금융 간편결제 앱 보안솔루션 문제점과 대응방안 –안드로이드 앱을 중심으로

Countermeasure of financial e-payment app security solution problem

김 경 곤(고려대학교)

차 례

■ keyword : | FinTech | Security Solution Bypass | App Security

1. 서론

금융산업은 우리의 삶에 있어서 필수적인 산업으로 경제생활을 하는 사람들에게 자금을 유통시켜주는 역할을 한다. IT기술이 점차 발전하면서 금융산업도 같이 변화하여 왔다. 창구에서 사람들이 직접 입출금을 하다가 이제는 인터넷 뱅킹으로 인해 사람들이 창구에 가는 일이 과거에 비해 줄어들었다. 금융산업은 지속적으로 IT기술과의 접목을 시도하고 있는데, 최근에는 금융산업과 IT기술의 결합인 핀테크(FinTech) 산업이 활성화 되고 있다.

핀테크 (Fintech) 산업에는 현재 국내에서 송금, 지급결제, P2P 금융 플랫폼 등 다양한 분야에서 핀테크 기술이 성장하고 있다.[1] 이 중 모바일 환경에서의 지급결제는, PC 기반 위주의 온라인 쇼핑이 모바일 기반으로 전환됨에 따라 2018년에는 전년 대비 53.7%가 증가하는 등 빠른 성장세를 보이고 있다.[2]

이러한 모바일 지급결제 서비스는 인증과정을 간편화하여 결제 할 수 있어 '간편결제 서비스'로 불린다. 하지만 간편한 결제 프로세스는 곧 더 많은 보안 위협이 있을 수 있다.

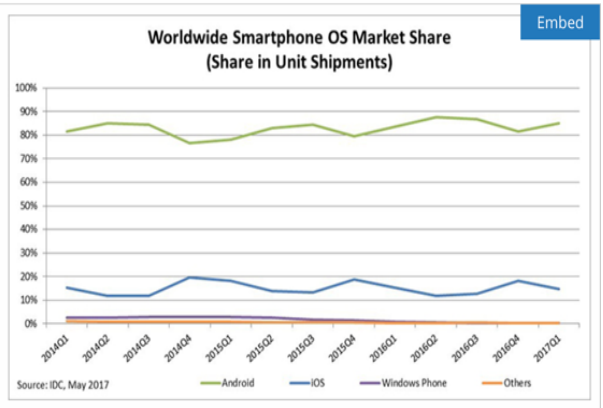
한국은행에서 2018년 2월에 발간한 2017년중 국내 인터넷뱅킹 서비스 이용현황에 따르면, 전체 인터넷 뱅킹 등록 고객 중 67.3%, 즉 절반 이상의 고객이 스마트폰 뱅킹을 이용하고 있다.[3] 뿐만 아니라 2014년 말부터 금융 규제가 완화되며 생겨난 간편결제 서비스는 새로운 결제 수단으로 각광 받으며, 모바일 환경을 기반으로 급격히 성장하였다.

이러한 모바일 기반 금융 서비스들은 높은 편리성과 접근성을 장점으로 많은 사람들이 일상생활에서 빈번히 이용하고 있다. 전자금융 서비스는 개인 및 기업의 자산과 직접적으로 연관되어 있기 때문에 더욱 높은 수준의 보안성이 요구되며, 만약 사이버 공격이 이루어졌을 시엔 금융정보 탈취나 부정결제 등 매우 심각한 피해가 우려된다.[4]

모바일 기반 금융 서비스들의 보안 위협은 모바일 OS의 보안 위협에 종속적일 수 밖에 없다. 그림 1과 같이 전 세계 스마트폰 OS의 시장 점유율중 85% 이상이 안드로이드 운영체제를 기반으로 하고 있다. 안드로이드 OS의 애플리케이션은 역공학이 쉬운 JAVA 언어로 구현되어 악성코드 삽입과 리패키징 공격 등의 수많은 위협에 노출 되어 있다.[5]

모바일 보안 솔루션 개발사 ARXAN의 조사 결과, 안드로이드 애플리케이션의 전 분야에서 90% 이상, 특히 금융분야 애플리케이션의 95% 가 해킹당한 이력이 있어 금융 애플리케이션이 해커들의 주 공격표적이 되고 있음을 확인할 수 있었다. 이를 방지하기 위해 대부분의 금융

