Attestation and Confidential Dump for IBM® Secure Execution on Linux

Steffen Eiden <steffen.eiden@ibm.com>
Marc Hartmayer <mhartmay@de.ibm.com>

KVM Forum 2022

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time

this information was published. Such trademarks may also be registered or common law trademarks in other countries.

A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

The following are trademarks or registered trademarks of other companies.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

AMD, the AMD Arrow logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc.

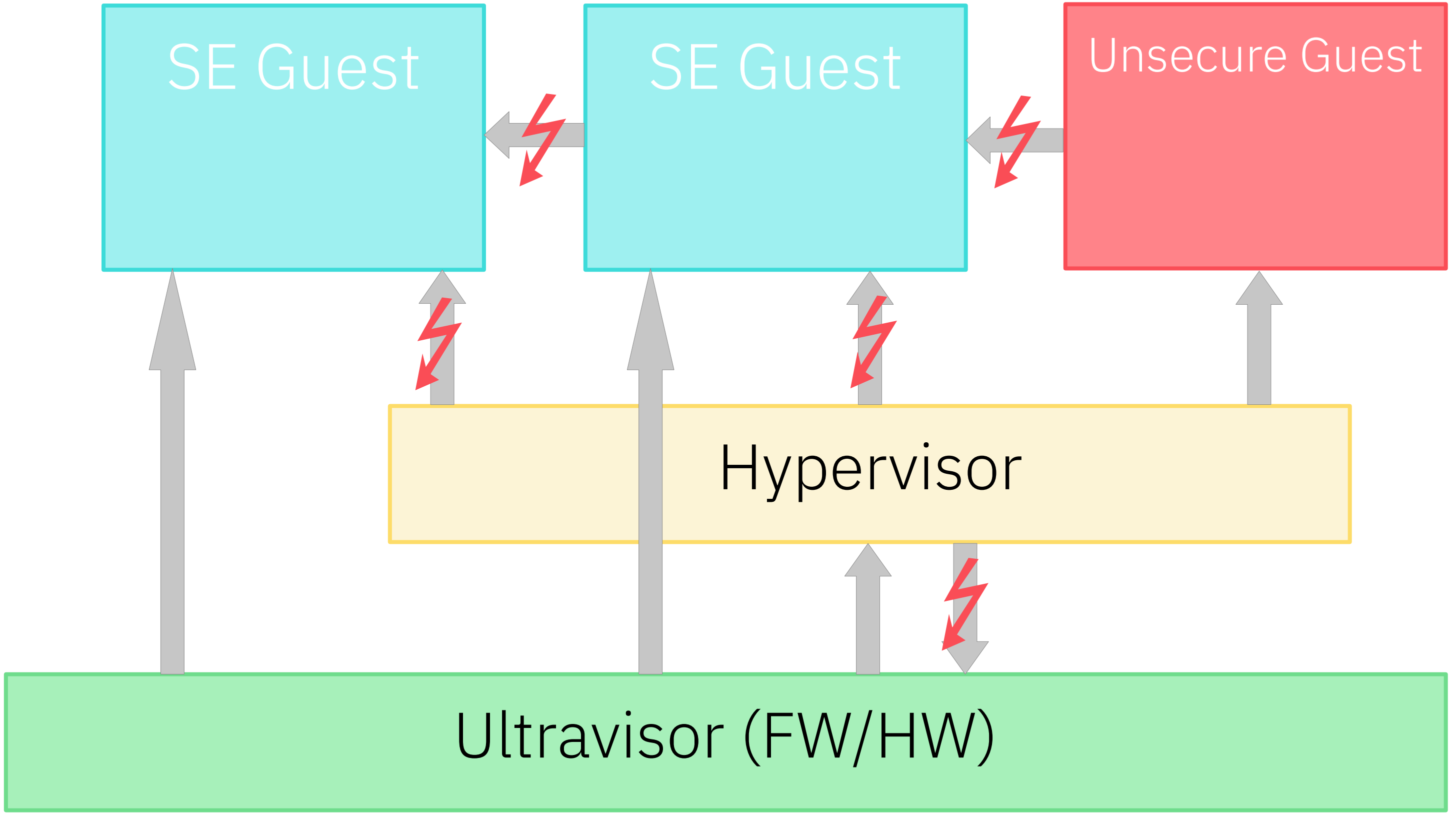
Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries.

Other company, product, or service names may be trademarks or service marks of others.

