

Persistent, Stealthy, Remote-controlled Dedicated Hardware Malware

Patrick Stewin and Iurii Bystrov

Security in Telecommunications (SecT)

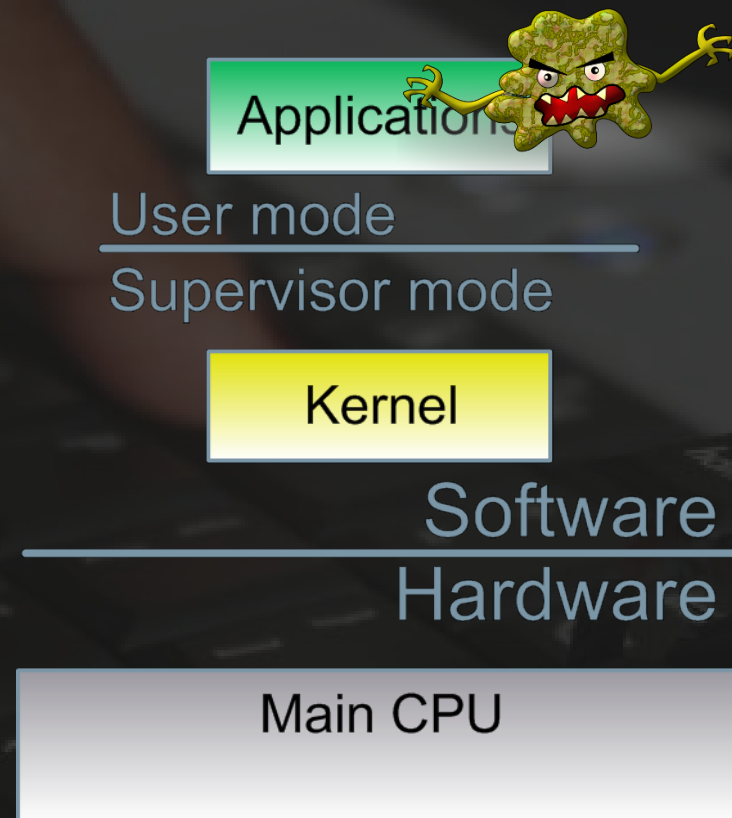
TU Berlin

patrickx@sec.t-labs.tu-berlin.de

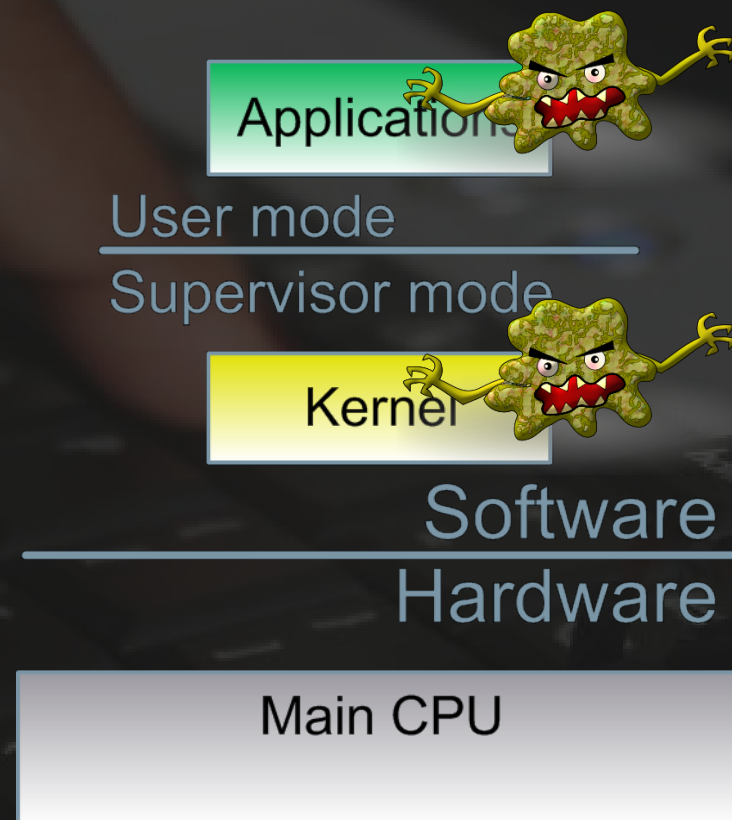
30C3, Hamburg, Germany



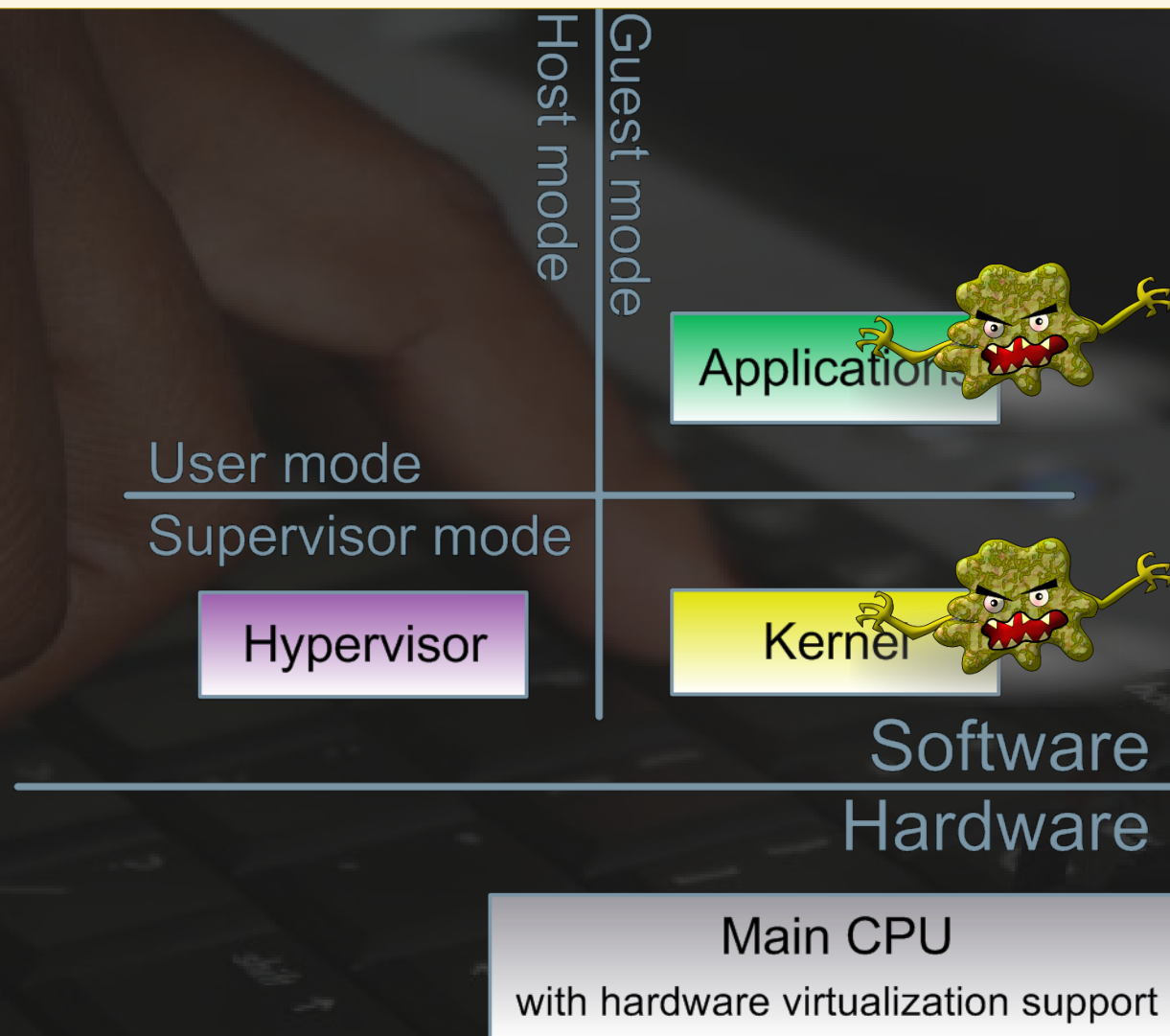
ARMS RACE Malware Developers ↔ Anti-Malware Community



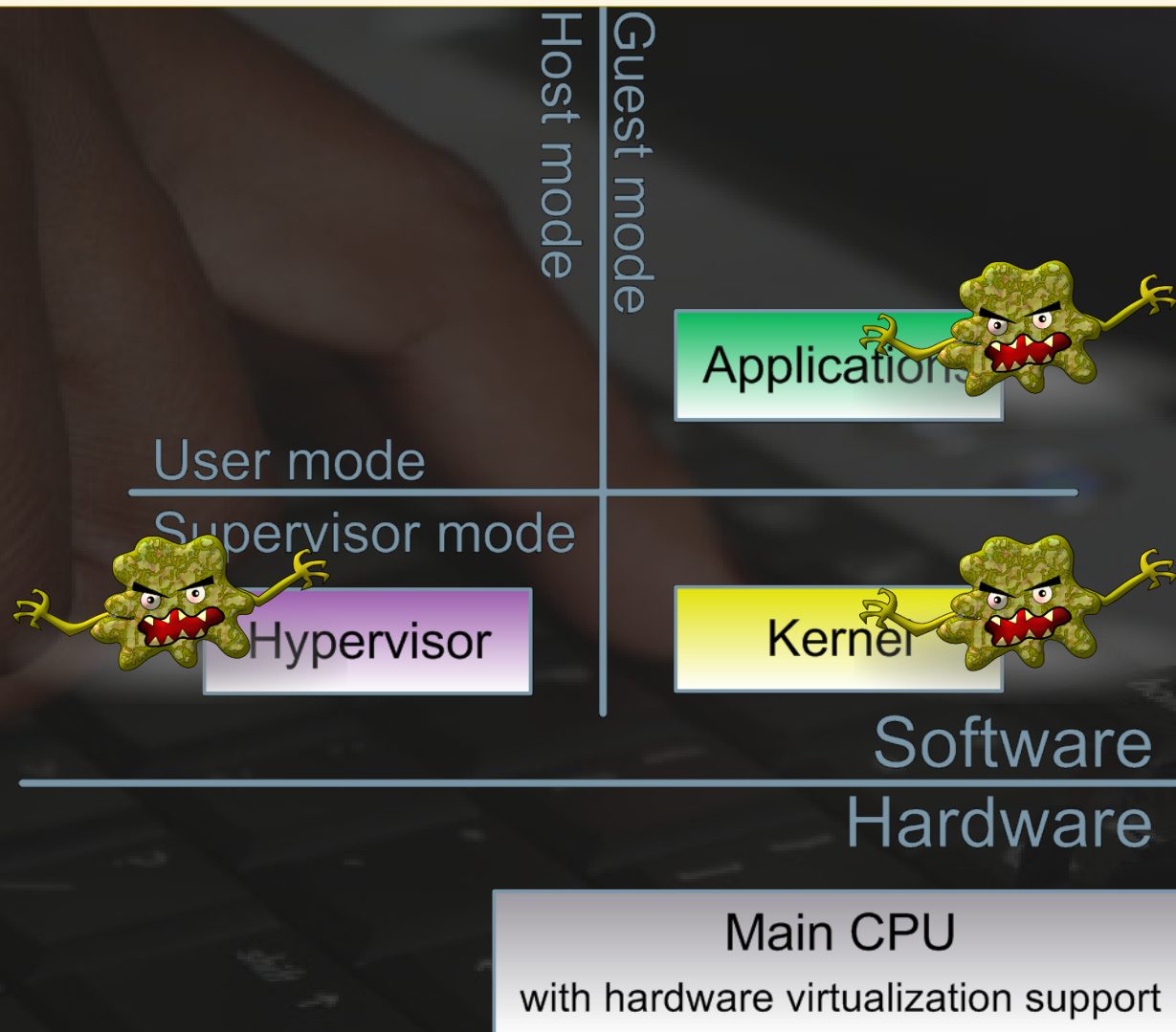
ARMS RACE Malware Developers ↔ Anti-Malware Community



