Sysadmin Miniconf at Linux.conf.au 2021

Saturday, Jan 23 · 15:45 – 16:05 UTC+11

FUJITSU

shaping tomorrow with you

Getting started with LinuxBoot Firmware on AArch64 Server

Naohiro Tamura (naohirot@fujitsu.com)

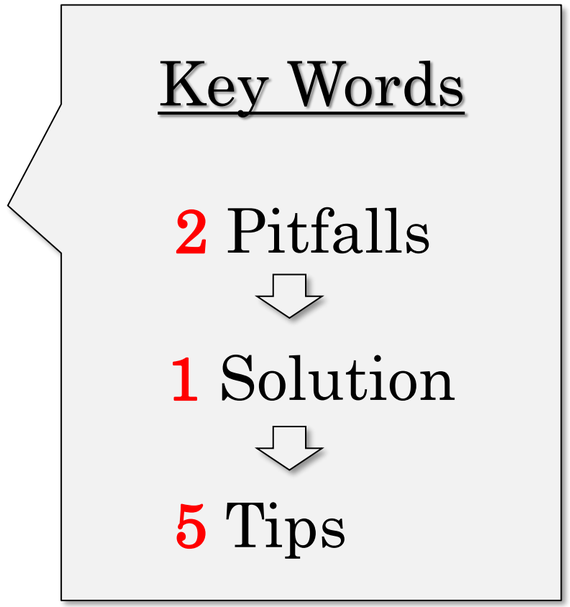
Professional Engineer
Fujitsu Limited

Slides: https://github.com/NaohiroTamura/LCA2021

Outline

FUJITSU

- My Motivation, Your Merit, and Our Goal
- What is LinuxBoot and its 2 Pitfalls?
- Solution
- How to create, boot and debug Flashrom
  - Tip 1: Create AArch64 OVMF 32MB Firmware File System
  - Tip 2: Configure LinuxBoot Kernel and Initramfs
  - Tip 3: Inject LinuxBoot into QEMU 64MB Flashrom
  - Tip 4: Boot Final OS from Local Disk
  - Tip 5: Debug LinuxBoot AArch64 Kernel using QEMU and GDB on x86_64
- What To Do Next?
- Summary



My Motivation, Your Merit and Our Goal

FUJITSU

My Motivation

Don’t Repeat My Struggle by sharing 2 Pitfalls, 1 Solution and 5 Tips.

- Last year I investigated LinuxBoot for AArch64 Server Project.
- Because **LinuxBoot is Mega Datacenter Customer’s Requirement for Security**.

ARMv8 is becoming important for SysAdmin.
Fujitsu A64FX won HPC Top500 last Jun. and Nov.
Apple M1 and AWS Graviton2 became available.
Bloomberg said Microsoft is planning.*

Your Merit

Be able to explain LinuxBoot AArch64 to your Boss and Colleagues with 100% Confidence.

- Because of getting LinuxBoot AArch64 Box Today without purchasing any additional HW at all.

