UEFI/EDK2 for RDK on Hikey

Kalyan Kumar N (LHG)

Agenda

- UEFI Bootloader For RDK
- Secure Boot Loader
- Development of RDK Boot Loader

UEFI Bootloader For RDK

- Standardization of the RDK set-top box firmware boot process
  - Increase industry awareness of UEFI/EDK2 solutions for set-top boot implementation
- Need secure boot with hardware root of trust with secure keys
- Implement RDK Bootloader and Disaster Recovery Image (DRI) requirements (use cases) using well defined standard.

UEFI/EDK2 Dev Environment

- QEMU

  https://wiki.linaro.org/LEG/UEFIforQEMU

- HiKey

  https://github.com/96boards/documentation/wiki/HiKeyUEFI

Secure Boot Loader

- Helps Prevents malicious code before OS Loads
- Validates UEFI applications (boot loaders and drivers) using AuthentiCode signatures embedded in these applications
- Trusted X.509 root certificates are stored in UEFI variables
- Enable / Disable Secure Boot

Secure Boot Keys:

- Platform Key (PK) - Trust relationship between platform owner & firmware
- Key Exchange Key (KEK) - Trust relationship between OS & firmware
- Signing database (DB) - whitelist authorised certificates

Secure Boot Loader

Basic steps for Implementing Secure Boot:

- Set platform key(PK) using setVariable() API
- Validated the System boot mode using Setup Mode
- Add KEK and DB Keys using setVariable() for validating Signed Images.

RDK Boot Loader

- Create new module (.inf) for RDK Boot Loader in EDK2 code

- Use EFI Runtime service Set/Get Variable() for setting/getting other Module EFI variable.

Secure Boot enable programmatically:

- Set EFI_CUSTOM_MODE_NAME to CUSTOM_SECURE_BOOT_MODE

- Use EFI_SIMPLE_FILE_SYSTEM_PROTOCOL for opening PK key and get File handle.

- Populate EFI_SIGNATURE_LIST data for PK key by reading File content

RDK Boot Loader

- Set PK_KEY with populated EFI_SIGNATURE_LIST data (PK cert).

- Attributes for setting Keys = EFI_VARIABLE_NON_VOLATILE | EFI_VARIABLE_TIME_BASED_AUTHENTICATED_WRITE_ACCESS | EFI_VARIABLE_BOOTSERVICE_ACCESS

- Same procedure for KEK and DB cert registration.

RDK Boot Loader

RDK kernel boot:

- Use”Loaded Image protocol” for loading kernel to physical memory
- Load options for kernel arguments
  - char load[]= "initrd=/initramfs";
  - CHAR16 LoadOption[30];
  - UnicodeSPrintAsciiFormat(LoadOption,sizeof(LoadOption),load);
  - ImageInfo->LoadOptions = LoadOption;

- Linux kernel(>= 4.5) treated as UEFI Application and can be launched using Start Image.

Signing Images

create key pair

- openssl req -new -x509 -newkey rsa:2048 -subj "/CN=my PK /" -keyout PK.key -out PK.crt -days 3650 -nodes -sha256

- **sbsign --key DB.key --cert DB.crt --output Image-Signed Image**

- **sbsign --key DB.key --cert DB.crt --output RDKImageLoader-signed.efi RDKImageLoader.efi**

Work in progress

- Signing monolithic Image of Kernel and Rootfs and validating
  Through UEFI bootloader

- DRI (disaster recovery Image) implementation using UEFI.

Thank You

#BUD17

For further information: www.linaro.org

BUD17 keynotes and videos on: connect.linaro.org