ACRN: A Big Little Hypervisor for IoT Development

Eddie Dong, Intel Open Source Technology Center

Key contributors: Christopher Cormack, Matthew Curfman, Jeff Jackson

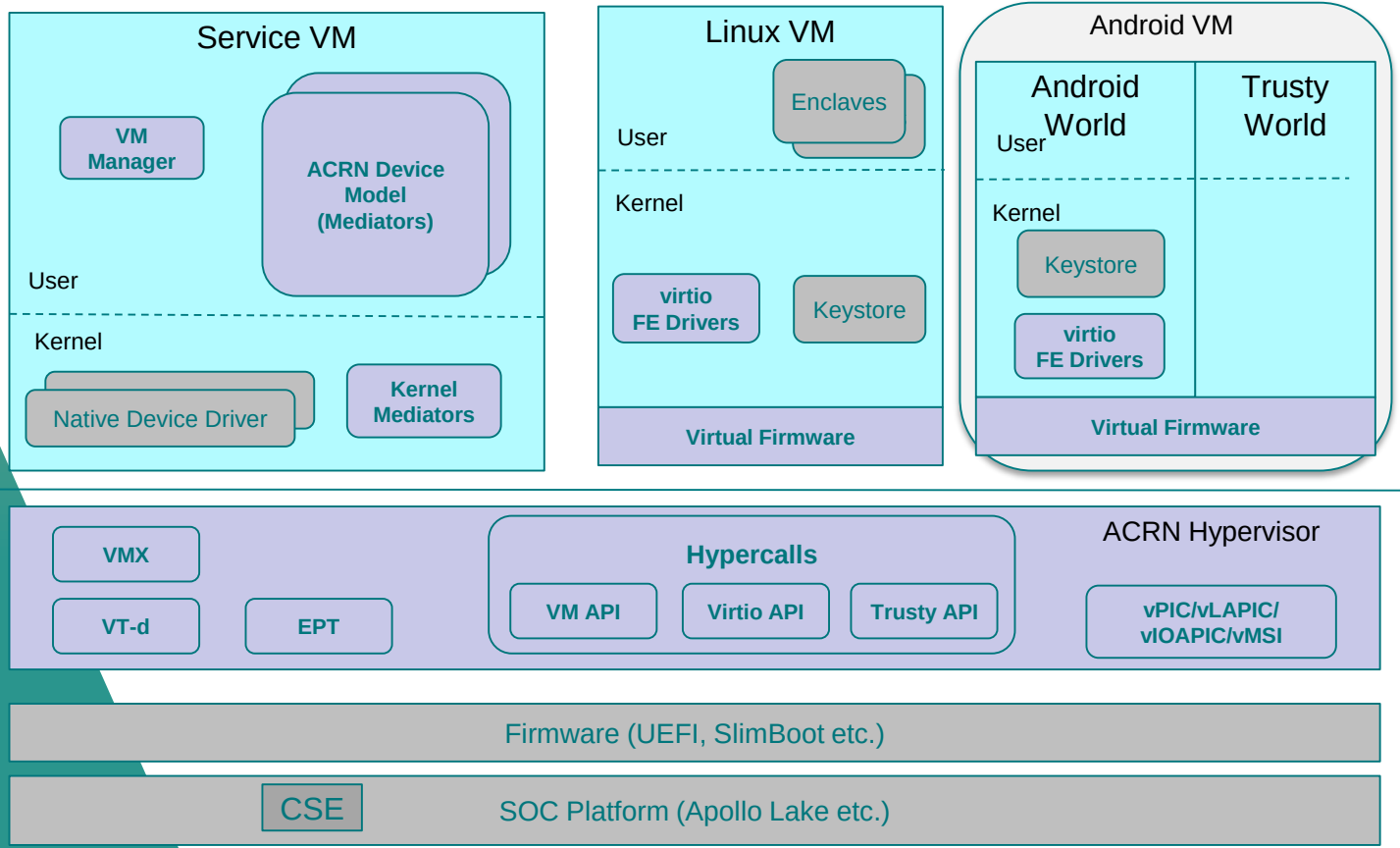
Table of Contents

What is ACRN

ACRN is a Big Little Hypervisor for IoT Development

ACRN™ is a flexible, lightweight reference hypervisor, built with real-time and safety-criticality in mind, optimized to streamline embedded development through an open source platform

