# TPM support in oVirt

Milan Zamazal <mzamazal@redhat.com>
Tomáš Golembiovský <tgolembi@redhat.com>

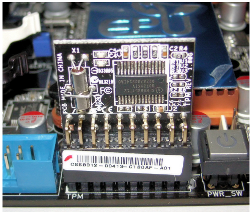
September 14, 2021

## What is Trusted Platform Module (TPM)?

A built-in hardware module with a small amount of secure persistent memory, providing some cryptography functions (random number generator, key generator, hash generator, signatures). It can be used for:

- Checking hardware and software integrity.
- Remote hardware and software attestation.
- Key management and protection for purposes such as disk encryption.
- Restricting users (licensing, DRM, . . . ).

If a TPM device is lost or reset, the stored attestations and keys are lost.



A TPM device

http://commons.wikimedia.org/w/index.php?curid=7637561

## Why TPM support in oVirt?

A TPM device and its associated facilities can be required (BitLocker, Windows 11) or useful (LUKS, securing boot, ...) in the guest OS.

Examples of typical usages:

- Disk encryption (BitLocker, LUKS)
- Securing Windows (boot, HyperV, ...)

## TPM in oVirt

Virtualization:

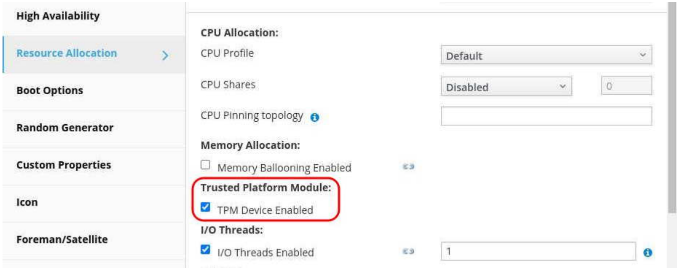
- TPM 2.0
- **Emulated** device only (no hardware support needed); host TPM devices not suitable for use by guests
- Virtual TPM device + associated persistent data
- swtpm — the (external) emulator used by QEMU
- swtpm keeps its data on the host, in `/var/lib/libvirt/`

Requirements:

- **UEFI**
- Cluster level >= **4.6**

# Enabling TPM for a virtual machine (VM)

It can be enabled or disabled (default) for a VM:



High Availability
Resource Allocation
Boot Options
Random Generator
Custom Properties
Icon
Foreman/Satellite

**CPU Allocation:**

| | |
|---|---|
| CPU Profile | Default |
| CPU Shares | Disabled, 0 |
| CPU Pinning topology | |

**Memory Allocation:**

☐ Memory Ballooning Enabled

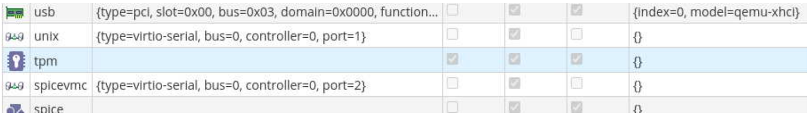
**Trusted Platform Module:**

☑ TPM Device Enabled

**I/O Threads:**

☑ I/O Threads Enabled: 1

New kind of a VM device: TPM



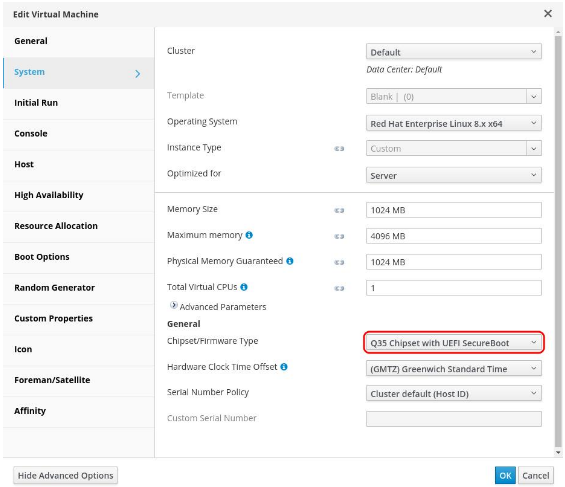
| | | | | | | |
|---|---|---|---|---|---|---|
| usb | {type=pci, slot=0x00, bus=0x03, domain=0x0000, function... | ☐ | ☑ | ☑ | {index=0, model=qemu-xhci} |
| unix | {type=virtio-serial, bus=0, controller=0, port=1} | ☐ | ☑ | ☐ | {} |
| tpm | | ☑ | ☑ | ☑ | {} |
| spicevmc | {type=virtio-serial, bus=0, controller=0, port=2} | ☐ | ☑ | ☐ | {} |
| spice | | ☐ | ☑ | ☑ | {} |

## Secure boot

- UEFI protocol for prevention of loading untrusted code during boot.
- Every software component is first validated against a set of known keys before it is executed.
- 4-5 key types used in the process, some stored in firmware, other in non-volatile memory (NVRAM) of the machine.
- Emulated by QEMU, NVRAM data stored in `/var/lib/libvirt/`.

