

안드로이드 악성코드 분석

[2] 분석 들어가기

2014-01-16



HNS Pioneer of Security Research
<http://www.hacknsecurity.com>

내용 요약

디바이스에 설치된 애플리케이션의 APK 파일을 추출 하기 위해서는 ADB 툴을 이용하는 방법과 파일 관리 애플리케이션을 이용하는 방법이 있음

ADB는 안드로이드 SDK에 포함되어 있는 툴로, 디바이스에 설치되어있는 기본 탑재 및 사용자가 설치한 APK 파일을 로컬 디렉토리로 다운받을 수 있게 함

APK 파일은 ZIP 파일 형식의 압축 패키지로 파일 압축 유틸리티를 이용해 그 구성 요소를 확인할 수 있음

APK 파일에는 오리지널 자바 소스가 컴파일 된 classes.dex 파일, 기능 실행 권한 등 기본적인 정보를 담고 있는 AndroidManifest.xml 파일, 서명과 인증서 정보가 있는 META-INF/ 디렉토리 외 각종 리소스 데이터가 저장되어 있음

단, classes.dex 파일을 비롯해 상당수의 중요 정보 파일은 컴파일 되어 있어서 단순 압축 해제를 하는 것만으로는 그 내용을 볼 수 없고, 각 구성요소를 디컴파일 하여 그 기능을 파악해야 함

목차

1. 개요

2. APK 파일 추출

2.1. ADB

2.2. 파일 관리 애플리케이션

3. APK 파일 구조

4. 결론

1. 개요

본 문서는 '안드로이드 악성코드 분석' 시리즈 중 두 번째인 '[2] 분석 들어가기'로, 본격적인 분석을 하기 위해 APK 파일을 디바이스에서 추출하는 과정과 APK 파일 구조에 대해 설명함

'안드로이드 악성코드 분석' 시리즈는 다음 순서로 해당 내용을 다룰 예정

- [1] 현황 - 안드로이드 악성코드 현황
- [2] 분석 들어가기 - APK 파일 구조, ADB 사용 등 분석에 필요한 기본 지식 및 방법
- [3] 기본 분석 - 주요 안드로이드 악성코드 샘플 파일을 대상으로 정적 분석, 동적 분석
- [4] 심화 분석 - 난독화된 APK 파일 분석 및 암호화, 시스템 감염, 최근 안드로이드 취약점
- [5] 방지 방법 - 각종 검증 및 탐지 서비스 등 악성코드 방지법

2. APK 파일 추출

안드로이드 모바일 디바이스에 설치된 애플리케이션을 분석하기 위해서는 해당 애플리케이션의 APK(Application Package File) 파일이 필요함

APK는 구글 안드로이드 운영체제에서 애플리케이션 배포 및 설치에 사용되는 ZIP 파일 형식의 압축 패키지로, 이 파일 안에는 소스코드 등 애플리케이션 작동에 관련된 모든 구성요소가 포함되어 있음

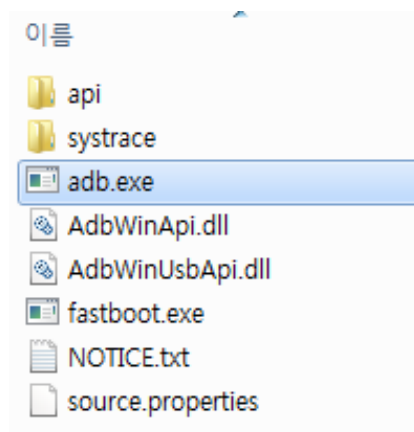
직접 온라인 상에서 APK 파일을 찾아 다운받는 법 외에 디바이스에 설치된 애플리케이션의 APK 파일을 추출하는 방법으로는 ADB 툴을 이용하는 방법과 파일 관리 애플리케이션을 이용해 백업하는 방법이 있음

2.1. ADB

ADB(Android Debug Bridge)는 안드로이드 SDK¹에 포함된 툴로, 다양한 명령어를 이용하여 USB로 연결된 디바이스나 가상 디바이스(AVD)와 통신할 수 있음

ADB 툴(adb.exe)은 <SDK 설치경로>/platform-tools/ 에서 찾을 수 있는데, ADB를 포함한 각종 SDK 툴을 실행하기 위해서는 시스템에 자바(JDK 또는 JRE)가 설치되어 있어야 함

