발표자_임경환 (단국대학교 일반대학원)
작성일_2019. 08. 20

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

Ⅰ. Introduction

1.1 biography

1.2 Background

Ⅱ. Literature survey

2.1 Previous study

2.2 Application level virtualization

Ⅲ. Our approach

3.1 Overview of Proposed method

3.2 Structure of our method

3.3 Contribution

IV. Experiment

4.1 Experimental environment

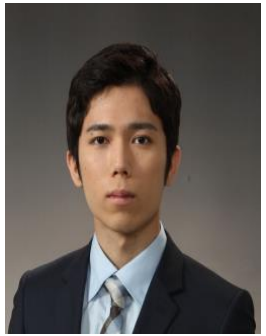
4.2 Experimental Result

V. Discussion

VI. Conclusion

1.1 biography

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법



임 경 환

E-mail limkh120@dnakook.ac.kr

학 력 **2016. 09 ~**
단국대 대학원 컴퓨터학과 (컴퓨터 보안 및 운영체제 연구실)
박사과정 (연구 분야 : 안드로이드 시스템 및 앱 보안)

2015. 03 ~ 2016. 08
단국대 대학원 컴퓨터학과 (컴퓨터 보안 및 운영체제 연구실)
석사

2009. 03 ~ 2015. 02
단국대학교 소프트웨어학과
학사

1.2 Background

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

Android Everywhere

Smart Watches

Smart glasses

Home Appliances

Cars

Smart TVs

Mobiles

1.2 Background

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

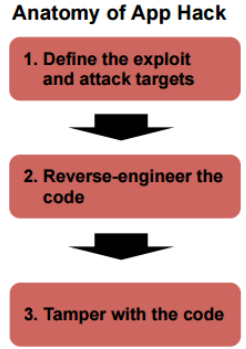
Problem: 안드로이드 앱은 해킹에 취약함

Top 10 Mobile Risks – Final List 2016 of OWASP

M5: Insufficient Cryptography

M8: Code Tampering

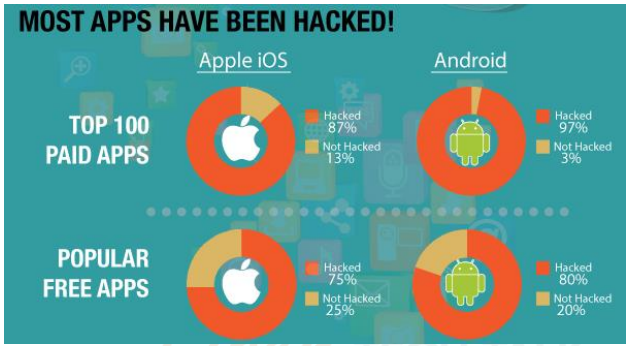
M9: Reverse Engineering



97% of the top 100 paid Android apps and 80% of the most popular free apps have been hacked by means of repackaging and cloning

[출처]