# Enabling secure boot for a virtual machine

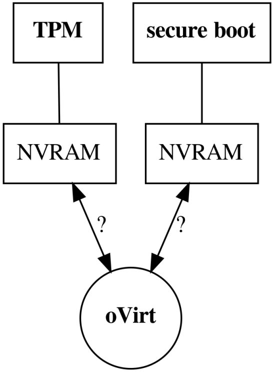
It can be enabled for a VM by selecting the corresponding Chipset/Firmware Type:



Cluster: Default
Data Center: Default
Template: Blank | (0)
Operating System: Red Hat Enterprise Linux 8.x x64
Instance Type: Custom
Optimized for: Server
Memory Size: 1024 MB
Maximum memory: 4096 MB
Physical Memory Guaranteed: 1024 MB
Total Virtual CPUs: 1
Advanced Parameters
General
Chipset/Firmware Type: Q35 Chipset with UEFI SecureBoot
Hardware Clock Time Offset: (GMTZ) Greenwich Standard Time
Serial Number Policy: Cluster default (Host ID)
Custom Serial Number:

## What do have TPM and secure boot in common?

**VM external data:** Persistent data in an emulated non-volatile memory, stored outside the VM process and storage.



## Storing VM external data

TPM and secure boot are not very useful without their data being stored persistently. The data is stored in multiple places:

- While a VM is running: Local file system on the host (`/var/lib/libvirt/`); transferred to destination hosts by libvirt on live migrations
- Persistently: The Engine database
- On storage: Snapshots (in OVF), OVA exports, hibernation

TPM data and secure boot NVRAM data are handled the same way but independently to each other.

## Handling stored VM external data

TPM device **removed** → TPM data **deleted** from the database.

Secure boot **disabled** → secure boot data **deleted** from the database.

Notes:

- VMs with freshly enabled TPM or secure boot get default initial external data once they are started and their data is retrieved from the host.
- The same applies for new VMs, unless they are created from a template or an OVA export already containing the external data.
- VM external data can be reset by disabling and enabling TPM or secure boot in the VM.

## VM external data retrieval

**Challenge: Synchronization of the stored data with the guest OS.**

VM external data is updated when a VM goes down. A **stale** rather than up-to-date external data may be stored in case of:

- VM power off from Engine.
- Unreachable or malfunctioning host / Vdsm.

Additionally, external data is retrieved **periodically**, to hopefully still have recent data in the situations above. But it's best effort only, having up-to-date data cannot be guaranteed in those cases.

Unclean shutdown from the guest OS or QEMU crash is OK, unless it causes inconsistency in the guest OS itself.

Data in the emulators can change even if there are no changes in the guest OS, this is not a problem.

## VM external data in snapshots

**Challenge: Storing data as of the same moment the snapshot is taken.**

Inconsistency of the stored external data and the guest OS file system → the guest OS may refuse to start or to work properly.

No problems with offline snapshots, they capture the exact stored state of the VM. But live snapshots are more complicated.

VM external data is stored in OVF in snapshots.

## Live snapshots without memory

**Warning:** A running VM may have **newer** external data than those stored in the Engine database.

Recommendations:

- If TPM or secure boot data has been altered recently, wait at least 2 minutes before making a snapshot, to allow periodic data synchronization to the Engine database and the snapshot.
- If you want to be extra sure, make an additional snapshot before changing TPM or secure boot data, to be able to return back to the previous configuration if something bad happens to the new one.

## Live snapshots with memory

Snapshots with memory are **disabled** for VMs with TPM:

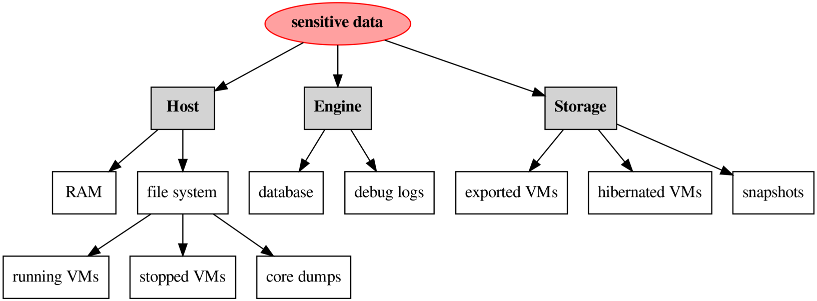
- libvirt currently doesn't provide means to guarantee external data in the snapshot correspond to the contents of RAM.
- Inconsistency of VM external data and RAM may lead to a guest OS crash or other problems after running the VM from the snapshot.

Secure boot doesn't prevent snapshots with memory, but the limitations of live snapshots without memory apply.

## Security

Live data stored in an emulator is by definition exposed more than offline data stored in a specialized secure hardware.

It may be present in many places:



## What can be done about security?

Only **trusted users** should have access to any of the sensitive areas.

Introducing special user permissions for operations putting VM external data on storage (snapshots, exports, hibernation) may be considered in future.

Note: Automatic VM external data encryption is not used. It doesn't help much here beyond security by obscurity – encryption keys must also be stored somewhere, resulting in the same problems and limitations.

## Limitations

- Live snapshots are not guaranteed to have VM external data consistent with the guest OS.
- Snapshots with memory are disabled when TPM is enabled (`https://bugzilla.redhat.com/1855367`).
- VM external data is not stored when exporting a VM to deprecated export domains.
- TPM can be enabled only for RHEL >= 7 and Windows >= 8.1 guest OSes (can be changed in `osinfo`).
- Security of data must be carefully considered.

## Future work

- Hiding sensitive data in Engine debug logs (not only VM external data).
- Support for snapshots with memory (depending on libvirt – `https://bugzilla.redhat.com/1855367`).
- Better consistency guarantees for VM external data in snapshots.
- Limiting permissions to put VM external data on storage (snapshots, exports, hibernation; `https://bugzilla.redhat.com/1352501`).

## Resources

- Feature page for TPM: `https://www.ovirt.org/develop/release-management/features/virt/tpm-device.html`
- Feature page for secure boot NVRAM data persistence: `https://www.ovirt.org/develop/release-management/features/virt/nvram-data.html`
- swtpm: `https://github.com/stefanberger/swtpm`
- tpm2-tools: `https://github.com/tpm2-software/tpm2-tools`
- Clevis: `https://github.com/latchset/clevis`

## Thank you!

**Questions?**

<users@ovirt.org>