KrakenBoot: Firmware-Level Cluster Provisioning via UEFI Surgery

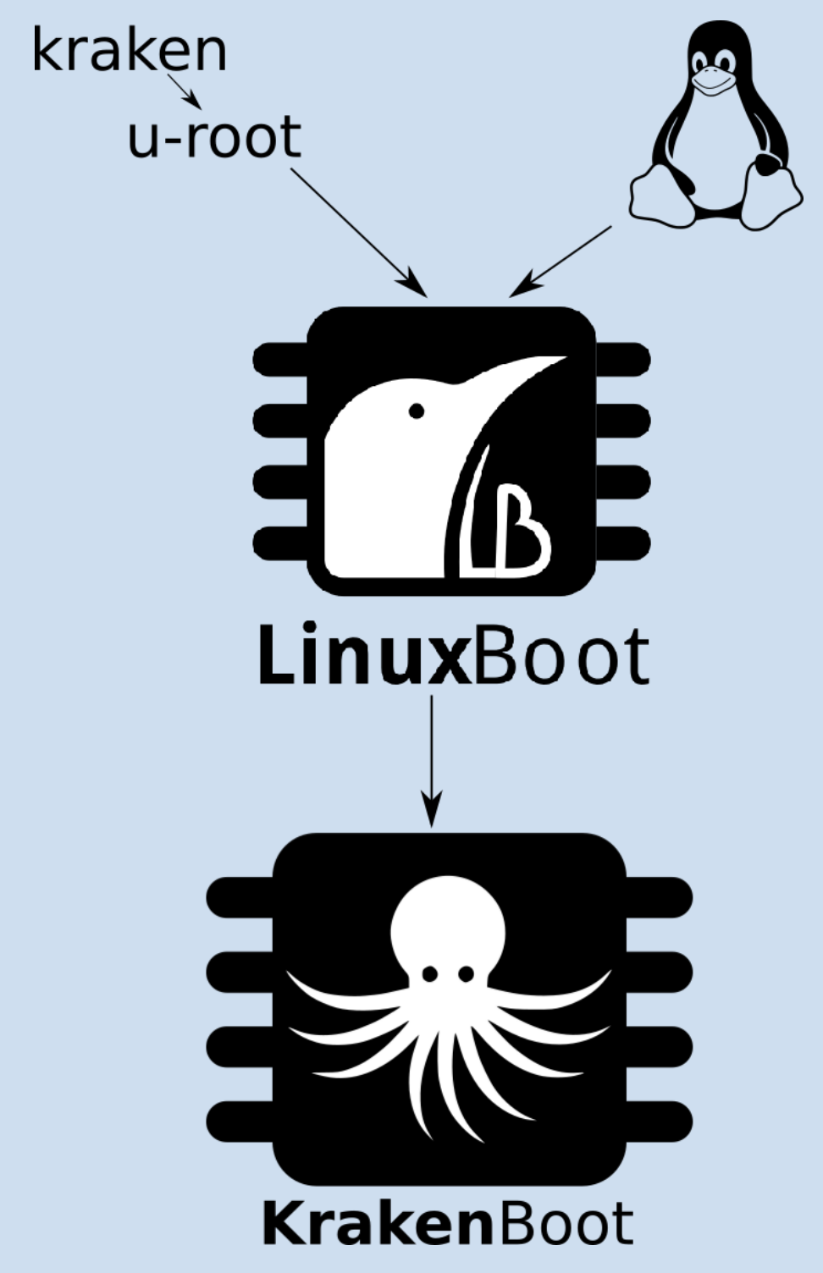
Devon Bautista – *Arizona State University*
LA-UR-19-27138

Mentors: Lowell Wofford, Cory Lueninghoener

1. Abstract

Traditionally, cluster booting and provisioning has relied on a multitude of dated software such as PXE boot, `tftp`, and DHCP, which have been proven to be fragile when combined. Furthermore, firmware shipped with motherboards is normally closed-source and often comes with unnecessary bloat which can increase boot times. Thus, a more robust and integrated solution is needed which solves both of these issues. **Kraken** is a distributed automation tool with applications in cluster booting and provisioning to reliably enforce node state. It essentially unifies the fragmented utilities described above for a more robust cluster provisioning solution. The **Unified Extensible Firmware Interface (UEFI)** is the *de facto* open firmware specification, but its implementation is largely proprietary and closed-source. Being able to control the UEFI boot process and subsequently fine-tune it for cluster nodes is necessary for tuned, performance systems like clusters. Therefore, we propose **KrakenBoot** – a project which surgically removes unnecessary (and possibly erratic) UEFI components and replaces them with Kraken/Linux as a pre-boot environment.

