WEAPONIZING HYPERVISORS TO FIGHT & BEAT CAR & MEDICAL DEVICE ATTACKS

Ali Islam – CEO Numen Inc

Dan Regalado – DanuX – CTO Numen Inc

NUMEN

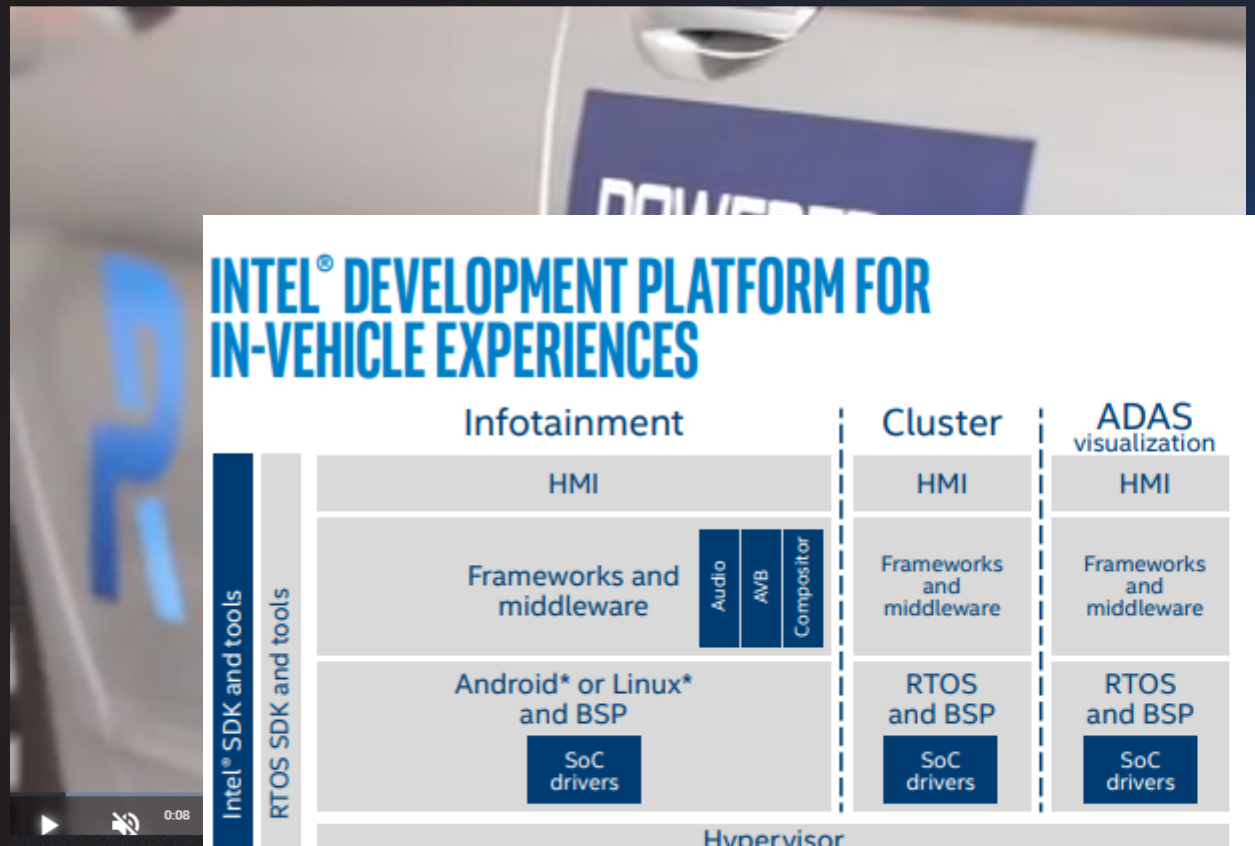
AGENDA



- Basic Concepts
- Embedded Environment
- Demo – Attacks & Use cases
- Q&A

NUMEN

Hypervisors & Strong Trending



INTEL® DEVELOPMENT PLATFORM FOR IN-VEHICLE EXPERIENCES

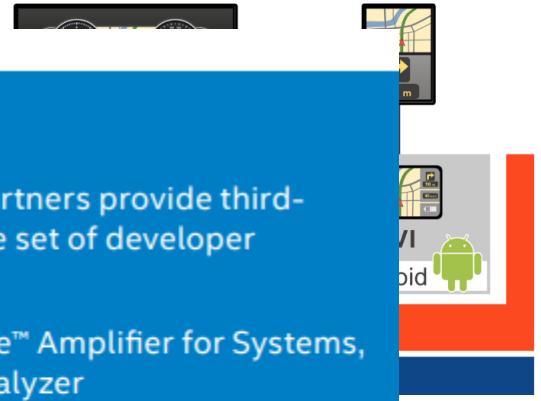
Figure 2. The Intel® development platform for in-vehicle experiences drives the consolidation of systems in the vehicle

Developer tools

To speed development, Intel and its partners provide third-party OS support and a comprehensive set of developer tools, including:

- The Intel® C++ Compiler, Intel® VTune™ Amplifier for Systems, and Intel® Graphics Performance Analyzer
- Reference stacks, including an IVI middleware and automotive boot loader
- Reference OS support for Linux*, Android*, Green Hill Integrity*, QNX CAR Platform for Infotainment*, and Wind River Helix*
- Hypervisors for multi-operating systems from QNX, Green Hill, and Mentor Graphics
- Performance-tuning tools for the Intel® architecture-based CPU and GPU and complete hardware development vehicles

Renesas' R-Car SoC and OpenSynergy's Hypervisor Enables Practical Shared Display of Meter Display and IVI Function



INTEGRITY GLOBAL SECURITY

Generation

d Infotainment with

Consumer Technology Association

Incorporated (NASDAQ: QCOM), announced today their efforts to support global automakers and Tier-1 suppliers with purpose-built, scalable solutions, designed to support a safe, secure consolidation of Android and Linux-based infotainment processing with critical ASIL-certified vehicle services into a single multicore-based electronic control unit (ECU). As a part of the relationship, Green Hills is working with Qualcomm Technologies to feature the Green Hills INTEGRITY® real-time operating system (RTOS), INTEGRITY Multivisor™ secure virtualization, and integrated MULTI® ASIL D-qualified software development environment as part of the new Qualcomm® Snapdragon™ Automotive Cockpit Platforms. Designed with a focus on production-readiness and consolidation of diverse safety and security

By 2020, the two companies look to install an infotainment solution in millions of vehicles throughout China.

