Guest-side changes for confidential guests in Android

Will Deacon <will@kernel.org>

Android Systems, Google

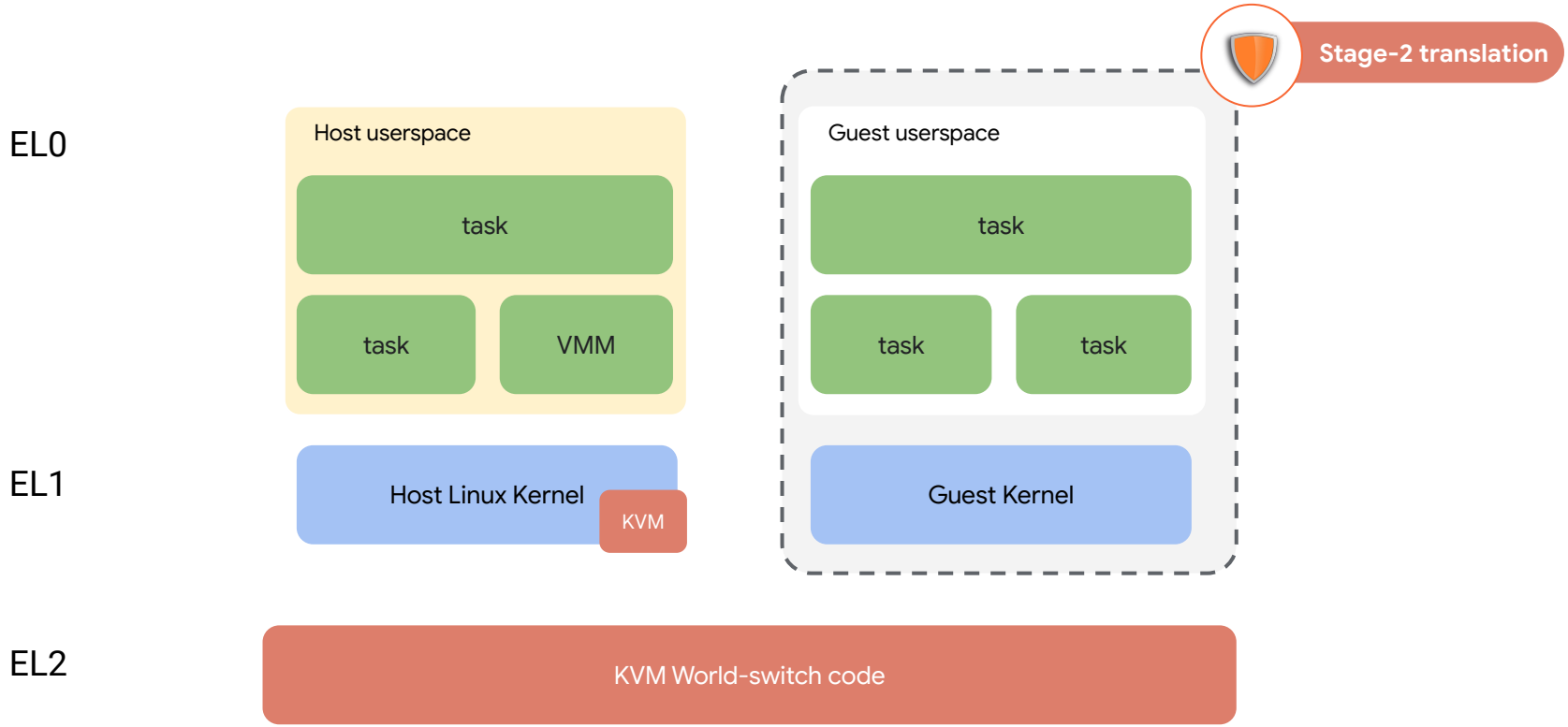
KVM Forum 2024, Brno

Android