2. Structure



- **LinuxBoot[1]** is used to generate firmware for certain mainboards. In this research the *Comanche C99X* ARM-based experimental board was used for testing.
- **Kraken[2]** itself is embedded in the initramfs, which is generated with **U-root[3]**.
- **The Linux Kernel[4]** is combined with the U-root initramfs and passed to LinuxBoot to generate the KrakenBoot firmware image.
- **KrakenBoot** integrates Kraken into the initramfs and handles the Kraken environment creation.

4. Results

- A KrakenBoot firmware image was successfully built for the Comanche C99X board.

Firmware Comparison		
	Stock	KrakenBoot
Boot Time[a]	3:45	N/A
Total DXE Size	7MiB	9MiB

- A larger DXE volume means less space wasted by padding and more space utilized by manually-chosen drivers, etc.
- Boot time was inconclusive due to time constraints for testing KrakenBoot on actual hardware.

[a] Boot times measured from power on until the user interaction (i.e. `login` or an interactive shell) starts.

5. References

[1] Ronald Minnich, et al. *LinuxBoot*. URL: `https://www.linuxboot.org/`.

[2] *Kraken*. URL: `https://github.com/hpc/kraken`.

[3] Ronald Minnich, et al. *u-root*. URL: `http://u-root.tk/`.

[4] Linus Torvalds, et al. *Linux*. Version 4.14.62. URL: `https://www.kernel.org/`.

3. Replacing UEFI Drivers with Linux and Kraken Initramfs

Figure 1 shows the 7-stage boot sequence of UEFI. The red markings indicate which components were removed while the green markings show which components were added.