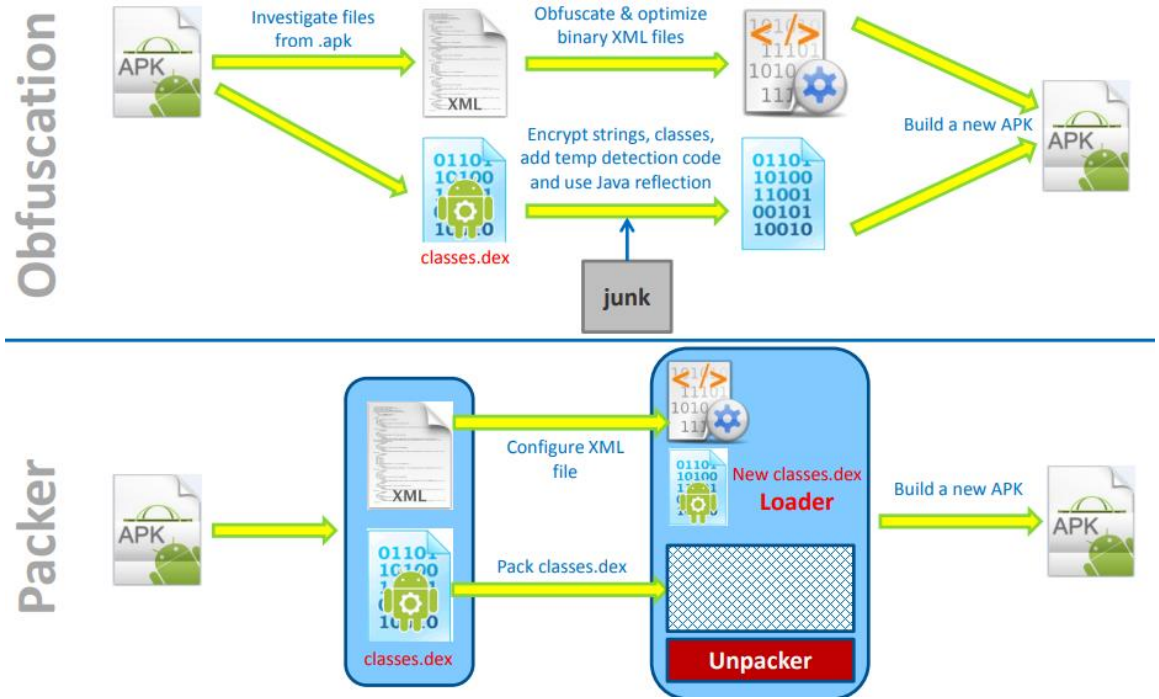
- "Dynamic Self-Protection and Tamperproofing for Android Apps using Native Code," 2015 IEEE 10th Int'l Conf. on Availability, Reliability and Security.
- Arxan Technologies, State of Mobile App Security: Apps under Attack, 25th Feb. 2015



Ⅱ. Literature survey

2.1 Previous study of Android app protection

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법



출처: SOPHOS Lab, 2014

Ⅱ. Literature survey

2.1 Previous study of Android app protection

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

- 코드 난독화, LVL (License Verification Library), Google Play Protect, SafetyNet, …
- **Packing** of Android Apps
  - Dynamic Code Loading
    - Encrypt the original classes.dex and store it in the assets folder
    - Decrypt of the entry point classes and restore the original outer interface of the app.
      - The rest of the app classes is decrypted on-demand
  - Dynamic Bytecode Re-encryption
    - Increase the encryption granularity to method level and re-encrypts the method's bytecode again after it has been executed.

- Dynamic self-protection and tamperproofing for android apps using native code, IEEE 10th International Conference on Availability, Reliability and Security, 2015
- Inside Android's SafetyNet Attestation, Blackhat Europe 2017
- Android Code Protection via Obfuscation Techniques: Past, Present and Future Directions, arXiv preprint arXiv:1611.10231, 2016
- Honey, i shrunk your app security: The state of android app hardening, Int'l Conf. on Detection of Intrusions and Malware, and Vulnerability Assessment, 2018
- Android Application Protection against Static Reverse Engineering based on Multidexing, J. Internet Serv. Inf. Secur. 6(4), 2016
- An Android Application Protection Scheme against Dynamic Reverse Engineering Attacks, JoWUA 7(3), 2016
- An anti-reverse engineering technique using native code and obfuscator-llvm for android applications, Proceedings of the ACM International Conference on Research in Adaptive and Convergent Systems, 2017

2.1 Previous study of Android app protection

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

Packers for Android Apps (Qihoo 360, Tencent, Liapp, …)

	Packer Protection Techniques							
Packer	Code Obfuscation	Dynamic Code Loading	Dynamic Code Modification	Debugger Detection	Append shared Libraries	Additional Class insertion	DVM Support	ART Support
APKProtect	✔	✔	✔	✔	✔	✔	✔	✗
Ali	✔	✔	✔	✔	✔	✔	✔	✗
Baidu	✔	✔	✔	✔	✔	✔	✔	✔
Bangle	✔	✔	✗	✔	✔	✔	✔	✔
Ijiami	✔	✔	✗	✔	✔	✔	✔	✔
HoseDex2jar	✔	✔	✗	✔	✗	✗	✔	✗
Pangxie	✔	✗	✗	✗	✗	✔	✔	✗

출처: Android Code Protection via Obfuscation Techniques: Past, Present and Future Directions, arXiv preprint arXiv:1611.10231, 2016.

