Confidential Computing with AMD SEV-SNP

Brijesh Singh
8/31/21
**INTRODUCING SEV-SNP**

- **Secure Nested Paging** (SEV-SNP) is the latest generation of AMD Secure Encrypted Virtualization (SEV) technology designed for Confidential Computing.

- SEV-SNP builds on existing AMD SEV and AMD SEV-ES (Encrypted State) features to provide *stronger security, additional use models, and more* to protected VMs:
  - SEV and SEV-ES supported in 1\textsuperscript{st} and 2\textsuperscript{nd} generation AMD EPYC Processors (2017)
  - SEV-SNP supported starting in 3\textsuperscript{rd} generation AMD EPYC Processors (2021)

- SEV-SNP is designed to protect a VM from a malicious hypervisor in specific ways:
  - Useful in public cloud and any scenario where the hosting environment cannot be trusted.

---

**SEV**
- Memory confidentiality

**SEV-ES**
- Register confidentiality

**SEV-SNP**
- New integrity protection and more
SEV-SNP OVERVIEW

THREAT MODEL

SEV-SNP is designed to protect the VM in specific ways:
- **Confidentiality** – Prevent hypervisor from reading guest data
- **Integrity** – Prevent hypervisor from modifying/replaying guest data
- **Physical Access** – Prevent “offline” physical attacks (e.g., cold-boot)
- **Interrupt Control** – Prevent malicious interrupt injection
- **CPUID** – Prevent hypervisor from lying about HW capabilities
- **Certain Side Channels** – Prevent certain speculative side channel attacks

SEV-SNP does not protect against certain attack vectors, including:
- **Availability** – Hypervisor retains control of resource allocation and scheduling
- **Advanced Physical Attacks** – Attacking voltage/data buses while system is running
- **Certain Side Channels** – Including PRIME+PROBE, page fault side channels, etc.

SEV-SNP security is enforced via a combination of hardware and guest software.
Memory integrity is enforced using a new DRAM structure called the Reverse Map Table (RMP)

There is 1 RMP for the entire system, it is created by software during boot

Basic properties:
- RMP contains 1 entry for every 4k of assignable memory
- RMP is indexed by System Physical Address (SPA)
- RMP entries may only be manipulated via new x86 instructions

The RMP indicates page ownership and dictates write-ability. Examples:
- A page assigned to a guest is only writeable by that guest
- A page assigned to the hypervisor cannot be used as a private (encrypted) guest page
- A page used by AMD firmware cannot be written by any x86 software
RMP CHECKS

The RMP directly protects against:
- Data corruption/replay (only assigned guest can write to a page)
- Memory aliasing (one page can only be mapped to one guest at a time)

RMP is checked on:
- **Writes in any mode**
- **Reads from SEV-SNP guests**

The RMP is not checked on reads in certain modes (e.g., HV mode) because memory encryption ensures confidentiality.
RMP VIOLATION FAULT (HOST)

Host RMP fault handler strategy
- Unmap the guest private pages from the direct map to avoid the RMP violation for the kernel addresses.
- User space write to guest private raise SIGBUS.

Host backing page support strategy
- Keep the host and RMP levels in sync either by splitting the large page or smashing the large RMP entry into multiple of 4K.

Example:
1. VMM allocates guest RAM backing memory from large page.
2. Guest issues a PSC to mark a region as 2MB private in the RMP table.
3. Guest later issues another PSC to make one of the subpages shared.
4. VMM attempts to write to the shared page.
   - The write access will cause #PF due the page size mismatch
5. To resolve the fault, the host page fault handler split the backing pages into 4K.
RMP VIOLATION FAULT (VM)

All the guest memory access go through the RMP checks.

- #NPF is extended to provide cause of an RMP violation
  - BIT 31 (RMP) is set if the fault was due to RMP check
  - BIT 33 (ENC) is set if the guest C-bit is 1, 0 otherwise
  - BIT 34 (SIZEM) is set if the fault was due to the size mismatch on PVALIDATE or RMPADJUST
  - BIT 35 (VMPL) is set if the fault was due to the VMPL check failure.

<table>
<thead>
<tr>
<th>C-Bit</th>
<th>Type of Access</th>
<th>Check</th>
<th>RMP Fault Handler Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Instruction fetch</td>
<td>Page is private</td>
<td>RMPUPDATE to mark page private</td>
</tr>
<tr>
<td></td>
<td>Page table access</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Data write</td>
<td>Page is private</td>
<td>RMPUPDATE to mark page private</td>
</tr>
<tr>
<td>0</td>
<td>Data write</td>
<td>Page is shared</td>
<td>RMPUPDATE to mark page shared</td>
</tr>
</tbody>
</table>
GHCB V2 CHANGES

SNP specific new VMGEXITs (spec link: developer.amd.com/sev)

- GHCB GPA Register
  - Some hypervisors may prefer that a guest use a consistent or specific GPA for the GHCB associated with vCPU
- Page State Change(PSC)
  - Allows guest to request page state changes using the GHCB protocol.
- Hypervisor feature query
  - Allows guest to query whether the hypervisor supports the SNP feature.
- Guest message request
  - Allows guest to send messages such as attestation report etc. to AMD-SP using the GHCB protocol.
- AP Creation
  - Allows guest to create or destroy or change the register state of AP using the GHCB protocol
- #HV doorbell page
  - Allows guest to register a doorbell page for use with the hypervisor injection exception.
- #HV IPI
  - Allows guest to send IPI to other vCPUs in the guest when the restricted injection feature is enabled.
- #HV timer
  - Allows guest to request timer support from the hypervisor when the restricted injection feature is enabled.
PAGE VALIDATION

SEV-SNP requires that private pages must be validated before the access.

