Updates on Coconut SVSM

Secure Services and Stateful Devices for Confidential Virtual Machines

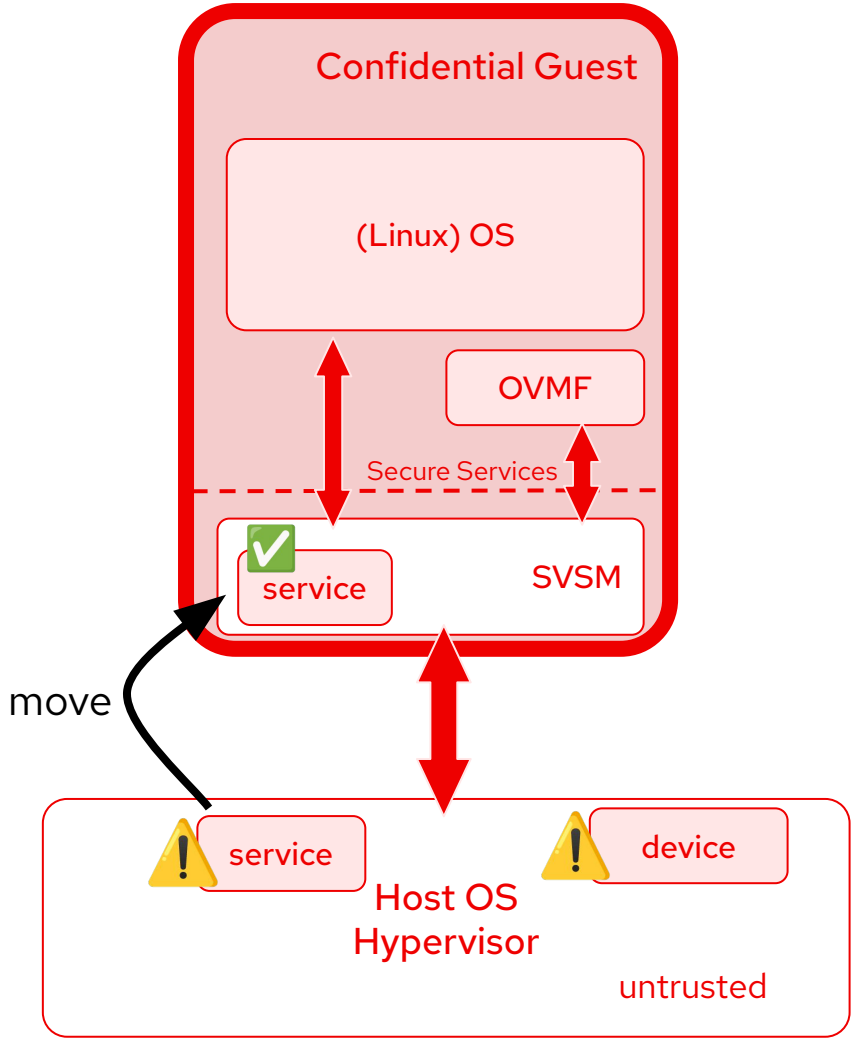
FOSDEM 2025

Stefano Garzarella
<sgarzare@redhat.com>

Oliver Steffen
<osteffen@redhat.com>

Red Hat

Secure Services for CVMs

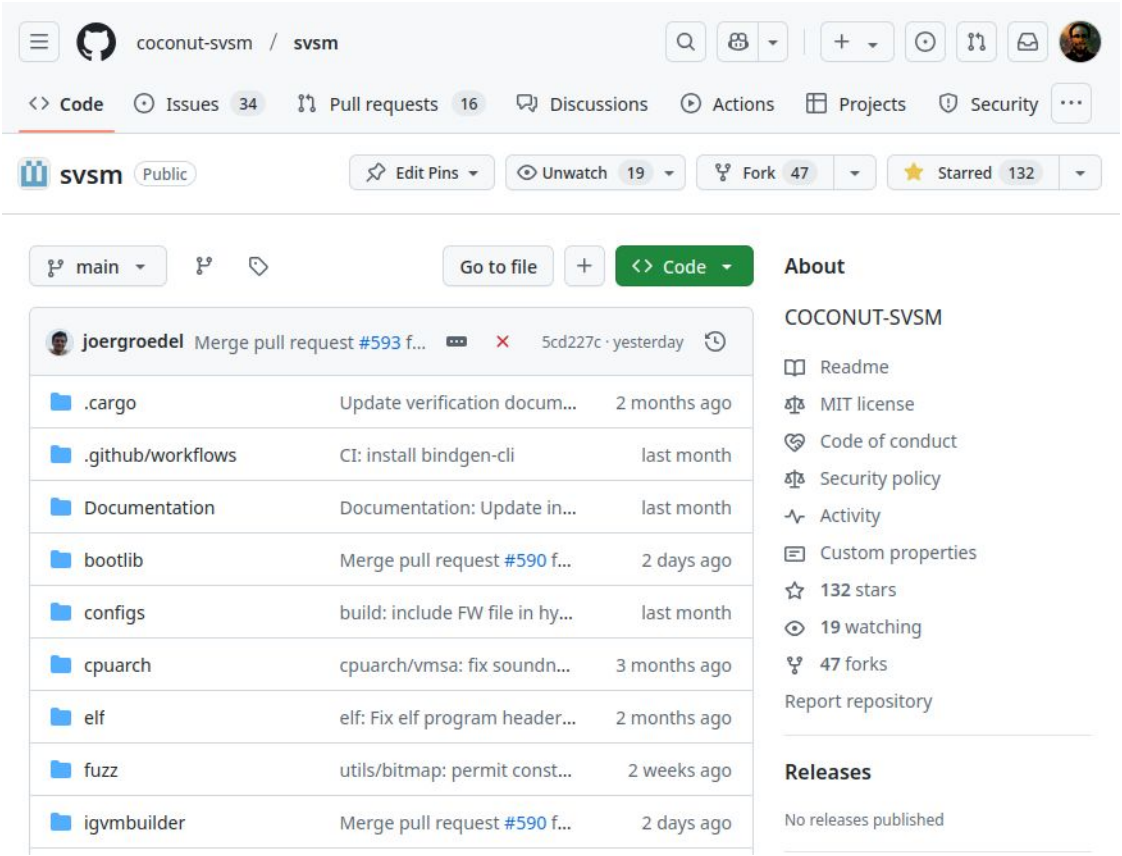


- Confidential Virtualization:
  Hardware is trusted, Host OS/cloud provider is not trusted
- Hardware guarantees confidentiality of guest memory and CPU state
- Problem: The host can't provide security relevant services anymore.

▶️ Service Module for confidential VMs:

- Execution environment for providing services and devices to confidential guest in a secure way
- Runs inside the TCB of the CVM
- Ex: AMD SEV-SNP: SVSM (Secure Vm Service Module)
- Useful for:
  - Emulating a vTPM
  - UEFI variable storage
  - Migration helper
  - APIC emulation + IRQ delivery
  - VC handling