Architecture Overview



ACRN as a Device Hypervisor

- Small footprint

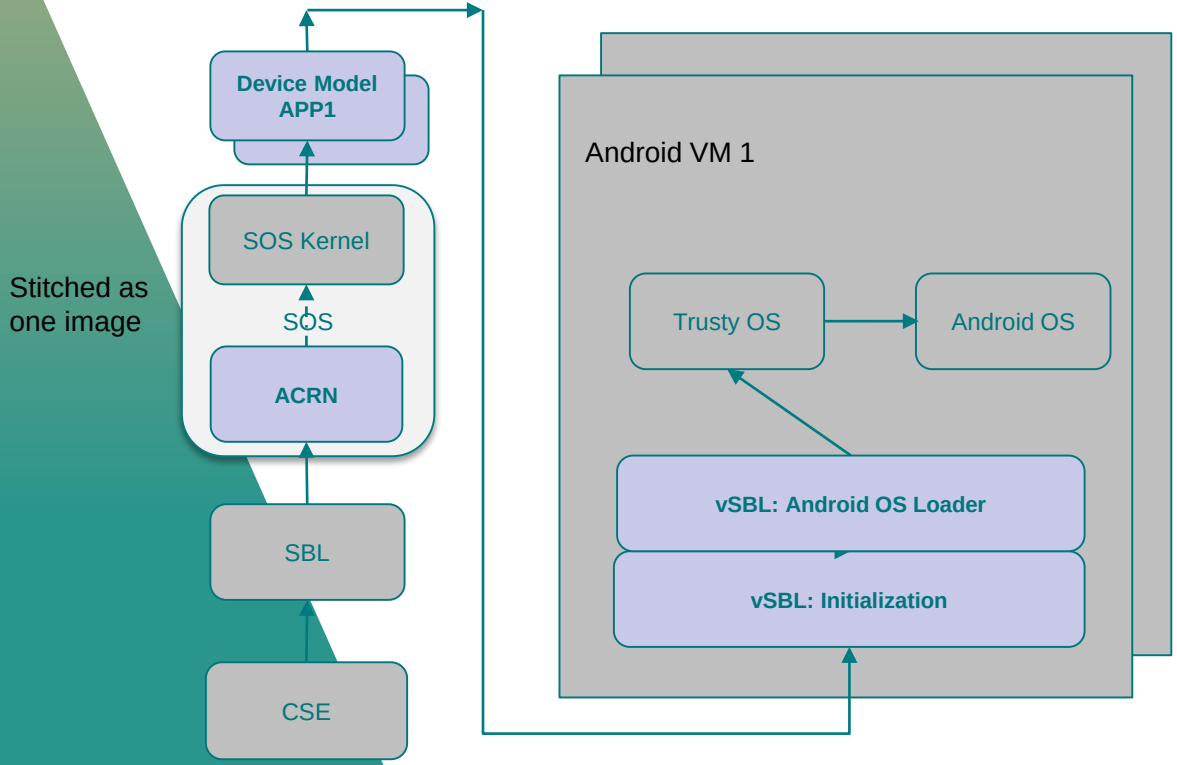
	KVM	Xen	ACRN
LOC	17M	290K	25K

- BSD licensee
- Be able to cherry pick piece of codes into OSV/OEM's own hypervisor
- Verified boot

- Rich I/O mediators

GPU	IPU	TSN	CSE	USB	Audio	Ethernet	Block	IOC	Touch
Mediated Passthru	Virtio	Virtio	Virito	Emu.	Virtio	Virtio	Virtio	Emu.	Virtio

Verified Boot Sequence with SBL



- CSE verifies SBL
- SBL verifies ACRN & SOS Kernel
- SOS kernel verifies DM & vSBL thru dm-verity
- vSBL starts the guest side verification process (reusing the Android verified boot mechanism)