A typical page validation flow:

1. Guest issues a PSC VMGEXIT.
   - Multiple PSC requests can be batched.
   - PSC VMGEXIT takes a RMP page size hint
2. Hypervisor handles the PSC VMGEXIT
   - Try to keep the NPT and RMP page level in sync.
   - Uses the RMPUPDATE to add/remove page from the RMP table.
3. Hypervisor resumes the guest.
4. Guest calls PVALIDATE to validate the page in the RMP table.

```
struct psc_hdr {
    u16 cur_entry;
    u16 end_entry;
};

struct psc_entry {
    u64 cur_page: 12,
    gfn: 40,
    op: 4,
    pagesize:1,
    rsvd: 7
};

struct snp_psc_desc {
    struct psc_hdr hdr;
    struct psc_entry entry[253];
};
```
PAGE VALIDATION OPTIONS...

- **Pre-validate (current)**
  - Guest BIOS validates the entire system RAM on boot

- **Lazy Validate (future)**
  - Guest BIOS validates the memory used by itself.
  - Guest BIOS published invalid memory region through newly added “Unaccepted” memory type..
  - Guest OS validate the remaining memory by going through the EFI memory map. It can validate on-demand or run a thread in background.
  - Guest OS can maintain of validated region and pass it to the kexec’ed kernel to avoid the double validation.
1. **Host OS initializes AMD Secure Processor (AMD-SP)**
   - AMD-SP generates random memory key (VEK)
   - Host OS selects key slot in Memory Controller

2. **Host OS allocates & initializes image memory**
   - Host OS places initial image into DRAM
   - AMD-SP reads memory, writes back out with VEK

- **Image memory consists of**
  - Initial guest BIOS (OVMF)
  - Initial CPU register state
  - Special information

- **Hypervisor flow:**
  - `SNP_GCTX_CREATE` – Create guest context
  - `ACTIVATE` – Assigned ASID
  - `SNP_LAUNCH_START` – Start launch context
  - `SNP_LAUNCH_UPDATE` (multiple) – Add page(s) to launch image
  - `SNP_LAUNCH_FINISH` – Close launch context, make guest runnable
TYPES OF GUEST PAGES (SEE SNP_LAUNCH_UPDATE)

- **PAGE_TYPE_NORMAL**
  - Standard data or instruction page. Contents and metadata included in Launch Measurement

- **PAGE_TYPE_VMSA**
  - Virtual Machine Save Area page. Contents and metadata included in Launch Measurement

- **PAGE_TYPE_ZERO**
  - Page of 0’s. Identical to PAGE_TYPE_NORMAL with a zero’d page

- **PAGE_TYPE_UNMEASURED**
  - Unmeasured (but encrypted) page. Can be used to pass information from the Hypervisor
  - Only metadata measured

- **PAGE_TYPE_SECRETs**
  - Special page used to hold AMD-SP provided keys and other information.
  - Only metadata measured

- **PAGE_TYPE_CPUID**
  - Special page used to provide secure CPUID information
  - Only metadata measured
VM MANAGEMENT COMMANDS

❑ New commands to create and manage SEV-SNP VMs
  - SNP_INIT
  - SNP_LAUNCH_START
  - SNP_LAUNCH_UPDATE
  - SNP_LAUNCH_FINISH
  - SNP_GUEST_REQ_{SET,GET}_RATE_LIMIT

❑ New object in Qemu to launch the SEV-SNP VM
  - $QEMU_CLI -object sev-snp-guest,id=sev0,policy=0x3 ...

❑ New host commands to query and control the system-wide configuration
  - SNP_PLATFORM_STATUS – Query the platform information through the AMD-SP (firmware)
  - SNP_{SET,GET}_CONFIG – Set or Get the certificate blob provided during the attestation report and reported TCB version etc
New driver (coco/sevguest.ko)
- The character device “/dev/sev-guest”
- IOCTLs to query attestation report and key derivation
  - SNP_GET_REPORT - Query the attestation report.
  - SNP_GET_DERIVED_KEY – Derive a key
  - SNP_GET_EXT_REPORT – Same as GET_REPORT with additional certificates imported through the SNP_SET_EXT_CONFIG.
SEV AND SEV-ES

SEV
- Guest >= 4.15
- Hypervisor >= 4.16
- Qemu >= 2.12
- OVMF >= vUDK2018
- Libvirt >= 4.5

SEV-ES
- Guest >= 5.10
- Hypervisor >= 5.11
- OVMF >= Stable202008
- Libvirt >= 4.5
- Qemu >= 6.0

In progress (patches discussed upstream)
- Live migration support
SEV-SNP

**Supported Features**
- Guest driver to query the attestation report
- Guest uses the firmware filtered CPUID values.
- Guest RAM backing page can be allocated from THP.
- Guest BIOS validates the entire guest RAM.
- Multiple vCPUs in Guest
FUTURE SNP DEVELOPMENT

- KVM Unit test and kself test
- Avacado test framework
- Restricted Interrupt Injection
- Lazy validate
- Kexec support in guest
- Live Migration
- Support backing pages from HugeTLB
- vTPM support
Q/A ?