Confidential Computing with OpenBSD

Hans-Jörg Höxer

Confidential Computing with ~~OpenBSD~~ vmd(8)

Hans-Jörg Höxer

About

Hans-Jörg Höxer

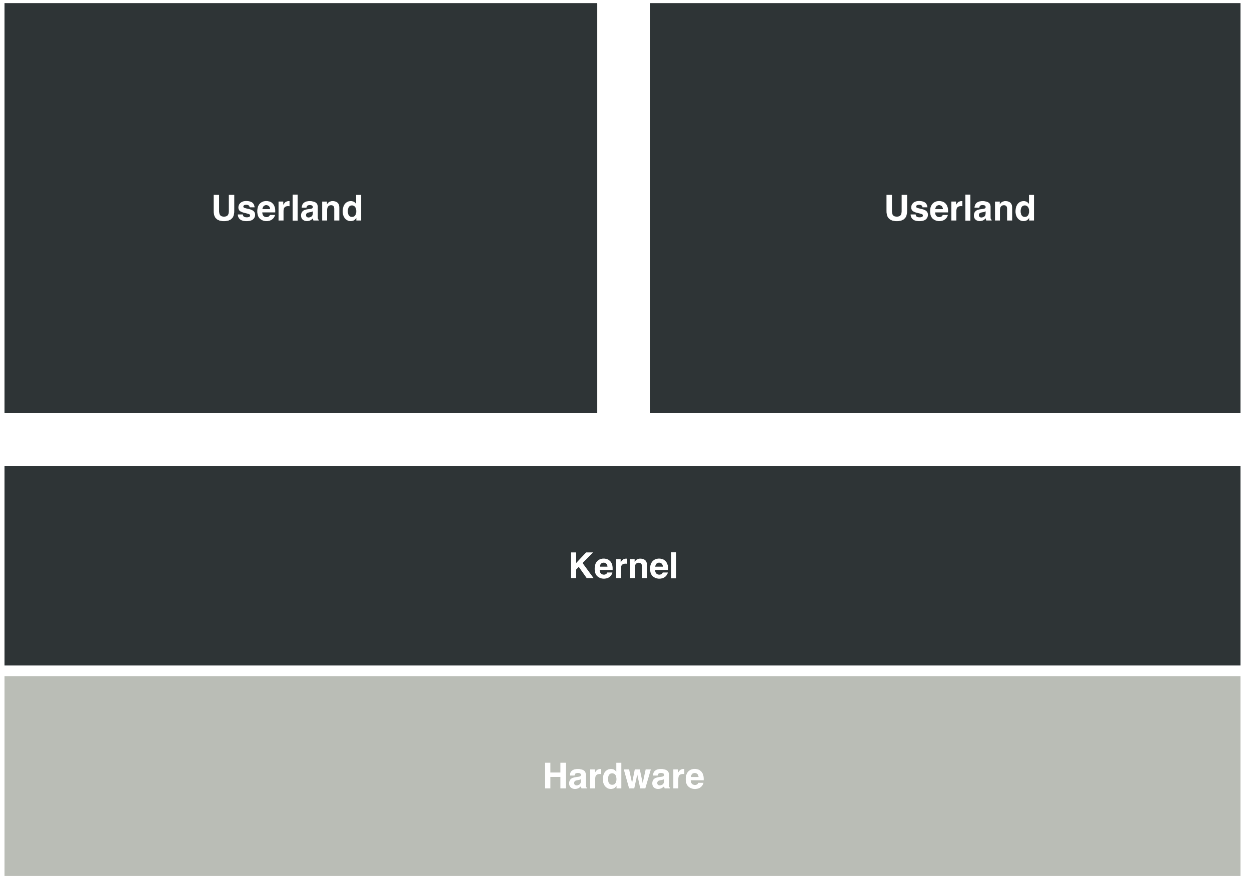
- Mid-2000s:
  - hshoexer@openbsd.org
- genua GmbH (www.genua.de):
  - hshoexer@genua.de
  - OpenBSD
  - Firewalls
  - VNP-Appliances

Confidential Computing

What is this all about?

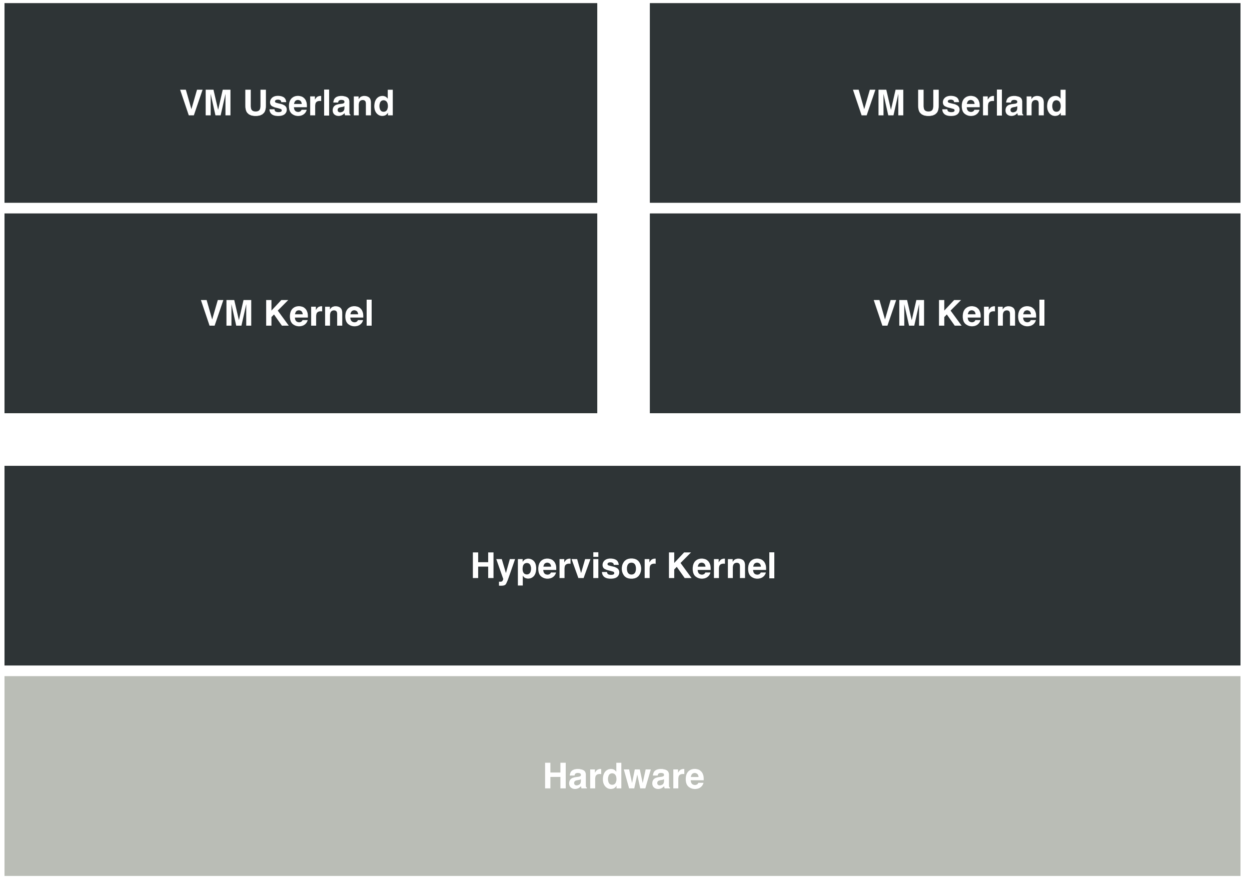
- Problem:
  - Sensitive data in an untrusted environment
- Supposed solution:
  - “Turn public cloud into private cloud”

