Virtio and the Chamber of Secrets

Interface design for Confidential Computing Systems

Michael S. Tsirkin

Distinguished Engineer
Chair of Virtio TC

Fall 2024

Unicorns by stable diffusion

Red Hat

OASIS

Agenda

- How are Confidential Computing and Virtio related
- Untrusted Virtio
  - Status and Issues
- Trusted Virtio
  - Status and Issues
- Migration
- Summary

Confidential Computing / VMs

- Reduce VM's trust in the hypervisor.

Can protect or migitate	No mitigation
1)Code execution 2)Rollback attacks 3)Information leaks 4)(some) physical access	DoS

- In fact, DoS is the main mitigation measure
- Virtio is used **heavily**
- Requirement: avoid introducing more trust in HV

Red Hat

Standard Virtio device models

Software (virtio, vhost, vhost-user)	Hardware (Passthrough with VFIO)	Mixed (vdpa)

device is accessible to HV
- access directly
- trap and emulate

Should not be trusted

Device driver MUST protect the guest VM.

Protecting guest: DMA

- SWIOTLB (lookaside buffer):
  limited memory accessible to untrusted devices
- Driver (through DMA API) copies data to/from guest memory
- Protects against TOCTOU
  - (basic) support since Virtio 1.0
  - Allocation/copy overhead

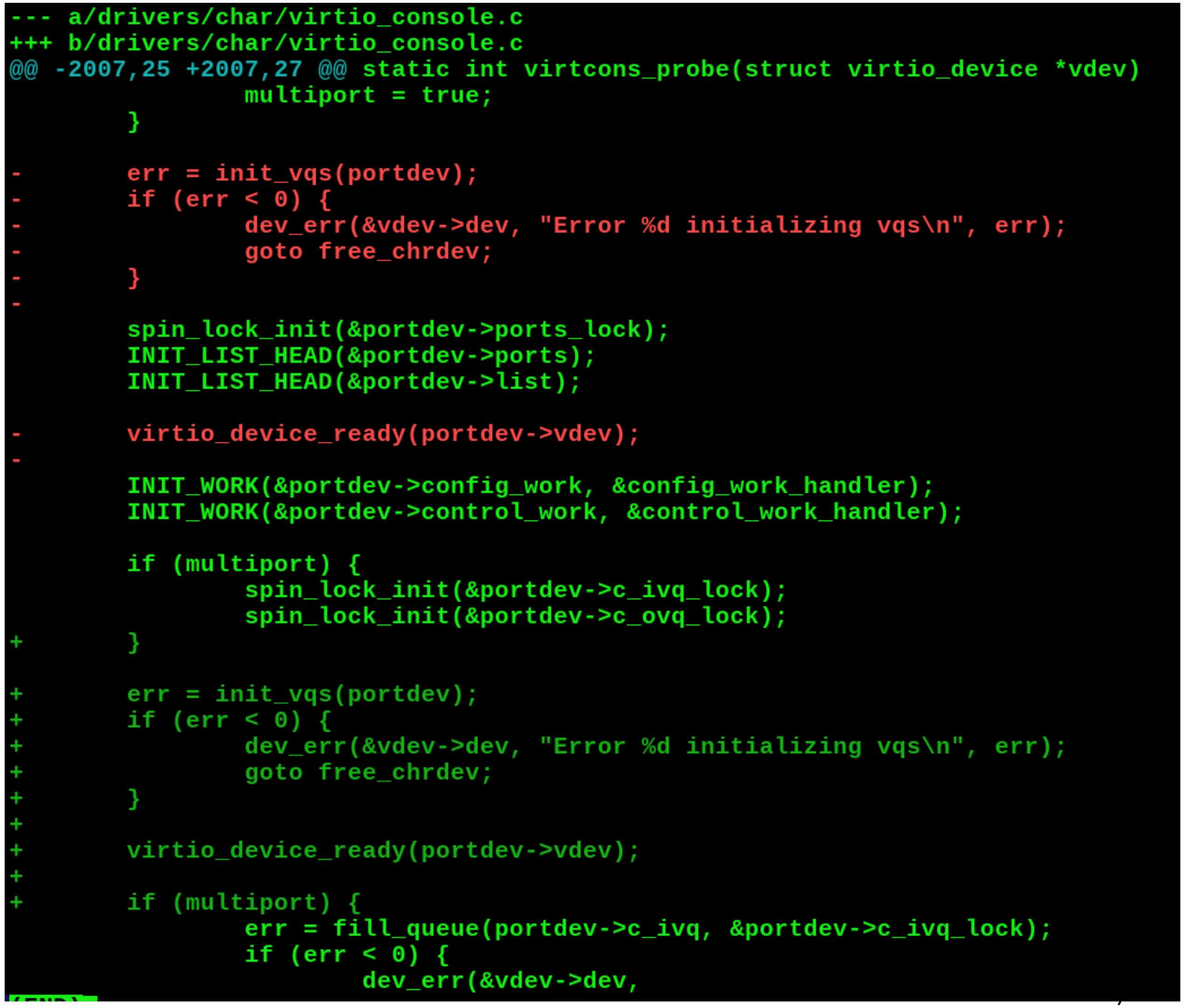
Red Hat

Input validation

- Constant vigilance required
- Virtio drivers fuzzed
  - Net
  - Block
  - Console
  - 9P
  - Vsock
- Transient execution: Spectre v1?

