Breaking virtualization by switching the cpu to virtual 8086 mode

Jonathan Brossard
CEO – Toucan System

jonathan@
toucan-system.com

Agenda

- Virtualization : big picture
- Attack surface analysis
- The need for new tools
- Introducing Virtual 8086 mode
- Practical fuzzing with vm86()

Virtualization : big picture

Market shares
Definitions

Virtualization : market shares

Source : Forrester Research 2009

78% of companies have production servers virtualized.

20% only have virtualized servers.

Virtualization : market shares

Source : Forrester Research 2009

VMWare is present in **98%** of the companies.
Microsoft virtualization products are used by 17%.
Citrix/Xen is used by 10%.

In a nutshell...

- As widespread as Apache or Bind
  - Proprierary software + very few builds + weak toolchains so it runs with other toolchains (no SSP, etc) = reliable exploitation.
- You don't need a « remote » exploit : you <u>buy</u> a shell at the same hosting provider.

Usage

- Cost reduction (shared hosting)
- Scalability (cloud computing)
- Run broken applications on broken Oses (legacy).

Definitions

Virtualization : Definitions

Virtualization

Virtualization is the name given to the simulation with higher level components, of lower level components.

NOTE: Virtualization of applications (as opposed to full Oses) is out of topic.

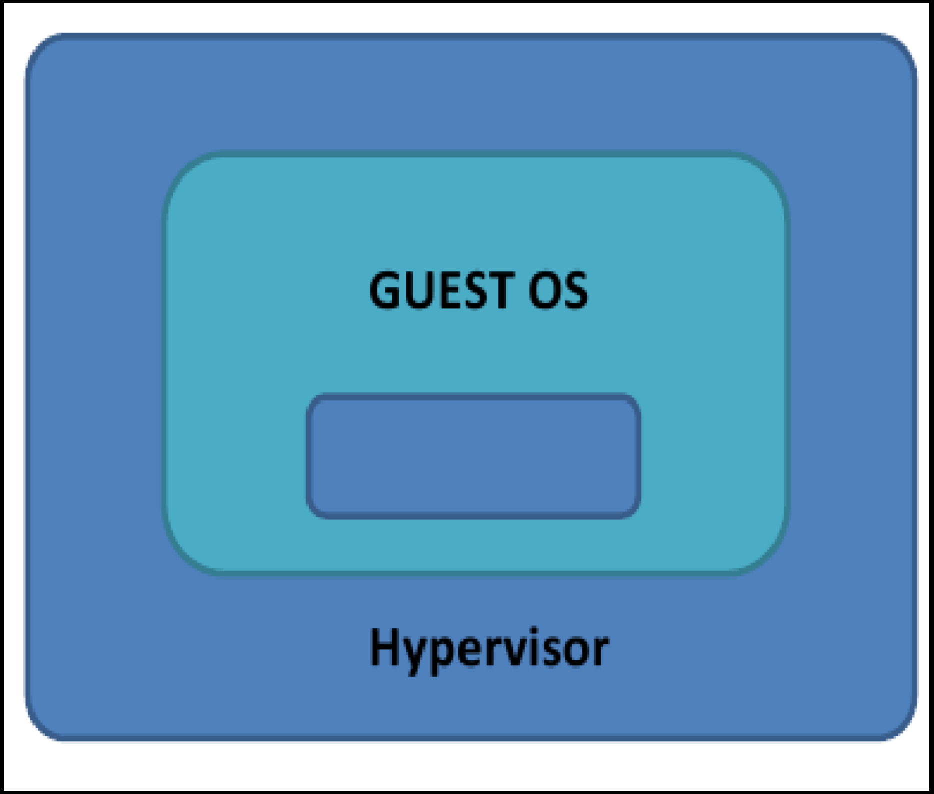
Virtualization : Definitions

Virtual Machine

A **virtual machine** (VM) is : "an efficient, isolated duplicate of a real machine".

-- Gerald J. Popek and Robert P. Goldberg (1974). "Formal Requirements for Virtualizable Third Generation Architectures", Communications of the ACM.

Paravirtualization



Virtualization : Definitions

Paravirtualization

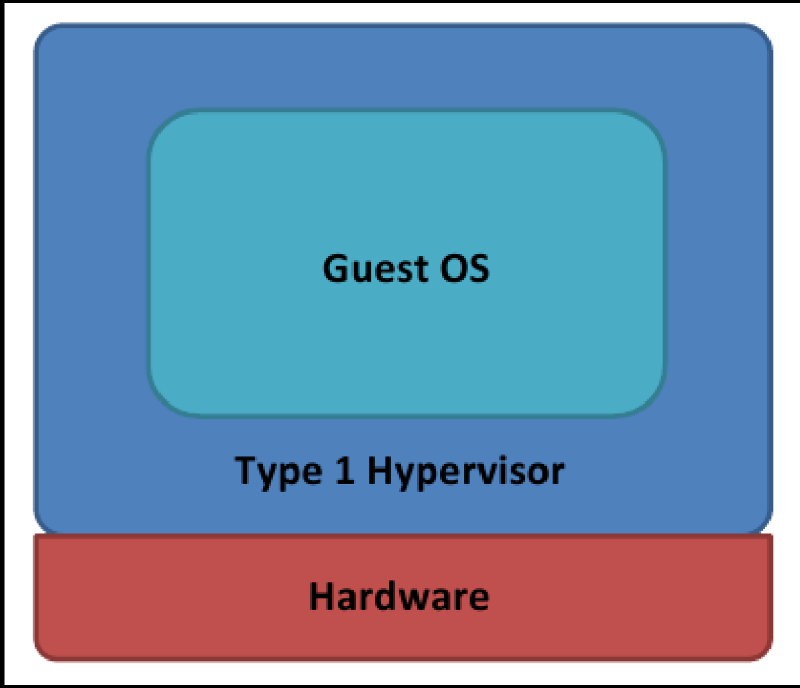
Requires the modification of the guest Oses (eg: Xen, UML, Qemu with kquemu, VMWare Workstation with VMWare Tools).

Opposed to « full virtualization ».

Virtualization : Definitions

There are two types of virtualizations : Virtual Machine Monitors (or **Hypervisors**) of **type I** and **type II**.

