presented by

Multi-ISA Driver Compatibility

What's the Future?

UEFI Fall 2023 Developers Conference & Plugfest

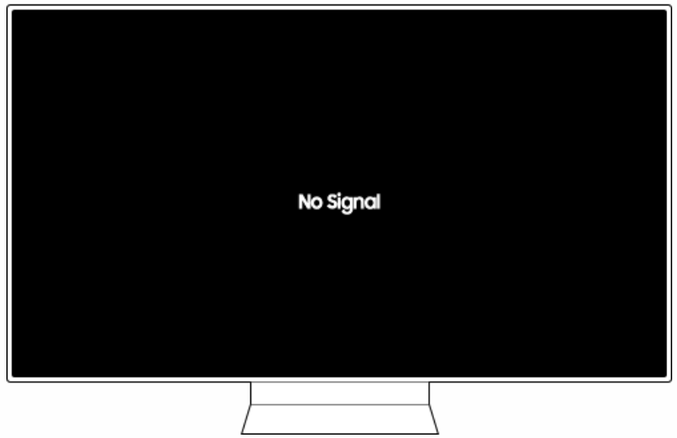
October 9-12, 2023

Andrei Warkentin (Intel)

Agenda

- Background
- Past and Current Approaches
- What's the Future?
- Discussion

Background



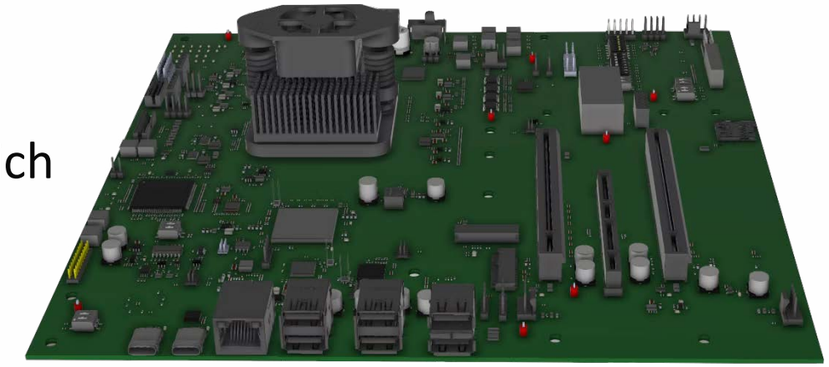
Typical non-x86 SBC

Off-the-shelf adapter (NIC/IPU, GPU, RAID)

