brightsight®

the number one
security lab
in the world

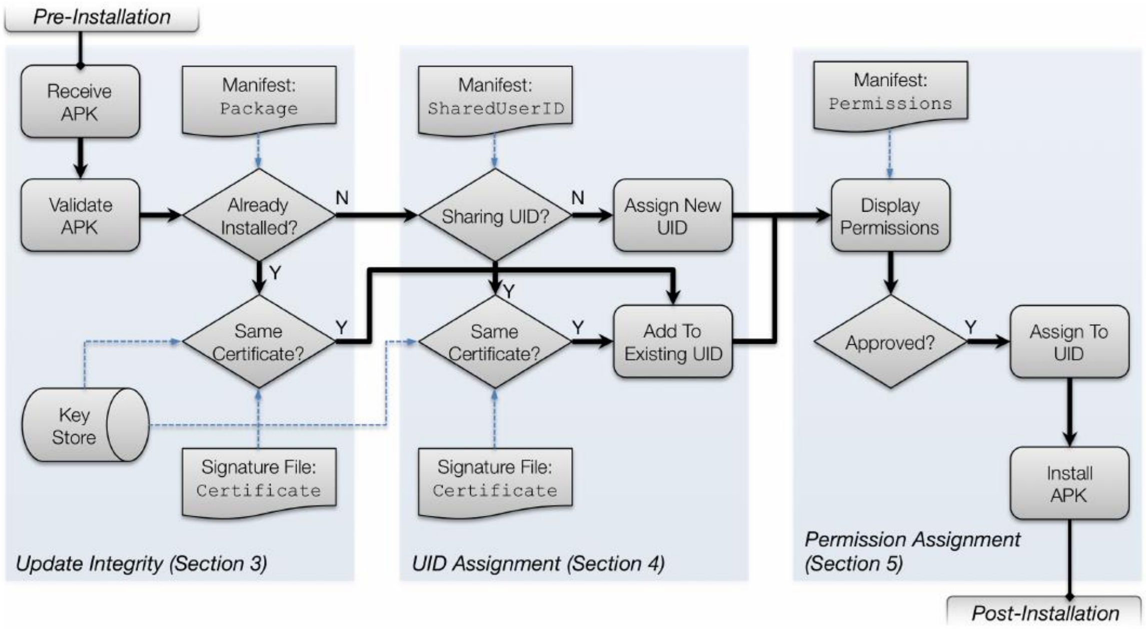
Android Security

FW/APP Authentication & Signing (B4/B4.1/B4.2)

Shiqi Li / Brightsight

Security Measures in Android

- Package installation (APK)
  a. apk validation
  b. pkg name check e.g. com.google.android.music
  c. pkg certificate check i.e. PKs compared
  d. Assign UID
  e. apk installation

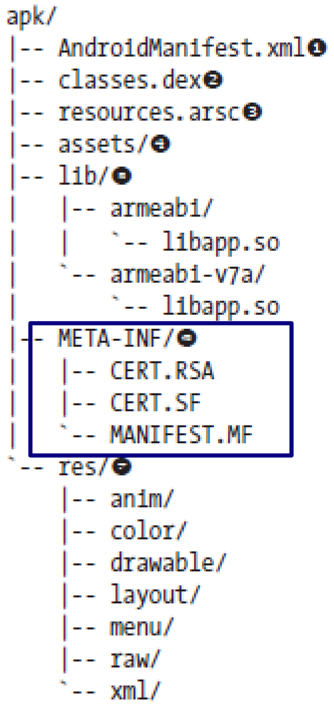


Security Measures in Android

- apk contents

APK format (ZIP archive ⇒ extension of JAR ⇒ extension of ZIP)

- **lib** - applications that use the native libraries via JNI. A sub-folder for each supported platform architecture (arm*)
- **resources.arsc -** binary with compiled resources such as strings and style (think CSS)
- **resources folder -** other references resources like images, animations etc. (sub-folder for each type)
- META-INF - apk securitah. Verification info. for every file outside of this directory is here.



