션을 전송한다면 보다 안전한 세션 관리가 가능하다. 추가로, HTTP only 속성을 사용하여 스크립트 코드로 쿠키값에 접근하는 행위를 방지할 수도 있다.

2.3 무차별 대입 공격 방어

무차별 대입 공격(Brute Force Attack)은 원시적인 공격 기법이지만 네트워크 환경에서 여전히 유효하며 빈번히 일어나는 공격이다[8]. 해당 기법을 방어하기 위한 여러 연구가 진행되고 있지만 가장 쉬운 방법으로 reCAPTCHA를 사용할 수 있다. reCAPTCHA는 컴퓨터가 쉽게 읽을 수 없는 이미지 형태의 텍스트를 제시한 뒤 해당 텍스트를 입력하여야만 이후 작업이 가능하도록 하는 기능이다. 반복된 작업이 감지되었을 때 해당 기능이 실행되도록 하면 크롤러나 스크립트 코드로 수행되는 반복 작업을 막을 수 있으므로 무차별 대입 공격을 어느 정도 무력화시킬 수 있다.



[그림 4] 무차별 대입 공격을 방어하는 reCAPTCHA 기법[9]

2.4 로그인 모니터링

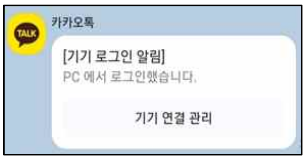
로그인을 시도한 장소에 근거한 추가 본인인증 요청이나, 다중 로그인 방지로 무단 로그인을 어느 정도 무력화시킬 수 있다. 원거리 통신망에서 계정 정보 유출이 일어났다면, 공격자는 최초 로그인 위치에서 멀리 떨어진 곳에서 로그인을 시도할 것이다. 이를 인지하여 추가적인 인증을 요구하거나 로그인을 거부할 경우 계정 정보 유출로 인한 피해를 완화시킬 수 있다.

다중 로그인 방지는 하나의 계정으로 두 개 이상의 디바이스에 동시에 로그인할 수 없도록 하는 기능이다. 이미 로그인이 되어 있는 상태에서 또 다른 로그인이 시도되면 기존 사용자에게 로그인 시도가 있음을 알리고, 불법 로그인을 확인한 사용자는 타 기기의 로그인을 즉시 해제할 수 있도록 개발을 한다면 공격 피해자는 계정 정보 유출을 빠르게 알아챌 수 있

다. 또한, 비밀번호를 바꾸고 서비스 제공자에게 신고하는 등 빠른 후속 조치가 가능하다. 국내 메신저 서비스 카카오톡은 PC 환경에서 정상 이용자 이외의 로그인이 시도될 경우 알림을 보내며, 정상 이용자의 모바일 디바이스로 본인 이외의 로그인을 강제로 해제할 수 있도록 개발해두었다.



[그림 5-1] 불법 로그인으로 인해 정상 이용자의 PC 로그인이 해제된 모습



[그림 5-2] 본인 이외의 로그인 기록이 정상 사용자의 모바일 디바이스에 전달된 모습



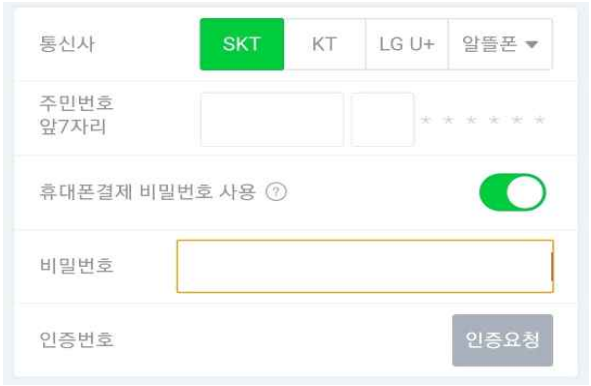
[그림 5-3] 불법 로그인을 확인한 정상 이용자가 타 PC의 로그인을 강제로 해제한 모습

2.5 추가 패스워드 활용

서비스 제공자는 일부 민감한 작업에 대하여 로그인 이후에도 추가적인 패스워드를 요구할 수 있다. 이를 통해, 공격자가 세션을 탈취하더라도 생길 수 있는 금전 피해를 완화 시킬 수 있다. 통신사 SKT는 휴대폰 결제 시 비밀번호를 추가로 입력하도록 하여 SMS 스니핑으로 인한 무단 결제를 방지하고 있다. 또한, 카 셰어링 서비스 그린카의 경우 결제 시 L포인트를 사용할 수 있는데, L포인트를 사용하기 위해서는 포인트 사용 비밀번호를 입력하도록 하여 포인트 무단 사용을 방지하고 있다.

III. 결론

본 논문에서는 가장 먼저 전자상거래의 확대와 함께 늘어난 온라인 현금자산의 보급에 대하여 설명하고 전자화폐와 포인트 서비스를 예



[그림 6-1] 휴대폰 결제 시 추가 결제 비밀번호를 요구하는 SKT



[그림 6-2] 포인트 사용 시 비밀번호를 요구하는 L포인트

로 제시하였다. 이후 서비스 제공자 측의 보안 강화의 필요성을 제시한 뒤, 보안 강화를 위해 적용될 수 있는 여러 가지 보안 기법들을 소개하였다. 전자상거래 서비스 이용자들이 보안 문제를 심각하게 생각하지 않는[2] 현실 속에 서비스 내부 보안 강화는 앞으로 일어날 보안 사고를 예방하기 위해 필수불가결한 것으로 보인다. 서비스 제공자는 선량한 관리자로서 의무를 다하기 위해 개발 및 관리에 있어 현실적으로 적용 가능한 모든 보안 기법들을 적용할 필요가 있다.

[참고문헌]

[1] TEO, Thompson SH; YU, Yuanyou. Online buying behavior: a transaction cost economics perspective. Omega, 2005, 33.5: 451-465.

[2] How big is the global e-commerce market?, https://hackernoon.com/how-big-is-the-global-e-commerce-market-93920e61f687 [Accessed : Oct 28, 2019]

[3] CHO, Jinsook. Likelihood to abort an online transaction: influences from cognitive evaluations, attitudes, and behavioral variables. Information & Management, 2004, 41.7: 827-838.

[4] SAXENA, Sameer, et al. Survey on Online Electronic Paymentss Security. In: 2019 Amity International Conference on Artificial Intelligence (AICAI). IEEE, 2019. p. 756-751.

[5] 한국 온라인 쇼핑 협회, 2015 온라인쇼핑 동향과 전망, http://www.korcham.net/FileWebKorcham/OnlineSeminar/File/20151202_05.pdf [Accessed : Oct 28, 2019]

[6] 설날 열차표 예매사이트, 평문전송 취약점 발견, https://www.boannews.com/media/view.asp?idx=45037&kind=1 [Accessed : Oct 28, 2019]

[7] WEDMAN, Shellie; TETMEYER, Annette; SAIEDIAN, Hossein. An analytical study of web application session management mechanisms and HTTP session hijacking attacks. Information Security Journal: A Global Perspective, 2013, 22.2: 55-67.

[8] NAJAFABADI, Maryam M., et al. Machine learning for detecting brute force attacks at the network level. In: 2014 IEEE International Conference on Bioinformatics and Bioengineering. IEEE, 2014. p. 379-385.

[9] Google reCaptcha Bypass Technique Uses Google's Own Tools, https://threatpost.com/google-recaptcha-bypass-technique-uses-googles-own-tools/124006/ [Accessed : Oct 29, 2019]

< 저 자 소 개 >



박 용 현 (Yonghyeon Park)
2019년 8월 : 성균관대학교 컴퓨터공학과 졸업
2019년 9월~현재 : 한국과학기술원 정보보호대학원 석사과정

관심분야 : 네트워크 보안, 안티 재밍



조 민 주 (Minju Jo)
2016년 3월~현재 : 동국대학교 경주캠퍼스 컴퓨터공학과 재학

관심분야 : 블록체인, 암호



김 남 형 (Namhyeong Kim)
2014년 3월~현재 : 순천향대학교 정보보호학과 재학

관심분야 : 시스템 최적화



정 재 영 (Jaeyoung Jeong)
2017년 3월~현재 : 한양대 원자력공학과 재학

관심분야 : 리버스 엔지니어링



이 경 민 (Kyungmin Lee)
2014 3월~현재 : 동국대학교 경주캠퍼스 컴퓨터공학과 재학

관심분야 : 소프트웨어 공학



이 준 범 (Junbeom Lee)
2013년 3월~현재 : 대구가톨릭대학교
정보보호학과 재학

관심분야 : 빅데이터



김 홍 진 (Hongjin Kim)
2001년~현재 : LG CNS 재직

QR코드와 추가인증서를 이용하는 간편결제 웹서비스

윤정민, 박종훈, 권혁민, 마민기, 이병천

중부대학교 정보보호학과

Easy Payment Web Service Using QR code and Additional Certificates

Jeongmin Yoon, Jonghun Park, Hyukmin Kwon, Mingi Ma, Byoungcheon Lee

Division of Information Security, Joongbu University

요 약

현재 널리 사용되는 결제방식은 공인인증서 기반의 전자서명 결제와 휴대폰에 모바일 간편결제 앱을 설치하여 사용하는 등의 방법이 이용되는데, 편의성 때문에 모바일 간편결제로 시장이 이동하고 있다. 공인인증서 기반의 전자결제는 사용자가 다수의 결제서비스를 이용하거나 복수의 컴퓨팅 기기들을 사용하게 되는 실제 환경에서는 인증서 관리가 복잡하다는 단점이 있고, 모바일 간편결제는 부인방지 서비스의 기본인 인증서를 사용하지 않고 앱과 서비스의 신뢰성에 크게 의존한다는 단점이 있다. 본 논문에서는 이 두 가지 접근방법을 모두 포괄하는 인증서 기반 간편결제 모바일 웹서비스를 개발하였고 이의 상용화 타당성을 검토하였다. 이 서비스에는 이중토큰을 이용한 인증유지, 추가인증서를 이용한 인증서 발급 및 관리, QR코드를 이용한 리다이렉트 결제 등의 기능을 포함하여 사용자는 어떤 환경에서도 표준 웹브라우저만을 이용하여 편리하게 간편결제를 사용할 수 있도록 하였다.

I. 서론

현재 널리 사용되는 결제방식은 공인인증서 기반 전자서명 결제와 모바일 간편결제의 두 가지 방식으로 나누어 볼 수 있는데, 사용자 편의성 문제로 공인인증서 사용을 지양하고 모바일 간편결제로 시장이 크게 이동하고 있다.

첫째, PC 환경에서 공인인증서를 이용한 전자서명 결제방식에서는 인증서를 활용할 수 있도록 인증서 관련 플러그인 프로그램을 브라우저에 설치하여야 하는데 이 경우에는 비표준 플러그인 프로그램을 설치해야 한다는 점, 인증서를 관리하기 어렵다는 점에서 많은 불편함이 있다. 특히 사용자가 하나의 공인인증서를 여러 컴퓨터에 복사하여 사용하는 것이 불편하기도 하고 보안성이 취약하다는 점이 비판되고 있다. 이런 이유로 인증서를 사용하지 않는 모바일 간편결제 서비스로 시장이 이동하고 있다.

둘째, 모바일 앱을 이용하는 모바일 간편결제 방식은 부인방지 서비스의 기본인 인증서를 사용하지 않고 앱과 서비스의 신뢰성에 크게 의존해야 한다는 단점이 있다.

한편 사용자가 여러 대의 컴퓨팅 기기, 모바일 디바이스를 사용하는 경우에는 인증서 및 인증키의 관리에 불편함이 있다. 이를 해결하기 위한 방식으로는 첫째, 하나의 디바이스에 인증키를 안전하게 관리하고 다른 컴퓨터를 이용하게 될 때는 결제를 안전한 디바이스로 리다이렉트 하여 결제하도록 하는 방식이 금융권을 중심으로 널리 개발되고 있다. 두 번째 방식으로는 하나의 마스터 디바이스에 마스터인증서를 인증기관으로부터 발급받고 추가 기기에는 사용자가 직접 추가인증서를 발급 및 관리하여 복수의 디바이스로도 인증서 기반의 결제를 수행하도록 하는 방식이 제안되었다[2,3].

이 논문에서는 QR코드를 이용하여 결제를 리다이렉트하여 수행하는 방식과 추가인증서를 이용하여 어느 디바이스를 사용하더라도 인증서 기반의 결제를 가능하도록 하는 기능을 함께 웹서비스로 구현하는 사례를 제시한다. 사용자는 모바일 디바이스에 특정 앱을 설치하지 않고도 표준 웹브라우저를 이용하여 인증서 기반의 결제를 진행할 수 있게 되었다. 이

러한 서비스의 운영 사례를 기반으로 이러한 기술의 상용화 타당성을 검토한다.

II. 관련연구

2.1 기존의 간편결제 서비스

현재 국내외적으로 많은 간편결제 서비스들이 개발 및 서비스되고 있다. 국내 서비스로는 카카오페이, 삼성페이, 네이버페이, 페이코 등이 있고 국외 서비스로는 애플페이, 알리페이, 위챗페이 등이 있다. 이들 서비스는 사용 기술 및 기능들이 각기 다르지만, 부인방지의 기반이 되는 기술인 인증서를 사용하지 않고 사용자에게 편의성을 제공하는 간편결제 서비스를 추구하고 있다는 공통점이 있다.

2.2 이중토큰을 이용한 인증유지 [1]

OAuth2.0, JWT(JSON Web Token) 방식의 토큰 인증은 서버가 인증된 클라이언트에게 발급하는 토큰을 서비스 요청 시 서버에 반복해서 전송하여 인증상태를 유지하는 방식을 사용하는데 이것은 도청 공격에 취약하므로 HTTPS와 같은 보안통신 기술과 함께 사용되어야 한다는 제약이 있다.

이중토큰을 이용한 인증유지 기술은 보안통신 채널을 이용하지 않고도 안전한 무상태 인증을 제공할 수 있도록 개선한 토큰인증 기술이다. 사용자가 서버에 초기인증에 성공하면 서버는 추후 인증을 위해 사용자에게 공개토큰과 비밀토큰의 쌍을 발급한다. 공개토큰은 사용자 정보 및 유효기간에 서버의 HMAC 서명 값을 포함하는 토큰인데 이것은 인증된 사용자임을 나타내기 위해 서버에 반복적으로 전송하는 정보로서 서명된 ID와 같은 역할을 한다. 비밀토큰은 사용자의 공개토큰에 대해 서버의 HMAC 서명이 포함된 토큰으로 외부로 노출되지 않고 일회용 인증정보 계산에만 사용하는 정보로서 서명된 패스워드와 같은 역할을 한다.

클라이언트가 서버에 인증요청 시 현재 시각과 비밀토큰을 이용하여 일회용 인증정보를 계산하고 <공개토큰, 현재시간, 일회용 인증정보>를 서버로 전송하게 된다. 서버는 사용자의 공개토큰으로부터 사용자의 비밀토큰을 계산하게 되고 일회용 인증정보의 유효성을 검증하여 클라이언트를 인증하게 된다. 이러한 방식은 서버가 사용자의 비밀토큰을 유지, 관리할 필요가 없어서 무상태 인증이 가능하고 현재 시각에 기반한 일회용 인증정보는 재사용할 수 없어서 보안통신을 사용하지 않아도 안전하다는 장점이 있다.

2.3 추가인증서를 이용한 인증확장 [2,3]

사용자가 복수의 컴퓨팅 기기들을 사용하게 되는 경우 인증서 기반의 전자서명을 사용하기 위해서는 하나의 인증서를 여러 대의 컴퓨팅 기기들에 복사하여 사용하는 것이 지금까지의 일반적인 방법이었으나 인증서와 개인키를 복사하여 사용하는 것은 불편하기도 하고 매우 위험한 일이다. 추가인증서를 이용한 인증확장 방식은 사용자가 하나의 인증기관으로부터 발급받은 하나의 마스터인증서를 기반으로 스스로 필요한 만큼 추가인증서를 발급하여 사용할 수 있도록 하는 방식이다.

마스터인증서는 서버가 인증된 사용자의 마스터 컴퓨터에 최초로 발급해주는 인증서이다. 다른 컴퓨터에서도 인증서를 사용할 필요가 있는 경우 사용자는 자신의 마스터인증서로 추가인증서를 직접 발급 및 배포하여 사용할 수 있다. 사용자는 자신이 발행하여 사용하고 있는 추가인증서의 현황을 확인하고 삭제 및 비활성화할 수 있어야 한다.

2.4 QR코드를 이용한 결제정보 리다이렉트

QR코드는 1차원 바코드와는 다르게 숫자로는 최대 7098자의 많은 데이터를 담을 수 있고, 오류 정정 기능 및 디자인 분야에서도 많은 이점을 가지고 있어서 직접 대면한 환경에서 정보를 직접 전달하는 용도에 매우 유용하다. 중국의 Alipay, Wechatpay에서는 QR코드를 카메라로 인식하면 결제로 연동되도록 하여 간편결제 서비스로 크게 성공하였다.

III. 간편결제 웹서비스 개발

3.1 시스템 개요

우리는 QR코드를 이용하여 결제페이지를 리다이렉트 하는 간편결제 기능, 인증서를 사용한 전자서명 결제, 복수의 기기들을 이용하는 때도 추가인증서를 이용한 인증서 관리 기능, 이중토큰을 이용한 인증유지 기능을 가지는 전자결제 웹서비스를 개발하는 것을 목표로 하였다. 사용자는 여러 대의 컴퓨터에 발급한 추가인증서를 이용하여 직접 전자서명 결제를 할 수도 있고, QR코드를 이용하여 결제정보를 다른 모바일 디바이스로 리다이렉트하여 전자서명 결제를 하도록 할 수도 있다.

이러한 웹서비스를 구현하기 위하여 다음의 [표 1]에 표시한 것과 같은 기술들을 사용하였다.

[표 1] 사용된 구현 기술

분야	기술
back-end	node.js, express
front-end	vue.js
DB	MySQL
cloud service	Heroku
암호 라이브러리	node-forge, bcrypt
토큰인증	jwt

현재 구현된 모델 서비스에서는 MySQL DB에 다음과 같은 테이블들이 운영된다.

- 사용자정보 테이블: 등록된 사용자 정보 관리
- 인증서 테이블: 사용자에게 발급하는 마스터인증서 및 추가인증서를 저장
- 장바구니 테이블: 사용자가 선택하는 상품을 저장
- 결제정보 테이블: 사용자가 결제한 정보를 저장
- 영수증 테이블: 판매자가 결제정보에 대해 서명한 영수증을 저장

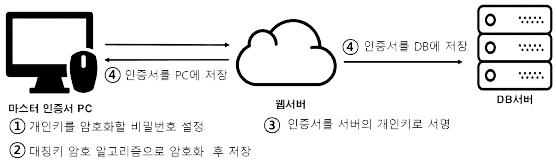
3.2 사용자등록, 로그인 및 인증유지

사용자는 서버에 계정을 등록할 수 있다. 사용자는 가입된 계정으로 아이디, 패스워드를 입력하고 로그인에 성공하게 되면 서버는 향후의 인증유지에 사용하기 위한 공개토큰과 비밀토큰의 쌍을 발급한다. 사용자의 컴퓨터는 브라우저의 로컬스토리지에 공개토큰과 비밀토큰을 저장한다. 이때 서버는 사용자에게 발급한 토큰들을 저장할 필요가 없다.

로컬스토리지에 공개토큰과 비밀토큰이 저장되어 있으면 로그인된 상태가 유지된다. 로그인된 상태에서 클라이언트가 서버에 서비스를 요청할 때에는 비밀토큰과 현재 시각을 이용하여 일회용 인증정보를 생성하고 이것을 현재시간, 공개토큰과 함께 서버에 전송한다. 서버는 공개토큰으로부터 비밀토큰을 계산하고 일회용 인증정보의 유효성을 검증한 이후 유효한 경우에만 서비스를 제공한다. 이 토큰은 유효기간이 설정되어 있는데 유효기간 이내에는 다시 로그인할 필요 없이 로그인된 상태를 유지하는데 토큰의 유효기간이 지나면 서버는 다시 로그인할 것을 요구하게 된다.

3.3 마스터인증서와 추가인증서 발급과 관리

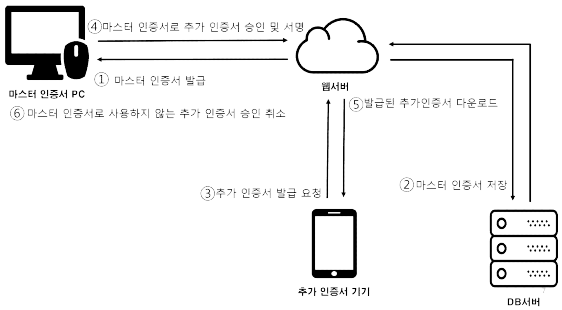
회원가입 후 처음 로그인에 성공하면 사용자는 서버에게 마스터인증서 발급을 요청할 수 있다. 마스터인증서를 발급받은 기기는 사용자의 마스터기기가 되며 사용자는 이것을 안전하게 관리하여야 한다. 마스터인증서는 브라우저의 로컬스토리지에 저장되며 개인키는 패스워드로 암호화하여 로컬스토리지에 저장된다.



[그림 1] 마스터인증서 발급

사용자가 또 다른 추가 컴퓨팅 기기에 추가인증서를 발급하여 사용하고자 하는 경우 추가 기기에서 로그인한 후 추가인증서 발급 신청을 할 수 있고 신청내역은 서버의 DB에 저장된다. 이때 사용자 아이디 이외에 기기를 구별하기 위한 기기 아이디 정보를 입력하게 된다. 이후 동일 사용자는 마스터기기로 로그인하여 신청내역을 검색할 수 있고 추가인증서 발급을 승인하여 추가인증서를 발급한다. 추가인증서는 마스터인증서의 개인키로 서명한 개인용 인증서가 된다. 동일 사용자는 추가 기기에서 서비스에 다시 접속하면 추가인증서 발급 내역을 검색할 수 있고 기기에 인증서를 다운로드하게 된다. 추가인증서 및 개인키는 브라우저의 로컬스토리지에 저장된다.

사용자는 서버에 로그인하면 인증서 관리 메뉴를 통해 자신의 인증서 발급 내역을 검색할 수 있고 기기 분실 및 고장 등으로 인해 추가인증서를 사용하지 못하게 되는 경우 해당 추가인증서를 삭제 또는 비활성화할 수 있다. 그러므로 사용자는 자신이 사용하는 마스터인증서 및 추가인증서의 내역을 언제든지 검색해볼 수 있고 직접 관리할 수 있어서 관리의 편의성과 안전성을 확보할 수 있다.



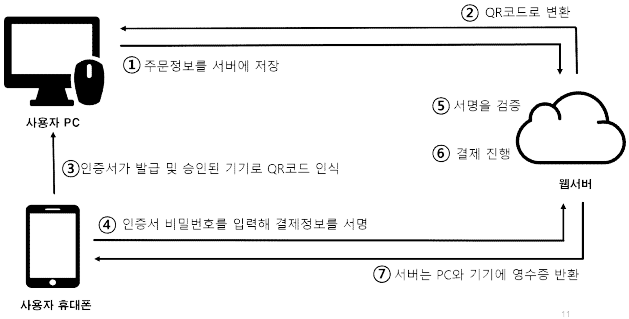
[그림 2] 추가인증서 발급 절차

3.4 상품 구매 및 결제 절차

사용자는 구매할 상품들을 선택하여 구매 버튼을 누르면 클라이언트는 구매요청 정보를 웹서버로 전송하고, 서버는 지불요청 페이지를 표시하게 된다. 사용자는 현재 사용 중인 기기에서 마스터인증서 또

는 추가인증서를 이용하여 직접 전자서명 지불을 할 수도 있고 또 다른 추가인증서가 장착된 휴대폰으로 QR코드를 스캔하여 지불정보를 휴대폰으로 리다이렉트하여 전자서명 지불을 할 수도 있다.

[그림 3]은 QR코드를 이용하여 결제를 휴대폰으로 리다이렉트하여 수행하는 대표적인 결제 시나리오를 보여준다.



[그림 3] QR코드 방식의 결제 시나리오

사용자가 상품을 선택해 구매요청을 하면 주문정보를 서버로 보내 서버에서는 DB에 주문정보를 만들고 결제 여부에 False를 넣는다. 서버에서는 주문정보에서 주문번호를 추출하여 <주문정보, 현재시간>을 전달하기 위한 QR코드를 만들어 화면에 표시한다. 사용자는 로그인된 스마트폰으로 해당 쇼핑몰의 모바일 페이지의 결제페이지로 이동하여 카메라를 활성화하고 QR코드를 인식하면 주문정보와 결제정보가 스마트폰의 화면에 나타난다. 사용자가 결제승인 버튼을 누르면 스마트폰에 발급된 추가인증서를 이용하여 결제정보에 서명하고 서버에 전송한다. 서버가 결제를 검증하여 정상으로 판정되면 결제 완료 페이지로 리다이렉트한다.

3.5 구현 및 데모

1) 서비스 운영: 현재 구현된 모델 서비스는 클라우드 PaaS인 Heroku에 포팅하여 다음의 주소에서 서비스되고 있다.

https://coconutpay.herokuapp.com/

2) 인증 및 로그인: 사용자 가입 후 성공적으로 로그인하면 [그림 4]에 보인 바와 같이 서버가 발급한 이중토큰이 브라우저의 로컬스토리지에 저장된다. 이 상태가 되면 사용자는 다시 로그인할 필요 없이 토큰의 유효기간(현재 7일) 동안 로그인된 상태로 유지되며 서버를 이용할 수 있다.



[그림 4] 로컬스토리지에 저장된 이중토큰

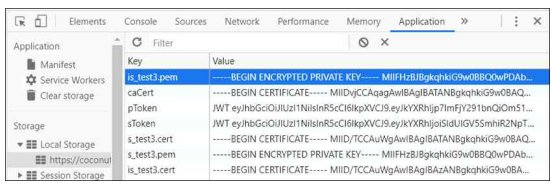
3) 인증서 발급 및 관리: 로그인한 사용자는 '인증센터'를 통해 마스터 PC에서 마스터인증서를 발급받을 수 있고, 동일한 아이디로 다른 기기에서 로그인하면 '추가인증서 발급'을 신청할 수 있다. [그림 5]에 보인 바와 같이 추가인증서 발급 신청을 하면 동일한 사용자는 마스터 PC에서 '인증서 관리' 메뉴를 통해 승인 버튼을 누르면 추가인증서가 발급되며, 동일 사용자는 추가 기기에서 이것을 다운로드하여 사용하게 된다. 또한, 이미 발행하여 사용하던 추가기기의 추가인증서를 사용하지 않고자 할 때는 [그림 6]의 '인증서 관리' 메뉴를 통해 해당 인증서를 비활성화할 수 있다. [그림 7]는 브라우저의 로컬스토리지에 저장된 인증서를 보여준다.



[그림 5] 인증서 발급, 관리를 위한 인증 센터



[그림 6] 인증서 관리 기능



[그림 7] 로컬스토리지에 저장된 인증서

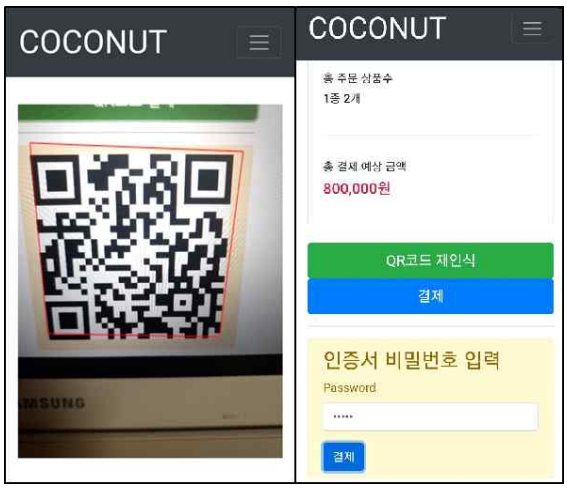
4) 주문 및 결제: 결제 단계에서는 현재 사용 중인 기기에 발급되어 설치된 추가인증서를 이용하여 직접 전자서명 결제를 하는 '바로 결제'와 QR코드를 이용하여 휴대폰으로 리다이렉트하여 결제를 진행하는 'QR코드 결제' 중에서 선택하여 결제할 수 있다. '바

로 결제'의 경우 현재의 기기에 마스터인증서 또는 추가인증서가 있는 경우 진행할 수 있으며 바로 인증서의 비밀번호를 입력하여 전자서명을 통해 결제가 진행된다. 'QR코드 결제'를 진행하는 경우 화면에 주문정보와 현재 시각이 담긴 QR코드를 화면에 표시하게 되며, 이 QR코드를 인증서가 보관된 모바일 기기의 카메라로 인식해 결제를 진행하게 된다.

휴대폰으로 QR코드를 인식하게 되면 QR코드에 담긴 주문정보를 화면에 표시하게 되며, 결제버튼을 클릭하고 인식한 기기의 인증서의 비밀번호를 입력하여 결제를 진행하게 된다.



[그림 8] 인증서 유무에 따라 결제방식 선택



[그림 9] 모바일 기기를 통한 QR코드 리다이렉트 결제

5) 영수증 발행: 사용자의 지불이 완료되면 판매자가 서명한 영수증을 발급하며 이것은 DB에 저장된다. 사용자는 거래 내역 페이지에서 판매자의 영수증 정보를 확인, 검증할 수 있다.

3.6 서비스의 활용성 분석

현재 구현된 서비스가 제공하는 특징적인 기능들은 간편결제 서비스에서 활용하기에 매력적인 기능들이라고 생각된다.

1) 이중토큰을 이용한 인증유지: 한번 로그인되면 토큰의 유효기간 동안 다시 로그인할 필요 없이 인증이 유지된다. 더구나 이러한 인증상태는 비밀토큰을 이용한 일회용 인증정보를 검증하므로 https 보안통신을 사용하지 않고도 인증이 안전하게 유지된다.

2) 인증서 기반 간편결제 웹서비스: 특정 앱을 설치하지 않고도 표준 웹브라우저를 이용하여 결제할 수 있도록 표준 웹서비스 기술들을 이용하여 전자서명 지불을 구현하였다. 기존의 간편결제 서비스에서는 제공하지 못했던 부인방지 가능한 인증서 기반 간편결제 기능을 제공한다.

3) 추가인증서를 이용한 복수기기 인증서 활용 환경: 사용자가 필요한 추가 기기의 수량만큼 추가인증서를 스스로 발행하여 사용할 수 있다.

4) 인증기관에 의존하지 않는 사설인증서 활용: 인증기관이 발행하는 공인인증서를 여러 서비스에 통용하여 사용하도록 시도하는 것은 경로검증 등이 복잡하고 인증서 관리에도 많은 어려움이 있다. 여기에서는 현재 로그인된 서버에서 자동으로 발급하고 브라우저에서 자동으로 관리되는 사설인증서를 해당 서비스에서만 이용하는 것이므로 인증모델이 간단하고 사용자의 이용도 편리하다.

5) 인증서의 안전한 관리: 여러 서비스에 통용하여 사용되는 것을 목표로 하는 공인인증서를 안전하게 관리하는 것은 매우 어려운 문제이며 개인키의 안전한 보관장소가 필요하다. 현재 구현된 서비스에서는 브라우저의 로컬스토리지에 패스워드 암호화하여 개인키를 보관하는데 이것은 해당 서비스에서만 접근하고 지불에 사용할 수 있다. 로컬스토리지에 저장되는 정보가 여러 가지 공격에 취약한 것은 사실이다. 그러나 이 방법은 브라우저에 의해 자동으로 관리되며 해당 서버에서만 사용될 수 있는 특수 인증서이고 사용자가 직접 관리할 수 있다. 사용자는 서버에 로그인하여 마스터인증서를 새로 발급받을 수도 있고 추가인증서도 직접 갱신할 수 있다. 무엇보다도 사용자가 자신의 인증서 사용 현황을 직접 파악하고 편리하게 관리할 수 있는 기능을 제공하여 편의성 및 안전성을 높일 수 있다.

IV. 결론

본 논문에서는 이중토큰을 이용한 인증유지, 추가인증서를 이용하는 복수기기 환경의 인증서 관리, 표

준 웹서비스 기술을 이용하는 인증서 기반 간편결제, QR코드를 이용한 리다이렉트 결제 등의 특징을 가지는 간편결제 서비스를 구현하였고 서비스의 활용성을 분석하였다. 이러한 특징적인 기술들은 간편결제 서비스 운영에 매우 유용하게 사용될 수 있다고 판단된다. 향후 실제 지불서비스에서 요구되는 여러 가지 기능들을 분석하고 구현하여 활용성을 더욱 높이는 연구개발이 필요하다.

[참고문헌]

[1] 이병천, “OAuth 2.0 MAC 토큰인증의 효율성 개선을 위한 무상태 난수화 토큰인증”, 정보보호학회논문지, Vol. 28, No. 6, pages 1343-1354, Dec. 2018.

[2] 이병천, “복수 K-FIDO 기기 환경에서의 인증키 관리”, 정보보호학회논문지, Vol. 27, No. 2, pages 293-303, Apr. 2017.

[3] 이병천, “하이브리드 인증을 위한 키관리서버 모델”, 보안공학연구논문지, Vol.13, No.1, pages 27-40, 2016.

Understanding Nine Central Bank Digital Currency Experiments Selected from the World

Edwin Ayisi Opare*, Kwangjo Kim**

*Global Information and Telecommunication Technology Program, KAIST

** School of Computing, KAIST

Abstract

Over the last decade, private-sector actors have issued several thousands of distributed ledger technology (DLT)-based digital currencies that neither possess intrinsic value nor are they backed by any tangible resources. The issuance of these private digital currencies such as Bitcoin, Ethereum and other altcoins threatens the stability of monetary policy and financial market infrastructures (FMIs). Facing the threat of disruption of monetary policy and financial market instability by such private sector issuances, many central banks have begun research and experimentation into the issuance of central bank-backed digital currencies (CBDCs) to guarantee financial market stability and monetary policy preservation. In this paper, we present a survey of nine CBDC experiments from the world with the goal of understanding the motivations and factors that influence the technical design considerations of the selected CBDC experiments.

Keywords: CBDC, digital currency, DLT, FMIs, monetary policy

I. Introduction

The invention of Bitcoin in 2009 by Satoshi Nakamoto[1] has given rise to the use of digital or crypto-currencies globally. In less than a decade since the introduction of Bitcoin, private sector actors have issued several thousands of digital currencies that lack intrinsic value and are not backed by any tangible resources[2].

Facing the threat of monetary policy and financial market instability by such private sector issuances, many central banks (CBs) have delved into research and experimentation on the issuance of central bank-backed digital currencies (CBDCs) to guarantee financial system stability and monetary policy preservation[3].

In a recent survey conducted by the Bank for International Settlements (BIS), of the 63 CB respondents of the survey, more than 70 per cent are currently investigating the possibility of issuing a CBDC[4].

In this paper, we present a survey of nine (9) completed CBDC experiments from the world to understand the motivations and factors that influence the technical design considerations of the selected experiments.

II. CBDC

A CBDC may be defined as monetary value similar to central bank-issued money that is stored electronically and represents a claim on asset on the CB[5]. It can be distributed in a decentralized manner and used to make payments.

CB-issued money from the perspective of accessibility are of two types: general-purpose (physical) money or cash, which is widely available and accessible to

the general-public; and wholesale electronic CB money (reserves or settlement balances) whose access is restricted to commercial banks (CMBs) and other high-value financial industry market participants[5].

Similar to the CB money types, there are two types of CBDCs: general-purpose CBDCs and wholesale CBDCs (W-CBDC). General purpose CBDCs can be broken down further into general-purpose account-based CBDC (GA-CBDC) and general-purpose value-based CBDC (GV-CBDC)[4].

The Bank for International Settlements (BIS) provides a further classification of money and CBDCs which are represented on Figure 1, the money flower. In Figure 1, the dark-gray shaded area represents the types of CBDCs issuable by a CB.

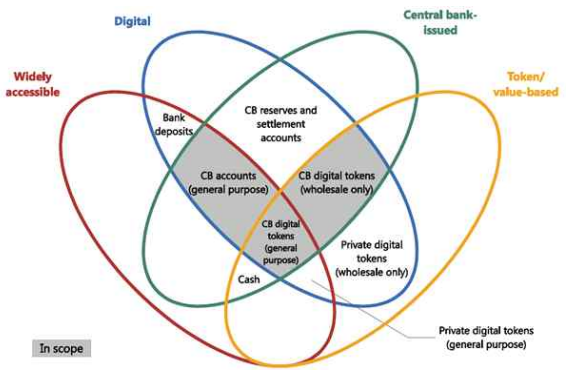


Figure 1: The money flower: a taxonomy of money (Source: BIS, 2019)

III. CBDC Experiment Selection

We search the World Wide Web for completed CBDC experiments and present our findings in this section. In particular, we crawl through resources of the World Economic Forum (WEF) which has been curating CBDC experiments of various types in one place at[6]. All CBDC experiments aggregated by the WEF as at August 10, 2019 are considered potential candidates for our survey subject to the fulfillment of our experiment selection criteria discussed in this section.

Additionally, we crawl the Whitepaper Database, a renowned data source in the cryptocurrency world where leading cryptocurrency projects such as Ethereum, Ripple, Tether, Stellar, and other altcoins were all published.

To qualify for inclusion in our survey, a CBDC experiment must meet the following criteria:

1. Must be published by the WEF, whitepaperdatabase.com or by the central bank/government implementing the CBDC experiment
2. Must have completed a proof-of-concept
3. Must have a detailed report about the experiment in English

Following our selection criteria, we present the selected CBDC experiments in Table 1 which summarize the candidate experiments surveyed in this paper.

Table 1. Selected CBDC Experiments

No.	Country	Experiment Name	Type of CBDC
1	Canada	Project Jasper	W-CBDC
2	Singapore	Project Ubin	W-CBDC
3	Hong Kong	None	W-CBDC
4	Brazil	None	W-CBDC
5	South Africa	Project Khokha	W-CBDC
6	Germany	BLOCKBASTER	W-CBDC
7	Thailand	Project Inthanon	W-CBDC
8	Bilateral 1 (Canada, Singapore)	Jasper-Ubin	W-CBDC
9	Bilateral 2 (EU, Japan)	Project Stella	W-CBDC

IV. CBDC Motivations and Technologies

In this section, we present the motivations and choice of technology (DLT or otherwise) for each selected CBDC experiment in Table 2.

Table 2. CBDC Experiment Motivations and Technologies

Country	Experiment Name	Motivation	DLT used
Canada	Project	Understand how	Ethereum

	Jasper	DLT could transform the future of payments in Canada	; Corda
Singapore	Project Ubin	Evaluate the implications of having a tokenized form of the Singaporean Dollar on DLT, and its potential benefits to Singapore's financial ecosystem.	Ethereum ; Corda; Hyperledger Fabric; Quorum
Hong Kong	N/A	Carry out an open-minded and in-depth examination of DLTs and identify possible applications of DLT in banking industry	Ethereum , Hyperledger Fabric
Brazil	Project SALT	Explore central bank use cases that could benefit from the potential of DLT	Ethereum ; Corda; Hyperledger Fabric; Quorum
South Africa	Project Khokha	Build a PoC wholesale payment system for interbank settlement using a tokenized South African rand on DLT	Quorum
Germany	BLOCKBASTER	Create a conceptual prototype for a Blockchain-based transfer and settlement of securities and cash	Hyperledger Fabric
Thailand	Project Inthanon	Access the potential of DLTs on Thailand's financial market infrastructure	Corda
Bilateral 1 (Canada, Singapore)	Jasper-Ubin	Enable cross-border high value transfer between different DLT platforms	Quorum, Corda
Bilateral 2 (EU, Japan)	Project Stella	Implement liquidity saving mechanism functionalities on DLT	Hyperledger Fabric, Elements, Corda

We note that all the CBDC experiments surveyed in this paper are W-CBDCs. Additionally, all the experiments were implemented on at least one of Ethereum, Corda, Quorum or Hyperledger Fabric DLT platforms.

V. Conclusion

In conclusion, we provide a summary of CBs motivation for undertaking CBDC experiments. We also highlight some of the pros and cons of existing DLT platforms.

1.1. Motivations for CBDC Experiments

CBs have indicated the following as their key goals for undertaking CBDC experiments:

- Availability: elimination of the single point of failure problem associated with the execution of transactions that rely on a central authority for payment, clearing, and settlement
- Performance: reduction in transaction processing time and cost as DLTs eliminate the need for third parties for transaction verification and validation, thereby removing transaction processing time and cost bottlenecks that otherwise exist with current financial market infrastructure (FMI) schemes
- Auditability: improved real-time payment, clearing and settlement monitoring and guarantees of transaction transparency by CBs and financial market participants
- Transparency: minimize the occurrence of fraudulent transactions through the use of a single shared ledger with capabilities for transaction traceability that allows for efficient transaction screening over relatively shorter times compared with screening for existing FMI implementations
- Cost reduction: operational simplification through the use of a shared ledger for improvements in back-office processing and reduction in reconciliation costs in securities clearing and settlement among transacting parties
- Trust: Elimination or reduction of counterparty risks through the use of

smart contracts for transaction automation to ensure that counterparties do not renege on their commitments in payment transactions

- Security: enhanced transaction data privacy due to the underlying public key cryptography systems underlying DLT implementations

1.1. Pros and Cons of DLTs

DLTs refers to a combination of technologies and capabilities that provides strong auditability and traceability guarantees to enable different actors to share in a trustless environment, access to the same data over multiple logical and geographic locations. Blockchain, a type of DLT popularized by Nakamoto's implementation and release of the Bitcoin core in 2009 [1].

While the bitcoin blockchain possesses a number of desirable attributes for the financial industry, a number of shortfalls with its original implementation undermine its suitability for FMIs.

Firstly, Bitcoin is a public, permissionless blockchain, meaning that anyone can join the Bitcoin network without a need for verification of their identity[7].

Secondly, the public nature of Bitcoin means that all Bitcoin transactions are publicly visible making it unsuitable for data privacy requirements required by FMIs and financial transactions[6].

Lastly, Bitcoin's consensus protocol, which is based on the Proof-of-Work (PoW) protocol, is resource intensive, requiring excessive amounts of energy and time to append new blocks to the transaction ledger of a blockchain system. In FMIs, payment transactions usually require a fraction of a second to be completed while Bitcoin only adds transactions to blocks and propagates such blocks to the blockchain ledger every 10 minutes[3].

To address the Bitcoin shortfalls, leading CMBs are collaborating with financial technology (Fintech) companies to develop permission-based DLT platforms as an alternative to the Bitcoin system.

Notable among the myriad of DLTs for the financial services industry includes the Quorum, Corda, and Hyperledger Fabric DLT platforms.

[References]

[1] S. Nakamoto. "Bitcoin v0.1 released." http://www.metzdowd.com/pipermail/cryptography/2009-January/014994.html

[2] V. Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform." [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[3] B. S. C. Fung and H. Halaburda, "Central Bank Digital Currencies: A Framework for Assessing Why and How," Bank of Canada, Ottawa, 2016.

[4] C. Barontini and H. Holden, "Proceeding with caution – a survey on central bank digital currency," Bank for International Settlement, 2019.

[5] W. Engert and B. S. C. Fung, "Central Bank Digital Currency: Motivations and Implications," Bank of Canada, Ottawa, 2017.

[6] World Economic Forum. "Central Bank/Macroeconomics DLT Research List. (accessed 13 October, 2019).

[7] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," October 31, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

블록체인 네트워크 기반의 암호화폐 동향 분석

임세진* 김현지* 서화정*†

*한성대학교 IT융합공학부

Analysis of trend of cryptocurrency based on blockchain network

Se-Jin Lim* Hyun-Ji Kim* Hwa-Jeong Seo*

*Division of IT Convergence Engineering, Hansung University.

요 약

최근 몇 년간 국내를 비롯하여 전 세계적으로 암호화폐에 대한 관심이 증가하고 있다. 암호화폐(Cryptocurrency)란 분산 원장(Distributed Ledger) 시스템 내에서 안전하게 전송되고, 해시 함수를 이용해 쉽게 소유권을 증명해 낼 수 있는 디지털 자산이다. 암호화폐는 법정화폐와 달리 화폐를 발행하는 중앙은행 없이 전 세계의 노드(node)에 P2P(Peer-to-Peer) 방식으로 분산 저장되어 운영된다. 일반적으로 암호화폐는 블록체인이나 DAG (Directed Acyclic Graph)를 기반으로 한 분산 원장 위에서 동작하는데, 본 논문에서는 블록체인 상에서 동작하는 암호화폐에 초점을 맞추어 블록체인 기술에 따른 암호화폐의 동향에 대하여 알아보고자 한다.

I. 서론

암호화폐는 본래 투자를 목적으로 만들어진 것은 아니지만, 이를 새로운 투자자산으로서 가치가 있다고 판단한 투자자들에 의해 한때 거래량과 가격이 폭등하기도 하였다. 이를 계기로 암호화폐가 대중들에게 널리 알려졌다. 암호화폐는 중앙정부에 의해 통제되고 운영되는 화폐가 아닌 민간 분야의 개발자에 의해 창조된 완전히 새로운 화폐 시스템이라는 점에 그 의의가 있다. 암호화폐가 동작하는 환경에는 블록체인과 DAG (Directed Acyclic Graph)을 기반으로 한 것이 있는데, 대부분의 암호화폐가 블록체인 상에서 구현되었으므로 블록체인을 기반으로 한 암호화폐를 중심으로 소개하고자 한다. 현재 유통되는 암호화폐는 1,400여개에 달하지만, 본 논문에서는 블록체인 기술의 큰 흐름을 기준으로 성격과 목적이 크게 달라진 대표적인 암호화폐에 대해서 알아보고자 한다.

II. 블록체인 기술과 암호화폐

2.1 비트코인(Bitcoin)

비트코인은 2009년 나카모토 사토시에 의해 개발된 암호화폐로, 중앙기관 없이 참여자들의 네트워크를 통해 모든 거래가 이루어지고 검증된다. 비트코인은 블록체인 기술로 구현한 최초의 암호화폐이므로 1세대 암호화폐라 할 수 있다. 비트코인은 블록체인의 핵심가치인 분권화와 탈중앙화에 기반한 시스템을 시도했다는 점에서 의의가 있으나, 금융 분야에서의 한정된 사용, 느린 거래 속도 및 낮은 확장성, 분산화된 시스템으로 인한 의사결정 과정에서의 합의 도출의 어려움 등의 한계점이 있다.

2.2 이더리움(Ethereum)

이더리움은 2015년 비탈릭 부테린(Vitalik Buterin)에 의해 개발되었다. 이더리움은 스마트 컨트랙트(smart contract)를 블록체인을 기반으로 하

여 구현한 최초의 암호화폐 플랫폼으로 2세대 암호화폐라 할 수 있다. 초기 비트코인은 기존의 화폐와 은행 시스템을 대체하기 위한 수단에 불과했다. 하지만 이더리움은 블록체인 상의 서비스 및 어플리케이션을 제공하기 위한 플랫폼으로 진화했다. 이더리움의 스마트 컨트랙트를 중심으로 계약의 자동화가 이루어졌다. 온라인 상에서 거래 중개자 없이 법적 효력을 지닌 계약 집행이 가능하다는 점에서 스마트 컨트랙트는 화폐의 성격이 강한 비트코인을 뛰어넘은 온라인 거래 플랫폼으로의 블록체인의 발전가능성을 보여주었다.

2.3 이오스(EOS)와 에이다(ADA)

이더리움의 단점으로 알려진 거래 속도, 개발 환경, 상호호환성, 의사결정 알고리즘 등을 보완한 대표적인 암호화폐에는 이오스(EOS)와 에이다(ADA)가 있다. 이오스는 블록체인 개발자인 댄 라리머를 주도로하여 개발된 퍼블릭 블록체인 플랫폼이다. 기존 블록체인 플랫폼의 비싼 수수료와 느린 트랜잭션 처리속도를 개선하기 위한 목적으로 개발되었다. 이오스의 많은 부분이 웹 어셈블리로 개발되었기 때문에 스마트 계약을 웹브라우저 상에서 빠르게 실행할 수 있는 장점이 있다. 에이다는 분산형 퍼블릭 블록체인인 카르다노(Cardano)라는 블록체인 플랫폼 위에서 동작하는 암호화폐이다. 하스켈(Haskel) 언어를 사용하여 구현되었기 때문에 성능 및 안정성이 뛰어나다.

III. 암호화폐의 동향

1세대 암호화폐라 할 수 있는 비트코인의 등장 이후, 결제의 시간적 지연, 확장성의 한계, 제한적인 용도, 높은 채굴 비용 등의 불편함이 부각되자 이를 해소하기 위한 다수의 알트코인(altcoin)이 등장했다. 알트코인은 '대안'을 뜻하는 alternative와 '비트코인'의 coin을 합성한 표현이다. 앞에서 블록체인 기술과 암호화폐의 추세를 보면 알트코인은 이전 블록체인 기술이 가지고 있던 문제점, 한계성을 보완하고 확장하는 방향으로 등장하고 있음을 알 수 있다. 3세대 암호화폐로 주목받고 있는 이오스(EOS), 에이다(ADA) 등은 새로운 합의 알고리즘을 기반으로 보다 혁신적이고 발전된 플랫폼을 선보이고 있어 향후 블록체인의 활용 범위를 더욱 넓혀 줄 것으로 기대되고 있다. 블록체인과 암호화폐는 초기에는 금융 분야를 중심으로 발전해왔으나, 기술 진화에 따라 활용가능 분야도 점차 넓어지고 있다. 무엇보다 현재 전 세계적으로 1,400여개의 암호화폐가 유통되고 있다는 사실을 통해 블록체인 기반 암호화폐의 무궁무진한 발전 가능성을 엿볼 수 있다.

IV.결론

본 논문에서는 블록체인 네트워크 상에서의 암호화폐의 발전 동향을 살펴보았다. 최근 등장하고 있는 암호화폐들은 이전 암호화폐의 단점과 한계점을 보완하는 방식으로 발전하고 있으며, 계속해서 발전하는 혁신적인 플랫폼을 통해 다양한 분야에 활용될 수 있을 것으로 기대된다. 현재 블록체인 플랫폼의 한계점을 극복해나간다면 암호화폐는 단순한 디지털 자산으로서의 가치를 넘어, 사회 시스템의 변화를 도모할 수 있는 핵심 기술로 널리 사용될 것이다.

[참고문헌]

[1] Hee-Jong Lee, Jae-Young Cho, A Study on Matters of Crypto-Currency,The Korean Association Of Comparative Private Law-2018-25(2), 657-696, May, 2018.

[2] Jun-Sang Kim, A Survey of Cryptocurrencies based on Blockchain, The Korean Society Of Computer And Information-2019-24(2), 67-74, February, 2019.

[3] Jae-Young Lee, The Blockchain 3.0 Era and the Future of Cryptocurrency, FUTURE HORIZON-2018-37, 32-35, November, 2019.

2019 한국정보보호학회 동계학술대회

CISC-W'19

Conference on Information Security and Cryptography-Winter 2019

구두발표 트랙 P

블록체인을 활용한 관광 분석 시스템 제안

장경배* 최승주* 권혁동* 김경호* 서화정*†

*한성대학교 IT융합공학부

A Proposal of tourism analysis system using Blockchain

Kyoung-Bae Jang* Seung-Ju Choi* Hyeok-Dong Kwon* Kyung-Ho Kim* Hwa-Jeong Seo*†

*Division of IT convergence engineering, Hansung University.

요 약

관광 사업의 발전은 지방자치단체나 국가의 입장에서 매우 중요한 요소이다. 때문에 해외 관광객들이 우리나라의 무엇을 좋아하는지 주요 지출 분야가 어디인지 파악하는 것은 매우 중요하다. 이러한 측면에서 한국 여행 중, 관광객의 돈이 어디에 쓰이는지는 중요한 데이터가 된다. 하지만 관광객들의 다양한 소비 내역을 수집하는 작업은 매우 까다롭다. 본 논문에서는 돈의 흐름을 효과적으로 수집하기 위해 블록체인[1] 코인을 활용한다. 블록체인에 기록되는 관광객들의 소비 데이터를 수집하고 분석하는 관광 분석 시스템을 제안하며 다양한 기대효과를 예상해 본다.

I. 서론

한류의 긍정적인 영향 및 쇼핑, 관광 등을 위해 한국을 방문하는 관광객이 증가함에 따라 국내 관광 사업이 더욱 활성화 되고 있다. 그러나 관광객들이 선호하고 가장 많이 찾는 실질적인 데이터는 부족한 상황이다. 관광 사업을 활성화시키기 위해서 이는 반드시 필요한 데이터이며 파악하기 가장 좋은 방법은 관광객들의 국내 소비 패턴을 분석하는 것이다. 의류, 관광, 식업 등 다양한 소비 분야와 이러한 소비가 발생하는 지역 및 장소를 파악해서 해당 사업들을 더욱 활성화시킬 수 있다면 관광 사업은 더욱 발전할 것이다. 하지만 전국 곳곳에서 다양하게 발생하는 소비 데이터들을 일괄적으로 수집하여 활용하기에는 큰 어려움이 따른다.

이에 블록체인을 활용한 관광 분석 시스템을 제안한다. 관광객과 상업자 사이의 거래를 블록체인 네트워크에서 진행하고 결제 내역을 트랜잭션 한다. 다양한 결제 내역이 블록체인에 일괄적으로 기록되기 때문에 기존 문제점인 데이터 확보의 어려움을 해결한다. 이를 통해 다양한 관광객의 소비 데이터를 활용 및 분석할 수 있고, 대한민국 관광 사업 발전에 기여할 수 있을 것이다.

II. 관련 연구

2.1 빅데이터를 활용한 관광 사업과 한계점[2]

대부분의 지방자치단체에서는 2016년부터 지역 발전을 위한 빅데이터 관련 사업을 진행 중이다. 그 중 관광 분야는 주로 관광객 수 집계나 관광객의 신용카드 소비 패턴으로 분석하는 추세이다. 하지만 현업 부서의 협조 부족, 데이터 확보의 어려움, 예산 부족이 주요 애로사항으로 꼽힌다. 이러한 상황에서 빅 데이터의 부적절한 활용이나 부정확한 통계 생산이 이루어지지 않도록 정부의 적절한 지원이 필요하다.

III. 블록체인을 활용한 관광 분석 시스템

3.1 관광 코인

본 논문에서 제안하는 블록체인을 활용한 관광 분석 시스템에서는 국내에서 원화 대신 사용할 수 있는 관광코인을 사용한다. 블록체인 기술은 암호 화폐의 기반이 되는 기술이기에 간편 결제를 구현하기에 매우 적합하다. 한국을 방문하는 외국인 여행자들은 자신들의 외화를 원화로 환전하는 대신 관광코인을 충전하여 사용한다. 관광코인 결제 시스템을 사용하는 가맹점이라면 어디든지 결제가 가능하다. 관광객의 입장에서는 네트워크 거래가 가능한 스마트폰만 있으면 앱을 통해서 간편한 결제가 가능해지는 것이다. 또한 결제 사용 내역들은 블록체인 상에 기록되기 때문에 자신들의 여행 소비 내역 또한 손쉽게 관리할 수 있다.

관광객들이 익숙하지 않은 나라에서 평소에 자신들이 사용하던 원화가 아닌 외화를 사용하고 관리 하는 것은 매우 까다롭다. 하지만 관광코인을 사용한다면 현금을 사용하지 않고 언제든지 충전할 수 있기 때문에 분실 위험이 없고 여비 관리 또한 편리해진다. 여행 전 또는 여행 도중 환전과 현금 관리 등으로부터 오는 번거로움에서 벗어날 수 있는 것이다.

3.2 전자 영수증

블록체인 기반의 코인을 사용하기 때문에 모든 결제 내역이 트랜잭션 되어 기록된다. 해당 정보는 블록체인 특성상 위변조에 대해 강인한 내성을 갖기 때문에 충분히 신뢰할 수 잇는 데이터이다. 이러한 결제 내역 정보는 전자 영수증으로 활용될 수 있으며 추후에 해당 결제에 대한 증명이 필요한 경우 사용될 수 있다.

3.3 관광코인 데이터 수집 및 분석

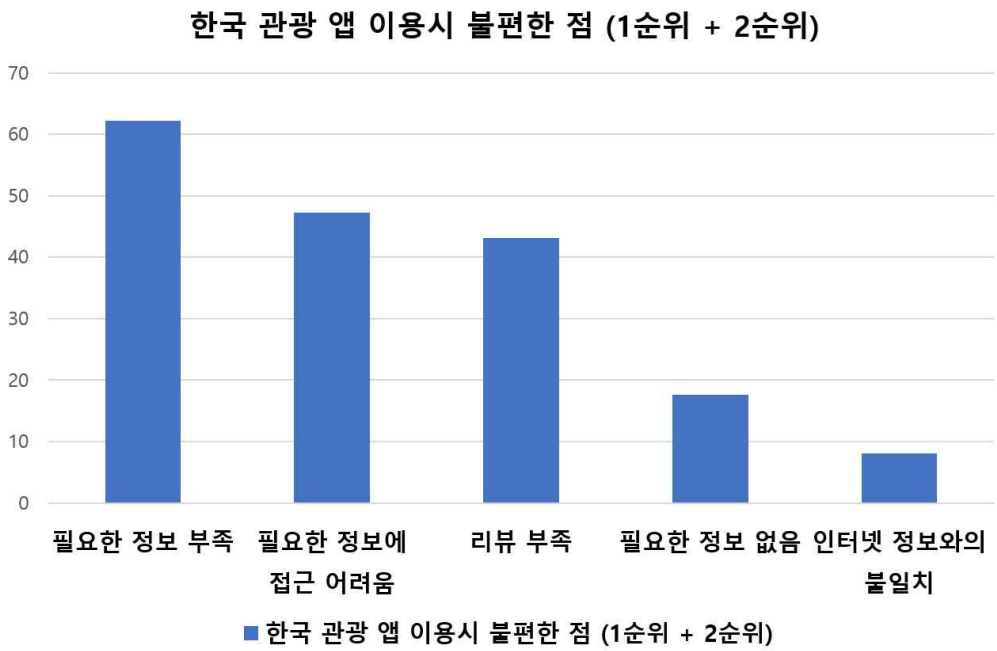
관광객이 결제한 데이터는 단순 결제정보에 국한되지 않고 관광 사업의 발전을 위한 데이터가 된다. 그러나 앞서 관련연구에서 보았듯 데이터 확보에는 어려움이 있다. 해외 관광객의 결제 방법은 해외 가맹 카드를 사용하거나 현금으로 결제하는 것이 일반적이다. 하지만 현금을 사용한 결제는 추적하기가 사실상 불가능하다. 그리고 가맹 카드를 사용하여 결제한 데이터는 해당 카드사를 제외하고는 공유되지 않는다. 이를 수많은 카드사에 일일이 요구할 수도 없는 상황이다. 하지만 관광코인으로 블록체인 상에서 거래가 이루어진다면 네트워크에 기록되는 결제 내역을 일괄적으로 수집할 수 있다.

수집한 결제내역을 기반으로 데이터 분석 기술을 활용한다면 더 큰 부가적 이익을 창출하는데 사용될 수 있을 것이다. 예를 들어 해당 시스템을 문화관광부 등의 정부기관에서 운영하게 된다면 어느 나라에서 온 어떤 연령대의 사람이 무슨 소비활동을 하였는지 분류가 가능해진다. 상대적으로 결제 내역이 부족한 상품은 손쉽게 비인기 상품으로 분류가 가능하며, 해당 상품에 대한 개선을 계획하거나 다른 상품으로 대체할 수 있다. 마지막으로 모든 지자체가 정보를 공유한다는 점에서 다른 지자체의 관광 상품 수요를 분석할 수 있다.

즉, 관광 자원의 불균형 또는 집중화를 해소할 수 있다. 외국인 관광객들의 한국 여행 성향을 파악하고 변화추이 비교분석 또한 가능하다. 각 지역별 및 분야별로 관광 사업을 발전시킬 지역과 분야를 블록체인에 트랜잭션 된 데이터를 바탕으로 파악할 수 있다.

3.4 관광지역 추천 및 정보 공유

한국에 온 관광객들이 기존 한국 관광 앱에서 느끼는 불편한 점[3]은 그림 1과 같다. 관광 가이드를 위한 양질의 정보가 부족한 것이다. 이를 해결하기 위해, 제안하는 시스템에선 관광코인의 흐름을 분석하여 한국에 방문하는 관광객에게 관광지 추천이나 관광 가이드를 제공한다. 해당 추천정보는 실제 관광객들의 소비를 반영하기 때문에 기존에 리뷰 작성 등 자신의 주관이 반영되는 정보에 비해 훨씬 객관적인 지표가 될 수 있다. 더하여 관광객이 자신의 결제 내역을 공개하기로 동의한다면 다른 관광객들에게 공유하여 사전 조사가 부족한 관광객이 입을 수 있는 피해를 미연에 방지할 수 있다. 그리고 정보 공유를 한 관광객에게 혜택을 지원하는 것으로 좀 더 많은 공유가 이루어지고 이로 인해 관광객의 더욱 편리한 정보 확보가

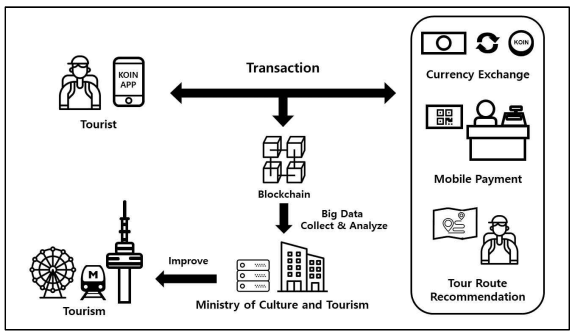


<그림 1> 한국 관광 앱 이용 시 불편한 점

이루어진다면, 관광객들의 한국 여행에 대한 만족도가 높아질 것이며 더 많은 소비를 촉진시킬 수 있다.

3.5 시스템 적용 구조

제안하는 블록체인을 활용한 관광 분석 시스템의 동작 구조는 그림 2와 같다.



<그림 2> 시스템 구조

언제 어디서든 스마트폰만 있다면 해당 서비스에서 제공하는 앱을 통해 코인을 자국의 돈으로 교환할 수 있다. 이렇게 교환한 코인은 모든 가맹점에서 사용할 수 있으며 이는 일반적으로 카드로 계산할 수 없는 길거리 상점 또한 포함이 된다. 예를 들어, 길거리 음식 판매자들은 앱에 가입하고 본인의 블록체인 상의 주소값을 QR 코드로 출력을 하여 제시해 놓는다. 해당 음식을 구매하고 싶은 관광객은 로그인 후 QR 코드를 스캔하여 판매자의 주소로 음식값을 코인으로 지불하게 된다. 판매자는 지불된 코인을 확인하고 음식을 판매 완료하게 된다. 이러한 절차를 통해 분실 및 도난 가능성이 있는 현금을 따로 관리할 필요가 없어지게 된다. 이와 같은 모든 결제 내역들은 블록체인에 기록이 되기 때문에 전자 영수증으로도 활용이 된다.

앞서 언급한 관광객들의 결제 내역들은 전부 블록체인에 기록이 되는데, 이를 바탕으로 음식, 오락, 축제 등 각각의 분야에서 가장 인기 있는 분야들을 판단하여 관광 사업 중 앞으로 발전시킬 분야를 결정하는데 지표로서 활용이 될 것으로 예상된다. 또한 해당 데이터들을 빅데이터 기술을 사용하여 분류 및 분석하여 관광객의 연령, 성별, 국적 등에 따라 지역별로 소비 패턴을 분석 가능. 분석한 데이터를 기반으로 관광지의 강한 특산품을 확인하여 이를 좀 더 발전시켜 관광 사업을 키울 수 있으며 부족한 상품 등은 보완하는 형식을 통해 관광지를 개선하는데 활용된다. 또한 외국인의 방한 시 이전 트랜잭션을 기반으로 새로 방문한 외

국인에게 관광 루트나 상품 등을 추천하는데 사용하여 외국인의 관광 가이드가 된다.

IV. 비교 분석

본 장에서는 기존의 관광 시스템과 블록체인을 활용한 관광 분석 시스템을 표 1과 같이 평가하여 비교 분석한다.

	기존 시스템	제안 시스템
외화 관리	복잡함	단순함
소비 내역 관리	어려움	쉬움
데이터 획득 및 분석	까다로움	용이함

<표 1> 제안 시스템 비교 분석

먼저 관광객의 입장에서 비교해본 결과, 기존 시스템에서 관광객들은 외화를 관리하는데 있어 어려움이 따른다. 하지만 제안하는 시스템에선 가상의 블록체인 코인을 사용한다. 물리적으로 존재하지 않기 때문에 관리하는데 부담이 없다. 또한 코인을 이용하여 결제 시, 블록체인에 전부 기록되기 때문에 전자영수증의 역할을 함과 동시에 소비 내역관리가 용이해진다. 그리고 예를 들어, 문화관광부와 입장에서 이러한 소비 내역은 관광 사업 발전을 위한 매우 중요한 데이터이다. 기존 시스템에서는 다양한 카드사, 현금 추적의 어려움 등의 이유로 획득이 어려웠지만 제안하는 시스템에서는 모두 블록체인에 기록되기 때문에 획득 및 분석이 용이하다.

본 논문의 제안 시스템의 가장 큰 차이점은 관광 시 블록체인 코인을 활용한 거래이다. 관광객은 가상으로 존재하는 코인을 사용하기 때문에 결제 및 관리가 편리하다. 그리고 코인을 활용한 거래는 모두 블록체인에 기록된다. 이로써 관광 사업 발전에 필요한 데이터를 기존 시스템보다 손쉽게 획득할 수 있다.

V. 결론

관광 사업 발전을 위해 관광객이 한국에서 선호하는 것이 무엇인지 파악하는 것은 매우 중요하다. 이를 파악하는 제일 좋은 방법은 관광객의 소비패턴을 수집하여 분석하는 것이다. 만약 제안하는 시스템의 블록체인 코인을 관광객들이 사용한다면, 블록체인의 장점을 이용하여 모든 관광객들의 소비내역을 획득할 수 있다. 즉, 기존의 데이터 획득이 어렵다는 문제점을 해결할 수 있다. 관광객의 입장에서도 코인이 제공하는 관리 및 결제의 편리함 때문에 사용의 이유가 충분하며, 한국 관광에 대한 만족도 또한 높아질 것이다.

본 논문에서 제안하는 시스템을 문화관광부와 같은 정부 측에서 사용한다면, 해외 관광객들의 한국 내 소비패턴 분석이 용이해질 것이다. 향후 연구 방향으로는 공개 데이터를 사용하여 실제로 분석 시스템을 구현하고, 문제점이 될 수 있는 블록체인의 트랜잭션 속도를 확인해야 한다. 따라서 본 논문의 주제에 대한 지속적인 연구가 필요하며, 한국 관광 사업 발전에 크게 기여할 수 있을 것이라 예상된다.

[참고문헌]

[1] Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electornic Cash System, 2008

[2] The Korean Association for Survey Research, Statistical Data Methodology research service report Using Big Data, 2016.

[3] N. K. Kim, 6billion Youke times…“Tourist information is inconvenient”, IT.chosun, 2015, http://m.news.zum.com/articles/20453021

블록체인을 이용한 호스트기반 악성코드 침입탐지 관제시스템에 관한 연구

최용철, 이덕규**

**서원대학교 정보보안학과

cho6nt@gmail.com, deokgyulee@gmail.com

Title

Yong cheol Choi, Deok Gyu Lee**

**Dept of information security, Seowon University.

요 약

최근 정보 보안 기술의 발전과 더불어 악성코드 또한 발전함에 따라 표적형 지능형 공격인 Advanced Persistent Threats (APT) 공격은 사회적으로 큰 문제를 야기하고 있다. APT 공격은 표적 시스템에 사회공학적 기법을 이용하여 침투하여 표적 시스템의 정교한 보안망을 무너뜨림으로써 장기간에 걸쳐 정보를 유출하거나 시스템을 파괴하여 표적에 대한 금전적인 피해를 입힌다. 또한 성공적으로 정보 유출을 하였을 경우 한번에 만족하지 않고 공격자의 뜻대로 언제든지 정보를 유출할 수 있도록 악성코드를 숨겨 드러내지 않는 지능적인 행동을 한다. 본 논문에서는 블록체인을 이용하여 호스트기반 악성코드 탐지 시스템을 활용해 기존의 침입탐지 시스템의 단점을 보완하는 방법을 제시한다.

I. 서론

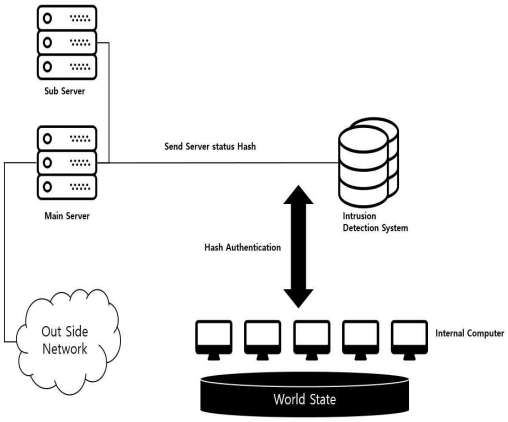
최근 정보 보안 기술의 발전과 더불어 악성코드 또한 발전함에 따라 새로운 사이버 공격이 나타나고 있다. 특히 표적형 지능형 공격인 Advanced Persistent Threats (APT) 공격은 사회적으로 큰 문제를 야기하고 있다. APT 공격은 표적 시스템에 사회공학적 기법을 이용하여 침투하여 표적 시스템의 정교한 보안망을 무너뜨림으로써 장기간에 걸쳐 정보를 유출하거나 시스템을 파괴하여 표적에 대한 금전적인 피해를 입힌다. 이처럼 APT 공격은 초기 침투에는 스피어 피싱과 같은 비교적 단순한 공격을 취하지만 초기 침입 이후에 장기간 정보를 유출하도록 백도어를 생성하고 내부 네트워크를 분석하여 악성코드를 전염시킨다. 또한 성공적으로 정보 유출을 하였을 경우 한번에 만족하지 않고 공격자의 뜻대로 언제든지 정보를 유출할 수 있도록 악성코드를 숨겨 드러내지 않는 지능적인 행동을 한다.[1] 본 논문에서는 블록체인을 이용하여 호스트기반 악성코드 탐지 시스템을 활용해 기존의 침입탐지 시스템의 단점을 보완하는 방법을 제시한다.[1)]

II.제안 방식

3.1 전체 시스템 구성

3.1.1 시스템 구성도

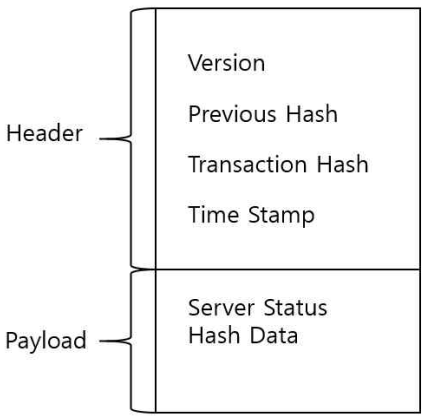
*본 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2019-0-00326, 블록체인 기반 물류정보의 실시간 트래킹을 통한 스마트 항만 응용 플랫폼 개발)



[그림 3] System flowchart

본 논문에서는 하이퍼레저 패브릭을 이용하여 구성한다. 제안하는 시스템 형식은 [그림 3]과 같다. 사내에서 서비스하고 있는 메인서버와 침해사고 발생 시 빠른 복구를 위한 미러링된 백업 서버가 존재한다. 침해대응 시스템과 메인서버, 그리고 사내에 있는 각각의 PC들이 블록체인 네트워크를 이루는 형식이다. 메인 서버가 트랜잭션을 발생시켜 스마트 컨트랙트를 작동하여 자신의 시스템의 현재 해쉬값을 블록에 등록하고, World State 데이터베이스에 해당 서버의 해쉬값이 저장되어있다. 악성코드에 의한 침해사고가 발생 했을 경우에는 Sub서버로 서비스를 포워딩 요청을 한 후, 해당 Main서버의 감염된 악성코드의 해쉬값과, 감염된 서버의 해쉬값도 함께 블록체인에 등록한 뒤 해당 규칙을 침입탐지 시스템에 등록한다.

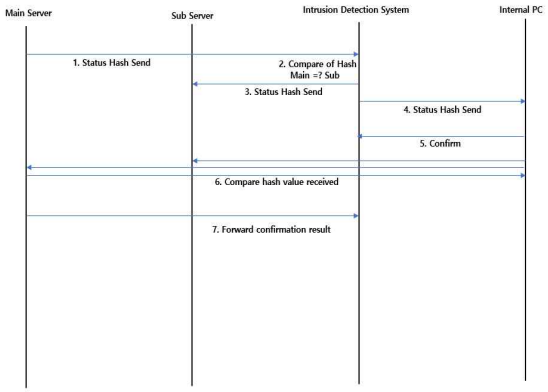
3.1.2 블록체인 구조



[그림 4] Block Format

제안하는 시스템에서 블록체인의 구조는 다음 [그림 4] 와 같다. 현재 블록체인의 버전이 명시되어있고, 이전 블록의 해쉬값과 트랜잭션의 해쉬값, 타임스탬프, 그리고 현재 서버 상태의 해쉬값과 특정 악성코드의 해쉬값이 저장되어있다.

3.1.3 블록생성 및 등록

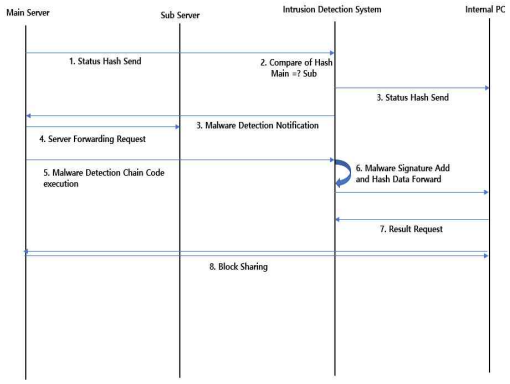


[그림 5] Process of Block Creation

본 논문의 시스템에서 블록을 생성하고 등록하는 과정은 다음과 같다. Main서버가 해쉬 값을 전송한다. 침입탐지 시스템에서 해쉬값을 전송받은 후, 미러링 되어있는 Sub서버와 Main 서버의 해쉬값을 비교한다. 비교하여 값이 같은 경우 내부 노드와 함께 해쉬값을 검증한다. 검증이 완료되면 결과를 침입탐지 시스템에게 전송하고, 모든 노드가 자신이 갖고있는 Main서버의 해쉬값과 비교한다. 비교한 결과 문제가 없는 경우 침입탐지 시스템에게 해당 결과를 전송하여 블록을 생성한다.

3.1.3 침입탐지

해당 시스템이 침입탐지를 하는 과정은 다음 그림과 같다. Main 서버가 해쉬값을 전송한다. 침입 탐지 시스템은 Sub서버와 Main서버의 해쉬값을 비교하는 과정에서 해쉬값이 다를 경우 Main서버에게 악성코드가 탐지되었다는 알림을 보낸다. Main서버는 악성코드탐지 알림을 받으면 Sub서버에게 포워딩 요청을 보낸 후 침입탐



[그림 6] Process of Detection

지 시스템에게 악성코드탐지에 관한 체인코드 실행을 요청한다. 체인코드실행 요청을 받은 침입탐지 시스템은 해당 Main서버에 감염된 악성코드의 해쉬값과 감염당시 서버의 해쉬값을 함께 블록에 등록하며 해당 블록을 다른 노드와 공유한다. 블록을 전달받은 노드들은 모든 노드간의 서로에게 전달받은 블록을 비교하며 블록을 추가한다. 해당 과정이 종료되면 침해사고를 일으킨 악성코드가 일으켰던 해당 행위에 대한 해쉬값을 모든 노드가 공유하고 있으므로 해당 노드에 참여하고있는 시스템에서는 해당 악성코드가 정의되어있기 때문에 추후 같은 방식의 공격 방어에 용이하다.

3.1.4 해당 시스템의 성능분석

본 논문에서 제안하는 시스템은 기존의 호스트기반 침입탐지 시스템의 단점이었던 모든 호스트에 설치하는 과정을 줄이고 비용을 절감시킬 수 있다. 또한 전체적으로 스캔 할 필요 없이 자신이 공유하고 받았던 블록을 이용하여 시그니처 및 휴리스틱 기반으로 악성코드를 탐지 할 수 있으므로, 탐지 성능도 우수하다고 볼 수 있다. 하이퍼레저 기반을 사용하고 있는 시스템이므로 해당 시스템을 사용하고자 하는 유저를 추가하기만 하면 자동으로 블록이 동기화 되므로 사용자 시스템의 구성 및 삭제가 용이하다. 또한 공격받은 악성코드의 해쉬값 또한 분산저장되므로 해당 해쉬값이 잘못 전송되거나 깨질 수 있는 경우를 블록체인을 이용하여 방지 할 수 있다. 하지만 블록을 공유하고 해쉬값을 전송하는 과정에 있어 노드수에 따른 전체적인 네트워크 트래픽이 저하될 수 있다. 따라서 대규모의 네트워크를 운영하고 있는 시스템에는 부적합 하다고 볼 수 있다. 또한 동시다발적인 공격 및, Sub서버까지 공격의 영향이 미치면 해당 시스템의 사용이 불가능하다는 단점이 존재한다.

III.결론

최근 표적형 지능형 공격인 Advanced Persistent Threats (APT) 공격은 사회적으로 큰 문제를 야기하고 있다. APT 공격은 표적 시스템에 사회공학적 기법을 이용하여 침투하여 표적 시스템의 정교한 보안망을 무너뜨림으로써 장기간에 걸쳐 정보를 유출하거나 시스템을 파괴하여 표적에 대한 금전적인 피해를 입힌다. 또한 성공적으로 정보 유출을 하였을 경우 한 번에 만족하지 않고 공격자의 뜻대로 언제든지 정보를 유출할 수 있도록 악성코드를 숨겨 드러내지 않는 지능적인 행동을 한다. 본 논문에서는 블록체인을 이용하여 호스트기반 악성코드 탐지 시스템을 활용해 기존의 침입탐지 시스템의 단점을 보완하는 방법을 제시하여 침입탐지에 관한 문제를 해결하는데 기여하고 싶다.

[참고문헌]

[1] 임형진 '의사 결정 트리를 이용한 악성 행위 분석 기반 침입 차단 시스템' 서울과학기술대학교 [2015]

[2] 정승일*, 김현우** '악성코드 탐지 시스템 Web-Anti-Malware' 카이스트 사이버 보안 연구센터 [2014]

[3] 김선우 '침입탐지시스템에서의 실시간 악성코드 탐지' 한양대학교 [2014]

[4] 장정우 '스카다 시스템에서 악성코드 탐지 및 보안관제 방안에 관한 연구' 고려대학교 [2014]

[5] 봉진숙 '하이퍼레저 패브릭 블록체인 기반의 개인건강정보 공유플랫폼' 숭실대학교 대학

원 [2019]

블록체인 기반 릴라제로 분산 학습 기법*

박승윤, 오희국†

*한양대학교 컴퓨터공학과

A blockchain-based Distributed Training method for Leela-zero*

Seungyun Park and Heekuck Oh†

*Department of Computer Science and Engineering, Hanyang University.

요 약

릴라제로의 학습과정에 참여하는 클라이언트는 하루 평균 200-250개이고 현재까지 대략 1500만개의 자가대국 수 그리고 90만개의 매칭게임을 가진다.[2] 계산을 해보면 하루 평균 15,000개의 자가대국 수를 실행하는 것이다. 이는 분산 컴퓨팅을 활용한 릴라 제로의 진행방식에 있어서는 비효율적 결과를 야기한다. 또한, 최근 2019년 세계 인공지능 바둑대회에서 릴라제로는 국내 바둑 인공지능인 '한돌'과의 대국에서 져 4등을 차지하였다. 이는 릴라제로의 기력에 대한 의구심을 들게 하였다.

블록체인은 위에서 언급한 릴라제로의 기력을 증강하는데 큰 솔루션이 될 수 있다. 현재 릴라제로 학습과정에 기여하는 노드들의 수는 분산 환경에서 진행된다는 점에 비하면 턱없이 부족한 수이다. 이는 블록체인을 기반으로 개발된 탈중앙화 어플리케이션에서 컴퓨팅 파워를 대여해주는 노드들에게 보상을 함으로서 참여를 유도할 수 있다.

I. 서론

릴라제로는 구글 딥마인드의 논문[1]을 바탕으로 알파고 제로를 재구현하기 위한 프로젝트이다. 이름 그대로 바둑의 규칙 이외에는 인간의 지식을 입력하지 않고 강화학습을 진행한다. 릴라제로는 1600 visits의 자가대국을 실행하고 어느 정도 학습이 진행된 가중치 파일로 시험대국 400판을 실행하여 기존의 가중치 파일을 상대로 55% 이상의 승률을 가진다면 새로운 가중치 파일 즉 ,Best-Network 가 된다. 이러한 점은 구글 딥마인드의 논문을 바탕으로 개발된 알파고 제로의 진행과정을 따른다. 릴라제로는 분산 컴퓨팅을 사용하여 학습을 진행한다. 릴라제로 개발자는 알파고 제로의 학습과정을 단일 컴퓨터로 학습과정을 재구현하기 위해서는 1700년이라는 긴 시간이 필요하다고 언급하였지만 릴라제로는 분산 컴퓨팅을 활용하여 2017년 11월에 랜덤 착수를 시작으로 2019년 10월 현재, 알파고 제로와 동일한 40블록으로 크기가 커졌으며, 이는 알파고 제로만큼의 기력을 가지게끔 하였다.

하지만 릴라제로의 학습과정에 참여하는 클라이언트는 분산 컴퓨팅을 활용한 릴라 제로의 진행방식에 있어서 상대적으로 비효율 결과를 가져온다. 또한, 최근 2019년 세계 인공지능 바둑대회에서 릴라제로는 국내 바둑 인공지능인 '한돌'과의 대국에서 져 4등을 차지하였다. 이는 릴라제로의 기력에 대한 의구심을 들게 하였다.

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2019R1A2C2003045).

† 교신저자 : hkoh@hanyang.ac.kr

블록체인은 위에서 언급한 릴라제로의 기력을 증강하는데 큰 솔루션이 될 수 있다. 첫 번째로, 현재 릴라제로 학습과정에 기여하는 노드들의 수는 분산환경에서 진행된다는 점에 비하면 턱없이 부족한 수이다. 이는 블록체인을 기반으로 개발된 탈중앙화 어플리케이션에서 컴퓨팅 파워를 대여해주는 노드들에게 보상을 함으로서 참여를 유도할 수 있다. 또한, 실제로 유휴한 컴퓨팅 자원을 가진 클라이언트들에게 참여를 유도하게 됨으로써 학습과정에 참여하는 노드들의 수를 대폭 확보할 수 있다. 본 논문에서는 릴라제로의 더딘 학습과정 문제점을 해결하고자 이더리움 블록체인을 활용하여 릴라제로의 학습과정에 기여하는 탈중앙화 어플리케이션 개발에 목표를 둔다.

이후 논문에서는 2장에서 개발과정에 필요한 배경지식과 3장에서 릴라제로에 대한 설명,4장에서 릴라제로와 블록체인 연동모델에 대한 설명 마지막으로 5장에서 결론을 맺는다.

II. 배경지식

블록체인은 확장하는 분산형 네트워크로, 네트워크에 참여하는 모든 노드가 거래내역 데이터를 분산 및 저장하는 기술이다. 기존의 중앙집중형 네트워크는 모든 데이터를 관리하는 신뢰할 수 있는 중앙 관리자가 존재하고 중앙관리자에 문제가 생긴다면 모든 시스템에 문제가 발생할 수 있다. 하지만 분산형 네트워크 즉, 블록체인은 데이터가 모든 네트워크에 분산되어 유지되기 때문에 중앙 관리자가 존재하지 않으며, 특정 노드에 문제가 발생이 되어도 다른 네트워크 노드에게도 동일한 데이터를 가지고 있기 때문에 문제가 발생하지 않는다.

2.1 스마트 컨트랙트 및 라이브러리

스마트 컨트랙트는 EVM 바이트코드라 불리는 튜링 완전한 바이트 코드 언어로 작성된다. 쉽게 말하면, 스마트 컨트랙트는 일련의 바이트코드 명령어들로 정의된 함수들의 집합이다. 이러한 스마트 컨트랙트를 이용하여 보다 복잡한 계약을 즉 다양한 패턴을 만들 수 있다. 이로 인해서, 이더리움은 스마트 컨트랙트를 활용하여 다양한 목적을 가진 탈중앙화 어플리케이션을 개발할 수 있다. 정리하자면 유저들은 트랜잭션을 다음과 같은 목적으로 전송한다.

1. 스마트 컨트랙트 생성
2. 스마트 컨트랙트 함수 호출
3. 스마트 컨트랙트 또는 다른 유저에게 이더 전송

스마트 컨트랙트는 일반적으로 솔리디티 언어로 작성된다. 이더리움에서는 EVM에서 실행

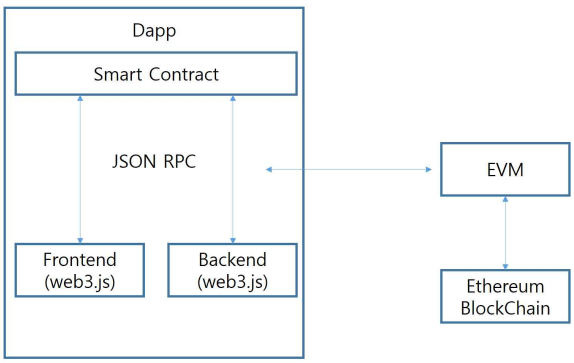


그림 1 Dapp 구성

되는 로우레벨 바이트코드로 컴파일해야 한다. 스마트 컨트랙트를 컴파일 하면 결과로써 ABI와 바이트코드를 얻을 수 있다.바이트코드는 EVM에서 실행될 코드의 역할을 하며, ABI는 머신코드에서 데이터 구조 및 기능에 대한 접근방법을 정의한다. 이더리움에서의 ABI는 EVM에 대한 컨트랙트 요청을 인코딩하고 트랜잭션에서 데이터를 읽어올 때 사용된다. 즉, ABI의 목적은 호출할 수 있는 스마트 컨트랙트에서 함수를 정의하고 각 함수가 인수를 승인하고 결과를 리턴하는 방법을 설명한다.

2.2 라이브러리

이더리움의 대표적인 라이브러리로 Web3, Ethers, light 등이 있다. 본 논문에서 사용하는 라이브러리는 Web3로 HTTP 또는 IPC 연결을 사용하여 로컬 또는 원격 이더리움 노드와 상호 작용할 수 있는 자바스크립트 API이다. 또한 Web3.js 는 ABI로 EVM과 통신한다. [그림

1] 은 Dapp에서 프론트엔드와 백엔드에서 web3를 사용함을 보여준다. 유저가 프론트엔드단에서 블록체인 정보를 불러올 때 web3 라이브러리로 객체를 만들어 사용하여 블록체인 데이터를 불러온다. 블록체인 데이터로는 블록의 데이터 뿐만 아니라 web3.eth.getAccounts 함수를 통해 스마트 컨트랙트를 작성하고 배포할 때 필요한 유저의 공개키 또한 가져올 수 있다.

2.3 IPFS 프로토콜

IPFS[3]는 InterPlanetic File System의 약자로서 모든 컴퓨팅 장치를 동일한 파일 시스템과 연결하는 피어 투 피어 분산 파일 시스템이다. 기존 HTTP Web 은 클라이언트가 서버에 요청을 보내면 서버에서 응답해주는 구조를 가지고 있다. 따라서 중앙 서버가 차단되면 접근할 수 없다는 단점이 존재한다. 즉, 중앙화로 인한 문제점이 존재한다. 하지만 IPFS는 중앙화된 서버없이 IPFS 노드들간의 피어 투 피어 통신을 제공함으로서 영구적인 웹을 만들 수 있다. 또한, 높은 처리량의 Content-addressed 블록 스토리지 모델과 하이퍼링크를 제공한다.

본 논문에서는 릴라제로의 학습데이터를 포함한 업로드 데이터를 블록체인에 저장하여 공유하는 시스템을 제안하지만 릴라제로의 학습데이터의 경우 100MB에 가까운 크기를 가지기에 현실적으로 불가능에 가깝다.이런 문제점을 해결하기 위해서 IPFS 프로토콜을 사용한다. IPFS 프로토콜은 데이터의 해시값을 기반으로 데이터에 접근하기 때문에 실질적으로 블록체인에 저장되는 값은 데이터의 해시값이므로 블록의 크기안에 저장될 수 있다.

III.릴라제로

릴라제로[2]는 MCTS 알고리즘과 Deep residual convolutional neural network stack을 사용하고 사람이 제공한 지식없이 개발된 오픈소스 바둑 프로그램이다. 릴라제로는 크게 Leelaz, AutoGTP, Leela-server로 구성되어 있다. Leelaz는 그림에서

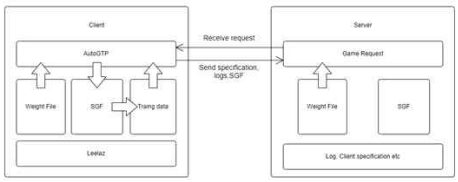


그림 2 릴라제로 동작 과정

보다시피 릴라제로를 돌리는 엔진으로써 여러 GUI툴을 활용하여 바둑게임을 실행할 수 있다. AutoGTP는 릴라제로의 Self-play 도구이다. AutoGTP가 실행되면 릴라제로는 서버로부터 best-network를 받아오며 게임을 실행한다 그리고 게임이 끝날때에 학습데이터와 SGF 파일을 서버에 업로드하는 역할을 한다. Leela-server는 릴라제로의 가중치 파일과 Self-play 그리고 Match 데이터에 대한 관리를 한다. 또한 클라이언트에게 Self-play Match에 대한 요청을 할 수 있으며 결과에 대한 정보를 보여준다. 그림에서 보다시피 릴라제로의 작동과정은 다음과 같다. 릴라제로 서버는 데이터베이스에 기력을 표현하는 Weight File과 게임 결과 데이터인 SGF를 저장한다. Weight File을 기반으로 게임 요청을 클라이언트에게 보낼 수 있다. 클라이언트는 서버로부터 받은 게임 요청과 Weight File을 AutoGTP를 통해 실행할 수 있다. 이 때에 클라이언트가 서버로부터 받은 Weight File의 해시값과 게임 요청에 담겨있는 Network 의 해시값을 검증하여 동일할 경우 게임을 실행한다. 게임이 종료되었을 경우, AutoGTP는 게임 결과 데이터인 .SGF 파일을 저장하며 서버로 응답할 수 있다. 또한, .SGF 파일을 기반으로 훈련데이터를 생성할 수 있다. 생성된 데이터를 가지고 Tensorflow를 활용하여 학습할 수 있다. 게임 결과를 받은 서버는 Self-Play 와 Match game 의 데이터 결과를 가지고 항상 최고의 기력을 가지는 Best-network를 유지한다. 릴라제로는 알파고제로의 논문을 바탕으로 개발된 AI 바둑 프로그램이므로 학습으로 생성된 Weight File이 기

존의 Best-network와의 승부에서 55%이상의 승률을 낸다면 그 Weight File을 Best-network로 업로드하는 방식을 그대로 따른다.

IV. 연동모델

4.1 업로드 데이터

본 논문에서 제안하는 모델에서, 블록체인에 업로드 하는 데이터는 릴라제로의 게임에서 필요한 Weight File과 게임의 결과값인 .SGF 파일이다. 또한, 게임 결과에서 파생되어 생성된 로그 데이터 또한 포함된다. 경우에 따라서 .SGF파일로부터 생성된 학습데이터 또한 업로드 데이터가 될수 있다.

4.2 업로드 데이터 이슈

제안하는 모델에서 업로드 데이터의 종류는 Weight File, .SGF, Log data, 학습데이터이다. Weight File 같은 경우 텍스트 형식으로 이루어진 파일이고 릴라 제로에서 텍스트 파일을 압축한 .gz 형식으로 Weight File을 운용한다. 100MB에 가까운 크기를 가지기에 현실적으로 블록체인에서 Weight File을 송수신한다는 것은 불가능에 가깝다. 본 논문에서는 블록체인에 릴라제로에서 사용되는 데이터를 직접 저장하는 방법을 사용하지 않고 IPFS를 활용하여 데이터의 해시값을 저장하는 방식을 사용하여 블록체인의 한계점을 해결하고자 한다.

4.3 연동모델

릴라제로 서버는 IPFS를 통해서 Weight file을 전송하며 클라이언트 측은 weight file을 수신하고 로컬에서 AutoGTP로 게임을 실행한다. 게임을 실행한 후에 결과값으로 생성된 SGF 파일을 다시 웹 기반 Dapp을 통해 전송하고 릴라제로 서버는 결과값 데이터를 수신하여 서버에 저장한다. 이에따라 컴퓨팅 파워를 제공한 클라이언트에게 보상을 하는 메커니즘을 가진다.

V.결론

본 논문에서는 기존 릴라제로의 학습과정에 블록체인을 연동하는 모델을 제안하였다. 또한 릴라제로의 학습데이터의 크기를 고려 IPFS 프로토콜을 사용하여 한계점을 극복하였으며 추후 릴라제로의 학습과정을 분산환경에서 효율적으로 진행하기 위한 Dapp을 개발하였다. 이는 릴라제로 모델에 블록체인을 활용하는 첫 모델 역할을 할 것으로 기대한다.

[참고문헌]

[1] Silver, David, et al. "Mastering the game of go without human knowledge." Nature 550.7676 (2017): 354.

[2] http://zero.sjeng.org/

[3] http://ipfs.io/

블록체인과 연합학습에 관한 연구

산디 라마디카, 이경현*

부경대학교 정보보호학협동과정

*부경대학교 IT융합응응공학과

sandika@pukyong.ac.kr, khrhee@pknu.ac.kr*

A Brief Discussion of Federated Learning and Blockchain

Sandi Rahmadika and Kyung-Hyune Rhee*

Interdiscplinary Program of Information Security, Graduate School PKNU

*Department of IT Convergence and Application Engineering

Pukyong National University, Republic of Korea

Abstract

Federated learning appears to the public to overcome the client's privacy in centralized learning by bringing the models to the data source or client device for training and inferencing. Federated learning utilizes computing and storage resources on the client's device, and it is privacy preserved by design. In line with this, blockchain is also popular since it does not require an intermediary in the transaction. However, these technologies are still in the early stages and face numerous challenges with its implementation. Therefore, in this paper, we present several points to be discussed related to federated learning and blockchain and the potential combination. In addition, we apply this technology to a simple case to find out its performance in general.

I. Introduction

Presently, federated learning and blockchain technology are broadly discussed. These technologies rely on the top of the decentralized form as a part of the distributed system. These technologies can be described as an intersection of on-device AI, decentralized ledger, and edge computing.

