AMD

SECURE ENCRYPTED VIRTUALIZATION – WHAT'S NEXT

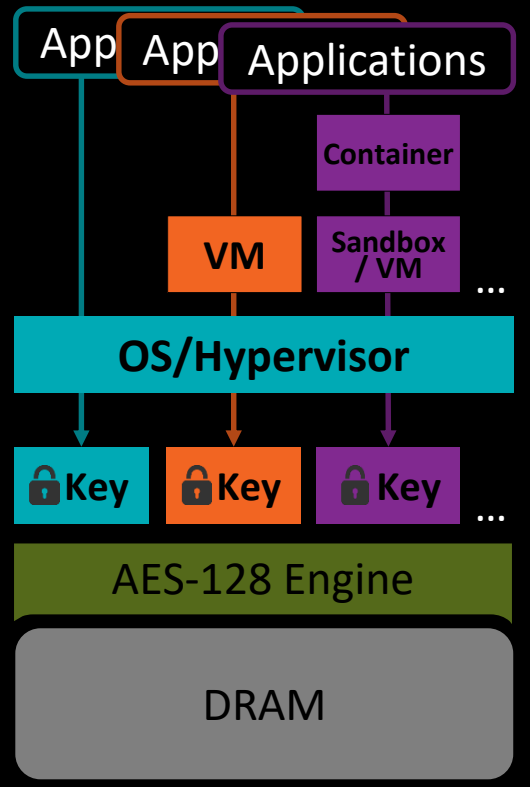
KVM FORUM - 2019

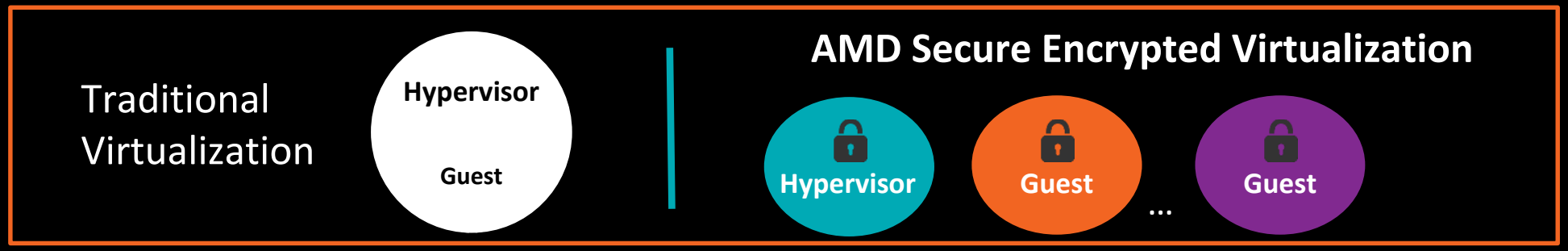
AGENDA

- SEV
  - Review
  - Live Migration
  - Migration Helper
  - Performance Work
- SEV-ES
  - Review
  - Status
- SEV-SNP