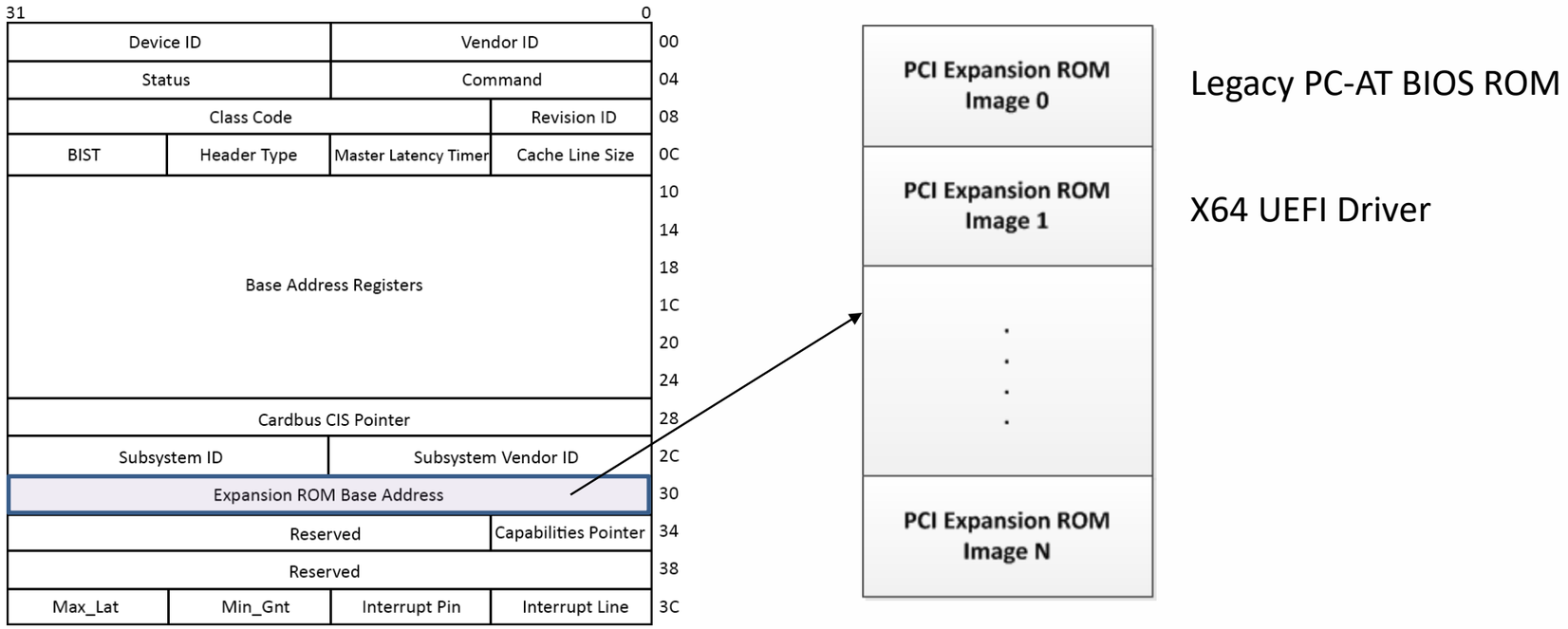
Nothing

Background

- Today ➔ $GENERIC computer world is already Multi-ISA!
  - X64, AArch64 SystemReady, LoongArch64
  - ACPI + UEFI based servers, PCs, etc.
  - PCIe/CXL connectivity for off-the-shelf devices, which are not **Multi-ISA.**
- Tomorrow ➔ More architectures and environments.
  - RISCV64
  - RISCV128
  - CHERI variants (Morello, …)