Federated learning is a breakthrough in the machine learning field. It converts the centralized raw data into a decentralized form. It turns the update of AI models upside-down by allowing the devices on edge to contribute to the training model [1]. Furthermore, it provides an effective way to distribute knowledge across a network while preserving the privacy of the individual nodes. On the other hand, blockchain comes with all the advantages of offering solutions that are faced in the centralized model. The core idea behind blockchain technology is a chain-shaped data structured as known as a chain of blocks [2].

In this research, we show the performance of federated learning in a simple use case. The time collection and local loss data are our objectives. Each client is assumed to have a built-in local training data. The connected clients compute and upload the updated AI model via the blockchain. The reward system based on blockchain smart contract (many issues has been solved):

1. Execution is active. If both parties agree with conditions, the smart contract will run the codes without interference.
2. The data record is available since the history of a transaction is recorded.
3. The smart contract is efficient. It is executed within seconds [3].
4. Low fee. There is no paperwork and middleman involvement.

5. The smart contract eliminates user error and fraud (no single point of failure).

II. Definition of Federated Learning

The statistics show that the number of sensors will grow by 1 trillion in 2030 [4]. Due to the increasing number of connected devices, it brings new challenges for AI, especially for the machine learning area. One of the essential problems of developing AI is the difficulties in collecting datasets from several sources. Moreover, the parties have data on each device, which is confidential. To tackle the issues, FL was introduced by the Google AI research team in 2016 as a strategy for improving communication efficiency [5].

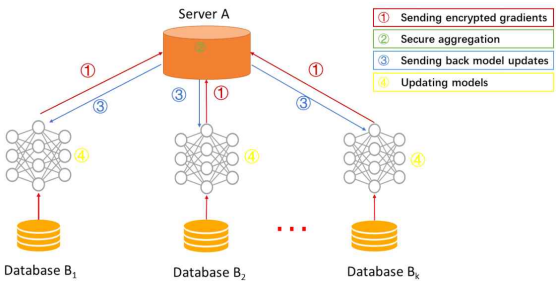


Fig. 1. Architecture of federated learning

$$|V_{FED} - V_{SUM}| < \delta \quad (1)$$

Federated learning has δ accuracy loss.

The architecture of federated learning in general can be seen in Figure 1. Define N data owners $\{F1, \cdots, FN\}$, all of whom wish to train a machine learning model by consolidating their respective data $\{D1, \cdots, DN\}$. A standard method is to put all data together and use $D = D1 \cup \cdots \cup DN$ to train a model $MSUM$. The federated learning system can be interpreted in which process any data owner $F1$ does not expose its data $D1$ to others. Formally, let δ be a non-negative real number as denoted in (1). The general objective of the scheme is to learn the AI model with parameters embedded in a real matrix which is derived from data stored across a significant number of clients. In the initial round, the server propagates the global model to the subset of total clients in the decentralized network.

III. AI on Blockchain

A blockchain-based loyalty rewards system can reduce system management costs with smart contracts that report secure, tracked, transparent transactions to legacy systems, reducing costs associated with errors and fraud. AI on smart contract integration and innovation may provide a more resilient and efficient path for a decentralized system that is interactive for the parties.

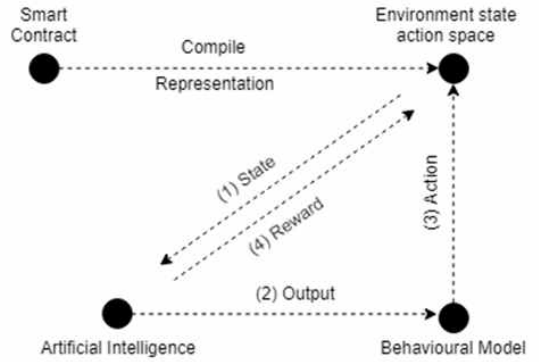


Fig. 2. The basic high-level structure of the AI smart contract model

The model in Figure 2 provides a way to represent smart contract structure design in the right framework, which handles the training of AI agents with an interpretable result. As the transition model towards the blockchain economy space, this could become a critical tool towards offering safer decentralized application (DApps) [6] and validating incentive structures pre-launch [7]. Intuitively, AI agents are trained to optimize the pre-defined objectives before simulating their interaction with real-world applications.

IV. Implementation and Discussion

For the implementation, we use the MNIST dataset for the digit recognition task (see Fig. 3). The learning task is addressed using multiple devices that have local training data

on every device. The collaborative approach uses several client's devices to train their local models with independent data. The clients each train their local model with a subset of the full dataset.

The aggregated server combines the local models updated from the clients into a master model, then sends the master model back to the clients. This process is called one round. When another round begins, the clients train their local copy of the master model by creating another set of individualized local models, send those local models back to the manager afterward. Once again, the aggregated server combines these individualized local models into a new master model and propagates the new master model down to the clients. The cycle of rounds repeats as long as it is necessary and useful.

```
0it [00:00, ?it/s]Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz
9920512it [00:01, 8680138.87it/s]
Extracting ./MNIST-data/raw/MNIST/raw/train-images-idx3-ubyte.gz to ./MNIST-data/raw/MNIS
  0%|          | 0/28881 [00:00<?, ?it/s]Downloading http://yann.lecun.com/exdb/mnist/tra
32768it [00:00, 129951.66it/s]
  0%|          | 0/1648877 [00:00<?, ?it/s]Extracting ./MNIST-data/raw/MNIST/raw/train-la
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz to ./MNIST-data/ra
1654784it [00:00, 2119191.91it/s]
0it [00:00, ?it/s]Extracting ./MNIST-data/raw/MNIST/raw/t10k-images-idx3-ubyte.gz to ./MN
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz to ./MNIST-data/ra
8192it [00:00, 48727.75it/s]
Extracting ./MNIST-data/raw/MNIST/raw/t10k-labels-idx1-ubyte.gz to ./MNIST-data/raw/MNIST
Processing...
Done!
```

Fig. 3. Gathering process for dataset

```
  0%|          | 0/30 [00:00<?, ?it/s]Training 30 round(s) with 1 worker(s) doing 1 epoch(s)

Beginning round 1
        Worker: 5344    local loss: 0.55
  3%||          | 1/30 [00:11<05:21, 11.09s/it]Finished round 1 with global loss: 0.53446

Beginning round 2
        Worker: 5344    local loss: 0.29
  7%|▌         | 2/30 [00:21<05:07, 10.99s/it]Finished round 2 with global loss: 0.39699

Beginning round 3
        Worker: 5344    local loss: 0.28
 10%|▉         | 3/30 [00:32<04:55, 10.93s/it]Finished round 3 with global loss: 0.33696

Beginning round 4
        Worker: 5344    local loss: 0.32
 13%|█▎        | 4/30 [00:43<04:42, 10.87s/it]Finished round 4 with global loss: 0.32599

Beginning round 5
        Worker: 5344    local loss: 0.28
 17%|█▋        | 5/30 [00:54<04:31, 10.85s/it]Finished round 5 with global loss: 0.29530

Beginning round 6
        Worker: 5344    local loss: 0.14
 20%|██        | 6/30 [01:04<04:19, 10.82s/it]Finished round 6 with global loss: 0.25776
```

Fig. 4. First six-rounds performance

Figure 4 is the client's performance for the first-six rounds training of dataset consist of 60.000 examples. Whilst, in Figure 5 presents the performance of the last-five rounds out of 30 rounds. We focus on the global loss data between the client and the aggregated server. We use a simple model of

```
Beginning round 25
        Worker: 5344    local loss: 0.05
 83%|████████▎ | 25/30 [04:30<00:54, 10.84s/it]Finished round 25 with global loss: 0.10432

Beginning round 26
        Worker: 5344    local loss: 0.07
 87%|████████▋ | 26/30 [04:41<00:43, 10.83s/it]Finished round 26 with global loss: 0.10549

Beginning round 27
        Worker: 5344    local loss: 0.02
 90%|█████████ | 27/30 [04:52<00:32, 10.82s/it]Finished round 27 with global loss: 0.09467

Beginning round 28
        Worker: 5344    local loss: 0.05
 93%|█████████▎| 28/30 [05:02<00:21, 10.83s/it]Finished round 28 with global loss: 0.10469

Beginning round 29
        Worker: 5344    local loss: 0.05
 97%|█████████▋| 29/30 [05:13<00:10, 10.84s/it]Finished round 29 with global loss: 0.09330

Beginning round 30
        Worker: 5344    local loss: 0.05
100%|██████████| 30/30 [05:24<00:00, 10.83s/it]Finished round 30 with global loss: 0.08948

Federated Training Time: 324.66
```

Fig. 5. The performance of last five rounds

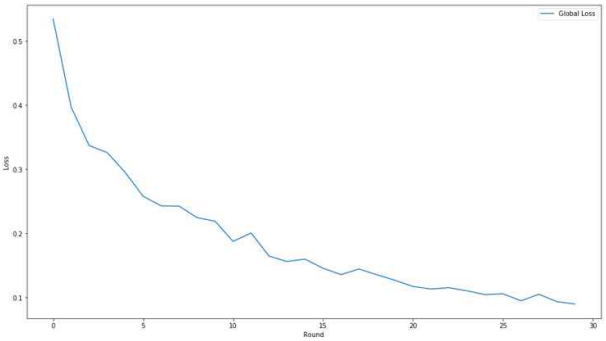


Fig. 6. Global loss data communication

the convolutional neural network to train the image data. The local training data is installed on the client's device such as a smartphone. Due to circumstances, we only set a client to train the model, and send the updated model to the aggregated server.

We implement the federated model using the python programming language that leverages PyTorch for the machine learning framework. We simulate the performance of federated learning for 30 rounds to get the result of a global loss for each round. As we can see in Figure 6, the global loss decreases as the number of rounds increases.

The system adapts well with increasing frequency of communication between the client and the aggregated server. Additionally, each model update requires a certain amount of time to be trained on client devices, so that the aggregated server has enough time to collect each model sent by the client.

The nature of the federated concept is decentralized learning that enables clients located at different geographical locations to conjoint and contributes to an AI model without revealing the personal data to the public. Intuitively, the federated learning ensures the privacy of the users since the personal data is considered sensitive.

Along the same line, blockchain technology helps to manage the consistency of the ledger without third party involvement. The revenue mechanism is also well designed in general. Therefore, the combination of these technologies promises to continue to be developed to improve existing systems. By adopting blockchain technology in the collaborative learning system, it helps to manage the consistency of the ledger (tamper-proof). The reward distribution for the users and the miners are possible to be developed.

Inefficiencies in the flow of information are essential in this system. The data is propagated across millions of users through their devices in highly lacking consistency or regularity in quality or performance. It will be complicated if it occurs on a huge scale. The techniques for updated data collection must also be able to adjust to the field. The consistency of each party also affects the quality of the AI model.

V. Conclusion

The development of collaborative learning, such as federated learning and blockchain, is so rapid that it brings goodness to human life. We also present the performance of federated learning in the digit recognition task using the MNIST dataset. With the mature development of federated learning and blockchain can prevent threats by attackers from threats and vulnerabilities associated with mobile wireless computing. AI on a smart contract allows every AI agent to write regular hash functions of their activity to a blockchain so that it is immutable and inspectable by the encryption keys that make the reward system much better. In terms of future work, we plan to analyze the data protection policies with the appropriate incentives based on our model.

Acknowledgments

This research was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2018R1D1A1B07048944) and partially was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program(IITP-2019-2015-0-00403) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation)

[References]

[1] Qiang Y., Yang L., Chen, and Yongxin Tong, "Federated machine learning: Concept and applications" ACM Transactions on Intelligent Systems and Technology (TIST), 10(2):12, 2019.

[2] S. Wu, Y. Chen, Q. Wang, M. Li, C. Wang, and X. Luo, "CReam: A Smart Contract Enabled Collusion-Resistant e-Auction," IEEE Transactions on Information Forensics and Security, 2018.

[3] M. Mylrea and S. N. G. Gourisetti, "Blockchain: A path to grid modernization and cyber resiliency," in 2017 North American Power Symposium, NAPS 2017, 2017.

[4] Iot analytics.https://iot-analytics.comn/sta te-of-the-iot-update/. Accessed: 2019-09-11

[5] Jakub Konecny, H Brendan McMahan, Daniel Ramage, and Peter Richtarik. "Federated optimization: Distributed machine learning for on-device intelligence", arXiv preprint arXiv:1610.02527, 2016.

[6]"Incentivai." [Online]. Available: https://incentivai .com/product/. [Accessed: 12-Oct-2019].

[7] Xuhui Chen, et.al, "When machine learning meets blockchain: A decentralized, privacy-preserving and secure design", IEEE International Conference on Big Data, 2018.

매직키 획득을 위한 펌웨어 분석

임한울, 김현욱, 윤주범

세종대학교 정보보호학과

nurilmaro@gmail.com, hyunwook711@naver.com, jbyun@sejong.ac.kr

Analysis of Firmware for the Acquisition of Firmware Magic Key

Hanwool Lim, Hyunwook Kim, Joobeom Yun

Department of Computer and Information Security, Sejong University.

요 약

최근 IoT 기술의 발달과 5G통신망 상용화의 시작으로 이전에 비해 다양한 분야에 임베디드 시스템의 디바이스들이 많아지게 되었다. 하지만 짧은 개발기간, 개발인력 부족 등과 같은 이유로 임베디드 디바이스의 펌웨어는 많은 취약점들을 잠재되어져 있다. 임베디드 시스템의 영향력이 점점 증가함에 따라 이러한 사항은 기업, 국가기관, 나아가 사람의 생명까지도 위협되어질수 있다. 이러한 문제 때문에 임베디드 펌웨어의 다양한 취약점들을 분석하기위한 방법들이 연구되어지고 있다. 그 중 한 방법으로 UART를 이용한 연결을 통해 펌웨어를 분석하는 방법이 있다. 과거에는 단순히 UART 연결만으로도 쉘을 획득할 수 있어 다양한 작업을 수행할 수는 있었지만 현재는 UART핀을 숨겨놓거나 혹은 UART 연결에 성공한다 하더라도 바로 쉘을 획득할 수 없게 하는 다양한 방어기법들이 존재한다. 그래서 본 논문은 쉘을 얻을 수 있는 방법 중 매직키(Magic Key)를 입력하여 쉘을 획득할수 있는 방법을 제시하고 있다.

I. 개요

최근 IoT(Internet of Things)기술의 발달과 5G통신망이 상용화되어짐에따라 이전에 단순히 스마트폰끼리만 통신하는 방식이 아닌 여러 가지 임베디드 시스템의 디바이스들을 포함한 일상생활에서 사용하는 모든 제품들이 점점 네트워크화 되어져 서로 통신할 수 있는 시대가 되어지고 있다. 하지만 그러한 현실에 비해 짧은 기획기간, 인력의 부족, 장비의 한정된 크기등과 같은 다양한 이유들로 인해 보안적인 측면에서는 많이 위험에 노출되어있다. 실제로 IoT 부문 OWASP Top10에 따르면 주요 취약점은 서버-클라이언트의 인증부분이 취약하거나 통신중에 패킷을 암호화하지 않는 등 기본적인 부분에 존재한다고 나와있다.[1]

펌웨어의 취약점들을 파악하기 위해서는 임베디드 디바이스와 PC를 연결한 다음 펌웨어를 분석하여야한다. 여러 가지 연결 방법이 존재하고 UART역시 여러 가지 방법들 중 하나이다. 하지만 최근에는 UART에 대한 보안상의 이유로 UART연결에 성공하더라도 쉘을 사용하지 못하게 막아두었을 가능성이 매우 높다. 쉘을 얻기위한 방법으로써는 다양한 방법들이 존재한다.

본 논문의 2장에서는 관련연구인 공유기에 대한 UART연결과 UART 연결 시 쉘을 획득할 수 있는 다양한 방법에 대해 설명하고, 3장에서 UART 연결에서 쉘을 획득할 수 있는 방법 중 하나인 매직키를 찾아낼 수 있는 방법에 대해서 설명한다. 4장은 3장에서 다룬 방법을 PC와 공유기에 직접 입력한 실험 결과를 보여주고 5장에서 결론을 제시하겠다.

II. 관련 연구

2.1 UART

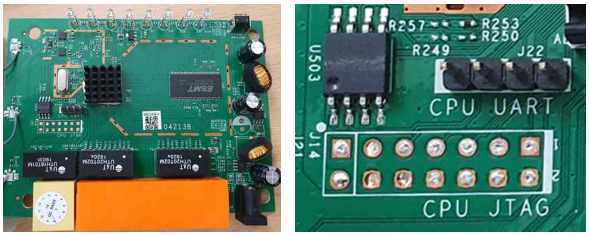
UART는 직렬 통신방식의 범용 비동기 송/수신기의 의미로써 Universal Asynchronous Receiver/Transmitter의 약자이다.

UART에 필요한 핀은 총 4가지로 데이터를

송신하기위한 TX핀, 데이터를 수신하기위한 RX핀 그리고 VCC핀과 GND핀으로 구성되어진다. TX와 RX의 경우 해당장비를 기준으로 송/수신을 구분한다. 예를들어 PC에서의 TX핀은 PC에서 데이터를 송신하는 핀을 의미하고 RX핀은 PC에서 데이터를 수신하는핀을 의미한다. 임베디드 디바이스와 PC가 UART 통신을 하기 위해서는 임베디드 디바이스의 TX와 RX가 PC의 TX와 RX와 교차되어 연결해야한다.[2]

2.2 공유기 UART 연결시도

이번 연구에 사용한 공유기는 N704bcm으로 무선 공유기에 해당한다. 해당 공유기의 내부구조는 [그림 1]과 같다.



[그림 1] IPtime N704bcm

최근에 출시되어지는 임베디드 디바이스의 경우에는 펌웨어에 악성코드를 심어 배포하거나 해킹의 우려가 있기 때문에 보안상의 이유로 UART연결을 할수 있는 핀이 제거되어져있거나 아예 UART연결을 막아두는 경우도 존재한다. 하지만 해당 공유기의 경우 UART핀이 직접적으로 밝혀져 있고 핀헤드 역시 드러나있기 때문에 핀을 찾거나 납땜과 같은 작업을 따로 할 필요가 없다.

공유기의 UART 핀헤더와 PC에 연결되어진 USBtoTTL을 점퍼케이블로 알맞게 연결한 다음 Baud Rate값을 115200으로 설정하여 UART 통신을 시도하면 [그림 2]와 같은 화면이 출력된다.



[그림 2] UART 통신화면

UART통신화면에는 펌웨어 버전, IP주소등 디바이스의 여러 가지 정보들을 출력하고 있다. 하지만 현재 공유기와의 UART통신의 경우 출력은 정상적으로 이루어지지만 단순히 UART연결만으로는 쉘이 획득되어져 있는 상태가 아니라 사용자가 입력한 값들을 받지 않고 있기 때문에 이는 제대로된 통신이라고 보기 어렵다.

2.3 UART 쉘 획득방법

최근 대부분의 임베디드 디바이스들은 보안상의 이유로 단순한 UART연결만으로 쉘을 획득할수 없는 경우가 많다. UART통신 자체를 막혀있는 경우 관리자 메뉴에서 UART통신을 활성화시켜 UART통신을 수행하거나 혹은 임베디드 디바이스의 제조사측에서 아예 UART통신을 수행할 수 없게 막아두는 경우도 존재한다. 하지만 각 임베디드 디바이스마다 쉘을 획득할 수 있는 방법이 존재하는 경우 추가적인 작업을 통해서 쉘을 획득할 수 있다. 쉘을 획득할 수 있는 방법은 다양한 방법들이 존재한다.[3]

이 장에서는 쉘을 획득하는 방법들에 대해서 알아본다.

(1) 쉘은 정상적으로 실행이 되었지만 입력이 되어지지 않는 경우가 있다. 이러한 경우에는 PCB에 UART핀을 제외한 다른 특정 핀들을 서로 점퍼케이블로 연결할 경우 입력기능이 활성화되어질수 있다.

(2) UART 연결을 시도하였을 때 쉘이 아예 출력되지 않는 경우가 있다. 이러한 경우에는 Ctrl+C와 같은 키를 조합하여 입력하거나 매직키를 입력하여 쉘을 획득할 수 있다.

III. 펌웨어 매직키 획득 기법

N704bcm공유기를 UART통신하였을 때 출력되어지는 정보들을 확인하다보면 [그림 3]과 같이 화면에 “press 매직키 to change default setting ”이라는 문장을 찾을 수 있다.

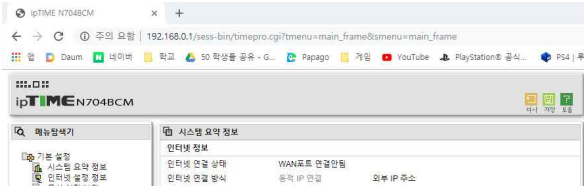


[그림 3] 매직키 입력요청

이를 통해서 해당 공유기는 UART연결하였을 때 쉘을 획득하기 위해서는 매직키의 입력이 필요하다는 것을 알 수 있다.

이 장에서는 펌웨어파일을 IDA 프로그램에 로드하여 해당 펌웨어의 매직키를 획득할 수 있는 방법에 대해서 설명한다.

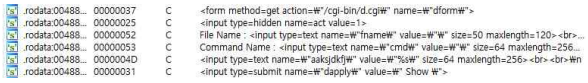
해당 공유기를 PC와 이더넷으로 연결하여 관리자 페이지에 접속하면 [그림 4]에 나온 관리자페이지의 URL과 같이 설정할 때 timepro.cgi라는 파일에 접근함을 알 수 있다.



[그림 4] IPtime 관리자 페이지

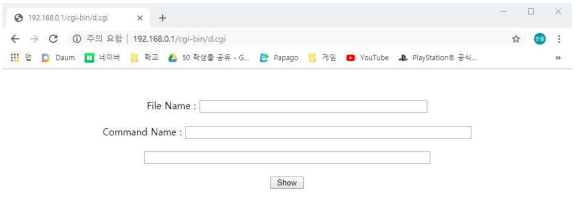
공유기의 경우 일반적으로 HTTP를 통해 서비스되어진다. 이 때 공유기는 보통 사용되어지는 php형식이 아닌 이미 컴파일 되어진 CGI형식을 사용한다. 또한 IPtime홈페이지에서 다운로드한 해당 공유기의 펌웨어 파일(v9.06)을 다운로드하여 binwalk를 사용하여 펌웨어 파일의 압축을 해제하고 CGI파일을 검색하면 timepro.cgi파일이 존재함을 확인할 수 있다. 관리자 페이지의 URL에 timepro.cgi가 존재하고 cgi파일 가운데 timepro.cgi파일의 용량이 가장 큰 것으로 보아 timepro.cgi파일이 이 중 가장 핵심파일일 것이라고 추측이 가능하다.

timepro.cgi파일을 분석하기 위해 IDA프로그램을 통해서 timepro.cgi파일을 로드한 다음 String View를 통해 보면 [그림 5]과 같이 수상한 코드들이 보인다.



[그림 5] timepro.cgi

d.cgi라는 파일의 구성형태를 보면 File Name과 Command Name이라는 이름의 입력값이 존재하고 그것을 전송하는 형태이다. 해당 구성요소를 통해서 d.cgi파일이 디버깅을 위한 페이지임을 추측할 수 있다. d.cgi의 페이지는 [그림 6]과 같다. 단 해당 페이지에 접근하기 위해서는 공유기에 원격지원을 실행할 수 있게 설정값을 변경해야한다.



[그림 6] d.cgi

UART통신때와 마찬가지로 매직키 없이 Command Name창에 명령을 입력하면 빈창만 출력되고 아무런 응답이 없다. 임의의 값을 입력한다음 시도해보아도 마찬가지이다. 세 개의 입력창 중 세 번째 입력창이 매직키를 입력하는 창임을 추측할 수 있다. [그림 5]의 d.cgi페이지의 코드를 보면 세 번째 입력창의 name이 'aaksjdkfj'로 지정되어져 있다. 임의의 명령어를 입력하면 URL에 aaksldkfj이름의 값을 가져와서 비교하는 것을 확인할 수 있다. 해당값을 IDA View를 통해 검색해보면 사용자가 입력한 값들을 하나하나 비교하는 코드를 확인할 수 있다. 해당 부분에서 비교하는 값들을 모아보면 #notenoughmineral^이라는 문자가 만들어진다. 이를 통해 해당 공유기의 매직키의 값이 #notenoughmineral^임을 파악할 수 있다.

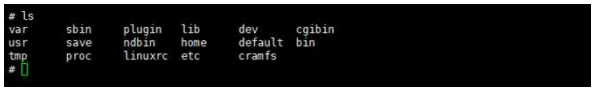
IV.실험결과

발견한 매직키 값을 d.cgi페이지의 세 번째 입력창에 입력하여 ls 명령어를 실행시킨 결과 [그림 7]과 같이 ls 명령어의 결과 값이 하단에 출력되어진다.



[그림 7] d.cgi페이지 ls명령 실행결과

이를 통해 올바른 매직키 값을 구했음을 확인할 수 있다. 디버깅 페이지가 아닌 UART핀으로 직접 연결했을 때 매직키가 완벽하게 입력되어지기 전까지 입력 값이 화면에 출력되어지지는 않지만 매직키를 입력하였을 때 정상적으로 쉘을 획득할 수 있다. 쉘을 획득하고 ls 명령어를 입력하였을 때 결과는 [그림 8]처럼 정상적으로 수행되어진다.



[그림 8] UART통신 매직키 입력결과

[그림 8]을 통해 알수 있듯 UART 통신에서도 펌웨어를 분석하여 알아낸 매직키를 입력하였을 때 똑같이 입력의 효과가 나타난다.

V.결론

본 논문에서는 UART연결 후 쉘을 획득할 수 있는 방법 중 하나인 매직키를 얻는 방법을 설명했다. 모든 임베디드 디바이스가 UART연결을 수행하였을 때 매직키를 통해서 얻을 수 있는 것은 아니고 매직키를 사용한다고 해도

Key값과 찾는 방법이 각 제조사, 펌웨어 버전에 따라서도 다르다. 현재 임베디드의 UART를 하드웨어적이나 소프트웨어적으로 다양하게 막아 놓고 있는 추세이지만 그래도 아직까지도 많은 임베디드 디바이스들이 UART를 이용하여 쉘에 접속할수 있거나 보안상 막아놓은 방법들을 우회하여 연결할수 있다. 본 논문에 설명한 방법들을 통해서 펌웨어의 구조를 분석한 결과 임베디드 디바이스의 특성으로 많은 정보들이 담겨있지않고 명령어 또한 제한적이다. 이후 펌웨어에 원하는 기능을 추가하여 커스터마이징을 하거나 임베디드 디바이스의 취약점 분석에 이용할수 있기 때문에 임베디드 시스템의 구조나 프로그래밍과 같은 분야에서 더 연구가 필요로 되어진다.

[참고문헌]

[1] Pamela Rentz, “OWASP Releases Latest Top 10 IoT Vulnerabilities”, January, 2019, 2019년 10월 30일 접속, https://www.techwell.com/techwell-insights/2019/01/owasp-releases-latest-top-10-iot-vulnerabilities

[2] 김현욱, “임베디드 장비 펌웨어 획득을 위한 UART 연결 방법 연구”, 정보보호동계학술대회, July, 2019

[3] Mongii,Grayhash, “임베디드 디바이스 Serial Port Hacking의 모든 것!”, February, 2016

[4] Cybermong,Grayhash, “임베디드 시스템 취약점 분석”, April, 2015

라즈베리 파이를 활용한 스마트 스피커 및 NAS 로그 분석*

신수민*, 박세준*, 강수진*, 조승기*, 이예솔*, 김기윤**, 김종성*,**

*국민대학교 정보보안암호수학과, **국민대학교 금융정보보안학과

Log Analysis of Smart Speakers and NAS using Raspberry Pi

Sumin Shin*, Sejun Park*, Soojin Kang*, Seungki Cho*, Yesol Lee*, Giyoon Kim**, Jongsung Kim*,**

*Dept. of Information Security, Cryptology and Mathematics, Kookmin University

**Dept. Financial Information Security, Kookmin University

요 약

교육용으로 제작된 라즈베리 파이는 특유의 범용성으로 다양한 애플리케이션 구축에 활용된다. 라즈베리 파이를 통해 구축 가능한 애플리케이션은 스마트 스피커와 NAS 등이 있으며, 최근 사생활 침해 또는 개인 정보 유출 피해의 문제가 제기되고 있다. 본 논문에서는 라즈베리 파이로 구축한 스마트 스피커와 NAS의 로그 및 데이터를 분석하였다. 이를 통해 실제 저장되는 데이터를 확인하고 포렌식적 활용 방안을 제시하였다.

I. 서론

라즈베리 파이 (Raspberry Pi)는 2012년 교육용으로 제작되어 올해 2월까지 2,500만대 이상 판매된 초소형, 초저가의 컴퓨터이다[1]. 다양한 컴퓨트 모듈을 탑재하고 있고, 모듈 설계가 오픈 소스로 되어있어 커스텀 입출력 보드를 생산할 수 있는 라즈베리 파이는 특유의 범용성으로 현재 교육용 외에도 IoT (Internet of Things) 환경 구축, 서버, 스마트 미러, 클러스터링, 게임기 그리고 NAS (Network attached storage) 등을 위해 폭넓게 사용되고 있다[2].

인공지능 스피커와 NAS는 대표적인 라즈베리 파이 활용사례이나, 정보보호에 관한 문제가 대두되고 있다. 인공지능 스피커의 경우 과도한 데이터 수집으로 인한 사생활 침해가 문제가 되고 있으며, NAS의 경우 해킹에 의한 개인 정보 유출 피해가 제기되고 있다[3, 4]. 본 논문에서는 라즈베리 파이로 구축한 스마트 스피커와 NAS에 저장된 데이터 및 로그를 파악하여 분석하였다.

본 논문은 2장에서 스마트 스피커와 NAS를 구축하여 획득한 로그를 설명하며 3장에서는 라즈베리 파이의 주요시스템의 로그를 설명한다. 마지막 4장에서는 결론으로 마무리한다.

II. 스마트 스피커 및 NAS 로그

2018년 기준 AI 스피커 시장 점유율 1, 2위는 알렉사 (Alexa)와 구글 어시스턴트 (Google Assistant)다[5]. 따라서 본 장에서는 알렉사와 구글 어시스턴트를 구축하여 분석했다. NAS는 라즈베리 파이에서 구축 가능한 방식 중 접근성이 좋은 데비안 운영체제 기반의 오픈소스 OMV (OpenMediaVault)를 구축하여 분석했다.

각 프로그램 구축 및 분석에 사용한 도구는 다음과 같다[표 1].

* 본 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2019R1F1A1060634)

[표 1] 구축 및 로그 분석 시 사용한 도구

	모델	운영체제	저장장치
알렉사	Raspberry pi 2 B	Raspbian Buster	SanDisk micro SDHC Class4
구글 어시스턴트	Raspberry pi 3 B	Raspbian Jessie Lite 2017-02	SanDisk Ultra
NAS	Raspberry pi 3 B	Raspbian Jessie Lite 2017-02	SAMSUNG EVO

2.1 알렉사

알렉사는 아마존에서 개발한 클라우드 플랫폼으로 아마존에서 제공하는 Alexa Skills Kit과 Alexa Voice Service가 합쳐져 인공지능 비서 혹은 인공지능 스피커로 사용된다. 현재 아마존에서는 외부 개발자들이 자신의 하드웨어나 서비스에 알렉사를 탑재할 수 있게 API (Application Programming Interface)를 제공하고 있다[6]. 알렉사는 현재 8개국의 언어를 지원하며 미디어 애플리케이션 조작, 스케쥴링, 쇼핑, 검색 및 IoT 기기 조작 등 다양한 명령을 음성으로 처리할 수 있다. 라즈베리 파이에 구축된 알렉사를 사용 시 저장되는 로그와 설명은 다음과 같다[표 2].

[표 2] 알렉사 사용 시 저장되는 로그

로그	데이터
/var/log/daemon.log	Main Title : 명령어
/var/log/syslog	(알렉사가 인식한 명령어 저장)
/var/log/daemon.log	Started Raspberry Pi Alexa. (알렉사 시작 로그)

(그림 1)은 실제 저장된 명령어 로그의 일부다.

```
Oct 29 22:19:40 raspberrypi bash[376]: # Main Title      : Tell me about Ghost
Oct 29 22:20:12 raspberrypi bash[376]: # Main Title      : Samsung
Oct 31 00:38:19 raspberrypi bash[386]: # Main Title      : What time is it in Seoul?
Oct 31 00:38:47 raspberrypi bash[386]: # Main Title      : Seoul, South Korea
Oct 31 00:41:36 raspberrypi bash[386]: # Main Title      : Tell me about Heroes of the Storm
Nov  2 03:05:50 raspberrypi bash[374]: # Main Title      : Do you know kimchi?
```

(그림 1) 알렉사 로그 데이터

알렉사 구축 시 저장되는 주요 데이터베이스와 포함된 데이터는 다음과 같다[표 3].

[표 3] 알렉사 주요 데이터베이스

데이터베이스	테이블	데이터
cblAuthDelegate.db	refreshToken	알렉사 실행 시 사용되는 인증 토큰값
deviceSettings	deviceSettings	시간대(timezone), 언어(locales), 알렉사 호출 단어(wake Word)
miscDatabase	capabilitiesDelegate_capabilitiesPublishMessage	각종 아이디 및 시리얼 번호 (Client ID, Product ID, Device Serial Number)

2.2 구글 어시스턴트

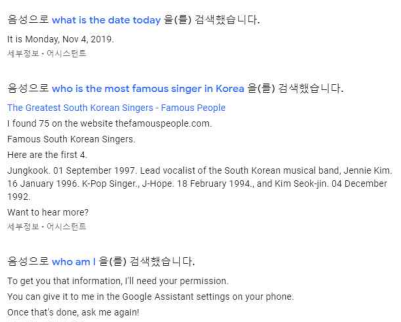
구글 어시스턴트는 구글이 개발한 인공지능 가상 비서다. 알렉사와 동일하게 구글에서 API를 제공하고 있으며 개발자는 이를 통해 하드웨어 혹은 서비스에 탑재할 수 있다. 사용 용도는 알렉사와 동일하다.

라즈베리 파이에 구축된 구글 어시스턴트 사용 시 저장되는 로그는 다음과 같다[표 4].

[표 4] 구글 어시스턴트 사용 시 저장되는 로그

로그	데이터
/var/log/daemon.log	Started Google Assistant. (구글 어시스턴트 실행시간)
	Startup finished in 시간 (kernel) + 시간 (userspace) = 시간 (사용 지속 시간)

알렉사와 다르게 명령어와 그에 대한 답변이 라즈베리 파이에 저장되지는 않지만, 사용 기록에 대한 정보가 남으며 구글의 '내 Google 활동'을 통해 지난 명령어를 확인할 수 있다[7].



(그림 2) '내 Google 활동'에 저장된 로그 데이터

2.3 NAS

NAS는 컴퓨터 네트워크에 연결된 데이터 저장 서버로 기업뿐만 아니라 개인적으로도 많이 사용된다[8]. OMV를 이용한 NAS는 단순 데이터 저장 및 공유뿐만이 아닌 웹페이지를 통한 NAS 접근을 지원하며 이메일 알림, 연결된 네트워크 디바이스 부팅, S.M.A.R.T.(Self-Monitoring, Analysis and Reporting Technology) 등을 지원한다. 라즈베리 파이에 OMV를 통해 구축된 NAS를 사용할 시 저장되는 로그와 설명은 다음과 같다[표 5].

[표 5] NAS 사용 시 저장되는 로그

로그	데이터
/var/log/nginx /openmediavault -webgui_access. log	OMV 접속 원격 주소, 시간 등 (공유 폴더에 접속한 기록)
/var/log/nginx /openmediavault- webgui_error.log	접속 주소, 서버 종류 등 오류 발생 시간, 오류 내용 등 (오류에 관한 기록)
/var/log/ syslog	FTP session opened. (FTP[1]로 공유 폴더에 접근) FTP session closed. (FTP로 공유 폴더에 접근 해제)

OMV를 통해 구축된 NAS는 '/etc/samba/smb.conf' 파일에 공유 폴더에 관한 정보가 저장된다. 'smb.conf' 파일에는 NAS를 통해 공유한 폴더의 이름, 실제 경로, 사용자 설정, 접근 권한 등 다양한 설정이 저장되어 있다(그림 3).

```
#======================= Share Definitions =======================
[Folder1]
path = /srv/dev-disk-by-id-usb-SanDisk_Ultra_USB_3.0_4C530000250712117372-0-0-part1/Folder1
guest ok = no
read only = no
browseable = yes
inherit acls = yes
inherit permissions = no
ea support = no
store dos attributes = no
vfs objects =
printable = no
create mask = 0664
force create mode = 0664
directory mask = 0775
force directory mode = 0775
hide special files = yes
follow symlinks = yes
hide dot files = yes
valid users = "[redacted]","[redacted]","[redacted]"
invalid users =
read list =
write list = "[redacted]","[redacted]","[redacted]"
```

(그림 3) smb.conf 내용

1) FTP (File Transfer Protocol) : 파일 전송을 하기 위한 프로토콜

III. 주요시스템 로그

본 장에서는 라즈베리 파이에 주요시스템에 관한 로그를 설명한다. 본 논문에서 정의한 라즈베리 파이의 주요시스템 로그는 네트워크와 계정에 관한 로그이다.

네트워크 로그의 경우 라즈베리 파이에서 구축한 애플리케이션의 정상작동에 필요하기 때문이며, 계정에 관한 로그는 악의적인 접근의 여부를 파악할 수 있기 때문이다. 시스템 로그 수집에 사용한 도구는 2장과 동일하다.

3.1 네트워크

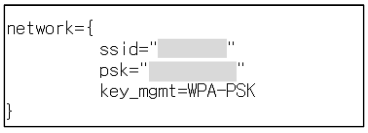
라즈베리 파이는 LAN 또는 WiFi를 통해 네트워크를 사용할 수 있으며 블루투스 통신 또한 가능하다. 블루투스 통신으로는 키보드, 마이크 그리고 스피커를 연결할 수 있다. 라즈베리 파이에 구축한 애플리케이션에 따라 네트워크와 블루투스 연결을 필수적으로 요구할 수 있다. 실제 라즈베리 파이에 저장되는 로그의 경로와 데이터는 다음과 같다[표 6].

[표 6] 네트워크 관련 로그

로그	데이터
/var/log/ daemon.log	ssid="**" auth_failures=1 reason=WRONG_KEY (WiFi 비밀번호를 잘못 입력했을 경우)
	ssid="**" auth_failures=1 reason=CONN_FAILED (올바른 WiFi 비밀번호를 입력했지만, 연결을 실패한 경우)
	CONNECTED - Connection to [wifi_mac_addr] (WiFi 연결 성공)
/var/log/ daemon.log	/org/bluz/hci0/dev_[device_mac_name] /fd0 : ready Starting Bluetooth service. (Bluetooth 연결 성공)
	Stopping Bluetooth. Stopped target Bluetooth. (Bluetooth 연결 해제)
	Operation not permitted connect error : Operation already in

	progress connect failed for [device_mac_name] (Bluetooth 연결 해제)

‘/etc/wpa_supplicant/wpa_supplicant.conf’ 파일에는 WiFi의 SSID (Service Set IDentifier)와 비밀번호가 평문 형태로 저장되어있다(그림 3).



(그림 4) wpa_supplicatn.conf 내용

3.2 계정

사용자의 계정과 관련된 로그는 ‘/var/run’ 경로에 존재하는 utmp 파일과 ‘/var/log’ 경로에 존재하는 btmp, wtmp, lastlog 파일에 저장된다. 네 개의 파일 모두 바이너리 형식이므로 터미널 창에서 명령어를 입력해서 내용을 확인해야 한다. 각각의 로그와 데이터는 다음과 같다[표 7].

[표 7] 계정 관련 로그

로그	데이터
/var/run/utmp	사용자 이름, TTY[2], 로그인 시간, 사용 시간 등 (현재 로그인되어있는 계정 정보)
/var/log/btmp	사용자 이름, TTY, 실패한 시간 등 (로그인에 실패한 계정 정보)
/var/log/wtmp	사용자 이름, TTY, 시간 정보, 상태 (과거와 현재의 로그인/로그아웃 기록) SSH를 통해 접속했을시 TTY 대신 pts
/var/log/lastlog	사용자 이름, TTY, 최근 로그인 시간 등 (계정별 마지막 로그인한 시간 정보)

IV. 결론

본 논문에서는 사생활 침해 및 개인 정보 유출 등의 문제로 보안 문제가 대두되고 있는 스마트 스피커와 NAS를 라즈베리 파이에 구축하여 로그를 수집하고 데이터를 정리했다. 스마트 스피커의 경우 사용자가 명령한 데이터들이 모두 저장되고 있음을 확인했고, NAS의 경우 접근 가능 경로 및 설정 등을 확인했다. 또한, 기본적인 시스템 로그의 일부를 분석했다. 시스템 로그의 경우 애플리케이션의 정상작동 확인을 위한 네트워크 로그와 정상 사용자의 접근 여부 확인을 위해 계정 로그가 남아있다.

스마트 스피커의 경우 과도한 개인 정보 수집은 없었으나 이용자의 타임라인을 구축할 수 있는 데이터를 획득할 수 있다. 또한, NAS의 경우 침해사고가 일어났을 때 시스템 로그와 함께 활용하여 대응할 수 있는 데이터를 획득할 수 있다. 따라서 본 논문은 용의자가 본 논문의 애플리케이션을 사용할 시 타임라인을 구축하거나 침해사고가 발생했을 시 도움이 될 것으로 기대한다. 마지막으로 라즈베리 파이는 특유의 범용성에 의해 악용될 여지가 있으므로 향후 더 많은 분석이 필요하다.

2) TTY (Teletypewriter) : 리눅스 디바이스 드라이브 중 콘솔 또는 터미널

[참고문헌]

[1] “Raspberry Pi Founder Shares 10 Things You May Not Have Known”, tom’sHARDWARE, 2019. (2019. 02. 11)

[2] “The huge list of Raspberry Pi uses”, https://www.patsnap.com/blog/raspberry-pi-uses

[3] “AI 스피커: 보안, 사생활침해 우려에도 믿어도 될까”, BBC, 2019. (2019. 07. 13)

[4] “NAS 표적, 쉬운 비밀번호는 즉시 교체해야”, IT동아, 2019. (2019. 07. 29)

[5] “US Smart Speaker Users, by Brand”, eMarketer, 2018. (2018. 04. 01)

[6] “Amazon developer”, https://developer.amazon.com/

[7] “My Google”, https://myactivity.google.com/myactivity

[8] “Network-attached storage”, https://en.wikipedia.org/wiki/Network-attached_storage

디지털 복합기 RAW 9100 포트의 위험성 연구

김준태*, 박소영*, 윤지인*, 윤형준*, 이용우*, 지한별**, 박의성**, 김홍진***

*KITRI 차세대 보안리더 양성 프로그램

**라온화이트햇

***LG CNS

A Study on the Risk of RAW 9100 Port in MFP

Jun-Tae Kim*, So-Young Park*, Jee-In Yoon*, Hyeong-Joon Yun*, Yong-Woo Lee*

*Best of the Best, Korea Information Technology Research Institute(KITRI)

요 약

디지털 복합기는 프린터, 스캐너, 팩시밀리 등의 기능이 통합된 장치로, 많은 기업과 공공기관에서 업무의 편의성, 비용 절감 등을 위해 사용하고 있다. 이에 따라, 디지털 복합기는 다양한 정보를 다루게 되어 수많은 보안 위험이 등장했다. 실제로 RAW 9100 포트를 이용한 프린터 5만대의 침해 사고가 발생했고 RCE 취약점도 발견되었다. 본 논문에서는 디지털 복합기 보안성 향상을 위해 RAW 9100 포트로 공격을 시도하여 위험성을 연구한 후 대응 방안을 제시한다.

I. 서론

IT 기술의 급속한 발전에 따라 구분되던 복사기, 프린터, 스캐너 등의 각 영역이 디지털 복합기라는 영역으로 통합·확대되었다. 디지털 복합기는 인쇄·스캔·복사·팩스뿐만 아니라 대용량 문서 저장 기능과 다양한 네트워크 응용 서비스 등을 제공하는 장치이다. 많은 기업과 공공기관에서는 이러한 디지털 복합기를 업무의 편의성, 비용 절감, 공간 절약 등을 위해 사용하고 있다.[1] 디지털 복합기의 접근성이 증가함에 따라 수많은 보안 위험이 등장하게 되었고, 제조사는 이에 대응하여 다양한 보안 기능이 탑재된 디지털 복합기를 출시하고 있다.

그러나 디지털 복합기의 보안 위험 인식 수준은 PC나 모바일 기기와 같은 대중적인 기기에 비해 낮은 편이다.[2] 또한, 제조사들이 경쟁적으로 보안 연구를 진행하기 때문에 보안 수준의 차이가 존재하여 기본 설정이 각각 다르다.

한편, HP에서는 RAW 9100 포트에 대한 RCE 취약점(CVE-2017-2741)이 발견되었고, 2018년에는 외부 네트워크에 공개된 프린터의 RAW 9100 포트를 통해 5만대를 해킹하는 사건이 발생했다.

이에 본 논문에서는 디지털 복합기 보안성 향상에 기여하기 위해 RAW 9100 포트의 제조사별 기본 설정 현황과 위험성을 연구하려고 한다.

II. 디지털 복합기의 인쇄 프로토콜

2.1 디지털 복합기 개요

디지털 복합기는 인쇄·팩스·스캔·복사 등의 다양한 사무 처리 기능이 통합된 장치이다. 또한, 디지털 복합기는 내장된 대용량 하드디스크를 통해 대용량 문서 저장이 가능하고, USB와 다양한 네트워크 응용 서비스(TCP/IP, WiFi, 웹서버, FTP 등) 등을 통해 사용자에게 편의성을 제공한다. 이러한 디지털 복합기는 디지털

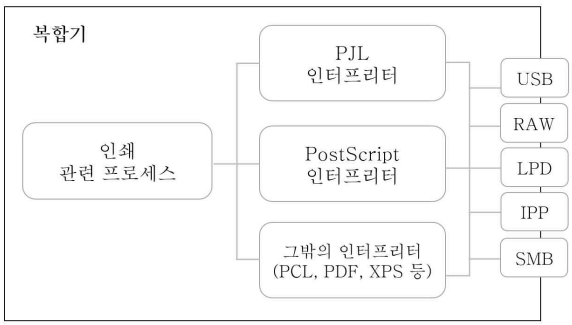
프린터, MFP(Multi Function Printer), 다기능 장치(MFD, Multi Fuction Device), 올인원(AIO, all-in-one) 등으로 불린다.



[그림 1] 디지털 복합기의 정의

2.2 네트워크 인쇄 프로토콜 개요

프린터 기능이 통합된 디지털 복합기는 사용자의 편의를 위해 USB, 네트워크 등을 통한 인쇄를 지원한다. 프린터에서 가장 일반적으로 사용되는 네트워크 인쇄 프로토콜은 RAW, LPD, IPP, SMB 이다.[4] 이러한 프로토콜들은 아래 [그림 2]와 같이 프린터에서 실행되는 프린터 언어(PJL, PostScript 등)로 변환된 인쇄 문서 등을 네트워크를 통해 디지털 복합기에게 전달하는 역할을 수행한다.



[그림 2] 복합기의 인쇄 관련 구성요소

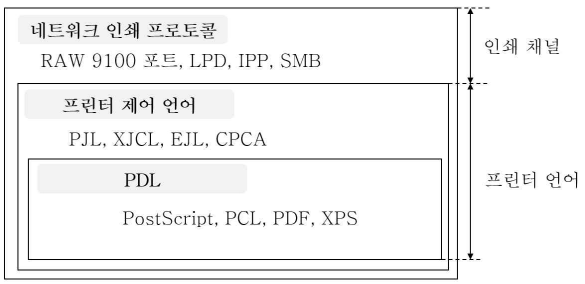
2.2.1 RAW 9100 포트

RAW 9100 포트는 엄연히 프로토콜은 아니며, 해당 포트에 전송된 모든 RAW 데이터는 TCP를 통한 병렬연결처럼 프린터에서 직접 처리된다. 해당 포트에 프린터 언어를 전송하면, 전송된 프린터 언어는 프린터에서 해석되고 프린터는 상태 및 오류메시지 등으로 클라이언트에게 응답한다. 즉, RAW 9100 포트는 인쇄뿐만 아니라 프린터 언어를 입력하고 결과를 얻을 수 있는 양방향 채널이 된다. 한편, RAW 9100 포트 인쇄는 JetDirect 또는 AppSocket으로도 불린다.

2.3 프린터 언어 개요

2.3.1 프린터 제어 언어

프린터 제어 언어는 복사 매수, 종이 크기와 같은 프린터의 세팅 및 인쇄 작업 등을 제어하기 위해 사용되는 언어로, [그림 3]과 인쇄 프로토콜과 PDL 사이에 존재하는 선택적인 계층이다. 프린터 제어 언어는 후지제록스의 XJCL(Xerox Job Control Language), 앱손의 EJL(Epson Job Language), 캐논의 CPCA (Common Peripheral Controlling Architecture) 등과 같이 제조사별로 개발되어 사용되는데, HP가 고안한 PJL(Printer Job Language)이 사실상 표준이 되었고, 다양한 프린터에서 지원된다.[4] 이번 연구에서 이러한 PJL을 이용하여 분석을 진행했다.



[그림 3] 프린터 언어의 캡슐화

2.3.2 PDL(Page Description Language)

PDL은 실제 문서의 모양을 표현하는 언어이다. 프린터에 전송되는 문서들은 프린터 드라이버에 의해 하나의 PDL로 변환되어 프린터에 전달되며, 프린터가 변환된 PDL 명령을 수행함에 따라 출력물이 인쇄된다. PDL도 프린터 제

어 언어와 같이 제조사별 언어가 존재하며, 가장 일반적인 PDL에는 PS(PostScript), PCL(Printer Command Language), PDF(Portable Document Format)가 있다.[5]

III. RAW 9100 포트 위험성 연구

3.1 제조사별 기본 설정 현황

먼저 연구를 진행하기에 앞서 국내외 4개 제조사 디지털 복합기 모델의 RAW 9100 포트 기본 설정값을 조사했다. [표 1]은 해당 결과이다.

구분	제조사 A	제조사 B	제조사 C	제조사 D
기능 존재 여부	O	O	O	O
활성화 여부	O	O	X	O

[표 1] 제조사별 RAW 포트 기본 설정

3개 제조사의 기기는 RAW 9100 포트를 기본적으로 허용했으며, 제조사 C의 기기의 경우는 비활성화되어 있었다.

3.2 RAW 9100 포트 위험성 연구

본 연구에서는 여러 디지털 복합기에서 통용되는 PJL을 이용하여 RAW 9100 포트의 위험성을 분석한다. 분석 과정은 디지털 복합기의 파일 시스템 접근, 파일 시스템 변조를 통한 원격 코드 실행 확인 순으로 진행했다.

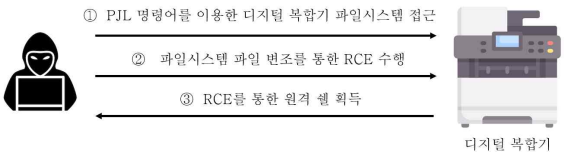
우선 PJL에는 파일 시스템에 접근이 가능한 명령어가 존재함을 확인했다. 그중 파일 시스템 내부의 파일과 디렉터리 목록을 출력해주는 FSDIRLIST와 호스트에서 디지털 복합기로 파일 쓰기가 가능한 FSDOWNLOAD 명령어를 조사했다.

FSDIRLIST 명령어	@PJL FSDIRLIST NAME=“0:/etc” ENTRY=1 COUNT=1000
FSDOWNLOAD 명령어	@PJL FSDOWNLOAD FORMAT:BINARY SIZE=112 NAME=“0:/etc/profile”

[표 2] PJL 명령어 예시

[표 2]과 같은 명령어를 작성한 뒤, RAW 9100 포트로 패킷을 전송했다. 디지털 복합기의 파일 시스템에 성공적으로 접근했으며 경로 탐색 및 파일 쓰기가 정상 수행되었다.

파일 시스템에 임의의 파일 읽기/쓰기가 가능한 경우, 공격자는 DoS(Denial of Service), 정보 유출(Information Disclosure), RCE(Remote Code Execution) 등 다양한 공격을 진행할 수 있다. 그 가운데 RCE를 통한 디지털 복합기 장악을 시도했다. 연구 대상 기기는 취약점이 알려진 버전의 리눅스 커널을 사용했으며, 해당 커널 영역의 이벤트 관리 파일을 변조하여 원격 쉘 접근 권한을 획득했다.



[그림 4] RAW 9100 포트를 이용한 RCE 과정

IV.결론

디지털 복합기는 기업·기관에서 수많은 정보를 처리하는 중요한 장치이다. 하지만 디지털 복합기에서 사용하는 RAW 9100 포트로 기기를 장악할 수 있는 위험성을 확인했고, 여러 제조사에서 기본적으로 해당 포트를 활성화하고 있다. 공격자가 디지털 복합기를 장악하면 기능을 손상·마비시키거나 기기 내부 혹은 외부 저장매체의 정보를 탈취·변조하는 등의 보안 사고를 야기할 수 있다.

RAW 9100 포트로부터 발생 가능한 위험을 줄이기 위한 대응 방안은 첫 번째로, 해당 포트를 비활성화하는 것이다. 두 번째로, 디지털 복합기의 네트워크 설정을 변경 가능한 관리자 웹페이지에 접근 통제 정책을 설정하는 것이다.

관리자 외 IP는 접근을 통제하고, 관리자 페이지의 기본 비밀번호를 변경해야 한다.

향후에는 디지털 복합기에서 발생할 수 있는 추가적인 보안 위협을 식별하고, 기본 보안 수준을 높일 수 있는 표준 가이드라인에 대한 연구가 필요하다.

[참고문헌]

[1] 이광우, 김승주, 기업 비밀정보 유출 방지 및 보호 관점에서의 디지털 복합기 보안 기술 동향 분석, 한국정보보호학회, 2월, 2010

[2] HP, Printer Security: The New IT Imperative, Survey, May, 2018

[3] 김경애, 프린터 5만대 해킹 수법, 국내에서 지속적으로 포착, 보안뉴스, 12월, 2018

[4] Jens Müller, Vladislav Mladenov, Juraj Somorovsky, and Jörg Schwenk, SoK: Exploiting Network Printers, IEEE, February, 2017

[5] Kritkorn Karntikoon, Cattalyya Nuengsigkapian, Korrawat Pruegsanusak, Suchan Vivatsethachai, Hacking Printers: MIT's Printers Security Analysis, May, 2018

임베디드 시스템 펌웨어 취약점 퍼저 분석

김현욱, 이규원, 김원철, 윤주범

세종대학교 정보보호학과

hyunwook711@naver.com, lee.kw926@gmail.com, atc1827@gmail.com, jbyun@sejong.ac.kr

Embedded System Firmware Vulnerability Fuzzer Analysis

Hyunwook Kim, Kyouwon Lee, Wonchul Kim, Joobeom Yun

Department of Computer and Information Security, Sejong University.

요 약

최근 인간의 편리를 위해 만들어진 IOT기술의 상용화로 많은 임베디드 디바이스들이 개발되어지고 있다. 이뿐만이 아니라 각종 산업현장이나 의료센터, 국방부등에서 다양한 기술들의 편리성을 위해 임베디드 디바이스가 개발되어 만들어지고 있다. 임베디드 디바이스들이 다양해지고 많아지고 있지만 임베디드 디바이스의 보안에 대해선 무방비한 상태로 출시되어지고 있는 제품들도 많아지고 있다. 그래서 최근 해커들은 사람들에게 가장 밀접해 있고, 필요로하는 임베디드 디바이스를 타겟으로 잡아 취약점을 탐색하고 공격하는 사례들이 최근에 급격히 증가했다. 이로 인해 보안전문가들은 임베디드 디바이스 안에 어느 취약점이 있는지 빠르게 파악하기위해 임베디드 디바이스를 위한 퍼저를 개발하고 있다. 본 논문에서는 임베디드 디바이스 퍼저 중 하나인 Firm AFL을 직접 설치해 실험한 결과를 보여준다.

I. 개요

인간의 편리성을 위해 개발되어지고 있는 임베디드 디바이스들은 현재 더 다양한 장비들이 나오고 있다. 개인적으로 편리하기 위해 만들어진 디바이스들이 존재하는가하면, 사업현장에서도 일처리를 간편하기 위해 개발되어지는 임베디드 디바이스들이 많이 개발되어지고 있다. 이로 인해 임베디드 디바이스의 공격 사례들이 날마다 증가하고 있고, 혹여 새로운 취약점이 발견되면 공격하거나 취약한 부분을 보안하기 위해 큰 액수의 돈으로 취약점이 거래되고 있다.

임베디드 디바이스의 펌웨어는 하드웨어를 제어하는 소프트웨어이다. 하드웨어의 한 부분, 한 부분을 제어하며 디바이스들이 잘 동작할 수 있게 하는 부분인데, 이러한 펌웨어에 취약점이 존재하면 취약한 부분의 공격으로 인해 디바이스 내의 중요한 정보를 얻을 수 있거나 관리자 권한을 얻을 수 있다. 이로 인해 보안전문가들은 디바이스 내에 취약점을 빠르게 판단하기 위해 임베디드 디바이스의 퍼징을 시도하기 위해 기존의 퍼저들을 변형시키면서 취약점을 찾는 연구를 진행하고 있다. 본 논문은 가장 유명한 퍼저중에 하나인 AFL(American Fuzzy Lop)을 변형시킨 Firm AFL에 대해 살펴본다.

본 논문의 2장에서는 관련연구인 퍼징과 임베디드 디바이스 퍼저, QEMU등을 서술하고, 3장에서는 Firm AFL의 설치과정을 서술, 4장은 FIirm AFL로 펌웨어를 실험해본 결과를 서술한다. 그리고 마지막 5장에서 결론으로 본 논문이 구성되어 있다.

II. 관련 연구

2.1 퍼징

퍼징은 특정 프로그램을 타켓으로 잡고 무작위 값을 대입하여 충돌 또는 메모리 누수를 발생시켜 취약점을 찾는 소프트웨어 기술이고 구조화된 입력을 취하는 프로그램을 테스트하는 기술이다. [그림 1]은 퍼징의 기본 동작을 그림으로 나타낸다.



[그림 1] 퍼징의 기본 동작

퍼징은 프로그램 구조를 얼마나 인지하는지에 따라 화이트 박스와, 블랙 박스 퍼징으로 나뉘고, 화이트 박스와 블랙 박스를 섞어 놓은 그레이 박스 퍼징 3가지로 나뉜다. 또한 퍼저는 입력이 처음부터 생성하는지, 아니면 기존의 입력값을 수정하는 식으로 입력값이 만들어지고 있다.

퍼징의 종류는 화이트 박스 퍼저, 심볼릭 익스큐션, 임베디드 디바이스 퍼저, 네트워크 퍼저 등으로 나눌 수 있는데, 본 논문에서 제시하는 퍼저는 펌웨어 퍼저이다.

2.2 임베디드 디바이스 퍼징[2]

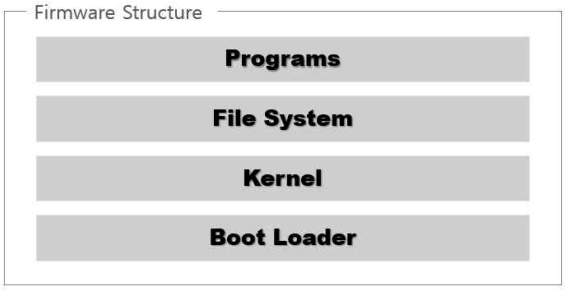
임베디드 디바이스를 대상으로 하는 퍼징 기술은 네트워크 퍼저와 펌웨어 퍼징이다. 네트워크 퍼징같은 경우는 우리가 흔히 사용하는 LAN을 임베디드 디바이스에 연결시켜 특정 프로토콜을 대상으로 무작위 값을 보내 퍼징하는 방식이다. 네트워크 퍼징같은 경우 네트워크에서 주고 받는 프로토콜을 상대로 퍼징을 진행하기 때문에 프로토콜 규격에 맞춰 다양한 값을 변형시켜가며 퍼징해야 한다. 네트워크 퍼징의 종류로는 TLS-Attacker, DELTA,,T-Fuzz, Sulley 등이 있다.

임베디드 디바이스를 대상으로 하는 또 다른 퍼징 기술은 임베디드 디바이스의 펌웨어를 획득해 가상 에뮬레이터에 펌웨어 이미지를 올려 가상화 에뮬레이터를 실행하면서 임의의 값을 대입해 퍼징하는 기술이다. 펌웨어를 가상 에뮬레이터를 사용해 퍼징을 하는 이유는 실제 디바이스에 퍼징을 하게될 경우 충돌이나 메모리 누수가 일어나도 모니터링할 방법이 없고 디바이스가 고장나도 어느 부분에서 고장났는지 알 수 없기 때문이다.

2.3 펌웨어

펌웨어는 임베디드 디바이스의 하드웨어 장치를 제어해주는 소프트웨어이다. 소프트웨어이기는 하나 장치를 제어하고 빠르게 부팅이 되야하고 항상 존재해야하기 때문에 비휘발성인 ROM 또는 PROM에 저장되어있다. 펌웨어는 임베디드 디바이스마다 아키텍가 다르고 동작하는 기능이 다르다. CPU의 스케줄링 방식, 인터럽트처리 방식 등에 따라 펌웨어가 달라진다.

[그림 2]는 가장 기본 적인 펌웨어의 구성을 나타내고 있다.



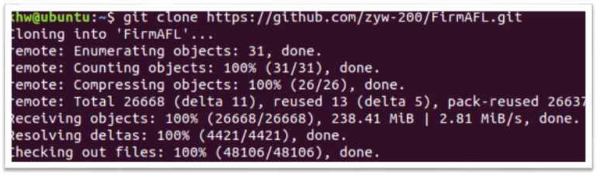
[그림 2] 펌웨어의 기본 구조[1]

[그림 2]를 살펴보면 부트 로더(Boot Loader) 위에 커널(Kernel)이 있고, 그 위에 파일 시스템을 얻는다. 그 이후 OS나 임베디드 디바이스에 필요한 프로그램을 올리면 동작하게 되는데, 여기서 부트 로더는 커널을 올리기 위해 주요 하드웨어인 CPU, 네트워크, RAM등을 초기화하고, 커널은 인터럽트 처리, 메모리 관리, 스케줄링 등을 처리한다. 파일 시스템은 플래시 메모리에 데이터를 올바르게 저장하고 관리하기 위한 역할을 수행한다.

III. 펌웨어 Fuzzing을 위한 Firm AFL 설치

본 연구에서 펌웨어를 퍼징하기 위해 사용한 도구는 Firm AFL[1]이다. Firm AFL은 임베디드 디바이스의 펌웨어를 QEMU라는 가상화 에뮬레이터에 올리고 가상화 에뮬레이터를 실행해 펌웨어를 실행하여 그 안에 무작위 값을 넣으며 퍼징을 하는 기술이다.

Firm AFL은 오픈소스이고 Github에서 다운받을 수 있다. 하지만 QEMU와 연동 시키고 여러 가지 빌드과정이 복잡하다. [그림 3]에서 보이는 명령어를 통해 github에서 Firm AFL을 다운 받는다.



[그림 3] Firm AFL 다운로드 과정

Github를 통해 다운로드를 진행하고 Firm AFL User_mode와 System_mode를 컴파일을 진행해 줘야한다. 하지만 여기서 QEMU[4]를 설치해야 하는데 QEMU를 컴파일하기 위해 Glib-2.22 / Gthread-2.0을 설치하고 빌드해야 한다. 또한 QEMU는 이미지화시킬 수 있는 QEMU-Utills를 설치하고 QEMU의 아키텍처를 사용하기 위해 QEMU-system-mips, arm과 같은 아키텍처들을 설치한다. 그리고 마지막으로 필요한 시스템이 Firmadyne[3]인데, Firmadyne는 임베디드 펌웨어의 에뮬레이션 및 동적 분석을 수행하지 위해 자동화된 시스템이다. 이 시스템을 설치해야 QEMU에 아키텍처가 제대로 설정될 수 있다. Firmadyne를 설치하는 과정은 Github에서 다운을 받고 설치 매뉴얼을 통해 바르게 설치해야 한다. Firm AFL을 실행하기 위해선 QEMU와 Firmadyne 시스템이 필요하고 매뉴얼에 맞춰 바르게 설치해야 Firm AFL이 제대로 동작한다.

IV.실험결과

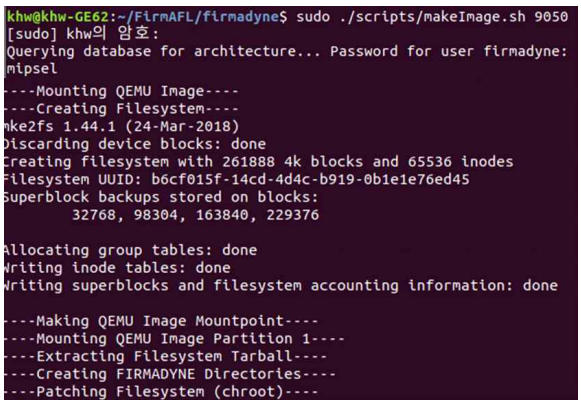
실험에 사용된 펌웨어는 DIR-815 공유기의 펌웨어 1.01버전과 DIR-850L 공유기의 펌웨어 1.03버전이다. Firm AFL을 실행하기에 앞서 펌웨어 이미지를 추출하고 QEMU에 마운트 시키고 네트워크를 삽입하는 과정이 필요하다.



[그림 4] 펌웨어 이미지 추출 과정

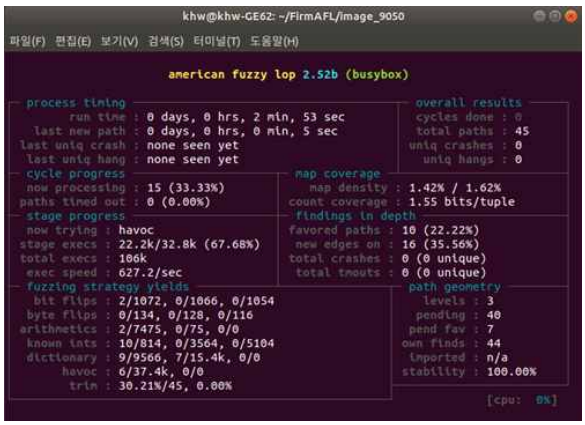
[그림 4]는 펌웨어를 추출하는 과정이다 Firmadyne의 기능 중에 추출기를 이용해 추출을 하게되면 Firmadyne의 DB에 저장된 펌웨어 ID를 찾아 압축한다. 이후 Firmadyne의 기능인 getArch를 이용해 펌웨어의 아키텍처를 찾는다.

[그림 5]는 이미지 추출작업과 아키텍처를 확인하는 작업이 끝나고 QEMU에 이미지를 마운트 시키는 그림이다. 이 기능 또한 Firmadyne의 makeImage기능을 사용해서 펌웨어의 ID와 펌웨어의 아키텍처를 입력하면 자동으로 QEMU에 마운트 시켜준다. 이 작업이 끝나면 Firm AFL 디렉토리 밑에 Image_펌웨어ID로 디렉토리가 생성된다. 이후 Firm AFL의 기능을 마운트된 디렉토리에 호환시킨 이후 Image_펌웨어ID디렉토리 밑에 start.py를 실행하면 Firm AFL을 실행하면 퍼징이 시작된다.



[그림 5] 펌웨어 이미지 QEMU 마운트 과정

실행할 때 가장 잘 확인해야 할 것은 아키텍처이다. 아키텍처가 맞지 않을 경우 낮은 확률로 마운트까지는 진행이 되지만, 실행할 때 QEMU를 실행하는 부분에서 Error 메시지가 발생한다. 본 논문에서는 5장 실험결과 초반에 언급한 펌웨어들을 10시간씩 퍼징을 시도했다 하지만 Crash가 발견되지 않았다. [그림 6]은 Firm AFL을 실행했을 때 보여지는 화면이다.



[그림 6] Firm AFL 실행 과정

V.결론

본 논문에서는 임베디드 디바이스의 펌웨어의 취약점을 찾기 위해 Firm AFL 퍼징을 분석했다. 펌웨어 퍼징은 가상화 에뮬레이터에 마운트를 시켜놓고 퍼징을 진행하는 것이기 때문에 가상화 에뮬레이터를 바르게 설치하고 사용하는 방법이 익숙해진 후 진행하는 것이 좋다. 또한 펌웨어를 마운트하는 과정에서 아키텍처를

어떻게 찾고 아키텍처와 펌웨어를 가상화 에뮬레이터에 잘 마운트하는 것이 관건이다. 본 논문에서 실험한 펌웨어들은 10시간을 퍼징을 했는데도 Crash가 발견되지 않았다. 또한, 아키텍처를 지원하지 않은 펌웨어 들도 많이 존재해 아키텍처들을 많이 지원할 수 있도록 QEMU의 아키텍처 관련 연구가 필요해 보인다. 이후 AFL이 외에 다른 퍼저들을 QEMU와 같은 가상화 에뮬레이터에 연동시켜 펌웨어를 퍼징할 수 있도록 연구를 진행할 예정이다.

[참고문헌]

[1] Yaowen Zheng, Ali Davanian, Heng Yin, Chengyu Song, Hongsong Zhu, and Limin Sun "FIRM-AFL: High-Throughput Greybox Fuzzing of IoT Firmware via Augmented Process Emulation", August, 2019

[2] Dae-il Jang, "A Design of IoT Protocol Fuzzer", May, 2017

[3] Daming D. Chen, Manuel Egele, Maverick Woo, and David Brumley, "Towards Automated Dynamic Analysis for Linux-based Embedded Firmware", February, 2016

[4] Daniel Bartholomew, "QEMU: a Multihost, Multitarget Emulator", June, 2006

윈도우 운영체제 환경에서의 파일리스 공격 기법 조사

이경호*, 김기범*

*ETRI부설연구소

Survey on Fileless Attack Techniques in Windows OS

KyoungHo Lee*, Kibom Kim*

*The Affiliated Institute of ETRI

요 약

파일리스 공격(Fileless Attack)은 일반적으로 실행 파일이 저장되는 파일 시스템이 아닌 메모리에서만 악의적인 코드를 실행하고 데이터를 은닉하기 때문에 탐지가 매우 어려워 최근 악성코드에서 많이 사용되고 있다. 이에 따라 파일리스 공격에 대한 자세한 분석이 필요한 상황이다. 본 논문은 윈도우 운영체제 환경에서 사용 가능한 파일리스 공격 기법인 스레드 인젝션과 APC 인젝션, 스레드 하이재킹, 아톰 봄빙 기법, 리플렉티브 DLL 인젝션, 메모리 모듈, 프로세스 할로윙 기법에 대한 공격 절차를 상세히 분석한다.

Abstract

Fileless attack executes malicious code and hide data in memory, not in file system normally storing executables. Recently, in this reason, malwares use fileless attack techniques to hide them self. Thus, we need to analyze fileless attack techniques. In this paper, we analyze thread injection, APC injection, thread hijacking, atom bombing, reflective dll injection, memory module and process hollowing in detail, which are mostly used fileless attack techniques on Windows OS.

I. 서론

파일리스 공격은 윈도우 운영체제의 내장 기능을 사용하여 취약점 없이 합법적인 프로세스에 임의의 코드를 주입하고 실행한다. 코드가 파일 시스템을 거치지 않고, 메모리에서 바로 실행되기 때문에 안티 바이러스 솔루션의 탐지 표면을 우회한다.

Ponemon Institute 보고서에 따르면 2017년 공격 중 77%가 파일리스 공격이었고, 파일리스 공격은 전통적인 파일 기반 공격보다 10배나 높은 성공률을 보였다[1]. 파일리스 악성코드의 비율은 2016년 초 3%에서 2016년 11월 13%로 증가한 만큼 향후 지속적으로 늘어날 것으로 보인다. 따라서 파일리스 공격에 대한 자세한 분석이 필요한 상황이다.

본 논문에서는 파일리스 공격의 공격 벡터인 코드 인젝션 기술들을 설명한다. 스레드 인젝션과 APC 인젝션, 스레드 하이재킹, 아톰 봄빙 기법과 리플렉티브 DLL 인젝션, 메모리 모듈 기법, 프로세스 할로윙 기법의 동작 방식을 분석한다. 마지막으로 결론을 맺고 향후 연구에 대해 의논한다.

II.파일리스 공격

2.1 스레드 인젝션

스레드 인젝션(Thread Injection)은 가장 전

통적인 코드 인젝션 기술이다[2]. 이 기술은 우선 VirtualAllocEx 함수를 사용하여 대상 프로세스 읽기, 쓰기, 실행 가능한 메모리를 할당한다. 다음, WriteProcessMemory 함수를 사용하여 대상 프로세스의 할당한 메모리에 실행하고자 하는 코드를 복사한다. 마지막으로, 코드를 시작점으로 CreateRemoteThread 함수를 호출하여 원격 스레드를 실행한다. 이제 대상 프로세스에서 주입한 코드가 실행된다.

2.2 APC 인젝션

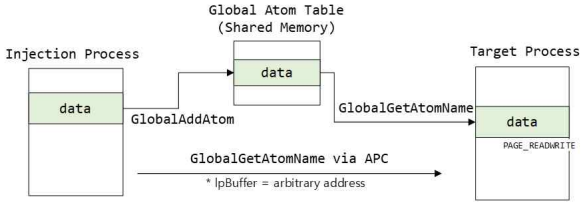
APC(Asynchronous Procedure Call) 인젝션은 임의의 코드를 대상 메모리에 할당한 후, 이 코드의 주소를 QueueUserApc 함수로 APC 큐에 등록한다. 이후 스레드는 시스템에 의해 비동기적으로 임의의 코드를 실행하게 된다. 스레드 인젝션이 CreateRemoteThread 함수로 코드를 직접 실행한다면, APC 인젝션은 APC 큐에 임의의 코드를 등록하여 실행되길 기다린다.

2018년에 새로운 코드 인젝션 기술로 소개된 얼리 버드 인젝션(Early Bird Injection)은 이 기술을 사용한다[3].

2.3 스레드 하이재킹

스레드 하이재킹(Thread hijacking)은 기존 프로세스의 실행 중인 스레드 중 하나를 정지시키고, 스레드 컨텍스트를 변경하여 임의 코드 주소로 실행 흐름을 바꾼다.

이 기술은 SuspendThread 함수로 정지시킨 스레드를 대상으로 GetThreadContext 함수를 호출하여 컨텍스트를 얻고, 스레드의 현재 실행 위치인 EIP 필드를 주입한 코드의 주소로 변경한다. 이후, SetThreadContext 함수를 호출하여 변경된 컨텍스트를 대상 스레드에 적용한다. 마지막으로 ResumeThread 함수를 호출하여 정지한 스레드를 재개한다. 이제 공격자는 주입한 임의 코드를 대상 프로세스에서 실행할 수 있다.



[그림 1] 아톰 테이블을 이용한 임의의 주소에 데이터 쓰기

2.4 아톰 봄빙

2016년에 Ensilo 연구팀에서 발견한 아톰 봄빙(Atom Bombing)은 아톰 테이블과 APC를 익스플로잇하는 코드 인젝션 기술이다[4][5]. 2017년에 Dridex 멀웨어가 사용하여 이 기술이 실제 악성코드에서 사용될 수 있음을 보였다[6].

이 기술은 그림 1과 같이 WriteProcessMemory 함수 대신 아톰 테이블에 데이터를 쓰는 GlobalAddAtom 함수와 아톰 테이블로부터 데이터를 읽는 GlobalGetAtomName 함수쌍을 사용하여 대상 프로세스의 임의의 메모리에 코드를 주입한다. 3개의 인자가 필요한 GlobalGetAtomName 함수를 호출하기 위해 NtQueueApcThread 함수를 사용한다. 이후 스레드 하이재킹 기술을 사용하여 코드를 실행한다.

2.5 리플렉티브 DLL 인젝션

리플렉티브 DLL 인젝션은 LoadLibaray 함수 호출 없이 대상 프로세스 스스로 인젝션 DLL을 로드하게 하는 기술이다[7].

이 기술은 우선 DLL 이미지를 VirtualAllocEx 함수와 WriteProcessMemory 함수를 사용하여 대상 프로세스 메모리에 기록한다. 다음, CreateRemoteThread 함수로 원격 스레드를 생성하여 대상 프로세스에서 DLL 로더 함수를 실행한다. 실행된 로더는 LoadLibrary 함수와 유사하게 DLL을 로드한다. 마지막으로 CreateRemoteThread 함수를 사용하여 DllMain 함수를 호출한다.

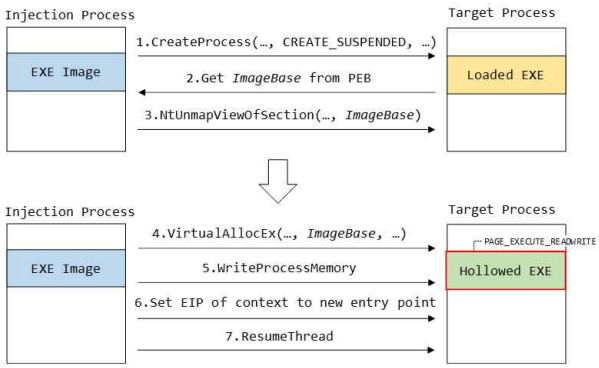
2.6 메모리 모듈

매뉴얼 맵(Manual Map) 기술로도 알려진 메모리 모듈(Memory Module)은 인젝션 DLL을 LoadLibrary 함수 호출 없이 인젝션 프로세스에서 직접 실행 가능하도록 로드한다[8][9][10]. 리플렉티브 DLL 인젝션과의 차이점은 대상 DLL을 메모리로 매핑하는 주체가 인젝션 프로세스이다.

2.7 프로세스 할로윙

프로세스 할로윙(Process Hollowing)은 실행되는 프로세스를 대체하는 공격 기술로 2007년 블랙햇에서 악성코드가 사용하는 은닉 기술들에 대해 다루면서 널리 알려졌다[11]. Nebbett's Shuttle 또는 PE Image switching, RunPE로도 불린다[12][13].

이 기술은 그림 2와 같이 우선 CreateProcess 함수로 프로세스를 생성 할 때 정지 상태로 생성하고, NtUnmapViewOfSection 함수를 호출하여 원본 프로세스 파일과 매핑된 메모리들을 해제한다. 다음, 리플렉티브 DLL 인젝션과 유사한 방식으로 대체할 EXE 파일을 실행 가능한 형태로 직접 로드한다. 마지막으로 SetThreadContext 함수를 사용하여 변경된 PE 이미지의 엔트리 포인트로 EIP를 변하고, ResumeThread 함수를 호출하여 메인 스레드가 엔트리 포인트부터 실행되게 한다.



[그림 2] 프로세스 할로윙 절차

III. 결론

본 논문에서는 최근 악성코드에서 많이 사용되는 파일리스 공격의 주요 공격 방식인 스레드 인젝션과 APC 인젝션, 스레드 하이재킹, 아톰 봄빙 기법과 리플렉티브 DLL 인젝션, 메모리 모듈 기법, 프로세스 할로윙 기법의 동작 방식에 대해 기술하였다.

향후 공격 측면에서 파일리스 공격의 지속성 유지 방법에 대한 조사를 진행할 예정이며, 탐지 측면에서 API 콜 체인 기반 파일리스 공격 탐지 기법, 메모리 기반 탐지 기법 등을 조사할 예정이다.

[참고문헌]

[1] Ponemon Institute, "The State of Endpoint Security Risk Report", 2018

[2] Ashkan Hosseini, "Ten Process Injection Techniques: A Technical Survey Of Common And Trending Process Injection Techniques", JULY 18 2017, https://www.endgame.com/blog/technical-blog/ten-process-injection-techniques-technical-survey-common-and-trending-process

[3] 알약, "해커들, 탐지를 피하기 위해 새로운 코드 인젝션 기술 사용하는 것으로 나타나", Apr 16 2018, https://blog.alyac.co.kr/1628

[4] Andrew Spangler, "ATOMBOMBING: YET ANOTHER ATTACK VECTOR TO DEFEND", Dec 6 2016, https://www.nuix.com/blog/atombombing-yet-another-attack-vector-defend

[5] Tal Liberman, "ATOMBOMBING: BRAND NEW CODE INJECTION FOR WINDOWS", Oct 27 2016, https://blog.ensilo.com/atombombing-brand-new-code-injection-for-windows

[6] EndGame, "Dropping Atom Bombs: Detecting DridexV4 in the Wild", May 2017

[7] Stephen Fewer, "Reflective DLL Injection", Oct 31 2008, https://pdfs.semanticscholar.org/d30c/d54b10948fef0046f61fd05cd18eed4cc561.pdf

[8] Joe Desimone, "Hunting In Memory", JUNE 13 2017, https://www.endgame.com/blog/technical-blog/hunting-memory

[9] Joachim Bauch, "Loading a DLL from memory", Apr 7 2010, https://www.joachim-bauch.de/tutorials/loading-a-dll-from-memory/

[10] DarthTon, "Blackbone - Windows memory hacking library", 2013, https://github.com/DarthTon/Blackbone

[11] Nick Harbour, “Stealth Secrests of the Malware”, 2007, Blackhat USA, https://www.blackhat.com/presentations/bh-usa-07/Harbour/Presentation/bh-usa-07-harbour.pdf

[12] Gary Nebbett, “Launch an exec / proc from memory”, Dec 14 2000, https://groups.google.com/forum/#!msg/comp.os.ms-windows.programmer.win32/Md3GKPc279A/Ax3bYgXhpD8J

[13] Andrea Fortuna, “RunPE: a practical example of Process Hollowing technique”, Nov 22 2017, https://www.andreafortuna.org/2017/11/22/runpe-a-practical-example-of-process-hollowing-technique/

국내 APT 및 기업 보안 사고에 대한 고찰

이창엽, 이진우, 이정훈, 노무승, 박현상

창원경일고등학교 3학년, 대구대학교 1학년, 인제대학교 1학년,

창원중앙고등학교 2학년, 무안 남악중학교 3학년

Anti-root (안티루트)

A Consideration on Domestic APT and Corporate Security Accidents

Lee Changyeop, Lee JinWoo, Lee Junghun

Rho Moo-Seung, Park HyunSang

Changwon Gyeongil High School third grader, Daegu University first grade,

Inje University first grade, Changwon Jungang High School second grader,

Muan Namak Middle School third grader

요 약

최근, 국내에서 다양한 APT 공격이 진행되고 있고, 개인정보 유출 및 기업에서 보안 사고가 발생하고 있다. 이러한 일들을 들어, 어떠한 문제점으로 인해 이러한 문제가 발생하는지, 그리고 이를 어떻게 해결해 나갈 수 있을지에 관심이 생기게 되어, 올해 발생한 보안 사고 및 APT 공격에 관해 조사해보고, 향후 어떤 방법으로 피해를 줄여나갈 수 있을지 연구해보고자 한다.

I. 서론

[1]APT란, 지능형 지속 위협이라고 하며, 공격자가 특정 대상 [기업 혹은 국가, 특정 단체 등]을 목표로 다양한 해킹 기법/기술을 통해, 은밀하고 지속적으로 공격하는 행위를 의미한다. 이러한 APT 공격은 기존의 공격과 달리, 지속적이며, 치밀하고, 공격 대상의 조직을 이해하고, 조직화된 계획을 통해 공격이 진행된다. APT 공격의 사이클은 사전적으로 공격을 진행하고자 하는 특정 대상의 정보를 수집[관계자 정보 및 이메일, 재직자의 정보 등]한 뒤, 이를 활용하여, 침입을 진행하고 그 뒤, C&C [2]서버 통신을 통해 통신망을 구성하고, 공격 대상 내에서 확산을 통해, 점차 공격 대상을 장악해나가면서, 원하는 데이터에 접근 및 지속적으로 유출 혹은 특정 대상이 제공하는 서비스 혹은 업무에 지장을 주는 형식으로 공격 사이클이 이루어진다. APT와 관련된 국내 주요 사건으로는 3.20 사이버 테러 및 원전 기밀 유출, 은행 전산망 마비와 같은 주요한 사건들이 존재한다. 최근에 들어, 개인정보 유출 사고 및 기업에서 정보보안과 관련된 사건/사고가 자주 발생하고 있으며, 또한 위에서 소개한 APT와 관련하여, 특정 단체, 조직, 국가가 행하는 APT 공격으로 인해 심각한 피해가 발생하고 있다는 점을 느끼게 되어, 국내 기업 및 APT 단체를 통해 발생한 주요 사건/사고를 정리하고, 공통점인 부분을 찾아내어, 이와 관련된 피해를 줄일 수 있는 방안을 찾아보고자 하여, 해당 연구를 시작하게 되었다.

1) Advanced Persistent Threat

2) Command&Control Server

II. 본론

Ⅱ-1 사례 분석

먼저, 공통점을 찾기 위해 국내에서 발생한 APT, 기업 및 개인정보 사고 사례를 수집, 선정, 분석을 진행했다. 분석한 사건들은 표 및 타임라인을 통해 표시하였으며, 그 중 일부 사례를 소개하고자 한다.

[Ⅱ-1-1] APT 사고 사례

[Ⅱ-1-1-1] 오퍼레이션 익스트림 잡 | 라자루스

2019년 1월 30일 'Job Description.doc'라는 이름을 가진 악성 파일이 확인되었으며, 해당 악성 파일은 특

정 웹 사이트[IP 카메라/ CCTV 등 판매 사이트]를 통해 악성코드가 유포 되었으며, 국내외 주요 침해사고 대응에서 발견됐던 doc [3]매크로와 거의 같은 코드 기법이 재활용됐으며, 공격 벡터 상 DOC 파일명도 기존과 동일하다 판단되어, 라자루스가 해당 공격의 배후라고 예상되었다. MS Word 파일의 내부 OLE 코드 구조 안에, 매크로 코드가 포함되어있으며, 매크로 VBA [4]내부 코드를 확인해볼 경우, 난독화된 스트링 코드들을 확인할 수 있으며, 복호화 명령어를 통해 추가 악성코드를 생성하고 실행을 하게 됨. 또한, 공격자는 문서 파일의 버전이 낮아, 매크로를 실행해야만 정상적인 문서가 보인다는 가짜 메시지를 포함시켜, 이용자가 매크로를 실행하도록 유도함. 이를 확인한 사용자가 콘텐츠 사용을 통해 매크로를 실행하게 될 경우, 악성 파일이 생성되고, 몰래 실행되게 됨. 또한, 매크로 실행 이후, 열람한 word 문서 내에는 'Job Description.doc' 문서 내용을 보내주어, 파일이 정상처럼 보이도록 유저를 속임. 추가로 생성된 악성 파일은 특정 웹사이트와 통신을 시도하고, 공격자의 추가 명령을 기다린다.

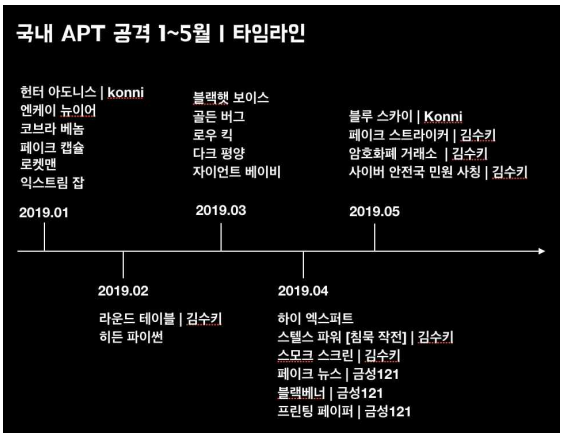


그림 1 | 국내 APT 공격 1-5월 사례 일부



그림 2 | 국내 APT 공격 6-9월 사례 일부

3) doc, word | MS 사의 word 파일 확장자
4) Visual Basic for Application

[Ⅱ-1-2] 기업, 개인정보 사고 사례

[Ⅱ-1-2-1] 티티호스팅 및 가비아

2014년 티티호스팅을 인수 후 2018년 12월 31일까지 정상적으로 운영되는 것으로 보였으나 개인정보 유출사고가 발생하였으며 2019년 04월 29일에 가비아가 티티호스팅 및 가비아에서 개인정보 유출사고가 발생한 사실을 인지하고 이를 공식적으로 발표하였음. 이에 2019년 05월 15일에서 16일까지 이틀 간 개인정보가 유출된 것으로 의심되는 고객에게는 이메일을 통해 유출 사실을 알렸으며 침해 사실 인지 후 지체없이 한국인터넷진흥원(KISA)에 신고하였으며 내부 기술 및 관리 보안 조치를 추가 강화하고 있다고 하였음. 이에 내부적 원인을 통해 개인정보 유출사고가 발생하였음을 알 수 있다.

[Ⅱ-1-2-2] 메가스터디

2019년 6월 12일 메가 스터디 측은 개인정보 유출사고를 인지하고 한국인터넷진흥원(KISA)에 신고하였으며 사고 당시 유출된 개인정보로는 ID, 이름, 연락처, 생년월일, 이메일, 성별, 암호화된 비밀번호가 유출되었으며 이번 개인 정보 유출 사고가 메가스터디의 첫 사고 사례가 아닌 2017년 7월 회원 이름, 생년월일이 포함된 개인정보 123만3859건이 유출되었고 과징금 2억1900만원, 과태료 1000만원의 징계를 받았던 적이 있으며 2차례 다 내부적 원인을 통해 개인정보 유출사고가 발생한 것으로 보아 재발 방지를 위한 대응을 강화하고있다고는 하지만 고객들의 불만은 쉽사리 사그라지지 않는 모습을 확인하였다.

사건	유출인원 유출 건수	시기
야놀자펜션앱	7만명	2019년 03월
천호엔케어	15만명	2019년 05월
가비아	7만 7천건	2019년 05월
메가스터디	123만 3859건	2019년 06월
땡글닷컴	8만 3천건 (해커 주장)	2019년 08월
홈플러스	5만명	2019년 09월
탈잉	38만명	2019년 09월

표 1 | 개인정보유출 사고 사례 (일부)

Ⅱ-2 사례 분석을 통한 공통점

[Ⅱ-2-1] APT 사고 공통점

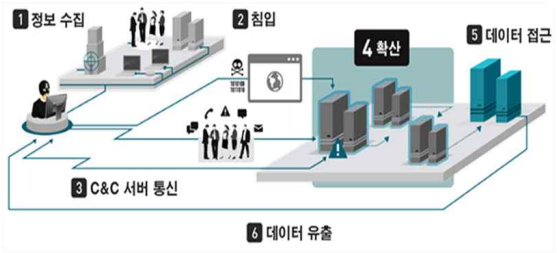


그림 3 | APT 진행 절차

대부분의 APT 사례들은 위와 같은 절차를 밟고 있었다. 실제로, 악성 메일 혹은 사회공학 기법을 통해, 특정 조직의 관계자가 해당 파일을 받아, 실행하게 될 경우, C&C 서버와 통신을 통해, 추가로 필요한 악성코드를 설치하거나, 통신을 통해 데이터를 유출하는 경우가 대다수였다. 또한 사례를 분석하면서 파악할 수 있는 것은, 정보수집과 같은 과정에서, 기업 내부자 혹은 관계자에게 직접 악성 이메일을 발송하거나, 업무 혹은 관심사와 관련된 사이트를 장악하여, 정보수집 및 침입 과정을 대부분 차지하고 있었으며, 1-day[5] 취약점을 활용하여, 공격에 활용하는 경우도 종종 발생하였으며, 대부분의 공격자가 문서형 악성코드를 통해, 사용자가 문서를 열람하여 오피스 계열의 경우, 매크로를 실행하거나, 한글 파일의 경우, EPS[6][포스트 스크립트]가 실행되면서, 악성코드에 감염되는 경우가 대다수였다. 특히, 문서형 악성코드의 경우, 암호를 걸어 문서를 소프트웨어를 통해 암호화를 진행하여, 제공하게 될 경우, AV 엔진이 이를 정상적으로 탐지할 수 없고, 분석에 어려움이 있기에 이러한 문서형 악성코드를 선호하는 것으로 보인다. 그리고, 특히 북한 조직 및 APT 조직의 경우, 암호화 화폐와 관련해서 많은 관심을 가지고 있는 것으로 보인다. 실제로, 거래소에서 보낸 메일로 속이거나, 거래소를 대상으로 진행한 공격을 보면, 암호화 화폐의 익명성, 자신의 신분이 밝혀질 위험이 낮기에, 이러한 암호화 화폐와 관련된 공격을 진행하는 것으로 보인다.

[Ⅱ-2-2] 기업, 개인정보 사고 사례 공통점

기업 및 개인정보 사고 관련 사례의 공통점으로는 주로 사람의 실수 혹은 평소에 정보보안에 대한 의식이 부족하여, 발생한 사례가 다소 존재하였으며, 또한 공격 기법 혹은 해커가 침입한 경로 및 사고로 이어지게 된 이유가 각기 달라, 공통점을 정리하는 것에는 다소 어려움이 존재하였다. 하지만 대다수의 보안 사고가 발생한 기업 및 단체의 경우, 앞에서 밝혔듯이 주로 사람의 실수로 인해, 해커가 외부에서 침입에 성공한 경우가 대다수이거나, 혹은 기본적인 보안 수칙을 지키지 않거나, 정기적인 보안 점검 및 수칙의 미비로 사고가 발생한 경우가 많았다. 실제로 하나투어의 경우, DB 외부직원이 하나투어 보안망 접속이 가능한 접근 통제 솔루션의 아이디, 비밀번호를 담은 스티키 노트를 저장한 사례가 존재하며, 가비아의 경우, 내부 직원 계정이 유출되어, 개인 정보 유출로 이어진 사례가 존재한다. 특히 개인정보 유출 및 보안사고와 관련된 처벌이 피해보다 상당히 가볍기 때문에, 사고가 발생해도 이를 가볍게 넘기는 경우가 대다수로 보인다. 이에 관해 개인정보 유출 및 보안사고와 관련된 처벌을 강화하고, 관련 법을 조정하여, 기업이나 단체에서 이를 가볍게 보지 않도록 할 필요가 있으며, 피해자가 그에 합당한 보상을 받을 수 있도록 도와야 한다고 생각한다. 또한, ISMS-P[7]와 같은 인증 및 보안에 대한 꾸준한 관심 및 투자가 필요한 시점이라 생각된다.

Ⅱ-3 대책 및 해결 방안 제시

[Ⅱ-3-1] 솔루션 개선 및 정보보안에 대한 투자

APT와 관련된 공격은 증가하고, 관련된 단체가 더욱 활발하게 활동하고 있으나, 지란지교시큐리티의 설문조사 결과에 따르면, 응답자의 60%가 APT 경험이 존재하나, 대응 솔루션 사용은 17%에 불가했다. 또한, APT 솔루션을 도입할 경우, 탐지 정확성과 경제적인 도입/운영 비용을 가장 많이 고려한다고 하니. 탐지 정확성을 높이기 위해, 기존 탐지 방법을 개선하거나, 비용 문제와 관련하여, 조치할 필요가 있어 보인다. 단, APT 솔루션을 개선하는 것에서 그치지 않고, 기업 및 단체, 기관에서 이러한 솔루션에 관심을 가지고, 정보보안에 투자를 점차 늘려나갈 필요가 있다고 생각된다. 또한, ISMS-P와 같은 제도를 도입하는 비용 및 시간 투자를 부정적으로 보고 차라리 벌금을 내는 것이 오히려 낫다고 생각하는 경우가 존재한다. 이러한 인식을 개선하고, 보안에 대한 투자를 늘릴 수 있도록 지원금을 제공하는 방식이나 다양한 방법으로 정보보안 강화를 독려할 필요가 있어 보인다.