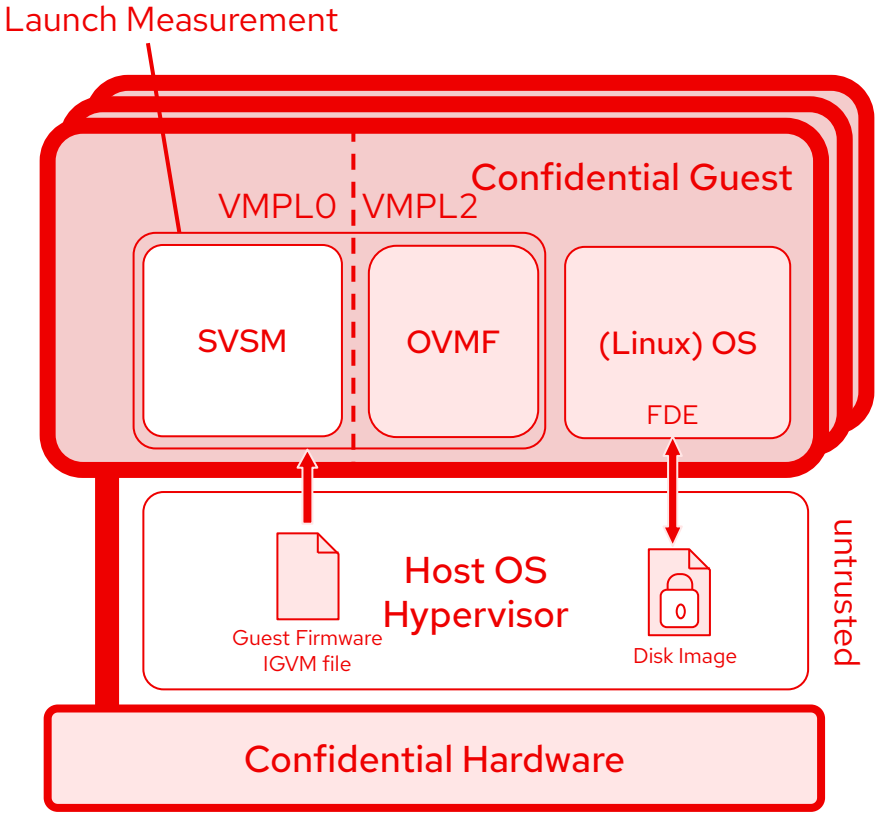
Coconut-SVSM



- Service Module for confidential VMs
- Supports AMD SEV-SNP, Intel TDX is WIP
- Supports QEMU, Hyper-V, and Vanadium
- Written in Rust
- Started in 2022
- MIT/Apache2.0
- Project is part of the Confidential Computing Consortium
- Provides a virtual TPM to the guest
- Currently requires enlightened guest (Service module mode)
  - Pavavisor and Service VM mode planned

https://github.com/coconut-svsm/svsm

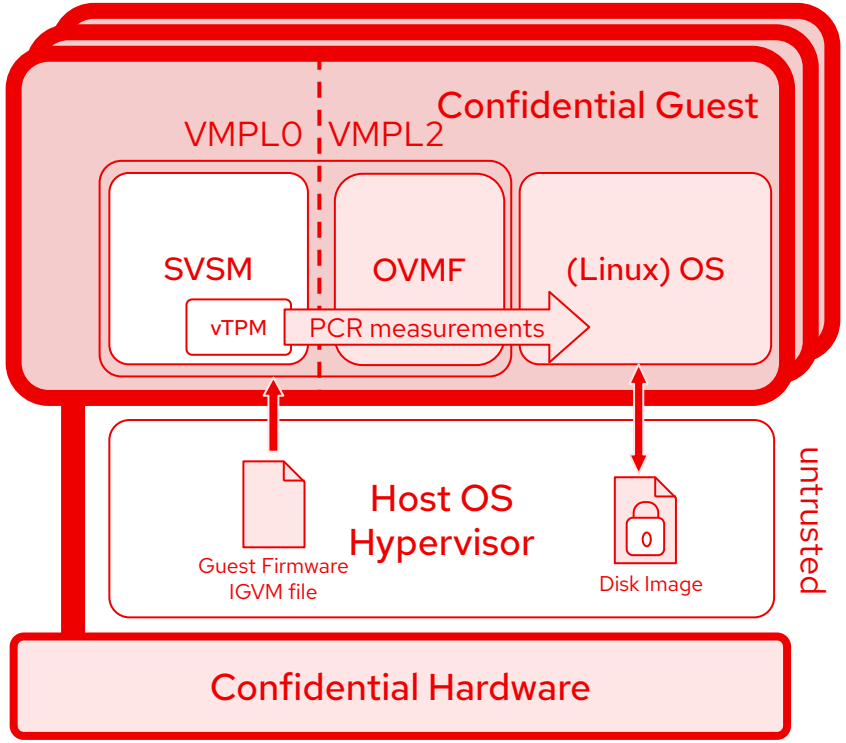
Current state of Coconut on AMD SEV-SNP



- Coconut packaged as IGVM file
  - Contains SVSM + OVMF and defines the initial VM state
- Initial launch measurement covers the initial state
- VM to request attestation report from HW and perform remote attestation

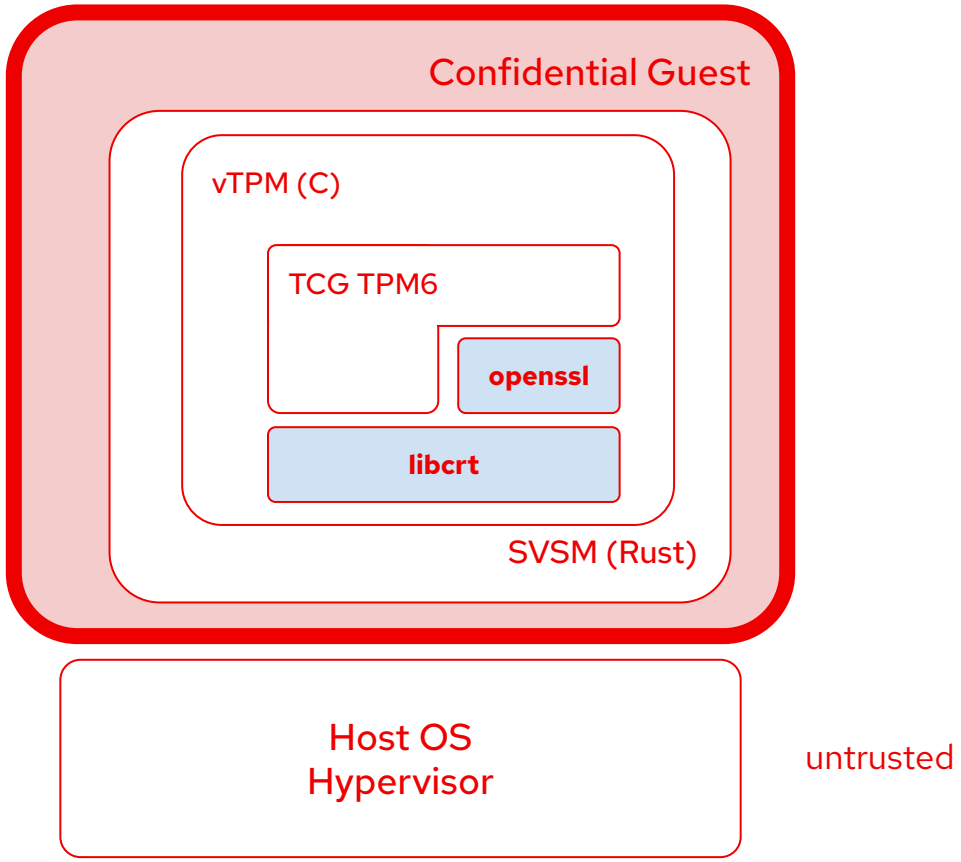
- SVSM runs at VMPL0
- Launches OVMF + Linux OS at VMPL2
- Linux and OVMF support running under an SNP-SVSM already

Current state of Coconut on AMD SEV-SNP



- Coconut SVSM provides an ephemeral vTPM
  - State is not preserved (no secure storage available)
  - PCRs for measured boot and IMA runtime measurements
  - SNP attestation report includes the TPM EK
    -> remote attestation can establish trust in the TPM
- OVMF: “In RAM” UEFI variable store
  - Volatile: User can’t customize SecureBoot settings, boot options, etc.
  - Not possible to implement securely in OVMF due to lack of SMM
  - SVSM could provide a EFI variable service
- Open Questions
  - How to automatically unlock the root disk?
  - When and how to do remote attestation?
    - Possible anywhere in the boot process, depending on use-case
  - Can we add persistent storage? ⇨ Later...

