System Firmware and Device Firmware Updates using Unified Extensible Firmware Interface (UEFI) Capsules

HARRY HSIUNG, INTEL CORPORATION

LINUX PLUMBERS CONFERENCE - AUGUST 2020
Building and Distributing UEFI Capsules for Firmware Update

1. Generate UEFI Capsule
2. Publish UEFI Capsule
3. Distribute UEFI Capsule
4. Process UEFI Capsule

Linux* Vendor Firmware Service (LVFS)
https://fwupd.org

* Other names and brands may be claimed as property of others
Platform Initialization (PI) Architecture
Firmware Phases

Pre EFI Initialization (PEI)
- Device, Bus, or Service Driver
- EFI Driver Dispatcher
- Boot Manager
- Intrinsic Services

Driver Execution Environment (DXE)
- Boot Dev Select (BDS)
- Transient System Load (TSL)

OS-Absent App
- Transient OS Environment
- Transient OS Boot Loader

OS-Present App
- Final OS Boot Loader
- Final OS Environment

Security (SEC)
- Pre EFI Initialization (PEI)
- Driver Execution Environment (DXE)
- Boot Dev Select (BDS)
- Transient System Load (TSL)
- Run Time (RT)
- After Life (AL)

Power on → [ . . . Platform initialization . . . ] → [ . . . . OS boot . . . . ] → Shutdown
PI Architecture Firmware Phases
Example UEFI Capsule Processing

Normal Boot

Pre EFI Initialization (PEI)

Driver Execution Environment (DXE)

Boot Device Select (BDS)

Final OS Boot Loader

Load UEFI Capsule

Call UpdateCapsule()

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Normal Boot

ResetSystem()

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()

Normal Boot

Normal Boot

Receive UEFI Capsules (LVFS) based on matching ESRT

Save UEFI Capsule

Load UEFI Capsule

Call UpdateCapsule()

Final OS Boot Loader

Process UEFI Capsules

Pass UEFI Capsule to matching Firmware Management Protocol

Coalesce UEFI Capsules

ResetSystem()
Process UEFI Capsule

**UEFI Capsule**

- UEFI Capsule Header
- FMP Header
- Auth Info
- Payload Header (Extensible)
- Payload

**System Firmware**

- SetImage()
- Authenticate

**FMP Driver**

- ImageTypeId
- GUID A
- Public Key(s)

**ESRT Table**

- GUID A

**Flash**

- Publish
- Update

FMP = UEFI Firmware Management Protocol
GUID = Globally Unique Identifier
## EDK II UEFI Capsule Features

EFI Development Kit II ([https://www.tianocore.org](https://www.tianocore.org))

<table>
<thead>
<tr>
<th>Feature</th>
<th>UDK2017 / UDK2018</th>
<th>edk2-stable201808</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate UEFI Capsule</td>
<td>Integrated EDK II Build</td>
<td>Standalone Python* Script</td>
</tr>
<tr>
<td>Update Granularity</td>
<td>Focused on Monolithic</td>
<td>Designed to support Multiple Components</td>
</tr>
<tr>
<td>Authentication</td>
<td>PKCS7 Single Key</td>
<td>PKCS7 Multiple Keys</td>
</tr>
<tr>
<td>Pre-Check</td>
<td>N/A</td>
<td>Power/Battery, Thermal, System</td>
</tr>
<tr>
<td>Update Indicator</td>
<td>Requires platform code</td>
<td>Built-in with Consistent UX and Progress Bar</td>
</tr>
<tr>
<td>Firmware Management Protocol</td>
<td>Requires full implementation</td>
<td>Produced by FmpDxe module customized using configuration data and small libraries.</td>
</tr>
<tr>
<td>Test Key Detection</td>
<td>Requires platform code</td>
<td>Built-in</td>
</tr>
<tr>
<td>Watchdog</td>
<td>Requires platform code</td>
<td>Built-in</td>
</tr>
<tr>
<td>ESRT Driver</td>
<td>Legacy + FMP</td>
<td>Smaller/Simpler FMP only version</td>
</tr>
</tbody>
</table>

* Other names and brands may be claimed as property of others.
Firmware Update Indicators

UEFI Graphics Console
EFI_GRAPHICS_OUTPUT_PROTOCOL

- System Logo
- User Experience(UX) Capsule Bitmap Message

UEFI Text Console
EFI_SIMPLE_TEXT_OUTPUT_PROTOCOL

- Update Progress - 100%
- Update Progress - 100%
- Update Progress - 100%
- Update Progress - 32%

Customize with a new DisplayUpdateProgressLib instance
FmpDxe Module Overview

FMP DXE Module
Configured through PCDs
Produces UEFI Firmware Management Protocol

- FmpAuthenticationLib
  - BaseCryptLib
    - OpensslLib
  - FmpPayloadHeaderLib
- FmpDeviceLib
- CapsuleUpdatePolicyLib

Generic
Device Vendor
Platform Vendor

PCD = Platform Configuration Database
## FmpDxe Module Configuration

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE_GUID</td>
<td>ESRT GUID Value</td>
</tr>
<tr>
<td>PcdFmpDeviceImageIdName</td>
<td>FMP Image Descriptor - Unicode string</td>
</tr>
<tr>
<td>PcdFmpDeviceBuildTimeLowestSupportedVersion</td>
<td>Build time FMP/ESRT default value</td>
</tr>
<tr>
<td>PcdFmpDeviceLockEventGuid</td>
<td>Event GUID to lock FW storage device. Default is End of DXE.</td>
</tr>
<tr>
<td>PcdFmpDeviceProgressWatchdogTimeInSeconds</td>
<td>Watchdog armed on each progress update</td>
</tr>
<tr>
<td>PcdFmpDeviceProgressColor</td>
<td>24-bit Progress Bar Color (0x00rrggbb)</td>
</tr>
<tr>
<td>PcdFmpDevicePkcs7CertBufferXdr</td>
<td>One or more PKCS7 Certs in XDR format. Encode with <code>BaseTools/Scripts/BinToPcd</code></td>
</tr>
<tr>
<td>PcdFmpDeviceTestKeySha256Digest</td>
<td>Set to <code>{0}</code> to disable test key detection</td>
</tr>
</tbody>
</table>