[Ⅱ-3-2] 기업 내 보안 교육 강화

대부분의 보안 사고는 사람으로 인해 발생하며, 장비 문제 혹은 0-day[8]와 같은 대처가 불가능한 공격은

5) 1-day | 취약점이 패치가 발표되었으나, 적용하지 않은 상태를 말함.
6) EPS | Encapsulated PostScript
7) 정보보호 및 개인정보보호 관리체계 인증
8) 0-day | 취약점이 발견되지 않은 경우를 말함

극소수에 불가하다. 이러한 시점에서 기업 및 단체, 기관에서는 사내 보안 교육 및 정기적인 보안 점검 및 보안 수칙을 사내에 전파해, 보안 사고를 예방할 필요가 있다. 또한, 보안 전공자가 아닌 비전공자의 입장에서는 단순히 이메일 혹은 사이트로 접근하는 행위를 통해, 해킹의 위험에 노출되는 것에 대해 그리고 왜 그런 보안사고가 발생하는 것인지 이해하기가 어려울 것이다. 이와 관련해서 시뮬레이션 혹은 이를 체험할 수 있는 플랫폼을 제작하여, 비전공자가 실수로 인해, 보안사고가 발생할 수 있다는 것을 이해하고, 경각심을 가질 수 있는 방향으로 연구를 진행될 필요가 있을 것 같다.

[Ⅱ-3-3] 관련 기관의 노력 및 협력

개인정보 유출 및 보안사고와 관련된 처벌을 강화하여, 보안에 투자할 비용보다, 벌금을 내는 것이 더 이득을 볼 수 있다는 관점을 바꿀 필요가 있을 것 같으며, KISA 및 국정원과 같은 국가기관에서 정기적으로 기업 및 단체, 개인을 대상으로 정보보안 교육 그리고, 정보보안과 친숙해질 다양한 기회를 제공해야 한다고 생각한다. 특히 개인정보 유출 및 보안사고와 관련된 처벌이 피해보다 상당히 가볍기 때문에, 사고가 발생해도 이를 가볍게 넘기는 경우가 대다수로 보인다. 이에 관해 개인정보 유출 및 보안사고와 관련된 처벌을 강화하고, 관련 법을 조정하여, 기업이나 단체에서 이를 가볍게 보지 않도록 할 필요가 있으며, 피해자가 그에 합당한 보상을 받을 수 있도록 도와야 한다고 생각한다. 또한, ISMS-P와 같은 인증 및 보안에 대한 꾸준한 관심 및 투자가 필요한 시점이라 생각된다.

III. 결론

본 논문에서는 결론적으로 APT와 관련된 솔루션을 도입 및 기존 솔루션에 변화를 줄 필요가 있다고 생각한다. 또한 변화에서 멈추지 말고, 기업, 단체, 기관이 정보보안에 대한 관심을 늘릴 필요가 있으며, 보안에 더 투자해야할 이유, 그리고 보안을 점차 강화할 수 있도록 정부 각 기관처에서 다양한 캠페인 및 도움을 줄 필요가 있다. 또한, 기업 내 보안 교육과 관련하여, 단순히 강사가 말로 전하는 교육이 아닌, 시각화 및 시뮬레이션을 통해, 직접 자신이 실수로 열람한 메일 혹은 접근한 사이트로 인해, 어떤 피해가 발생하게 되는지 경험할 수 있는 솔루션이 연구될 필요가 있다고 생각한다. 마지막으로는 생각보다, 여전히 기본적인 보안 수칙 혹은 점검 부족으로 발생하는 보안 사고가 많이 존재한다. 이러한 사고를 예방하기 위해 기업 및 단체, 기관에서는 정기적으로 보안 점검을 수행하고, OWASP[9]와 같은 알려진 공격 기법에 대해 정기적으로 검사하고, 모의해킹 업체를 통해 자사의 보안상태를 점검하는 등, 다양한 노력이 필요할 것으로 보인다.

[참고문헌]

[1] 'IT World 용어풀이 APT(Advanced Persistent Threat)', IT World, 2015.09.04.,

http://www.itworld.co.kr/news/95741

[2] '중국의 APT 단체들, 서로 힘 합치고 있나 보안뉴스', 2019.07.08.

https://m.boannews.com/html/detail.html?mtype=1&idx=81212

[3] 'APT 성행해도 대응 솔루션 사용 기업은 17%에 불과...APT 대응에 미흡', 보안뉴스, 2019.03.21.

https://m.boannews.com/html/detail.html?mtype=1&idx=78033

[4] "가비아 이어 자회사 '티티호스팅'도 개인정보 유출", 보안뉴스, 2019.05.21.

https://www.boannews.com/media/view.asp?idx=79696&page=12&kind=1

[5] "'메가스터디', 깊어지는 성인대상 교육사업 적자에 개인정보유출 논란까지", 뉴스워커 2019.09.06.

www.newsworker.co.kr/news/articleView.html?idxno=40551

[6]APT 공격과 대응방안, 이글루시큐리티, 2016.08.02

http://www.igloosec.co.kr/BLOG_APT%20%EA%B3%B5%EA%B2%A9%EA%B3%BC%20%EB%8C%80%EC%9D%91%EB%B0%A9%EC%95%88?searchItem=&searchWord=&bbsCateId=1&gotoPage=1

[7] 라자루스 APT 조직, 오퍼레이션 익스트림잡으로 공격 수행, 알약, 2019.01.31

https://blog.alyac.co.kr/2105

9) Open Web Application Security Project

네트워크 퍼저를 이용한 FTP 서버의 취약점 분석

최인호, 윤정준, 윤주범

세종대학교 정보보호학과

polarisinho@naver.com, lovlove7@naver.com, jbyun@sejong.ac.kr

FTP Server Vulnerability Analysis by Network Fuzzer

Inho Choi, Jeongjun Yun, Joobeom Yun

Department of Computer and Information Security, Sejong University.

요 약

일반적으로 편의성과 효율성 때문에 FTP 서버를 많이 이용하고 있다. 이 논문에서는 FTP 프로토콜과 퍼징이라는 기술을 설명하고, 네트워크 퍼저인 Simple FTP Fuzzer(FTP전용)와 BooFuzz(Sulley기반)을 선정하여, 우리에게 친근한 FTP 서버를 퍼징한다. 결과로, FTP 서버에 Crash를 발생시키고 취약점 분석을 통하여, 정보보호 분야에서 퍼징 기술의 중요성에 대하여 논한다.

I. 개요

일반적으로 사람들이 생각하는 '인터넷'이라면 웹 브라우저를 이용한 화려한 화면의 WWW 방식을 말한다. HTTP 프로토콜을 통해 접속되는 WWW는 일반 문자를 비롯하여 사진, 음악/동영상 콘텐츠까지 간편하게 사용할 수 있다는 장점이 있지만, 많은 양의 파일을 일괄적으로 주고받기에는 결정적으로, 속도도 느리고 파일 제어도 번거롭기 때문에 대량의 파일을 네트워크를 통해 주고 받을 때는 파일 전송 전용 서비스인 FTP를 사용하는 것이 훨씬 유리하다. FTP는 인터넷을 통한 파일 송수신만을 위해 고안된 서비스(또는 프로토콜)이기 때문에 동작 방식이 대단히 단순하고 직관적이다. 그런 만큼 사용법도 간단하다. 무엇보다 WWW 방식보다 파일을 빠른 속도로 한꺼번에 주고받을 수 있다는 것이 FTP의 가장 큰 장점이다. 또한, 기본적으로 명령어 기반의 통신 서비스이기 때문에, 사용이 복잡하고 어려울 수 있어 많은 사람들이 별도의 FTP 클라이언트 프로그램을 사용한다.

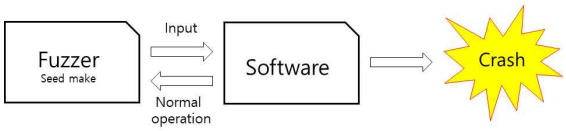
FTP는 인터넷을 통하여 어떤 한 컴퓨터에서 다른 컴퓨터로 파일을 송수신할 수 있도록 지원하는 프로토콜이다. FTP를 사용하여 파일을 송수신할 때에는 원격 호스트 컴퓨터를 이용할 수 있는 사용자 ID와 password 가 있어야 원격지 호스트에 접속하여 파일을 송수신할 수 있는데, 여기에서 사용하는 FTP 서버에 취약점이 존재하면 취약한 부분의 공격으로 인해 FTP 서버 프로그램이 오작동을 일으키거나 허가받지 않은 원격지 내의 중요한 정보를 얻을 수 있다. 본 논문은 퍼저의 성능이 상이 함을 고려하여, 네트워크 퍼저 두 가지 Simple FTP Fuzzer(FTP전용)와 Boofuzz(Sulley기반)를 선정하였고, 이를 이용하여 FTP 서버를 퍼징한다. 결과로 FTP 서버에 Crash을 발생시키고 취약점을 분석함으로 퍼징 기술의 중요성을 논해 본다.

본 논문의 2장에서는 관련 연구인 퍼징과 3장에서는 퍼저와 FTP서버의 설치, 4장은 퍼저로 FTP 서버를 퍼징하고 결과를 서술한다. 그리고 마지막 5장에서 결론으로 본 논문이 구성되어 있다.

II. 관련 연구

2.1 퍼징

퍼징은 특정 프로그램을 타켓으로 잡고 무작위 값을 대입하여 충돌 또는 메모리 누수를 발생시켜 취약점을 찾는 소프트웨어 기술이고 구조화된 입력을 취하는 프로그램을 테스트하는 기술이다. [그림 1]은 퍼징의 기본 동작을 그림으로 나타낸다.



[그림 1] 퍼징의 기본 동작

퍼징은 프로그램 구조를 얼마나 인지하는지에 따라 화이트 박스와, 블랙 박스 퍼징으로 나뉘고, 화이트 박스와 블랙 박스를 섞어 놓은 그레이 박스 퍼징 3가지로 나뉜다. 또한 퍼저는 입력이 처음부터 생성하는지, 아니면 기존의 입력값을 수정하는 식으로 입력값이 만들어지고 있다.

퍼징의 종류는 화이트 박스 퍼저, 심볼릭 익스큐션, 임베디드 디바이스 퍼저, 네트워크 퍼저 등으로 나눌 수 있는데, 본 논문에서 제시하는 퍼저는 네트워크 퍼저이다.

2.2 FTP 서버 퍼징

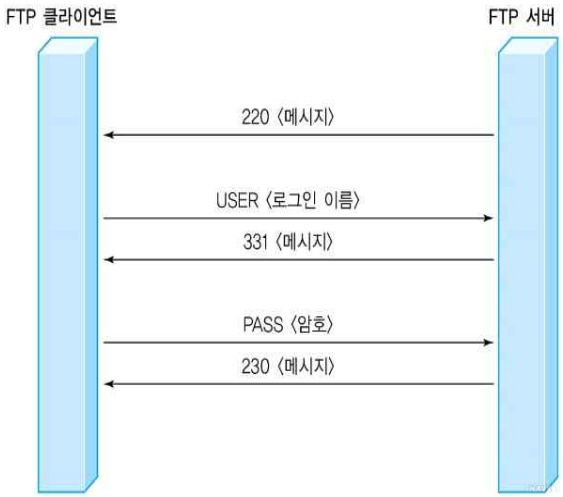
FTP서버를 대상으로 하는 퍼징 기술은 네트워크 퍼저이다. 네트워크 퍼징 같은 경우는 우리가 흔히 사용하는 FTP 프로토콜 등을 대상으로 무작위 값을 보내 퍼징하는 방식이다. 네트워크 퍼징 같은 경우 네트워크에서 주고받는 프로토콜을 상대로 퍼징을 진행하기 때문에 프로토콜 규격에 맞춰 다양한 값을 변형시켜가며 퍼징해야 한다. 네트워크 퍼징의 종류로는 BooFuzz, Peach, TLS - Attacker, DELTA, T-Fuzz, Sulley, Simple FTP Fuzzer 등이 있다.

2.3 FTP 로그인

앞에서 설명했듯이. FTP 클라이언트와 서버는 FTP 명령의 처리를 위해 제어 채널을 사용하며, 클라이언트가 전송한 명령에 대해 서버가 응답하는 방식으로 진행된다. FTP 서버에 원하는 작업을 요청하기 위해서 FTP 클라이언트는 간단한 FTP 명령을 이용해 작업한다. FTP 명령과 응답은 모두 제어 채널을 통해 전달된다.

다른 인터넷 응용 프로그램에서 사용하는 프로토콜처럼 FTP 응답 내용은 세 자리의 응답 코드와 설명문으로 구성된다. 응답 코드의 첫 번째 숫자가 2인 경우는 성공을, 5인 경우는 실패를 의미한다.

FTP 로그인은 FTP 클라이언트가 FTP 서버와의 제어 채널 연결을 설정한 후에 로그인 이름과 암호를 전송하여 FTP 세션 연결을 설정하는 것이다. [그림 2]처럼 클라이언트는 FTP 서버의 21번 포트와 TCP 연결 설정을 시도한다. 서버로부터 FTP 서비스의 이용이 가능함을 의미하는 응답 코드 220을 수신하면 연결이 성공한 경우다.



[그림 2] FTP 로그인 절차

[그림 2]를 이후 USER 명령을 사용해 로그인 이름을 제공하고, PASS 명령으로 로그인 암호를 전송한다. 서버에서는 해당 사용자에 대한 인증 절차를 통해 로그인 가능 여부를 판단한다. 부적절한 사용자는 오류로 처리되지만, 정상적인 경우는 그림처럼 응답 코드 230을 전송하여 로그인 절차가 성공적으로 이루어졌음을 회신한다. 일단 성공적으로 로그인하면 FTP 세션이 설정되어 다른 FTP 명령을 사용해 파일 송수신 기능 등을 수행할 수 있다.

III. 퍼저 및 타켓 FTP서버 설치

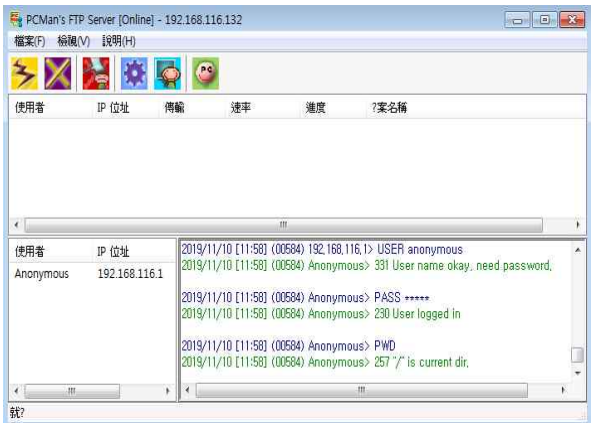
퍼저는 Simple FTP Fuzzer(FTP전용)와 BooFuzz(Sulley기반)이다. 두 퍼저는 Python으로 작성된 소프트웨어로, FTP 서버에서 사용자 이름과 비밀번호 필드를 퍼징하여, 취약점을 찾아낼 수 있는 퍼징 도구이다. 모두 오픈소스이

고 Github에서 다운받아서 설치 할 수 있다.

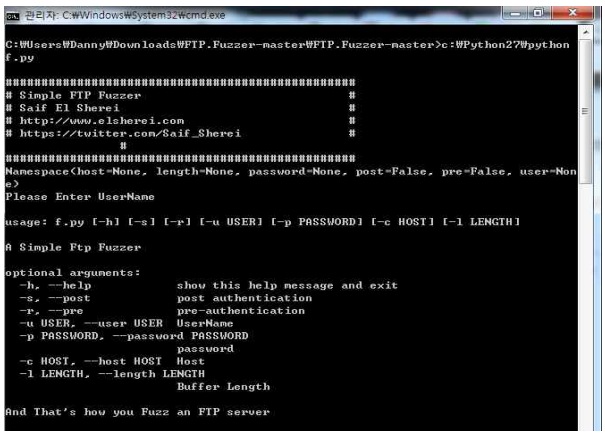
FTP는 PCMANFTP Server Version 2.0이고 별도의 설치없이, 포터블로 동작 가능하다.

IV.실험 및 결과

실험에 사용된 FTP 서버 프로그램은 위에서 설명한 PCman이다. Simple FTP Fuzzer를 실행하기에 앞서 [그림 3]은 타켓인 FTP 서버가 동작하는 그림이다.

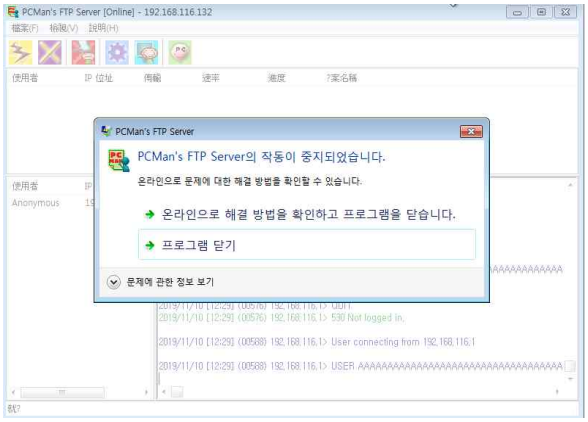


[그림 3] 원격지 호스트 FTP 서버



[그림 4] Simple FTP Fuzzer 구동

[그림4]의 Simple FTP Fuzzer를 구동하게 되면 몇 가지 간단한 명령어와 함께 어떠한 FTP 서버를 퍼징 할 것이냐 라는 문구가 등장한다. 이 논문에서는 선인증(pre-authentication) 방식을 이용하여 퍼징한다. 여기에서 호스트 IP와 user 그리고 password를 입력하고 length를 2000을 주게 되면 [그림 5]과 같이 호스트 FTP서버 프로그램이 작동을 멈추는 현상이 발생한다.



[그림 5] FTP 서버 오작동

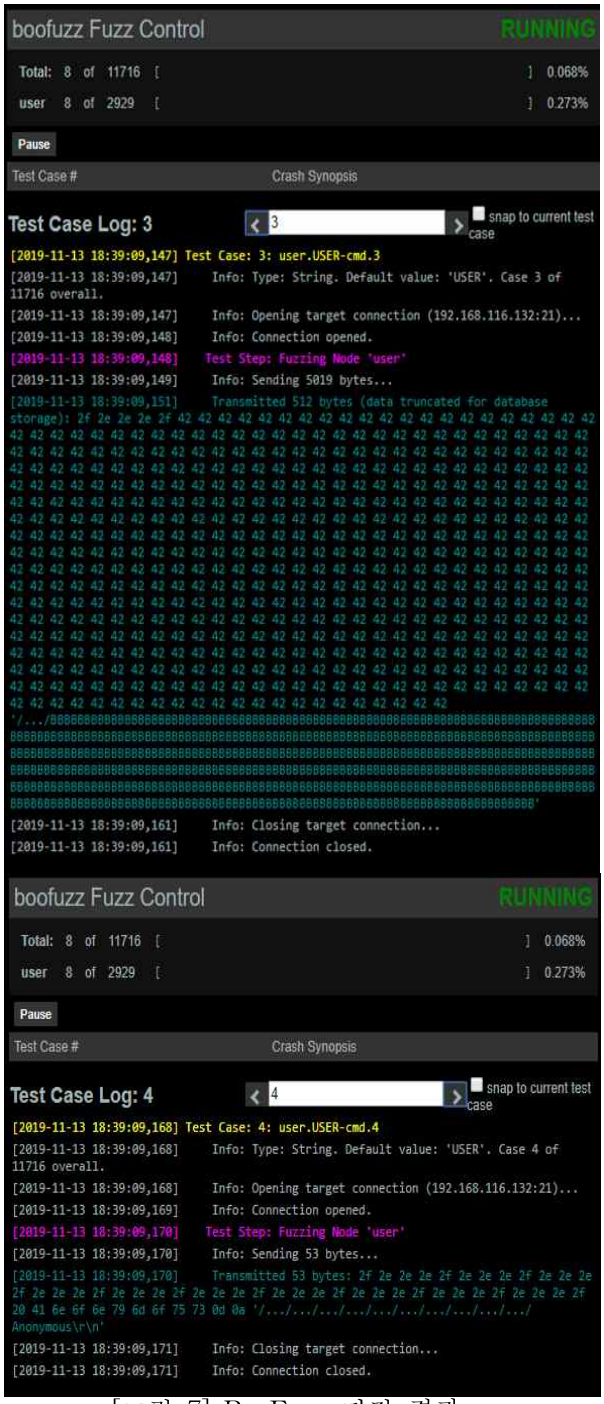
User 필드에 버퍼 오버플로우 취약점이 발생한 것이다.

버퍼 오버플로우는 프로그램이 실행될 때 입력받는 값이 버퍼를 가득 채우다 못해 넘쳐흘러 버퍼 이후의 공간을 침범하는 현상. 쉽게 말해 양동이(버퍼)에 물(값)을 받을 때(입력 받기) 물을 너무 많이 받아 바닥에 흘러 넘쳐버리는 것이라고 생각하면 된다

아래 [그림 6]과 같이 FTP 프로토콜의 Input 형식에 맞게 정의한 ftp.py 소스코드 파일을 실행하여, BooFuzz를 구동하였다.



[그림 6] Boofuzz 구동


boofuzz Fuzz Control
RUNNING
Total: 8 of 11716 [ ] 0.068%
user 8 of 2929 [ ] 0.273%
Pause
Test Case #
Crash Synopsis
Test Case Log: 3
3
snap to current test case
[2019-11-13 18:39:09,147] Test Case: 3: user.USER-cmd.3
[2019-11-13 18:39:09,147] Info: Type: String. Default value: 'USER'. Case 3 of 11716 overall.
[2019-11-13 18:39:09,147] Info: Opening target connection (192.168.116.132:21)...
[2019-11-13 18:39:09,148] Info: Connection opened.
[2019-11-13 18:39:09,148] Test Step: Fuzzing Node 'user'
[2019-11-13 18:39:09,149] Info: Sending 5019 bytes...
[2019-11-13 18:39:09,151] Transmitted 512 bytes (data truncated for database storage): 2f 2e 2e 2e 2f 42 42 42

차량 위협 모델과 보안 정책

고동의*, 박상우*
*안동 경일고등학교, *대구 달성고등학교,

*Vehicle threat model and security policy

Dong-Eui Ko*, Sang-Woo Park*
*Andong Gyeongil-Highschool, *Deagu Dalseong-Highschool

요 약

이동 통신이 발전함에 따라 차량도 함께 발전하면서 많은 모듈들이 생겨나기 시작했고 이 모듈들을 제어하기 위해 전자제어 장치들이 나왔지만 편의성만 추구하다 보니 보안은 많이 취약한 상황이 되었다. 5세대 이동 통신까지 나온 이 시대에 자율주행 차량, 커넥티드 카 등 여러 기술을 가진 차량이 나오고 있다. 그러나 테슬라 전기자동차가 단시간에 해킹되는 등 차량 부분은 보안이 취약한 상황이다. 여러 보안 정책이 나왔지만 여전히 예전 공격기법에 취약한 차량이 많이 존재한다. 본 논문에서는 차량의 취약 벡터를 알아보고 보안 정책에 대해 알아본다.

I. 서론

현대의 차량들은 CAN, 블루투스, USB, Key-Fob, 인포테인먼트 콘솔 심지어는 WIFI 등 많은 프로콜을 제공하고 있다. 이러한 운전자와 차량 간에 편리한 서비스를 제공해 주지만 동시에 운전자의 목숨을 위협하는 존재가 될 수 있다.

CAN의 경우 브로드캐스팅 형식의 통신을 하기에 OBD-II 커넥터를 이용하여 CAN 신호를 분석해 특정 신호로 조작 할 수 있다. 블루투스는 차량과 운전자의 무선 연결을 담당하는 프로토콜이지만 이를 공격한다면 차량 인포테인먼트 시스템에 접근할 수 있게 된다. USB 프로토콜을 이용하여 인포테인먼트 시스템에 악성코드를 설치하거나 취약점을 찾아 공격할 수 있다. Key Fob 프로토콜은 RF 신호를 이용하여 Key Fob 신호를 복사하거나 Jamming 공격을 통해 정상적인 통신을 방해할 수 있다.

DREAD risk assessment model은 Microsoft에서 이전에 사용된 컴퓨터 보안 위협을 평가하는 시스템의 일부이며 위험 등급 보안 위협에 대한 Mnemonic을 제공하는데 이 등급 체계를 활용하여 차량 위험 등급을 평가하기도 한다.

본 논문에서는 각 프로토콜에 보안 문제점을 제시한 후 문제점을 보완할 수 있는 정책에 대해 기술한 후 결론을 맺는다.

II. 보안 문제점

2.1 블루투스 프로토콜 문제점

차량에 있는 블루투스 프로토콜은 운전자의 휴대폰과 연결할 수 있는 기능을 제공해준다. 여기서 발생한 취약점은 블루투스 인증 번호가 단순한 정수 4자리로 추측하기 쉬운 번호로 되어있어 단순 brute force 공격으로 차량 블루투스에 접근할 수 있다. brute force 공격으로 인증번호를 알아낸 후 blueborn 공격을 통해 차량 시스템을 제어할 수도 있고 bluesnarfing 공격으로 운전자와 차량 통신이 도청 가능하다.

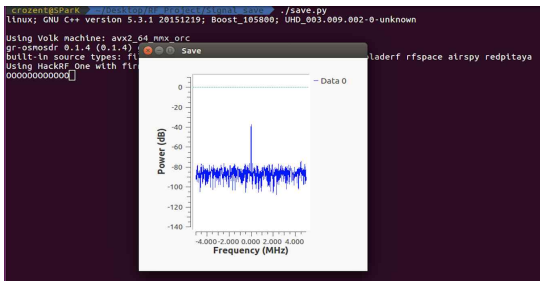
2.2 Key-Fob 문제점

이 프로토콜은 차량 무선 리모컨 키(Key-Fob)의 문제점이다. Key-Fob는 RF 라디오 신호를 이용해 차량의 문을 열거나 트렁크를 열 수 있는데

단순 RF 통신을 하는 것이 문제이다. 가령, RF 신호를 녹화 후 다시 전송하기만 해도 Key-Fob 에서 보낸 신호와 똑같은 작동을 하게 된다.

2.2.1 Replay Attack

차량의 문을 여는 RF 신호를 녹화하여 전송한다면 차량 문이 열릴 것이라는 가설을 세울 수 있다. 이 가설을 증명하기 위해 차량 신호를 녹화한다.



[그림 1] 차량 RF 신호 녹화

[그림 1]과 같이 녹화된 신호를 차량으로 전송하게 되면 차량 문을 여는 명령을 수행하는 신호가 담긴 RF 신호가 차량으로 전달되기 때문에 문이 열리게 된다. 차량 내 RF 통신은 별도의 인증 체계가 없어 단순 RF 신호 재생만으로도 문을 열 수 있다는 문제점을 증명하였다.

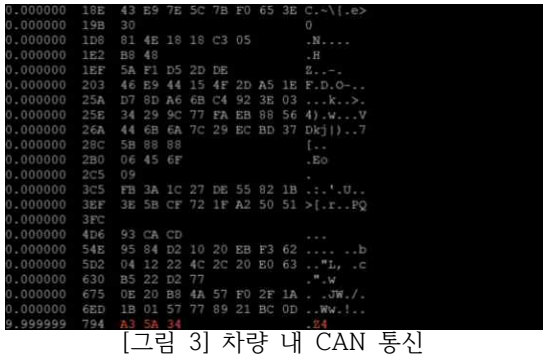
2.3 CAN 통신의 취약점

CAN-BUS 통신은 차량의 마이크로 컨트롤러나 장치들이 호스트 컴퓨터 없이 통신하기 위해 설계된 표준 통신 규격이다. 표준 규격인 OBD-II 포트로 연결할 수 있으며 주로 차량 운전석 좌측 하단에 위치해 있다.



[그림 2] 차량 내 OBD-II 포트

OBD-II 는 2005년 1월부터 국내에 판매되는 모든 차량에 의무화되어있어 2005년도 이후 차량은 좌측 하단에 [그림 2]와 같은 포트를 볼 수 있다. [그림 2] 포트에 OBD-II 커넥터와 CAN 컨버터를 랩탑에 연결하여 차량 내 CAN 통신을 볼 수 있다.



[그림 3] 차량 내 CAN 통신

CAN 통신은 네트워크 패킷처럼 각 모듈들이 통신하는데 [그림 3]과 같이 통신 데이터를 볼 수 있다. 가령 차량 내 엑셀 통신 패킷을 찾아 녹화하여 Replay Attack을 하거나 패킷을 조작해 전송하여 엑셀의 속도를 높이거나 방향을 재어 할 수 있다.

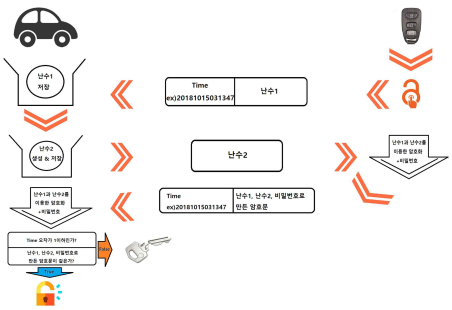
III. 보안 정책

3.1 블루투스 보안 정책

차량 내 인포테인먼트 시스템은 주로 안드로이드나 리눅스를 사용하고 있다. 안드로이드 운영체제의 blueborn 취약점은 CVE-2017-0781 ~ 0783, 0785 취약점이 있으며 리눅스 운영체제의 blueborn 취약점은 CVE-2017-1000250 ~ 1000251 취약점이 있다. 현재 차량에 보유하거나 네트워크에 연결된 장치를 식별하고 있어야 하며 제조사가 이를 파악하여 블루투스를 업데이트해야 한다.

3.2 Key-Fob 보안 정책

Replay Attack을 막기 위해서 난수를 활용하는 방법이 있다.



[그림 4] 난수 생성 및 인증

운전자가 자동차의 문을 열기 위해서 스마트키의 열림 버튼을 누르게 된다면 스마트키에서 난수 1을 생성하고 생성한 시간 및 난수 1을 암호화한 뒤 자동차에 방출한다. 자동차는 스마트키로

받아온 난수 1과 난 수 1 생성시간을 저장하고 또 다른 난수 2를 생성한 후 위와 같은 방식으로 난수 2를 스마트키로 방출한다. 이러한 과정을 통하여 자동차와 스마트키는 동일한 두 개의 난수를 가지게 된 다. 각각의 장비는 이 두 가지 난수를 이용해 특정 암호화 방식을 이용하여 암호문을 만든다. 스마트키는 암호문과 암호문 생성시간을 포함하여 암호화한 뒤 자동차로 방출한다. 스마트키 가 생성한 암호문과 자동차가 생성한 암호문이 동일하고 난수 1 생성시간과 암호문 생성시간 의 오차가 1초 이하라면 문이 열리게 된다.

```
public Hashtable<String, Long> makeNaN(){
    nan.put("TIME", System.currentTimeMillis());
    nan.put("RAND", (long) (rand.nextInt(1000000)+1));
    System.out.println("[스마트키]1. 난수 생성완료 & 전송");
    return nan;
```

[코드 1] 난수 생성 및 전송

[그림 4]와 같은 정책을 적용하면 차량의 RF 신호를 녹화하더라도 인증 체계를 거쳐야 하기에 Replay Attack을 막을 수 있다.

3.3 CAN-BUS 보안 정책

CAN-BUS 통신의 가장 큰 문제점은 암호화되지 않은 상태로 통신하는 것이다. 5G 시대에 들어서면서 자율주행 기능이 있는 스마트 카가 암호화되지 않은 CAN-BUS 통신을 하게 된다면 공격자의 의도대로 차량 속도나 방향 등을 제어할 수 있게 된다.

3.3.1 CAN-BUS 통신 모듈 ID 암호화

CAN-BUS 통신을 하는 모듈의 ID를 암호화하여 공격자가 목표하는 모듈의 ID를 찾을 수 없도록 하는 방법이다.

3.3.2 Dummy 데이터 통신을 활용한 난독화

CAN-BUS 통신에 사실상 무의미한 ID 값을 추가하여 공격자가 보기에 마치 진짜 모듈들이 통신하는 것처럼 보이도록 한다. 이로 인해 공격자는 공격 대상을 찾기 어렵게 된다.

IV. 결론

21세기에 들어오며 통신이 발전하여 1세대 이동통신(1G)부터 5세대 이동통신(5G)까지 발전해 왔으며 차량 또한 4세대 이동통신부터 맞춰가며 발전하기 시작했다. 특히 5세대 이동통신 시대는 인공지능이 탑재된 자율주행 자동차 기술과 차량 간 통신인 V2V (Vehicle to Vehicle) 기술이 나오면서 보안 또한 중요해지고 있다.

위 내용과 같이 아직까지 보안이 적용되지 않은 CAN-BUS 통신부터 보안이 점점 적용되고 있는 Key Fob 의 RF 통신, 그리고 보안은 적용되었으나 종종 차량에 적용되어있지 않은 블루투스 까지 다양한 보안 문제점이 나오고 있다. 5G 기술이 나오며 차량에도 적용되고 있는 이 시대에 차량 보안을 소홀히 한다면 운전자의 목숨은 물론 교통에도 혼란을 주게 될 것이기에 차량 보안도 소홀히 하지 말아야 할 것이다.

[참고문헌]

Craig Smith (2016).The Car Hacker's Handbook(A Guid for the Penetration Tester)

블루투스 취약점
https://www.norma.co.kr/blog/2018/01/11/blueborne-취약점-정리/

White-Box Cryptography의 사용 동향과 취약점에 관한 조사 연구

박예찬*

*한양대학교 국방정보공학과

A Study on the Trends and Vulnerability of White-Box Cryptography

Ye-Chan Park*

*Military Information Engineering, Hanyang University.

요 약

암호키가 누출되지 않을 것을 가정하고 만드는 블랙박스 암호화와는 다르게, 공격자가 내부를 들여다볼 수 있다는 가정 아래에 설계되는 화이트박스 암호화가 최근 주목을 받고 있다. 본 논문은 정부 기관은 물론이고 대기업과 핀테크 기업까지 화이트박스 암호화를 도입하는 동향과 추세에 대해서 조사하였다. 또한 화이트박스 암호화가 강조되고 있지만 부 채널 공격과 화이트박스 자체의 취약점을 가지고 있으므로 이것에 대해서 조사를 하고 주의를 당부하며 차세대 화이트박스 암호화 연구 방향을 제시한다.

I. 서론

스마트폰의 등장 및 다양한 멀티미디어 기기들이 출현함에 따라 해당 기기의 이용자 인증 방식도 중요하게 대두되고 있다. 특히 이용자가 단말기를 통해서 모든 환경을 제어할 수 있는 화이트박스 공격 상황들이 도래하고 있기 때문에 인증을 위한 키를 적절히 암호화 하는 것도 중요한 과제로 떠오르고 있다.

2009년 S. Chow가 논의한 디지털 권리 관리(DRM)를 위해 개발 된 화이트박스 암호화가 최근 각광을 받고 있는 추세이다[17]. 이는 소프트웨어만으로도 중간 값 및 암호키를 보호할 수 있다는 장점을 포함해서 오픈 플랫폼의 단말에서도 안전함을 보장할 수 있는 기존의 블랙박스 암호화 방식을 뛰어넘는 대체제로 사용될 것이라고 논의되어 지고 있다[1].

또한 DRM에서 이용할 뿐만 아니라 최근 주목 받고 있는 삼성페이, 제로페이 등의 스마트폰을 통한 결제 시스템과 같은 핀테크 기술에서도 적용이 가능할 것으로 사료된다[5][10].

화이트박스 암호는 최근의 기사와 매체에 따르면 한국전자통신연구원과 같은 국가 기관은 물론이고, 삼성SDS, 카카오뱅크등과 같은 사기업에서도 클라우드, 핀테크 보안을 위해서 도입 및 염두하고 있거나 현재 도입을 해서 사용하고 있는 추세이다[1][2][3][4][7].

이런 화이트박스 암호화도 다양한 공격들에 대한 난이도를 충분히 높여 놓은 것에 대해서 인정되고 있는 반면에, 해외의 여러 연구들에 따르면 그레이박스 공격점에 대해서는 추가적인 논의가 필요하다고 당부되는 시점이다[13][15][16].

본 논문에서는 화이트박스의 사용 추세에 대해서 논의를 하고 이런 화이트박스 암호도 그레이 박스 공격에 대해서 취약한 부분이 드러나고 있으며 이를 통해서 화이트박스 암호화를 사용할 때 주의해야 할 사항과 향후 발전 방향에 대해서 논의해 보고자 한다.

II. 배경 지식

2.1 화이트박스 암호화

전통적인 블랙박스 암호화의 경우에는, 암호화를 설계하는 공격자가 암호화 모듈 내부를 절대로 볼 수 없을 것이라는 가정 하에 모듈을 설계하게 된다. 하지만 화이트박스 암호화의 경우에는 입력 값과 출력 값뿐만 아니라 중간 값까지도 노출될 수 있다는 가정 하에 설계가 된다. 따라서 공격자가 연산 도중에 나오는 중간 값을 확인할 수 있더라도 이것도 암호화 되어 있기 때문에 암호키를 추론하기가 어렵다는 장점을 가지고 있다.

2.2. 그레이 박스 공격의 정의

그레이박스 공격은 보안을 유지해야 할 하드웨어에서 나오는 부차적인 특성(온도, 전자파, 전력 등)에 해당하는 공격을 지칭한다. 이는 다른 말로 Side-Channel Attack으로도 불리우고 있으며, 하드웨어가 누수하는 정보는 물론이고 하드웨어에 다른 강제적인 힘을 가하여 정보를 누수하게 만드는 방식도 일컫는다.

III. 관련 동향

3.1. 국내 사업 동향

핀테크 보안 기업 아톤(대표 김종서)의 경우, 문화상품권 관정 업체인 한국문화진흥과는 모바일 OTP에 화이트박스 암호화를 적용하는 솔루션 공급 계약에 체결한 바가 있으며, 인터넷 은행인 케이뱅크 등과의 업체에 화이트박스 암호화가 적용된 솔루션에 대한 협약을 체결한 바가 있다[7].

이니텍(대표 장홍식)의 경우에는 작년 초 국내 최초로 화이트박스 암호 솔루션을 출시한 바가 있으며, 가상화폐 거래소에 전자지갑 암호화에 사용 될 수 있다. 또한 생체 인증 솔루션에도 적용과 불안한 단말 환경에서도 적용이 가능하다고 밝힌 바가 있다[4].

카카오뱅크(대표 이용우 윤호영)는 화이트박스 암호화 기술이 인터트러스트(한국지사 대표 강승환)에 위탁하고 활용하고 있으며, 인터트러스트는 암호화 키가 메모리 상에 평문으로 노출되는 것이 아닌 암호화 키도 별도로 관리가 되며, 화이트박스 암호화를 통해 각기 다른 모바일 디바이스의 경우에도 소프트웨어적인 암호화 방식이라 동일하게 적용이 가능한 장점이 있다고 밝힌 바가 있다[2].

삼성SDS(대표이사 사장 홍원섭)는 클라우드 시장에 화이트박스 암호화 기술을 접목시켰으며, 현재 화이트박스 암호화의 가장 문제점으로 대두되고 있는 속도 부분에서 크게 향상시켜 타 암호화 대비 30배 이상 빠르게 개발했다고 한다[3]. 또한 화이트박스 암호화로 글로벌 보안 인증을 취득한 국가는 전 세계에 4개국 뿐인데, 그 중에 한국에서는 삼성SDS가 유일하다[6]. 그 중에서 삼성SDS의 플랫폼인 Nexsign에서 지문, 홍체 등의 바이오 정보를 이용한 인증 시스템을 암호화 하는 기술로써 화이트박스 암호를 사용함을 알 수 있다[18].

3.2. 국내 학계 동향

국내에서는 화이트박스 암호 및 응용 기술 동향 분석이라는 논문[1]으로 한국전자통신연구원에서 발간하는 논문집에 게재된 바가 있다. 또환 한국전자통신연구원에서는 2014년에 화이트박스 암호화를 활용해서 화이트박스 공격에 대응할 수 있는 모바일 단말기에서의 연구[12]를 IEEE에 제출한 바가 있다. 금융보안원에서 발간한 전자금융과 금융보안 2018년 4월자에서는 글로벌 금융시장의 화이트박스 암호화 기법의 사용 동향[5]에 대해서 연구된 바가 있다.

3.3. 해외 학계 동향

Estuardo Alpirez Bock의 논문[10]에 따르면 화이트박스 암호화 기술은 모바일 결제 수단으로써 사용될 수 있는 장점에 대해서 논하고 있지만 비압축적인 특성이 다음절에 나올 Code Lifting 공격에 대해 취약하다는 논의가 되어지고 있다.

Arunima Saha의 논문[8]에 다르면 화이트박스 암호화 기술을 IoT 기기에서 Code Lifting 공격 혹은 차분전력분석 등과 같은 그레이박스

공격에 대해서 보완하기 위한 연구를 진행하였다. 화이트박스 암호화의 취약점에 대해서 논하고 발전시키는 방향으로 연구를 진행한 바가 있다.

IV. 그레이 박스 공격과 화이트박스 암호화의 취약점

4.1. 그레이 박스 공격

Cold Boot Attack 전원이 공급되지 않아도 DRAM에 올라간 정보는 일정시간 삭제되지 않으며 온도를 낮추면 더 오랜 시간동안 정보가 삭제되지 않아 그 시간 내에 다른 장치에 해당 메모리 칩을 탑재하고 암호 키를 분석해 내는 방식이다[11].

뿐만 아니라 잘 알려진 Side Channel Attack으로는 Pascal Sasdrich의 논문[16]에 따르면 Differential Power Analysis Attack과 Differential Computational Analysis Attack 그리고 Mathmatical Founctions에 대해서도 언급을 하고 있으며 이에 관한 화이트박스 취약점에 대해 소개를 하고 있다.

4.2. 화이트박스 암호화 취약점

Cold Lifting Attack 공격자는 비밀키를 추출해 내는 것 보다는 화이트박스 실행 환경 자체를 복제해서 사용하는 것에 초점을 둔다. 이 취약점은 화이트박스 실행 자체를 감싸는 External Encoding 으로 암호화 시켜서 예방을 할 수 있다.

White-Box Inversion 공격자는 테이블을 추출 할 수 있고, 모든 라운드의 계산을 역으로 하면서 복호화 할 수 있다. 이 취약점은 Cold Lifting Attack의 취약점과 마찬가지로 External Encoding 방식을 이용해서 예방할 수 있다.

그리고 화이트박스 암호화는 여러 차례 진화를 하였다. 2005년 Billet에 연구에는 AES에 대한 공격이 키가 노출됨을 보였고[9], 이를 연구자들이 여러 차례 보완해서 2010년에는 Karroumi의 논문[14]을 통해서 안전한 화이트박스 암호화가 되기까지의 취약한 부분을 개선하고 발전시켰음을 확인할 수 있다.

4.3. 현존하는 화이트박스 암호화 단점

화이트박스 암호화를 DRM에서 사용할 경우에 룩업 테이블을 서버에 저장해야 하므로 서버가 처리해야 할 데이터의 양이 상당히 많아지게 된다[1]. 또한, 콘텐츠를 암호화 하고 암호키만 전송하면 되는 기존의 블랙박스 암호화 방식과는 다르게 테이블 전체를 전송해야 하기 때문에 다량의 데이터 전송에 대한 처리 부분도 개선이 필요하다.

V. 결론 및 향후 연구

현재 국내 및 국외에서 화이트박스 암호화와 그에 관련된 솔루션들을 제작하는 등 관심이 모이고 있다. 하지만 아직 남겨진 부채널 공격의 위협이나 자체 취약점이 존재하고 있으므로 화이트박스 암호화 이러한 부분에 대해서 대응을 할 수 있게 다른 암호화 방식을 섞어서 사용하는 방식이나 난독화를 더 거치는 방식에 대해서 고민을 해보아야한다.

또한 스마트폰도 초기 출시 대비 어느 정도 사양이 많이 향상 되었지만, 최근에 많은 관심을 받고 있는 IoT 기기의 경우에는 성능이 좋지 않기 때문에 다량의 정보를 전송하는 화이트박스 암호화의 사용이 어려울 것[10]으로 사료된다. 따라서 데이터 사이즈와, 전송 부분에 대해서 지속적인 연구가 요구된다.

끝으로, WhibOx Contest에서 화이트박스 암호화에 관련한 경진대회(Capture the Flag)를 매년 열고 있다[18]. 이와 관련된 Louis Goubin의 논문[15]에서는 해당 대회에서 사용된 여러 가지 방법들을 소개하고 있으므로 각종 최신의 화이트박스 암호화의 취약점에 대해서 파악할 수 있다.

[참고문헌]

[1] 김신효, 이윤경, 정병호, “화이트박 스 암호 및 응용기술 동향 분석”, 전자통신동향분석 제 25권 제 5호, 2010년 10월

[2] 손경호, “카카오뱅크 통신 보안지키는 '화이트박스 암호화'”, 지디넷코리아, 2017년 10월 24일자

[3] 신진섭, “삼성SDS, 화이트박스암호・동형암호 기술로 클라우드보안시장 공략”, 탑데일리 2019년 3월 14일자

[4] 안수민, “이니텍, 화이트박스 암호 솔루션 국내 최초 출시”, 전자신문, 2018년 2월 28일자

[5] 윤석근, Douglas Kinloch, “글로벌 금융시장의 화이트박스암호 기술 동향 및 보안 고려사항”, 전자 금융과 보안 제 14호, 2018년 10월

[6] 황민규, “삼성SDS, 비밀병기 '동형암호' 공개 "클라우드 공격 완벽 봉쇄"”, 조선비즈, 2019년 3월 14일자

[7] ITWorld 편집부, “아톤, 한국문화진흥과 모바일 OTP 솔루션 공급계약 체결” ITWorld, 2019년 9월 9일자

[8] Arunima Saha, Chungath Srinivasan, White-Box cryptography based data encryption-decryption scheme for IoT environment, 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), 2019

[9] Billet, Olivier, Henri Gilbert, and Charaf Ech-Chatbi. Cryptanalysis of a white box AES implementation. Selected Areas in Cryptography. Springer Berlin Heidelberg, 2005.

[10] Estuardo Alpirez Bock, Alessandro Amadori, Joppe W. Bos, Chris Brzuska, Wil Michiels, Doubly half-injective PRGs for incompressible white-box cryptography, Topics in Cryptology – CT-RSA 2019 pp 189-209, 2019

[11] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten, “Lest We Remember: Cold Boot Attacks on Encryption Keys”, 17th USENIX Security Symposium

[12] Jong-Yeon Park, Jae-Dock Lim, Jung-Nyeo Kim, Dong-Guk Han, A Whitebox Cryptography application for Mobile device security against whitebox attacks, 2014 IEEE

[13] Joppe W. Bos, Charles Hubain, Wil Michiels, Philippe Teuwen, Differential Computation Analysis: Hiding your White-Box Designs is Not Enough, IACR Cryptology ePrint Archieve, 2014

[14] Karroumi, Mohamed. Protecting white-box AES with dual ciphers. Information Security and Cryptology-ICISC 2010. Springer Berlin Heidelberg, 2011. 278-291.

[15] Louis Goubin, Pascal Paillier, Matthieu Rivain, and Junwei Wang, How to Reveal the Secrets of an Obscure White-Box Implementation, Journal of Cryptographic Engineering, April 2019

[16] Pascal Sasdrich, Amir Mor, Tim Guneysu, White-Box Cryptography in the Gray Box, Fast Software Encryption pp 185-203, 2016

[17] S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot, A White-Box DES Implementation for DRM Applications, In Security and Privacy in Digital Rights Management, ACM CCS-9 Workshop, 2002

[18]https://www.ncloud.com/marketplace/samsungsds

[19] https://whibox-contest.github.io/

암호 키관리 상호운용 프로토콜(KMIP)과 암호 키관리 솔루션(KMS) 동향 조사

김도원*, 최은영*, 박해룡*

*한국인터넷진흥원

A Survey on the Key Management Interoperability Protocol and Key Management Solution trend

Do-won Kim*, Eun-young Choi*, Hae-ryong Park*

*Korea Internet & Security Agency

요 약

암호 키는 보안 시스템에 있어 핵심적인 요소라고 볼 수 있다. 다양한 인증방식과 여러 가지 암호기술들을 적용하여, 암호 키가 취약하다면 근본적인 안전성이 흔들리게 된다. 그렇기 때문에 보안을 요구하는 어떤 시스템에서도 암호 키관리는 항시 중요하다.

최근 엔터프라이즈 환경은 그 복잡성이 점점 심화되면서, 다양한 환경들이 결합하고 있다. 여러 환경들이 결합되다보니 암호 키를 사용하기 위한 저마다의 프로세스를 갖고 있고, 이러한 암호 키관리를 통합하여 관리할 수 있는 표준 기술이 필요하다.

본 논문에서는 이러한 표준 기술로 활용될 수 있는 암호 키관리 상호운용 프로토콜(KMIP)와 암호 키관리 솔루션(KMS)에 대하여 조사하였다.

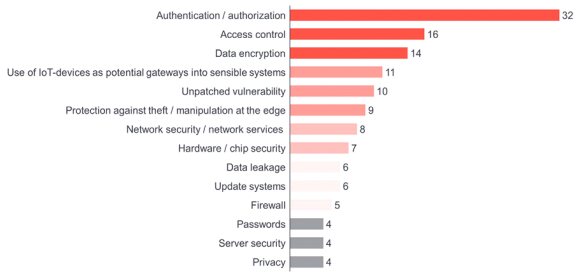
I. 서론

암호기술은 암호화, 무결성, 인증 등 다양한 보안 기능을 제공하며, 어떤 시스템에서든 암호기술들이 대부분의 보안 기능을 제공하고 있다.

암호기술을 운용하기 쉬운 일반적인 IT환경뿐만 아니라, IoT같은 제약된 환경에서도 암호기술의 중요성은 동일하다. IoT 회사들을 대상으로 IoT 보안에서 가장 중요한 세가지를 뽑는 투표에 따르면(그림1), 인증/인가, 접근제어, 데이터 암호화가 가장 높은 득표수를 얻었으며, 이 요소들은 모두 암호기술과 밀접한 관련이 있다.

블록암호, 해시함수 등 종류별로 수많은 연구들로 안전성이 검증된 다양한 암호알고리즘들이 존재한다. 하지만, 암호알고리즘을 안전하게 운용하기 위해서는 암호 키관리에 대한 고려가 필수적이다.

※ 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-00267)



(그림 1)중요한 IoT 보안 요소(출처 : IoT Analytics)

거의 모든 암호알고리즘은 안전한 키를 사용하는 것을 기반으로 안전성을 보장할 수 있다. 수많은 인증과 암호기술들을 적용하더라도, 암호 키가 취약하다면 시스템의 안전성은 근본적으로 흔들리게 된다.

암호 키관리는 암호키를 생성하고, 안전하게 보관하며, 키를 사용하는데 필요한 모든 절차들을 포함하는 것이다. 하지만, 암호 키관리와 관련된 표준들에서 요구하는 사항들은 결코 간단하지 않다.

엔터프라이즈 환경은 여러 공급 업체 환경에서 운영되며, 해당 환경의 복잡도는 점점 복잡해지고 있다. 또한, 여러 공급 업체는 암호화 및 암호 키관리를 위한 별도의 프로토콜과 처리 프로세스를 갖고 있기 때문에, 각 환경들의 보안과 접근성을 조화시키는 것은 어려우며 관리비용 또한 지속적으로 증가한다. 이러한 문제점을 해결하기 위해서는 표준화된 암호 키관리 기술이 필요하다.

본 논문에서는 암호 키관리를 위해 별도로 적용할 수 있는 KMIP(Key Management Interoperability Protocol)[1]와 상용 키관리 솔루션들(Key Management Solution, KMS)을 조사하였다. 2장에서는 KMIP의 특징과 구성을 분석하고, 3장에서는 상용 KMS들을 조사하였다. 4장에서는 결론으로 마친다.

II. KMIP

KMIP는 OASIS에서 제정한 키관리 시스템 프로토콜 표준이다. KMIP의 사용주체는 클라이언트와 KMS이다. KMIP는 다양한 키관리 프로세스들과 키관리 프로토콜 메시지의 표준화를 목표로 하고 있다.

2.1 KMIP 표준

KMIP 표준은 Specification, Profile, Test-Case, Usage-Guide 총 4가지로 구성되어 있다[표 1]. 현재 각 문서마다 1.4버전이 등재되어 있고, 2019년 10월31일 기준으로 2.0 버전의 Specifacion과 Profile이 추가되었다.

[표 1] KMIP 표준 구성

구분	내용
Specification	○ 프로토콜의 전반적인 설명 - 프로토콜 패킷 구성 - 헤더/바디 구조 등
Profile	○ Specification을 준수하는 방법 - 환경에 따른 KMIP 사용방법 - 적합성 검사 기반 자료
Test Case	○ 실제 사용 예시 - XML, TTLV 값을 통한 시연 - 시나리오를 통한 사용 케이스
Usage Guide	○ Specification의 부가 설명 자료

SNIA에서는 KMIP에 대한 적합성 검사를 진행한다. KMIP 서버/클라이언트를 개발하면 해당 검사를 통해 스펙을 준수하였는지 검사할 수 있다. 해당 검사를 통과하면, OASIS KMIP TC Wiki에 등재될 수 있으며, 국내에서는 한컴시큐어의 KMS 제품이 적합성 검사에 통과하였다.

2.2 기본 구성[2]

KMIP에서 다루는 객체들은 크게 Managed-Object[표 2]와 Base-Object[표 3]로 구분된다. 해당 객체들은 Operation을 통해 제어한다.

Managed-Object는 KMS에 의해 안전하게 저장되고, 정상적으로 사용되어지는 객체들이다.

[표 2] Managed-Object

구분	설명
Certificate	- X.509 인증서
Symmetric Key	- 대칭키
Public Key	- 공개키(비대칭키)
Private Key	- 비밀키(비대칭키)
Split Key	- 분할 키
Template	- 암호 키 정책
Secret Data	- 단순 랜덤 비트열
Opaque Object	- 사용자가 확장 정의한 Managed Object

Base-Object는 프로토콜 메시지에 사용하는 객체들이다.

[표 3] 주요 Base-Object

구분	설명
Attribute	- Request 메시지의 키에 대한 메타 데이터 정보
credential	- 클라이언트를 식별/인증하기 위한 정보
KeyBlock	- 키를 등록하고 조회할 때 사용하는 정보
Key Wrapping Data	- 키 랩핑할 때 사용하는 정보
Transparent Key Structures	- 기타 표준에서 제공하는 암호키 표현방법 (ex. RSA의 n, e 값)

2.3 KMIP 메시지 구조

KMIP Request 메시지는 (그림 2)와 같으며,

Response 메시지는 이와 유사하다.

Header는 일반적인 프로토콜의 헤더 역할을 하며, Batch Item은 메시지의 바디 역할을 한다. 예를 들어, Header에는 프로토콜 내 패킷에 대한 정보가 들어가며, Batch Item의 Operation에 "생성(Create)"가 들어가며, Payload에는 Operation에 따라 서버에서 필요한 다양한 정보들이 들어가야 한다.

Header

Protocol Version	Maximum Response Size
Client Correlation Value	Server Correlation Value
Asynchronous Indicator	Attestation Capable Indicator
Attestation Type	Authentication
Batch Error Continuation Option	Batch Order Option
Time Stamp	Batch Count

Batch Item(s)

Operation	Unique Batch Item ID
Request Payload	Message Extenstion

(그림 2) KMIP Request 메시지

III. 키 관리 솔루션(KMS)

KMS는 암호키 또는 암호키와 관련된 객체들에 대하여, 생성, 등록, 분배, 저장, 파기 등의 절차를 관리해주는 솔루션이다. 본 장에서는 KMS의 특징과 상용 솔루션을 조사하였다.

3.1 특징

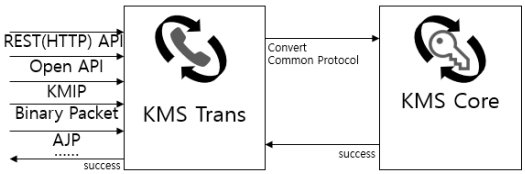
3.1.1 HSM

KMS는 하드웨어 방식 및 소프트웨어 방식으로 제공할 수 있다. 단순히 암호화 키를 관리하는 목적이라면 단순 소프트웨어 형태로도 관리가 가능하다.

하지만, 높은 수준의 보안이 요구된다면, 안전한 저장공간으로 활용될 수 있는 추가적인 하드웨어 장비가 필요하다. HSM은 키 혹은 관련 정보가 일반적인 소프트웨어 형태가 아닌, 특별한 방법으로 보호될 수 있다.

3.1.2 KMS 중계서버

KMS의 중계서버는 여러 API나 프로토콜을 대응하기 위해, 일종의 프록시서버의 기능을 한다고 볼 수 있다.

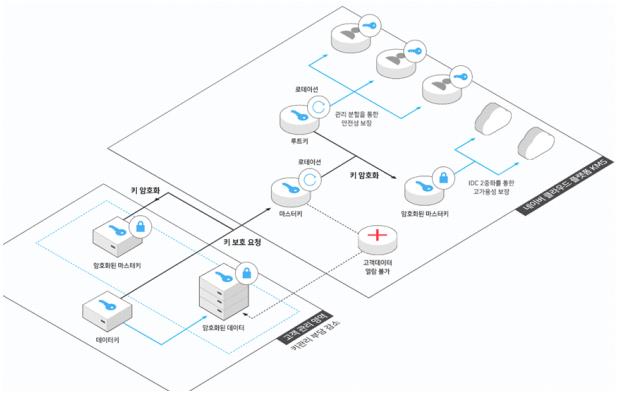


(그림 3) KMS 중계 서버

3.2 상용 솔루션

3.2.1 Naver Cloud Key Management Service[3]

네이버의 클라우드 시스템을 이용한 KMS이다. 분할 저장된 루트키를 이용하여 사용자의 마스터키를 암호화하고, 해당 마스터키를 이용하여 사용자의 데이터키를 보호한다. 마스터키와 루트키는 자사의 클라우드 시스템에 보관되어 있는 구조이다. 데이터←데이터키←마스터키←루트키의 형태로, 3중으로 암호화되어 있다고 볼 수 있다.



(그림 4) 네이버 클라우드 기반 KMS

3.2.2 AWS Key management Service[4]

AWS managed key, Customer Managed key(CMK), Custom key store(CKS) 세가지 방식으로 서비스를 제공한다.

AWS managed key는 AWS 서비스들이 내

부적으로 키를 제공하며, 사용자는 직접적으로 제어가 불가능하다. CMK 방식은 AWS가 사용자의 키를 보호하여, 어느정도 제어가 가능하다. CKS는 AWS의 또다른 키 관리형 서비스인 CloudHSM을 활용한 방식으로, 사용자가 모든 부분에서 제어가 가능하다.

AWS(Amazon Web Services) KMS는 일반적인 KMS들과 별반 다르지 않으며, 대신 다른 AWS 서비스들과 연계되어 보다 좋은 통합 서비스를 제공하는 장점을 갖는다.

3.2.3 한컴시큐어 XecureKeyManager[5]

한컴시큐어의 XecureKeyManager는 국내 KMS의 선두주자라고 볼 수 있다. 2016년 한컴시큐어는 OASIS KMIP TC에 가입하였다.

제품은 하드웨어와 소프트웨어 형태 모두 지원하며, KMIP 표준을 준수한다. 또한, 암호모듈 인증(KCMVP)을 받은 암호모듈을 탑재하였다.



(그림 5) 한컴시큐어 XecureKeyManager

IV. 결론

본 논문에서는 암호 키관리 상호운용 프로토콜(KMIP)와 암호 키관리 솔루션(KMS)들을 살펴보았다.

암호 키를 안전하게 관리하고 사용하는 것은 모든 보안시스템에 있어 중요하다. 점점 복잡해지는 엔터프라이즈환경에서, 통합적으로 암호키관리를 운영하는 것은 안전성이나 비용 등 여러 가지 측면에서 유리할 수 있다. 이러한 점에서 KMIP나 상용 KMS들이 활용될 수 있다.

[참고문헌]

[1] OASIS Key Management Interoperability Protoco(KMIP) TC, https://www.oasis-open.org/committees/kmip/

[2] Key Management Interoperability Protocol Specification Version 1.4, http://docs.oasis-open.org/kmip/spec/v1.4/os/kmip-spec-v1.4-os.html

[3] 네이버 클라우드 플랫폼 - 키 관리 서비스, https://docs.ncloud.com/ko/kms/kms-1-1.html

[4] AWS Key Management Service(KMS), https://aws.amazon.com/ko/kms/

[5] 한컴시큐어 - Xecure Key Manager, http://www.softforum.com/solutions/solution_03_02.asp

안전한 바이오인증을 위한 함수 암호 구현 연구 동향

전성윤, 임종혁, 이문규*

*인하대학교 컴퓨터공학과

Survey of functional encryption implementations for secure biometric authentication

Seong-Yun Jeon, Jong-Hyuk Im, Mun-Kyu Lee*

*Department of Computer Engineering, Inha University

요 약

함수 암호는 암호문과 키가 주어질 경우 평문을 직접 획득할 수는 없지만 평문에 특정 함수가 적용된 결과는 계산할 수 있도록, 즉 제한적인 복호화가 가능하도록 하는 암호이다. 최근에는 함수 암호 중 특히 내적 함수 암호를 적절히 활용하여 안전한 바이오인증이 가능하도록 하는 방안이 소개된 바 있으며, 이를 위해 내적 함수 암호를 설계하고 실용적인 구현 결과를 제시한 결과들도 발표되고 있다. 이 논문에서는 이러한 내적 함수 암호의 구현에 대한 최근 연구 동향을 소개하고 그 성능을 비교한다.

I. 서론

최근 모바일 및 온라인 서비스가 보편화되어 온라인 비대면 활동이 늘어남에 따라 사용자 인증 또한 많은 서비스 영역에 적용되었다. 이러한 영향으로 사용자 인증 정보의 도용이 심각한 사회적, 경제적인 문제를 야기할 수 있게 되었다. 최근까지 널리 사용되고 있는 사용자 인증 방식인 패스워드, PIN (Personal Identification Number), IC 카드 등 지식이나 소유 기반의 인증에 비해 얼굴, 지문, 홍채 등의 바이오인증 정보를 이용한 바이오인증은 사용자가 인증 정보를 따로 관리 또는 휴대할 필요가 없다는 이점이 있어 다양한 서비스에 적용되고 있다. 그러나 바이오 정보는 지식이나 소유 기반의 인증 정보와는 달리 사용자마다 고유하고 변하지 않는다는 특성 때문에 유출될 경우에 지속해서 악용될 수 있어 보호의 필요성이 크고 민감한 개인정보이다 [1].

추가적인 하드웨어 없이 민감한 바이오인증 정보의 유출을 방지하기 위해서 고려할 수 있는 방법으로는 cancelable biometrics [2]나 fuzzy commitment [3]와 같은 변환 기법들이 있다. 그러나 이러한 변환에 기반을 둔 방법들은 인식률이 저하되는 단점을 갖고 있다. 또 다른 보호 방법으로 원격 서버에 바이오인증 정보를 저장한 후 원격으로 인증을 수행하는 방법도 고려 가능하나, 원격 서버에 대한 신뢰성 확보라는 새롭게 해결해야 할 문제가 있다. 한편, 암호화된 상태로 바이오인증 정보를 원격 서버에 저장하면 이러한 원격 서버에 대한 신뢰성을 확보할 수 있는데, 다만 서버가 직접적으로 암호화된 바이오인증 정보를 복호화할 수 있으면 저장된 바이오인증 정보를 보호할 수 없으므로, 암호화된 상태로 인증만을 가능하게 하는 방법이 필요하다. 이러한 문제를 해결하여 안전하게 서버에 저장된 바이오인증 정보를 이용해 원격 인증을 수행할 수 있도록 하는 방법으로 함수 암호(Functional Encryption: FE)를

고려할 수 있다. 함수 암호란 함수 f와 연관된 비밀키의 소유자는 데이터 $\boldsymbol{y}$에 대한 암호문으로부터 $f(\boldsymbol{y})$의 값을 복원할 수 있는 암호다[4].

본 논문에서는 안전하고 실용적인 원격 바이오인증을 구성하기 위한 함수 암호 구현 연구의 최신 동향을 분석한다.

II. 함수 은닉을 지원하는 내적 암호

내적 암호(Inner-Product Encryption: IPE)란 함수 암호의 일종으로, 함수 f_x가 함수 내에 인코딩된 벡터 $\boldsymbol{x}$와 입력 벡터 $\boldsymbol{y}$의 내적, 즉 $f_x(\boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$로 정의될 때, 벡터 $\boldsymbol{y}$의 암호문을 입력으로 받아 $f_x(\boldsymbol{y})$를 계산할 수 있는 암호이다[5]. 만약, 내적 암호가 함수 f에 연관된 벡터 $\boldsymbol{x}$까지 보호하기 위해서는 추가로 함수 은닉(Function-Hiding)이라는 속성을 지원해야 하며[6], 이러한 스킴을 함수 은닉 내적 암호(Function-Hiding Inner-Product Encryption: FHIPE)라 부른다. FHIPE는 setup, left encryption, right encryption, decryption의 네 가지 기본 연산으로 구성되며, 이 중 left 및 right encryption은 각각 key generation 및 encryption으로 불리기도 한다[7]. 다음은 이들 네 가지 기본 연산에 대한 개요이다.

- *IPE.Setup*: 주어진 안전성 파라미터 λ에 따라 연산을 위한 파라미터 및 IPE 구성을 위한 파라미터를 계산하여 공개 파라미터 pp와 마스터 비밀키 msk를 출력한다.
- *IPE.LeftEnc*: 마스터 비밀키 msk와 벡터 $\boldsymbol{x}$가 주어지면, 암호문 ct_1을 출력한다.
- *IPE.RightEnc*: 마스터 비밀키 msk와 벡터 $\boldsymbol{y}$가 주어지면, 암호문 ct_2를 출력한다.
- *IPE.Decrypt*: 공개 파라미터 pp와 암호문 ct_1, ct_2가 주어지면, $\boldsymbol{x}$, $\boldsymbol{y}$ 두 벡터 사이의 내적 $z = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$을 출력한다.

III. 내적 암호 기반의 바이오 인증

2018년에 Kim 등은 위와 같은 함수 은닉 속성을 지원하는 IPE를 이용해 바이오인증을 구성하는 방법을 제안했다[7]. 해당 방법을 간략히 설명하면, 바이오인증 정보로 등록할 벡터 $\boldsymbol{x}$를 *IPE.LeftEnc*로 암호화한 암호문 ct_1을 서버에 저장하고, 인증을 위한 벡터 $\boldsymbol{y}$를 *IPE.RightEnc*로 암호화한 암호문 ct_2를 서버에게 제공하면, 서버는 *IPE.Decrypt*를 이용하여 ct_1과 ct_2로부터 계산된 내적 z를 이용하여 두 벡터 간의 유사도를 계산하여 인증을 수행한다. 만약 IPE가 함수 은닉 속성을 지원하지 않는다면, 서버는 등록 단계에서 저장된 벡터 $\boldsymbol{x}$를 공개 파라미터 pp만으로 알 수 있기 때문에 IPE를 이용한 원격 바이오인증은 구성할 수 없다.

일반적으로 바이오인증을 수행하기 위해서는 등록된 벡터와 인증 대상 벡터 간의 유사도 계산이 필요한데, [7]에서는 FHIPE를 이용하여 계산된 내적을 기반으로 유클리드 거리(의 제곱)을 계산하여 유사도를 계산하는 방법을 제안하였다. FHIPE를 이용해 k 차원의 두 벡터 $\boldsymbol{x}$, $\boldsymbol{y}$ 간의 유클리드 거리의 제곱인 $\| \boldsymbol{x} - \boldsymbol{y} \|^2$를 계산하기 위해서는 다음과 같은 세 연산을 정의해야 한다.

- *IPE.EncodeX*: k 차원의 벡터 $\boldsymbol{x}$로부터 $k+2$ 차원의 벡터 $\boldsymbol{x}' = (\| \boldsymbol{x} \|^2, -2x_1, -2x_2, \cdots, -2x_k, 1)$를 계산한 후, $\boldsymbol{x}'$과 msk를 입력으로 *IPE.LeftEnc*를 수행하여 암호문 ct_1을 계산한다.
- *IPE.EncodeY*: k 차원의 벡터 $\boldsymbol{y}$로부터 $k+2$ 차원의 벡터 $\boldsymbol{y}' = (\| \boldsymbol{y} \|^2, y_1, y_2, \cdots, y_k, 1)$를 계산한 후, $\boldsymbol{y}'$과 msk를 입력으로 *IPE.RightEnc*를 수행하여 암호문 ct_2를 계산한다.
- *IPE.Euclid*: 두 암호문 ct_1, ct_2와 공개 파라미터 pp를 이용하여 *IPE.Decrypt*를 수행하여 $\boldsymbol{x}'$과 $\boldsymbol{y}'$의 내적 $\langle \boldsymbol{x}', \boldsymbol{y}' \rangle$를 계산하여 출력한다. $\langle \boldsymbol{x}', \boldsymbol{y}' \rangle = \| \boldsymbol{x} - \boldsymbol{y} \|^2$이 성립하므로, *IPE.Euclid*는 두 벡터 $\boldsymbol{x}$와 $\boldsymbol{y}$의 유클리드 거리의 제곱을 출력하게 된다. 상기 관계가 성립한다는 것은 다음 전개식을 통해서 확인할 수 있다.

  $\langle \boldsymbol{x}', \boldsymbol{y}' \rangle = (\| \boldsymbol{x} \|^2 \times 1 - (x_1 \times y_1 + x_2 \times y2 +$

$$\cdots + x_k \times y_k + 1 \times \| \boldsymbol{y} \|^2)$$
$$= (\| \boldsymbol{x} \|^2 - 2\langle \boldsymbol{x}, \boldsymbol{y} \rangle + \| \boldsymbol{y} \|^2)$$
$$= \| \boldsymbol{x} - \boldsymbol{y} \|^2.$$

IV. 함수 은닉형 내적 암호 구현 동향

표 1은 최근에 발표된 FHIPE 구현 결과의 성능을 비교한 표이다. 표에서 k는 내적 계산 대상인 벡터의 차원이다. 문헌에 따라서는 *LeftEnc*, *RightEnc*가 *KeyGen* 및 *Encrypt*, 또는 반대로 *Encrypt* 및 *KeyGen*으로 표기되기도 하나, 본질적으로 이 두 연산은 기능상 서로 차이가 없는 대칭적인 연산으로 볼 수 있으므로 본 논문에서는 이들을 별도로 구분하지 않고 *LeftEnc*, *RightEnc*로 통일하여 표기하였다. 또한 *Setup* 연산은 오프라인으로 이루어지는 연산으로, 바이오 데이터의 등록이나 인증의 성능에 영향을 미치지 않으므로 비교 대상에서 제외하였다. 표에 비교된 결과 중 [7]은 FHIPE 스킴을 제안하면서 최초로 실용적인 결과를 제시한 논문이며, [8]은 또다른 FHIPE 스킴을 제안한 논문이다. 이 두 논문에서는 FHIPE 구현을 위해 PC 상에서 Python 언어를 이용하였다. 한편 [9]와 [10]는 [7]의 스킴을 C++ 언어로 구현한 결과이다. 이 중 [9]는 PC가 아닌 스마트 디바이스 상에서의 구현이며, 표에 제시된 실험치는 각각 *LeftEnc*, *RightEnc*, *Decrypt* 대신 이들을 활용하여 바이오인식을 위한 계산까지 수행하는 *EncodeX*, *EncodeY*, *Euclid* 연산의 결과이다. 다만, 이들은 벡터의 차원이 k 대신 $k+2$가 된다는 점 이외에는 *LeftEnc*, *RightEnc*, *Decrypt*와 계산량의 차이는 미미하다. 한편, [11]에서 구현한 스킴은 함수의 결과로 내적값을 반환하는 대신, 내적이 0인지의 여부, 즉 직교성(orthogonality)만을 판단하는 함수암호이다. 이 스킴은 PC 상에서 C++로 구현되었으며, 두 벡터가 수직일 경우 true, 수직이 아닐 경우 false를 반환한다. 위 다섯 가지 구현 결과에서 확인할 수 있듯이, 최적화의 방향에 따라 차이는 있을 수 있지만 대부분의 경우에는 FHIPE의 연산들 중 Decrypt 연산이 가장 시간이 많이 소모된다. 그러나 바이오인증에서 주로 많이 사용되는 128차원 특성벡터를 고려할 경우에도 Encrypt 연산과 Decrypt 연산을 합쳐 1초 이내의 시간이 소모되므로, 함수암호를 적절히 활용하면 실시간 원격 바이오인증 응용을 효과적으로 구현할 수 있음을 확인할 수 있다.

	k	*LeftEnc*	*RightEnc*	*Decrypt*
[7]	10	1.9 ms	7.5 ms	40.1 ms
	30	4.1 ms	21.1 ms	112.3 ms
	50	6.6 ms	34.9 ms	194.4 ms
	100	14.5 ms	71.4 ms	366.4 ms
[8]	10	43.8 ms	3.3 ms	109.9 ms
	30	66.3 ms	7.0 ms	232.5 ms
	50	123.1 ms	11.2 ms	344.4 ms
	100	391.3 ms	23.2 ms	638.5 ms
[9]*	10	15.7 ms	150.7 ms	46.2 ms
	30	12.3 ms	153.6 ms	63.3 ms
	50	29.7 ms	389.2 ms	143.1 ms
	100	46.9 ms	623.5 ms	232.4 ms
[10]	16	1.3 ms	7.6 ms	60.3 ms
	32	2.5 ms	14.6 ms	128.7 ms
	64	4.92 ms	28.85 ms	274.7 ms
	128	10.0 ms	57.5 ms	589.2 ms
[11]	16	6 ms	2 ms	10 ms
	32	12 ms	4 ms	19 ms
	64	22 ms	9 ms	37 ms
	128	46 ms	20 ms	73 ms

* [9]의 성능은 스마트 디바이스 상에서 측정한 *EncodeX*, *EncodeY*, *Euclid*의 성능임

표 1. 함수 은닉형 내적 암호 구현들에 대한 성능 비교

ACKNOWLEDGMENT

이 논문은 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(No. 2017R1D1A1A09000915) 및 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구(No.2017-0-00380, 차세대 인증 기술 개발)의 결과임

[참고문헌]

[1] 전명근, “생체정보 개인정보보호 현황 및 전망”, KISA Report, vol.10, pp21-29, 2018.

[2] N.K Ratha, J.H Connell, R.M Bolle, “Enhancing Security and privacy in biometrics based authentication systems,” IBM systems journal, pp. 614-634, 2011.

[3] A. Juels., M. Wattenberg, “A Fuzzy Commitment Scheme,” CCS ‘99 Proceedings of the 6th ACM conference on Computer and communications security, ACM CCS ‘99, pp. 28-36, Nov. 1999.

[4] D. Boneh, A. Sahai, B. Waters, “Functional Encryption: Definitions and Challenges,” Theory of Cryptography, TCC 2011, pp 253-273, Mar. 2011.

[5] M. Abdalla, F. Bourse, A.D. Caro, D. Pointcheval, “Simple Functional Encryption Schemes for Inner Products,” Public-Key Cryptography, PKC 2015, pp. 733-751, Mar. 2015.

[6] A. Bishop, A. Jain, L. Kowalczyk, “Function-Hiding Inner Product Encryption,” Advances in Cryptology, ASIACRYPT 2015, pp. 470-491, Nov. 2015.

[7] S. Kim, K. Lewi, A. Mandal, H.W. Montgomery, A. Roy, D.J. Wu, “Function-Hiding Inner Product Encryption Is Practical,” Security and Cryptography for Networks, SCN 2018, pp. 544-562, Sep. 2018.

[8] 전성윤, 임종혁, 이문규, “함수 암호 기반의 바이오인증 구현 및 성능 평가,”한국통신학회 학술대회논문집, Vol.2019, No.1, 2019

[9] 전성윤, 임종혁, 이문규, “효율적인 프라이버시 보호형 응용을 위한 함수암호의 구현 및 성능 평가,”한국정보과학회 하계컴퓨터통신 워크샵, 2019

[10] Q. Zhao, Q. Zeng, X. Liu, “Improved Construction for Inner Product Functional Encryption,” Security and Communication Networks, vol. 2018, Article ID 6561418, 12 pages, 2018.

[11] M. Barbosa, D. Catalano, A. Soleimanian, B. Warinschi, “Efficient Function-Hiding Functional Encryption: From Inner-Products to Orthogonality,” Topics in Cryptology - CT-RSA 2019, pp. 127-148, February 2019.

해시 기반 서명 기법을 이용한 티켓 기반 서비스 제공 방법

지청민*, 김지민*, 홍만표**

*아주대학교 컴퓨터공학과, **아주대학교 사이버보안학과

Ticket-based service provision method using hash-based signature scheme

Cheongmin Ji*, Jimin Kim*, Manpyo Hong**

*Department of Computer Engineering, Ajou University,

**Department. of Cyber Security, Ajou University

요 약

최근 양자컴퓨터 기술의 발전과 NIST의 양자 내성 암호 공모전으로 인해 해시 기반 서명 기법에 관한 관심이 높아지고 있다. 해시 기반 서명 기법은 오직 해시 함수의 안전성에 의존하는 서명 기법으로서, 하나의 공개키/개인키 쌍으로 가능한 서명 횟수가 제한적인 단점을 가진다. 본 논문에서는 이러한 해시 기반 서명 기법의 단점을 활용하여 제한된 횟수만큼의 사용을 보장하는 티켓 기반 서비스 제공 방법을 제안한다. 또한, 기반이 되는 해시 기반 서명 기법 및 매개변수의 선택에 따른 최대 사용 가능 횟수, 보안 수준, 티켓의 크기 등을 분석한다.

I. 서론

최근 양자컴퓨터 기술의 발전과 NIST의 양자 내성 암호(Quantum-Resistant Cryptography) 공모전으로 인해 양자 컴퓨터 상용화 이후에도 안전할 것으로 여겨지는 암호 기법에 관한 관심이 높아지고 있다. 2019년 10월에는 구글에서 54큐비트 Sycamore 양자 프로세서[1]를 발표하여 양자 우월성(Quantum supremacy)의 실현이 머지않았음을 보였으며, 2019년 1월 발표된 NIST 양자 내성 암호 공모전 Round 2 후보군에 SPHINCS+[2]가 포함되어 해시 기반 서명 기법의 활용에 대한 기대가 높아지고 있다.

해시 기반 서명은 격자 기반, 코드 기반, 다변수 기반 서명과 함께 대표적인 양자 내성 암호로서[3] 오직 해시 함수의 collision resistance 문제에 기반한다. 양자컴퓨터가 상용화될 경우 현재 널리 쓰이고 있는 공개키 암호 방식이 Shor의 알고리즘[7]에 의해 무력화되는 것과 달리 해시 기반 서명은 Grover 알고리즘[8]을 활용하더라도 기존 대비 절반 수준의 보안 강도를 보인다[9]. 그러나 해시 기반 서명은 다른 서명 방식보다 서명의 크기가 크고 하나의 공개키/개인키 쌍으로 서명할 수 있는 메시지의 개수가 제한적($1\sim2^{60}$)이다. 더욱이 더 많은 메시지에 서명하기 위해서는 서명/검증 비용 및 서명의 크기가 더 커지는 trade-off가 존재한다.

본 논문에서는 이러한 해시 기반 서명 기법의 단점인 제한적 서명횟수의 특성을 활용하여 제한된 횟수만큼의 사용을 암호학적으로 보장하는 티켓 기반 서비스 제공 방법을 제안한다. 제안하는 방법은 어떤 해시 기반 서명 기법 및 매개변수를 선택하는가에 따라 최대 사용 가능 횟수, 보안 수준 등이 달라질 수 있다.

본 논문의 구성은 다음과 같다. II. 장에서는 주요 해시 기반 서명 기법에 관해 서술하고, III. 장에서는 해시 기반 서명 기법을 활용한 티켓 기반 서비스 모델을 제시한다. IV. 장에서는 구체적인 서비스 제공 방법을 설명하며, V. 장에서는 해시 기반 서명 기법별 성능 분석, 마지막으로 VI. 장에서 결론을 서술한다.

II. 관련 연구

해시 기반 서명 기법은 격자 기반, 코드 기반, 다변수 기반 암호와 함께 대표적인 양자 내성 암호로서[3] 1979년 L. Lamport가 제안한 L-OTS(Lamport One-Time Signature) [4]가 최초의 해시 기반 서명 기법이다. 이후 R. Winternitz가 L-OTS의 서명 크기를 줄인 W-OTS[5]를 제안하였으며, L-OTS와 W-OTS는 해시 기반 서명 기법 중에서도 단 한 번의 서명만 가능한 One-Time Signature로 분류된다[6].

2001년에는 A. Perrig가 해시 충돌을 이용한 새로운 방식의 서명 기법인 BiBa(Bins and Balls)[10]를 제안하였고, 다음 해 L. Reyzin 등이 BiBa를 개선한 새로운 방식의 서명 기법인 HORS(Hash to Obtain Random Subset)[19]를 제안하였다. HORS는 TV-OTS[11], TSV[12], HORSIC[13], SPHINCS+의 FORS(Forest of Random Subset), Gravity-SPHINCS의 PORS(PRNG to Obtain

Random Subset) 등의 여러 후속 연구에서 각 기법의 빌딩 블록 및 기반으로 활용된다. BiBa 및 HORS는 매개변수의 적절한 조정을 통해 특정 보안 수준 이상을 유지하면서 r번의 서명이 가능한 Few-Time Signature로 분류된다[6]. 각 논문에서 제시하는 적절한 매개변수 설정을 통해 하나의 공개키/개인키 쌍으로 약 1~10회 내외의 서명이 가능하다.

OTS 또는 Few-Time Signature를 기반으로 실용적으로 사용될 수 있을 만큼의 서명횟수를 지원하는 서명 기법 또한 활발히 연구됐다. 1979년 R. C. Merkle이 MSS(Merkle Signature Scheme)[5]을 제안한 이래, CMSS[17], GMSS[18], XMSS family[14, 15, 16] 등의 Stateful 및 SPHINCS+ 등의 Stateless 서명 기법이 제안되었다. 이들은 모두 Many-Time Signature[6]로 분류되며, 매개변수 설정에 따라 한 번의 키 생성으로 적게는 2^{20}회에서 많게는 2^{60}회까지의 서명이 가능하다.

III. 기본 티켓 모델

3.1 티켓 기반 서비스 제공 모델

그림 1은 본 논문에서 제안하는 티켓 기반 서비스 제공 모델을 나타낸다. 사용자(user)는 티켓 발급자(ticket issuer)에게 인증을 거친 뒤 적절한 티켓을 발급받는다. 추후 사용자는 해당 티켓을 서비스 제공자(service provider)에게 전송하고 그에 맞는 적절한 서비스를 제공받는다. 티켓 발급은 최초 한 번이며, 티켓 사용은 최초 티켓 발급 시 정해진 횟수만큼 사용될 수 있다. 티켓 사용 횟수는 사용자와 서비스 제공자가 각자 기록하며, 상대방의 기록은 신뢰하지 않는다.

3.2 가정 사항

티켓 발급자는 사전에 사용자의 ID 및 패스워드를 안전하게 등록받으며, 그림 1의 ①과 ②는 안전한 채널로 이루어진다고 가정한다. 티켓 발급자의 공개키 PK_{TI} 또한 서비스 제공자에게 사전에 안전한 채널로 전달한다고 가정한다. 그림 1의 ③에서 사용자가 서비스 제공자에게 티켓 전송 이후 ④에서 어떻게 서비스를 제공하는지에 대해서는 다루지 않는다.

3.3 티켓 요구사항

티켓은 다음과 같은 요구사항을 가진다.

1) 무결성: 정당한 사용자가 정당한 티켓 발급자에게 인증을 거쳐 발급받은 티켓만 유효하다.
2) 재사용 가능: 사용자는 티켓이 처음 발급될 때의 정해진 사용 횟수만큼 해당 티켓을 사용할 수 있어야 한다.
3) 재전송 공격 방지: 이전에 사용된 티켓을 악의적 사용자 또는 공격자가 재사용할 경우 서비스 제공자는 이를 확인할 수 있어야 한다.
4) 부인 방지: 사용자와 서비스 제공자 간에 서로의 티켓 사용 횟수를 증명할 수 있어야 한다.

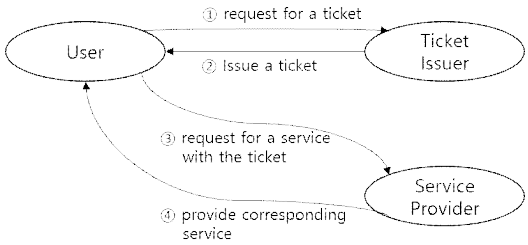


그림 1. 티켓 기반 서비스 제공 모델

IV. 티켓 기반 서비스 제공 방법

3.1 티켓 발급 요청

사용자는 티켓 발급 요청 이전에 사전에 선택된 해시 기반 서명 기법을 사용하여 공개키/개인키 쌍(PK_{user}/SK_{user})을 생성한 뒤 아래와 같이 티켓 발급 요청 REQ_{Ticket}을 티켓 발급자에게 전송한다. ID_{user}는 user의 고유 식별 아이디, PK_{user}는 사용자가 사전에 생성한 공개키, Φ는 사용자가 요청하고자 하는 서비스 정보, r은 티켓의 사용 횟수를 의미한다.

$$REQ_{Ticket} = (ID_{user}, PK_{user}, \Phi, r) \quad (1)$$

3.2 티켓 발급

티켓 발급자는 사용자가 티켓을 발급받을 권한을 가지고 있다면 아래와 같이 티켓 요청에 대한 응답으로 티켓 T를 전송한다. T에는 사용자의 ID 및 공개키, 제공 가능한 서비스 정보 및 티켓의 사용 가능 횟수, 그리고 이 모든 데이터를 티켓 발급자의 개인키로 서명한 σ_{TI}가 포함된다.

$$\sigma_{TI} = Sig_{SK_{TI}}(ID_{user}, PK_{user}, \Phi, r) \quad (2)$$

$$T = (ID_{user}, PK_{user}, \Phi, r, \sigma_{TI}) \quad (3)$$

3.3 서비스 요청 및 제공

사용자는 티켓 T를 수신한 뒤, $i(0 < i \leq r)$번째 서비스 요청 시 서비스 제공자에게 아래와 같이 $REQ_{Service}$를 전송한다. $\sigma_{user,i}$는 사용자가 티켓 T와 i를 자신의 개인키 SK_{user}로 서명한 것을 나타낸다.

$$\sigma_{user,i} = Sig_{SK_{user}}(T, i) \quad (4)$$

$$REQ_{Service} = (T, i, \sigma_{user,i}) \quad (5)$$

서비스 제공자는 수신된 $REQ_{Service}$에서 먼저 T가 정당한 티켓인지 확인하기 위해 σ_{TI}를 통해 T에 대한 서명 검증을 수행한다. 검증을 통과할 경우, $\sigma_{user,i}$를 통해 T에 포함된 사용자의 공개키 PK_{user}를 대상으로 검증을 수행하여 티켓이 정당한 사용자에 의해 제시된 것인지 검증한다. 두 번째 검증을 통과하면 서비스 제공자는 T에 포함된 ID_{user}를 통해 이전에 해당 사용자에게 서비스를 제공한 기록이 있는지를 자신의 데이터베이스에서 확인한다. 이전에 서비스를

분류	기법	공개키 크기	서명 크기	키 생성 비용	서명 비용	검증 비용	$REQ_{Service}$ 크기	서명 횟수	보안 수준
OTS	L-OTS	1,600	800	160	80	80	>2,438	1	2^{80}
FTS	BiBa	10,000	110	1024	2048	23	>10,110	10	2^{80}
	HORS	10,000	130	1024	1	14	>10,130	1	2^{80}
FTS + hash chain	HORSIC	10,000	80	13K	≈2	22	>10,080	1~665	2^{80}
		10,000	80	16K	≈2	41	>10,080	1~325	2^{80}
MTS	XMSS^{MT}	64	5,605	$2{,}535\times10^6$	12,417	2,318	>5669	2^{40}	2^{256}
		64	18,469	309,504	4,170	9,152	>18,533	2^{40}	2^{256}
		64	8,392	$3{,}803\times10^6$	13,417	3,477	>8,456	2^{60}	2^{256}
		64	27,688	464,256	4,170	13,728	>27,752	2^{60}	2^{256}
	SPHINCS$^+$ -256	64	29,792/ 49,216	68M~1,210M CPU cycles	1,558M~ 13,842M CPU cycles	20M~41M CPU cycles	>29,856/ 49,280	2^{64}	2^{256}

표 1. 해시 기반 서명 기법별 특징(크기의 단위는 byte, 비용은 별도의 표기가 없는 경우 해시 함수의 수행 횟수를 의미)

제공한 기록이 없다면 해당 사용자의 ID_{user}와 $\{\sigma_{user,i}, i\}$ 쌍을 새로이 데이터베이스에 기록하고, 이전 기록이 존재한다면 해당 사용자의 기록에서 기존에 제출된 서명 및 티켓 사용 횟수 $\{\sigma_{user,j}, j\}$를 확인하여 현재 제출된 T의 재전송 여부 및 사용 횟수를 확인(아래 (6) 참조)한다. 마지막으로, 모든 검증을 통과한다면 $\{\sigma_{user,i}, i\}$를 데이터베이스에 업데이트하고 사용자에게 Φ에 해당하는 서비스를 제공한다.

$$(i = j+1)\,\wedge\,(i < r) \tag{6}$$

V. 성능 분석

4.1 해시 기반 서명 기법별 특징

본 논문에서 제시한 티켓 기반 서비스 제공 방법에서는 사용자와 티켓 발급자가 각각 개인키/공개키 쌍을 생성한다. 사용자의 개인키/공개키 쌍은 2장에서 서술한 해시 기반 서명 기법을 이용하여 생성하는 것이 적절하며, 티켓 발급자의 개인키/공개키 쌍은 티켓의 사용 횟수, 티켓 발급의 규모, 보안 요구사항 등을 고려하여 RSA, ECDSA 등의 전통적인 비대칭키 암호 기법을 사용하거나 XMSS, SPHINCS+ 등의 해시 기반 서명 방식을 사용할 수 있을 것이다. 본 논문에서는 사용자의 전자 서명 기법에만 초점을 둔다.

표 1은 본 논문에서 제안하는 티켓 기반 서비스 제공 모델에서 활용될 수 있는 주요 해시 기반 서명 기법의 특징을 나타낸다. BiBa, HORS 등의 Few-Time Signature 기법은 일반적으로 약 1~10회의 서명만이 가능하나 서명 생성 및 검증의 계산 비용이 매우 낮다. HORSIC은 낮은 비용으로 수 백회에서 수천 회 수준의 서명이 가능하지만, 해시 체인의 소모 패턴이 불규칙적이므로 최소 서명 가능 횟수를 예측할 수 없어 티켓에 활용하기에는 비효율적이다. XMSS 및 SPHINCS+는 현실적으로 최대 2^{60}회 수준의 서명이 가능한 반면, 키 생성 비용이 매우 큰 단점을 보인다. 다만 SPHINCS가 stateless 기법이므로 티켓 기반 애플리케이션에서는 장점이 있다. $REQ_{Service}$의 크기는 선택된 해시 기반 서명의 공개키 및 서명의 크기에 영향을 받으며, 기법별로 최소 2~50KB 수준의 크기를 가진다.

4.2 보안성 분석

본 절에서는 3.3에서 제시한 각 티켓의 요구사항별 보안성에 관해 서술한다.

(1) 무결성: T의 무결성은 티켓 발급자의 서명 σ_{TI}를 통해 검증된다. 임의의 공격자는 티켓 발급자의 개인키를 알 수 없으므로 서명을 생성해낼 수 없다.

(2) 재사용 가능: $REQ_{Service}$ 전송을 위해 사용자는 자신의 개인키를 소모하게 되므로 해시 기반 서명의 특성에 따라 사전에 설정된 서명 가능 횟수 내에서 정당한 서명을 생성해낼 수 있으며, 서비스 제공자 또한 $\{\sigma_{user,i}, i\}$ 값을 통해 정확히 r회의 서비스 제공을 할 수 있다.

(3) 재전송 공격 방지: 임의의 공격자가 사용자가 전송한 $REQ_{Service}$ 를 도청하여 재전송하더라도, 서비스 제공자는 조건 (6)을 통해 재전송 여부를 확인할 수 있다. 사용자 역시 재전송 공격이 불가능하다.

(4) 부인 방지: 사용자가 사용 횟수 i를 부인하더라도 서비스 제공자가 기존에 사용자로부터 받은 $\sigma_{user,i}$를 제시하면 사용자의 티켓 사용 사실이 증명된다. 반대로 서비스 제공자 또한 사용자가 $\sigma_{user,i}$를 전송하지 않는 한 사용 횟

수를 i 이상 거짓으로 늘릴 수 없다.

VI. 결론

본 논문에서는 해시 기반 서명 기법을 이용한 티켓 기반 서비스 제공 방법에 대해 다루었다. 특히 해시 기반 서명의 단점인 제한된 사용 횟수를 이용하여 티켓으로 활용하기 위한 한 방법을 제시하였으며, 주요 해시 기반 서명 별 특징 및 티켓 기반 서비스 모델로의 적합성에 대해 논의하였다. 본 연구 및 추후 연구를 통해 해시 기반 서명 기법의 응용 분야가 확대될 수 있을 것으로 기대된다.

Acknowledgement

본 연구는 보건복지부의 재원으로 한국보건산업진흥원의 보건의료기술연구개발사업 지원에 의하여 이루어진 것임(과제고유번호 : HI18C0316)

[참고문헌]

[1] ARUTE, Frank, et al. Quantum supremacy using a programmable superconducting processor. Nature, 2019, 574.7779: 505-510.

[2] BERNSTEIN, Daniel J., et al. SPHINCS. 2017.

[3] CHEN, Lily, et al. Report on post-quantum cryptography. US Department of Commerce, National Institute of Standards and Technology, 2016.

[4] LAMPORT, Leslie. Constructing digital signatures from a one-way function. Palo Alto: Technical Report CSL-98, SRI International, 1979.

[5] MERKLE, Ralph C. A certified digital signature. In: Conference on the Theory and Application of Cryptology. Springer, New York, NY, 1989. p. 218-238.