2.1 Previous study of Android app protection

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

- **패킹 기법의 한계**
  - 앱 수정 필요: stub dex (unpacker) 추가, lib 삽입, manifest.xml 수정, repackaging
  - 동적 분석에 취약: 많은 연구들 수행되어 왔음.
    - Memory dumping
    - Unpacking

- Dexhunter: toward extracting hidden code from packed android applications, European Symposium on Research in Computer Security, 2015.
- We Can Still Crack You! General Unpacking Method For Android Packer (NO ROOT), BlackHat Aisa 2015
- Unpacking the Packed Unpacker Reversing an Android Anti-Analysis Native Library, BlackHat USA 2018
- AppSpear: Automating the hidden-code extraction and reassembling of packed android malware, Journal of Systems and Software, 140, 2018
- Adaptive Unpacking of Android Apps, 2017 IEEE/ACM 39th Int'l Conf. on Software Engineering, 2017
- Things You May Not Know About Android (Un)Packers: A Systematic Study based on Whole-System Emulation, NDSS, 2018
- DexX: a double layer unpacking framework for Android, IEEE Access 6 (2018)

2.2 Application level virtualization

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법

- 어플리케이션 수준 가상화
  - 개념 및 특징:
    - OS를 추상화하고 앱과 OS 사이의 모든 상호작용을 관리하는 계층을 도입
      - 동일한 앱을 dual-instances 형태로 실행하기 위함
      - 한 스마트폰에서 두개 이상의 facebook 계정을 동시에 로그인 할 수 있음
    - 앱을 단말에 설치하지 않고 동적으로 로딩 및 실행하게 해 줌
  - 대표적 도구: DroidPlugin, VirtualApp, ...

- An android dynamic data protection model based on light virtualization, 2013 15th IEEE International Conference on Communication Technology, 2013
- Anception: Application virtualization for android, arXiv preprint arXiv:1401.6726, 2014
- A lightweight virtualization solution for android devices, IEEE Transactions on Computers, 64(10), 2015
- Anti-Plugin: Don't let your app play as an Android plugin, Proceedings of Blackhat Asia (2017)
- Jekyll and Hyde is Risky: Shared-Everything Threat Mitigation in Dual-Instance Apps, Proceedings of the ACM 17th Annual International Conference on Mobile Systems, Applications, and Services, 2019
- Demystify Application Virtualization in Android and its Security Threats, Proceedings of the ACM on Measurement and Analysis of Computing Systems 3.1, 2019