Contents

IBM® Secure Execution Revisited
Attestation
Confidential Dump

IBM® Secure Execution



IBM® Secure Execution

Guest owner prepares a SE boot image including:

– Guest owner public ECDH key. This key is used for establishing a shared secret between UV and guest owner:

$$ECDH(pub_{owner}, priv_{UV}) = secret_{shared} = ECDH(priv_{owner}, pub_{UV})$$

→ **Only Ultravisor (UV)** can decrypt and **execute** the SE image

– Guest owner secrets in SE header:

- Customer Communication Key (CCK)
- Keys for components decryption

Kernel, cmdline and initrd are always encrypted, authenticated and integrity protected

→ **Allows** the **storage of secrets** in these components

Attestation

Implicit Attestation

IBM Secure Execution does not require external attestation to prove that a guest is secure.

If the image contains a unique secret, a successful login implicitly *attests* a SE guest image.

The problem. Is there one?

Why nevertheless?

Explicit attestation on IBM z16™ is useful when

- Proving to a 3rd party without passing image secrets
- Verify that the guest is a specific image instance
- Needing trusted information about
  - SE guest image instance
  - Execution environment

Use cases

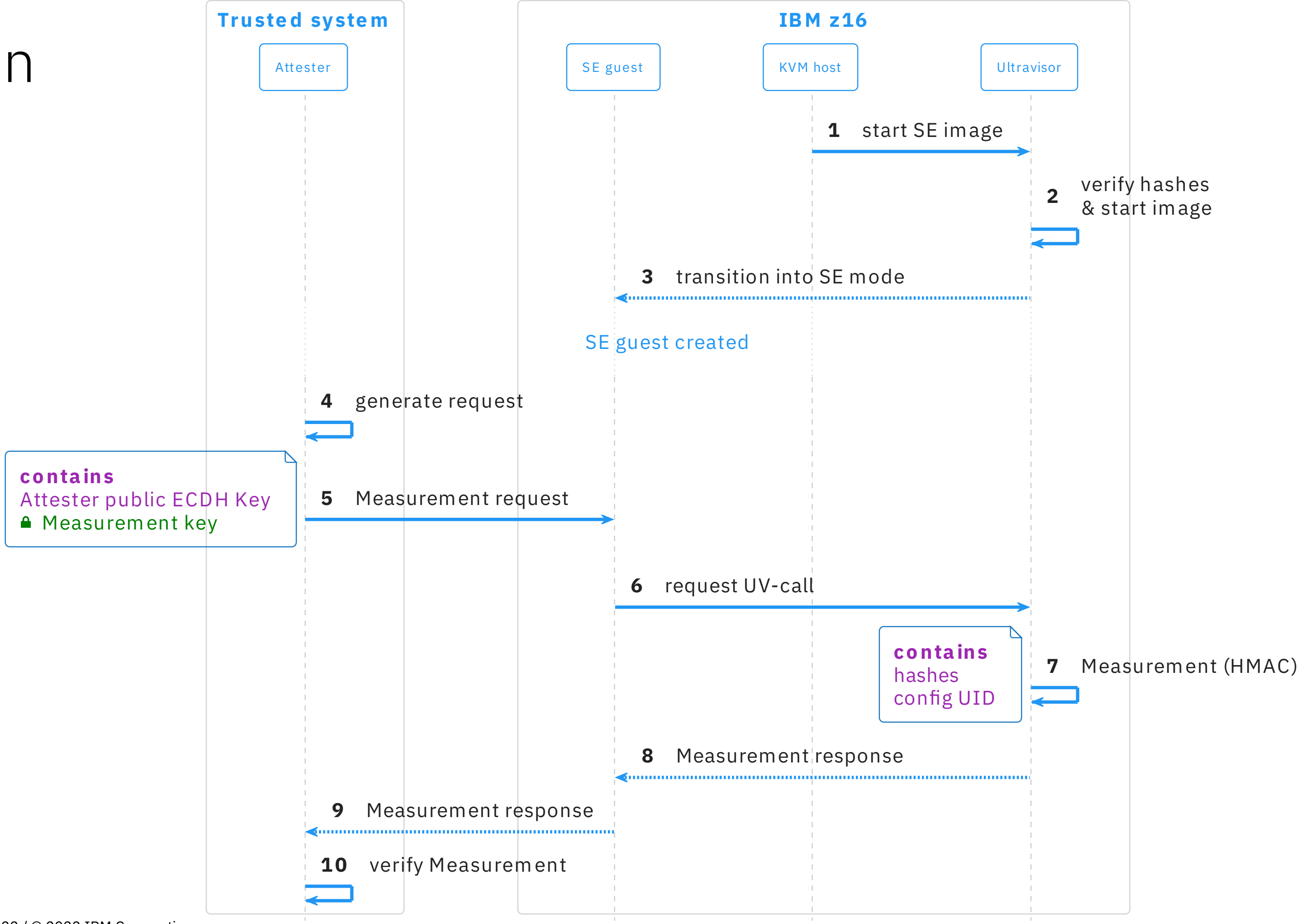
Become compliant

- Attestation request by 3rd party

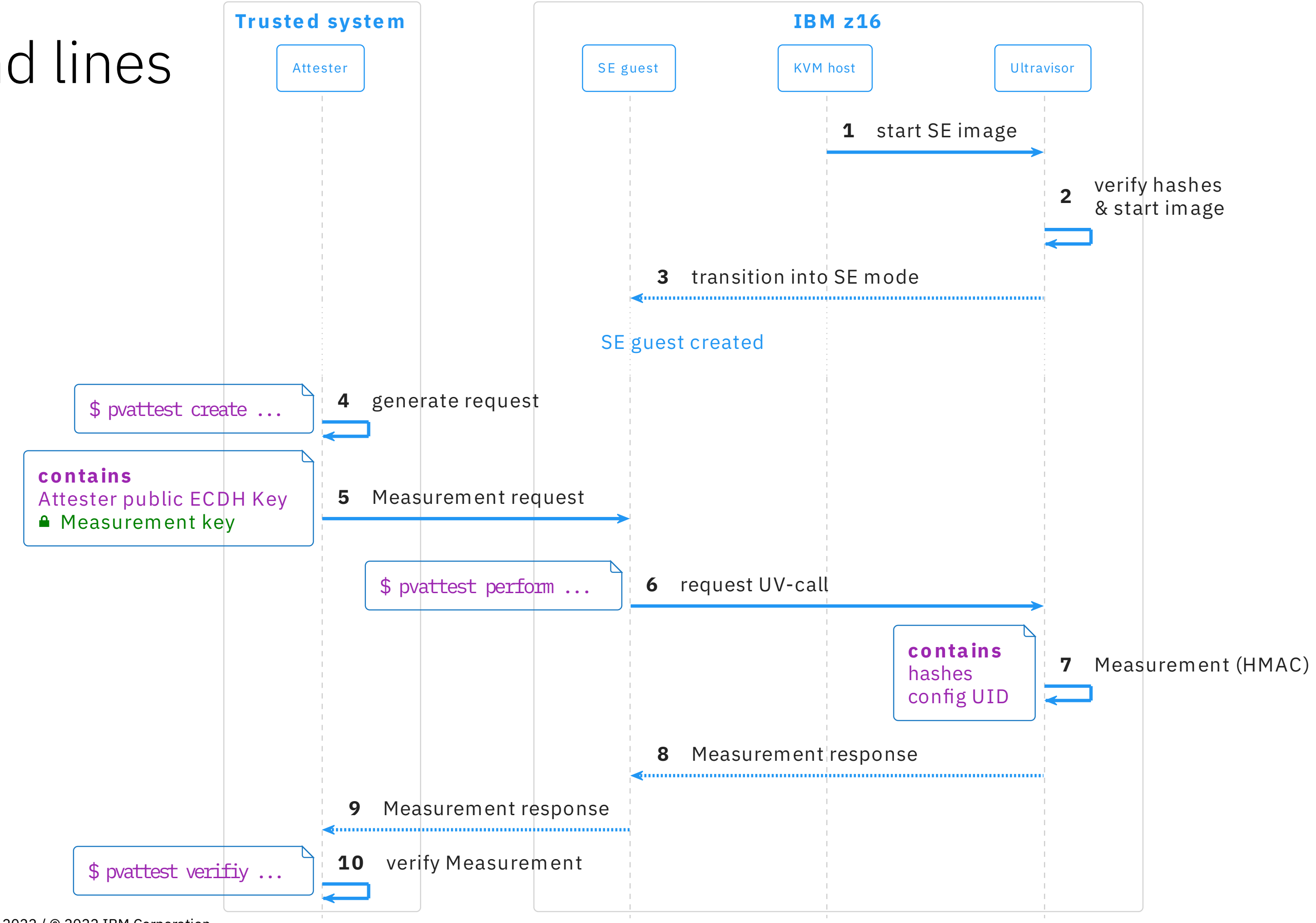
Customize an already prepared generic SE image