[6] ZYCH, Mateusz Dominik. Quantum Safe Cryptography Based on Hash Functions: A Survey. 2018. Master's Thesis.

[7] SHOR, Peter W. Algorithms for quantum computation: Discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. Ieee, 1994. p. 124-134.

[8] GROVER, Lov K. A fast quantum mechanical algorithm for database search. arXiv preprint quant-ph/9605043, 1996.

[9] 서화정, 차세대 양자 내성 암호와 NIST 양자 내성 암호 2단계 공모전 동향, ITFIND 주간기술동향 1911호, 2019

[10] PERRIG, Adrian. The BiBa one-time signature and broadcast authentication protocol. In: Proceedings of the 8th ACM conference on Computer and Communications Security. ACM, 2001. p. 28-37.

[11] WANG, Qiyan, et al. Time valid one-time signature for time-critical multicast data authentication. In: IEEE INFOCOM 2009. IEEE, 2009. p. 1233-1241.

[12] LI, Qinghua, et al. Multicast authentication in the smart grid with one-time signature. IEEE Transactions on Smart Grid, 2011, 2.4: 686-696.

[13] LEE, Jaeheung, et al. HORSIC: An efficient one-time signature scheme for wireless sensor networks. Information Processing Letters, 2012, 112.20: 783-787.

[14] BUCHMANN, Johannes, et al. XMSS-a practical forward secure signature scheme based on minimal security assumptions. In: International Workshop on Post-Quantum Cryptography. Springer, Berlin, Heidelberg, 2011. p. 117-129.

[15] HÜLSING, Andreas, et al. Forward secure signatures on smart cards. In: International Conference on Selected Areas in Cryptography. Springer, Berlin, Heidelberg, 2012. p. 66-80.

[16] HÜLSING, Andreas, et al. Mitigating multi-target attacks in hash-based signatures. In: Public-Key Cryptography-PKC 2016. Springer, Berlin, Heidelberg, 2016. p. 387-416.

[17] BUCHMANN, Johannes, et al. CMSS-an improved Merkle signature scheme. In: International Conference on Cryptology in India. Springer, Berlin, Heidelberg, 2006. p. 349-363.

[18] BUCHMANN, Johannes, et al. Merkle signatures with virtually unlimited signature capacity. In: International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2007. p. 31-45.

[19] REYZIN, Leonid; REYZIN, Natan. Better than BiBa: Short one-time signatures with fast signing and verifying. In: Australasian Conference on Information Security and Privacy. Springer, Berlin, Heidelberg, 2002. p. 144-153.

공동소유 환경을 제공하는 새로운 ID 기반 프록시 재암호화+

김원빈*, 이임영**

순천향대학교 컴퓨터학과

New ID-Based Proxy Re-Encryption Provides a Co-Owned Environment

Won-Bin Kim*, Im-Yeong Lee**

Dept of Computer Science and Engineering, Soonchunhyang University*,**

요 약

현재, 프록시 재암호화 기술은 다양한 영역에 이용되고 있다. 일반적으로 프록시 재암호화는 데이터 소유자가 데이터를 암호화하여 프록시에 보관하고, 제 3자인 요청자가 해당 데이터를 요청할 때 요청자에게 복호화 권한을 발급하는 형태를 보인다. 따라서 데이터의 소유자와 요청자는 개인의 형태로 운영된다. 하지만 기업, 단체 등 다수의 참여자가 존재하는 환경에서는 데이터의 소유자가 여러 명인 공동소유 환경을 보일 수 있다. 이러한 공동소유 환경에서는 공동소유자 그룹의 멤버가 언제든지 추가, 제거될 수 있기 때문에 암호화된 데이터의 복호화, 재암호화를 수행할 수 있는 권한이 언제든지 변경될 수 있음을 고려해야 한다. 이에 따라 본 연구에서는 N:1 프록시 재암호화 환경에서 그룹 멤버의 추가, 제거가 가능한 새로운 방식의 프록시 재암호화 형태를 제안한다.

I. 서론

프록시 재암호화는 프록시 서버에 데이터를 암호화 하여 보관한 뒤, 제 3자의 요청에 따라 암호화된 데이터를 변환하여 제 3자가 복호화 할 수 있도록 권한을 부여하는 기술이다. 따라서 클라우드를 이용한 데이터 공유 및 전달에 매우 유용하게 이용될 수 있다.

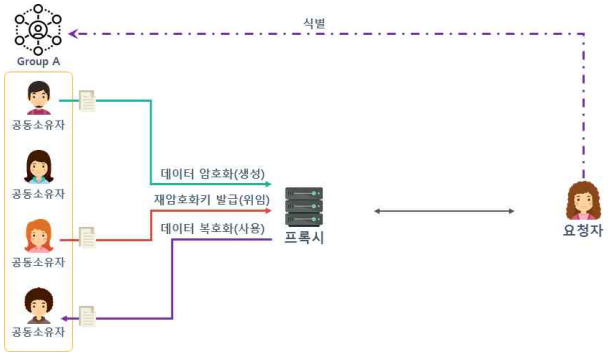
ID 기반 프록시 재암호화는 상대방의 ID를 공개키로 이용하거나 ID를 이용하여 공개키를 유도하여 상대방을 식별할 수 있는 프록시 재암호화 기법이다[1]. 따라서 ID 기반 프록시 재암호화는 소유자와 요청자가 1:1로 서로를 식별하고, 데이터를 주고받는 환경을 가정한다. 하지만 ID 기반 프록시 재암호화 환경에서 데이터의 소유자가 1명이 아닌 여러 명인 환경이 제시될 수 있다. 예를 들어, 기업 내 특정 부서, 단체, 군사환경 등은 소속의 특징에 따라 구성원 여럿이 동시에 데이터의 소유를 가질 수 있다. 이러한 환경에서는 공동소유자가 언제든지 추가·제거 될 수 있으며, 추가·제거된 사용자에게 기존 데이터의 접근을 허용·차단 할 수 있는 방법이 필요하다. 본 연구에서는 이러한 문제를 해결하기 위해 하나의 데이터를 여러명이 공동소유하는 환경에서 프록시 재암호화를 통해 데이터를 제 3자에게 전달하는 새로운 환경을 제안하며, 이를 N:1 프록시 재암호화 환경이라 명명한다.

+ 이 논문은 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성 지원사업(IITP-2018-2015-0-00403) 및 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2019R1A2C1085718)

II. 관련 연구

2.1 프록시 재암호화

프록시 재암호화 기술은 Alice가 소유한 데이터를 프록시 서버가 재암호화 하여 Bob에게 전달하는 기술을 의미한다[2]. 프록시 재암호화 기술은 다양한 형태가 존재하며 본 연구에서는 ID Based 방식을 이용하여 수행되었다. ID



(그림 1) N:1 프록시 재암호화의 개요도

Based 방식은 위임자인 Bob의 ID를 Bob의 공개키로 이용하는 방식이며, 재암호화 키의 생성을 위해서는 Bob의 ID가 이용된다. 이후, Bob은 재암호화 데이터를 복호화하기 위해 자신의 개인키를 이용하여 복호화를 수행한다[3].

2.2 그룹 키

그룹 키는 특정 그룹에 속한 그룹 멤버들이 공통으로 가지는 키를 의미한다. 이러한 그룹 키는 그룹 리더·키 분배 센터의 키 분배 형태와 그룹 멤버들끼리 파라미터를 공유하여 키를 동의·합의하는 등 다양한 형태가 존재한다. 본 제안방식에서는 그룹 멤버들이 ID 기반 공개키 환경에서 서로 동일한 키를 유도하는 방법으로 X. Lv et al을 참고하였으며, 이를 응용하여 N:1 프록시 재암호화를 설계하였다[4].

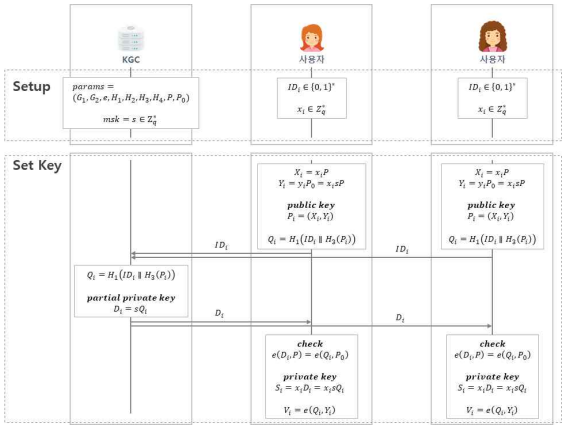
III. 보안 요구사항

본 제안방식에서는 프록시 재암호화 기술에 다음과 같은 사항을 요구한다.

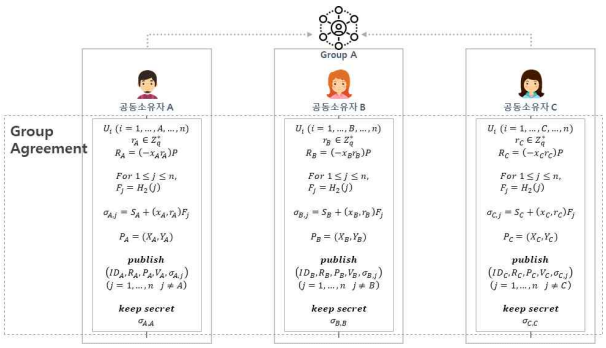
• 기밀성(Confidentiality) : 클라우드 스토리지에 업로드된 데이터는 정당한 소유자 이외에는 원본을 확인할 수 없도록 해야 한다.

• 단방향성(Unidirectional) : 소유자가 위임자의 암호문을 해독하지 못하면서도 소유자가 위임자에게 복호 권한을 위임할 수 있도록 해야 한다.

• 갱신가능(Renewable) : 공동소유 환경에서 그룹 멤버는 언제든지 추가·제거될 수 있다. 따라서 이러한 환경을 제공할 수 있는 환경을 고려해야 한다.



(그림 2) 초기화 단계



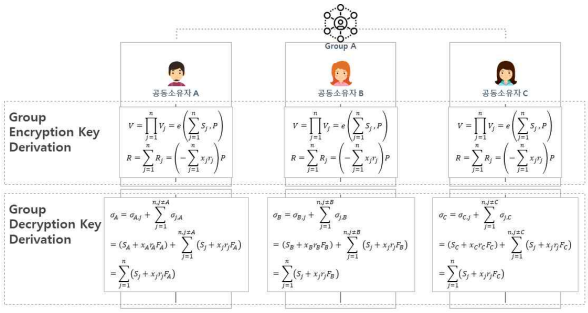
(그림 3) 그룹 형성 단계

IV. 제안방식

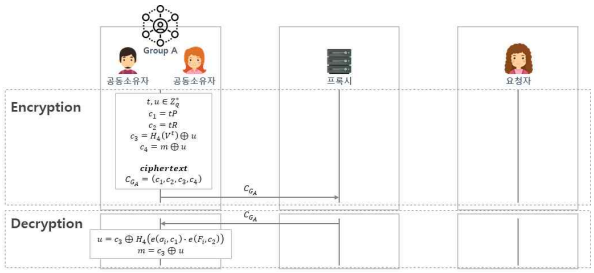
본 논문은 가입과 탈퇴가 자유로운 ID Based N:1 프록시 재암호화 방식을 제안하며, 각 방식은 초기화 단계(Setup, Set Key)(그림 2), 그룹 형성 단계(Group Agreement)(그림 3), 그룹 키 생성 단계(Group Encryption Key Derivation, Group Decryption Key Derivation)(그림 4), 암·복호화 단계(Encryption, Decryption)(그림 5), 재암호화 단계(ReKeyGen, Re-Encryption, Re-Decryption)(그림 6) 그리고 그룹 갱신 단계(Renew)(그림 7)로 구성되어 있다.

4.1 Setup

KGC(Key Generation Center)와 각 사용자는 시스템 이용을 위한 파라미터, ID를 생성한다(그림 2 참조).



(그림 4) 그룹키 생성 단계



(그림 5) 암·복호화 단계

4.2 Set Key

각 사용자는 공개키를 생성하고, 이를 이용하여 KGC에게 부분 개인키를 발급받는다. 이후 각 사용자는 부분개인키를 이용하여 개인키를 생성한다(그림 2 참조).

4.3 Group Agreement

그룹에 참여하고자 하는 사용자는 각자 공개·비밀 파라미터를 생성하고 이를 다른 공동소유자에게 전송한다(그림 3 참조).

4.4 Group Encryption Key Derivation

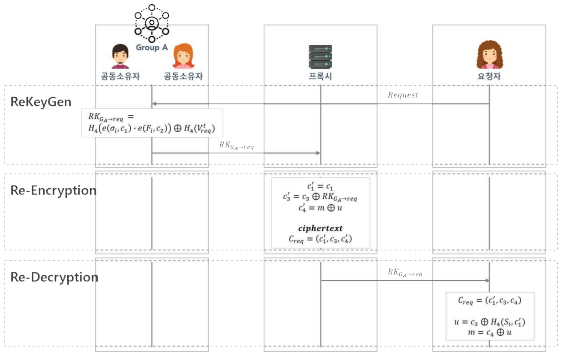
공동소유자는 다른 멤버에게 수신한 파라미터를 이용하여 그룹 암호화 키를 생성한다(그림 4 참조).

4.5 Group Decryption Key Derivation

공동소유자는 다른 멤버에게 수신한 파라미터를 이용하여 그룹 복호화 키를 생성한다(그림 4 참조).

4.6 Encryption

공동소유자는 그룹 암호화 키를 이용하여 원본 메시지 m의 암호문을 생성하고, 프록시에



(그림 6) 재암호화 단계

보관한다(그림 5 참조).

4.7 Decryption

공동소유자는 프록시에 보관된 암호문을 다운로드하여 그룹 복호화키로 복호화하여 원본 메시지 m을 획득한다.(그림 5 참조).

4.8 ReKeyGen

제 3자인 요청자가 공동소유자에게 데이터를 요청할 경우, 공동소유자는 그룹 복호화 키와 요청자의 공개키를 이용하여 재암호화키를 생성하고, 이를 프록시제 전송한다(그림 6 참조).

4.9 Re-Encryption

그룹으로부터 암호문, 재암호화키를 전송 받은 프록시는 재암호화를 수행하여 요청자에 대한 암호문을 생성한다(그림 6 참조).

4.10 Re-Decryption

프록시는 재암호화된 암호문을 요청자에게 전송하며, 요청자는 자신의 개인키를 이용하여 전송 받은 암호문을 복호화하고 메시지 m을 획득한다(그림 6 참조).

4.11 Renew

만약 공동소유자 그룹 멤버가 변경될 경우, 새로운 공동소유자 그룹의 암호화, 복호화 키를 생성하고, 이를 이용하여 재암호화키를 생성한다. 그리고 재암호화키를 프록시에 전송하여 기존 암호문을 새로운 공동소유자 그룹의 암호문으로 변경한다. 마지막으로 기존 암호문을 프록시에서 제거한다(그림 7 참조).

V. 제안방식 분석

• 기밀성(Confidentiality) : 클라우드 스토리지에 업로드 되는 데이터는 데이터 소유자가 자신의 공개키로 암호화를 수행한 뒤 업로드를 한다. 따라서 업로드된 데이터는 소유자의 개인키로만 복호화를 수행할 수 있다. 만약 소유자가 위임자의 공개키를 이용하여 재암호화 키를 생성하고, 암호문을 재암호화 할 경우, 위임자의 개인키로만 암호문을 복호화 할 수 있다.

• 단방향성(Unidirectional) : 소유자는 위임자의 공개키를 취득하여 재암호화 키를 생성할 수 있다. 하지만 공개키 암호화 방식의 특성상 위임자의 공개키를 이용하여도 위임자의 공개키로 암호화된 암호문을 복호화 할 수 없다.

• 갱신가능(Renewable) : 그룹 구성원이 변경될 경우, 변경된 그룹 멤버의 암호화 키 및 복호화키를 재생성한다. 재생성된 키와 기존 그룹의 키를 이용하여 기존 그룹의 암호문을 새로운 그룹의 암호문으로 변경할 수 있도록 재암호화를 수행한다. 이를 통해 기존 그룹의 암호문이 새로운 그룹의 암호문으로 변경되어 기존 그룹의 멤버의 접근을 차단하거나 새로운 그룹 멤버의 접근을 허용하게 된다.

VI. 결론

본 제안방식은 클라우드 스토리지 환경에서 데이터를 보다 효율적으로 공유하기 위한 기술을 다룬다. 또한 이를 위해 기존의 1:1 프록시 재암호화 환경을 개선하여 N:1 방식의 프록시 재암호화 환경을 제안하였다. 이 환경은 여러 공동소유자가 하나의 데이터를 소유하는 환경으로, 그룹에 속한 멤버라면 누구든지 데이터를 생성하고, 재암호화 키를 생성할 수 있는 방법이다. 하지만 이 방법에서는 그룹 멤버인 공동소유자가 언제든지 추가·제거될 수 있기 때문에 그룹 멤버의 변경이 가능해야한다. 이를 위해 본 연구에서는 그룹키 생성방식과 프록시 재암호화의 특징을 결합하여 그룹 멤버의 변경 시 변경된 그룹의 키를 재생성하고, 이를 통해 기존 그룹의 암호문에서 새로운 그룹의 암호문으로 손쉽게 변경할 수 있는 방식을 제안하였다. 결과적으로 이를 통해 본 제안방식에서는 데이터 공동소유 환경에서 비교적 안전하고 효율적으로 프록시 재암호화를 통해 데이터를 공유할 수 있는 환경을 제공한다.

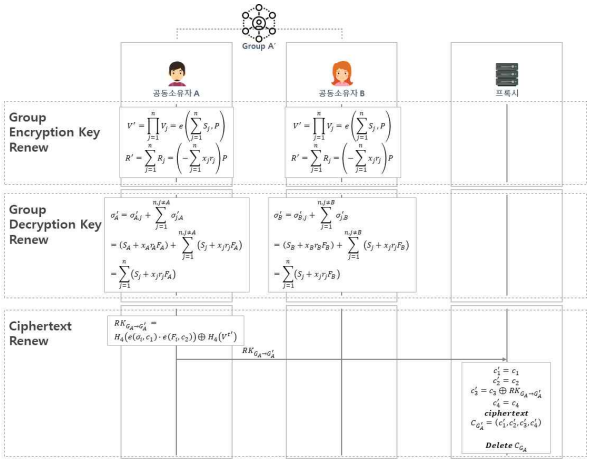
[참고문헌]

[1] Green, Matthew, and Giuseppe Ateniese. "Identity-based proxy re-encryption." Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2007.

[2] Chu, Cheng-Kang, and Wen-Guey Tzeng. "Identity-based proxy re-encryption without random oracles." International Conference on Information Security. Springer, Berlin, Heidelberg, 2007.

[3] Shao, Jun, and Zhenfu Cao. "Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption." Information Sciences 206 (2012): 83-95.Security Symposium, 2013.

[4] Lv, X., Li, H., & Wang, B. . Authenticated asymmetric group key agreement based on certificateless cryptosystem. International Journal of Computer Mathematics, 91(3), 447-460. 2014.



(그림 7) 그룹 갱신 단계

OSINT 기반 사이버 위협정보 수집 및 관리기술 연구

김민석, 강지훈*
*(주)디플랫폼

Research on cyber threat information data collection and management method using OSINT

Minseok Kim*, Jihun Kang*
*D-Platform Co., Ltd.
{mins, jhkang}@d-platform.co.kr

요 약

사이버 위협정보(Cyber Threat Information)는 지속적으로 지능화 및 증가하는 침해사고의 위협에서 잠재적인 악의적 활동을 식별할 수 있는 중요한 지표로 활용되어지고 있는 데이터이다. 본 논문에서는 접근에 제한적인 사이버 위협정보(Cyber Threat Information)에 대한 데이터 확보 및 연구를 위한 OSINT 기반의 수집 및 관리 기술을 소개한다.

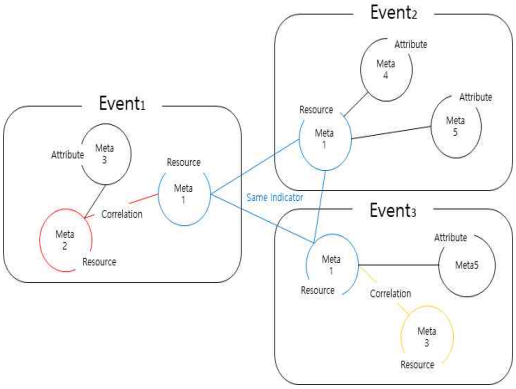
I. 서론

지능형 사이버 위협 대응(CTI; Cyber Threat Intelligence)은 조직 내부뿐만 아니라 외부 조직 등에서 발생한 위협들을 수집 및 분석하여, 내부의 보안 시스템에도 이를 통합, 적용시키는 것을 의미한다. 현재의 국내/외 사이버 위협은 지속적으로 증가하고 지능화되어지고 있으며 조직적으로 보다 체계적인 위협으로 발전하고 있다. 이에 따라 궁극적인 CTI(Cyber Threat Intelligence)를 위해서 국내/외 사이버 위협정보(Cyber Threat Information)에 대한 수집이 선행되어야 하며, 이를 위하여 시스템 구축과 사이버 위협정보 관리에 대하여 연구 해보고자 한다.

II. 사이버 위협정보 저장 구조

먼저 OSINT(Open Source Intelligence)[1]는 특정 몇 가지의 정보 영역을 표현하는 용어로 사용되어지지 않으나, 이 경우 사이버 위협정보에 대한 OSINT 영역으로 국한한다. 또한 시스템 개발에서 OSINT 기반의 사이버 위협정보를 언제, 어디서, 무엇을, 어떻게, 왜, 누가 수집할 것인지에 대한 고민이 필요한데, 이와 같은 관점에서 연구내용을 서술한다.

수집의 대상이 되는 사이버 위협정보는 침해지표(IoC; Indicator of Compromise)를 비롯하여 OSINT 사이트에서 제공하는 다양한 유형의 데이터들이 될 수 있으며, 대표적으로 IP Address, Domain, URL, Hash(악성코드에 대한 Binary 해시 MD5, SHA1, SHA256 값) 등이 해당한다. 시스템에서는 침해지표 데이터를 기준으로 확장된 사이버 위협정보 데이터베이스(Mongodb)를 구성하기 위하여 데이터 유형에 따라 세부 분류를 정의한다.



[그림 1] 사이버 위협정보 저장 구조 개념도

2.1 과거 이력 정보(Events)

OSINT를 통해 수집되는 사이버 위협정보를 과거 이력으로 보는 이유는 실제 침해사고 기준의 시점에서 보았을 때, OSINT 사이트상에서 실시간으로 게재되기 어렵기 때문이 가장 크다.



[그림 2] Events 단위 설정 예시(malc0de.com)

과거 이력 정보(Events)는 시스템에서 동일 OSINT 사이트를 통해 동일시점에 수집되는 IoC 데이터들의 묶음 또는 그룹 단위로 표현될 수 있다. [그림 2]와 같이 간단한 예로, 웹(Web) 상에서 특정 테이블 정보를 크롤링(Crawling) 한다면, 해당 테이블의 1줄(row)이 Events 단위 기준으로 생성된다.

[표 1] Events Collection

키 이름(Key)	Description
event_id	해당 event에 대한 고유 ID
ctime	수집 시스템의 수집 시간
dtime	탐지/분석 시간 또는 외부시스템(해당 채널)의 수집, 게시 시간
channel	수집 채널(Channel) 명
origin	수집 오리진(Origin) 명
seed	데이터 수집에 근거가 되는 원본 파일(html, csv, json 등) 저장 경로
ioc_list	event에 포함되는 IoC에 대한 ID(meta_id) 리스트

'event_id' 는 해당 과거 이력 정보(Events)를 저장할 데이터베이스에서 관리를 목적으로 고유하게 사용되어지며, 각 데이터의 문자열 정보를 기반으로 고유화(Unique)하여 사용하며, 'ctime' 과 'dtime' 은 데이터 수집과 관련된 날짜/시간형태의 데이터이다. UTC 표기를 별도로 하지 않은 시간/날짜 형태의 데이터에 대해서는 Whois 질의를 통해 해당 서버 로컬 시간대역을 국내 기준 시간(UTC+09:00)으로 적용하였다. 'channel' 과 'origin' 은 OSINT 사이트에 대한 분류인데 채널(channel)은 대부분 해당 OSINT 사이트 domain name을 기준으로 구분 짓는다. 또한 오리진(origin)은 동일 채널 내에서 수집 방식 및 event 유형에 따라 분류하여 정의한다. 'ioc_list' 는 해당 event를 통해 수집된 IoC에 대한 ID 정보인데, 사용자가 데이터베이스 질의 시 주로 IoC 데이터 범주 내에서 검색을 시작하기 때문에 확장 및 편의를 위해 해당 정보를 포함한다. 또한 동일한 유형 및 값을 가진 데이터는 Events 내 document에서 'ioc_list' Key값을 통해 지속적으로 동일한 'collect_id' 로 참조되며, 'label'의 값 중 'resource'에 해당하는(이 경우, 주로 검색 대상 자원) document가 참조된 횟수를 통해 특정 자원에 대한 과거 수집이력을 알 수 있다.

2.2 주요 자원(Resource) 및 속성(Attribute)

[표 2] 주요 자원 및 속성 구분 예시

구분	데이터 유형
주요 자원 (Resource)	ip
	domain
	url, uri
	hash(md5, sha1, sha256, ...)
	cve_code
속성 (Attribute)	port
	domain name
	isp
	asn
	path
	username
	password
	sld/tld
	filename
	file size
	cvss(v2, v3)

주요 자원과 속성을 구분하는 주요한 기준은 해당 자원을 침해지표와 같이 악성으로 볼 수 있는지에 대한 여부를 기준으로, 데이터베이스 상에서 아래의 [표 3]과 같은 형태로 Meta Collection 저장구조를 정의한다.

[표 3] Meta Collection

키 이름(Key)	설명
collect_id	해당 자원에 대한 고유 ID
label	주요 자원 분류(IoC와 같은 주요 침해 자원: Resource / 그 외 속성 정보: Attribute)
type	데이터 유형 분류(IP, Domain, SHA1, MD5, CVE-CODE, ...)
value	해당 문자열 값

앞서 서술한 과거 이력 정보(Events)의 묶음 내에서 각 데이터를 표현하는 문자열로 IP인 경우 데이터 유형(Type)을 'ip', 임의의 IP '192.168.0.1'와 같은 형태의 문자열 데이터를 값(Value)로, 해당 유형에 따라 주요 자원 분류(label)를 포함하여 [표 3]와 같은 형태로 저장한다.

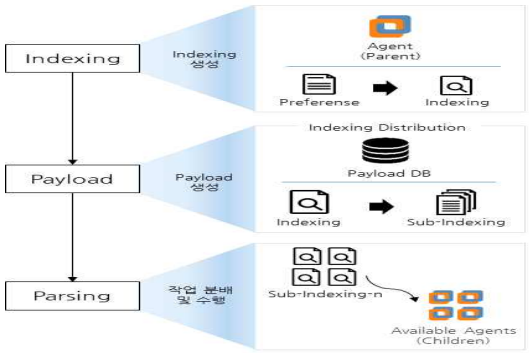
2.3 주요 자원의 연관관계(Correlations)

앞서 설명한 사이버 위협정보 데이터베이스 저장 구조의 주된 목적은 자원들의 문자열 값 저장(Meta) 및 특정 IoC의 과거 이력들(Events) 뿐만 아니라 여러 과거 이력들(Events) 거치며 해당 IoC가 특정 시점의 Event외에 다른 Event에서 기존에 Event에 없었던 새로운 IoC와의 연관관계를 생성하는 것을 포함한다. 이를 위하여 IoC간의 연관관계 표현이 필요하며 아래의 [표 4]와 같이 저장구조를 정의한다.

[표 4] Correlations Collection

키 이름(Key)	설명
correlation_id	연관관계 고유 ID
src_collect_id	해당 Meta 자원 ID
dst_collect_id	대상 Meta 자원 ID
relation_name	연관관계 명
des	추가 설명(description)

III. 사이버 위협정보 수집 과정



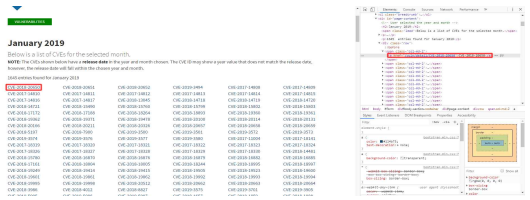
[그림 3] 데이터 수집 과정 개념도

사이버 위협정보 수집 시스템은 주로 크롤링(Crawling)기술 및 API 등을 활용하여 OSINT 사이트 내 정보를 수집을 목적으로 하며 데이터 수집 과정을 3가지 단계로, 작업 대상 및 순서 선정(Indexing), 작업 생성(Payload), 수집 수행(Parsing) 구성 된다.

3.1 작업 대상 및 순서 선정

작업 대상 및 순서 선정 목적은 데이터 수집을 위하여 작업 진행률(Events에서 설정한 단위 기준)을 파악하고, 수집할 수행할 작업량에 따라 분산처리를 위한 작업 분배를 위함이다. 또한 Indexing의 경우 Events 단위를 기준으로

[그림 4]와 같이 수집 대상이 되는 웹 페이지의 URL, 웹 페이지 내 테이블의 pagination 및 라인번호, 특정 데이터 파일 경로 등이 될 수 있다.



[그림 4] Indexing 대상 예시(nvd.nist.gov)

3.2 작업 생성

작업 대상 및 순서 선정을 통해 만들어진 Indexing은 Payload 데이터베이스를 통해 수집 작업을 생성하게 된다. 또한 생성된 indexing을 작업량에 따라 분할한다.

3.3 수집 수행

정의된 수집 엔진(Crawling 등)을 통해 실제 데이터를 파싱하는 과정을 수행하며, 이 때 사전에 생성된 작업을 [그림 3]에서와 같이 작업 생성 과정에서 sub-indexing으로 생성된 작업 대상으로 할당받아 처리한다. 또한 작업 오류 발생 시 해당 작업 중이던 payload 및 indexing에 오류를 기록한다. 또한 데이터베이스의 저장은 Indexing 단위(Events 단위)를 수행할 때 마다 [그림 1]의 하나의 Event와 같이 Events, Correlation, Meta 정보 객체(JSON)를 생성하고 하나의 Event를 저장하는 방식으로 수행하며 이를 데이터베이스(Mongodb, bson)에 저장 한다.

IV. 결론 및 향후 연구 방향

본 논문은 사이버 위협정보 수집 및 관리 기술을 위해 필요한 저장 구조와 수집 과정을 제안하고, 이를 통해 IoC들의 연관관계 및 과거이력들을 파악할 수 있도록 구성하였다.

본 논문에서 다루는 OSINT 기반의 사이버 위협 정보 수집의 궁극적인 목적인 CTI(Cyber Threat Intelligence)에 활용하기 위해서는 단편적인 데이터 파편으로서 보안제품의 차단 기준이 아닌 보다 복잡한 보안제품 통합 과정[3]이 필요하며, ATT&CK Matrix[4]와 같은 기반으로 사이버 위협정보가 Tactics, Techniques, Procedure 과정의 패턴 분석 등에 활용할 수 있다. 또한 연구에는 Mongodb 데이터베이스를 사용하였지만 연구 결과물의 성능과 시각적인 요소 등의 장점을 위해 그래프 데이터베이스[5]를 기반으로 구현할 수 있다.

[참고문헌]

[1] OSINT, https://en.wikipedia.org/wiki/Open-source_intelligence, Wikipedia

[2] IoC, Nate Lord, What are Indicators of Compromise?, https://digitalguardian.com/blog/what-are-indicators-compromise, DIGITALGUARDIAN, 2018.

[3] Jeff Berg, The Role of Cyber Threat Intelligence in Security Operations, https://www.fireeye.com/blog/products-and-services/2016/11/the_role_of_cyberth.html, FIREEYE, Nov, 2016.

[4] ATT&CK, B. E. Strom et al, “MITRE ATT&CK: Design and Philosophy,” MITRE White paper, Jul. 2018.

[5] 그래프 데이터베이스, 김낙현, 김병익, 이슬기, 조혜선, 박준형. 사이버 위협정보 분석을 위한 그래프 데이터베이스 성능 분석. 한국정보과학회 학술발표논문집, 1106-1107. 2017.

ACKNOWLEDGMENT

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-00158, 국가 차원의 침해사고 대응을 위한 사이버 위협 인텔리전스 분석(CTI) 및 정보 공유 기술 개발)

학생회비의 투명성 제고를 위한 PBFT 합의 기반의 회계 감사 시스템 아키텍처

김건오*, 김민혁*, 김선균*, 이소연*, 노시완**, 이경현*

*부경대학교 IT융합응용공학과

**부경대학교 일반대학원 정보보호학협동과정

An Audit System Architecture Based on the PBFT Consensus for the Improvement of Transparency in Student Union Fee

Geonoh Kim*, Minhyeok Kim*, SeonGyun Kim*, SoYeon Lee*, Siwan Noh and Kyune-Hyune Rhee

*Department of IT Convergence and Application Engineering, Pukyong National University.

**Interdisciplinary Program of Information Security Graduate School, Pukyong National University

요 약

학생회는 학생들이 자발적으로 납부하는 학생회비로 운영되는 학생자치조직이다. 하지만 폐쇄적인 학생회비 관리방식으로 학생회비 집행의 투명성에 관한 문제가 꾸준히 제기되어 왔다. 최근 블록체인 기술의 대두와 함께 블록체인의 투명성을 이용한 회계 관리 솔루션들이 제안되었으나 학생들의 납부에 의존적인 학생회비의 특성을 고려하지 않는 단순 회계관리 솔루션이거나 납부된 학생회비를 토큰화하여 블록체인에 기록하여 이후 토큰과 실물화폐를 교환해주는 거래소와 같은 신뢰개체의 참여를 필요로 하는 한계가 있었다. 본 논문에서는 학생회비 운영관리의 특성과 보안요구사항을 분석 및 정의하고 집행기록을 블록체인에 기록하는 대신 학과 참여구성원들의 합의 내역을 Pedersen commitment를 사용한 예산편성 찬반 투표 스마트컨트랙트와 PBFT 합의 기반 학생회비 집행내역 감사 기술을 이용하여 블록체인에 기록하는 학생회비 감사 시스템 아키텍처를 제안한다.

I. 서론

학생회는 학생이 주체가 되어 어떠한 일을 의논하고 결정하고 실행하는 학생자치조직으로서 일반적으로 학교의 모든 학생들을 대표하는 총학생회와 단과대학별 자치조직인 단과대학 학생회로 구성되어 있다. 학생회의 운영은 일부 교내 지원금을 제외하면 학생자치조직이므로 학생들이 납부하는 학생회비에 의존한다. 하지만 학생회비의 투명성과 납부의 자율성 및 과도한 납부금액 등으로 인해 많은 문제가 제기되고 있다[1].

대표적으로 학생회비 집행의 권한을 학생회 임원들의 자율에 맡겨 발생하는 투명성 문제가 있다. 2014년 고려대학교 전자·정보공학과 학생회장은 학생회비 및 학과점퍼 구입명목으로 학과 신입생 109명으로부터 총 2,181만원을 걷었으나 이중 1,600여만원을 개인적인 용도로 사용하다 적발되었고 2013년 인하대학교 생활과학대 학생회장은 학생회비 700만원 중 142만원을 회계에서 누락하고 이를 횡령하다 고발당하는 사건 등이 빈번하게 발생하여 학생회비 집행의 투명성 해결이 현안으로 떠오르고 있다.

이를 방지하기 위해 각 학교에서는 학교차원의 감사 제도를 도입하여 학생회비의 투명성을 제고하는 방안을 추진하고 있으나 감사인의 선정이 불투명하고 같은 학교 조직 내 참여구성원간의 상호감사로 감사 실효성의 의문이 제기되고 있다[2]. 최근에는 블록체인 기술의 탈중앙성과 투명성을 이용하여 학생회비 관리에 적용

하는 방법이 고려되고 있으나 블록체인 토큰으로 발행된 디지털자산과 명목화폐(fiat money)를 교환해주는 신뢰개체를 가정하여 실 구현에 한계를 지닌다[3].

본 논문에서는 블록체인을 이용하여 학생회비 집행의 투명성 보장과 함께 기존 명목화폐로 납부된 학생회비의 사용내역을 블록체인 토큰으로 블록체인에 기록하는 대신, 학과 내 학생들 중 선정된 감사인들과 학생회비 관리를 위한 모임통장을 개설하여 회비 이용 내역을 감사인들에게 공개하고 감사인들은 확인한 사실을 다른 감사인들에게 알려 최종적으로 감사인들 중 일정 수 이상의 합의가 이루어졌을 때 블록체인에 검증가능한 감사정보를 기록하여 최종적으로 학과 내 구성원 누구든지 학생회비 집행의 유효성을 확인할 수 있는 시스템 모델을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 학생회비의 일반적인 관리체계와 문제점을 분석하고 3장에서는 보안요구사항을 정의, 보안요구사항을 만족하는 시스템 모델을 제안하고 4장에서 결론을 맺는다.

II. 기존 방법 분석

학생회비 관리방법은 각 학교별 혹은 학생회별로 상이하며 일반적인 관리 방법이 존재하지 않는다. 이 장에서는 본교 학생회의 관리 방법 및 알려진 관리 방법들을 종합하여 기본적인 학생회비 관리방식을 정의하고 여기서 우리가 목표로 하는 학생회비 관리 방법을 위해 필요한 보안 요구사항들을 정의한다. 또한 학생회비 납부방식 및 금액에 대한 이슈 또한 운영의 투명성과 함께 큰 문제로 남아있지만 본 논문의 연구범위가 아니므로 논문에서는 언급하지 않는다.

2.1 학생회비 운영관리

학생회비는 당해 연도 예산안을 편성하고 심의·확정된 이후 실제 집행이 이루어진다. 학생회비 운영관리는 1년 단위로 이루어지며 본 논문에서는 학과 신입생들이 입학 시 4년 치의 학생회비를 납부하는 경우를 고려한다. 기본적인 학생회비 운영관리 과정은 다음과 같다.

학생회는 새로운 학년이 시작하기 전에 당해 연도의 예산안을 편성하고 이를 토대로 필요한 예산 및 신입생들이 납부해야할 학생회비를 정하고 이를 공지한다. 납부된 학생회비는 학생회비 운영을 위한 학생회 임원 명의의 신규 계좌를 개설하여 관리한다. 이후 예산안에 계획된 목적에 대해 집행 전 세부 집행 계획을 수립하고 학생회 임원들의 동의 후 이를 토대로 집행 후 증빙자료를 상위 조직에 제출한다. 이후 매 분기 혹은 일정 기간마다 집행 내역을 결산하여 공개하며 추가적으로 학생들의 요청시마다 증빙자료를 공개한다.

2.2 문제점 분석

2.1절에서는 기본적인 학생회비 운영관리 방법을 정의하였다. 이 절에서는 이러한 운영방법의 문제점을 지적하고 분석한다.

- **운영관리의 중앙화**: 학생회의 운영은 일부 학교의 지원을 제외하면 일반적으로 소속 학생들이 납부하는 학생회비에 의존하지만 예산안 편성 및 집행 계획 수립은 학생회 임원들에 의해서만 이루어진다. 때문에 필요 예산대비 과도한 예산 혹은 불필요한 목적의 예산 편성 및 집행은 집행 이후 결산과정이나 감사에서 적발되기도 한다.
- **운영관리의 투명성**: 학생회비의 투명성을 위해 학생회는 주기적으로 결산내역 및 증빙자료를 공개하거나 학생들의 요청시 이를 공개하지만 영수증 분실 혹은 증빙자료 조작 등 실제 집행이후 증빙자료 조작 및 은폐를 시도할 가능성이 있으며 학생회비 관리 계좌 역시 학생회 임원의 개인계좌와 같이 일부 구성원들만 확인할 수 있어 공개를 하더라도 이에 대한 신뢰성을 보장하기 어렵다.

이와 같은 문제는 학생회 임원들에 의해서만 운영관리가 이루어져 실제 학생회비를 납부한 학생들의 참여가 배제되어 발생하므로 본 논문에서는 블록체인을 이용하여 학생회에 집중된 권한을 분산하고 투명한 운영관리를 위한 방법을 제안한다.

III. 제안 시스템 모델

2장에서는 기존 학생회비 관리방법과 여기서 발생할 수 있는 문제점들을 정의하였다. 이 장에서는 이를 개선한 시스템 모델과 보안요구사항을 제시한다.

3.1 보안 요구사항

2.2절에서는 학생회비 운영관리에서 발생하는 문제점을 지적하였다. 이러한 문제점에 기반하여 제안 시스템 모델이 만족해야하는 보안요구사항을 다음과 같이 정의한다.

- **운영관리의 투명성**: 학생회비 집행과 관련된 모든 활동(예산안 편성, 집행)의 적절성을 참여구성원 모두 언제든지 학생회의 동의 없이 검증할 수 있어야한다.
- **탈중앙화된 운영관리**: 학생회비 집행과 관련된 모든 활동(예산안 편성, 집행)은 학생회 임원들의 독단으로 처리할 수 없고 반드시 학생들의 동의를 얻어야한다.

3.2 시스템 아키텍처

이 절에서 우리는 3.1절에서 정의한 보안요구사항을 만족하는 솔루션을 적용한 학생회비 관리 시스템의 모델을 제안한다.

운영관리의 투명성을 위해 우리는 블록체인을 적용할 수 있다. 본 논문에서는 명목 화폐로 집행된 내역을 블록체인에 토큰화하여 기록하는 대신, 예산안 편성의 적절성 및 예산 집행성과의 평가결과를 블록체인에 기록하여 누구든지 이 결과를 쉽게 검증할 수 있도록 한다. 제안 시스템에서는 명목화폐로 운영되는 학생회비 운영관리의 투명성을 위해 모임통장을 사용한다. 모임통장은 개인명의 통장이지만 설정한 멤버들이 실시간으로 통장명의자를 통하지 않고 직접 거래내역을 확인할 수 있다. 하지만 단순히 참여구성원 모두와 모임 통장을 개설하는 것은 비효율적이고 현실적이지 못하다. 제안 시스템에서는 이런 문제를 해결하기 위해 참여구성원 중에서 일정수의 감시자(Observer)를 선정하여 학생회 임원들과 모임통장을 개설하고 실제로 예산의 집행이 이루어진후 감시자들의 확인 결과를 블록체인에 감시자들간의 합의를 통해 기록한다.

탈중앙화된 운영관리를 위해 단순히 학생회가 편성한 예산안을 공지하고 이에 대한 이의신청을 받는 것으로 해결할 수 있지만 실제 예산안에 편성에 대한 학생들의 참여를 유도하긴 어렵고 매번 투표를 진행하는 것도 비효율적이다. 때문에 본 논문에서는 간단한 예산안 찬반투표를 수행하는 스마트 컨트랙트를 블록체인에 배치하고 예산안 편성 및 세부 예산집행 계획에 대한 찬반투표를 수행하고 결과를 블록체인에 기록하여 누구든지 투표의 투명성과 결과를 쉽게 검증할 수 있도록 한다.

3.3 제안 시스템 모델

제안 시스템은 다음과 같이 구성되어있다.

- **학생회**는 학생회비 1년 예산안과 세부 예산집행 계획을 편성하여 학생들에게 공지하고 이에 대한 찬반 투표를 스마트 컨트랙트로 배포한다.
- **감시자**들은 학과 참여구성원 중 임의로 선정되어 학생회 모임통장 사용내역을 감시하고 예산 집행마다 계획대로 정확하게 집행되었는지 확인하여 다른 감시자들과의 합의를 통해 블록체인에 기록한다.
- **학생**들은 감시자를 제외한 학과 참여구성원으로 블록체인에 기록된 내용을 검증하고 투표에 참여하여 예산안의 적절성을 확인한다.

학생회는 1년 예산안과 학생회비 납부금액을 공지하고 이에 대한 찬반투표를 블록체인 스마트 컨트랙트로 배포한다(블록체인 네트워크는 고정된 참여자들로 구성된 프라이빗 체인을 가정한다). 찬반 투표 컨트랙트는 Pedersen Commitment[4]를 사용한다. 학생들은 공지된 예산안에 대한 찬성(1) 혹은 반대(0)를 선택하고($m_i \in \{0,1\}$) Commitment $c_i \leftarrow Com(r_i, m_i)$를 생성한 뒤 c_i는 네트워크의 다른 참여자들에게 전파한다. 이후 $g^{r_i}(r_i \in \{0, q-1\})$을 계산하여 컨트랙트에 제출한다. 일정 시간 후 컨트랙트는 n명이 제출한 값 $g^r = g^{r_1 + r_2 + \cdots + r_n}$을 계산하여 공개하고 참여자들은 $\pi = g^{-r} \prod_{i=1}^{n} c_i$를 계산하여

$h^s = \pi$를 만족하는 찬성표의 합 s를 계산할 수 있다.

예산안에 계획된 예산 집행 전에 학생회는 예산의 세부 집행계획을 공지하고 투표한다. 이후 투표결과에 따라 학생회가 예산을 집행하면 k명의 감시자들은$(1 < k < n)$ 모임통장을 통해 집행내역을 각자 확인할 수 있다. 예산이 계획에 따라 정상적으로 집행되었다면 감시자들은 확인결과를 서로에게 공유하여 최종적으로 집행결과의 적절성을 블록체인에 기록한다. 하지만 감시자들은 신뢰 그룹이 아니므로 부적절한 예산집행에 대해서도 적절하다는 결과를 제출할 수 있다. 본 논문에서는 학생회 임원들과 감시자들로 구성된 프라이빗 네트워크를 구성하고 실용적 비잔틴 장애 허용(Practical Byzantine Fault Tolerance)[5] 합의를 통해 블록체인에 기록한다. 학생회 임원은 집행결과 메시지를 생성하여 감시자 대표(Primary node)에게 전송한다. 감시자 대표는 다른 모든 감시자들(Replica)에게 전달받은 메시지를 전달하고 각 감시자들은 메시지의 유효성(즉, 실제 통장 기록과 메시지의 일치 여부)를 확인하여 검증결과를 다른 감시자들에게 전파한다. 이후 2/3이상의 감시자가 동의 메시지를 전파한 경우에만 블록체인에 집행결과를 기록한다.

최종적으로 블록체인에는 예산안 편성의 적절성에 대한 투표결과와 유효한 예산 집행결과 메시지가 기록되어 운영관리의 적절성을 학생회를 통하지 않고 언제든 검증할 수 있다.

IV. 결론

학생회는 학생들의 자율적인 관리 조직으로 모든 운영을 학생들의 자치에 맡겨왔지만 특유의 폐쇄적인 운영관리 방식으로 운영관리의 불투명성이 많은 논란이 되어왔다. 본 논문에서는 학생회비를 블록체인 토큰으로 변환하여 사용기록을 기록하는 대신 학생회비 운영관리에 대한 학생 참여구성원들의 합의 내역을 블록체인에 기록하여 학생회 임원들의 중앙집중적인 관리 구조를 개선하고 운영의 투명성을 보장하면서 거래소와 같은 신뢰개체의 참여를 가정하지 않는 학생회비 운영관리 시스템 아키텍처를 제안하였다. 향후 논문에서 제안한 아키텍처를 Hyperledger Fabric이나 Ethereum 개발 플랫폼을 이용하여 구현, 실증 테스트를 수행할 계획이다.

Acknowledgement

'이 논문은 교육부와 한국연구재단의 지원으로 링크플러스사업단 캡스톤디자인 활동의 일환으로 작성되었습니다.'

[참고문헌]

[1] 김한빛찬, "학생 자치의 노력과 한계," 한국대학학회, 대학: 담론과 쟁점, vol.1, pp.138-146, 2018.

[2] "끊이지 않는 대학가의 학생회비 횡령 논란." 숭대시보, 2015년9월14일 수정, 2019년11월9일 접속.

[3] 신승민, 서상민, 송동훈, 최효선, R.Grzegorz, 고석주, 이광열, "블록체인 기반 학교 재정 장부 자동 관리 시스템," 한국정보처리학회 추계학술발표대회, 2018.

[4] T.P.Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing." CRYPTO'91, 129-140, 1992.

[5] M.Castro, and B.Liskov, "Practical Byzantine fault tolerance," OSDI, Vol. 99, No. 1999, pp. 173-186, 1999..

OSINT 개념과 활용방안 연구

유 정훈,† 고 승 철‡

*수원대학교

Study on OSINT Concept and Use Cases

Jung-Hoon Yoo*,† Sung-cheol Goh‡

The University Of Suwon.

요 약

사이버 세상의 공격이 본격화 되고 있다. 과거부터 지금까지도 다양한 공격을 통해 고객의 재산상 피해를 증대 시키고 있다. 빠르게 진화하는 공격유형을 분석 해야만 자산을 최대한 보호 할 수 있다. 이에 따라 본 논문에서는 OSINT의 개념과 국내·외 연구 동향을 살펴본 후 조사방법론을 통해 활용하는지와 향후 연구방안을 제시하고자 한다.

ABSTRACT

Attacks in the cyber world are in full swing. From the past to the day, various attacks have increased the damage to the property of customers. You must analyze the rapidly evolving type of attack to protect your assets as much as possible.

Accordingly, this paper examines the concepts of OSINT and domestic and foreign research trends and proposes ways to use it through research methodologies and future research plans.

Keywords: OSINT, Open Source intellgence, Social Engineering, Humit, Techint, Cybint

I. 서론

1.1 연구의 배경 및 목적

정보통신의 발전으로 인해 사이버 공격이 빈번하게 발생하고 있다. 정보통신 서비스는 과거에는 단편적인 공격에 대한 분석만 하였다면 현재는 실시간성의 공격을 분석해야 하는 상황이다. 지속적인 사이버 공격이 증가됨에 따라 많은 양의 로그와 다양한 로그 형태를 분석하기는 어렵다. 다시 말해, 융합사회의 발전으로 다양한 사이버 공격유형과 공격 기술을 판별하는 능력을 갖추어야 한다. 과거의 사이버 공격들은 어느 정도 분석이 가능하지만, 짧은 시간 동안 수많은 공격 이벤트를 분석하기는 역부족이다. 이로 인해 공개정보정찰 기술의 연구가 지속적으로 필요한 상황이다. 이에 본 연구에서는 OSINT[1] 개념을 정립하고 국내·외 OSINT 기술 발전방향 및 활용사례를 살펴본다. 향후 김병익 등[2]의 연구결과를 통해 사이버 분석방안과 대응대책을 제안 하고자 한다.

1.2 연구의 배경 및 목적

연구의 범위는 공격 예방관점에서 OSINT 기술을 활용하여 어떤 유형들을 분석 할 수 있는지 살펴보고자 한다. 더불어, OSINT 보안에 대한 선행연구는 아직 많지 않은 상태이다. 이슬기 등[3]는 발표한 내용을 살펴보면 OSINT 기반으로 한 사이버 위협정보에 대한 유사도를 빅데이터를 활용하여 다양한 포맷과 유사도를 분석 한 후 활용 방안을 제시하였다. William Sousan 등[4]은 TIDS 플랫폼 구축을 통해 오픈소스 공개 수집(OSINT) 기술을 활용하여 사전 원시 데이터를 분석 할 수 있는 기반을 마련하였다. 다만, 앞서 살펴본 선행 연구사례를

바탕으로 기술 발전 동향과 활용 사례를 살펴본 후 OSINT 기술을 통해 분석을 하는데 있어 도움이 되고자 한다. 추가적으로 향후에 선행에 있어 부족 했던 내용을 토대로 환경을 구축하여 내부데이터와 외부데이터의 가중치를 비교하려고 한다.

II. 서론

2.1 OSINT 개요

OSINT(Open Source Intelligence)는 사이버공간에 있는 공개된 정보를 수집하는 인텔리전트 기법으로 통칭하고 있다. 이에 따라 OSINT의 데이터는 IT시스템과 다른 형태로 진화할 것이다[5].

2.2 OSINT의 종류

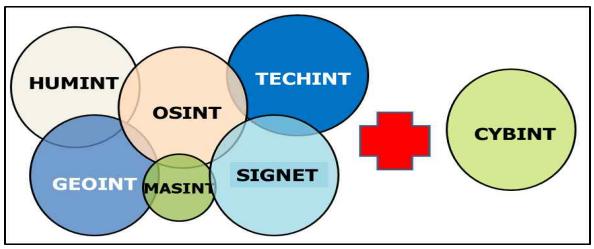


Fig. 1. Different types of open intelligence technologies

Fig. 1.과 같이 정보를 수집하는 방식은 3가지 방식으로 존재한다. 사람이 직접 수집하는 방식은 HUMINT, 기술을 통해 수집하는 방식은 TECHINT (SIGINT: 신호분석, 기후 예측:GEOINT , 주파수 분석:MASINT), 공개된 정보를 통해 수집하는 기술을 OSINT라고 한다. 최근에는 사이버 기술에 대한 수집 기술은 CYBINT라고 한다.

2.3 국내·외 OSINT 연구 동향

2.3.1 국외 OSINT기술 연구현황

미국의 경우 Public Law의 931조항을 보시면 OSINT 라는 내용이 정의가 되어 있다. 그 내용으로는 사이버 운영인력에 대한 전략 방안으로 예산, 기술, 정책, 평가에 대한 다양한 내용이 나와 있다. 이를 통해 2004년 ISVG (Institute for the Study of Violent Groups) Project 라는 미연방정보지원을 통해 테러리즘과 폭력그룹 데이터베이스 구축 프로젝트와 미국토안보부 지원을 통해 2005년에 메릴랜드 대학교에서 시행되었던 테러리즘과 테러리스트 대응 연구를 하는 START 프로젝트가 있다. 특히, ISVG의 프로젝트의 경우 데이터베이스 구축을 통해 서로 다른 언어권의 정보를 활용하여 정보의 활용성을 더욱더 높였다. 2009년에는 START로 통합되어 서로의 장점을 서로 보완하였다. 유럽의 경우에는 온라인 환경을 대상으로 조직적 범죄연구를 하였다. 이처럼 여러 국가에서 OSINT기술을 활용하여 테러리즘과 범죄분야에서 많이 활용 하고 있다[6].

2.3.2 국내 OSINT기술 연구현황

국내의 경우에는 아직 관련 법령은 없다. 다만, 국내의 경우에는 2012년 북한의 활동을 모니터링 하고자 북한 공개 정보 센터 구축을 하자는 논의를 한 바 있다. 또한, 한병우[6]의 단어조합 검색을 이용한 불법 유해 정보 탐지기법을 OSINT 기술을 통해 탐지 자동화를 통한 연구 사례도 있다. 현 시점으로 2018년까지 한국인터넷진흥원의 상황전파문, MISP 등 11종의 OSINT 채널로 수집하고 있고 2차정보는 연관성 및 연결성을 위해 IP2location 채널을 이용하여 정보를 수집하고 있다[7].

2.3.3 국내·외 OSINT 기술을 통한 시사점

서구의 국가의 경우에는 오래전부터 OSINT 기술을 활용한 전략, 정책, 기술, 매뉴얼, 보고서 등의 다양한 연구가 진행 중에 있다. 다만, 국내의 경우에는 이러한 분야에 대한 연구가 미흡한 실정이다.

III. OSINT 활용방안 제시

선행 논문을 참고하여 OSINT기반의 조사 분석 절차를 통해 활용방안을 제시 한다.

3.1 OSINT 기반의 조사 분석 절차



Fig. 2. osint-based investigation procedure

Fig. 2.과 같이 OSINT 기반의 조사절차 방법론을 만들어 업무 현상에서 활용 할 수 있다.

3.1.1 수집단계

위협정보 관점에서 이용자들이 주로 쓰는 검색엔진이나 이용자 Feed 및 크롤러 엔진을 활용하여 수집대상의 공개정보 데이터를 분석하여 질적인 정보를 추출하는 단계이다.

Table 1. Collection Types of Cyber attacks

Collectio Type	Detailed description (ex: example)
Search Engine	Search engine that can search various information ex: google, duck duck go, pastbin, the internet archive, etc
FEED	Public threat intelligence data source repositories provided by organizations and individuals ex:Abuse.ch(Ransomware, Trojan) Proofpoint ET intelligence (information about emerging threats), MISP, etc
Crawer	Ability to collect various information such as web document location (URL), link information, document, etc. ex: spiderfoot crawler

Table 1과 같이 외부에서 일어나고 있는 사이버공격에 대한 다양한 공격 형태를 수집 분석 할 수도 있고 비교할 수 있다.

3.1.2 처리단계

앞서 설명한 수집단계상의 위협정보를 전처리 진행후 다양한 툴을 이용하여 이메일이나 IP정보, 각종문서에 대한 다양한 정보를 추출한다.



Fig. 3. Information extraction using maltego

Fig. 3.과 같이 Maltego Tools을 활용하여 이메일 정보만으로도 다양한 정보 수집처리가 됨을 하나의 예시를 통해 확인 하였다.

3.1.3 분석단계

파일이나 URL의 다양한 위협정보를 분석하는 단계이다.

Table 2. Free Automated Malware Analysis Service

ITEM	Detailed content
AMAaaS	Android apk Hunting
any.run	Malware Hunting
URLhaus	malicious URLs tracked
Hybrid Analysis	unknown threats analyze and Detect

Table 2와 같이 자동화된 악성분석 툴을 이용하여 다양한 정보에 대해 악성파일내용을 분석한다. 이에 담당자가 사고를 대응하거나 예방할 수 있도록 분석 시간을 단축 시켜 준다.

3.1.4 시각화 단계

관리자나 담당자가 에게 보기 쉽게 해당 공개정보를 보여주는 단계이다.

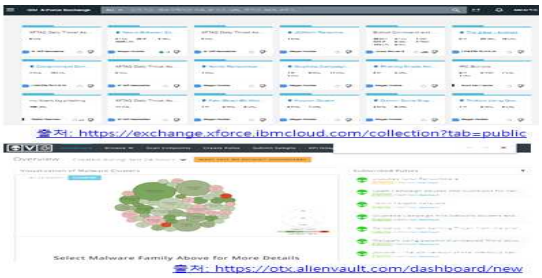


Fig. 4. Visualization through classification of corporate malicious file types

Fig. 4.와 같이 여러 가지 유형에 따른 수집유형에 따른 상세분석도 할 수 있고 연계되어 있는 위협 정보 및 분류별로 공격 유형을 시각화된 형태로 한눈에 볼 수 있다.

3.1.4 협력 단계

이러한 위협정보들을 관련 기관과의 공유를 통해 위험 리스크를 사전에 예방하고 분석할 수 있다.

IV. 결론

본 논문에서는 공개정보 인텔리전스 기술의 개념과 기술 동향을 살펴보았고 다양한 공격에 대한 이슈들을 OSINT 조사 절차를 통해 어떻게 활용 접근 방식을 제시하였다. 이러한 연구를 통하여 다음과 같은 효과가 기대된다.

- OSINT 관점의 수집 정보 제시
- OSINT 관점의 활용 방안 제시
- 사이버 공격에 대한 가중치 분석방안 제시
- 위협 공개 정보에 대한 위험요소 제공 가능

[참고문헌]

[1] Sparks, Timothy, "Why haven't technologies fixed open source intelligence?" (2014). Masters Theses. 13(7~8 pages)

[2] Byungik Kim, Nakhyun Kim, Byung-ik Kim, Seulgi Lee, Hyeisun Cho, Jun-hyung Park., "A Study on Cyber Threat Intelligence Analysis (CTI) Platform for Proactive Detection of Cyber Attacks Based on Automated Analysis," Journal Proceedings of Symposium of the Korean Institute of communications and Information Sciences, 11, 2017, pp. 578~579.

[3] Seulgi Lee, Nakhyun Kim, Hyeisun Cho, Byungik Kim, Junhyung Park., "A study on the similarity analysis of cyber threat indicators based on OSINT," Korean Information Science Society, 12, 2017, pp. 1114~1115.

[4] Sousan, William; Zhu, Qiuming; and Hospodka, Peter,D., Kim, S., Lee, Y., and Kim, C., "Collecting Open Source Intelligence via Tailored Information Delivery Services," Computer Science Faculty Publications. Paper 20.

[5] Fahimeh Tabatabaei, Douglas Wells., "OSINT in the Context of Cyber-Security.", ResearchGate, 02, 2018, Pages 213-231.

[6] Tak, Hee-sung, Park, Jun-hee, Cheong, Jin-seong, Yoon, Ji-won, "Building Crime Prevention System Utilizing Big Data (II)", Korean Institute of Criminology, 12, 2015, 15-B-18.

[7] Byeong Woo Han, Ji Won Yoon., "Illegal and Harmful Information Detection Technique Using Combination of Search Words," Journal of the Korea Institute of Information Security & Cryptology 26(2), 2016.4, 397-404(8 pages).

[8] Seong-Joo, Kim., "Development Direction of Military Counterintelligence in Accordance with Information Environment Changes," 6, 2019 Defense & Security Vol. 1 No. 1(106 pages).

커넥티드 의료기기 트래픽 분석 기반 사이버보안 자산 관리 연구

이준석*, 정일안**, 정동섭**

*전남대학교 정보보안협동과정, **(주)휴네시온

A Study on Cybersecurity Asset Management based on Traffic Analysis of Connected Medical Devices

Jun-Suk Lee*, Il-Ahn Cheong**, Dong-Seob Jung**

*Interdisciplinary Program of Information Security, Chonnam National University

**HUNESION Co., Ltd.

요 약

병원 현장에서 다수의 복합적인 의료기기를 대상으로 악성코드 감염으로 인한 피해가 점차 늘어가고 있지만, 의료기기의 특성상 기존 ICT와 유사한 대응 방식으로 조치를 취하기 어려운 상황이다. 본 논문에서는 사이버보안 관점에서 커넥티드 의료기기를 보호하기 위해서 먼저 의료기기 자산을 식별하고 관리하는 방안을 제안한다. 실제 DICOM 통신 트래픽에 의료기기를 식별하는 고유한 디바이스 식별자 정보가 없더라도, SOP 관련 특정 태그 정보를 기준으로 의료기기를 식별함으로써 의료기기 특성에 적합한 사이버보안 자산 관리 체계를 마련하는데 기여한다.

I. 서론

병원 네트워크 환경에서 다수의 의료기기를 대상으로 악성코드에 감염되는 피해가 지속적으로 발생하고 있다. 2017년 WannaCry 랜섬웨어의 경우에 영국에서 81개 의료기관 피해로 20,000건의 수술 또는 진단이 취소되었고, MEDJACK의 경우에 X-Ray, MRI, CT Scan, PACS 등 의료기기를 통해 병원 네트워크에 침투하는 공격이 수차례 발생했다고 TrapX 연구에 의해 보고되었다[1]. 이러한 사이버 공격은 점차 지능화되고 있는 추세로 구형 OS가 탑재된 의료기기, 약물주입기, 원격수술로봇 등 대상도 확대되고 있다. 이와 같은 커넥티드 의료기기에 대한 사이버보안 위협 대응이 미비한 것은 의료기기 고유의 특성 때문이다. 다수의 의료기기들은 병원에서 오래 사용하는 장비로 구형 OS나 SW가 탑재되어 있어 보안 패치나 업데이트가 제공되지 않는 경우가 많고, 의료기기의 가용성이 중요하기 때문에 의료기기용 백신 등 추가적인 소프트웨어 설치를 허용하지 않는다. 그리고 국내 병원은 외산 의료기기가 다수를 차지하기 때문에 보안대책 마련이 쉽지 않고, 점차 무선 네트워크나 접속 인터페이스가 다양화되고 이동성과 연결성이 증가되고 있어 이에 대한 보안 대책도 마련해야 할 시점이다. 먼저 보호가 필요한 커넥티드 의료기기 자산을 식별하기 위해서 의료기기의 특성을 고려하여 일반 기기(PC, 서버, 단말 등)와 구분되는 정보를 선별하고, 의료기기의 통신 트래픽 분석을 통해 그 정보를 추출하는 방법이 필요하다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 커넥티드 의료기기의 보안에 대해 소개한다. 3장에서는 커넥티드 의료기기의 통신 트래픽 분석을 통해 의료기기 자산을 식별하고 관리하는 방안에 대해서 설명한다. 마지막으로 4장에서는 결론을 맺는다.

II. 커넥티드 의료기기 보안

국내 상급종합병원의 경우 1만여 개의 복합적인 의료기기가 설치되어 있어, 관리가 용이하고 의료현장에 적합한 분류체계가 필요하며 안전 및 보안 인식이 고취되어야 한다. 그리고 보통 10% 정도가 인터넷과 연결되어 있는데 최근 출시되는 의료기기는 인터넷과 연결되며 향후 60% 이상 증가할 것으로 보고 있다[2]. 본 장에서는 의료기기 보안 기술 동향과 국내외 의료기기 관련 사이버보안 가이드 현황을 설명한다.

2.1 의료기기 보안 기술

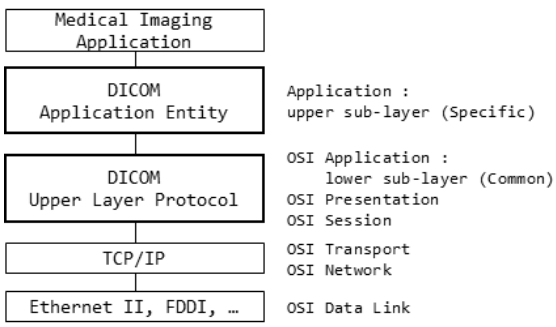
의료 현장에서는 의료기기들이 항상 사용 중에 있으므로 기기들을 파악하고 가시성을 확보해 놓은 다음 그 기기에 대한 사용 패턴을 모니터링하면서 관리하는 방법이 필요하다. 국외 Medigate 사는 병원 네트워크의 의료기기 보안과 자산(asset) 관리 플랫폼으로 디바이스 유형, 네트워크 통신, 기기 흐름 패턴 등 이상행위를 분석하고 대응하는 기술을 개발하였다. Zingbox 사는 의료기기 네트워크 행위에 대한 딥인사이트 분석을 통해 벤더, 모델, 시리얼 번호, OS 등 기기 정보를 탐색한다. 또한, 기존 보안 솔루션 업체들도 병원 의료기기의 특성에 맞춘 비침습적 방법으로 네트워크 행위를 분석하고기 구축된 네트워크 장비(스위치, 라우터 등)나 NAC과 연동하여 차단 정책을 적용함으로써 대응한다.

2.2 의료기기 사이버보안 가이드

미국 FDA는 시판 전/후 의료기기 사이버보안 가이드라인 문서 발간을 통해, 의료기기 사이버보안 위협으로 인한 환자의 위험을 완화하고 감소시키기 위해 설계 및 개발 단계부터 사이버보안 이슈들을 고려해야 한다고 권고하고 있다[3, 4]. 또한, 사이버보안 위협과 취약점들을 관리하고, 위험관리 프로그램을 실행할 것을 권고하고 있다. 국내 식품의약품안전처는 의료기기 제조사에 “사이버보안 허가 및 심사 가이드라인(민원인 안내서)”을 배포하고, 의료기기 사이버보안 요구사항을 식별, 보호, 탐지, 대응, 복구 단계로 구분하여 총 24개의 요구사항을 정의하였다[5]. 특히, 식별 및 보호 항목이 21개로 다수이고, 의료기기를 식별하기 위해 ‘사용자(의료기기) 접속 인식’ 항목이 포함되어 있다. 또한, 과학기술정통부, KISA, IoT 보안 얼라이언스 공동으로 “스마트의료 사이버보안 가이드” 문서를 배포하고, 병원에서 운영되고 있는 의료기기, 시스템, 네트워크에 대한 보안 위협(의료기기 분실·도난, USB 악성코드 감염 등)과 대응 방안을 제시되어 있다[6].

2.3 의료기기 정보교환 표준규격

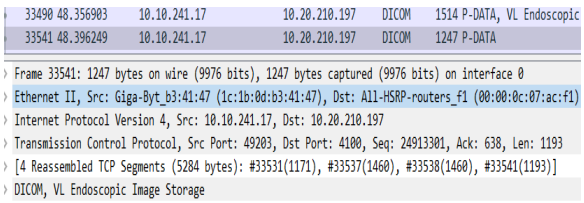
의료분야에서 의료영상 저장 및 전송 장치로 PACS (Picture Archiving and Communications System)가 사용되고 있는데, 각 의료기기 제조사 또는 영상 장비마다 다른 디지털 영상 정보는 표준화 영상 포맷인 DICOM (Digital Imaging and Communications in Medicine)으로 영상을 획득한다. DICOM은 의료 영상을 구성하고 교환하는 방법과 관련 정보들을 기술한 규격이다[7]. 모든 의료기기는 DICOM 서비스 및 관련된 명령, 문법, 역할 등을 적합성 명세(Conformance Statement) 문서에 기술해야 한다. 현재 사용되고 있는 DICOM 표준은 객체지향 설계 방식을 도입하고, 영상 장비를 네트워크에 직접 연결이 가능하게 되어 필요한 Header Information 정보와 Image 정보를 결합하여 영상을 전송하는 기능을 지원한다. [그림 1]은 DICOM 프로토콜 계층을 나타낸 것으로, TCP/IP 계층 상위의 DICOM Upper Layer Protocol(OSI 응용 계층의 공통 하위 서브-계층)과 DICOM Application Entity(OSI 응용 계층의 특정 상위 서브-계층)로 구성되어 있다.



[그림 1] DICOM 프로토콜 계층

III. 의료기기 자산 식별 및 관리

본 논문에서는 커넥티드 의료기기를 보호해야 하는 자산(asset)으로 간주하고 식별 대상으로 한다. 일반적으로 의료기기에서 영상 정보를 전송하는 DICOM 통신 트래픽을 선별하고 분석하여 의료기기의 유형, 제조사, 모델명 등 정보를 획득한다. DICOM 태그 정보에는 기기를 식별하기 위한 고유한 디바이스 식별자(Unique Device Identifier) 정보가 있지만, 실제 DICOM 통신 트래픽에서는 그 정보를 포함하지 않는 경우가 많다. 따라서 다른 태그 정보들을 통해서 의료기기를 식별하는 방안도 고려해야 한다. [그림 2]와 같이, 네트워크 인터페이스 카드(NIC)의 MAC 정보로 의료기기를 식별하는 방식은 의료기기 제조사가 아닌 NIC 칩 제조사(1C:1B:0D := GIGA-BYTE)를 의미하므로 정확성이 낮아 적절하지 않다. 또한, IP 주소 정보로 의료기기를 식별하는 방식도 유동 IP 주소나 운영 정책에 따라 변경되는 경우에 정확성이 낮게 되어 적절하지 않다.

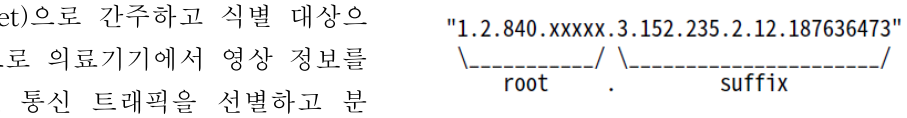


[그림 2] DICOM P-DATA 패킷 정보

본 절에서는 의료기기를 식별하기 위한 정보로 SOP Class UID와 SOP Instance UID 정보를 기준으로 우선 OID 데이터베이스 또는 GUDID (General Unique Device Identification Database)를 검색하고, 추가적으로 Modality, Device Serial Number 등 태그 정보로 식별하는 방법을 제시한다.

3.1 SOP Instance UID

객체 식별자 OID(Object IDentifier)는 인터넷에서 사용하는 인터넷주소체계 등과 유사하게 객체 타입의 유일성을 보장하기 위한 식별체계를 의미한다. 표준문서, 국가, 암호 알고리즘, 네트워크, 코드식별 등 다양한 객체를 구분하기 위해서 OID를 사용한다. 등록된 OID에 대한 정보저장 및 조회를 위해 OID 트리 등록시스템(http://www.oid-info.com)을 이용하여 조회한다[8].

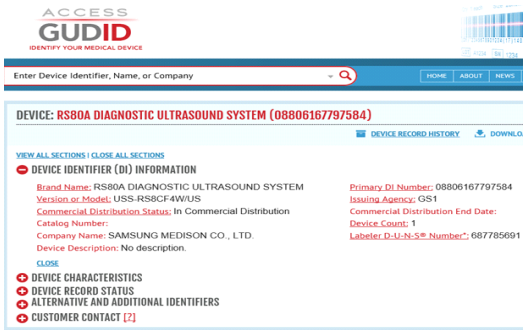


[그림 3] 의료기기 OID 식별자 형식

[그림 3]에서 'root' 파트의 1.2는 ISO와 ANSI 회원을 의미하고, 840은 특정 회원의 국가코드(U.S.)이며 xxxx는 ANSI에 할당된 특정 조직을 의미한다. 디바이스 식별과 관련한 'suffix' 파트의 3.152는 제조사가 정의한 디바이스 유형과 시리얼 번호를 의미한다. 나머지 suffix 파트의 4가지 요소는 이미지와 관련되어 있고, Study.Series.Image 번호(235.2.12)와 인코딩된 이미지 취득 날짜/시간(187636473)을 의미한다[9].

3.2 FDA GUDID

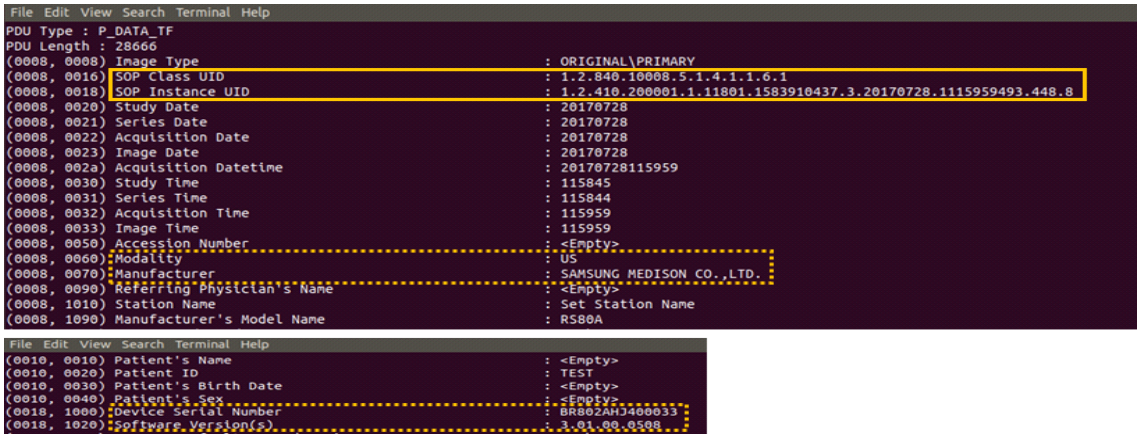
[그림 4]의 Access GUDID는 FDA에서 제공하는 의료기기 식별 정보 검색 서비스이다[10]. 기기 식별자나 모델 또는 제조사로 검색하면 디바이스 식별 정보, 특징, 기록 상태, 추가 정보 등을 제공하고 있다. 이와 유사하게 국내는 식약처의 의료기기통합정보시스템이 있다.



[그림 4] Access GUDID 검색 서비스

3.3 의료기기 통신 트래픽 분석

DICOM 통신으로 전송하는 커넥티드 의료기기들의(MRI, CT Scan 등) 상세한 의료기기 자체 정보를 분석하기 위해, 먼저 수집한 트래픽에서 <표 1>에 나열된 A-ASSOCIATE-RQ 패킷의 각 필드 값들이 모두 일치하면 DICOM 통신 패킷으로 판단한다.



[그림 5] DICOM 메시지 분석 정보

<표 1> A-ASSOCIATE-RQ PDU

PDU bytes	Field name	Values
1	PDU-type	01H
2	Reserved	00H
7-8	Protocol version	1
9-10	Reserved	0000H
43-74	Reserved	00H (all)

그 이후에 P-DATA-TF 패킷들을 모아서 조립하고 Presentation-data-value 필드를 분석하여 DICOM 메시지 정보를 추출한다. [그림 5]의 DICOM 메시지 분석 정보에서 SOP Class UID의 속성 값을 통해 의료기기의 유형을 Ultrasound Image Storage로 식별하고, SOP Instance UID의 속성 값을 통해 의료기기 제조사(Samsung Medison)와 종류(초음파 진단기)를 식별한다. 추가적으로, Modality 속성 값으로 의료기기의 유형이 US(Ultrasound)라고 식별하고, 디바이스 시리얼 번호(BR802AHJ400033)와 소프트웨어 버전(3.01.00.0508)을 식별한다.

IV. 결론

본 논문에서는 의료 현장의 다수의 복합적인 커넥티드 의료기기들에 대한 사이버보안 관점에서 자산 식별과 체계적 관리의 필요성에 대해 살펴보았다. DICOM 태그 정보에는 기기를 식별하기 위한 고유한 디바이스 식별자 정보가 있지만, 실제 DICOM 통신 트래픽에서는 그 정보를 포함하지 않는 경우가 많았다. DICOM의 다른 태그 정보들을 통해 의료기기를 식별하는 방안에 대해 제안하였고, SOP Class UID와 SOP Instance UID 정보를 기준으로 우선 OID 데이터베이스를 검색하고, 추가적으로 Modality, Device Serial Number 등 태그 정보로 의료기기를 식별하였다. 다만, OID 또는 FDA의 GUDID 정보가 불충분하거나 DICOM 통신 트래픽에서 추출한 정보와 일치되도록 통합적인 의료기기 식별 정보 관리 체계가 마련되어야 한다.

[참고문헌]

[1] TrapX Investigative Report, MEDJACK.4 Medical Device Hijacking, TrapX research labs, 2018.

[2] ZDNet Korea, 백신도 못 까는 의료기기 - 보안성 강화 시급, 2018.

[3] FDA, Premarket Submissions for Management of Cybersecurity in Medical Devices, 2014.

[4] FDA, Postmarket Management of Cyber-security in Medical Devices, 2016.

[5] 식품의약품안전처, 의료기기의 사이버 보안 허가·심사 가이드라인(민원인 안내서), 2017.

[6] IoT보안얼라이언스, 스마트의료 사이버보안 가이드, 2018.

[7] DICOM, https://www.dicomstandard.org

[8] OID Repository, https://www.oid-info.com

[9] DICOM PS3.5 2019d, Data Structures and Encoding, dicomstandard.org, 2019.

[10] GUDID, https://www.accessgudid.nlm.nih.gov

영국 사례를 통해 본 한국 규제 샌드박스 시행의 현황 및 시사점

김지혜*, 염흥열**

*한국정보기술단,**순천향대학교 정보보호학과

Current Status and Implications of the Korean Regulatory Sandbox according to UK's case

Jihye Kim*, Heung-Youl Youm**

*KISCA. **Department of Information Security Engineering, SoonChunHyang University.

요 약

한국 규제 샌드박스가 2019년에 시행되었다. 규제 샌드박스를 통해 핀테크 분야의 규제 불확실성이 해소되고 혁신 금융서비스 개발이 촉진될 것으로 기대한다. 한국의 규제 샌드박스는 금융을 포함하여 전 산업분야에 적용하는 특징이 있다. 그러나 시행에 있어 실효성에 대한 문제가 제기되고 있고 이를 선결하기 위한 규제 샌드박스 선진사례를 살펴볼 필요가 있다. 제도 시행의 구체적인 활성화를 위해 규제 샌드박스의 도입국인 영국의 규제 샌드박스를 연구·분석하여 한국의 규제 샌드박스와 비교하였고 두 나라간의 특징을 도출하여 긍정적 기대효과를 내기 위한 시사점을 제시하였다. 또한 우리나라 규제 샌드박스 제도의 발전 방향을 제시한다.

I. 서론

규제란 일반적으로 개인 및 법인 등의 활동에 대해 국가가 개입하여 효율, 형평성, 안전 등의 공익을 보장하기 위한 목적으로 제한하는 것을 말한다[참고문헌]. 경제사회질서의 구현을 위해서 규제는 필수불가결하다. 하지만 4차 산업혁명 시대가 도래함에 따라 빅데이터, 인공지능, IoT, 등 새로운 혁신 기술이 등장하여 기존 규제체계 내에서는 적용이 불가한 경우가 다수 발생하고 있다. 따라서 복합적인 측면에서 규제를 검토하여야 한다. 이를 위해 시행된 제도가 규제 샌드박스이다.

규제 샌드박스란 새로운 제품이나 서비스를 출시할 때 기존 규제에 막혀 지체되거나 무산되는 일이 없도록, 일정한 조건 아래 기존 규제를 면제하거나 유예하여 시장에서 테스트·출시할 수 있도록 허용해주는 제도이다.[1] 규제 샌드박스 제도는 2016년 영국에서 시작하여 전 세계 주요국으로 확산되었다. 한국은 당해년도인 2019년에 규제 샌드박스를 시행하였다. 선진사례인 영국의 사례와 한국의 사례를 비교·분석하여 향후 국내 규제 샌드박스의 안정적 시행을 위한 개선점을 제시한다.

본 논문의 II에서는 영국의 핀테크 규제 샌드박스를 살펴보고, III장에서는 국내 샌드박스 현황을 제시하며, 이를 근거로 III장에서 영국과 우리나라의 규제 샌드박스를 주요 특성 별로 비교 분석한 결과를 제시했다. 마지막으로 IV장에서는 우리나라 제도의 발전 방향을 제안했다.

II. 영국의 핀테크 규제 샌드박스

영국의 규제 샌드박스는 2015년에 영국 재무산하 독립청인 금융청(FCA, Financial Conduct Authority)이 금융서비스 혁신 기업들에게 안전

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2019-0-00660, 차세대 ICT 환경에서의 보안 및 개인정보보호 기술 국제 표준화 추진)

한 '실험의 장'을 제공하기 위해 규제 샌드박스를 도입하였고 2016년에 본격적으로 첫 시행을 하였다. 규제 샌드박스 신청기업은 FCA에 의해 평가를 거쳐 선정되면 정부와 함께 최적의 샌드박스를 설계하고 사업 테스트와 모니터링을 통해 사업개시 여부를 결정한다.

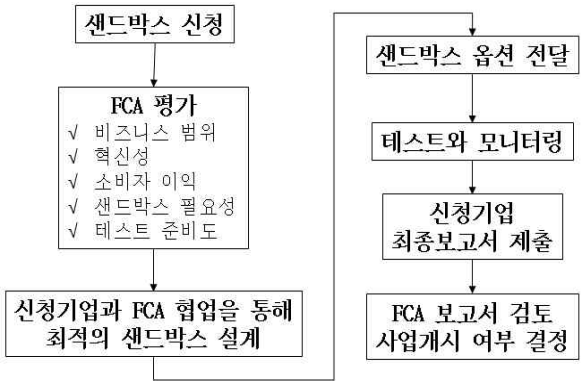


그림 1 영국 규제 샌드박스 추진 절차[2]

영국 규제 샌드박스는 코호트 단위로 모집하며 모집 신청 기간은 약 2개월 정도로 소요된다. 코호트는 영국 금융감독청이 샌드박스 프로그램을 그룹 단위로 모집하며 '특정한 행동양식 등을 공유하는 집단'이라는 뜻의 사회학적 단어인 코호트를 활용하여 표현한다.[3]

선정평가 기준은 비즈니스 범위, 혁신성, 소비자의 이익, 규제 샌드박스의 필요성, 테스트 준비도로 구성되어 있다.

[표 1] 영국의 규제 샌드박스 심사 항목[2]

심사 항목	심사 내용
비즈니스 범위	신청된 서비스는 영국 금융 서비스 시장에서 현행 규제나 사업에 관하여 개혁이나 혁신을 바라고 있는가
혁신성	신청된 서비스는 기존 서비스와 유사하거나 또는 크게 다른 성질을 갖고 있는가
소비자의 이익	테스트 기간을 통해 신청된 서비스는 소비자에게 특정 가능한 편익을 제공할 수 있을 것인가
규제 샌드박스의 필요성	샌드박스를 적용하는 목적은 합리적인가
테스트 준비도	신청자는 투자, 법제의 이해, 리스크 감소 대책 마련을 하고 있는가

2019년도에 제5차 코호트의 업체 선정이 마무리되어 지난 3년간 131개의 업체가 선정되었고 73개의 업체가 테스트를 완료하였으며 29개의 업체는 현재 테스트를 진행 중이다.[2] 2018년에 선정된 제4기 코호트에서는 처음으로 암호 자산에 대한 실험도 규제 샌드박스에 신청할 수 있도록 허용하였다.

1년간의 경과를 평가한 결과, 테스트 참여기업은 80% 이상이 스타트 기업이며, 절반 이상은 금융권 업무에 해당했다. 분산원장기술, 생체인식, 온라인IPO플랫폼, 금융정보 통합, 로보어드바이저 등의 신기술 금융서비스를 통해 소비자 효용 증대 뿐 아니라, 시장진입기간 단축, 비용절감, 금융접근성 향상 등 긍정적인 평가를 받았다. 그러나 은행 고객정보 부족으로 서비스의 테스트 결과에 대한 근거 미흡하고, 금융기관간 시스템이 상이하여 정보의 통합이 어려우며, 테스트 과정에서 추가적인 규제를 확인한 경우 스타트업이 즉각적으로 새로운 규제요건을 충족시키기 어려움을 발견했다.[4]

현재 영국은 참가 회원국을 모집해 글로벌 샌드박스 구축을 위한 국제금융혁신네트워크(GFIN, Global Financial Innovaion Network)를 결성했다[3]. GFIN은 새로운 기술 혹은 서비스 모델에 대한 혁신과 경험을 각국시장에 공유하거나 적용하는데 도움을 주고 다국적인 사업 아이템을 갖고 있는 기업의 환경 조성을 위해 설립하였다. GFIN이 운영하는 글로벌 샌드박스는 2018년 5월 기준 17개 기관이 참여하였고 44개 업체가 신청하여 그중 8개의 업체가 선정되었다. 선정된 업체는 테스트 후 가동 예정단계에 있다.[3]

III. 한국의 핀테크 규제 샌드박스

한국의 법령체계는 원칙적 금지·예외적 허용 방식을 기반으로 하기 때문에 법령에서 규제 적용에 대한 여부를 판단하고 있지 않은 사안은 규제하고 있다. 이 체계는 과학기술 분야에도 적용되어 명백한 허용 규정이 없거나 실무상 허용한 사실이 없는 새로운 시도는 일단 금지한다. 신기술 사업은 여러 부처의 규제가 복합적으로 작용하여 쉽게 해결하기가 어렵다. 신기술 사업의 성장을 위하여 규제 완화라는 제도 장치의 필요성이 대두되었고 한국은 금융위원회가 2016년부터 금융, IT 융합 등에 따른 새

[표 2] 한국 규제 샌드박스 관련 법안[5]

구분	ICT규제샌드박스	산업융합규제샌드박스	금융규제샌드박스	규제자유특구
법안	정보통신융합법	산업융합촉진법	금융혁신지원특별법	지역특구법
국회통과	2018.09.20	2018.09.20	2018.12.07	2018.09.20
시행	2019.01.17	2019.01.17	2019.04.01	2019.04.17
소관부처	과학기술정보통신부	산업통상자원부	금융위원회	중소벤처기업부
주요내용	-제3조의2: 우선허용·사후규제원칙 -제36조의2: 일괄처리 -제36조 및 제37조: 현행 임시허가제도 보완(기간연장) -제38조의2: 실증을 위한 규제특례	-제3조의2: 우선허용·사후규제원칙 -제8조의2: 갈등조정위원회 구성 및 운영 -제10조의2~6:신속확인, 실증을 위한 규제 특례, 임시허가 등	-제2조와 제4조: 혁신금융서비스의 정의, 지정 -제13조: 혁신금융심사위원회의 설치 및 구성 -제17조: 규제적용특례 -제24조: 규제신속확인	- 법률명칭 변경 : 규제자유특구 및 지역 특화발전특구에 관한 규제특례법 - 제2조의 13: 규제자유특구의 정의

로운 상품·서비스의 원활한 성장을 지원하기 위해 영국의 샌드박스를 벤치마킹한 금융규제 테스트베드 도입을 추진하여 투자자에게 보다 효율적이고 넓은 선택의 기회를 제공할 수 있는 규제 샌드박스를 처음으로 실시하였다.

한국의 규제 법령의 근간인 '원칙적 금지·예외적 허용'의 포지티브 규제의 한계를 극복하기 위해 신기술·신사업 분야에 대해 '원칙적 허용·예외적 금지'의 네거티브 규제 방식인 '선허용-후규제' 대원칙을 규제 샌드박스에 도입하였다.[6]

한국형 샌드박스는 네거티브 규제 체계를 기초로 하는 대신에 관련분야 이해당사자 및 전문가들의 신충검토를 거쳐서 일정한 제한 조건을 설정하고 해당 조건 하에서 현행 규제 집행을 유예하거나 면제 할 수 있도록 하고 있다.[7]

한국의 규제 샌드박스는 핀테크 산업 분야뿐만아니라 해양산업, 스마트관광, 탄소섬유 등 더 광범위한 규제 샌드박스가 도입되고 있다.

이에 따라 표 2를 보면 규제 샌드박스를 주관하는 기관도 핀테크 분야의 금융위원회, ICT융합 분야의 과학기술정보통신부, 산업 융합 분야의 산업통상지원부와 지역혁신 분야의 중소벤처기업부에 이르기까지 국가차원의 광범위한 구조를 이루고 있는 것 또한 특징이다.

기존 논의된 다양한 입법안을 기초로 '신속확인·실증특례·임시허가' 3가지 유형으로 구성된다. 각 부처는 규제 샌드박스 지정을 위해 관계부처 협의·사전검토 등을 거쳐 심의위원회를 개최하고 심의결과를 바탕으로 신청과제들에 대해 규제특례 적용 여부를 결정하고 있다.

[표 3] 규제 샌드박스 유형별 내용과 절차[8]

	신속확인	실증규제특례	임시허가
목적	실증과 테스트 목적 (구역·기간·규모 제한)		시장출시 목적 (구역·규모 제한 정도가 낮거나 없음)
주요내용	기업이 개발한 신기술·서비스· 제품의 규제가 모호한 경우 법령 적용 여부와 규제존재 여부 등을 신속히 확인 (30일내 회신)	신기술·서비스· 제품이 법령공백, 규제 적용 부적합, 금지 및 불허인 경우 제한된 구역·기간(2년이내 1회연장) · 규모 내에서 규제 적용 배제	신기술·서비스· 제품의 법령공백, 규제 적용 부적합인경우 시장출시가 가능하도록 임시허가 (2년이내 1회연장)
절차	① 과제 신청 과제 접수	② 소관부처 회신 (30일 이내)	③ 사전검토 위원회 심의위원회	④ 과제 시행·운영 (2+2년)	⑤ 결과 보고서 법제도 정비

IV. 영국과 한국의 규제 샌드박스 비교

영국과 한국의 규제 샌드박스를 비교분석한 결과, 기본 운영방식은 대체로 유사하였으나 서비스의 지정기간, 규제 개입 범위 등에서 차이가 있었다. 한국 규제 샌드박스는 금융 분야 외에도 정보통신·산업융합 분야, 지역특구에 적용된다는 차이점을 볼 수 있다. 다양한 분야를 포괄하는 입법안은 최초의 시도이기 때문에 주무부처가 하나인 영국과 달리 4개의 부처가 사업에 해당한다. 또한 영국은 실증 특례 중심으로 3년간의 실증 테스트를 통해 성과를 분석하여 약 80퍼센트의 신청기업이 테스트 완료 후 정식으로 서비스를 출시하였다. 한국은 현재 초기 도입 단계로 신청된 서비스에 대한 실증 테스

[표 4] 영국과 한국의 규제 샌드박스 비교

항목	영국	한국
주무 부처 (사업 담당)	재무부	과기정통부, 산업부, 중기벤처부, 금융위
시행	16년 5월	19년 1월
대상 산업	금융 핀테크	금융을 포함한 전 산업
모집 기간	평균 7개월 간격 모집(코호트)	수시
주요 특징	- 세계 최초 시행 - 코호트 운영 - 선정 5~8개월 소요 - 실증특례 중심 - GFIN 결성	- 포괄적 네거티브 규제 - 한시적 규제 유예 - 부처 별 운영 - 실증특례, 임시허가, 신속확인 3종 제도 신설

트를 하여 실효성을 확보하는 단계에 있다.

V. 결론

세계적인 핀테크 허브인 영국의 산업 성숙도나 금융 규제수준을 고려하면 영국은 핀테크 선진국에 해당한다. 약 3년 정도 선행된 영국의 규제 샌드박스를 통해 국내 규제 샌드박스에 대한 시사점을 유추해 볼 수 있었다. 비교분석을 통해 한국 규제 샌드박스의 특징을 명확히 이해하고 이를 기반으로 제도의 지속적인 활성화를 위한 시사점을 도출하였다.

첫 번째로 한국 규제 샌드박스 활성화를 위한 정책적 방안으로서 절차 상 생성된 정보를 공개해야 한다. 현재 각 부처의 사이트에서 신청 방식을 안내하고 있으나 실제 관련 법령 조문을 명확히 적시하지 않고 있다. 어떤 사항이 신청기관에 규제 유예를 해주는지 정확한 공개가 필요하다. 두 번째로 한국의 규제 샌드박스의 성공적인 운영과 실현은 제도 도입의 취지에 부합하기 이해서는 신속한 규제 불확실성 해소와 혁신금융서비스 지정 및 인허가의 실효성 제고 여부에 달려있다. 금융사고의 사회경제적 파급효과를 고려하여 소비자 보호방안에 대한 심사는 엄격히 하되, 기타 심사기준이나 인허가 등에서는 적극 행정을 통해 제도의 실효성을 제고해야한다. 입법안을 기초로 '신속확인·실증특례·임시허가' 유형이 또 하나의 진입장벽이 되어서는 안된다. 세 번째로는, 한국의 규제 샌드박스의 시행이 안정화되면 국내의 해외진출 희망기업이 해외 금융규제에 선제적으로 대응하고 서비스를 테스트할 수 있도록 국제금융혁신네트워크(GFIN)의 가입 타당성을 검토해야 한다. 가입시, 국내 규제 수준의 적정성을 점검할 수 있다. 한국이 규제 샌드박스는 올해 시행되어 박차를 가하고 있다. 지속적인 규제와 운영 정비가 전제될 때 실효성 있는 규제 샌드박스 제도운영이 가능할 것으로 보인다.