Agent-less vs Agent (AV)

- Sophisticated Invisibility (VMI) - Cat and mouse game
- No messing up the actual device functionality
- Helps with regulations and certifications

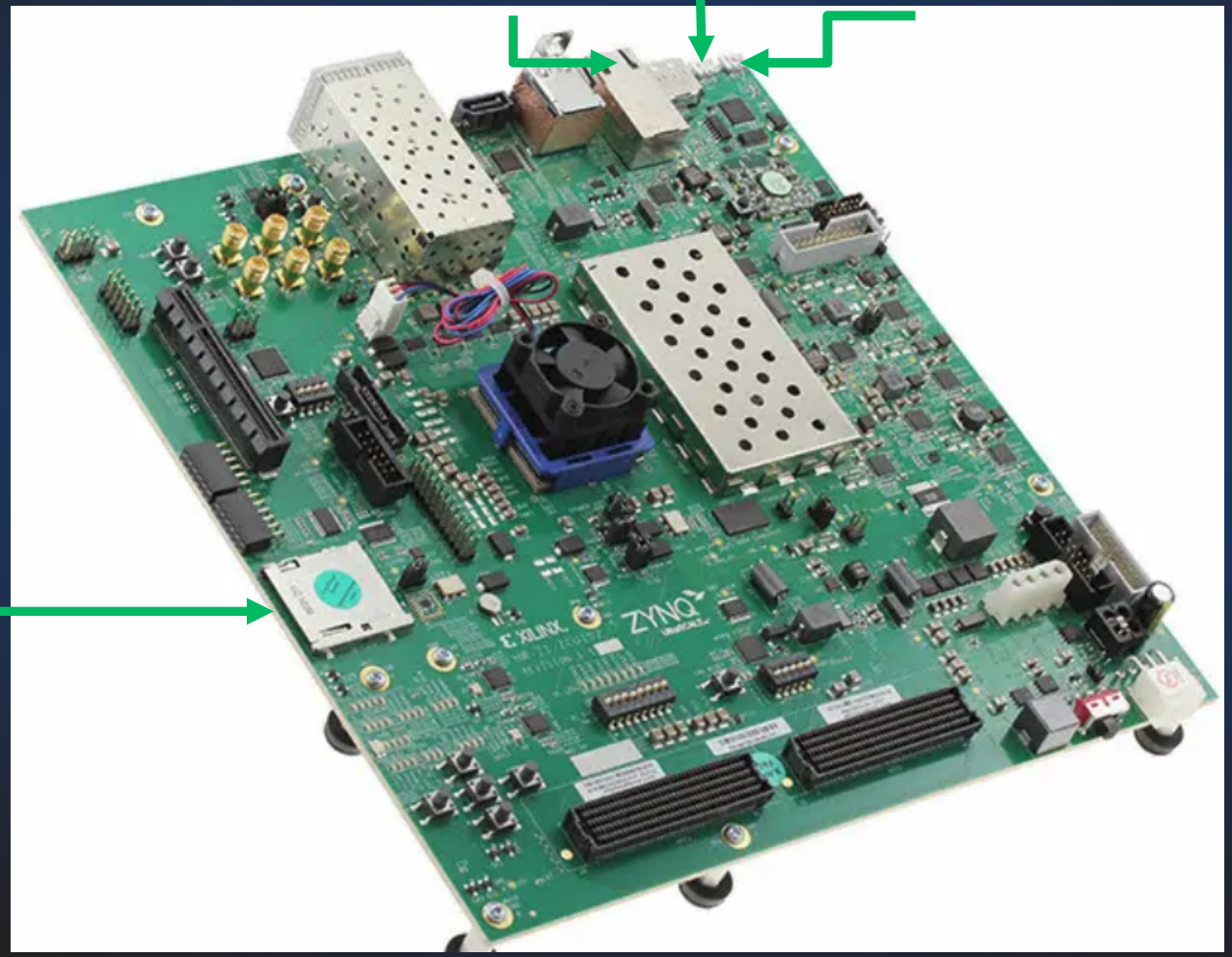
NUMEN

Let’s start the Journey

Setting up the Environment on a Zynq UltraScale+ MPSoC ZCU 102



Zynq UltraScale+ MPSoC ZCU 102



- DDR4 – 4 GB
- Quad-core Cortex A-53
- Dual-core Cortex R5F

Src: https://www.digikey.com/product-detail/en/xilinx-inc/EK-U1-ZCU102-G/122-2035-ND/7035245#images-1

NUMEN

Booting the board with JTAG

- Using Xilinx System Debugger CLI (xsdb) which reads a tcl file

```
proc run {fsbl uboot bl31 pmu} {
        targets -set -filter {name =~ "PS8" || name =~ "PSU" }
        # According to AR 67871
        mwr 0xffff0000 0x14000000
        mwr 0xffca0038 0x1FF
        mask_write 0xFD1A0104 0x501 0x0

        after 1000
        targets -set -filter {name =~ "MicroBlaze PMU"}
        dow $pmu
        con
        after 1000

        # download and run FSBL
        targets -set -filter {name =~ "Cortex-A53 #0"}
        # downloading FSBL
        dow $fsbl
        con
        after 1000
        stop

        # Set the GEMS to be NS
        mwr 0xff240000 0x492
        mwr 0xff240004 0x492

        # Unmask SMMU interrupts.
        mwr 0xFD5F0018 0x1f

        dow $uboot
        dow $bl31
        con
}
```

```
set fsbl /home/builder/ZCU102_BSP_2018.3/images/zynqmp_fsbl.elf
set uboot /home/builder/ZCU102_BSP_2018.3/images/u-boot.elf
set bl31 /home/builder/ZCU102_BSP_2018.3/images/bl31.elf
set pmu /home/builder/ZCU102_BSP_2018.3/images/pmufw.elf
```

PMUFW – Setup clock and platform management
FSBL – First Stage Bootloader – Initializes U-Boot
U-Boot – Boots the Hypervisor, Kernel and rootfs
BI31 – ARM Trusted Firmware

```
U-Boot 2018.01 (Dec 06 2018 - 09:36:05 +0000) Xilinx ZynqMP ZCU102 rev1.0

I2C:   ready
DRAM:  Hit any key to stop autoboot:  0
ZynqMP>
```

NUMEN

U-Boot Configuration

- Preparing Device Tree Blob (DTB) xen.dtb file (dts below):

```
chosen {
        bootargs = "earlycon console=ttyPS0,115200 clk_ignore_unused";
        stdout-path = "serial0:115200n8";
        #address-cells = <0x2>;
        #size-cells = <0x1>;
        xen,xen-bootargs = "console=dtuart dtuart=serial0 dom0_mem=1G bootscrub=0 dom0_max_vcpus=2 sched=credit altp2m=1";
        xen,dom0-bootargs = "console=hvc0 earlycon=xen earlyprintk=xen maxcpus=1 clk_ignore_unused root=/dev/mmcblk0p3 rw rootwait";

        dom0 {
                compatible = "xen,linux-zimage", "xen,multiboot-module";
                reg = <0x0 0x80000 0x3100000>;
        };
};
```

- Preparing the hypervisor:

  # mkimage -A arm64 -T kernel -a 0x1400000 -e 0x1400000 -C none -d xen-zcu102-zynqmp xen.ub

ZynqMP> tftpb 0x1380000 xen.dtb
ZynqMP> tftpb 0x80000 Image-2018.3
ZynqMP> tftpb 0x1400000 xen.ub
ZynqMP> bootm 0x1400000 - 0x1380000

```
root@zcu102:~# uname -a
Linux zcu102 4.14.0-xilinx-v2018.3 #1 SMP Thu Dec 6 10:01:26 UTC 2018 aarch64 GNU/Linux
```