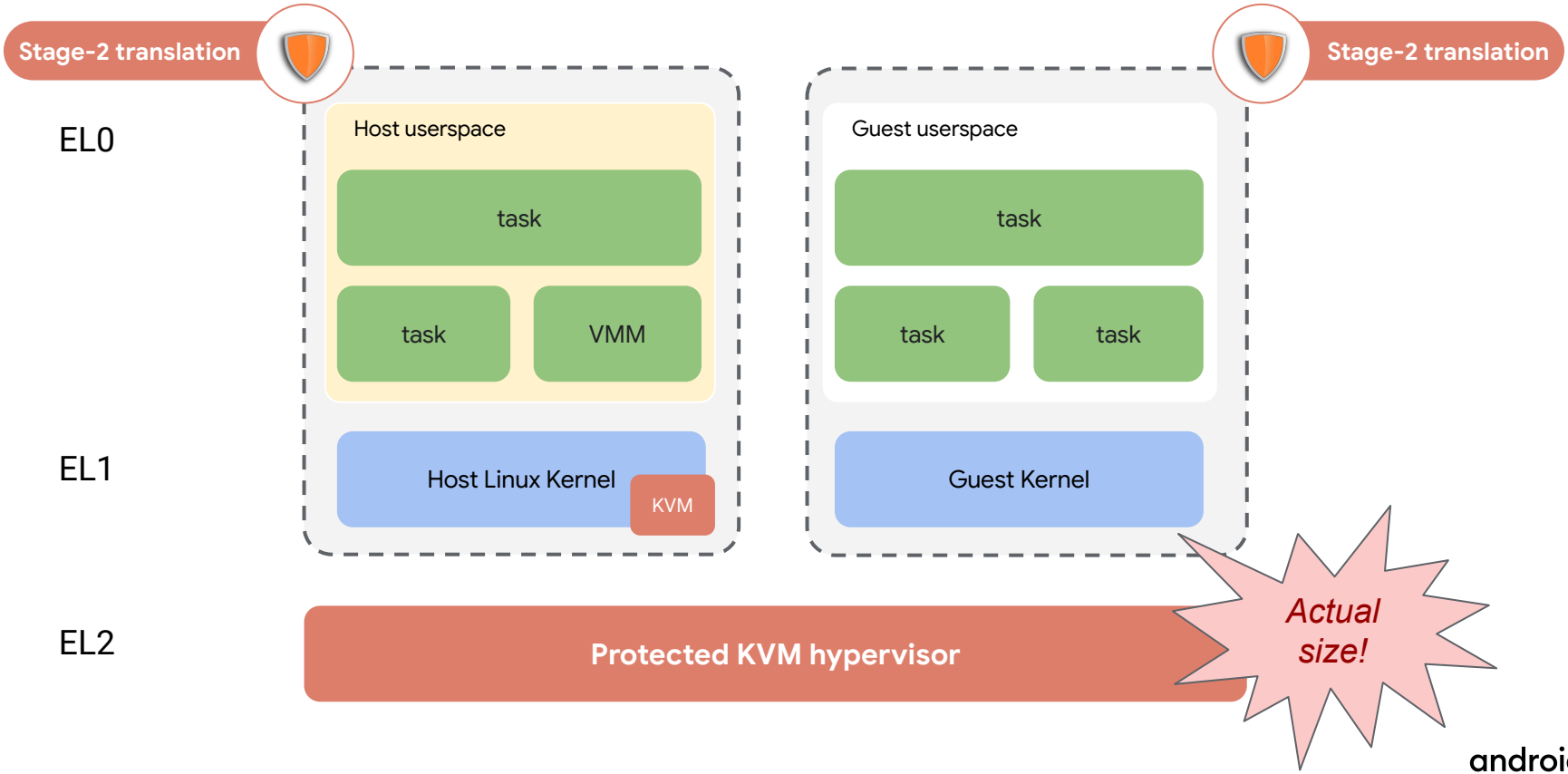
$ whoami

- Upstream kernel hacker
- Arm64 & IOMMU co-maintainer
- Android systems team at Google
- pKVM developer
- Homebrewer
- I’d rather be fishing

