Hypervisor Memory Forensics

Mariano Graziano and Davide Balzarotti

SANS DFIR EU SUMMIT

October 2013 - Prague

S3 GROUP - EURECOM
http://s3.eurecom.fr

S3 GROUP

Faculty

Davide Balzarotti

Aurelien Francillon

Research Engineers

Andrea Lanzi

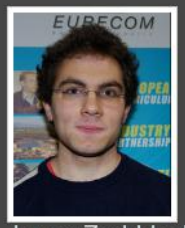
Luca Bruno

S3 GROUP

Phd Students


Jelena Isachenkova


Davide Canali

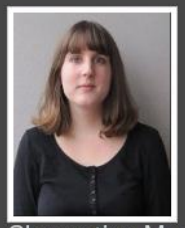


Jonas Zadddach


Mariano `emdel` Graziano


Giancarlo Pellegrino


Andrei Costin


Clementine Maurice

Actaeon

- Memory forensics of virtualization environments
- Locate any **Intel** Hardware assisted Hypervisor
- Detect **nested** Virtualization
- Transparent Guest **Introspection**

Actaeon [Use Cases]

- Hypervisors are everywhere:
  - Xen, KVM, VirtualBox, Vmware, Hyper-V, bhyve
  - Cloud (Amazon, Microsoft, Google, Apple)
  - Domestic use (Running multiple operating systems)
  - Security Solutions (Sandboxes, DeepDefender, Bromium etc)
  - POC Malware (BluePill, Vitriol)

- The forensics community needs tools for digital investigations of virtual environments

What Actaeon is NOT

- Real time hypervisor detector
- Physical memory dumper
- Hypervisor-based malware detector

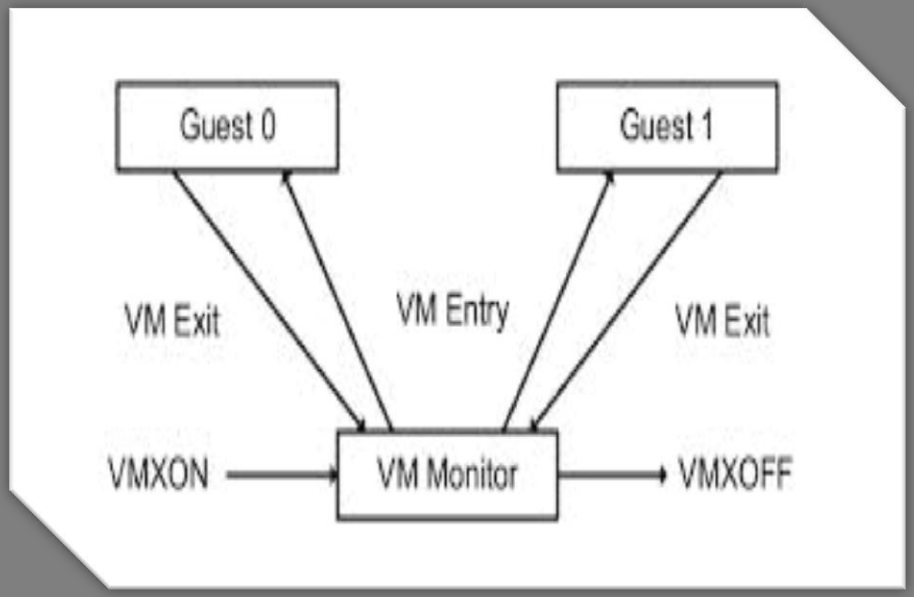
Actaeon framework

- VMCS memory layout dumper
- Hyperls
- Volatility patch for guest introspection

VMCS Dumper

[Theory]

- Intel gives process level support for virtualization
- There are 2 main VMX operations: root and non root



*From the Intel Manual

VMCS Dumper

[VMCS]

- Virtual Machine Control Structure
- VMCS controls both VMX non root operation and VMX transitions
- The format to store the VMCS data is implementation specific
- Every field is associated with a 32 bit value (its encoding) used by VMREAD/VMWRITE instructions
- The VMCS data is divided in 6 groups

VMCS Dumper

[Reversing]

- Custom Hypervisor initialization code (based on HyperDbg) :
  - VMCS memory region allocation
  - Fill the region with an 16 bit incremental counter
  - Perform VMREAD operations
  - Same approach valid for nested VMCS structures