NUMEN

Building the rootfs

- PetaLinux: Xilinx-based and therefore not universal
- Yocto: Universal but builds a Busybox limited rootfs
  - ✓ Real pain to compile new libraries
- Debootstrap: Way to go, Debian-based FileSystem ☺

NUMEN

Dev environment

- You do not want to make changes directly on the board
- Schroot to the rescue
  - ✓ Chroot into the rootfs but from a mounting point via QEMU

```
builder@DanilA-Lab:~/Xilinx$ uname -a
Linux DanilA-Lab 4.13.0-36-generic #40~16.04.1-Ubuntu SMP Fri Feb 16 23:25:58 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
builder@DanilA-Lab:~/Xilinx$ sudo schroot -c zcu102
[sudo] password for builder:
(zcu102)root@DanilA-Lab:/home/builder/Xilinx# uname -a
Linux DanilA-Lab 4.13.0-36-generic #40~16.04.1-Ubuntu SMP Fri Feb 16 23:25:58 UTC 2018 aarch64 GNU/Linux
(zcu102)root@DanilA-Lab:/home/builder/Xilinx#
```

NUMEN

Let’s get the damn ARM Syscalls out!

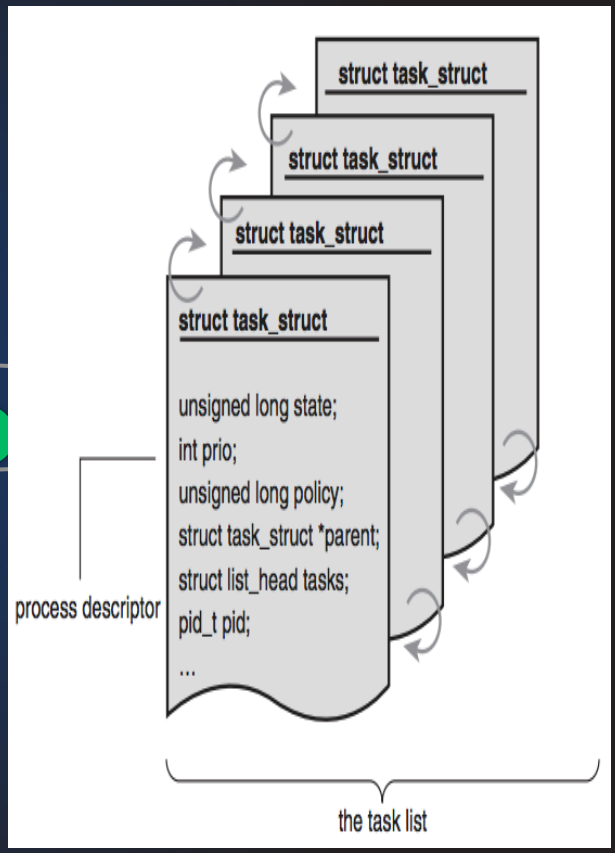


VMI & Semantic Gap



Understand meaning using OS specific knowledge

linux_name = 0x4f0;
linux_tasks = 0x280;
linux_mm = 0x2d0;
linux_pid = 0x334;
linux_pgd = 0x40;

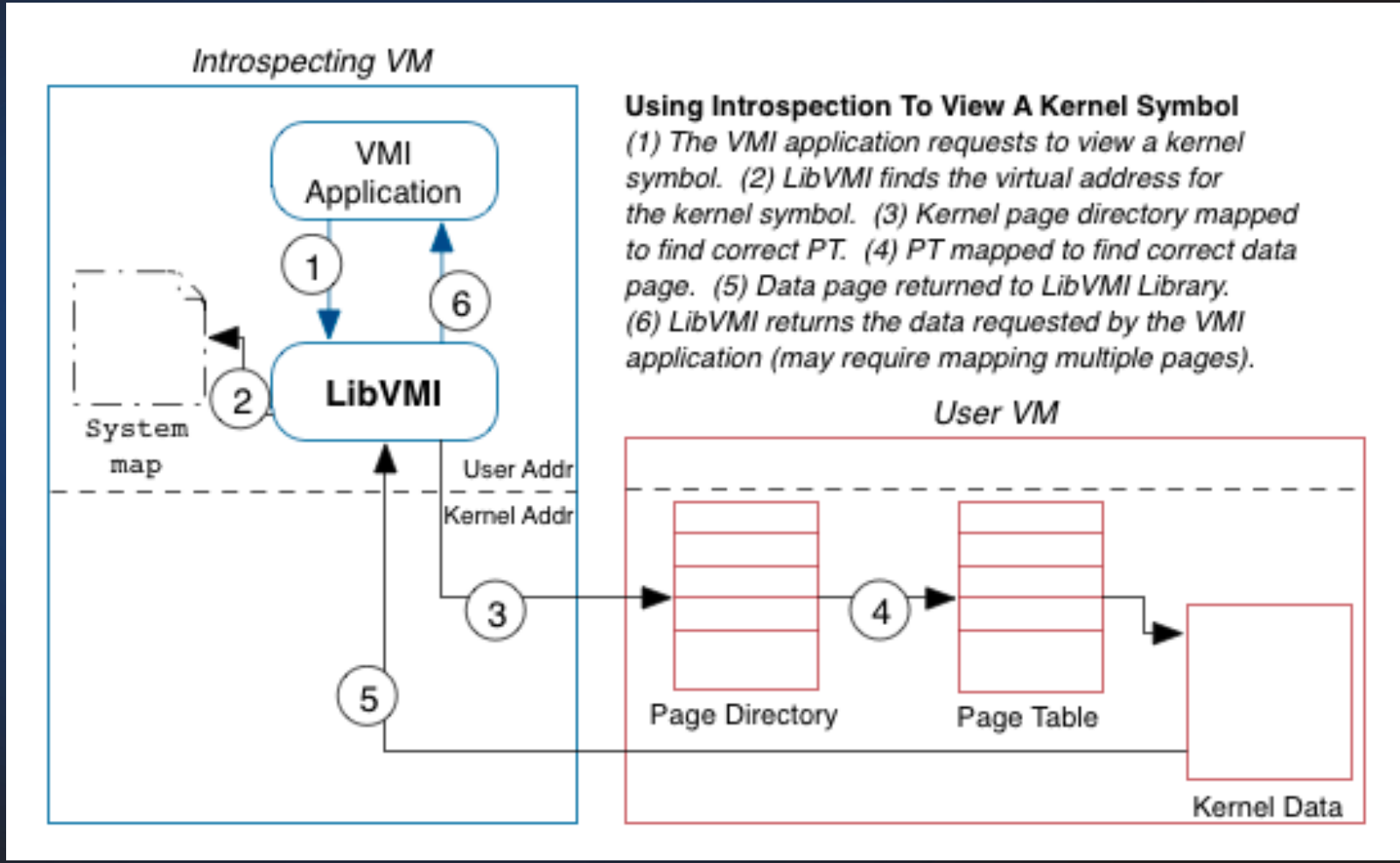


https://notes.shichao.io/lkd/ch3/

NUMEN

Kernel Symbol Value Example

```
status_t vmi_read_ksym(
    vmi_instance_t vmi,
    const char *sym,
    size_t count,
    void *buf,
    size_t *bytes_read
);
```



http://libvmi.com/docs/gcode-intro.html

NUMEN

Single Stepping

- Hardware Breakpoints
- Software Breakpoints - CPU assisted
- Software breakpoints – No CPU Assistance