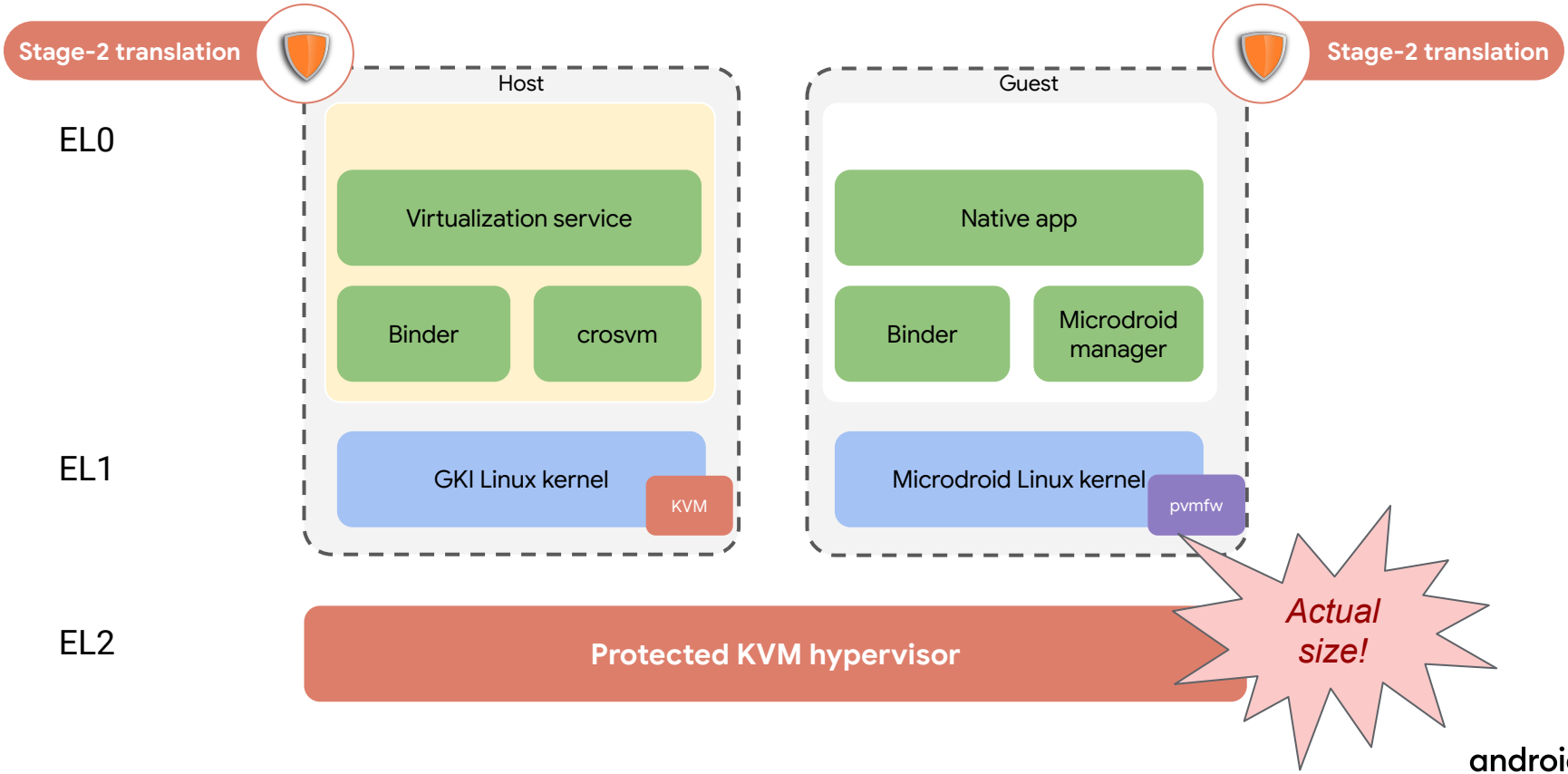
KVM port on armv8.0A (nVHE)