Red Hat

Initialization and cleanup bugs



```
--- a/drivers/char/virtio_console.c
+++ b/drivers/char/virtio_console.c
@@ -2007,25 +2007,27 @@ static int virtcons_probe(struct virtio_device *vdev)
                multiport = true;
        }

-       err = init_vqs(portdev);
-       if (err < 0) {
-               dev_err(&vdev->dev, "Error %d initializing vqs\n", err);
-               goto free_chrdev;
-       }
-
        spin_lock_init(&portdev->ports_lock);
        INIT_LIST_HEAD(&portdev->ports);
        INIT_LIST_HEAD(&portdev->list);

-       virtio_device_ready(portdev->vdev);
-
        INIT_WORK(&portdev->config_work, &config_work_handler);
        INIT_WORK(&portdev->control_work, &control_work_handler);

        if (multiport) {
                spin_lock_init(&portdev->c_ivq_lock);
                spin_lock_init(&portdev->c_ovq_lock);
+       }

+       err = init_vqs(portdev);
+       if (err < 0) {
+               dev_err(&vdev->dev, "Error %d initializing vqs\n", err);
+               goto free_chrdev;
+       }
+
+       virtio_device_ready(portdev->vdev);
+
+       if (multiport) {
                err = fill_queue(portdev->c_ivq, &portdev->c_ivq_lock);
                if (err < 0) {
                        dev_err(&vdev->dev,
```

Stack/application level protection

- TLS
- Dmcrypt
- Dmintegrity
- Rollback protection?

Red Hat

More devices

- Virtio-input (since we have console)
- Virtio-scsi (since we have blk)
- Virtio-snd (why not?)
- Virtio-rng (needed?)

Possible?

- Virtio-fs ?
- Virtio-crypto ?
- Virtio-pmem?
- Virtio-balloon ? Could be useful.

Audit/Fuzzing challenges

- + __iomem
- + dma_addr_t
- - dma_sync

Note: unlike __user

Filtering

- Device filter
  - Guest decides which drivers to allow
  - If not allowed, probe does not run
- Features
  - Virtio has a lot of flexibility, reducing attack surface is desired
  - Limit the supported features, configurations?

restore trust in devices

- Bring device into TCB
- MUST NOT be accessible to HV
- For PCI devices - TDISP

TDISP in action

- TEE Device Interface Security Protocol
- End to end encryption of guest to device communication
- Designed to protect against many types of software and physical attacks

Locking

- HV is still responsible for device discovery, some setup (e.g. scan/sriov) and allocation to guests
- To assigned device to guest, it has to be locked
- Can not be changed by HV while locked

Measurement report

- DEVICE_INTERFACE_REPORT
- Signed by device
- E.g. MMIO_RANGES
- Can include device specific info

TDISP limitations

- 3 main ways to access a PCI device:

IO R/W	Memory R/W	Config R/W

- Only memory encrypted
- HV can trap and emulate IO/Config
- Insecure

Red Hat

PCI Config uses in Virtio

- RO – helps driver locate registers
  - Common cfg / device cfg / vq notification / ISR / shared memory
- RW – gateway for 32-bit firmware if memory is > 4G
  - Slow

Red Hat

Using measurement for RO config

- Arguably a bug that the TDISP spec does not include this
- Add ranges or RO registers to the report
- Alternatively, add to device specific area in the report
- We then need to define format for this area – worth it?

Red Hat

Using lock to protect config

- Disable RW registers upon lock
- Give up on 32 bit / high memory support

Avoid PCI Config

- Relocate to a known offset in PCI Memory
- Possibly verbatim or with consmetic changes, to minimize driver work

- Compatiblity: detect TDISP? Unattractive
- Or, allocate new device IDs

Red Hat

VDPA

- VDPA: a mixed device
  - Data path – passthrough
  - Control path - emulated
- Popular due to hardware simplicity
- What does control path include:
  - Programming queues (size/address)
  - Reset
  - Features, etc

Red Hat

VDPA vs TDISP

- Does not seem practical
- HV can redirect DMA arbitrarily
- Confuse guest by lying about features
- Or device config
- Include in DEVICE_INTERFACE_REPORT / DEVICE_SPECIFIC INFO?
- Practical?
- VDPA can not tweak. Negates benefits?

Red Hat

VFIO/virtio

- VDPA-like trick to implement a transitional device over a modern device
- VIRTIO_ADMIN_CMD_LEGACY:
  - Exposes direct access to VF’s IO memory through PF
- MUST be disabled upon interface lock

Red Hat

VM Migration

- Moving state between devices: SRC, DST
- By the HV
- But how do we prevent HV attacks?

Migration: untrusted Virtio

- HV saves state from SRC and restore on DST
- can corrupt the state
  - but then it can, anyway
- Guest must validate at all times

Memory tracking: untrusted Virtio

- Device can change memory as it is migrated
- HV can track changes (e.g. shadow VQ) and re-copy
- Can corrupt memory
  - But it is public, so it can anyway
- Guest must copy and validate at all times

Migration: TDISP Virtio

- Can not trust HV
- On SRC device saves state in encrypted and signed form
- On DST device checks the signature and restores the state
- A bit vague

Memory tracking: TDISP Virtio

- Device tracks memory changes
- Signals the HV to retransmit
- Leaks which memory pages are accessed
- Rollback protection?

Red Hat

Summary

- Many improvements possible
- Non-trusted Virtio - driver work
- Trusted Virtio – spec work

Red Hat

Questions? New Virtio MLs

- Virtio-comment@lists.linux.dev - driver/device devel
- Virtio-dev@lists.linux.dev - spec development

- Courtesy of Linux Foundation

Red Hat