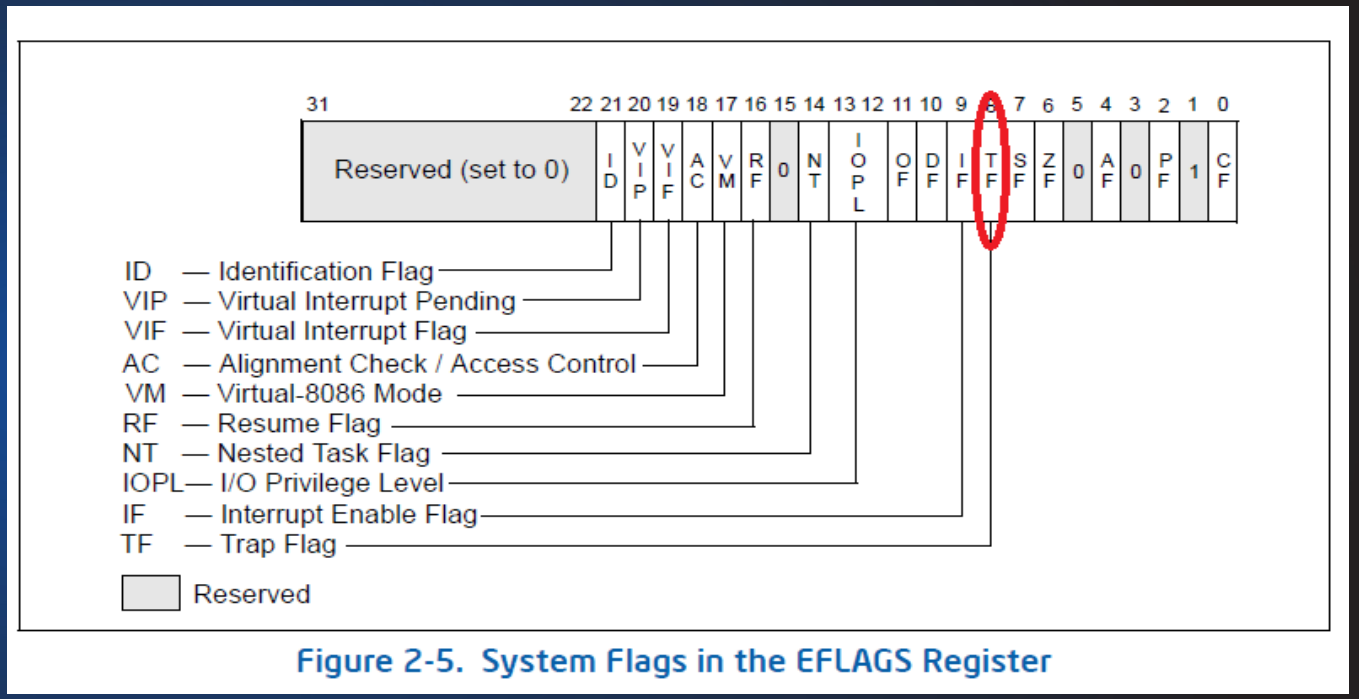
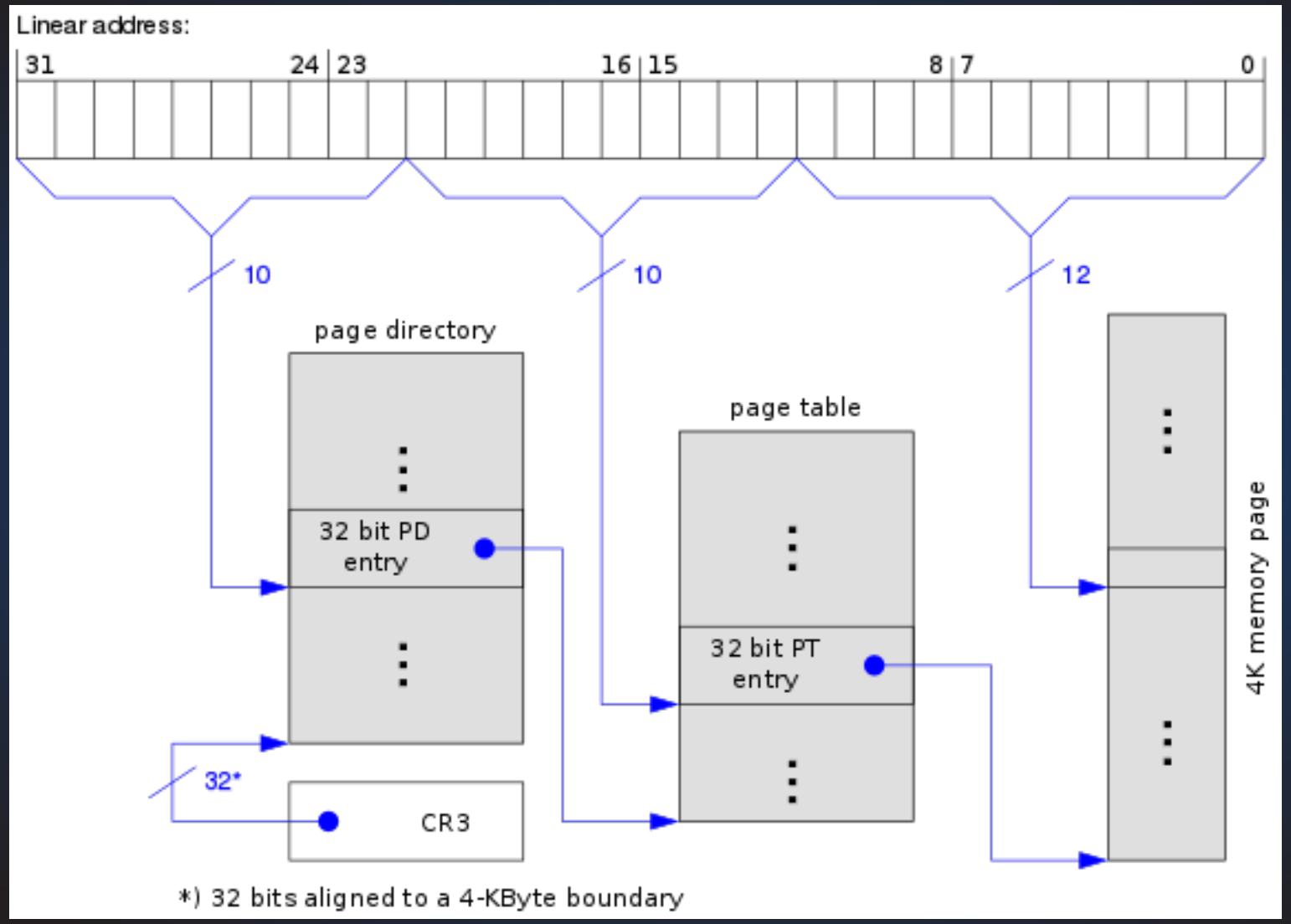