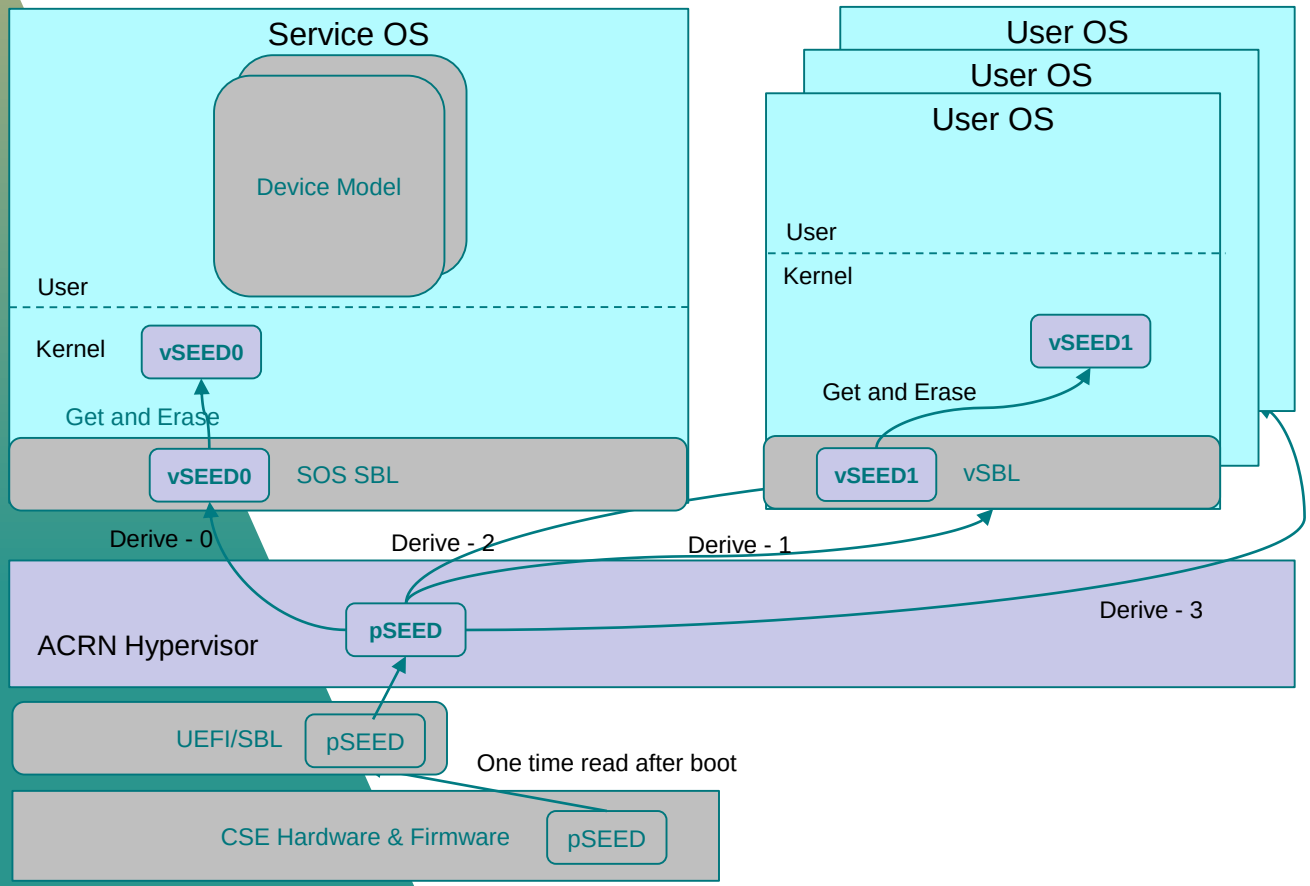
- NOTE: Each user VM has a DM APP instance in SOS

Verified Boot Sequence with UEFI

- UEFI verifies ACRN & OS Bootloader & SOS Kernel
- SOS kernel verifies DM and vSBL thru dm-verity
- vSBL starts the guest side verified boot process

