OF != UEFI

Considerations for Directly Booting Linux in Power Logical Partitions

Nayna Jain and George Wilson, IBM

Linux Plumbers

System Boot and Security Microconference

September 18, 2024

Disclaimer

- These slides represent the authors' views, not necessarily IBM's.
- All design points disclosed herein are subject to finalization and upstream acceptance.
- The features described may never ultimately exist or take the described form in a product.
- IBM is a registered trademark of International Business Machines Corporation in the United States and/or other countries.
- Linux is a registered trademark of Linus Torvalds.
- Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Other company, product, and service names may be trademarks or service marks of others.

Parallel Universes

Open Firmware

- Originated by Sun for SPARCstations
- Was IEEE 1275-1994 Spec
- Multiple vendor implementations
- PowerPC, ARM, x86_64
- Reference source & specs from OpenBIOS
- ISA-independent FCode drivers
- DeviceTree HW representation
- OS handover by executing client program
- Run Time Abstraction Services (RTAS)
- . . .

Unified Extensible Firmware Interface

- Originated by Intel for Itanium
- UEFI Forum Spec
- Multiple vendor implementations
- x86_64, ARM, etc.; ports for PowerPC
- Reference source from TianoCore EDK II
- ISA-independent EBC drivers
- ACPI/SMBIOS HW representation
- OS handover by executing EFI binary
- EFI runtime services
- . . .

Power Servers

- All Power servers today ship with the PowerVM hypervisor included in FW
- PowerVM is a type 1 hypervisor that owns the HW and presents Logical Partitions (LPARs) in which guest OSs execute
- **Partition FW (PFW) is OF based**
- Must support Linux (LE, BE deprecated), AIX (BE), and IBM i (BE); various BSD flavors can also run
- **The entire FW stack, including Partition FW, is signed, verified, and measured; updates are authenticated; secure boot extends integrity verification into OSs**
- The Power platform is documented in the *Power Architecture Platform Reference* (PAPR), a new public version of which will soon be published
- The current public version of the PAPR is the *Linux on Power Architecture Platform Reference* (LoPAPR), which I reference but we consider deprecated
- The FW code is proprietary and the Linux team is firewalled off from it

Boot Image Characteristics

Client Program Form

- Per LoPAPR § B.10.4, "LoPAPR only supports the 32-bit version (i.e., ELFCLASS32) for 32 and 64 bit platforms."
- GRUB core.elf is indeed built as a 32-bit BE ELF
- But LoPAPR § B.11.4 says, "OF must recognize a client program that is formatted as ELF, as defined in *System V Application Binary Interface, PowerPC Processor Supplement [15]*, and PE, as defined in *Peering Inside the PE: A Tour of the Win32 Portable Executable File Format [13]*. Other formats may be handled in an implementation-specific manner."
- **The Partition FW team states that in practice**
  - **It is possible to boot a 64-bit LE binary with proper ELF header entries**
  - **Booting PE client programs is not supported, though that might open up some interesting possibilities for common tooling across platforms**
- There may be no need to implement a stub bootloader

Boot Image Characteristics (cont'd)

Image Size

- The **real-base** (OF starting base address) and **load-base** (default address at which the client program will be loaded) **OF configuration variables must be set appropriately to reserve memory for the larger image size**
- **These should be settable via ELF notes**
- However, the PFW team states that past attempts to set the variables via the ELF notes failed
- Requires test and, potentially, updates to PFW

Client Architecture Support (CAS) Negotiation

What It Is

- **After OF handover to the client program, the client program and OF exchange information on their respective properties**
- The properties include such items as memory layout, hash vs radix PT, and much more
- Real Mode Area (RMA) size is negotiated by GRUB in ieee1275/init.c
- Other properties are negotiated in Linux prom_init.c
- Some property changes require a reboot; some are sticky across reboots

Client Architecture Support (CAS) Negotiation (cont'd)

The Problem

- **Properties that affect DT, once negotiated, are fixed for the lifetime of the boot cycle and cannot be re-negotiated**
- When the initial kernel kexec's a second kernel, the second kernel negotiates properties requiring a DT change will reboot and the first kernel will again negotiate and reboot
- Therefore, arbitrary kernels cannot be kexec'd
- A potential solution proposed is to add a new Run Time Abstraction Services (RTAS) call to perform CAS negotiation in the hypervisor and then resume execution after the RTAS call

