Keymaster and Gatekeeper

Joakim Bech and Victor Chong
1. Introduction to Keymaster and Gatekeeper
2. OP-TEE enablement
   a. Current status
   b. How to try it out
   c. What is next?
Keymaster

- Access control on keys - Who, when?
- Version binding - Rollback prevention using key + OS patch level
- Client binding - Associate a key with a certain application
- Expiration - How long is a key valid?
- Root of Trust Binding - All keys must be bound to the ROTK/HUK
- Velocity - Prevents brute force attacks
- Attestation - Ensure that keys are stored in hardware backed environment
- Authorization Tags - Set of properties and types
- Android O: Mandatory!
Gatekeeper

- Password and pattern authentication
- Enrolls and verifies passwords
- Leverages hardware-backed secret key
- Responsible for throttling brute force attacks
- Sign authentication attestations and sends them to Keymaster
Architecture Overview

Image source: https://source.android.com/security/
OP-TEE: KM/GK

Rich OS / REE

- Native Applications
- Keymaster / GK
- TEE supplicant
- GlobalPlatform TEE Client API
- Networking
- TEE framework and driver
- Storage / RPMB

TrustZone OS / TEE

- Keymaster
- Gatekeeper
- GlobalPlatform TEE Internal API
- TEE Core
- TEE functions / libraries
- HAL
- Hardware resources

Secure Monitor

USER MODE

PRIVILEGED MODE
AOSP components

Image source: https://source.android.com/security/
Agenda

1. Introduction to Keymaster and Gatekeeper
2. OP-TEE enablement
   a. Current status
   b. How to try it out
   c. What is next?
Where to start?

- Starting from scratch (more or less)
  - Sept ‘17- an ‘18: On the roadmap (related PKCS#11 work took precedence)
  - Feb-April ‘18: Running with modest pace
  - May-June ‘18: Full speed, several engineers involved

- Third party company donated a KM2/GK implementation for OP-TEE
  - Sanity check, test, review == good! →
    - Throw away what we had been doing prior to this
  - July ‘18 - until now: Integrate new KM2/GK solution
    - Integration into HiKey 6220 build system
    - Fixes/changes for Treble enablement
    - Solving VTS test regressions
    - Refactor code and cleanup for upstreaming
Current status

- All(*) KM/GK VTS test cases are passing
- Creating a stable manifest based where changes are on forks
- Refactor code
  - Remove pseudo-TA supporting KM
  - Move pseudo-TA functionality into the KM TA

(*) After upgrading to a new AOSP version we have two regressions
How to try it?

- Build

```bash
$ git clone https://github.com/linaro-swg/optee_android_manifest -b yvr18
$ cd optee_android_manifest
$ ./sync.sh -v p -bm pinned-manifest-stable_yvr18.xml
$ ./build-p.sh
```
How to try it?

- Flash (HiKey 6220)
  - Put board in recovery mode by connecting jumpers 1-2 and 3-4
  - Run command
    
    $$ \texttt{cp -a out/target/product/hikey/*} . \texttt{img device/linaro/hikey/installer/hikey/}$$
    
    $$ \texttt{sudo ./device/linaro/hikey/installer/hikey/flash-all.sh /dev/ttyUSB<x>}$$
    
    $x$ = device number that appears after rebooting with the 3-4 jumper connected
    
    e.g.
    
    $$ \texttt{sudo ./device/linaro/hikey/installer/hikey/flash-all.sh /dev/ttyUSB0}$$
  - Power off board
  - Remove jumper 3-4
  - Power on board
How to try it?

- Test

$ adb root
$ adb shell /data/nativetest64/VtsHalKeymasterV3_0TargetTest/VtsHalKeymasterV3_0TargetTest

# help
$ adb shell /data/nativetest64/VtsHalKeymasterV3_0TargetTest/VtsHalKeymasterV3_0TargetTest --help

NOTE

- Run above commands from a terminal that is NOT the board console
- This is test build so there will be lots of debug prints!
- Test can take up to an hour to complete
How to try it?