Our Goal

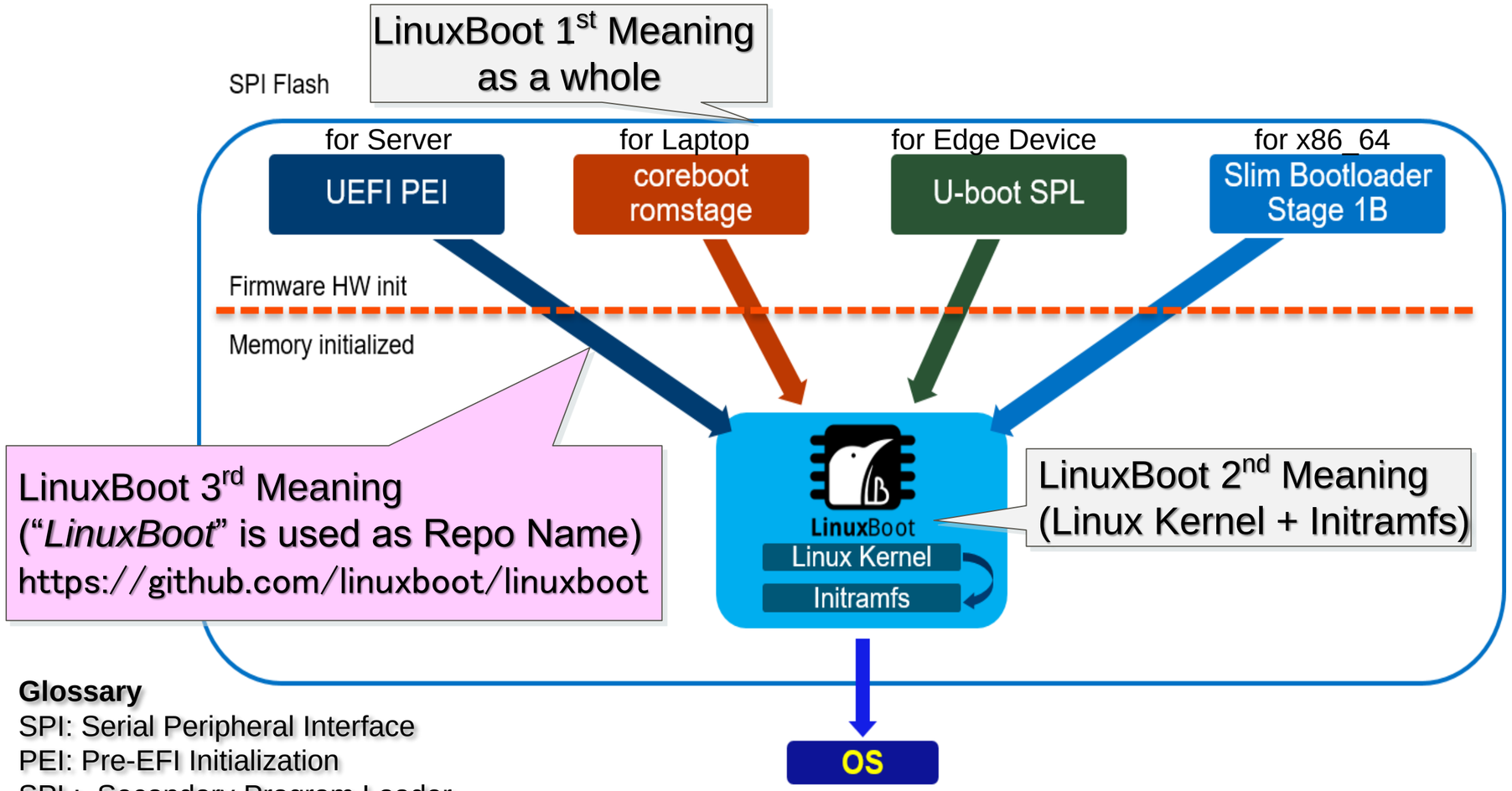
Boot Final OS, CentOS 8.2 AArch64, from LinuxBoot Flashrom using QEMU by Ourselves

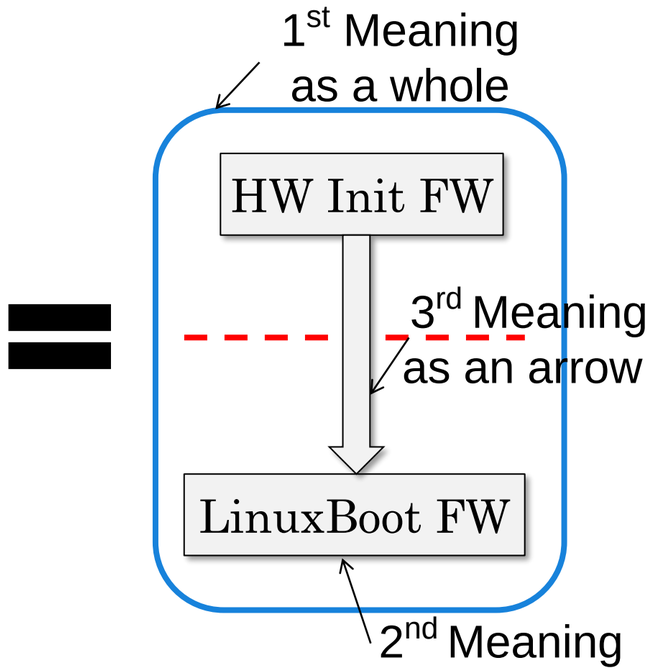
- Consider CentOS 8.2 as RHEL 8.2 which enterprise customers mostly use
- All steps to reproduce are available at https://github.com/NaohiroTamura/LCA2021

* Source: https://www.bloomberg.com/news/articles/2020-12-18/microsoft-is-designing-its-own-chips-for-servers-surface-pcs

What is LinuxBoot?

- "LinuxBoot" has Three Meanings depending on Contexts.
  - **We focus on LinuxBoot 3rd Meaning (UEFI PEI to LinuxBoot 2nd) because it's for Server**