1. Attest image
2. Deploy own instance-dependent secrets

Attestation



Command lines



Trusted system

Attester

IBM z16

SE guest

KVM host

Ultravisor

1 start SE image

2 verify hashes & start image

3 transition into SE mode

SE guest created

`$ pvattest create ...`

4 generate request

contains
Attester public ECDH Key
🔒 Measurement key

5 Measurement request

`$ pvattest perform ...`

6 request UV-call

contains
hashes
config UID

7 Measurement (HMAC)

8 Measurement response

9 Measurement response

`$ pvattest verifiy ...`

10 verify Measurement

Current state

Hardware:

IBM z16

Kernel:

v5.19

QEMU, libvirt and genprotimg (s390-tools):

No changes – just works

pvattest (s390-tools):

v2.22.0

Confidential Dump

Guest vs. hypervisor initiated guest dumping

Guest initiated

Pro

- No hypervisor interaction required
- Guest knows its data best

Contra

- Not always possible, e.g. bug in memory management, early boot problem, ...
- Dumping modifies guest state
- Needs extra memory for dumper
- Must be set-up (e.g. kdump)

Hypervisor initiated

Pro

- Reliability
- Doesn’t modify guest state
- Guest initiated dumping is not always available

Contra

- Hypervisor interaction required
- Transport of dump
- Hypervisor needs access to guest state

→ Under SE, hypervisor does not have access to guest state, so how can you do hypervisor initiated dumps?

