Open. Together.
Go Forth and Modify: Fiano

Gan Shun Lim  Software Engineer, Google
Ryan O’Leary  Software Engineer, Google
Many thanks to

- Ron Minnich (Google)
- Julien Viard de Galbert (Splitted-Desktop Systems)
- Andrea Barberio (Facebook)

References taken from

- Nikolaj Schlej (UEFITool)
- Teddy Reed (UEFI Firmware Parser)
The problem

- Vendors provide binary UEFI blobs without source
- Want to edit binary UEFI firmware images
- UEFI was designed to be modular, should be easy in theory
- Applications:
  - LinuxBoot (see Chris Koch’s talk)
  - Removing unnecessary DXEs to reduce attack surface
  - Security forensics
  - Debugging
  - Rapid prototyping
LinuxBoot

- LinuxBoot adds Linux to your UEFI firmware image.
- Netboot and diskboot are performed by Linux.
- See Chris Koch’s talk for specifics

https://www.linuxboot.org
UTK as a Build Tool for LinuxBoot

Existing UEFI Firmware Image

UTK

LinuxBoot Firmware Image

utk old.bios replace_pe32 Shell linux.efi save linuxboot.bios
DXECleaner

Original UEFI Image

- DXE Firmware Volume
  - DXE
  - DXE
  - DXE
  - DXE
- PEI Firmware Volume

LinuxBoot Image

- DXE Firmware Volume
  - DXE
  - DXE
  - DXE
  - DXE
- PEI Firmware Volume
  - Linux

Cleaned Image

- DXE Firmware Volume
  - DXE
  - DXE
- PEI Firmware Volume
  - Linux

Insert Linux kernel
Removal of Unneeded DXEs
Load bearing

Test
Remove DXE

Original UEFI Image → LinuxBoot Image → Cleaned Image

Open. Together.
Demo of DXECleaner

https://asciinema.org/a/OjPGXgylNGreaAbsaJM4bVTsj
Anatomy of a UEFI Image

- Firmware Image
  - Intel Firmware Descriptor
  - ME Region
  - BIOS Region
  - GBE Region
  - UEFI Firmware Volume
  - UEFI File
  - UEFI Section
  - INTEL Specific

- UEFI File
  - UEFI File
  - UEFI Section

- OPEN SYSTEMS FIRMWARE

- Embedded Software

Open. Together.
Anatomy of a UEFI Image

- Firmware Image
  - Intel Firmware Descriptor
  - ME Region
    - UEFI Firmware Volume
    - BIOS Region
      - UEFI File
      - UEFI Section
    - GBE Region
      - UEFI File
      - ...
What can utk do?