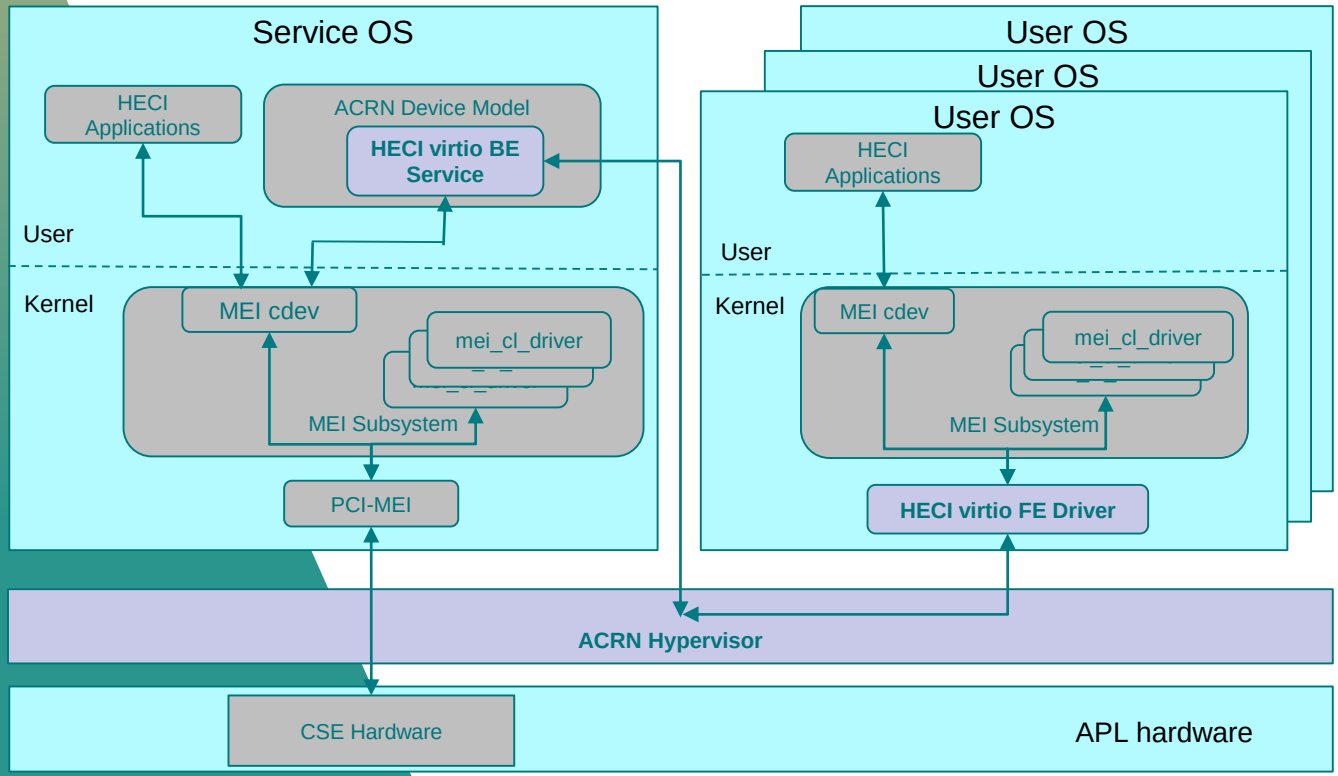
- NOTE: ACRN remains EFI runtime services and boot time services (without interrupt)

SEED Virtualization



- HV gets pSEED from ABL, which retrieves from CSE through HECI.
- Hypervisor implements Key derivation function (KDF) to generate child seeds (vSEED) per request
- HMAC-SHA256 for Android VM
- HMAC-SHA512 for Linux VM
- Present the derived vSEED to guest VM. Each guest cannot see/derive the other guest's vSEED.

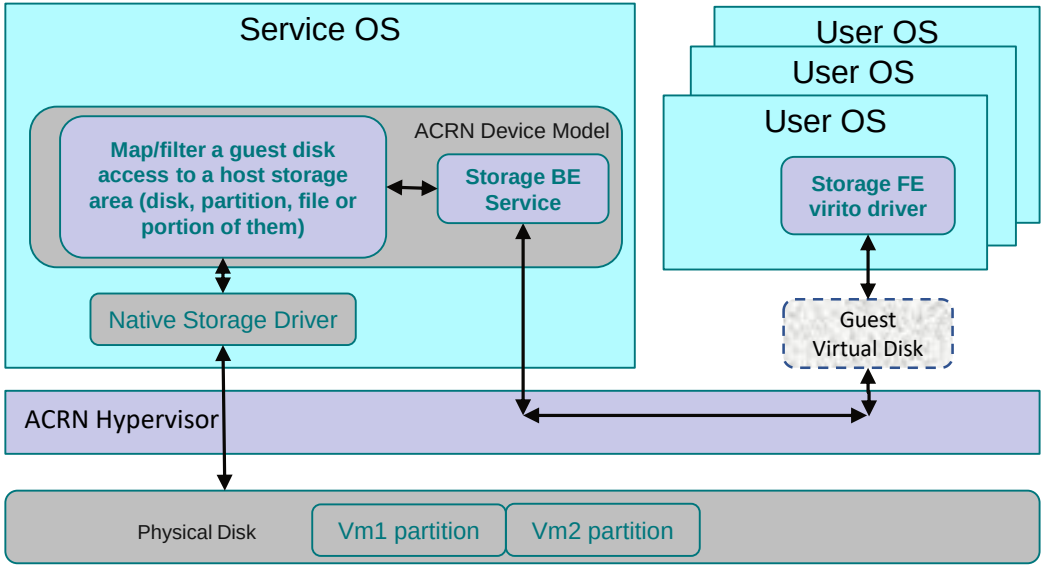
HECI (Host Embedded Controller Interface)