Problem: We don’t trust the hypervisor

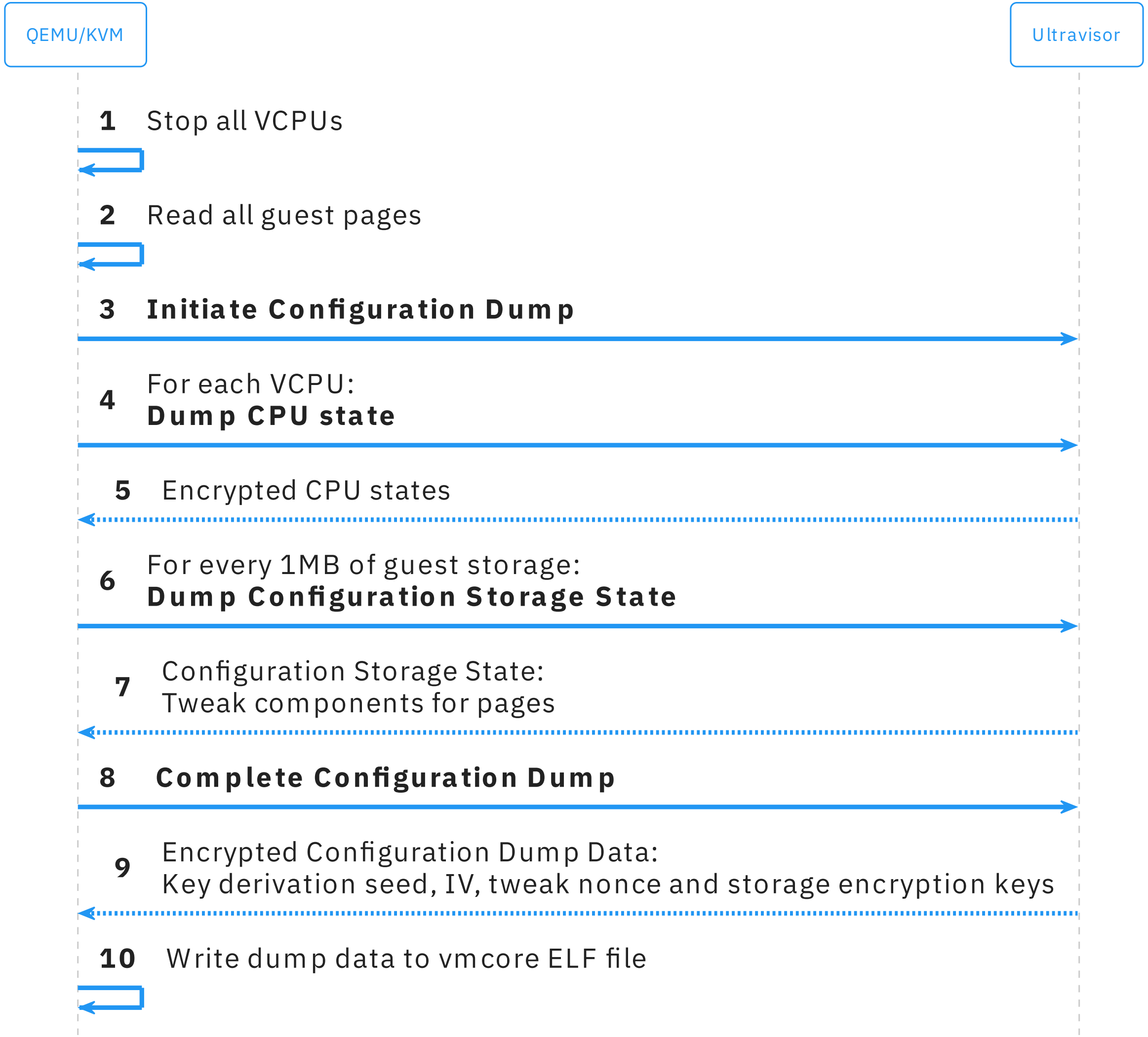
⇒ New Hardware/Firmware support

- Opt-in to enable confidential dump support via SE-header flag
- New Ultravisor calls (uses CCK for dump data protection and encryption)

1. Initiate Configuration Dump
2. Dump CPU state
3. Dump Configuration Storage[1] State
4. Complete Configuration Dump