[참고문헌]

[1] 정책위키, 규제 샌드박스, http://www.korea.kr/special/policyCurationView.do?newsId=148857563#L1

[2] FCA, https://www.fca.org.uk/firms/global-financial-innovation-network

[3] 이현준, "주요국 혁신의 엔진, 규제 샌드박스", 정보통신산업진흥원 이슈리포트 2019-10호, pp.3-4, 2019.05

[4] 배영임, "규제샌드박스 성공적 안착을 위한 제언", 이슈&진단, pp.1-6, 2019.02

[5] 국가법령센터, http://www.law.go.kr

[6] 이재훈, 정희영, "규제 샌드박스, 한국의 새로운 규제 패러다임", 한국과학기술기획평가원 통권 제261호, 2019.03

[7] 육상현, "규제 샌드박스의 이해", 전자공학회지 46(6), pp.27-34, 2019.06

[8] ICT 규제샌드박스, https://www.sandbox.or.kr

정보보안 사고 사례 분석을 통한 정보보호 평가 항목 제안

주영국*

*고려대학교 컴퓨터정보통신대학원

Proposal of information security evaluation items through analysis of information security incidents

Young-Guk Ju*

*Graduate school of Computer & Information Technology, Korea University

요 약

서비스 거부 공격(DDoS), 랜섬웨어, 개인정보 유출 등 다양한 침해사고가 발생하고 있다. 하지만 정보보안 사고 예방 및 보안 강화를 위한 조직 및 예산 투자는 부족한 것으로 조사되었다. 본 논문에서는 기관 및 기업에서 정보보호 정책을 수립 및 운영할 때 정보보안 사고 사례를 분석하여 사고의 원인이 되었던 위협 사항들을 조치할 수 있도록 정보보호 평가항목을 제안하고자 한다.

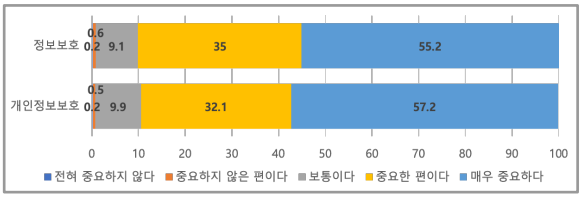
I. 서론

2010년부터 2018년까지 9년간 국내 홈페이지 및 기업에서 발생한 침해사고 신고접수 건수는 매년 증가하는 추세로, 2010년 53건에서 2018년 500건으로 약 9.4배가 증가했다.[1]

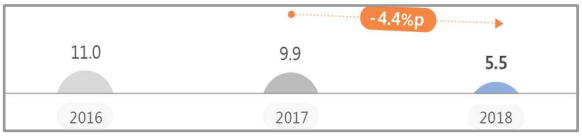
[표 1] 침해사고 신고접수 건수

구분	침해사고 신고접수
2010년	53
2011년	63
2012년	91
2013년	82
2014년	175
2015년	225
2016년	247
2017년	287
2018년	500
총 계	**1,723**

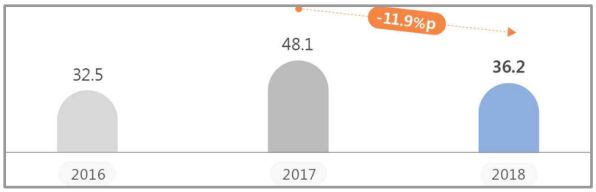
2018년 정보보호 실태조사 결과에 따르면 정보보호 및 개인정보보호의 중요성에 대한 인식은 90.2%, 89.3%로 대다수 정보보호에 대한 중요성을 인식하고 있다. 하지만 정보보호(개인정보보호) 전담 조직을 보유하고 있는 조직은 5.5%이며, 예산을 편성한 기업은 36.2%로 조사되었다. 조사에 따르면 정보보안은 중요하다고 인식하고 있으나, 정보보호 및 개인정보보호를 위한 투자는 미흡한 것으로 나타났다. [2]



[그림 1] 정보보호 및 개인정보보호 중요성 인식률



[그림 2] 정보보호(개인정보보호) 조직 보유율



[그림 3] 정보보호(개인정보보호) 예산 편 성율

본 논문에서는 기관 및 기업에서 정보보호(개인정보보호) 조직 및 예산 편성에 미흡한 부분을 고려하여 정보보안 사고 기반의 최소한의 정보보호 평가항목을 도출하고자 한다.

II. 관련 연구

본 논문에서는 정보보안 사고의 원인을 분석하고 정보보호 및 개인정보보호 관리체계(ISMS-P : Personal Information & Information Security Management System, 이하 ISMS-P) 평가항목과 매핑하여, 정보보호 평가항목을 도출하고자 한다.

2.1 정보보호 및 개인정보보호 관리체계(ISMS-P) 인증의 이해

ISMS-P 인증은 2001년 7월 시행한 정보보호 관리체계(ISMS) 인증과[3] 2010년 11월에 시행한 개인정보보호관리체계(PIMS) 인증이[4] 2018년 11월 정보보호 및 개인정보보호 관리체계(ISMS-P)인증으로 통합된 인증이다.[5] 정보통신망 이용촉진 및 정보보호 등에 관한 법률에 따라 ISMS 인증 의무대상자가 정의되어 있으며, 의무대상자는 ISMS-P인증에서 ISMS 인증만 선택적으로 취득할 수 있다. ISMS-P인증 항목 구성 및 인증 의무대상자는 [표 2]와 [표 3]과 같다.[6]

[표 2] 정보보호 및 개인정보보호 관리체계(ISMS-P) 인증 항목 구성

인증 기준			적용여부	
영역	항목	세부항목	ISMS	ISMS-P
관리체계 수립 및 운영	16	42	O	O
보호대책 요구사항	64	192	O	O
개인정보 처리 단계별 요구사항	22	91	X	O
총계	**102**	**325**	-	-

[표 3] 정보보호 관리체계(ISMS) 의무대상자 기준

구분	의무대상자 기준
ISP (정보통신망 서비스)	「전기통신사업법」 제6조제1항에 따른 허가를 받은 자로서 서울특별시 및 모든 광역시에서 정보통신망서비스를 제공하는 자
IDC (집적정보 통신시설)	정보통신망법 제46조에 따른 집적정보통신시설 사업자
다음의 조건 중 하나라도 해당하는 자	연간 매출액 또는 세입이 1,500억 원 이상인 자 중에서 다음에 해당되는 경우 - 「의료법」 제3조의4에 따른 상급종합병원 - 직전연도 12월 31일 기준으로 재학생 수가 1만 명 이상인 「고등교육법」제2조에 따른 학교
	정보통신서비스 부문 전년도(법인인 경우에는 전 사업연도를 말한다) 매출액이 100억 원 이상인 자
	전년도 직전 3개월간 정보통신서비스 일일평균 이용자 수가 100만 명 이상인 자

2.2 정보보안 사고 사례 분석

본 논문에서 분석할 정보보안 사고 사례는 최근 공격 기법 트랜드를 반영하기 위해 2011년 이후의 정보보안 사고를 대상으로 한다. 사고의 원인이 명확하게 밝혀지지 않은 정보보안 사고는 대상에서 제외했다.

[표 4] 정보보안 사고 분석 대상 목록

일시	피해 기관 및 기업	유형
2011. 3.	정부기관, 포털, 은행	사이트 마비
2011. 4.	현대캐피탈	개인정보유출
2011. 4.	농협	사이트 마비
2011. 5.	한화손해보험	개인정보유출
2011. 7.	SK컴즈	개인정보유출
2012. 7.	KT	개인정보유출
2013. 3.	MBC, KBS, 신한은행, 농협 등	전산망 마비
2013. 4.	SC제일은행	개인정보유출
2013. 4	씨티은행	개인정보유출
2014. 1.	KB국민카드, 롯데카드, NH농협은행 등	개인정보유출
2014. 3.	KT	개인정보유출
2014. 3	국토교통부	개인정보유출

일시	피해 기관 및 기업	유형
2015. 9.	뽐뿌	개인정보유출
2016. 3.	정부 서울 청사	정보 조작
2016. 7.	인터파크	개인정보유출
2016. 9.	국군 사이버 사령부	중요정보유출
2017. 3.	위드이노베이션	개인정보유출
2017. 6.	빗썸	개인정보유출
2017. 6.	나야나	랜섬웨어 감염
2017. 9.	하나투어	개인정보유출
2018. 6	빗썸	암호화폐 탈취

정보보안 사고 원인 분석은 정부기관의 발표자료, 뉴스 보도자료, 보안 전문 업체 분석자료 등의 자료를 참고하였다. 2011년 이후의 21개 정보보안 사고 원인을 분석한 결과 아래 [표 5]와 같은 원인으로 보안사고가 발생한 것으로 분석되었다.[6][7][8][9]

[표 5] 정보보안 사고 주요원인

No.	사고 주요원인
1	관리자 계정 관리 미흡
2	외부자 보안 이행여부 관리 미흡
3	물리적 출입통제 관리 미흡
4	PC 외부 반출입 관리 미흡
5	서버 접근 통제 미흡
6	DB세션 타임아웃 미설정
7	DB서버의 로그 관리 미흡
8	고객 패스워드 암호화 미흡
9	개인정보 암호화, 비식별화 미흡
10	도입 및 개발 시 보안 절차 관리 미흡
11	백업 관리 미흡
12	웹 홈페이지, 어플리케이션 프로토콜 시스템 등 취약성 점검 및 관리 미흡
13	업무용 단말기 보안 미흡
14	보조저장매체 관리 미흡
15	패치시스템 보호대책 미흡
16	백신 소프트웨어 관리 미흡
17	DDoS공격 대응 등 침해사고 대응 체계 미흡
18	개인정보 등 중요자료 유출 모니터링 미흡
19	출력물 보안 관리 미흡

Ⅲ. 정보보호 평가 항목 제안

정보보안 사고 원인을 분석하여 사고의 주요원인을 ISMS-P 인증 항목과 매핑하여 18개의 주요 정보보호 평가 항목을 제안한다.

[표 6] 정보보호 평가 항목

주요원인	평가 항목
관리자 계정 관리 미흡	퇴직 및 직무 변경 관리
외부자 보안 이행여부 관리 미흡	외부자 보안 이행 관리
물리적 출입통제 관리 미흡	출입통제
PC외부 반출입 관리 미흡	반출입 기기 통제
서버 접근 통제 미흡	정보시스템 접근
DB세션 타임아웃 미설정	응용프로그램 접근
DB서버의 로그 관리 미흡	데이터베이스 접근
고객 패스워드 암호화 미흡 개인정보 암호화, 비식별화 미흡	암호정책 적용
도입 및 개발 시 보안 절차 관리 미흡	보안요구사항 정의
백업 관리 미흡	백업 및 복구관리
웹 홈페이지, 어플리케이션 프로토콜 시스템 등 취약성 점검 및 관리 미흡	공개서버 보안
업무용 단말기 보안 미흡	업무용 단말기기 보안
보조저장매체 관리 미흡	보조저장매체 관리
패치시스템 보호대책 미흡	패치관리
백신 소프트웨어 관리 미흡	악성코드 통제
DDoS공격 대응 등 침해사고 대응 체계 미흡	사고 예방 및 대응체계 구축
개인정보 등 중요자료 유출 모니터링 미흡	이상행위 분석 및 모니터링
출력물 보안 관리 미흡	개인정보 표시제한 및 이용 시 보호조치

제안한 18개의 평가항목별 세부 사항은 아래 [표 7]과 같다.

[표 7] 평가항목 및 세부 사항

평가항목	세부 사항
퇴직 및 직무 변경 관리	퇴직, 직무변경 등 인사변경 내용이 유관부서과 공유하고, 정보자산 반납, 접근권한 회수등의 적절한 조치 수행
외부자 보안 이행 관리	외부자가 계약서, 협정서 등에 명시된 정보보호 및 개인정보보호 요구사항 준수여부 주기적 점검 수행
출입통제	보호구역은 출입절차에 따라 출입이 허가된 자만 출입하도록 하고, 출입기록, 접근권한 주기적 검토 수행
반출입 기기 통제	정보시스템, 모바일기기 등을 보호구역 내 반출입하는 경우 정보유출, 악성코드 감염 등의 보안사고 예방 절차 수행
정보시스템 접근	정보시스템 별 적절한 접근 통제를 수행하고, 접속 세션타임 아웃 적용

평가항목	세부 사항
응용프로그램 접근	응용프로그램의 접근 권한을 차등부여하고, 비인가자의 접근 차단
데이터베이스 접근	데이터베이스 내 정보에 접근이 필요한 응용프로그램, 정보시스템, 사용자를 명확히 식별하고 적절한 접근통제 적용
암호정책 적용	개인정보 및 중요정보의 저장/전송/전달 시 암호화 수행
보안요구사항 정의	정보시스템 신규 도입, 개발, 변경 시 시큐어 코딩 등 정보보호 및 개인정보보호 측면의 보안 검토 수행
백업 및 복구관리	백업 대상/주기/방법/절차 등의 백업 정책에 따라 백업을 수행하고, 재해/재난 대처를 위한 백업매체 분리 보관
공개서버 보안	공개서버 DMZ 영역 설치 및 중요정보 외부 노출 여부를 주기적 점검
업무용 단말기기 보안	업무용 단말기의 보안설정, 무선망 통제 등의 보안 정책 정의 및 주기적 점검
보조저장매체 관리	통제구역, 제한구역 등에서의 보조저장매체 사용 제한
패치관리	서버, PC 등 정보자산의 패치정책을 수립 및 수행하며, 패치 관리시스템의 접근통제 등의 보호대책 수행
악성코드 통제	백신 소프트웨어 등 보안프로그램을 설치하고 최신의 상태 유지
사고 예방 및 대응체계 구축	내외부 침해시도 탐지, 대응, 분석, 공유에 대한 절차 수립 및 운용
이상행위 분석 및 모니터링	내외부에 의한 침해시도, 개인정보유출 시도 등을 탐지, 대응할 수있는 절차 및 모니터링 시스템 마련
개인정보 표시제한 및 이용 시 보호조치	개인정보의 조회 및 출력 시 용도데 따라 최소한의 개인정보만 표시하며, 업무상 필요하지 않은 개인정보는 삭제 또는 비식별화 적용

IV. 결론

국내에서 운영 중인 ISMS-P 인증 기준은 21개 분야, 102개 항목, 325개 세부항목으로 구성되어 있어, 항목의 중요성 및 우선순위를 파악하는데 많은 시간이 소요된다.

정보보안 사고는 관리/기술/물리 등 정보보호 영역 중 가장 약한 곳에서 정보보안 사고는 발생한다. 따라서 본 연구에서는 과거 발생한 정보보안 사고를 되풀이 하지 않기 위해 사고를 분석하였으며, 18개의 평가 항목 및 세부사항을 제안한다. 본 논문의 관련연구에서 분석한 정보보안 사고 사례는 제안하는 정보보호 평가항목을 모두 만족했을 때 사고가 발생하지 않았거나, 사고로 발생한 개인정보 유출, 전산망 마비 등의 피해가 감소하였을 것이다.

본 논문에서 제안하는 정보보호 평가항목을 기반으로 내부 정보보안의 약점을 조치한다면 과거에 발생한 정보보안 사고를 예방할 수 있을 것으로 기대한다.

[참고문헌]

[1] e-나라지표 해킹사고 건수 http://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1363

[2] 과학기술정보통신부, 2018년 정보보호 실태조사 요약보고서.pdf, 2019.4

[3] 한국인터넷진흥원, ISMS 인증제도 안내서, pp. 11-26, 2017.4

[4] 한국인터넷진흥원, PIMS 인증제도 안내서, pp. 12-14, 2017.4

[5] 한국인터넷진흥원, 정보보호 및 개인정보보호 관리체계 인증제도 안내서, 2019.1

[6] 한국인터넷진흥원 ISMS-P 제도소개, https://isms.kisa.or.kr/main/ispims/intro

[6] 허은정, 사례분석을 통한 정보보안사고방지방안 연구, pp. 19-26, 2018.6

[7] 김미래, 개인정보 유출사고의 원인분석을 통한 주요 보안위협 선정 및 활용방안에 관한 연구, pp 28-32, 2018.7

[8] 한국인터넷진흥원, 포커스3-국내 주요 인터넷사고 경험을 통해본 침해사고현황.pdf, 2013.10

[9] 위키백과, 대한민국의 정보 보안 사고 목록, https://ko.wikipedia.org/wiki/대한민국의_정보_보안_사고_목록

기초 오펜시브 시큐리티 기법 실습을 위한 파이썬 기반 교보재 개발

이해영*, 김태형*, 임윤수*, 박도규*, 최승용*

*청주대학교 소프트웨어융합학부

Development of Python Based Training Aids for Basic Offensive Security Technique Exercises

Hae Young Lee*, Tae Hyeng Kim*, Yoonsu Lim*, Do Kyu Park*, Seongyong Choi*

*Division of Software Convergence, Cheongju University

요 약

본 논문은 대학 학부 과정에서 기초적인 오펜시브 시큐리티 기법을 실습할 수 있는 파이썬 기반 교보재를 설명한다. 학생이 주어진 파이썬 스니펫을 확장하여 직접 오펜시브 시큐리티 기법을 코드로 작성 및 실행하고, 공격에 대해 대응 기법을 실습할 수 있다. 필요한 경우 대상 체계를 취약하도록 만드는 설정 파일과, 수업 후 과제로 부여할 수 있는 예제도 제공한다. 이를 통해 학생의 체계에 대한 이해도 및 정보보호 교육 효과를 높일 수 있을 것으로 기대한다.

I. 서론

보안 교육에서 오펜시브 시큐리티 기법을 활용하여 실습을 수행하면, 대상 체계에 대한 이해를 향상시킬 수 있으며, 이에 따라 교육 효과의 향상도 기대할 수 있다[1]. 그러나 현재 오펜시브 시큐리티 기법 교육 대부분[1~3]이 단순 공격 도구를 활용하는 수준에 머물고 있다. 예를 들어, arpspoof 도구[4]로 ARP 스푸핑(ARP spoofing)을 수행하는 수준이다. 만약 학습자가 직접 오펜시브 시큐리티 기법을 코드로 작성하여 실습할 수 있다면, 체계에 대한 이해 및 교육 효과도록 보다 더 향상시킬 수 있을 것이다.

본 논문에서는 기초적인 오펜시브 시큐리티 기법을 실습하는데 활용할 수 있는 파이썬(Python) 기반 교보재를 설명한다. 교보재는 학부생 대상 교육을 가정하며, 실습 시간에 활용할 수 있는 오펜시브 기법 스니펫 모음(snippets), 필요한 경우 대상 호스트를 취약하게 만드는 설정 파일(setups), 그리고 실습 후 복습 및 심화 학습을 위한 과제 예제(assignments)로 구성된다.

II. 파이썬 기반 실습 교보재

본 절에서는 기초 오펜시브 시큐리티 기법 실습에서 활용할 수 있는 파이썬 스니펫에 대해 설명한다.

2.1 스니펫 모음

파이썬 스니펫 모음은 대학 학부 전공 과정에서 기초적인 보안 공격 기법을 실습할 때 교보재로 활용할 수 있도록 개발 중이며, 깃헙(GitHub)을 통해 제공[1]된다. 각 스니펫은 해당 보안 공격 기법 실습에 필요한 핵심적인 코드만을 포함한다. 학생은 실습 환경 및 주어진 정

1) https://github.com/haelee/allbypythonself

보에 맞춰 스니펫을 수정·보완하며 보안 공격 기법을 실습할 수 있으며, 이에 대한 대응 기법도 함께 실습할 수 있다.

<그림 1>의 텔넷 대상 사전 공격(dictionary attack) 스니펫(dictionary.py)으로, telnetlib을 사용하여 대상 호스트의 텔넷 서버와 연결하고, 로그인을 수행하는 코드만 포함한다. 학생은 스니펫을 주어진 계정 ID와 사전에 있는 단어를 반복적으로 입력하는 코드로 발전시키고, 이를 실행하여 대상 ID의 패스워드를 알아낸다. 학생은 대응으로 패스워드 복잡성 및 계정 잠금 임계값을 설정한다[5].

```
import os
import telnetlib

t = telnetlib . Telnet ("10.0.2.4")
t . read_until ("login: ")
t . write ("victim\n")
t . read_until ("Password": ")
t . write ("rockyou\n")
t . close ()
```

<그림 1> 사전 공격 실습용 스니펫

<표 1> 파이썬 스니펫 목록

분야	스니펫	기법
암호 및 시스템	crack.py	패스워드 크래킹
	dictionary.py	사전 공격
네트워크	arpspoof.py	ARP 스푸핑
	icmpredirect.py	ICMP 리다이렉트 공격
	portscan.py	SYN 스캐닝
	smurf.py	스머프 공격
	synflood.py	SYN 플러딩
	tcpreset.py	TCP 리셋 공격

<표 1>은 2019년 10월 1일 현재 제공되는 파이썬 스니펫 목록이다. 네트워크 분야 실습용 스니펫은 데이터 링크 수준에서 패킷을 조작할 수 있는 Scapy 프로그램[6]을 사용한다. 시스템 분야 실습용 스니펫 중, 패스워드 크래킹 스니펫은 리눅스 crypt 루틴과의 인터페이스를 구현한 crypt 모듈을, 사전 공격 스니펫은 텔넷 클라이언트를 구현한 telnetlib 모듈을 사용한다.

2.2 설정 파일

제공 스니펫 중 일부는 대상 호스트에 취약점이 존재해야 실습이 가능하다. 예를 들어, 스머프 공격(Smurf attack)은 로컬 네트워크 내의 모든 호스트에서 net.ipv4.icmp_echo_ignore_broadcasts를 0으로 설정해야 공격이 가능하다. 설정 파일은 이와 같은 설정을 수행하는 스크립트 모음이다. 학생은 공격 기법 실습 전 대상 호스트에서 해당 설정 파일을 실행한다.

2.2 과제 예제

실습수업 후 학생에게 스니펫을 보다 발전시키도록 과제를 부여할 수 있다. 예를 들어, 패스워드 크래킹 스니펫은 실습에서 하드 코딩된 해시 값을 사용자로부터 입력받거나 /etc/shadow 파일에서 읽어오도록 수정할 수 있다. 사전 공격 스니펫의 경우, 스레드를 사용하여 보다 효율을 높일 수도 있다. 과제 예제는 이러한 과제의 예로, 현재 학부 1~2학년 수준으로 제공한다.

III. 결론 및 향후 연구

본 논문은 학부 교과에서 보안 공격·방어 메커니즘에 대한 이해를 돕기 위한 파이썬 기반의 실습 교보재를 소개하였다. 실습 교보재는 스니펫, 설정 파일, 과제 예제로 구성되며, 스니펫을 활용한 보안 공격 코드 작성, 설정 파일을 사용한 취약점 설정, 보안 공격·방어 수행, 공격 코드 보완하는 형태로 실습 및 과제를 진행할 수 있다.

파이썬 기반 실습 교보재를 네트워크 보안, 암호학, 운영체제 관련 학부 교과 및 동아리 비교과 프로그램에서 활용하였다. 정식으로 설문을 실시하지는 않았으나, 학생의 피드백(강의평가 등)을 볼 때 공격·방어 메커니즘 이해에 많

은 도움이 된 것으로 평가된다.

향후에는 보다 다양한 오펜시브 시큐리티 기법 실습을 위한 교보재를 제공할 계획이다. 우선적으로 DNS 캐시 오염, DNS 증폭 공격 등이 추가될 예정이다. 또한, 정규 교과 활용 결과를 조사할 계획이다.

[참고문헌]

[1] B. Wilson, "eaching security defense through web-based hacking at the undergraduate level," *Journal of Computing Science in Colleges*, 2017.

[2] M. Timchenko, D. Starobinski, "A Simple Laboratory Environment for Real-World Offensive Security Education," *Proc. of ACM SIGCSE*, 2015.

[3] W. Du, "The SEED Project: Providing Hands-on Lab Exercises for Computer Security Education," *IEEE Security and Privacy Magazine*, 2011.

[4] arpspoof. https://su2.info/doc/arpspoof.php

[5] 한국인터넷진흥원, 주요정보통신기반시설 기술적 취약점 분석·평가 방법 상세가이드, 2017.12.

[6] Scapy. https://scapy.net

개인인증을 위한 심전도 데이터 취급 간 준용해야 할 유의사항 및 관련 지침

김경훈, 김동영, 김준서, 박예찬, 박인현, 오명교, 이우진, 이현정, 최재진*

*한국정보기술연구원 Best of the Best

Precautions and Related Guidelines while Handling ECG Data for Personal Authentication

Kyung-Hoon Kim, Dong-Young Kim, Jun-Seo Kim, Ye-Chan Park, In-Hyun Bark, Myung-Gyo Oh, Woo-Jin Lee, Hyung-Jung Lee, Jae-Jin Choi*

*Korea Information Technology Research Institute Best of the Best.

요 약

지문, 홍채, 안면 등 신체의 해부학적 지표를 이용하여 본인인증을 수행하는 일반적인 바이오인증 방식의 여러 제한점으로 인해 생리학적 지표 중에서 가장 인식률이 우수하다고 알려진 심전도를 중심으로 다양한 연구가 진행되고 있지만, 정보통신망법상 개인정보로 분류되는 심전도 데이터의 취급 간 유의사항은 다소 경시되어 있다. 이에 본 논문을 통해 심전도 데이터를 포함한 바이오정보의 수집・저장・이용・파기 간 개인정보 관리자가 준수해야 할 법령, 지침 및 가이드라인을 제시함으로써 관련 연구자들에게 개인정보 보호에 대한 경각심을 제고하고, 향후 데이터 유출을 비롯한 침해사고 발생 시 법적 피해를 최소화하는 데에 공헌하고자 한다.

I. 서론

지문, 홍채, 안면 등 사람의 해부학적 지표에 기반한 바이오인식 기술은 핀테크, 헬스케어, 위치기반서비스 등 IoT 서비스의 확대로 해킹이나 개인정보 유출의 위협에 대비하기 위한 안전한 보안 기술로 부상하고 있다[12]. 또한 공인인증서 사용 의무 폐지 및 사용 강제 규정 개정[13], 비대면 금융거래 간 바이오인증을 포함한 복수의 방식 허용[4] 등 금융규제 완화로 인하여 금융 분야를 포함한 컴퓨터 보안, 통신, 출입관리, 의료복지, 공공분야 및 검역 등 광범위하게 적용될 것으로 예상된다[14].

하지만 현재 상용화되어 사용하고 있는 대부분의 바이오인식 기술은 사용자의 부재 시에도 사용이 가능하다는 제한점이 존재하고, 지문, 홍채 등 해부학적 지표를 이용한 바이오인식 기술은 지문패턴의 손실과 같이 일부 바이오정보의 채취에 대한 어려움과 함께 위변조의 위협이 존재하여 한국인터넷진흥원, 한국정보통신기술협회 바이오인식 프로젝트 그룹(PG505) 등 다양한 연구기관에서는 생리학적 특성인 뇌파・심전도・근전도 등의 생체신호 중 자체 시험결과 가장 개인 식별성이 우수하다고 판단되는 심전도를 대상으로 새로운 바이오인증 기술을 개발하고 있다[8].

바이오정보의 일종인 심전도 데이터는 개인을 식별할 수 있는 신체적 또는 행동적 특징에 관한 정보이기 때문에[9] 정보통신망 이용촉진 및 정보보호 등에 관한 법률(이하 정보통신망법)상 개인정보에 해당한다. 따라서 개인정보의 법적 생명주기 요구사항에 따라 수집・저장・이용・파기되어야 하지만, 다양한 연구에서 개인정보 생명주기의 적법한 절차에 관한 내용은 어디에서도 찾아보기 어려웠다[6, 9, 16,

18-19].

이에 본 논문에서는 심전도 데이터를 제공하는 연구대상자(이하 정보주체)로부터 개인인증 연구를 목적으로 심전도 데이터를 수집하여 활용하고자 하는 개인, 단체 또는 기관(이하 사용자)이 수집・저장・이용・파기 간 준수해야 하는 관련 법령을 나열하고 이에 따른 적법한 절차를 제시하고자 한다. 따라서 심전도뿐만 아니라 다양한 바이오정보를 수집하여 활용하고자 하는 관련 연구자들에게 개인정보보호의 관심 및 경각심을 제고하고, 향후 정보유출을 비롯한 침해사고 발생 시 민・형사적 책임이 공격자가 아닌 관리자에게 부담시킴으로써 기존 법제를 준용하여 법적 피해를 최소화하는데 그 목적을 둔다.

II. 심전도 데이터 취급 간 유의사항 및 관련 지침

2.1. 개요

개인인증을 위한 심전도 데이터 수집・저장・이용・파기간 사용자가 고려해야 할 관련 법령, 행정규칙, 표준 및 가이드라인은 아래와 같다.

관련 규정	이용 단계			
	수집	저장	이용	파기
개인인증을 위한 생체신호 정보 시험용 DB 구축지침	○	○		
개인인증용 심전도 및 광용적맥파 특징점 데이터 교환 포맷		○	○	
개인정보의 기술적・관리적 보호조치 기준		○	○	
개인정보의 안전성 확보조치 기준		○	○	○
개인정보의 암호화 조치 안내서		○	○	
개인정보 보호법	○	○	○	○
개정「정보통신망법」중 개인정보보호규정 안내서	○	○		○
금융보안을 위한 바이오인식 운영 지침	○	○	○	○
바이오정보 보호 가이드라인	○	○	○	○
정보통신망법	○	○	○	○

Table 1. 심전도 데이터 취급 간 단계별 참고 법령 (가나다순)

2.2 수집

개인인증을 위해 심전도 데이터를 수집하고자 할 때 가장 먼저 고려되어야 하는 사항은 심전도 데이터의 수집 및 이용 목적과 보유 기간의 정의이다. 사용자는 개인정보를 수집하는 경우 정보통신망법 제22조제1항에 따라 개인정보의 수집 및 이용 목적, 항목, 보유 및 이용 기간을 정보주체에게 명확히 알리고 동의를 받아야 하며, 개인정보 보호법 제22조제1항에 따라 각각의 동의 사항을 구분하여 정보주체가 이를 명확하게 인지할 수 있도록 알리고 각각 동의를 받아야 한다. 따라서 정보주체는 생체정보 수집 동의서 작성 간 아래와 같은 항목의 검토 및 작성이 필요하다.

목적	항목	보유・이용기간	동의여부
이용자 식별 및 본인 인증	심전도정보 (원본/특징정보)	원본정보 : 특징정보 생성 시 까지 특징정보 : 2019년 12월 31일 까지	□ 동의함 □ 동의안함

Table 2. 심전도정보 필수 수집 및 이용 동의 예시

심전도 데이터는 크게 수집한 심전도의 날 값인 원본데이터와 날 값의 데이터 가공을 통해 특징을 추출한 특징데이터로 나누어진다[11]. 원본데이터의 경우 특징데이터가 생성되는 순간 지체 없이 파기하는 것이 원칙이지만, 특징데이터 생성 이후에도 알고리즘 고도화와 같은 이유로 원본데이터를 파기하지 않고자 한다면 명확한 이유(목적)와 보유 기간을 기존 심전도 데이터의 수집・이용 동의와 구분하여 고지 후 동의를 받아야한다[11]. 특징정보 추출 이후에 원본데이터를 추가적으로 보유하는 경우 이를 동의하지 않는다는 이유로 서비스의 제공을 거부하여서는 안 되며 이는 정보통신망법 제23조제3항의 위반사항이라고 할 수 있다.

목적	항목	보유・이용기간	동의여부
심전도 인증 알고리즘 고도화	심전도정보 (원본정보)	2019년 12월 31일 까지	□ 동의함 □ 동의안함

Table 3. 원본정보 선택 수집 및 이용 동의 예시

사용자는 정보통신망법 제23조제1항, 개인정보 보호법 제3조제1항에 따라 정보주체에게 인증 또는 식별 목적에 필요한 범위에서 최소한의 개인정보를 수집・이용해야 한다. 만일 정보주체의 개인 식별을 위해 심전도 데이터 이외에 추가적으로 개인정보를 수집・저장하고자 하는 경우, 명확한 근거에 기초한 최소한의 개인정보(이름 및 전화번호 등)만을 수집하여야 한다.

본 생체신호 평가기관은 수집된 심전도 데이터의 이용자 식별을 위해 최소한의 개인정보를 수집합니다.	
◎ 이름 :	◎ 전화번호 :

Table 4. 정보주체의 인적사항에 대한 필수 수집 동의 예시

사용자는 만 14세 미만의 아동으로부터 심전도 데이터를 수집・이용・제공 등의 동의를 받고자 하는 경우 정보통신망법 제31조제1항에 따라 정보주체의 법정대리인 동의가 필수적이며 이를 위반하는 경우 동법 제71조제1항제8호에 따라 처벌받을 수 있다.

□ 만 14세 이상입니다.

Table 5. 법정대리인 필요 여부 판단을 위한 필수 수집 동의 예시

정보주체는 정보통신망법 제30조에 따라 사용자에게 수집된 심전도 데이터의 수집・이용・제공 등의 동의를 철회하거나 정정을 요구할 수 있다. 따라서 사용자는 개인정보 보호법 제38조제4항에 따라 정보주체가 개인정보의 열람, 정정・삭제, 처리 정지 등의 요구를 할 수 있는 구체적인 방법과 절차를 마련하고, 이를 정보주체가 알 수 있도록 공개하여야 한다. 또한 동법 동조 제5항에 따라 정보주체의 열람 등 요구에 대한 거절 등 조치에 대하여 불복이 있는 경우 이의를 제기할 수 있도록 필요한 절차를 마련하고 안내하여야 한다.

연구책임자 : KITRI BEST OF THE BEST 8기 오명교	☎ 010-4929-7541
서울특별시 금천구 서부샛길 606 대성디폴리스지식산업센터 A동 27층 BEST OF THE BEST	

Table 5. 연구책임자의 소속 및 연락수단 등의 명시 예시

2.3 저장 및 이용

수집된 심전도 데이터는 정보통신망법상 개인정보이므로 개인정보의 기술적・관리적 보호조치 기준에 따라 관리되어야 한다. 동 기준 제6조제3항에 따르면 정보통신망을 통해 이용자의 개인정보 및 인증정보를 송・수신하고자 하는 경우 SSL 인증서나 웹서버에 암호화 응용프로그램을 설치해야 한다고 명시하고 있다. 따라서 심전도 측정 센서 등의 장치를 통해 심전도 데이터를 수집 및 입력할 때 제3자에 의해 위・변조된 심전도 데이터가 처리되지 않도록 적절한 보안 조치를 취하여야 한다.

사용자는 정보통신망법 제28조, 개인정보의 기술적・관리적 보호조치 기준 제6조제1항, 동법 동조 제4항, 개인정보의 안전성 확보조치 기준 제7조 등에 따라 정보주체로부터 획득한 심전도 데이터를 정보통신망을 통하여 송신하거나 기기, 보조저장매체 등에 저장 및 이를 이용하여 전달할 경우 안전한 암호 알고리즘으로 데이터를 암호화하여 저장하여야 한다. 만약 심전도 데이터를 서버로 전송하여 처리한다면 데이터 유출과 같은 침해사고 발생 시 피해 범위가 커지므로, 수집된 기기에서 암호화 처리 후 전송하거나 보관하는 조치가 우선적으로 고려되어야한다 [11].

이때 '안전한 암호 알고리즘'이란 국내 및 미국, 일본, 유럽 등의 국외 암호 연구 관련 기관에서 권고하는 암호 알고리즘을 의미하며, 공공기관은 국가정보원의 검증대상 암호 알고리즘을, 민간부문(법인・단체・개인)은 국내・외 전문기관(KISA, NIST, ECRYPT 등)에서 권고하는 암호 알고리즘을 기반으로 개인정보 암호화에 사용할 수 있다 [22].

사용자가 정보주체로부터 수집한 심전도 데이터를 업무용 컴퓨터 및 보조저장매체에 암호화하여 저장하고자 하는 경우, 문서 도구 자체 암호화(한글파일 저장 간 비밀번호 설정 등), 암호 유틸리티를 이용한 암호화(파일압축 간 암호설정, Windows OS의 BitLocker 기술 등), DRM, 디스크 암호화 등의 방법을 선택할 수 있다 [22].

2.4 파기

사용자는 수집한 심전도 원본데이터에서 특징데이터를 추출하였을 경우, 원본데이터의 수집 및 이용 목적을 달성한 것으로 볼 수 있으므로 정보통신망법 제29조제1항에 따라 지체 없이 해당 개인정보를 복구・재생할 수 없도록 파기하여야 한다. 다만, 사용자의 필요로 정보주체의 동의를 받아 원본정보를 이용하는 경우에는 동의 받은 목적이 달성되거나, 보유 및 이용기간이 종료되는 경우 지체 없이 원본 정보를 파기하여야 한다 [11].

특징데이터 역시 정보주체의 수집・이용・제공 동의 등의 철회 요청이 발생한다면 즉각 파기를 진행하여야 하나, 특정한 이유로 즉각 파기가 어려운 경우 이 내용을 알리고 파기 가능한 날짜를 고지하여야 한다. 이때 파기 요청이 들어온 날짜부터는 해당 데이터를 분리 보관하고 사용해서는 안 되며, 데이터의 접근을 막기 위해 물리적으로 분리하는 것이 바람직하다. 부득이하게 물리적으로 분리가 어려운 경우라면 물리적 분리와 동등한 수준으로 논리적 분리하여 관리하여야 한다[11].

심전도 데이터의 파기를 진행하는 경우 원본 데이터의 복구・재생할 수 없도록 하여야한다. 저장매체 전체의 데이터를 파기하는 경우 로우레벨포멧이나 와이핑과 같은 프로그램을 이용하거나 디스크 플레터 천공, 파쇄기, 소각, 디가우저와 같은 물리적인 파기를 실시한다. DB서버에 저장된 일부 데이터를 파기하는 경우라면 서비스 중인 DB의 해당 개인정보 위에 임의의 값(Null값 등)을 덮어쓰기한 후 삭제한다. DB의 특정 부분에 덮어쓰기가 곤란한 경우라면 테이블 데이터에 대한 논리적인 삭제도 허용되나, 신속하게 다른 데이터로 덮어쓰기 될 수 있도록 운영하여야 한다[10].

III. 결론

본 연구는 심전도 인증 시스템의 상용화에 앞서 심전도 데이터의 원활한 수집・저장・이용・파기와 같은 생체주기의 관리를 위하여 기존 연구와 앞서 이루어질 연구에 대한 법적 주의사항을 단계별로 정리함으로써 심전도 인증의 도입 및 상용화를 위한 법적 기틀을 제시하고자 하였다. 다만, 개인정보의 관리에 관한 법제도 및 가이드라인 등은 현재 지속해서 제・개정되고 있으므로 사용자의 관련 법령 등에 대한 재확인이 사전에 필수적으로 이루어져야 할 것이다.

향후 연구로는 심전도 데이터 뿐 아니라 광용적맥파를 비롯한 다양한 생체신호 및 지문, 홍채, 정맥 등의 통상적인 바이오정보에 대한 개인정보 관리실태를 지속 확인 및 연구하여 다양한 사용자들에게 바이오정보 취급방법을 제고하기 위한 연구를 지속할 예정이다.

[참고문헌]

[1] 개인정보 보호법, 법률 제14839호 (2017).

[2] 개인정보의 안전성 확보조치 기준, 행정안전부고시 제2019-47호 (2019).

[3] 개인정보의 기술적·관리적 보호조치 기준, 방송통신위원회고시 제2015-3호 (2015).

[4] 금융위원회. (2016). 비대면 실명확인 운영 현황 및 향후 계획. 서울: 동 기관.

[5] 김동진. (2016). 바이오인증 최신 활용 및 보안 동향. 용인: 금융보안원.

[6] 김슬기, 김대은. (2018). 손가락에서 측정된 심전도 파형 모양을 이용한 생체인식. 제어로봇시스템학회 국내학술대회 논문집, 155-156.

[7] 김재성. (2016, 05-06). 생체신호 인증기술 및 표준화 동향, TTA Journal, 165, 18-24.

[8] 김재성. (2017). 다중 생체신호를 이용한 인증 알고리즘 기술. 나주: 한국인터넷진흥원.

[9] 김지훈, 박광석. (2016.05-06). 생체신호를 이용한 개인인증 기술 및 DB 구축. TTA Journal, 165, 41-46.

[10] 방송통신위원회. (2015). 개정 「정보통신망법」 중 개인정보보호규정 안내서. 과천: 동 기관.

[11] 방송통신위원회, 한국인터넷진흥원. (2017). 바이오정보 보호 가이드라인. 나주: 동 기관.

[12] 연구성과실용화진흥원. (2016, 02). 생체인식 기술 및 시장동향. 서울: 동 기관.

[13] 유신투자증권. (2017). 신정부출범과 4차산업혁명 - 생체인증. 서울: 동 기관.

[14] 윤일영. (2018, 02). 바이오와 보안의 융합, 생체인식 기술. 서울: 융합정책연구센터.

[15] 이새움, 김재성. (2017). 심전도를 이용한 생체신호 인증 알고리즘. 한국통신학회논문지, 42(12), 2350-2356.

[16] 임서현, 민경란, 이종실, 장동표, 김인영. (2014). 단일 리드 심전도를 이용한 개인 식별. 의공학회지, 35, 42-49.

[17] 정보통신망 이용촉진 및 정보보호 등에 관한 법률, 법률 제16021호 (2018).

[18] 조주희, 조병준, 이대종, 전명근. (2017). 주성분 분석기법을 이용한 심전도 기반 개인인증. 전기학회논문지 P, 66P(4), 258-262.

[19] 한국정보통신기술협회. (2016). 개인인증을 위한 생체신호 정보 시험용 DB 구축지침. 성남: 동 협회.

[20] 한국정보통신기술협회. (2016). 금융보안을 위한 바이오인식 운영 지침. 성남: 동 협회.

[21] 한승진. (2016, 05-06). 금융권 바이오인식 기술 운영 현황. TTA Journal, 165, 25-35.

[22] 행정자치부, 한국인터넷진흥원. (2017). 개인정보의 암호화 조치 안내서. 나주: 동 기관.

기업환경에서 초기 IoT 보안 도입을 위한 지침

조한섭*

*고려대학교 컴퓨터정보통신대학원

Guidelines for Initial Adoption of IoT Security in the Enterprise Environment

Han-Seob Jo*

*Graduate School of Computer & Information Technology, Korea University.

요 약

본 논문은 기업환경에서 초기 IoT 보안 도입을 위한 지침을 수립함으로써 기업 내 영업 비밀을 안전하게 보호하고, 정보의 유출 예방 및 서비스 중단 예방을 목적으로 한다. 전체 IoT 서비스는 센서, 기기에서부터 응용서비스까지 광범위하지만 본 논문은 센서, 기기, 게이트웨이 및 근거리 저전력 통신을 대상으로 하며 그 외 대상은 어플리케이션, 인프라, 네트워크 등 영역별 존재하는 보안 지침을 적용함으로써 혼재되고 중복되는 보안 부분을 제거하여 초기 IoT 보안을 도입하는 기업환경에 효율적 지침을 제공하는데 목적이 있다.

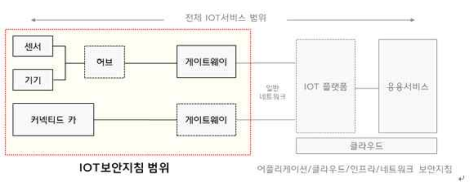
I. 서론

본 논문은 기업환경에서 초기 IoT 보안 도입을 위한 지침을 수립함으로써 기업 내 영업 비밀을 안전하게 보호하고 정보의 유출 예방 및 서비스 중단 예방을 목적으로 한다.

1. 총칙

1.1 적용 범위

전체 IoT 서비스는 센서, 기기부터 응용서비스까지이나, 본 IoT 보안 지침은 센서, 기기, 게이트웨이 및 근거리 저전력 통신을 대상으로 하며, 그 외 대상은 영역별 보안 지침을 따라 적용한다.



[그림 1] IoT 보안지침 범위

1.2 관련 규정

본 지침에서 규정하지 않은 보안사항은 어플리케이션, 인프라, 네트워크 등 영역별 존재하는 보안 지침을 동일하게 준수해야 한다.

기기간 상호인증 구현 시 국제 표준을 준수한다.

개인정보를 취급할 경우 기기가 이용되고 데이터를 수집하는 권역의 법을 준수하여 관리한다.

기기 혹은 서비스는 비즈니스 성격에 따라 관련 국제표준을 준수하여 도입 및 개발, 관리, 운영되어야 한다. 특히, 상용서비스 연동 시 국제표준을 우선하여 준수한다.

2. IoT 보안 절차

2.1 절차 설명

IoT 기술을 활용한 시스템을 구축, 변경하는 경우 기기 및 기술에 대한 적정한 보안 세부

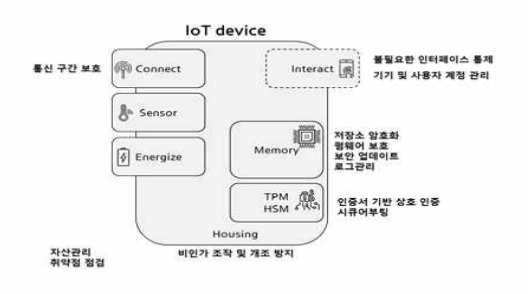
항목을 선별하기 위해 IoT 보안 심의를 수행한다.

2.2 심의 절차.

보안팀과 업무주관 부서는 적용되는 기기와 기술에 대한 IoT 보안 세부 항목을 정한다.

업무주관 부서는 선정된 보안 세부 항목에 따라 도입되는 기기를 선정하고 조치한다.

3. IoT 보안 세부 항목



[그림 2] IoT 보안 세부 항목

3.1 통신 구간 보호

IoT 기기와 서버 간 통신 시 중요 정보는 평문 전송하지 않는다.

중요 정보를 전송 시에는 통신 구간을 암호화 하여 전송한다. 통신 구간 암호화 기술은 SSL을 원칙으로 하고 SSL을 적용하기 어려운 경우 암호화된 소켓 통신을 개발하여 적용한다.

SSL을 적용한 경우에도 공격자의 공격이 가능하므로 중요한 값은 암호화 또는 마스킹 처리해야 한다.

SSL 서버 인증서를 검증하여 신뢰할 수 있는 경우에만(공인인증) 통신을 허용한다.

망 외 접속시에는 IPSec VPN, SSL VPN으로 통신을 암호화해야 한다.

공장 내 근거리 통신을 구성하는 경우에도 인증과 암호화를 지원하지 않는 통신 프로토콜(IR 적외선 등)은 사용하지 않는다.

3.2 저장소 암호화

IoT 기기 내에 데이터를 저장하지 않도록 하며, 데이터가 저장되는 경우 암호화 하여 저장한다.

IoT 기기 내 중요 정보를 저장해야 하는 경우에는 하드웨어적인 보안 모듈(TPM)을 이용하여 데이터를 암호화 저장해야 한다.

암호화 기능이 제공되지 않거나 안전한 암호화 기준을 만족하지 못하는 알고리즘을 제공하는 기기에는 중요 정보를 저장하지 않는다.

암호화된 정보를 암/복호화하는 함수와 암호화 키는 안전한 하드웨어(ROT)에 저장한다.

IoT 기기의 초기 설정 시 보안 모듈과 데이터 암호화를 활성화한다.

3.3 펌웨어 보호

OS구동 시 시큐어부팅을 통해 서명된 OS 동작만을 허용해야 한다.

펌웨어 설치 시 백도어 코드 및 계정 여부를 확인해야 한다.

펌웨어 버전 출력이 되지 않도록 해야 한다.

펌웨어 다운그레이드(이전 버전, 덜 안전한 버전)를 방지해야 한다.

펌웨어에 포함된 중요 정보를 분석할 수 없도록 암호화 또는 난독화 한다.

기기 내에서 구동되는 프로세스는 시스템 자원(메모리, 파일 등)에 접근하는 권한을 최소화하여 제어한다.

인증되지 않은 소프트웨어와 파일이 기기에 설치되는 것을 방지하기 위한 기능이 있어야 한다.

원격으로 소프트웨어 업그레이드를 하는 경우 소프트웨어 이미지는 승인 된 서명 확인 후 진행되어야 한다.

기기의 소프트웨어 서명 루트는 변경 방지 메모리(ROT)에 저장되어야 한다.

원격 업그레이드 소프트웨어는 전송되는 동안 암호화되어야 한다.

워치 독 타이머는 비활성화 할 수 없도록 해야 한다.

개발, 테스트 이후 운영 되는 기기 내 소프트웨어는 모든 디버그 기능을 해제한다.

펌웨어 및 소프트웨어는 코딩 보안 표준을 준수하여 개발되어야 하고 각종 위협에 대한 모의 테스트를 통해 보안 취약점을 검증해야 한다.

3.4 인증서 기반 상호 인증

기기의 자격 증명이나 암호화 키는 신뢰할 수 있는 저장소(ROT)에 보관하여 고유 기기를 식별하고 인증하도록 한다.

보안 속성값은(UID 등) 보안칩 기반(TPM)으로 제공하는 암호화 방식 또는 네트워크 보

안 프로토콜을 사용하여 상호인증을 수행하며 데이터를 보호한다.

기기 간 상호인증 구현 시 국제 표준을 준수한다.

기기 간 상호인증 시, 인증방법에 따라 신뢰할 수 있는 시간에 대한 인증이 가능해야하며 이를 통해 무결성을 검증할 수 있어야 한다.

일정 횟수이상 반복되는 무단 상호인증 실패에 대해 차단한다.

3.5 보안 업데이트

업그레이드를 받는 기기는 무결성이 훼손된 시스템과 통신하지 않도록 소프트웨어의 신원, 무결성, 목록을 검증해야 한다.

하드웨어 기반 RoT(Root of Trust)를 통해 기기와 업데이트 서버 간 안전하게 통신 할 수 있도록 보장한다.

업그레이드 전 소프트웨어의 승인 된 서명을 확인 후 업그레이드 수행한다.

원격 업그레이드 소프트웨어는 암호화 되어야 하며 암호화 통신을 통해 파일을 전송해야 한다.

업데이트 서버와 기기 사이에 상호 인증 기능을 제공하여 위장 서버나 중간자 공격 등의 취약점에 대응해야 한다.

업데이트 서버와 기기는 무결성을 확보된 상태에서 업데이트를 수행한다. 무결성이 확보되지 않은 기기는 자동 업데이트를 적용하지 않는다.

펌웨어 업데이트 시 파일 고유 해시값을 비교하여 변조 여부에 대한 무결성 검증을 해야하며, AES-128 이상의 암호화 알고리즘을 사용해야 한다.

업데이트 시 노출되는 정보를 최소화해야 한다.

주기적으로 펌웨어를 업데이트하여 최신을 유지한다.

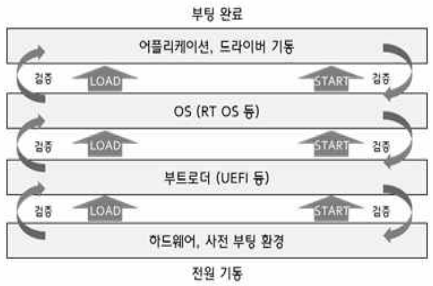
보안 부팅을 지원하지 않는 경우 펌웨어 업데이트가 설치되면 사용자 데이터와 암호를 지우고 해당 기기 사용을 정지해야 한다.

기기의 업데이트 신뢰성을 확인할 수 없는 경우는 운영자가 물리적으로 직접 업데이트한다.

3.6 시큐어부트(Secure Boot)

시큐어부트(Secure Boot)가 지원되는 TPM 칩이 장착된 기기는 기본 사용으로 설정해야 한다.

각 부트 단계마다 미리 생성한 서명과 각 부트 과정에서 생성한 서명이 서로 일치하지 않을 경우, 부팅을 중단함으로써 손상 및 검증되지 않은 OS의 실행을 방지해야 한다.



[그림 3] 시큐어부트 단계 별 검증 과정

시스템 부팅 프로세스가 있는 기기는 환경설정인 부트로더(예: BIOS 설정)를 임의 조작하지 못하도록 비활성화해야 한다. 만일 부트로더를 임의 조작하도록 해야 한다면 반드시 별도 프로그램을 통해 인증을 통하여 접근 후 설정하도록 해야 한다.

3.7 불필요한 인터페이스 통제

기기 외부와 통신이 가능한 포트 및 인터페이스는 최소화해야 한다.

불필요한 포트는 기기를 운영 적용하기 전 제거해야 한다.

해당 포트를 사용 할 경우는 허가되고 인증된 개체하고만 통신 가능하도록 제한한다.

관리자 / CLI 인터페이스 접근을 기본으로 허용하지 않아야 한다.

기기 내 비밀번호는 필수 설정하고 초기 비밀번호는 변경한다.

3.8 비인가 조작 및 개조 방지

기기는 관리자 승인 없이 분해되어서는 안된다.

기기에 대한 비인가 조작 및 개조를 방지(Tamper Proofing)하기 위한 솔루션이 적용되어야 한다.

비인가 조작 탐지 시 주요키 및 민감 데이터가 초기화되도록 해야 한다. 초기화가 어려울

경우, 기기 자체가 비활성화 되어야 한다.

역공학을 막기 위한 보호 장치가 있어야 한다.

공장 초기화 시 기기가 업데이트되지 않으면 정상 작동되지 않음을 경고해야 한다.

3.9 기기 및 사용자 계정 관리

기기 및 사용자는 관리 (등록, 초기화, 분실, 폐기 등) 되어야 한다.

다중 요소 인증이 제공되는 기기는 이를 적용한다.

사용자 권한은 최소 권한으로 초기 설정되어야 한다.

계정정보, 서명 값, OS 접속정보, 암호화 키 등의 중요한 값은 노출되지 않아야 한다.

사용자 별 기능 접근 제어가 제공되어야 하며, 사용자 변경 시 재 인증 기능이 제공되어야 한다.

인증 토큰 및 세션 키가 중복 발급되지 않아야 한다.

계정, 기기 ID는 고유하게 (1개씩) 부여되어야 한다.

기기의 초기 설정/재설정 단계에서 기본 인증정보(아이디, 비밀번호 등)를 필수 변경한다.

개발 및 유지보수 목적의 계정은 업무 종료 후 삭제한다.

패스워드 생성/재생성 시 안전한 패스워드 규칙을 적용한다.

특정 기간이 지나면 암호키와 인증 비밀번호의 만료를 권고하고 변경하도록 한다.

3.10 로그 관리

로그는 안전한 장치에 보관 되어야한다.

로깅 시 기기의 시작/종료(Shutdown), 로그인, 접근시도 이력은 필수로 남겨야 하며, 시스템에 필요한 정보는 추가적으로 선택하여 남기도록 한다.

로그파일은 비인가자로부터 접근이 불가능하여야 한다.

타 시스템으로 민감한 로그를 전송하여야 할 때에는 암호화되어 전송하도록 한다.

로그는 주기적으로 모니터링 및 분석되어야 한다.

로깅 시간이 현재 시간과 일치해야 한다.

패스워드, 개인정보는 로그에 남지 않아야 한다.

3.11 취약점 점검

중요한 인프라 변경이나 애플리케이션 업데이트 발생 시 또는 정기적으로 연 1회 이상 네트워크 및 애플리케이션 계층 침투테스트를 실시하여야 한다.

II. 결론

본 논문에서는 기업환경에서 초기 IoT 보안 도입을 위한 지침을 수립해 보았다.

이를 통해 IoT 보안 도입을 고려하는 기업환경에서 영역별 기 존재하는 보안과의 중복을 줄이고 효율적으로 IoT 보안을 도입할 수 있는 가이드라인이 될 수 있을 것이라고 생각한다.

더 나아가 각 기업의 세부적 환경에 맞게 해당 지침을 연구하고, 구체화 해 나가는 지속적인 노력이 필요하다.

[참고문헌]

[1] 한국인터넷진흥원, IoT 공통 보안 원칙, 2015

[2] 한국인터넷진흥원, IoT 공통보안 가이드, 2016

[3] 미래창조과학부, 사물인터넷(IoT)정보보호 로드맵, 2014

[4] SK인포섹, 사물인터넷(IoT) 보안 가이드, 2018

[5] 한국인터넷진흥원, 홈·가전 IoT 보안 가이드, 2017

In-body OTP : 이식형 의료 장치의 고유성 확보를 위한 체내형 One Time Password 생성 방법

안성규*,정혜림*,박기웅†

세종대학교 시스템 보안 연구실*, 세종대학교 정보보호학과†

In-Body OTP : One Time Password generation method for the ensure the uniqueness of the implantable medical device

Sung-Kyu Ahn*,Hyelim Jeong*, Ki-Woong Park†

* System Security Laboratory, Sejong University

† Department of Computer and Information Security, Sejong University

요 약

4차 산업혁명을 거치며 의학분야와 ICT 분야의 융합이 이루어지면서 다양한 형태의 의료방법이 탄생했다. 체내 이식형 의료 장치 분야는 ICT 분야와 융합되면서 임베디드 분야의 IoT 기반 기술로 발전하였다. 이러한 발전은 한편으로 해킹의 위험성을 초래할 수 있다. 이러한 문제를 해결하기 위해, 본 논문에서는 체내 이식형 의료 장치의 고유성 및 인증과정을 수행하기 위한 "In-Body OTP"기술을 제시한다. 이 기술을 통해 이식형 의료장치는 각각의 고유성을 가짐으로써 인증기술 등에 사용되어 이식형 의료 장치의 보안성 및 안전성을 확보할 수 있다.

I. 서론

4차 산업혁명 이후 의학분야와 ICT 분야의 융합이 급속도로 이루어지며, 의학과 IT가 접목된 제품이 제작되어 환자의 생명을 지키고 유지하는것에 큰 도움이 되고 있다. 일례로는 원격 로봇팔 수술, 환자 정보에 따라 의약품 자동 분배기, 인공지능을 사용한 암 진단 시스템, 체내에 이식·침습되어 사용하는 이식·침습형 의료장치(또는, 이식·침습형 의료장치, Implantable Mediacal Device)가 있다[1]. 이러한 기술은 임베디드 개념의 IoT기반의 기술로써 발전하면서 환자의 삶의 질을 높이고 고통을 줄여주는 긍정적인 역할을 수행하지만, 악의적인 목적으로 사용 될 가능성이 있다[2][3]. 특히, 이식형 의료장치는 환자의 신체 내부에 이식되어 사용됨으로 인해, 환자의 생명에 가장 가까운 의료기기로써 보안 측면 위험으로 인해 생명에 직접적인 영향을 끼칠 수 있는 부정적인 위험요소로써 사용될 수 있다. 하지만 이식형 의료 장치는 본래의 의료적 목표를 달성하기 위한 최소한의 컴퓨팅 파워 및 배터리 등으로 구성되어 있다. 이러한 자원 문제로 인해 기기에 직접 부가적인 보안 장치를 추가하기 위해서는 기존의 컴퓨팅 파워보다 높은 성능의 컴퓨팅 파워가 필요하며, 이는 곧 높은 전력 소모를 의미한다. 결국, 환자는 배터리 교체를 위해 몸속에 있는 기기를 꺼내야 하는 의료적 행위를 반복적으로 진행해야 하는 부정적인 결과를 초래한다[4].

이러한 문제를 해결하기 위해 본 논문에서는 이식형 의료기기에 보안 장치의 추가를 최소화하면서 보안성을 확보할 수 있는 '이식형 의료장치를 위한 체내 OTP 시스템'에 대한 개념인 'In body OTP"를 제시한다.

본 논문은 2장에서 본 논문에서 제시하는 '이식형 의료 장치를 위한 채네 OTP 시스템'의 개념을 제시한다. 마지막으로 3장에서 결론을 제시한다.

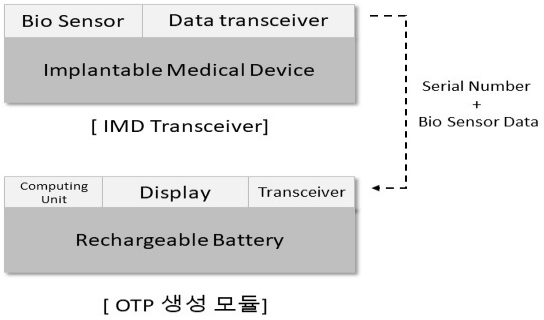
II. In body-OTP

본 장에서는 본 논문에서 제시하는 이식형 의료 장치를 위한 보안 기술과 관련된 연구와 체내 통신 기술 관련 연구에 대해 제시한다.

기본적으로 이식형 의료 장치를 인증하기 위한 시스템에는 In body-OTP가 환자에게 이식되는 과정에서 이식형 의료 장치의 고유번호와 환자의 일반적인 바이오 정보 (심장박동, 뇌파 등)을 획득하고 디지털화하여 저장된다.

1.1 In body-OTP 구조

In body-OTP의 구성은 크게 [그림1]과 같이 송수신기능이 부착된 IMD Transceiver와 송수신기가 장착된 OTP 생성 모듈로 구성되어 있다.

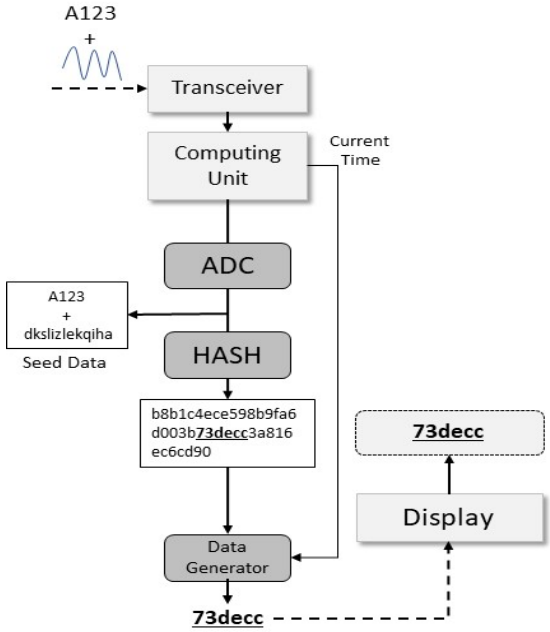


[그림 1] In body-OTP 구조도

IMD Transceiver는 이식형 의료 장치의 기본적인 의료 행위를 포함한 기기와 데이터 송수신 기능을 수행하는 Data transceiver 모듈이 추가로 구성되어 있다. OTP 생성 모듈은 배터리 및 MCU, 송수신 모듈, 컴퓨팅 모듈 및 디스플레이로 구성되어 있다.

1.2 In body-OTP 구현

IMD Transceiver는 기본적으로 이식형 의료 장치와 같은 기능을 수행한다. 만약 이식형 의료 장치의 인증이 필요한 시점이 되었을 때, [그림 2]와 같이 이용자는 OTP 생성 모듈의 수신 기능을 사용하여 체내 통신[5][6]을 통해 IMD Transceiver로부터 기기의 고유번호와 현재 측정되고 있는 신체 데이터를 수신한다. OTP 생성기는 수신한 기기의 고유번호와 디지털화된 바이오 데이터를 결합하여 OTP를 생성하기 위한 Seed 데이터값을 생성한 뒤, 해시 연산을 수행하여 최종값을 도출하고, 현재 시각을 기준값으로 하여 해시 데이터의 특정 부분을 추출한 뒤 디스플레이 모듈로 전달한다[7]. 디스플레이 모듈은 해당 값을 디스플레이에 출력하여 사용자가 확인할 수 있도록 한다. 디스플레이 출력의 최소 시간은 10초이며, 10초가 지난 뒤에는 다른 바이오 데이터를 이용하여 OTP를 생성한다.



[그림 2] OTP 데이터 생성과정

사용자의 OTP를 입력받은 시스템은 기존에 보유하고 있던 기기 고유번호와 바이오 데이터에 해시 연산을 반복해 가면서 해당 시간 단위에 해당하는 값을 도출하게 된다.

III. 결론

의학 분야와 ICT 분야의 접목으로 인해 많은 IoT 기반의 의료기기가 생성되고 있지만, 이러한 의료기기는 질병 치료 및 환자의 생명 유지 등과 같은 의료 목적으로 사용됨으로써, 기존의 ICT 기술을 동일하게 접목하기 어려운 실정이다[8]. 특히 이식형 의료 장치 분야의 경우, 환자의 질병을 모니터링 하고 특정 상황에 맞는

동작을 위해 최적화된 기기를 사용함으로 인해, 의료기기를 대상으로 하는 보안 기술을 적용하기 어렵다. 본 논문은 이런 문제점을 해결하기 위해 In body-OTP를 제시하였으며, 이 기술은 별도의 무선네트워크가 아닌 신체 네트워크를 통해 데이터를 송수신하고, 시간과 환자의 생체정보를 기준으로 인증데이터를 생성함으로써, 인증데이터가 무선환경에서 유출되거나 가로채기 당하는 문제점을 해결할 수 있다. 추후 연구로써는 본 논문에서 제시한 In body-OTP를 활용하여 이식형 의료 장치 인증 플랫폼 모듈을 구성하는 연구를 진행할 예정이다.

[참고문헌]

[1] 식품의약품안전평가원 “2018년 신개발 의료기기 전망 분석 보고서”, 2018.03.

[2] Tabasum, Aliya, et al. "Cybersecurity Issues in Implanted Medical Devices." 2018 International Conference on Computer and Applications (ICCA). IEEE, 2018.

[3] Papp, Dorottya, Zhendong Ma, and Levente Buttyan. "Embedded systems security: Threats, vulnerabilities, and attack taxonomy." 2015 13th Annual Conference on Privacy, Security and Trust (PST). IEEE, 2015.

[4] Yaqoob, Tahreem, Haider Abbas, and Mohammed Atiquzzaman. "Security Vulnerabilities, Attacks, Countermeasures, and Regulations of Networked Medical Devices - A Review." IEEE Communications Surveys & Tutorials (2019).

[5] Li, Maoyuan, et al. "The modeling and simulation of the galvanic coupling intra-body communication via handshake channel." Sensors 17.4 (2017): 863.

[6] Sana Ullah, Henry Higgins, 곽경섭. (2008). 내장형 및 부착형 인체센서네트워크의 연구 동향 및 이슈. 한국통신학회지(정보와통신), 25(2), 18-25.

[7] M'Raihi, David, et al. "Totp: Time-based one-time password algorithm." Internet Request for Comments (2011).

[8] Belkhouja, Taha, et al. "Light-Weight Solution to Defend Implantable Medical Devices against Man-In-The-Middle Attack." 2018 IEEE Global Communications Conference (GLOBECOM). IEEE, 2018.

IoT 환경을 위한 무인증서 기반의 게이트웨이 중재 서명 방법에 관한 연구+

이대휘*, 이임영**

순천향대학교 컴퓨터학과

A Study on Certificate-Free Gateway Arbitration Signature Method for IoT Environment

Dae-Hwi Lee*, Im-Yeong Lee**

Dept of Computer Science and Engineering, Soonchunghyang University

요 약

IoT(Internet of Things) 환경을 위한 많은 서비스들이 출시되고 있다. 다양한 IoT 서비스들의 신뢰성 확보를 위해서 다양한 보안 요구사항들이 있으며, 특히 IoT 환경에서 무선으로 전송되는 메시지들에 대한 무결성의 확보는 필수적이다. 하지만 IoT 환경과 같은 초경량, 저전력 기술을 필요로 하는 환경은 기존의 공개키 기반 서명 기술을 그대로 적용하기 힘든 문제가 있다. 이를 경량화 하기 위해 지속적으로 연구되고 있다. ID기반 암호는 공개키 검증 과정을 줄이는 암호 기술이지만, KGC(Key Generation Center)에서 참여자들의 공개키와 개인키 쌍을 모두 생성해주어 이후에 KGC는 모든 참여자들의 키를 다시 생성할 수 있다는 키 에스크로 문제가 발생한다. 이를 해결하기 위해 무인증서 기반의 공개키 암호 시스템 CL-PKC(Certificateless Public Key Cryptography)가 제안되었다. 본 논문에서는 IoT 환경을 위해 CL-PKC를 이용한 서명 방법인 CLS(Certificateless Signature)에 대해 분석하며, 기존 CLS에서 부인 방지 기능을 향상시키기 위해 IoT 환경에서 게이트웨이를 통해 중재 서명을 수행하는 방법에 대해 제안한다.

I. 서론

IoT 서비스 환경에 참여하고 있는 객체들은 각자 인터넷에 연결되어 다른 객체와 통신할 수 있다. IoT 환경에서 전송되는 메시지에 대해 무결성을 제공하기 위해서는 서비스의 신뢰성을 확보하기 위해 필수적이다. 메시지의 무결성을 제공하기 위한 기술로는 디지털 서명이 있으며, 서명도 마찬가지로 경량화 되어야만 IoT 환경에 적용할 수 있을 것이다.

따라서 본 논문에서는 IoT 환경에서 적합한 서명 방법인 무인증서 기반의 암호 시스템인 CL-PKC(Certificateless Public Key Cryptography)를 사용한 서명 방법인 CLS(Certificateless Signature)에 대해 분석하고, CLS에서 서명의 부인방지의 기능을 강화하기 위한 무인증서 기반 게이트웨이 중재 서명을 수행하는 방법에 대해 연구한다.

+ 본 연구는 2019년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(No. NRF-2019R1A2C1085718)과 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터 육성 지원사업(IITP-2019-2015-0-00403)의 연구결과로 수행 되었음

II. 관련 연구

ID 기반 암호 시스템은 기존 PKI의 검증 문제를 해결하기 위해 등장하였다. ID 기반 암호 시스템은 이미 알려진 사용자의 식별자에 기반한 공개키를 사용하기 때문에, 공개키의 검증 과정을 없애 기존 PKI에서 발생하는 키 분배나 오버헤드 문제를 해결할 수 있다. 하지만 ID 기반 암호에서는 KGC(Key Generation Center)가 사용자의 비밀키까지 모두 생성하는 키 에스크로에 대한 문제가 발생한다.

키 에스크로 문제를 해결하기 위해 여러 암호

기술들이 제안되었으며, CL-PKC도 그 중 하나이다. CL-PKC는 KGC에서 공개키와 개인키를 전부 생성하지 않고 일부분만 생성하여 사용자에게 돌려준다. 이 일부분의 키를 부분 비밀키라고 말하며, 사용자는 부분 비밀키를 이용해 자신의 완전한 키 쌍을 생성하게 된다. 완전한 키 쌍은 사용자가 직접 생성하기 때문에 키 에스크로 문제를 해결할 수 있고, PKI보다 오버헤드가 적어 IoT 환경에 적합하다.

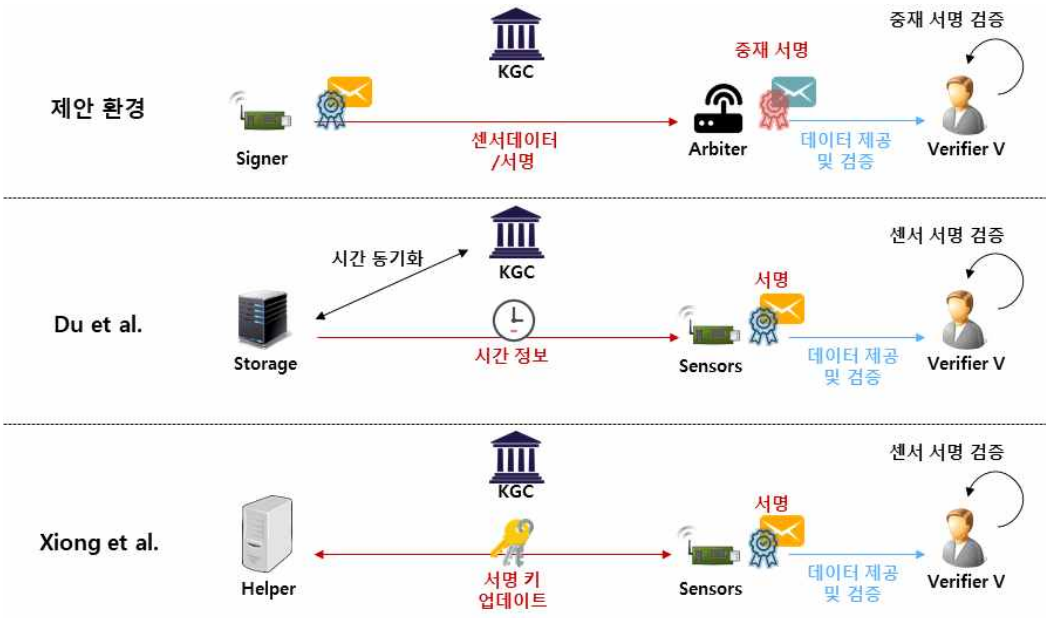
하지만, CL-PKC에서 사용되는 공개키는 인증서를 사용하지 않아 사용자의 식별자에 대한 인증을 할 수 없다. ID에 대한 인증이 수행되지 않기 때문에 부인 방지도 제공하지 않아 공개키에 대한 대체 공격이 발생할 수 있는 문제가 있다. CLS에서 공개키 대체 공격은 서명자의 개인키를 이용해 생성된 서명의 검증을 우회할 수 있는 공격자의 대체된 공개키가 서명자의 공개키인지 공격자의 공개키인지를 인증할 수 없기 때문에 발생한다[2]. 이를 방지하기 위한 연구는 현재 활발하게 진행되고 있다[3-4].

III. 제안 방식 설계

본 논문에서는 IoT 환경을 위한 무인증서 서명인 CLS 기법에 있어서 공개키 대체 공격에 안전하도록 인증과 부인 방지 기능을 제공하는 방식에 대해 설계한다. 특히, 기존 Du et al.[3]과 Xiong et al.[4]에서는 시간 동기화나 helper를 이용한 키 동기화를 이용해 공개키에 대한 안전성을 확보하였다. 이렇게 외부 객체와 통신할 필요 없이, 본 제안 방식에서는 IoT 통신 환경에서 사용되는 게이트웨이를 통한 중재 서명을 수행하여 이용해 공개키에 대한 안전성을 제공하고자 한다.

IoT 환경에서 메시지를 생성하고 서명하는 주체는 센서 디바이스이다. 게이트웨이는 이를 수집하여 중재 서명을 수행한다. 중재 서명의 검증은 메시지를 획득하는 서비스 이용자, 즉 소비자가 수행하며 서명자인 디바이스와 중재 서명자인 게이트웨이의 검증을 동시에 수행하게 된다. (그림 1)는 기존 Du et al., Xiong et al.과 제안하는 방식의 비교 그림이다.

- 공개키 대체 공격 방지 : 공개키에 대한 인증과 식별자를 가진 서명자에 대한 부인 방지 기능이 필요하다.
- 키 유출 방지 : KGC는 부분 비밀키를 생성할 때 사용자들의 키를 계산할 수 없어야 하며, 통신상에 전송되는 메시지들을 통해서도 키가 유출되어서는 안 된다.
- 서명 위조 방지 : 정당한 메시지의 서명에 대한 위조가 불가능해야 한다. 이는 키 유출과 공개키 대체 공격을 방지하고 서명과 검증 과정에 대한 안전성 분석이 필요하다.



(그림 1) 기존 방식과 제안 방식의 환경 비교

위의 보안 요구사항을 통해 제안하는 방식은 등록 단계, 서명 및 중재 서명 단계, 검증 단계의 세 단계로 구성된다.

2.1 등록 단계

Step 1. 디바이스는 KGC를 통해 부분 비밀키를 발급받기 위해 자신의 검증용 공개키 및 개인키 쌍을 생성한다.

Step 2. 디바이스는 KGC에게 자신의 식별자와 검증용 공개키를 전송하고, KGC는 이를 이용해 디바이스의 부분 비밀키를 생성하여 안전하게 디바이스에게 돌려준다.

Step 3. 디바이스는 부분 비밀키와 검증용 공개키 쌍을 이용해 완전한 키 쌍을 생성한다.

2.2 디바이스 서명 및 중재 서명 단계

Step 1. 디바이스는 자신이 생성한 메시지에 서명하기 위해 세션 정보와 자신의 완전한 개인키를 이용해 서명을 생성한다.

Step 2. 디바이스는 서명과 메시지를 중재자인 게이트웨이에게 전송한다.

Step 3. 게이트웨이는 디바이스의 공개 정보를 이용해 KGC로부터 디바이스에게 생성하여준 키로 서명한 서명이 맞는지 검증을 수행한다.

Step 4. 게이트웨이는 기존의 메시지에 대해 디바이스가 검증되었다는 세션 정보, 그리고 자신의 서명을 추가하여 중재 서명을 수행한다.

2.3 검증 단계

Step 1. 게이트웨이는 검증자에게 디바이스의 완전한 공개키와 자신의 완전한 공개키를 검증자에게 전송한다.

Step 2. 검증자는 디바이스, 게이트웨이의 공개된 정보들을 이용해 서명을 검증하여 유효성을 확인한다. 이 때, 디바이스와 게이트웨이의 유효성을 동시에 검증할 수 있다.

IV. 결론

본 논문에서 제안하는 프로토콜은 IoT 환경의 참여 객체인 게이트웨이에 중재 서명의 개념을 이용해 CLS를 적용하여 두 객체 사이에 전송되는 메시지에 대한 인증을 제공한다. 제안 방식은 여러 보안 요구사항들에 대해 만족할 수 있도록 설계되었으며, 기존 방식들에서는 특히 공개키 대체 공격과 키 유출, 그리고 이를 통한 서명 위조에 대해 문제가 있었기 때문에, 이를 해결하기 위해 게이트웨이의 중재 서명을 적용하고 속도를 높여 효율성까지 제공할 수 있다. 향후에는 1:1로 통신하는 환경이 아닌 1:N 환경에서 사용될 수 있는 중재-집계 서명을 이용하여 더욱 효율성을 높이기 위한 서명 기법을 연구하려 한다.

[참고문헌]

[1] K. E. Skouby, and P. Lynggaard, “Smart home and smart city solutions enabled by 5G, IoT, AAI and CoT services”, In Proceedings of 2014 International Conference on Contemporary Computing and Informatics, pp. 874-878. 2014.

[2] X.Jia, D.He, Q.Liu and K.K.R.Choo, “An efficient provably-secure certificateless signature scheme for Internet-of-Things deployment”, Ad Hoc Network, Vol 71, pp. 78-87, 2018.

[3] H. Du, Q. Wen, and S. Zhang, “A provably-secure outsourced revocable certificateless signature scheme without bilinear pairings”, IEEE Access, Vol. 6, pp. 73846-73855, 2018.

[4] H. Xiong, Q. Mei and Y. Zhao, “Efficient and Provably Secure Certificateless Parallel Key-Insulated Signature Without Pairing for IIoT Environments”, IEEE SYSTEMS JOURNAL, 2019.

종단 간 보안을 지원하는 MQTT-SN 보안 아키텍처 설계 및 구현

남혜민* 박창섭+

*단국대학교 대학원 컴퓨터학과

+단국대학교 소프트웨어학과

Design and Implementation of MQTT-SN security architecture supporting the end to end security

Hye-Min Nam* Chang-Seop Park+

*Department of Computer Science, Dankook University.

+Department of Software, Dankook University.

요 약

MQTT-SN(Message Queuing Telemetry Transport - Sensor Network)프로토콜은 센서 기반의 IoT(Internet of Things)환경에서 사용되는 메시지 전송 프로토콜이다. 이 MQTT-SN 프로토콜은 메시지 전송 중간에 중개자(Broker)를 둔 발행-구독 모델(Publish-Subscribe Model)이다. 따라서 각 IoT 장치들은 메시지 전달 시 반드시 중개자를 통해 메시지를 주고받게 된다. 따라서 기존의 암호화 채널을 만드는 DTLS(Datagram Transport Layer Security) 프로토콜 방식은 종단 간 보안을 지원 할 수 없다. 본 논문에서는 MQTT-SN 프로토콜에서 종단간 보안을 지원하기 위해 발행자와 구독자에게 주제(Topic) 별 인증서를 기반으로 ECQV 인증서를 발급하는 인증 서버를 추가하고, 발급된 인증서를 기반으로 상호 대칭키를 도출할 수 있도록 기존 네트워크 전송 패킷을 수정한 새로운 MQTT-SN 프로토콜을 제안한다.

Keywords: MQTT-SN, ECQV, ECDH, End-to-End security, IoT

I. 서론

사물인터넷 환경에서는 무선 통신을 통해 각종 기기들이 수집하는 방대한 양의 데이터가 사물 간에 전달된다. 이 무선 통신에 사용되는 전용의 메시지 전송 프로토콜이 존재하며 대표적으로 CoAP과 MQTT-SN이 있다[1]. 각 프로토콜은 서로 다른 네트워크 모델을 사용하고 있다. CoAP의 경우 클라이언트-서버 모델로 데이터 및 명령 전달을 위해 클라이언트와 서버 간에 직접 채널을 제공한다. 반면에 MQTT-SN의 경우 발행-구독 모델로 발행자(데이터 생성)와 구독자(데이터 소비) 사이에 중개자를 통해 간접 채널을 제공한다.

두 프로토콜 모두 각 표준에서 메시지 보안을 위해 DTLS를 제안하고 있다. 직접 채널을 제공하는 CoAP의 경우, DTLS를 적용하면 종단 간 보안이 가능하다. 하지만, 중개자를 사이에 둔 MQTT-SN의 경우에는 DTLS로는 종단 간의 보안이 보장되지 않는다.

따라서 이 논문에서는 MQTT-SN에서 종단 간 보안을 지원하기 위해서 인증서를 발급하는 서버를 추가한다. 이후 발급된 인증서를 기반으로 발행자와 구독자간의 공유하는 키를 만들어 사용하는 아키텍처를 제안하고자 한다.

본 논문은 다음과 같이 구성된다. 2장에서는 종단 간의 보안을 지원하기 위해서 제안하고 있는 새로운 아키텍처에 대해 설명한다. 3장에서는 구현을 소개하고 4장에서 결론을 맺는다.

II. 위협 분석 & 제안 프로토콜

기존 MQTT-SN 프로토콜의 보안적인 특징과 프로토콜에서 보안방법으로 제시된 DTLS 핸드셰이크를 통한 보안 채널생성 시에 가능한 위협에 대해 설명하고 이를 해결하기 위한 새로운 프로토콜의 메커니즘에 대해 소개하도록 하겠다.

2.1 기본 MQTT-SN 보안 위협

그림 1.은 MQTT-SN의 패킷 구조이다. 이를 살펴보면 데이터 전송에 관한 필드만 존재하는 것을 확인 할 수 있다. 이는 무결성을 보장하지 못해 재생 공격 등의 대상이 될 수 있으며 기밀성 역시 적용되어 있지 않아 패킷 감청을 통한 통신상에서 데이터 유출이 될 수 있다.[3]

2.2 DTLS-MQTT-SN 보안 위협

따라서 MQTT-SN 프로토콜에서는 보안 방법으로 DTLS 핸드셰이크를 통한 보안 채널 생성을 제시하고 있다. 그림 2.와 같이 인접한 노드 사이에 핸드셰이크 과정을 통해 보안 채널을 생성하고 그 보안 채널 내에서 데이터를 전송하게 된다. 이를 통해 무결성, 기밀성을 보장하게 된다. 하지만, 이 방식은 인접한 노드 간에서 이루어지는 보안 채널이다. 따라서 종단 간 보안이 이루어지지 못하고 중개자 단계에서 전송되는 내용이 노출될 수 있다. 이는 악의적인 중개자 혹은 외부 공격에 의해 장악된 중개자에 의해 메시지 노출, 변조의 공격이 모두 가능하다는 보안 위협이 존재한다.[4]

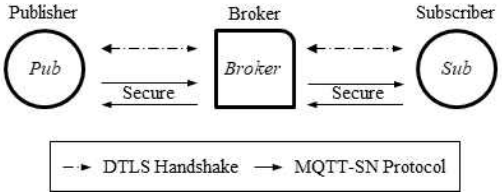
Message Header	Message Variable Part
(2 or 4 octets)	(n octets)

Table 1. General Message Format

Message Variable Part		
CliedtId	Data	Duration
Flags	GwAdd	GwId
MsgId	ProtocolId	Radius
ReturnCode	TopicId	TopicName
WillMsg	WillTopic	

Table 2. Message Variable Part

[그림 1] MQTT-SN 패킷 구조

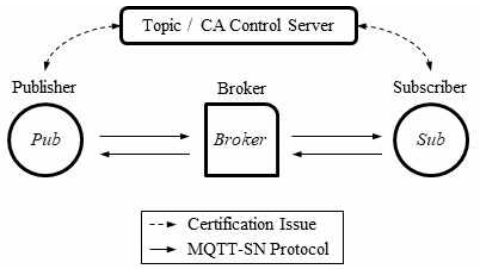


[그림 2] DTLS - MQTT-SN

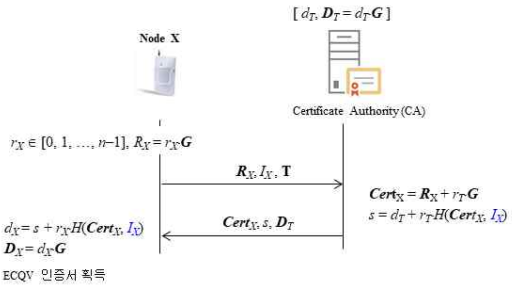
2.3 제안 프로토콜

새롭게 제안하는 프로토콜은 DTLS에서 지원하지 못했던 종단 간 보안을 적용하는 것이 가장 큰 목적이다. 이를 위해 기존 MQTT-SN 프로토콜에 존재하는 3 분류의 참가자(발행자, 구독자, 중개자) 외에 인증 서버를 추가하였다.

인증 서버는 노드(발행자와 구독자)에게 개인 인증서를 발급해주고, 각 노드들이 요청하는 토픽의 인증서를 전달한다. 각 토픽은 인증서 형태로 관리되고 토픽의 인증서를 가진 노드만이 일련의 상호키 교환 알고리즘을 통해 공유키를 생성 할 수 있다. 이후 공유키로 데이터를 암호화하여 전송함으로써 종단 간 보안이 적용된다.



[그림 3] 제안 프로토콜



[그림 4] 개인 인증서 발급

인증 서버를 추가한 이유는 중개자가 보안을 집중한 서버보다 물리적으로 취약하기 때문이다. 본래 중개자에서 관리해야하는 내용을 별도의 전문 서버를 둠으로써 보안을 향상시키고 중개자에게는 부담을 줄일 수 있게 된다. 따라서 결과적으로 그림3.과 같이 인증 서버로부터 인증서를 발급받는 개체는 발행자와 구독자뿐이다. 중개자는 인증서를 발급받지 않으며 단순 전달자로서의 역할을 수행하게 된다. 그림4.를 보면, 발행자와 구독자는 개인 인증서의 발급에 필요한 인자들을 생성·전달하게 되고 이와 동시

에 특정 토픽을 구독, 발행 할 것인지 선택하여 인자들과 함께 전달한다. 인증 서버는 전달받은 내용을 사용하여 개인 인증서를 생성, 선택한 토픽 인증서와 함께 개인 인증서를 전달한다.

인증 서버는 개인의 인증서를 발급하는 것 뿐 만아니라 토픽을 인증서 별로 관리하며 접근 제어 목록으로 사용할 수 있다. 그림 5.의 표 4처럼 인증 서버는 네트워크에 참가하고 있는 기기들의 ID정보를 미리 가지고 있다. 따라서 최초 요청 시 네트워크에 참가하고 있는 기기인지 확인하게 된다. 이후 요청하는 토픽이 존재하는지 검증하고 중복으로 토픽을 요청하는지 확인하는 단계가 진행된다. 이러한 일련의 과정을 마치면 각 노드의 인증서 발급 단계가 종료된다.

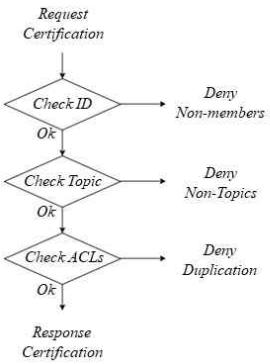


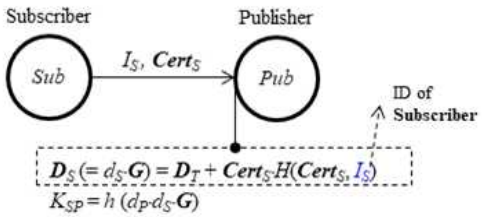
Table 3. Topic – Cert Management

Topic	Certification	ID
Topic_1	T-Cert_1	ID_1, ID_3
Topic_2	T-Cert_2	ID_2
Topic_3	T-Cert_3	NULL
⋮	⋮	⋮

Table 4. Device Topic Management

ID	Type	Topic
ID_1	Pub	Topic_1
ID_2	Sub	Topic_2
ID_3	Sub	Topic_1
⋮	⋮	⋮

[그림 5] 인증 서버 메커니즘



[그림 6] ECQV 인증서 기반 ECDH 알고리즘

인증서 발급이 완료되면 구독자와 발행자가 각 개인의 인증서의 교환을 통해 상호 공유 할 수 있는 키를 만드는 작업이 수행된다.

그림6.은 인증서 교환을 통한 상호 공유키 생성과정을 발행자 입장에서 표현한 것이다. 우선 전달받은 구독자의 인증서($\boldsymbol{Cert_S}$)와 ID 값(I_S)을 사용하여 구독자의 공개키($\boldsymbol{D_S}$)를 도출하는 과정이 진행된다. 공개키 도출이 완료되면 ECDH 알고리즘을 통해 상호 공유키($dpds\boldsymbol{G}$)가 생성된다. 만들어진 상호 공유키($dpds\boldsymbol{G}$)는 마스터키로 취급되며 실제 메시지의 기밀성과 무결성을 위해 사용되는 키는 암호화해시 함수가 적용된 세션 키(K_{SB})이다. 구독자 역시 동일한 과정으로 세션 키를 생성한다. 생성된 세션 키가 동일할 경우, 암호화한 데이터를 복호화가 가능할 것이고, 동일한 해시 기반 메시지인증 코드(HMAC)를 만들 수 있을 것이다. 세션 키는 발급 단계가 정상적으로 진행된 후 얻을 수 있는 인자들로 만들어진다. 따라서 세션 키가 동일하다는 것은 인증 서버를 통해서 검증된 네트워크 기기이면서 동일한 토픽를 선택하여 구독, 발행 중이라는 의미로 해석 할 수 있다.

Length	MsgType	Flags	TopicId	MsgId	Certification	ClientId	Data	HMAC
(octet 0)	(1)	(2)	(3-4)	(5-6)	7:26	27:34	35:49	50:69

Table 5. Enhanced Publish Message Format

[그림 7] 수정 된 Publish Packet Format

이러한 세션 키를 만들기 위해서는 인자 전달이 필요하다. MQTT-SN 프로토콜에는 중간에 중개자가 존재하여 그림6.과 같은 방식으로 직접 각 노드 종단 간 전달이 불가능하다. 따라서 중개자를 통해 그대로 전달되는 메시지 포맷을 이용하여 전달하도록 한다. 구독자의 경우 Subscribe 메시지 포맷을 발행자의 경우 Publsih 메시지 포맷에 전달하고자 하는 인자 필드를 추가한다. 추가되는 필드로는 인증서, 노드ID, HMAC이 있으며 각 길이는 20, 8, 20 bytes 이다.

III. 구현

본 논문에서 구현은 인증 서버로부터 각 노드(구독자와 발행자)가 동일한 토픽을 대상으로 인증서가 이미 발급이 완료되어 가지고 있다는 가정으로 진행하였다.

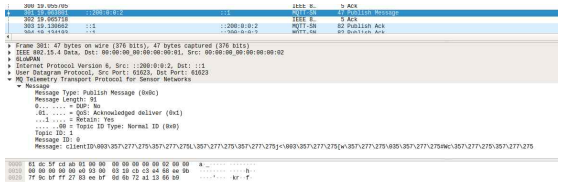
본 구현에서 집중해서 다룬 부분은 제한된 성능을 내는 IoT 기기가 ECQV 인증서로부터 공개키를 추출하고 소개한 키 교환 알고리즘을 통해 공유키를 만들어 기밀성을 위해 추가한 AES-CTR 방식의 암호화에 사용되고, 무결성

을 위해 추가한 HMAC - SHA1의 연산에 실제로 이용이 가능한 것인지에 대한 테스트이다. 따라서 구현에서는 IoT환경에서 인증서 발급을 위해 필요한 인자 값(EC parameter)을 만드는데 걸리는 시간과 인증서로부터 공개키를 추출하는데 걸리는 시간(ECQV), 인자 값으로부터 상호 공유키를 만드는데 걸리는 시간(ECDH), 암호화 연산에 걸리는 시간을 측정하였다(AES-CTR, HMAC-SHA1).

(단위 : ms)

Algorithm	Time
AES-CTR	1.861
HMAC-SHA1	5.645
Make EC parameter (secp160r1)	1096.557
ECQV (secp160r1)	1148.986
ECDH* (secp160r1)	216.766

또한 상호 공유키를 만들기 위해 Publish 메시지에 필드를 추가하였고, 해당 필드를 통해 공유키 생성에 필요한 인증서와 ID 정보를 전달 할 수 있도록 기존 MQTT-SN 코드를 수정하였다.[5,6] 다음 그림8.은 실제로 그림7.의 Publish 패킷 포맷을 수정하여 통신을 진행하는 것을 Wireshark를 통해 캡처한 화면이다.



[그림 8] MQTT-SN Publish Message packet capture

IV. 결론

본 논문에서는 각 노드에 인증서를 발급하고 토픽을 인증서 별로 관리하는 인증 서버를 새로 추가하여 MQTT-SN 프로토콜에서 구독자와 발행자 간의 종단 간 보안을 적용했다. 이로써 중개자가 악의를 가지거나 이미 악의적인 공격자에 의해 점령되어 있는 중개자가 존재하더라도 종단의 기기들은 인증서버로부터 받은 인증서를 기반으로 만들어 진 공유키를 사용하여 전송되는 메시지에 대한 기밀성과 무결성을 만족 시킬 수 있었다.

다만 인증 서버에서 이미 정해진 네트워크 참여자를 가정하고 있었다는 점에서 네트워크에 참여자를 추가하는 과정과 토픽별 인증서를 갱신하여 다시 공유키를 만들어야 하는 시점에서의 키 재생성 메커니즘과 같은 키 관리 부분이이 추가되어야 네트워크의 가입과 탈퇴가 잦은 IoT 환경에 적합한 보안 프로토콜이 될 것이라고 생각한다. 해당 내용들에 대한 보완을 추후 연구에서 진행 할 것이다.

[참고문헌]

[1] Kraijak, Surapon, Panwit Tuwanut. "A survey on internet of things architecture, protocols, possible applications, security, privacy, real-world implementation and future trends." 2015 IEEE 16th International Conference on Communication Technology. IEEE, 2015.

[2] 성정기, 김은기. (2017). ECQV 묵시적 인증서의 설계 및 구현에 관한 연구. 한국정보통신학회논문지, 21(4), 744-752.

[3] A. Banks and R. Gupta, “MQTT Version 3.1.1,” OASIS Standard, Apr. 2014

[4] Wang, Xinlei, et al. "Performance evaluation of attribute-based encryption: Toward data privacy in the IoT." 2014 IEEE International Conference on Communications. IEEE, 2014.