pKVM overview



pKVM in Android



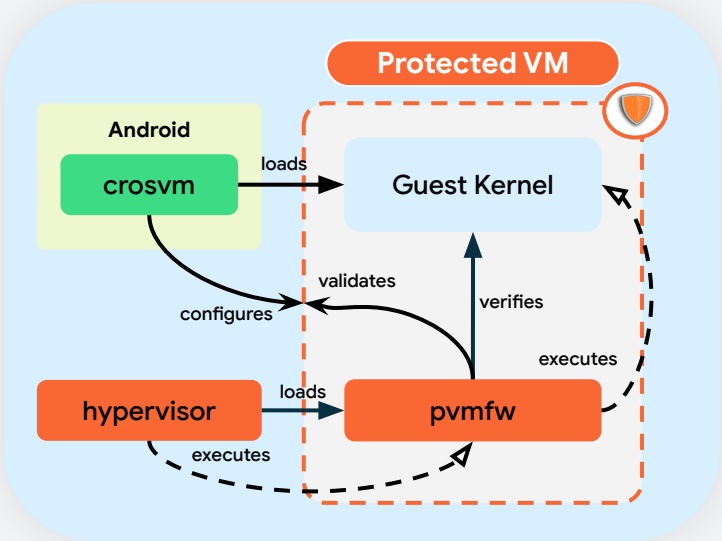
01

Protected boot

Without trusting the host

Protected VM boot

- All secondary vCPUs are initially “off” (per PSCI)
- `KVM_SET_ONE_REG` limited for the primary vCPU
  - Only `X0-X14`
- Memory initialised directly by crosvm
  - Page donated to the guest on stage-2 fault
- Hypervisor initialises PC of primary vCPU to enter PVMFW
  - Which is then loaded on demand from EL2
- New capability for userspace to configure the pvmfw load address:
  - `KVM_CAP_ARM_PROTECTED_VM_FLAGS_SET_FW_IPA`
  - `KVM_CAP_ARM_PROTECTED_VM_FLAGS_INFO`
- PVMFW is written in Rust and open-source:
  - https://android.googlesource.com/platform/packages/modules/Virtualization/+/HEAD/guest/pvmfw/