Boot from Disk

- **The OF client program is installed in the PowerPC Reference Platform (PReP) partition** (called Single program image in LoPAPR § B.10.1); there is no filesystem, just the ELF binary
- Partition FW searches the boot device for a PReP partition and attempts to load and hand over control to the client program
- The client program must contain suitable ELF notes as defined in the PAPR
- The PReP partition is typically sized 10M max
- **Larger PReP partitions must be configured at OS installation to contain the larger boot image**
- A FAT filesystem package is described in LoPAPR § B.10.1 and FAT16 PFW support should be usable; it would be interesting to have a boot partition similar to an ESP that could be managed as a filesystem

Boot from Network

- BOOTP or manual adapter configuration and TFTP file transfer
- The TFTP implementation is based on RFC 1350 and, because it has a 32 MB limit, the initramfs is too large to fetch directly
- **Nowadays, netboot fetches and executes GRUB and GRUB subsequently fetches the kernel and initramfs**
- In order to directly boot the even larger composite boot image, the current limitation must be alleviated
- Larger file sizes could be achieved by implementing the RFC 2348 blocksize option
- Also, the PFW could be updated to include HTTP(S) as an alternative

Boot from ISO

- A Common Hardware Reference Platform (CHRP) script in the /boot directory of the boot media designates the client program
- **There should be not extraordinary considerations for ISO Boot**
- However, CHRP boots have provoked Real Mode Area (RMA) memory issues so this needs to be tested

Device Discovery, Boot Images, and Selection

- Devices can be presented and can be selected by the System Management Services (SMS) menu, albeit using OF device names
- **But how will discovery and selection of bootable kernel images be handled?**
  - Initial kernel
  - Kexec'd kernels
- **Will editing the kernel command line be supported?**
- **What about other OSs**, e.g., FreeBSD?
- Will chain loading be supported somehow?

Secure Boot

Signed Image

- The boot image must be signed with an appended signature as is done for kernel modules, which provides backwards compatibility with unsigned boot images
- Appropriate ELF notes must be present in the boot image header

Out-of-Tree Modules

- **If a composite image including initramfs is signed by a distribution, how will out-of-tree modules required for storage and network devices be loaded?**
- The initramfs must be verified and measured
- Either the composite image must be re-signed or the initramfs must be verified by some other means, such as IMA
- If signed modules are to be loaded from some location, where would they be stored?
- **This needs to work for the network boot case**
- Perhaps separately signed auxiliary components could be appended after the signed composite image

Secure Boot (continued)

Authenticated Variables

- We would need to consider how to apply our existing 2-level authenticated variable scheme; we do not sign distro certificates and have no MOK

Secure Boot Advanced Targeting (SBAT)

- Appears to be as useful as it is for GRUB to prevent having to store too many revocation hashes

Trusted Boot

PCR Semantics

- PCR semantics need to be considered
- Power attempts to match the semantics of the TCG PC and UEFI Specs as much as practicable for FW, bootloader, and kernel
- **The PCRs should agree with existing expectations for both sealing and remote attestation purposes**

Infinite Logs

- There's also the problem of the next kexec'd kernel and infinite logs and PCR chains extended without reset so they continue to match
- May not be a big problem in practice
- **Power only supports SRTM**, not DRTM, launches
- DRTM, were it implemented, would likely look like something other than an ISA enhancement

Additional Considerations

- We also need to consider the effects of direct kernel boot for Live Partition Mobility (LPM) so that LPARs can migrate as freely as possible

Community Buy-in, Transition, and the Future

- **Linux distros should agree on a standard common boot mechanism**
- IBM does not want to preclude the ability to boot other OSs on Power
- A very important point is **all platforms will require GRUB support until completely transitioned to direct kernel boot, which could be a long time**
- What can we do as a community to facilitate GRUB support?
- Finally, the IBM Linux on Power team wants to support direct kernel boot
- **Next steps are to experiment and to continue interacting with internal and external stakeholders, as I'm doing here, to ensure that the experience on the Power platform is similar to UEFI**

References

- GRUB ieee1275/init.c: https://git.savannah.gnu.org/cgit/grub.git/tree/grub-core/kern/ieee1275/init.c
- Linux on Power Architecture Platform Reference v2.9+, https://openpowerfoundation.org/wp-content/uploads/2020/07/LoPAR-20200611.pdf
- Linux prom_init.c: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/arch/powerpc/kernel/prom_init.c
- OpenBIOS Project: https://openfirmware.info/

Thank you!

Nayna Jain
<naynjain@us.ibm.com>

George Wilson
<gcwilson@us.ibm.com>