VMCS Dumper

[Demo]

DEMO

HyperIs

[Scanning]

- Memory scanner looking for VMCS structures
- We use selected VMCS fields:
  - REVISION_ID
  - VMX_ABORT_INDICATOR
  - VMCSLINKPOINTER
  - HOST_CR4
- These fields cannot be obfuscated

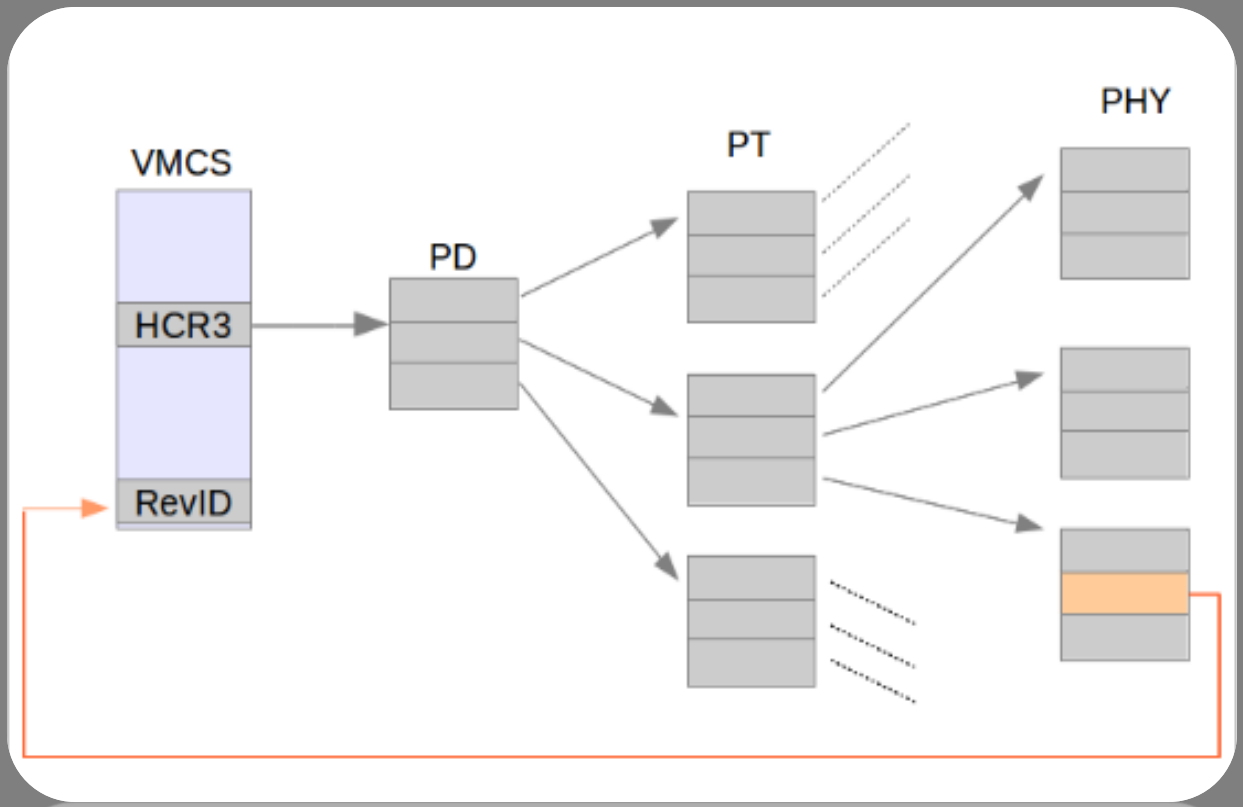
HyperIs

[Validation]

- HOST_CR3 property:
  - The HOST_CR3 register points to the hypervisor page tables
  - The page tables need to map the page containing the VMCS
- For every VMCS candidate we extract the HOST_CR3
  - We walk the entire page tables
  - We obtain all the allocated physical pages
  - The VMCS is validated if and only if it is in the set of the allocated physical pages

HyperIs

[Validation]