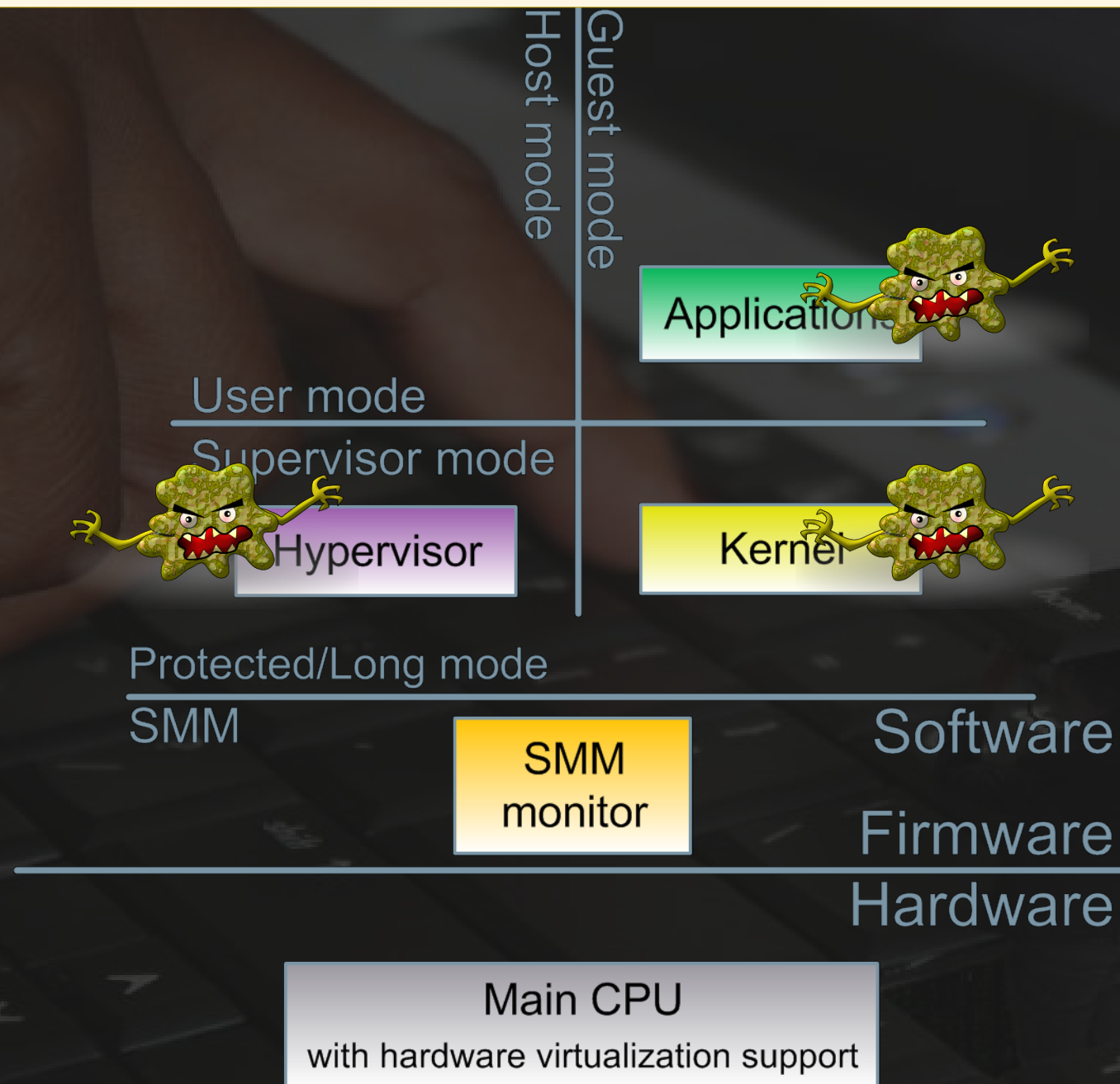
ARMS RACE Malware Developers ↔ Anti-Malware Community



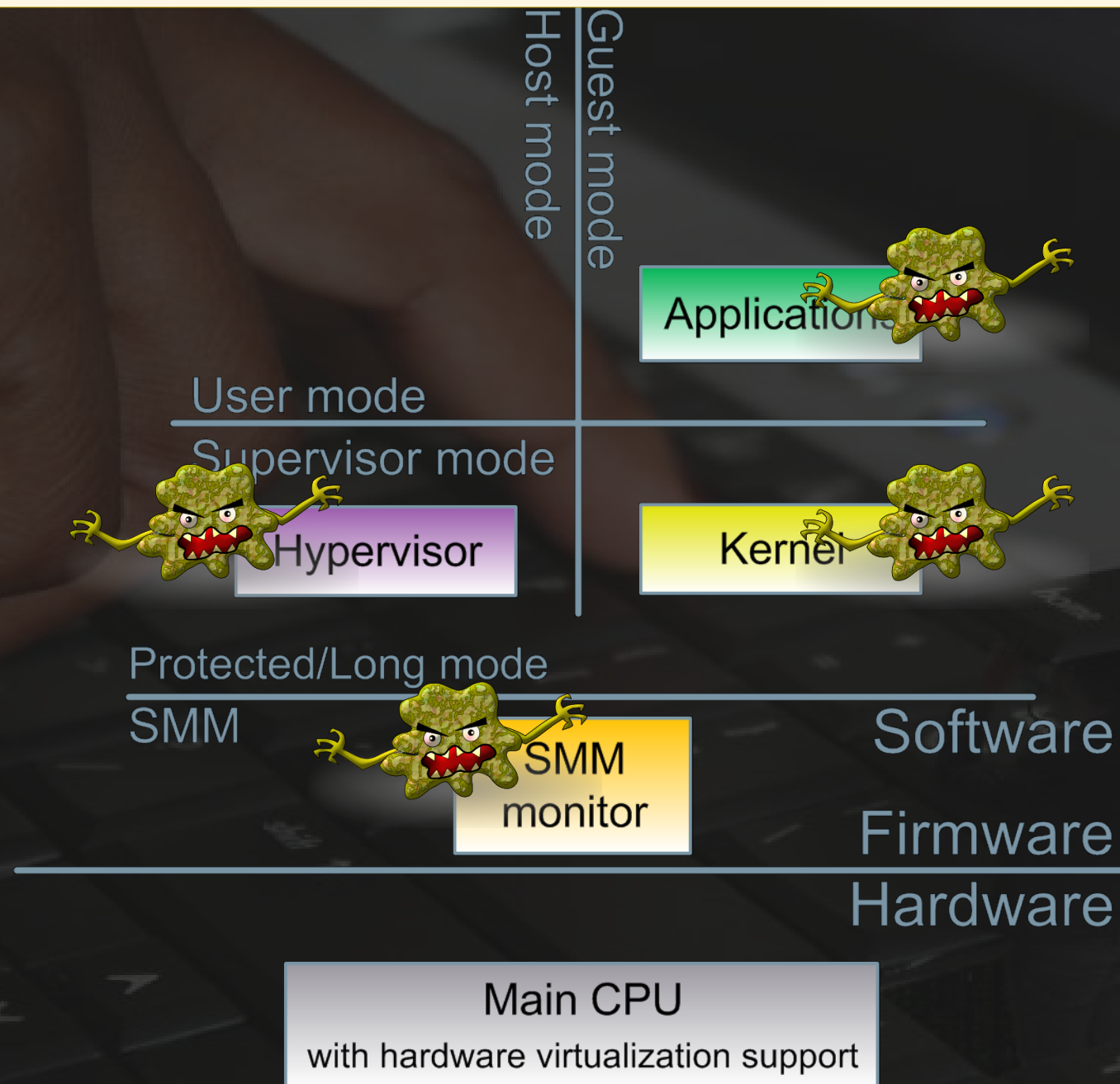
ARMS RACE Malware Developers ↔ Anti-Malware Community