3.1 Overview of Proposed method

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

Goal : 어플리케이션 수준 가상화 및 패킹을 적용한 안드로이드 앱 역공학 방지

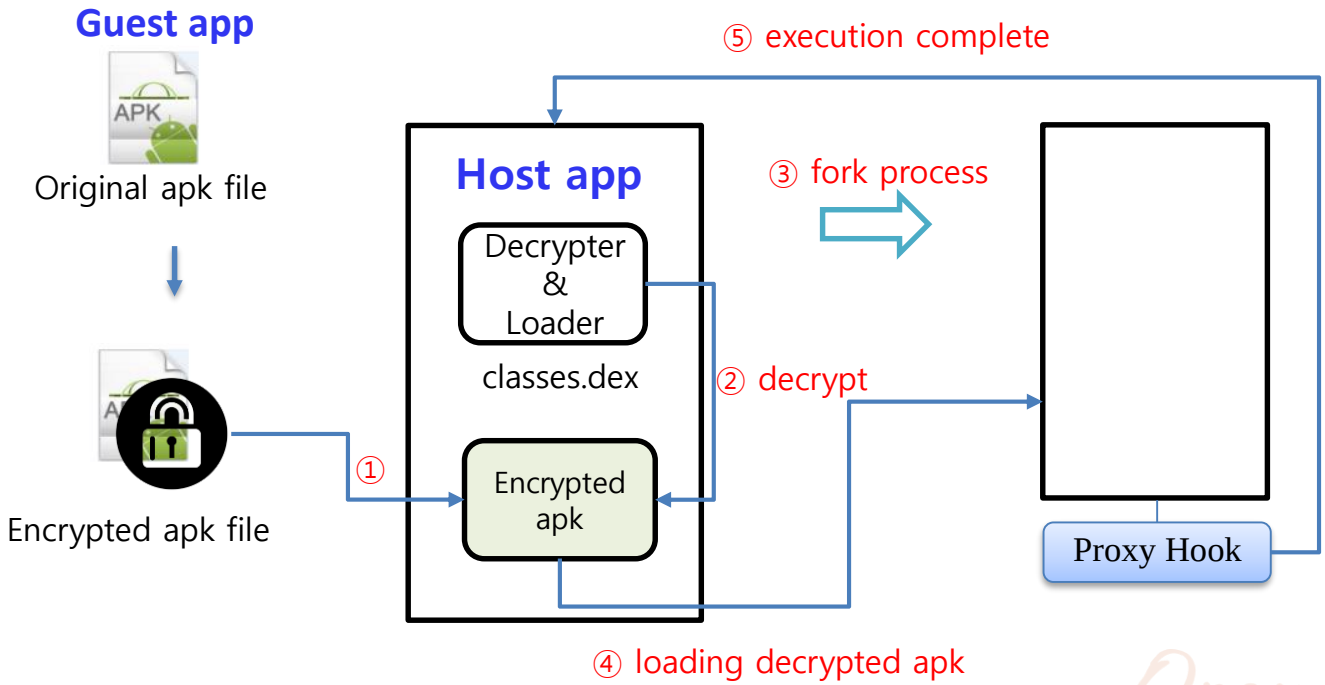
- Host app
  - DroidPlugin (가상화 프레임워크) 기반으로 구현

- Guest apps (보호 대상 앱. 패킹/암호화되어 배포)