Untrusted Environments



Generic OS

Untrusted Environments



VM Userland

VM Userland

VM Kernel

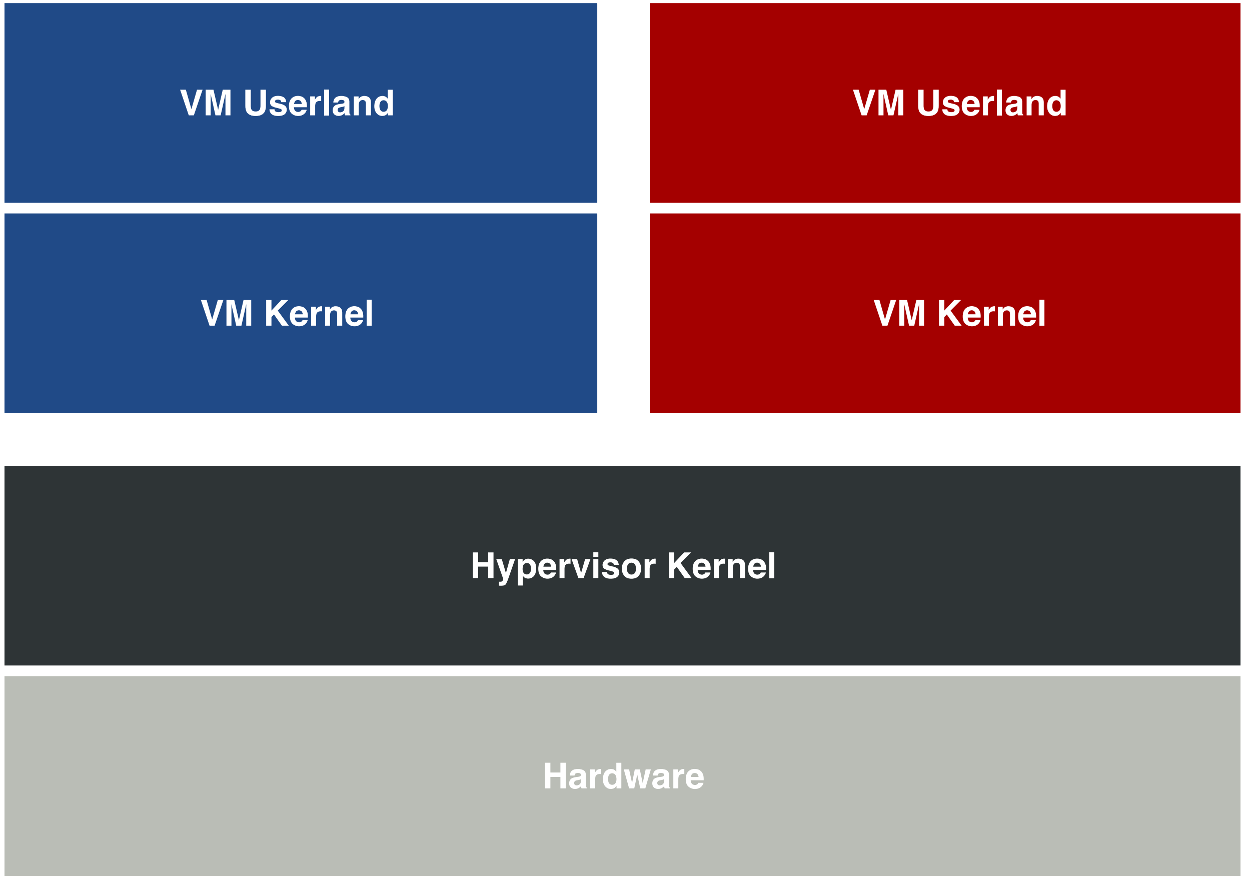
VM Kernel

Hypervisor Kernel

Hardware

Virtualisation

Untrusted Environments



VM Userland

VM Kernel

VM Userland

VM Kernel

Hypervisor Kernel

Hardware

Confidential VM

Confidential Computing

Claims

- Techniques to protect computing workload from its untrusted environment
  - Data confidentiality
  - Data integrity
  - Code integrity
- Isolation levels
  - Function or library isolation
  - Application isolation

  ⭐Virtual machine isolation

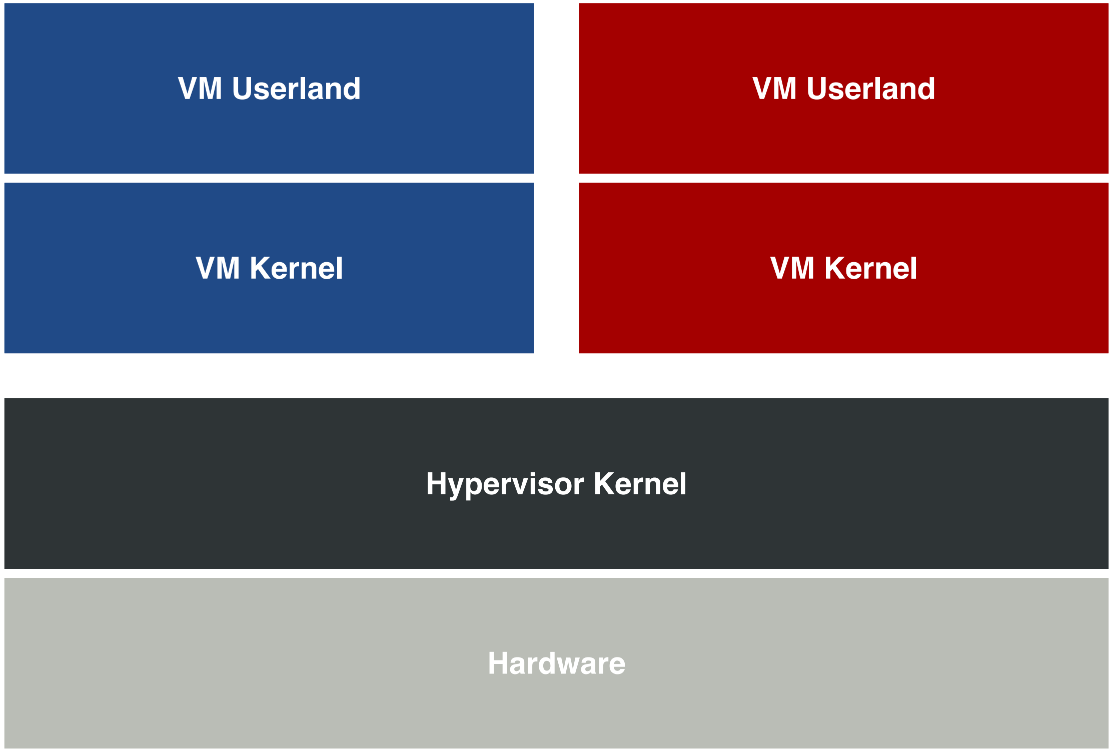
Confidential Computing

Hardware Support

- Hardware support:
  - ⭐Runtime encryption
  - Attestation
  - Strong isolation
- Examples:
  - **AMD SEV**, Intel TDX, Arm CCA (virtual machines)
  - Intel SGX (library, function)