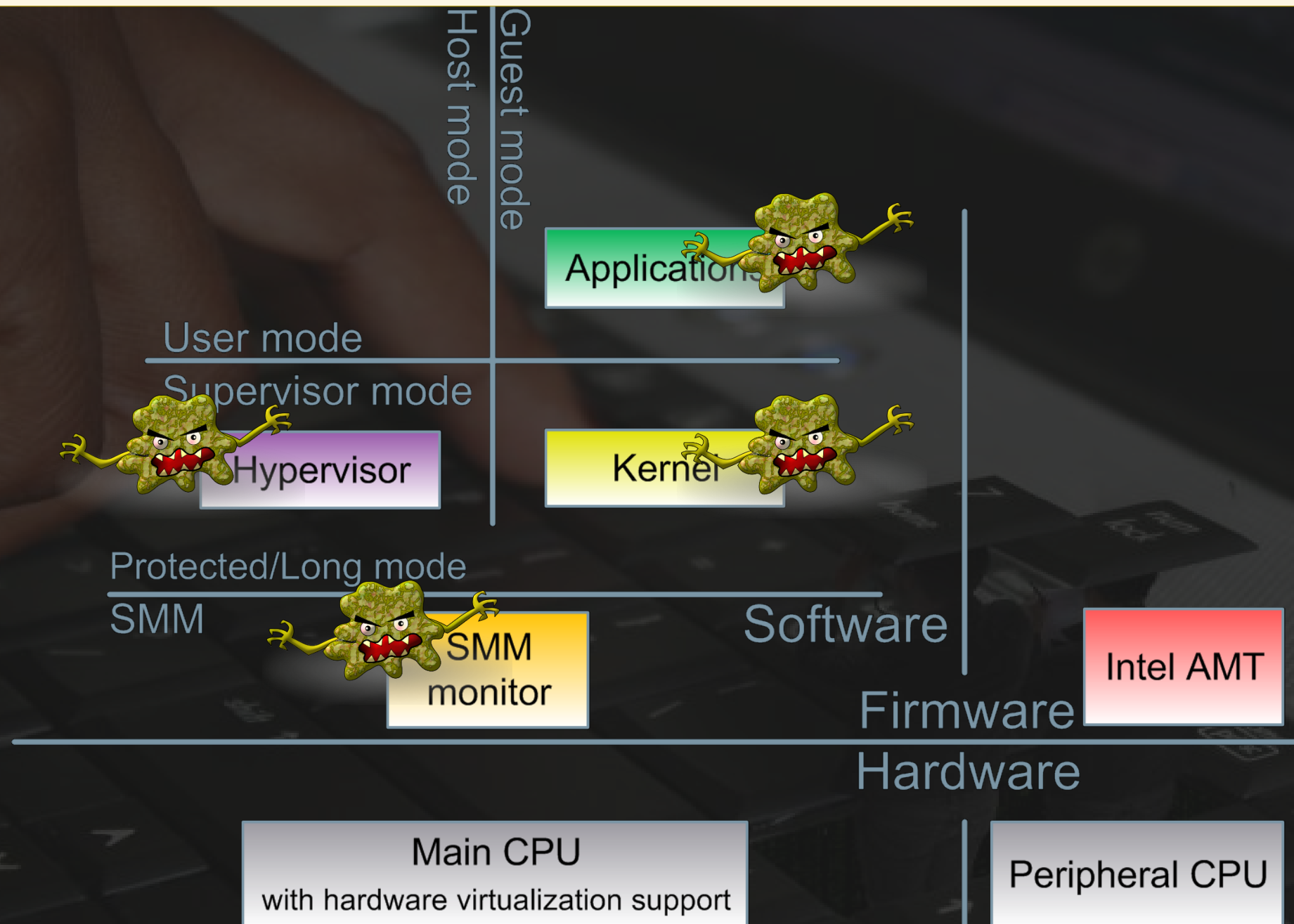
ARMS RACE Malware Developers ↔ Anti-Malware Community



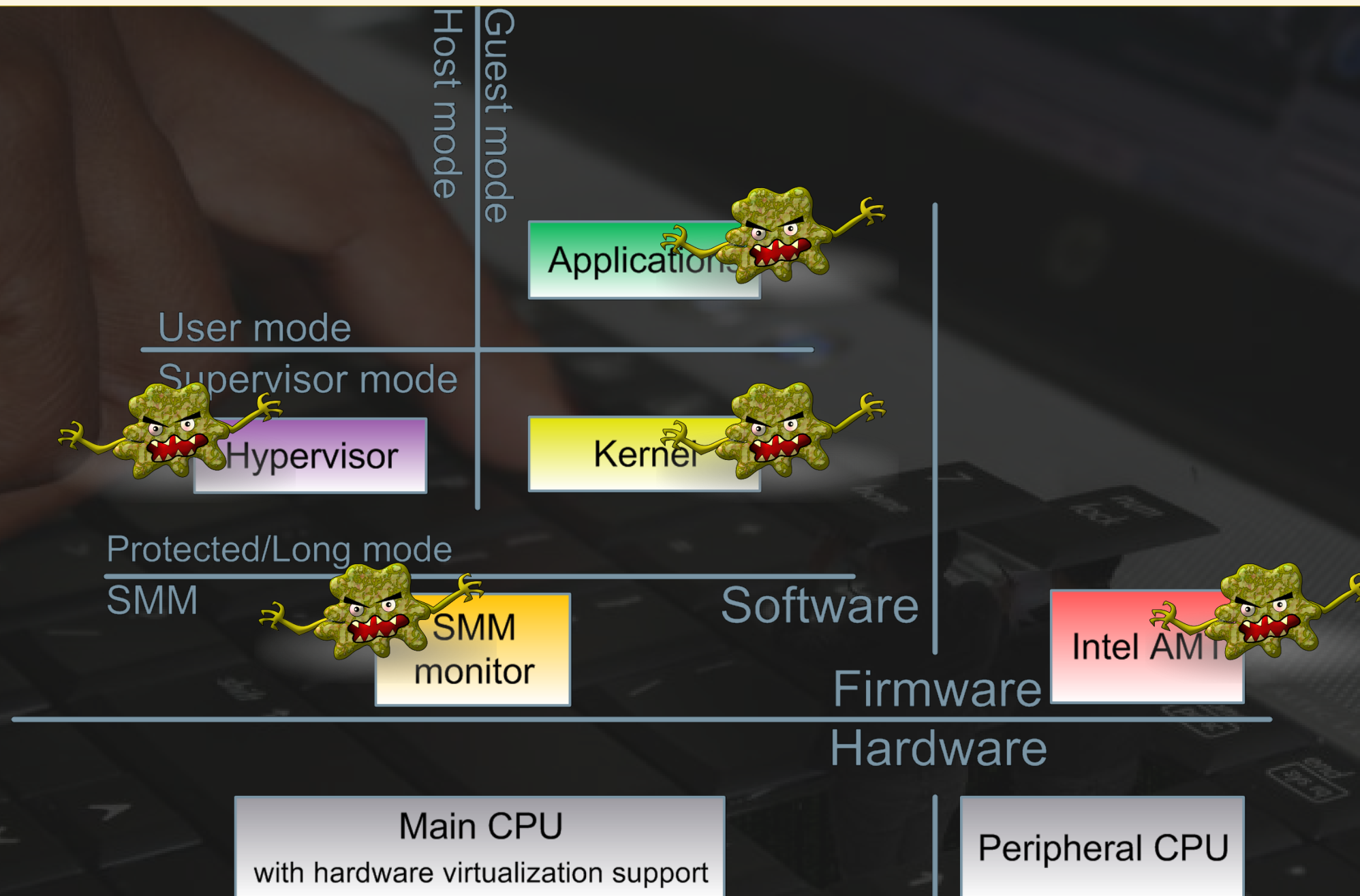
ARMS RACE Malware Developers ↔ Anti-Malware Community



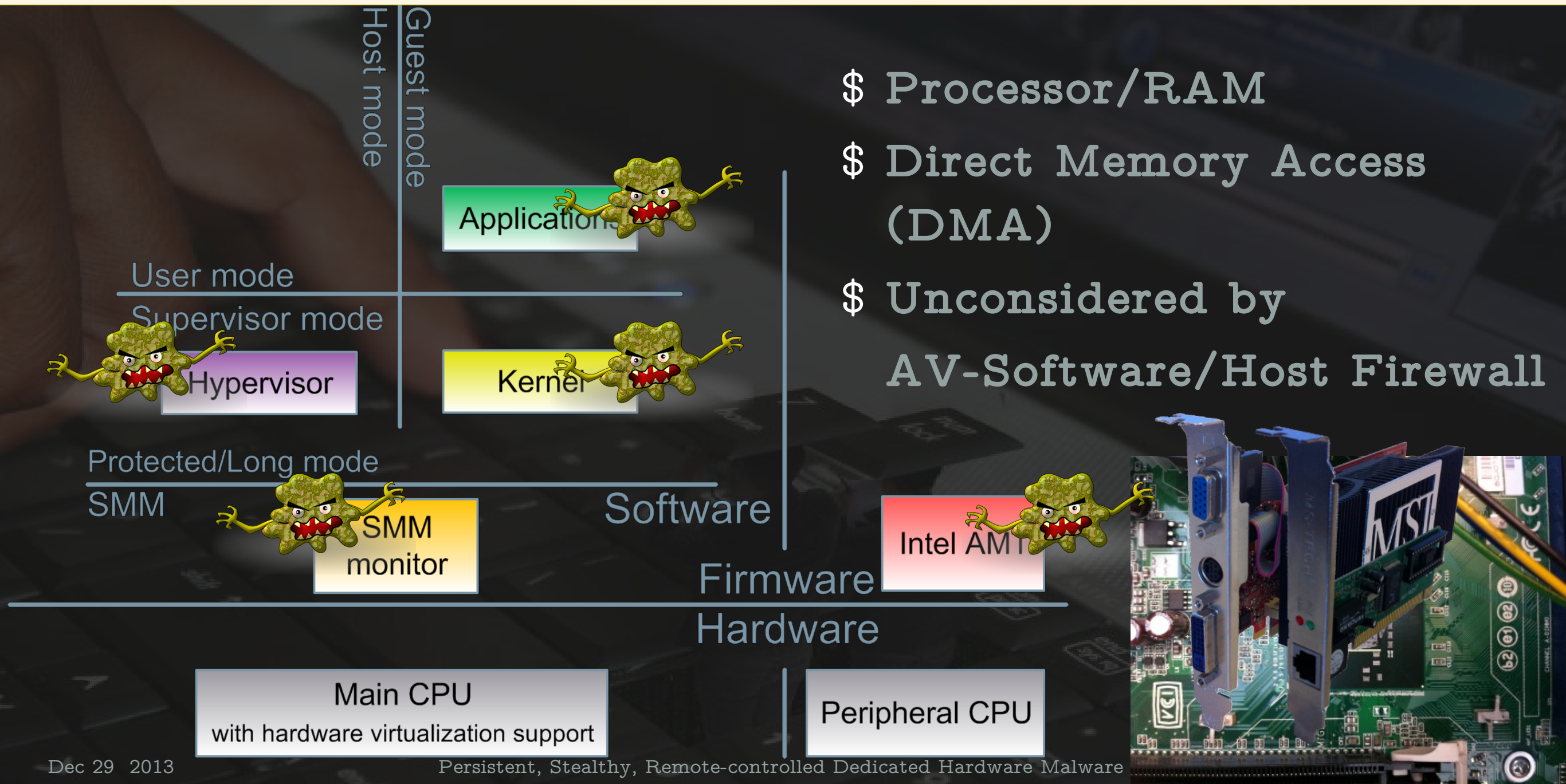
ARMS RACE Malware Developers ↔ Anti-Malware Community



ARMS RACE Malware Developers ↔ Anti-Malware Community



ARMS RACE Malware Developers ↔ Anti-Malware Community



```
[patrickx@30C3:~$] cat 'Overview'
```

① DmA based keystroke loGGER

② Out-of-Band network channel

③ Covert network channel



DAGGER

[patrickx@30C3:~\$] cat 'What is DAGGER?'