애플리케이션이 무결성 검증, 디버깅 방지, 악성코드검사 등의 보안기능을 제공하는 보안 솔루션을 탑재하였다. 또한, 해당 보안솔루션에서 더 체계적이고 안전한 검증 수행을 위해 동적 로딩을 이용하는 등 다양한 연구가 선행되었다.[6]



▶▶ 그림 1. 스마트폰 OS 시장 점유율 - IDC

그러나 해당 연구들에서 제시한 방법으로 검 증을 수행할지라도, 애플리케이션 서비스의 모 든 기능이 개발된 후 보안솔루션이 탑재되기 때문에, 애플리케이션과 보안 솔루션의 독립적인 구조상 검증 결과 값을 위·변조하는 것만으로도 보안 솔루션을 우회할 수 있다.

본고에서는 이러한 방법을 통해 실제로 보안솔루션이 우회 가능한지 검증하기 위해 금융 애플리케이션 중 이용률이 높은 간편결제 애플리케이션 5개를 선정하여 분석 및 우회실험을 진행하였다.

본고의 2장에선 관련 연구들을 살펴보고, 3장에서는 애플리케이션과 보안 솔루션의 동작 과정에 대해 설명하고, 4장에서는 결과값 위· 변조를 통한 보안 솔루션 우회를 실험을 통해 검증한다. 5장에서 대응 방안을 제시하고, 끝으로 6장에서 결론을 서술한다.

2. 관련 연구

본 장에서는 안드로이드 애플리케이션의 보안성을 향상시키기 위한 기술들에 대한 선행 연구들을 알아본다. 금융 분야의 애플리케이션 사용률이 증가하며 이의 보안성에 대한 연구의 필요성이 대두되고 있다. 정상 안드로이드 애플리케이션의 위·변조를 통하여, 중요정보를 탈취할 수 있는 악성코드를 심거나 거래루틴 수정을 통한 부정거래 등의 공격을 수행할 수 있다. 이러한 공격을 막고 애플리케이션의 보안성을 향상시키기 위한 보안검증 기능 우회 취약점과 그 대응방안에 대한 연구들이 진행되고 있다.

금융 애플리케이션이 실행될 때 함께 실행되는 백신 애플리케이션의 검사 지점 우회 취약점과[4], 애플리케이션의 위·변조를 탐지하기 위한 무결성 검증 기능 우회 취약점 등[5] 보안 검증기능을 우회할 수 있는 취약점에 대한 연구가 실제 뱅킹 애플리케이션을 대상으로 한 실험을 통해 진행되었다.

특히 뱅킹 애플리케이션에서 무결성 검증기능을 우회하고, 리패키징을 통해 사용자가 송금하고자 하는 대상을 공격자로 변경하여 부정 거래를 진행하는 시나리오를 기반으로 한 실험을 통하여 금융 애플리케이션이 위·변조 되었을 때의 위험성을 제시하는 연구 또한 진행되었다[7].

애플리케이션의 보안 검증기능을 우회할 수 있는 취약점에 대한 연구가 진행됨과 동시에, 이러한 취약점에 대응하기 위한 여러 기법들에 대한 연구 또한 진행되었다. APK 파일의 클래스를 동적으로 로딩하여 정적 분석과 변조를 어렵게 하는 연구와[8], 동적으로 정해지는 메모리에 적재된 클래스 주소를 Key값으로 해시계산을 하는 기법을 통해 검증기능을 강화하는 등의 연구가 제시되었다[6].

또한 의사 명령어와 메소드 진입 분기문을 중요 클래스의 메소드 앞에 삽입함으로써 역공학 도구들의 분석을 방해하는 연구와[9], 단말 내의 이벤트를 추출하여 악성앱의 루팅 공격을 탐지하는 기법에 대한 연구[10] 등, 안드로이드 애플리케이션의 보안 검증기능 향상에 대한 다양한 연구가 진행되고 있다.

본고에서는 '애플리케이션과 보안 솔루션의 독립적 구조상 발생하는 검증기능 우회 취약점' 방식을 제시하며, 정교한 검증기법에 대한 상세한 루틴 분석 없이도 검증기능을 우회할 수 있기 때문에 이에 대한 연구와 대책이 필요한 실정이다.

3. 간편결제 앱 보안 솔루션 현황

본 장에서는 선택한 간편결제 앱 5종에 대해 각기 보안솔루션 적용 현황을 조사하였다.