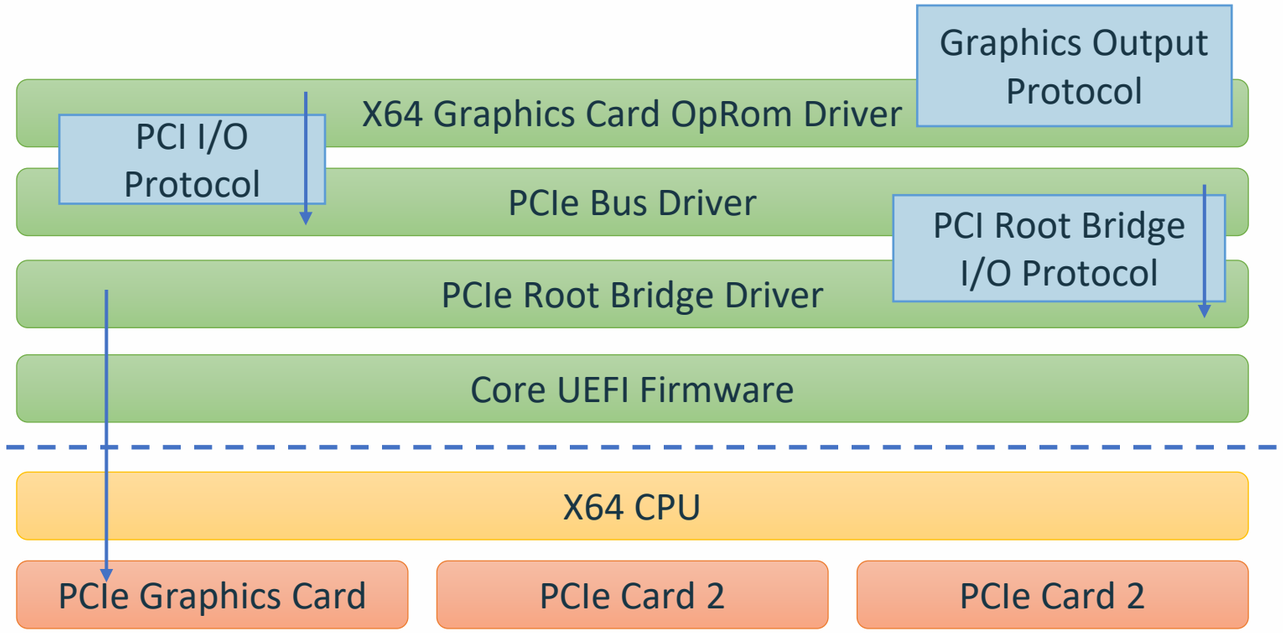
PCIe® Firmware Drivers



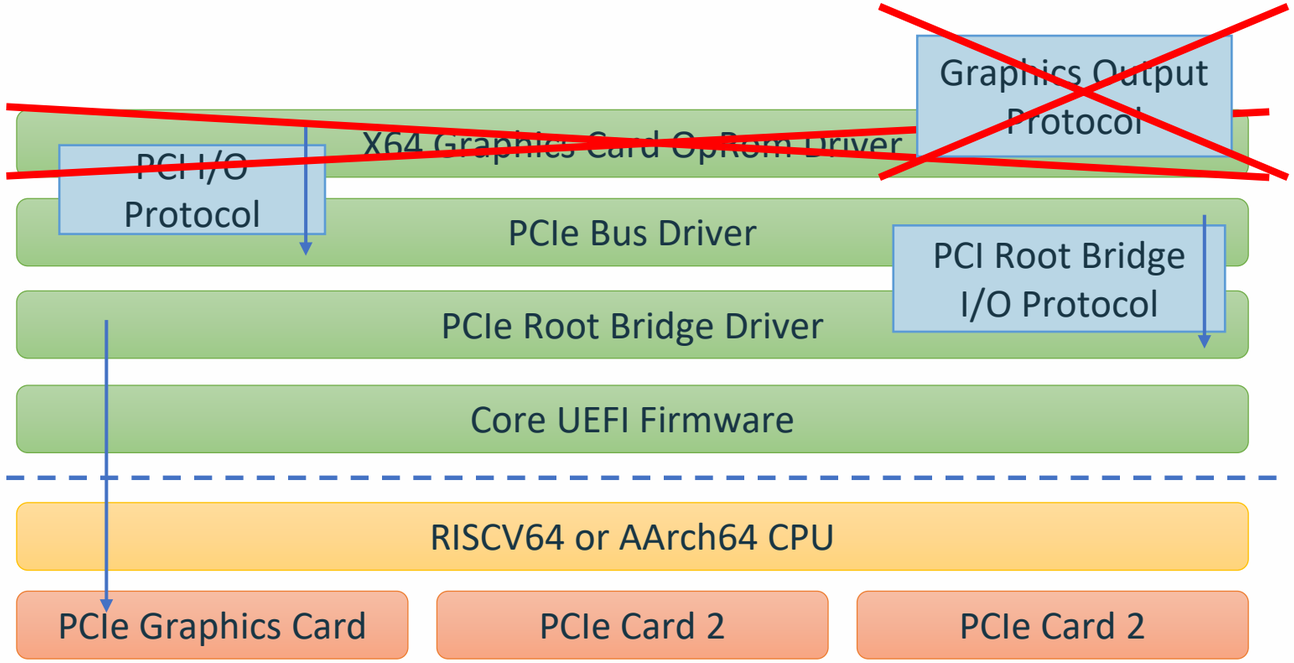
Legacy PC-AT BIOS ROM

X64 UEFI Driver

Life of PCIe® in x64 UEFI



Life of PCIe® Elsewhere



Past and Current Approaches

Past Approach – EFI Byte Code

Legacy PC-AT BIOS ROM

X64 UEFI Driver

EBC UEFI Driver

- ☑ Specifically made for this scenario!
  - Processor Independence
    - sizeof(VOID*) is a runtime operation.
    - VM takes care of 32 vs 64 vs 128-bit issues.
  - TianoCore comes with an interpreter.
- ❌ Not used by the industry!
  - No tooling – the only supported and proprietary C compiler has been retired.
    - Some OSS now exists
      - https://github.com/yabits/ebcvm / ELVM
      - https://github.com/pbatard/fasmg-ebc