- HECI emulator implements a virtio PCIe device to support multiple User OS.
- HECI BE will communicate with HECI FE driver to send & receive the HECI messages.
- HECI client layer protocol will read/write to SOS MEI cdev directly. And HECI bus messages will emulate in the BE.

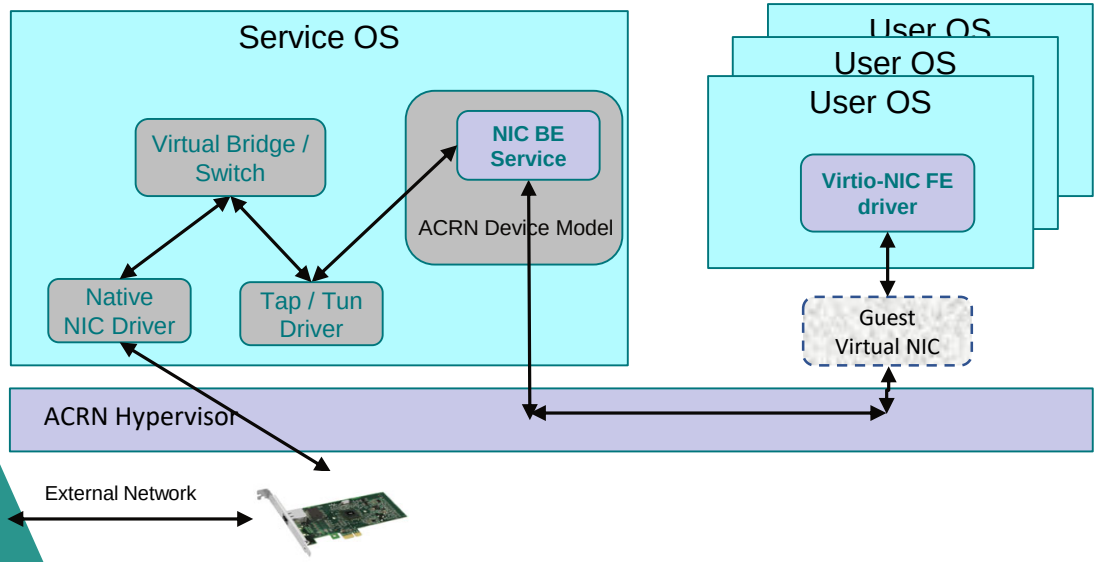
MEI: Intel Management Engine Interface Linux driver; mei_cl_driver: mei client driver