Figure 2-5. System Flags in the EFLAGS Register

http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html

NUMEN

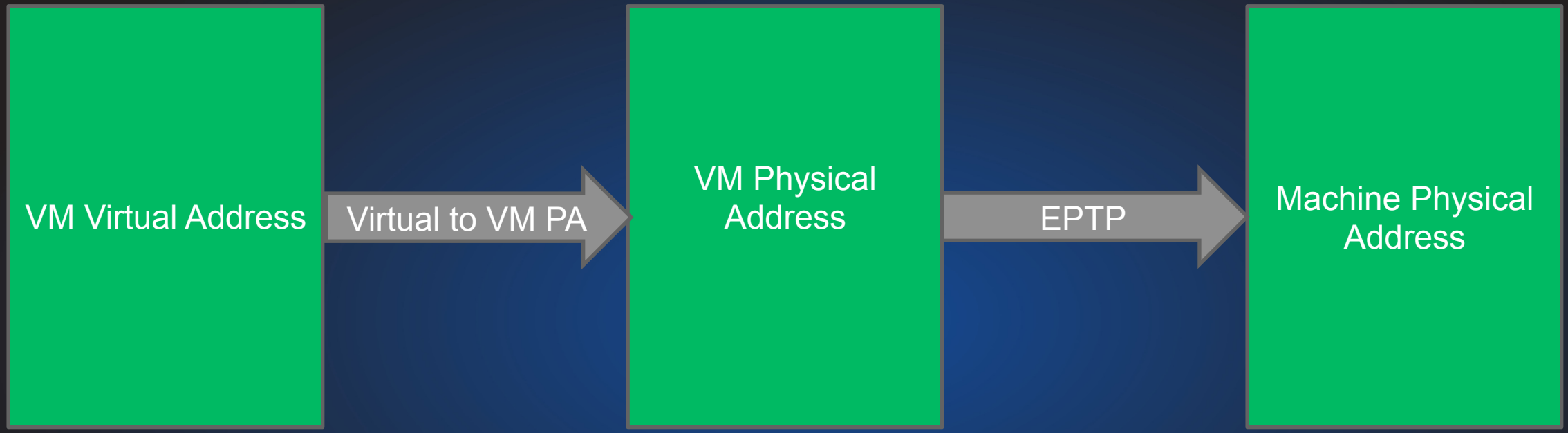
Extended Page table(s)



https://de.wikipedia.org/wiki/Datei:X86_Paging_4K.svg

NUMEN

p2m Translation

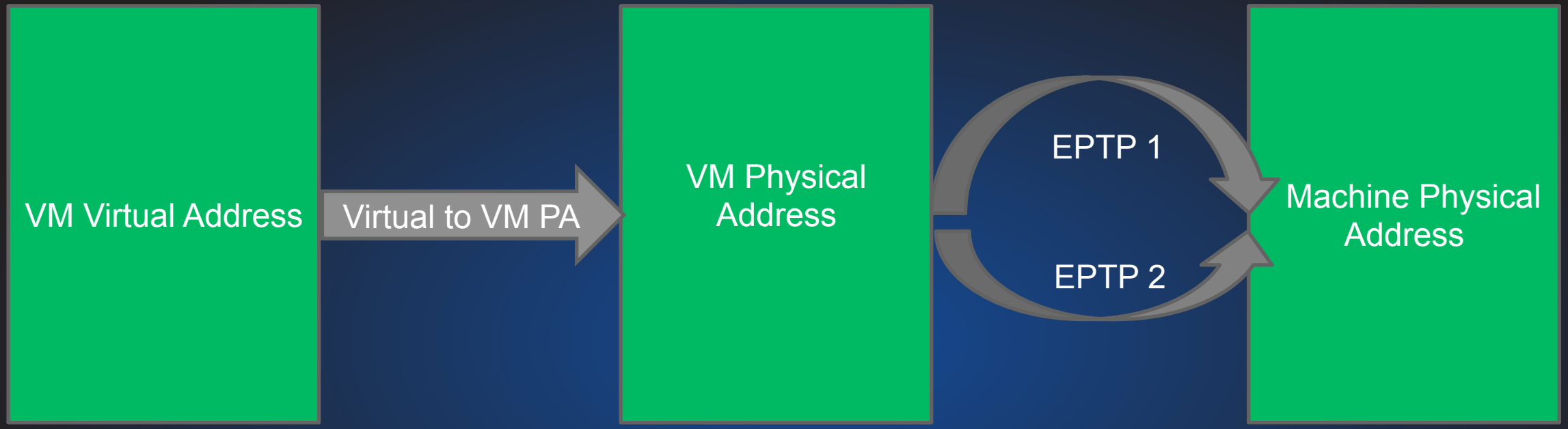



```
347  /* VMCS field encodings. */
348  #define VMCS_HIGH(x) ((x) | 1)
349  enum vmcs_field {
350      VIRTUAL_PROCESSOR_ID            = 0x00000000,
351      POSTED_INTR_NOTIFICATION_VECTOR = 0x00000002,
352      EPTP_INDEX                      = 0x00000004,
353  #define GUEST_SEG_SELECTOR(sel) (GUEST_ES_SELECTOR + (sel) * 2) /* ES ... GS */
354      GUEST_ES_SELECTOR               = 0x00000800,
355      GUEST_CS_SELECTOR               = 0x00000802,
356      GUEST_SS_SELECTOR               = 0x00000804,
```

EPT pointer (EPTP) is stored in the Virtual Machine Control Structure (VMCS) - A per VM data struct in the memory and managed by VMM