SEV REVIEW

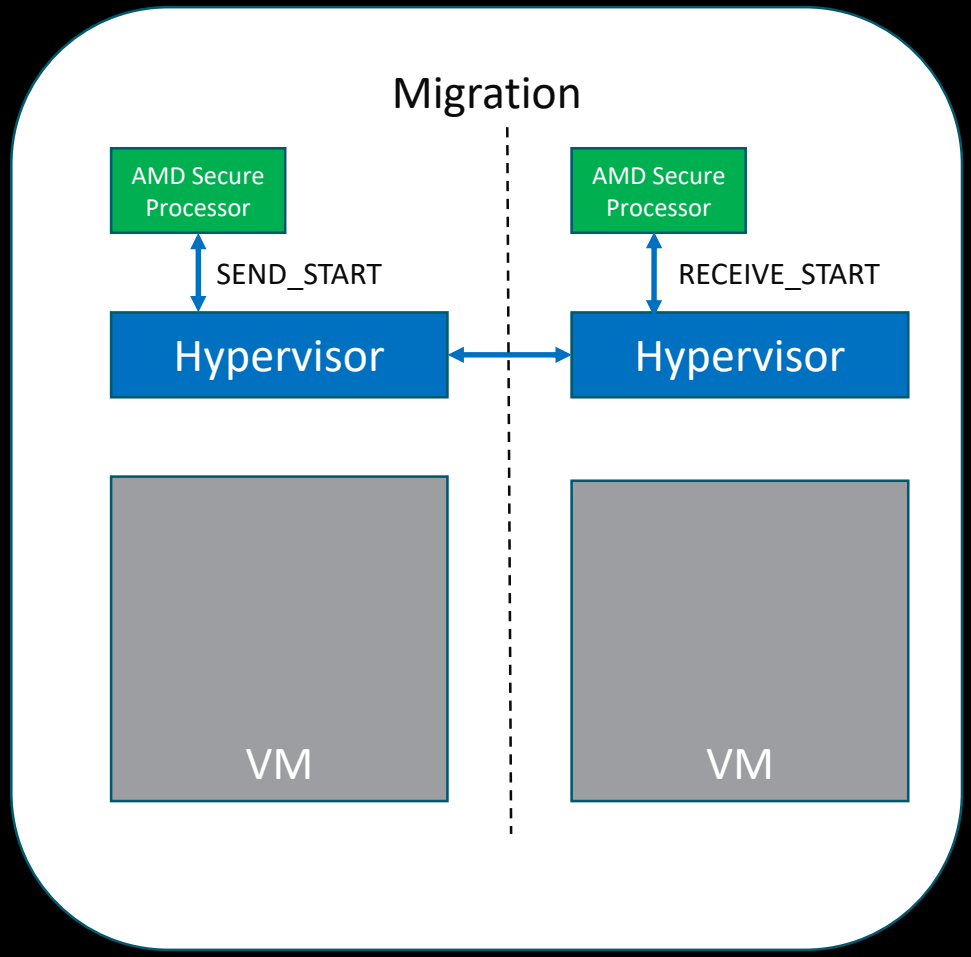
- Protects VMs/Containers from each other, administrator tampering, and untrusted Hypervisor
- One key for Hypervisor and one key per VM or VM/Sandbox with multiple containers
- Cryptographically isolates the hypervisor from the guest VMs
- Integrates with existing AMD-V™ technology
- System can also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





