VT入门

转载

litterflybug 于 2013-10-30 18:04:36 发布 1789 收藏
分类专栏： vbox

 vbox 专栏收录该内容
6 篇文章 0 订阅
订阅专栏

看雪安全论坛 > Windows平台 > 『编程技术』
【原创】VT入门---- 闭门造VT [有码]
用户名 记住 忘记密码?
密 码

KSSD 注册账号 搜索论坛 日历事件 论坛帮助

网监张局

【原创】VT入门---- 闭门造VT [有码]

标 题: 【原创】VT入门---- 闭门造VT [有码]
作 者: 网监张局
时 间: 2011-12-24,18:36:53
链 接: http://bbs.pediy.com/showthread.php?t=144656

前言

传说中的VT貌似很神秘的样子,关于VT入门的资料又很少,于是研究了一番
由于资源有限,自身水平亦有限,并且是闭门造车之作,如有错误的地方请指正,不胜感激!
关于VT可以先参考海风月影写的关于VT调试器http://bbs.pediy.com/showthread.php?t=96122

运行环境
操作系统: windows XP
CPU : intel i3-390M
状态: 单核运行

驱动没有卸载部分,测试前请先保存好文档,在 boot.ini 文件中添加 /numproc=1 重启

驱动没有卸载部分,调试前请先保存好文档.在 boot.ini 文件中添加 /numproc=1,重启
VT简介

Intel VIrtual Techonlogy , intel 硬件虚拟化技术 ,在硬件级别上完成计算机的虚拟化

为实现硬件虚拟化 ,VT增加了 12条新的 VMX指令

[VMCS控制 5 条]
VMPTRLD
VMPTRST
VMCLEAR
VMREAD
VMWRITE

[VMX命令 5条]
VMLAUNCH
VMCALL
VMXON
VMXOFF
VMRESUME

[Guest software 2条]
INVEPT
INVVPID

12条指令对应的机器码(xen-3.4.1\xen\include\asm-x86\hvm\vmx\vmx.h)
代码中使用 _emit

```
#define VMCALL_OPCODE ".byte 0x0f,0x01,0xc1\n"
#define VMCLEAR_OPCODE ".byte 0x66,0x0f,0xc7\n" /* reg/opcode: /6 */
#define VMLAUNCH_OPCODE ".byte 0x0f,0x01,0xc2\n"
#define VMPTRLD_OPCODE ".byte 0x0f,0xc7\n" /* reg/opcode: /6 */
#define VMPTRST_OPCODE ".byte 0x0f,0xc7\n" /* reg/opcode: /7 */
#define VMREAD_OPCODE ".byte 0x0f,0x78\n"
#define VMRESUME_OPCODE ".byte 0x0f,0x01,0xc3\n"
#define VMWRITE_OPCODE ".byte 0x0f,0x79\n"
#define INVEPT_OPCODE ".byte 0x66,0x0f,0x38,0x80\n" /* m128,r64/32 */
#define INVVPID_OPCODE ".byte 0x66,0x0f,0x38,0x81\n" /* m128,r64/32 */
#define VMXOFF_OPCODE ".byte 0x0f,0x01,0xc4\n"
#define VMXON_OPCODE ".byte 0xf3,0x0f,0xc7\n"
```

驱动初在始化一个VMCS (Virtual Machine Control Structures)内存区域后,启动VMM (Virtual Machine Monitor) ,
得到最高权限从而管理硬件资源, 操作系统是运行于ring0下
VMM要监控管理整个系统的资源,因而VMM的权限是大于操作系统,它处于一个全新的级别ring -1
用一个形象的比喻来讲就是:当前操作系统的的“政权”被VMM颠覆了
下面这个图是: VMM 与Guest OS 的联系

下面简单介绍一下VMM的建立过程,看看“政权”是如何被颠覆的
首先通过CPUID检查CPU是否支持VT

硬件环境的检测,通过指令CPUID检测CPU是否支持VT
ECX的第5个bit标志代表对 VT 的支持与否
(更多检测可以看
http://bbs.pediy.com/showthread.php?t=96122

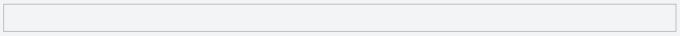
回贴部分)
初始化主要的内存区域
使用VMXON 进入 VMX (虚拟机指令扩展指令集)操作

执行VMXON指令 , EFLAGS.CF 可判断执行是否成功

```
__asm
{
PUSH 0
PUSH PhysicalVMXONRegionPtr.LowPart

_emit 0xF3 // VMXON [ESP]
_emit 0x0F
_emit 0xC7
_emit 0x34
_emit 0x24

PUSHFD
POP eFlags

ADD ESP, 8
}
```

接下来开始初始化VMCS区域,(相当于部署军队,准备起义)
这个结构只能够被VMCLEAR, VMPTRLD, VMREAD,和VMWRITE操作。
VMM运行后guest 执行这些指令时会引发#VMExit事件

VMCS 结构如下图



VMCS 是一个4K的内存区域
在逻辑上，虚拟机控制结构被划分为 6 部分：
1) GUEST-STATE 域：虚拟机从根操作模式进入非根操作模式时，处理器所处的状态；
2) HOST-STATE 域：虚拟机从非根操作模式退出到根操作模式时，处理器所处的状态；
3) VM 执行控制域：虚拟机在非根操作模式运行的时候，控制处理器非根操作模式退出到根操作模式；
4) VM 退出控制域：虚拟机从非根操作模式下退出时，需要保存的信息；
5) VM 进入控制域：虚拟机从根操作模式进入非根操作模式时，需要读取的信息；
6) VM 退出信息域：虚拟机从非根操作模式退出到根操作模式时，将退出的原因保存到该域中。

在这里初始化的代码比较多,要初始化 host 与 guest 的 CR0，CR3，CR4，IDTR，GDTR，LDTR，Rflag，SYSENTER_CS，SYSENTER_E
初始化时使用VMWRITE 指令设置 , 设置时主要的宏是

```
GUEST_ES_SELECTOR = 0x00000800,
GUEST_CS_SELECTOR = 0x00000802,
GUEST_SS_SELECTOR = 0x00000804,
GUEST_DS_SELECTOR = 0x00000806,
GUEST_FS_SELECTOR = 0x00000808,
GUEST_GS_SELECTOR = 0x0000080a,
GUEST_LDTR_SELECTOR = 0x0000080c,
GUEST_TR_SELECTOR = 0x0000080e,
HOST_ES_SELECTOR = 0x00000c00,
HOST_CS_SELECTOR = 0x00000c02,
HOST_SS_SELECTOR = 0x00000c04,
HOST_DS_SELECTOR = 0x00000c06,
HOST_FS_SELECTOR = 0x00000c08,
HOST_GS_SELECTOR = 0x00000c0a,
HOST_TR_SELECTOR = 0x00000c0c,
…
```

使用 VMWRITE 指令集成的函数

```
VOID WriteVMCS( ULONG encoding, ULONG value )
{
__asm
{
PUSHAD
PUSH value
MOV EAX, encoding

_emit 0x0F
_emit 0x79
_emit 0x04
_emit 0x24

POP EAX
POPAD
}
}
```

例如初始化 GDT , IDT 使用的是以下代码

```
__asm
{
SGDT gdt_reg
}
temp32 = 0;
temp32 = gdt_reg.BaseHi;
temp32 <<= 16;
temp32 |= gdt_reg.BaseLo;
Log( "Setting Host GDTR Base" , temp32 );
WriteVMCS( HOST_GDTR_BASE, temp32 );



__asm
{
SIDT idt_reg
}
temp32 = 0;
temp32 = idt_reg.BaseHi;
temp32 <<= 16;
temp32 |= idt_reg.BaseLo;

Log( "Setting Host IDTR Base" , temp32 );
WriteVMCS( HOST_IDTR_BASE, temp32 );
```

比较重要的是设置 #VMExit 事件处理入口 HOST_RIP

```
WriteVMCS( HOST_RIP, (ULONG)VMMEntryPoint ); //0x6C16
```

最后执行 VMLAUNCH 指令 ,正式启动虚拟机(建立了新的政权)

```
__asm
{
_emit 0x0F
_emit 0x01
_emit 0xC2
}
```

至此,整个VMM的帝国已经运行起来,帝国将会管理整个系统资源
接下来的就是#VMExit的事件循环处理(类似在调试器中下断点后,等待事件发生)
运行过程中Guest OS 遇到要监控的指令,会发生 #VMExit 事件 (相当于被调试程序执行到中断事件)
此时由VMM 处理,例如cpuid ,
处理完后由 VMRESUME 继续执行 (相当于在 OD 中 F9 继续运行)

#VMExit事件的类型

#VMExit事件的类型

#VMExit事件分为二种
1 无条件事件CPUID GETSEC INVD XSETBV 所有 VMX指令
2 有条件事件I/O访问,中断事件, MSR寄存器访问, HTL 等 (要设置VMCS相应部分触发)

VMM使用VMREAD读取虚拟机状态,主要的一些变量有
VM_EXIT_REASON = 0x00004402, //退出代码
VM_EXIT_INTR_INFO = 0x00004404, //中断信息
VM_EXIT_INTR_ERROR_CODE = 0x00004406,
IDT_VECTORING_INFO = 0x00004408,
IDT_VECTORING_ERROR_CODE = 0x0000440a,
VM_EXIT_INSTRUCTION_LEN = 0x0000440c, //指令长度
VMX_INSTRUCTION_INFO = 0x0000440e,
GUEST_ES_LIMIT = 0x00004800,
GUEST_CS_LIMIT = 0x00004802,
GUEST_SS_LIMIT = 0x00004804,
GUEST_DS_LIMIT = 0x00004806,
…
其中最重要的就是VM_EXIT_REASON , 可以看作是消息类型
主要的类型有
```
#define EXIT_REASON_EXCEPTION_NMI 0
#define EXIT_REASON_EXTERNAL_INTERRUPT 1
#define EXIT_REASON_TRIPLE_FAULT 2
#define EXIT_REASON_INIT 3
#define EXIT_REASON_SIPI 4
#define EXIT_REASON_IO_SMI 5
#define EXIT_REASON_OTHER_SMI 6
#define EXIT_REASON_PENDING_VIRT_INTR 7
#define EXIT_REASON_PENDING_VIRT_NMI 8
#define EXIT_REASON_TASK_SWITCH 9
#define EXIT_REASON_CPUID 10
#define EXIT_REASON_HLT 12
#define EXIT_REASON_INVD 13
#define EXIT_REASON_INVLPG 14
#define EXIT_REASON_RDPMC 15
#define EXIT_REASON_RDTSC 16
#define EXIT_REASON_RSM 17
#define EXIT_REASON_VMCALL 18
#define EXIT_REASON_VMCLEAR 19
#define EXIT_REASON_VMLAUNCH 20
#define EXIT_REASON_VMPTRLD 21
#define EXIT_REASON_VMPTRST 22
#define EXIT_REASON_VMREAD 23
#define EXIT_REASON_VMRESUME 24
#define EXIT_REASON_VMWRITE 25
#define EXIT_REASON_VMXOFF 26
#define EXIT_REASON_VMXON 27
…

DWORD VmxRead(DWORD in_code)
{
DWORD m_vmread = 0;
__asm
{
PUSHAD

MOV EAX, in_code
```

```
_emit 0x0F // VMREAD EBX, EAX
_emit 0x78
_emit 0xC3

MOV m_vmread, EBX
POPAD
}
return m_vmread;
}

VOID VMMReadGuestState( )
{
HandlerLogging = 0;
ExitReason = VmxRead(VM_EXIT_REASON); //0x4402
ExitInterruptionInformation = VmxRead( VM_EXIT_INTR_INFO); //0x4404
ExitInstructionLength = VmxRead(VM_EXIT_INSTRUCTION_LEN); //0x440c
ExitQualification = VmxRead(EXIT_QUALIFICATION) ; //0x6400
ExitInterruptionInformation = VmxRead(VM_EXIT_INTR_INFO); //0x4404
ExitInterruptionErrorCode = VmxRead(VM_EXIT_INTR_ERROR_CODE); //0x4406

IDTVectoringInformationField = VmxRead(IDT_VECTORING_INFO); //0X00004408 // IDT-Vectoring Information Field
IDTVectoringErrorCode = VmxRead(IDT_VECTORING_ERROR_CODE); //0X0000440A // IDT-Vectoring Error Code
ExitInstructionLength = VmxRead(VM_EXIT_INSTRUCTION_LEN); //0x0000440C // VM-Exit Instruction Length
ExitInstructionInformation = VmxRead(VMX_INSTRUCTION_INFO) ; //0x0000440E //VM-Exit Instruction Information
GuestEIP = VmxRead(GUEST_RIP); //0x0000681E; //GuestEIP
GuestESP = VmxRead(GUEST_RSP); //0x0000681c //esp
GuestCR3 = VmxRead(GUEST_CR3); //0X6802 GuestCR3

}
```

CPUID指令的拦截与修改

当 guest OS 执行 cupid 时,会得到这样一个错误号 10 (VMX_EXIT_CPUID)
当执行到CPUID这条指令的时候,响应,然后就可以修改cpuid的返回值

```
if( ExitReason == VMX_EXIT_CPUID )
{
if( GuestEAX == 0 )
{
DbgPrint("CPUID EIP == %08X \n" , GuestEIP );
//0x34EC2B
__asm
{
POPAD
MOV EAX, 0
CPUID


//修改CPUID返回值
MOV EBX, 0x80808080
MOV ECX, 0x90909090
MOV EDX, 0x10101010
JMP Resume
}
}
else
{
__asm
{
```

{
POPAD
MOV EAX, GuestEAX
CPUID
JMP Resume
}
}
}

最后运行的效果

运行VT前 cupid

运行VT后 cpuid

顺道打印了执行CPUID程序的 EIP,与输出的 00401009一致

```
00401003 |. 53 push ebx
00401004 |? B8 00000000 mov eax,0
00401009 |. 0FA2 cpuid
0040100B |. 894424 0C mov dword ptr ss:[esp+C],eax
0040100F |? 894C24 08 mov dword ptr ss:[esp+8],ecx
00401013 |? 895C24 04 mov dword ptr ss:[esp+4],ebx
00401017 |? 68 10614000 push cpuid.00406110 ; ASCII "---------cpuid------",LF
```

上传的附件 bin.rar (20.0 KB, 365 次下载)
intel-vt.rar (33.4 KB, 560 次下载)

此帖于 2011-12-24 18:45:21 被 网监张局 最后编辑 原因: 格式

[公告]请注意言行举止，不要让大家觉得不适!

网监张局
查看公开信息
查找 网监张局 发表的帖子
查找 网监张局 发表的所有主题
查看 网监张局 发表的精华帖

网监张局

普通会员

资 料:
注册日期: Oct 2011
帖子: 38
精华: 1
现金: 308 Kx
致谢数: 0
获感谢文章数：1

获会员感谢数：2   2 2011-12-24, 18:39:09

EXIT_REASON 的所有类型注释

VirtualBox\src\VBox\VMM\VMMR3\HWACCM.cpp

```
EXIT_REASON(VMX_EXIT_EXCEPTION          ,  0, "Exception or non-maskable interrupt (NMI)."),
    EXIT_REASON(VMX_EXIT_EXTERNAL_IRQ       ,  1, "External interrupt."),
    EXIT_REASON(VMX_EXIT_TRIPLE_FAULT       ,  2, "Triple fault."),
    EXIT_REASON(VMX_EXIT_INIT_SIGNAL        ,  3, "INIT signal."),
    EXIT_REASON(VMX_EXIT_SIPI               ,  4, "Start-up IPI (SIPI)."),
    EXIT_REASON(VMX_EXIT_IO_SMI_IRQ         ,  5, "I/O system-management interrupt (SMI)."),
    EXIT_REASON(VMX_EXIT_SMI_IRQ            ,  6, "Other SMI."),
    EXIT_REASON(VMX_EXIT_IRQ_WINDOW         ,  7, "Interrupt window."),
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_TASK_SWITCH        ,  9, "Task switch."),
    EXIT_REASON(VMX_EXIT_CPUID              , 10, "Guest software attempted to execute CPUID."),
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_HLT                , 12, "Guest software attempted to execute HLT."),
    EXIT_REASON(VMX_EXIT_INVD               , 13, "Guest software attempted to execute INVD."),
    EXIT_REASON(VMX_EXIT_INVPG              , 14, "Guest software attempted to execute INVPG."),
    EXIT_REASON(VMX_EXIT_RDPMC              , 15, "Guest software attempted to execute RDPMC."),
    EXIT_REASON(VMX_EXIT_RDTSC              , 16, "Guest software attempted to execute RDTSC."),
    EXIT_REASON(VMX_EXIT_RSM                , 17, "Guest software attempted to execute RSM in SMM."),
    EXIT_REASON(VMX_EXIT_VMCALL             , 18, "Guest software executed VMCALL."),
    EXIT_REASON(VMX_EXIT_VMCLEAR            , 19, "Guest software executed VMCLEAR."),
    EXIT_REASON(VMX_EXIT_VMLAUNCH           , 20, "Guest software executed VMLAUNCH."),
    EXIT_REASON(VMX_EXIT_VMPTRLD            , 21, "Guest software executed VMPTRLD."),
    EXIT_REASON(VMX_EXIT_VMPTRST            , 22, "Guest software executed VMPTRST."),
    EXIT_REASON(VMX_EXIT_VMREAD             , 23, "Guest software executed VMREAD."),
    EXIT_REASON(VMX_EXIT_VMRESUME           , 24, "Guest software executed VMRESUME."),
    EXIT_REASON(VMX_EXIT_VMWRITE            , 25, "Guest software executed VMWRITE."),
    EXIT_REASON(VMX_EXIT_VMXOFF             , 26, "Guest software executed VMXOFF."),
    EXIT_REASON(VMX_EXIT_VMXON              , 27, "Guest software executed VMXON."),
    EXIT_REASON(VMX_EXIT_CRX_MOVE           , 28, "Control-register accesses."),
    EXIT_REASON(VMX_EXIT_DRX_MOVE           , 29, "Debug-register accesses."),
    EXIT_REASON(VMX_EXIT_PORT_IO            , 30, "I/O instruction."),
    EXIT_REASON(VMX_EXIT_RDMSR              , 31, "RDMSR. Guest software attempted to execute RDMSR."),
    EXIT_REASON(VMX_EXIT_WRMSR              , 32, "WRMSR. Guest software attempted to execute WRMSR."),
    EXIT_REASON(VMX_EXIT_ERR_INVALID_GUEST_STATE,  33, "VM-entry failure due to invalid guest state."),
    EXIT_REASON(VMX_EXIT_ERR_MSR_LOAD       , 34, "VM-entry failure due to MSR loading."),
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_MWAIT              , 36, "Guest software executed MWAIT."),
    EXIT_REASON_NIL(),
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_MONITOR            , 39, "Guest software attempted to execute MONITOR."),
    EXIT_REASON(VMX_EXIT_PAUSE              , 40, "Guest software attempted to execute PAUSE."),
    EXIT_REASON(VMX_EXIT_ERR_MACHINE_CHECK  , 41, "VM-entry failure due to machine-check."),
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_TPR                , 43, "TPR below threshold. Guest software executed MOV to CR8.
    EXIT_REASON(VMX_EXIT_APIC_ACCESS        , 44, "APIC access. Guest software attempted to access memory a
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_XDTR_ACCESS        , 46, "Access to GDTR or IDTR. Guest software attempted to exec
    EXIT_REASON(VMX_EXIT_TR_ACCESS          , 47, "Access to LDTR or TR. Guest software attempted to execut
    EXIT_REASON(VMX_EXIT_EPT_VIOLATION      , 48, "EPT violation. An attempt to access memory with a guest-
    EXIT_REASON(VMX_EXIT_EPT_MISCONFIG      , 49, "EPT misconfiguration. An attempt to access memory with a
    EXIT_REASON(VMX_EXIT_INVEPT             , 50, "INVEPT. Guest software attempted to execute INVEPT."),
    EXIT_REASON_NIL()
```

```
    EXIT_REASON_NIL(),
    EXIT_REASON(VMX_EXIT_PREEMPTION_TIMER   , 52, "VMX-preemption timer expired. The preemption timer count
    EXIT_REASON(VMX_EXIT_INVVPID            , 53, "INVVPID. Guest software attempted to execute INVVPID."),
    EXIT_REASON(VMX_EXIT_WBINVD             , 54, "WBINVD. Guest software attempted to execute WBINVD."),
    EXIT_REASON(VMX_EXIT_XSETBV             , 55, "XSETBV. Guest software attempted to execute XSETBV."),
    EXIT_REASON_NIL()
```

此帖于 2011-12-24 18:45:47 被 网监张局 最后编辑

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮！

共 2 位会员
感谢 网监张局 发表的文章： salwtp (2012-02-14), 小小天 (2011-12-25)

网监张局
查看公开信息
查找 网监张局 发表的帖子
查找 网监张局 发表的所有主题
查看 网监张局 发表的精华帖

pDriObj

初级会员

资 料:
注册日期: Oct 2009
帖子: 27
精华: 0
现金: 93 Kx
致谢数: 0
获感谢文章数：0
获会员感谢数：0 3 2011-12-24, 19:00:48

这个必须得顶!!!

[招生]15PB开始接受第002期报名!

pDriObj
查看公开信息
查找 pDriObj 发表的帖子
查找 pDriObj 发表的所有主题

雪yaojun

初级会员

资　料:
注册日期: Dec 2007
帖子: 538
精华: 0
现金: 145 Kx
致谢数: 11
获感谢文章数：7
获会员感谢数：7　4　2011-12-24, 19:04:36

这是神马？？？？

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮！

雪yaojun
查看公开信息
查找 雪yaojun 发表的帖子
查找 雪yaojun 发表的所有主题

Fido

初级会员

资　料:
注册日期: Jan 2008
帖子: 829
精华: 0
现金: 354 Kx
致谢数: 19
获感谢文章数：6
获会员感谢数：6　5　2011-12-24, 19:08:45

强帖留名啊...我的天啊..膜拜啊...

[公告]请注意言行举止，不要让大家觉得不适！

Fido
查看公开信息
查找 Fido 发表的帖子
查找 Fido 发表的所有主题

查找 Fido 发表的所有主题

xiejienet

初级会员

资　料:
注册日期: Jun 2007
帖子: 490
精华: 0
现金: 391 Kx
致谢数: 6
获感谢文章数：3
获会员感谢数：3　6　2011-12-24, 19:12:47

虽然看不懂,但是觉得好牛逼

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮！

xiejienet
查看公开信息
查找 xiejienet 发表的帖子
查找 xiejienet 发表的所有主题

雪yaojun

初级会员

资　料:
注册日期: Dec 2007
帖子: 538
精华: 0
现金: 145 Kx
致谢数: 11
获感谢文章数：7
获会员感谢数：7　7　2011-12-24, 20:11:14

---

cpuid是拦截个什么过程呀？？楼主可不可以详细说明一下。
要是代码加个缩进就好了。

[公告]请注意言行举止，不要让大家觉得不适！

雪yaojun
查看公开信息
查找 雪yaojun 发表的帖子
查找 雪yaojun 发表的所有主题

panti

初级会员

资　料:
注册日期: Jun 2008
帖子: 212
精华: 0
现金: 193 Kx
致谢数: 5
获感谢文章数：0
获会员感谢数：0　8　2011-12-25, 04:28:20

是好东西，不知道怎么看
后面的可以拦截CPU指令是什么意思？在虚拟机中拦截？

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮！

panti
查看公开信息
查找 panti 发表的帖子
查找 panti 发表的所有主题

网监张局

普通会员

资　料:
注册日期: Oct 2011
帖子: 38
精华: 1
现金: 308 Kx
致谢数: 0
获感谢文章数：1
获会员感谢数：2　9　2011-12-25, 17:23:27

引用:
最初由 panti发布
是好东西，不知道怎么看
后面的可以拦截CPU指令是什么意思？在虚拟机中拦截？
后面是VMM已经启动，VMM 监控操作系统的运行
而CPUID这个指令是会主动触发#VMExit的,不用配置VMCS

1 无条件事件CPUID GETSEC INVD XSETBV 所有 VMX指令

如果事件ID == VMX_EXIT_CPUID 就代表执行的是 CPUID

[公告]请注意言行举止，不要让大家觉得不适！

网监张局
查看公开信息
查找 网监张局 发表的帖子
查找 网监张局 发表的所有主题
查看 网监张局 发表的精华帖

游戏神通

初级会员

资　料:
注册日期: Jul 2010
帖子: 166
精华: 0
现金: 30 Kx
致谢数: 1
获感谢文章数：1
获会员感谢数：1　10　2011-12-25, 19:30:19

不错 顶　.

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮！

游戏神通
查看公开信息
访问 游戏神通 的个人网站
查找 游戏神通 发表的帖子
查找 游戏神通 发表的所有主题

nevergone

普通会员

资　料:
注册日期: Nov 2006
帖子: 246
精华: 3
现金: 39 Kx
致谢数: 0
获感谢文章数：0
获会员感谢数：0　11　2011-12-25, 22:48:20

学习。谢谢分享

[公告]请注意言行举止，不要让大家觉得不适!

nevergone
查看公开信息
查找 nevergone 发表的帖子
查找 nevergone 发表的所有主题
查看 nevergone 发表的精华帖

wowocock

普通会员

资　料:
注册日期: Jul 2007
帖子: 103
精华: 1
现金: 255 Kx
致谢数: 0
获感谢文章数：0
获会员感谢数：0　12　2011-12-26, 14:13:30

虽然已经很老了，但还是不错。

[公告]如果你觉得有人语言挑衅，请点每帖右上角的“举报”按钮!

wowocock
查看公开信息

查找 wowocock 发表的帖子
查找 wowocock 发表的所有主题
查看 wowocock 发表的精华帖

wowocock

普通会员

资　料:
注册日期: Jul 2007
帖子: 103
精华: 1
现金: 255 Kx
致谢数: 0
获感谢文章数：0
获会员感谢数：0　13　2011-12-26, 14:16:39

引用:
最初由 panti发布
是好东西，不知道怎么看
后面的可以拦截CPU指令是什么意思？在虚拟机中拦截？
参考很久以前 ROOTKIT.COM上的例子，在VMEXIT里处理，修改GUEST各寄存器的返回值，来达到修改CPUID结果的效果。

VMM不是只能接管虚拟机环境中执行的指令而引发的#VMExit，你那个CPUID好象直接就是在R3真实机下的一个程序，这难道也能被VMM接管