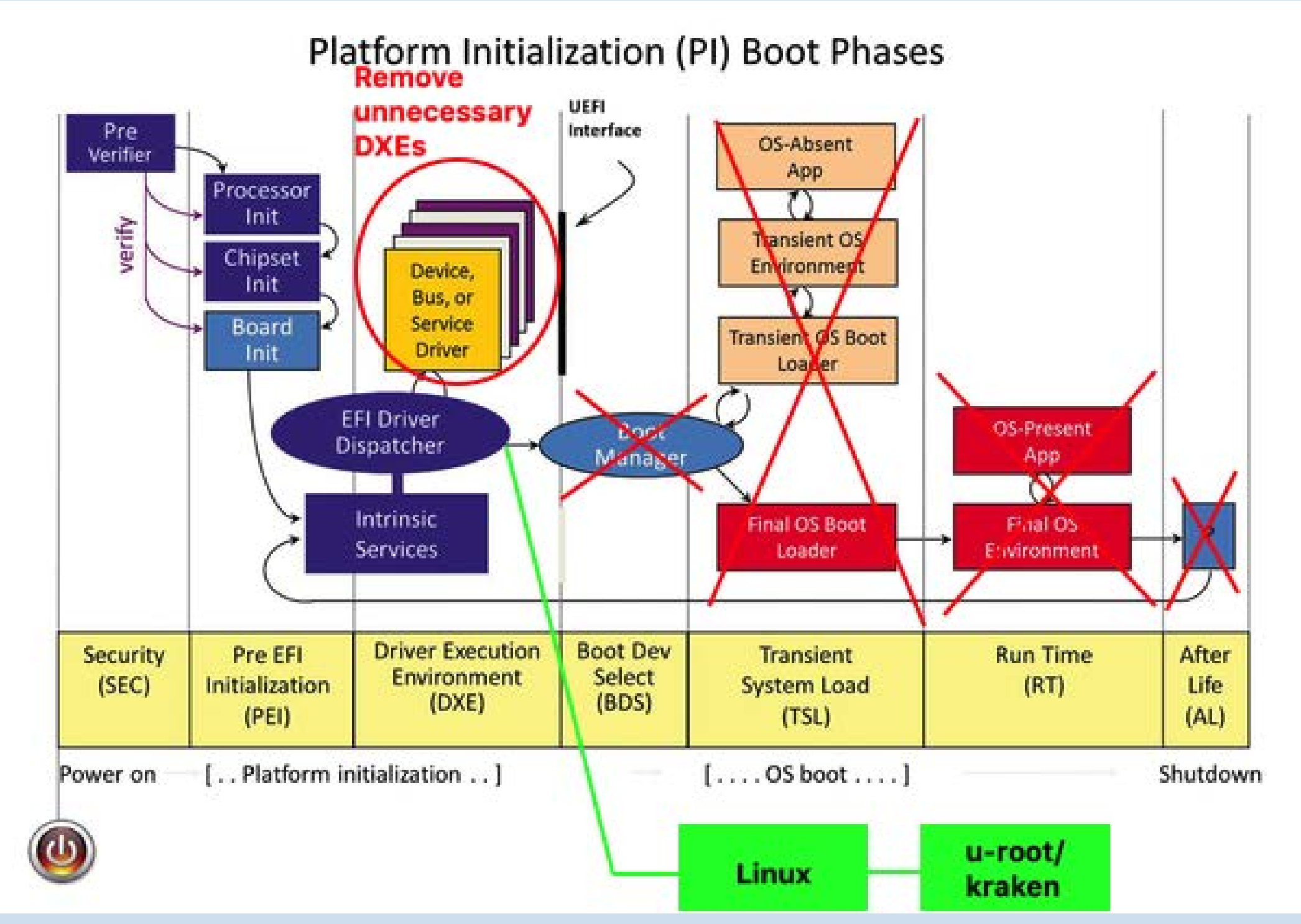
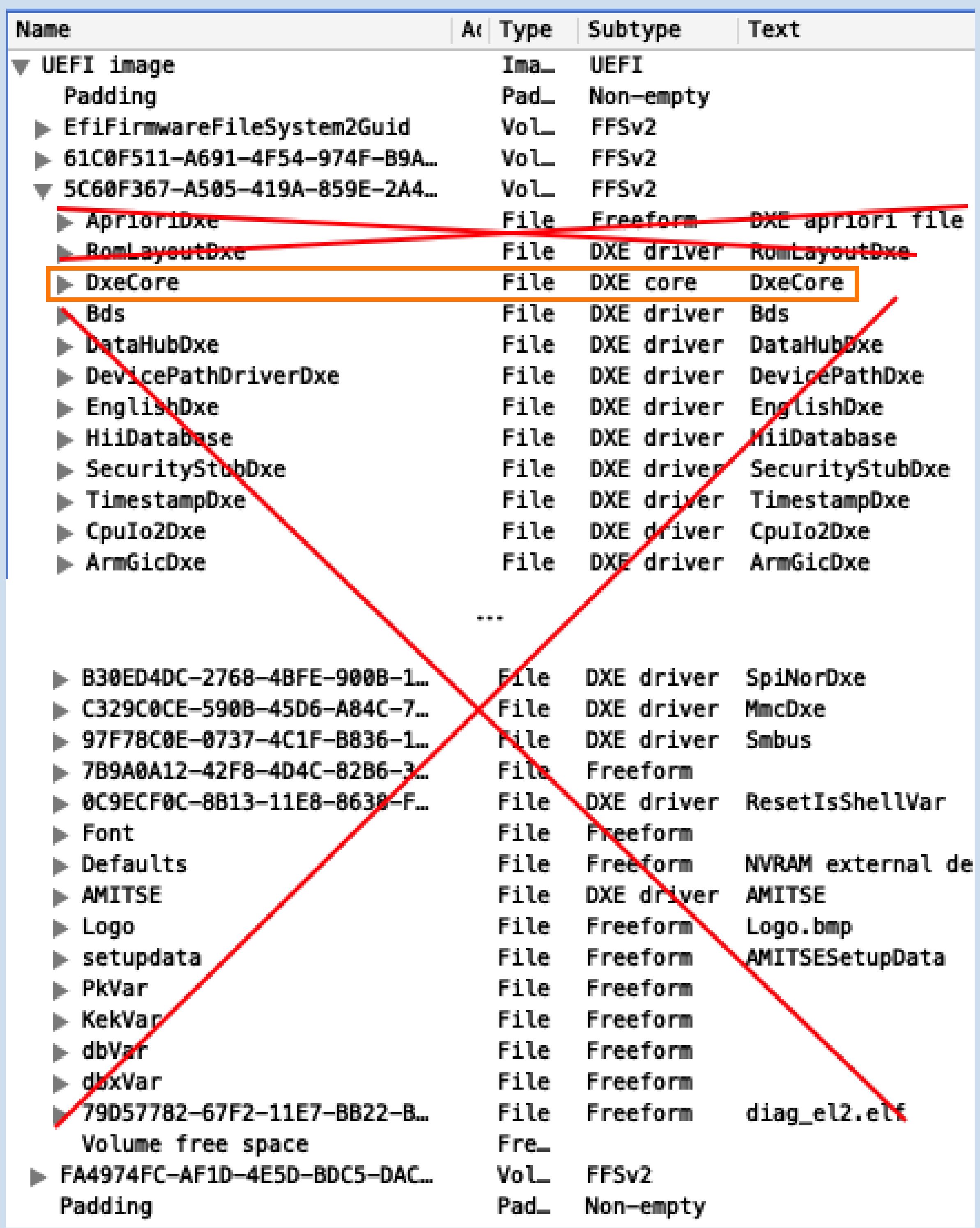