NUMEN

Multiple p2m Translations

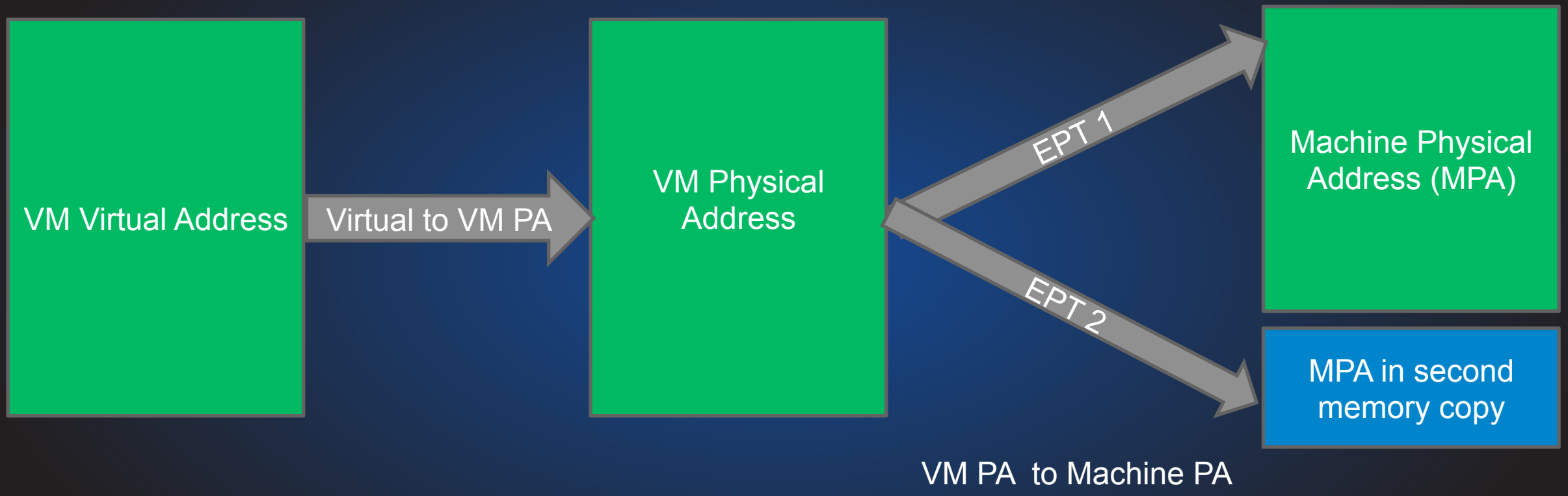



```
26 #include <asm/i387.h>
27 #include <asm/hvm/support.h>
28 #include <asm/hvm/trace.h>
29 #include <asm/hvm/vmx/vmcs.h>
30
31 typedef union {
32     struct {
33         u64 r       :   1,  /* bit 0 - Read permission */
34         w           :   1,  /* bit 1 - Write permission */
35         x           :   1,  /* bit 2 - Execute permission */
36         emt         :   3,  /* bits 5:3 - EPT Memory type */
37         ipat        :   1,  /* bit 6 - Ignore PAT memory type */
38         sp          :   1,  /* bit 7 - Is this a superpage? */
```

Extend Page Table Entry (epte) struct from Xen code

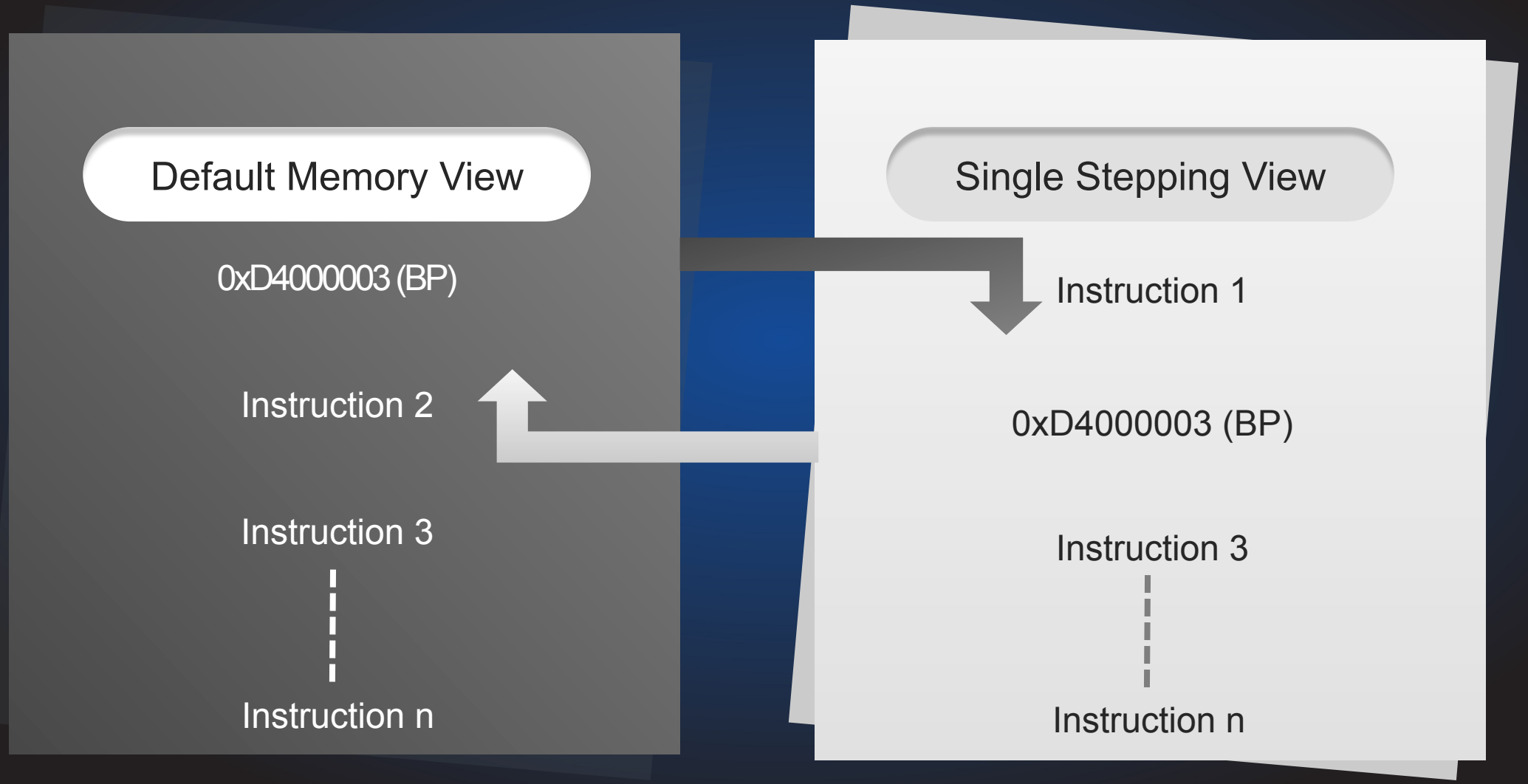
NUMEN

Multiple p2m Translations (continued)