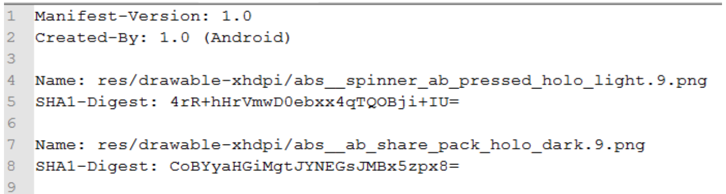
```
apk/
|-- AndroidManifest.xml❶
|-- classes.dex❷
|-- resources.arsc❸
|-- assets/❹
|-- lib/❺
|   |-- armeabi/
|   |   `-- libapp.so
|   `-- armeabi-v7a/
|       `-- libapp.so
|-- META-INF/❻
|   |-- CERT.RSA
|   |-- CERT.SF
|   `-- MANIFEST.MF
`-- res/❼
    |-- anim/
    |-- color/
    |-- drawable/
    |-- layout/
    |-- menu/
    |-- raw/
    `-- xml/
```

Security Measures in Android

apk validation

- Signing:

- MANIFEST.MF - SHA1 of the actual files

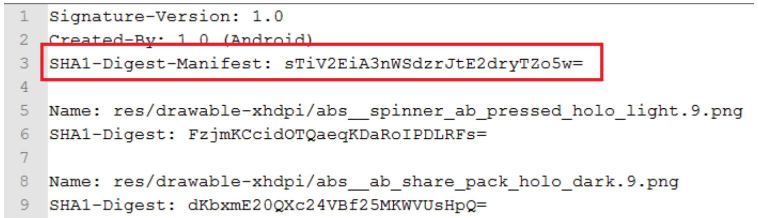


```
Manifest-Version: 1.0
Created-By: 1.0 (Android)

Name: res/drawable-xhdpi/abs__spinner_ab_pressed_holo_light.9.png
SHA1-Digest: 4rR+hHrVmwD0ebxx4qTQOBji+IU=

Name: res/drawable-xhdpi/abs__ab_share_pack_holo_dark.9.png
SHA1-Digest: CoBYyaHGiMgtJYNEGsJMBx5zpx8=
```

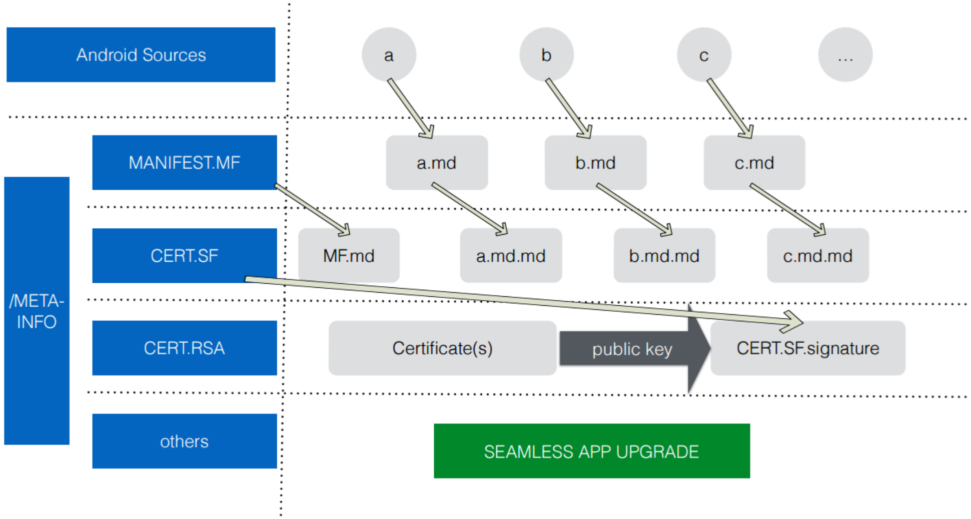
MANIFEST.MF



```
Signature-Version: 1.0
Created-By: 1.0 (Android)
SHA1-Digest-Manifest: sTiV2EiA3nWSdzrJtE2dryTZo5w=

Name: res/drawable-xhdpi/abs__spinner_ab_pressed_holo_light.9.png
SHA1-Digest: FzjmKCcidOTQaeqKDaRoIPDLRFs=

Name: res/drawable-xhdpi/abs__ab_share_pack_holo_dark.9.png
SHA1-Digest: dKbxmE20QXc24VBf25MKWVUsHpQ=
```