\$ It has nothing to do with the *Dagger Complex* :)



Picture: Armin Kübelbeck (CC BY-SA 3.0)

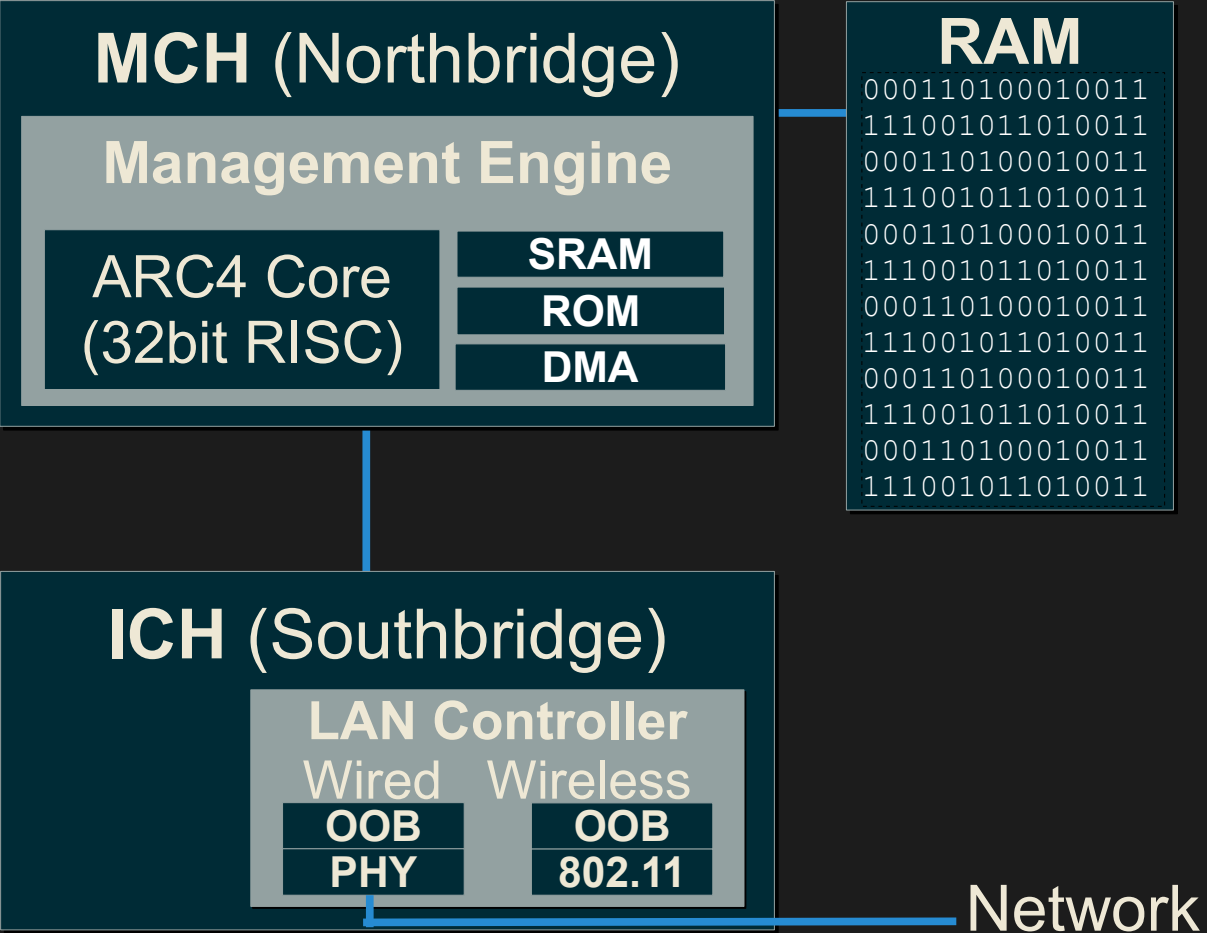
```
[patrickx@30C3:~$] cat 'What is DAGGER?'
```

- \$ Written in C / ARC4 assembly
- \$ Part of academic research project
- \$ Not only a keylogger anymore
- \$ Access to host memory
(DMA read/write)
- \$ Isolated network channel
- \$ 32bit/64bit based
attack targets
- \$...



[patrickx@30C3:~\$] cat 'Our Attack Environment'

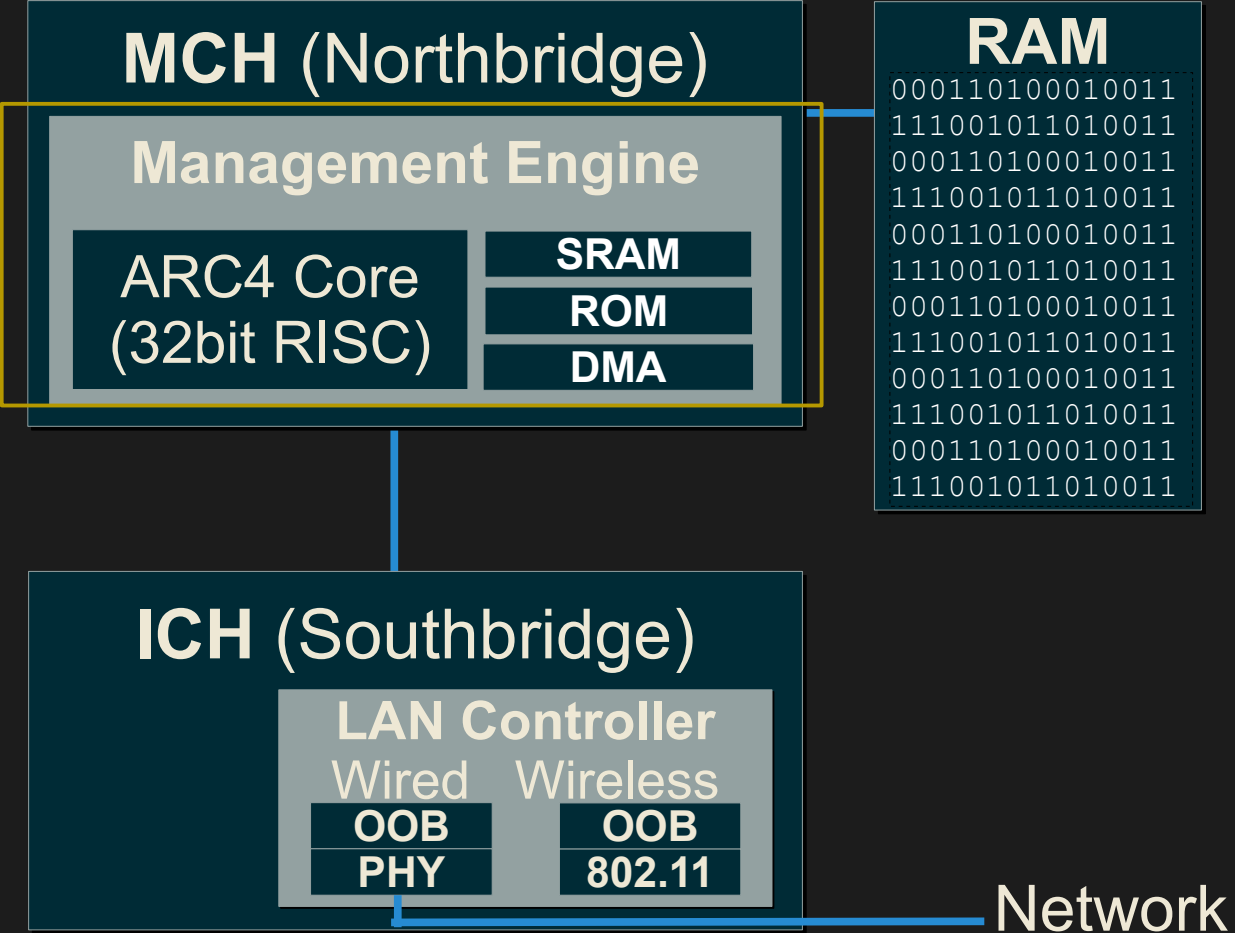
\$ Manageability Engine



(Q35 Chipset)

[patrickx@30C3:~\$] cat 'Our Attack Environment'

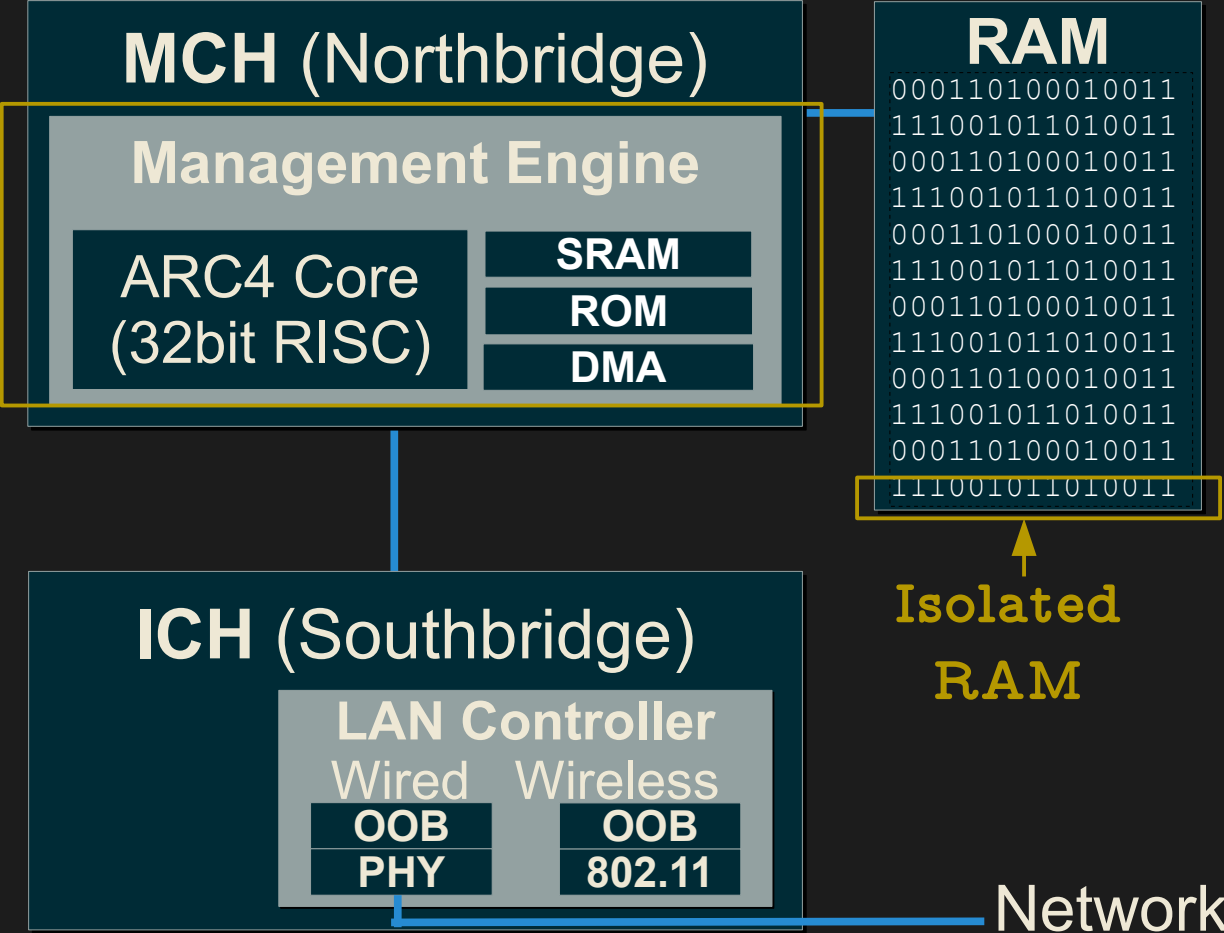
\$ Manageability Engine



(Q35 Chipset)