Storage Virtualization

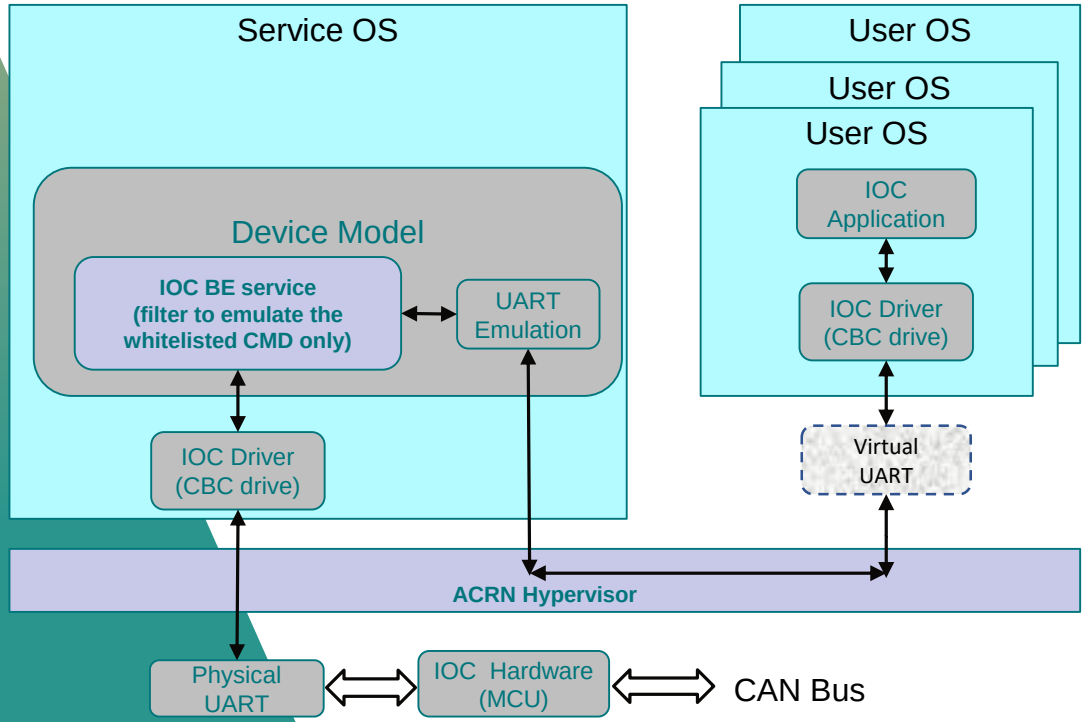


- Map a host storage area (SAR), i.e., disk / partition / file, as a guest disk
- Map a portion of host SAR (start_LBA, size) as a guest disk