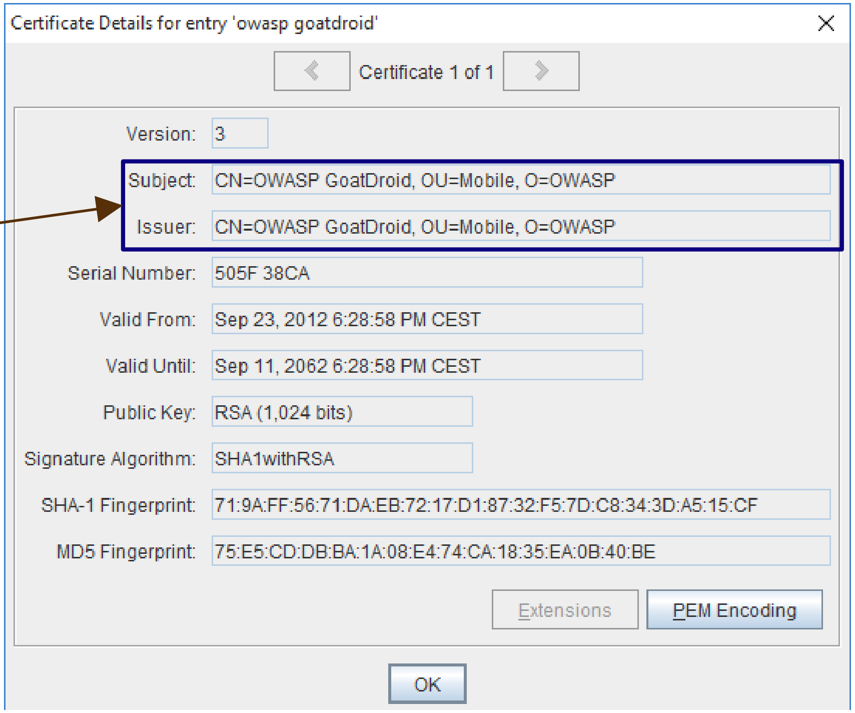


- CERT.RSA (an actual PKCS#7 certificate containing (digest algorithm + signature value) + signing certificate) in fact in CMS format: The **Cryptographic Message Syntax** (CMS) is the IETF's standard for cryptographically protected messages (wiki)

Security Measures in Android

apk validation

- CERT.RSA
  - can be self-signed
    - subject == issuer



Certificate Details for entry 'owasp goatdroid'

Version: 3

Subject: CN=OWASP GoatDroid, OU=Mobile, O=OWASP

Issuer: CN=OWASP GoatDroid, OU=Mobile, O=OWASP

Serial Number: 505F 38CA

Valid From: Sep 23, 2012 6:28:58 PM CEST

Valid Until: Sep 11, 2062 6:28:58 PM CEST

Public Key: RSA (1,024 bits)

Signature Algorithm: SHA1withRSA

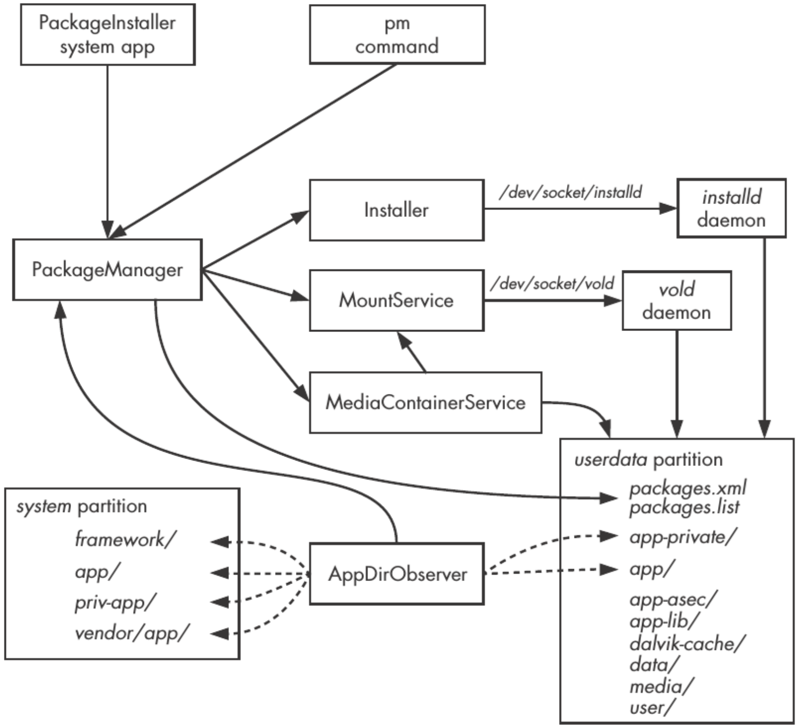
SHA-1 Fingerprint: 71:9A:FF:56:71:DA:EB:72:17:D1:87:32:F5:7D:C8:34:3D:A5:15:CF