Figure 1: Modifying the UEFI boot process

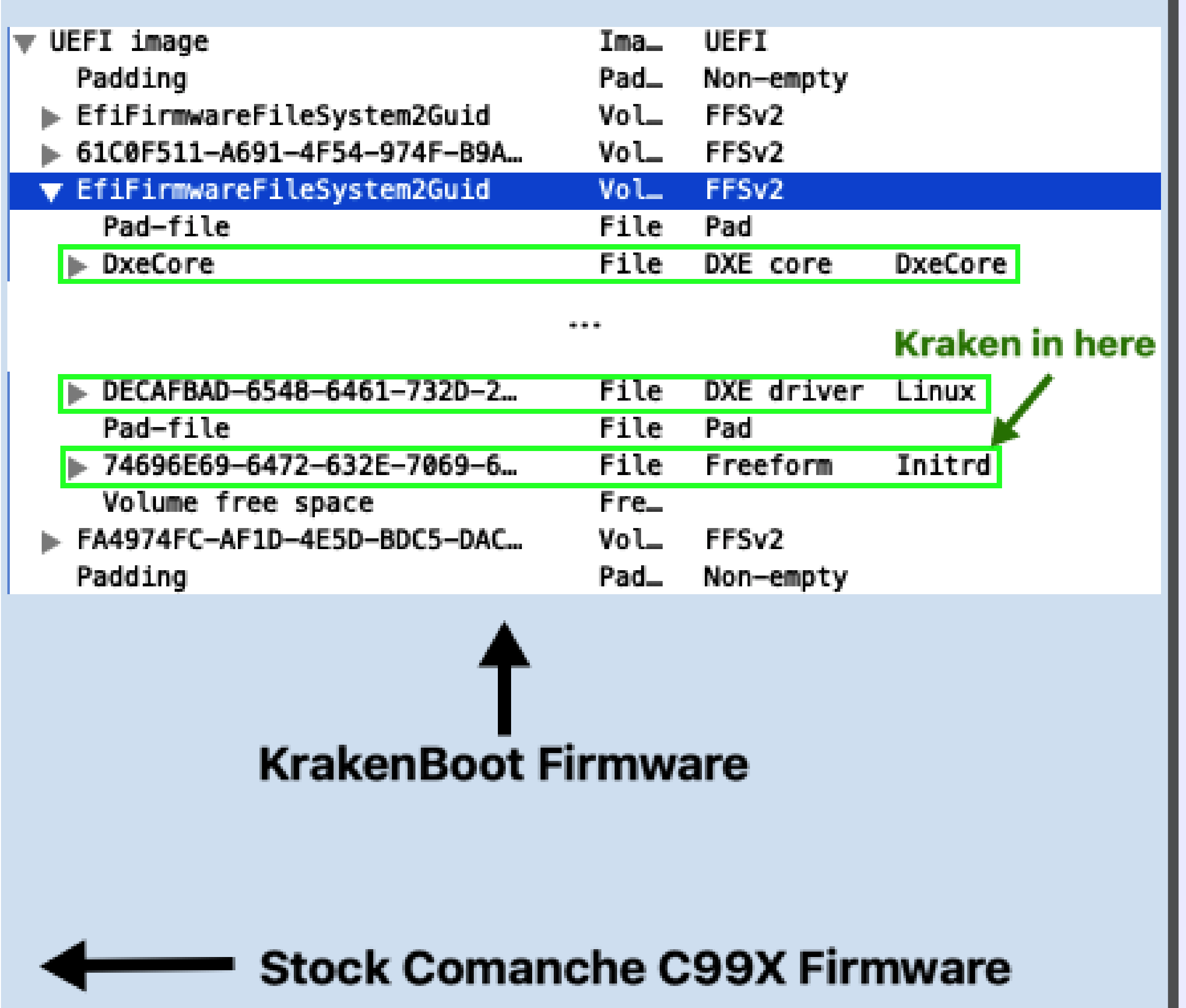
- Most UEFI drivers in the DXE Phase (Phase 3) are scrapped in favor of drivers in the Linux kernel.
- Instead of booting using unknown vendor driver binaries, a known kernel-initramfs pair is launched. **The boot process from here is known!**
- By eliminating unnecessary DXE drivers and compiling a custom Linux kernel, the firmware:
  - has a known boot process from the DXE phase.
  - is slimmed down only to what it needs.
- Some closed-source parts of the UEFI implementation (Phases 1-2) are necessary; we can't get rid of them.



```
Name                                 A( Type  Subtype      Text
▼ UEFI image                            Ima…  UEFI
    Padding                             Pad…  Non-empty
  ▶ EfiFirmwareFileSystem2Guid          Vol…  FFSv2
  ▶ 61C0F511-A691-4F54-974F-B9A…        Vol…  FFSv2
  ▼ 5C60F367-A505-419A-859E-2A4…        Vol…  FFSv2
    ▶ AprioriDxe                        File  Freeform     DXE apriori file
    ▶ RomLayoutDxe                      File  DXE driver   RomLayoutDxe
    ▶ DxeCore                           File  DXE core     DxeCore
    ▶ Bds                               File  DXE driver   Bds
    ▶ DataHubDxe                        File  DXE driver   DataHubDxe
    ▶ DevicePathDriverDxe               File  DXE driver   DevicePathDxe
    ▶ EnglishDxe                        File  DXE driver   EnglishDxe
    ▶ HiiDatabase                       File  DXE driver   HiiDatabase
    ▶ SecurityStubDxe                   File  DXE driver   SecurityStubDxe
    ▶ TimestampDxe                      File  DXE driver   TimestampDxe
    ▶ CpuIo2Dxe                         File  DXE driver   CpuIo2Dxe
    ▶ ArmGicDxe                         File  DXE driver   ArmGicDxe

                                 …

    ▶ B30ED4DC-2768-4BFE-900B-1…        File  DXE driver   SpiNorDxe
    ▶ C329C0CE-590B-45D6-A84C-7…        File  DXE driver   MmcDxe
    ▶ 97F78C0E-0737-4C1F-B836-1…        File  DXE driver   Smbus
    ▶ 7B9A0A12-42F8-4D4C-82B6-3…        File  Freeform
    ▶ 0C9ECF0C-8B13-11E8-8638-F…        File  DXE driver   ResetIsShellVar
    ▶ Font                              File  Freeform
    ▶ Defaults                          File  Freeform     NVRAM external de
    ▶ AMITSE                            File  DXE driver   AMITSE
    ▶ Logo                              File  Freeform     Logo.bmp