[1] For s390x *storage* means *memory*

Dumping: QEMU/KVM perspective



1 Stop all VCPUs

2 Read all guest pages

3 **Initiate Configuration Dump**

4 For each VCPU:
Dump CPU state

5 Encrypted CPU states

6 For every 1MB of guest storage:
Dump Configuration Storage State

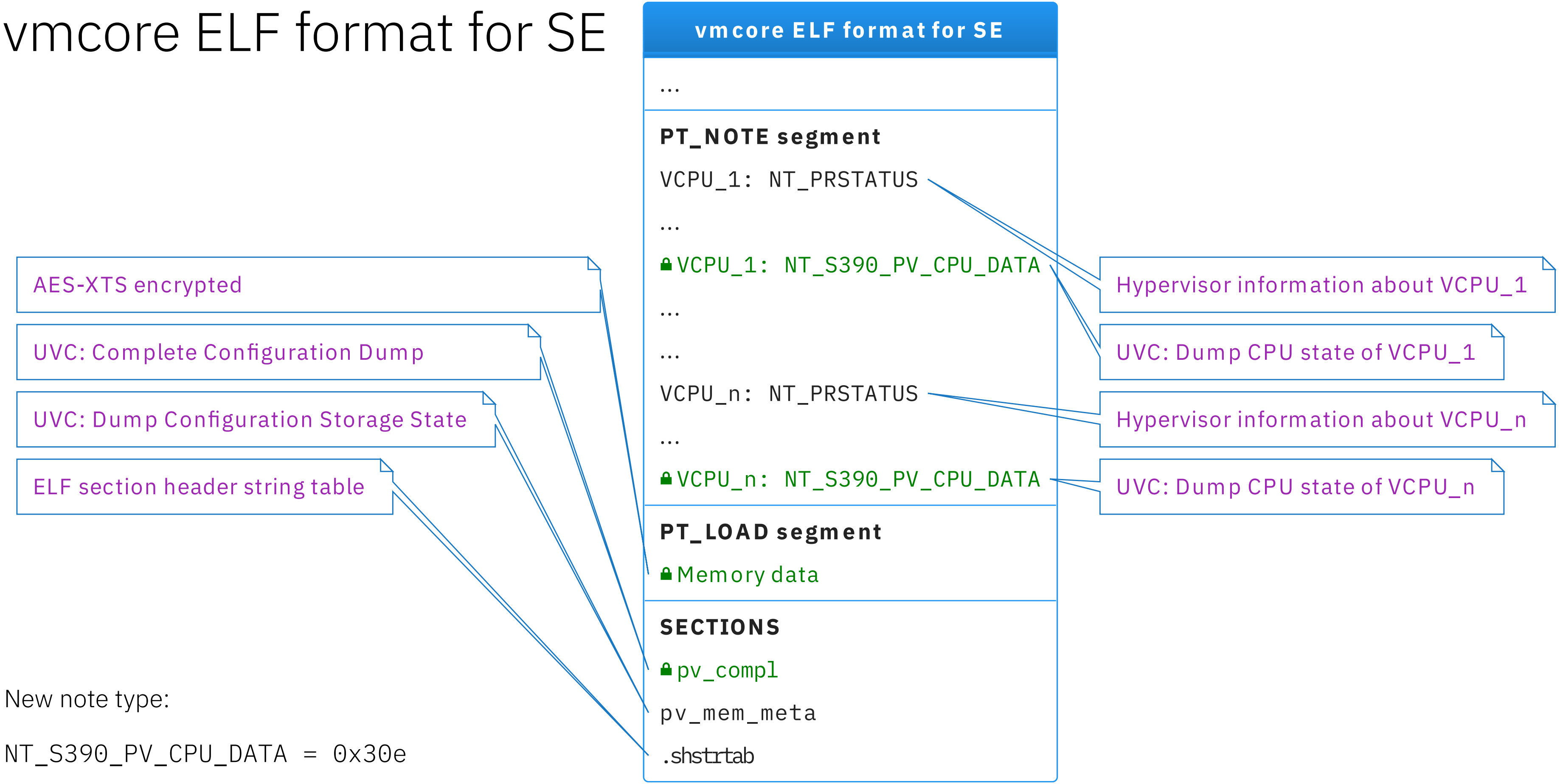
7 Configuration Storage State:
Tweak components for pages