Network Virtualization

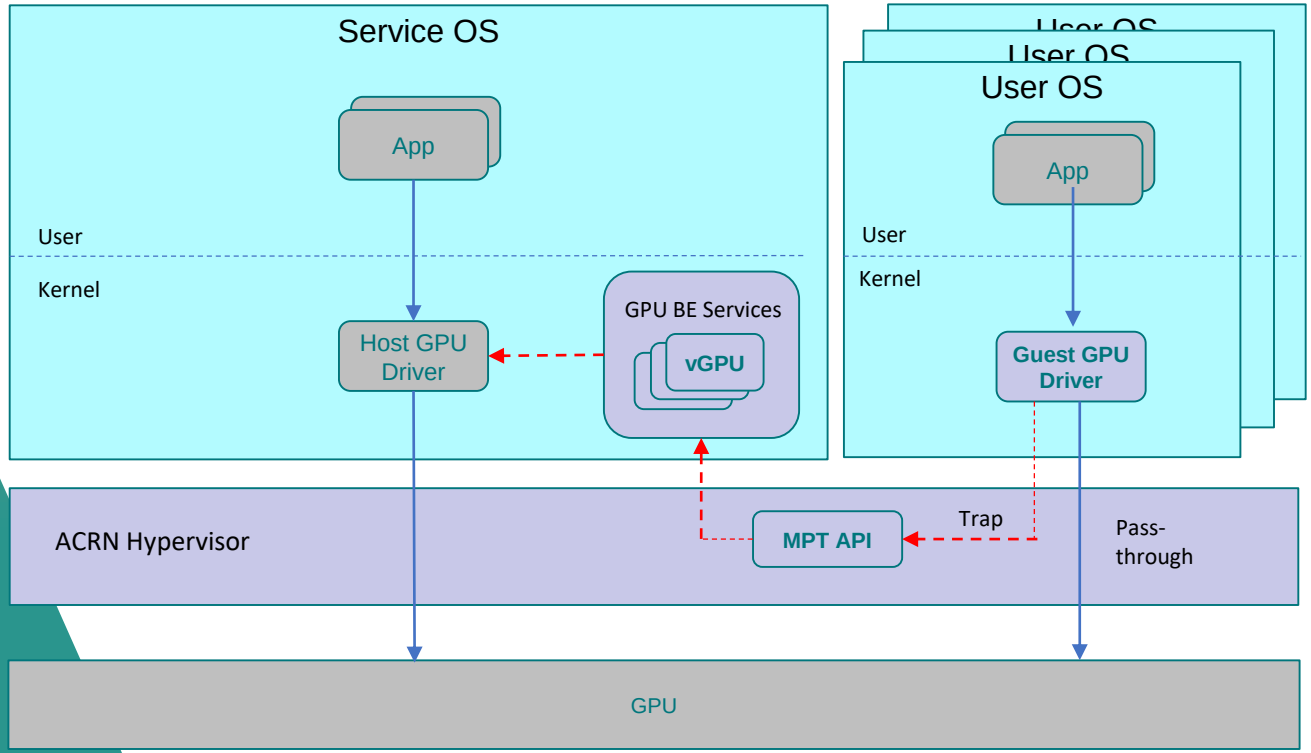


IOC (I/O Controller) Virtualization

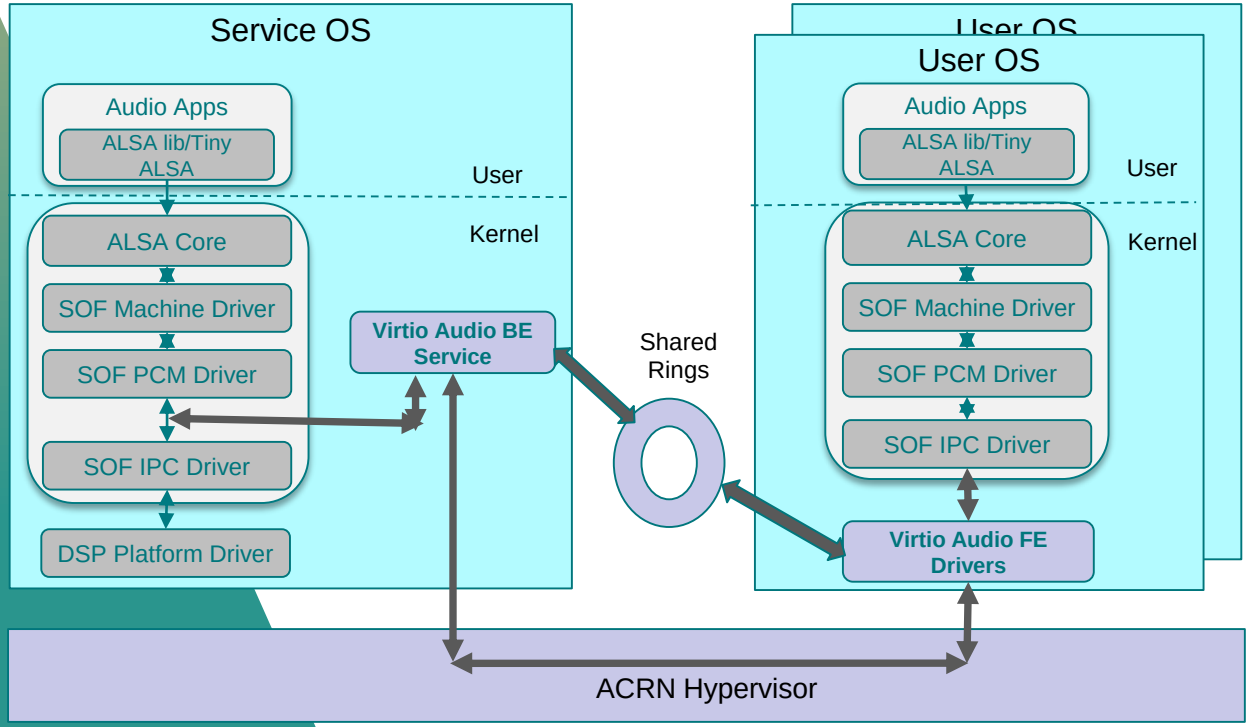


- SOS owns IOC, but UOS may access part features
- Whitelisted CMDs from UOS may be forwarded / emulated
- Support Intel IOC controller only, OEMs may extend