¹ SDK 다운로드 <http://developer.android.com/tools/help/adb.html>



(1) 터미널이나 CMD 등 커맨드 라인 셸에서 ADB를 실행하고 USB로 연결된 디바이스에 접속
\$ adb shell

이때 연결되는 디바이스는 'USB 디버깅' 모드가 활성화 되어 있어야 함

```
C:\Users\Wdell\Downloads\adt-bundle-windows-x86_64-20131030\adt-bundle-windows-x86_64-20131030\platform-tools>adb shell
shell@android:/ $
```

(2) 디바이스에 설치된 모든 APK 파일과 그 디렉토리 목록을 확인
\$ pm list packages -f

사용자가 설치한 애플리케이션은 /data/app 디렉토리에, 기본 탑재된 애플리케이션(Gmail, 메모, 연락처 등)은 /system/app에 위치해있음

참고로 루팅을 하지 않은 디바이스에서는 /data/app 디렉토리에 있는 APK 파일을 pull을 할 수는 있지만, 직접적인 접근은 불가능함(drwxrwx--x)

따라서, 직접 접근을 하기 위해서는 디바이스를 루팅한 뒤, ADB를 루트 권한으로 실행(\$ su)하여야 함²

```
shell@android:/ $ pm list packages -f
pm list packages -f
package:/system/framework/framework-res.apk=android
package:/system/app/TStock.apk=com.ATsolution.TStock
package:/system/app/AhnLabU3Mobile.apk=com.android.ahnmobilesecurity
package:/system/app/Tag.apk=com.android.apps.tag
```

² <http://stackoverflow.com/questions/1043322/why-do-i-get-access-denied-to-data-folder-when-using-adb>

```
package:/data/app/com.kakao.talk-2.apk=com.kakao.talk
package:/system/app/KyoboBookStore.apk=com.kyobo.ebook.ebookcase.s2
package:/data/app/com.metago.astro-1.apk=com.metago.astro
package:/system/app/AppleMint.apk=com.monotype.android.font.applemint
package:/system/app/ChocoEUKor.apk=com.monotype.android.font.chococooky
```

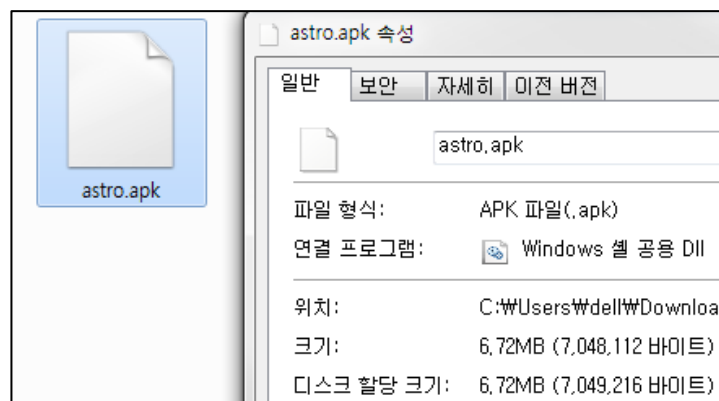
(3) ADB 종료

\$ exit

(4) 원하는 APK 파일을 선택한 후 사용자 로컬 디렉토리로 추출(복사)

\$ adb pull <디렉토리>/<패키지명.apk> <로컬 패키지명.apk>

```
C:\Users\Wdell\Downloads\adt-bundle-windows-x86_64-20131030\adt-bundle-windows-x86_64-20131030\sdk\platform-tools>adb pull /data/app/com.metago.astro-1.apk C:\Users\Wdell\Downloads\Android\apkfiles\astro.apk
4153 KB/s (7048112 bytes in 1.657s)
```



2.2. 파일 관리 애플리케이션

SDK를 설치하지 않고도 파일 관리 애플리케이션을 이용해 조금 더 간편하게 APK 파일을 추출할 수 있음

단, ADB 툴과 달리 /system/app에 있는 기본 탑재 애플리케이션은 추출할 수 없다는 단점이 있음

안드로이드에는 'ASTRO 파일 관리자', '파일 익스퍼트', 'ES 파일 탐색기' 등 여러 파일 관리 애플리케이션이 있는데 사용 방식은 대부분 비슷함³