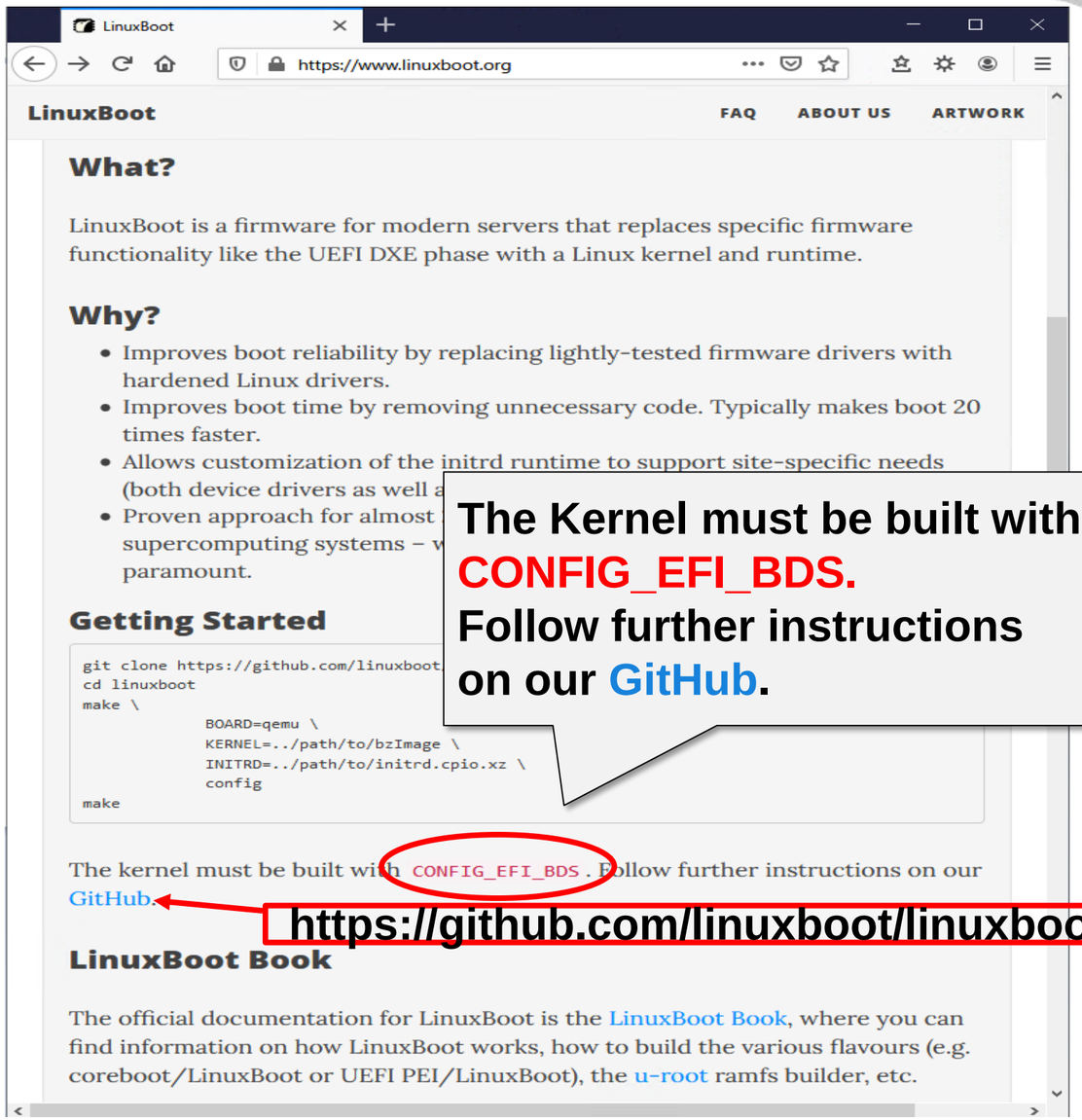
Essence of LinuxBoot

Glossary
SPI: Serial Peripheral Interface
PEI: Pre-EFI Initialization
SPL: Secondary Program Loader
BDS: Boot Device Selector

Source: https://www.linuxboot.org/

The 1st Pitfall: No BDS Kernel Param and Patch for AArch64

- No such Kernel Param CONFIG_EFI_BDS
  - The GitHub provides no further instructions
- But found BDS Kernel Patch in HEADS repo
  - This Kernel Patch is to x86_64 arch dependent code, so **it's NOT applicable to AArch64**
    - https://github.com/osresearch/heads/blob/master/patches/linux-4.14.62/0000-efi_bds.patch
- Basically What is BDS ?
  - BDS (Boot Device Selector)



Source: https://www.linuxboot.org/

The 2nd Pitfall: No LinuxBoot BDS Source Code for AArch64

- BDS is a phase of UEFI Boot. LinuxBoot BDS selects Flashrom Device and boot
  - https://github.com/linuxboot/linuxboot/blob/master/dxe/linuxboot.c

2nd Pitfall is that BDS Source Code is for x86_64, **but not AArch64**

So What should we do?