Ⅲ. Our Approach

3.1 Overview of our method

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법



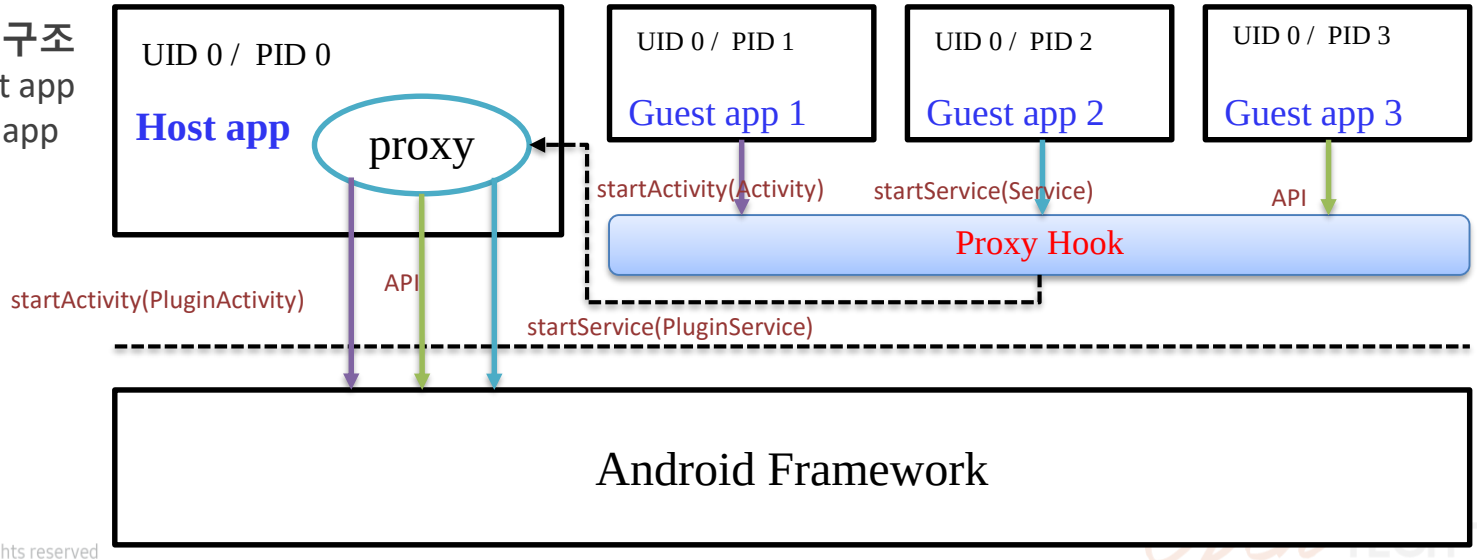
3.2 Structure of our method

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

OS를 추상화하고 앱과 OS 사이의 모든 상호작용을 상호 연결하는 계층

Guest Apps' Requests ⇔ **Host App** ⇔ Android Framework

Master/Slave 구조
- Master : Host app
- Slave : guest app



3.3 Contribution

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

제안 기법 장점

- No firmware modification
- No app modification
  - The solution should not require to access to apps' source code
  - Only APK files are needed.
- No installation
  - 설치된 앱 리스트에 나타나지 않음 (pm 서비스 상에서)

☞ 어플리케이션 수준 가상화 및 패킹을 통한 앱 역공학 및 변조 방지

IV. Experiment

4.1 Experimental environment

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

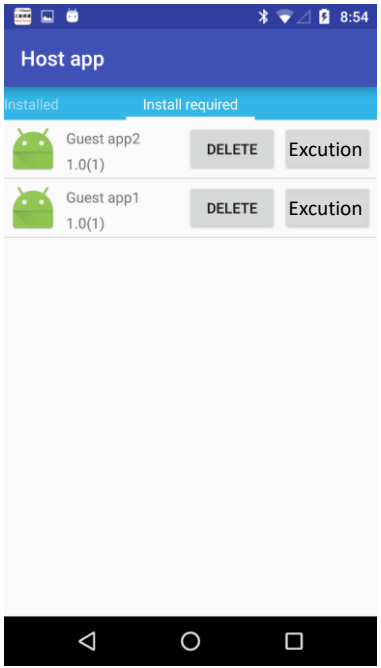
- Devices : Nexus 5
- Android OS version : 6.0.1 (Marshmallow)
- Linux Kernel : 3.4.0-gcf10b7e

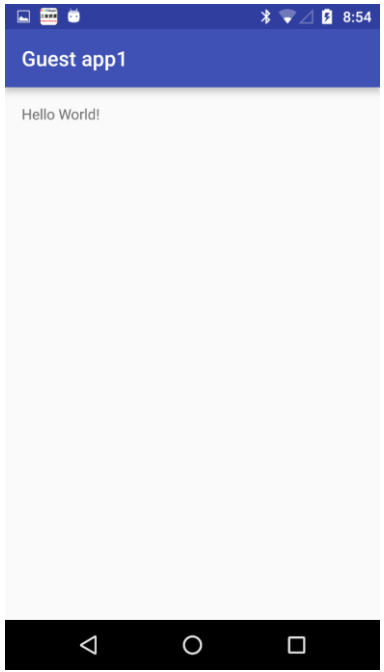
- DroidPlugin
  - Virtualization Framework Library
  - A plugin Framework
- Host app based on DroidPlugin
- Simple guest app
- 구현 및 성능 평가
  Cold-start delay 측정