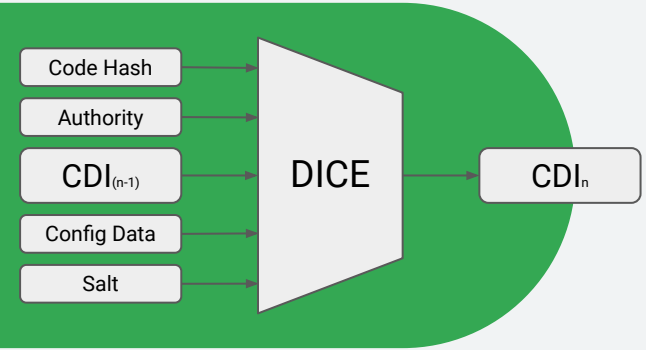


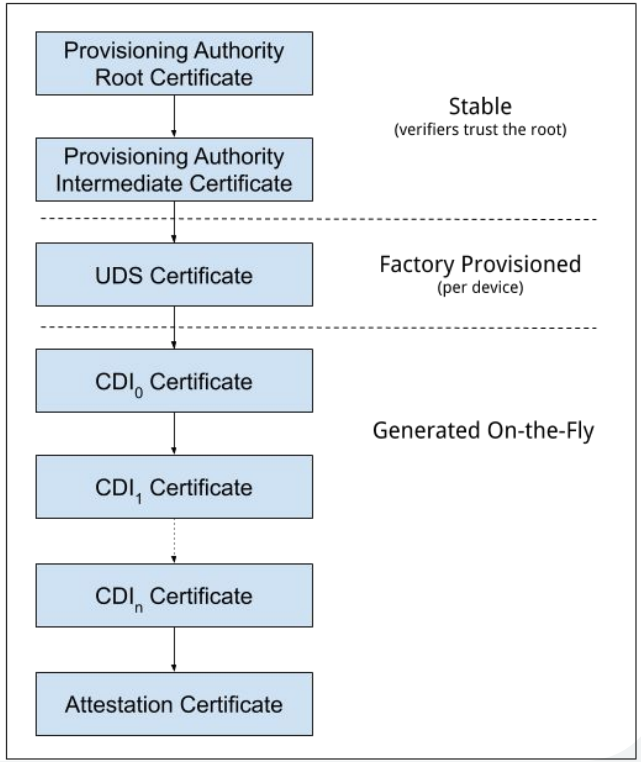
PVMFW & DICE

DICE chains attest of a boot sequence and provide each stage with a **certificate** and a **private key** (CDI),
Each stage adds a layer to the chain through cryptographic operations:

The reserved memory loaded into the pVM by the hypervisor contains a **DICE chain**, which pvmfw extends with the measurements from its **verified boot**, producing a new chain that the guest can make use of to:

- Perform cryptographic operations with a key that was protected from the host
- Attest of its identity to external entities
- Derive new chains for its payloads (*e.g.* user-space applications)





https://embedded-recipes.org/2023/schedule/pocket-sized-virtual-machines/

02

Hypercalls

Services for protected guests
(or, ABIs are hard)

Discovery

Hypercalls are allocated in the *“Vendor Specific Hypervisor Service Call”* range of the Arm SMCCC specification using KVM’s UID.

- `ARM_SMCCC_KVM_FUNC_FEATURES`
  - Query hypercall availability
- `ARM_SMCCC_KVM_FUNC_HYP_MEMINFO`
  - Stage-2 page size
- `ARM_SMCCC_KVM_FUNC_MMIO_GUARD_INFO`
  - **Err, Stage-2 page size**

``ARM_SMCCC_KVM_FUNC_HYP_MEMINFO``

Query the memory protection parameters for a protected virtual machine.

Presence:	Optional; protected guests only.		
Calling convention:	HVC64		
Function ID:	(uint32)	0xC6000002	
Arguments:	(uint64)	R1	Reserved / Must be zero
	(uint64)	R2	Reserved / Must be zero
	(uint64)	R3	Reserved / Must be zero
Return Values:	(int64)	R0	``INVALID_PARAMETER (-3)`` on error, else memory protection granule in bytes
	(int64)	R1	``KVM_FUNC_HAS_RANGE (1)`` if MEM_SHARE and MEM_UNSHARE take a range argument.

Memory

When a page is initially mapped (*“donated”*) into a protected guest, it is made inaccessible to the host.

- `ARM_SMCCC_KVM_FUNC_MEM_SHARE`
  - Share a page RWX with the host
- `ARM_SMCCC_KVM_FUNC_MEM_UNSHARE`
  - Remove host access to a previously shared page. **Never used.**
- `ARM_SMCCC_KVM_FUNC_MEM_RELINQUISH`
  - Unmap a page from the guest and return ownership to the host

```
``ARM_SMCCC_KVM_FUNC_MEM_SHARE``
--------------------------------

Share a region of memory with the KVM host, granting it read, write and execute
permissions. The size of the region is equal to the memory protection granule
advertised by ``ARM_SMCCC_KVM_FUNC_HYP_MEMINFO`` times the number of granules
set in R2. See the ``KVM_FUNC_HAS_RANGE`` paragraph for more details about this
argument.

+---------------------+----------+----+-----------------------------------------+
| Presence:           | Optional; protected guests only.                        |
+---------------------+----------+----+-----------------------------------------+
| Calling convention: | HVC64                                                   |
+---------------------+----------+----+-----------------------------------------+
| Function ID:        | (uint32) | 0xC6000003                                   |
+---------------------+----------+----+-----------------------------------------+
| Arguments:          | (uint64) | R1 | Base IPA of memory region to share      |
|                     +----------+----+-----------------------------------------+
|                     | (uint64) | R2 | Number of granules to share             |
|                     +----------+----+-----------------------------------------+
|                     | (uint64) | R3 | Reserved / Must be zero                 |
+---------------------+----------+----+-----------------------------------------+
| Return Values:      | (int64)  | R0 | ``SUCCESS (0)``                         |
|                     |          |    +-----------------------------------------+
|                     |          |    | ``INVALID_PARAMETER (-3)``              |
|                     +----------+----+-----------------------------------------+
|                     | (uint64) | R1 | Number of shared granules               |
+---------------------+----------+----+-----------------------------------------+
```

