Android System Updates

Lecture 8

Security of Mobile Devices

2018

Outline

Bootloader

- Low-level program executed when device is powered
- Initialize hardware
- Identify and load the main OS

Bootloader

- Usually locked
  - Boot only OS image signed by device manufacturer
  - Trusted and unmodified OS runs on the device
- Unlocking the bootloader is needed for:
  - Installing a custom Android build
  - Installing a recent Android version on an old device

Unlocking the Bootloader via Fastboot

- Start device in fastboot mode:
  - `adb reboot bootloader`
  - Or by pressing a key combination while booting
- Connect mobile device to host via USB
- In CLI:
  - `fastboot oem unlock`

Unlocking the Bootloader via Fastboot

- Confirmation screen
  - Warning regarding installing untested third-party builds
  - Warning regarding deleting all your data
- Locking again:
  - `fastboot oem lock`
  - Prevents booting third-party builds
- *tampered* flag
  - Set when unlocking the bootloader for the first time
  - Disallow certain operations / display warning

OEM unlocking via Settings

- Enable Developer options
  - Press a number of times on the Build number
- Enable OEM unlocking from Developer options

Outline

Fastboot

- Original purpose: write device partitions
  - Partition image sent to the bootloader
  - Written to a specific block device
- Porting Android to a new device
- Factory reset
  - Writing partition images from the device manufacturer

Samsung Galaxy S7 Edge

```
hero2lte:/ # ls -l /dev/block/platform/155a0000.ufs/by-name/
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 BOOT -> /dev/block/sda5
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 BOTA0 -> /dev/block/sda1
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 BOTA1 -> /dev/block/sda2
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 CACHE -> /dev/block/sda15
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 CPEFS -> /dev/block/sdd1
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 CP_DEBUG -> /dev/block/sda17
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 DNT -> /dev/block/sda10
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 EFS -> /dev/block/sda3
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 HIDDEN -> /dev/block/sda16
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 OTA -> /dev/block/sda7
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 PARAM -> /dev/block/sda4
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 PERSDATA -> /dev/block/sda13
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 PERSISTENT -> /dev/block/sda11
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 RADIO -> /dev/block/sda8
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 RECOVERY -> /dev/block/sda6
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 STEADY -> /dev/block/sda12
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 SYSTEM -> /dev/block/sda14
lrwxrwxrwx 1 root root 15 2018-01-06 17:33 TOMBSTONES -> /dev/block/sda9
lrwxrwxrwx 1 root root 16 2018-01-06 17:33 USERDATA -> /dev/block/sda18
```

Partition Layout

- Most partitions - device-specific and proprietary data
- `aboot` - bootloader
- `modem` - baseband software
- `boot` - kernel and rootfs RAM disk image
- `system` - all other system files
- `userdata` - user files
- `cache` - temporary files and OTA images
- `recovery` - recovery OS image

Fastboot Protocol

- Over USB
- Host sends commands and data to the bootloader
- Bootloader responds with OKAY, FAIL, INFO or DATA
- Flash or boot custom kernels only if bootloader is unlocked

Fastboot Commands

- `devices` - connected devices that support fastboot
- `getvar` - information about the bootloader
- `reboot` the device
- `reboot-bootloader` - reboot in fastboot mode
- `erase`, `format` a partition

Fastboot Commands - Writting and Booting Images

- `flash` patition image-name - write a disk image to a partition
- `update` zip-file - write multiple partition images
- `flashall` - writes boot.img, system.img and recovery.img to boot, system and recovery partitions
- `flash:raw boot` kernel ramdisk - creates boot image from kernel and RAM disk and writes it to boot partition
- `boot` boot-image - boot an image without writing it to the device
- `boot` kernel ramdisk - boot an image created from kernel and RAM disk