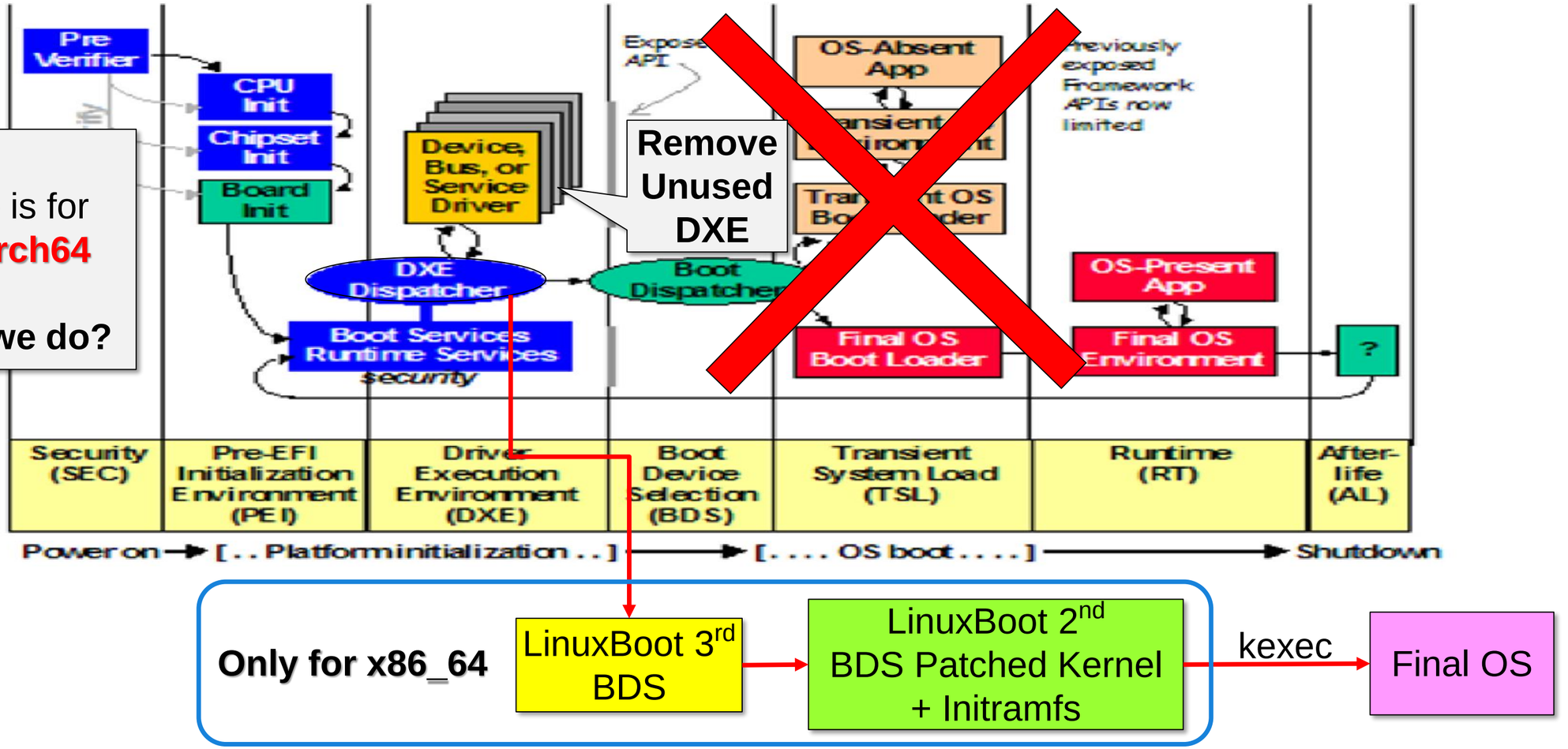
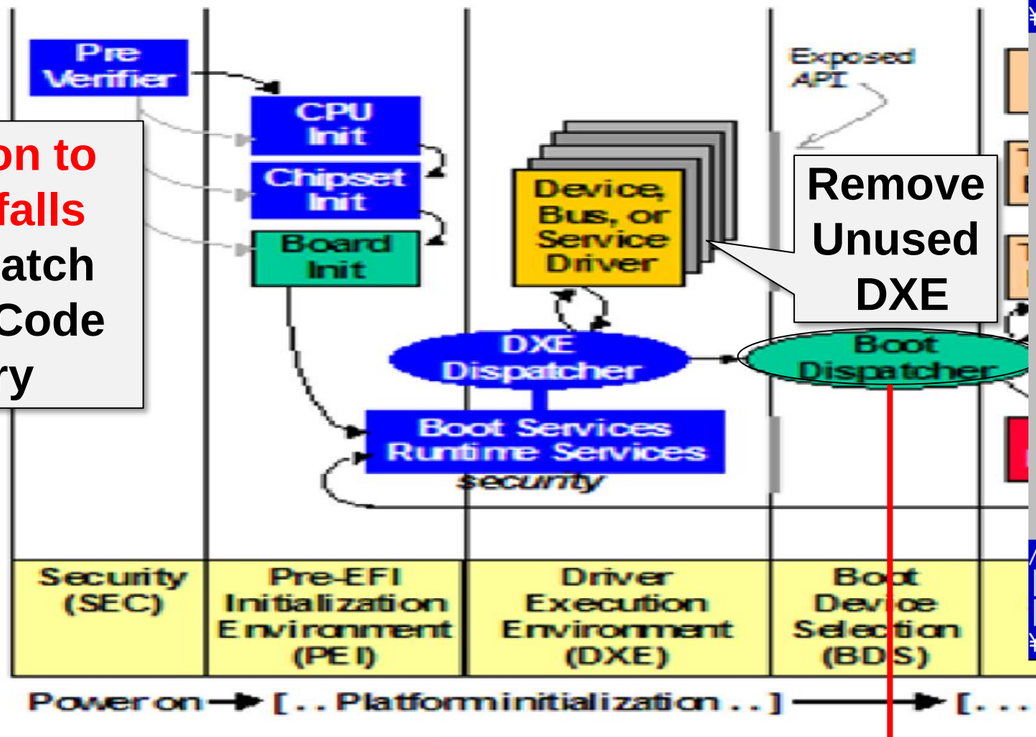