MMIO

By default, MMIO exits cannot be handled by the VMM:

- No syndrome information in the host
- No mechanism to set the destination register
- No mechanism to skip the faulting instruction

```
``ARM_SMCCC_KVM_FUNC_MMIO_GUARD``
---------------------------------

Request that a given memory region is handled as MMIO by the hypervisor,
allowing accesses to this region to be emulated by the KVM host. The size of the
region is equal to the memory protection granule advertised by
``ARM_SMCCC_KVM_FUNC_HYP_MEMINFO``.

+---------------------+----------+----+---------------------------------------------+
| Presence:           | Optional; pKVM protected guests only.                       |
+---------------------+----------+----+---------------------------------------------+
| Calling convention: | HVC64                                                       |
+---------------------+----------+----+---------------------------------------------+
| Function ID:        | (uint32) | 0xC6000007                                       |
+---------------------+----------+----+---------------------------------------------+
| Arguments:          | (uint64) | R1 | Base IPA of MMIO memory region              |
|                     +----------+----+---------------------------------------------+
|                     | (uint64) | R2 | Reserved / Must be zero                     |
|                     +----------+----+---------------------------------------------+
|                     | (uint64) | R3 | Reserved / Must be zero                     |
+---------------------+----------+----+---------------------------------------------+
| Return Values:      | (int64)  | R0 | ``SUCCESS (0)``                             |
|                     |          |    +---------------------------------------------+
|                     |          |    | ``INVALID_PARAMETER (-3)``                  |
+---------------------+----------+----+---------------------------------------------+
```

MMIO

By default, MMIO exits cannot be handled by the VMM:

- No syndrome information
- No mechanism to set the destination register
- No mechanism to skip the faulting instruction

- `ARM_SMCCC_KVM_FUNC_MMIO_GUARD_ENROLL`
  - **Not needed with hypervisor auto-enroll!**
- `ARM_SMCCC_KVM_FUNC_MMIO_GUARD_MAP`
- `ARM_SMCCC_KVM_FUNC_MMIO_RGUARD_MAP`
  - Allow MMIO exits for a given IPA range
- `ARM_SMCCC_KVM_FUNC_MMIO_GUARD_UNMAP`
- `ARM_SMCCC_KVM_FUNC_MMIO_RGUARD_UNMAP`
  - Prevent MMIO exits for a given IPA range. **Shouldn't be needed!**

``ARM_SMCCC_KVM_FUNC_MMIO_GUARD``

Request that a given memory region is handled as MMIO by the hypervisor,
allowing accesses to this region to be emulated by the KVM host. The size of the
region is equal to the memory protection granule advertised by
``ARM_SMCCC_KVM_FUNC_HYP_MEMINFO``.

Presence:	Optional; pKVM protected guests only.		
Calling convention:	HVC64		
Function ID:	(uint32)	0xC6000007	
Arguments:	(uint64)	R1	Base IPA of MMIO memory region
	(uint64)	R2	Reserved / Must be zero
	(uint64)	R3	Reserved / Must be zero
Return Values:	(int64)	R0	``SUCCESS (0)``
			``INVALID_PARAMETER (-3)``

TRNG (SMC)

There's not a lot of entropy in a protected VM....