A pay	A 앱카드	B 앱카드	A 유통 pay	B 유통 pay
A pay	A 앱카드	B 앱카드	A 유통 pay	B 유통 pay
A	A	A	C	A
C	C	C		C
	D	B		자체보안
	자체 보안	E		D

▶▶ 그림 2. 간편결제 앱에 구축된 보안솔루션

테스트에 선정된 간편결제 앱은 최소 1개에서 최대 4개까지의 보안솔루션을 탑재하고 있다. 각 보안솔루션의 기능을 살펴보면 다음 그림과 같이 루팅 체크, 무결성 체크, 안티 디버깅, 악성코드 검사, 애뮬레이터 체크 기능을 가지고 있다.

구분	루팅 체크	무결성 체크	안티 디버깅	악성코드 검사	애뮬레이터 체크
A	O	O	O		
B		O		O	
C	O	O	O		O
D	O		O	O	
E	O				
자체보안	O	O			
자체보안	O	O	O		

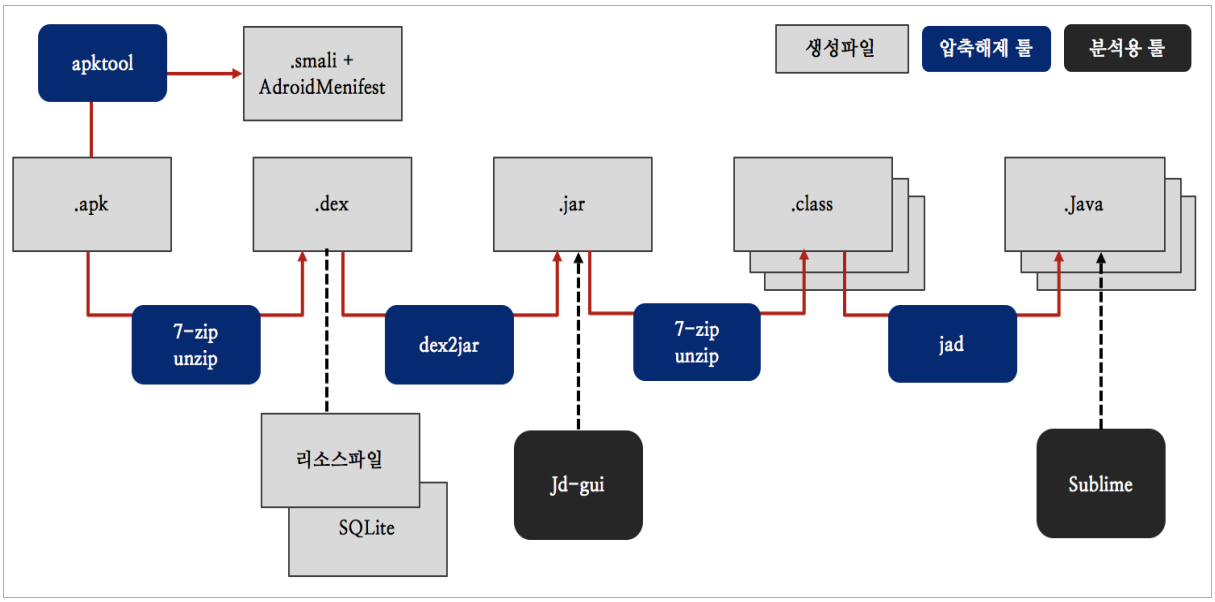
▶▶ 그림 3. 각 보안 솔루션의 보안 기능 나열

루팅 체크란 해당 단말기가 루팅되어있는지를 확인하는 것이다. 루팅이 된 단말기는 사용자 및 공격자가 루트 권한으로 단말기를 구동하는 것인데, 정상적인 사용자는 단말기를 루팅하지 않기 때문에 루팅을 체크하는 것은 사용자가 악의적으로 앱을 분석하거나, 또는 해당 단말기가 악의적인 사용자에 의해 루팅되어 공격에 악용될 수 있기 때문에 루팅을 체크한다.