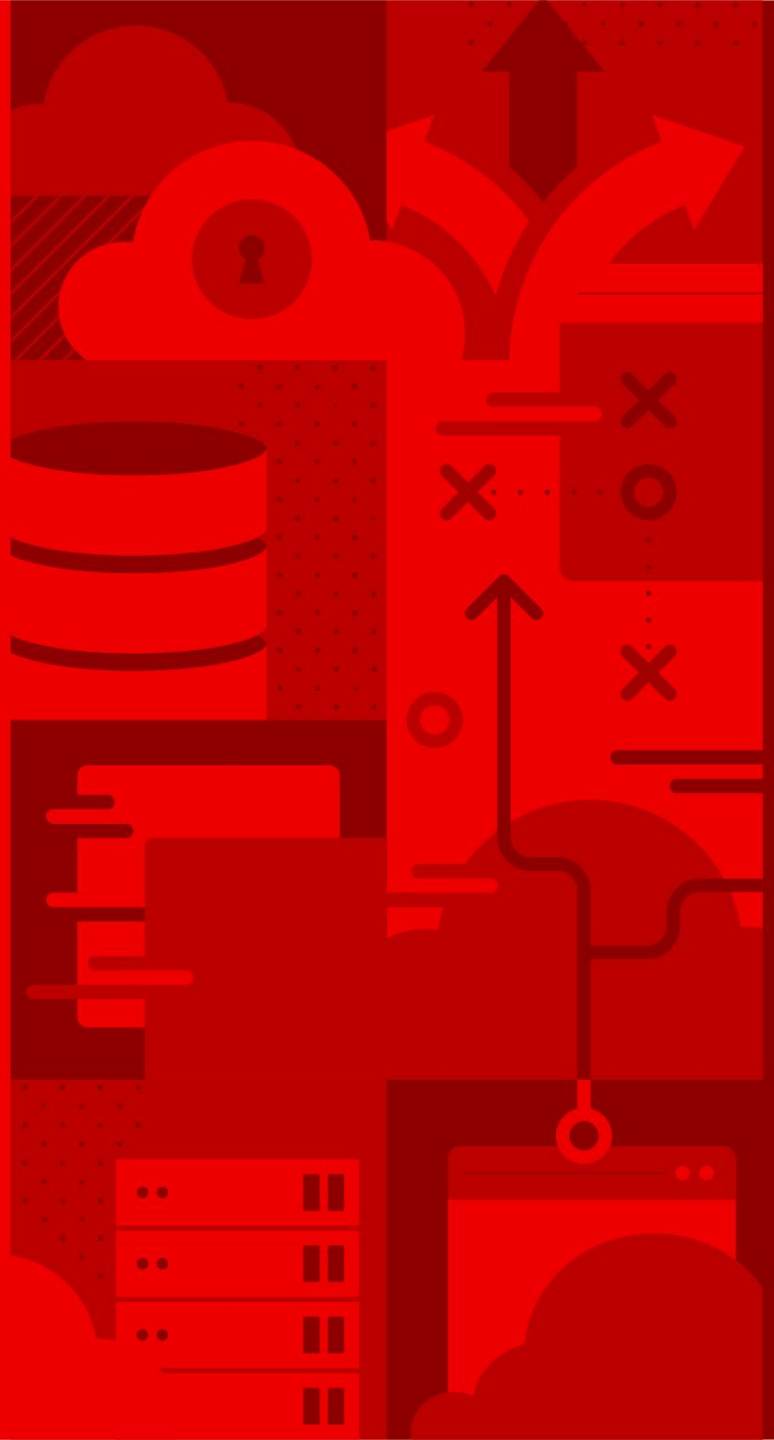
Coconut's virtual TPM



- First usable service Coconut provides 🥳
- Uses the TCG reference implementation (in C)
- Small C lib and OpenSSL
- Uses SVSM-Guest interface calls (AMD SEV-SNP)
  - OS requires enlightened drivers
- TPM is stateless / ephemeral:
  EK regenerated at boot, NV-storage is not preserved
- Useful for measured boot

Roadmap

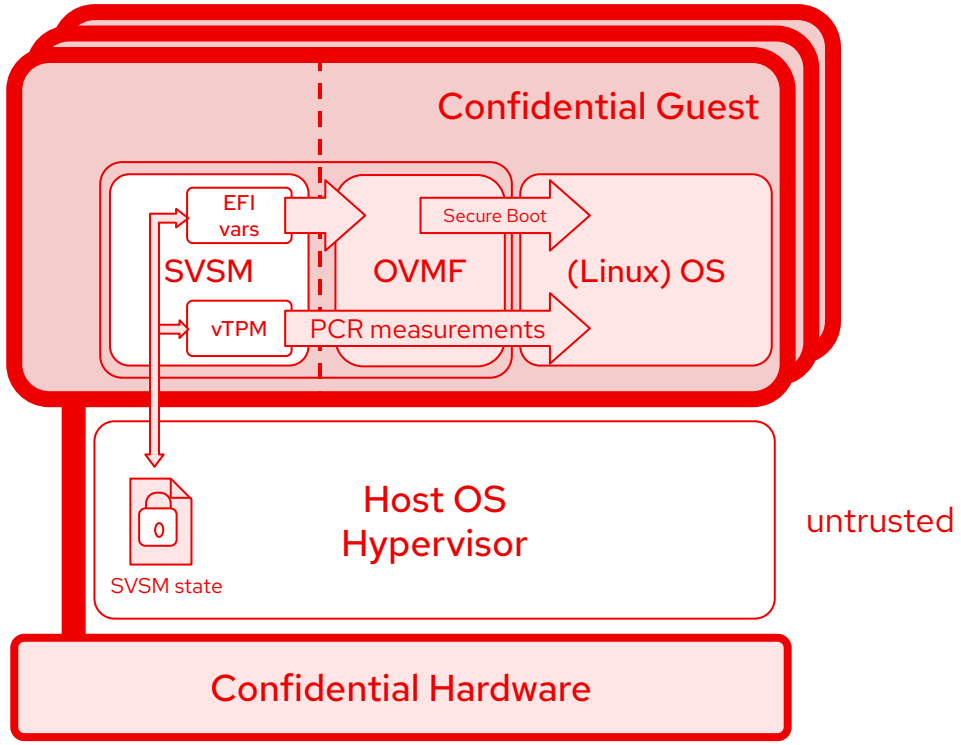
- First official (development) release planned!
- User mode: use regular privilege levels to implement user tasks and split kernel/user-land
  - needs system call interface, support library, memory management for user mode
- x2APIC Support: required for non-AMD platforms
- Paravisor mode for non-enlightened guests
- Working on upstreaming
  - vTPM driver for Linux and EDK2
  - QEMU support for SVSM + IGVM
- Add secure persistent storage for vTPM state and other uses