[patrickx@30C3:~\$] cat 'Our Attack Environment'

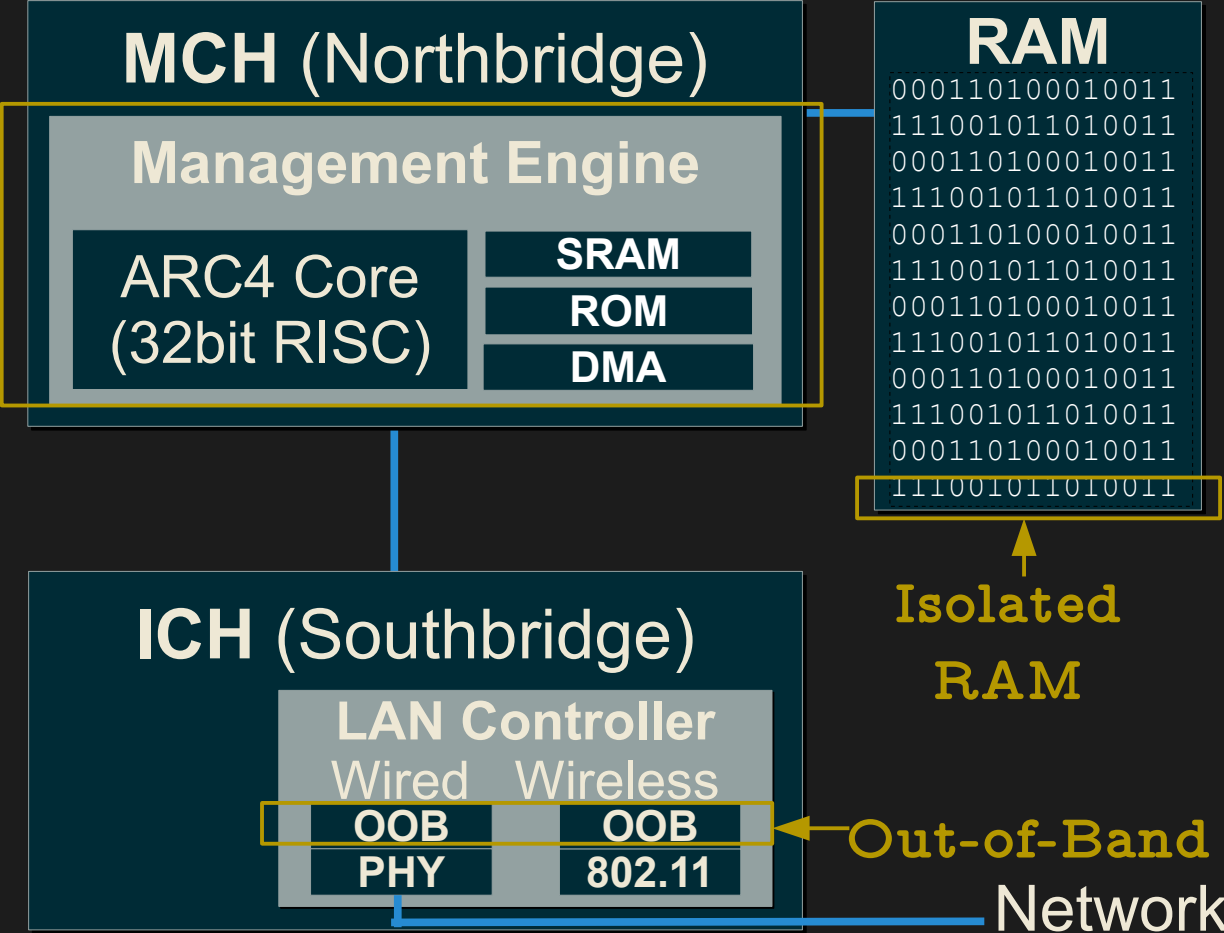
\$ Manageability Engine



(Q35 Chipset)

[patrickx@30C3:~\$] cat 'Our Attack Environment'

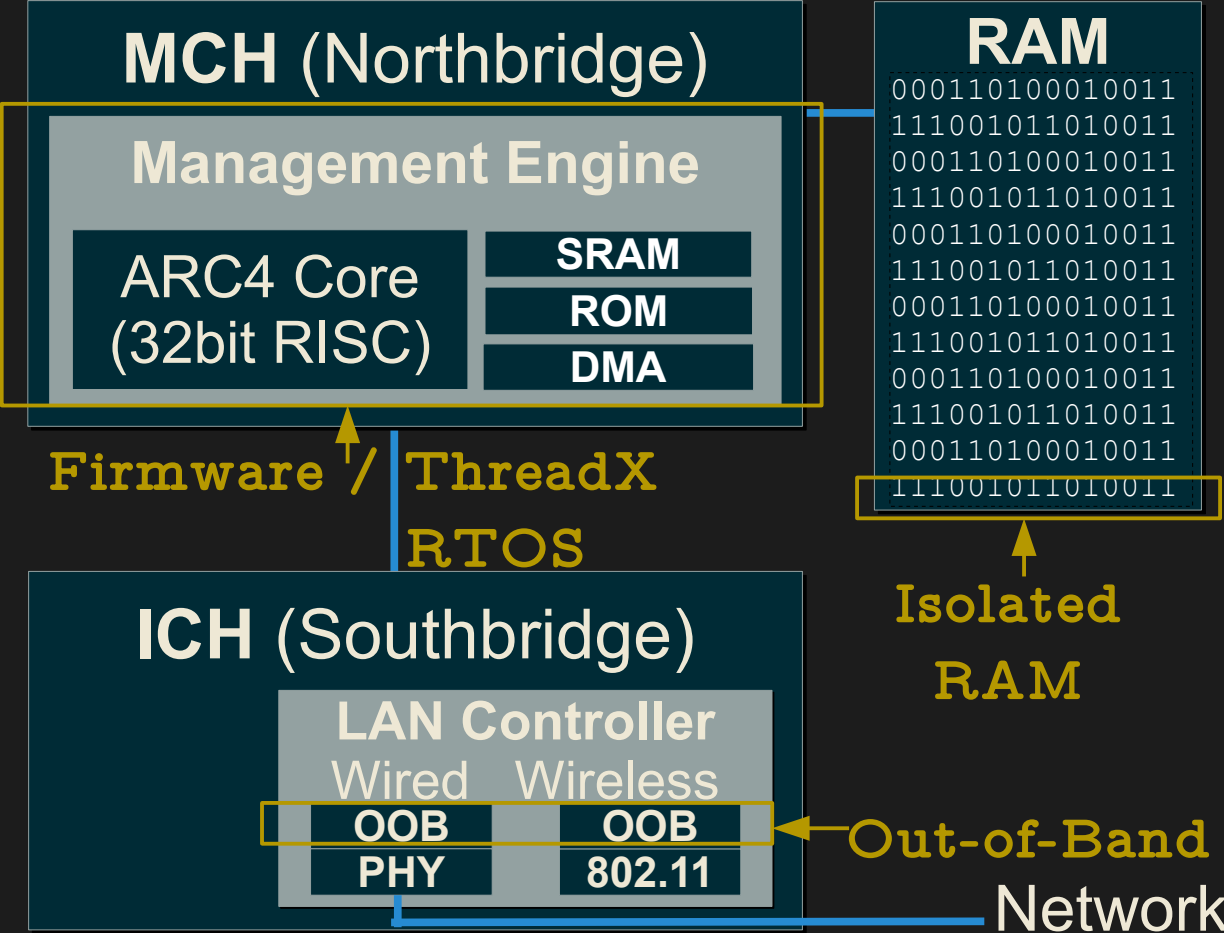
\$ Manageability Engine



(Q35 Chipset)

[patrickx@30C3:~\$] cat 'Our Attack Environment'