Figure Source: https://edk2-docs.gitbook.io/edk-ii-build-specification/2_design_discussion/23_boot_sequence

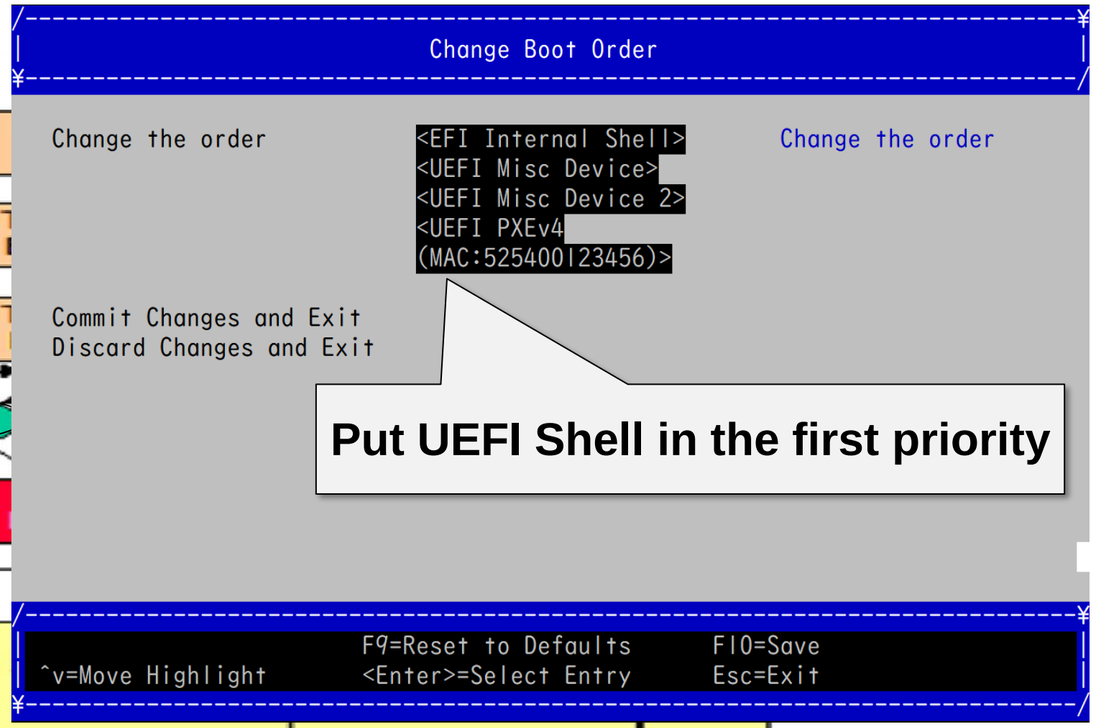
Solution: Replace UEFI Shell with LinuxBoot in Flashrom

- Fiano replaces UEFI Shell with LinuxBoot then BootManager calls LinuxBoot 2nd
  - https://github.com/linuxboot/fiano

This is the Solution to the 1st and 2nd Pitfalls because Kernel Patch and BDS Source Code are NOT necessary



Remove Unused DXE



Put UEFI Shell in the first priority

How do we create, boot and debug Flashrom?

Solution for AArch64

LinuxBoot 2nd as UEFI Shell Kernel (no patched) + Initramfs

kexec

Final OS

Figure Source: https://edk2-docs.gitbook.io/edk-ii-build-specification/2_design_discussion/23_boot_sequence

5 Tips for LinuxBoot AArch64

FUJITSU

- How to create, boot and debug Flashrom
  - Tip 1: Create AArch64 OVMF 32MB Firmware File System
  - Tip 2: Configure LinuxBoot Kernel and Initramfs
  - Tip 3: Inject LinuxBoot into QEMU 64MB Flashrom
  - Tip 4: Boot Final OS from Local Disk
  - Tip 5: Debug LinuxBoot AArch64 Kernel using QEMU and GDB on x86_64

- Flashrom Size Requirement and Challenge
  - Flashrom Size is 32MB
    - Low End Physical AArch64 Server has only 32MB Flashrom.
    - Trusted Firmware (8MB)+ UEFI(8MB) + LinuxBoot (Kernel + Initramfs) < 32MB
  - AArch64 Kernel has to be stored in uncompressed (3 times larger than compressed)
    - Because AArch64 doesn't support Self Decompression PE/COFF Kernel Image, but X86, x86_64 and AArch32 do
    - FYI, CentOS 8.2 generic kernel size gzip 8MB and gunzip 25MB