AMD Secure Encrypted Virtualisation

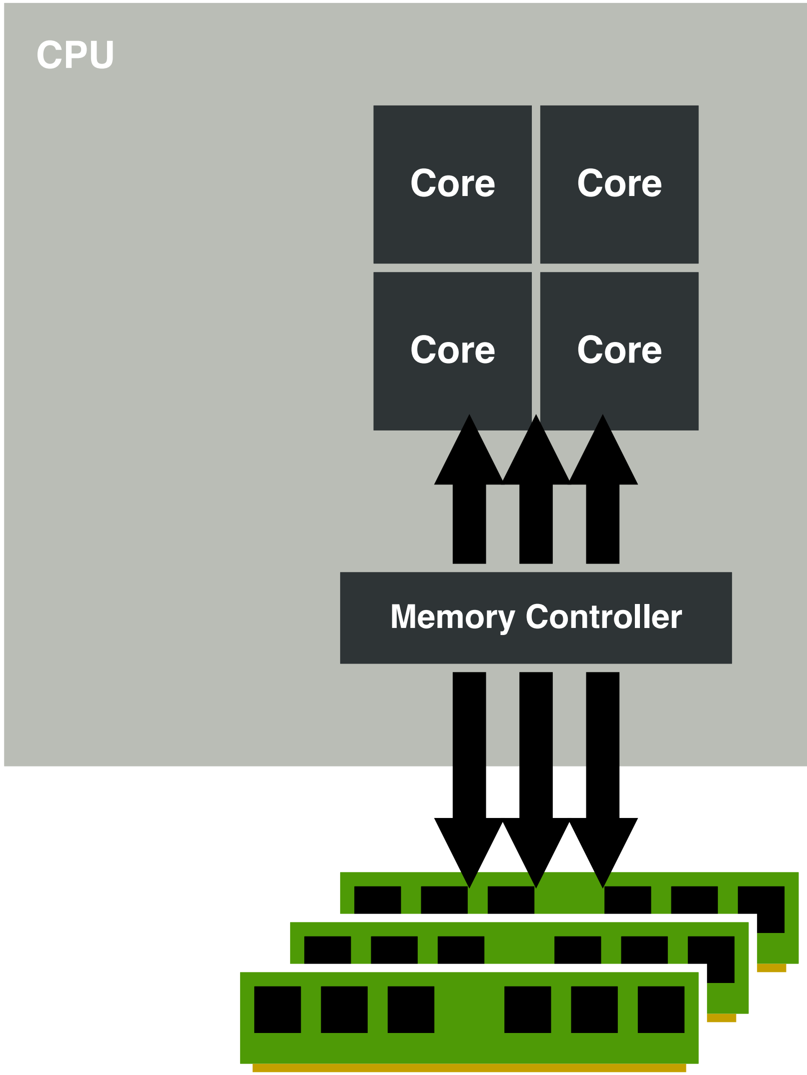
Confidential VM



Confidential VM

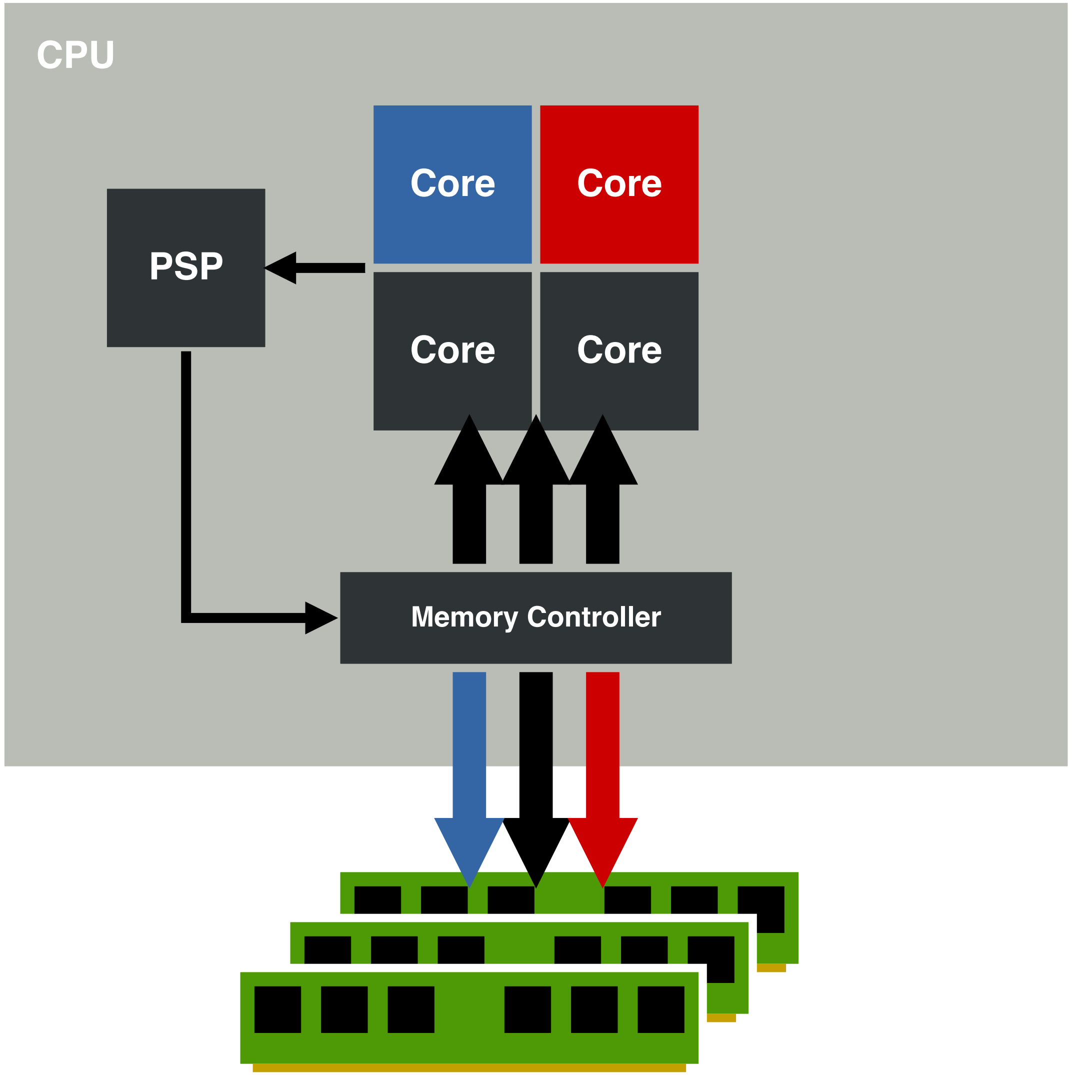
AMD SEV

Architecture



AMD SEV

Architecture



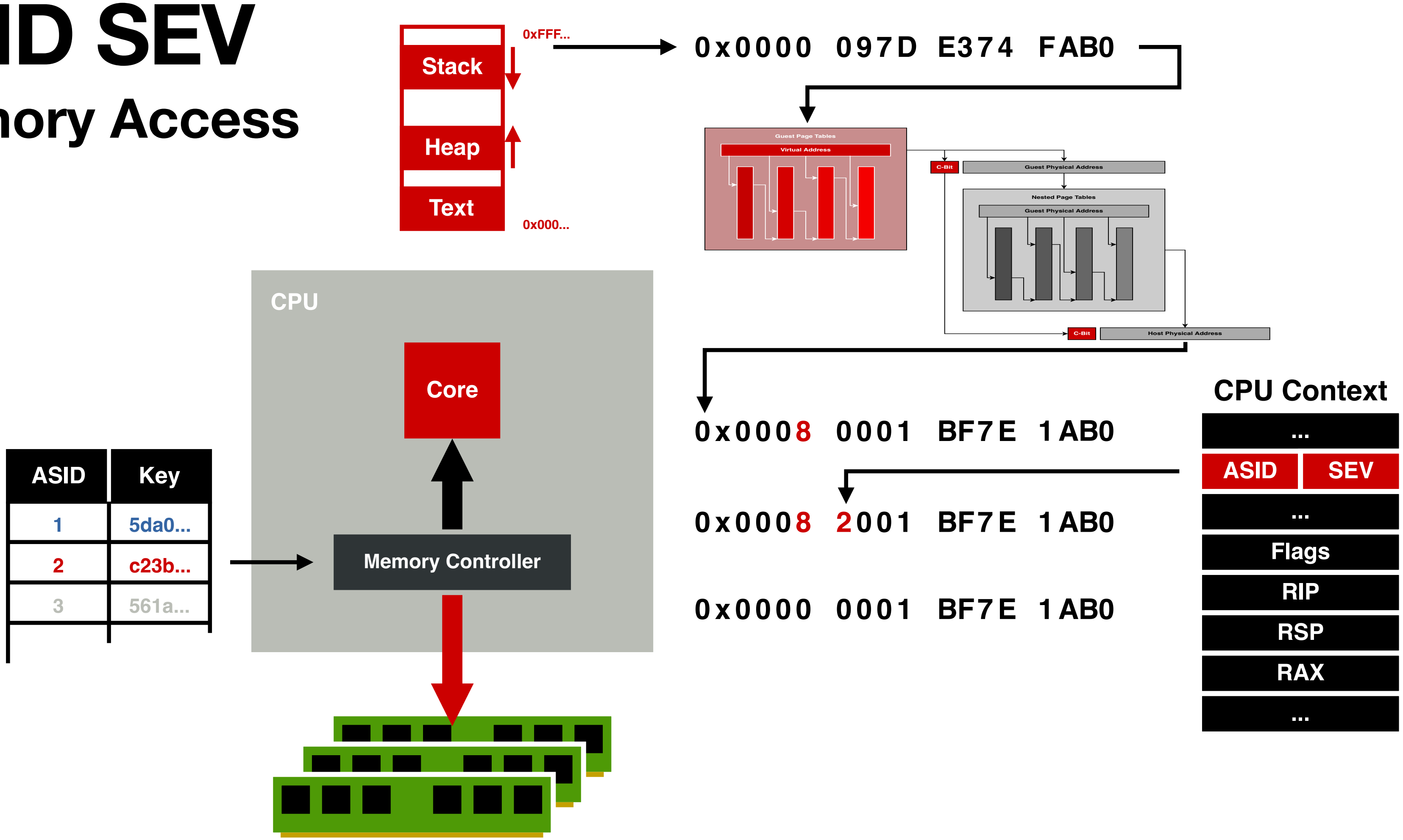
AMD SEV

Secure Encrypted Virtualisation

- Guest VM controls encryption!
  - Page tables:
    - “Crypt bit” (C-bit)
    - Private data
    - Public data — shareable
- Departure from x86 security model:
  - Hypervisor < Guest VM

AMD SEV

Memory Access



0x0000 097D E374 FAB0

0x0008 0001 BF7E 1AB0

0x0008 2001 BF7E 1AB0

0x0000 0001 BF7E 1AB0

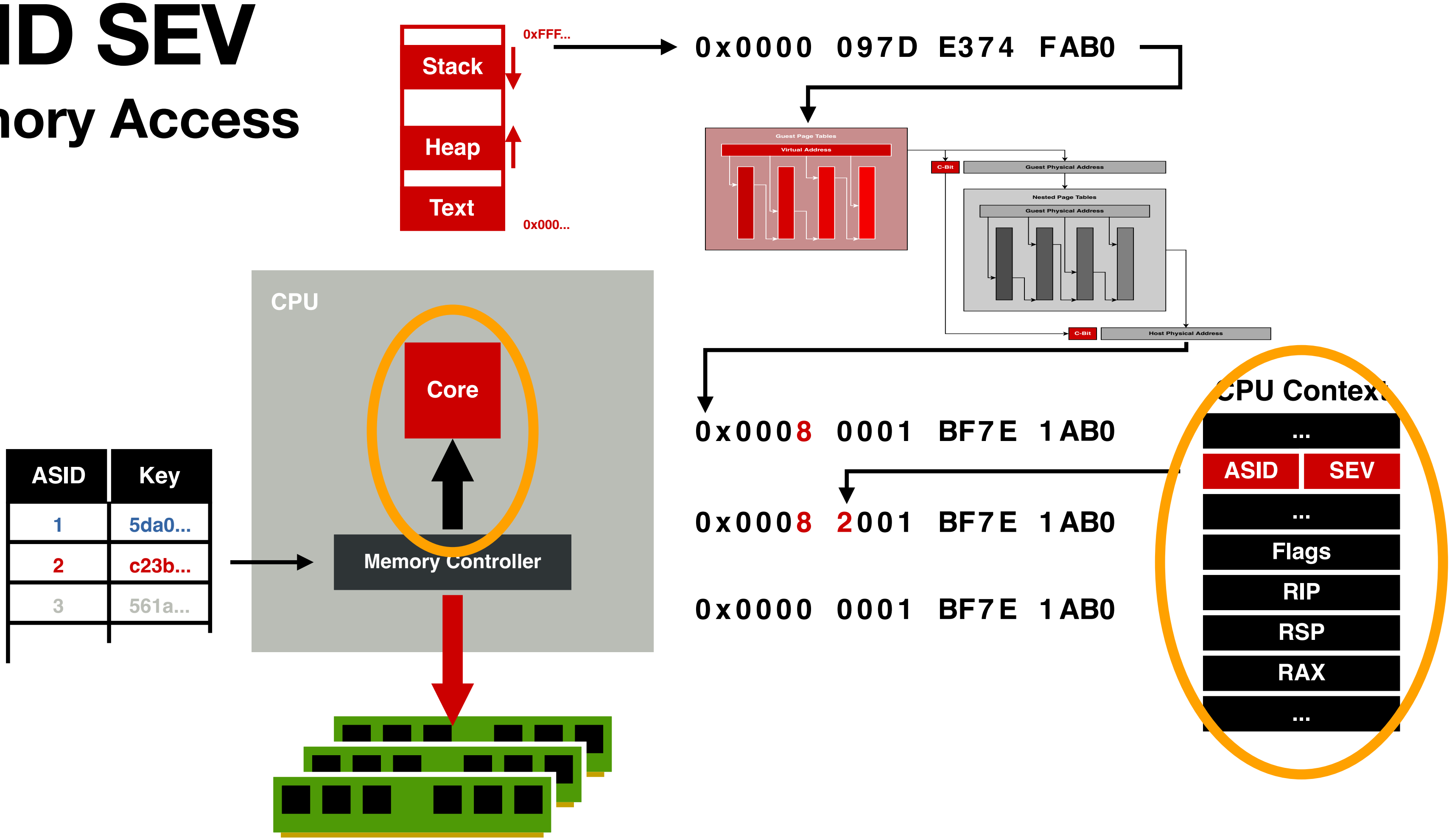
ASID	Key
1	5da0...
2	c23b...
3	561a...

CPU Context

...	
ASID	SEV
...	
Flags	
RIP	
RSP	
RAX	
...	

AMD SEV

Memory Access



Stack
Heap
Text
0xFFF...
0x000...

0x0000 097D E374 FAB0

Guest Page Tables
Virtual Address
C-Bit
Guest Physical Address
Nested Page Tables
Guest Physical Address
C-Bit
Host Physical Address

0x0008 0001 BF7E 1AB0

0x0008 2001 BF7E 1AB0

0x0000 0001 BF7E 1AB0

CPU Context
...
ASID
SEV
...
Flags
RIP
RSP
RAX
...

CPU
Core
Memory Controller

ASID	Key
1	5da0...
2	c23b...
3	561a...

AMD SEV

Limitations

- Limitations:
  - VCPU state visible to hypervisor
  - No integrity protection
  - Local attestation
- Solutions:
  - SEV-ES
  - SEV-SNP

AMD SEV

Security

- AMD-SB-3011 Guest memory vulnerabilities:
    - CVE-2024-21978, CVE-2024-21980, CVE-2023-31355
- Attacks on PSP:
    - Buhren, Krachenfels, Jacob, Seifert, 2021, “One Glitch to Rule Them All: Fault Injection Attacks Against AMD’s Secure Encrypted Virtualization”
    - Buhren, Werling, Seifert, 2019, “Insecure Until Proven Updated: Analysing AMD’s SEV Remote Attestation”
- ¯_(ツ)_/¯