IV. Experiment

4.2 Experimental Result

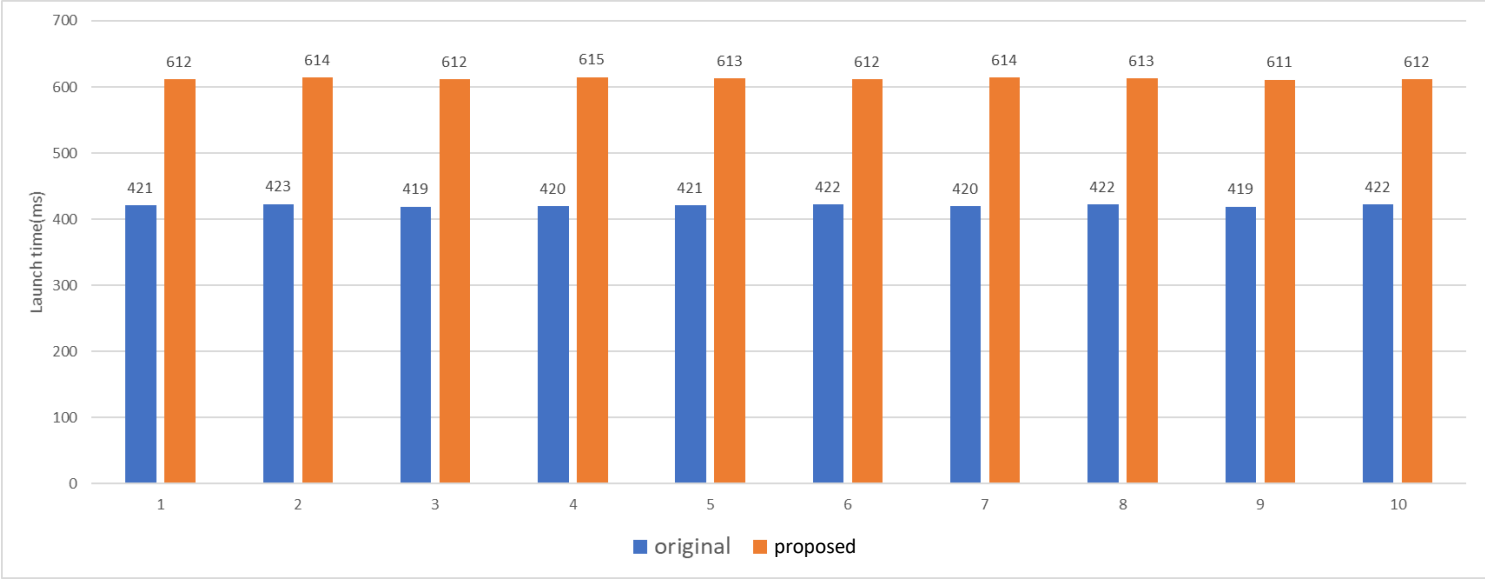
어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법





4.2 Experimental Result

어플리케이션 수준 가상화 기반의 효과적인 안드로이드 앱 역공학 방지 기법



Min. overhead : 190ms

Avg. overhead : 192ms

Max. overhead : 195ms

Original : Guest app을 직접 실행

Proposed : Host app에 의해 Guest app 실행 + decryption

V. Discussion (Limitations)

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

- 일반 상용 앱을 대상으로 제안 기법 적용할 수 있도록 개선 필요
- 가상화 프레임워크에서 동작하는 Guest App들이 동일한 UID를 갖는 문제가 존재
  - Guest apps 간의 Sandbox 우회 문제
  → `secomp` 시스템콜을 이용한 프로세스별 권한 조정
- Host App의 Permissions 상속 문제
  → Guest apps 의 permission을 조사하여 제한하는 메커니즘 구현 필요
- Memory dumping

VI. Conclusion

어플리케이션 수준 가상화 기반의 효과적인
안드로이드 앱 역공학 방지 기법

Summary & Existing problem:

- 앱에는 Business logic, Credentials, Personal Information을 포함되어 있음
- 안드로이드 앱은 코드 변조, 역공학에 취약하여, 지적재산권 및 개인정보 침해 위협 존재
- 코드 난독화, LVL, SafetyNet 우회 가능
- 기존 패커들은 Stub dex 및 library 추가를 위한 repackaging 필요

Our Approach: 어플리케이션 수준 가상화 및 암호화를 이용한 안드로이드 앱 역공학 방지

- No firmware modification
- No app modification
- No installation

Thank you

Samsung
Security Tech
Forum 2019

SAMSUNG
Research