NOTE (cont.)

- The `vendor.hwcomposer-2-1` service repeatedly fails to start and continuously clutters the console with error messages.
  - This is **NOT** an OP-TEE related issue
  - Linaro AOSP engineers currently working on fix
  - Temporary workaround - delete `/vendor/etc/init/android.hardware.graphics.composer@2.1-service.rc` and reboot
How to try it?

- Pre-built binaries

http://people.linaro.org/~victor.chong/prebuilt/pie/kmgk/yvr18/

<table>
<thead>
<tr>
<th>Name</th>
<th>Last modified</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Directory</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boot.img.xz</td>
<td>2018-09-07 07:50</td>
<td>7.1M</td>
<td></td>
</tr>
<tr>
<td>cache.img.xz</td>
<td>2018-09-07 07:50</td>
<td>732</td>
<td></td>
</tr>
<tr>
<td>fip.bin</td>
<td>2018-09-07 07:53</td>
<td>1.6M</td>
<td></td>
</tr>
<tr>
<td>flash-all.sh</td>
<td>2018-09-07 07:54</td>
<td>2.1K</td>
<td></td>
</tr>
<tr>
<td>hisi-idt.py</td>
<td>2018-09-07 07:53</td>
<td>8.4K</td>
<td></td>
</tr>
<tr>
<td>l-loader.bin</td>
<td>2018-09-07 07:53</td>
<td>133K</td>
<td></td>
</tr>
<tr>
<td>nvme.img</td>
<td>2018-09-07 07:53</td>
<td>128K</td>
<td></td>
</tr>
<tr>
<td>ptable-aosp-4g.img</td>
<td>2018-09-07 07:53</td>
<td>17K</td>
<td></td>
</tr>
<tr>
<td>ptable-aosp-8g.img</td>
<td>2018-09-07 07:53</td>
<td>17K</td>
<td></td>
</tr>
<tr>
<td>ramdisk.img.xz</td>
<td>2018-09-07 07:50</td>
<td>1.5M</td>
<td></td>
</tr>
<tr>
<td>system.img.xz</td>
<td>2018-09-07 07:52</td>
<td>338M</td>
<td></td>
</tr>
<tr>
<td>userdata.img.xz</td>
<td>2018-09-07 07:52</td>
<td>3.9M</td>
<td></td>
</tr>
<tr>
<td>vendor.img.xz</td>
<td>2018-09-07 07:52</td>
<td>4.9M</td>
<td></td>
</tr>
</tbody>
</table>
Sample test output

[==========] Running 106 tests from 12 test cases.
[----------] Global test environment set-up.

[----------] 1 test from KeymasterVersionTest
[  OK ] KeymasterVersionTest.SensibleFeatures (1 ms)
[ RUN ] NewKeyGenerationTest.Rsa
[  OK ] NewKeyGenerationTest.Rsa (520247 ms)
<snip>
[ RUN ] KeyDeletionTest.DeleteAllKeys
[  OK ] KeyDeletionTest.DeleteAllKeys (0 ms)
[----------] 3 tests from KeyDeletionTest (11384 ms total)

[----------] Global test environment tear-down
[==========] 106 tests from 12 test cases ran. (1295946 ms total)
[  PASSED ] 106 tests.
What is next?

● Refactor the pseudo and dynamic Trusted Application
● Improve SE policies → allow running as non-root user
● Upstream Keymaster + Gatekeeper (and OP-TEE?) to the AOSP project
● Keymaster v4
● Fingerprintd?
Backup
Keymaster Functions

getHardwareFeatures()    begin()
addRngEntropy()           update()
generateKey()             finish()
getKeyCharacteristics()   abort()
importKey()               
exportKey()               
deleteKey()               
deleteAllKeys()           
destroyAttestationIds()   

Gatekeeper Functions

enroll()
verify()