SMAPWN

A FASTER WAY FOR DETECTING DOUBLE FETCH VULNERABILITIES IN WINDOWS KERNEL

by Artem Shishkin
@honorary_bot

DISCLAIMER

I am presenting the contents of this presentation in my personal capacity. The views expressed are solely my own and do not necessarily reflect the views of Intel Corporation or its affiliates.

WHOAMI

- Security researcher at Intel Corporation
- I like Windows kernel

DOUBLE FETCH VULNERABILITIES

- TOCTOU:

  pStruct->size checked

  … ← this is when we change it

  pStruct->size used

- Two sequential references to the same address in this case
- Can we track it?

WE CAN TRACK IT

- And the guys did that using Bochs
  - Bochspwn by Mateusz "j00ru" Jurczyk and Gynvael Coldwind
  - https://research.google.com/pubs/archive/42189.pdf
- Bochs means instrumenting the whole system
  - A bit slow
- BSDaemon and NadavCh did that using Simics
  - https://github.com/rrbranco/poc_gtfo/blob/master/pocorgtfo15.pdf
- Are there different ways of tracking?

SMAP

- Supervisor Mode Access Prevention
- When a code runs on Ring 0 (CPL = 0) and tries to access a user mode page (PTE.U\S bit = 1) a page fault is raised
- Configured by CR4.SMAP bit
- STAC \ CLAC instructions may be used to control SMAP operation
- Windows does not use SMAP … yet (RS5 will)

THE IDEA

- Use SMAP to track kernel references to usermode memory
- Gather the needed info
  - Code address that caused an exception
  - Address that the code tried to access
  - Timestamp
- Apply filtering if needed
  - Filter the log data
  - Apply in runtime to increase performance

IMPLEMENTATION #1

- A debugger script
  - Enable SMAP
  - Place a breakpoint on KiPageFault
  - Check if it is caused by SMAP violation
  - Log all necessary data
  - Disable SMAP
  - Perform a single-step
  - GOTO 1

IMPLEMENTATION #1 ADVANTAGE

- The fastest way to implement the idea
- Maybe even the easiest way as well

IMPLEMENTATION #1 DRAWBACKS

- Slow
  - Debug transport latency
  - WinDBG / KD latency
- Unstable
- Not thread safe
  - There is a chance for a context switch while CR4.SMAP is disabled

IMPLEMENTATION #2

- A debugger plugin
  - Enable SMAP
  - Implement a custom page fault handler
  - Implement a custom debug trap or fault handler
  - Patch system handlers
  - Go!

IMPLEMENTATION #2 ADVANTAGES

- Target OS operates way faster
- Since the target is being kernel debugged, the PatchGuard will not complain about kernel modifications

IMPLEMENTATION #2 DRAWBACKS

- Still not the fastest way of target execution
  - Because log go to WinDBG \ KD, which causes some latency
- Kernel debugger is required

IMPLEMENTATION #3

- Autonomous kernel mode driver
  - Implements the same stuff as #2

IMPLEMENTATION #3 ADVANTAGES

- Maximum target OS execution speed
  - Since logging is local
- No need for kernel debugging

IMPLEMENTATION #3 DRAWBACKS

- Still you have to tame the PatchGuard

IMPLEMENTING #2

- Implement custom exception handlers as an injectable asm code
- Locate a space in Windows kernel to place the asm code
- Load the asm code
- Patch the OS IDT entries