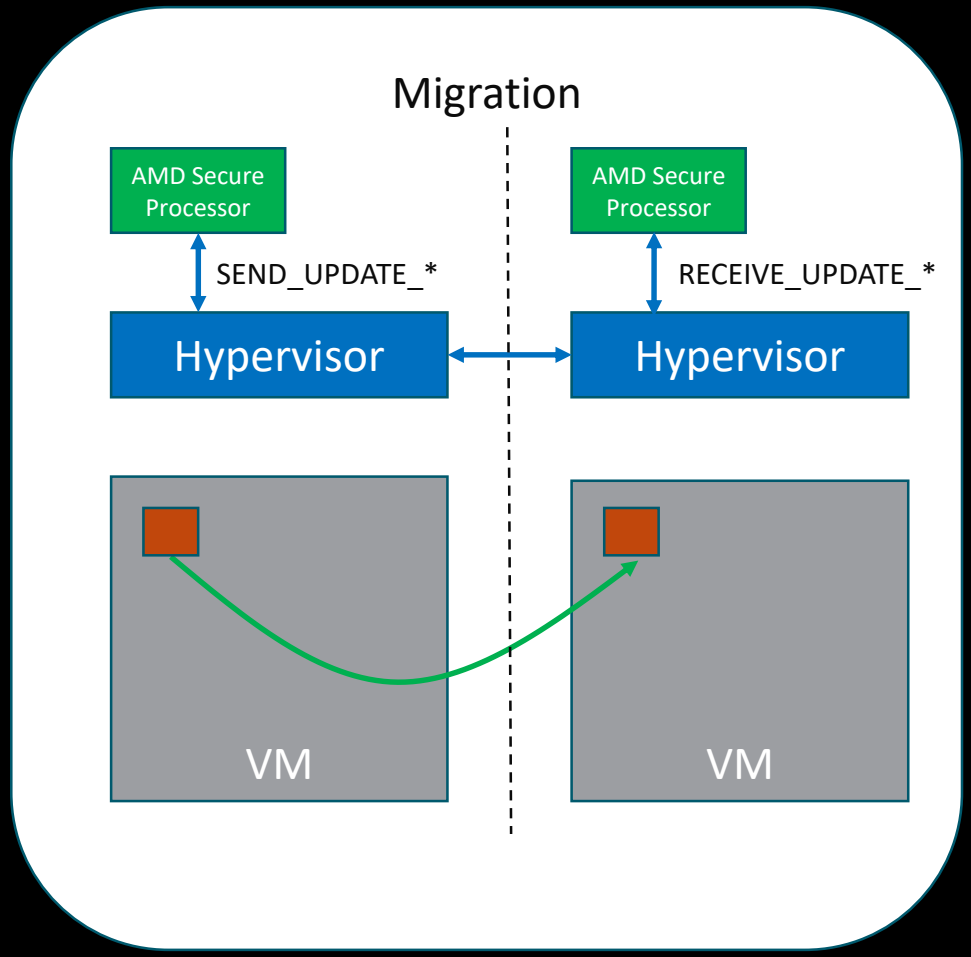
SEV LIVE MIGRATION

- Hypervisor can’t just copy encrypted memory to destination machine
  - VM encryption key is not migrated
    - Different key for source and destination SEV VMs
  - Physical location in memory matters
    - Even if key was identical, location in DRAM matters
- SEV firmware required to securely copy encrypted memory
  - Source and Destination machines negotiate transport keys
  - Encrypted pages are wrapped using transport keys



SEV LIVE MIGRATION…

- Hypervisor maintains a bitmap of guest page encryption state
  - Hypercall from guest notifies of change in encryption state
- Source Hypervisor
  - Uses SEND API for encrypted pages
    - Transforms the page for transport
- Destination Hypervisor
  - Uses RECEIVE API for encrypted pages
    - Transforms the page for use in the guest
- At completion, guest page encryption state is migrated