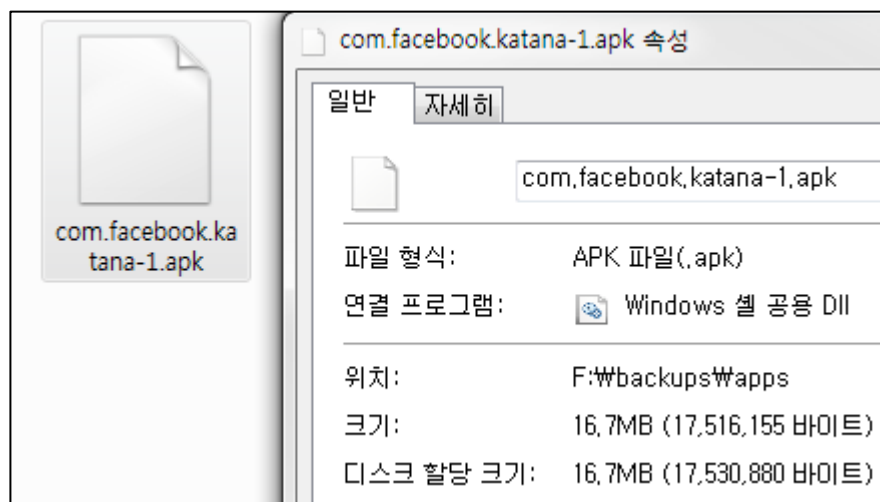
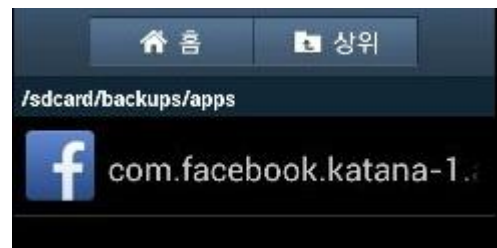
³ 여기에서는 'ASTRO 파일 관리자'를 이용함



(1) 애플리케이션 관리자 메뉴에서 원하는 애플리케이션을 선택한 뒤 백업



(2) 디바이스 내장 메모리(여기에서는 /sdcard/backups/apps)에 해당 애플리케이션의 APK 파일이 저장됨

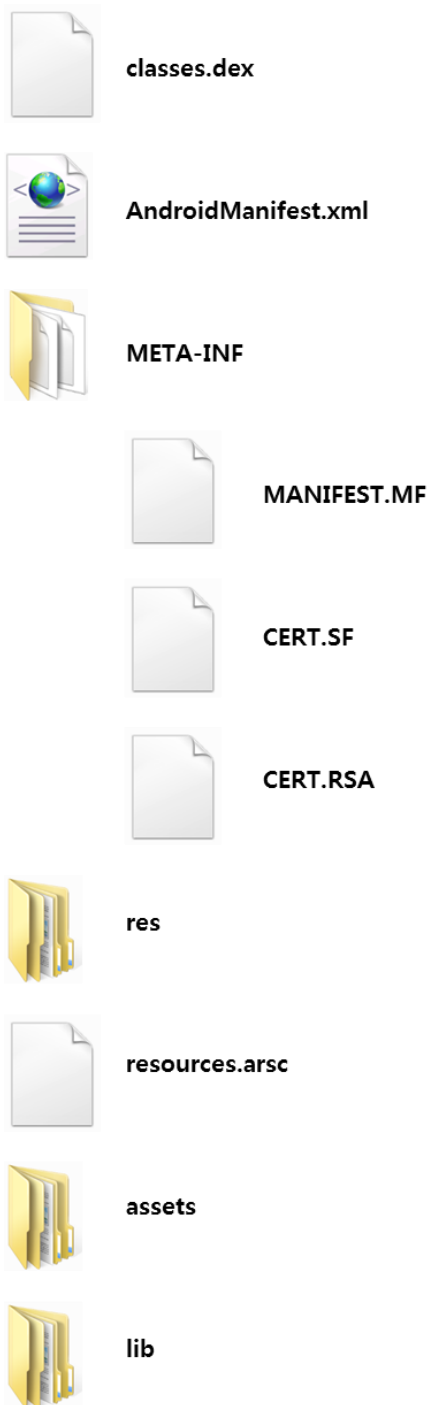


3. APK 파일 구조

APK 파일은 ZIP 파일 형식의 압축 패키지이기 때문에 7-Zip이나 WinZip과 같은 일반적인 파일 압축 해제 유틸리티를 이용해 쉽게 그 구성 요소를 확인할 수 있음

단, 단순히 압축 해제를 한 것만으로는 암호화된 파일 내용을 바로 볼 수는 없고, 이를 보기 위해서는 Apktool 등을 이용한 디컴파일 과정이 필요함('안드로이드 악성코드 분석 - [3] 기본 분석'에서 상세 설명)