[5] A. Ignacio da Silva, “MQTT-SN and Contiki-OS,” https://blog.aignacio.com/2017/06/25/6lowpan-mqtt-and-contiki-os/

[6] eclipse, “ RSMB: Really Small Message Broker ” https://github.com/eclipse/mosquitto.rsmb

스마트폰 환경의 보안부팅 기술 비교 분석

이정국*, 김진우*, 이상길*, 이상한**, 이철훈*

*충남대학교 컴퓨터공학과

**ETRI부설연구소

A Comparative Analysis of Secure Boot Technology in Smartphone Environment

Jeong-Guk Lee*, Jin-Woo Kim*, Sang-Gil Lee*,
Sang-Han Lee**, Cheol-Hoon Lee*

*Dept. Computer Science & Engineering, Chung-Nam University

**The Attached Institute of ETRI

요 약

최근 스마트폰 보급률이 높아지고 스마트폰을 기반으로 하는 모바일 뱅킹의 사용자 수와 규모가 증가하고 있다. 특히 모바일 뱅킹과 같은 서비스는 보안 취약성으로 인해 보안 위협이 발생할 수 있고 그 피해가 심각하므로 신뢰성 있는 스마트폰에서 수행되어야 할 것이다. 신뢰성 있는 스마트폰을 위해 다양한 보안기능 중 부팅 시작 단계부터 시스템에 대한 무결성을 검증하는 보안부팅부터 시작하게 된다. 본 논문에서는 스마트폰 환경에서의 보안부팅에 대해서 조사하고 이를 바탕으로 보안부팅을 비교하여 각 스마트폰 환경에 따라 유사한 점과 상이한 점을 확인하였다. 향후 보안부팅 이후에도 스마트폰을 신뢰할 수 있게 하는 보안기술에 대한 조사 및 비교 분석이 필요할 것이다.

I. 서론

국내의 스마트폰 이용자 수는 점점 늘어나 2019년 2월에 국민의 95%가 스마트폰을 사용하고 있다[1]. 이렇게 스마트폰의 이용자 수가 늘어나면서 스마트폰을 기반으로 하는 금융서비스인 모바일 뱅킹의 사용자 수도 함께 늘어나고 있다. 2019년 6월 말 모바일 뱅킹 등록고객 수는 1억 1천만 명이 넘고 모바일 뱅킹 이용실적 건수는 일평균 9천만 건 이상이며, 이용금액은 하루에 평균 6조가 넘을 정도로 모바일뱅킹의 규모는 점차 성장하고 있다[2].

모바일 뱅킹과 같은 서비스는 보안 취약성으로 인해 금융정보 유출과 같은 보안 위협이 발생할 수 있고 그 피해가 심각해 보안에 매우 민감하다. 이러한 모바일 뱅킹 서비스를 스마트폰을 통해 수행하기 위해서는 사용하는 스마트폰을 신뢰할 수 있어야 할 것이다. 신뢰성 있는 스마트폰을 위해 각 스마트폰마다 다양한 보안기능을 제공하고 있으며 이러한 보안기능은 부팅 시작 단계부터 시스템에 대한 무결성을 검증하는 보안부팅부터 시작하게 된다.

본 논문에서는 보안부팅에 대해 서술하고 스마트폰에서 가장 많이 사용하는 모바일 운영체제인 구글의 Android, 애플의 iOS의 보안부팅의 기능과 특징을 설명한다.

그리고 삼성전자에서 Android에 추가적인 보안 성능을 제공하는 Knox의 보안부팅에 대한 기능과 특징을 설명하고 각 보안부팅을 비교 분석한다.

II. 보안부팅

부팅 단계에서 감염되는 악성코드는 부팅 후 탐지가 어렵다. 보안부팅은 부팅 단계에서의 악성코드 감염으로부터 시스템을 보호하기 위한 보안 설계 중 하나로서 시스템의 H/W에 무결성 정보를 저장하기 위한 RoT(Root of Trust, 신뢰의 뿌리)를 제공한다[3]. 그리고 [그림 1]과

같이 RoT를 기반으로 하여 시스템에서 로드하는 부트로더에 대한 검증을 수행하고 실행하도록 하며 이러한 과정을 반복하여 이전의 부팅 컴포넌트를 다음의 부팅 컴포넌트에서 검증하도록 한다. 이러한 방식을 기본으로 하여 각 제조사마다 고유한 방법을 통해 보안부팅을 적용하였다.

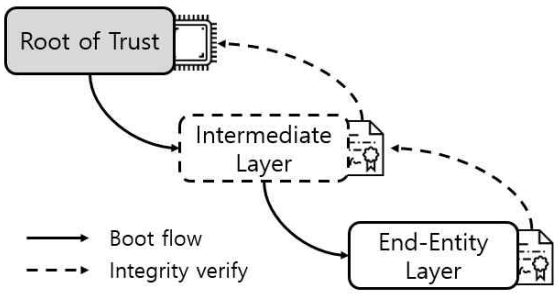


그림 1. 보안부팅의 동작 과정

Android, Knox, iOS는 모두 ARM 보안프로세서를 사용하며, 외부와 격리된 프로세서 내부의 보안 영역인 Trustzone[4]에 무결성 정보를 저장한다. 저장된 무결성 정보는 RoT로 사용하며 매 부팅 시 이를 기반으로 하여 보안의 무결성을 검증하는 보안부팅을 수행한다.

III. Google Android

Android의 전세계적 스마트폰 시장 점유율은 2019년 10월 기준 76%이상으로 막강하다[5]. 이러한 점유율에는 Android가 오픈소스를 기반으로 설계가 되어 스마트폰 제조사가 원하는 대로 Android를 만들 수 있다는 것이 한몫 했다. 하지만 다양한 제조사에서 다양한 Android 기기가 출시되며 각 기기별로 보안상의 차이점이 발생하게 되었다. 이에 구글은 공통 보안 환경을 위해 다양한 보안 구조를 설계하여 4.4버전부터 보안 환경을 위한 보안부팅을 지원하기 위해 DM-Verity(Device-Mapper-verity)와 검증된 부트(Verified Boot)를 추가하였다.

먼저 DM-verity[6]는 Android에서 사용되는 저장소와 드라이버 블록의 검증을 위한 기술로서 각 블록의 해시를 [그림 2]처럼 계층화된 트리구조의 해싱테이블로 관리하여 최종적으로 루트 해시만 비교하여 검증하면 된다.

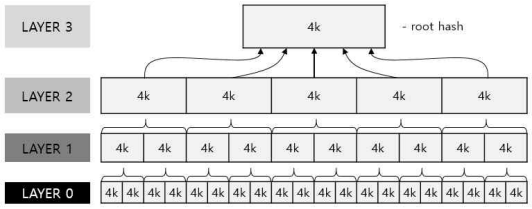


그림 2. DM-verity hash table

검증된 부트[7]는 하드웨어로 보호되는 BootROM를 RoT로 하여 보안부팅을 진행한다. 각 부트 단계에서 SHA-256, RSA-2048이상의 알고리즘을 사용하도록 한다.

부트 중 오류가 발생하면 그 상태에 따라 경고화면을 표시하게 되고 사용자의 선택으로 다음 단계의 부팅으로 진행할 수 있다.

IV. Samsung Knox

삼성전자에서는 자사가 개발하는 Android의 보안을 강화하기 위해 보안 솔루션인 Knox를 탑재하여 출시하고 있다. Knox는 기기에 자체 개발한 다양한 장치를 통해 일반영역과 보안영역으로 구분하게 되고 보안영역에서 보안이 필요한 기능 실행과 보안 자료의 보관을 수행하게 된다[8].

삼성 Knox의 보안부팅 과정은 SSBK (Samsung Secure Boot Key)를 기반으로 동작하는데 이는 비대칭 키 쌍으로 비밀키는 Knox를 탑재한 기기를 제조하는 과정에서 부트스트랩의 모든 과정에서 서명을 수행하며 공개키는 기기의 제조 과정에서 OTP(One-Time Programmable) 퓨즈라는 수정이 불가능한 영역에 저장한다.

Knox는 SSBK의 공개키를 RoT로 하여 보안부팅을 진행한다. 이러한 보안부팅이 이루어지는 과정에서 오류가 발견된 경우에 Knox Warranty Fuse(Knox 보증 퓨즈)를 설정하여 Knox와 관련된 기능과 Knox와 연결된 중요 정보를 사용할 수 없도록 한다.

V. Apple iOS

애플의 모바일 운영체제인 iOS는 스마트폰뿐만 아니라 iPod, iPad와 같은 모바일 환경에

서도 사용할 수 있도록 개발된 운영체제이다. iOS를 탑재한 시스템은 강력한 보안기능을 갖추고 있지만 보안상의 취약점을 악용하여 iOS 커널을 조작하고자 하는 탈옥(Jailbreak)과 같은 공격 시도가 꾸준히 있었으며 애플은 이를 방지하기 위해 iOS에 대한 지속적인 업데이트과 다양한 보안 기능을 제공하여 보안을 강화하고 있다[9].

iOS는 [그림 3]에서 볼 수 있듯이 제조단계에서 하드웨어에 저장되어 수정이 불가능한 BootROM을 RoT로 하여 RSA인증 방식을 통한 보안부팅을 수행한다.

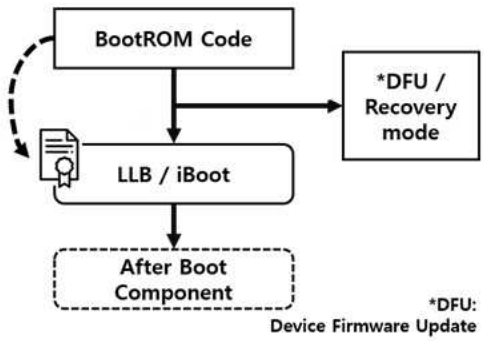


그림 3. iOS의 보안부팅

다음 부트로더를 검증하였을 때 인증에 실패하게 되면 DFU/Recovery mode를 통해 복구를 수행하여 기기의 공장 초기화를 수행한다.

VI. 스마트폰 환경의 보안부팅의 비교

스마트폰 환경에서 각 제조사들이 구현한 보안 부팅을 정리하면 [표 1]과 같다.

본 논문에서는 적용대상, Root of Trust(RoT), 암호화 알고리즘, 부팅 오류 시 처리방법, 업데이트 제공 주체로 나누어 비교해보았다.

여러 제조사에서 Android를 탑재한 다양한 기기를 출시하여 보안상 차이점이 발생하고 있어 구글은 공통 보안 환경을 위해 다양한 보안 구조를 설계하여 보안부팅을 지원하는 기술을 제공하고 있다. Knox는 삼성의 Android 스마트폰의 보안을 강화하기 위해 개발한 것으로 다양한 보안 기능을 갖추고 있고 그 중 보안부팅이 있다. 그리고 iOS의 경우 Apple의 스마트폰에 적용되며 강력한 보안기능 중 하나로 보안부팅을 수행한다.

각 스마트폰은 ARM Trustzone에 저장된 RoT를 기반으로 하여 공개키 암호화 알고리즘을 사용하여 보안부팅을 수행한다. Android와 iOS는 하드웨어에 저장되어 수정이 불가능한 BootROM을 RoT로 하며 Knox의 경우 비대칭키 쌍인 SSBK를 RoT로 한다.

Knox에서 보안부팅 중 오류가 발견되면 Know 보증 퓨즈를 설정하여 Knox와 관련된 기능과 Knox내의 정보를 사용할 수 없도록 하며 iOS에서는 보안부팅 중 인증에 실패한 경우 DFU/Recovery와 같은 기기 공장 초기화와 같은 복구를 수행하게 된다. 이에 비해 Android는 보안부팅 중 오류가 발생하였을 때 경고화면을 표시하게 되고 사용자의 결정에 의해 다음 단계의 부팅으로 진행할 수 있다. 이러한 점을 살펴보았을 때 Knox와 iOS는 보안부팅 과정 중 사용자가 제어할 수 있는 방법이 없어 부팅 중 오류만 없으면 보안부팅이 제대로 이루어지는데 비해 Android의 경우에는 보안부팅 과정 중 오류가 발생한 경우 사용자가 제어할 수 있도록 하여 보안부팅이 올바르게 수행하지 못하도록 할 수 있다.

Knox의 업데이트는 삼성에서 직접 제공하며 iOS 역시 애플에서 직접 제공하므로 취약점을 완화하는 업데이트의 주기가 빠르고 바로 업데

표 1. 스마트폰 환경에서의 보안부팅 비교

이름	적용대상	Root of Trust	암호화 알고리즘	부팅 오류 시 처리방법	업데이트 제공 주체
Android	여러 제조사의 스마트폰	BootROM	공개키(RSA-2048 이상)	장치의 부팅 상태에 따라 경고화면 표시	각 제조사
Knox	삼성의 Android 스마트폰	SSBK	공개키	Knox 보증 퓨즈를 설정하여 시스템에 저장된 중요정보 잠금	삼성
iOS	Apple의 스마트폰	BootROM	공개키(RSA-2048 이상, ECC-256이상)[10]	인증 실패시 DFU/Recovery 모드를 통한 공장 초기화	Apple

이트 할 수 있어 보안도 그만큼 안전하게 된다. 이에 비해 Android의 경우 새로운 업데이트가 출시되면 각 기기 제조사에서 해당 기기에 맞춰 업데이트를 개발하는 과정이 필요하다. 이는 기기 제조사들이 시간과 비용을 투자하여 업데이트를 개발해야 한다는 것이다. 그러나 경제적인 이유 등으로 기기 제조사에서 업데이트를 제공하지 않는 경우가 발생할 수 있다. 이로 인해 상위 버전에서는 해결된 취약점을 업데이트가 되지 못해 하위 버전으로 남아있는 기기들은 그대로 가지고 있을 수 밖 에 없다.

VII. 결론 및 향후 연구과제

스마트폰의 이용자가 증가하면서 스마트폰을 기반으로 하는 서비스가 늘어나고 있다. 특히 모바일 뱅킹과 같은 금융서비스의 경우 실행하는 스마트폰을 신뢰할 수 있어야 한다. 보안 부팅은 부팅 단계부터의 신뢰성을 제공한다.

본 논문에서는 스마트폰 환경에서의 보안부팅에 대해 조사하여 비교분석하였다. 스마트폰 환경에서의 보안부팅은 ARM Trustzone에 저장된 RoT를 기반으로 공개키 알고리즘으로 수행한다는 유사점이 있으며 환경에 따라 부팅 오류 시 처리방법이나 업데이트 제공 주체에 대해서는 상이하다는 것을 확인할 수 있었다.

향후 보안부팅 이후에도 스마트폰의 신뢰를 유지할 수 있는 보안기술에 대한 조사 및 비교분석이 필요할 것이다.

ACKNOWLEDGEMENT

이 논문은 2019년도 ETRI부설연구소의 재원으로 ETRI부설연구소의 지원을 받아 수행된 연구 사업임(2019-129)

[참고문헌]

[1] 고영태, “국민 95%가 스마트폰 사용…보급률 1위 국가는?”, KBS 뉴스, 2019.02.11., http://mn.kbs.co.kr/news/view.do?ncd=4135732

[2] 김다운, “'손안의 은행' 시대…모바일뱅킹 이용자·규모 증가”, 아이뉴스24, 2019.10.01., http://www.inews24.com/view/1211934

[3] W. A. Arbaugh, D. J. Farber and J. M. Smith, "A secure and reliable bootstrap architecture," Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097), pp.65-71, 1997.

[4] ARM, ARM Security Technology: Building a SecureSystem using Trustzone ® Technology, ARM Limited., 2009.

[5] “Mobile Operating System Market Share Worldwide”, StatCounter Global Stats, D.S.Milojicic, https://gs.statcounter.com/os-market-share/mobile/worldwide

[6] “Implementing dm-verity”, Android Open Source Project, https://source.android.com/security/verifiedboot/dm-verity

[7] “Verified Boot”, Android Open Source Project, https://source.android.com/security/verifiedboot

[8] Samsung Research America, Whitepaper: Samsung Knox Security Solution, Samsung Electronics Co., Ltd, 2017.

[9] Apple, iOS Secrutiy:iOS12.3, Apple Inc, 2019.

[10] Apple, Apple Inc. Certification Authority Certificate Policy, Apple Inc, 2014

안드로이드에서 JNI를 사용하는 공유라이브러리 동적분석에 대한 연구

최정수*

*고려대학교 컴퓨터정보통신대학원

Study on Shared Library Dynamic Analysis Using JNI In Android

Jeong-Su Choi*

*Graduate School of Computer & Information Technology, Korea University.

요 약

JNI(Java Native Interface)를 사용하는 안드로이드 어플리케이션의 공유라이브러리를 분석한는 3가지 방법에 대해서 고안하고 각각의 장단점을 비교했다. 또한 자동화 방법을 제시, 구현하여 취약점 분석에 접목할 수 있도록 연구를 수행했다.

I. 서론

프로그래밍을 하다 보면 베이스로 사용하고 있는 프로그래밍 언어 외에 다른 프로그래밍 언어를 사용하여 개발된 라이브러리를 사용하는 일이 가끔 있다. 예를 들어 안드로이드에서 GPS를 사용하는 어플리케이션을 개발할 때 개발자는 Application Framework에 있는 Location Manager를 사용하게 된다. Location Manager는 내부적으로 C++ 언어로 작성되어 있는 GPS 라이브러리 libgps.so에 있는 함수들을 사용한다. 이때 Location Manager는 안드로이드의 Application Framework이기 때문에 JAVA 언어로 작성되어 있다. 따라서 JAVA 언어에서 C++ 언어로 작성되어 있는 함수를 호출하는 방법이 필요하다. 이때 사용하는 것이 자바 네이티브 인터페이스(Java Native Interface, JNI)이다, JNI는 자바 가상 머신에서 실행되고 있는 JAVA 코드가 다른 언어들로 작성된 라이브러리들을 호출하거나 호출되도록 도와주는 프레임워크이다.

JNI(Java Native Interface)를 사용하는 안드로이드 어플리케이션의 공유라이브러리를 분석한는 방법은 모든 바이너리를 분석하는 방법과 같이 정적분석, 동적분석 크게 2가지로 나눌 수 있다.

정적분석은 공유라이브러리 파일(.so)이 ELF(Executable and Linkable Format) 파일 포맷이기 때문에 일반적인 바이너리를 분석하는 방법으로 접근한다. 하지만 공유라이브러리 파일(.so)을 동적분석 하는 것은 독립적으로 실행할 수 없기 때문에 일반적인 바이너리와 달리 어려움이 존재한다. 특히 안드로이드 어플리케이션 중 JNI를 사용하는 공유라이브러리를 동적분석하는 것은 안드로이드 기기 내부에 개발환경(gcc)등을 제공하고 있지 않기 때문에 다른 방법을 생각 해야한다.

본 연구에서는 JNI(Java Native Interface)를

사용하는 안드로이드 어플리케이션의 공유라이브러리를 동적분석 할 때 효율적으로 분석할 수 있는 방법을 제시한다.

II. 결론

JNI(Java Native Interface)를 사용하는 안드로이드 어플리케이션의 공유라이브러리를 동적분석 하기 위한 방법으로 3가지를 고안했다.

첫 번째 방법은 가장 간단한 방법으로 Frida를 사용하여 공유라이브러리의 함수를 후킹하는 방법이다. Frida는 Multi Platform Hooking Frameworks로 안드로이드 후킹을 지원한다.

```
Interceptor.attach
    (Module.findExportByName
        ("{{ libaray name }}", "{{ function name }}"),
        {
            onEnter: function (arg) {
                {{ SOMETHING CODE }}
            },
            onLeave: function (ret) {
                {{ SOMETHING CODE }}
            }
        }
```

[그림 1] Frida Native Function Hocking Example

[그림 1]과 같이 'Module.findexportByName' 함수를 사용하여 첫 번째 인자에 모듈 이름, 두 번째 인자에 함수 이름을 넣어 주면 해당 함수를 후킹 할 수 있다.

Frida를 통해 후킹하는 방법의 장점은 간단하게 코드를 작성하여 해당 함수의 Arguments와 Return Value를 수정할 수 있다는 점이다. 하지만 단점으로는 어플리케이션이 처음부터 실행되기 때문에 Overhead가 발생할 수 있으며 분석하고 싶은 원하는 함수를 직접 호출 할 수는 없다는 점이다.

두 번째 방법은 분석하고자 하는 어플리케이션을 리패키징하는 방법이다. 안드로이드 어플리케이션은 컴파일 과정을 거쳐 소스코드에서 DEX 바이너리 파일로 변경되며 프로그램을 묶어주는 패키징과 코드사인 작업이 추가되어 APK로 만들어진다. 리패키징은 어플리케이션으로 부터 소스코드를 획득하고 소스코드를 변경한 뒤 다시 어플리케이션을 제작하는 과정을 말한다. 따라서 분석하고자 하는 공유라이브러리의 함수를 호출하도록 코드를 추가하거나 변경함으로서 동적분석을 진행 할 수 있다.

어플리케이션을 리패키징하는 방법의 장점은 어플리케이션을 원하는대로 바꿀 수 있기 때문에 원하는 곳에서 공유라이브러리 함수를 호출할 수 있다는 점이다. 하지만 단점으로는 DEX 바이너리를 사람이 읽을 수 있도록 표현한 Smali 코드를 직접 수정해야한다. Smail 코드는 어셈블리어와 같기 때문에 sudo 코드와는 거리가 멀고 숙련도를 필요로 한다.

세 번째 방법은 어플리케이션을 새로 만드는 방법이다. 동적 링커는 Native 함수의 이름을 아래와 같은 규칙에 따라 찾는다. 따라서 JNI를 사용하여 공유라이브러리를 만들게 되면 아래와 같은 명명법이 쓰인다.

* the prefix Java_
* a mangled fully-qualified class name
* an underscore ("_") separator
* a mangled method name
* for overloaded native methods, two underscores ("__") followed by the mangled argument signature

```
package core.cyber.whitehat2018;

import android.os.Bundle;
import android.support.v7.app.c;
import android.view.View$OnClickListener;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import java.util.Random;

public class moleGame extends c {
  static {
      System.loadLibrary("native-lib");
  }
  public native String cs(int arg1) {
  }
```

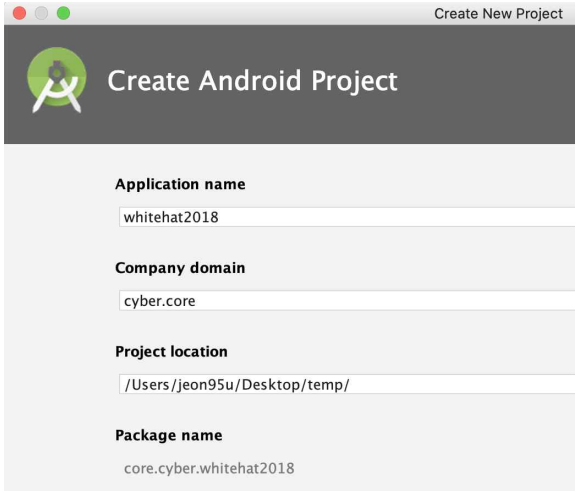
[그림 2] JNI Example Java Code

Exports

Name	Address
B4C	00000A50
B4C_len	00000A20
B4D	00000850
B4D_len	00000800
Java_core_cyber_whitehat2018_moleGame_cs	00001310

[그림 3] .so Dynamic Symbol Table

예를 들어 [그림 2]와 같은 Java 코드가 있으면 JNI Native function 이름 생성 규칙에 의해 공유라이브러리 상에서 cs 함수의 이름은 [그림 3]과 같다.



[그림 4] 새로운 어플리케이션 생성

따라서 [그림 4]와 같이 Application name, Company domain을 만들면 Package name과 분석하고자 하는 공유라이브러리의 함수 이름과 같게 되고 추가적으로 프로젝트 안에서 moleGame이라는 클래스 안에 cs 함수를 선언해 주면 분석하고자 하는 공유라이브러리를 그대로 가져와서 사용할 수 있다.

새로운 어플리케이션을 만드는 방법의 장점은 Java 코드로 분석가가 원하는 코드를 즉시 만들어서 테스트 해볼 수 있다는 장점이 있다. 하지만 단점으로는 새로운 APK를 빌드해야하는 오버헤드가 발생한다.

여기서 새로운 어플리케이션을 만드는 과정의 자동화도 생각해 볼 수 있다. 안드로이드 스튜디오에서의 기본적인 Build 시스템은 Gradle을 사용하며 Gradle은 GUI 외에도 CLI를 제공한다. 분석하고자 하는 공유라이브러리 파일의 Dynamic Symbol Table을 확인하여 JAVA_로 시작하는 함수를 찾아낸다. package 이름과 class 이름, 함수 이름을 분리하여 안드로이드 스튜디오 프로젝트를 생성하고 Gradle을 사용하여 build 하면 어플리케이션이 만들어진다.

이 연구를 통해 JNI(Java Native Interface)를 사용하는 안드로이드 어플리케이션의 공유라이브러리를 동적분석 하기 위한 3가지 방법을 개발하였다. 세 번째 방법의 경우 자동화도 가능하기 때문에 더 나아가 퍼징을 하는 등 취약점 분석에 대한 연구로 발전시킬 수 있다.

[참고문헌]

[1] Gang Tan, Andrew W. Appel, Srimat Chakradhar, Anand Raghunathan, Srivaths Ravi and Daniel Wang, Safe Java Native Interface, 2006

[2] William Enck, Damien Octeau, Patrick McDaniel, and Swarat Chaudhuri, A Study of Android Application Security, August, 2011

[3] Valerio Costamagna, Cong Zheng, ARTDroid: A Virtual-Method Hooking Framework on Android ART Runtime, 2016

[4] Goh Kondoh, Tamiya Onodera, Finding Bugs in Java Native Interface Programs, 2008

[5] N Totosis, C Patsakis, Android Hooking Revisited, 2018

[6] Java Native Interface Design Overview https://docs.oracle.com/javase/7/docs/technotes/guides/jni/spec/design.html

LibFuzzer를 이용한 APK 클래스 모듈 분석

박기현, 이성원, 황신운, 윤종희*

영남대학교 컴퓨터공학과

APK class module analysis with LibFuzzer

Gi-Hyun Park, Sung-Won Lee, Shin-Woon Hwang, Jonghee Youn

Dept. of Computer Engineering, Yeungnam University.

요 약

전 세계 스마트폰의 대부분을 차지하고 있는 안드로이드 OS는 하루에도 수백 개의 새로운 애플리케이션이 등장하고 있다. 안드로이드 애플리케이션이 생성되는 만큼 새로운 애플리케이션 테스트 도구들도 제시되고 있으나, 기존의 소프트웨어 테스트 도구를 활용한다면 현재보다 더 효율적인 테스트가 가능할 것이다. 본 논문에서는 기존의 소스코드 기반 소프트웨어 테스트 도구와 Java 리프팅 도구를 활용해 안드로이드 애플리케이션 분석 방법을 제안하고자 한다. 안드로이드 애플리케이션에서 추출한 Java 코드 파일을 LLVM IR로 변환한 후 C, LLVM IR 퍼징 테스트 도구인 LibFuzzer를 이용하여 원하는 특정 Java 클래스 모듈에 대해 취약점 테스트를 수행한다.

I. 서론

현재 오픈소스들을 대상으로 한 소스 기반 테스팅 연구들이 많이 진행되고 있다. 하지만 대부분의 코드 기반 테스트 도구는 특정 언어에만 국한되며, 다른 언어에 적용하기엔 문법적인 특성으로 문제가 발생한다. 이에 본 논문에서는 APK에서 안드로이드 소스 추출 도구 dex2jar[1], jd[2]를 이용해 Java 코드를 추출한 후, Java 코드를 LLVM IR로 변환해주는 리프팅 도구 JLang[3] 및 C, LLVM IR 기반 퍼징 테스트 도구 LibFuzzer[4]를 결합하여 APK의 특정 클래스 모듈을 대상으로 퍼징 테스트가 가능함을 확인한다.

* 교신저자(youn@yu.ac.kr)

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2019-2016-0-00313) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation), the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (No.2018R1D1A1B0705647) and Yeungnam University Research Grant.

II. 테스트 파이프라인 제안

테스트 가능성을 제안하기 위해 다음과 같은 도구들을 활용하고자 한다. Android APK에서 dex, class, java 추출 도구 dex2jar, jd와 Java를 LLVM IR로 변환하는 리프팅 도구 JLang을 활용한다. JLang 은 Java 코드를 사전에 컴파일하여 아키텍처의 범용성을 높이는 도구로써 이러한 리프팅 도구는 프로그램 언어를 중간언어로 변환해 원하는 아키텍처로 컴파일하는 과정에 도움을 준다. 코드 기반 퍼징 테스트 도구인 LibFuzzer는 C, LLVM IR 기반으로 동작하며, 테스트 대상 함수를 호출하는 Harness 파일을 작성하여 테스트를 수행할 수 있다.

2.1 테스트 환경 전체적인 구조

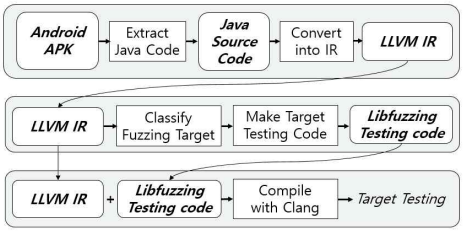


그림 1 테스트 프레임워크 구조

본 연구에서 제안하는 테스트 방법은 그림 1과 같이 크게 3단계 흐름으로 진행된다. 첫 번째 단계에서 APK 내 class 코드를 추출 및 Java 코드로 변환한다. 변환된 코드는 JLang 도구를 활용해 LLVM IR 코드로 변환한다. 여기서 생성된 LLVM IR 코드는 자체 제공하는 라이브러리를 포함하는 컴파일 과정을 거쳐야 한다. 두 번째 단계인 LibFuzzer의 Harness 코드작성은 첫 번째 단계에서 생성한 LLVM IR 코드를 기반으로 만든다. 테스트하고자 하는 함수의 이름, 인자 형태 및 반환 값을 가져와 형식에 맞게 작성하며, 이때 작성되는 테스트 코드는 JLang에서 제공하는 LLVM IR 파일 형식에 LibFuzzer의 LLVMFuzzerTestOneInput을 LLVM IR로 변환 후 덧붙여 작성한다. 마지막으로 작성된 코드는 LibFuzzer와 JLang의 동적 라이브러리를 포함하는 컴파일 과정을 거쳐 퍼징 바이너리를 생성한다.

III. 실험 및 결과

분석 방법의 적절성을 검증하고자 본 연구에서 제안하는 방법을 따라 실제 클래스 코드를 대상으로 변환 과정을 거쳐 LibFuzzer 테스트를 진행했다. 그림 2 예제 코드는 class가 되기 전 Java로 작성된 코드로 사용자의 입력값이 범위에 속하는 경우 예외처리되는 코드이다.

```
1. public static int fuz(int score){
2.         if(score >= 90) { return 1; }
3.         if(score >= 80 && score <=89) { return 2; }
4.         if(score >= 70 && score <=79) {
5.                 throw new RuntimeException("Crash!");
6.         }
7.         if(score >= 60 && score <=69) { return 4; }
8.         if(score <=59) { return 5; }
9.         return 0;
10.}
```

그림 2 예제 소스코드

테스트는 다음과 같은 순서로 진행됐다. 테스트하고자 하는 class 코드를 Java로 변환한 후 생성된 Java 파일을 JLang을 이용해 IR로 변환했다. 대상 함수에 대한 IR 소스코드를 바탕으로 LLVMFuzzerTestOneInput 함수를 작성했으며, 다시 LLVM IR로 변환한다. 변환된 두 IR(JLang IR 코드와 LibFuzzer IR 코드) 코드를 합친 후 JLang과 LibFuzzer의 컴파일 옵션을 결합해 바이너리를 생성한다. 그 결과 Java 코드를 LibFuzzer 실행 파일로 생성하여 퍼징 테스트를 동작할 수 있었으며, 함수를 대상으로 무작위 입력을 통해 범위 내에 속하는 특정 입력값을 찾아내어 예외처리를 발생시킬 수 있었다.

IV.결론

본 논문에서는 C언어 기반 테스트 도구를 Java 리프팅 도구와 결합해 APK의 특정 클래스 모듈에 대한 테스트 방법을 제안했다. 이러한 방식은 소스코드가 주어지지 않은 APK를 대상으로 코드 기반(LLVM IR) 테스트가 가능함을 보였으며, 새로운 대상 형식에 기존의 도구를 활용하여 범용적으로 사용할 수 있었다. 또한, Android 애플리케이션 시장이 활성화되고 LLVM IR을 이용한 프로젝트들이 증가하는 만큼 관련 도구를 복합적으로 활용 가능한 테스트 지침이 될 수 있다.

Acknolegdge

[참고문헌]

[1] https://github.com/pxb1988/dex2jar

[2] http://jdownloader.org/

[3] https://github.com/polyglot-compiler/JLang

[4] https://llvm.org/docs/LibFuzzer.html

행위 기반의 지속인증을 위한 스마트폰 인증 연구 동향 분석*

황은비, 박래현, 권태경

연세대학교 정보대학원 정보보호 연구실

A survey on Smartphone Authentication for Continuous Authentication based on User Behavior: A survey

Eunbi Hwang, Leo Hyun Park, Taekyoung Kwon

Information Security Lab, GSI, Yonsei University.

요 약

오늘날 스마트폰은 그 기능이 발전함에 따라 매우 많은 사용자 개인정보를 담고 있으며, 최근에는 스마트폰을 통한 은행 업무도 가능해짐에 따라 조금 더 강력한 스마트폰의 보안이 요구되고 있다. 현재까지 사용자 인증을 위해 사용되어온 PINs, 패턴락, 얼굴인식, 지문인식 등은 여러 가지 공격에 취약하며, One shot 인증으로 지속적인 사용자 인증이 불가능하다. 지속인증은 이러한 One shot 인증의 위험성을 보완하고자 제시되었다. 본 논문에서는 사용자 행위 기반의 지속인증을 위한 인증 연구의 동향을 소개한다.

I. 서론

오늘날 스마트폰은 개인적인 통신뿐만 아니라 은행 어플리케이션을 통해 계좌 이체, 공인인증서 발급 등의 은행 업무를 볼 수 있다. 따라서 스마트폰을 잃어버린다든지 누군가 훔쳐갔을 때, 사용자 인증을 통한 높은 보안성이 만족되지 않는다면 스마트폰에 저장된 개인 정보는 물론이고 계좌 유출로 인한 경제적 위험을 초래할 수 있다.

현재까지 스마트폰의 개인정보 유출을 막기 위한 사용자 인증기법으로는 PINs, 패턴락, 얼굴인식, 지문인식 등이 존재한다. 하지만 이 인증기법들은 smuge 공격, side-channel 공격, shoulder-surfing 공격 등에 취약하다 [3,7]. 또한, 해당 인증기법들은 One shot 인증으로 한번 사용자 검증이 완료되면 스마트폰이 꺼지기 전까지 또 다른 사용자 인증을 필요로 하지 않는다. 이는 사용자의 스마트폰이 사용자 인증을 완료한 후 켜진 상태에서 스마트폰을 도난당하거나 공격자가 One shot 인증의 취약점을 이용하여 사용자 스마트폰 암호를 풀었을 때, 또 다른 인증을 요구하지 않기 때문에 보안에 위험하다.

지속인증 (Continuous Authentication)은 이러한 One shot 인증의 위험성을 보완하고자 제시되었다. 스마트폰을 사용하는 동안 이루어지는 지속인증은 사용자의 행위에 따라 사용자를 인증이 이루어지게 된다.

본 논문에서는 사용자의 행위로써 소프트 바이오 인식, 터치 입력, 활동 패턴, 응용프로그램 사용 패턴 기반의 사용자 지속인증 기법에 대한 연구 동향을 소개한다. 또한, 이러한 관점에서 향후 본 연구실의 행위 기반의 지속인증을 위한 스마트폰 인증 연구 방향을 소개한다.

* 본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2019-2016-0-00304)

II. 지속인증 연구 동향

2.1. 소프트 바이오 인식기반 사용자 지속인증 (Soft Biometrics)

Niinuma 등은 컴퓨터 사용자가 로그 아웃을 하지 않고 자리를 비울 때 다른 사람이 컴퓨터를 사용하는 보안의 문제를 해결하고자 사용자 로그인 후 사용자의 얼굴색과 옷색과 같은 소프트 바이오 인식을 기반으로 사용자를 검증하는 지속인증 스킴을 제시했다 [6].

해당 인증 스킴은 총 4개의 모드가 존재한다. 첫 번째 모드에서는 먼저 패스워드를 통해 사용자 인증을 한 후 사용자의 얼굴을 탐지하고 사용자의 몸이 위치와 크기를 파악한다. 이를 통해 얼굴색과 옷색에 관한 히스토그램을 생성하고, 얼굴의 Eigenface 표현을 계산하여 사용자 견본 (template)으로 저장한다.

두 번째 모드에서는 사용자의 얼굴과 몸의 색 히스토그램과 PCA기반 얼굴 인식 기술(Eigenface)을 통해서 1초마다 사용자의 얼굴 인식이 진행된다. 다음으로 사용자 견본과 사용자 인식결과를 비교하여 유사도를 계산한다. 이 때, 유사도가 임계값 보다 작게 되면 세 번째 모드로 넘어가게 된다.

세 번째 모드에서는 해당 인증 시스템에서 사용자가 있는 환경에서 빛의 변화가 있는지 파악하고 빛의 변화가 포착되면 사용자의 견본을 업데이트하여 지속인증을 유지 시킨다.

하지만 만약 세 번째 모드에서 만약 사용자가 인증 기기 앞에 있지 않다면 네 번째 모드로 넘어가서 시스템을 잠가버리고 첫 번째 모드로 돌아가게 된다.

해당 지속인증 스킴은 지속인증 중에 사용자의 얼굴색과 옷색으로 구분하는 soft biometrics 인증기법을 사용하여 인증의 유용성을 높였다. 또한, 지속인증 중에 공격자가 발생했다 판단했을 시 얼굴인증이나 지문인식과 같은 hard biometrics 인증기법을 사용하여 다시 사용자 로그인을 하는 방법을 통해 보안성을 높였다.

2.2. 사용자 터치 입력 기반 사용자 지속인증 (Touch biometrics)

Xu 등은 사용자의 터치스크린에서의 터치 작동에 기반 한 사용자 인증 스킴을 제시했다[4].

해당 연구는 사용자의 특징을 모델링 하기 위해 터치 작동 중 keystroke, handwriting, slide, pinch를 사용자 feature set으로 추출했다. 해당 연구의 터치 작동에 기반한 사용자 인증 스킴은 터치 작동 구별 모델과 사용자 인증 모델의 2가지의 모델을 사용하였으며, 터치 작동 구별 모델로는 multi-classification 모델을 사용자, 인증 모델로는 one-class SVM을 사용하였다.

해당 연구는 32명의 사용자와 한 달간의 실험을 통해 해당 논문에서 지정한 4가지의 터치 작동 feature가 사용자를 구분할 수 있는 바이오 특성임을 입증하였다. 또한, 실험을 통해 해당 연구에서 구현한 사용자 분류모델의 사용자 인증 정확도가 80% 이상임을 알 수 있었다.

2.3. 사용자 활동 패턴 기반 사용자 지속인증

Ehatisham-Ul-Haq 등은 걷기, 뛰기, 서 있기, 앉아 있기, 계단 오르기, 계단 내려가기의 6가지 활동 패턴 기반으로 사용자 지속인증 스킴을 제안했다 [3]. 이 스킴은 특히 스마트폰의 여러 위치(팔 위쪽, 손목, 허리, 왼쪽 주머니, 오른쪽 주머니)에서의 사용자 활동 유형에 따른 사용자 인증 정확도를 제공한다.

해당 연구는 피지컬 활동 인지를 위한 존재하는 데이터셋 (Shoaob et al., 2013, 2014)을 이용하였다. 또한, 해당 데이터 셋에서 피지컬 활동 인지 연구 (Anjim and Ilyas, 2013; Incel et al., 2013; Shoaib et al., 2014, 2013; Su et al., 2014)에서 사용되어왔던 16개의 features를 추출했다. 다음으로 SVM 모델을 이용하여 활동 패턴 기반 사용자 지속인증 스킴을 제시하였다.

제시된 지속인증 스킴의 정확도 파악을 위해 10명의 참여자를 대상으로 실험을 진행했다. 실험은 스마트폰을 여러 신체 부위에 위치시키고, 각 위치마다 6가지의 활동을 진행하였다. 해당 실험은 스마트폰에 내장되어있는 acceleroneter,

gyroscope, magnetometer 센서를 통해 feature 데이터를 추출하여 활동 패턴 기반 사용자 지속인증 스킴을 평가했다.

걷기와 뛰기 활동에서 왼쪽 주머니와 오른쪽 주머니에 스마트폰이 있을 때, 사용자구별 정확도가 100%이고, 스마트폰이 손목과 허리에 있을 때, 99.6% 정확도로 사용자 분류가 가능했다. 서 있는 상황에서는 허리 벨트에 스마트폰이 걸려있을 때 정확도 99.4%로 사용자 인증이 가장 잘 되었으며, 앉아 있는 상황에서는 왼쪽 주머니에 있을 때 정확도 99.3%로 사용자 인증이 잘 되었다. 계단을 오르고 내리는 활동에서는 오른쪽 주머니에 스마트폰이 있을 때 각각 98.7%, 96.5%의 정확도로 사용자를 인증할 수 있음을 알 수 있었다.

2.4. 웨어러블 디바이스를 통한 사용자 활동 패턴 기반 지속인증

Wu 등은 wearable device를 개발하여 사용자의 손가락 끝에 착용 시 자동으로 기기를 착용한 사람이 진짜 사용자인지 지속 인증할 수 있는 스킴을 제안했다 [2]. 해당 기기는 행동을 측정하는 acceleration 과 angular velocity 센서와 심박 수를 측정하는 photoplethysmography (PPG) 센서로 이루어있으며 기기를 통해 센서 데이터를 추출할 수 있다. 이런 센서 데이터를 통해 사용자의 동적인 활동에서의 사용자의 피지컬 데이터를 얻을 수 있다.

해당 센서 데이터 수집을 위해 총 40명 (남자:30명, 여자: 10명) 대상으로 걷기, 앉기, 과격한 움직임(빠른 걸음, 계단 오르내리기)의 행위를 30일 동안 각각 하루에 20번씩 진행하여 총 7200개의 데이터를 생성했다.

Wearable device를 이용한 사용자 활동 패턴 기반 지속인증 스킴은 먼저 걷기, 앉기, 과격한 움직임(빠른 걸음, 계단 오르고 내리기)과 같은 사용자의 행위를 구분한 다음 사용자를 구분한다. 이때, acceleration과 angular velocity의 평균과 분산, PPG의 평균값을 사용자 행위 feature로 한 K-nn 알고리즘을 이용하였다. 또한, 사용자 구분은 time-domain, frequency-domain, wavelet-domain 값을 feature로 한 SVM 알고리즘을 이용했으며 해당 스킴의 사용자 지속인증의 정확도는 98.5%였다.

또한, Mare 등은 스마트 시계 또는 피트니스 밴드 등의 손목밴드를 스마트폰과 연동시켜 진짜 사용자가 스마트폰을 쥐고 있음을 지속적으로 인증하는 스킴인 CSAW (Continuous Smart phone Authentication using Wristbands)을 제안했다 [5].

해당 스킴에서 사용되는 손목밴드는 accelerometer 와 gyrosecope 센서가 내장되어있어야 한다. 따라서 해당 센서들을 통해 손목밴드 기반 사용자 지속인증 스킴은 사용자가 핸드폰을 주머니에서 꺼내거나 핸드폰을 들어 올리는 행위 등을 구분할 수 있다.

해당 스킴의 초기 인증은 사용자가 스마트폰을 사용하기 전에 발생하는 자연적 상호작용(스마트폰 들어 올리기, 스마트폰 잡기)을 활용하여 빠르게 구현한다. 추가적으로, 초기 인증 후에 지문인식, 안면인식, 또는 다중 요소 인증 시스템 중 하나인 안드로이드 스마트 락과 함께 사용한 인증기법을 사용할 수 있다. 초기 인증과 사용자 인증이 시행되고 난 이후에는 지속인증이 진행된다. 해당 지속인증 스킴은 스마트폰을 사용하고 있는 내내 confidence score를 계산하며, confidence score 값이 일정 수치보다 작게 되면 현재 사용자가 진짜 사용자가 아니라고 판단되어 스마트폰이 다시 잠금이 걸리게 된다.

CSAW는 단순히 스마트폰을 들어 올릴 때에 99% 정확도로 사용자를 인증하며, 많은 사람들이 사용하는 손목밴드 (스마트 워치, 피트니스 밴드)를 사용하여 사용성이 편리한 특징을 지녔다. 또한, confidence score를 통해 사용자 지속인증 알고리즘의 보안성도 높일 수 있었다.

2.5. 응용프로그램 사용 기반 사용자 지속인증

Mahbub 등은 사용자의 스마트폰 응용프로그램 사용 패턴을 기반으로 하는 사용자 지속인증 스킴을 제시했다 [1].

해당 연구는 스마트폰의 어플리케이션 사용

량 공공 데이터인 UMDAA-02와 Securacy Application Usage Dataset을 이용하여 시행되었다. 이때, UMDAA-02 (University of Maryland Active Authentication Dataset)의 어플리케이션 사용량 데이터를 사용했으며, 해당 데이터셋에서 사용자 간의 스크린의 잠금과 잠금해제 시의 시간, 전화의 시작과 끝의 타임스탬프, 현재 실행되고 있는 어플리케이션의 정보를 알 수 있다. 다음으로 Securacy Application Usage 데이터셋에는 현재 실행되고 있는 어플리케이션, 설치 또는 삭제되거나 업데이트된 어플리케이션, 어플리케이션 서버 연결, 기기 위치 등의 정보를 제공한다.

응용프로그램 사용 패턴 기반 사용자 지속인증은 먼저 사용자의 응용프로그램 데이터로부터 현재 응용프로그램의 상태를 시간 정보와 일치시킨다. 다음으로 응용프로그램 상태 정보와 시간 정보를 합쳐놓은 데이터에서 특정 사용자의 어플리케이션 사용 패턴을 모델링 한다. 해당 state-space model은 서로 다른 어플리케이션을 하루 또는 주간, 주말 동안 다른 시간에 사용하는 것에 대한 패턴들을 모델링 하는 것으로써 사용자마다 가지고 있는 생활 패턴이 다르게 존재하기 때문에 사용자 인증 모델로써 사용 가능하다. 다음으로 훈련된 sate-space model을 통해 테스트 데이터는 사용자 검증 점수를 생성하고 해당 점수를 임계값을 기반으로 사용자를 인증하게 된다.

III. 결론 및 향후 계획

본 논문에서는 행위 기반의 사용자 지속인증 기법에 관한 연구 동향에 대해 살펴보았다. 지속인증에서 사용자의 동적 활동과 사용자의 얼굴색, 옷색을 행위 데이터로 활용하여 사용자 인증을 진행할 수 있다. 또한, 사용자의 터치 작동과 사용자의 응용프로그램 사용 패턴에 따라 사용자를 구분할 수 있다.

한편, 본 논문에서 소개된 사용자 활동기반의 지속인증기법은 동적인 활동 (걷기, 계단 오르내리기 등)기반으로 이루어진다. 하지만 보다 실제 인증 환경을 고려한다면 정적인 활동 (누워있기, 기대어있기 등) 또한 고려 되어야 한다. 따라서 본 연구실은 사용자의 정적인 사용환경을 고려한 사용자 인증에 관한 연구를 진행할 것이며, 이때, 검증의 신뢰성을 높이기 위해 유저 스터디의 규모를 높이는 방향으로 연구를 진행할 것이다.

[참고문헌]

[1] MAHBUB, Upal, et al. Continuous Authentication of Smartphones Based on Application Usage. IEEE Transactions on Biometrics, Behavior, and Identity Science, 2019.

[2] WU, Guannan, et al. A continuous identity authentication scheme based on physiological and behavioral characteristics. Sensors, 2018, 18.1: 179.

[3] EHATISHAM-UL-HAQ, Muhammad, et al. Continuous authentication of smartphone users based on activity pattern recognition using passive mobile sensing. Journal of Network and Computer Applications, 2018, 109: 24-35.

[4] XU, Hui; ZHOU, Yangfan; LYU, Michael R. Towards continuous and passive authentication via touch biometrics: An experimental study on smartphones. In: 10th Symposium On Usable Privacy and Security ({SOUPS} 2014). 2014. p. 187-198.

[5] MARE, Shrirang, et al. Continuous Smartphone Authentication using Wristbands.

[6] NIINUMA, Koichiro; PARK, Unsang; JAIN, Anil K. Soft biometric traits for continuous user authentication. IEEE Transactions on information forensics and security, 2010, 5.4: 771-780.

[7] KU, Yeeun, et al. Draw It As Shown: Behavioral Pattern Lock for Mobile User Authentication. IEEE Access, 2019.

P2P 네트워크 기반 악성코드 분석 생태계 구현 방법

최지헌*

*아주대학교 소프트웨어학과

A method to implement malware analysis ecosystem based on P2P network

Jiheon, Choi*

*Division of Software, Ajou University.

요 약

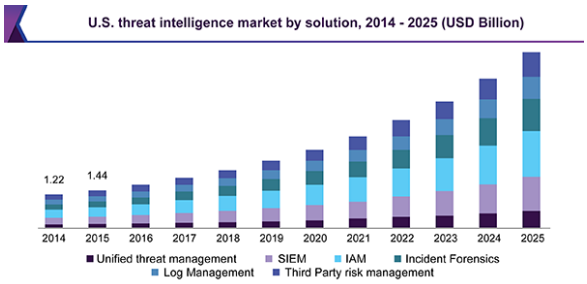
최근 다양한 경로를 통해 랜섬웨어와 같은 악성코드가 무차별적으로 유포되며 발생하는 피해가 증가함에 따라, 사전에 악성코드 감염을 방지하는 기술이 지속적으로 요구되면서 사이버 위협 인텔리전스(Cyber Threat Intelligence) 시스템의 필요성이 대두되고 있다. 종래의 시스템 운영은 중앙에서 악성코드 샘플과 같은 위협 데이터를 관리하다보니 저장소를 운영함에 있어 막대한 비용이 투입되는 부담을 지니고 있다. 또한, 다양한 방법으로 수집된 위협 데이터들이 고가에 거래되고 있어, 자본력이 충분하지 않은 구매자에게는 진입 불가능한 기술시장이 되었다. 이에 따라, 구매력 및 진입장벽의 문제를 해소함과 동시에 블록체인을 바탕으로 시스템 요소가 상호 유기적으로 연결됨으로써 새로운 위협 정보를 찾아내고 공유되도록 하기 위한 분석 생태계 구현 방법에 대해 소개하고자 한다.

I. 서론

선진국과 개발도상국을 불문하고 사이버 범죄(공격)의 빈도수는 해가 갈수록 가파르게 증가하고 있으며 이에 대한 우려와 해결방법이 대두되고 있다. 이에 산업계와 정부는 위협을 완화하거나 제거하기 위해 수집된 위협 요소들을 바탕으로 최적의 해결방법을 찾기 위해 노력하고 있다.

전 세계 사이버 위협 인텔리전스 시장의 규모는 2016년 약 32억 2천만 달러로 추정된다[1]. 그러나 다양한 방법과 막대한 투입비용으로 수집된 위협 데이터들은 고가에 거래되고 있어, 자본력이 충분하지 않은 구매자에게는 진입 불가능한 시장이 되었다,

따라서 본 논문에서는 구매력 및 진입장벽의 문제를 해소함과 동시에 수집, 분석, 저장의 시스템 요소가 상호 유기적으로 연결되어 새로운 위협 정보를 찾아내고, 수집된 정보들을 블록체인을 통해 공유하여, 악성코드 분석으로 이러지는 생태계를 구현하는 방법을 제시한다. 최종적으로는 생태계의 구현을 통해 실현 가능성을 입증하는 것을 목적으로 한다.



[그림 1] Threat Intelligence 시장 분석 자료

II. 제안 모델

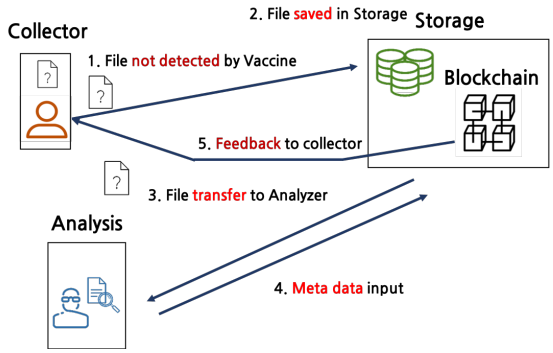
2.1. 사이버 위협 인텔리전스 (Cyber Threat Intelligence) 정의

가트너에서는 사이버 위협 인텔리전스에 대

해서 다음과 같이 정의한다. 매커니즘, 지표, 조언 등을 비롯하여 기존 또는 새롭게 등장한 위협 또는 자산(Assets)에 대한 위험에 대한 현재 존재하거나 발생이 임박한 위협에 대한 증거 기반의 지식이며, 이는 위협 대응에 필요한 의사결정에 사용된다 [2].

2.2. 블록체인 기반 사이버 위협 인텔리전스 플랫폼 (Cyber Threat Intelligence Platform)

생태계 시스템은 그림 2와 같이, 크게 세 가지의 역할을 가진 노드로 구성된다. 수집 노드는 백신을 통해 악성 행위가 의심되는 파일을 탐지하여 스토리지 노드로의 전송을 수행하고, 스토리지 노드에서는 수집 노드에서 전송받은 파일을 분산하여 저장하거나 분석노드의 성과에 대한 코인 보상을 수행하며, 분석노드는 악성코드 분석가로 구성되어 스토리지 노드에 올라온 파일을 분석하여 블록체인 메인넷에 트랜잭션 업로드 하는 역할을 수행한다.



[그림 2] 생태계 시스템 구성도

2.2.1. 수집 노드

수집 노드는 파일 시그니쳐(File signature) 기반으로 동작하는 백신이 실행되고 있는 컴퓨터로써, 일반 사용자(End-user)가 참여할 수 있는 노드가 될 수 있다. 백신 실행 결과값에 따라 수집노드는 P2P 네트워크에 연결되어 있는 스토리지 노드에 분산하여 전송할 수 있다. 파일 전송이 이루어질 때 스토리지 노드 파일 저장소에는 파일을 제공한 수집 노드의 peer ID가 기록된다. 분석 노드에 의해 분석이 완료된 후 악성코드로 판별을 경우 트랜잭션으로 업로드 된 분석 리포트에 대한 투표(Voting) 수에 따라 파일을 제공한 수집 노드에 대한 보상이 이루어진다.

2.2.2. 스토리지 노드

스토리지 노드는 두 가지 특징을 지닌다.

(1) 수집 노드로부터 전송받은 파일을 분산해시 테이블(Distributed Hash Table, DHT) 기술을 이용하여 저장하는 스토리지 기능 수행

(2) 생태계 구현 검증을 위해 최소한으로 구성된 블록체인 기능을 지니고, 위임 지분 증명(Delegated Proof of Stake) 방식으로 구현

스토리지 노드는 위협 정보를 구매하기에 자본력이 부족한 구매자들을 대상으로 하며. 본인의 컴퓨터 자원을 악성코드 분석 생태계에 제공함으로써 생태계 발전을 위하여 기여한 정도에 따라 공유된 위협정보를 제공받을 수 있다.

2.2.3. 분석 노드

주로 자본력이 부족한 악성코드 분석가가 악성코드 샘플을 구하기 위해 참여하며, 분석 완료된 파일은 저장된 스토리지 노드의 peer ID 값과 함께 파일의 메타데이터를 포함한 분석가의 리포트 형태로 메인넷에 트랜잭션으로 업로드 된다. 블록체인 블록의 형태는 다음과 같다.

```
{
    "index": 1,
    "timestamp": 1544281137074,
    "postList": [],
    "replyList": [],
    "voteList": [],
    "transactionList": [],
    "nonce": 0,
    "hash": "",
    "previousBlockHash": ""
},
```

[그림 3] 블록체인 제네시스 블록[1)]

그림 3과 같이, 블록 안에는 악성코드 분석가

1) 제네시스 블록: 블록체인에서 생성된 첫 번째 블록을 말한다.

의 의견이 담긴 리포트와 그에 대한 댓글이 저장된다. 또한, 리포트에 대해 투표(Voting)한 노드의 peer ID, public key가 저장된다. 피드 형태로 구현된 웹페이지에서 투표할 수 있으며 평점이 높을수록 악성코드 리포트의 신뢰도는 향상된다. 이는 수집 노드에서 동작하는 백신의 악성코드 DB를 업데이트하는 기준이 될 수 있다. 마지막으로, 투표 수에 따라 분석노드는 블록체인 기반으로 발행된 코인을 통해 보상 받게 된다.

Ⅲ. 향후 연구 과제

추가 연구를 통해 블록체인 네트워크로 구성된 생태계에 참여함으로써 노드들의 보상 체계를 구체화해야 하고, IPFS (InterPlanetary File System) 기술 [3]을 사용하여 스토리지 노드에서 파일을 분산 저장한 후 파일을 읽어올 때 속도가 저하되는 문제 등의 파일 분산 처리 성능 향상은 추가 연구가 필요할 것이다.

Ⅳ. 결론

본 논문에서는 P2P 네트워크 기반으로 동작하는 블록체인을 이용하여 Threat Intelligence를 위한 악성코드 분석 생태계와 그 필요성에 대해 다루었다. 종래의 위협 인텔리전스 시스템의 중앙 스토리지 운영 문제를 해결하고, 블록체인 네트워크를 통해 위협 정보를 공유하면서 중요한 정보자산에 대한 위협 대응을 보다 더 빠르게 할 수 있도록 제안하였다. 이 연구를 통해 미래의 위협 정보들은 분산되어 P2P 네트워크 상에서 상호 유기적으로 연결됨으로써 각 노드들 간의 협업을 통해 보안 분야의 수준을 더욱 향상시킬 것으로 기대된다.

[참고문헌]

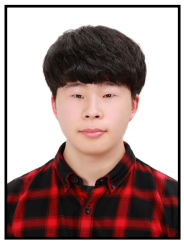
[1] Gartner, “Market Guide for Security Threat Intelligence Products and Services”, 2019.1.

[2] Gartner, “Definition : Threat Intelligence”, 2013.5.

[3] Juan Benet, “IPFS-content addressed, versioned, P2P file system”, arXiv preprint arXiv:1407.3561 (2014)

[4] BRUNO JOACHIM KWEKA, “An Access Control Based on Smart Contract on IPFS Storage”, 부경대학교 대학원 석사학위논문, 2019. 13~ 14p

< 저 자 소 개 >



최 지 헌 (Jiheon Choi)

2019년~현재 : 아주대학교 소프트웨어학과 재학

2019년~현재 : ㈜와파스시스템즈 선임연구원

관심분야 : 웹 보안, 블록체인

임베디드 시스템의 RDP[1]와 ARP 취약점 분석 및 해결방안

정이원*, 허태겸*
*대구대학교 정보보호영재교육원

RDP and ARP Vulnerabilities on Embedded Systems

Yi Won Chung*, Tae Gyeom Heo*
*Information Security Institute for the Gifted, Daegu University

요 약

본 연구에서는 임베디드 시스템을 이용함에 있어서 ARP와 RDP Bluekeep 취약점을 분석하고 가상 임베디드 머신에 대한 모의 해킹으로 관련 시스템의 취약점이 미치는 부정적인 파급효과를 규명하였다. 또한, 본 연구결과는 RDP Bluekeep 취약점을 통해 관리자 권한을 쉽게 탈취할 수 있다는 점 그리고 ARP Spoofing 기법을 통해 클라이언트의 각종 프로토콜로 진행되는 통신의 내용을 유출할 수 있다는 점을 실증적으로 보여준다. 이러한 취약점에 대한 분석을 통해 해결방안을 제시하고자 하였다.

ABSTRACT

This study examines the ARP and RDP Bluekeep vulnerabilities on using Embedded Systems and identifies the possible implications of such vulnerabilities by performing penetration testing on virtualized embedded machines. Furthermore, this study elaborates on that the Administrative privileges can be easily taken away through the RDP Bluekeep vulnerability, and that all packets containing communication information of various protocols could be severely leaked by the ARP Spoofing method. The result of this study presents the solutions for these vulnerabilities.

Ⅰ. 서론

본 연구에서는 기업체, 교육 및 공공 기관에서 사용하는 임베디드 시스템의 여러 가지 취약점을 분석하고 이를 악용할 수 있는 가능성과 해결방안에 대해 살펴보고자 한다.

Bluekeep 취약점 (CVE-2019-0708)을 통해 Use After Free 메모리 공격 기법을 이용하여 관리자 권한에 접근하는 것과 ARP Spoofing을 이용하여 실제 라우터를 가장함으로써 클라이언트의 모든 패킷을 탈취하는 중간자 공격을 통한 직접적 정보 탈취 등의 공격 방법이 이를 뒷받침한다.

이러한 기초적인 공격으로 쉽게 개인정보 및 사생활 유출이 발생할 수 있는데 이는 곧 기업 또는 법인에 금전적 손실을 초래할 수 있고 사회적으로 큰 타격을 입힐 수 있다. 따라서 이러한 손실과 타격을 입히는 행위에 대하여 대응할 수 있는 대책을 모색하기 위해 최대한 비슷한 형태로 구성한 가상 머신을 토대로 본 연구를 진행하였다.

1) Remote Desktop Protocol: 원격 데스크톱 프로토콜

II. 배경 지식

2.1. 임베디드 시스템

임베디드 시스템은 기계나 기타 제어가 필요한 시스템에 대해, 제어를 위한 특정 기능을 수행하는 컴퓨터 시스템으로 장치 내에 존재하는 전자 시스템이다. 대표적인 임베디드 시스템으로는 은행 ATM, 자동차 내비게이션, 슈퍼마켓의 결제용 컴퓨터 또는 공공기관의 안내용 키오스크가 있다. 임베디드 시스템의 안정성을 추구하는 특성상 윈도우 7과 그 이전의 운영체제를 사용하고 있는 기기들이 매우 많다.

2.2. ARP Spoofing

ARP Spoofing은 근거리 통신망(LAN) 하에서 주소 결정 프로토콜(ARP) 메시지를 이용하여 상대방의 데이터 패킷을 중간에서 가로채는 중간자 공격 기법이다. 이 공격은 데이터 링크 상의 프로토콜인 ARP를 이용하므로 근거리 통신에서만 사용할 수 있는 공격이다.

2.2. Use After Free 취약점

메모리의 heap 영역에서 할당된 공간을 비우고 다시 사용할 때 일어날 수 있는 취약점이다. 메모리 블록이 비할당되고 난 뒤 그에 상응하는 포인터가 NULL 값으로 설정되지 않고 다음에 그 메모리 블록이 사용되려는 그 순간 메모리의 코드를 수정하면 원하는 코드를 실행하게 할 수 있다.

2.3. RDP

RDP (Remote Desktop Protocol)은 원격 접속용으로 개발된 프로토콜로, 다른 컴퓨터에 세션 할당 방식으로 GUI (그래픽 사용자 인터페이스를) 제공하는 프로토콜이다. TeamViewer의 원격 데스크톱과 같이 로그인된 상태에서 화면을 공유하는 것이 아닌 자체의 세션 기반이므로 다른 컴퓨터에서 대상 컴퓨터로 로그인을 하면 대상 컴퓨터의 계정은 자동으로 로그아웃이 되는 특징이 있다.

2.4. NAC

네트워크 접근 제어(Network Access Control)는 바이러스 백신, 호스트 침투 방지, 취약점 분석과 같은 보안 기술, 사용자 또는 시스템 인증, 그리고 네트워크 보안을 하나로 묶으려는 시도를 의미한다. 본 연구에서는 여러 세트의 프로토콜을 이용하여 보안정책을 정의하고 실행하고자 한다.

III. 본론

3.1. 네트워크 모델링

본 논문에서는 임베디드 시스템을 원격 조종으로 관리하는 컴퓨터로 정의한다. 이러한 네트워크는 이를 운용 목적의 특성상 보안 장비에 높은 비중의 투자가 이뤄지지 않는 것이 일반적 특징이다.

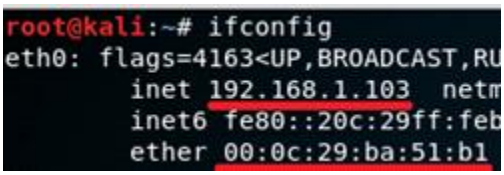
3.2. 취약점 분석

(1) ARP Spoofing으로 패킷 감청

특정 패킷이 telnet이나 SSL을 사용하지 않는 웹 같은 데이터가 암호화되지 않는 프로토콜은 사용해서 전송될 경우 패킷을 얻을 수 있으면 무단 정보 탈취가 손쉽게 이루어진다. 단순히 단일 클라이언트에 프록시 서버를 설치하여 패킷을 감청하는 것도 가능한 것에서 같은 네트워크망 내에 있으면 이루어질 수 있는 ARP Spoofing, Router Hijacking을 통해서 공격할 수 있다.

다음 실험은 ARP Spoofing의 시연을 통하여 공격 대상 컴퓨터의 패킷을 탈취해오는 중간자 공격을 요약해 제시한 것이다. 실험하는데 이용한 컴퓨터는 두 대이며, 공격자인 Kali Linux와 공격 대상의 Windows 10의 운영체제가 각각 설치되어 있다.

첫 번째 절차는 두 컴퓨터의 IP를 파악하는 것이다. 공격 대상은 (192.168.1.105), 공격자는 (192.168.1.103)인 것을 확인한다.



<그림 1 – 공격자의 IP와 MAC Address[2)]>



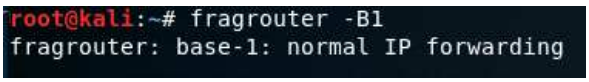
<그림 2 – 공격 대상의 IP>

다음으로 게이트웨이인 (192.168.1.1)의 MAC Address를 파악한다.



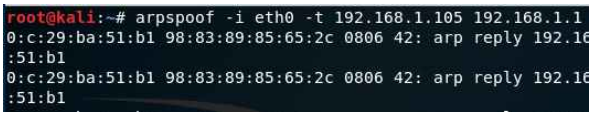
<그림 3 – 라우터의 실제 MAC Address>

공격자에서 라우터 에뮬레이터인 fragrouter를 통해 가상 라우터를 생성한다.



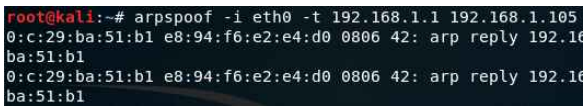
<그림 4 – fragrouter 가상 라우터 생성>

공격 대상에게 공격자가 라우터로 가장하기 위해 ARP Reply를 전송한다.



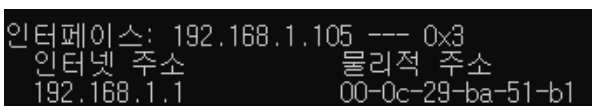
<그림 5 – arpspoof로 공격 대상을 스푸핑>

그런데 공격 대상의 모든 패킷이 공격자로 전달하게 되면 패킷들이 진짜 라우터에 도달하지 않아 공격 대상의 인터넷 연결이 끊어지게 되므로 진짜 라우터에 패킷을 전달하기 위해 똑같이 공격자가 공격 대상을 가장하여 ARP Reply Packet을 전송한다.



<그림 6 – 실제 라우터를 스푸핑>

라우터의 IP 주소에 할당된 MAC Address가 공격자의 것임을 확인할 수 있다.



<그림 7 – 변조된 MAC Address>

2) Media Access Control Address: 데이터 링크 계층에 소속된 인터페이스에 할당되는 주소

이제 공격 대상의 모든 패킷이 공격자의 컴퓨터로 전송되므로 공격자가 Wireshark 패킷 캡처 프로그램을 실행하여 패킷을 캡처하고 공격 대상 컴퓨터에서 (https://www.google.com/)에 접속하면 다음과 같이 공격 대상의 IP 주소에서 전송되는 패킷이 나타나는 것을 확인할 수 있다.

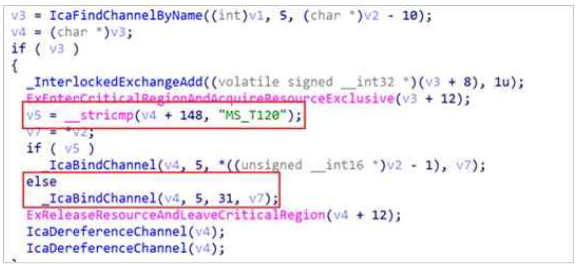


<그림 8 – 전달되는 공격 대상의 패킷>

(2) RDP Bluekeep

최근에 발견된 BlueKeep 취약점(CVE-2019-0708)의 경우 패치 권고안에 따르면 공격자는 인증 없이도 서버에 코드 실행이 가능해진다. 보안업체 Errata Security 조사(2019)에 의하면 약 95만대의 윈도우 시스템이 해당 취약점에 노출되어 있는 것으로 조사되었다. RDP는 연결 과정에서 클라이언트가 서버에 정적 채널을 요청하면 서버는 요청된 채널에 대하여 채널 번호를 할당하고 해당 채널을 통해 통신하게 된다.

Microsoft에서 배포한 보안 패치를 Hex-rays의 IDA Pro 프로그램을 이용하여 분석해보면 채널 번호와 객체를 바인딩하는 "_IcaBindChannel" 함수가 변경된 것을 확인할 수 있다. "_IcaBindChannel" 함수 호출 전에 "MS_T120" 문자열이 존재할 경우 세 번째 인자가 강제로 31로 변경되도록 패치되었다.



<그림 8 – 패치된 _IcaBindChannel>

취약점에 노출된 "MS_T120" 채널은 일반적으로 요청되는 채널이 아니고 이미 할당되어 있다. 그런데 사용자가 명시적으로 "MS_T120" 채널을 요청할 경우 임의의 번호와 "MS_T120" 객체가 바인딩 된다. 이렇게 하나의 객체에 두 개의 채널 번호가 할

당되면 마지막에 해당 채널이 두 번이나 비할당되면서 메모리 오류가 발생하여 Use After Free(UAF) 취약점을 사용할 수 있다.

해당 함수의 라이브러리를 분석해보면 "_IcaCopyDataToUserBuffer"를 호출하는 함수와 "_MCSPortData" 함수를 찾을 수 있는데 전자의 함수는 아래의 일정량의 데이터를 EAX 레지스터로 전송할 수 있게 한다.

```
kd> r
eax=48474645 ebx=018d3054 ecx=0000000c edx=00000080 esi=018d3050 edi=018d30c4
eip=728f8758 esp=0188fdc4 ebp=0188fdd0 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
rdpwsx!MCSPortData+0x29:
001b:728f8758 83e800          sub     eax,0
kd> db edi
018d30c4  41 42 43 44 45 46 47 48-49 4a 4b 4c 01 00 54 00  ABCDEFGHIJKL..T.
018d30d4  22 00 00 00 03 00 00 00-00 00 00 00 01 00 00 00  "...............
```

<그림 9 – 전송 문자열의 일부가 EAX에 할당>

후자의 함수는 CPU의 EAX 레지스터값이 2일 경우 "_MCSChannelClose" 함수를 호출하며 해당 채널 객체를 비할당 한다. Microsoft에서 배포한 RDP 패킷 매뉴얼과 위에 HGFE 문자열이 할당된 것을 토대로 다음과 같은 패킷을 전송하면 시스템이 크래시 된다.

#EAX에 2를 호출하여 채널 비할당
"\x00\x00\x00\x00\x02\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00"

#모든 채널 비할당 요청

#TPKT Header
"\x03\x00\x00\x09"

 # PER encoded PDU
"\x02\xf0\x80"

 # shutdownRequestPduData
"\x21\x80"

<표 1 – 시스템 크래시 RDP 패킷>

위 패킷에서 "모든 채널 비할당 요청" 대신 다른 요청을 하여 관리자 권한을 탈취할 수도 있다.

IV. 결론

본 연구에서는 임베디드 시스템의 네트워크에 대한 여러 가지 취약점을 분석하고 ARP Spoofing 공격 기법과 RDP Bluekeep 취약점을 이용하여 가상 머신에 대한 모의 해킹을 하였다. 이를 통해서 위험성을 검증하였으며 통합 보안관리 솔루션의 필요성을 시사하였다.

ARP Spoofing의 경우 네트워크 스위치별로 MAC 주소를 동적 형태가 되어 실시간으로 Gateway를 찾는 방식이 아닌 고정 형태로 설정이 되어야 하고 스파이웨어 같은 경우 관리자에 의해서 검증이 된 프로그램만 설치해야 한다. RDP 취약점의 경우에는 RDP 프로토콜을 최대한 내부에서만 사용해야 하고 사용이 불가피할 때에는 고급 수준의 네트워크 인증 절차를 도입함과 더불어 RDP 의 개발사인 Microsoft 사에서 정기적으로 긴급하게 배포되는 보안 패치를 신속하게 설치하는 것, 그리고 Snort와 같은 네트워크 침입 감지 및 방지 시스템을 통해 "MS_T120" 문자열을 필터링함으로써 문제를 방지할 수 있다.

이를 기반으로 통합적으로 관리할 수 있는 NAC 솔루션이 충분히 개발될 수 있을 것이며 상용화가 되면 효용성이 뛰어나 이익창출의 가능성이 클 것으로 예상된다.

향후 연구과제로는 Snapshot 형식으로 배포할 수 있어 관리가 쉬운 클라우드 컴퓨팅이라고 할 수 있다. 그래서 클라우드 컴퓨팅을 상용화하는데 비용을 절감하는 방법과 빠른 속도를 유지하여 효율성을 높일 수 있는 프로토콜과 임베디드 시스템용 Zero Client(클라우드 서버 접속 전용 운영체게 탑재 기기)에 대한 연구를 이어 나가야 할 것이다.

[참고문헌]

[1] 황현욱, 박은신, 박종백 (2001). ARP Spoofing 공격과 대응방법에 관한 연구. 한국통신학회 학술대회논문집, (), 1821-1824.

[2] "CVE-2019-0708", 이글루 시큐리티 블로그, 2019년 7월 3일 수정, 2019년10월23일 접속, http://www.igloosec.co.kr/BLOG_BlueKeep%20(CVE-2019-0708)

[3] "CVE-2019-0708 Proof of Concept", GitHub, 2019년 5월 31일 수정, 2019년11월16일 접속, https://github.com/n1xbyte/CVE-2019-0708

허니팟을 활용한 캠퍼스 네트워크 공격 유형 분석1)

오인영*, 정지호*, 김정태**, 강구홍*

*서원대학교 정보통신공학과

**한국전자통신연구원 정보보호연구본부

Attack Pattern Analysis on Campus Network Using Honeypots

In-Young Oh*, Ji-Ho Jeong*, JungTae Kim**, Koohong Kang*

*Dept. of Information and Communications Eng., Seowon University

**Electronics and Telecommunications Research Institute, Network & System Security Research Section

요 약

다양한 컴퓨팅 자원을 보유하고 있는 캠퍼스 네트워크는 많은 해커들의 주요 공격 목표가 되고 있다. 본 논문에서는 DMZ 영역과 내부 네트워크에 각각 2개의 허니팟을 구성하고 이들 허니팟에서 수집된 이벤트를 분석하여 오늘날 캠퍼스 네트워크를 대상으로 이루어지고 있는 공격 유형을 분석한다. 공격 유형은 크게 무작위 공격(brute force attack)과 응용계층의 취약점 공격으로 나누어 분석하였으며 SQL Injection 공격에 대한 세부 분석 결과를 다루었다. 특히, 내·외부 두 개의 허니팟의 이벤트를 상호 비교 분석함으로써 방화벽의 효율성을 일정 수준 파악할 수 있다. 본 연구에서는 방화벽은 대부분의 무작의 공격은 잘 방어하는 반면, 응용계층의 취약점 공격에 대한 대응은 다소 부족한 것으로 조사되었다.

I. 서론

캠퍼스 네트워크에는 대학이 운영하는 학사 관리 시스템, 웹서버, 그리고 교육에 필요한 다양한 컴퓨터 등을 포함하는 매우 다양한 컴퓨팅 자원들이 있다. 뿐만 아니라, 보안에 상대적으로 취약한 학생들이 사용하는 다양한 종류의 컴퓨터들이 있어 캠퍼스 네트워크는 항상 해커들의 주요 공격 목표가 되고 있다.

본 논문에서는 캠퍼스 네트워크의 방화벽을 기준으로 DMZ 영역과 내부 네트워크에 각각 2개의 허니팟을 구성하고 SNORT를 활용하여 10일간에 걸쳐 수집된 공격 이벤트를 분석함으로써 오늘날 캠퍼스 네트워크를 대상으로 이루어지고 있는 다양한 공격 유형을 분석한다. 공격 유형은 크게 무작위 공격(brute force attack)과 응용계층의 취약점을 이용한 공격으로 나누어 분석하였다. 또한 응용계층 취약점 공격 중에서 가장 흥미로운 SQL Injection 공격에 대한 세부 분석 결과를 제시하였다. 한편, 외부와 내부에 설치된 두 개의 허니팟에서 수집된 이벤트를 상호 분석함으로써 캠퍼스 네트워크에 사용되고 있는 방화벽의 실제적인 효율성을 분석하였다.

II.관련 연구

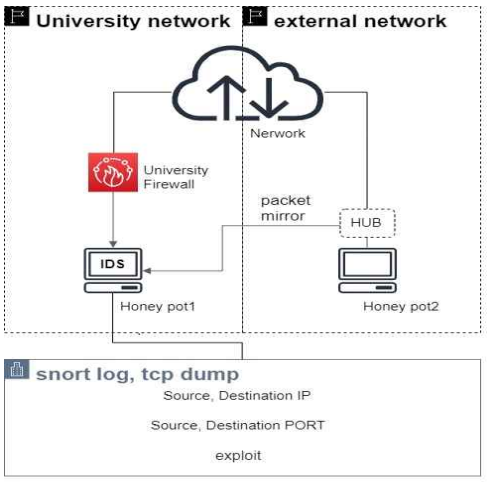
Snort는 실시간 트래픽 분석과 패킷 로깅을 수행하는 오픈소스 침입탐지 시스템이다. 2019년 현재 sectools.org의 125개의 보안 시스템 순위에서 5위의 랭킹을 유지하고 있다. Snort의 특징은 사용자들의 커뮤니티를 통해 지속적으로 탐지 Rule이 제공되어 지속적으로 업데이트

1) 본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발).

되며, 또한 사용자가 직접 로컬 Rule을 설정하여 사용할 수 있다. Snort는 이렇게 제작된 Rule을 기반으로 패킷을 검사하여 Rule의 조건과 일치하는 경우 알람을 발생시키는 시스템이다.

III. 허니팟 구성

공격 트래픽 검출을 위한 허니팟 구성은 방화벽을 통과하기 전 DMZ에 위치한 외부 허니팟과 내부 네트워크 내에 위치한 내부 허니팟으로 구성하였다. 이들 각각의 허니팟은 두 개의 머신으로 동작하며 운영체제는 Ubuntu 16.04 와 window server 2008을 사용하였고 HTTP, mySQL, SSH, telnet 등의 응용프로그램을 설치하여 무작위 공격(brute force attacks)이나 취약점 공격의 목표가 될 수 있도록 하였다. <그림 1>은 허니팟의 전체 구성도이다. 허니팟1에 IDS가 설치된 모니터링 서버를 구성하고 허니팟2에는 미러링 장비를 설치하였다. 미러링 장비를 통해 모니터링 서버는 양쪽의 패킷을 모두 수집할 수 있도록 구성 하였다. 모니터링 서버는 Ubuntu로 사용했으며 IDS인 Snort와 Tcpdump를 사용하여 이벤트 로그와 패킷로그를 수집할 수 있도록 하였다.



<그림 1> 허니팟 구성도

snort와 tcpdump로 수집된 로그는 모니터링 서버에 저장되며 snort는 사용자가 보기 쉽도록 Snort 관련 프로그램 중 하나인 Barnyard2[1]를 사용하여 mySQL에 기록된다. Tcpdump 로그는 자세한 분석을 위해 wireshark 프로그램을 사용하였다.

IV. 기록 분석

본 장에서는 2019년 9월 27일부터 10월 12일 사이에 10일간에 걸쳐서 방화벽 내. 외부 허니팟에서 수집된 다양한 이벤트에 대한 분석 결과를 설명한다.

4.1 이벤트 횟수 분석

본 논문에서는 snort에서 제공하는 기본 룰(rule) 셋을 사용하고 자체 제작 룰을 배제하였다. Snort 버전 2.9.9.0에서 제공하는 룰 수는 약 3만개 정도이고 허니팟에 발생된 알림에 직접 적용된 룰 수는 380개이다. 이 경고들을 크게 무작위 공격과 응용프로그램 취약점 공격으로 분류하여 분석하였다. 먼저, 무작위 공격은 다음과 같이 두 가지 접근방식으로 세분화 하였다.

- 1대N 공격
  무작위 공격은 하나의 외부 IP에서 허니팟의 N개의 포트에 접근하는 경우이다.
- N대1 공격
  N개의 외부 IP에서 허니팟의 하나의 포트에 반복적으로 접근하는 방식이다.

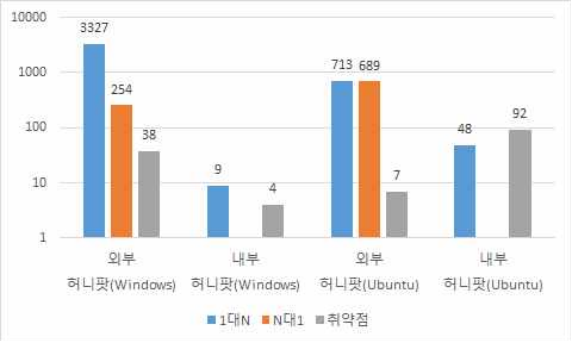
한편, 취약점 공격은 허니팟에 설치된 다양한 응용프로그램 및 운영체제의 취약점을 노린 공격이다. <표 1>은 허니팟 머신별 수집위치와 수집기간을 요약해 놓았다.

<표 1> 허니팟별 수집 위치

수집 날짜	윈도우	우분투
9월 27-30	외부	내부
10월 4-7	외부	내부
11월 11-12	내부	외부

<그림 2>는 무작위 공격(1대N 및 N대1 공격)과 취약점 공격에 따른 이벤트 횟수를 허니팟의 위치와 운영체제별로 구분하여 보여준다. 그림에서 보듯이, 외부허니팟과 내부허니팟의

전체 이벤트 수를 비교하면 운영체제의 종류에 상관없이 공통적으로 내부 허니팟의 경우 이벤트 수가 급감한 것을 볼 수 있다. 학내 방화벽은 무작위 공격에 대해서는 매우 높은 차단 효과를 보여주지만 방화벽에 보호되는 내부 허니팟에서 소량의 무작위 공격 이벤트가 여전히 발생하는 것으로 확인되었다. 즉 학내 방화벽이 무작위 공격 가능성이 있는 패킷을 100% 막지 못한다는 것을 알 수 있다. 한편, <그림 2>에서 취약점 공격의 이벤트 수는 무작위 공격과는 다르게 허니팟의 위치에 상관없이 비슷한 수준의 이벤트가 발생한 것을 볼 수 있다. 즉 학내 방화벽은 응용프로그램 취약점을 노린 패킷의 경우에는 적극적인 대응이 부족한 것으로 조사되었다.



<그림 2> 허니팟에 검출된 이벤트 수

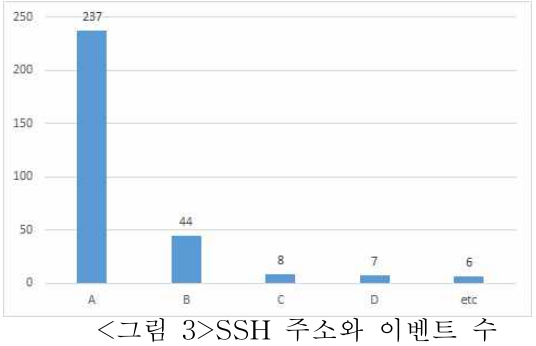
4.2 무작위 공격 패턴

본 절에서는 전체 이벤트 중에서 20% 이상을 차지하는 무작위 공격의 공격 대상(응용)들을 분석한다. 본 논문에서 사용하는 이벤트의 룰은 동일한 이벤트가 1분에 5회 이상 탐지될 경우 로그파일에 기록되기 때문에 실제 공격 횟수는 본 연구에서 보여주는 이벤트 수 보다 많다. <표 2>는 무작위 공격 목표의 포트와 그 포트에 발생한 이벤트 로그 수이다. MSSQL, SSH, mySQL순으로 많은 접근 이벤트가 발생했으며 기타 포트에 약간의 접근 기록이 남았다.

<그림 3>은 SSH를 노린 외부 IP 주소 (top 4)의 개수와 해당 IP의 이벤트수를 보여준다. IP 주소는 보안상의 이유로 <그림 3>에서 보듯이 알파벳으로 대체하였으며 etc는 이벤트 로그가 5회 이하인 IP들의 집합이다. <표 3>은 IP 주소별 접근정보를 정리한 표이다. 분당 패킷수는 패킷수÷지속성으로 외부에서의 공격 간격을 알 수 있다. 가장 많은 공격 시도를 한 A는 중국의 IP로 공격 지속성은 10월 12일 1시 45분부터 10시 13분까지 약 9시간에 걸쳐서 98068회의 패킷이동이 발생하였다.

<표 2> brute force 공격 목표

응용프로그램	포트	이벤트 수
SSH	22	302
MSSQL	1433	424
mySQL	3306	156
그외		41



<그림 3>SSH 주소와 이벤트 수

<표 3> SSH 접근 외부 IP별 공격 정보

국가	A	B	C	D
이벤트	237	44	8	7
지속성	8.5H	1.5H	0.3H	0.3H
패킷수	98068	17095	3009	2831
분당패킷수	192.2	189.9	167.1	157.2

4.3 취약점 공격 패턴

본 절에서는 학내 방화벽이 감지하지 못한 취약점 부분들을 분석한다. <그림 4>는 무작위 공격을 제외한 공격 패턴을 분석한 차트이다. <그림 4>에서 보듯이 가장 많은 공격 유형으로 원격 코드 실행이 발생하였다(42.77%). 원격 코드 실행은 컴퓨터 응용프로그램의 취약점이나 IoT, 라우터 장비 등의 취약점의 가능성이 있는 장비들을 노리는 공격이다. 두 번째로 많은 이벤트는(25.16%) 취약점 정보 수집으로 허니팟에 구현해둔 아파치의 phpinfo에 접속 시도를 해보거나 인터넷 전화를 해킹하는 Sipvicious의 스캔시도가 감지되었다. 이외에도 널리 알려진

SQL Injection, 멜웨어, DOS 순으로 감지되어 로그에 기록되었다.



<그림4> brute force를 제외한 침입 방식

4.4 SQL Injection 패킷 분석

SQL Injection 공격은 개인정보에 대한 중요성 관점에서 가장 치명적인 공격으로 분류된다. 본 연구에서 수집된 SQL Injection 이벤트 패킷 중 Referer 내용을 <그림 5>와 같다. 이 패킷은 SQL Injection을 통한 원격코드실행이 목표인 패킷이다. <그림 5>의 (1)은 문자열을 자르기 위한 해시 값으로, 이후 응용프로그램의 취약점 함수를 호출하기 위해 사용된다. <그림 5>의 (2)는 php 배열로 id, name 배열은 임의의 값을 채우고 num 배열에 SELECT 구문을 사용하여 SQL injection을 실행한다. select 이후 9개의 컬럼값이 있으며 2번째와 9번째 이외에는 컬럼을 채우기 위한 무의미한 값이다. 2번째 컬럼인 0x2720756e696f6e2f2a의 문자열은 'union/*으로 주석처리를 위한 값이다. 9번째 컬럼값인 0x7b24…는{${$RCE'];die/**/(md5(DIRECTORY_SEPARATOR));//}} 문자열로 에러를 반환하는 die함수를 실행하는 컬럼이고 DIRECTORY_SEPARATOR 값을 반환한다. 해당 응용프로그램이 설치되어 있다면 웹쉘까지 접속 가능한 취약점이지만 공격자는 사전 조사를 위해 스캐닝 가능한 코드를 넣은 것으로 보인다.

V.결론

본 논문에서는 허니팟을 구성 하여 현재 캠퍼스 네트워크를 대상으로 일어나는 공격 유형들을 분석하였다. 특히 내·외부로 구분된 허니팟을 구성함으로써 방화벽으로 차단되는 공격 유형을 포함한 모든 이벤트를 snort를 활용하여 분석하였다. 방화벽은 응용계층의 취약점을 활용한 외부 공격에 대해서는 적극적인 차단 효과가 다소 부족한 것으로 조사되었다. 이에 대한 대응방안으로 2차 분석 툴 활용 및 응용계층 취약점 분석 세분화를 통해 IDS/IPS 보안장비에 적용할 예정이다. 한편, 다양한 IoT 장비들을 노리는 원격 코드 실행 공격이 대량 검출되어, 이에 대한 후속 연구가 이루어져야 할 것이다.

IP_src w.x.y.z	src_port 10418
IP_dst a.b.c.d	dst_port 80
룰명 : ET WEB_SPECIFIC_APPS ECSHOP user.php SQL INJECTION via Referer 패킷 info : POST /USER.PHP HTTP/1.1 Referer: 45ea207d7a2b68c49582d2d22adf953a ads (1) |a:3:{s:3:"num";s:147:"*/select 1,0x2720756e696f6e2f2a, 3,4,5,6,7,8,0x7b247b24524345275d3b6469652f2a2a2f286d6435284449524543544f52595f534550415241544f5229293b2f2f7d7d,0--";s:2:"id";s:9:"'union/*";s:4:"name";s:3:"ads";} (2) 45ea207d7a2b68c49582d2d22adf953a (1)	

<그림5>SQL injection 패킷정보

[참고문헌]

[1] Noah Dietrich, Snort 2.9.9.x on Ubuntu 14 and 16 with Barnyard2, PulledPork, and BASE, January 8, 2017, available at https://sublimerobots.com.

[2] I. Mokube and M. Adams, "Honeypots: concepts, approaches, and challenges," Proceeding of the 45th annual southeast regional conference, pp.321-326, 2007.

[3] A. Patel, Q.S. Qassim, and C. Wills, "Survey of intrusion detection and prevention systems," Information Management & Computer Security, vol.18, no.4, pp.277-290, Oct. 2010.

시스코 IOS FunctionTrace 알고리즘에 관한 연구

조현우* 이소연* 이예은* 안건희* 박성진* 박광호* 김경곤* 신정훈*

*KITRI Best of the Best 8기

*고려대학교 정보보호융합학과 산학협력중점 교수

*Theori

A Study on Cisco IOS FunctionTrace Algorithm

Hyeon-Woo Cho* So-Yeon Lee* Ye-Eun Lee* Gun-Hee Ahn*
Seong-Jin Park* Kwang-Ho Park* Kyoung-Gon Kim* Jung-Hoon Shin*

*KITRI Best of the Best 8th

*Division of information security, Korea university

*Theori

요 약

시스코 IOS(Internetwork Operating System)는 단일형 운영체제로, 소프트웨어 분석에 있어서 함수를 처음부터 추적하여서 취약점 분석에서는 패치된 분석을 발견하기에는 많은 어려움이 존재한다. 따라서, 본 논문에서는 기존 리눅스에서는 함수추적 기능(Ftrace)이 존재하며 이를 착안하여 시스코 IOS에서의 함수추적에 용이한 Function Trace 알고리즘을 제안하고 이에 대한 타당성에 대해 기술한다.

Ⅰ. 서론

IOS(Internetwork Operating System)는 시스코사의 라우터 및 스위치에서 사용하는 독자 개발한 운영체제이다. 시스코사의 장비에서는 IOS 및 IOS XE 등의 운영체제를 사용하며, 리눅스 커널 위에서 동작하는 시스코 IOS XE와는 다르게 하드웨어에서 동작하는 단일 운영체제 형태의 시스코 IOS는 흔히 사용하는 분석 도구를 이용하는 것이 어렵다.

소프트웨어 분석은 소스 코드를 직접 분석하는 것이 가장 효율적이다. 하지만, 약 40만개 함수가 내장된 단일형 운영체제인 시스코 IOS의 소스 코드를 직접 분석하는 것은 불가능에 가깝다. 그리고 메모리 할당 부분 또는 I/O가 일어나는 과정과 같이 특정 기능을 실행하는 부분을 추적하는 것에도 어려움이 있다.[1] 또한, 시스코 IOS 취약점 분석을 하기 위해서는 Binary Diffing 도구를 통해 상이한 버전의 시스코 IOS를 패치 분석 시도 시, 수십만 개의 함수로 이루어진 시스코 IOS의 특성상 패치된 부분의 위치를 찾는 것은 어렵다. 따라서, 본 연구에서는 리눅스의 커널의 내부 동작을 추적 및 디버깅하기 위한 Ftrace 기능을 보고 착안하여, 시스코 IOS의 특정 기능의 함수 추적을 위한 MIPS 아키텍쳐 기반의 FunctionTrace의 알고리즘에 대해 제안한다.[2][5]

본 논문에서는 다음과 같은 내용을 기술한다. 2장에서는 리눅스의 함수 추적기인 Ftrace와 시스코 IOS FunctionTrace의 알고리즘에서 사용하는 DBI 및 관련 연구에 대해 서술하고, 3장에서는 시스코 IOS 함수 추적을 위한 알고리즘에 대해서 제안하며, 4장에서는 제안한 알고리즘에 대한 정당성을 증명한다. 마지막 5장에서는 연구의 시사점과 및 향후 연구 방향에 관해 기술한다.

Ⅱ. 관련연구

2.1 리눅스 Ftrace

Ftrace는 리눅스 커널 2.6.28 버전부터 제공하는 함수 추적기이며, 커널이 호출하는 함수를 추적할 수 있는 기능이다. Ftrace는 인터럽트, 스케줄링, 커널 타이머 커널 동작을 상세히 추적해준다. 또한, 함수를 어느 프로세스가 실행하는지, 함수 실행 시각을 알 수 있으며, Ftrace 로그를 분석하

면 시스템 동작에 부하를 주지 않는 특징이 있어 소프트웨어 및 취약점 분석에 있어서 유용한 도구이다.[3] 리눅스 커널은 오픈소스이기 때문에 코드를 삽입하는 데 어려움이 없다. 하지만 시스코 IOS는 이미 컴파일된 바이너리 파일이기 때문에 리눅스와 같이 기존 코드에 추가적인 삽입이 불가능하다. 따라서, 3장부터는 기존 코드의 실행 흐름을 해치지 않으며 시스코 IOS 함수 분석에 용이한 Ftrace 알고리즘을 제안한다.

2.2 DBI

Instrumentation 기법 중 하나로 사용되고 있는 것이 DBI(Dynamic Binary Instrumentation)로 불리는 동적 기법이 있다. DBI 는 런타임에서 실행 코드를 삽입하여 바이너리 응용 프로그램의 동작을 분석하는 방법이다. 또한, 동적 기법들은 바이너리의 실행 중에 메모리상에서 계측하고자 하는 위치에 동적으로 생성된 Instrumentation 코드를 삽입하거나, 계측하고자 하는 코드 영역을 Instrumentation 코드가 있는 영역으로 복사하고 제어 흐름만을 이동시켜 계측하는 방법을 사용한다. DBI는 실행 중 계측하고자 하는 위치의 제어 흐름만을 우회시키거나 JIT 컴파일 시점에 Instrumentation 코드를 삽입하는 방식을 사용한다.[4] 시스코 IOS는 코드 영역에 쓰기 권한이 부여되어 DBI 코드 삽입이 가능하다. 따라서, 시스코 디버그 상태의 RSP(Remote Serial Protocol)를 이용해 코드 패치가 가능하여 DBI 방식을 사용하였다.