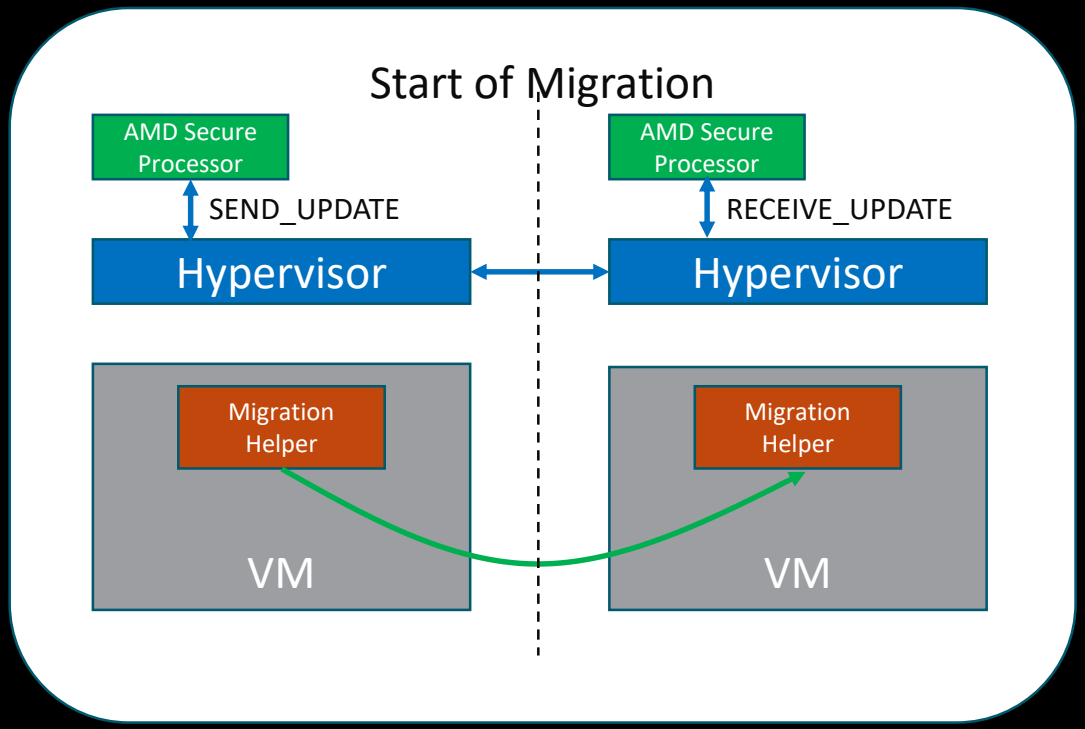


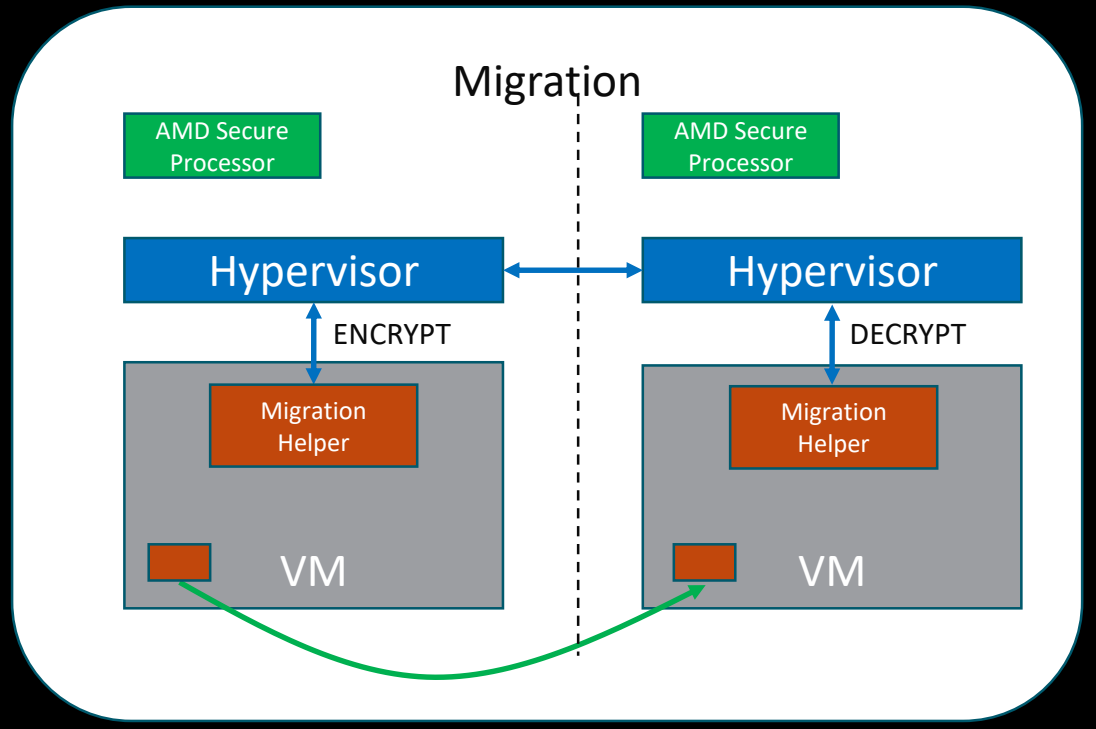
SEV MIGRATION HELPER

- AMD Secure Processor becomes a performance bottleneck
- Migration can be broken down into Authentication and Data Movement
- Authentication
  - Performed by AMD Secure Processor
  - Enforces guest migration policy
- Data Movement - Migration Helper (MH)
  - Proof of Concept
  - Runs on a “hidden” vCPU
    - Not visible to the guest
    - Runs only migration helper code with full access to guest memory
  - Listens for migration commands
    - INIT – Generates the migration key
    - ENCRYPT – Encrypts page for migration
    - DECRYPT – Decrypts page for use by guest

SEV MIGRATION HELPER…

- Source hypervisor uses SEND API to migrate the MH
- Destination hypervisor uses RECEIVE API to receive the MH
- MH migrates the remainder of the guest
  - Uses CPU AES instruction, AMD Secure Processor not involved
  - Migration at multiple Gbps