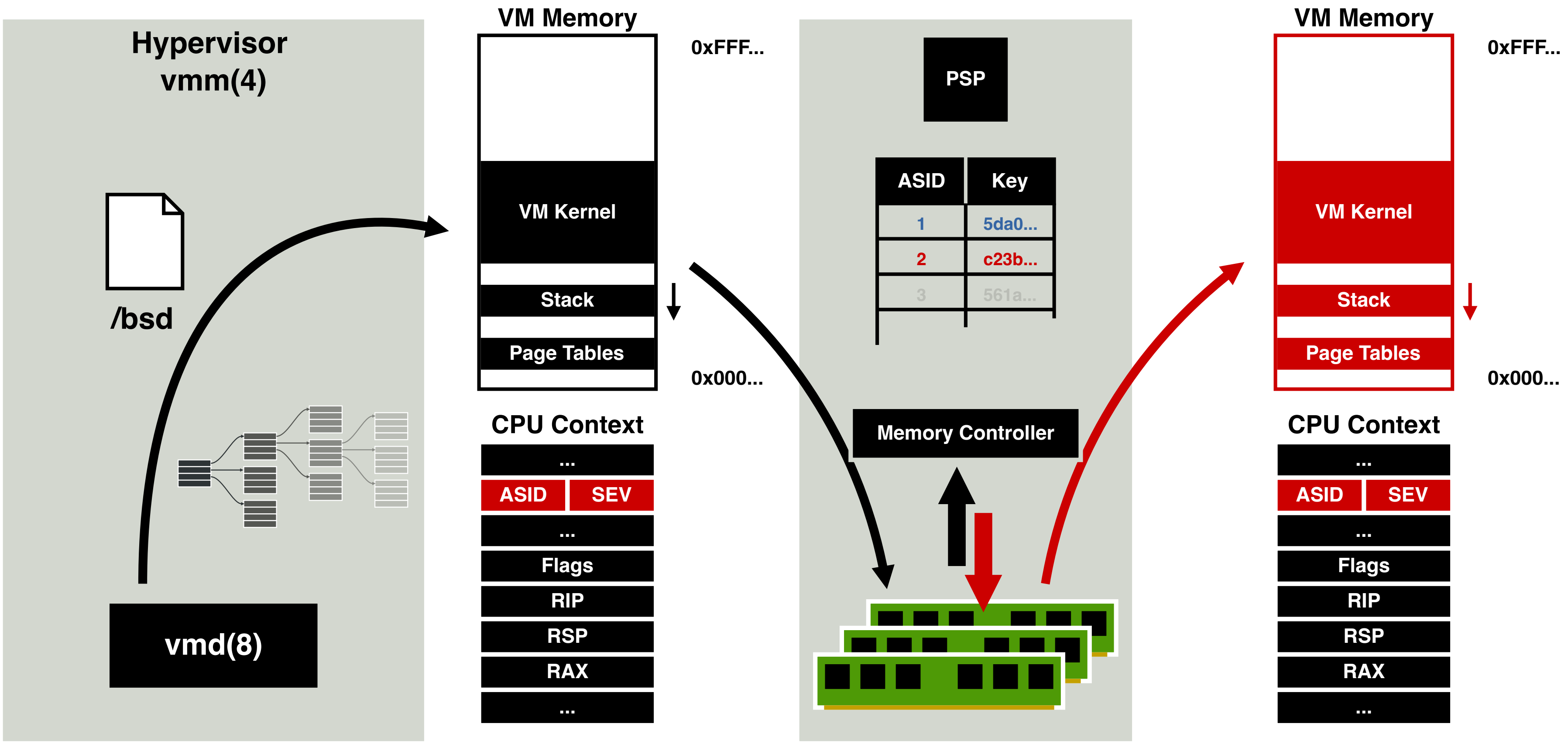
OpenBSD

Confidential VM

- Personal goal:
    - Learn about Confidential Computing
- OpenBSD as research/learn platform:
    - vmd(8)
    - vmm(4)
    - Run confidential OpenBSD guest on OpenBSD host

➡As simple as possible

The big picture



Hypervisor
vmm(4)

/bsd

vmd(8)

VM Memory

0xFFF...

VM Kernel

Stack

Page Tables

0x000...

CPU Context

...

ASID SEV

...

Flags

RIP

RSP

RAX

...

PSP

ASID	Key
1	5da0...
2	c23b...
3	561a...

Memory Controller

VM Memory

0xFFF...

VM Kernel

Stack

Page Tables

0x000...

CPU Context

...

ASID SEV

...

Flags

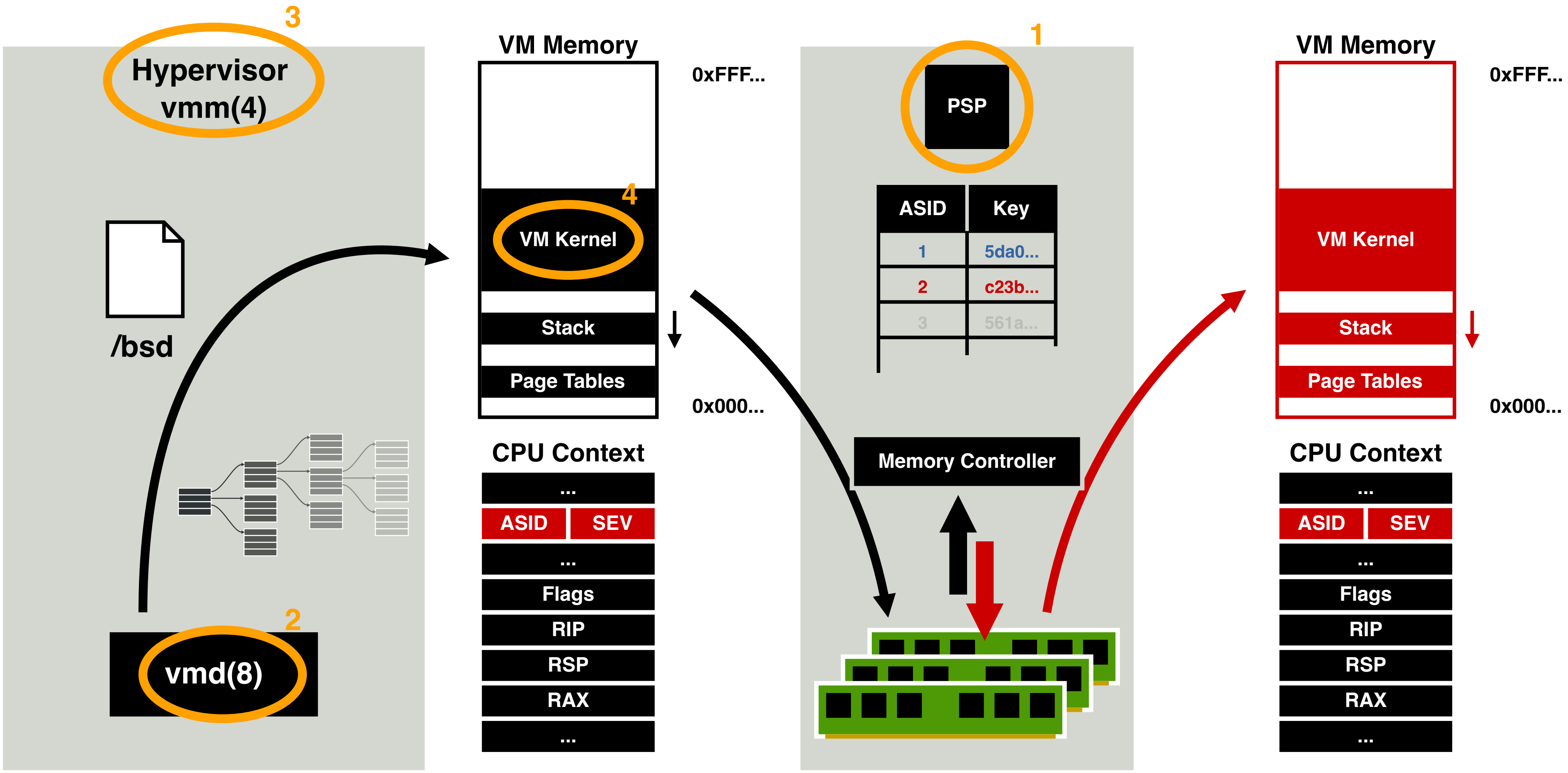
RIP

RSP

RAX

...

The big picture



ASID	Key
1	5da0...
2	c23b...
3	561a...

How to start?