NUMEN

Single Stepping on ARM



BP = Breakpoint = SMC

https://dl.acm.org/citation.cfm?id=3274698

NUMEN

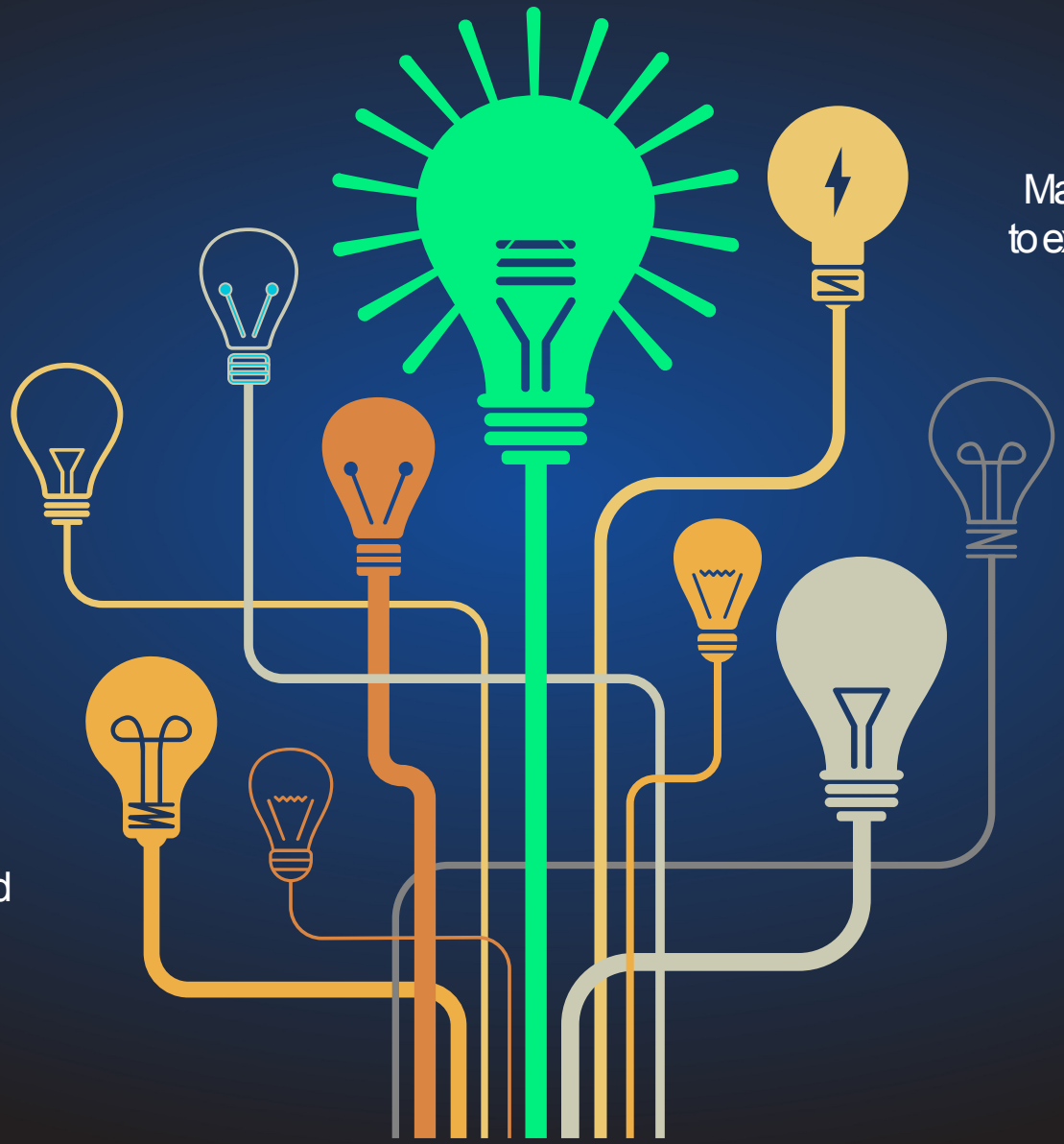
Hooking and Syscall Monitoring on ARM

Add & Register Hook

vmi_register_event()
&
Write to memory 0xD4000003 (SMC) at the start of each API function .

Callback

Do you analysis when the control gets to your registered callback.



Singlestep

Make sure to singlestep in order to execute the original functionality

Clean

After you are done, make sure to remove all hooks and exit VMI. Otherwise the VM might crash or become unstable
vmi_destroy();

NUMEN

Syscalls Monitoring in ARM

(ARM-Syscalls.mp4)

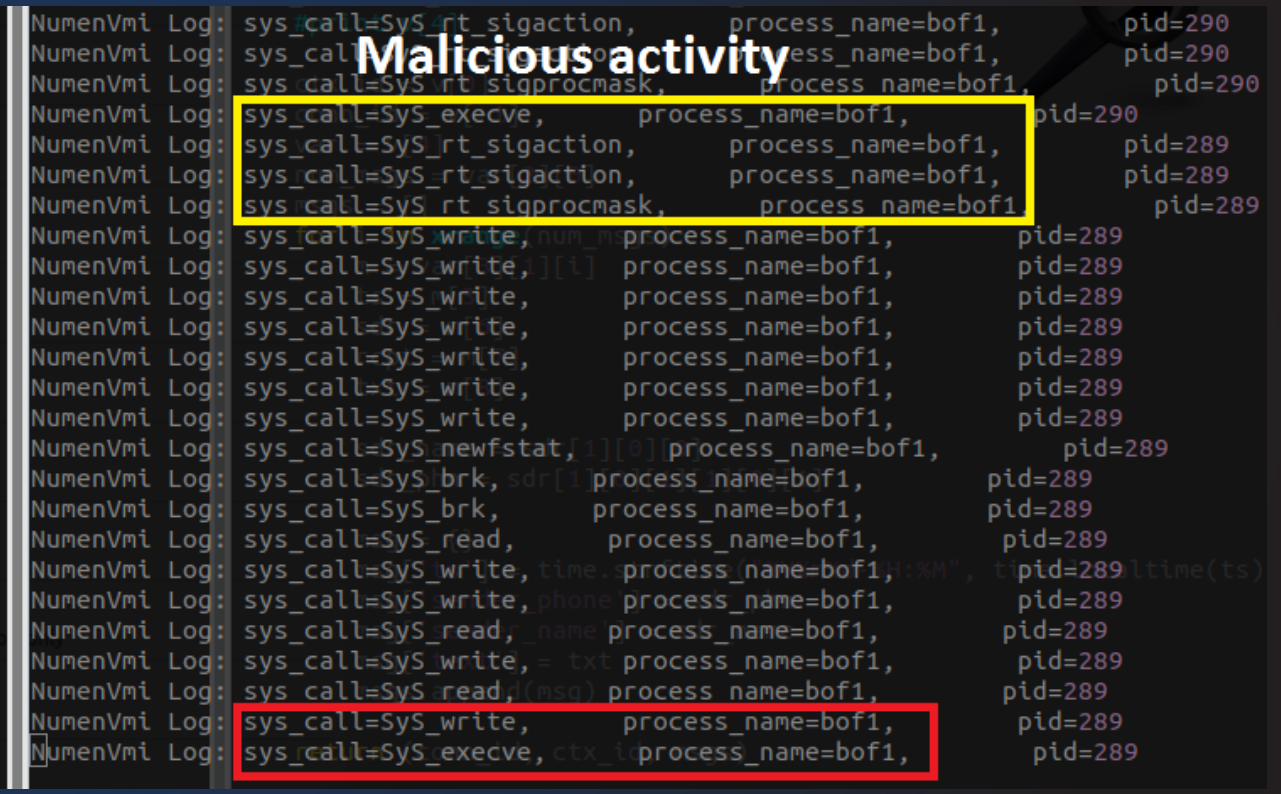


Attacks and Detection scenarios



Memory corruption attack





Exit gracefully

Shell spawn at the end

Easy sequence-based detection

NUMEN

Shellcode execution delay

```
sc =  ( "\xe0\x7f\x40\xb2" +      #mov  x0, <big number>
        "\x00\x04\x00\xf1" +      #subs x0, x0, #0x1
        "\xe0\xff\xff\xb5" +      #cbnz x0, 4
        asm(shellcraft.execve("/bin/sh")) )
```

- Syscall monitoring cannot be on all the time
- Not using syscall (sleep) to delay execution
- Traditional AV challenge

NUMEN

Solution approach

- Create a “triggered memory view” hooking only suspicious syscalls: execve, connect, clone, etc all the time
- As soon as the shellcode spawns, full hooking on that process is enabled!