SVSM state persistence

Red Hat

SVSM State: persistence

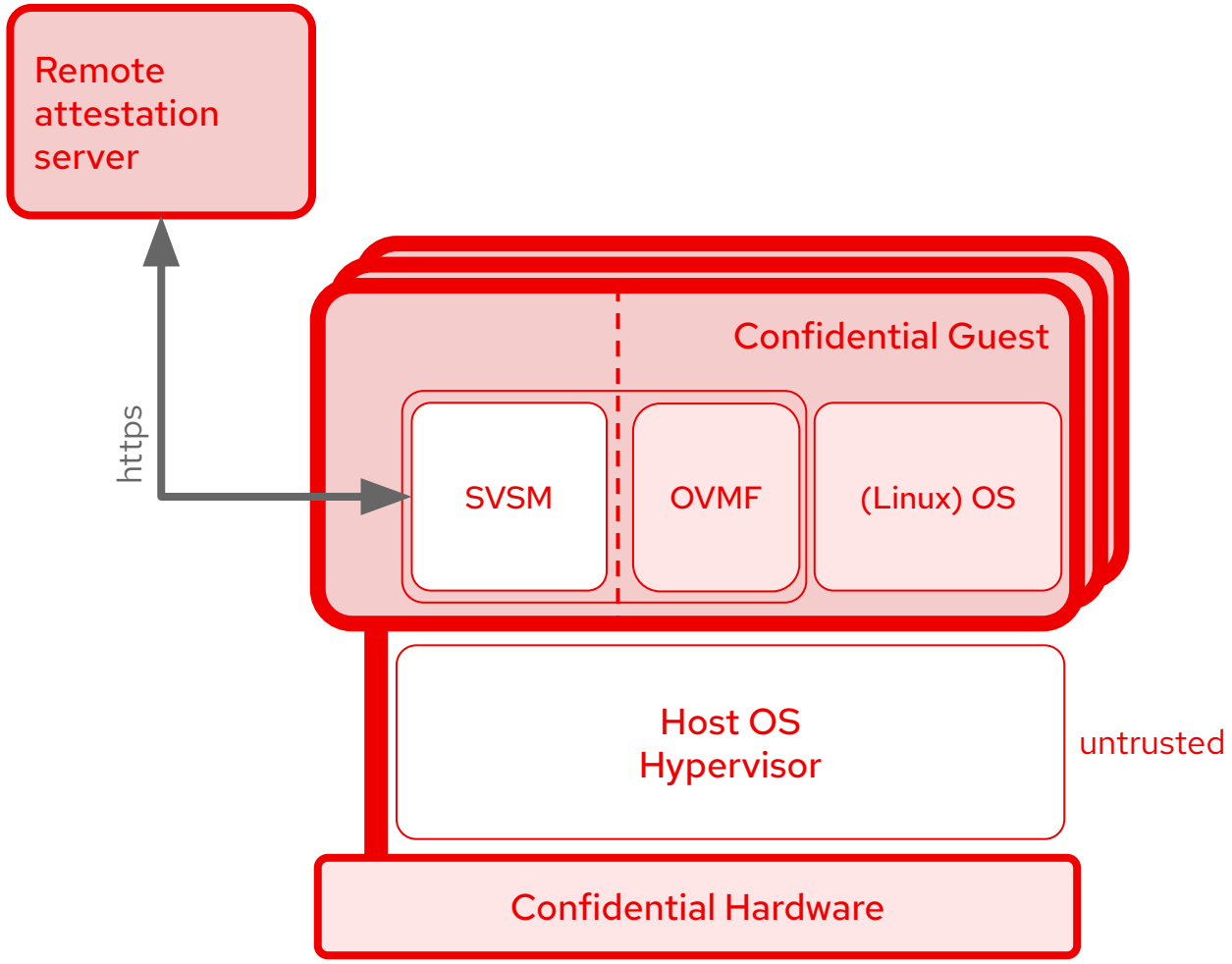


- Support **stateful services**
  - vTPM, UEFI variable store, etc.
- SVSM State = vTPM state + UEFI variables + …
  - Add a **storage driver** to SVSM
    - Storage backend provided by the host
  - Use **encryption**
    - Host is not trusted!
  - Support **multiple drivers** for different hypervisors

How to decrypt the SVSM state ?

Early attestation in SVSM

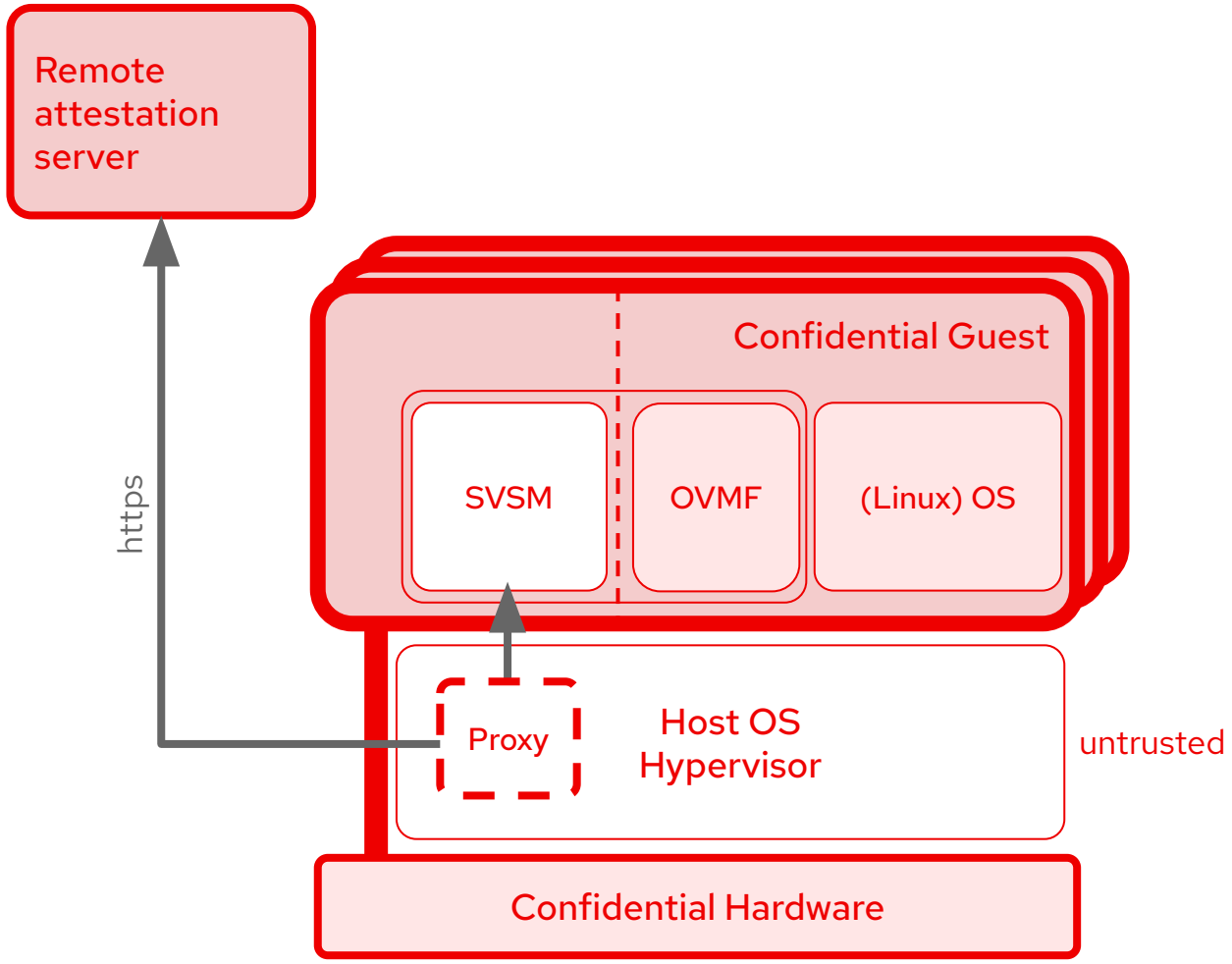
- **Encrypted** state
  - Unlocked only after a successful remote attestation
- **Remote attestation**
  - HW generates an attestation report (evidence)
    - signed by HW's vendor certificate
  - Remote server (trusted) checks the evidence
    - Expected SW running on a genuine HW
  - Remote server sends back the SVSM state key
    - Unlock vTPM state, UEFI variable service, etc.
- Challenges
  - **Network stack** not available in SVSM
  - Support **multiple** remote attestation **protocols**