    ▶ setupdata                         File  Freeform     AMITSESetupData
    ▶ PkVar                             File  Freeform
    ▶ KekVar                            File  Freeform
    ▶ dbVar                             File  Freeform
    ▶ dbxVar                            File  Freeform
    ▶ 79D57782-67F2-11E7-BB22-B…        File  Freeform     diag_el2.elf
      Volume free space                 Fre…
  ▶ FA4974FC-AF1D-4E5D-BDC5-DAC…        Vol…  FFSv2
    Padding                             Pad…  Non-empty
```



```
▼ UEFI image                            Ima…  UEFI
    Padding                             Pad…  Non-empty
  ▶ EfiFirmwareFileSystem2Guid          Vol…  FFSv2
  ▶ 61C0F511-A691-4F54-974F-B9A…        Vol…  FFSv2
  ▼ EfiFirmwareFileSystem2Guid          Vol…  FFSv2
      Pad-file                          File  Pad
    ▶ DxeCore                           File  DXE core     DxeCore

                                 …

    ▶ DECAFBAD-6548-6461-732D-2…        File  DXE driver   Linux
      Pad-file                          File  Pad
    ▶ 74696E69-6472-632E-7069-6…        File  Freeform     Initrd
      Volume free space                 Fre…
  ▶ FA4974FC-AF1D-4E5D-BDC5-DAC…        Vol…  FFSv2
    Padding                             Pad…  Non-empty
```

Figure 2: A comparison of stock and KrakenBoot firmware marking the removal of many vendor-specific DXE drivers in favor of Linux drivers.

Figure 2 above shows how many of the vendor-specific DXE drivers are replaced with a modified `DxeCore` and Linux/initramfs pair. This is the pre-boot environment from which the Kraken provisioner will run. From here, one can take control of how OS booting/provisioning is done, etc.

6. Conclusion & Future Work

- We successfully created a KrakenBoot firmware image identical in size to the original Comanche C99X firmware, but with more efficient spatial usage of the DXE volume of the vendor firmware while getting rid of many vendor-specific binary blobs.
- Improving boot time was inconclusive due to time constraints during testing, but we hope to subsequently continue with testing.
- Choosing which DXE drivers to omit for the testing board needs more extensive research and as of now requires manual sifting.
- Expanding the KrakenBoot project to other platforms is certainly feasible, albeit each platform needs to be supported manually on a case-by-case basis.