AGENDA

- Overview
  - Use cases
  - System architecture
  - Technical details

- Linux Integration
  - Page tables
  - DMA

- Q&A
MOTIVATION -- CLOUD

- **Hypervisor must enforce full isolation between co-resident VMs**
  - Typically using hardware virtualization support like AMD-V
  - “Logical isolation” using page tables, VM intercepts, etc.
  - Sometimes breaks down
    - QEMU “VENOM” (CVE-2015-3456)
    - VirtualBox bug (CVE-2014-0983)
    - Etc.

- **Cloud users must fully trust the cloud hoster**
  - Hypervisor has full access to guest secrets in memory
  - Hypervisor enforces all isolation
  - Not ideal for users or cloud hosters
HARDWARE MEMORY ENCRYPTION - ATTACKS
DEFENDED BY AMD SME + SEV

User Access Attacks
- Administrator scrapes memory of guest data areas
- Administrator injects code into a guest VM
- Hypervisor bug allows hosted guest to steal data from other guests

Physical Access Attacks
- Probe the physical DRAM interface
- Install HW device that accesses guest memory
- Freeze then steal DIMMs
- Steal NVDIMMs
HARDWARE MEMORY ENCRYPTION - HW SUPPORT

**AMD Secure Memory Encryption (SME) / AMD Secure Encrypted Virtualization (SEV)**

- Hardware AES engine located in the memory controller performs inline encryption/decryption of DRAM
- Minimal performance impact
  - Extra latency only taken for encrypted pages
- No application changes required
- Encryption keys are managed by the AMD Secure Processor and are hardware isolated
  - not known to any software on the CPU

**Defense against unauthorized access to memory**
A Dedicated Security Subsystem

- AMD Secure Processor integrated within SoC
  - 32-bit microcontroller (ARM Cortex-A5)
- Runs a secure OS/kernel
- Secure off-chip NV storage for firmware and data (i.e., SPI ROM)
- Provides cryptographic functionality for secure key generation and key management
- Enables Secure Platform boot (Hardware validated boot)

*Hardware root of trust provides foundation for platform security*
HW MEMORY ENCRYPTION - SECURE MEMORY ENCRYPTION (SME)

- Protects against physical memory attacks
- Single key is used for encryption of system memory
  - Can be used on systems with VMs or Containers
- OS/Hypervisor chooses pages to encrypt via page tables
- Support for hardware devices (network, storage, graphics cards) to access encrypted pages seamlessly through DMA

Defense against unauthorized access to memory
SME – TECHNICAL DETAILS

1. Call CPUID Fn8000_001F to get information on memory encryption support
2. During boot, enable MemEncryptionModEn (SYSCFG[23])
3. Set the C-bit (enCrypted) on pages to be encrypted

Notes:
- C-bit location determined by CPUID call (example: address bit 47)
- C-bit is only supported when CR4.PAE=1 and paging is enabled
- Some address bits may be reserved in this mode, see CPUID

<table>
<thead>
<tr>
<th>[47]</th>
<th>[46:X]</th>
<th>[X-1:0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encrypted</td>
<td>(Reserved)</td>
<td>Physical Address</td>
</tr>
</tbody>
</table>
HW MEMORY ENCRYPTION - SECURE ENCRYPTED VIRTUALIZATION (SEV)

- Protects VMs/Containers from each other, administrator tampering, and untrusted Hypervisor
- One key for Hypervisor and one key per VM, groups of VMs, or VM/Sandbox with multiple containers
- Cryptographically isolates the hypervisor from the guest VMs
- Integrates with existing AMD-V technology
- System can also run unsecure VMs

Enhances isolation of VMs
SPI Flash holds all non-volatile state. All secret SPI flash data is encrypted with chip-unique keys.