Fastboot Commands - Example

- Pixel XL

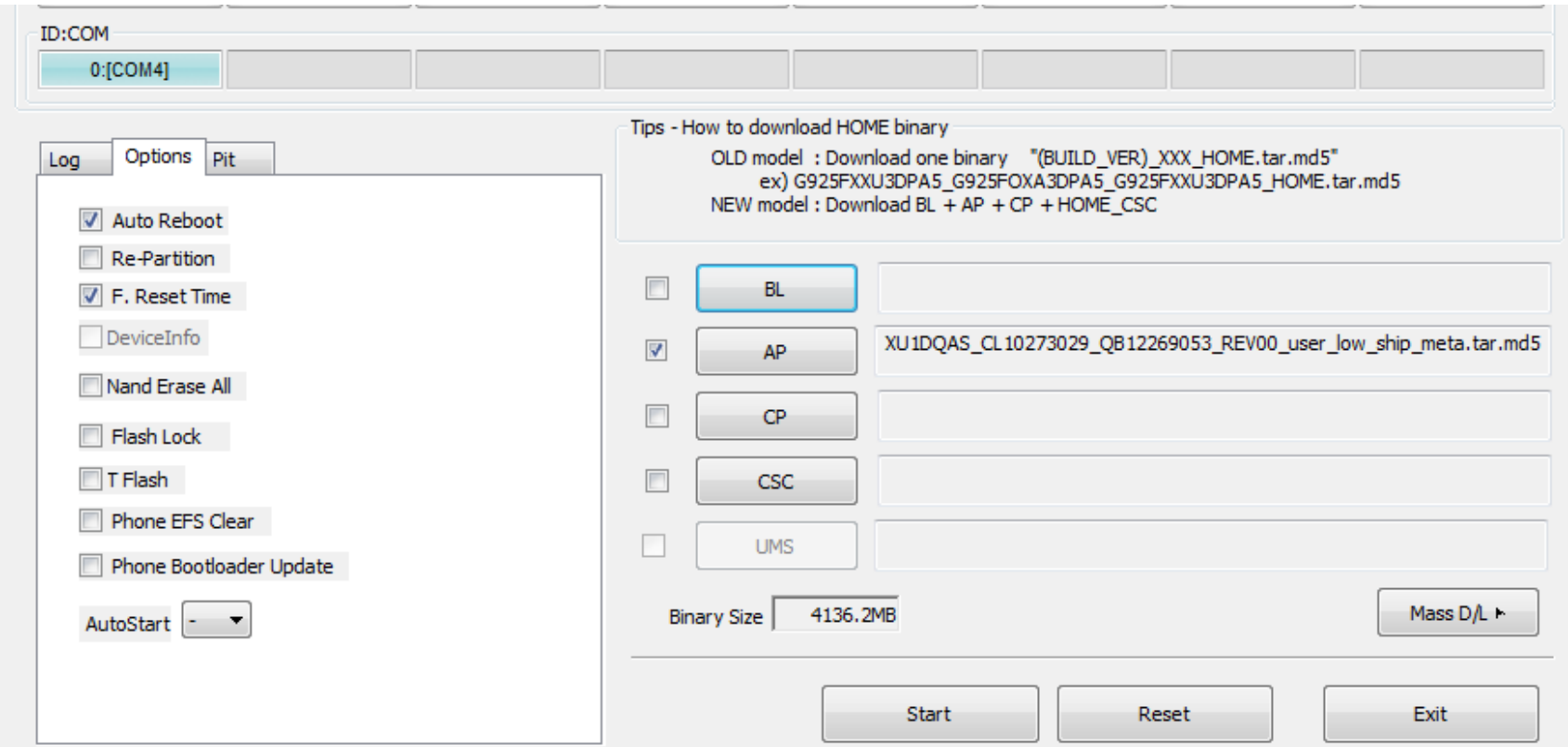
```
$ fastboot devices
HT73L0203468    fastboot

$ fastboot getvar version-bootloader
version-bootloader: 8996-012001-1710040120
finished. total time: 0.050s

$ fastboot getvar version-baseband
version-baseband: 8996-130091-1710201747
finished. total time: 0.050s
```

Writing Images on Samsung Devices

- No fastboot on Samsung devices
- Images written in Download mode with Odin program on Windows



Outline

Recovery OS

- Minimal OS used for factory reset and OTA updates
- Started using:
  - `adb reboot recovery`
  - Or a specific combination of keys
- Stock or custom recovery

Stock Recovery

- Minimal functionality
- Update system software
- Without erasing user data
- Simple UI, operated with buttons
- Menu:
  - reboot
  - apply update from ADB
  - factory reset
  - wipe cache partition

Custom Recoveries

- Created by third party
- Not signed with manufacturer's keys
- Needs an unlocked bootloader
- Boot: `fastboot boot recovery.img`
- Flash `fastboot flash recovery recovery.img`

Custom Recoveries - Features

- Provides additional functionality
  - Full partition backup and restore
  - Root shell with a full set of device management utilities
  - Support for mounting external USB devices
  - Disable OTA package signature checking
    - OS modification
    - Custom OS

TWRP

- Team Win Recovery Project (TWRP)
- Many additional features
- Open Source, actively maintained
- Based on AOSP stock recovery
- Touch screen

TWRP - Features

- Supports encrypted partition backups
- Installs system updates from USB devices
- Backup and restore to/from external devices
- Integrated file manager
- Scripting language to specify actions from main OS

Outline

System Updates

- Updates applied by stock recovery
- OTA updates
  - Main OS downloads the OTA package
  - Instructs recovery OS to apply update
- Tethered updates
  - User downloads OTA package on PC
  - `adb sideload otafile.zip`
- Same updating process, different ways to obtain the package