  - Different performance profile - interpreted code.
  - Didn't make a come-back when the Arm ecosystem explored this space

Current AArch64 Approach

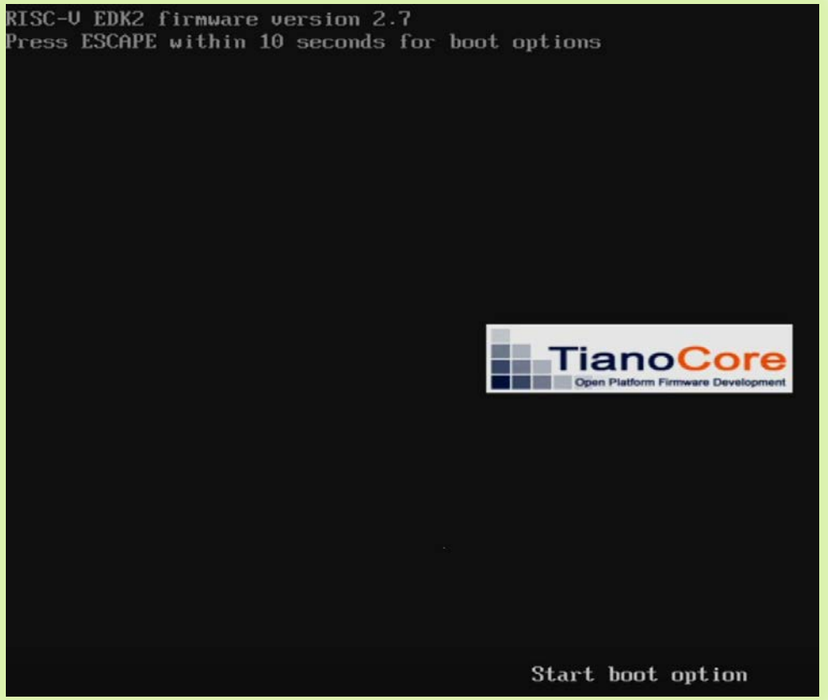
X86EmulatorPkg

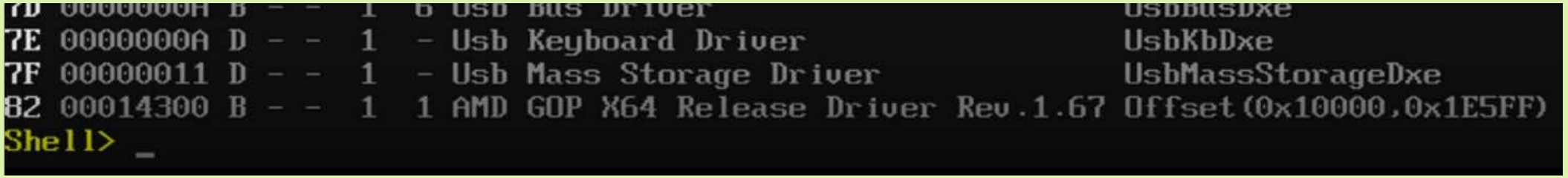
- ☑ Supports x64 OpRoms and UEFI applications on AArch64 systems
  - Open Source UEFI Boot Service Driver
  - Targets 64-bit AArch64 systems (servers, workstations)
  - Developed by Linaro engineers six years ago
  - Uses BT (Binary Translation), via Qemu Tiny Code Generator
  - https://github.com/ardbiesheuvel/X86EmulatorPkg
- ❌ Not trivially portable to RISC-V!
  - Old TCG code of unknown provenance
  - Backporting RISC-V support sounds hard (and time consuming) unless you're a Qemu guru