XDR = External Data Representation using Variable-Length Opaque Data format from RFC 4506
## CapsuleUpdatePolicyLib APIs
### Platform Specific Library

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckSystemPower()</td>
<td>Is system power/battery ok for FW update?</td>
</tr>
<tr>
<td>CheckSystemThermal()</td>
<td>Is system temperature ok for FW update?</td>
</tr>
<tr>
<td>CheckSystemEnvironment()</td>
<td>Is the system environment ok for FW update?</td>
</tr>
<tr>
<td>IsLowestSupportedVersionCheckRequired()</td>
<td>Skip lowest supported version check? (e.g. Service Mode)</td>
</tr>
<tr>
<td>IsLockFmpDeviceAtLockEventGuidRequired()</td>
<td>Skip firmware storage device lock action? (e.g. Manufacturing Mode)</td>
</tr>
</tbody>
</table>
# FmpDeviceLib APIs

## Device Specific Library

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RegisterFmpInstaller()</td>
<td>Future expansion for add-in controllers.</td>
</tr>
<tr>
<td>FmpDeviceGetSize()</td>
<td>Size of <strong>currently stored FW image</strong>.</td>
</tr>
<tr>
<td>FmpDeviceGetImageTypeIdGuidPtr()</td>
<td>ESRT/FMP GUID. Overrides FILE_GUID value.</td>
</tr>
<tr>
<td>FmpDeviceGetAttributes()</td>
<td>FMP Attributes Supported/Settings.</td>
</tr>
<tr>
<td>FmpDeviceGetLowestSupportedVersion()</td>
<td>LSV from <strong>currently stored FW image</strong>.</td>
</tr>
<tr>
<td>FmpDeviceGetVersionString()</td>
<td>Unicode version string from <strong>currently stored FW image</strong>.</td>
</tr>
<tr>
<td>FmpDeviceGetVersion()</td>
<td>32-bit version value from <strong>currently stored FW image</strong>.</td>
</tr>
<tr>
<td>FmpDeviceGetImage()</td>
<td>Retrieve copy of <strong>currently stored FW image</strong>.</td>
</tr>
<tr>
<td>FmpDeviceCheckImage()</td>
<td>Check if a new FW image is valid for this device.</td>
</tr>
<tr>
<td>FmpDeviceSetImage()</td>
<td>Update FW storage with a new FW image.</td>
</tr>
<tr>
<td>FmpDeviceLock()</td>
<td>Lock FW storage to prevent any further changes.</td>
</tr>
</tbody>
</table>
ESRT GUIDs and Keys
Multiple Components

System Firmware

FMP Driver
ImageTypeId GUID A

FMP Driver
ImageTypeId GUID B

FMP Driver
ImageTypeId GUID C
  Public Key(s)

FMP Driver
ImageTypeId GUID D
  Public Key(s)

Vital Product Data
Public Key(s)

ACPI ESRT
GUID A
GUID B
GUID C
GUID D
ESRT GUIDs and Keys

3rd Party FMP Driver

System Firmware

FMP Driver
ImageTypeId
GUID A
3rd Party Key(s)

Import Driver
Replace with System Key(s)

Vital Product Data (VPD)
Public Key(s)

FMP Driver
ImageTypeId
GUID A
System Key(s)

FMP Driver
ImageTypeId
GUID B

ACPI ESRT
GUID A
GUID B

3rd Party UEFI Capsules must be re-signed with System Key
ESRT GUIDs and Keys

3rd Party FMP Driver

System allows UEFI Capsules from 3rd Party to be installed
Add FMP to Existing Device Driver

System Firmware

Device Driver
- FMP Library
  - ImageTypeId: GUID A
  - Public Key(s)

FMP Driver
- ImageTypeId: GUID B
  - Public Key(s)

FMP Driver
- ImageTypeId: GUID C
  - Public Key(s)

ACPI ESRT
- GUID A
- GUID B
- GUID C
Summary

EDK II supports new UEFI Capsule Features for Firmware Update
  ◦ Simplifies FMP support for system firmware and integrated devices
  ◦ Multiple authentication keys with flexible key storage options.
  ◦ System update pre-check (Power/battery, thermal, and system).
  ◦ Improved UX with progress indicators during update.
  ◦ Built-in support for test key detection & watchdog timer.
  ◦ Simplified ESRT driver using FMP instances

EDK II GenerateCapsule.py used to Generate UEFI Capsules

Publish and Distribute UEFI Capsules for Firmware Updates using Linux Vendor Firmware Services (LVFS)
Call to Action

Add UEFI Capsule based Firmware Update to platforms
Implement UEFI Capsule based Firmware Update for devices
Take advantage of latest EDK II FmpDevicePkg features
Use Linux Vendor Firmware Service (LVFS) to publish and distribute UEFI Capsule based Firmware Updates

Provide feedback and contribute!

- TianoCore  https://www.tianocore.org/
- LVFS  https://fwupd.org/