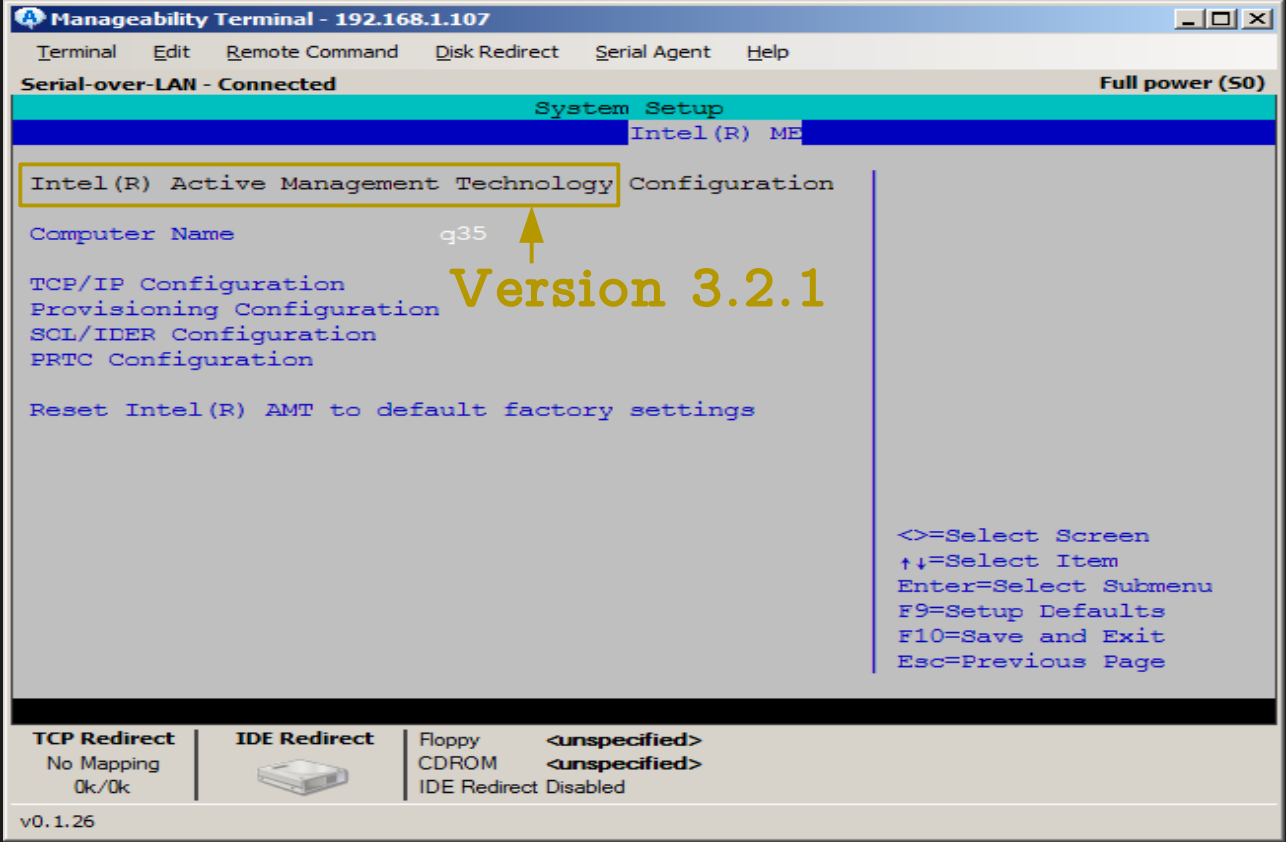
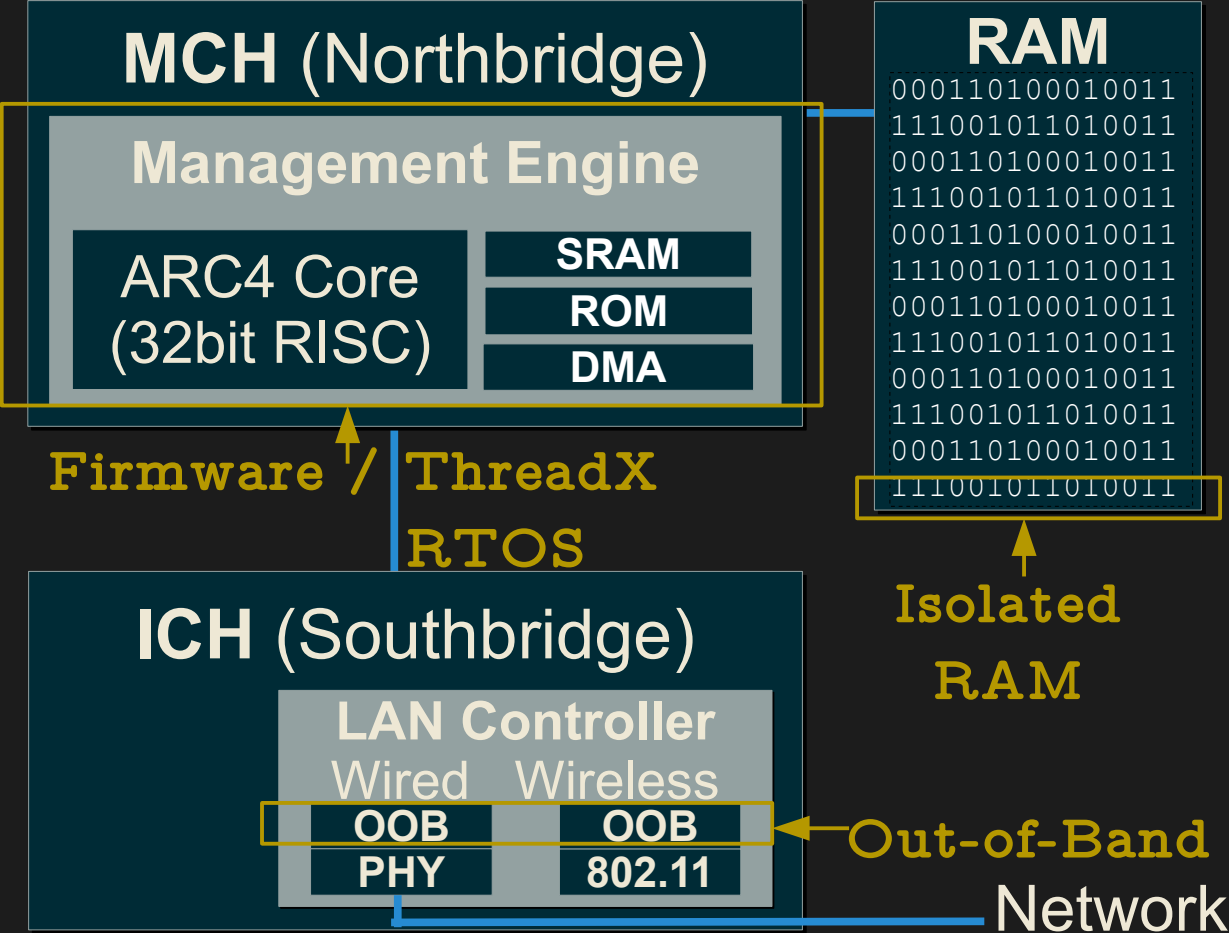
\$ Manageability Engine



(Q35 Chipset)

[patrickx@30C3:~\$] cat 'Our Attack Environment'

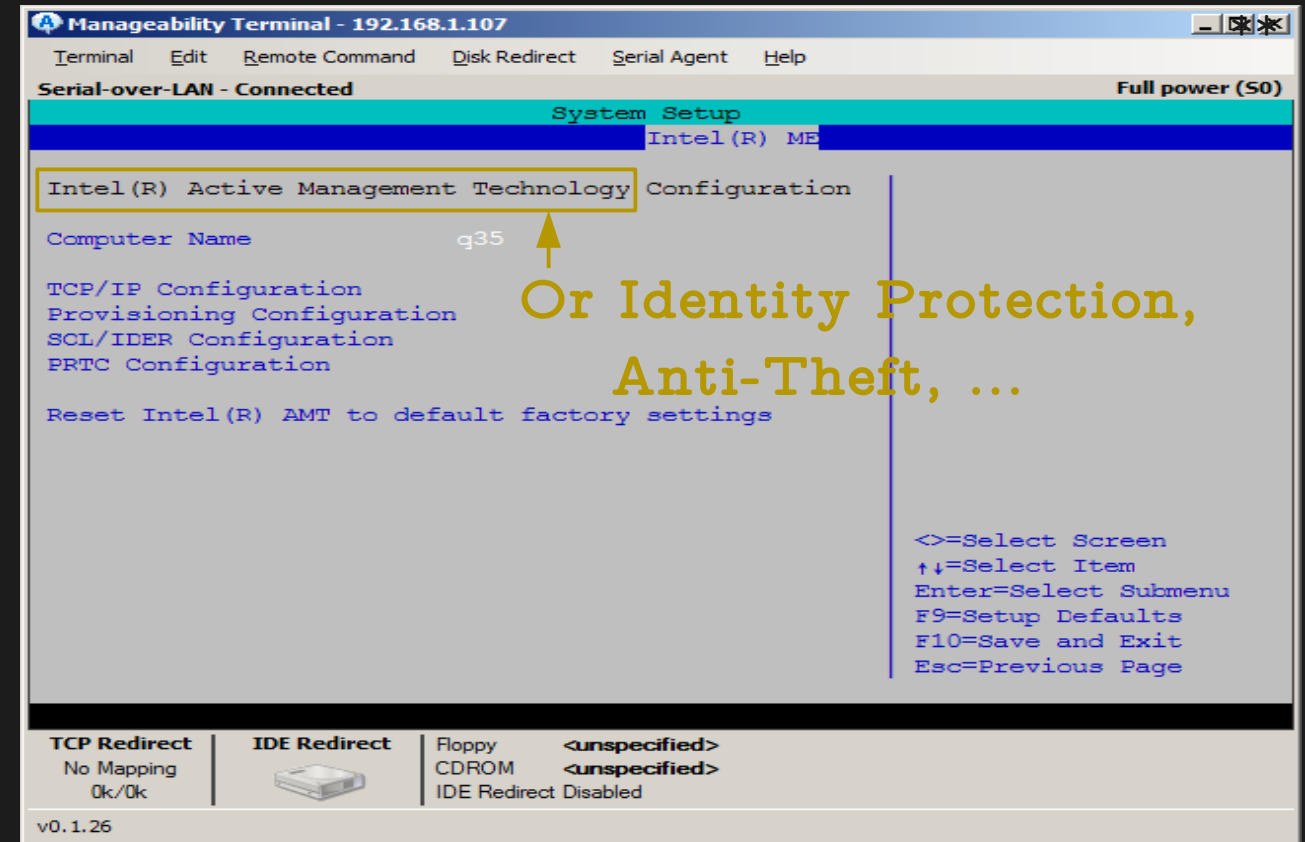
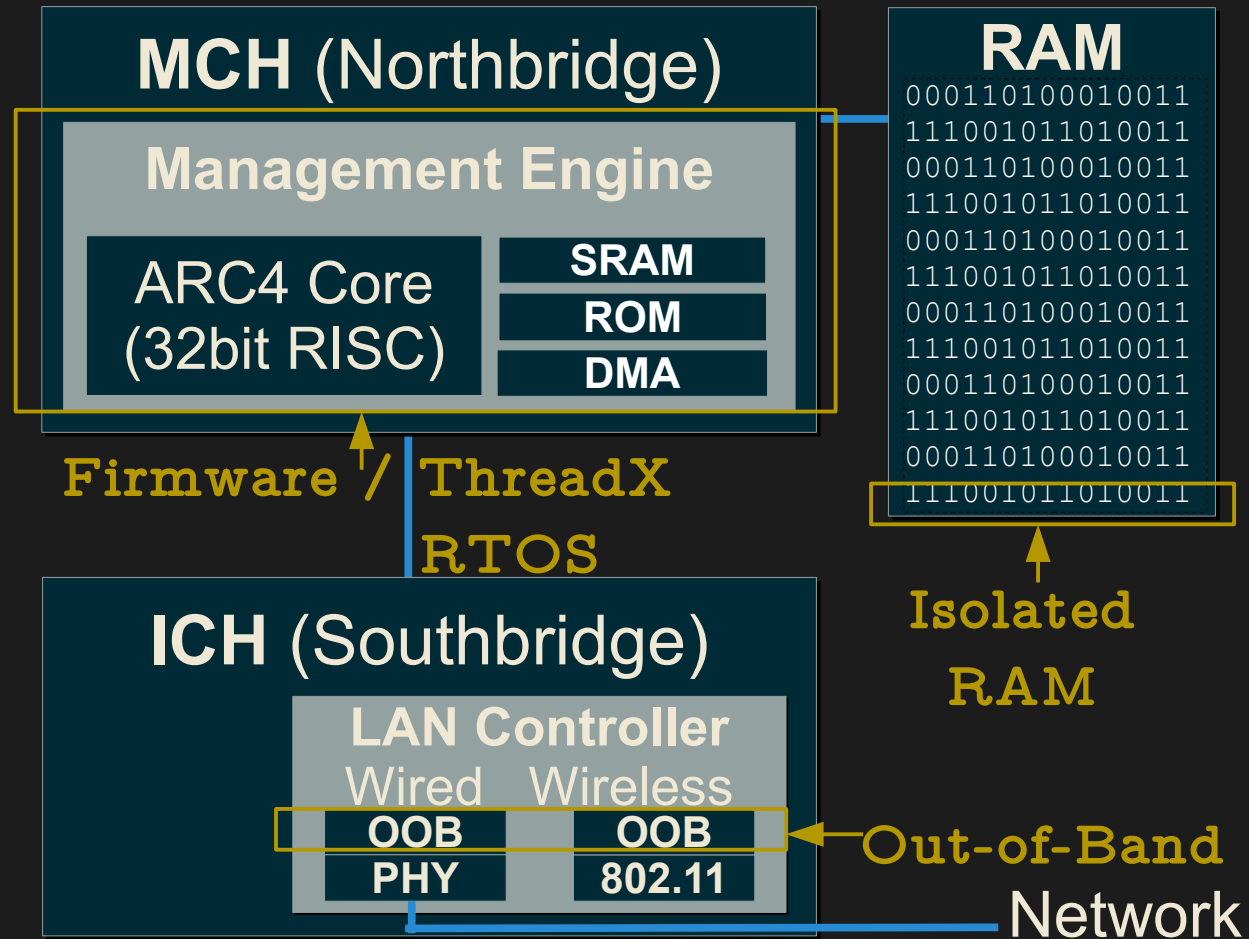
\$ Manageability Engine