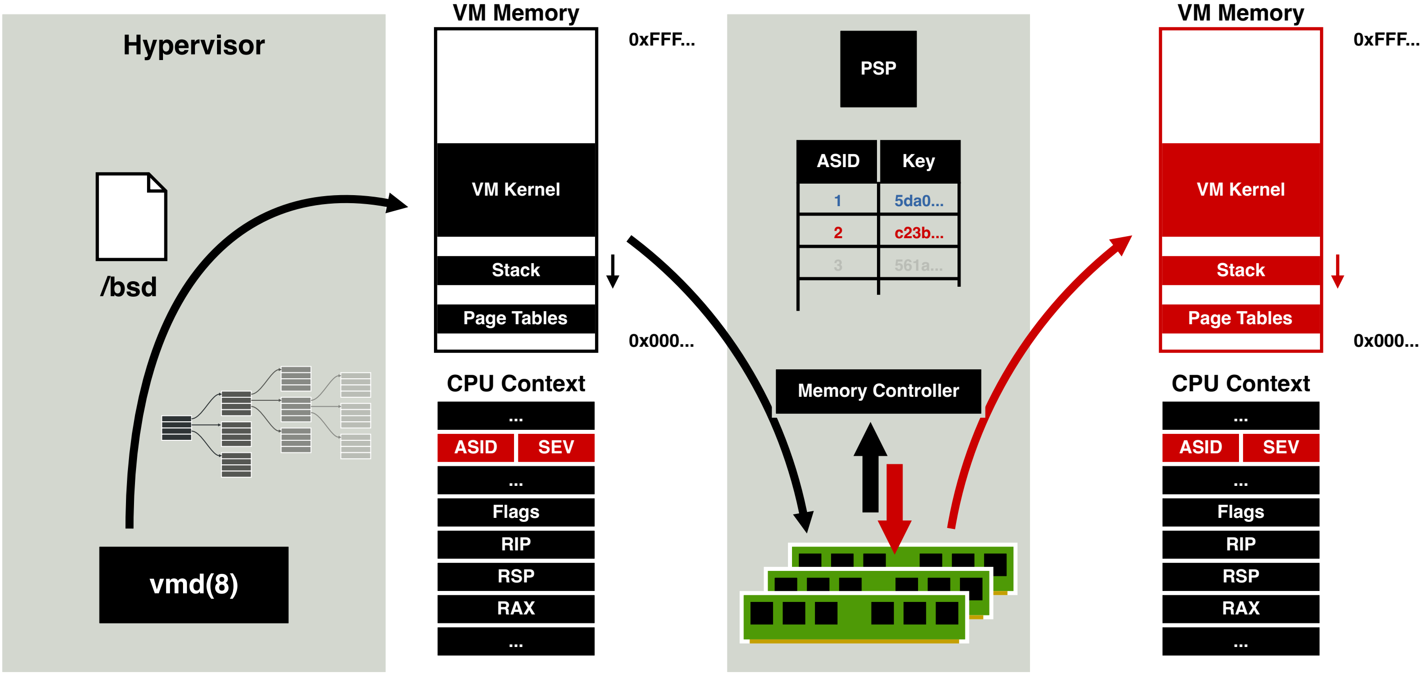
The plan — Simplicity first

- bsd.rd single-user as guest
- Fully encrypted
- No DMA, no virtio(4)
- Only IN/OUT instructions:
  - PIT i8253, RTC mc146818, PIC i8259, UART ns8250
- Hardcode everything — C-bit
- 12/2023

Round One

Minimal psp(4) support

- Mailbox interface
- Simple commands:
  - INIT
  - PLATFORM_STATUS
- Launch protocol:
  - LAUNCH_START
  - LAUNCH_UPDATE_DATA
  - LAUNCH_MEASURE
  - LAUNCH_FINISH
- Some more



ASID	Key
1	5da0...
2	c23b...
3	561a...

Round One

*

Minimal psp(4) support

- LAUNCH_UPDATE_DATA:
  - vmd(8) provides virtual address
  - psp(4) wires mapping (uvm_map_pageable(9))
  - Converts to physical address (pmap_extract(9))
  - PSP encrypts

Round One

Minimal vmd(8) and vmm(4) support

- vmd(8):
  - Only “direct kernel exec”
  - Page tables use predefined PG_CRYPT
  - Encrypt memory — psp(4)
- vmm(4):
  - Set SEV enable flag in VMCB

Round One

Guest kernel bsd.rd

- Hard code:
  - PG_CRYPT 0x0008 0000 0000 0000 (bit 51)
  - PG_FRAME 0x0007 FFFF FFFF F000
  - Initial page tables in locore
  - pmap(9)

➡bsd.rd boots single-user

- ~2 months (12/2023 to 01/2024)

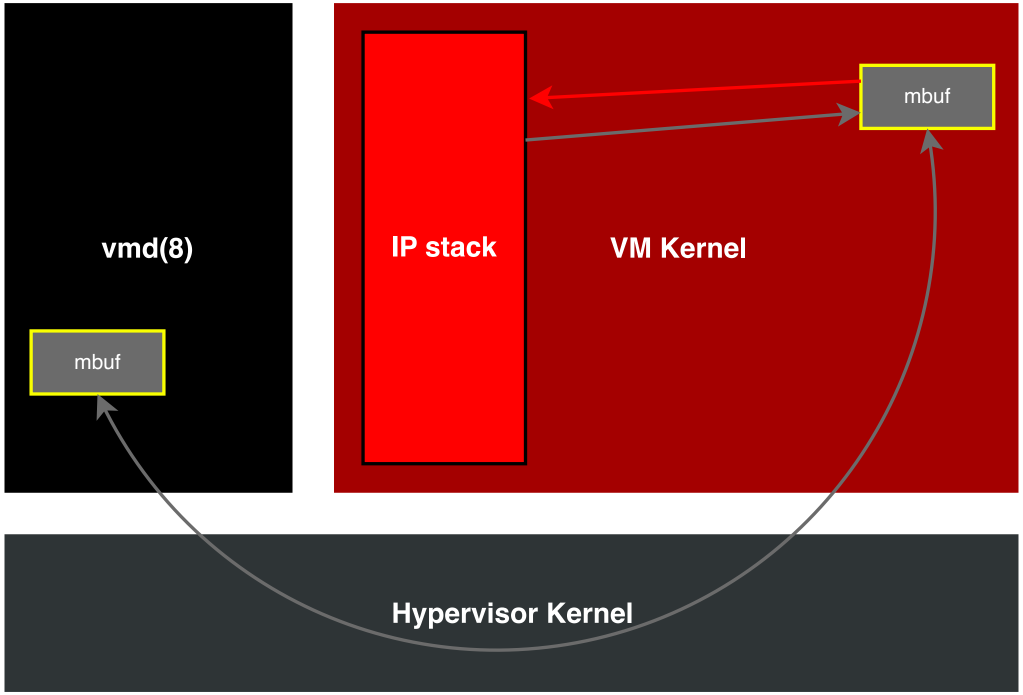
Round Two

The GENERIC kernel

- locore:
  - Detect SEV guest mode
  - C-bit position
  - Physical bit reduction
  - Configure pg_crypt and pg_frame — similar to pg_nx
- pmap(9)
  - Use pg_crypt
  - Use pg_frame instead of PG_FRAME