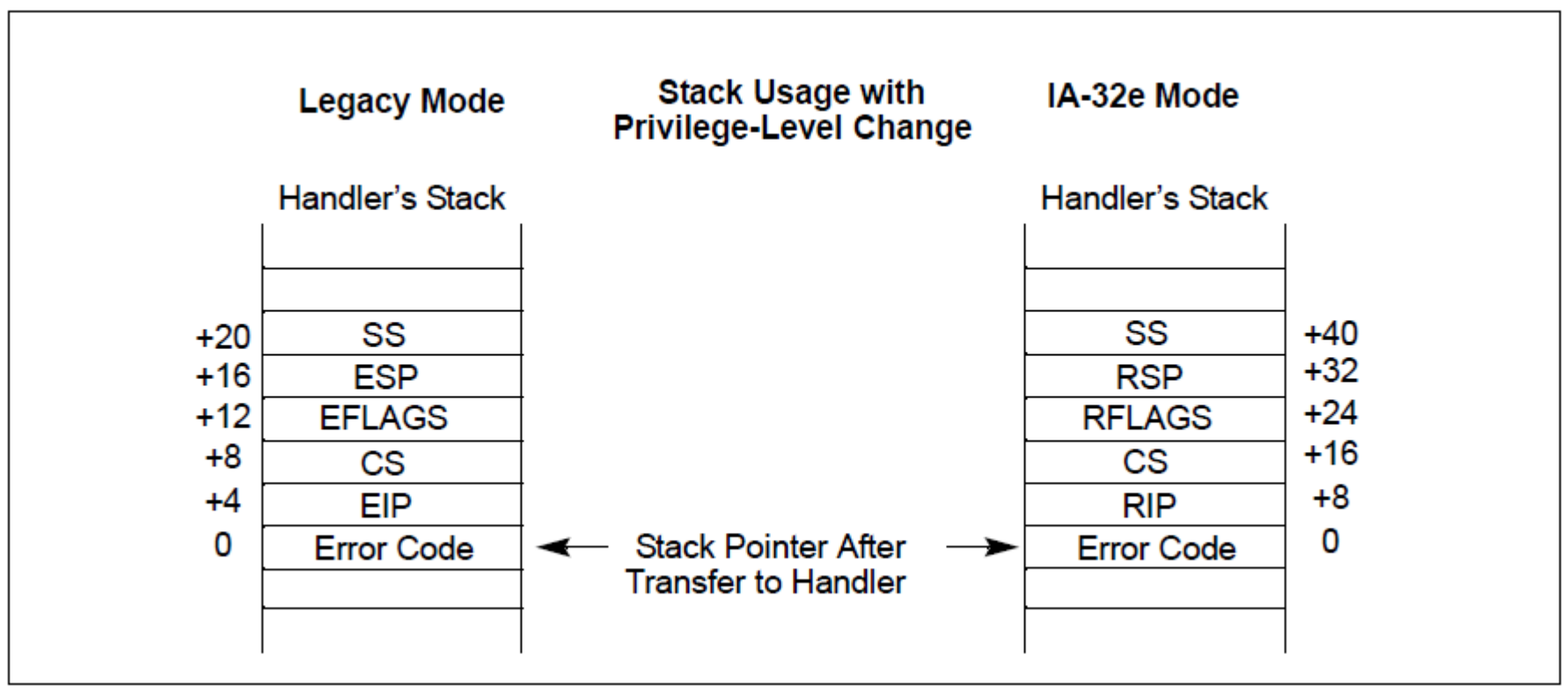
CUSTOM PAGE FAULT HANDLER

- Check whether the page fault is SMAP related
  - Check the error code of #PF
  - Check if saved RIP is a kernel address (higher part of virtual address space)
  - Check if CR2 (linear page fault address) is a user address
- Disable SMAP by setting RFLAGS.AC flag
- Log the needed info
  - Saved RIP is a code address that caused an exception
  - CR2 is an address the kernel module tried to access
  - RDTSC as a timestamp
  - Anything else, symbol info may be used to resolve any kernel info
- Format the log buffer and call nt!DbgPrintEx to send the data to the debugger
- Set saved RFLAGS.TF to 1, that will cause a single-step exception on thread resume
- IRETQ or jump to the original Page Fault Handler

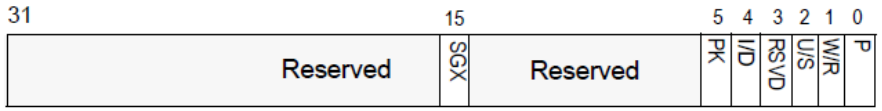
CUSTOM DEBUG TRAP OR FAULT HANDLER

- Check if a SMAP logging event resume (ID flag of RFLAGS)
  - Otherwise it is just a regular single step in the debugger
- Enable SMAP by resetting RFLAGS.AC flag
- Reset TF for saved RFLAGS
- IRETQ