GPU Virtualization



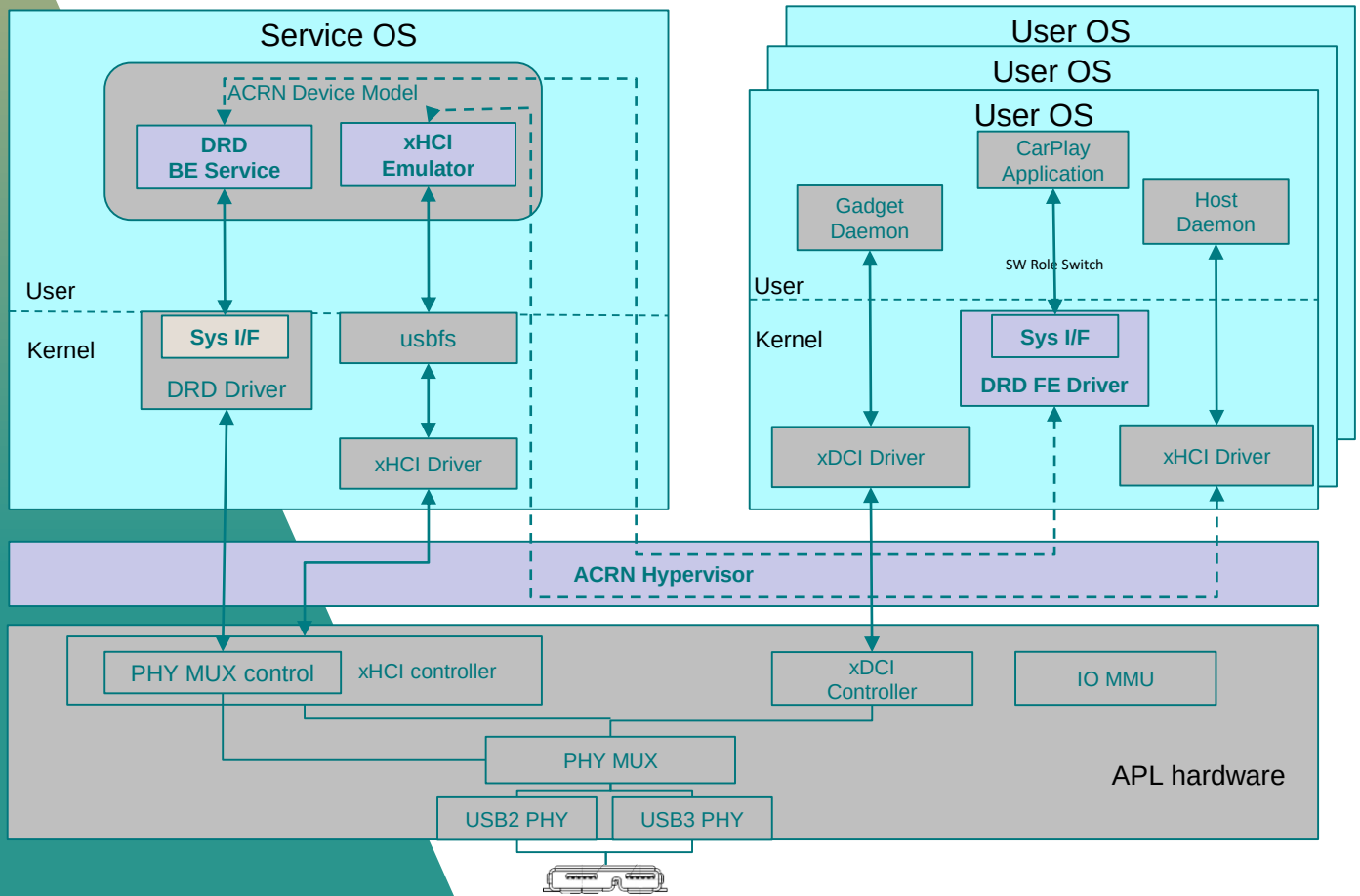
Audio Virtualization



- ALSA (Advanced Linux Sound Architecture) lib - same user API across VMs
- SOF FE driver forwards IPC commands to its counterpart SOF BE service (kernel space) thru virtio shared rings
- The commands carry the address of audio data (not data)
- Service OS can directly access the memory of User OS
- FE driver communicate with IPC driver thru ops callback of platform driver
- BE service communicate with IPC driver thru IPC TX/RX interface of IPC driver

*SOF: Sound Open Firmware; PCM: Pulse-code modulation; IPC: Inter-Processor Communication

USB Virtualization



- xHCI emulator provides multiple instances of virtual xHCI controller to share among multiple User Oss, each USB port can be dedicatedly assigned to a VM.
- xDCI controller can be passed through to the specific user OS with I/O MMU assistance.
- DRD BE service emulate the PHY MUX control logic. And DRD FE driver provide sysfs interface to user space of user OS to switch DCI/HCI role in CarPlay SW.

Call for Participation

https://projectacrn.github.io/index.html

Joining ACRN Community Today!!!