8 **Complete Configuration Dump**

9 Encrypted Configuration Dump Data:
Key derivation seed, IV, tweak nonce and storage encryption keys

10 Write dump data to vmcore ELF file

vmcore ELF format for SE



vmcore ELF format for SE

...

PT_NOTE segment

VCPU_1: NT_PRSTATUS

...

🔒VCPU_1: NT_S390_PV_CPU_DATA

...

...

VCPU_n: NT_PRSTATUS

...

🔒VCPU_n: NT_S390_PV_CPU_DATA

PT_LOAD segment

🔒Memory data

SECTIONS

🔒pv_compl

pv_mem_meta

.shstrtab

AES-XTS encrypted

UVC: Complete Configuration Dump

UVC: Dump Configuration Storage State

ELF section header string table

Hypervisor information about VCPU_1

UVC: Dump CPU state of VCPU_1

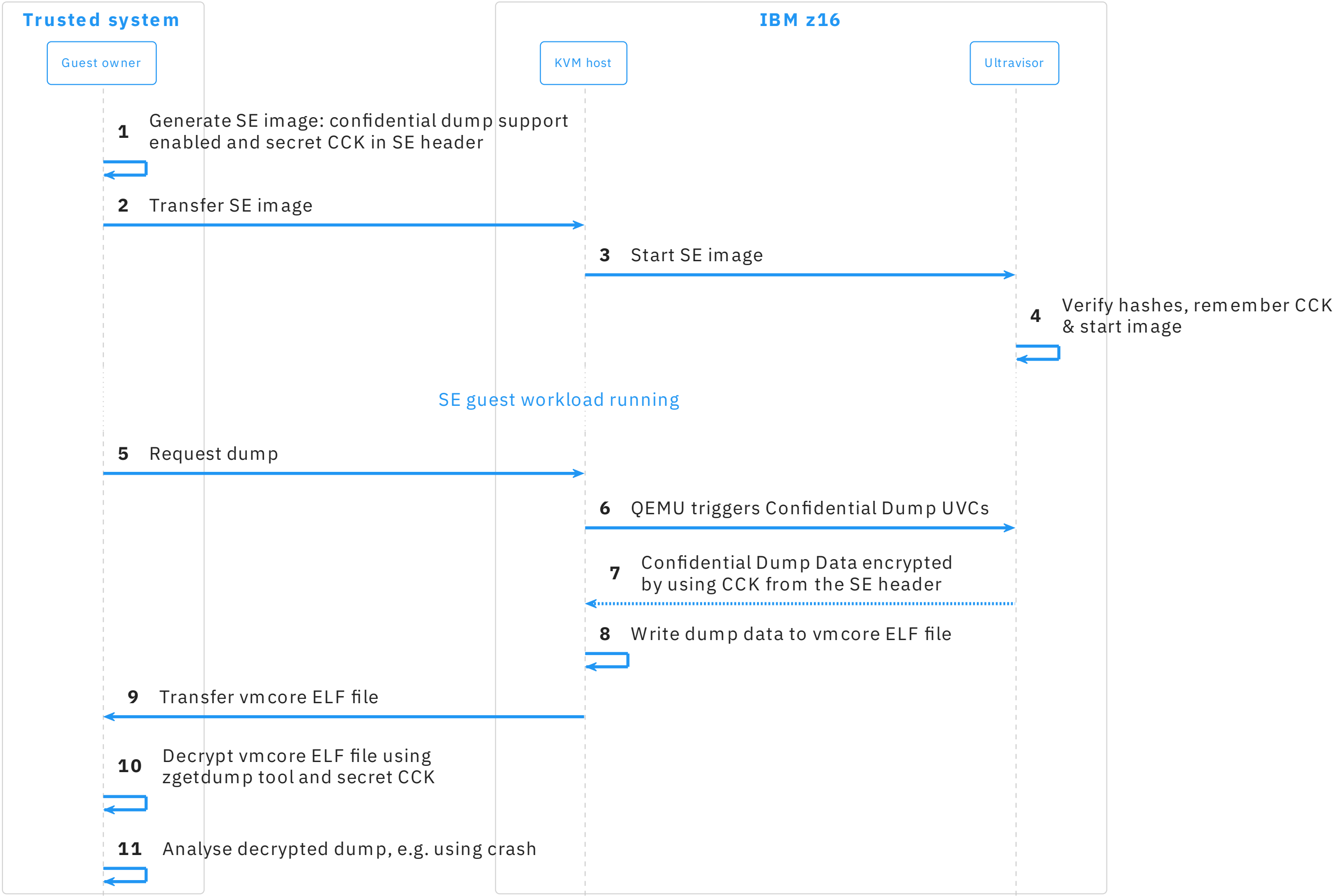
Hypervisor information about VCPU_n

UVC: Dump CPU state of VCPU_n

New note type:

`NT_S390_PV_CPU_DATA = 0x30e`

Life Cycle



Trusted system

Guest owner

IBM z16

KVM host

Ultravisor

1 Generate SE image: confidential dump support enabled and secret CCK in SE header

2 Transfer SE image

3 Start SE image

4 Verify hashes, remember CCK & start image

SE guest workload running

5 Request dump

6 QEMU triggers Confidential Dump UVCs

7 Confidential Dump Data encrypted by using CCK from the SE header

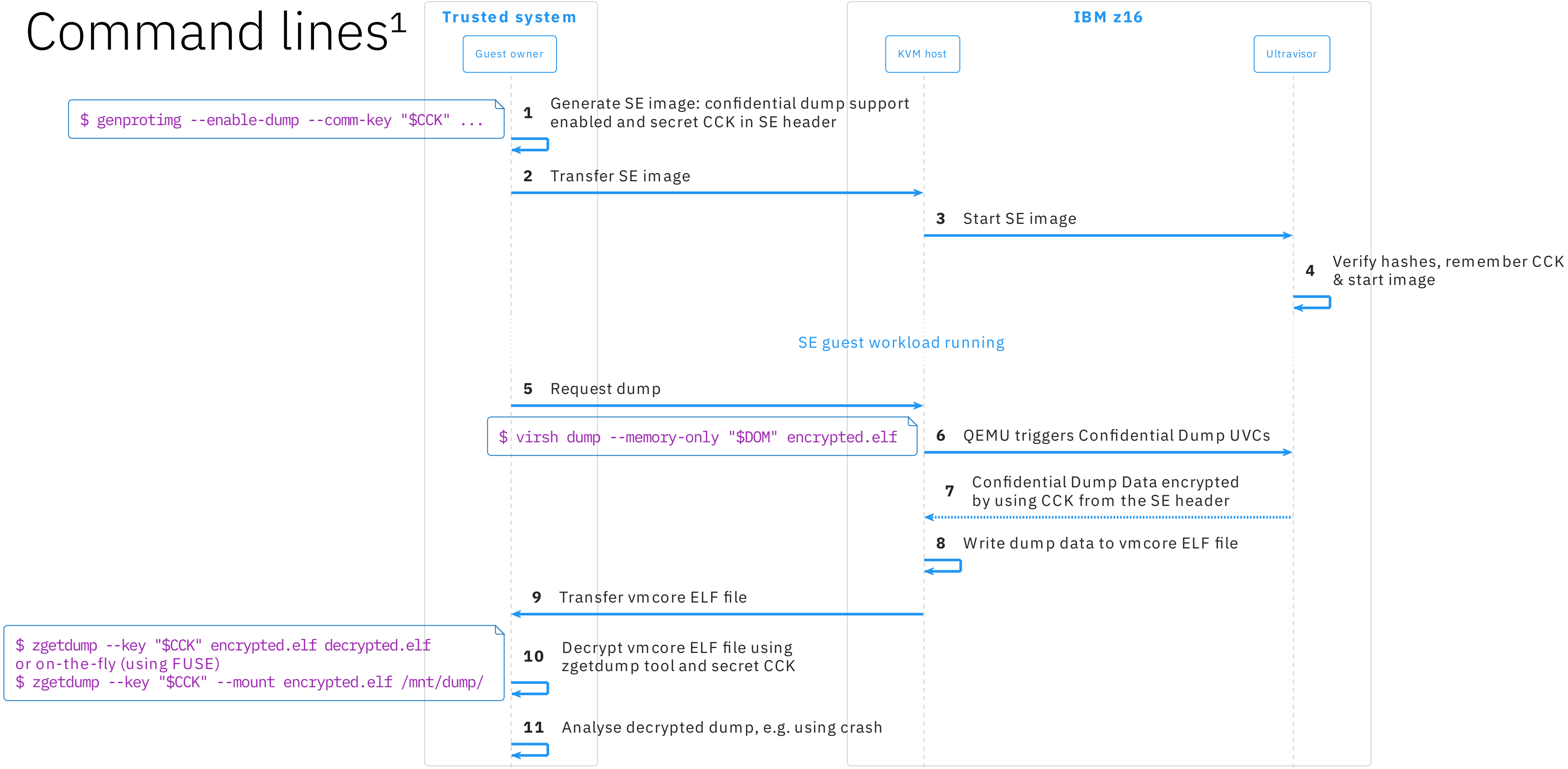
8 Write dump data to vmcore ELF file

9 Transfer vmcore ELF file

10 Decrypt vmcore ELF file using zgetdump tool and secret CCK

11 Analyse decrypted dump, e.g. using crash

Command lines[1]



Trusted system

Guest owner

IBM z16

KVM host

Ultravisor

```
$ genprotimg --enable-dump --comm-key "$CCK" ...
```

1 Generate SE image: confidential dump support enabled and secret CCK in SE header

2 Transfer SE image

3 Start SE image

4 Verify hashes, remember CCK & start image

SE guest workload running

5 Request dump

```
$ virsh dump --memory-only "$DOM" encrypted.elf
```

6 QEMU triggers Confidential Dump UVCs

7 Confidential Dump Data encrypted by using CCK from the SE header

8 Write dump data to vmcore ELF file

9 Transfer vmcore ELF file

```
$ zgetdump --key "$CCK" encrypted.elf decrypted.elf
or on-the-fly (using FUSE)
$ zgetdump --key "$CCK" --mount encrypted.elf /mnt/dump/
```

10 Decrypt vmcore ELF file using zgetdump tool and secret CCK

11 Analyse decrypted dump, e.g. using crash

[1] QEMU and zgetdump under review, can therefore change

Current state

Hardware:

IBM z16

Kernel:

v6.0-rc1

QEMU:

Under review[1]

Libvirt:

No changes – just works

genprotimg (s390-tools):

v2.21.0

zgetdump (s390-tools):

WIP

[1] https://lists.gnu.org/archive/html/qemu-devel/2022-08/msg01772.html

Summary

Attestation

Verify integrity of SE image instance

- Implicit Attestation on IBM Secure Execution
- Explicit Attestation after transition into SE mode
  - Identify specific image instance
  - Attest without revealing secrets

Confidential Dump

- Opt-in required by setting a SE-header flag
- Reliable and secure way for hypervisor initiated dumping
  - Actual guest state is encrypted
- No *QEMU Monitor Protocol* API changes[1]
  - → No changes in libvirt
- `zgetdump` tool will handle decryption
  - On-the-fly decryption using FUSE possible
  - Decrypted dump can be analysed, e.g. using `crash`

[1] Still under review, can therefore change

Thank you!

Steffen Eiden <seiden@linux.ibm.com>
Marc Hartmayer <marc@linux.ibm.com>
—

ibm.com

IBM