- We cannot trust the host to provide entropy
- We'd like to keep EL2 simple and non-blocking
- `ARM_SMCCC_TRNG_*`
    - Pass these through directly to TZ
    - *Rate-limiting a concern?*

```
/*
 * Forward TRNG calls to EL3, as we can't trust the host to handle
 * these for us.
 */
switch (fn) {
case ARM_SMCCC_TRNG_FEATURES:
case ARM_SMCCC_TRNG_RND32:
case ARM_SMCCC_TRNG_RND64:
        arg1 = smccc_get_arg1(vcpu);
        fallthrough;
case ARM_SMCCC_TRNG_VERSION:
case ARM_SMCCC_TRNG_GET_UUID:
        arm_smccc_1_1_smc(fn, arg1, &res);
        smccc_set_retval(vcpu, res.a0, res.a1, res.a2, res.a3);
        memzero_explicit(&res, sizeof(res));
        break;
}
```

03

I/O

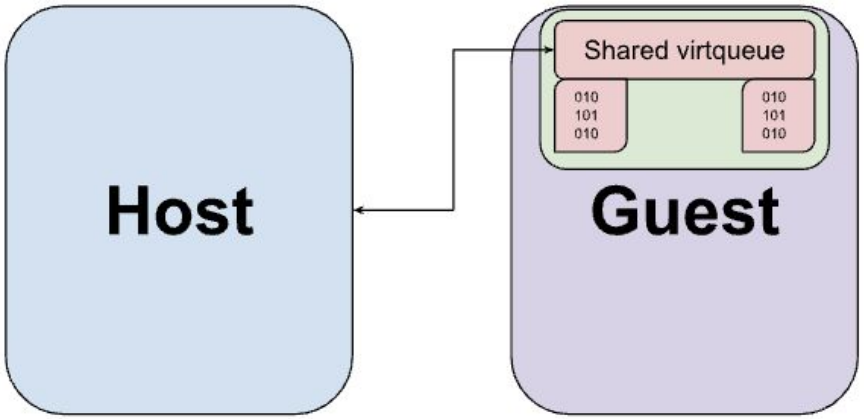
Bouncing via a shared window

Restricted DMA

Per-device SWIOTLB buffers

- Not all I/O data resides in nicely sized/aligned buffers for the stage-2 page size, so we have to bounce using SWIOTLB
- The infamous `VIRTIO_F_ACCESS_PLATFORM` feature in combination with restricted DMA buffers does the trick

```
restricted_dma_reserved {
      compatible = "restricted-dma-pool";
      size = <0x00 0xe00000>;
      alignment = <0x00 0x1000>;
};
pci {
      compatible = "pci-host-cam-generic";
      memory-region = <&restricted_dma_reserved>;
            ...
```



Protected virtio: guest data is copied through a shared window

earlycon=

Early console is accessed long before hypercalls have probed

- Debuggable VMs can be provisioned on some devices
    - Changes the VM identity
    - Enables additional logging and introspection utilities in the guest

- PVMFW pre-registers the 8250 UART with `MMIO_GUARD` for these VMs

04

Upstream

State in Linus’ tree

Upstreaming status

Guest-side changes

- Restricted DMA ✅
- Open DICE misc device ✅
- KVM hypercall discovery ✅
- Memory sharing
    - `set_memory_{de,en}crypted()` ✅
    - GIC ITS 🟡
- MMIO guard
    - `ioremap()` hooks ✅
- Optional drivers
    - Vcpufreq 🟡
    - Per-cpu stall detector ✅
    - Enlightened virtio balloon ❌

Hooks in `arch/arm64/` designed to be reused by the CCA support

`drivers/virt/coco/pkvm-guest/`

Android

Unleash the demo.

Wish me luck...

Thank you