Red Hat

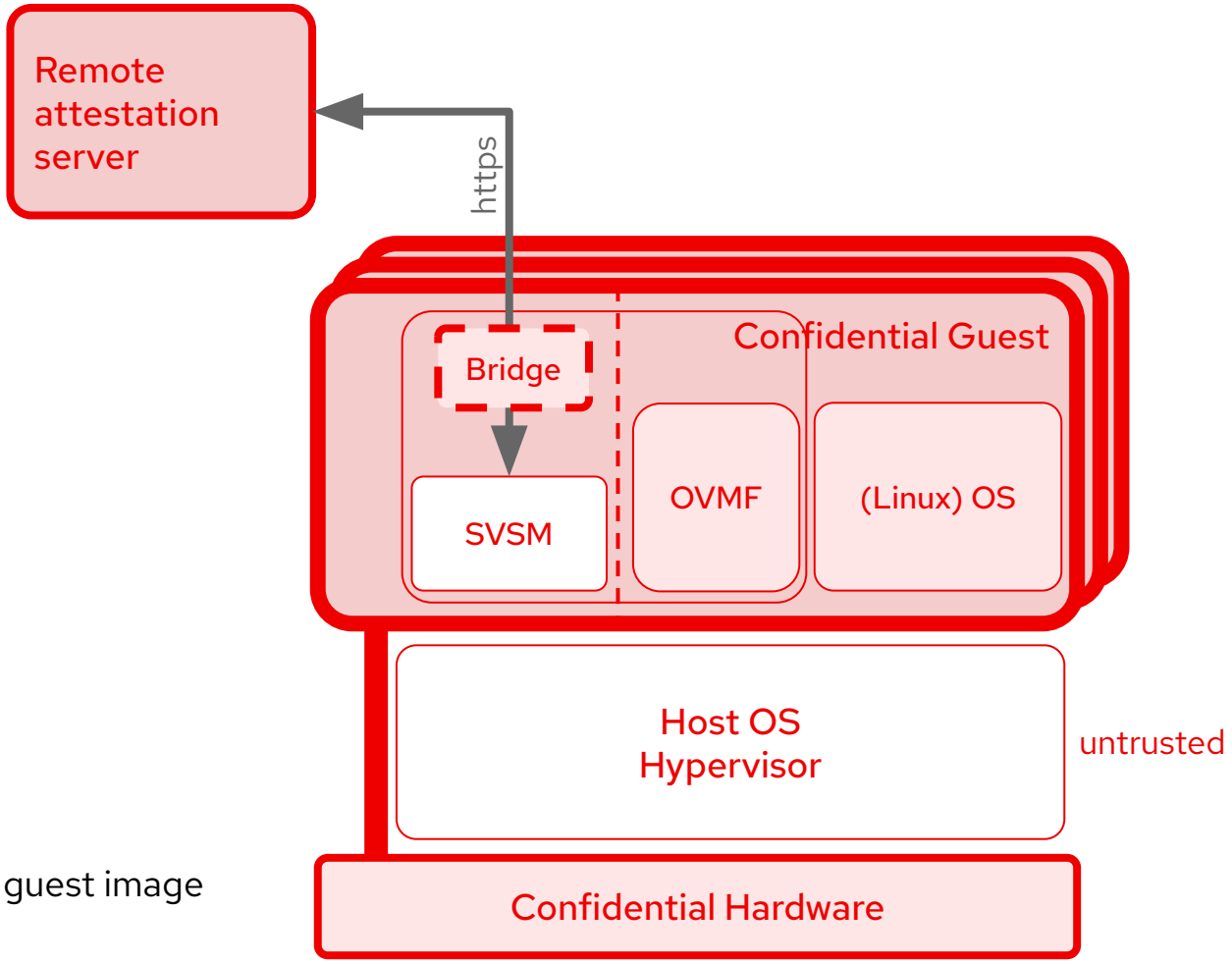
Attestation proxy

- **Proxy** application running on the **host**
  - Simple application forwarding requests coming from SVSM to the https connection
- Pro
  - No network stack in SVSM
  - Use features already supported by VMM (e.g. vsock, serial port)
- Cons
  - Require network connectivity in the host
  - TLS ended in the host



Attestation bridge

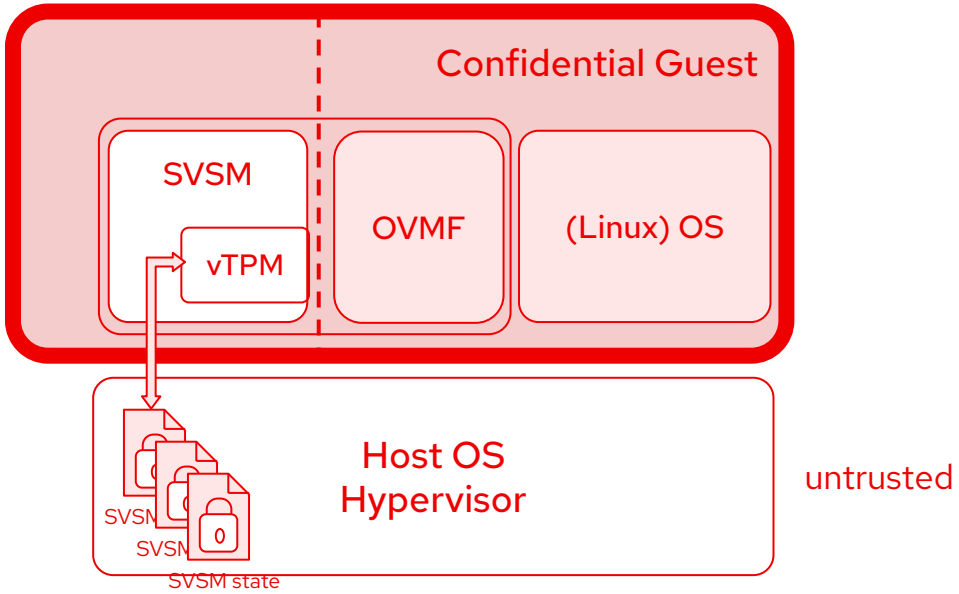
- **Bridge** application running in the **guest**
  - UEFI application
  - Minimal service OS
  - SVSM user space application
- Pro
  - Self-contained in the guest firmware
  - Host network connectivity not required
- Cons
  - Bridge will be part of launch measurement
  - Bridge requires network setup
  - Boot phase a bit more complex
    - SVSM needs to boot the bridge first, then the real guest image



HPE proposal:
https://mail.8bytes.org/pipermail/svsm-devel/2025-January/000442.html

SVSM state: Rollback and clone attacks mitigation

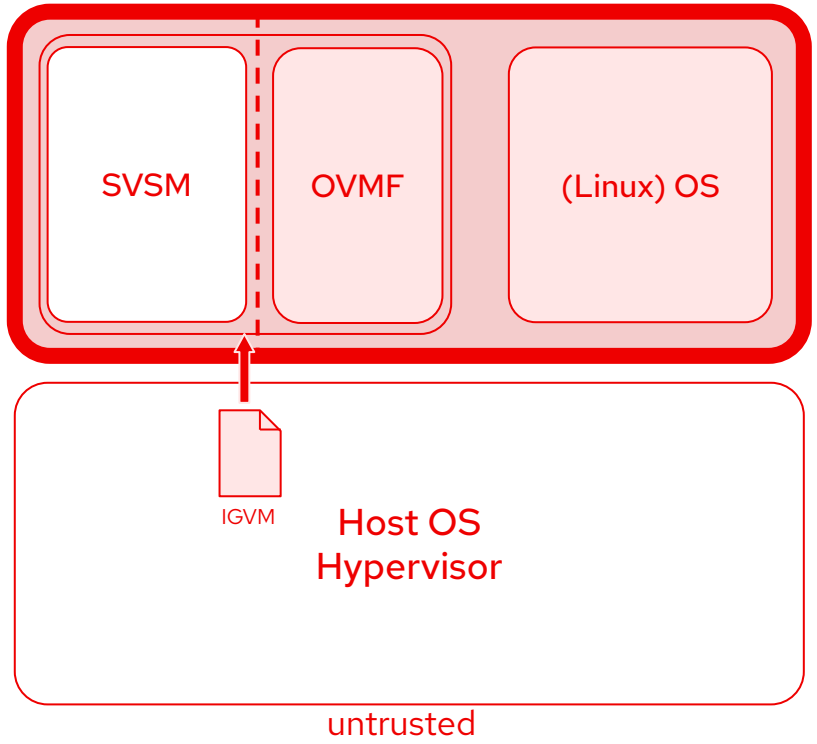
- Malicious host could perform some **attacks** with the **persistent state** of SVSM
  - Rollback: reuse an old state
    - TPM monotonic counters could be unreliable
    - SecureBoot updates can be undone
  - Clone: spawn a copy
    - Same TPM identity for different instances
- How to **mitigate** these attacks
  - Rollback: boot counter
  - Clone: only one successful attestation per boot request
- TCG **Virtualized Platform WG**
  - Ongoing discussions
    - possible changes to the TPM specification
    - attestations protocols
  - https://trustedcomputinggroup.org/work-groups/virtualized-platform/
  - https://github.com/TrustedComputingGroup/Virtualized-Platform-WG