Note: Hypervisor serves as untrusted communication channel for guest owner <-> AMD Secure Processor communications.
1. Hypervisor loads BIOS/OS image into DRAM
   - BIOS/OS image is not encrypted
   - All confidential information resides on the virtual encrypted hard drive
SEV KEY MANAGEMENT

LAUNCHING A GUEST

1. Hypervisor loads BIOS/OS image into DRAM

2. User supplies their DH key
   - Hypervisor contacts AMD Secure Processor to get
     a) Platform Certificate
     b) Diffie-Hellman exchange key
SEV KEY MANAGEMENT
LAUNCHING A GUEST

1. Hypervisor loads BIOS/OS image into DRAM
2. User supplies their DH key
3. Hypervisor prepares BIOS/OS image
   • Allocates an Address Space Identifier (ASID)
   • Requests firmware generate / load an AES encryption key in memory controller
   • Requests firmware to encrypt image
SEV KEY MANAGEMENT

LAUNCHING A GUEST

1. Hypervisor loads BIOS/OS image into DRAM
2. User supplies their DH key
3. Hypervisor prepares BIOS/OS image
4. Hypervisor sets up Virtual Machine Control Block with ASID and runs the guest
1. Hypervisor loads BIOS/OS image into DRAM
2. User supplies their DH key
3. Hypervisor prepares BIOS/OS image
4. Hypervisor sets up Virtual Machine Control Block with ASID and runs the guest
5. Hypervisor sends launch receipt to user
   • Includes measurement of guest memory
   • Includes platform authentication information
SEV KEY MANAGEMENT
LAUNCHING A GUEST

1. Hypervisor loads BIOS/OS image into DRAM
2. User supplies their DH key
3. Hypervisor prepares BIOS/OS image
4. Hypervisor sets up Virtual Machine Control Block with ASID and runs the guest
5. Hypervisor sends launch receipt to user
6. User supplies disk decryption key to the guest
   • E.g., BitLocker or dm-crypt key
   • Sent via secure channel (e.g., TLS)
Address Space ID (ASID) determines VM encryption key
- ASID is tagged with all data within the SoC
- ASID determines encryption key to use when data enters/leaves SoC

HW and Guest page tables determine if a page is “private” or “shared”
- Instruction code pages always “private”
- Guest page tables always “private”
- Data pages can be “private” (C=1) or “shared” (C=0) depending on page tables
- Before CR4.PAE=1, all pages are “private”

All DMA must occur to “shared” pages

Example use: All guest pages are “private” except for DMA pages
In host (ASID=0) mode, the host C-bit determines the encryption status of a page
- C=1 => Encrypted with SME (host) key
- C=0 => Unencrypted

In guest mode with a non-SEV guest, the nested page table determines the encryption status (same as above)

In guest mode with SEV, both the guest and nested tables are consulted:

<table>
<thead>
<tr>
<th>Guest Page Table</th>
<th>Nested Page Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>C=0 Unencrypted</td>
<td>C=0 Unencrypted</td>
</tr>
<tr>
<td>C=1 Encrypted</td>
<td>C=1 Encrypted</td>
</tr>
<tr>
<td>C=1 Encrypted</td>
<td>C=1 Encrypted</td>
</tr>
<tr>
<td>C=1 Encrypted</td>
<td>C=1 Encrypted</td>
</tr>
</tbody>
</table>
SEV ADDRESS TRANSLATION EXAMPLE

Guest virtual Address: 0x8765_4321

Example (0x8765_4321)

Guest Page Tables
Address
Nested Page Tables
Address
Guest Physical Address
C-bit
System Physical Address
C-bit
Guest Data Request
SEV ADDRESS TRANSLATION EXAMPLE

Guest virtual Address
Guest page Tables
Guest physical Address

Example (0x8765_4321)
(use gCR3)

Guest Page Tables
Address
Nested Page Tables
Address
Guest Physical Address

Guest Physical Address

C-bit

0x8000_ABCD_E321

0xABCD_E321

0xACAC_A321

0x8000_ACAC_A321
SEV ADDRESS TRANSLATION EXAMPLE

Guest virtual Address
- Guest page Tables
- (use gCR3)
- 0x8765_4321

Example (0x8765_4321)

Guest physical Address
- Nested Page Tables
- 0x8000_ABCD_E321

Host virtual address without C-bit
- 0xABCD_E321

Other text:
- Host virtual address
- System Physical Address
- C-bit
- Guest Data Request
SEV ADDRESS TRANSLATION EXAMPLE

Guest virtual Address
Guest page Tables
Guest physical Address
Host virtual address without C-bit
Nested Page Tables
Host physical address