무결성 체크는 사용하는 앱이 변조되지 않았는지를 확인하는 것이다. 안드로이드의 경우 악의적인 사용자는 APK 앱을 내려받은 후에 디컴파일하여 Smali 파일로 바꾼다. Smali 파일로 바꾼 다음에 해당 코드를 수정해서 다시 리패키징을 하면 해당 앱이 변조된다. 이렇게 앱이 변조되는 것을 막는 기능이 무결성 체크 기능이다. 그림 3과 같이 APK파일은 2가지 방식으로 디컴파일 되는데 하나는 java 파일 형태로, 하나는 Smali 파일 형태로 생성된다.

안티 디버깅은 앱을 분석하기 위해 디버거를 연결할 때 탐지하는 기능으로, 앱을 분석하지 못하게 하는 기능이다. 안티 디버깅 기능이 있으면 디버거를 실행하는 순간 디버거가 실행 종료 되거나, 앱이 실행 종료 된다.

악성코드 검사는 보통의 금융 앱에서 제공하는 기능 중 하나로, 해당 단말기에 악성코드가 심겨져 있는지를 본다. 보통 악성코드는 사용자 정보를 빼거나, SMS 정보를 탈취하거나, 네트워크 세션을 열어서 외부에서 불법적인 네트워크에 붙도록 하는 코드를 담고 있다. 루팅된 단말기는 루트 권한으로 단말기를 실행할 수 있기 때문에 보다 많은 권한을 가지고 있다. 그렇기 때문에 루팅



▶▶ 그림 4. APK 파일의 분석 단계

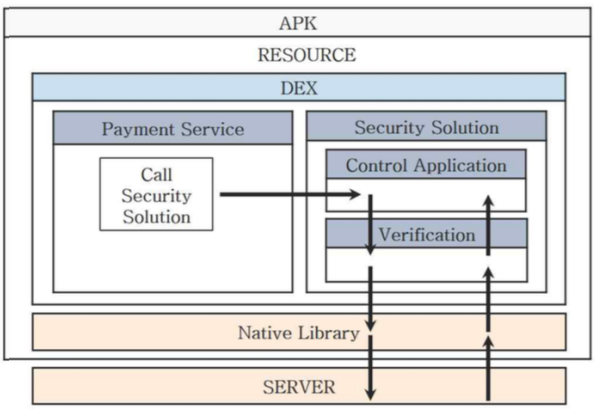
된 단말기와 악성코드 검사는 동일한 상황에서 구동되는 경우가 많다.

애뮬레이터 체크는 실제 단말기에서 앱이 구동되는 것이 아니라, 공격자가 앱을 분석하기 위해 애뮬레이터를 실행하고, 해당 애뮬레이터에서 앱을 분석한다. 이럴 경우, PC에 주로 설치해서 사용할 수 있기 때문에 네트워크 패킷을 보거나, 또는 웹 앱의 경우 웹 페이지에 전달되는 매개변수들을 모두 보고 수정할 수도 있다. 그렇기 때문에 애뮬레이터에서 구동된다는 것은 정상적인 사용자가 아니라고 볼 수 있기 때문에, 이에 대한 검증을 한다.

4. 보안 솔루션의 적용 구조 및 검증기능 우회 가능성

4.1 안드로이드 앱의 보안 솔루션 적용 구조

안드로이드 애플리케이션의 apk 파일은 Java로 구현된 코드와 리소스 파일이 컴파일 되어 구성된다. 애플리케이션의 Java class파일은, 안드로이드의 달빅(Dalvik) 가상머신이 인식할 수 있도록 DEX(Dalvik Executable, 달빅 바이트코드) 파일로 변환된다. 또한 안드로이드는 JNI(Java Native Interface)를 통해 Java가 아닌 C/C++, 혹은 어셈블리어로 작성된 네이티브 라이브러리를 달빅 가상머신에서 사용할 수 있도록 지원 한다.