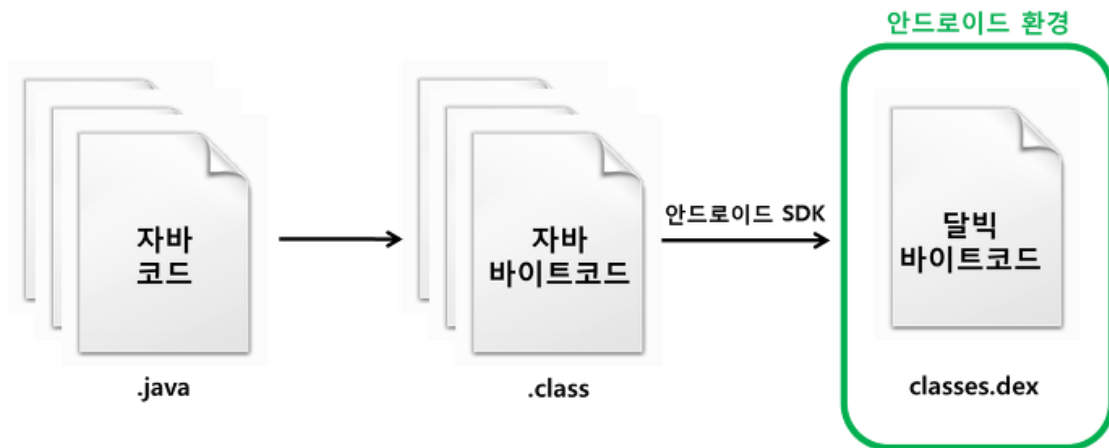
애플리케이션 생성 과정에 따라 포함하고 있는 리소스 디렉토리 등 구성 요소간 다소 차이는 있을 수 있으며 아래는 일반적인 APK 파일에 공통적으로 포함되어 있는 파일과 디렉토리임



(1) classes.dex

자바 코드(.java)를 컴파일 해서 만들어진 자바 바이트코드(.class)가 안드로이드의 달빅 가상머신(Dalvik VM) 환경에서 실행될 수 있도록 하나의 달빅 실행파일 포맷(.dex)으로 만들어져 저장됨

즉, classes.dex는 애플리케이션 소스 정보를 담고 있는 핵심이라고 할 수 있음



(2) AndroidManifest.xml

패키지 이름, 버전, 구성 컴포넌트, 기능 실행 권한, 참조 라이브러리 등 애플리케이션에 대한 기본적인 정보를 명시하고 있는 XML 형식 파일⁴

APK 파일 내에서는 암호화 인코딩(컴파일)된 상태로 저장되기 때문에, 내용을 보기 위해서는 디컴파일을 하여야 함

(3) META-INF/

APK 파일의 무결성을 확인할 수 있는 서명 데이터 MANIFEST.MF, CERT.SF, CERT.RSA 파일이 저장되는 디렉토리⁵

eclipse에서 APK 파일을 생성할 때, 패키지에 포함된 모든 파일에 대한 각 해쉬값을 Base64로 계산하여 META-INF 디렉토리에 저장함

그리고 생성된 APK 파일을 설치하기 전에 디바이스 단에서 다시 한 번 모든 파일의 해쉬값을 계산하여 META-INF 디렉토리에 저장된 값과 비교하는데, 만약 이 값이 같지 않을 경우 해당 APK 파일 설치가 불가능함

즉, 사용자가 서명 이후에 이미지나 소스코드 등을 임의로 변경할 수 없으므로 APK 파일이 변조되는 것을 방지할 수 있음⁶

⁴ <http://developer.android.com/guide/topics/manifest/manifest-intro.html>

⁵ 서명되지 않은 APK 파일에는 이러한 파일이 없고, 반대로 기존에 만들어진 APK의 서명 정보를 지우려면 META-INF/ 내 항목을 삭제하면 됨