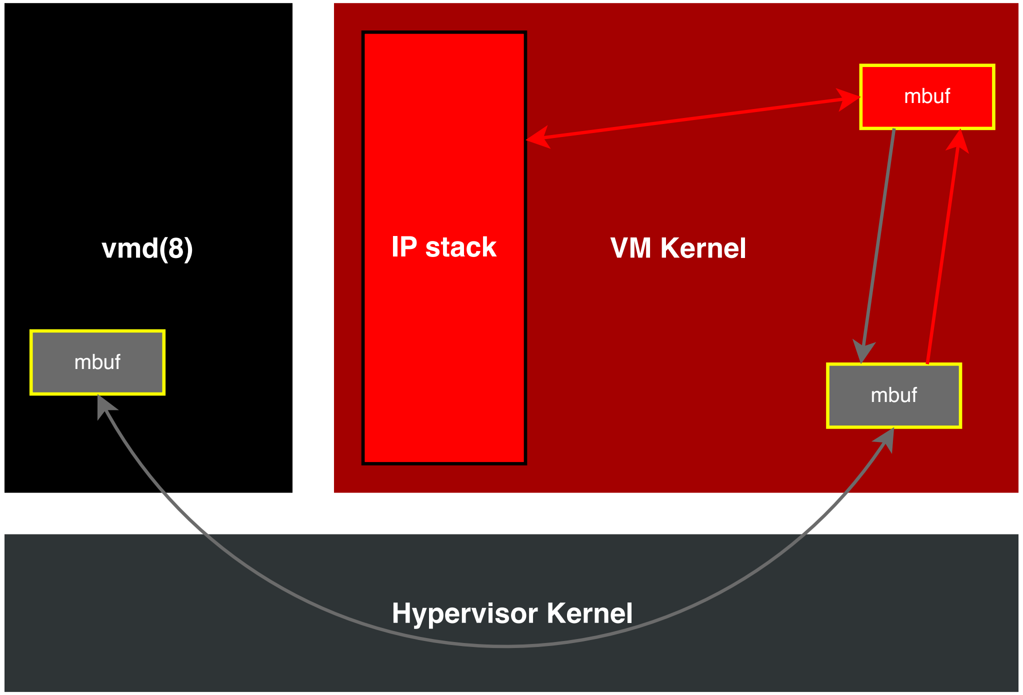
Round Two

DMA for virtio(4) — bounce buffers



Round Two

DMA for virtio(4) — bounce buffers



Round Two

bus_dma(9)

```
for each DMA xfer {
        bus_dmamem_alloc();  /* allocate some DMA'able memory        */
        bus_dmamem_map();    /* map it into the kernel address space */

        bus_dmamap_load();   /* initialize the segments of dmamap    */
        bus_dmamap_sync();   /* synchronize/flush any DMA cache      */

        for (i = 0; i < dm_nsegs; i++) {
                /* Start the DMA, wait until it's done */
        }

        bus_dmamap_sync();   /* synchronize/flush any DMA cache      */
        bus_dmamap_unload(); /* prepare dmamap for reuse             */

        bus_dmamem_unmap();  /* free kernel virtual address space    */
        bus_dmamem_free();   /* free DMA'able memory                 */
}
```

Round Two

bus_dma(9)

- bus_dma_segment_t:

```
struct bus_dma_segment {
  bus_addr_t      ds_addr;          /* DMA address */
  bus_size_t      ds_len;           /* length of transfer */
  …
};
```

Round Two *

bus_dma(9)

- bus_dma_segment_t:

```
struct bus_dma_segment {
  bus_addr_t      ds_addr;         /* DMA address */
  bus_size_t      ds_len;          /* length of transfer */
  vaddr_t         _ds_va;          /* mapped loaded data */
  vaddr_t         _ds_bounce_va;   /* mapped bounced data */
  …
};
```

Round Two

*

bus_dma(9)

- bus_dmamap_create(9):
    - Allocates DMA segments
    - Allocate bounce buffers
    - Map with PMAP_NOCRYPT
- bus_dmamem_map(9):
    - Map into kernel address space
- bus_dmamap_load_*(9):
    - Set _ds_va and _ds_bounce_va
    - Set ds_addr to bounce buffer
- bus_dmamap_sync(9):
    - bcopy() from/to _ds_va and _ds_bounce_va

Round Two

Improve initial guest kernel load

- vmd(8) only encrypts:
  - ELF kernel image
  - Page tables
  - GDT
  - Initial stack
  - Boot arguments
  - Initial random seed

Round Two

Self-hosting Confidential VM

- Same kernel for host and guest!
- Confidential VM works :-)
  - 05/2024
- …almost :-(
  - vio(4) stalls
  - vioblk(4) crashes (during make build)

Round Three

Thank god, it’s open source!

- virtio(4) debugging and fixing by sf@
- bus_dma(9) bounce buffer debugging and testing by bluhm@
- psp(4) <-> ccp(4) cleanup jsg@
- Input mlarkin@, dv@, kettenis@, dlg@
- Getting stuff committed by bluhm@

➡Stable SEV enabled guest VM on OpenBSD hypervisor

  - make build survives
  - ~09/2024

Does SEV actually work?

The heat is on…

- Dump memory (RAM)
- Measure compressibility per page
- Plot heat map

hexdump

Warm boot marker

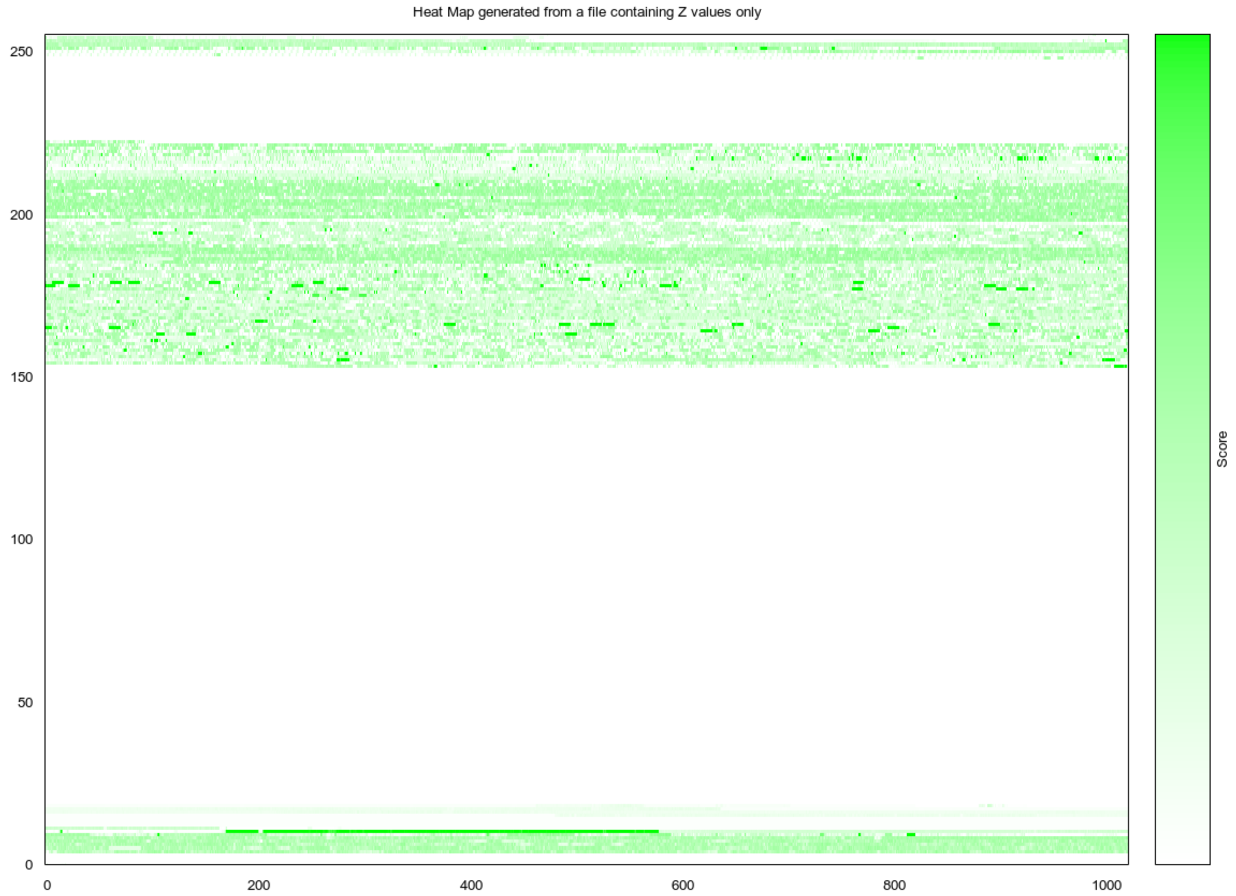
- Without SEV

```
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000470  00 00 34 12 00 00 00 00  00 00 00 00 00 00 00 00  |..4.............|
00000480  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
```

