Securing Linux VM boot with AMD SEV measurement

Dov Murik & Hubertus Franke

IBM

Work of many people

- Colleagues from IBM
  - Tobin Feldman-Fitzthum, James Bottomley, Jim Cadden
- edk2/OVMF community
- QEMU community

Confidential Computing setting

- Goal: Protect the guest from the hypervisor
- Cloud Service Provider (untrusted)
- Host machine (untrusted)
- Guest Owner
- Guest VM
- Sensitive guest computing workload
- Encrypted memory

The problem

- Memory encryption is not enough
- Guest Owner has no idea what's running in the guest
  - Need to verify that the desired workload is indeed running in the guest

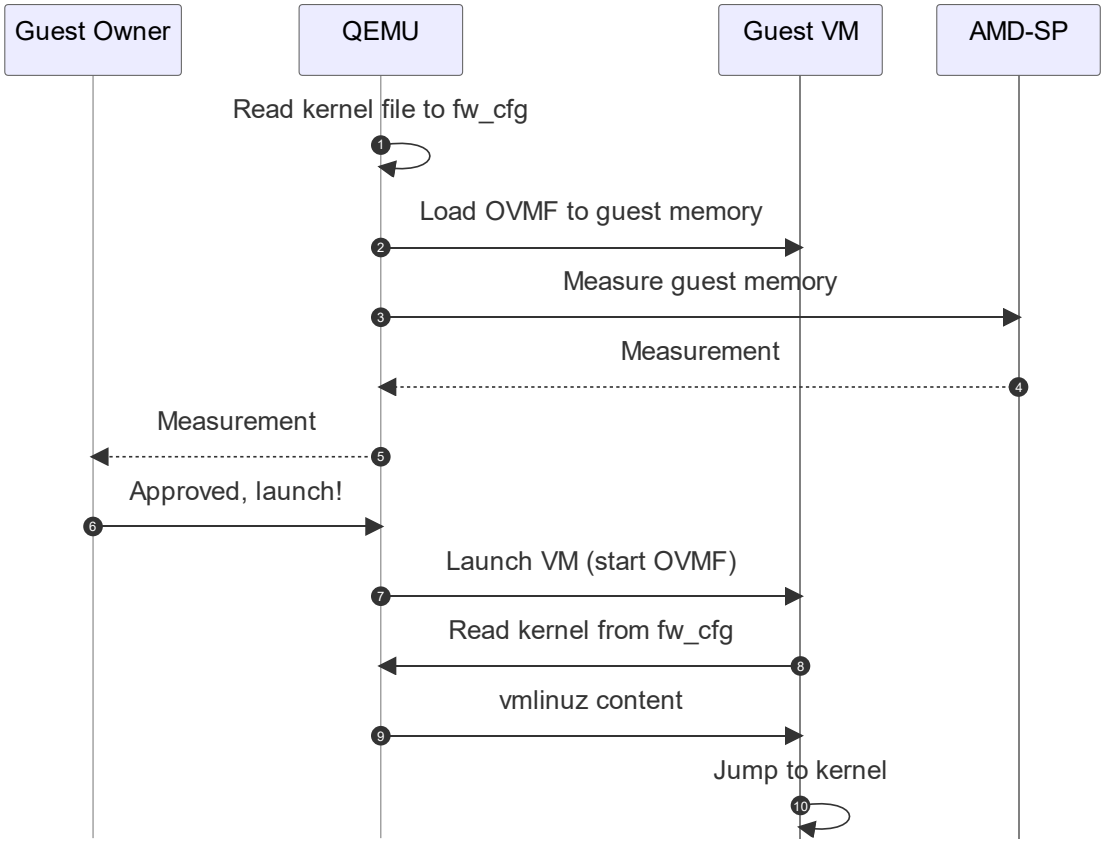
AMD SEV

- AMD-SP (Secure Processor) hardware
  - Also called PSP (Platform Secure Processor)
- VM memory is encrypted
- Guest launch measurement
  - Hash of initial guest memory before VM starts
  - Signed by AMD-SP
- Guest secret injection
  - Only at launch, immediately after verifying the measurement

VM boot process with -kernel

- Example QEMU command line:
  - `qemu-system-x86_64 -kernel vmlinuz-5.13.0 \`
    `-initrd initrd.img-5.13.0`
- QEMU reads these files to a fw_cfg “device”
- QEMU loads OVMF into guest memory
- SEV measures memory
- Guest owner approves → Launch!
- Jumps to OVMF
- OVMF reads kernel / initrd / cmdline from fw_cfg
- Loads it into memory
- Jumps to kernel

VM boot process with -kernel



Host attack on boot with -kernel

- Host runs:
  - `qemu-system-x86_64 -kernel` **`malicious-5.13.0`** …
- QEMU loads a malicious guest kernel
- QEMU loads OVMF into guest memory
- SEV measures memory
- Guest owner **approves** → Launch!
- Jumps to OVMF
- OVMF reads the **malicious kernel** from fw_cfg
- Loads it into memory
- Jumps to malicious kernel

Vulnerability

- AMD-SP hardware measured OVMF
- … but **didn’t measure** kernel / initrd / cmdline
  - (as they are not part of the initial VM memory)

Solution

- “Extend the measurement”
- Add a list of hashes (of kernel / initrd / cmdline) to the initial guest memory
- AMD-SP will measure OVMF + list of hashes
- OVMF will verify hashes when loading kernel / initrd / cmdline from fw_cfg