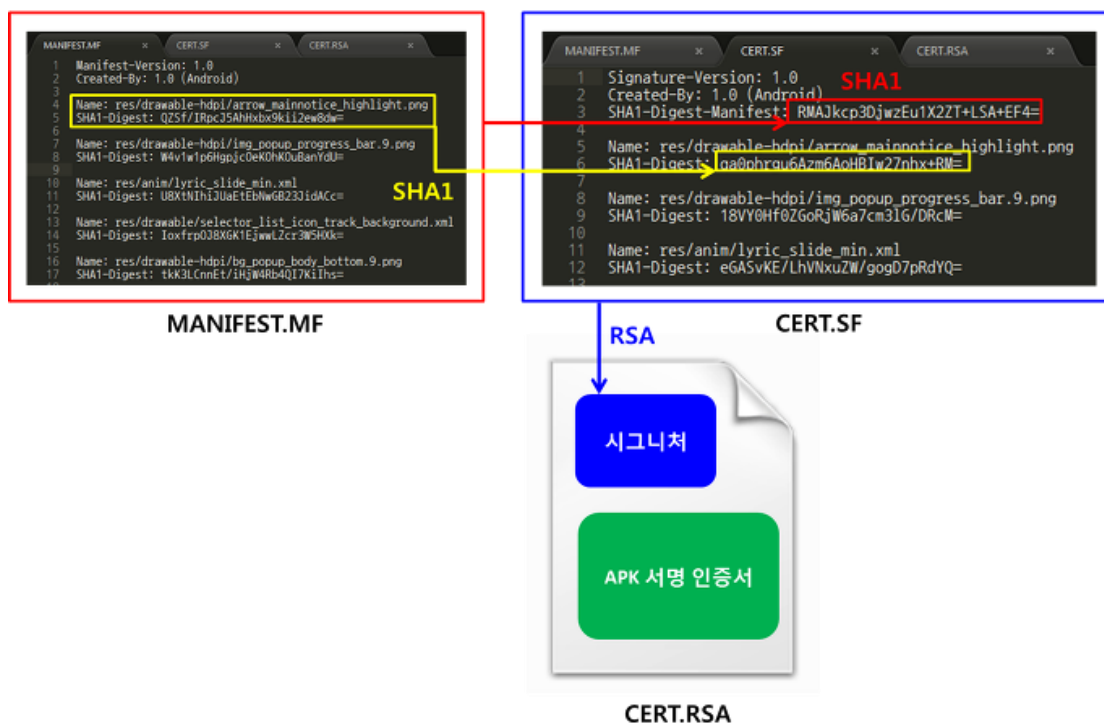
⁶ <http://android-anything.diandian.com/post/2011-09-28/5377936>
<http://www.nowherenearithaca.com/2011/12/difference-between-signed-and-unsigned.html>

META-INF/MANIFEST.MF - APK 파일이 가지고 있는 이미지, XML 파일 등의 리소스와 이에 대한 SHA-1 해쉬값을 저장하고 있음

META-INF/CERT.SF - MANIFEST.MF에 나와 있는 해쉬값을 토대로 다시 한 번 인코딩한 값을 저장하고 있음

META-INF/CERT.RSA - CERT.SF 파일에 대한 RSA 시그니처와 애플리케이션 파일의 서명 인증서 정보를 저장하고 있음

META-INF 디렉토리



(4) res/

애플리케이션 작동에 필요한 각종 리소스(텍스트 스트링, 이미지, 레이아웃 XML, 소리 파일 등)가 있는 디렉토리로, 컴파일되지 않은 상태로 res 하위에 저장됨⁷

(5) resources.arsc

ARSC(Application Resource) 파일로, 애플리케이션 작동에 필요한 각종 리소스 중 일부가 빠르게 로딩될 수 있도록 바이너리 포맷으로 컴파일되어 저장됨⁸

⁷ <http://hyeonstorage.tistory.com/152>

⁸ <http://www.openthefile.com/ext/arsc/547>

(6) assets/

리소스처럼 애플리케이션 작동에 필요한 데이터이지만 그 사용 빈도가 낮은 것들로, 컴파일되지 않은 원시 파일 그대로 저장됨⁹

(7) lib/

디바이스 프로세서(ARM, x86 등)에 맞춰진 코드가 컴파일된 상태로 저장됨

4. 결론

안드로이드 APK 파일을 ADB나 파일 관리 애플리케이션을 이용하여 추출한 뒤 압축을 풀어보면 비교적 쉽게 그 파일 구조를 파악할 수 있음

단, 압축 해제된 파일의 대부분은 암호화 되어 있기 때문에 바로 그 내용을 살펴볼 수는 없음

APK 파일 구성 요소 중 가장 핵심 파일은 개발자(공격자)가 직접 작성한 classes.dex 파일로, 이를 가지고 오리지널 자바 소스로 디컴파일하여 기능을 파악하는 것이 관건임

⁹ <http://underclub.tistory.com/340>