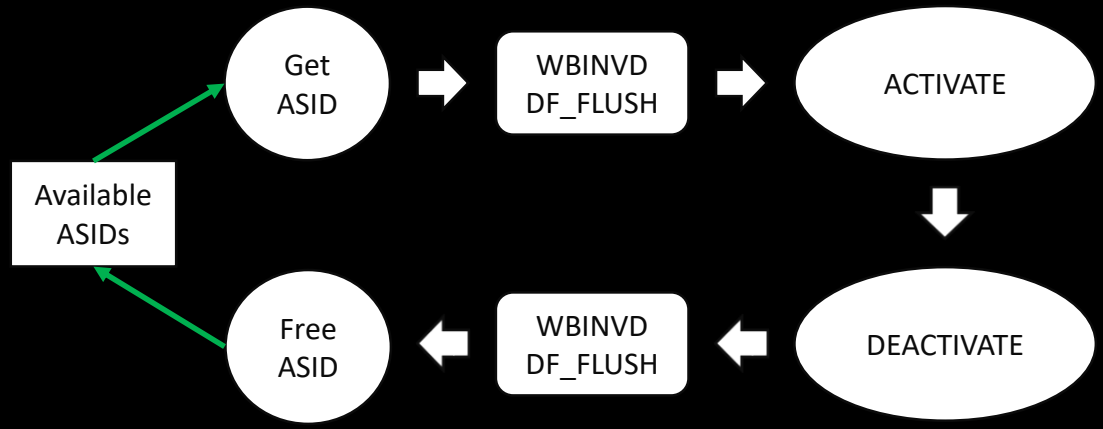


SEV PERFORMANCE

- All guest memory is currently pinned
  - Physical location in memory matters
    - Can’t just “move” a guest encrypted page
  - Higher initial resource requirements
  - Slower guest startup
- Investigating Options:
  - Prevent page movement (migration/swapping) of an SEV guest page
    - Mark page as an SEV guest page
    - Eliminate need to pin all the guest memory before boot
  - SEV firmware required to securely move encrypted memory
    - Need to hook into page migration / compaction / swap path
      - Need to determine if the page is encrypted – build upon Live Migration bitmap
    - Move encrypted page using new COPY API
      - If COPY API is not supported, prevent page from being moved
    - Prevent page from being swapped
      - No API currently in place to swap out a page

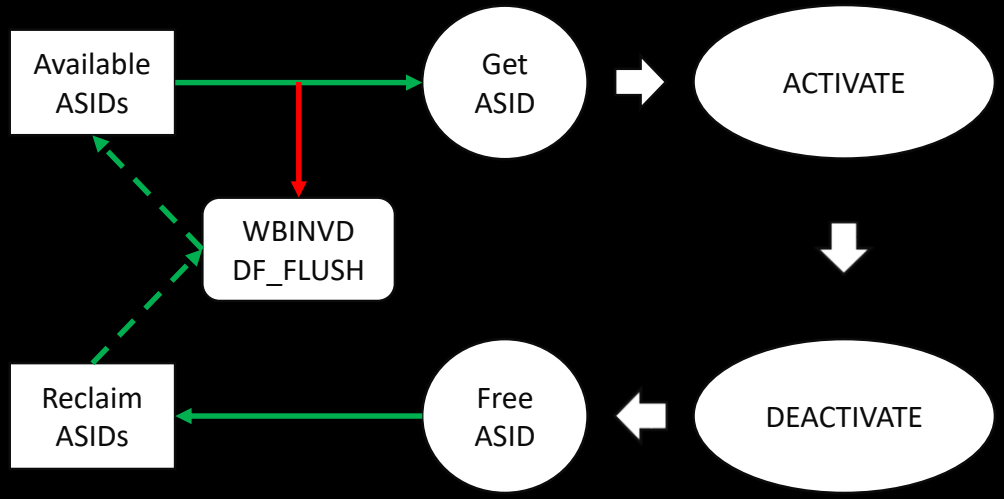
SEV PERFORMANCE…

- SEV ASID Activation
  - 1st generation EPYC supports 15 SEV ASIDS, 2nd generation EPYC supports 509 SEV ASIDS
  - WBINVD / DF_FLUSH command done on every LAUNCH and DEACTIVATE
    - Very expensive operations