Controlling Recovery Operations

- Main OS controls recovery through `android.os.RecoverySystem` API
- Writes options to `/cache/recovery/command`
- `/sbin/recovery` process reads the `command` file
- Options:
  - –send-intent
  - –update-package
  - –wipe-data
  - –wipe-cache

Download OTA package

- Device checks OTA servers periodically
- Obtains URL of OTA package and description
- Download package to cache or data partition
- Verify signature
- Ask user to install update

OTA Signature Verification

- Package is code signed
- Signature applied over the whole file
- Verification, in main OS:
  - `verifyPackage()` of `RecoverySystem`
  - Zip file with X.509 certificates
  - Default: `/system/etc/security/otacerts.zip`
- Success -> reboot in recovery mode to apply update

OTA Signature Verification

- Verification in recovery OS:
  - Using set of public keys from recovery OS
  - Extracted from OTA signing certificates
  - In mincrypt format in file `/res/keys`
- Signature algorithms:
  - 2048-bit RSA with SHA-1
  - 2048-bit RSA with SHA-256
  - ECDSA with SHA-256
  - 256-bit EC keys using NIST P-256 curve

System Update General Steps

- Data from OTA package
  - Update boot, system, vendor partitions
- File containing new recovery saved on system partition
- Device rebooted normally
  - Load boot partition
  - That loads system partition
  - Executes binaries from system partition
- Compare recovery partition with the file saved on system
  - Flash recovery with file contents

System Update Process

- Execute the update command from OTA package
  - `META-INF/com/google/android/update-binary`
  - Recovery API version, pipe file descriptor, path to OTA package
- Executes `updater-script` (*edify* language)
  - Sequence of function calls to apply update
  - Copying, deleting, and patching files
  - Formatting and mounting volumes
  - Setting file permissions and SELinux labels

Updater-script (1)

- Mounts system partition
- Verifies device model and current build
  - Incompatible build => soft brick
- Verifies the hash of each patched file
  - OTA - binary patches applied on previous file version
- Verifies partitions without filesystem (e.q. boot, modem)

Updater-script (2)

- Patches all filesystems and partitions
- Extracts new recovery patch in `/system/`
- File owner, permissions and capabilities of patched files
- Set SELinux security labels of all files
  - `u:object_r:system_file:s0`

Updater-script (3)

- Patch baseband software (in modem partition)
- Unmount system partition
- Finally recovery:
  - Clears the cache partition
  - Saves logs to `/cache/recovery`
  - No errors -> reboots in main OS
  - Errors -> Restarts update process after reboot

Update Recovery OS

- Recovery patch extracted by not applied
  - Interrupted recovery update -> unusable system
- Recovery updated from the main OS
  - After main OS update and boot
- `flash_recovery` service in `init.rc`

Update Recovery OS

- `/system/etc/install-recovery.sh` script
- Verifies the recovery partition
- Hash is ok -> Applies patch
- Hash not ok -> Logs message

Block OTA Updates

- From Android 5.0
- Handles entire partition as one file
- Aplies a single binary patch
- Enables dm-verity for system partition

Block OTA - Update Types

- Applies update at block level, not filesystem level
- Full update:
  - Large package, full image
  - Same result as flashing the image with fastboot
- Incremental update:
  - Smaller package, patches

A/B System Updates - Advantages

- Recent method
- Uses 2 sets of patitions called slots
- Workable booting system while OTA update
- Reduce chance of obtaining an unusable device after update
- While the system is running, while user is using the device
  - Reboot to updated disk partition
  - Does not take a longer time

A/B System Updates - Advantages

- OTA update fails -> old OS
- OTA applied but fails to boot -> old OS
- dm-verity error => old image is booted
- Streamed updates
  - No need to download entire package before installation
  - Useful when not enough free space

A/B System Updates

- Two sets of partitions called slots (A and B)
- System runs from current slot - other slot is not used
- One slot is updated - other slot has a working system
- In case of errors -> rollback to the working system
- No partition in the current slot should be updated

A/B System Updates - Attributes

- Bootable attribute = includes a functional system that can boot
- Current slot is bootable, the other slot may be:
  - Old, functional version
  - New version
  - Invalid data
- Only one active/preferred slot - used on the next boot

A/B System Updates - Attributes

- Successful attribute
  - Set in userspace
  - Slot with the attribute bootable
  - Slot able to boot, run, update
- Bootable slot not marked successful (after several attempts)
  - Becomes unbootable
  - Change active slot to another bootable slot

Outline

Bibliography

- Android Security Internals, Nicolay Elenkov, 2015
- Android Hacker's Handbook, Joshua J. Drake, 2014
- `https://source.android.com/devices/tech/ota/`

Keywords

- Bootloader
- OEM Unlock
- Fastboot
- System partition
- Boot partition
- Recovery partition
- Stock Recovery
- Custom Recovery
- TWRP
- OTA Update
- Block OTA Update
- A/B Update