Example (0x8765_4321)
(use gCR3)

0x8000_ABCD_E321

0xABCD_E321

(use nCR3)

0xACAC_A321

Guest Page Tables
Address
Nested Page Tables
Address
Guest Physical Address
C-bit
System Physical Address
C-bit
Guest Data Request
SEV ADDRESS TRANSLATION EXAMPLE

- Guest virtual Address
  - Guest page Tables
  - Guest physical Address
  - Host virtual address without C-bit
  - Nested Page Tables
  - Host physical address
  - Final system physical with C-bit

Example (0x8765_4321)
- (use gCR3)
  - 0x8000_ABCD_E321
- (use nCR3)
  - 0xABCD_E321
  - 0xACAC_A321

- Guest Page Tables
  - Address
  - Guest Physical Address
  - C-bit

- Nested Page Tables
  - Address
  - System Physical Address
  - C-bit

- Guest Data Request

0x8000_ACAC_A321
| SME/SEV COMPARISON |
|-------------------|-----------------|-----------------|
| **Protects against** | Physical attack | Physical and vulnerable HV |
| **Keys** | 1 | 1 per VM |
| **Encryption enabled by** | Native page tables | Guest page tables |
| **DMA** | Any page | Only to shared pages |
| **Default page behavior when CR4.PAE=0** | Unencrypted | Private (encrypted with guest key) |
| **Requires AMD Secure Processor x86 Driver** | No | Yes |
| **Software impacted** | Native OS/HV | HV and guest OS |
Linux Enablement
SOFTWARE SUMMARY

AMD is developing
- AMD Secure Processor firmware to implement key management tasks (distributed in AGESA)
  - Signed by AMD, source not public
- Linux driver to facilitate HV to AMD Secure Processor communication
  - Open source

Other major components
- Linux kernel support for SME/SEV
  - RFC patches have been sent to LKML
- KVM/QEMU support
  - Managing ASIDs, facilitating guest owner communication, etc.
LINUX SUPPORT - SME

⚠️ Initial setup
- Boot before 64-bit mode is unencrypted
- Kernel is then encrypted in-place by copying pages to ones with the C-bit set
- New pages are built with the C-bit set on all entries

⚠️ Runtime
- C-bit is set on all new page allocations
- Interfaces for requesting unencrypted pages for special purposes

⚠️ Exceptions
- EFI, ACPI – Placed in memory by BIOS, accessed as unencrypted
- SMP – Initial boot of AP’s is unencrypted, startup code is copied to an unencrypted page until the AP can begin using encryption
- DMA – Any non 48-bit capable devices use an unencrypted bounce buffer
LINUX CHANGES - SEV

- Similar to SME, but some differences
  - Pages are encrypted (private) by default
  - Initial kernel/boot data is encrypted during setup so no need to encrypt-in-place
  - All DMA must occur to shared pages

- All pages are marked private by the kernel except
  - SWIOTLB pages
  - Pages allocated via `dma_alloc_coherent`

- New `dma_ops` redirect accesses as appropriate so no driver modifications are required

- Para-virtualization features (e.g., pv-clock) use shared pages
OTHER CHANGES

- **AMD Secure Processor driver**
  - Marshals data between HV (e.g., KVM) and AMD Secure Processor MMIO interface
  - Developed by AMD

- **Hypervisor modifications**
  - Provisioning – Establishing trust in the platform and establishing secure communication with a guest owner
  - Launch – Encrypting and measuring new VM guests
  - Migration – Send/Receive functionality to support migration
  - Key Slot Management – Overcommitting of hardware key slots in the memory controller
REFERENCES

- **AMD Memory Encryption** – Overview of SME and SEV features

- **AMD64 Architecture Programmer’s Manual Volume 2: System Programming** (sections 7.10 and 15.34)
- **Secure Encrypted Virtualization Key Management**

Linux patches
- RFC patches for SME sent on 4/26 ([http://marc.info/?l=linux-doc&m=146171137011274&w=2](http://marc.info/?l=linux-doc&m=146171137011274&w=2))

Also please see talks at Xen Summit and KVM Forum
The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ATTRIBUTION

© 2016 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names are for informational purposes only and may be trademarks of their respective owners.