MD5 Fingerprint: 75:E5:CD:DB:BA:1A:08:E4:74:CA:18:35:EA:0B:40:BE

Security Measures in Android

apk installation - high level process

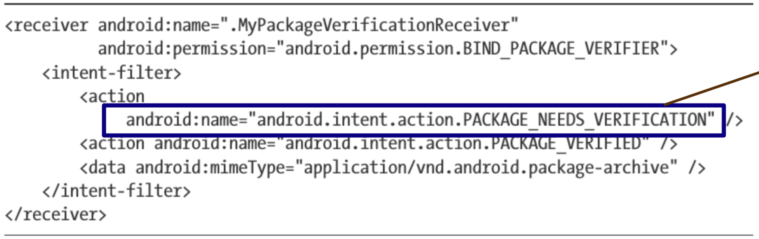
- a. PackageInstaller system app handles
  - Installation GUI and PackageManager service
- b. PackageManager service
  - runs as "system" user
  - hands off verification to a verifier
- c. pm binary implements overloaded install()
- d. AppDirObserver



Security Measures in Android

apk installation - apk verification agents

- Android AOSP does not ship with a verifier! However Google Play serves as one in most devices
- apk verification needs:
  - 1 *required verifier* AND
  - 0 or more *sufficient verifiers*



```
<receiver android:name=".MyPackageVerificationReceiver"
          android:permission="android.permission.BIND_PACKAGE_VERIFIER">
    <intent-filter>
        <action
            android:name="android.intent.action.PACKAGE_NEEDS_VERIFICATION" />
        <action android:name="android.intent.action.PACKAGE_VERIFIED" />
        <data android:mimeType="application/vnd.android.package-archive" />
    </intent-filter>
</receiver>
```

(signature | system) permission

Security Measures in Android

apk installation - ways to install apk

1. Via an application store client (such as the Google Play Store) ⇒ most popular

2. Load apk and open (if the “Unknown sources” option in system settings is enabled). ⇒ app *sideloading* through *PackageInstaller* system app

3. From a USB-connected computer with the adb install Android SDK command which, in turn invokes the pm command line utility with the install parameter. This method is used mostly by application developers.

Security Measures in Android

- Vulnerabilities
  - Hide and Ignite - insert malicious code in META-INF files and run it dynamically later [3]
  - Masterkey - insert malicious “duplicate” resource in archive. first one is checked, second is extracted instead! [4]

Evaluation Concerns

- Firmware authentication
  - System Services,
  - System Applications,
  - Dynamic Link Libraries,
  - Boot secripts (init)
  - Filesystem Integrity
  - System configuration files (SELinux security policy, MAC, …)
- Verification tests
  - try to install arbitrary apk
  - uninstall existing app (if possible) and install fake app with same pkg.name
- Source code review
  - Check for existance of Google Play alikes ⇒ point to system app that acts as primary verifier
- Alternative possible solutions
  - Vendors strip google play from AOSP and make their own *required verifier* where they can use their own CA certs to verify apps (PKI like).

PCI PTS security concerns

DTR B4/B4.1

The evidences shall

- show that the device cryptographically authenticates firmware/application integrity
- show that the device authenticate external components for FW/SW update
- show that device rejects unauthorized firmware/application
- show which component performs authentication of firmware/application
- Controls provide for unique accountability and utilize key sizes appropriate for algorithms
- provide complete table of processing elements (as also given in A4)
- show, If applicable, detail various types update images differentiated from each other
- show in source code that
  - FW/SW are authenticated by secure firmware
  - if HMAC is used no leaking of timing information
  - if CBD MAC used, detail method to mitigate vulnerabilities
- show how Public keys are loaded during manufacturing, and how default values are changed

PCI PTS security concerns

DTR B4.2

The evidences shall show that

- any unsigned files cannot be launched and will be deleted
- unsigned files cannot affect device security
- loading unsigned files cannot affect device security

References

1. Barrera, David, et al. "Understanding and improving app installation security mechanisms through empirical analysis of android." *Proceedings of the second ACM workshop on Security and privacy in smartphones and mobile devices*. ACM, 2012.
2. Elenkov, Nikolay. *Android Security Internals: An In-depth Guide to Android's Security Architecture*. No Starch Press, 2014.
3. Xiao et al. - What can you do to an apk without its private key except repacking? Black Hat Mobile 2015, London. https://www.blackhat.com/docs/ldn-15/materials/london-15-Xiao-What-Can-You-Do-To-An-APK-Without-Its-Private-Key-wp.pdf
4. Android Master Key vulnerability. http://resources.infosecinstitute.com/android-master-key-vulnerability-poc/

Questions?