SEV-SNP and SVSM

Evaluating the Performance Impact of Privilege Levels of Confidential Virtual Machines

Alexander Schindler
Advisor: Patrick Sabanic
Chair of Computer Systems
https://dse.in.tum.de/

Submission: 15/11/2024

Motivation: Research context

- Benefits of remote hosted VMs vs. self hosted hardware
- BUT: huge concerns for security i.e. privacy

How to make sure that privacy is established on outsourced hardware?

- Needs hardware modifications → Extended Virtualization Interfaces
- Needs modifications of system software → HV, Firmware, Guest OS

State-of-the-art

- Intel and AMD are developing solutions
- Focus on AMD: SEV-SNP Interface
- Concepts:
  - In-memory encryption, keys in SP
  - Per-VM ephemeral keys, managed by chip
  - Also: Secured Registers and Hardened Memory Mapping
  - VMPLs: Hardware managed security levels within VM

Research gap

- Problem of Securing the entire guest OS: possible?

- SVSM: small programm with high privileges monitors certain security Aspects of guest OS

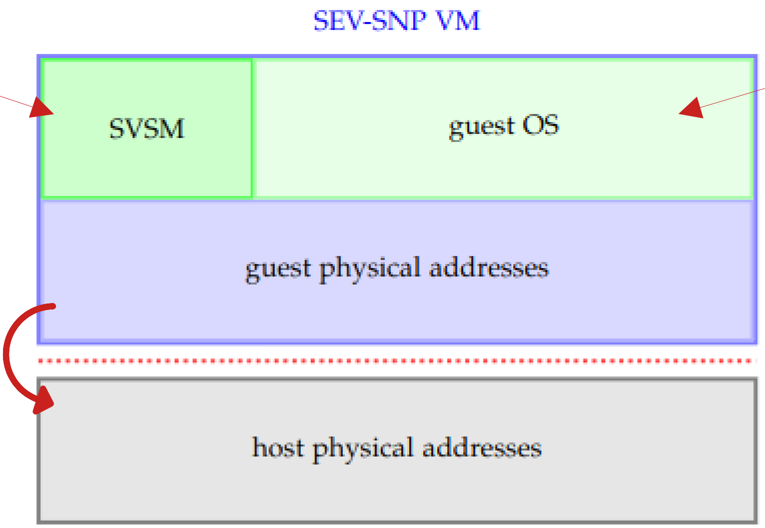
- No feature complete version of SVSM yet, not much data on practicality

Visualisation: A VM with SVSM loaded

In VMPL 0

In higher VMPL
(less privilege)

Mapping
managed by HV,
but controlled by
SEV-SNP



Problem Statement

- Problem 1: Not enough data about practicality of SVSM for huge data centers

  ➝ Goal: Retrieve data on performance

- Problem 2: No mechanism to validate guest-OS integrity

  ➝ Goal: Interface to measure guest-OS memory on-demand

  – Is it possible to secure the guest kernel and arbitrary memory (e.g. modules, user applications, …) with SEV and SVSM?

Part One: Performance Comparison

- Two perspectives: Customer and (Datacenter) Vendor
- Customer: What is the performance impact within the VM?
- Vendor: What is the impact on boot-time per VM?

General performance impact: UnixBench

Boot-time impact: bpf-tooling

Part One: Performance Comparison

Results

Plain vs SVSM

	1VM	8VMs
File copy, large buffer	-31.4%	-15.0%
Proc creation	+14.2%	-8.6%
Shell 1conc.	-0.2%	+13.9%
Shell 8conc.	-0.6%	+12.6%

CVM vs SVSM

1VM	8VMs
+341.6%	+307.6%
+66.8%	+27.2%
+26.7%	-7.8%
+31.8%	-4.9%

Positive: SVSM-VM is that many % better than the comparison
Negative: SVSM-VM is that many % worse than the comparison

Part One: Performance Comparison

Evaluation

- For plain vs. svsm: apparently heavy io is worse with svsm, for tasks closer to “real world” it seems to perform better when parallelized
- For cvm vs. svsm: svsm outperforms cvm in most scenarios, slight drop when parallelized

Why?

Artificial tasks, slightly different setups, errors/interference, etc. → “weird” results

Also: for SVSM the guest kernel functions don’t need to do SEV work

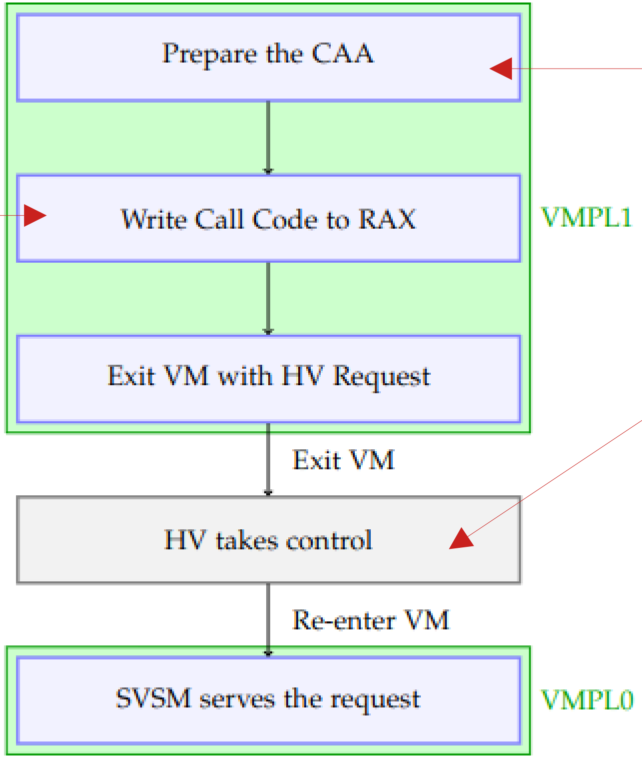
Part Two: Measuring GuestOS Memory

- Currently: Hard to detect malicious OS
  - SVSM can limit certain capabilities but OS or Applications could still leak data
  - End goal: verify entire OS kernel before booting
- Extending CoconutSVSM:
  - New protocol: given arbitrary gPA, return signed hash (computed in SVSM)
- Extend Linux guest Kernel:
  - Kernel Module to interface new protocol

But: Guest OS itself not yet secured (needs Firmware modification)

Part Two: Measuring GuestOS Memory

Initiate Protocol Call:



Parameters in Registers

For use in between VMPLs, encrypted from HV

Request can be denied but not manipulated: CAA secured by SEV-SNP, Request and Response Validated in guest

Part Two: Measuring GuestOS Memory

- Protocol call secured from HV and outside world (by SEV-SNP)
- Hash calculated within SVSM → no interference from HV, guest OS etc.
- Address to calculate hashes for come from outside:
  - Firmware: could make protocol call for kernel memory after it has been loaded into memory (not yet implemented)
  - Guest OS: can dynamically calculate memory hashes for modules, applications, etc. → compromised kernel could alter addresses in request or attempt replay attacks

Part Two: Measuring GuestOS Memory

→ Security not established until guest OS is validated from the outside

Only first part is implemented (SVSM protocol) and the additional functionality for dynamic validation (interface to protocol from within guest OS)

Summary

- SEV-SNP with SVSM has a lot of potential
- Performance Tradeoffs: There seems to be an impact especially for in-VM performance → Only for users where confidentiality outweighs performance significantly
- SVSM can be extended to further harden VM, but relatively high effort
  - SVSM, Firmware (for guest Kernel) and guest Kernel need to be modified
  - Firmware modification for guest OS validation not yet realized

Questions?