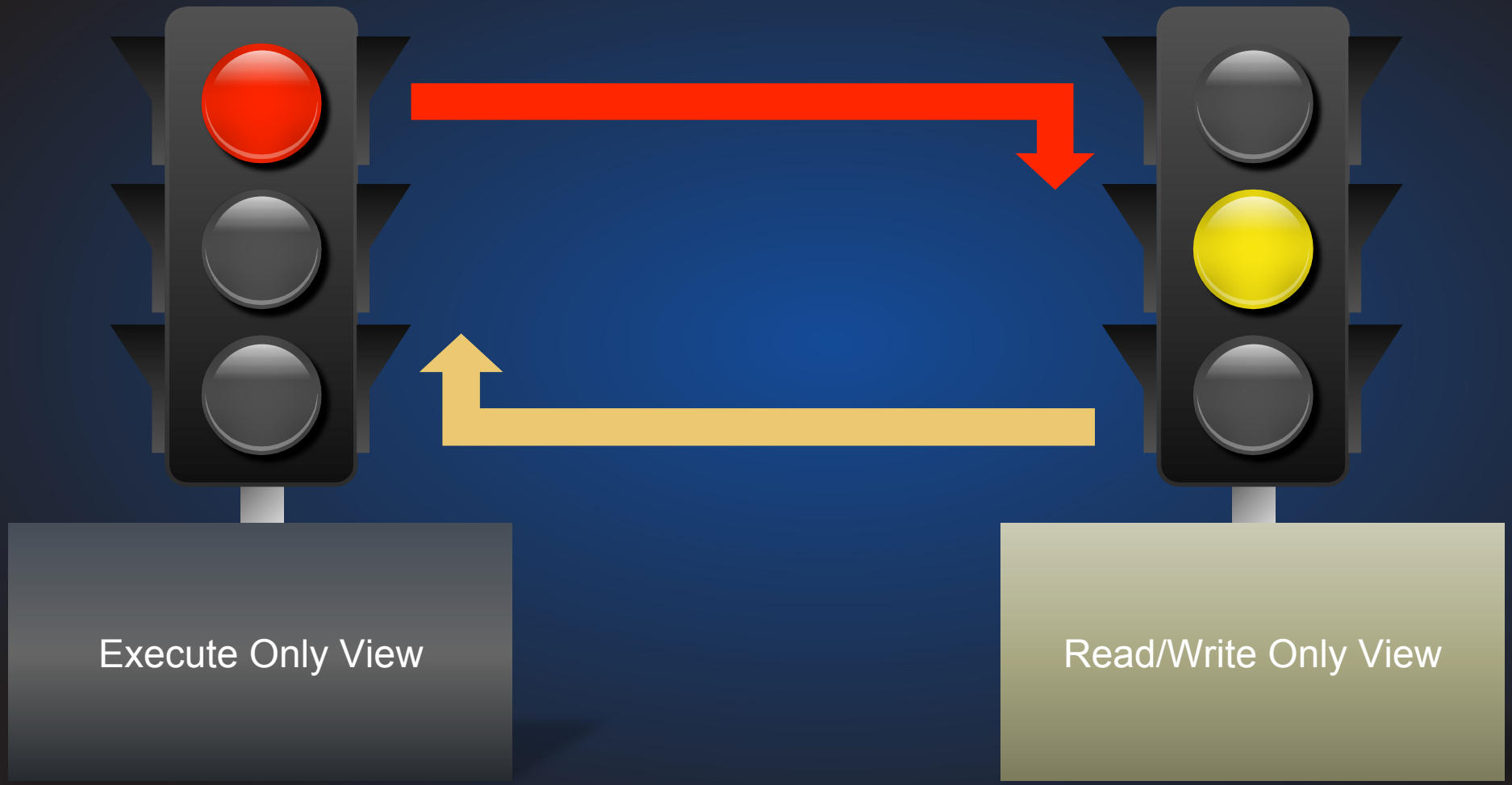
NUMEN

Malware hypervisor-aware

- The malware is able to read kernel memory and identify SMC hooks
  - ✓ Stops running or wipes the system!
- Even in some conditions is able to remove the hooks!
  - ✓ Worst scenario, detection bypass!

NUMEN

Stealthiness using memory views



NUMEN

Policy Enforcement – Network Use Case

Once you have a good handle on Virtual Machine Introspection, there are many possibilities.

1) Traverse a task list and see if there is any socket handle for a particular task struct
    1.1) A socket is a special type of file. So check if there is any additional file handle

2) Hook the network related APIs (e.g. connect).
    2.1) More active approach vs the passive one in step 1.

NUMEN

Policy Enforcement – Network Use Case

```
root@zcu102:/home/ali/policy_Nvmi# ls
Makefile  NInspector  nvmi.h  nvmi_arm.c  nvmi_arm.o
root@zcu102:/home/ali/policy_Nvmi# ./NInspector stretch /home/poc/prod/sys_map-4.15.0-20

                      Numen Introspection Framework v5.0


Alternative Memory Views Created
Memory pages created and configured for ARM single stepping & Adaptive VMI. Monitoring now..

NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=sd-resolve,         pid=204
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=sd-resolve,         pid=204
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=sd-resolve,         pid=204
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
NumenVmi Policy Log: Attempt to communicate outside detected for process_name=systemd-timesyn,        pid=194
```

Our patent pending Numen Adaptive Monitoring (NAM) is a combination of different techniques to achieve exceptional performance

NUMEN

Remediation

- Its not easy to remediate from outside without putting any agent inside. Lets say kill a process.
- How about manipulating with one of the frequently called APIs?
- Maybe make one of the string parameter NULL?
- Just a basic way. There can be other more mature ways.

NUMEN

PRACTICAL RECOMMENDATIONS FOR END TO END SYSTEM

- Software Breakpoints
- Efficient Single Stepping Mechanism
- Event Mechanism
- Efficient translations caching
- Multiple mappings support for p2m (physical to machine)
- Memory page permissions management



Releasing tool to the public

- Tool to perform syscall monitoring for ARM & Intel ☺
- All files needed to setup a working environment:
  - ✓ Booting the board: zynqmp_fsbl.elf, u-boot.elf, bl31.elf, pmufw.elf
  - ✓ Environment: xen.dtb, Kernel-Image, Xen-Hypervisor (version 11.0), DomU-Configuration files, xen startup scripts.
  - ✓ Test: ARM64-based malware and exploit samples.
- Dropbox link: xxxxxxxxxxxxxxxxxxxxx

NUMEN

Takeaways

- “Smart” Hypervisors on ARM are needed, not only for isolation
- ARM Syscall Hooking is great achievement but just the beginning, the detection strategies is what makes the difference
- Switching between memory views for detection strategies is a new way to detect maliciousness from VMI

NUMEN

Special Thanks

- Stefano Stabellini: For his great help on Xen troubleshooting
- Matt Leinhos: For his great features on ARM/Intel VMI
- For those 3 of you guys, you know who you are ☺

Without you, no way to complete this effort



Q & A

@Ali_Islam_Khan
@danuxx