LinuxBoot 2nd (Kernel + Initramfs) has to be less than 16MB without Kernel Compression

Tip 1: Create AArch64 OVMF 32MB Firmware File System

FUJITSU

- OVMF File Size is only 2MB, so no room to replace UEFI Shell with LinuxBoot 2nd
  - OVMF (Open Virtual Machine Firmware) is UEFI implementation for QEMU and KVM
    - AArch64 https://github.com/tianocore/edk2/tree/master/ArmVirtPkg

- How to extend Firmware File System to 32MB?
  - Increase Flash Device # of Blocks in OVMF Source Code
    - FD Block Size = 4096 Byte
    - FD Size 2MB = 512 Blocks
    - FD Size 32MB = 8,192 Blocks

Apply Patch and Rebuild OVMF

    - https://github.com/NaohiroTamura/edk2/compare/edk2-stable202008...aarch64-flashrom.patch

      ArmVirtPkg/ArmVirt.dsc.inc | 7 ++++++-
      ArmVirtPkg/ArmVirtQemu.fdf | 4 ++++
      2 files changed, 10 insertions(+), 1 deletion(-)

Tip 2: Configure LinuxBoot Kernel and Initramfs

FUJITSU

- LinixBoot 2nd (Kernel + Initramfs) has to be One File, and Size < 16MB

- How to Minimize Kernel with embedded Initramfs?
  - Repeat Kernel Config Trial and Error using GDB
    - CONFIG_EFI_STUB=y
    - CONFIG_INITRAMFS_SOURCE="../initramfs.linux_arm64.cpio.xz“
    - CONFIG_INITRAMFS_COMPRESSION_XZ=y
    - # CONFIG_MODULES is not set
    - Enable ACPI Support
    - Minimized Kernel Defconf is available (https://github.com/NaohiroTamura/LCA2021/blob/master/linuxboot-5.9.0-aarch64_defconfig)
  - Chose u-root as Initramfs
    - u-root is implemented in Golang for Security (https://github.com/u-root/u-root)
    - Build with minimum commands
      $ GOARCH=arm64 u-root -build=bb -o=initramfs.linux_arm64.cpio -uinitcmd=boot core github.com/u-root/u-root/cmds/boot/boot
    - XZ compress to 3.5MB
      $ xz --check=crc32 -9 --lzma2=dict=1MiB --stdout initramfs.linux_arm64.cpio | dd conv=sync bs=512 of=initramfs.linux_arm64.cpio.xz

LinuxBoot 2nd became15MB
(Kernel 5.9 with embedded u-root)

Tip 3: Inject LinuxBoot into QEMU 64MB Flashrom

FUJITSU

- QEMU ‘virt’ machine requires 64MB Flashrom, but not 32MB

- How to replace UEFI Shell with LinuxBoot 2nd?
  - First use ‘dd’ to extend 32MB Flashrom to 64MB by just filling out Zero
    - OVMF RPM Spec file https://src.fedoraproject.org/rpms/edk2/blob/master/f/edk2.spec#_384-386

    $ dd of="arm/QEMU_EFI-pflash.raw" if="/dev/zero" bs=1M count=64

    $ dd of="arm/QEMU_EFI-pflash.raw" if="arm/QEMU_EFI.fd" conv=notrunc

  - Then use ‘replace_pe32’ subcommand of Fiano ‘utk’
    - https://github.com/linuxboot/fiano

Input: OVMF Flashrom 64MB

LinuxBoot 2nd PE/COFF Image 15MB
(Kernel 5.9 with embedded u-root)

```
$ utk QEMU_EFI-pflash.raw replace_pe32 Shell build-5.9.15/arch/arm64/boot/Image ¥
> save QEMU_EFI-pflash-linux.raw
```

Output: Linux Boot Flashrom 64MB

Tip 4: Boot Final OS from Local Disk

FUJITSU

- CentOS 8 follows Boot Loader Spec that u-root (Initramfs) hasn't implemented yet
  - Boot Configuration Format, Grub2 'menuentry', is changed
    - https://systemd.io/BOOT_LOADER_SPECIFICATION/

- How to boot Final OS, CentOS 8.2 from Local Disk using QEMU?
  - Apply Quick Hack Patch to u-root and rebuild LinuxBoot Flashrom
    - https://github.com/NaohiroTamura/u-root/compare/04f343dd1922457c530a90b566789fe1707d591d...centos8-bls-support.patch

```
$ /opt/qemu-5.1.0/bin/qemu-system-aarch64 -m 8192 ¥
-drive if=pflash,format=raw,readonly,file=QEMU_EFI-pflash-linux.raw ¥
-drive if=pflash,format=raw,file=vars-template-pflash.raw ¥
-device virtio-rng-pci -nographic -serial mon:stdio ¥
-machine virt,accel=tcg -cpu cortex-a72 -smp 4 ¥
-hda centos8-aarch64-lvm.qcow2
```

LinuxBoot Flashrom
Linux Kernel 5.9 (embedded u-root Initramfs)