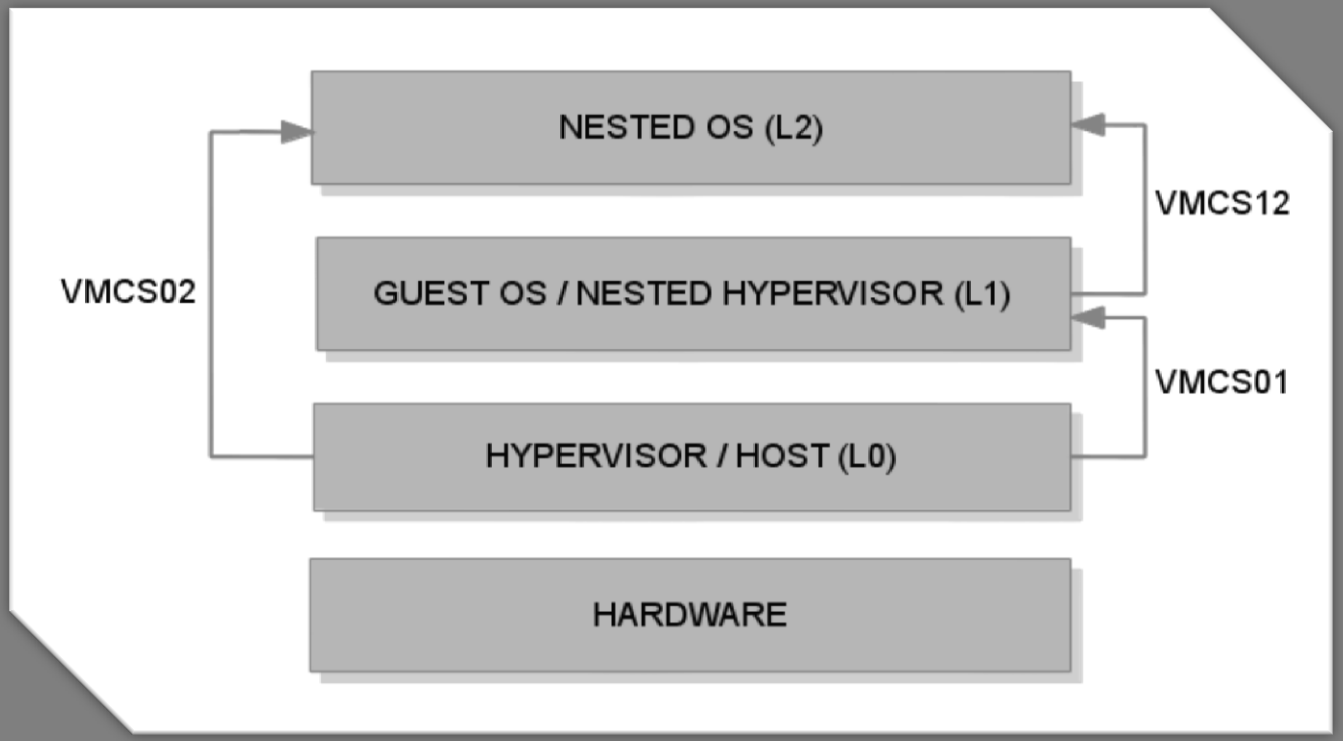
HyperIs

[DEMO]

DEMO 1

HyperIs [Nested]

- A guest virtual machine can run an hypervisor
- In x86 only one hypervisor is in root mode



DEMO 2

Guest Introspection [EPT]

- Extended Page Tables (EPT): "New" Intel Hardware feature
- Address translation from Guest Physical Addresses (GPA) to Host Physical Addresses (HPA)
- It has different stages (very similar to IA-32e)

Guest Introspection [Algorithm]

- We extract the EPT_POINTER from the VMCS
- We translate, when required, all the GPA to HPA through the EPT table
- We patched Volatility to use this pointer during the address translation

Guest Introspection

[DEMO]

DEMO

Limitations

- Actaeon supports only Intel hardware assisted hypervisors (No AMD support, no paravirtualization)
- Actaeon supports EPT (no shadow page tables)
- Dump is not our concern (VT-d disabled)

Future Works

- We are working to support:
  - Hyper-V
  - Introspection for Linux Guests
  - VMCS Shadowing
  - VMWare ESXi
  - AMD

Questions?

Mariano Graziano

graziano at eurecom dot fr

@emd3l

Davide Balzarotti

balzarotti at eurecom dot fr

@balzarot