Device Provisioning and OTA updates

Mark.gross@intel.com

ABS2011

Pre-ramble

- Pre-OS and provisioning related requirements.
- What's available from AOSP
- What we've been doing
  - I will not say much about the OTA agent you'll need to get that from your carrier.

- I am not an expert. But ask questions anyway.
  - This slide deck is not really a good reference
  - Lots of detail **not** covered here

Pre OS and Provisioning requirements

Device Update

- Storage formating / partitioning
- FW
- Kernel + initrd
- System partition
- Modem
- Security stuff
- update of recovery
- Boot loader

Manufacturing

- Loading Manufacturing and test payload
- per device data to store
  - 3g calibration
  - Gps calibration
  - MAC ID's
  - Device unique serial numbers and keys
- Speed
- Ease of use for line workers

Development

- OS updates
  - Kernel / Drives
  - System
- Fast turn around time
- Hands free automation

Reliability

- Error logging and reporting
  - Not really so great today.
- Robustness against brickage
- Payload / Provisioning hand shake
- Battery + charger awareness

Other stuff

- Packaging
- Signing of update packages
- Partial updates
- Versioning
- Carrier dependencies
- UI
- Roll back
- Security
- Lots of stuff I'm not dealing with yet.

Device provisioning and update support provided to some extent by AOSP

Fastboot client

- Built as part of the host tools.
- Walks usb stack looking for fastboot gadgets.
- System/core/fastboot

Fastboot target

- bootable/bootloader/lagacy/*
  - fastboot_protocal.txt
- git://android.git.kernel.org/kernel/lk.git
  - App/aboot
-

Fastboot gaps

- “popping” error logging from target
- Add hock commands
  - Security risk
- Download speed

Provisioning / Recovery

- Bootable/recovery.git
  - Tools/ota
  - Recovery.c
- Recovery
  - Platform/recovery.git
- OTAProvisioning.git ← empty
- MasterClearReciver.java
- Recovery.java

Handshake between OS's

- Documented in comments in Recovery.c
  - Commands
  - Logs
  - Intents
- Reboot reasons

Provisioning OS
(aka Intel's recovery image)

Provisioning OS

- Started off as a adb_gadget kernel driver hack
- Used aboot from lk.git
- Used to be just kboot with a fastboot gadget + aboot daemon
- Evolving into a “recovery OS” based on kboot (without kexec) for x86.
- Deals with platform specific details.
-

Provisioning OS

- Started off using tarball packaging.
- Moving to loop back iso's for packaging
- Signed packages
- Standard update processing for each type of update
  - FW, kernel/initrd, system, Modem
- Device specific partitioning
- Similar to “dual booting”

Hardening

- Device Provisioning is a common attack vector for rooting devices.
- Remove shell
- Limit operations to only trusted ones.
-

MRST/MFLD OSIP

- similar to menu.lst
- Array of OSII values each OSII can be a loadable boot image
- Manipulate OSII array to control what image boots.

UMG specific detail

- FW boots multiple targets based on search path of a data structure called the OSIP.
- Default osii to boot is the Main OS (android or manufacturing)
- Direct OSIP manipulation enables boot into provisioning OS
- Provisioning OS can re-write the main OS (kernel + initial ram disk)
- Re-writing OSIP defaults boot to Main OS after provisioning is finished with its processing.

Questions?