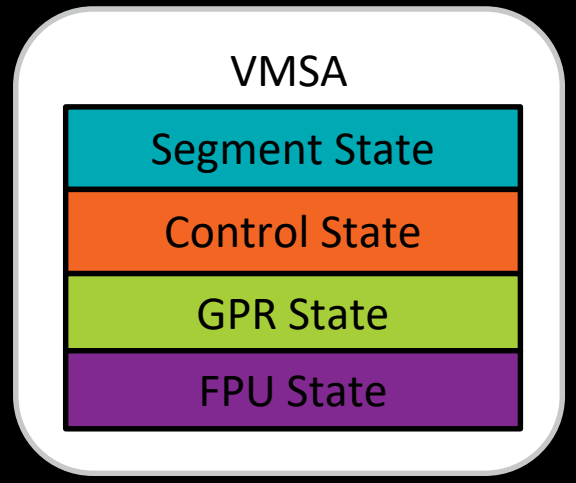
SEV PERFORMANCE…

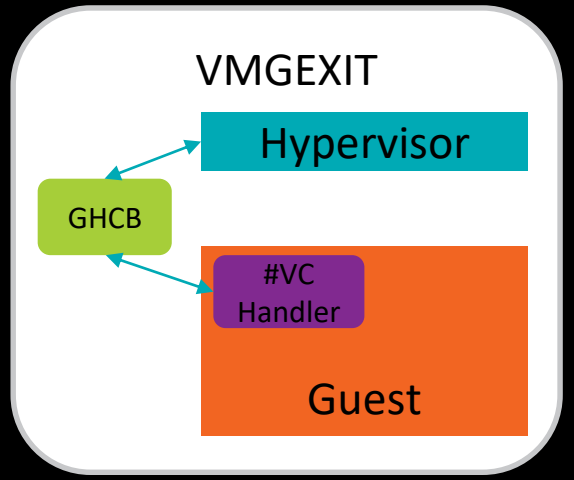
- SEV ASID Reclaim
  - Reduce the number of WBINVD / DF_FLUSH command invocations
  - Track ASIDs on DEACTIVATE as reclaimable
  - Issue WBINVD / DF_FLUSH command when there is no available SEV ASID
    - Reclaims ALL ASIDs



SEV-ES REVIEW

- Guest register state protection
  - Register state is initialized with known state (Initial Processor State)
  - Register state is encrypted and measured as part of the SEV LAUNCH process
  - Integrity check performed on each VMRUN
  - World switches now swap ALL register state
- VMCB under SEV-ES
  - Control Area (VMCB) and Save Area (VMSA) now separated
    - VMCB now points to VMSA
  - VMSA extended to save more state
- Guest-Hypervisor Communication Block (GHCB)
  - Allows guest ←→ hypervisor communication of the state needed to satisfy the guest service request
  - Shared (un-encrypted) page between the hypervisor and the guest
  - GHCB specification (in process)
    - Defines the format of the GHCB and how to communicate with the hypervisor





SEV-ES

- Current Status:
  - Guest-Hypervisor Communication Block protocol (near) final
    - Adding requirements for ensuring proper CPUID and MSR values are set for the guest
  - OVMF patches submitted
    - RFC stage
  - Kernel patches being completed
    - A few areas to address to be able to use a single kernel as hypervisor and guest
  - Qemu patches being completed
    - Uses the memory encryption policy object to indicate SEV-ES
    - More investigation on register accesses
  - GitHub trees:
    - https://github.com/AMDESE

SEV-SNP

- SEV-SNP – Secure Encrypted Virtualization - Secure Nested Paging
  - Next step in the evolution of SEV
  - SEV/SEV-ES provides Confidentiality
    - SEV – Encryption of VM memory
    - SEV-ES – Adds Encryption of VM registers
  - SEV-SNP builds on SEV-ES and adds Integrity Protection
    - Prevents replay attacks, corruption attacks, remapping attacks

- Linux Security Summit presentation on Friday @ 2:20PM
  - Upcoming x86 Technologies for Malicious Hypervisor Protection - David Kaplan, AMD
- White Paper
  - “AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More”

REFERENCES

- Links to the following reference material can be found at https://developer.amd.com/sev
  - White Papers & Specifications
    - Protecting VM Register State with SEV-ES Whitepaper
    - Guest Hypervisor Communication Block Specification
    - AMD64 Architecture Programmer's Manual Volume 2: System Programming
      - Sections 7.10, 15.34 and 15.35
- Code / Patches
  - https://github.com/AMDESE

DISCLAIMER

- The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD’s products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.
- AMD, the AMD Arrow logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.
-