Ⅲ. FunctionTracer 알고리즘

시스코 IOS FunctionTrace는 코드 영역 하단의 Nop Padding 영역을 이용한 DBI 방식을 차용하여 설계하였다. Instrumentation 코드를 실행하게 되면, 삽입한 DBI 코드가 실행되며, 이때 추적할 함수의 조건이 존재한다.

```
addiu $sp, $sp, 0x??
(lw|sw) $??, 0x??($sp)
(lw|sw) $??, 0x??($sp)
```

그림 1. 추적 함수 조건

1. 일반적인 Stack Prologue(addiu $sp, $sp, -0x??)로 시작해야 한다.
2. $sp 레지스터의 값을 수정해 스택 공간을 만든 뒤 실행되는 2개의 Instruction이 sw 또는 lw이며, 스택에 값을 보관 또는 읽어야 한다.

해당 패턴에서 각각의 함수마다 다른 값은 스택의 크기, 레지스터의 종류, 스택의 오프셋, Instruction이 있다. 해당 조건을 만족해야 하는 이유는 3.6장에서 설명한다.

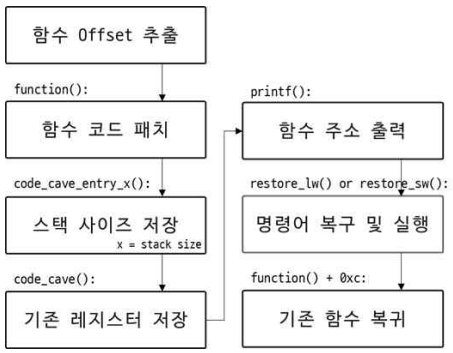


그림 2. FunctionTrace 구조도

그림 2는 FunctionTrace 대략적인 구조도이며, 3.1장부터는 해당 구조에 대한 프로세스와 FunctionTrace 알고리즘에 대해 제안한다.

3.1 함수 주소 Offset 및 명령어 추출

DBI 코드를 삽입하기 전, IDA에서 추적하고자 하는 함수 주소의 Offset 및 명령어를 추출한다. 시스코 IOS 자체 메모리 보호 기법인 주솟값의 랜덤화로 인해 함수 주솟값이 아닌 Offset으로 저장한다. 코드 패치 이후 함수의 기존 명령어를 복구하기 때문에 명령어를 저장해야 한다. 하지만, 기존 어셈블리 공간은 한정되어있기 때문에 명령어와 오프셋을 하나의 레지스터에 저장하여 code cave 함수에서 명령어를 복구한다. 해당 함수를 추출하는 단계는 다음과 같다.

1. 바이너리에서 해당 프로토콜에 대한 문자열을 추출한다. (함수에 심볼이 존재하지 않아 문자열로 추측)
2. 해당 문자열을 사용하는 함수를 추출한다.
3. Depth 값에 따라 해당 함수가 호출하는 함수를 추출한다. (타겟 프로토콜을 처리하는 함수 추출 목적)
4. 추출한 함수에서 베이스 주솟값을 계산한다. 그 후 함수 Offset, 스택 크기, 첫 번째 명령어 레지스터, 첫 번째 명령어 스택 Offset 과 첫 번째 명령어, 두 번째 명령어 레지스터, 두 번째 명령어 스택 Offset과 두 번째 명령어 순서로 function list에 저장한다.

명령어를 복구하는 방식은 3.6장에서 서술한다.

3.2 함수 코드 패치

시스코 IOS는 코드 영역에 쓰기 권한이 부여되어 DBI 코드 삽입이 가능하다. 따라서, 시스코 디버그 상태의 RSP(Remote Serial Protocol)를 이용하여 코드 패치가 가능하다. 시스코 IOS의 빈 메모리 영역에 code_cave_entry, code_cave, restore lw or sw 함수와 offset_list, reg_stack _offset_list 데이터를 삽입한다.

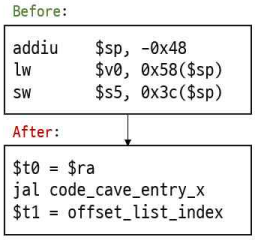


그림 3. DBI 코드 패치 전후

추적할 함수는 다음과 같이 패치한다. 우선 t0 레지스터에 기존 함수의 복귀 주소를 저장한다. t1 레지스터에는 첫 번째 명령어 레지스터, 첫 번째 명령어 스택 Offset, 두 번째 명령어 레지스터, 두 번째 명령어 스택 Offset에 따라 offset_list_index를 저장한다. t0, t1 레지스터를 사용하는 이유는 t레지스터는 임시 레지스터로 함수의 지역변수 저장 시에 사용하여, 해당 레지스터를 수정하여도 기존 코드 실행에 영향을 주지 않기 때문이다. 그 후 jal을 이용해 code_cave_entry 함수를 호출한다. 이때, x는 해당 함수의 스택 크기이다. MIPS는 RISC 아키텍처로, 파이프 라인을 사용하기 때문에 jal로 함수를 호출하기 전 아래의 명령어도 실행한다. 따라서 코드 패치도 파이프라인을 고려하여 설계하였다.

3.3 스택 크기 저장

code_cave_entry함수는 t5 레지스터에 해당 함수의 스택 크기를 저장하고 code_cave 함수를 호출한다.

3.4 기존 레지스터 저장

code_cave 함수를 호출하면 기존의 코드 흐름을 방해하지 않기 위해 사용하던 레지스터를 모두 heap memory의 하단에 저장한다.

3.5 함수 주소 출력

code_cave 함수에서 레지스터를 저장한 뒤 해당 함수의 주소를 출력하기 위해 인자 값을 저장하는 데 쓰이는 레지스터인 a0, a1 레지스터를 사용한다. a0 레지스터에는 printf의 첫 번째 인자인 “[DEBUG] sub_%08x called\n”, a1 레지스터에는 ra 레지스터를 이용하여 해당 함수의 주솟값을 넣어준 뒤 printf 함수를 호출한다.

3.6 명령어 복구 및 실행

code_cave에서 함수 주소를 출력한 뒤, 다시 기존 함수의 명령어를 실행해야 한다. 이는 3.2장에서 저장한 t1 레지스터를 이용하여 삽입한 offset_list 배열에서 값을 읽은 뒤 t1 레지스터에 저장하는 방식으로 실행된다.
t1 레지스터의 상위 2바이트는 첫 번째 명령어를, 하위 2바이트는 두 번째 명령어 저장 시에 사용한다. 저장한 값은 그림4와 같이 파싱한다.

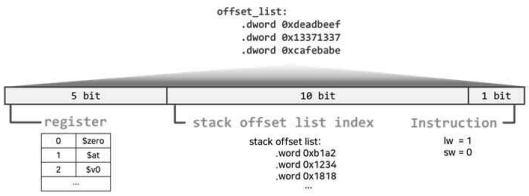


그림 4. Offset 구조

읽어온 t1 레지스터는 상위 2바이트는 첫 번째 명령어를, 하위 2바이트는 두 번째 명령어를 저장하고 있다. t2 레지스터에는 첫 번째 명령어의 스택 오프셋 인덱스, t3 레지스터에는 두 번째 명령어 레지스터, t4 에는 두 번째 명령어 스택 오프셋 인덱스 그리고, t1 레지스터에는 첫 번째 명령어의 레지스터를 저장한다. 이후 t2 레지스터의 하위 1비트를 이용하여 ‘1’일 시에 restore_lw 함수, ‘0’일 시 restore_sw 함수를 호출한다. restore_lw 와 restore_sw 함수는 다음과 같은 switch case 구문으로 구성되어 있다.

```
restore_sw:
    $t2 = reg_stack_offset_list[$t2]
    $t2 = $t2 + $sp
    switch($t1)
        case 0:
            sw $zero, ($t2)
            break
        ……
        case 31:
            sw $ra, ($t2)
            break
    return
```

그림 5. restore_sw 함수 의사코드

reg_stack_offset_list에서 t2 레지스터를 이용해 기존 함수의 실제 스택 오프셋을 불러온다. 그리고 t1의 값에 따라 명령어에 사용되는 레지스터를 선택하여 첫 번째 명령어 코드를 실행한다. 다시 code_cave 함수로 돌아와 두 번째 명령어도

같은 방식으로 실행한다. 이후 3.3장에서 저장한 t5 레지스터를 이용해 스택 크기를 할당해준다.

3.7 기존 함수 복귀

Function Tracing을 끝낸 code_cave 함수는 기존 ra 레지스터를 이용해 함수로 돌아간다. 이때 3.2장에서 저장한 복귀 주소를 ra 레지스터에 저장하여 코드 실행 흐름에 영향을 끼치지 않도록 한다.

Ⅳ. 결과 분석

3장에서 제안한 시스코 IOS 함수추적 알고리즘을 이용하여 FunctionTrace를 구현하였다.

```
[*] write code_cave
[*] write offset_list
[*] write reg_stack_offset_list
[*] write fmt_str
[DEBUG] Below addresses are based on IDA
[*] Tracing 501 functions....
[DEBUG] tracing sub_30a7c800()...sub_30a70580() on IDA
[DEBUG] tracing sub_31ae0598()...sub_31ad4318() on IDA
[DEBUG] tracing sub_32204804()...sub_321f8584() on IDA
```

그림 6. FunctionTrace 실행 결과

FunctionTrace를 실행 결과로 코드가 삽입된 주소, 메모리 영역, 추적할 함수들이 출력된다. 이를 그림 6을 통해 결과 값을 확인할 수 있다.

4.1 SNMP 프로토콜 추적을 통한 타당성 검증

```
[DEBUG] sub_43F10250() called
[DEBUG] sub_43F10250() called
[DEBUG] sub_4145A10C() called
[DEBUG] sub_414592FC() called
[DEBUG] sub_4145A10C() called
[DEBUG] sub_414592FC() called
[DEBUG] sub_4145A10C() called
[DEBUG] sub_414592FC() called
[DEBUG] sub_414598B0() called
[DEBUG] sub_42720224() called
[DEBUG] sub_4145A10C() called
[DEBUG] sub_414592FC() called
[DEBUG] sub_414598B0() called
```

그림 7. SNMP 프로토콜 추적 결과

FunctionTrace를 이용해 SNMP 프로토콜 패킷을 추적한 결과 그림7과 같이 함수가 출력되었다. 이는 Time Synchronization이 중요한 네트워크 장비에서도 빠르게 실행되며 실제 코드 흐름에 영향을 주지 않는 것을 알 수 있다.

4.2 코드삽입 실행시간을 통한 실용성 검증

실제 실행은 시스코 라우터 C2911에서 진행하였으며 실제 삽입에 실행되는 시간은 다음과 같다.

표 1. 실행시간 비교

함수 갯수	실행 시간1	실행 시간2	실행 시간3	평균 실행 시간
50	41.894	40.799	40.493	41.062
100	47.368	47.564	47.168	47.366
500	105.496	104.968	104.906	105.123
1000	183.889	183.377	183.159	183.475

단위 : 초

C2911에서 평균 실행시간은 100개의 함수일 때는 약 47초, 1000개의 함수일 때 183초가 소모되었으며, 이는 평균적으로 함수를 추적하는 것에 적절한 시간이다. 이를 통해, 해당 알고리즘의 정당성을 증명한다.

Ⅴ. 결론 및 향후 연구

단일형 운영체제인 시스코 IOS에 존재하는 함수들을 일견하게 찾아 분석하는 것은 불가능한 일이다. 하지만, 리눅스의 함수 추적기와 같은 기능이 존재한다면, 시스코 IOS 취약점 분석 및 다양한 연구에서 많은 활용이 될 수 있을 것이다. 따라서, 본 연구에서는 40만 개가 넘는 시스코 IOS의 함수들의 종속관계를 파악할 수 있는 FunctionTrace의 알고리즘을 제안하고, FunctionTrace를 개발하여 결과를 통해 해당 알고리즘의 정당성을 증명하였다.

또한, 해당 방식은 시스코 IOS에 국한된 것이 아닌 다른 OS와 아키텍처에도 사용될 수 있어 추후 다양한 연구들이 진행될 것으로 예상된다. 본 연구는 FunctionTrace 알고리즘을 통해 실제 시스코 IOS에서의 분석 대상 함수를 추적하여 Smart Fuzzing과 같은 취약점 분석 또는 연구에 이용될 수 있다.

[참고문헌]

[1] 김진우, Ftrace로 커널 읽기, https://brunch.co.kr/@alden/24#comment, last accessed 2019/11/07

[2] Jinb-park, [Linux kernel Trace]01_1-ftrace 사용기,http://blog.daum.net/tlos6733/127, last accessed 2019/11/07

[3] AustinKim, ftrace 설정과 로그 분석하기, https://kldp.org/node/161282, last accessed 2019/11/08

[4] 이민수, 이제현, 김호빈, 류찬호, “바이너리 분석 도구 효율성 평가를 위한 Instrumenta tion 성능 측정기법”, 정보보호학회논문집 27, pp. 1331-1345, Dec. 2017

[5] Steven Rostedt, Ftrace linux Kernel Tracing, in Proc. of Linuxcon, Aug. 2010

머신러닝 프라이버시에 관한 연구 : FEDERATED LEARNING

아피파툴 무카로*, 푸트리 라마와티*, 김호원*

*부산대학교 전기전자컴퓨터공학과

{afifatul.mukaroh, putrirhmwati, howonkim}@gmail.com

STUDY ON PRIVACY PRESERVING MACHINE LEARNING : FEDERATED LEARNING

Afifatul Mukaroh*, Putri Rahmawati*, Howon Kim*

*Department of Electrical and Computer Engineering,

Pusan National University

Abstract

Privacy of individuals is important because it maintains people autonomy and individuality. In order to preserve privacy, data related personal information should be protected. Data itself is the main resource in Machine Learning area. The more complete and valid the data, the more powerful the machine learning performance. But as more as the data is exploited, as more as the privacy of an individual is prone to be violated. Federated Learning is one of technique in privacy preserving machine learning. This paper explains about Federated Learning, the method, the development, and the challenges.

Keywords : Privacy Preserving, Machine Learning, Federated Learning

I. Introduction

Privacy of individuals is important because it helps them to maintain their autonomy and individuality [1]. We define individuals based on their personal information. Beside that, privacy also has its functional benefits. For example anonymity and pseudonymity that protects the privacy of people's identity. The other important benefit of privacy or confidentiality is also for identity fraud. Therefore, people have a right to decide how their personal information moves in the economy or society.

European Union (EU) has a law that concern in privacy or data protection, which is called General Data Protection Regulation (GDPR). It will give fines or penalties to those who violate the privacy and security standards [2]. Because of this law, there are many companies that could be hit by EU.

Data itself is the most important resource for companies that work in Machine Learning area. The more complete and valid the data, the more powerful the machine learning performance. But as more as the data is exploited , as more as the privacy of an individual is prone to be violated. Federated Learning is one of technique that popular in privacy preserving machine learning. This paper explains about Federated Learning, the method, the development, and the challenges.

II. Federated Learning

Federated Learning solve privacy issue in Machine Learning by letting the data remain

local or not being sent to the data center. The learning process is conducted in user device itself. Instead of sending the raw data, model parameter of the learning process is sent to the data center [5].

Define N data owners $F_i, i = 1...N$ who all wish to train a machine learning model by consolidating their respective data D_i. A conventional method is to put all data together and use $D = D_i,\ i = 1 \cdots N$ to train a model $Msum$. However, this solution is not possible to implement due to legal issues such as privacy and data security [3].

To solve this problem, Federated Learning is a learning process in which data owners collaboratively train a model M_{FED} and in the process any data owner F_i does not expose its data D_i. In addition, the performance or accuracy of M_{FED} (denoted as V_{FED}) should be very close to the performance or accuracy of $Msum$, $Vsum$. That is : $|V_{FED} - Vsum| < \delta$, where δ is non-negative real number. Federated learning algorithm has δ-accuracy loss.

Let matrix D_i denotes the data held by each data owner i. Each row of the matrix represents a sample, and each column represents a feature. At the same time, some data sets may also contain label data. Features space is denoted as X, label space as Y and I as sample ID space. So complete training dataset is (I,X,Y).

Federated learning is classified into horizontally federated learning, vertically federated learning and federated transfer learning based on how data is distributed among various parties in the feature and sample ID space [4]. Horizontal Federated Learning is when data sets share the same feature space but different in samples. Participants train independently and share only subsets of updates of parameters. Vertical federated learning is applicable to the cases that two data sets share the same sample ID space but differ in feature space. Federated Transfer Learning is when two data sets differ not only in samples but also in feature space.

III. Federated Learning Method

To perform Horizontal Federated Learning training in Machine Learning, do 4 steps below iteratively until the loss function converges [4]: (i) Participants compute training gradients locally, encrypt the gradient, and send the results to server; (ii) Server aggregate the gradient securely; (iii) Server send back the aggregated gradient to participants; (iv) Participants update their respective model with the decrypted gradients. Figure 1 shows the architecture of Horizontal Federated Learning, where k is total participant with the same data structure.

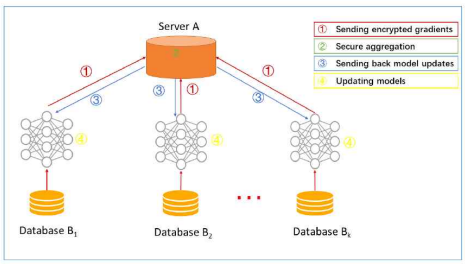


Figure 1. Horizontal Federated Learning Architecture[4]

To perform Vertical Federated Learning training, there are two parts that should be done as shown in Figure 2, [4]. Part a is alignment techniques to confirm the common users of both parties using encryption-based user ID. After alignment is done, the data is used for machine learning training (Part b). In this part b, a collaborator C involves to ensure the confidentiality. A trusted third party C can be government or secure computing node. Part b divided into 4 steps [4] : (i) Collaborator C creates encryption pairs, send public key to A and B; (ii) A and B encrypt and exchange the intermediate

results for gradient and loss calculations (iii) A and B computes encrypted gradients and adds additional mask. B also computes encrypted loss; then send it to C; (iv) C decrypts and send the decrypted gradients and loss back to A and B; A and B unmask the gradients, update the model parameters.

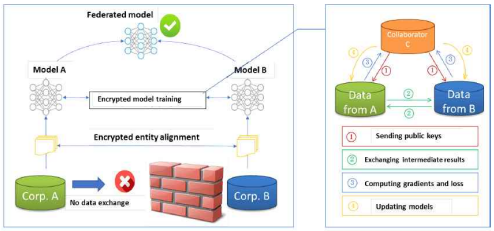


Figure 2. Vertical Federated Learning Architecture [4]

In Federated Transfer Learning, since the samples and feature space are different so the overlapping samples are a very small set. Suppose party A and B has small overlapping set but we are interested in learning the labels for all the data set in party A. Using Federated transfer learning, we can use architecture in Figure 2 but the details of the intermediate results that are exchanged between party A and party B is changed. It only learns a common representation between the features of party A and B, and minimizing the errors in predicting the labels for the target-domain party by leveraging the labels in the source-domain party (B in this case).

IV.Federated Learning Development

Federated Learning has been introduced to the public mainly through Google's blogs. But it also has been developed more by some papers. In 2016 Brendan [9] proposed Model Averaging in Federated Learning to train data in relatively few rounds of communication. The key insight is that despite optimizing the non-convex loss functions, it does parameter averaging over updates from multiple clients. In 2017, Brendan [7] developed Federated Learning again for communication efficiency with iterative model averaging. Jakub [10] in 2017 also proposed two ways to reduce communication cost in Federated Learning. First way is structured updates, where it directly learn an update from a restricted space parametrized using a smaller number of variables. Second way is sketched updates, where it learn a full model update and then compress it using a combination of quantization, random rotations, and subsampling before sending it to the server. Also for communication-efficient federated learning, Yang [8] developed it with Asynchronous Model Update and Temporally Weighted Aggregation in 2019. Not only in communication part, Federated Learning also has many development in aggregation part. In 2016, Keith [12] proposed a secure aggregation protocol to protect the privacy of each user's model gradient. The protocol consist of 4 steps : (i) Dropped User Recovery using Secret Sharing; (ii) Double-Masking to Thwart a Malicious Server; (iii) Exchanging Secrets Efficiently; and (iv) Minimizing Trust in Practice.

For the implementation, Davy [6] has implemented Federated Learning for Intrusion Detection. Andrew [11] implemented it for next-word prediction in a virtual keyboard for smartphones. Konstantin [14] implemented it for Human Activity Recognition. It also has been implemented in Mobile Packet Classification [13].

V. Federated Learning Challenges

Although Federated Learning solves the problem in privacy violation, there are some challenges that come up with it. Tian Li [5] divided Federated Learning challenges into 4 : expensive communication, systems heterogeneity, statistical heterogeneity, and privacy concern.

Expensive communication comes up because of the possibility that massive device in the federated network can be slower than local network. System heterogeneity is related to the storage, computational, connectivity, and other hardware specification of each local device that may differ among other. Some devices may also be unreliable to do local learning. Statistical heterogeneity means that every device may learn or generate model from non-identical data. Privacy concern is related to sensitive information of a user that possibly to be learnt also by federated learning. For example like pin of a user in the context of a next word prediction task.

VI. Conclusion

Federated Learning is a technique in privacy preserving machine learning. In federated learning, instead of sending the raw data for learning, the learning is conducted in local device and the output parameter is sent. Although federated learning can preserve the privacy, there are still some other issues that become challenges. It needs to be developed more to cope these challenges. In the future Work, Federated Learning may be compared with other techniques like Differential Privacy or Secure Multi-Party Computation.

Acknowledgment

This work is supported by Smart City R&D project of the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (MOLIT), Ministry of Science and ICT (MSIT) (Grant 18NSPS-B149388-01)

[References]

[1] Privacilla.org, “Why is Privacy Important?”, Privacy Fundamental, Feb 12 2001, <www.privacilla.org/ fundamentals/ whyprivacy.html>[Accessed at Nov 8 2019].

[2] GDPR.EU, “What is GDPR, the EU’s new data protection law?“,<gdpr.eu/what-is-gdpr/> [Accessed at Nov 9 2019].

[3] WeBank AI Group, Federated Learning White Paper V1.0, 2018.

[4] Q. Yang, Y. Liu, T. Chen, and Y. Tong, Federated Machine Learning: Concept and Applications, ACM Trans. Intell. Syst. Technol., Vol. 10, No. 2, Article 12, 2019

[5] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, Federated Learning: Challenges, Methods, and Future Directions, arXiv:1908.07873v1, 2019

[6] Davy P. , Vera R., Ilias T., Jan S., Wouter J., and Elisabeth I.Z., Chained Anomaly Detection Models for Federated Learning: An Intrusion Detection Case Study, Applied Sciences, 2018.

[7] H. Brendan M., Eider M., Daniel R., Seth H., and Blaise A, A., Communication-Efficient Learning of Deep Networks from Decentralized Data, arXiv:1602.05629v3, 2017.

[8] Yang C., Xiaoyan S., and Yaochu J. Communication-Efficient Federated Deep Learning with Asynchronous Model Update and Temporally Weighted Aggregation, arXiv:1903.07424v1,2019.

[9] H. Brendan M., Eider M., Daniel R., and Blaise A.A., Federated Learning of Deep Networks using Model Averaging, arXiv:1602.05629v1, 2016.

[10] Jakub K., H. Brendan M., Felix X. Y., Ananda T. S., Dave B., and Peter R., Federated Learning: Strategies For Improving Communication Efficiency, arXiv:1610.05492v2, 2017.

[11] Andrew H., Kanishka R., Rajiv M., Swaroop R., Francoise B., Sean A., Hubert E., Chlo´e Kiddon, Daniel Ramage, Federated Learning for Mobile Keyboard Prediction, arXiv:1811.03604v2, 2019.

[12] Keith B., Vladimir I., Ben K., Antonio M., H. Brendan M., Sarvar P., Daniel R., A., and Karn S., Practical Secure Aggregation for Federated Learning on User-Held Data, arXiv:1611.04482v1, 2016.

[13] Evita B., Bálint T., and Athina M., A Federated Learning Approach for Mobile Packet Classification, arXiv:1907.13113v1, 2019.

[14] Konstantin S., Vladimir V., Sarunas G., Human Activity Recognition Using Federated Learning, IEEE Intl Conf on Parallel & Distributed Processing with Applications, 2018.

머신러닝을 활용한 피싱사이트 탐지 방안

김경석*, 김태호*, 박우인*
*수원대학교 정보보호학과

Phishing Site Detection Method Using Machine Learning

Kyoung-Seok Kim*, Tae-Ho Kim*, Woo-In Park*
*Division of information security, Suwon University.

요 약

매년 피싱사이트의 수는 증가하고 있으며, 그로 인한 피해도 비례해 증가하고 있다. 하지만 현재 나와 있는 피싱사이트를 탐지하는 도구들은 리스트 기반의 탐지 방법을 사용하고 있다. 이는 알려진 피싱사이트 탐지에는 효과적이나, 알려지지 않은 피싱사이트의 탐지는 불가능하다. 따라서 본 논문에서는 효과적인 탐지를 위해 알려진 피싱사이트를 탐지하기 위한 리스트 기반의 탐지 방법과, 알려지지 않은 피싱사이트를 탐지하기 위해 머신러닝 모델을 결합한 탐지 방안을 제시한다.

I. 서론

Private data와 Fishing의 합성어인 피싱(Phishing)은 공격자가 이메일과 같은 통신 수단을 활용해 신뢰할 수 있는 사람 또는 기관으로 위장하여 사용자의 개인정보, 금융정보 등을 부정하게 얻으려는 사회공학적 공격 기법이다.

피싱사이트 탐지를 위해 블랙리스트를 사용하는 방법이 보편적으로 사용되고 있다. 그러나 블랙리스트만 사용하는 탐지 방안은 수명이 짧은 피싱사이트를 탐지하는 데 비효율적이며, 알려지지 않은 피싱사이트를 탐지할 수 없다. 본 논문에서는 이러한 점을 보완하기 위해 머신러닝을 활용한 피싱사이트 탐지 방안을 제안한다.

본 논문은 다음과 같이 구성된다. 2장에서는 피싱사이트의 feature를 이용한 피싱사이트 탐지 방안을 제안한다. 3장에서는 제안된 탐지 방안을 활용하여 알려지지 않은 피싱사이트를 탐지한 실험 결과를 분석하고, 4장에서는 결론을 맺는다.

Ⅱ. 탐지 방안

2.1 탐지 전략

피싱사이트 탐지를 위한 다양한 연구로 탐지 기법이 발전하고 있지만, 최근의 피싱사이트는 수명이 짧고 원시사이트를 그대로 참조하거나 원시사이트로 연결된 많은 수의 링크를 가지는 등의 방법으로 탐지를 어렵게 만들고 있다.

이런 피싱사이트를 탐지하는데 기존 방식은 한계가 있기 때문에, 피싱사이트의 feature를 분석하고 머신러닝 알고리즘과 1차적으로 도메인을 검증해줄 수 있는 필터링 리스트를 함께 사용해 최대한 많은 피싱사이트를 탐지하고자 한다.

본 논문에서 제안하는 피싱사이트 탐지 방안은 리스트와 머신러닝 모델을 결합한 방식으로 1차적으로 리스트를 통해 검증하고 2차적으로 머신러닝 모델에서 검증한다.

2.1.1 데이터 수집

피싱사이트를 탐지할 수 있도록 머신러닝 모델과 리스트를 구성하기 위해서는 정상 도메인과 피싱사이트 도메인 데이터를 수집하고 분석할 필요가 있다. 정상 도메인 데이터는 Alexa 2017 상위 8,495개, 피싱 도메인 데이터는 Phishtank에서 Valid된 피싱 도메인 9,141개를 수집하여 분석해 feature로 구성하였다.

2.1.2 리스트 구성

피싱사이트의 탐지율을 높이기 위해 기존의 피싱사이트를 탐지하는 방법 중 하나인 리스트를 활용하여 1차적인 탐지를 수행한다. Alexa 2017 상위 5만 개 데이터를 수집해 리스트로 등록하였고, Phishtank의 Valid된 피싱사이트 약 5만 개를 수집하여 리스트로 등록하였다. 이를 통해 피싱사이트 탐지와 오탐률 개선의 두 가지 효과를 만족시킬 수 있다.

2.1.3 머신러닝 알고리즘 활용

점점 정교해지고 수명이 짧아지는 피싱사이트를 탐지하기 위해 머신러닝 알고리즘을 활용해 탐지하고자 한다. 참고문헌[1]에서 분석된 feature와 새롭게 분석한 feature를 통해 하나의 도메인이 총 18개의 feature 값을 가질 수 있도록 했고, 머신러닝이 충분한 학습을 거쳤을 때 효과적인 탐지를 수행할 수 있도록 진행했다. 각 feature는 2.2에서 설명한다.

피싱사이트의 효과적인 탐지를 위해서는 탐지율이 높게 나올 수 있도록 적절한 머신러닝 알고리즘을 사용해 검증해야 한다. 본 논문에서는 3개의 알고리즘을 테스트한 후, Fig. 1.에서 가장 정확도가 높은 Random Forest 알고리즘을 사용하여 머신러닝 학습을 진행하였다.

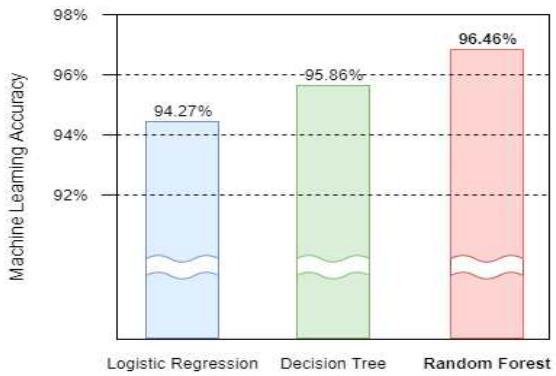


Fig. 1. 머신러닝 알고리즘 테스트

2.1.4 피싱사이트 탐지

1차적으로 필터링 리스트에서 도메인을 검사 후 머신러닝 모델에서 탐지하게 된다. 도메인이 리스트에 존재하지 않는다면 도메인이 생성된 시간이 비교적 최근이라는 뜻이고, 해당 도메인이 분석된 feature로 학습된 머신러닝 모델을 통과하게 되면서 피싱사이트인지 정상사이트인지 판별되는 시스템을 갖춘다.

2.2 피싱사이트 feature

2.2.1 URL 길이

피싱사이트의 URL 길이는 일반적으로 정상사이트에 비해 길다. 전체 데이터셋의 URL 길이를 추출한 결과 정상사이트의 평균 URL 길이보다 피싱사이트의 평균 URL 길이가 긴 것으로 확인되었다.

2.2.2 URL 내에 '-' 문자 포함 여부

일부 피싱사이트의 경우 URL 내에 '-' 문자를 사용하여 사용자로 하여금 정상적인 웹사이트라고 착각하게 한다.

2.2.3 서브 도메인의 개수

피싱사이트들은 일반적으로 여러 개의 서브 도메인을 갖고 있다. 서브 도메인 내에 원시사이트의 도메인을 포함시켜 사용자가 의심하지 않도록 한다.

2.2.4 SSL/TLS 적용 여부

SSL/TLS를 사용하는 경우 URL이 'https://'로 시작한다. SSL/TLS를 사용하는 사이트들은 안전한 사이트일 확률이 높지만, 최근에는 단기간의 인증서를 발급받아 SSL/TLS를 사용하는 피싱사이트들이 증가하고 있다.

2.2.5 SSL/TLS 위장

일부 피싱사이트의 경우, 실제로 SSL/TLS를 사용하지 않으면서 URL 내에 'HTTPS' 문자를

포함시켜 SSL/TLS를 사용하는 것처럼 사용자를 속이는 경우가 있다.

2.2.6 도메인 등록 기간

피싱사이트의 특성상 짧은 기간 동안만 활동하기 때문에, 도메인 등록 기간이 1년 미만인 경우 피싱사이트로 의심해볼 수 있다.

2.2.7 HTML 소스코드 내 도메인 존재 여부

통계적으로 정상사이트의 경우 HTML 소스코드 내에 자신의 도메인을 포함하는 경우가 많고, 피싱사이트는 원시사이트의 HTML 소스코드를 일부만 수정하여 그대로 사용하는 경우가 많기 때문에 HTML 소스 코드 내에 피싱사이트의 도메인이 존재하지 않을 확률이 높다.

2.2.8 HTML 소스코드의 길이

피싱사이트들은 원시사이트의 모습을 비슷하게 만들기 때문에, 원시사이트에 존재하는 기능이 없을 확률이 높다. 이것은 HTML 소스코드의 길이로 이어지며, 전체 데이터셋의 HTML 소스코드의 길이를 추출한 결과 정상사이트의 평균 HTML 소스코드 길이는 144,976자, 피싱사이트의 평균 HTML 소스코드의 길이는 30,731자로 확인되었다.

2.2.9 <head>, <body> 태그 중복 사용 여부

일반적인 정상사이트는 HTML 소스코드 내에 <head>, <body> 태그를 각 하나씩 사용한다. 전체 데이터셋의 HTML 소스코드 내에서 <head>, <body> 태그를 하나라도 2번 이상 사용한 사이트는 전체 데이터의 12%이고, 이 중에서 피싱사이트의 비율은 87%로 확인되었다.

2.2.10 페이지 대표 아이콘(Favicon)의 경로

통계적으로 정상사이트는 Favicon 이미지가 내부 서버 경로에 있지만. 피싱사이트는 Favicon을 사용하지 않거나, 외부 서버(원시사이트의 서버) 경로에 위치한다.

2.2.11 <form> 태그의 action 속성 값 검사

<form> 태그의 action 속성을 통해 서버에 정보를 전송할 수 있다. 대부분의 피싱사이트들은 action 속성 값이 외부 도메인(원시사이트의 도메인)이거나, 비어있는 경우가 많다.

2.2.12 <meta>, <script>, <link> 태그의 비율 검사

통계적으로 정상사이트는 위 태그들을 적절하게 사용하고 있지만, 피싱사이트의 경우 위 태그들이 차지하는 비율이 높다.

2.2.13 <a> 태그의 href 속성 값 검사

<a> 태그의 href 속성을 통해 다른 페이지로 하이퍼링크를 생성할 수 있다. 일부 피싱사이트에서는 href 속성에 하이퍼링크를 생성하지 않거나, 원시사이트로 연결되어 있는 것으로 확인되었다.

2.2.14 <script> 태그의 외부 링크 비율

<script> 태그의 src 속성을 통해 자바스크립트 파일을 불러올 수 있다. 일부 피싱사이트는 자바스크립트를 전부 외부 사이트에서 가져오거나, 하나도 가져오지 않는 경우가 있다.

2.2.15 외부 요청 URL 비율

통계적으로 정상사이트는 피싱사이트에 비해 웹 페이지 내에 포함된 자원(이미지, 비디오 등)을 내부 서버에서 가져오는 비율이 높다.

2.2.16 리다이렉션 횟수

보통 단축 URL을 사용하는 경우 한 번 리다이렉션되지만, 일부 피싱사이트는 두 번 이상 리다이렉션하여 사용자가 입력한 URL과 다른 URL로 연결되도록 하고 있다.

2.2.17 피싱 도메인 및 IP 사용 여부

PhishTank와 Stopbadware 사이트에 피싱사이트가 자주 사용하는 도메인과 IP에 대한 통계 보고서가 있다. 해당 보고서를 참조하여 일부 피싱사이트를 탐지할 수 있다.

2.2.18 웹 사이트 순위

Alexa 데이터베이스는 웹사이트 트래픽, 연결된 링크 수 등을 통해 웹 사이트의 순위를 측정한다. 대부분의 피싱사이트는 짧은 기간 동안 활동하기 때문에, Alexa에서 제공하는 순위가 낮거나 존재하지 않는다.

Ⅲ. 성능 분석

본 논문에서 제안된 탐지 기법의 성능을 분석하기 위하여 다음과 같은 분석 환경을 구성하였다.

· 학습 데이터셋 구성
 - 학습 데이터셋 : 8,000개
 - 테스트 데이터셋 : 9,636개
· 머신러닝 알고리즘 구성
 - Logistic Regression
 - Decision Tree
 - Random Forest

정상, 피싱 사이트 데이터셋 17,636개 중 무작위 8,000개로 머신러닝 모델을 학습시키고 나머지 9,636개 데이터로 테스트했다. 필터링 리스트를 사용하지 않고 sklearn 모듈의 cross_val_score 함수를 이용해 교차검증을 진행한 결과 Random Forest 알고리즘을 사용한 모델이 약 96%의 탐지율을 나타냈다. fig. 2.에서 confusion matrix 데이터를 통해 분류 모델에 대한 성능 평가를 할 수 있다.

이 수치는 참고문헌[3]에서 전처리 단계를 적용한 정탐률이 78%인 것을 고려하면 유의미한 수치라고 판단할 수 있다.

이는 알려지지 않은 최신 피싱사이트를 대상으로 했을 때를 가정하면 유효한 수치이며, 리스트와 결합되었을 경우 더 높은 정확도를 가질 수 있다.

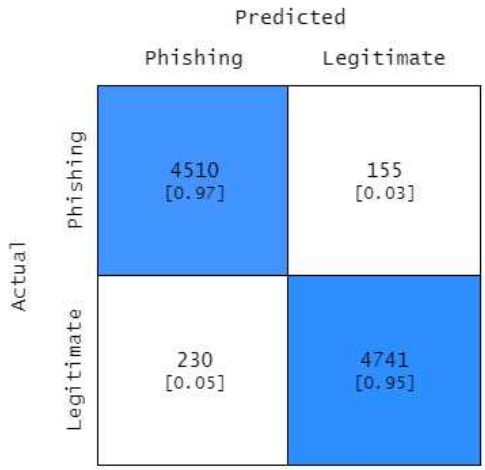


Fig. 2. confusion matrix

본 논문에서 제안한 방안은 기존의 탐지 방식을 우회하기 위해 SSL/TLS 통신을 하는 피싱사이트나 알려지지 않은 최신 피싱사이트를 탐지하는데 효과적일 수 있다.

Ⅳ. 결론

피싱사이트가 점점 고도화되고 수명이 짧아지는 만큼 기존방식으로는 탐지에 어려움이 있기 때문에 리스트와 머신러닝 알고리즘을 결합하여 피싱사이트 탐지 방안을 제안하였다.

본 논문에서 제안한 방식으로는 알려지지 않은 피싱사이트에 대해 96%의 탐지율과 3%의 미탐률을 보였다. 이는 기존에 이용되는 블랙리스트 기반의 탐지 방식과 비교할 때 우수한 탐지율이다.

하지만 웹사이트 내부 디렉터리로 접속할 경우 메인 웹사이트에 비해 탐지율이 낮아지는 경향이 있다. 따라서 다양한 형태의 웹사이트 데이터를 수집하여 학습시킬 필요가 있고, 피싱사이트 형태가 빠르게 변해감에 따라 feature의 분류 기준을 지속적으로 조정할 필요가 있다.

[참고문헌]

[1] Chiheb Chebbi, Mastering Machine Leaning for Penetration Testing: Develop an extensive skill set to break self-learning systems using Python, Packt Publishing, June, 2018

[2] Rami Mustafa A Mohammad, Lee McCluskey, Fadi Thabtah. (2015). Phishing Websites Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/phishing+websites

[3] Jeonguk Park, Gihwan Cho, A Unknown Phishing Site Detection Method in the Inerior Network Environment, pp. 1-8, Apr, 2015

위협 모델링을 통한 POS 시스템 보안 요구 사항 분석

김민수* 김대웅* 박건호* 박신철* 이승재* 임지훈* 이상섭**

*한국정보기술연구원 차세대 보안리더 양성 프로그램

**삼성 리서치

Trustworthy POS System: Security Requirements Analysis with Threat Modeling

Min-soo Kim* Dae-woong Kim* Gun-ho Park*
Shin-chul Park* Seung-jae Lee* Ji-hun Lim* Sang-sup Lee**

*Best of the Best 8th

**SAMSUNG Research

요 약

POS 시스템은 기존 단말기 영역에서 웹 페이지, 모바일로 영역을 확장했고, 더 많은 상거래 종사자들이 편리함과 접근성을 이유로 POS 시스템을 이용하게 되었다. 이때 POS 시스템은 결제 과정에서 사용자의 금융 정보를 취급하므로 무결성, 기밀성, 가용성이 보장되도록 설계 및 구현되어야 한다. 본 논문에서는 기존 POS 단말기와 VAN사에 한정된 POS 네트워크를 POS 업체 서버까지 확장한다. 이어서, Data Flow Diagram, STRIDE와 같은 위협 모델링 기법을 POS 시스템에 적용하여 위협을 식별하고 이에 따른 보안 요구사항을 도출하는데 필요한 위협을 기술한다. 이것을 바탕으로 신뢰할 수 있는 POS 시스템을 구축하는데 이바지한다.

I. 서론

POS(Point of sales, 판매시점 정보관리), POS 시스템, 카드 결제기 등의 다양한 이름으로 불리는 포스기는 소비자에게는 재화와 용역에 대한 지불 처리를 위해 현금, 카드, 포인트 등의 다양한 결제 방법을 제공하고, 판매자에게는 재고 및 재무관리 등의 서비스를 제공하여 매장 운영에 필수 요소가 되었다.(이하 '포스기'라 한다)

여신금융협회는 과거 포스기 보안 사고를 의식하여 최근까지도 'POS 단말 보안 요구 사항 표준'을 개정하여 안전한 포스기 보급을 위한 노력을 진행하고 있다[1]. 하지만, 이미 보급된 구형 기기를 취급하고 있는 업소가 다수이며[2], 사용자의 보안의식 결여로부터 오는 위험이 존재한다. 또한, 포스기는 결제가 이뤄지기 위해 VAN(Value Added Network, 부가가치통신망)사, 카드사와 결제 정보를 주고받는다. 이때, 내, 외부망을 함께 사용하기 때문에 필연적으로 어택 벡터가 증가한다. 실제로, 금융당국이 포스기의 보안 요구 사항에 대한 표준과 관련 법안을 운영하였지만[3], 최근까지도 침해 사고가 종종 보고되고 있다[4]. 금융 민감 정보를 다루는 포스기의 보안 강화의 필요성에 공감하여, 보안 사고 및 국내 포스기 업체의 현황을 토대로 향후 발생 가능한 위협 요소들을 파악하고자 본 연구를 진행한다.

본 논문의 구성은 다음과 같다. 2장 관련 연구에서 DFD(Data Flow Diagram)를 작성하고 현재까지 보고된 자료들을 바탕으로 3장에서는 시스템의 경계와 범위를 정하고 자산을 식별한다. 4장에서는 위협 모델 평가를 진행하고, 5장에서 본 논문의 결론을 맺는다.

II. 관련 연구

2.1 포스 통신 구조

포스기는 서비스 비용 지불에 대한 결제 데

이터의 처리와 전송 절차에 관여한다. 또한, 소프트웨어 유지 관리 및 비즈니스 데이터 처리를 위해 외부 객체와 통신하기 위한 네트워크 망에 연결되어 있다.

포스기는 카드 정보를 카드 리더기로 읽어 VAN사에 전송하여 결제 승인을 요청하고, 승인 응답을 받아 그 결과를 로컬 데이터베이스에 저장한다. VAN사는 카드 발급과 매입을 수행하는 카드사에 거래 데이터 중계 업무를 위탁받아 가맹점 업무와 거래 네트워크를 담당한다. 결과적으로 국내 신용카드 거래 구조는 Fig 1.과같이 나타낼 수 있다.

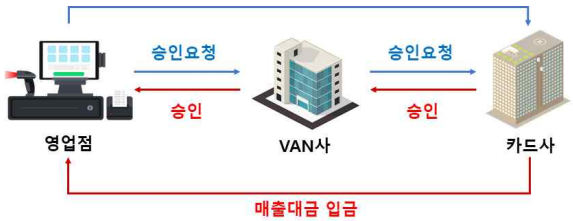


Figure 1: 신용카드 거래 절차 (VAN사 역할)

이러한 일련의 과정에서 포스기에서 보호되어야 할 기본 자산인 카드 정보, 암호 키, 포스기 보안 관련 저장 데이터는 메모리와 저장소, 네트워크 상에 존재한다.

2.2 관련 표준안

포스기 관련 표준으로는 TTA에서 제정한 POS 시스템 보안 요구 사항과 2015년 개정된 여신전문금융업법이 있다[3]. 그러나 TTA 표준은 법적 강제성이 없다. 또한, 포스기 서비스 영역이 확대됨에 따라 등장한 모바일을 비롯한 클라우드 포스, 간편 결제 수단에 대한 보안사항은 적용되지 않는다.

여신금융협회가 발표한 신용카드 단말기 정보보호 기술기준은 오직 '민감한 카드 정보'에 대해서만 다룬다. 하지만 포스기는 카드 정보뿐 아니라 판매 정보, 시스템 정보, 보안 기능 관련 정보 등 여러 형태의 자산이 존재한다. 이렇듯 법 개정에 따라 포스기 단일 단말에 대한 보안 수준은 높아졌지만, 실제 운영 환경에서 고려해야 하는 문제는 남아있다.

2.3 관련 침해 사고 사례 및 연구동향

국내외적으로 포스기를 대상으로 한 위협은 꾸준하게 존재해왔다. 미국의 경우 Target, Home depot, K-Mart 등 대형 유통 기업의 포스기에서 대규모 카드 정보 유출 사건이 발생하였고[5], 2017년 북한 해커 조직 Lazarus의 멀웨어 RAtankbaPOS는 국내 포스 단말기를 공격한 사례가 있다[6]. 최근에는 해커 조직 Fin7이 배포한 신종 멀웨어가 발견되었으며[7], 이외에도 포스기에 대한 위협이 지속되고 있다.

이러한 공격에 대응하기 위해 포스기에 대한 다양한 연구가 진행되고 있다. Attivo Networks는 기존에 발견된 포스 공격 사례를 바탕으로 위협에 대한 통계 분석을 진행하였다. 이를 통해 소매 업체를 규모별로 분류하여 공격 시나리오를 검증하였다[5]. 그리고 D. Chastuhin 외 1인은 미국에서 널리 사용되는 포스 소프트웨어에서 백엔드와의 통신 구간에서 발견한 취약점을 밝힌 연구 결과를 발표했다[8]. 이와 같이 관련 연구를 반영하여 신종 위협에 대응하기 위해 아직 인식되지 않은 위협을 발견하여 식별하기 위해 위협 모델링을 적용한다.

III. 본론

포스기에 대한 보안 위협의 분석 범위와 위협 요소를 구체화하기 위해 연구 가정 사항을 설정하고 DFD를 작성한다.

3.1 가정 사항 설정

본 연구에서는 포스기의 네트워크 구조를 결제 정보가 전송되는 흐름을 중점으로 분석하며, 포스기 형상관리 구간과 결제 승인 요청 구간을 포함한다. 분석 가정 사항으로는 크게 3가지가 있다. 첫째, VAN 또는 PG(Payment Gateway)사를 포함하는 복잡한 결제 과정을 하나의 결제 서버로 가정한다. 둘째, 포스 오너와 포스 관리자의 인증 과정에서 접근 제어 서버 또는 데이터 스토어를 경유한다고 가정한다. 셋째, 다양한 형태로 저장되는 로컬 데이터들을 하나의 데이터 스토어로 가정한다.

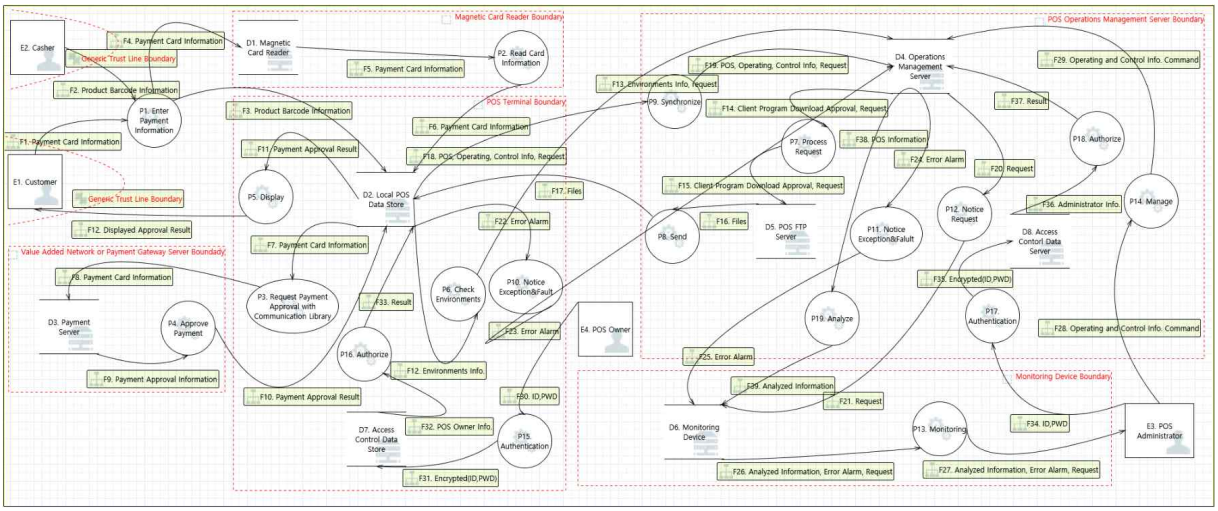


Figure 2: DFD

Table 1: POS Network DFD

Elements	Name	Symbol
External entity	Customer	E1
	Cashier	E2
	POS Administrator	E3
	POS Owner	E4
Process	Enter Payment Information	P1
	Read Card Information	P2
	Request Payment Approval with Communication Library	P3
	Approve Payment	P4
	Display	P5
	Check Environments	P6
	Process Request	P7
	Send	P8
	Synchronize	P9
	Notice Exception&Fault	P10
	Notice Exception&Fault	P11
	Notice Request	P12
	Monitoring	P13
	Manage	P14
	Authentication	P15
	Authorize	P16
	Authentication	P17
	Authorize	P18
	Analyze	P19
Datastore	Magnetic Card Reader	D1
	Local POS Data Store	D2
	Payment Server	D3
	Operations Management Server	D4
	POS FTP Server	D5
	Monitoring Device	D6
	Access Control Data Server	D7
	Access control Data Store	D8
		D9

포스기 소프트웨어에 제공되는 매뉴얼과 기존에 보고된 포스기의 네트워크 구조도를 종합하였고[2], 그 결과는 Fig 2.와 같다.

분석 대상은 국내 포스기로 선정하였고, 대금을 결제하는 고객과 결제를 중개하는 VAN사 또는 PG사, 운영정보를 파악하여 유지 보수를 제공하는 포스 제조사, 모니터링 서버를 포함하였다. 위 분석 대상의 영역을 붉은 점선 박스로 경계를 표시하였으며, 각 영역에서 보호되어야 할 자산은 Table 2.로 나타내었다.

Table 2: Asset of Boundary

Boundary	DB	Asset
VAN 또는 PG서버	D3	결제 및 금융정보
카드 리더기 영역	D1	카드정보
포스 단말기 영역	D2	개인·결제·포스 시스템·매장 정보
	D7	포스 오너 계정 정보
포스 제조사 서버	D4	개인·결제·포스시스템·매장 정보, 제조사 사업정보, 소프트웨어 운영정보
	D5	소프트웨어 관련 파일
	D8	관리자 계정 정보
모니터링 영역	D6	모니터링 관련 로그

IV. 평가

앞서 언급한 분석 범위에 따라 DFD를 기반으로 포스기의 데이터 흐름을 파악하고 식별한 위협을 STRIDE로 범주화했다. 이는 Microsoft가 제안한 보안 위협 모델링 방법으로, 자산에 대한 보안 위협을 각 기준으로 분류하여 식별하는 도구이다. 그 결과는 Table 3.과 같다.

Table 3: Number of Threat

Category	Number
Spoofing	57
Tampering	16
Repudiation	19
Information disclouser	31
Denial of Service	55
Elevation of priviledge	22

또한, 현재까지 보고된 국내외 포스기 보안 위협에 대한 자료를 수집해 Attack Library를 구축하여 참고문헌에 기재하였다. 이를 바탕으로 공격 방법을 구체화할 수 있는 위협을 가진 각 요소들을 Table 4로 분류하였다.

Table 4: STRIDE

	S	T	R	I	D	E
External Entity		O		O		O
Process		O		O	O	O
Data Store				O		O
Data Flow	O			O		

V. 결론

포스기 서비스 영역이 확대됨에 따라 새로운 위협요인들이 학계에 보고되고, 실제 공격 사례들이 등장하여 관련 기관들에 의해 연구되고 있다. 그러한 주요 논문과 보고서를 통해 포스기에 대한 보안 위협을 체계적으로 정량화하기 위해 Attack Library를 구축하고 STRIDE 기법을 적용하여 분석하였다. 이를 기반으로 최종적으로 총 200개의 위협을 도출하였으며, 각 위협 요소에 대해 전략적으로 대응 방안을 마련한다면 포스기 보안성을 강화할 수 있다. 또한, 위협에 대한 예측 가능성을 높여 향후 기존 포스 단말 요구 사항 표준의 개정 작업에 도움이 될 것으로 기대된다.

본 논문에서는 포스기의 위협 모델링 분석을 통해 예상 공격 요인들을 파악하였으나, 실제 운영 환경에서 테스트를 구축할 수 없다는 한계가 있다. 따라서 실질적인 공격 가능 여부 및 파악되지 않은 공격 지점에 대한 검증이 완료되지 않았다. 그러므로 향후 포스 기반의 상거래에 대한 안정성을 확보하기 위해서는 관련 유관기관과의 협조를 통해 실효성 있는 조사를 진행하고, 각종 표준안들에 대한 준수 여부를 파악할 필요가 있다.

[참고문헌]

[1] 정보통신단체표준(TTAS), POS 시스템 보안 요구 사항, TTAK.KO-12.0181/R1, 2014.12.

[2] 이소진, "POS 단말기 공급업체의 보안 규제필요성에 관한 연구," 석사학위논문, 고려대학교, 2016년 6월.

[3] 여신금융협회(2015). 신용카드 단말기 정보보호 기술기준

[4] 금융보안원(2018). 전자금융과 금융보안 제14호

[5] “Point-of-Sale System Attacks: The Role of Early Detection for Breach Prevention”, Attivo Networks, 2017.

[6] Darien Huss, North Korea Bitten by Bitcoin Bug: Financially motivated campaigns reveal new dimension of the Lazarus Group, proofpoint, DEC 2017.

[7] RSA Research, The Carbanak/FIN7 Syndicate: A Historical Overview of an Evolving Threat, RSA, Nov 2017.

[8] D.Chastuhin and V.Egorov, “Get of the money: Hacking POS and POP Systems”, 2017.

[9] Slava Gomzin. HACKING POINT OF SALE: Payment Application Secrets, Threats, And Solutions, John Wiley&Sons, 2014.

[10] "A SPECIAL REPORT ON: Attacks on point-of-sales systems", Symantec, 2017

[11] Ricardo J., "Evolution and Characterization of Point-of-Sale RAM Scraping Malware", Journal of Computer Virology and Hacking Techniques, Vol. 13, pp-179 - 192, Aug. 2017.

[12] W. Whitteker, Point of Sale (POS) Systems and Security, SANS, Oct 2014.

PDF 악성 코드 탐지 방법과 발전에 관한 연구

이영한*, 김요한*, 하희리*, 백윤흥*,

*서울대학교 전기 정보공학부, 반도체 공동 연구소

A Study of PDF malware detection techniques and their development

Younghan Lee*, Yohan Kim*, Whoi Ree Ha* and Yunheung Paek*

*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC),

Seoul National University

Abstract

We are living in the era of exploding number of malware where the attackers attempt to implement their exploits in various different forms. With recent rise in usage of PDF files, the attackers have expanded their targets to such format. As PDF files are consist of numerous objects, it was a suitable medium to hide and camouflage the malicious contents within PDFs. Therefore, many research have focused on improving PDF malware detection techniques by employing machine learning. A typical process of PDF malware detector includes three phases: feature extraction from PDF files, feature selection, and finally, model learning for classification. In this paper, two different approaches of detecting PDF malware will be introduced and how the current techniques can be improved will be discussed.

I. Introduction

In recent years, a large number of application has adopted machine learning (ML) in their classification tasks. Such applications include speech recognition and image processing. Coherently, a field of security also adapted a variety of ML-based techniques in pursuit of improving the detection accuracy while reducing false alarm rate. There are several studies that empirically proved a great potential and effectiveness of ML in tackling security challenges such as malware detection. Portable Document Format (PDF) malware have been earning their popularity in past few years as PDF became one of the most popular file format used. It is reported that as malicious PDF files is known to be the most dangerous type of attack exploited by adversaries these days, ML-based techniques are being actively studied and researchers have tried to mitigate these attacks. In order to prepare for the flooding of coming zero-day PDF malware, many have attempted to improve the performance PDF malware classifiers by applying ML techniques. The paper that set the milestone for tackling PDF malware detection problem with ML techniques was PDFrate-v1. It used metadata

and structural features of PDF documents as an input for training a random forest (RF) model for detection. Another paper called Hidost attempted to extract the files into a structural map and used it as a feature set. Support vector machines (SVMs) are used as classification model. The rest of this paper is structured as the following: background, explanation of current PDF malware detection techniques, future work and conclusion.

II. Background

2.1 PDF Structure

PDF files are consist of four parts: header, body, cross-reference table and trailer. Header contains the version number of the PDF specification. Body is where objects are located which holds all data of the document. Finally, a trailer specifies the offset value for the reader to find the reference table that helps find the wanted object quickly.

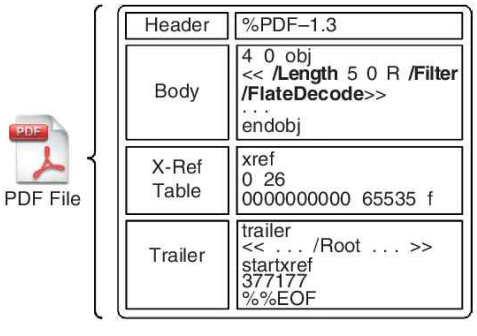


Figure 1. Structure of PDF file

2.2 PDF Malware Types

There are mainly three types of PDF malware. Firstly, JavaScript-based attack exploits a vulnerability of PDF reader by means of JavaScript code which can be hidden in objects in PDF files. Secondly, ActionScript-based attacks exploits the fact that PDF files can show Flash content. By embedding ActionScript that contains memory corruption code with ShockWave Flash in PDF, adversaries can hide the malicious content. Finally, File-embedding technique capitalize on the fact that PDF reader can parse PDFs that are embedded with other file types.

III. PDFRate-v1

3.1 Feature Extraction

To extract structural feature from PDF, PDFRate-v1 utilized a parser which was manually design. It extracts metadata from documents and object/stream are encrypted. Much of metadata items are inherently numeric.

3.2 Feature Selection

Feature selection process is very important as selected feature must characterize PDF documents. Authors heuristically chose few features that would represent the whole of PDF file. The list of manually selected features includes names of authors, size of a file and data encoding method etc. The total number of features is 202.

3.3 Random Forest Model

Random forest is suitable model for such problem as it achieves a high classification accuracy and it is easy to use. As the name suggests, random forest classification method contains many decision trees. Each small decision tree is generated

from subset of training set and each of tree votes in classification phase to determine the result.

3.4 Classification

The goal of classification is to determine if the input is malicious or benign. It can be designed as a binary classification and the performance of classifier is measured by drawing receiver operating characteristic (ROC) curves which depicts true positive and false positive rates.

IV. Hidost

4.1 Feature Extraction

In order to draw a directed rooted tree graph from the structure of PDF, Hidost utilized a parser called Poppler. It is a PDF rendering library. Once all the features are extracted from the document, Hidost simplified the multimap by the types. For example, all string type features were converted to a constant integer value, 1.

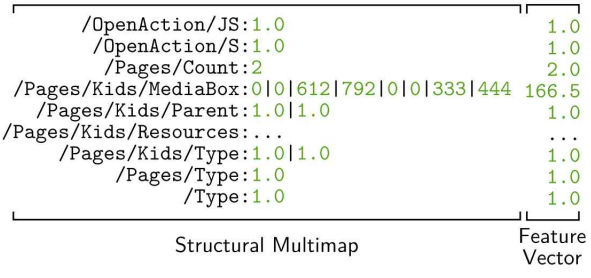


Figure 2. Feature simplification

4.2 Feature Selection

For feature selection, a simple rule was applied. Each feature holds a count number of how many files contained that feature. Only features that more than a certain number of files had were added to the list of selected features.

4.3 Model Learning

For model learning, the same random forest model as PDFRate-v1 was applied. For the method of implementing random forest model, scikit-learn Python machine learning library was used. For the experiments, the size of forest was set to 200 trees.

V. Development of PDF Malware Detection

5.1 Limiting the queries

In coherent to the improvement in defense mechanism, the attackers came up with various techniques to bypass the classifiers. The attackers will send a lot of queries to the malware classifier and return the result whether the input file was malicious or benign. Then the adversaries will make few perturbation to the original malware to bypass the classifier while maintaining the original malicious content. Therefore, the only defense technique valid to hinder this attack is to limit the number of queries that can be submitted to the classifier.

VI. Conclusion

Machine learning has been adopted in many fields of studies as it showed many promising results. With the rise of complicated malware especially regarding PDF documents, the defense mechanism had to improve accordingly. The first attempt to apply machine learning method in malware classification was done by PDFRate-v1. It used a manually designed feature extraction and selection process. On the other hand, Hidost accomplished the better result with heuristic approach in feature extraction and

selection process. The most recent defense tool is to limit the number of queries to the malware classifier so that the attacker will not be able to perturb the malware slightly. Attack and defense regarding PDF malware is a tight arms race that improves dramatically. It is important to be aware of both sides of the story to understand the current status of PDF malware.

VII. Acknowledgement

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발) 또한, 이 논문은 2019년도 두뇌한국21플러스 사업에 의하여 지원되었음

[References]

[1] Šrndić, N. and Laskov, P., 2016. Hidost: a static machine-learning-based detector of malicious files. EURASIP Journal on Information Security, 2016(1), p.22.

[2] Smutz, C. and Stavrou, A., 2012, December. Malicious PDF detection using metadata and structural features. In Proceedings of the 28th annual computer security applications conference (pp. 239-248). ACM.

[3] Smutz, C. and Stavrou, A., 2016, February. When a Tree Falls: Using Diversity in Ensemble Classifiers to Identify Evasion in Malware Detectors. In NDSS.

[4] Maiorca, D., Biggio, B. and Giacinto, G., 2019. Towards adversarial malware detection: Lessons learned from PDF-based attacks. ACM Computing Surveys (CSUR), 52(4), p.78.

[5] Šrndic, N. and Laskov, P., 2013. Detection of malicious pdf files based on hierarchical document structure. In Proceedings of the 20th Annual Network & Distributed System Security Symposium (pp. 1-16).

대량 악성코드 자동화 분석 프레임워크

이성원, 전현규, 박기현, 윤종희*

영남대학교 컴퓨터공학과

Mass Malware Automation Analysis Framework

Sung-Won Lee, Hyeon-Kyu Jeon, Gi-Hyun Park, Jonghee Youn

Dept. of Computer Engineering, Yeungnam University.

요 약

대량의 악성코드를 분석하고 결과를 확인하기 위해 새로운 악성코드 자동화 분석 프레임워크를 설계, 구현하였다. 본 논문에서 제안하는 악성코드 분석 프레임워크는 정적분석과 동적분석 방식을 통합적으로 사용할 수 있는 자동화 분석 도구이다. 단일 악성코드에 대한 분석 리포트만 얻을 수 있었던 기존의 알려진 도구들과는 달리 파일을 집합 단위로 분석하고, 분석 결과 데이터를 여러 단계로 가공하여 전체의 특징별 분류, 개수 합산과 같이 결과를 파일 집합 수준에서 요약할 수 있는 특징을 가진다.

I. 서론

대량의 악성코드를 효율적으로 분석하고 분석 결과에 대한 사용자 정의의 2차 분류, 해석을 위한 프레임워크를 구축하였다. 분석 환경에는 악성코드에 대한 정적분석·동적분석을 손쉽게 수행할 수 있도록 자동화 기능을 제공하며, 대량의 파일에 대한 효율적인 분석과 다중처리를 위한 스케줄러를 지원한다.

* 교신저자(youn@yu.ac.kr)

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2019-2016-0-00313) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation), the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (No.2018R1D1A1B0705647) and Yeungnam University Research Grant.

II. 분석 프레임워크 설계

기존에 잘 알려진 샌드박스형 분석 도구들(Cuckoo Sandbox[1], Limon Sandbox[2], HaboMalHunter[3] 등)의 경우 누구나 사용할 수 있게 오픈된 소스이고 각 도구만의 장점이 존재하지만, 본 논문에서 목표로 하는 대량분석에는 한계점과 문제점들이 존재했다. 대표적인 한계와 문제점으로는 분석 환경에 대한 OS 버전과 커널 버전의 제약, 단일 파일에 대한 분석 위주 시스템, 그리고 분석 시작부터 완료 후 결과 데이터를 통한 분류까지의 일관된 파이프라인이 없다는 부분을 꼽을 수 있다. 본 논문에서 제시하는 프레임워크는 6개의 설계목표를 기반으로 구축되었다. 1-오래된 버전의 OS, 커널에서도 분석, 2-단일 파일만이 아닌 파일 집합에 대한 일괄 분석, 3-분석 결과를 여러 단계에 통해 처리·가공·해석, 4-사용자 커스터마이징 과 기능 확장성, 5-분석 완료된 건에 대한 추가적인 데이터 가공, 6-다양한 환경에 대한 호환성,

이렇게 6개의 쟁점을 해결한 프레임워크를 구축하였다.

2.1 분석 환경의 전체적인 구조

분석 환경의 전반적인 구조는 그림 1과 같은 형태를 지니고 있다. 실제 분석 작업들이 관리되는 분석 프레임워크와 이에 대한 세부 설정과 작업 내용을 결정하기 위한 사용자 지정의 프론트엔드, VM 관리자, 에이전트, 분석기 스크립트로 이루어진다.

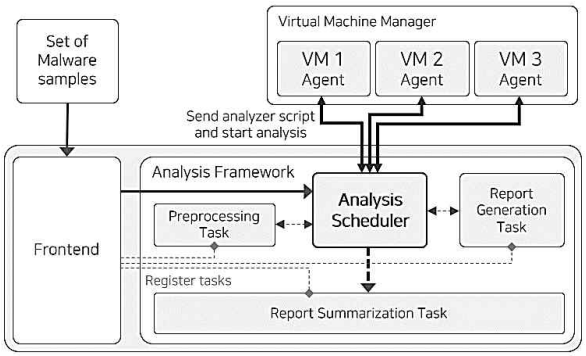


그림 1 분석 환경 구조

2.2 프론트엔드

프론트엔드는 프레임워크를 참조하는 실행 파일이며, 분석에 대한 각종 환경설정을 지정하고 분석작업을 시작시키는 구성요소이다. 정적분석 도구나 가상화 소프트웨어의 경로를 지정하고, 동적분석에 사용할 VM 프로파일을 등록한다. 전처리 작업과 리포트 생성 작업, 리포트 요약 작업의 내용 들을 각각 등록한 후, 분석할 파일 목록을 생성하고 분석 프레임워크에 전달하여 분석 프로세스를 시작한다. CLI 및 GUI 애플리케이션 또는 웹 애플리케이션과 같이 사용자 필요에 의해 다양한 형태로 구현될 수 있다.

2.3 분석 프레임워크

분석 프레임워크는 정적분석기 및 동적분석기와 분석 작업 전후로 데이터를 처리하고 가공하기 위한 다양한 작업 객체들과 스케줄러를 포함하고 있는 집합체이다. .NET Core 런타임 상에서 동작하는 클래스 라이브러리로서 구현되었으며, 각각의 가상머신 인스턴스를 캡슐화한 분산된 작업자 모델을 포함하고 있다. 분석 프레임워크는 실제로 샌드박스를 제어하기 위한 엔진이며, 자체 스케줄러를 통해 정적분석을 비롯한 전처리 작업, 동적분석, 개별 리포트 생성 작업, 전체 리포트 요약 작업을 관리하고, 이들을 적절한 시점에 호출한다. 전처리, 리포트 생성, 리포트 요약의 각 작업 범주에는 복수개의 작업이 등록될 수 있으며, 이 전체 흐름 자체를 필요에 따라 다중 인스턴스로 동작시킬 수 있다. 전처리 작업, 리포트 생성 작업, 리포트 요약 작업은 기본적인 유즈케이스에 따라 프리셋이 내장되어 있으며, 프리셋 외에도 사용자가 직접 구현하여 등록할 수 있도록 설계되어 있다.

악성코드의 실질적인 분석이 이루어지는 동적분석기 에서는 리눅스에서 기본적으로 제공하는 도구들과 오픈소스 기반의 몇몇 도구들을 사용하여 일차적인 분석을 수행하며, 분석 결과를 기능과 목적에 따라 추적, 상태 모니터링, 네트워크, 프로세스, 기타라는 5가지 범주로 분류한다. 내부적으로 사용되는 도구에 대한 목록은 표 1에서 확인할 수 있다.

범주	도구
추적	strace, ltrace
상태 모니터링	pstree, lsof, service, netstat, lastlog /proc/mounts .bash_history 등
네트워크	tcpdump, tshark
메모리	volatility
정적분석	MinGW(ldd, file, readelf, strings), upx

표 1 분석에 사용되는 도구 목록

III. 전체 분석 파이프라인

제출된 악성코드 샘플에 대해 정적 분석 단계와 동적 분석 단계, 그리고 분석 결과의 해석 단계를 거쳐 최종적으로 악성코드가 분류된다.

전체 분석 파이프라인은 그림 2와 같다.

정적분석 단계에서는 2.3의 표1에서 언급한 정적분석 도구들을 사용하여 파일 타입 인식, ELF 헤더 확인, 패킹여부 확인, 문자열 추출 등의 정적분석을 수행한다. 정적분석을 통해 확인한 내용을 통해 동적분석이 가능할 것으로 판단되면 준비된 VM을 이용하여 동적분석을 시도한다. 동적분석에서 pstree와 service 명령어를 이용하여 프로세스의 변화를 확인하였고, lsof 명령어를 이용하여 로드된 라이브러리 목록을 추출하였다. netstat과 tcpdump를 이용하여 네트워크 행위를 확인하였다. 그 외에 쉘 명령어의 기록이나 로그인 기록, 마운트 정보와 같은 부가정보를 확인하였다.

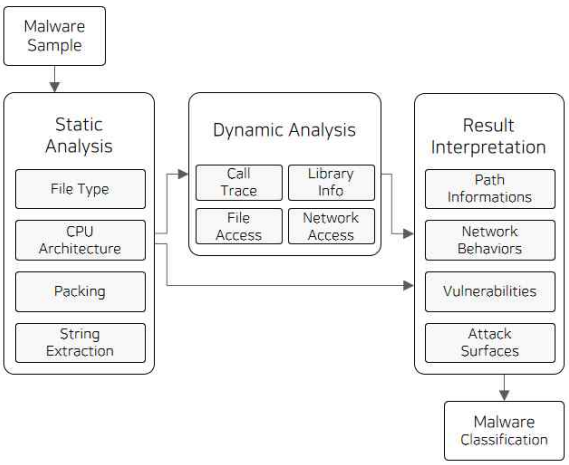


그림 2 전체 분석 파이프라인

결과 해석 단계에서는 정적분석 단계와 동적분석 단계에서 추출된 정보들을 이용하여 악성코드가 실제로 어떠한 경로에 접근하는지, 어떠한 네트워크 행위를 하였는지, 어떠한 종류의 취약점을 활용하는지, 어떤 공격 표면을 목표로 하는지 종합적으로 판단 후 사용자가 설정한 기준으로 결과 및 통계를 출력, 분류한다.

IV. 결론

현재 악성코드 자동화 분석을 위한 여러 연구 들이 진행 중이다[4]. 하지만 다양한 대상과 목적, 특징을 가진 악성코드를 분석하기 위한 완성도 있는 자동화 분석도구가 존재하지 않으며, 대량분석에도 적합하지 않다. 반면 본 논문에서 설계한 분석 프레임워크는 .NET Core 플랫폼과 MinGW를 통해 크로스 플랫폼으로 구현되어 Windows, Linux, Mac OS 등에 제한없이 사용할 수 있고, i386, amd64 외 다양한 아키텍쳐에 대한 동적 분석을 지원하는 등의 높은 호환성과 이식성을 지닌다. 또한, 분석 스케줄러를 통한 효율적인 자원 관리를 통해 분석과정에서의 시간을 단축시키고, 사용자가 원하는 기준과 분류를 쉽게 적용가능하기 때문에 향후 다양한 연구주제에 적용하고 기초로 활용할 수 있다.

[참고문헌]

[1] https://cuckoosandbox.org/

[2] K.MONNAPPA, Automating linux malware analysis using limon sandbox, Black Hat Europe 2015, 2015.

[3] https://github.com/Tencent/HaboMalHunter

[4] E.Cozzi, M.Graziano, Y.Fratantonio, D.Balzarotti, Understanding linux malware, 2018 IEEE Symposium on Security and Privacy (SP), IEEE, 2018.

Windows 10 Timeline을 이용한 사용자 행위 분석

현주연* 김재희** 김가은** 김성욱** 김하늘** 송찬영** 김영철*** 허원석****

*KITRI Best of the Best

**KITRI Best of the Best

***삼일회계법인

****KITRI Best of the Best

User behavior analyze with the Windows 10 Timeline

Ju-Yeon Hyun* Jae-Hee Kim** Ga-Eun Kim** Seong-Wook Kim** Ha-Neul Kim** Chan-Young Song** Young-Cheol Kim*** Won-Seok Heo****

*KITRI Best of the Best

**KITRI Best of the Best

***SamilPwC

****KITRI Best of the Best

요 약

개인 PC의 사용량이 증가함에 따라 해당 사용자와 관련된 많은 정보들을 가지고 있고, 우리는 이를 쉽게 볼 수 있다. 또한 Windows Redstone4 업데이트로 인해 생긴 Timeline 기능을 통해 사용자의 행위와 관련된 많은 정보들을 볼 수 있다. 해당 기능은 사용자의 행위를 시각화하는 기능으로 동일한 MS계정 내에서 공유된다. 본 논문에서는 Timeline 기능과 이로 인해 생긴 Timeline 아티팩트를 이용하여 사용자 행위를 추적하고 이에 대해 분석한다.

I. 서론

최근 전자 기기를 통한 범죄 수사가 많이 이루어지고 있는 추세이다. 이 때, 수사관들은 전자 기기에 남아있는 흔적들을 이용하여 사용자의 행위를 재구성한다. 기존 컴퓨터 내에 저장되어 있는 흔적들은 온라인으로 어떤 브라우저를 사용해서 어떤 것을 하였는지 알 수 없고, 만약 그 흔적들이 삭제된다면 삭제된 여부를 알아차리기 힘들다. 따라서 본 논문에서는 이러한 행위들을 확인하고, 분석에 수월함을 제공하기 위해 새로 추가된 Timeline 기능과 Timeline 아티팩트를 이용하여 온라인 상의 사용자 행위와 흔적 삭제 여부에 대해 연구하였다.

II. 본론

2.1 본 연구의 개요

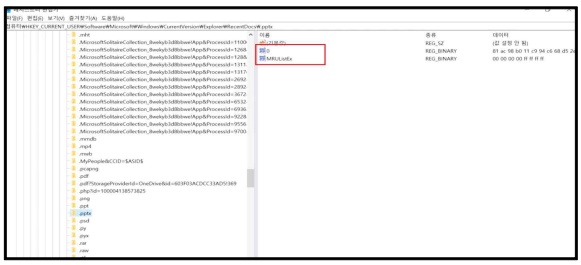
최근 개인 PC와 같은 전자 기기들의 발달로 인해, 전자 기기를 이용한 범죄율이 증가하고 있다. 이에 따라 수사 절차에서 디지털 증거의 영향력을 무시할 수 없고, 수사관이 얻을 수 있는 참고자료도 증가하고 있다. 이러한 범죄가 발생하였을 때, 포렌식 수사 과정에서는 사건과 관련된 전자 기기를 압수하여 사용자의 행위를 추적하고 이를 분석하며, 사건을 재구성한다. 만약 용의자가 흔적을 삭제했다면, 사건 재구성 과정에서 해당 행위에 대해서는 분석이 어렵다. 또한 인터넷을 사용했다면 인터넷 사용한 것은 알 수 있지만, 어떤 브라우저를 사용하여 어떤 url에 접속했는지, 무엇을 보았는지는 알기 힘들다. 하지만 Windows Redstone4 업데이트로 인해 생긴 Timeline 기능과 그 기능의 아티팩

트를 통해 위와 같은 행위들(흔적 삭제 및 자세한 인터넷 사용 흔적)에 대해 알 수 있다. 따라서 본 논문은 Timeline 기능과 Timeline 아티팩트를 이용하여 온라인 상의 사용자 행위와 흔적 삭제 여부를 알아내는 데에 중점을 두고 연구하였다.

2.2 기존 연구 사례

2.2.1 레지스티리를 통한 삭제 여부 확인

레지스티리란 Windows 계열 시스템에서 사용하는 시스템 구성 정보를 저장한 데이터베이스를 말한다. 이는 'regedit'이라는 Windows 자체의 프로그램으로 확인 및 편집을 할 수 있다. 레지스티리 내에는 [그림 1]과 같이 파일 실행 및 최근에 열람했던 문서에 대한 기록이 남는다.

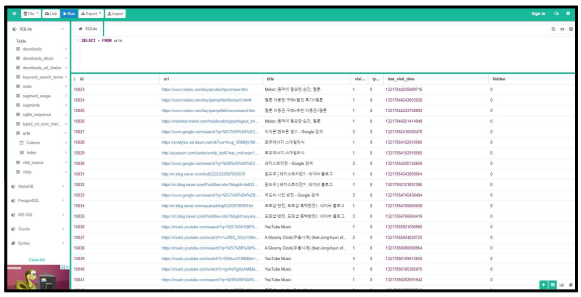


[그림 1] regedit로 확인한 MRU 레지스티리

이 기록들은 레지스티리 편집기인 'regedit'의 기능 중 하나인 '삭제' 기능으로 해당 기록들(이하 레지스티리 키 값들)을 삭제할 수 있다. 이렇게 삭제된 키 값들은 발견하기가 힘들고, 만약 발견한다면 Windows에서 제공하는 'NTBackup' 툴이나 'Workshop'이라는 툴을 이용하여 기존 백업 본과 삭제된 백업 본을 비교 분석하며 확인할 수 있다.

2.2.2 웹 아티팩트를 통한 온라인 행위 확인

웹 아티팩트란 브라우저 내의 활동으로 인해 생기는 기록으로써, 히스토리, 다운로드, 캐시, 쿠키 이렇게 총 4가지가 있다. 그 중 하나인 History DB를 열어보면 아래의 [그림 2]처럼 접속했던 사이트의 주소가 자세하게 남아있는 것을 확인할 수 있다.



[그림 2] History DB내의 기록

웹 아티팩트도 만약 삭제된다면 레지스티리처럼 알아차리기 힘들고, 삭제된 것을 복구하는 것 또한 'IE10Analyzer'이라는 툴을 사용하여 복구해야 한다.

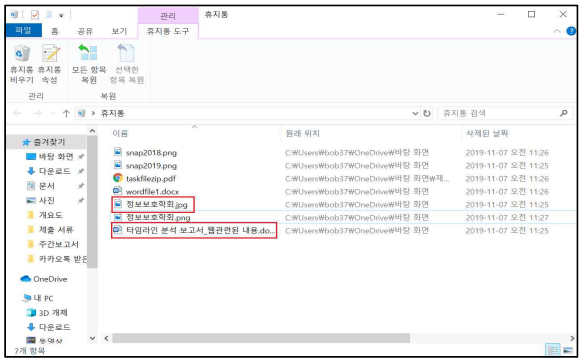
2.3 본 논문의 추가 분석 내용

2.3.1 흔적의 삭제 여부 확인

기존의 연구를 토대로 레지스티리를 통해 삭제 여부를 확인하려면, 툴을 따로 사용해야 한다. 그러나 Timeline 기능을 이용한다면, 레지스티리와 다르게 툴 사용 없이 그 자체 만으로도 흔적의 삭제 여부가 확인이 가능하다. 만약 용의자가 악성 코드가 내포된 *.doc과, *.jpg을 실행시킨 후 해당 파일을 삭제했다고 가정하면, 먼저 아래의 [그림 3]처럼 Timeline 기능을 통해 이를 실행시킨 기록을 확인할 수 있다.



[그림 3] Timeline 기능을 통해 확인한 파일 실행 기록



[그림 4] 휴지통에서 삭제된 파일 확인

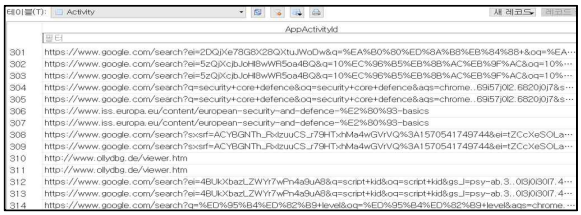
[그림 3]과 같이 Timeline 기능을 통해 용의자가 실행한 파일의 이름과 파일 확장자 명을 확인할 수 있다. 이미지 파일 같은 경우는 해당 이미지가 썸네일로 만들어 지면서 어떤 이미지인지도 확인이 가능하다. 또한, [그림 4]처럼 삭제된 파일이 많아 해당 파일을 찾기 어려운 경우에도 [그림 3]에서처럼 확인한 정보를 통해 쉽게 해당 파일을 찾아낼 수 있다. 만약 [그림 4]처럼 같은 이름의 다른 확장자 명을 가진 파일(이하 정보보호학회.jpg 와 정보보호학회.png)이 있다면, [그림 3]에서 확인한 확장자 명을 통해 용의자가 삭제한 파일을 정확하게 알아낼 수 있다.

2.3.2 온라인 상의 사용자 행위 분석

Timeline 기능과 Timeline 아티팩트 DB내에는 사용자가 컴퓨터 내에서 실행한 애플리케이션의 기록 뿐만 아니라, [그림 5]와 같이 인터넷 접속과 같은 온라인 상의 기록도 기록된다.



[그림 5] Timeline 기능 상에 남는 인터넷 접속 기록



[그림 6] Timeline 아티팩트 DB 상에 남는 인터넷 접속 기록

그리고 [그림 6]과 같이 Timeline 아티팩트 DB에는 웹 아티팩트처럼 접속했던 실제 사이트 주소가 좀 더 자세하게 남는다.



[그림 7] 특정 사이트 썸네일

Timeline 기능은 웹 아티팩트와 달리 접속한 기록들을 썸네일로 보여준다. 방송사에서 제공하는 VOD나 블로그에 접속한 경우는 [그림 7]과 같이 해당 플랫폼에서 사용한 이미지들을 사용하여 썸네일로 보여준다. 반면, 일반 사이트는 아래 [그림 8]처럼 Timeline 기능의 썸네일이 특정 이미지가 아닌 검색한 기록 키워드로 보여준다.



[그림 8] 검색 기록이 나와있는 썸네일

이처럼 Timeline 기능을 사용하면, 사용자가 검색한 기록 키워드와 특정 썸네일 이미지를 통하여 해당 사이트의 추가 정보를 쉽게 획득할 수 있다.



[그림 9] 접속 기록의 Starttime 과 Endtime

Timeline 아티팩트 DB를 통해 웹 아티팩트보다 좀 더 자세한 내용들을 확인할 수 있다. [그림 9]처럼 사이트 접속 시작시간(이하 Start Time)과 접속 종료시간(이하 End Time)과 같이 자세한 기록도 남는다. 또한 DB 내에는 각각의 브라우저가 AppId와 Payload의 값으로 구분되어 기록되어져 있다. 이렇듯, 인터넷 접속 기록만이 아니라 아래의 [그림 10]과 [그림 11]과 같이 DB를 통해 사용한 웹 브라우저의 유형을 확인할 수 있다.



[그림 10] DB를 통해 확인할 수 있는 사용한 브라우저의 종류



[그림 11] Chrome AppId

예를 들어 [그림 11]에서처럼 Chrome은

'webactivities.activity.windows.com'

라는 AppId를 가진다. 이처럼 Timeline 아티팩트 DB는 각각의 브라우저 별로 AppId를 다르게 하거나 Payload의 구분된 값을 통해 브라우저 별로 정보를 제공한다.

III. 결론 및 향후 연구 방향

본 논문에서는 Redstone4 업데이트로 Windows 10에 생긴 Timeline 기능과 Timeline 아티팩트 DB를 통해 흔적의 삭제 여부 및 용의자의 온라인 상 활동에 대해 연구했다. 본 연구를 통해 용의자의 흔적 삭제 여부와 온라인 상의 활동을 확인할 수 있었다. 이는 용의자가 거짓 진술을 할 경우 Timeline 기능과 Timeline 아티팩트 DB를 확인함으로써, 증거인멸과 온라인 상의 상세 활동을 파악할 수 있다는 기대 효과를 가져올 수 있다. 또한, 용의자가 사용한 웹 브라우저, 접속한 사이트 및 검색 기록들을 수사관이 확인할 수 있으므로 구체적인 접근이 가능해지고 사건을 좀 더 정확히 파악할 수 있을 것이다.

향후 연구 방향으로는 Timeline 아티팩트 DB 내에 있는 UserTimeZone이라는 항목의 데이터를 분석하여 해당 사용자가 어느 TimeZone에서 웹 사이트 접속을 했는지에 대해 분석을 할 것이다. 그리고 시간 값들을 가지고 있는 항목들로 해당 사이트에 접속한 총 시간에 대해 분석할 것이다. 이렇게 분석한 값들을 가공하여 수사 과정에서 사용자의 행위를 특정 지을 수 있도록 할 것이다.

본 논문은 KITRI BoB 프로그램의 지원을 받아 작성되었습니다.

[참고문헌]

[1] 서승희, 서울과학기술대학교, 2018.04, 디지털 포렌식 관점에서의 인스타그램 사용자 행위 분석

[2] Horsman, Graeme, 2019, A Forensic Exploration of the Microsoft Windows 10 Timeline

[3] https://www.slideshare.net/CostasKatsavounidis/windows-1809-timeline

[4] 박정민, 경일대학교, 2019.06, ElasticSearch와 Kibana를 이용한 웹 아티팩트 시각화

[5] https://m.blog.naver.com/PostView.nhn?blogId=rbdi3222&logNo=220597850076&proxyReferer=https%3A%2F%2Fwww.google.com%2F

[6] http://blog.plainbit.co.kr/wp-content/uploads/2018/05/Windows-File-Execution-Open-Artifacts_181225.pdf

[7] https://asecurity.dev/2018/02/%EC%9C%88%EB%8F%84%EC%9A%B0-%EC%82%AD%EC%A0%9C%EB%90%9C-%EB%A0%88%EC%A7%80%EC%8A%A4%ED%8A%B8%EB%6%AC-%ED%99%95%EC%9D%B8-regslack/

[8] https://blog.naver.com/ifkiller/70157957647

[9] https://moaimoai.tistory.com/45

SNI 차단 및 암호화 연구

윤재웅* 김경호** 서화정**†

*한성대학교 컴퓨터공학부

**한성대학교 IT융합공학부

A Study of SNI Blocking and Encryption

Jae-Woong Yun*

*School of Computer Engineering, Hansung University.

Kyung-Ho Kim** Hwa-Jeong Seo**†

**Division of IT convergence Engineering, Hansung University.

요 약

정부는 지난 2월 고도화된 인터넷 접속차단 정책인 SNI 차단을 도입했다. 주요 논란점은 차단 과정에서 불법적으로 패킷의 일부분이 해킹된다는 부분이다. 이로 인해 TLS 1.3에서 제안된 SNI의 암호화인 ESNI가 주목받고 있다. 이에 본 논문에서는 이전부터 현재 SNI 차단까지 과정 및 TLS 1.3에서 제안된 ESNI의 특징 및 드래프트에서 Open Issue로 언급된 ESNI를 통한 0-RTT 가능성 대해서 알아본다.

I. 서론

지난 2월, 정부는 HTTPS(Hypertext Transfer Secure) 및 우회 접속 방식으로 불법 사이트를 제공하는 해외의 유해 웹사이트에 대한 접속 차단 기능을 고도화하기 위해 SNI(Server Name Indication)차단 시행을 발표했다[1].

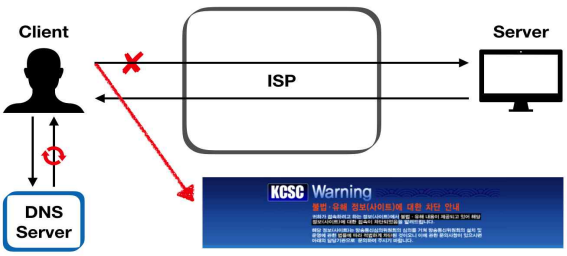
본 논문에서는 정부의 차단 방식과 이 과정에서 발생할 수 있는 보안상의 문제점을 해결할 수 있는 방법에 대해서 기술한다. 그리고 TLS(Transport Layer Security) 1.3에서 제안된 ESNI(Encrypted SNI)에 특징을 알아보고, 키 교환방식에 따른 기존 TLS 1.3에서 제안된 0-RTT(Round Trip Time) 과정을 드래프트의 Open Issue에서 언급한 Full Handshake 과정 없이 수행할 수 있는 가능성을 확인해본다.

구성으로 2장에서 과거부터 현재까지 진행된 차단방식들인 DNS, HTTP, HTTPS(SNI) 차단을 살펴보며, 3장에서는 SNI의 암호화인 ESNI의 특징 및 조건을 알아보고, 4장에서는 기존 제시된 방식과 ESNI를 통한 방식으로 0-RTT를 알아본다. 마지막으로 5장에서 결론으로 마무리한다.

II. 인터넷 차단

2.1 DNS 차단

DNS(Domain Name System)는 호스트의 도메인 이름을 호스트의 네트워크 주소로 바꾸거나 그 반대의 변환을 수행할 수 있게 해준다. DNS에 주소를 요청할 때 해당 도메인 네임이 불법 사이트 목록에 등재된 것이라면 [그림 1]과 같이 실제 IP 주소가 아닌 “warning.or.kr"라는 경고 웹페이지로 리다이렉션 한다. 이러한 기술은 파밍(Pharming)이라고 부르며 불특정 다수에게 메일을 발송하여 위장된 웹페이지로 접속하게 한 후에 개인정보를 빼내가는 피싱(Finshing)의 발전된 기법이다.

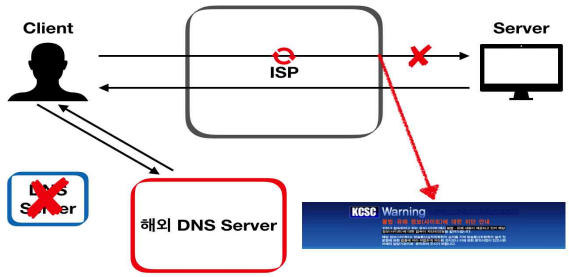


[그림 1] DNS 차단으로 리다이렉션

국내 인터넷 서비스 제공자(Internet Service Provider)가 운영하는 것이 아닌 구글 같은 해외 업체에서 제공하는 DNS를 이용하여 차단을 해결할 수 있다.

2.2 HTTP 차단

HTTP(HyperText Transfer Protocol)는 웹상에서 동작되는 프로토콜로 주로 HTML 문서를 주고받는 데에 쓰인다. HTTP 통신은 평문으로 수행되어 보안에 취약한 문제점을 가지고 있다. [그림 2]와 같이 DNS로부터 IP 주소를 수신한 웹브라우저가 국내 ISP(Internet Service Provider)에게 해당 주소로의 연결을 요청하는 순간, 이것이 불법 사이트 목록에 등재된 것일 경우 이를 차단하는 방식이다.



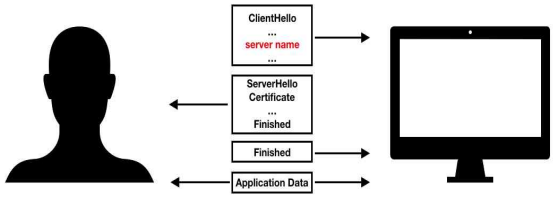
[그림 2] HTTP 차단으로 리다이렉션

TLS 프로토콜로 HTTP 통신을 암호화한 HTTPS를 통하여 클라이언트와 웹 서버 간 통신을 암호화하여 이를 해결할 수 있다.

2.3 HTTPS(SNI) 차단

SNI(Server Name Indication)는 TLS 프로토콜의 확장이며 이를 이용하여 동일한 IP 주소에 여러 개의 인증서를 사용할 수 있다. 따라서 동일 IP의 여러 호스트 인증이 가능하게 된다.

HTTP 통신의 암호화를 위해 TLS를 사용하는데, 이 경우 인증이 완료된 클라이언트와 웹 서버 사이의 통신은 암호화되는 반면 인증 과정에서의 통신은 암호화되지 않고 진행된다. [그림 3]과 같이 인증 과정 초기에 SNI 패킷을 주고받는데, 클라이언트와 웹 서버 간의 TCP+TLS Handshake 과정에서 아직 암호화 키를 전달받기 전 이기 때문에 ClientHello에서 SNI는 암호화되지 않은 평문으로 전달된다.



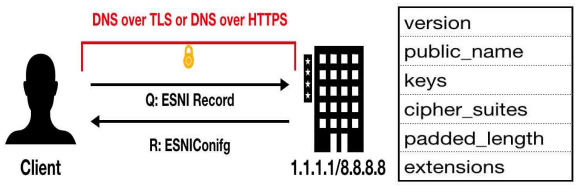
[그림 3] 평문으로 전달되는 SNI

SNI 정보는 네트워크상의 트래픽 모니터링은 물론 장애 문제를 해결하고자 하는 네트워크 관리자에게 매우 중요한 도구이며 네트워크에 가시성을 제공한다. 하지만 SNI 패킷에 포함된 도메인 정보를 확인하여 특정 웹사이트를 차단하는 것이 가능하다. 이로 인해 제 삼자에게 쉽게 노출이 되어 보안 문제가 생길 수 있다[2]. 다시 말해서, SNI 차단 방식은 HTTPS 통신을 하더라도 암호화가 시작되기 전에 SNI 필드에 호스트 이름이 평문으로 표시되는 것을 이용하여 ISP가 웹사이트를 차단하는 방식이다.

III. ESNI

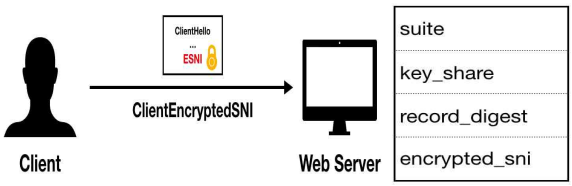
3.1 SNI 암호화를 위한 키 교환

ESNI는 TLS 1.3에서 제안된 SNI 암호화이다. 암호화를 하기 위해서 웹 서버는 신뢰할 수 있는 DNS 서버에 공개키를 미리 올려놓는다. 그리고 클라이언트가 DNS에 요청 시 TXT 레코드를 통해 ESNIConifg 구조체를 받는다. 이 구조체는 [그림 4]의 값들을 갖고 있다.