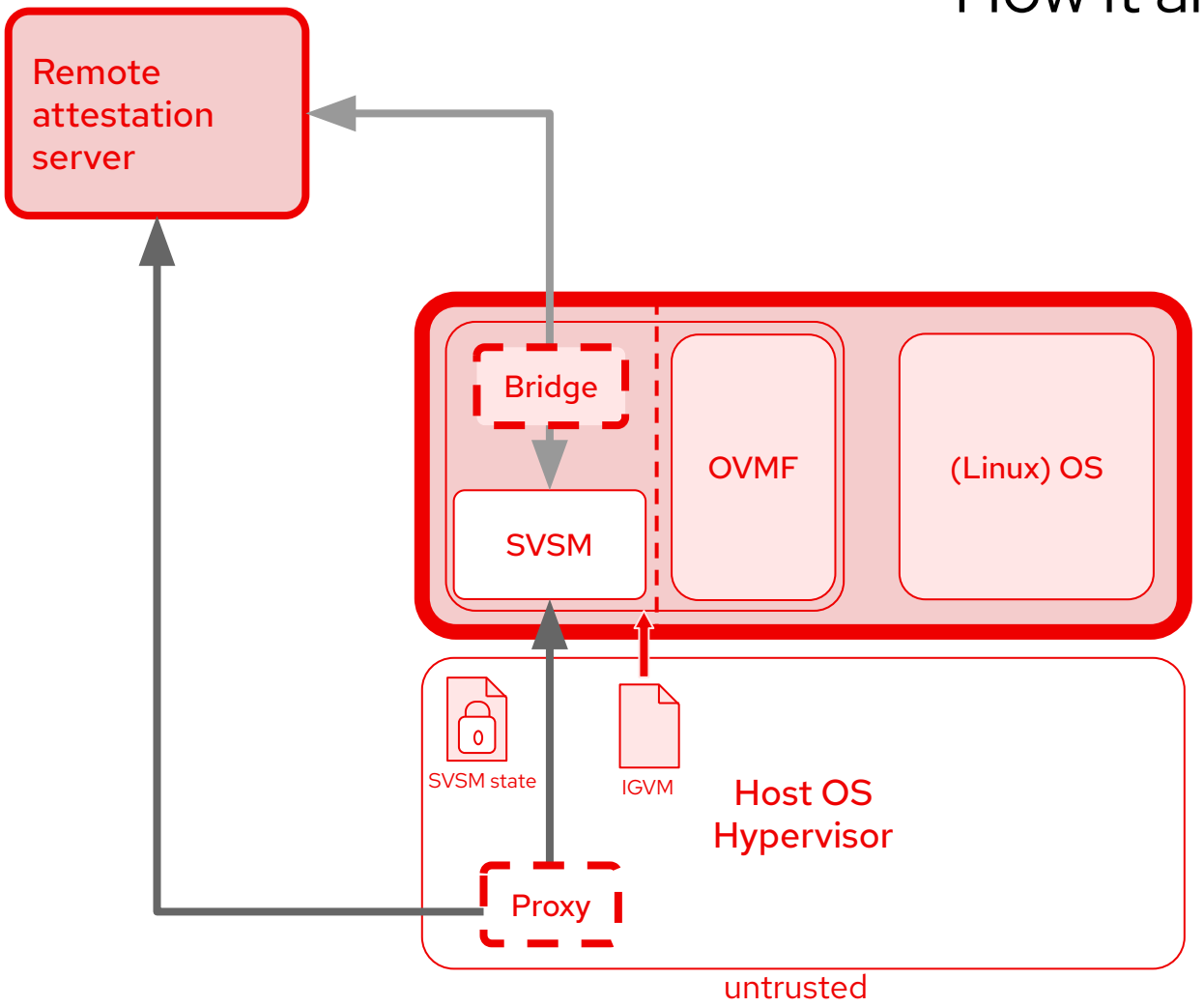
Red Hat

“How it all works”

- SVSM boots up from IGVM file

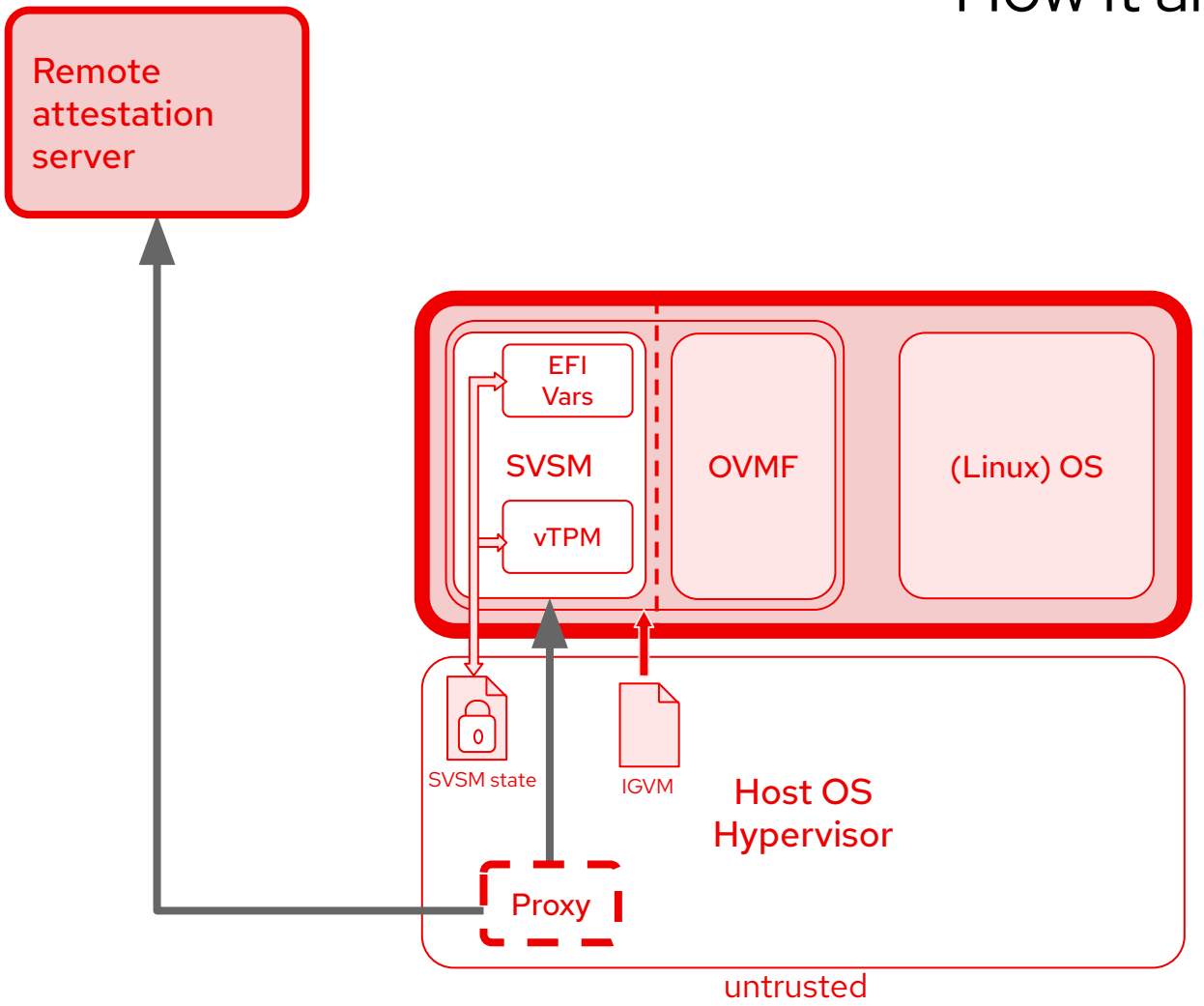


“How it all works”