Type I Hypervisor

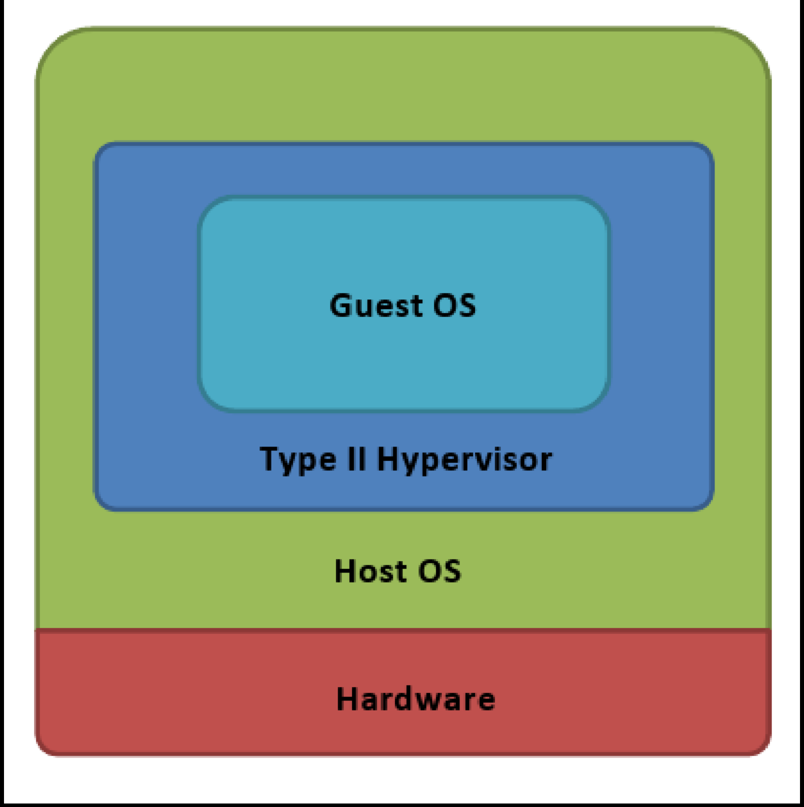


Virtualization : Definitions

Hypervisors of type I

Run on bare metal (eg: Xen, Hyper-V, VMWare ESX).

Type II hypervisor

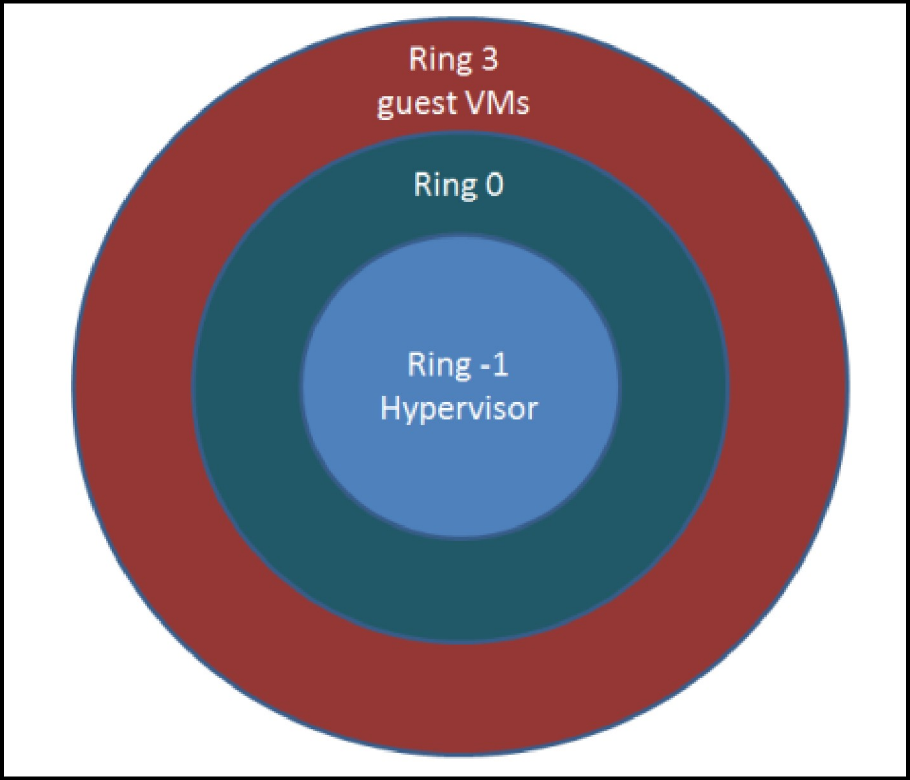


Virtualization : Definitions

Hypervizors of type II

Run as a process inside a host OS to virtualize guests Oses (eg: Qemu, Virtualbox, VMWare Workstation, Parallels).

Hardware assisted virtualization



Hardware assisted virtualization

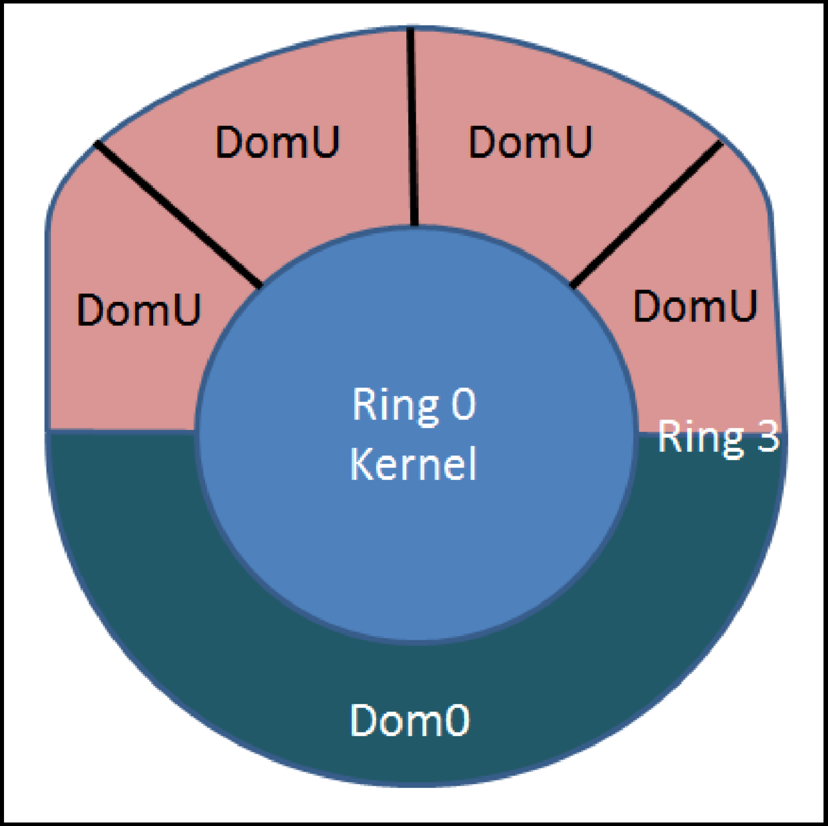
- Takes advantage of AMD-V On Intel VT-x CPU extentions for virtualization.
- x64 Only.
- The hypervisor is running in « ring -1 ».
- Much like the NX bit : requires the motherboard to support it and activation in the BIOS.

Virtualization : Definitions

Isolation

Isolation of the userland part of the OS to simulate independant machines (eg: Linux-Vservers, Solaris « Zones », BSD « jails », OpenVZ under GNU/Linux).

Isolation



Attack surface analysis

Depending on your perspective...

What are the risks ?
Where to attack ?

Privilege escalation on the host

Exemple :

VMware Tools HGFS Local Privilege Escalation Vulnerability

(http://labs.idefense.com/intelligence/vulnerabilities/display.php?id=712)

Demos

Attacking setuid binaries in vmware.

Virtual machines file fuzzing on virtualbox.

Privilege escalation on the Guest

Exemple :

CVE-2009-2267 « Mishandled exception on page fault in VMware » Tavis Ormandy and Julien Tinnes

Demo

Overwriting the MBR under vserver and instrumenting the original bootloader with keyboard/motherboard PIC programming through ioports

(see also « Invisible Man » tool from Jonathan Brossard, Defcon 2008).

Attacking other guests

Exemple:

Vmare workstation guest isolation weaknesses (clipboard transfer)

http://www.securiteam.com/securitynews/5GP021FKKO.html

DoS (Host + Guests)

Exemple:

CVE-2007-4591 CVE-2007-4593 (bad ioctls crashing the Host+Guests)

Demo

Attacking vmware kernel modules with ioctls.

Escape to host

Exemples :

Rafal Wojtczuk (Invisible things, BHUS 2008)
Kostya Kortchinsky (« Cloudburst », BHUS 2009).

IDEFENSE VMware Workstation Shared Folders Directory Traversal Vulnerability (CVE-2007-1744)

Escape to host

This is the real hard thing, and what we will focus on on the rest of this talk.

Attack surface analysis : usage

Hosting two companies on the same hardware is <u>very common</u> (shared hosting).

Getting a shell on the same machine as a given target may therefor be a matter of paying a few euros a month.

Attack surface : conclusion

Rooting the Host OS from the Guest is practical (Kostya Kortchinsky BHUS 2009, Rafal Wojtczuk BHUS 2008).

Seemingly minor bugs (local, DoS) do matter : virtualization amplifies consequences.

Note : public, reliable, packed attack tools exist (Claudio Criscione, HITB Kuala Lumpur 2010)

The need for dedicated methodologies and tools

The need for new tools : example

How do you attack a hard drive with software ?
What about a screen or a keyboard ?

=> Unusual attack surface.

How to dynamically test a virtual Hard Drive ? Naive approach

Standard API :

```
ssize_t read(int fd, void *buf, size_t count);
ssize_t write(int fd, const void *buf, size_t count);
```

This would mostly fuzz the kernel, not the Virtual Machine :(

We need something (much) lower level.

(low level) attack vectors

Ioports:

outb, outw, outl, outsb, outsw, outsl, inb, inw, inl, insb, insw, insl, outb_p, outw_p, outl_p, inb_p, inw_p, inl_p

Problems: sequence, multiple ports

Ioctls:

int ioctl(int d, int request, ...)

Problems : arbitrary input size !

Demos

Attacking Vmware's Direct Memory Access (DMA) with ioports fuzzing.

How did we used to do it « back in the days » ?

MS Dos : direct access to the hardware (interrupts : BIOS, HD, Display, …)

Can we get back to this ?

Introducing the Virtual 8086 mode

Introducing the Virtual 8086 mode

Introduced with Intel 386 (1985)

Purpose : run 16b applications under 32b Operationg systems (eg : MS DOS COM files under Windows).

Introducing the Virtual 8086 mode

Intel x86 cpus support 3 main modes

- Protected mode
- Real mode
- System Management Mode (SMM)

Nice things about Real mode / Virtual 8086 mode

Direct access to hardware via interruptions !

example:

```
Mov ah, 0x42 ; read sector from drive
Mov ch, 0x01 ; Track
Mov cl, 0x02  ; Sector
Mov dh, 0x03 ; Head
Mov dl, 0x80  ; Drive (here first HD)
Mov bx, offset buff ; es:bx is destination

Int 0x13          ; hard disk operation
```

Complexity

ax*bx*cx*dx (per interruption)

Id est: [0;65535]^4 ~ 1.8 * 10^19

=> still huge. But it's possible to call every function on every device (just not with all possible parameters).

=> much better than ioctl()'s arbitrary input length !

Attacking a Hard Disk : exemple functions tested

Int 13/AH=00h - DISK - RESET DISK SYSTEM
Int 13/AH=01h - DISK - GET STATUS OF LAST OPERATION
Int 13/AH=02h - DISK - READ SECTOR(S) INTO MEMORY
Int 13/AH=03h - DISK - WRITE DISK SECTOR(S)
Int 13/AH=04h - DISK - VERIFY DISK SECTOR(S)
Int 13/AH=05h - FLOPPY - FORMAT TRACK
...
Int 13/AH=09h - HARD DISK - INITIALIZE CONTROLLER WITH DRIVE PARAMETERS
Int 13/AH=0Ah - HARD DISK - READ LONG SECTOR(S) (AT and later)
Int 13/AH=0Bh - HARD DISK - WRITE LONG SECTOR(S) (AT and later)
Int 13/AH=0Ch - HARD DISK - SEEK TO CYLINDER
Int 13/AH=0Dh - HARD DISK - RESET HARD DISKS
Int 13/AH=0Eh - HARD DISK - READ SECTOR BUFFER (XT only)
Int 13/AH=0Fh - HARD DISK - WRITE SECTOR BUFFER (XT only)
Int 13/AH=10h - HARD DISK - CHECK IF DRIVE READY
Int 13/AH=11h - HARD DISK - RECALIBRATE DRIVE
Int 13/AH=12h - HARD DISK - CONTROLLER RAM DIAGNOSTIC (XT,PS)
...

Attacking a Hard Disk : model/vendor specific ints

Int 13/AH=43h - IBM/MS INT 13 Extensions - EXTENDED WRITE
Int 13/AH=44h - IBM/MS INT 13 Extensions - VERIFY SECTORS
Int 13/AH=45h - IBM/MS INT 13 Extensions - LOCK/UNLOCK DRIVE
Int 13/AH=46h - IBM/MS INT 13 Extensions - EJECT MEDIA
Int 13/AH=47h - IBM/MS INT 13 Extensions - EXTENDED SEEK
Int 13/AH=48h - IBM/MS INT 13 Extensions - GET DRIVE PARAMETERS
Int 13/AH=49h - IBM/MS INT 13 Extensions - EXTENDED MEDIA CHANGE
…

Int 13/AX=5504h - Seagate ST01/ST02 - RETURN IDENTIFICATION
Int 13/AX=5505h - Seagate - ??? - PARK HEADS
Int 13/AX=5505h - Seagate ST01/ST02 - PARK HEADS
Int 13/AX=5506h - Seagate ST01/ST02 - SCSI Bus Parity
Int 13/AX=5507h - Seagate ST01/ST02 - RESERVED FUNCTIONS

How to attack a Hard drive ?

By calling all those functions, our coverage is much much better than if we just had used read()/write() on disk.

How to attack a keyboard ?

Int 16/AH=00h - KEYBOARD - GET KEYSTROKE
Int 16/AH=01h - KEYBOARD - CHECK FOR KEYSTROKE
Int 16/AH=02h - KEYBOARD - GET SHIFT FLAGS
Int 16/AH=03h - KEYBOARD - SET TYPEMATIC RATE AND DELAY
Int 16/AH=04h - KEYBOARD - SET KEYCLICK (PCjr only)
Int 16/AH=09h - KEYBOARD - GET KEYBOARD FUNCTIONALITY
Int 16/AH=0Ah - KEYBOARD - GET KEYBOARD ID
Int 16/AH=10h - KEYBOARD - GET ENHANCED KEYSTROKE
Int 16/AH=11h - KEYBOARD - CHECK FOR ENHANCED KEYSTROKE
Int 16/AH=12h - KEYBOARD - GET EXTENDED SHIFT STATES
Int 16/AH=20h - KEYBOARD - GET 122-KEY KEYSTROKE
Int 16/AH=21h - KEYBOARD - CHECK FOR 122-KEY KEYSTROKE
Int 16/AH=22h - KEYBOARD - GET 122-KEY SHIFT STATUS
...

How to attack a screen ?

Int 10/AH=00h - VIDEO - SET VIDEO MODE
Int 10/AX=0070h - VIDEO - Everex Micro Enhancer EGA/Viewpoint VGA - EXTENDED MODE SET
Int 10/AX=007Eh - VIDEO - Paradise VGA, AT&T VDC600 - SET SPECIAL MODE
Int 10/AX=007Fh/BH=00h - VIDEO - Paradise VGA, AT&T VDC600 - SET VGA OPERATION
Int 10/AX=007Fh/BH=01h - VIDEO - Paradise VGA, AT&T VDC600 - SET NON-VGA OPERATION
Int 10/AX=007Fh/BH=02h - VIDEO - Paradise VGA, AT&T VDC600 - QUERY MODE STATUS
Int 10/AX=007Fh/BH=03h - VIDEO - Paradise VGA, AT&T VDC600 - LOCK CURRENT MODE
Int 10/AX=007Fh/BH=04h - VIDEO - Paradise VGA, AT&T VDC600 - ENTER MDA EMULATION MODE
Int 10/AX=007Fh/BH=05h - VIDEO - Paradise VGA, AT&T VDC600 - ENTER CGA EMULATION MODE
Int 10/AX=007Fh/BH=06h - VIDEO - Paradise VGA, AT&T VDC600 - ENTER MONOCHROME VGA MODE
Int 10/AX=007Fh/BH=07h - VIDEO - Paradise VGA, AT&T VDC600 - ENTER COLOR VGA MODE
...

Introducing the Virtual 8086 mode

Problem is… is this even possible inside a virtual machine ?

Introducing the Virtual 8086 mode

The kernel boots in (16b) real mode, and then switches to protected mode (32b).

The cpu normally doesn't get back to real mode untill next reboot.

Introducing the Virtual 8086 mode

Corollary

The hypervisor could run under any mode. protected mode in practice (being it ring0, ring1 or ring3).

All of the guests run <u>only</u> in protected mode.

Now how to swith to Virtual 8086 mode ? It this even possible ?

Leaving protected mode ?

```
         -------------------                    SMI (interrupt)
      |->|Real Address Mode| -------------------------------------------|
      |  ------------------- <----------------------------------|       |
      |  | PE=1    ^ PE=0 (requires ring0) or                   |rsm or |
      |  v         | reset                                      |reset  V
      |  -------------------                                    ---------
reset |  | Protected Mode  | -------> SMI (interrupt) ------> | SMM Mode |
      |  ------------------- <------- rsm instruction <------   ---------
      |  | VM=1    ^ VM=0                                       |       ^
      |  v         |                                            |rsm    |
      |  ------------------- <----------------------------------|       |
      |- |Virtual 8086 Mode| -------------------------------------------|
         -------------------                    SMI (interrupt)
```

(Ascii Art : Courtesy of phrack 65)

Setting the VM flag in CR0 under protected mode would get us to Virtual Mode
Removing the PE flag from CR0 would get us back to real mode

Leaving protected mode ?

linux-2.6.31/arch/x86/kernel/reboot.c:

```
static const unsigned char real_mode_switch [] =
{
    0x66, 0x0f, 0x20, 0xc0,                      /*    movl  %cr0,%eax        */
    0x66, 0x83, 0xe0, 0x11,                      /*    andl  $0x00000011,%eax */
    0x66, 0x0d, 0x00, 0x00, 0x00, 0x60,          /*    orl   $0x60000000,%eax */
    0x66, 0x0f, 0x22, 0xc0,                      /*    movl  %eax,%cr0        */
    0x66, 0x0f, 0x22, 0xd8,                      /*    movl  %eax,%cr3        */
    0x66, 0x0f, 0x20, 0xc3,                      /*    movl  %cr0,%ebx        */
    0x66, 0x81, 0xe3, 0x00, 0x00, 0x00, 0x60,    /*    andl  $0x60000000,%ebx */
    0x74, 0x02,                                  /*    jz    f                */
    0x0f, 0x09,                                  /*    wbinvd                 */
    0x24, 0x10,                                  /* f: andb  $0x10,al         */
    0x66, 0x0f, 0x22, 0xc0                       /*    movl  %eax,%cr0        */
};
```

Trouble is...

This obviously won't work inside a virtual machine !

Because CR[1-4] registers are themselves emulated

IS THIS « GAME OVER » ?

Actually not quite ...

Truth is : we don't need to switch back to real mode/virtual 8086 mode !

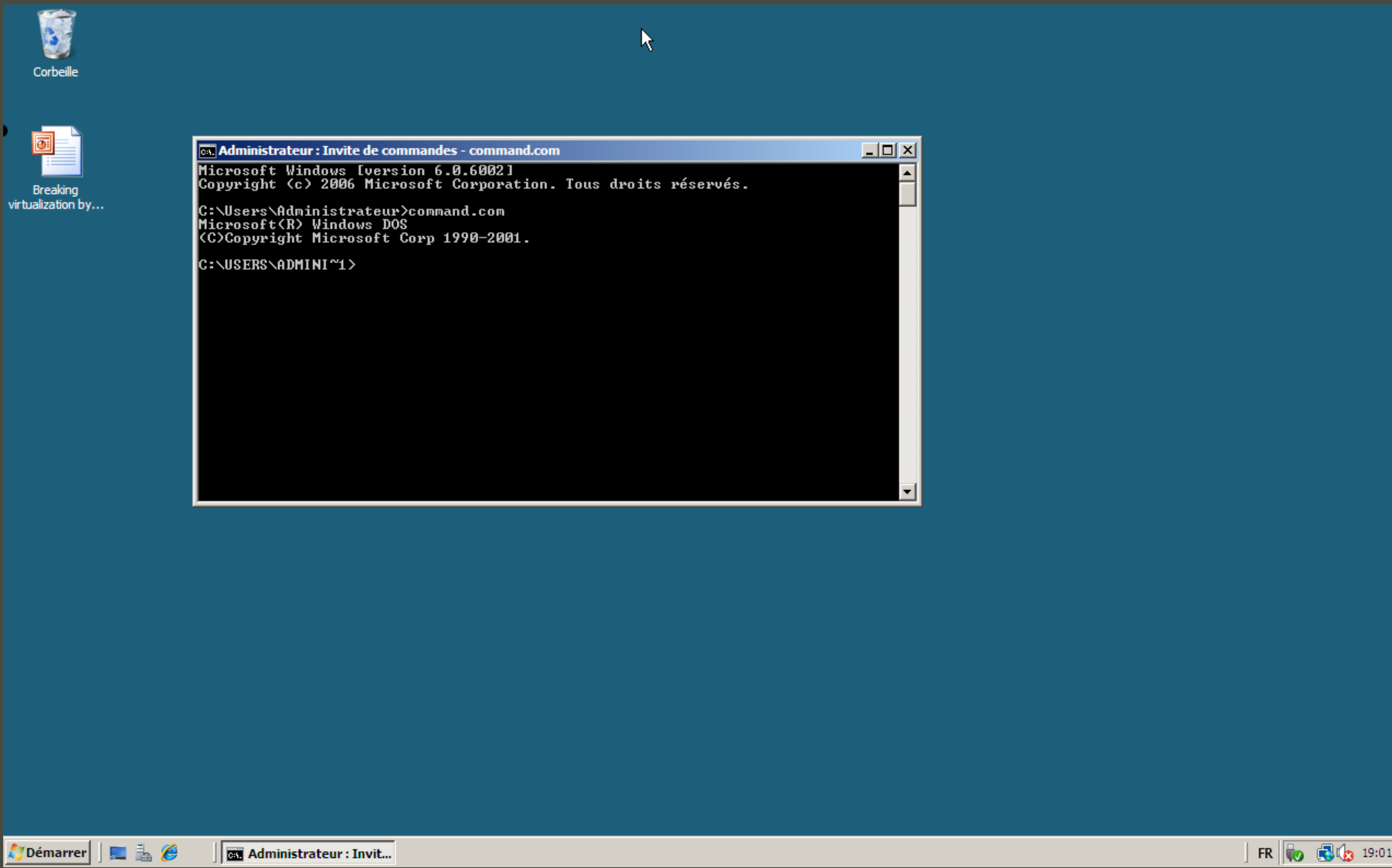
Most Operating systems offer a way to run 16b applications (eg: MS DOS) under protected mode by emulating a switch to Virtual 8086 Mode.

Notably Windows (x86) and Linux (x86).

The Windows case

NTVDM : ntvdm.exe

« Windows 16b Virtual Machine »


Corbeille
Breaking virtualization by...
Administrateur : Invite de commandes - command.com
Microsoft Windows [version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation. Tous droits réservés.
C:\Users\Administrateur>command.com
Microsoft(R) Windows DOS
(C)Copyright Microsoft Corp 1990-2001.
C:\USERS\ADMINI~1>
Démarrer
Administrateur : Invit...
FR
19:01

The Linux case

The linux kernel provides an emulation of real mode in the form of two syscalls:

```
#define __NR_vm86old     113
#define __NR_vm86        166
```

The Linux case

```
#include <sys/vm86.h>

int vm86old(struct vm86_struct *info);

int vm86(unsigned long fn, struct
   vm86plus_struct *v86);
```

```
struct vm86_struct {
	struct vm86_regs regs;
	unsigned long flags;
	unsigned long screen_bitmap;
	unsigned long cpu_type;
	struct revectored_struct
			int_revectored;
	struct revectored_struct
	int21_revectored;
};
```

The Linux case

linux-2.6.31/arch/x86/include/asm/vm86.h:

```
struct vm86_regs {
        long ebx;
        long ecx;
        long edx;
        long esi;
        long edi;
        long ebp;
        long eax;
    (…)
        unsigned short es, __esh;
        unsigned short ds, __dsh;
        unsigned short fs, __fsh;
        unsigned short gs, __gsh;

};
```

In a nutshell

- The switch to Virtual mode is entirely emulated by the kernel (this will work inside a VM)
- We can still program using old school interruptions (easy !)
- Those interruptions are delivered to the hardware (id est: either the emulated one, or the real one).

=> We just got a « bare metal (possibly virtualized) hardware interface »

The x64 case…

The x64 case

X64 cpus in 64b long mode can't swith to Virtual mode.

That's too bad : we'd like to fuzz latest Vmware ESX or Microsoft HyperV (necessarily under x64).

But under virtualization, the switch to VM86 mode is being emulated by the kernel...

The x64 case

Using kernel patches, we can add VM86 capabilities to a x64 GNU/Linux kernel.

EG: http://v86-64.sourceforge.net to run Dosemu under x64.

What's not possible in real hardware becomes possible under a virtualized environment !

Practical use : Fuzzing using vm86()

Practical use : Fuzzing using vm86()

Looking at the IVT allows us to fuzz all the hardware detected after BIOS Post, efficently (no calls to empty/dummy interrupts).

Practical use : Fuzzing using vm86()

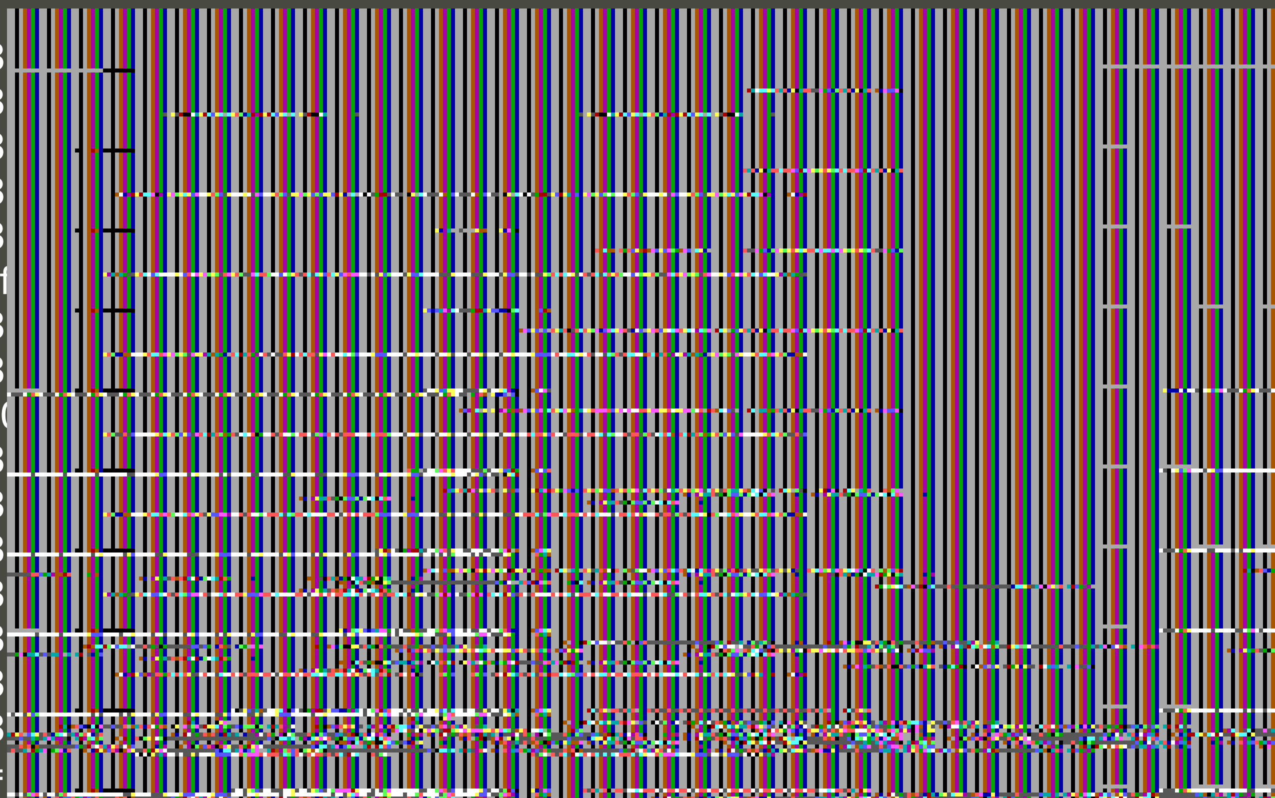
Exemple bugs !

Practical use : Fuzzing using vm86()

Bugs in hypervizors...

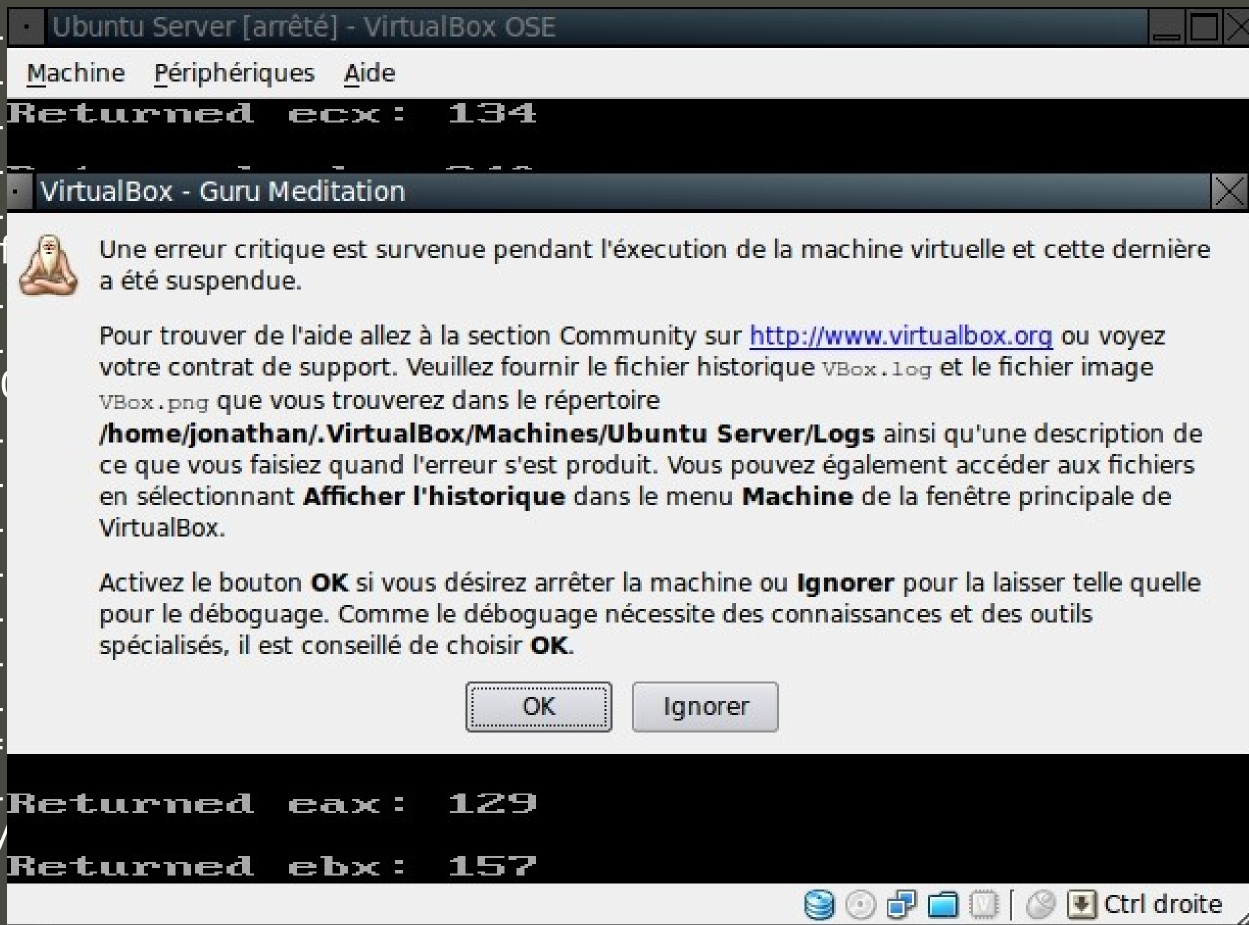
Virtualbox

00:21:13
00:21:13
00:21:13
00:21:13
00:21:13 000
EIP=f
00:21:13
00:21:13
_ZL1(
00:21:13
00:21:13
00:21:13
00:21:13
00:21:13
00:21:13
00:21:13 0
.esi=



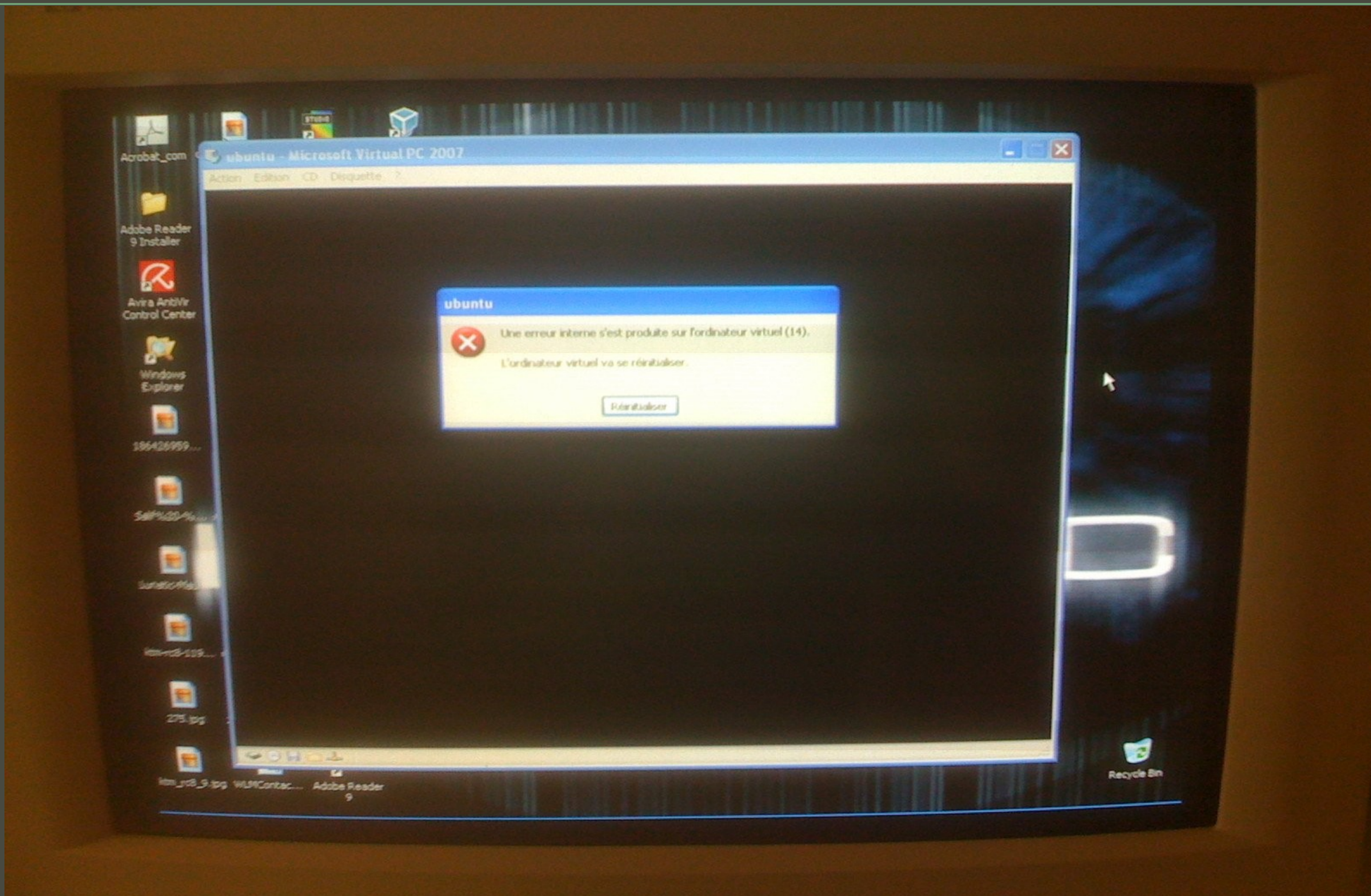
Virtualbox (take 2)

00:02:51
00:02:51
00:02:51)
00:02:51
00:02:51 ab000
EIP=f
00:02:51
00:02:51
_ZL1
00:02:51
00:02:51
00:02:51
00:02:51
00:02:51
00:02:51
00:02:51 00
.esi=
00:02:51 v up di nt zr ac
pe cy



More (guest) bugs

Virtual PC



ubuntu - Microsoft Virtual PC 2007
Action Edition CD Disquette ?
ubuntu
Une erreur interne s'est produite sur l'ordinateur virtuel (14).
L'ordinateur virtuel va se réinitialiser.
Réinitialiser
Acrobat_com
Adobe Reader 9 Installer
Avira AntiVir Control Center
Windows Explorer
Recycle Bin

Parallels (Guest)

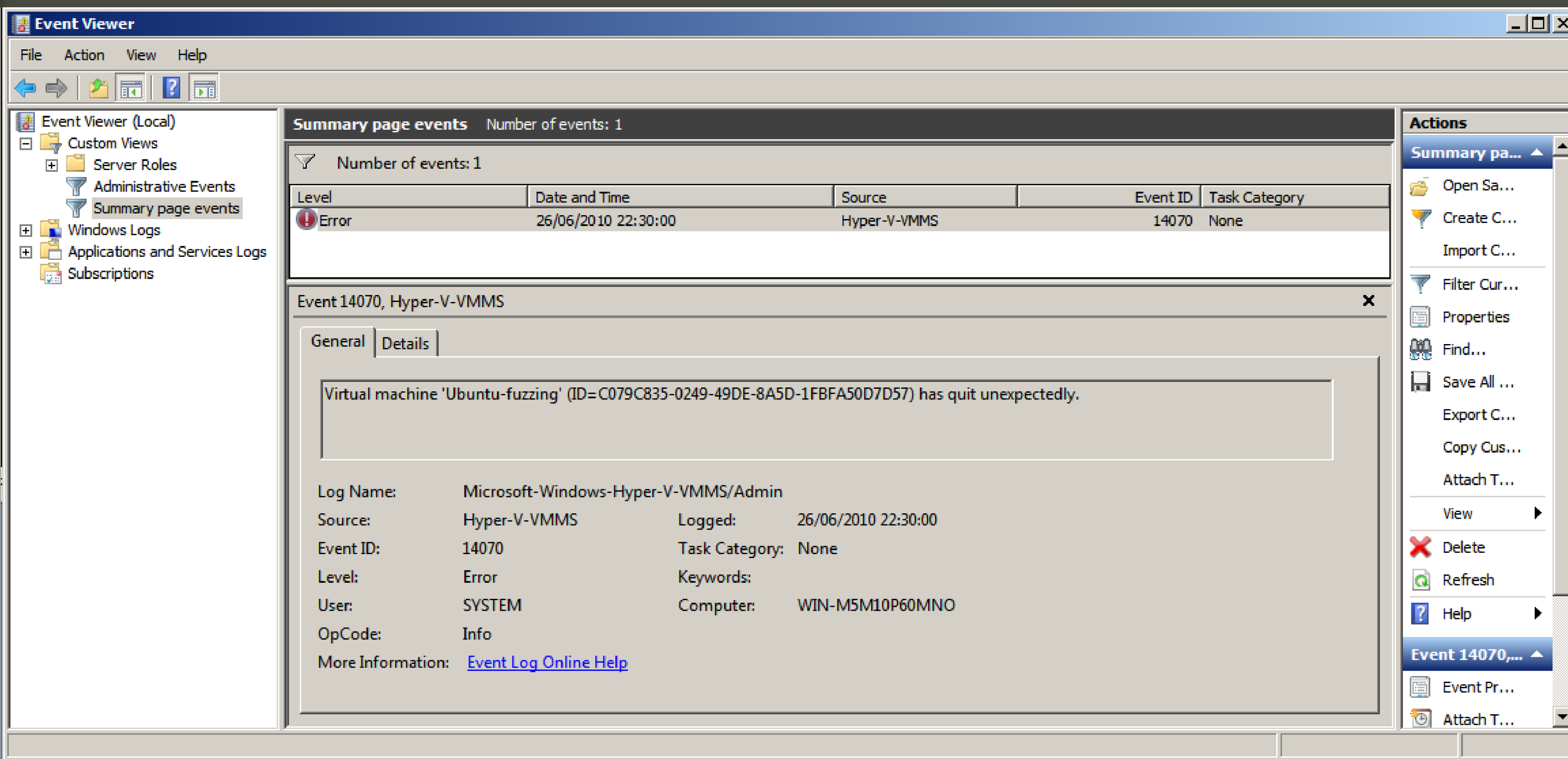
-------------- Guest processor state ---------------
Inhibit Mask=0

CS=FF63 [0000FFFF 0000F30F] V=1
SS=FFD3 [0000FFFF 00CF9300] V=1
DS=0018 [0000FFFF 00CFF300] V=1
ES=0018 [0000FFFF 00CFF300] V=1
FS=FF9B [0000FFFF 00CF9300] V=1
GS=0018 [0000FFFF 00CF9300] V=1

EAX=000000A9 EBX=00005148 ECX=0000F686 EDX=0000000B
ESI=00002D72 EDI=000007E4 EBP=00002E99 ESP=00000FFA
EIP=0000FE96 EFLAGS=00023202

What about x64 ?

Attacking Microsoft HyperV



DEMO

Fuzzing the virtualised hardware though VM86 mode under Vmware an Virtualbox.

Practical use : Fuzzing using vm86()

Exemple bugs !

Attacking hot-plugged hardware ?

=> We need PCI capabilities.

Attacking hot-plugged hardware ?

- PCI fuzzing over VM86:

Int 1A/AX=B102h - PCI BIOS v2.0c+ - FIND PCI DEVICE
Int 1A/AX=B103h - PCI BIOS v2.0c+ - FIND PCI CLASS CODE
Int 1A/AX=B106h - PCI BIOS v2.0c+ - PCI BUS-SPECIFIC OPERATIONS
Int 1A/AX=B108h - PCI BIOS v2.0c+ - READ CONFIGURATION BYTE
Int 1A/AX=B109h - PCI BIOS v2.0c+ - READ CONFIGURATION WORD
Int 1A/AX=B10Ah - PCI BIOS v2.0c+ - READ CONFIGURATION DWORD
...

- Or dedicated PCI configuration/memory fuzzer.

Limitations of VM86 fuzzing

Hardware has to been accesible though interrupts (typically known at BIOS POST).

Thank you for coming

Questions ?