BT for Everybody

Rewrite of X86EmulatorPkg

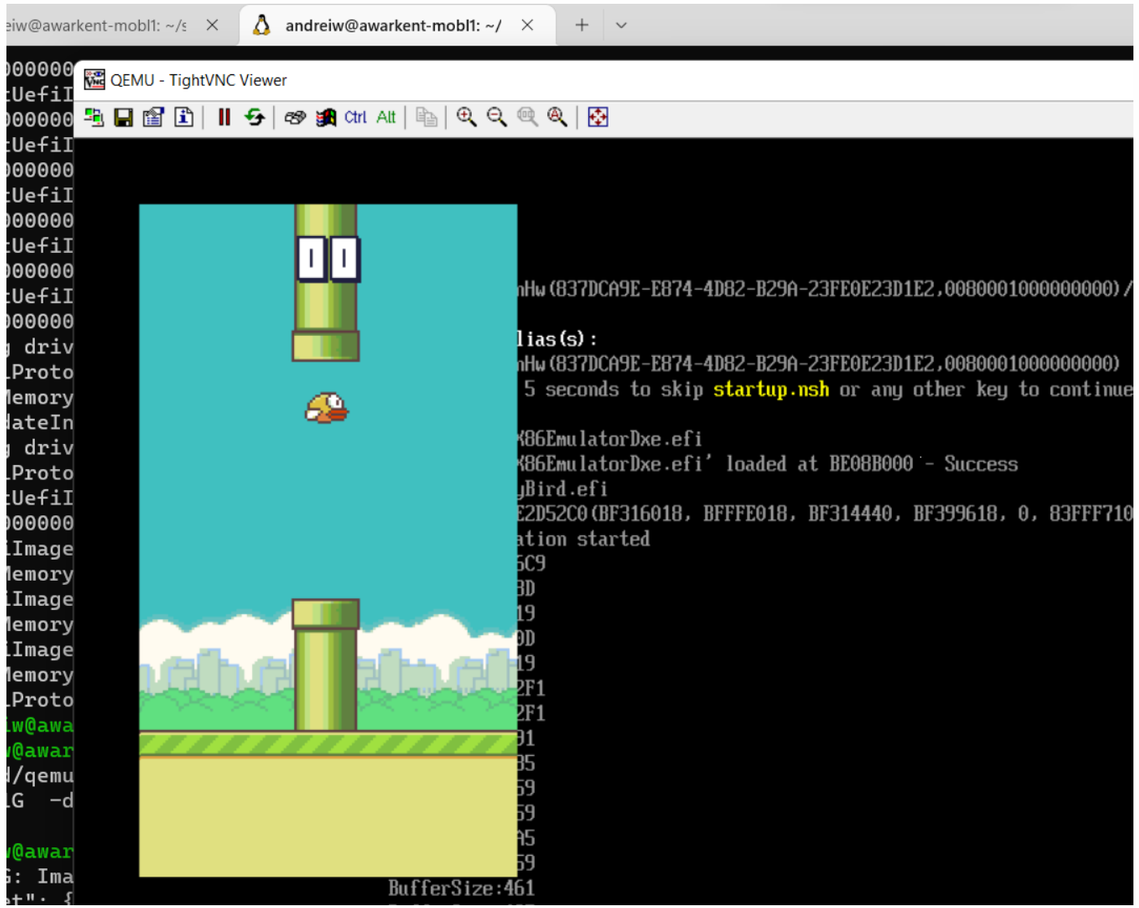
- Portable: Supports AArch64 and 64-bit RISC-V UEFI hosts.
- Tested on real hardware
  - AArch64: Raspberry Pi and Ampere Altra
  - RISCV64: StarFive VisionFiveV2
- 64-bit x64 and AArch64 UEFI Boot Service emulation
- Clean: Abstracts Qemu/TCG with Unicorn Engine API
- https://github.com/intel/MultiArchUefiPkg
- RISE Project in the Firmware WG
- Correctness, perf, size