'virt' machine supports ACPI

Final OS CentOS 8.2

Tip 4: Boot Final OS from Local Disk (Console Log)

```
Memory Type Information settings change.
[Bds]Booting EFI Internal Shell
[Bds] Expand Fv(64074AFE-340A-4BE6-94BA-91B5B4D0F71E)/FvFile(7C04A583-9E3E-4FIC-AD65-E05268D0B4D1) -> Fv(64074A
FE-340A-4BE6-94BA-91B5B4D0F71E)/FvFile(7C04A583-9E3E-4FIC-AD65-E05268D0B4D1)
BdsDxe: loading Boot0002 "EFI Internal Shell" from Fv(64074AFE-340A-4BE6-94BA-91B5B4D0F71E)/FvFile(7C04A583-9E3
E-4FIC-AD65-E05268D0B4D1)
InstallProtocolInterface: 5B1B31A1-9562-11D2-8E3F-00A0C969723B 23AAF5440
Loading driver at 0x00235AE0000 EntryPoint=0x002364AF9BC
Loading driver at 0x00235AE0000 EntryPoint=0x002364AF9BC
InstallProtocolInterface: BC62157E-3E33-4FEC-9920-2D3B36D750DF 23AAF2718
ProtectUefiImageCommon - 0x3AAF5440
  - 0x0000000235AE0000 - 0x000000000F70000
SetUefiImageMemoryAttributes - 0x0000000235AE0000 - 0x0000000000010000 (0x0000000000004008)
SetUefiImageMemoryAttributes - 0x0000000235AF0000 - 0x000000000009F0000 (0x0000000000020008)
SetUefiImageMemoryAttributes - 0x00000002364E0000 - 0x0000000000570000 (0x0000000000004008)
BdsDxe: starting Boot0002 "EFI Internal Shell" from Fv(64074AFE-340A-4BE6-94BA-91B5B4D0F71E)/FvFile(7C04A583-9E
3E-4FIC-AD65-E05268D0B4D1)
EFI stub: Booting Linux Kernel...
EFI stub: Generating empty DTB
EFI stub: Exiting boot services and installing virtual address map...
SetUefiImageMemoryAttributes - 0x00000023BE80000 - 0x0000000000040000 (0x0000000000000008)
```

```
Booting Linux on physical CPU 0x0000000000 [0x410fd083]
Linux version 5.9.15 (ubuntu@bionic) (aarch64-linux-gnu-gcc (Ubuntu/Linaro 7.5.0-3ubuntu1~18.04) 7.5.0, GNU ld
(GNU Binutils for Ubuntu) 2.30) #1 SMP Tue Dec 22 11:18:00 UTC 2020
efi: EFI v2.70 by EDK II
efi: SMBIOS 3.0=0x23bef0000 MEMATTR=0x239488698 ACPI 2.0=0x238830000 RNG=0x23bffcd98 MEMRESERVE=0x238b63f18
efi: seeding entropy pool
```

Tip 5: Debug LinuxBoot AArch64 using QEMU and GDB

FUJITSU

- Terminal 1

```
$ /opt/qemu-5.1.0/bin/qemu-system-aarch64 -s -S -m 8192 ¥
-drive if=pflash,format=raw,readonly,file=QEMU_EFI-pflash-linux.raw ¥
-drive if=pflash,format=raw,file=vars-template-pflash.raw ¥
-device virtio-rng-pci -nographic -serial mon:stdio ¥
-machine virt,accel=tcg -cpu cortex-a72 ¥
-hda centos8-aarch64-lvm.qcow2
```

When LinuxBoot Kernel doesn't start, GDB debug helps us find missing driver.

- Terminal 2

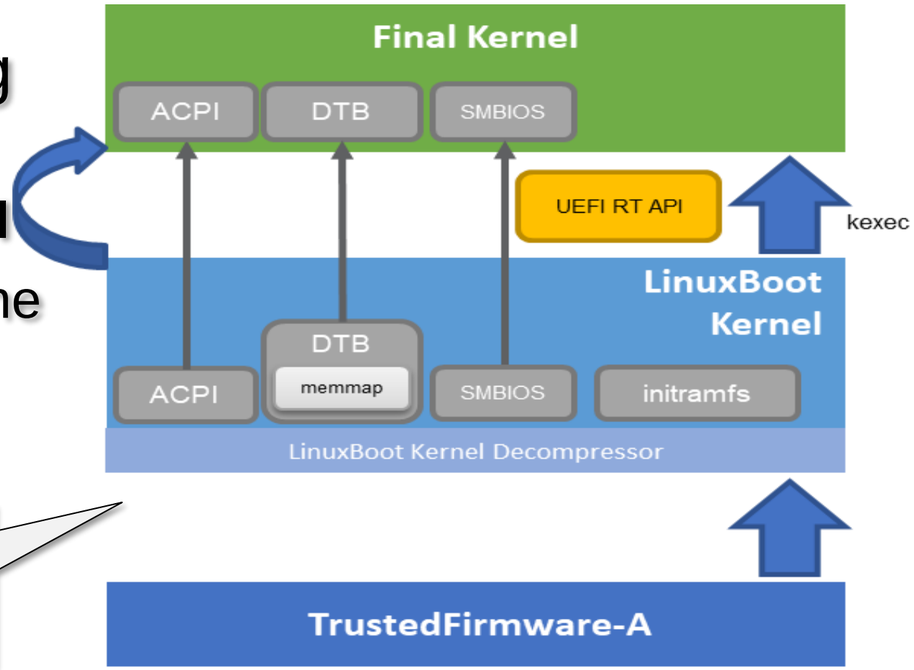
```
$ /opt/gdb-9.2/bin/aarch64-gnu-linux-gnu-gdb build-5.9.15/vmlinux
...
Reading symbols from build-5.9.15/vmlinux...
(gdb) target remote :1234
Remote debugging using :1234
0x0000000000000000 in ?? ()
(gdb) b start_kernel
Breakpoint 1 at 0xffffffe0010990da4: file /home/ubuntu/LCA2021/linux-5.9.15/init/main.c, line 847.
(gdb) c
Continuing.
Breakpoint 1, start_kernel () at /home/ubuntu/LCA2021/linux-5.9.15/init/main.c:847
847     {
(gdb)
```