(Q35 Chipset)

```
[patrickx@30C3:~$] cat 'Our Attack Environment'
```

\$ Manageability Engine



(Q35 Chipset)

```
[patrickx@30C3:~$] cat 'ARC Historical Overview'
```

\$ **M**athematical, **A**rgonaut, **R**otation & **I/O**: **MARIO** chip :)

\$ SuperFX

\$ ARC

\$ 1st ME generation: ARCTangent-A4 (ARC4)

\$ 2nd ME generation: ARCTangent-A5/ARCompact

\$ Rootkit in your laptop → [Sko12]

\$ ME related project: “[...]ME replacement compatible with the coreboot firmware [...]” → <http://intelvp.ro> or <http://me.bios.io>

\$ 30C3 assembly !

Artikbot (CC BY-SA 3.0)

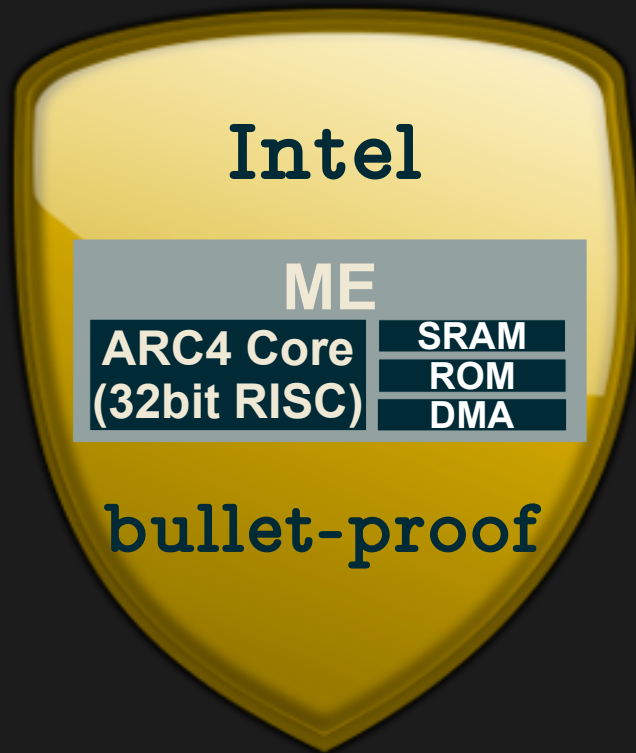


http://www.youtube.com/watch?v=k8dxLr_xVv4



[0:21:44]

```
[patrickx@30C3:~$] cat 'Our Attack Environment'
```



- \$ Nonvolatile storage isolation
- \$ Signed firmware
- \$ Measured launch
- \$ Access control
- \$...

→ DAGGER infiltration via memory remapping trick described in [Ter09] → Very good starting point!

```
[patrickx@30C3:~$] cat 'Searching for Keystrokes'
```



[patrickx@30C3:~\$] cat 'Searching for Keystrokes'



```
[patrickx@30C3:~$] cat 'Linux Target'
```

\$ Kernels tested: 2.6.32/3.0.9(32bit) / 3.5.0(64bit)

\$ Signature scan:

USB Request Block Structure

```
⋮  
struct usb_device *dev  
⋮  
  
⋮  
dma_addr_t transfer_dma  
⋮
```



A vertical double-headed arrow indicates a constant offset between the 'dev' pointer and the 'transfer_dma' field. The text 'Constant offset' is written vertically next to the arrow.

USB Device Structure

```
⋮  
  
⋮  
char *product  
⋮
```



A vertical double-headed arrow indicates a constant offset between the 'product' pointer and an unknown field. The text 'Constant offset' is written vertically next to the arrow.

```
[patrickx@30C3:~$] cat 'Linux Target'
```

\$ Kernels tested: 2.6.32/3.0.9(32bit) / 3.5.0(64bit)

\$ Signature scan:

USB Request Block Structure

```
⋮  
struct usb_device *dev  
⋮  
  
⋮  
dma_addr_t transfer_dma  
⋮
```

Constant
offset

If pointer mod 0x400 == 0

&&

If field mod 0x20 == 0

Start URB signature scan

1

USB Device Structure

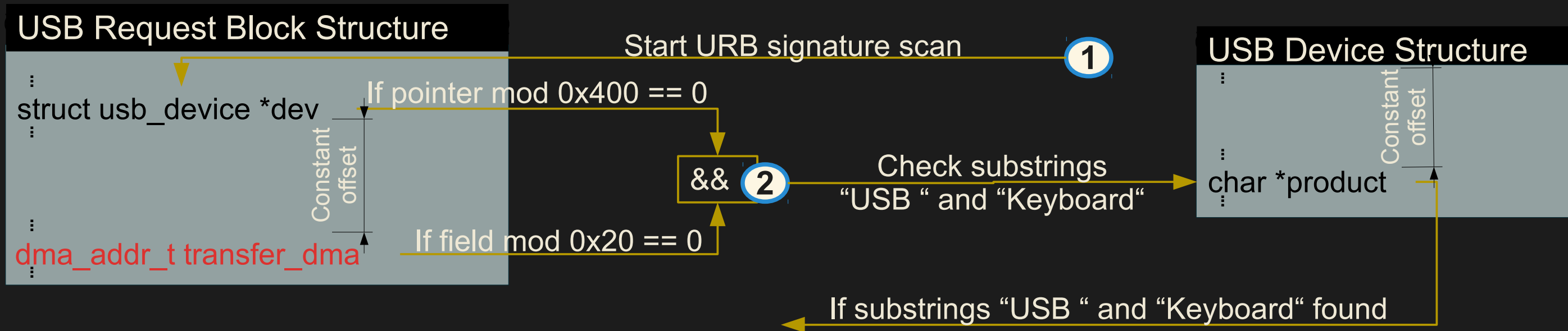
```
⋮  
  
⋮  
char *product  
⋮
```

Constant
offset

```
[patrickx@30C3:~$] cat 'Linux Target'
```

\$ Kernels tested: 2.6.32/3.0.9(32bit) / 3.5.0(64bit)

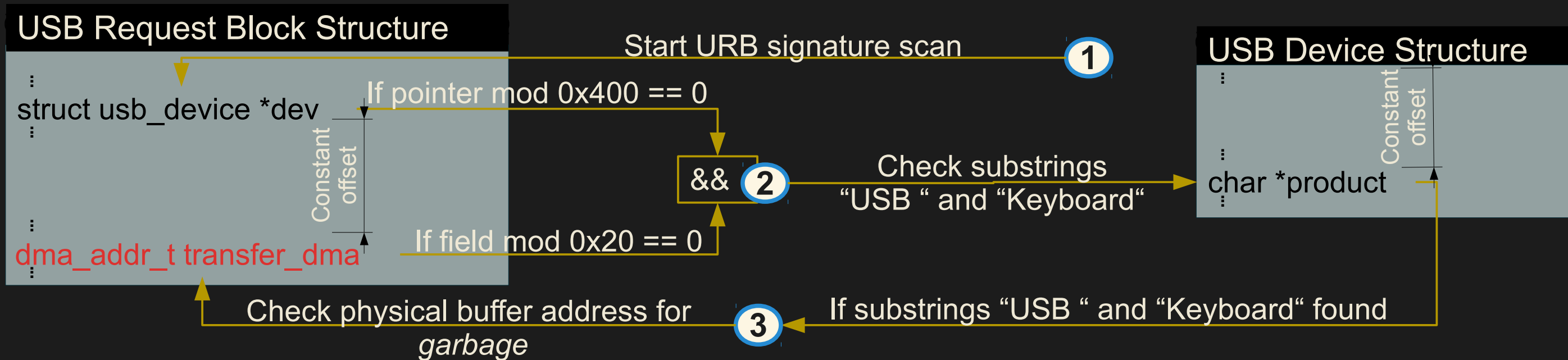
\$ Signature scan:



```
[patrickx@30C3:~$] cat 'Linux Target'
```

\$ Kernels tested: 2.6.32/3.0.9(32bit) / 3.5.0(64bit)

\$ Signature scan:



[patrickx@30C3:~\$] cat 'Linux Target'