BT for Everybody

MultiArchUefiPkg

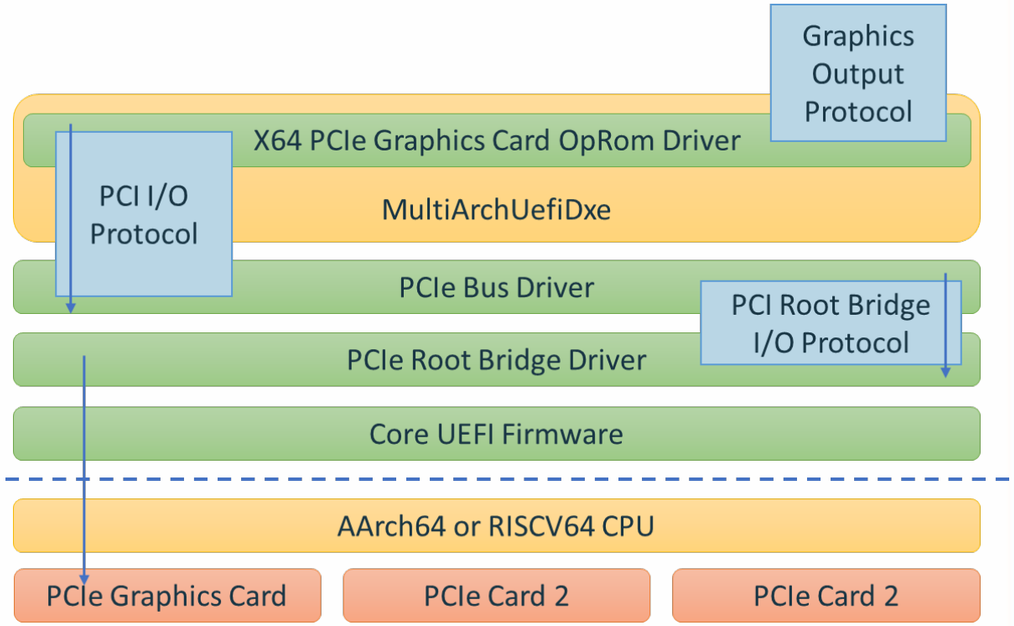


BT for Everybody

MultiArchUefiPkg

- Possible entirely due to narrowly-defined EFI ABI
- Models Boot Services environment, with certain services filtered or disabled
- Tiano support for foreign binaries - **EDKII_PECOFF_IMAGE_EMULATOR_PROTOCOL**
- Emulation is only interesting if thunking goes both ways!
  - No-Execute handler traps for native → emulated
  - Unicorn No-Execute handler traps for emulated → native

```
UINT64
EFIAPI
Fn(UINT64, UINT64, UINT64, UINT64,
   UINT64, UINT64, UINT64, UINT64,
   UINT64, UINT64, UINT64, UINT64,
   UINT64, UINT64, UINT64, UINT64);
```