[그림 4] ESNIConfig 구조체

클라이언트는 웹 서버의 공개키와 자신의 공개키로부터 대칭키를 유도한다. 이 대칭키를 통해 [그림 5]의 ClientEncryptedSNI 구조체로 암호화한다. 그리고 클라이언트의 공개키와 함께 ClientHello 패킷을 통해 보낸다. 웹 서버는 자신의 개인키와 클라이언트로부터 받은 공개키를 사용하여 동일한 대칭키를 유도하여 ClientEncryptedSNI 구조체를 복호화한다. 여기서 대칭키 유도는 DH(Diffie-Hellman) 방식을 사용한다.



[그림 5] ClientEncryptedSNI 구조체

키 교환 시 고려해야 할 문제점은 HTTPS 기반의 웹 서버와 DNS 서버 상호 간에 동기화가 동시에 이루어져야 한다는 것이다. SNI 암호화에 사용되는 키의 만료기간으로 인한 갱신이 요구되는데 이러한 갱신이 이 두 서버에 동시에 반영되어야 한다. 일반적으로 웹 서버의 경우 ISP가 직접 관리하지만, DNS 서버는 ISP 제공 서버 혹은 외부의 네임서버를 사용하는 경우가 많기 때문에 어려움이 예상된다.

시험적으로 ESNI를 시행하고 있는 Cloudflare의 경우 웹 서버는 전방 익명성을 향상하기 위해 매 시간 키 갱신을 하며, 수 시간 동안의 키를 갖고 있는다[3]. 웹 서버는 키 갱신 문제를 감지하여 클라이언트에게 새롭게 갱신된 ESNI 레코드를 제공한다. 이때 기존의 ESNI를 완전히 비활성하지 않고 새로 갱신된 레코드를 통해 재연결하도록 한다.

하지만 ESNI(Encryption SNI)를 이용하고 평문으로 통신하는 DNS를 사용한다면 SNI를 암호화한 의미가 무색해진다. 중간 경로의 관찰자는 암호화된 SNI의 사용여부와는 관계없이 클라이언트가 보내는 평문 DNS 질의를 관찰하는 것만으로 사용자가 어떤 웹사이트에 접속하는지 알 수 있다.

즉 ESNI는 DNSSEC, DoT 또는 DoH와 같은 DNS 인증 및 암호화를 이용해야 인터넷상의 감시와 검열 문제 해결에 효과가 있다[4].

3.2 DNSSEC

DNSSEC(DNS Security Extension)은 DNS 데이터 대상의 “데이터 위조-변조 공격”을 방지하기 위한 인터넷 표준기술이다. DNS의 위조-변조 가능성을 원천적으로 차단하기 위해 DNSSEC의 공개키 암호화 방식의 전자서명 기술을 사용한다.

클라이언트가 얻은 DNS 정보가 실제로 맞는 정보인지에 대한 검증은 제공하지만, DNS 레코드는 물론 DNSSEC에 의해 서명된 레코드에 대해서는 암호화를 제공하지 않는다.

3.3 DNS-over-TLS

DoT(DNS-over-TLS)는 클라이언트와 DNS 서버 간의 TLS 프로토콜 기반 암호화된 통신을 통해 DNS 요청을 근본적으로 감청할 수 없게 만들 뿐만 아니라 중간자 공격을 차단함으로써 DNS 스푸핑 공격의 가능성을 줄인다.

3.4 DNS-over-HTTPS

DoH(DNS-over-HTTPS)는 DNS 요청을 평문이 아닌 HTTPS를 통해 DNS 정보에 접근한다. 이를 통하여 DoH는 DNS 요청을 HTTP 요청인 것처럼 위장하기 때문에 높은 보안성 효과 및 확장성을 제공한다.

3.5 DoT vs DoH

DoT는 TCP를 기본 연결 프로토콜로 사용하고 TLS 암호화 및 인증을 통해 계층화하는 반면 DoH는 HTTPS 사용하여 연결한다.

DoT는 자체 전용 포트인 853을 사용하며 DNS 요청이 암호화되기 때문에 내용을 알 수는 없지만 자체 전용 포트 번호로 인해 DoT를 사용하고 있다는 것을 알 수 있다. 반면, DoH는 기존 HTTPS 트래픽의 표준 포트인 443을 사용하므로 HTTPS 통신과 DoH 통신을 구분할 수 없다. 즉, DoT와는 다르게 DoH는 사용하고 있다는 것을 숨길 수 있다[5].

IV. 0-RTT

4.1 TLS 1.2 재연결

HTTP를 암호화하는 과정을 매번 Full Handshake를 하는 것은 비효율적이다. 그래서

이미 한번 과정을 거친 통신을 간소화하는 두 가지 방법으로 Session ID, Session Ticket이 있다.

Session ID와 Session Ticket 모두 초기 Full Handshake 과정에서 서버로부터 값을 할당받는다. 이 값으로 다음 Handshake 과정부터는 상호 인증을 간소화하여 수행할 수 있게 된다.

RFC 5077에 의하면 Session ID보다 Session Ticket의 사용을 더 권장한다. 그 이유는 Session ID는 서버가 발행한 값들을 모두 메모리에 기억해야 하지만 Session Ticket은 클라이언트가 해당 내용을 기억하고 서버로 전달하는 방식이라 메모리 자원에 영향이 적기 때문이다. 그러나 클라이언트나 서버 둘 중 하나라도 Session Ticket을 지원하지 않는 경우에는 Session ID를 사용하게 된다.

4.2 TLS 1.3의 0-RTT

TLS 1.3의 0-RTT의 시범적 도입으로 기존 방법보다 빠르게 처리할 수 있는 가능성을 제시했다. 0-RTT는 기존의 Session ID, Session Ticket과 같은 역할을 수행하는 PSK(Pre Shared Key)를 클라이언트의 ClientHello 단계에서 요청할 Application Data와 함께 미리 암호화를 미리한 후 전송하는 방식이다. 서버는 ClientHello에서 PSK를 확인하고 검증을 수행고 오류가 없다고 판단되면 ServerHello에 암호화된 Application Data를 전송하게 된다[7].

4.3 ESNI를 통한 0-RTT

ESNI 드래프트의 Open Issue에서 언급된 것으로 가능성을 확인해 본다. SNI를 암호화하기 위해 웹 서버의 공개키를 사전에 신뢰할 수 있는 DNS 서버에 올린다. 이 과정을 통해 클라이언트는 웹 서버와 통신하기 전 이미 자신과 웹 서버의 공개키를 DH(Diffie-Hellman) 방식을 통해 ESNI-PSK로 유도할 수 있다. 이는 기존의 TLS 1.3의 0-RTT 방식인 PSK를 DNS 프로토콜을 통해 더욱 이전 과정에서 교환한 것이다. 그러므로 기존 PSK를 통한 0-RTT를 더 이른 과정에서 초기 Handshake 과정 없이 수행할 수 있다는 가능성이다[6].

V. 결론

본 논문에서는 인터넷 차단 방식에 대해서 과거부터 현재까지 살펴보았고, SNI의 암호화 방법을 알아보았다. 또한 ESNI와 DoH, DoT, DNSSEC의 조합을 통하여 통신의 전체적인 암호화에 도달 할 수 있으며, 기존의 TLS 1.3의 0-RTT를 Open Issue에서 언급한 ESNI를 통하여 초기 과정 없이 사용할 수 있는 가능성을 확인했다.

[참고문헌]

[1] 김평수. (2019). 안전한 웹사이트 접속을 위한 IETF 표준 기술 동향 분석. 한국통신학회지(정보와통신), 36(6), 32-40.

[2] A. Ghedini, Encrypt it or lose it: how encrypted SNI works, Cloudflare, September 2018.

[3] A. Ghedini, Encrypt that SNI: Firefox edition, Cloudflare, October 2018.

[4] A. Ghedini, Encrypt it or lose it: how encrypted SNI works, Cloudflare, September 2018.

[5] P. Nohe, What is the difference between DNS over TLS & DNS over HTTPS?, The TLS Store, December 2018.

[6] E. Rescorla RTFM, Inc. K. Oku Fastly N. Sullivan Cloudflare C.Wood Apple, Inc. Encryption Server Name Indication for TLS 1.3 draft-ietf-tls-esni-05, November 04, 2019.

[7] N.Sulliva, Introducing Zero Round Trip Time Resumption(0-RTT), Cloudflare, March. 2017.

리눅스 오픈소스 바이너리 프로텍터 분석

문현준*, 송준환*, 윤주범†

*세종대학교 정보보호학과

soos5152@naver.com, wnsghks30@naver.com, jbyun@sejong.ac.kr

Analysis of Linux Open Source Binary Protectors

Hyunjun Mun*, Joonhwan Song*, Joobeom Yun†

*Department of computer and information security, Sejong University.

요 약

소프트웨어를 난독화, 암호화 하는 것은 중요기술의 복제를 방지하고 디지털 저작권을 지키기 위해 필수적인 기술이다. 이 기능을 수행하는 리눅스 오픈소스 바이너리 프로텍터의 몇 가지 예제를 살펴보고 각 프로텍터가 사용하는 암호화 알고리즘을 분석한다. 그리고 각 프로텍터의 원리, 기능, 특징 등을 살펴보고 비교분석을 실시한다. 본 논문에서는 이 비교분석을 기반으로 사용자에게 적절한 리눅스 바이너리 프로텍터 선택 기준을 제시한다.

I. 서론

디지털 산업과 사이버 보안 분야에서 소프트웨어를 난독화, 암호화 하는 것은 중요기술의 복제를 방지하고 디지털 저작권을 지키기 위해 필수적인 기술이다. 일반적으로 바이너리 파일을 난독화, 암호화하여 분석하기 어렵게 만드는 기술을 소프트웨어 보호 스키마(binary protection schemes)라고 하는데[1], 이를 위한 도구가 바이너리 프로텍터(binary protector)이다. 대표적으로 실행파일 압축에 사용되는 패커(packer) 또한 바이너리 프로텍터로 사용할 수 있다.

패킹된 실행파일은 용량 감소의 기능뿐만이 아니라 바이너리를 암호화 해 소스 코드와 리소스를 숨길 수 있다. 이러한 특성을 이용하여 패커를 기초적인 바이너리 프로텍터로서 사용할 수 있고, 일반적인 바이너리 프로텍터는 코드 난독화나 안티시뮬레이션 등의 리버싱 기법을 추가로 사용한다.

본 논문은 바이너리 프로텍터가 지녀야 할 기본적인 요소들에 대해 설명하고, 몇 가지 리눅스 바이너리 프로텍터의 암호화 알고리즘을 비교분석한다. 각각의 바이너리 프로텍터들의 기능상의 특징에 대해 분석하여 목적에 맞는 적절한 바이너리 프로텍터를 선택하는데 도움이 되고자 한다. 전체 구성은 다음과 같다. 2장은 바이너리 보호를 위한 몇 가지 암호화 기법을 소개한다. 3장은 각 암호화 기법에 따른 대표적인 도구를 소개한다. 4장은 3장에서 소개한 도구들을 비교해 각 방식의 장단점에 대해 설명한다. 5장 결론은 목적에 맞는 바이너리 프로텍터의 선택에 관한 가이드를 제시하였다.

II. 리눅스 바이너리 보호 기법

2.1 UCL 암호화

UCL은 ANSI C 로 작성된 데이터 압축 라이브러리다.[2] 이 라이브러리는 오픈소스이며 GNU GPL 조건하에 재생산과 배포가

가능하다. 본래 암호화 기법이 아니라 데이터 압축 알고리즘이나, 서론에서 언급했듯이 데이터를 압축시킬 경우 데이터의 리소스를 숨길 수 있기 때문에 암호화의 기능 또한 수행할 수 있다. UCL은 무손실 압축을 지원하며 높은 압축률을 보여준다. 본래 암호화 알고리즘이 아니기 때문에 key가 필요하지 않으며 압축해제(decompress)가 가능하다. UCL 라이브러리를 이용한 바이너리 패커 툴로 UPX[3]가 존재한다.

2.2 XOR 암호화

XOR연산은 AND연산과 OR연산과는 다른 독특한 패턴을 가지고 있다. XOR연산을 실행할 경우 [그림 1]와 같이 기존의 값을 유추하기 힘든 값이 나온다. XOR연산의 또 다른 특징은 XOR 연산된 값에서 한 번 더 XOR 연산을 실행할 경우 연산을 실행하기 전 본래의 값이 나온다는 것이다. 이를 암호화에 적용하면 간단한 XOR 연산만 이용해 쉽고 빠르게 암호화와 복호화 기능을 수행할 수 있다. 사용자에게 key값을 입력받고 이 key을 이용해 XOR연산을 실행해 암호화를 한다. 그 후 사용자가 원문이 필요한 경우 다시 key값으로 XOR 연산을 실행해 복호화 작업을 수행할 수 있다. XOR암호화를 활용한 오픈소스 바이너리 프로텍터 툴로 Sherlocked[4]와 Pocrypt[5]가 존재한다.



[그림 1] OR, AND, XOR 연산 비교

2.3 RC4 암호화

RC4 암호화 기법은 대칭키 암호화 기법 중 스트림 암호에 속한다. 이 기법은 암호화 알고리즘이 단순하기 때문에 구현이 용이하고 속도가 빠르지만 암호화 강도는 약한 편이다. RC4 또한 2.2의 XOR연산을 수행하지만 셔플링 기법을 사용하여 의사 난수를 생성한다는 차이점이 존재한다. key값을 입력받고 이를 기반으로 완전한 난수는 아니지만 이와 유사한 의사난수를 생성한 다음 평문과 바이트 단위로 XOR 연산을 수행하여 암호문을 생성한다. 복호화를 수행할 시 마찬가지로 XOR 연산을 수행하여 본래의 평문으로 복원한다. RC4 암호화를 활용한 오픈소스 바이너리 프로텍터 툴로 ELFcrypt2[6]가 존재한다.

III. 바이너리 프로텍터

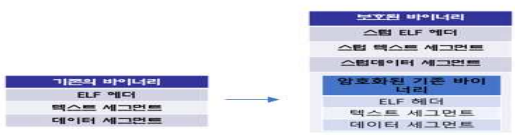
이 장에서는 앞서 소개한 알고리즘들을 기반으로 한 실제 바이너리 프로텍터들이 어떻게 동작하는지 설명한다.

3.1 UPX

UPX는 UCL 압축 알고리즘을 사용한 패커이다. 원본소스를 압축해 저장하고, 이를 실행 시 메모리에서 압축 해제해 실행한다. 기본적으로 실행파일을 압축해 배포를 쉽게 하고, 분석을 어렵게 하는 패커의 기본적인 기능을 가지고 있다. 그러나 바이너리 프로텍터에서 지원하는 안티 리버싱 기능들은 지원하지 않기 때문에 프로텍터로서는 거의 사용되지 않고, 패커의 장점을 살려 악성코드 배포에 많이 사용된다.

3.2 Sherlocked

Sherlocked는 XOR 암호화 방식을 사용한 바이너리 프로텍터이다. 사용자로부터 key를 입력받아 파일을 XOR 암호화하며, 결과를 x86-64 ELF Excutable 파일로 저장한다. 이때 복호화를 위한 stub_shellcode를 결과 파일에 저장한다. 여기서 stub(이하 스텁)이란 “메모리에서 암호화를 해제하고, 이어 해제된 바이너리를 실행시키는 것만이 목적인 단순한 런타임 엔진”[1]을 의미한다. 이 때문에 결과 파일은 기존 파일보다 용량이 커지게 되며, 추가된 스텁과 XOR 암호화로 인해 기존 파일의 소스코드와 리소스는 숨겨지게 된다. [그림 2]은 스텁된 파일의 구조를 보여준다.



[그림 2] 스텁된 실행 파일 구조 [1]

스텁은 암호화된 파일을 복호화 하고, 프로그램을 실행하는데 필요한 작업(동적링커 매핑, 매개인자 전달 등)을 한 뒤 프로그램을 실행 시킨다. Sherlocked에서는 /tmp 경로에 일시적으로 복호화 된 프로그램을 쓰고, 실행 종료 후 해당 프로그램을 /tmp에서 삭제한다. 위와 같은 방식으로 암호화된 내용을 보호하는 동시에 프로그램을 실행할 수 있다. 또 Sherlocked는 UPX가 설치되어 있을 시 암호화한 파일을 자동으로 패킹해 2단계의 보호 레이어를 제공한다.

3.3 Pocrypt

Pocrypt는 Sherlocked와 마찬가지로 사용자로부터 key를 입력받아 XOR암호화를 사용하지만, 프로그램 전체가 아닌 함수를 대상으로 암호화를 한다. 암호화할 대상 파일에 pocrypt의 소스코드를 포함시키고, 암호화할 함수를 지정해 코드를 작성하면, pocrypt에서 제공하는 암호화 모듈인 pocrypter를 이용해 암호화 할 때 해당 함수가 암호화 된다. 여기서 암호화 한 함수는 따로 임의의 ELF section에 저장된다.

```
./pocrypter PASSWORD .secure ./target
```

[그림 3] pocrypter로 target 암호화

[그림 3]은 pocrypt를 사용한 target 프로그램의 암호화 방법이며, 인자는 순서대로 key, 암호화 한 함수를 저장할 ELF section 이름, 대상 프로그램이다.

```
[14] .text             PROGBITS         0000000000000680  00000680
     0000000000000242  0000000000000000  AX       0     0     16
[15] .fini             PROGBITS         00000000000008c4  000008c4
     0000000000000009  0000000000000000  AX       0     0     4
```

[그림 4] 일반적인 파일의 Section header

```
[14] .text             PROGBITS         0000000000000a70  00000a70
     000000000024a71f  0000000000000000  AX       0     0     16
[15] .secure           PROGBITS         000000000024b18f  0024b18f
     0000000000000032  0000000000000000  AX       0     0     1
[16] .fini             PROGBITS         000000000024b1c4  0024b1c4
     0000000000000009  0000000000000000  AX       0     0     4
```

[그림 5] pocrypt로 암호화 한 파일의 Section header

[그림 4]는 일반적인 파일의 section header이고, [그림 5]는 pocrypt로 암호화한 파일의 section header이다. 기존에 없던 .secure section이 생긴 것을 확인할 수 있는데, 이곳에 암호화 된 함수가 저장되어 있다. 사용자는 암호화에 사용한 key를 사용해야만 암호화된 함수에 접근할 수 있다.

3.4 ELFcrypt2

ELFcrypt2는 RC4 암호화 알고리즘을 사용하며, 사용자로부터 key를 입력받아 Sherlocked와 같이 전체 파일을 암호화한다. 그러나 상술한 프로텍터들과는 달리 스텁 기능을 암호화 한 파일에 내장하지 않고 따로 스텁 전용 프로그램을 제공한다. 이는 사용의 빈도가 적고, 저장장치 내에 암호화해 보관할 프로그램에 더욱 적합한 방식이다.

IV. 실험결과

각 바이너리 프로텍터들의 암호화 알고리즘, 용량 변화, readelf를 통한 ELF reading, 키 요구 유무, 그리고 각 툴이 핵심적으로 제공하는 기능을 Table 1로 정리했다.

툴 이름	암호화 알고리즘	용량 변화	ELF Header Reading	ELF Section Reading	키 요구	핵심 기능	스텁 제공
UPX	UCL	감소	O	X	X	압축	내부 스텁
Sherlocked	XOR	증가	O	X	O	파일 전체 보호 2단계 레이어	내부 스텁
Pocrypt	XOR	X	O	O	O	함수 보호	내부 스텁
ELFcrypt2	RC4	X	X	X	O	파일 전체 보호	외부 스텁

[Table 1] 바이너리 프로텍터 비교

UPX는 실행파일을 압축하기 때문에 용량이 감소한다. [그림 6]는 파일의 코드와 데이터를 담고 있는 ELF Section header가 암호화 되어 감춰짐을 보여 준다.

```
root@ubuntu:/home/user/ELFcrypt# readelf -S test_upx

There are no sections in this file.
```

[그림 6] UPX가 적용된 파일의 Section header reading

그러나 UPX로 압축한 파일에는 별도의 안티 리버싱 기술이 적용되지 않아 바이너리 프로텍터로서는 조금 부족하다.

```
root@ubuntu:/home/user/sherlocked# readelf -S target_sh

There are no sections in this file.
```

[그림 7] Sherlocked가 적용된 파일의 Section header reading

Sherlocked는 스텁을 위한 stub_shellcode가 암호화된 파일에 내장되기 때문에 암호화된 파일의 용량은 증가하지만 [그림 7]처럼 ELF Section header 영역을 확실하게 암호화 하고 UPX로 한 번 더 압축해 2단계의 보호 레이어를 제공한다. 실행 후 메모리상의 바이너리를 삭제하므로 안티 리버싱적 특징 또한 가지고 있다. 즉 바이너리 프로텍터로서 기본적인 기능들을 갖추었다고 할 수 있다.

Pocrypt는 다른 도구들과는 달리 지정된 함수들만을 보호한다. 때문에 [그림 5]와 같이 ELF Section header 영역의 코드와 데이터들을 암호화하지는 않으며, 특정 함수의 대한 보호에 초점을 맞추고 있다.

```
root@ubuntu:/home/user/ELFcrypt# readelf -h target_ec_crypted
readelf: Error: target_ec_crypted: Failed to read file's magic number
```

[그림 8] ELFcrypt2 ELF header reading

[그림 8]은 ELFcrypt2를 적용한 파일의 ELF header reading 결과이다. 상술한 다른 프로텍터들과 달리 ELF header까지 암호화하여 readelf로 읽을 수 없기 때문에 Section header영역 또한 읽을 수 없다. 스텁을 암호화 파일에 내장하는 방식이 아닌 별도로 제공하기 때문에 용량변화가 없다. 이러한 방식으로 인해 자주 사용할 파일에는 부적절하지만 저장장치 내에 암호화 하여 파일을 보관하여 해커나 악의적 사용자로부터의 접근에 대비할 수 있는 장점을 가진다.

V. 결론

바이너리 프로텍터는 대상 파일의 소스코드 및 데이터를 암호화하여 보호하고, 메모리상에서 정상적으로 작동할 수 있게 하는 스텁 기능이 필수적이다. UPX는 바이너리 프로텍터로 기능할 수 있는 최소한의 조건을 갖추었다. 허나 본래 프로텍터가 아니라 패커의 용도로 개발되었기 때문에 프로그램을 보호할 기능이 부족하다. 그러므로 프로텍터보다는 패커로서 사용하는 것을 권장한다. 기본적으로 바이너리 프로텍터를 사용한다면 실행 후 메모리 삭제와 2단계 레이어를 가진 Sherlocked가 적합하다. 이를 기반으로 난독화나 안티 디버깅, 안티 에뮬레이션등의 기술을 추가할 수 있다. 특정 함수를 보호하고 사용을 통제하고 싶다면 pocrypt통한 함수 암호화 기능이 유용하다. 마지막으로, 저장장치 내에 보관할 파일에 대한 악의적인 접근을 막기 위해서는 별도의 스텁을 제공하는 ELFcrypt2가 효과적이다.

[참고문헌]

[1] Ryan “elfmaster” O’Neill, 리눅스 바이너리 분석, (에이콘 출판사, 2016), pp.171~174

[2] “UCL data compression library”, oberhumer.com, 2014년 7월 20일 수정, 2019년 10월 29일 접속, http://www.oberhumer.com/opensource/ucl

[3] “UPX: the Ultimate Packer for eXecutables”, github, 2018년 8월 26일 수정, 2019년 10월 29일 접속, https://upx.github.io/

[4] “elfmaster/sherlocked: Universal script packer-- transforms any type of script into a protected ELF executable, encrypted with anti-debugging.”, github, 2014년 10월 31일 수정, 2019년 10월 29일 접속, https://github.com/elfmaster/sherlocked

[5] “PoCrypt. A Proof Of Concept For Dynamically Decrypting Linux Binaries”, DMO Blog Tempus Fugit, 2016년 1월 26일 수정, 2019년 10월 29일 접속, http://papermint-designs.com/dmo-blog/2016-01-pocrypt-a-proof-of-concept-for-dynamically-decrypt-linux-binaries

[6] “droberson/ELFcrypt: Simple ELF crypter. Uses RC4 encryption”, github, 2018년 5월 26일 수정, 2019년 10월 29일 접속, https://github.com/droberson/ELFcrypt

시스템 호출 데이터를 활용한 비정상 행위 탐지 연구

김요한*, 하희리*, 김현준*, 백윤흥*,
*서울대학교 전기 정보공학부, 반도체 공동 연구소

A Study of anomaly detection techniques using system call data

Yohan Kim*, Whoi Ree Ha*, Hyunjun Kim* and Yunheung Paek*
*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC),
Seoul National University

Abstract

In recent years, the research on anomaly detection showed much improvement in its both performance and efficiency. As the name suggests, an anomaly detection is dedicated to detecting any abnormal behavior present in the system. Therefore, defining the normal state is the most important task. System call appeared to be a suitable unit of action that the system takes. Researchers have used system call data to determine if the system is operating out of the norm. In this paper, we explore how the sequence of system call data can be interpreted by deep learning model, long short term memory (LSTM) models and how the current technique can be improved by adding the system call argument data to the input vector.

I. Introduction

In the early era of system security field, detecting an intrusion to the system was mainly handled by pre-defined rules by human experts. Such rules can range from a certain event occurred in the system to an exploited vulnerability. However, this approach had an innate shortcomings as the effectiveness of the defense mechanism was bound by the human capability. On top of this, the attackers will aim to modify the attack scenario to bypass the system defense mechanism. Such attacks that are not yet revealed to the public are called zero-day attacks. In order to cope with these types of attacks, it is important to not rely only on rules and heuristics but generalize the normal state. Once the normal state is defined, by employing an anomaly detection, the system can recognize the attacks. System call sequence can be used to train LSTM to model the norm. In this paper, the methodology of such technique will be introduced and how it can be improved by adding the argument of the system call to the input vector.

II. Background

2.1 System call

The operating system operates in two separate regions, kernel mode and user

mode. System call allows users to make use of all functions in kernel mode. In other words, a process can use functions by directly accessing the hardware. There are few different types of system call. First is process control which includes create process and execute. Other types are as follow: file manipulation, device management, information maintenance and communication. Figure 1 shows few actually system calls used in the system. Each system call is assigned with an integer number.

%eax	Name	Source	%ebx	%ecx	%edx
1	sys_exit	kernel/exit.c	int	-	-
2	sys_fork	arch/i386/kernel/process.c	struct pt_regs	-	-
3	sys_read	fs/read_write.c	unsigned int	char *	size_t
4	sys_write	fs/read_write.c	unsigned int	const char *	size_t
5	sys_open	fs/open.c	const char *	int	int
6	sys_close	fs/open.c	unsigned int	-	-
7	sys_waitpid	kernel/exit.c	pid_t	unsigned int *	int
8	sys_creat	fs/open.c	const char *	int	-
9	sys_link	fs/namei.c	const char *	const char *	-
10	sys_unlink	fs/namei.c	const char *	-	-
11	sys_execve	arch/i386/kernel/process.c	struct pt_regs	-	-
12	sys_chdir	fs/open.c	const char *	-	-
13	sys_time	kernel/time.c	int *	-	-

Figure 1. System call table

2.2 LSTM

LSTM is a special type of recurrent neural network which was design to handle a training that requires a memory. The main difference that distinguishes LSTM from a regular RNN is that it contains four interactive gates. Those four gates are called, forget gate layer, input gate layer, cell state update and output gate layer. By being able to configure each gate individually LSTM thrives in a task where a sequence of data is provided. As system call data is given in sequential manner LSTM was the most suitable choice. Firgure 2 shows a simple version of LSTM inner structure.

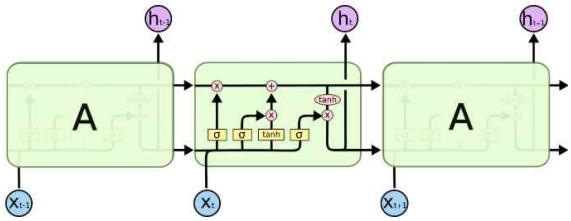


Figure 2. LSTM structure

III. Detection Model

3.1 System call sequence

In order to convert the system call sequence data into a appropriate vector form, all names were mapped to a integer value. The total number of system call was 312 which meant that the size of vector had to be at least that size.

3.2 System call argument

System call takes few different type of argument which includes int, float, char, and string. To simplify the data, only integer and string types of argument were considered and converted into a vector form before being concatenated into a single vector with sequence data.

3.3 Input Vector

Input vector to LSTM model will be a two fold: system call sequence vector and system call argument vector. It is possible to naively concatenate two vectors into a single vector with setting the size of concatenated vector to be the maximum size of either vector. The unknown element in the vector array will be embedded with a constant value, 0.

3.4 LSTM Model

As for a deep learning model, a simple and plain LSTM was implemented. Figure 3 illustrates the design choice for combined system call sequence and argument. Two different vector interpreters were used. Softmax layer was implemented for the sequential data and decoder part of auto-encoder was employed for argument data.

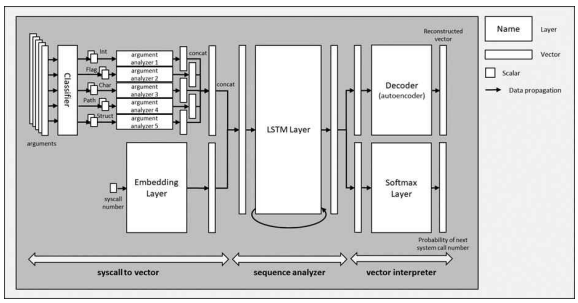


Figure 3. LSTM model

3.5 Decision Phase

Decision phase is where the detection scheme decides if an attack has been carried out. If the anomaly score of either one of types of input data is appeared to be over the threshold set, then, the flag is raised. A simple “or” gate can be implemented in between to judge the abnormality. The score is equivalent to the perplexity of LSTM model.

IV. Experiment

4.1 Normal data

The normal state is recorded by executing a routine task of mysql server. A mysql script that repeatedly generates tables and databases was run for a certain period of time. The system call data was recorded by using Intel processor trace which is embedded in intel CPU chips.

4.2 Attack data

A known attack proof of concept scenario from www.exploit.com was reproduced. CVE-2016-6663 which is related to mysql vulnerability exposed to the public was carried out. The system call sequence and argument were recorded the same way as the normal data.

V. Result

5.1 Detection

The anomaly detection system will raise a flag once it detects an abnormal activity in the protecting system. The threshold can be adjusted to accurately detect the attack. The value of threshold may vary between different types of attacks. We successfully detected the local privilege escalation attack by maneuvering the threshold value.

VI. Conclusion

In the flood of upcoming cyber security threat, the anomaly detection has never been as important. It is not sufficient to only protect the system from a known attacks but it is crucial to prepare for zero-day attacks. Anomaly detection technique is the most promising in such problem. In recent development, system call sequence and argument have been recognized as the input for deep learning model. On top of this, LSTM model was implemented for the purpose of interpreting a sequential data.

In the future, an experiment can be carried out with varying environment settings for the deeper insight.

VII. ACKNOWLEDGEMENT

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2018-0-00230, (IoT 총괄/1세부) IoT 디바이스 자율 신뢰보장 기술 및 글로벌 표준 기반 IoT 통합보안 오픈 플랫폼 기술개발 [TrusThingz 프로젝트])의 지원을 받아 수행된 연구이며 2019년도 두뇌한국21플러스사업에 의하여 지원되었음.

[References]

[1] Kang, D. K., Fuller, D., & Honavar, V. (2005, June). Learning classifiers for misuse and anomaly detection using a bag of system calls representation. In Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop (pp. 118-125). IEEE.

[2] Tandon, G., & Chan, P. K. (2003). Learning rules from system call arguments and sequences for anomaly detection.

[3] Eskin, E., Arnold, A., Prerau, M., Portnoy, L., & Stolfo, S. (2002). A geometric framework for unsupervised anomaly detection. In Applications of data mining in computer security (pp. 77-101). Springer, Boston, MA.

[4] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 15.

[5] Kruegel, C., Mutz, D., Valeur, F., & Vigna, G. (2003, October). On the detection of anomalous system call arguments. In European Symposium on Research in Computer Security (pp. 326-343). Springer, Berlin, Heidelberg.

[6] Xu, M., Chen, C., & Ying, J. (2004). Anomaly detection based on system call classification. Journal of Software, 15(3), 391-403.

Multi-Variant Execution Environment를 위한 데이터 난수화 기법

김현준*, 황동일*, 김지환*, 백윤흥*

*서울대학교 전기정보공학부, 반도체공동연구소

Data Randomization Technique for Multi-Variant Execution Environment

Hyun-Jun Kim*, Dong-Il Hwang*, Jee-Hwan Kim* and Yun-Heung Paek*

*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC), Seoul National University.

요 약

대부분의 임베디드 소프트웨어는 C 혹은 C++ 언어로 작성이 되어 있다. 따라서 이들에 대해 수많은 메모리 취약점들이 존재하며, 공격자들이 해당 컴퓨터 시스템에 침입하는 것을 가능케 한다. Multi-Variant Execution Environment (MVEE) 기법은 멀티코어 임베디드 프로세서를 활용하여 메모리 취약점을 이용한 공격을 효율적으로 방어한다. MVEE에 의해 제공되는 보안은 variants 간 난수화 정도에 따라 달라진다. 기존 MVEE에서는 일반적으로 variants 간의 데이터 레이아웃을 난수화하는데, 이 방식을 사용하면 메모리 에러와 관련된 몇 개의 특정 공격을 탐지하지 못한다. 본 논문에서는 데이터 공간 난수화 (DSR, Data Space Randomization)를 MVEE의 variants에 적용하여 MVEE에 의해 제공되는 보안을 강화하였다. 그리고 DSR로 인해 추가로 발생하는 오버헤드가 적당히 작음을 실험적으로 확인하였다.

I. 서론

임베디드 시스템상에서 C/C++로 작성된 소프트웨어는 많은 메모리 취약점을 가지고 있다. 이를 보완하기 위해 학술적/기업 차원에서 많은 연구가 진행되었지만, 여전히 이 취약점들은 컴퓨터 시스템의 보안을 위협하고 있다. 이러한 취약점을 찾아내기 위한 정적 분석 기법은 속도가 너무 느리며, 상용 소프트웨어에서 취약점을 잘 찾아내지 못한다는 단점을 가지고 있다. 이에 따라 실시간으로 특정 유형의 공격을 방어할 수 있는 동적 방어 기법들이 제안되었다. 이러한 방어 기법 중 메모리 안전을 보장하는 기법이 가장 강한 기법이라고 볼 수 있으며, 이를 위해서는 메모리 에러가 발생한 직후에 곧바로 공격을 탐지해야 한다. 하지만 동적 기법은 성능/메모리 오버헤드가 크며, 레거시 소프트웨어와 호환성 문제를 일으키기도 한다. 다른 기법으로는 제어 흐름 무결성 (CFI, Control Flow Integrity) 기법, ASLR 등이 있으며, 이들은 특정 공격만을 대상으로 하여 추가 성능 오버헤드를 최소화하였다. 하지만 이 기법들은 복잡한 공격이나 새로운 공격에 취약하다는 단점이 있다.

Multi-Variant Execution Environment (MVEE)[1] 는 성능과 보안을 동시에 잡을 수 있는 강력한 방어 기법이다. 여러 임베디드 컴퓨팅 시스템에 구성된 멀티코어 환경을 활용하여 다양한 프로그램 variants를 실행시키고, 이를 모니터링하여 악성 침입을 탐지하는 기법이다. 공격자들이 시스템에 악의적인 영향을 끼치기 위해 시스템 콜을 사용한다는 점을 이용하여 MVEE는 시스템 콜 boundary에서 variant 간에 차이가 발생하는지 모니터링한다. 각 variant의 데이터 레이아웃을 난수화하면, variant들은 정상적인 상태에서는 동일한 행위를 수행하지만, 공격이 발생하면 서로 다른 행위를 수행한다.

MVEE가 보장하는 보안은 이 variant들이 얼마나 다양화(diversify)되어있는가에 따라 결정된다. MVEE의 기본적인 설정은 스택/힙/글로벌 영역을 랜덤 위치에 두는 것이다. 공격자가 어떤 데이터를 특정 메모리 영역에 쓰려고 하거나, 제어 흐름을 특정 코드 메모리로 우회하는 것을 시도하면, variant 중 하나에서 프로그램 충돌이 발생할 가능성이 매우 크며, MVEE가 공격 시도를 탐지할 수 있게 된다. 하지만 공격자는 상대 메모리 주소를 사용해서 variant들이 동일한 행위를 하게 하여 탐지를 회피할 수 있다. 이 문제를 해결하기 위해 variant들에 대해 좀 더 fine-grain한 난수화 기법을 적용해서 객체 간 거리도 난수화시켜야 한다. 하지만 모든 객체의 위치를 난수화시키는 것은 현실적으로 불가능하다.

이 문제를 해결하기 위해서 본 논문에서는 DSR-MVEE를 제안하고자 한다. 이 기법의 메인 아이디어는 데이터 공간 난수화 (DSR)[2]를 통해 MVEE의 variant들을 난수화시키는 것이다. DSR은 데이터 레이아웃을 난수화하는 것이 아니라 데이터 그 자체를 난수화시킨다. 즉, 메모리의 모든 객체를 유니크 키 (unique key)로 암호화한다. 연속된 위치의 객체들이 다른 키로 암호화되기 때문에 공격자가 두 객체의 키들을 모두 알아내지 않는 이상 메모리 취약점을 공격할 수 없다. 만약 DSR을 MVEE의 variant마다 다른 난수화 시드로 초기화하게 되면, 객체 간 상대적 거리가 동일하더라도 메모리 에러를 기반으로 한 공격을 MVEE 모니터로 감지할 수 있다. SPEC benchmark를 사용한 실험을 통해 MVEE 시스템에 DSR을 적용하여 큰 성능 오버헤드 추가 없이 보안을 강화할 수 있음을 보였다.

II. DSR-MVEE

공격자들은 일반적으로 C/C++ 프로그램의 메모리 취약점을 활용하여 메모리에 있는 시스템 콜 파라미터를 변형하며, 시스템에서의 악성 행위를 야기한다. 메모리 취약점은 공격자가 타깃 객체의 특정 필드를 의도되지 않은 메모리 명령어로 덮어씌울 수 있게 한다. 가장 많이 알려진 취약점으로 버퍼 오버플로우와 use-after-free 취약점이 있다. MVEE는 난수화된 데이터 레이아웃을 가진 variants를 실행시킨다. 공격자가 이 variants 중 한 variant의 타깃 객체를 변형시키려고 하면, 결과적으로 다른 variant의 완전히 다른 객체를 변형하게 된다. 이로 인해 variants 중 하나가 충돌하게 되고, 공격 탐지가 가능하다. 충돌이 일어나지 않더라도 공격자가 원하는 방향으로 시스템을 변경하지 못하게 한다.

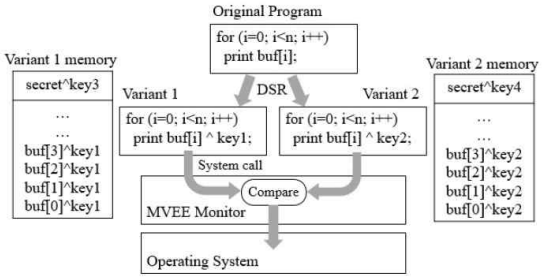


그림 1. DSR을 적용한 MVEE 아키텍쳐

[그림 1]에서 MVEE와 DSR 기반 variants가 애플리케이션에서 secret을 추출하려는 시도를 어떻게 막을 수 있는지 알 수 있다. 프로그램에 secret이 있다고 가정하고, 메모리 영역 상으로 바로 옆에 버퍼가 있다고 가정하자. 원본 프로그램은 for 루프에서 버퍼를 연속으로 읽는다. 만약 n이 버퍼의 크기보다 클 경우 버퍼 오버플로우 취약점이 발생하여 공격자가 프로그램에서 secret 정보를 추출할 수 있게 된다. DSR을 이 프로그램에 적용한다면 secret과 buf를 각각 유니크 키로 XOR 연산을 통해 암호화하게 된다. [그림 1]에서 Variant 1과 Variant 2의 메모리 영역을 확인할 수 있으며, 각각 다른 키를 통해 암호화되어 있다. 이 상태에서 buf를 읽을 경우 buf에 해당하는 키와 XOR 연산을 수행하여 buf의 기존 값을 출력하게 된다. 따라서 print 시스템 콜이 불리게 되었을 때, 두 variants 모두 buf의 값을 시스템 콜 파라미터로 넣어주게 된다. MVEE는 두 variant의 파라미터 값들이 동일함을 확인한 다음, 시스템 콜을 실행시킨다. 하지만 만약 버퍼 오버플로우

발생이 발생하여 secret 값을 읽을 경우, 원래 값 대신 Variant 1에서는 secret^key3^key1이, Variant 2에서는 secret^key2^key4가 읽히게 된다. 그리고 결과적으로 두 variants에 대해 system call에 들어가는 파라미터 값이 다르게 된다. MVEE는 이를 통해 공격을 탐지하고 프로그램을 중단시켜 추가적인 시스템 손상을 방지한다. (이 경우에는 secrete 값의 유출을 방지한다.) 이 기법을 사용하면 모든 객체를 다른 키로 암호화하기 때문에 난수화 granularity가 문제시되지 않는다. 객체 사이에서 발생하는 메모리 에러는 variant 사이에서 다른 값을 읽거나 쓰기 때문에 MVEE로 바로 탐지가 가능하다.

III. 실험 결과

DSR-MVEE의 성능을 확인하기 위해, DSR-MVEE를 직접 구현한 다음, SPEC2060 벤치마크의 일부 프로그램을 통해 성능 오버헤드를 검증하였다. MVEE 플랫폼으로 x86 오픈소스 MVEE인 ReMon[3] 을 사용하였다. 그리고 context-sensitive한 DSR 기법[5] 을 사용하였다. ReMon를 기반으로 DSR을 사용해서 variants를 만들도록 코드를 수정하였다. 그리고 DSR 기법을 사용하지 않는 기존 MVEE와 성능을 비교하였다.

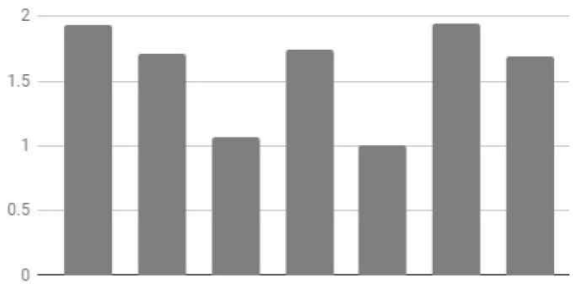


그림 2. DSR-MVEE의 성능 오버헤드

[그림 2]는 본 실험의 결과를 나타내고 있다. DSR를 적용함으로서 발생한 geomean 오버헤드는 68%이다. 데이터 암호화를 위해 XOR 연산을 코드에 instrument 했기 때문에 약간의 오버헤드가 발생하였다. 이는 MVEE의 보안 성능에 대비하여 충분히 작은 오버헤드이다.

IV. 결론

본 논문에서는 DSR를 통해 MVEE variants 들의 데이터를 난수화하는 기법을 제안하였다. 이를 통해 객체 사이에서 발생하는 메모리 취약점에 대한 공격을 효과적으로 방어하여 MVEE의 보안 성능을 크게 향상시켰다. 이 기법을 사용하여 임베디드 시스템의 security critical 어플리케이션에 적은 오버헤드로 강력한 보안을 제공할 수 있다.

V. ACKNOWLEDGEMENT

이 논문은 2019년도 두뇌한국21플러스사업에 의하여 지원되었으며, 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00213, 능동적 사전보안을 위한 사이버 자가변이 기술 개발)

[참고문헌]

[1] Volckaert, Stijn, et al. "GHUMVEE: efficient, effective, and flexible replication." International Symposium on Foundations and Practice of Security. Springer, Berlin, Heidelberg, 2012.

[2] Bhatkar, Sandeep, and R. Sekar. "Data space randomization." International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, Berlin, Heidelberg, 2008.

[3] Volckaert, Stijn, et al. "Secure and efficient application monitoring and replication." 2016 {USENIX} Annual Technical Conference ({USENIX}{ATC} 16). 2016.

[4] Hwang, Shin, et al. "Data Randomization for Multi-Variant Execution Environment." International SoC Design Conference (ISOCC) 2019, Jeju,

2019.

[5] Belleville, Brian, et al. "Hardware assisted randomization of data." International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Cham, 2018.

효율적인 Multi-Variant Execution Environment에 대한 연구

장지원*, 김지환*, 황동일*, 신장섭*, 백윤흥*,

*서울대학교 전기·정보공학부, 반도체공동연구소

A Study on Efficient Multi-Variant Execution Environment

Jiwon Chang*, Jeehwan Kim*, Dongil Hwang*, Jangseop Shin* and Yunheung Paek*

*Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center (ISRC),

Seoul National University

요 약

임베디드 시스템들은 주로 속도가 빠르고 호환성이 좋은 C나 C++ 언어를 사용하는데, 이들은 버퍼 오버플로우와 같은 메모리 취약성을 갖고 있다. Multi-Variant Execution Environment (MVEE)는 임베디드 시스템의 메모리 취약성을 이용한 공격에 대한 보안을 강화시키는 새로운 메커니즘이다. MVEE는 현대 멀티코어 프로세서를 사용하여 여러 variant들을 실행하고 이들을 활용하여 보안 상태를 확인한다. 현대 멀티코어 처리환경이 variant 프로세스들을 효율적으로 병렬실행 가능하게 하지만, 이전 연구들을 살펴보면 여전히 시스템 콜이 집약된 워크로드에서는 성능 오버헤드가 크다. 본 연구에서는 최근 오픈소스 MVEE를 활용해 성능 오버헤드에 영향을 미치는 요소를 ngnix 프로그램을 이용한 실험을 통해 알아보고자 한다.

I. 서론

코드 평가 회사인 TIOBE에서 프로그래밍 언어들의 대중성을 조사한 바에 따르면 2019년 10월 한 달간 사용된 프로그래밍 언어 중 C와 C++는 각각 16.18%, 6.23%의 점유율로 2위, 4위를 기록한 것을 알 수 있다.[1]

분명 C와 C++는 다른 프로그래밍언어들보다 실행 속도가 빠르고 호환성이 좋지만, 메모리 취약성이 존재하여 메모리 공격에 노출되어있다. 예를 들어, C와 C++는 프로그램에서 배열에 접근할 때, 할당된 메모리 범위 내의 인덱스에 올바르게 접근하는지에 대한 여부를 확인하지 않는다. 이러한 취약성을 이용해 공격자는 C나 C++로 구현된 프로그램 메모리 안에 값을 조작하거나 삽입할 수 있는 버퍼 오버플로우 (buffer overflow) 공격이 가능하다.

수행 속도와 전력 소모에 민감한 임베디드 시스템들은 주로 C나 C++ 언어로 작성되기 때문에, 앞서 말한 메모리 취약성을 띄게 된다. 메모리 취약성 기반의 공격자들로부터 시스템을 보호하기 위하여 Multi-Variant Execution Environment (MVEE) 기술이 개발되었다. 현대 멀티코어 환경을 이용하는 MVEE는 하나의 프로그램을 다각화 된 여러 개의 variant들로 복제하여 이들을 병렬로 동시에 수행한 뒤 행동을 비교해 공격자의 시스템 침입을 감시한다. 멀티코어 환경에서 MVEE는 효율적으로 높은 보안성을 제공하며 동작하지만, 최근 MVEE 관련 논문들에서 여전히 MVEE가 시스템 콜이 집약된 프로그램에서는 큰 성능 오버헤드를 나

타냄을 보여주고 있다.

본 논문에서는 최신 오픈소스 MVEE[3]의 동작 메커니즘을 분석하여 성능 오버헤드의 요인을 찾고, 이들의 다양한 조합으로 ngnix 프로그램을 이용한 성능 측정을 통해 MVEE의 성능 개선 방안을 제시하고자 한다.

II. MVEE 소개

Multi-variant execution environment (MVEE)의 개념은 Cox에 의해 2006년 최초로 개발되었다.[5] 처음 Cox가 제안한 N-variant 시스템은 그림 1과 같이 여러 개의 variant들과 폴리그래퍼, 모니터로 구성되어있다. 폴리그래퍼는 외부로부터 받은 입력 값을 variant들에게 전달하고, 여러 개의 variant들은 각자 프로그램을 수행한 후 모니터에게 출력 값을 전달한다. 모니터는 variant들의 행동을 비교하여 모두 일치할 경우에만 출력 값을 외부로 내보낸다. 이때 각 variant들의 메모리 주소 공간은 서로 다르게 분산되어 설계되어 있다. 이렇게 설계함으로써 메모리의 특정 절대주소를 참조하는 공격에 노출되었을 때, 하나의 variant에 대해서 참조 공격이 성공하여도 다른 variant들에서는 유효하지 않은 주소를 참조하게 된 것이므로 세그멘테이션 오류가 발생해 모니터가 variant들 간의 행동 불일치를 식별하여 공격을 탐지하게 된다.

최근 MVEE의 구현[2]을 살펴보면, 모니터는 분리된 프로세스로 구현되어있다. variant들은 리눅스 ptrace 인터페이스를 통하여 모니터에 부착되어, 시스템 콜이 variant에 의해 발생될 때 마다 커널은 모니터 프로세스를 깨워 발생된 시스템 콜을 검토하게 한다. 그러나 이 메커니즘은 모니터로 스위칭 하는 것이 context switch를 포함하기에 큰 성능 오버헤드가 발생하게 된다. 이러한 성능 오버헤드를 줄이기 위해, in-process 모니터를 기존 MVEE에 도입한 ReMon[3]이 제안되었다. in-process 모니터는 variant 프로세스 안에 포함되어 주소 공간을 공유한다. 이 점을 이용해 공격자가 in-process 모니터의 정보를 건들일 수 있기 때문에, in-process 모니터의 보안성은 cross-process 모니터의 보안성보다 낮아지게 된다. ReMon은 이러한 in-process 모니터의 단점을 보완하기 위해 시스템 콜을 보안 수준에 따라 여러 그룹으로 나누고, 오직 낮은 보안 수준의 시스템 콜만 in-process 모니터에 의해 다뤄지도록 한다. 이 메커니즘을 이용하여, ReMon은 여러 벤치마크에서 성공적으로 성능 오버헤드를 줄일 수 있었지만, 여전히 몇몇의 프로그램에서는 큰 오버헤드가 발생하고 있다.

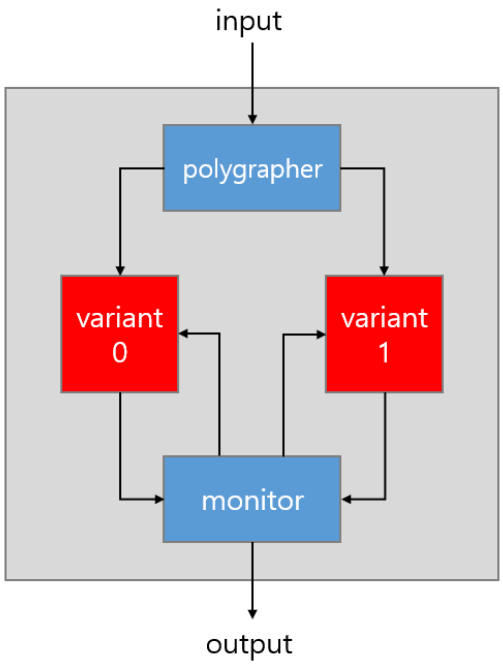


그림 1. N-variant System Framework

ReMon의 성능 오버헤드를 일으키는 요인을 알아보기 위해, 먼저 그림 2를 통해 ReMon의 전체적인 메커니즘을 분석하고자한다. variant 프로세스가 시스템 콜을 issue하게 되면(①) 그 제어권은 kernel broker에게 전달된다. 커널은 보안 정책을 보고 그 시스템 콜이 in-process 모니터에 의해 처리되어야 하는지(②), cross-process 모니터에 의해 처리되어야 하는지(②') 결정하게 된다. 결정된 모니터는 variant들의 시스템 콜 argument를 검증하여 시스템 콜을 처리할지(③) 여부를 판단한다. 필요에 따

라서, 모니터는 variant를 다른 variant들이 같은 argument의 같은 시스템 콜에 도달할 때까지 기다리게 할 수 있다. 이것을 시스템 콜의 lock-step 수행이라 한다. 다른 variant가 시스템 콜에 도달하기 전에, 공격자가 이미 하나의 variant를 공격해 시스템의 이상동작을 일으킬 수 있으므로 lock-step 수행은 security-critical한 시스템 콜에게 중요하다. 시스템 콜 이후 모니터는 모든 variant들에 대한 시스템 콜 결과들을 복제한다.(④, ④') 마지막으로, 제어권은 다시 기존 variant 프로세스에게 돌아간다.(⑤, ⑤')

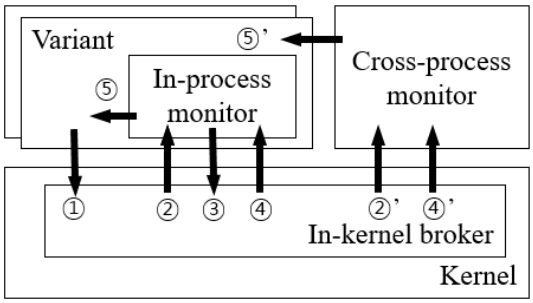


그림 2. Overview of ReMon

MVEE의 기본적인 동작과정을 알아보았으니, 이제 성능 오버헤드의 요인을 분석하고자 한다. 첫째로, 대부분의 시스템 콜들이 cross-process 모니터에 의해 다뤄지므로 security-critical한 시스템 콜이 많을 경우에 잦은 context switch로 인해 큰 성능 오버헤드가 발생하게 된다. 둘째로, 일부 security-critical한 시스템 콜들은 lock-step으로 수행이 되어서 variant의 수행을 빈번하게 중단하게 한다. 마지막으로, 각 시스템 콜에 대한 control flow는 유저와 커널 사이 경계를 2배로 왔다 갔다 하게 된다. 이러한 설계 디자인은 공격자가 모니터로부터 우회하려는 것을 막을 수 있다. 예를 들어, variant가 커널을 거치지 않고 in-process 모니터에게 제어권을 전달하게 되면, 공격자는 모니터를 우회하여 직접적으로 조작된 argument로 시스템 콜을 수행시킬 수 있게 된다. 시스템 콜이 다른 프로세스로 context switch를 하지 않는 경우에도 user context를 저장하고 복원하기 때문에 오버헤드가 발생한다. 결국 각 시스템 콜들은 사실상 2번 불리게 되므로 전체 성능 오버헤드에 영향을 주게 된다.

III. 실험 결과

앞서 말한 오버헤드 요인들에 의해 성능 오버헤드가 얼마나 발생되는지 정확하게 측정하는 것은 앞으로 MVEE 설계의 발전에 중요한 첫걸음이다. 실험 환경으로 MVEE에서 큰 오버헤드가 발생되는 nginx 웹 서버와 Intel Core i7-4790을 사용하였다. ReMon의 논문과 동일하게 nginx 1.5.12 version을 이용하였다. localhost에서 wrk http benchmark를 실행시켜 웹 서버에 의해 처리되는 throughput을 측정하였다. 먼저 MVEE가 동작하지 않는 상태에서 nginx를 실행하였고, 이후 여러 다른 MVEE 구성으로 nginx를 실행하였다. 그림 3에서 나타나는 웹 서버의 throughput은 MB/s 단위이다.

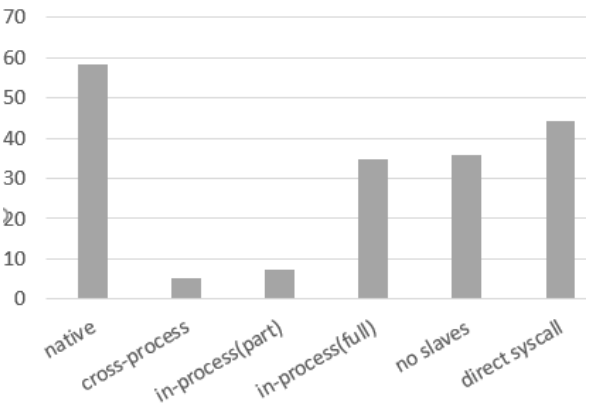


그림 3. Throughput by sources of overheads

첫 번째 native는 MVEE의 개입 없이 58.25 MB/s 의 throughput을 보여준다. http 요청이 존재할 때는 7개의 시스템 콜이 불리게 된다. 모든 시스템 콜들이 cross-process 모니터에 의해 다뤄지게 되면 throughput은 5.09 MB/s 로 급격히 감소하게 된다. throughput은 MVEE의 구성을 하나씩 변경할 때마다 증가한다. 5개의 시스템 콜만 in-process 모니터에 의해 다뤄진 경우 throughput 조금 증가하게 된다. 이에 따라, 모든 시스템 콜을 in-process 모니터에 의

해 다뤄지도록 하게 되면 throughput이 크게 증가하지만 여전히 native throughput의 60%정도 밖에 도달하지 못한다. 오직 하나의 master variant가 실행될 때는 lock-stepping이 필요하지 않고 master variant는 slave variant를 기다릴 필요 없이 실행된다. 하나의 master variant만 실행하는 것은 throughput을 미미하게 증가시킨다. 마지막으로, libc 라이브러리를 바꿔서 in-process 모니터가 kernel broker를 거치지 않고 직접적으로 불리도록 하였다. 이로 인해 throughput은 15% 증가하게 된다. native와 비교하였을 때 나머지 성능 오버헤드는 모니터 자체를 수행하는데 발생되는 오버헤드로 예상된다. 이 실험 결과로부터, (1) 시스템 콜을 cross-process 모니터로 모니터링 하는 것은 비효율적이라는 결론을 지을 수 있다. in-process 모니터의 보안성을 cross-process 모니터만큼 높여서, 모든 시스템 콜을 in-process 모니터에서 다뤄지게 하면 성능을 증가시킬 수 있다. (2) 또한, 시스템 콜을 2번 부르지 않고도 공격자가 in-process 모니터를 우회하지 못하도록 막는 방법을 고안한다면 MVEE의 성능이 향상될 것이다. (3) 모니터링의 동작을 최적화 시켜 나머지 성능 오버헤드를 줄일 수 있다.

IV. 결론

본 논문에서는 MVEE를 개선시킨 ReMon의 동작과정과 성능 오버헤드에 영향을 끼치는 요인들을 알아보았고, nginx 프로그램을 활용한 성능 측정을 수행하였다. 실험 결과를 통해, MVEE의 성능 오버헤드에 영향을 주는 3가지 문제점을 제시하였으며, 앞으로 이를 개선하는 연구가 활발히 진행되어 MVEE의 성능이 더욱 개선될 것이라 예상된다.

V. ACKNOWLEDGEMENT

이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00213, 능동적 사전보안을 위한 사이버 자가변이 기술 개발)

또한, 이 논문은 2019년도 두뇌한국21플러스 사업에 의하여 지원되었음.

[참고문헌]

[1] http://www.tiobe.com/tiobe-index/

[2] Volckaert, S., De Sutter, B., De Baets, T., & De Bosschere, K. (2012, October). GHUMVEE: efficient, effective, and flexible replication. In International Symposium on Foundations and Practice of Security (pp.261-277). Springer, Berlin, Heidelberg.

[3] Volckaert, S., Coppens, B., Voulimeneas, A., Homescu, A., Larsen, P., De Sutter, B., & Franz, M. (2016). Secure and efficient application monitoring and replication. In 2016 USENIX Annual Technical Conference (pp. 167-179).

[4] Kim, J., Hwang, D., Shin, J., Paek, Y., (2019). Towards Efficient Multi-Variant Execution Environment. In International SoC Desgin Conference (ISOCC).

[5] Cox, B., Evans, D., Filipi, A., Rowanhill, J., Hu, W., Davidson, J., Knight, J., Nguyen-Tuong, A., Hiser, J., (2006). N-variant systems: A secretless framework for security through diversity. In 2006 USENIX Security.

Power Delivery Chip 기반의 은닉 채널

정혜림*,안성규*,박기웅†

세종대학교 시스템 보안 연구실*, 세종대학교 정보보호학과†

Covert Channel using Power Delivery Chip

Hye-Lim Jeong*,Sung-Kyu Ahn*, Ki-Woong Park†

*Department of Computer and Information Security, Sejong University.

요 약

하드웨어 디바이스의 발전이 급증하면서 은닉 채널 기술의 연구도 함께 발전하였다. 본 논문에서는 디바이스의 충전 모듈인 Power Delivery Chip과 디바이스 간에 은닉 채널을 제안한다. Power Delivery Chip은 디바이스로 전력 양을 조절하여 보낼 수 있도록 재설계를 통해 디바이스로 은닉 채널 신호를 전송하게 된다. 디바이스는 해당 은닉 채널의 신호를 수신하도록 수신 기능을 수행하는 악성코드를 감염시킨다. 제안과 함께 Power Delivery Chip과 디바이스로 스마트폰의 안드로이드에서 본 제안의 가능성을 설명하였다.

I. 서론

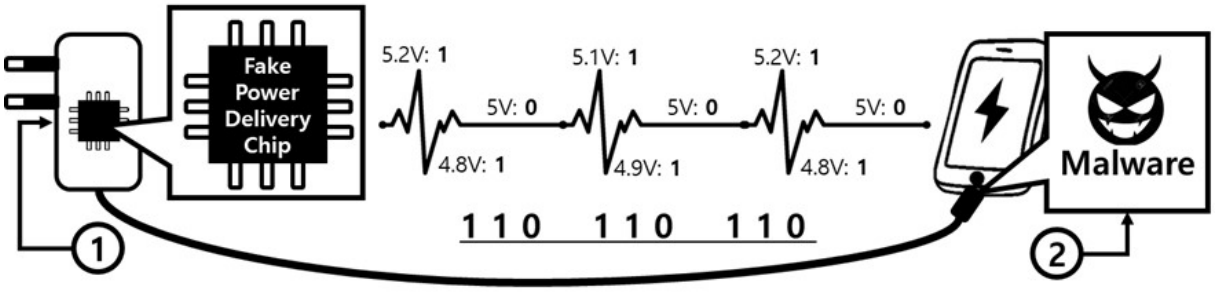
디바이스의 다양화와 사용률의 급증과 함께 은닉 채널 기술의 연구도 최근 많이 진행되고 있다. 은닉 채널 공격은 디바이스의 전력 소모 및 전자기파와 같이 누수 정보를 사용하여 디바이스를 침해하는 기법으로 일반적인 사용자들은 이러한 채널에 대해 인지하지 못한다. 이러한 은닉 채널 공격 기법으로 본 논문에서는 디바이스를 충전에 사용되는 USB 내 Power Delivery Chip을 이용한 은닉 채널 기법을 제안한다.

USB Power Delivery Chip(Power Delivery Chip)이란 USB에 연결된 기기의 전력을 전달하는 기능을 수행한다. 이 과정에서 해당 chip은 연결된 디바이스에게 전력의 양을 결정할 수 있다. 일반적인 충전 모드와 고속 충전 모드가 그 예이다. 최근엔 Power Delivery Chip 내 디바이스 인증 기능을 탑재하여 이 기능이 수행되면 데이터 송수신을 차단하기도 한다. 그러나 이러한 전력의 양을 결정할 수 있는 Power Delivery Chip을 조작하여 미세 전류를 조작한다면 디바이스로 특정 신호를 보낼 수 있는 은닉 채널을 제안한다.

본 논문에서 제안하는 USB 내 Power Delivery Chip을 이용한 은닉 채널 기법은 일반적으로 사용되는 Power Delivery Chip을 공격자가 신호를 보내도록 재설계해야 하며, 디바이스 내 전력의 양을 측정하고 이를 수정할 수 있도록 악성코드에 감염되어 있어야 한다.

II. Power Delivery Chip을 통한 은닉채널

본 논문에서는 디바이스의 USB Power Delivery Chip에 충전 전력의 양을 조절하여 디바이스로 신호를 전달하는 은닉 채널을 제안하였다. 해당 제안을 수행되기 위해서는 재설계된 Power Delivery Chip과 해당 은닉 채널의 수신



[그림 1] Power Delivery Chip 기반의 은닉 채널 디자인

지가 되는 악성코드가 필요하다. 이러한 은닉 채널의 구성은 그림 1과 함께 설명한다.

2.1 Power Delivery Chip

공격자는 디바이스에 연결될 Power Delivery Chip에 Chip 재설계로 신호를 입력한다. 입력의 신호는 그림 1과 같이 신호를 미세하게 조정을 하여 신호를 전송한다. 그림 1의 경우 5V의 충전량을 가정하고 작성하였다. 이와 같은 미세한 전류는 공격자가 어떤 신호를 전송하는지에 따라 전류 값이 달라지기 때문에 디바이스에서 전력을 상시 감시하더라도 패턴을 찾아내지 않는 이상 해당 은닉 채널을 탐지하기 어렵다. 본 논문에서 제안하는 설계에서는 5v에서 '0'이며, 그 외에서는 '1'로하여 설계하였으나 추후 연구로 실제 Power Delivery Chip을 설계하는 과정에서 일반적인 충전 상황의 전력의 범위에 따라 '1'의 신호 범위도 달라질 것이다.

2.2 디바이스 내 악성코드

스마트폰에서 안드로이드 프로그래밍으로 충전되는 전량을 측정할 수 있다. 안드로이드 프로그래밍을 위해 제공되는 BatteryManager 클래스를 사용하게 되면 전량의 미세한 차이 또한 측정 가능해진다. 그림 1과 같이 이러한 신호를 해석하여 신호로 변환한다면 은닉 채널에서 데이터를 송수신할 수 있을 것이다.

다른 디바이스에서도 전량을 측정할 수 있는 상황이 존재한다면 앞서 설명한 스마트폰 안드로이드 같은 경우와 같이 은닉 채널의 전량 신호를 측정하고 정보를 송수신할 수 있을 것이다.

III. 결론

본 논문에서는 은닉 채널로써 디바이스의 충전 모듈인 Power Delivery Chip에서 미세한 전량 조절을 통해 신호를 보내고 악성코드에 감염된 디바이스로 해당 신호를 수신하는 은닉 채널을 제안하였다. 추후 연구로는 본 제안의 은닉 채널을 구성하고 실험을 통해 검증할 것이다.

[참고문헌]

[1] ZHANG, Xiaosong, et al. A covert channel over volte via adjusting silence periods. IEEE Access, 2018, 6: 9292-9302.

[2] POOVENDRAN, Radha. No Free Charge Theorem: A Covert Channel via USB Charging Cable on Mobile Devices. In: Applied Cryptography and Network Security: 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings. Springer, 2017. p. 83.

[3] GURI, Mordechai; MONITZ, Matan; ELOVICI, Yuval. USBee: air-gap covert-channel via electromagnetic emission from USB. In: 2016 14th Annual Conference on Privacy, Security and Trust (PST). IEEE, 2016. p. 264-268.

[4] 조재범 기자. 삼성전자, 세계 최초 '전력제어+보안칩' 시스템LSI 내놔 ,2019,05,28,

ttp://biz.newdaily.co.kr/site/data/html/2019/05/28/2019052800016.htm 보안 뉴스

[5] NVDC I2C Battery Buck-Boost Charge Controller with System Power Monitor & Processor http://www.ti.com/lit/pdf/SLYY145?jktype=tech_docs

[6] BatteryManager, https://developer.android.com/reference/android/os/BatteryManager.html

음향 신호를 이용한 은닉채널 동향 연구[1)]

이선우* 이동훈*

*고려대학교 정보보호대학원

Research on Covert Channel using Acoustic Signal

Sun-Woo Lee* Dong Hoon Lee*

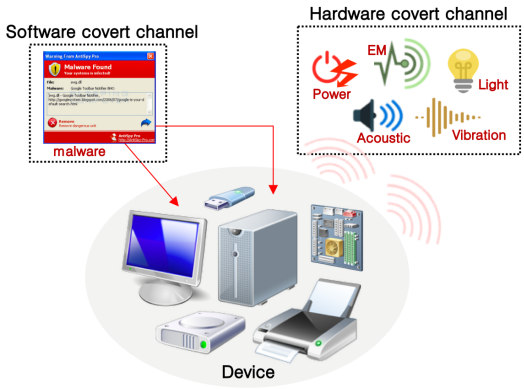
*Graduate School of Information Security, Korea University.

요 약

최근 ICT(Information Communication Technology) 기술 및 신호처리 기술이 발달함에 따라 모바일 기기의 Covert Channel을 이용하여 데이터를 유출시키거나 악성 데이터를 삽입하는 다양한 공격 기법들이 연구되고 있다. 이러한 Covert Channel을 통한 공격은 기업의 중요 원천 기술뿐만 아니라 민감한 국가 정보까지도 유출시킬 수 있는 중요한 문제이다. 이에 본 논문에서는 모바일 기기의 Covert Channel를 형성할 수 있는 매체들 중 음향 신호를 이용한 공격 방법에 대해 서술한다.

I. 서론

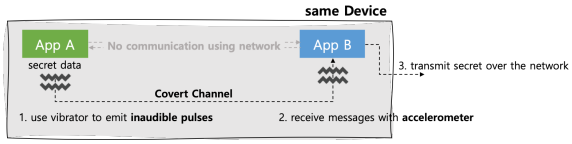
최근 ICT(Information Communication Technology) 기술과 신호처리 기술의 발달로, 모바일 기기의 Covert Channel을 이용하여 데이터를 유출시키거나 악성 데이터를 삽입하는 다양한 공격 기법들이 연구되고 있다. 기기의 Covert Channel을 통한 공격은 사용자의 인식 없이 기업의 중요 원천 기술뿐만 아니라 민감한 국가 정보까지도 유출시킬 수 있으므로 중요한 문제이다. 이에 미국의 경우, TEMPEST라는 의도치 않은 채널을 통해 발생할 수 있는 정보유출에 대한 감시시스템을 제안하였으며[2], 해당 시스템은 NATO(North Atlantic Treaty Organization, 북대서양 조약 기구) 인증 중 하나로 지정되어있다. IoT 시장 또한 급격히 성장하고 있는 현황[1]에서, 기기의 Covert Channel을 통한 공격의 위험성은 더욱더 증가할 것이다.



[그림 1] 기기의 Covert Channel로 이용될 수 있는 물리적(Hardware) 및 논리적(Software) 매체 종류

기기의 Covert Channel로 이용될 수 있는 매체는 [그림 1]과 같이 물리적(Hardware) 매체와 논리적(Software) 매체로 나뉠 수 있다. 물리

1) 본 연구 논문은 미래창조과학부의 출연금으로 수행하고 있는 한국전자통신연구원 ETRI 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발[19HH3700] 위탁연구과제의 연구결과입니다.



[그림 2] Intra-device Communication 공격 시나리오

적 매체에는 열, 음향 신호 및 빛 등이 있으며, 논리적 매체에는 네트워크 프로토콜, 시스템 메모리 및 시간 동기화 프로그램 등이 있다. 본 논문에서는 기기의 Covert Channel로 이용될 수 있는 여러 매체들 중 물리적 매체인 음향 신호를 이용한 공격 방법을 서술한다. 음향 신호는 모든 모바일 기기에서 추가적인 장치 없이 전송 및 수신이 가능하다. 따라서 음향 신호를 이용한 Covert Channel 공격은 다른 매체를 이용한 공격보다 비교적 위험성이 더 크다.

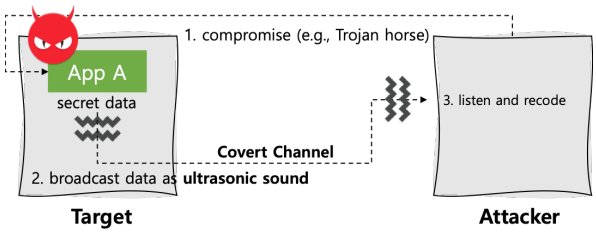
II. 음향신호를 이용한 은닉채널 동향

본 절에서는 음향신호를 이용한 Covert Channel 공격 방법을 서술한다.

1.1 Inaudible Sound as a Covert Channel in Mobile Device [3]

본 논문은 모바일 기기에서 초음파 신호를 이용하여 형성한 Covert Channel을 통해 사용자의 인식 없이 데이터를 유출시키는 공격을 수행한 논문이다. 모바일 기기에서는 추가적인 장치 없이 기기에 내장된 Speaker로 초음파 신호를 생성할 수 있다. 또한 기기에 내장된 Gyroscope 센서는 초음파 신호에 민감[6]하기 때문에 이를 이용함으로써, 모바일 기기에서 초음파 신호를 수신할 수 있다.

본 논문에서는 두 가지의 공격 시나리오를 제시하였다. [그림 2]는 타겟 기기 내에서 두 개의 앱 간의 데이터 유출을 다루고 있으며, [그림 3]은 공격자의 기기와 타겟 기기 사이에서의 데이터 유출을 나타낸다. 두 가지의 공격 시나리오는 모두 모바일 기기 내의 Gyroscope 센서에 Access Control이 되어 있지 않다는 점을 이용한다. 따라서 공격자는 사용자의 Permission 및 인식 없이, 타겟 기기의 Gyroscope 센서에 접



[그림 3] Inter-device Communication 공격 시나리오

근하여 센서 값을 얻을 수 있다.

우선 공격자는 이미 유출시킬 데이터를 가지고 있으며, 타겟은 공격자가 기기의 Gyroscope 센서 값을 읽을 수 있도록 생성한 앱를 실행하고 있다고 가정한다. 공격자는 유출시킬 데이터를 On-off 변조 기법을 이용하여 초음파 신호로 변환한다. 이후 App A에서 진동센서 또는 스피커를 이용하여 변조된 신호를 전송하면, [그림 2]처럼 기기 내 App B에서 또는 [그림 3]처럼 공격자의 기기 내의 Gyroscope 센서를 통해 초음파 신호를 수신하여 데이터를 얻을 수 있다.

1.2 How to phone home with someone else's phone [4]

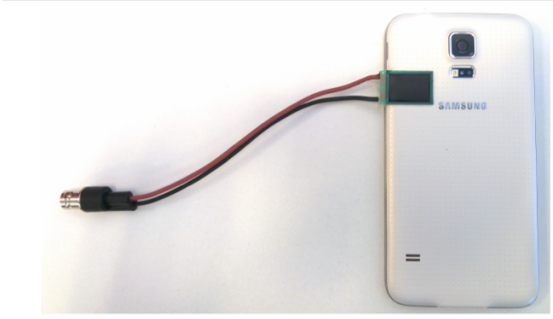
본 논문은 모바일 기기 내 MEMS(micro-electromechanical) Gyroscope센서를 이용하여 데이터를 유출시키는 방법을 제안하였다.

iOS, Android 및 Windows 웹페이지에서는 *ondevicemotion()*와 *ondeviceorientation()* 명령어를 통해 사용자의 인식 없이 Gyroscope 센서에 접근할 수 있으며, 샘플링 속도는 Chrome과 Safari 브라우저에서 60Hz, Firefox에서 100Hz, 그리고 Android 브라우저에서는 20Hz이다. 또한 App에서도 Gyroscope 센서에 사용자의 Permission없이 접근할 수 있으며, 샘플링 속도는 iOS는 100Hz, Android는 200Hz이다.

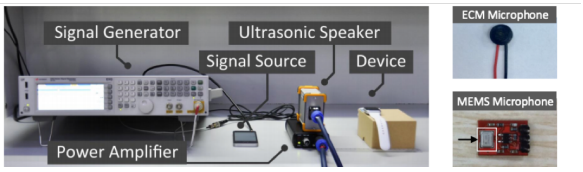
[그림 4]는 공격 시나리오의 전체 개요를 나타낸다. 이때 타겟은 공격자가 기기의 Gyroscope 센서 값을 읽을 수 있도록 의도적으로 생성한 앱 또는 웹페이지를 실행하고 있다고 가정한다. Implant는 유출시킬 데이터를 수집하는 역할을 하며, 이후 해당 데이터를 AM (Amplitude



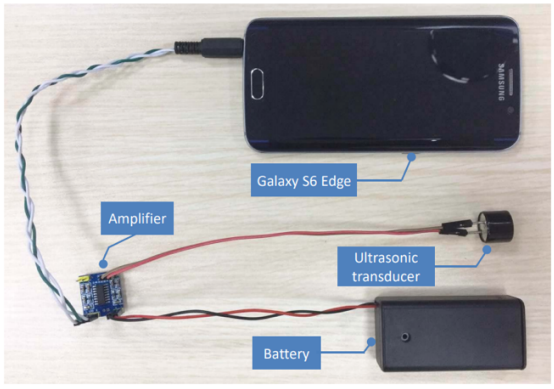
[그림 4] 초음파 신호를 이용한 데이터 유출 공격 시나리오의 개요



[그림 5] 실험 환경



[그림 6] 고성능 신호 생성기를 이용한 실험 환경



[그림 7] 모바일 기기를 이용한 실험환경

Modulation) 변조기법을 이용하여 초음파 신호로 변조시킨다. 초음파 신호가 타겟 기기에 전송되어 타겟 기기의 Gyroscope 센서에 영향을 주면, 앱 또는 웹페이지에서 Gyroscope 센서 값을 읽어 들여 이를 공격자의 C&C(Control and Command) 서버로 전송한다. 공격자는 최종적으로 Gyroscope 센서 값으로부터 데이터를 복조하여 데이터를 얻을 수 있다. 실험환경은 [그림 5]와 같이, 초음파 신호를 전송할 수 있는 Speaker를 타겟 기기에 부착하여 기기 내의 Gyroscope 센서에 영향을 줄 수 있다.

1.3 Dolphinattack: Inaudible voice commands [5]

본 논문은 초음파 신호를 이용하여 모바일 기기의 VCS(Voice Controllable System) 시스템에 악성 명령어를 삽입하는 공격을 수행한 논문이다.

VCS 시스템은 Voice Capture, Speech Recognition 및 Command Execution 단계 순으로 이루어져있다. Voice Capture 단계는 음성을 수신하여 증폭하고 필터링하여 잡음을 제거하는 단계이다. Speech Recognition는 Activation과 Recognition의 세부 단계로 이루어져있다. Activation 단계는 “Hey Siri”와 같이 사전에 정의된 단어로, VCS 시스템을 활성화시키는 단계이며, Recognition 단계는 사용자의 음성을 인증하는 단계이다. 마지막으로 Command Execution은 사용자가 말한 명령을 수행하는 단계이며, 이 단계에서는 음성 인증은 이루어지지 않는다.

본 논문에서는 여러 TTS(text-to-speech) 시스템들을 통해 생성한 음성 신호를 고주파수 대역의 초음파 신호로 변조하여 모바일 기기에 전송함으로써 공격을 수행하였다. [그림 6, 7]은 실험 환경을 나타낸 것으로, [그림 6]은 고성능의 신호 생성기를 이용하였으며 [그림 7]은 Samsung Galaxy S6 Edge 모바일 기기를 이용하였다. 고성능의 신호 생성기와 모바일 기기를 이용하여 VCS 시스템을 공격한 결과는 각각 [그림 8]과 [그림 9]와 같다.

TTS Systems	voice type #	# of successful types	
		Call 12..90	Hey Siri
Selvy Speech [51]	4	4	2
Baidu [8]	1	1	0
Sestek [45]	7	7	2
NeoSpeech [39]	8	8	2
Innoetics [59]	12	12	7
Vocalware [63]	15	15	8
CereProc [12]	22	22	9
Acapela [22]	13	13	1
Fromtexttospeech [58]	7	7	4

[그림 8] 고성능 신호생성기를 이용하여 VCS 시스템을 공격한 결과

f_c (kHz)	20	21	22	23	24
Word recognition rate	80%	100%	16%	100%	0%
Sentence recognition rate	80%	100%	0%	100%	0%

[그림 9] 모바일 기기를 이용하여 VCS 시스템을 공격한 결과

Ⅲ. 결론

본 논문에서는 초음파 신호를 이용하여 모바일 기기에 Covert Channel을 형성한 후, 데이터를 유출시키거나 악성 데이터를 삽입하는 공격 동향을 살펴보았다. 초음파 신호의 경우 모바일 기기 자체에서 송수신이 가능하기 때문에, 공격자는 Covert Channel을 형성하기가 쉽다. Covert Channel을 이용한 공격은 사용자의 인식 없이 이루어진 다는 점에 있어서 위험성이 크다. 최근 모바일 기기의 시장규모 및 기술 발전이 급격히 성장하고 있다. 따라서 모바일 기기에 대한 Covert Channel의 존재 여부를 판단하는 기술 및 이를 이용한 공격을 막을 수 있는 기술을 연구해야한다.

[참고문헌]

[1] https://www.sciencetimes.co.kr/?news=iot-기기-2022년에-2-5배-증가 [access]

[2] https://web.archive.org/web/20081224072647/http://www.nsa.gov/ia/industry/tempest.cfm

[3] Deshotels, Luke. "Inaudible sound as a covert channel in mobile devices." 8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14). 2014.

[4] Farshteindiker, Benyamin, et al. "How to phone home with someone else's phone: Information exfiltration using intentional sound noise on gyroscopic sensors." 10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16). 2016.

[5] Zhang, Guoming, et al. "Dolphinattack: Inaudible voice commands." Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017.

[6] Son, Yunmok, et al. "Rocking drones with intentional sound noise on gyroscopic sensors." 24th {USENIX} Security Symposium ({USENIX} Security 15). 2015.

USBEE의 은닉 채널을 활용한 데이터 유출 방지를 위한 주파수 탐지 기술 제안

황태윤*, 김서현*, 한혜경*, 윤지원*

*고려대학교 정보보호대학원

Proposed frequency detection technology to prevent data leakage using USBEE's covert channel

Tae Yoon Hwang*, Seo Hyun Kim*, Hye Kyung Han*, Ji Won Yoon*

*Graduate school of information security, Korea University.

요 약

Covert channel이란 은닉채널을 의미하며 두 송, 수신자간 비밀 채널을 개설하여 정보를 교환할 수 있는 통신 방법이다. 은닉채널에 대한 연구는 여러 매체를 통하여 채널을 형성하는 방향에 대해서 연구가 되어왔고 본 논문에서는 이동되는 데이터에 따라 특정 주파수를 내는 악성코드를 재구현하여 악성코드가 감염된 컴퓨터에서 저장매체로의 데이터 전송이 일어날 때 발생하는 특정 주파수를 탐지하는 방법론을 제안한다.

I. 서론

은닉채널(Covert Channel)을 사용하여 두 사용자간에 비밀 채널을 개설하여 정보를 송, 수신 하거나 손상된 시스템에서 정보를 유출하는 것은 지능형 공격의 목표중 하나이다. 일반적인 기업들의 서버에서는 논리적 망분리, 방화벽, IDS 및 IPS등의 일반적인 보안조치를 취하고 있어 어느 정도의 수준의 데이터 유출 위험은 감소하고 있다.

이렇게 민감한 데이터의 유출을 막기 위한 일반적인 방법이 사용되고 있으나 인터넷 연결이나 물리적 접근이 없이도 Air-gap을 이용하여 악성코드가 감염된 컴퓨터에서 비밀 데이터를 유출하는 광범위한 방법들이 연구되었다. 따*라서 본 논문에서는 Air-gap 네트워크를 이용하여 정보유출이 어떻게 이뤄지는지 간단한 실험 결과를 보여주고 그로 인해 나타나는 주파수의 패턴을 분석하여 Air-gap 네트워크 공격을 어떻게 탐지할 것인지에 대한 방법론을 제기한다.

* 본 연구 논문은 미래창조과학부의 출연금으로 수행하고 있는 한국전자통신연구원 ETRI 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발[19HH3700] 위탁연구과제의 연구결과입니다.

II. 사전 지식

2.1. Covert Channel

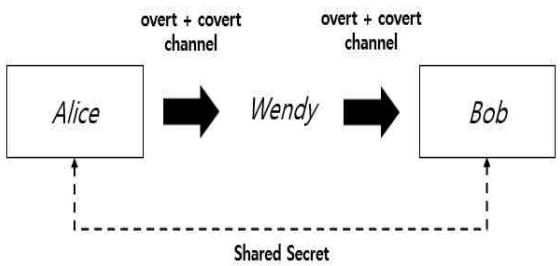


그림 1. Covert Channel

은닉 채널은 송, 수신자간의 비밀 통신을 위한 숨겨진 채널을 뜻한다. 두 사용자간에는 특정 프로토콜을 정해놓고 이동하는 네트워크 패킷 또는 물리적 현상을 통해서 데이터를 유출할 수 있도록 한다. Covert Channel은 죄수문제에 대입하여 설명이 가능하다. 이는 Simmons에 의해 처음 제기되었으며 Coveert Channel에 대한 전체적인 컨셉을 설명할 수 있다[1].

Simmons는 Covert channel에 대한 설명을 감옥에 갇혀있는 죄수문제에 대입하여 그림 1과 같이 설명한다. 죄수 Alice와 Bob은 Wendy라는 간수를 통해서만 쪽지를 주고받을 수 있다고 가정한다. 따라서 Alice와 Bob이 비밀의 메시지를 전송하려면 간수인 Wendy가 알아챌 수 없는 비밀 채널이 필요하다. 따라서 두 사람간의 프로토콜을 미리 정의해놓고 overt message 및 covert message를 정의한 쪽지를 간수를 통해서 전달하고 비밀 프로토콜을 공유한 사용자들끼리만 covert message를 통해서 통신할 수 있는 구조이다.

2.2. 관련 연구

관련 연구로는 네트워크, 빛, 열, 전자기파, 소리 등의 여러 매체를 통하여 Covert chennel을 생성하는 연구들이 진행되었다. Deshotels, Luke.[2]는 모바일 기기에서의 악성 앱을 통하여 메시지 값을 변환시킨 진동 신호를 생성한 후 은닉 채널 형성에 관한 연구를 진행하였다. Guri, Mordechai, et al[3].은 FM 신호를 사용하여 격리된 네트워크와 주변의 감염된 휴대폰 간의 Air-gap을 연결하여 Covert channel을 형성하는 malware인 AirHopper를 제안하였다. Claeys, Timothy, et al[4].은 기기의 열을 이용하여 clock drift 현상으로 유발되는 clock skew를 이용하여 블루투스 통신환경에서의 covert channel을 형성하는 방식을 제안하였다. Akhtari, Sara, Neda Moghim, and Mojtaba Mahdavi[5].은 TCP/IP 네트워크 패킷을 조작하여 송수신자간의 Covert channel을 형성하는 방법을 제안하였다. 또한 Ameri, Aidin, and Daryl Johnson[6].도 네트워크 프로토콜을 조작하여 Covert channel을 생성한 연구를 진행하였으며 각 서버간 시간 동기화를 위한 Network Time Protocol을 이용하여 내부 데이터 프레임 조작을 이용한다. Guri, Mordechai, et al[7].은 LCD 모니터를 활용한 Image 기반의 데이터를 전송하는 Covert channel을 제안하였다.

III. 정보 유출 탐지

3.1. USBEE

본 논문에서는 Guri, Mordechai, Matan Monitz, and Yuval Elovici[8].가 제안한 USBEE를 구현한 후 데이터 이동에 따른 주파수 변화를 살펴본 후 이를 탐지하기 위해 Peak detection 방식을 제안한다.

[8]에 따르면 USB는 VBUS, GND, D+, D-의 데이터 버스가 존재한다. 따라서 '0', '1'로 이뤄진 바이너리 데이터들은 데이터 버스로 이동하게 된다. 이중 D+, D- 에 흐르는 전압은 들어오는 데이터에 따라 전압이 유지되거나 바뀌게 된다. 이 때 들어오는 데이터에 따라 데이터 버스의 전압이 바뀌게 된다면 EMR을 발생시키게 된다. 따라서 EMR을 장비로 탐지하여 B-FSK 필터를 거쳐 0 또는 1의 신호로 나타내게 되면 전송되는 데이터가 '1'일 경우 주파수가 높게 나타나고 '0'일 경우 주파수가 낮게 나오는 것을 볼 수 있다.

```
void signal(unsigned int *buf, int size, double freq){
  int i = 0;
  unsigned int x = 0;
  double t = freq / 4800 * 2;

  for(i = 0, x = 0x0; i < size * 8; i++){
    x = x << 1;
    if((int)(i * t) % 2 == 0){
      x++;
    }
    if((i % 32) == 31){
      *(buf++) = x;
      x = 0x00000000;
    }
  }
}
```

표 1. Signal generation code

위의 알고리즘은 [8]에서 제안된 USBEE의 일부 알고리즘이며 해당 코드를 통해서 USB에 쓰여지는 데이터에 따라 EMR을 발생시키는 것이 가능하다.

3.2. USBEE 구현 및 탐지

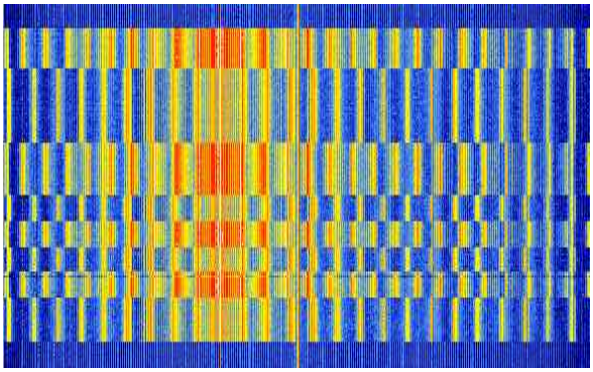


그림 2. 데이터 이동 주파수 탐지

데이터를 전송하는 악성코드의 구현은 C의 형태로 하였고 나타나는 주파수를 탐지하기 위해서 USRP 장비와 리눅스 소프트웨어인 gqrx를 사용하였다. 특정 주파수를 발생시키도록 데이터 이동에 따른 주파수를 탐지해서 형태를 STFT를 거쳐 보게 되면 전송되는 데이터에 따라 평상시보다 다른 주파수 형태를 그림 2와 같이 볼 수 있다. 그리고 주파수 데이터를 덤프떠서 살펴보면 4바이트 단위의 raw file이 나오게 된다. 이를 4바이트 단위로 묶어 복소수 형태로 파싱한 후 배열로 선언하였다. 해당 데이터들은 가공되지 않은 형태이기 때문에 다시 STFT를 거쳐야 한다. 또한 값을 잘 나타내기 위해서 log를 통해 스케일링을 해주고 하나의 축을 중심으로 평균값을 구하여 선형의 데이터 형식을 만들었다.

그림 3에서와 같이 결과물의 그래프가 나오게 된다. 진폭이 평소보다 높게 나타나는 곳에서 높낮이 변화가 나타나게 되는데 이는 그림 2에서 나타난 주파수의 변화와 형태가 동일하게 나온다.

이를 더 선명하게 나타내기 위해서 각 값에 diff를 취한 후 peak detection 알고리즘을 통한 형태를 살펴보면 그림 3과 같이 나타나는 것을 확인할 수 있다. 여기에서 peak로 지정된 부분을 살펴보면 평소의 주파수를 제외하고 일정 간격으로 peak 값이 생성되는 것을 확인할 수 있는데 이를 그림 2와 비교해보면 평소의 주파수와 달리 데이터 이동에 따른 주파수의 변화가 생길 때 위와 같이 일정 간격으로 peak가

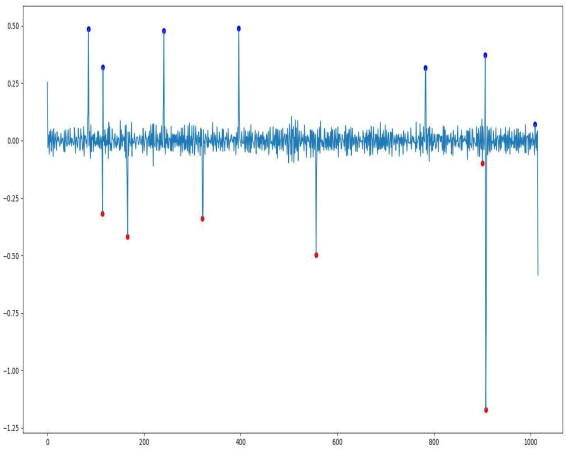


그림 3. diff 형태 그래프

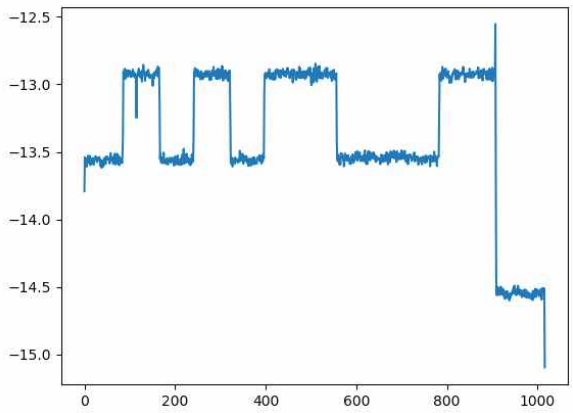


그림 4. 주파수 그래프

생기는 것을 확인할 수 있다. 따라서 감염된 컴퓨터에서 데이터를 전송할 때 일정한 주파수 변화가 특정 간격으로 나타난다면 데이터 전송에 따른 주파수 유출이 일어난다는 것을 알아낼 수 있다.

IV. 결론

본 논문에서는 데이터를 전송할 때 주파수를 발생시키는 USBEE 악성코드를 재구현 후 이를 어떻게 탐지하는지를 소개하였다. 단순한 알

고리즘을 사용하여 그래프 해석에 대해서는 아직 인력이 필요하지만 감염된 컴퓨터의 데이터 이동에 따른 주파수가 발생할 때 탐지할 수 있는 방법을 본 논문에서 제시하였다.

[참고문헌]

[1] S. Craver, “On Public-Key Steganography in the Presence of an Active Warden,” Proc. 2nd Int’l. Wksp. Information Hiding, Apr. 1998, pp. 355 - 68.

[2] Deshotels, Luke. "Inaudible sound as a covert channel in mobile devices." 8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14). 2014.

[3] Guri, Mordechai, et al. "AirHopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies." 2014 9th International Conference on Malicious and Unwanted Software: The Americas (MALWARE). IEEE, 2014.

[4] Claeys, Timothy, et al. "Thermal covert channel in bluetooth low energy networks." WiSec. 2019.

[5] Akhtari, Sara, Neda Moghim, and Mojtaba Mahdavi. "Binary middleman covert channel in ExOR protocol." 2017 Iranian conference on electrical engineering (ICEE). IEEE, 2017.

[6] Ameri, Aidin, and Daryl Johnson. "Covert channel over network time protocol." Proceedings of the 2017 International Conference on Cryptography, Security and Privacy. ACM, 2017.

[7] Guri, Mordechai, et al. "An optical covert-channel to leak data through an air-gap." 2016 14th Annual Conference on Privacy, Security and Trust (PST). IEEE, 2016.

[8] Guri, Mordechai, Matan Monitz, and Yuval Elovici. "USBee: air-gap covert-channel via electromagnetic emission from USB." 2016 14th Annual Conference on Privacy, Security and Trust (PST). IEEE, 2016.