Default Port 1234

What TO DO Next?

- Develop Kernel Decompressor UEFI Application for Fiano
  - Fiano replaces UEFI Shell with the Decompressor, then the Decompressor calls LinuxBoot 2nd
  - It' really peculiar why NOT only AArch64 kernel self-decompressor implemented
    - Because each loader such as Grub2, u-root and etc has to implement decompressor repeatedly
    - Found discussion once on the mailing list in Jan 2014 (http://lists.infradead.org/pipermail/linux-arm-kernel/2014-January/224746.html), but no more

- Watch LinuxBoot 3rd ARM Company is implementing
  - At OCP Summit 2020 ARM presented LinuxBoot which Trusted Firmware calls Kernel Decompressor **skipping UEFI**
  - When it's ready, we can try it using QEMU SABA-Ref machine
    - SABA (Server Base System Architecture)
    - https://github.com/tianocore/edk2-platforms/tree/master/Platform/Qemu/SbsaQemu



Trusted Firmware-A directly calls Decompressor **without UEFI**

Source: https://2020ocpvirtualsummit.sched.com/event/bXVn/open-system-firmware-on-arm

Summary

FUJITSU

- Explained 2 Pitfalls, 1 Solution and 5 Tips.

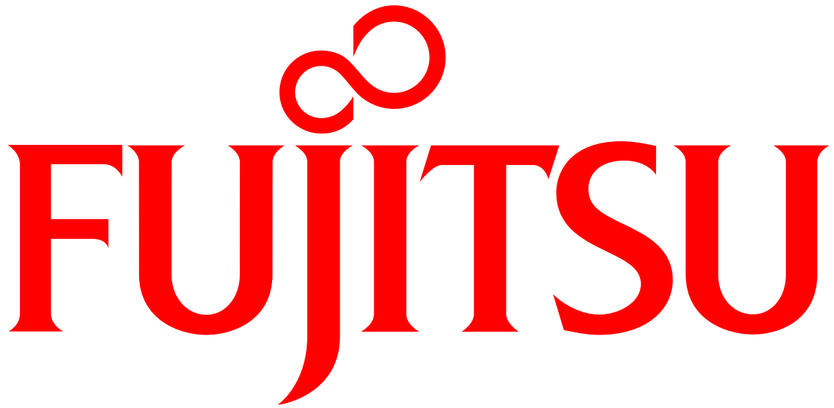
- You can boot Final OS, CentOS 8.2, from LinuxBoot Flashrom using QEMU
  - All steps to reproduce are available at https://github.com/NaohiroTamura/LCA2021

- Try it by yourself and explain LinuxBoot AArch64 to your Boss and Colleagues
  - Please send me an email or submit an Issue to the GitHub if you had any problem.

References

- Arm SystemReady and the UEFI firmware ecosystem
  - https://cfp.osfc.io/osfc2020/talk/KB3H9V/
- Open System Firmware on Arm *
  - https://2020ocpvirtualsummit.sched.com/event/bXVn/open-system-firmware-on-arm
- Go Forth and Modify: Fiano *
  - https://2020ocpvirtualsummit.sched.com/event/bXWK/go-forth-and-modify-fiano
- Firmware security, why it matters and how you can have it
  - https://2019.linux.conf.au/schedule/presentation/110/
- EDKII OVMF AArch64
  - https://github.com/tianocore/edk2/tree/master/ArmVirtPkg
- u-root
  - https://github.com/u-root/u-root
- LinuxBoot
  - https://github.com/linuxboot/linuxboot
- fiano
  - https://github.com/linuxboot/fiano

*) Downloading slide needs to enter the OCP Virtual Summit from https://www.opencompute.org/summit/virtual-summit



shaping tomorrow with you