STACK LAYOUT AFTER EXCEPTION

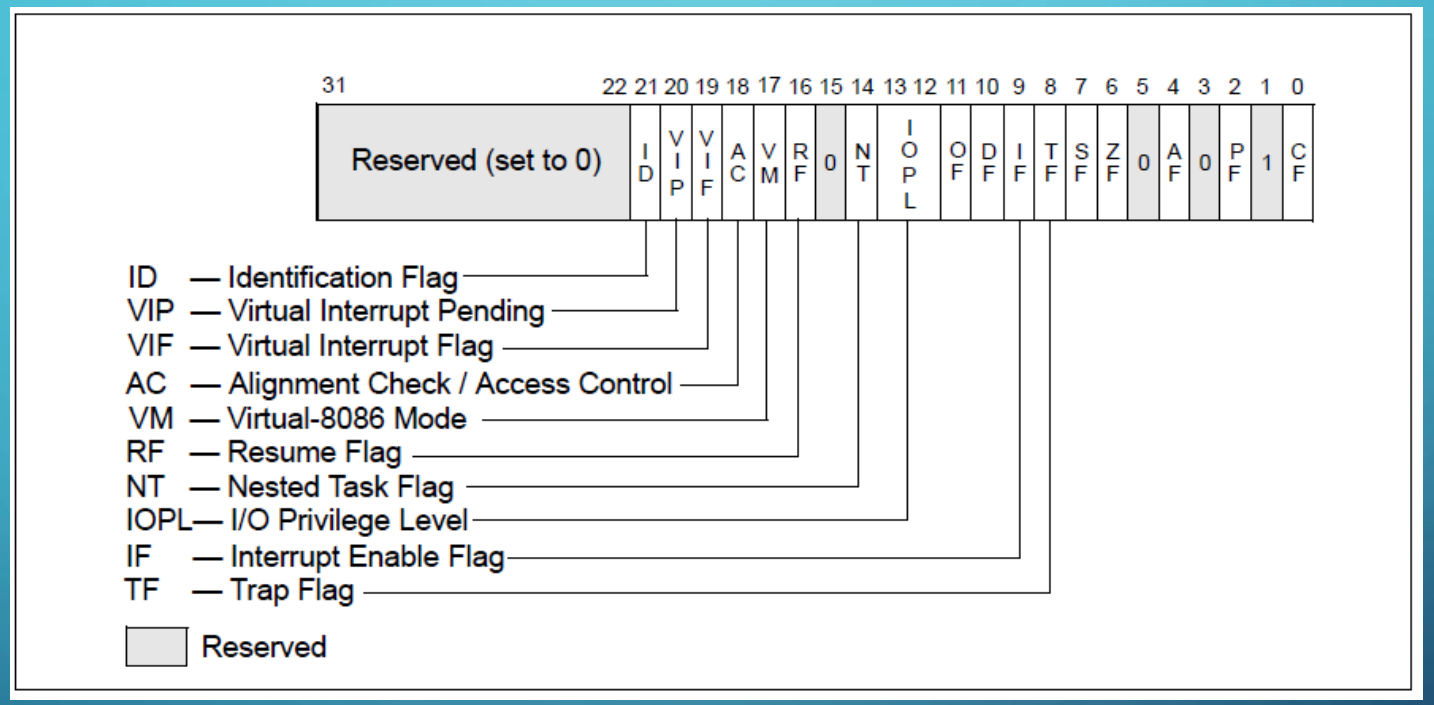


PAGE FAULT ERROR CODE



P	0	The fault was caused by a non-present page.
	1	The fault was caused by a page-level protection violation.
W/R	0	The access causing the fault was a read.
	1	The access causing the fault was a write.
U/S	0	A supervisor-mode access caused the fault.
	1	A user-mode access caused the fault.
RSVD	0	The fault was not caused by reserved bit violation.
	1	The fault was caused by a reserved bit set to 1 in some paging-structure entry.
I/D	0	The fault was not caused by an instruction fetch.
	1	The fault was caused by an instruction fetch.
PK	0	The fault was not caused by protection keys.
	1	There was a protection-key violation.
SGX	0	The fault is not related to SGX.
	1	The fault resulted from violation of SGX-specific access-control requirements.

RFLAGS REGISTER



- ID flag can be used as a spare bit

LOCATING A PLACE FOR PLUGIN DATA

- KUSER_SHARED_PAGE
  - Has plenty of space
  - Located at fixed address
  - Will not be reused
- Patch kernel PTE to make it executable
- Load your code and data there

WINDBG PLUGIN

- Automatic mode
  - Just logs everything
- Manual mode
  - Executes until next SMAP violation
- Target module filtering
  - Can target against a specific kernel module in runtime

IT'S WORKING

- Sample output

```
0: kd> g
000001FF50C597F0 FFFFF8031D96E61E 00000C81811FBDC9 0000000000000003
000001FF50C597F8 FFFFF8031D96E622 00000C818329484B 0000000000000003
000001FF50C59800 FFFFF8031D96E62C 00000C818533F1A1 0000000000000003
000000AB9E9FF818 FFFFF8031D96DD2D 00000C81873D895F 0000000000000003
000000AB9E9FF688 FFFFF8031DA50A10 00000C8189464D71 0000000000000001
000000AB9E9FF680 FFFFF8031DA50A18 00000C818B4C8451 0000000000000001
000000AB9E9FF678 FFFFF8031DA50A20 00000C818D60A849 0000000000000001
000000AB9E9FF670 FFFFF8031DA50A28 00000C818F727ECB 0000000000000001
```

  - Referenced address
  - Code that referenced
  - TSC
  - Error code
- The results are to be post-processed

THANK YOU!

- And a huge thanks to my team “STORM” at Intel for support and participation
- And the guys who already used the approach

QUESTIONS?