- Poke around
- Extract files
- Modify files in memory and save.
<table>
<thead>
<tr>
<th>Node</th>
<th>GUID/Name</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOS</td>
<td>FFF12B8D-7696-4C8B-A985-2747075B4F50</td>
<td>EFI FV FILETYPE_FIRMWARE_VOLUME_IMAGE</td>
<td>0x1256a7</td>
</tr>
<tr>
<td>FV</td>
<td>8C8CE578-8A3D-4F1C-9935-896185C32DD3</td>
<td>EFI SECTION_GUID_DEFINED</td>
<td>0x12568f</td>
</tr>
<tr>
<td>FV</td>
<td>E77F1DB2D792</td>
<td>EFI SECTION_RAW</td>
<td>0x7c</td>
</tr>
<tr>
<td>File</td>
<td>9E21FD93-9C72-4C15-8C4B-E77F1DB2D792</td>
<td>EFI SECTION_FIRMWARE_VOLUME_IMAGE</td>
<td>0xe0004</td>
</tr>
<tr>
<td>File</td>
<td>8C8CE578-8A3D-4F1C-9935-896185C32DD3</td>
<td>EFI FV FILETYPE_FREEFORM</td>
<td>0x2c</td>
</tr>
<tr>
<td>File</td>
<td>1B45CC0A-156A-428A-AF62-49864DA0E6E6</td>
<td>EFI SECTION_RAW</td>
<td>0x14</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI FV FILETYPE_PAD</td>
<td>0xc4fa</td>
</tr>
<tr>
<td>File</td>
<td>52C05B14-0B98-496C-BC3B-04B50211D680</td>
<td>EFI FV FILETYPE_PEI_CORE</td>
<td>0x3c</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI SECTION_RAW</td>
<td>0xc484</td>
</tr>
<tr>
<td>File</td>
<td>8C8CE578-8A3D-4F1C-9935-896185C32DD3</td>
<td>EFI SECTION_PE32</td>
<td>0xc484</td>
</tr>
<tr>
<td>File</td>
<td>1B45CC0A-156A-428A-AF62-49864DA0E6E6</td>
<td>EFI SECTION_USER_INTERFACE</td>
<td>0x14</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI SECTION_VERSION</td>
<td>0xe32f3a</td>
</tr>
<tr>
<td>File</td>
<td>9B3ADA4F-AE56-4C24-8DEA-F03B7558AE50</td>
<td>EFI FV FILETYPE_PEIM</td>
<td>0x40</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI FV FILETYPE_PAD</td>
<td>0x40</td>
</tr>
<tr>
<td>File</td>
<td>A3610442-E69F-4DF3-82CA-2360C4031A23</td>
<td>EFI FV FILETYPE_PEIM</td>
<td>0x211e</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI FV FILETYPE_PAD</td>
<td>0x60</td>
</tr>
<tr>
<td>File</td>
<td>9D225237-FA01-464C-A949-BABCB02D31D0</td>
<td>EFI FV FILETYPE_PEIM</td>
<td>0x21d6</td>
</tr>
<tr>
<td>File</td>
<td>E77F1DB2D792</td>
<td>EFI FV FILETYPE_PAD</td>
<td>0x28</td>
</tr>
</tbody>
</table>
More commands for poking around

- **utk <romimage> find .*Shell.***
  - Takes a regexp, dumps json about the struct in question
- **utk <romimage> find 7C04A583-9E3E-4f1c-AD65-E05268D0B4D1**
  - Find can also take a GUID, (in this case it’s the EFI Shell GUID)
- **utk <romimage> dump .*Shell.* shell.bin**
  - dump uses find to search for the requested ffs, and dumps the whole ffs to a binary
- **utk <romimage> count**
  - Counts the number of each type of firmware.
Let’s change things!

- `utk <romimage> remove .*Shell.* save <newromimage>`
  - Removes an FFS, could be Dxe, or Pei
  - Remove takes the same arguments as Find.

- `utk <romimage> replace_pe32 .*Shell.* bzImage save <newromimage>`
  - Replaces the PE32 executable in the Shell with another PE32 executable. In the case of Linuxboot, it can just be a Linux kernel!
Chain commands together

- utk <romimage> \\n  ○ remove .*Ip.* \\n  ○ remove .*Dhcp.* \\n  ○ replace_pe32 .*Shell.* bzImage \\n  ○ save <newromimage>

- Commands can be chained together for more complex operations
TLDR

- Easily scriptable UEFI image editing tool
- Written in Go, unit-tested, type safe
- Avoids rebuilding entire UEFI images.
  - Speed
  - Availability of source
- Automated DXE removal
Call to Action

UTK
https://github.com/linuxboot/fiano

Try utk
sudo apt-get install go
go get github.com/linuxboot/fiano/cmds/utk
utk --help

Take a look at the issue tracker. help wanted tags are great to start. File bugs, create pull-requests, update documentation, ...

LinuxBoot Book
https://github.com/linuxboot/book
See the UTK chapter.

New Firmware
Let us know if you want to try UTK on your firmware. We’re always excited about seeing UTK work for new firmware!

Laptop Stickers
Get them now!
Open. Together.
OCP Global Summit | March 14–15, 2019