Hashes GUIDed table

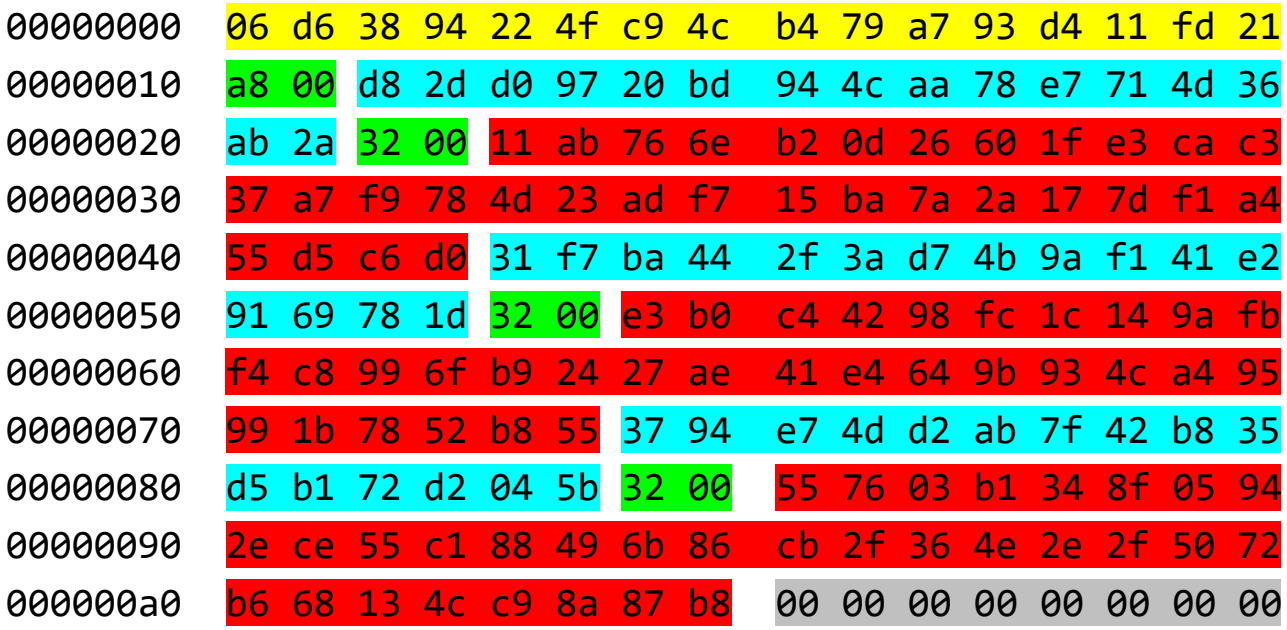


Table header GUID

Length field

Entry GUID

Entry SHA256 hash

Padding

Solution details

- QEMU loads OVMF into guest memory
- QEMU loads **hashes of kernel+initrd+cmdline** into guest memory
- SEV measures all guest memory
- Guest owner approves → Launch!
- Jumps to OVMF
- OVMF reads the kernel from fw_cfg
- OVMF **verifies kernel** against the expected hash
  - Same for initrd and kernel command-line
- Loads it into memory
- Jumps to kernel

Attack mitigation

- Host uses wrong kernel / initrd / cmdline
  - Measurement won't match
- Host replaces OVMF with own version which doesn't verify the hashes
  - Measurement won't match
- Host fills expected hashes but passes wrong content via fw_cfg
  - Measurement OK, but OVMF will refuse to load the content because it doesn't match the expected hash

Caveat

- kernel/initrd/cmdline are readable by the (untrusted) host
  - as is OVMF now
- Only use when the kernel+initrd are not confidential
- Alternatively: use encrypted disk boot
  - KVM Forum 2021 talk: Encrypted Virtual Machine Images for Confidential Computing (James Bottomley, IBM & Brijesh Singh, AMD)

Implementation status

- OVMF part
  - Designate memory area for hashes list
  - Verify fw_cfg blobs against the hashes list
  - Status: Merged to master in July 2021
- QEMU part
  - Calculate hashes and populate the OVMF designated memory area
  - Status: Reviewed; expected in v6.2

Accessing injected secrets

- Once we have a properly measured guest, Guest Owner can inject secrets (secure channel)
- OVMF and QEMU already support that
- But there’s no easy way to access them in the guest userland
- We proposed an sev_secret kernel module which exposes the injected secrets in a securityfs dir
  - Ongoing discussion (linux-coco mailing list)

sev_secret module usage

```
# modprobe sev_secret
# ls -l /sys/kernel/security/coco/sev_secret
-r--r----- 1 root root 0 Jun 28 11:54 736870e5-84f0-4973-92ec-06879ce3da0b
-r--r----- 1 root root 0 Jun 28 11:54 83c83f7f-1356-4975-8b7e-d3a0b54312c6
-r--r----- 1 root root 0 Jun 28 11:54 9553f55d-3da2-43ee-ab5d-ff17f78864d2
-r--r----- 1 root root 0 Jun 28 11:54 e6f5a162-d67f-4750-a67c-5d065f2a9910

# xxd /sys/kernel/security/coco/sev_secret/e6f5a162-d67f-4750-a67c-5d065f2a9910
00000000: 7468 6573 652d 6172 652d 7468 652d 6b61  these-are-the-ka
00000010: 7461 2d73 6563 7265 7473 0001 0203 0405  ta-secrets......
00000020: 0607                                     ..

# rm /sys/kernel/security/coco/sev_secret/
                              e6f5a162-d67f-4750-a67c-5d065f2a9910
```

(wipes secret from memory)

Future plans

- Improve Guest Owner’s experience
  - Every change in kernel / initrd / cmdline invalidates the expected measurement
- Adapt this scheme to support newer generations
  - AMD SEV-ES (measure CPU state)
  - AMD SNP
  - Intel TDX

Securing Linux VM boot with AMD SEV measurement

Dov Murik & Hubertus Franke

IBM

KVM FORUM