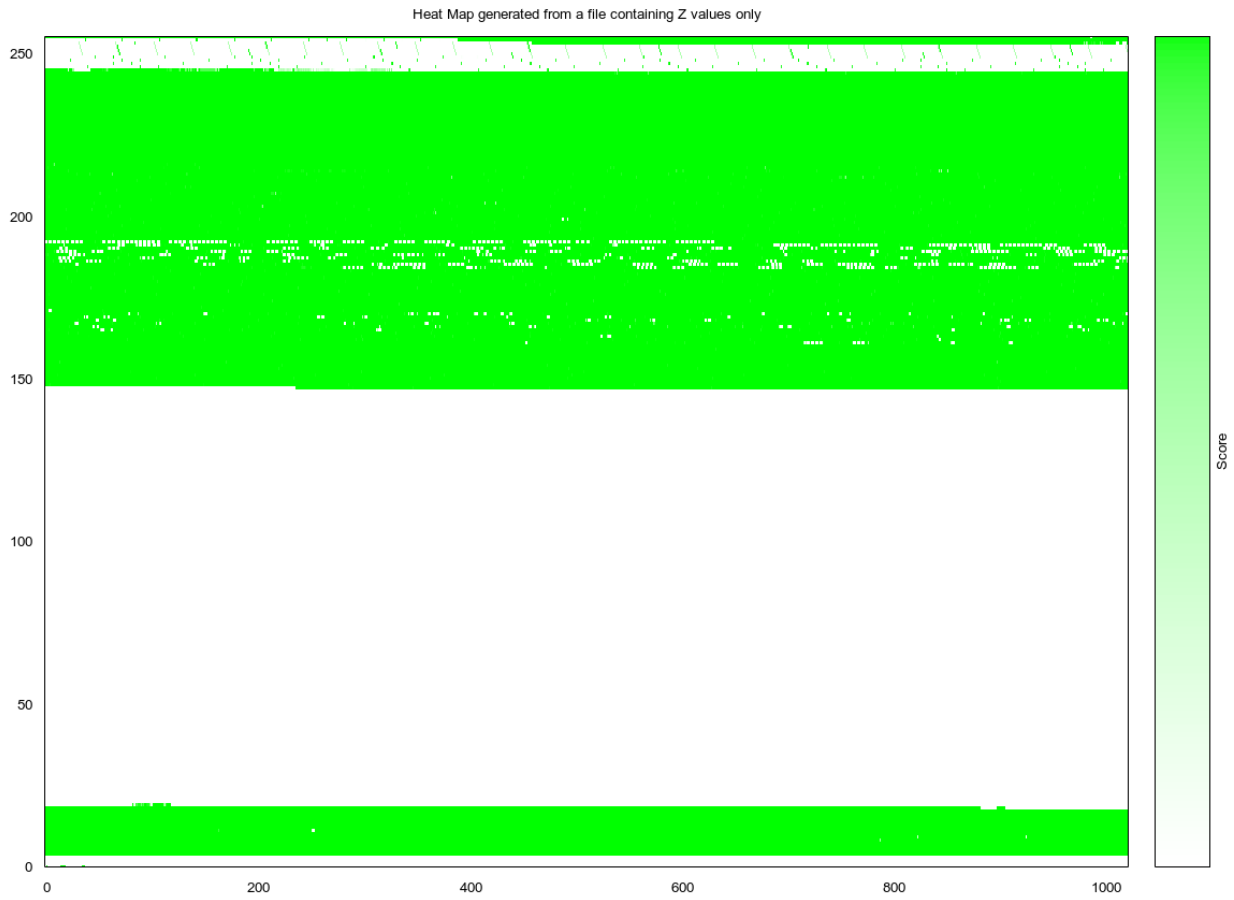
- With SEV

```
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000470  28 30 32 e7 6e d3 4f 45  08 7e 3d 6f dc 71 71 22  |(02.n.OE.~=o.qq"|
00000480  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
```

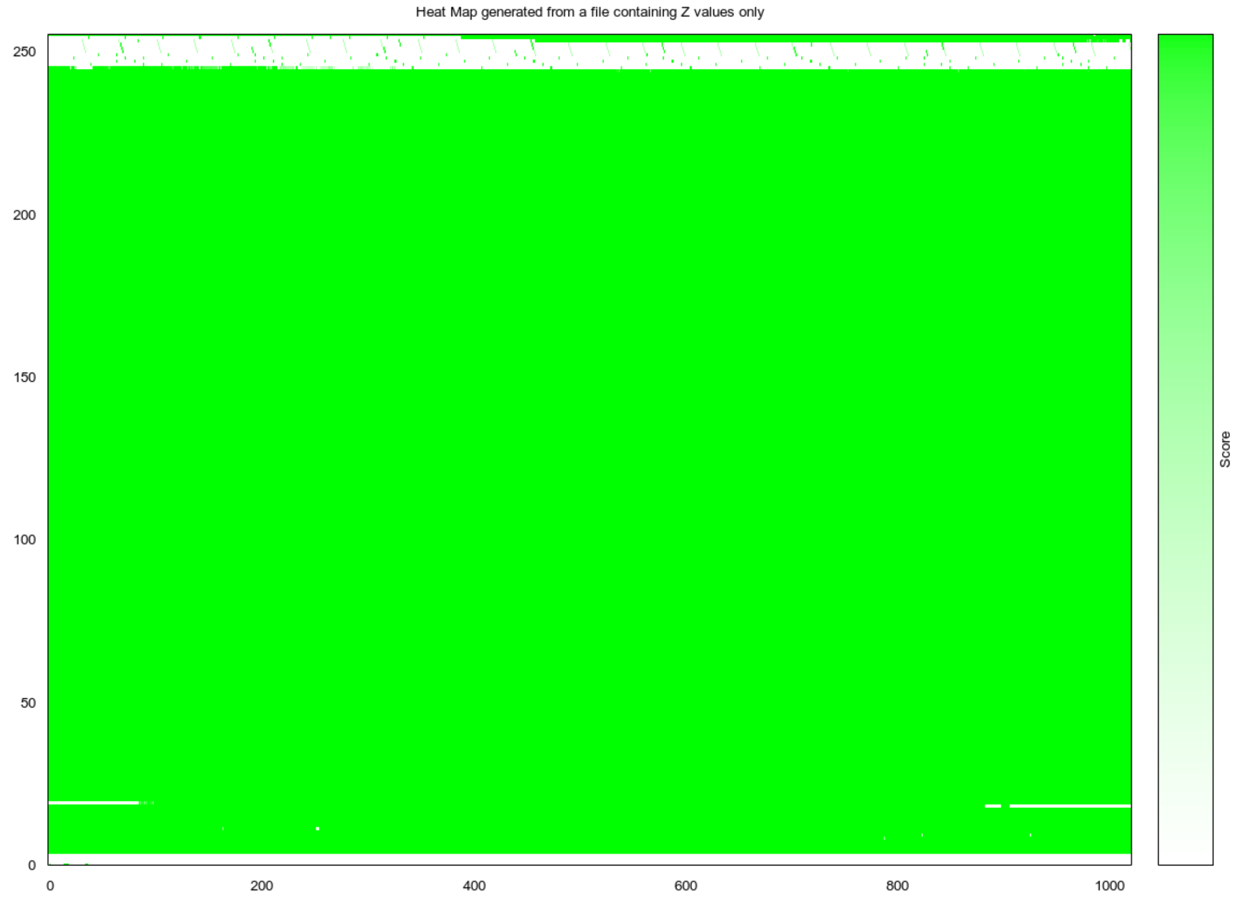
No encryption



SEV enabled


Heat Map generated from a file containing Z values only
250
200
150
100
50
0
0
200
400
600
800
1000
Score

“Page zero hack”



Conclusion

It’s a long way home

- Accomplished:
  - ⭐SEV enabled OpenBSD guest on OpenBSD host
- Next steps:
  - SEV-ES:
    - Already in progress
    - Compatibility with KVM/qemu
  - Fix all the bugs
  - Optimize DMA
  - Performance?
  - Attestation?
  - …

Thanks!

- genua:
  - Mia Teschauer
  - Jan Klemkow (jan@)
  - Alexander Bluhm (bluhm@)
  - Stefan Fritsch (sf@)
- tech@openbsd:
  - mlarkin@, dv@, dlg@, kettenis@, jsg@, Hrvoje Popvski, …

Questions?

Don’t forget to remember!