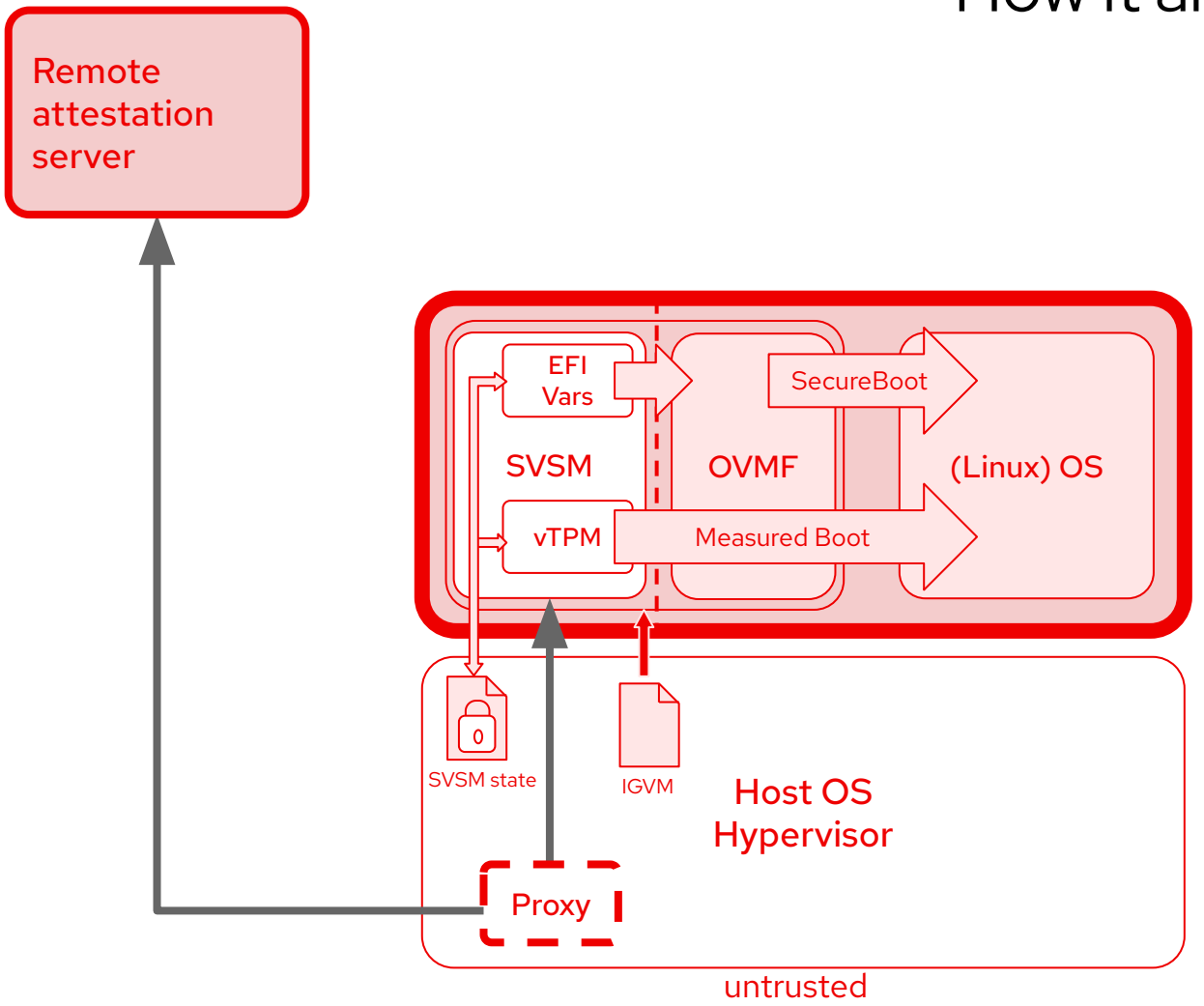
- SVSM boots up from IGVM file
  - Uses proxy or bridge to connect to attestation server
    - Sends attestation report
    - Receives key for SVSM state store
  - Unlocks state storage

“How it all works”



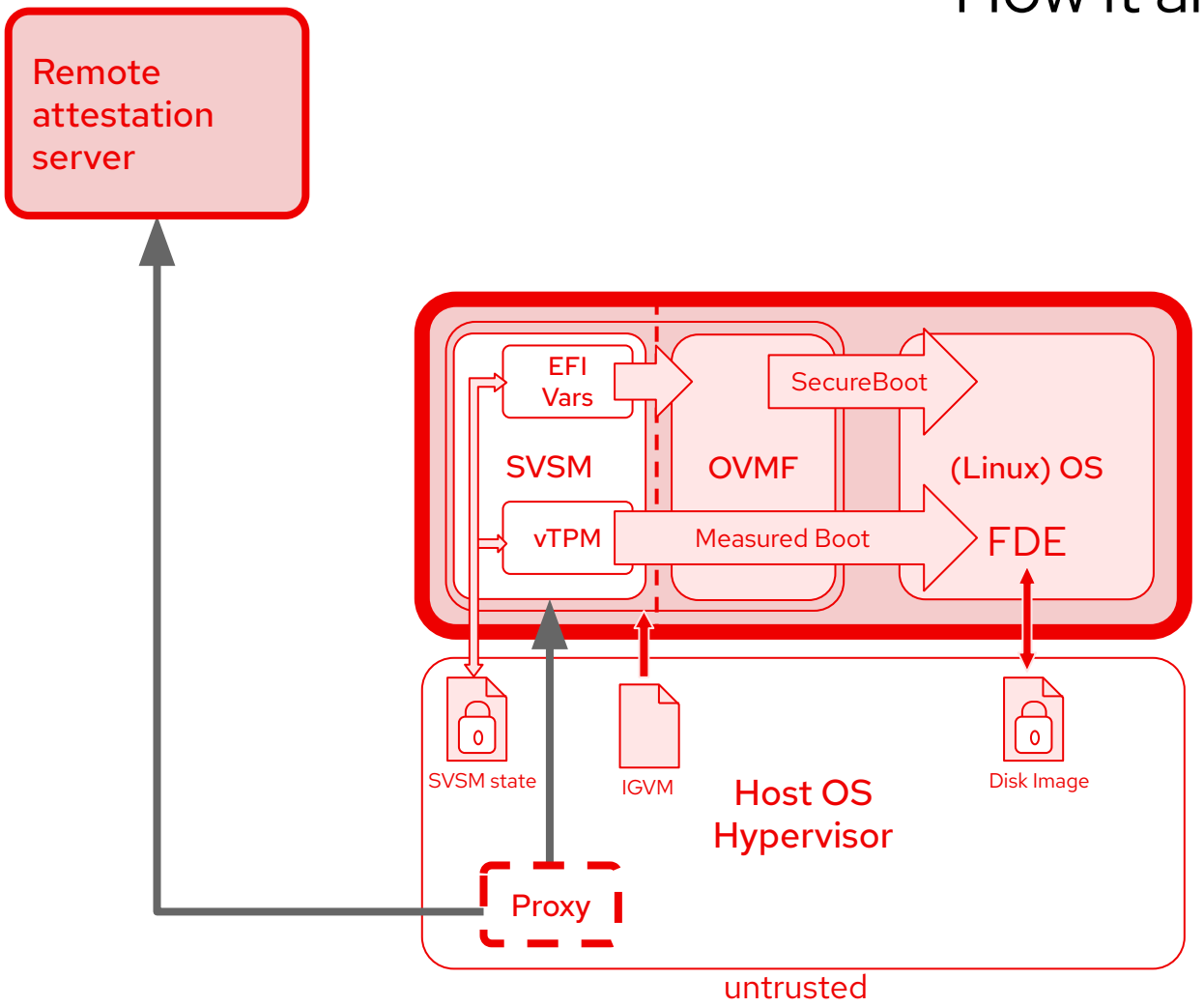
- SVSM boots up from IGVM file
  - Uses proxy or bridge to connect to attestation server
    - Sends attestation report
    - Receives key for SVSM state store
  - Unlocks state storage
  - Initialize vTPM and UEFI variable service from that
  - Continues boot process and launches OVMF