Price of Binary Translation

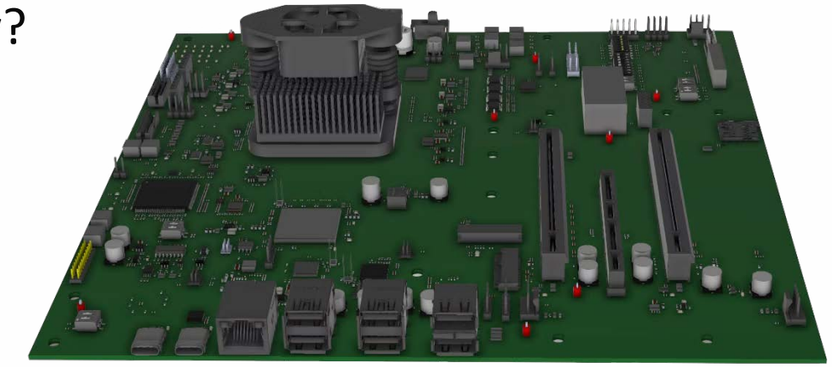
Good enough for OpRoms on cards out there, but you need testing and qualification. And you're always one OpRom upgrade from a non-POSTing system.

- Complexity
  - 500KiB for x64, ~1.7MiB for x64 + AArch64 BT
  - Tiano deps, BT bugs
  - Modelling certain BS calls without leaks is complex (Image Exit, SetJmp/LongJmp across native or across BT code)
- Technical debt
- 128-bit support requires sandboxing
- Fragility
  - ISAs are a moving target
  - "logic bombs" due to tooling or programmer error – very real
    - Code that must be loaded > 4GiB
    - Stack that must be below 4GiB
- Fragility Cont'd
  - OpRom environment unconstrained
    - Allocating, manipulatng and executing memory of type BootServiceCode?
    - Using self-modifying code?
    - Manipulating privileged state or set a CPU exception handler?
    - Changing page protection attributes (e.g. marking itself executable).
    - CallsingExitBootServices?
- Licensing headaches
  - MultiArchUefiPkg is approved for release under GPLv2 and LGPLv2.1+. EmulatorDxe itself is covered by the LGPL v2.1+, but it is statically linked to the Unicorn Engine library, which has a mix of LGPL and GPLv2 code.
  - X86EmulatorPkg is similar due to similar ancestry.

What’s the Future?

An IHV-Only Problem?

- It’s not reasonable to support every useful architecture
  - How many adapters ship with x64 + AArch64 today?
  - How many will additionally bundle RISCV64, RISCV128 and LOONGARCH support?
- Most cards only bother with x64



Embracing Emulation via EBC

- EBC was ahead of its time, and it is technically still the soundest option.
  - For example, it just disallows SetJmp/LongJmp
  - 1 driver to support x64, AArch64, LoongArch64, RISCV64 and anything else
  - Will support 128-bit ISAs
- Tooling can be fixed
  - LLVM for EBC?
  - https://speakerdeck.com/retrage/llvm-backend-development-for-efi-byte-code
  - Development discouraged by deprecation of EBC from spec
- Performance can be fixed
  - EBC JIT

Embracing Emulation via WASM

- Tooling is great, but… not for the scenario
- ISA abstraction not so much
  - WASM32 vs WASM64?
  - What about 128-bit?
- UEFI is single address space, WASM is meant to be sandboxed
  - Relocations?
  - Common memory?
  - Unique function IDs and thinking?

Embracing Emulation via eBPF

- Tooling is great.
- eBPF in the Linux kernel comes with a security model and validation
  - Enforces size, complexity, lack of looping, “safe” accesses
  - Doesn’t apply to UEFI
- 64-bit register size not future-proof

Embracing Constrained x64?

- IHVs build X64 code. Let’s meet them half-way!
- Reduce X64 ISA down to a minimum non-privileged set of instructions
  - Fixed ISA, not a moving target
  - Much simpler and smaller BT
  - An additional –march= flag to a compiler
- Compiled OpRoms continue to work on existing x64 systems
- Plenty of existing tools for static analysis, etc. No need to upset an existing flow

Let’s Discuss the Approach to Invest In?

Thanks for attending the UEFI Fall 2023 Developers Conference & Plugfest

For more information on UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by