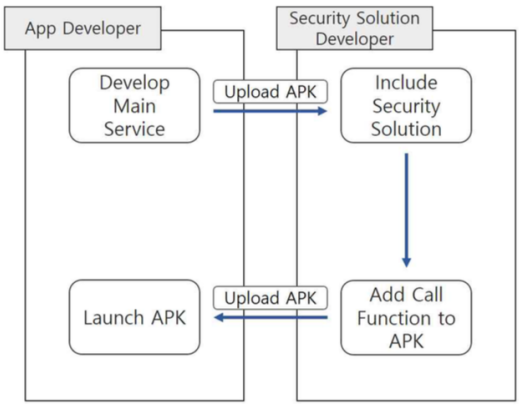
▶▶ 그림 5. 안드로이드 보안 솔루션의 구동 과정

이러한 안드로이드 애플리케이션의 구조에서 보안 솔루션은 현재 일반적으로 그림 5와 같이 적용되어 동작한다. 애플리케이션의 구동 시, DEX 파일 내에서 주 서비스는 보안 솔루션을 호출한다. 보안 솔루션은 검증 기능을 네이티브 라이브러리에서 수행한 후, 검증 결과 값을 반환한다. 반환된 결과 값을 토대로 무결성의 훼손, 변조된 OS(루팅), 디버깅 환경에서의 애플리케이션 실행 등 보안성을 저해할 수 있는 상황을 식별하여 애플리케이션을 제어함으로써 보안 위협을 차단한다.

4.2 검증 결과값 위·변조를 통한 검증기능 우회 취약점

4.1절에서 설명한 바와 같이 애플리케이션 보안 솔루션은 전달받은 반환 값을 토대로 보안 검증과 제어를 수행한다. 때문에 만약 보안 검증에 실패했다고 하더라도, 네이티브 라이브러리나 DEX의 반환 값을 정상 응답 값으로 위·변조한다면 보안 솔루션의 검증 기능이 쉽게 우회될 수 있다.

현재 일반적인 보안 솔루션의 적용 과정은 그림 6과 같다. 애플리케이션 개발사는 메인 서비스의 개발을 모두 마친 후, 보안성의 확보를 위해 원본 apk 파일을 보안 솔루션 제공업체에 넘긴다. 보안 솔루션 제공업체는 개발사에서 받은 apk 파일에 보안 검증 기능을 수행하는 네이티브 라이브러리나 DEX 코드를 포함하고, 이를 앱 구동시 호출하게 하여 보안 솔루션을 적용한다.



▶▶ 그림 6. 안드로이드 보안 솔루션의 적용 과정

표 1은 이와 같은 실험 방법을 통해 간편결제 앱 보안솔루션을 우회한 결과를 보여준다.

표 1. 간편결제 앱 보안 솔루션 우회 결과

Payment Application	Decompile	Repackaging	Bypass Points	Bypass Solutions
A	O	O	DEX	O
B	O	O	DEX	O
C	O	O	DEX	O
D	O	O	DEX	O
E	O	O	Native Library, DEX	O

5. 대응 방안

본고에서 제시한 검증기능 우회 취약점을 막기 위한 대응방안을 설계 시, 개발 시, 운영 시의 세가지 시점에서 나누어 제시한다.

5.1 설계 시 대응방안

본고에서 제시한 취약점이 발생하는 주요 원인 중 하나는 애플리케이션과 보안 솔루션의 독립적 동작이다. 때문에 이를 보완하기 위해선 애플리케이션 과 보안 솔루션을 종속적으로 설계할 필요가 있다.

보안 솔루션의 탑재 과정을 보면 애플리케이션 서비스의 개발 시, 보안 솔루션의 적용에 대한 고려가 부족하다. DEX와 Native Library의 종속적 설계를 위해선 애플리케이션 서비스를 설계할 때부터 보안 솔루션의 적용 방안에 대해 고려해야한다.

5.2 개발 시 대응방안

검증기능을 손쉽게 우회할 수 있었던 이유는 애플리케이션의 분석에 제약이 없었기 때문이다. 디컴파일 시 가장 많이 사용되는 툴인 apktool의 에러를 유발하여 디컴파일 되지 못하게 하거나[11], 소스코드에 사용되는 문자열을 암호 화하여 역분석을 통한 루틴 파악을 어렵게 하는 등, 애플리케이션의 디컴파일 및 리패키징에 성공하더라도 쉽게 구조를 파악할 수 없도록 소스코드 난독화를 강화해야한다.

5.3 운영 시 대응방안

간편결제 서비스의 개발이 완료되고 운영 중에 보안성을 확보하기 위한 방안으로는 FDS(Fraud Detect System)와 정기적인 업데이트를 통한 분석 무효화 등이 있다.

FDS는 사용자가 평소 거래하는 정보를 수집하여 이 정보들의 상관관계를 분석하고, 이를 통해 이상행위 여부를 판단하는 부정거래 탐지 시스템이다.[12] 이를 종속적 설계와 연계하여, 만약 정상적으로 검증이 수행되지 않았는데 트랜잭션 요청이 들어올 시, 비정상 거래로 간주하고 요청을 거부하는 등의 탐지기법을 활용할 수 있다.