“How it all works”



- SVSM boots up from IGVM file
  - Uses proxy or bridge to connect to attestation server
    - Sends attestation report
    - Receives key for SVSM state store
  - Unlocks state storage
  - Initialize vTPM and UEFI variable service from that
  - Continues boot process and launches OVMF
- OVMF launches OS using secure boot and measured boot

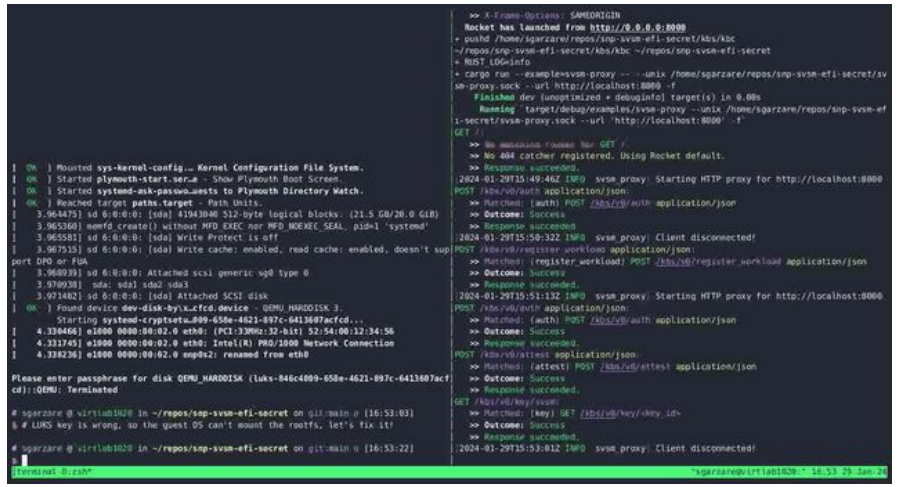
“How it all works”



- SVSM boots up from IGVM file
  - Uses proxy or bridge to connect to attestation server
    - Sends attestation report
    - Receives key for SVSM state store
  - Unlocks state storage
  - Initialize vTPM and UEFI variable service from that
  - Continues boot process and launches OVMF
- OVMF launches OS using secure boot and measured boot
- OS is able to unlock FDE via TPM’s PCR policy
  - Boot continues

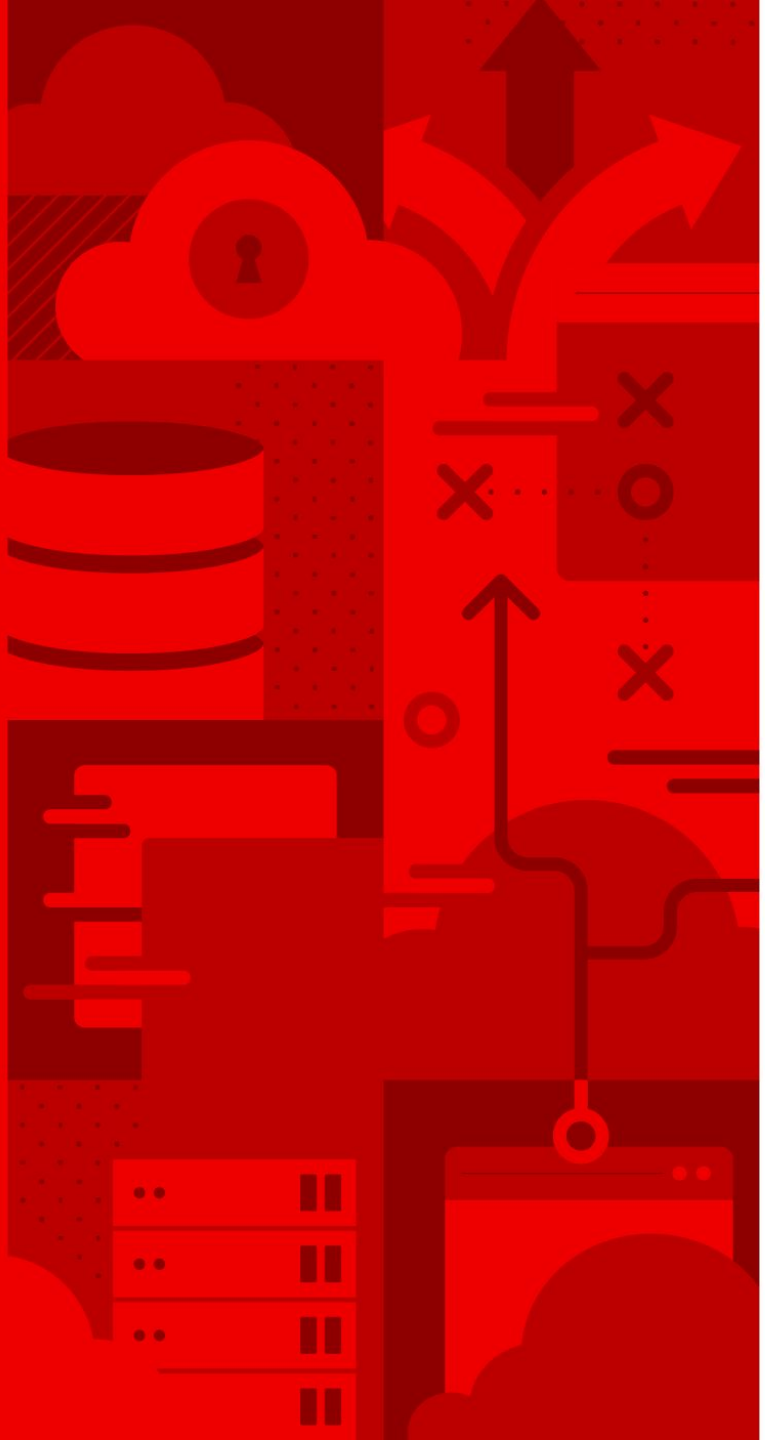
How to try SVSM with Fedora?

- **Demo**: https://github.com/stefano-garzarella/snp-svsm-vtpm
  - Remote attestation via host proxy
  - Encrypted SVSM persistent state (virtio-blk)
    - Unlocked after successful attestation
  - Loading of TPM state from the virtio-blk device
  - LUKS key sealed/unsealed with TPM's PCR policy
    - RootFS automatically unlocked



- **COPR** repo: https://copr.fedorainfracloud.org/coprs/g/virtmaint-sig/sev-snp-coconut/
  - **virt packages** with COCONUT SVSM enablement patches from https://github.com/coconut-svsm/
    - Linux kernel (host/guest)
    - QEMU
    - edk2
    - SVSM

https://red.ht/svsm



Thank you!

Stefano Garzarella
<sgarzare@redhat.com>

Oliver Steffen
<osteffen@redhat.com>

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat