POSITIVE TECHNOLOGIES

Very Mighty eXtension for debugging

Artem Shishkin

Debugging essentials

Debugging prerequisites

— Ability to pause program execution

- Any asynchronous event is suitable (exception or interrupt)

— Ability to examine program CPU context (registers state)

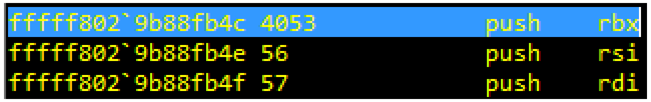
— Ability to examine program memory

- Memory is shared (so as any hardware)

Debugging capabilities

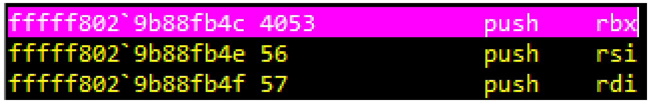
— INT 3 (#BP)

- 0xCC opcode
- Involves memory modification



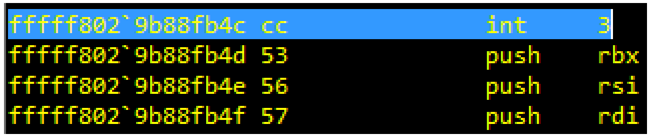
```
fffff802`9b88fb4c 4053            push    rbx
fffff802`9b88fb4e 56              push    rsi
fffff802`9b88fb4f 57              push    rdi
```

Original code



```
fffff802`9b88fb4c 4053            push    rbx
fffff802`9b88fb4e 56              push    rsi
fffff802`9b88fb4f 57              push    rdi
```

What you see in a debugger



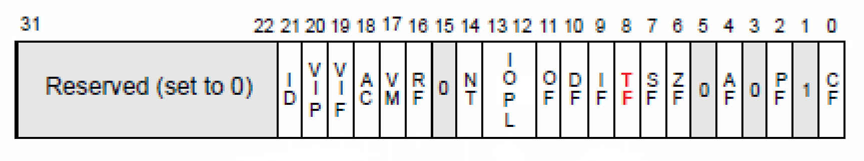
```
fffff802`9b88fb4c cc              int     3
fffff802`9b88fb4d 53              push    rbx
fffff802`9b88fb4e 56              push    rsi
fffff802`9b88fb4f 57              push    rdi
```

What is really happening

Debugging capabilities

— INT 1 (#DB)

- Single stepping
  - Through setting TF in eflags register



31 … 22	21	20	19	18	17	16	15	14	13 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved (set to 0)	ID	VIP	VIF	AC	VM	RF	0	NT	IOPL	OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

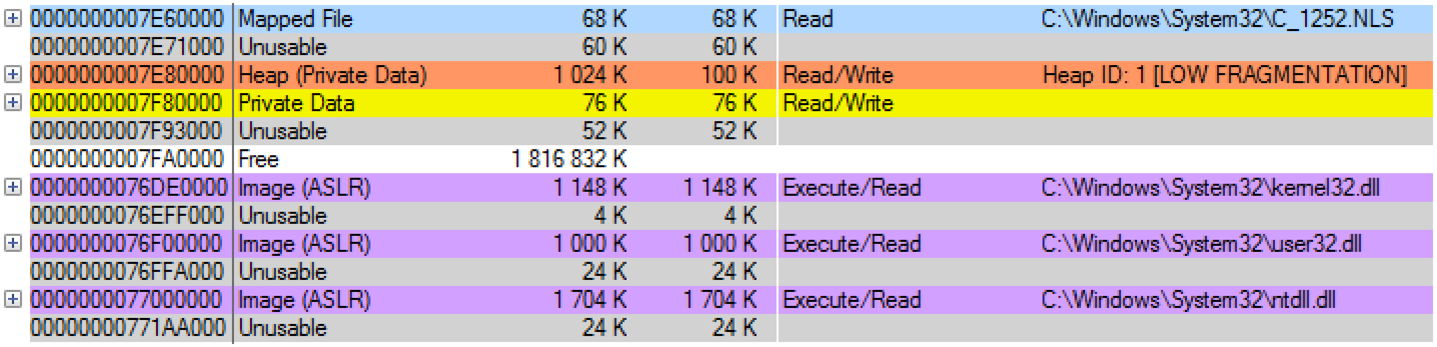
- Debug registers
  - Through modifying DR0-DR7 registers
  - Up to 4 linear address breakpoints (Reads, Writes, Executes)

- Involves register modification

Debugging capabilities

— INT 0x0E (#PF)

- Memory access trapping
- Trapping page access (Reads, Writes, Executes)
- Involves page table modification (Bits P, RW, XD)



⊞	000000007E60000	Mapped File	68 K	68 K	Read	C:\Windows\System32\C_1252.NLS
	000000007E71000	Unusable	60 K	60 K		
⊞	000000007E80000	Heap (Private Data)	1 024 K	100 K	Read/Write	Heap ID: 1 [LOW FRAGMENTATION]
⊞	000000007F80000	Private Data	76 K	76 K	Read/Write	
	000000007F93000	Unusable	52 K	52 K		
	000000007FA0000	Free	1 816 832 K			
⊞	0000000076DE0000	Image (ASLR)	1 148 K	1 148 K	Execute/Read	C:\Windows\System32\kernel32.dll
	0000000076EFF000	Unusable	4 K	4 K		
⊞	0000000076F00000	Image (ASLR)	1 000 K	1 000 K	Execute/Read	C:\Windows\System32\user32.dll
	0000000076FFA000	Unusable	24 K	24 K		
⊞	0000000077000000	Image (ASLR)	1 704 K	1 704 K	Execute/Read	C:\Windows\System32\ntdll.dll
	00000000771AA000	Unusable	24 K	24 K		

Anti-…-anti-debugging

OS debugging integration

— Modifies OS structures

  - PEB. BeingDebugged
  - nt!KdDebuggerEnabled

— Modifies control-flow

  - Event suppressing

— Exposes information about debugging session

  - ProcessDebugPort info class

— Refer to “The Ultimate Anti-Debugging Reference” by Peter Ferrie

Debugging impact

— Execution is paused, but time is not

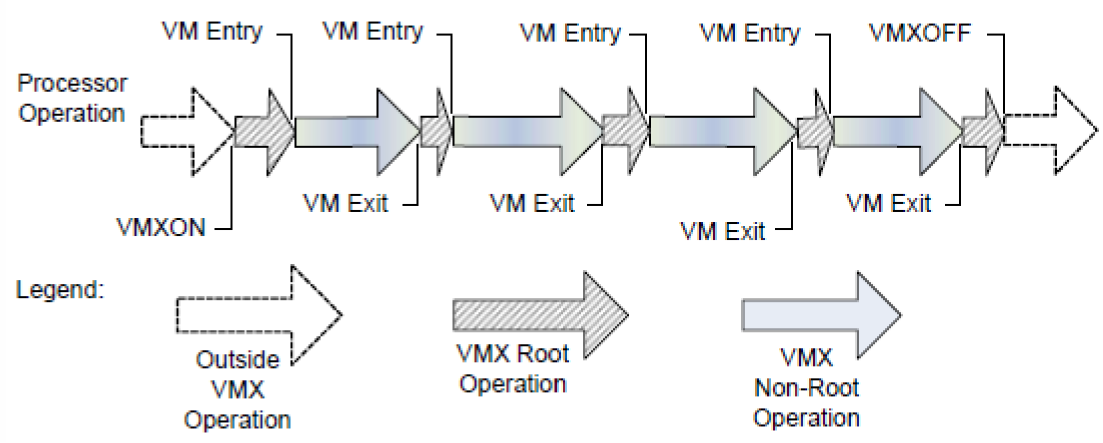
- GetTickCount
- rdtsc, rdtscp
- Performance monitoring
- OS specific (KdpTimeSlipDpc)

VMX basics

Virtual Machine Extensions

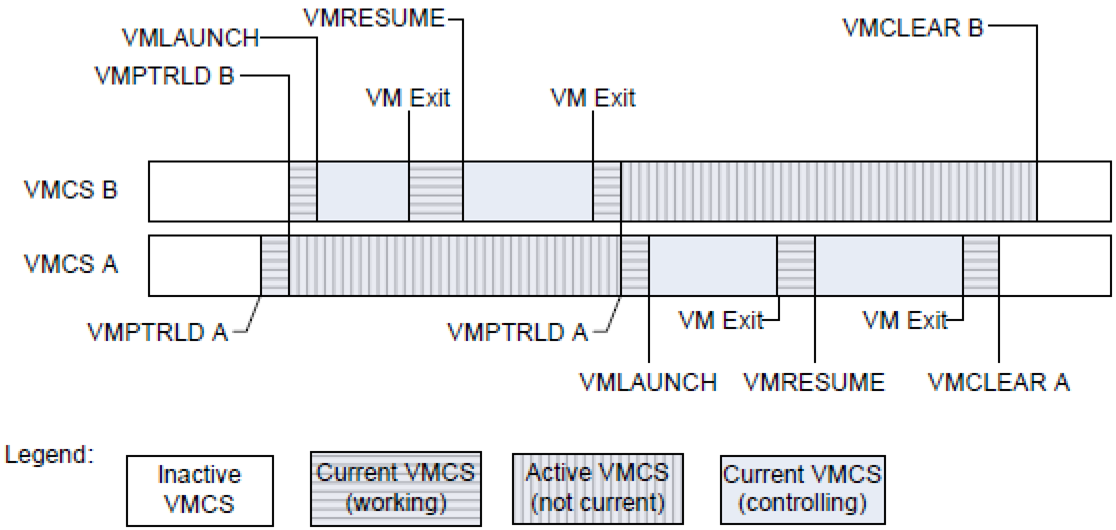
— Different processor execution mode

— Mode switching between Host (VMM) and Guest (OS)



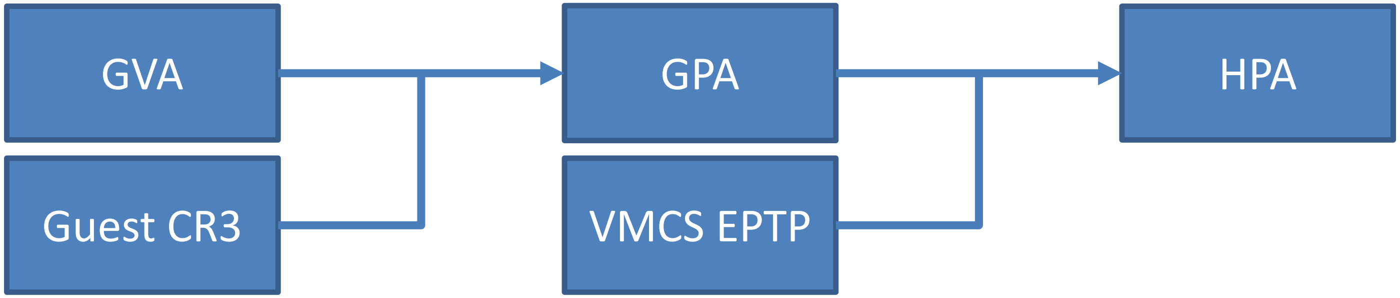
VMCS

- Virtual Machine Control Structure
  - Guest state
  - Host state
  - Virtual machine settings
  - Can be dynamically switched



EPT

- Second Level Address Translation (SLAT)
- Extended Page Table
  - Guest physical address to host physical address mappings
  - Page-level access control for guest physical addresses (reads, writes, executes)



VM Exits

— Events that cause guest mode switch to host mode

- Interrupts and exceptions
- EPT violations
- Certain instructions execution
- Special periodic timer ticks
- Instruction fetches under certain conditions
- System state related changes

and more…

Adapting VMX for debugging

VMX and debugger similarities

— Guest is paused when Host executes

— Full CPU context access

— Full memory access

Debugging events

— VM Exits can be treated like debugging events

VM Exit	Debugging event
Any VM Exit	Debugger break-in
Any VM Resume	Debugger continue
Monitor Trap Flag	Single-step event
VM Exit Instruction Execution	Breakpoint
EPT violation	Page fault

— A simple debugger needs nothing more

Outstanding capabilities

Additional events

- Address space switching
  - Used for switching between processes
- Special interrupts
  - Gives an ability to trace processor bootstrap code
- System structures modification
  - Used for debugging OS startup code
- Hardware access through IO ports and MMIO
  - Used for debugging hardware

Guest isolation benefits

— Stealth debugging

- Breakpoints hiding through EPT modification
- Hardware filtering through EPT modification, IO ports interception, VT-d, MSR access interception

— Time control

- Ability to conceal host execution time

— Blue-pilling

- Ability to convert your machine into virtual one on-the-fly at any time (well, at any time that you are able to gain execution control)

Full hardware access

— Full memory control
  - Disregarding address space
  - Disregarding privilege level

— Full context control

— Full MSR control

Virtual Machine Introspection

Analyzing the execution environment

— Perform in-place memory forensics

- Extended with CPU state

— Full hardware access provides full information about software

- Current module can be detected using module header
- Current kernel can be detected using CPU state
- Symbol information can be used to restore high-level OS data structures

Known issues

Virtualized memory is physical memory

— OS memory manager relies on virtual memory

- Memory pages can be not mapped (on-demand paging)
- Memory pages can be trimmed
- Memory pages can be moved
- Memory manager can interpret non-present pages however it wants

Virtual machine monitor robustness

- — VMX Guest operation is different from ordinary operation
  - VMM has to emulate a set of instructions
- — Stealthness is not free of charge
  - All detection vectors have to be inspected and tested with care
  - Some anti-detection tricks are highly difficult to implement
- — Host mode operation is also not free of charge
  - VMM has to be fast in order the Guest to operate smoothly

Implementation case

User interaction

— Debuggee is a remote machine

- Difficult to share the hardware between host and guest

— Communication is done via a set of transports

- Windows KD as an example

— Debugger is small and stupid

- Heavy analysis is performed by a debugging client

— Minimize data exchange

- Transport can be slow (like serial)
- Offload client features to the VMM if possible

Breakpoints

— Ordinary int 3

— Hide through EPT (allow execution only)

- Can be emulated on read or write
- Can be single-stepped on read or write

— Global

- Filter using CR3, VA and GPA

Debugging hints

— Maximize memory pages presence

- Disable swap
- DisablePagingExecutive (for Windows)
- Learn OS memory manager – absent pages can be mapped elsewhere

— Suppress interrupts

- Modify IF bit in eflags
- Modify guest interruptibility state

Questions?

- https://twitter.com/honorary_bot
- https://github.com/honorarybot

- https://github.com/ptresearch

- https://www.ptsecurity.com/products/#multiscanner

Thank you!

Artem Shishkin

ashishkin@ptsecurity.com

POSITIVE TECHNOLOGIES