또한 공격자들이 난독화를 해제하고 루틴을 분석하여 애플리케이션의 기능을 파악하였다면, 정기적인 업데이트를 통해 문자열 암호화 알고리즘, 애플리케이션 루틴 등의 코드를 변경하여 기존의 분석을 무효화하고 공격을 어렵게 만들 수 있다.

6. 결론

본고에서는 금융산업에서 IT기술과 접목된 핀테크 산업에서 활발하게 사용하고 있는 간편결제 앱에 적용된 보안 솔루션과 보안 솔루션 우회 기법, 그리고 대응 방안에 대해 살펴봤다. 앞으로 사용자들은 보다 편리한 방식으로 금융 거래를 하기 위해 간편결제와 같은 방법들은 더 많이 적용될 것이다. 최근에는 지문 인식만 가지고도 결제와 송금이 되는 금융 앱들도 생겨났다. 이러한 상황에서 금융 앱은 더욱 강한 보안 기능이 필요하며, 이에 따라 금융 앱에 다양한 보안 솔루션이 적용될 것이다.

본고에서 설명한 보안 솔루션 우회 기법은 기존 연구들과 다른 하나의 방식을 제시함에도 대부분의 보안 솔루션들을 우회할 수 있었다. 앞으로 모바일 앱 보안 솔루션을 개발할 때, 기존에 알려진 방법들을 보완할 수 있는 보안 솔루션 개발이 필요하며, 보안 솔루션 설계부터, 개발, 운영까지 총체적으로 고려하는 것이 필요하다.

참고문헌

[1] Survey on Mobile Payment Service(Fintech1)," Korea consumer Agency, pp. 1-2, May. 2016

[2] 한국은행, 2017년중 지급결제동향

[3] 한국은행, 2017년중 국내 인터넷뱅킹서비스 이용현황

[3] 월간 마이크로소프트웨어, 2002년 9월호

[4] 이우진, 이경호, 안드로이드 뱅킹 어플리케이션 내 중간언어 분석을 통한 보안 검사 지점 우회 취약점 연구, 정보보호학회 논문지, 2017

[5] 김순일, 김성훈, 이동훈, 안드로이드 스마트폰 뱅킹 앱 무결성 검증 기능의 취약점 연구, 정보보호학회논문지, 2013, 743-755

[6] 김정민, 이극, 동적인 key값을 이용한 안드로이드 뱅킹 앱 무결성 강화 연구, 한남대학교 석사학위

[7] Jin-Hyuk Jung, Ju Young Kim, Hyeong-Chan Lee, Jeong Hyun Yi, "Repackaging Attack on Android Banking Applications and Its Countermeasures," Wireless Personal Communications, Vol. 73, Issue. 4, pp

[8] Hyunjo Kim, Jin-Young Choi, "Research on Secure Coding and Weakness for Implementation of Android-based Dynamic Class Loading," Journal of Korea Multimedia Society, Vol. 19, No. 10, pp. 1792-1807, Oct. 2016

[9] Chanhee Lee, Yoon-Sik Jeong, Seong-Je Cho, "A Method to Protect Android Applications against Reverse Engineering," Journal of Security Engineering, Vol. 10, No. 1, pp. 41-50, Feb. 2013

[10] Chanhee Lee, Yoon-Sik Jeong, Seong-Je Cho, "A Method to Protect Android Applications against Reverse Engineering," Journal of Security Engineering, Vol. 10, No. 1, pp. 41-50, Feb. 2013

[11] "Dex Education: Practicing Safe Dex," Blackhat USA 2012, Jul. 2012, http://www.strazzere.com/papers/Dex

[12] E-Finance And Financial Security," Financial Security Institute, Issue. 1, pp. 67-98, Jul. 2015

저자소개

● 김 경 곤(Kyoung Gon Kim)

- 2008년 2월 : 숭실대학교 컴퓨터학과 학사
- 2015년 2월 : 고려대학교 정보보호대학원 정보보호학과 석사
- 2011년 12월 ~ 2016년 8월 : 딜로이트 안진 회계법인 Senior Manager
- 2006년 8월 ~ 현재 : 고려대학교 정보대학 정보보호융합전공 산학협력중점교수

<관심분야> : 모바일 보안, 네트워크 보안, Offensive Security