\$ Mapping virtual to physical memory addresses

\$ 32bit:
subtract constant offset
→ 0xc0000000

\$ 64bit:
see Documentation/x86/x86_64/mm.txt

user space	0x0000000000000000
hole	0x00007fffffffffff
guard hole	0xffff800000000000
all phys. memory	0xffff880000000000
hole	0xffffc80000000000
vmalloc/ioremap space	0xffffc90000000000
hole	0xffffe8fffffffffff
virtual memory map	0xffffea0000000000
unused hole	0xffffeafffffffffffff
kernel text mapping	0xfffffffff8000000
module mapping space	0xffffffffffa0000000
	0xfffffffffffff00000

```
[patrickx@30C3:~$] cat 'Windows Target'
```

\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:

```
[patrickx@30C3:~$] cat 'Windows Target'
```

\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:

KiInitialPCR

⋮

KdVersionBlock

```
[patrickx@30C3:~$] cat 'Windows Target'
```

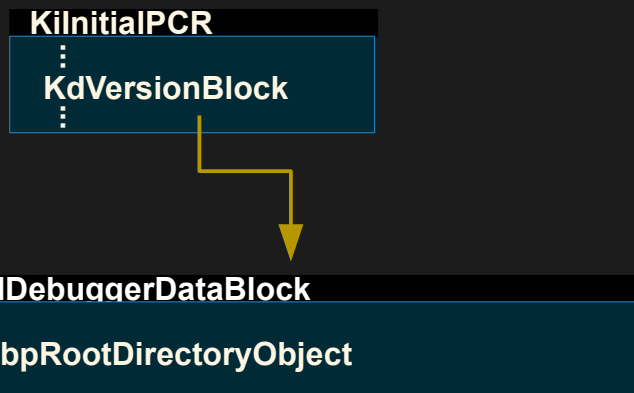
\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



```
[patrickx@30C3:~$] cat 'Windows Target'
```

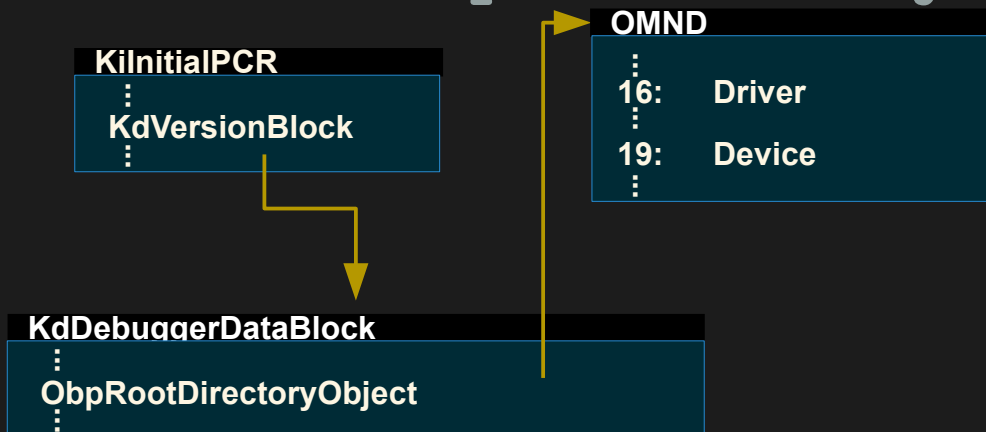
\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



```
[patrickx@30C3:~$] cat 'Windows Target'
```

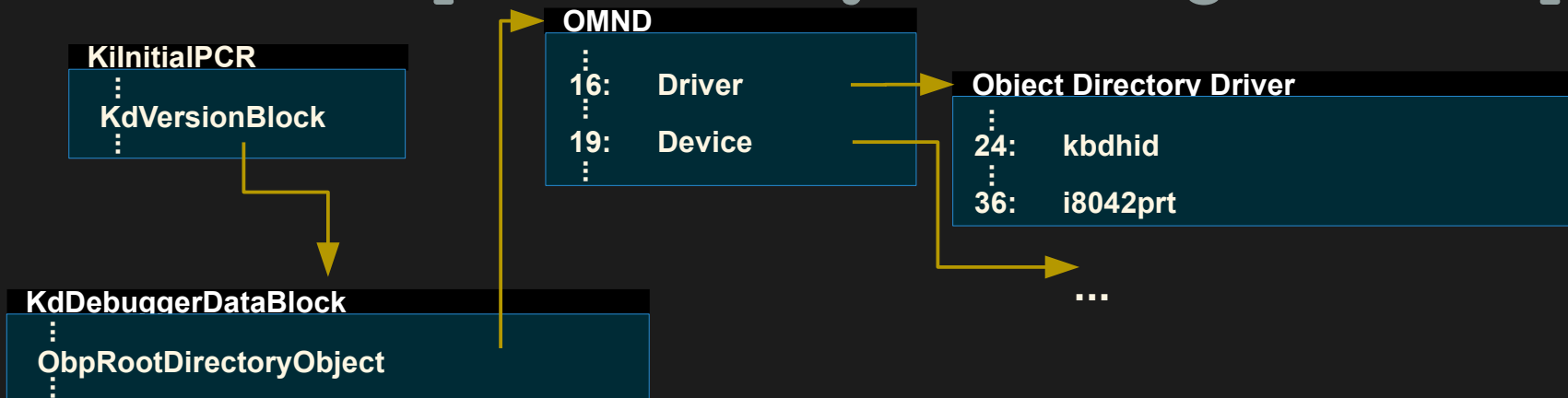
\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



```
[patrickx@30C3:~$] cat 'Windows Target'
```

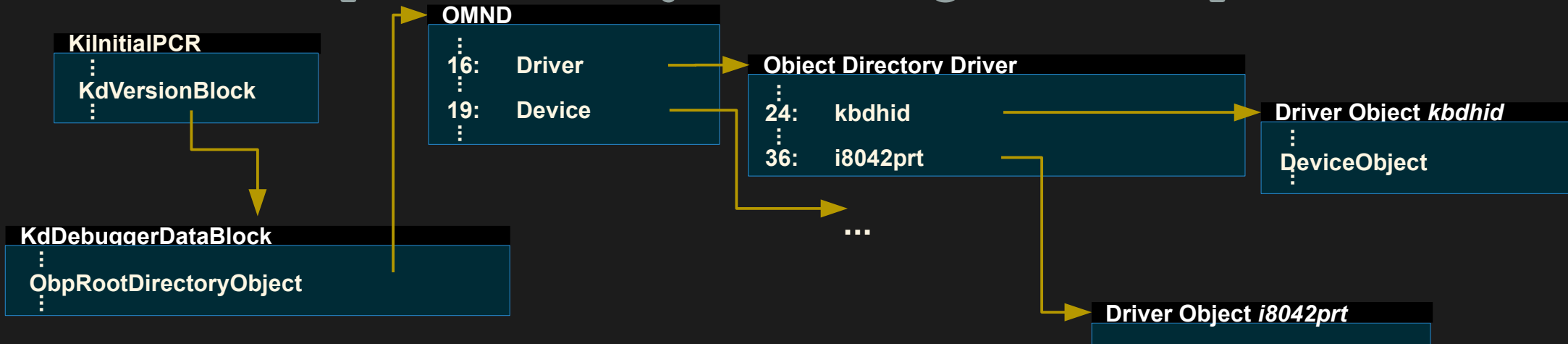
\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



```
[patrickx@30C3:~$] cat 'Windows Target'
```

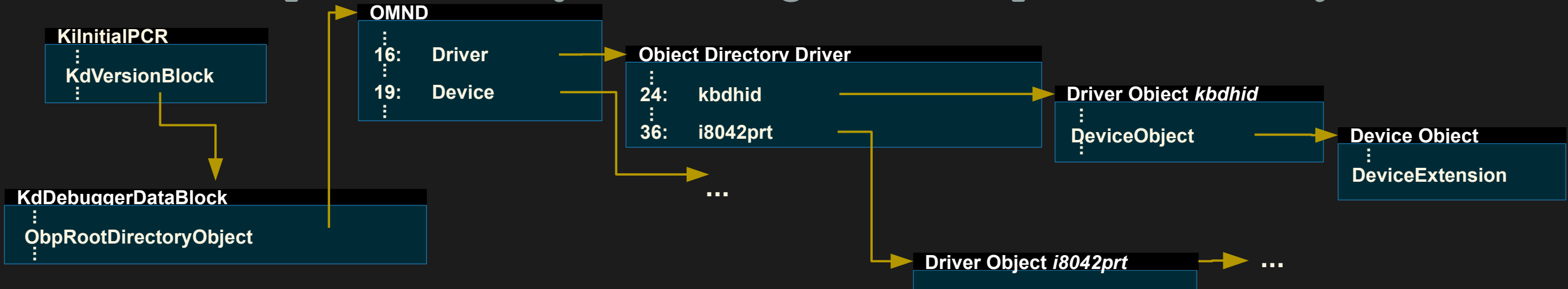
\$ Kernels tested: Vista / 7

\$ CR3 value required

(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



```
[patrickx@30C3:~$] cat 'Windows Target'
```

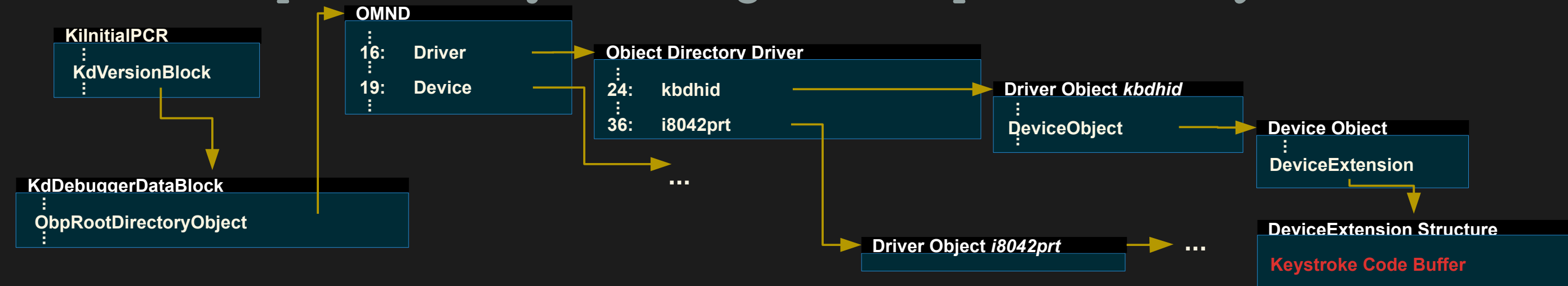
\$ Kernels tested: Vista / 7

\$ CR3 value required

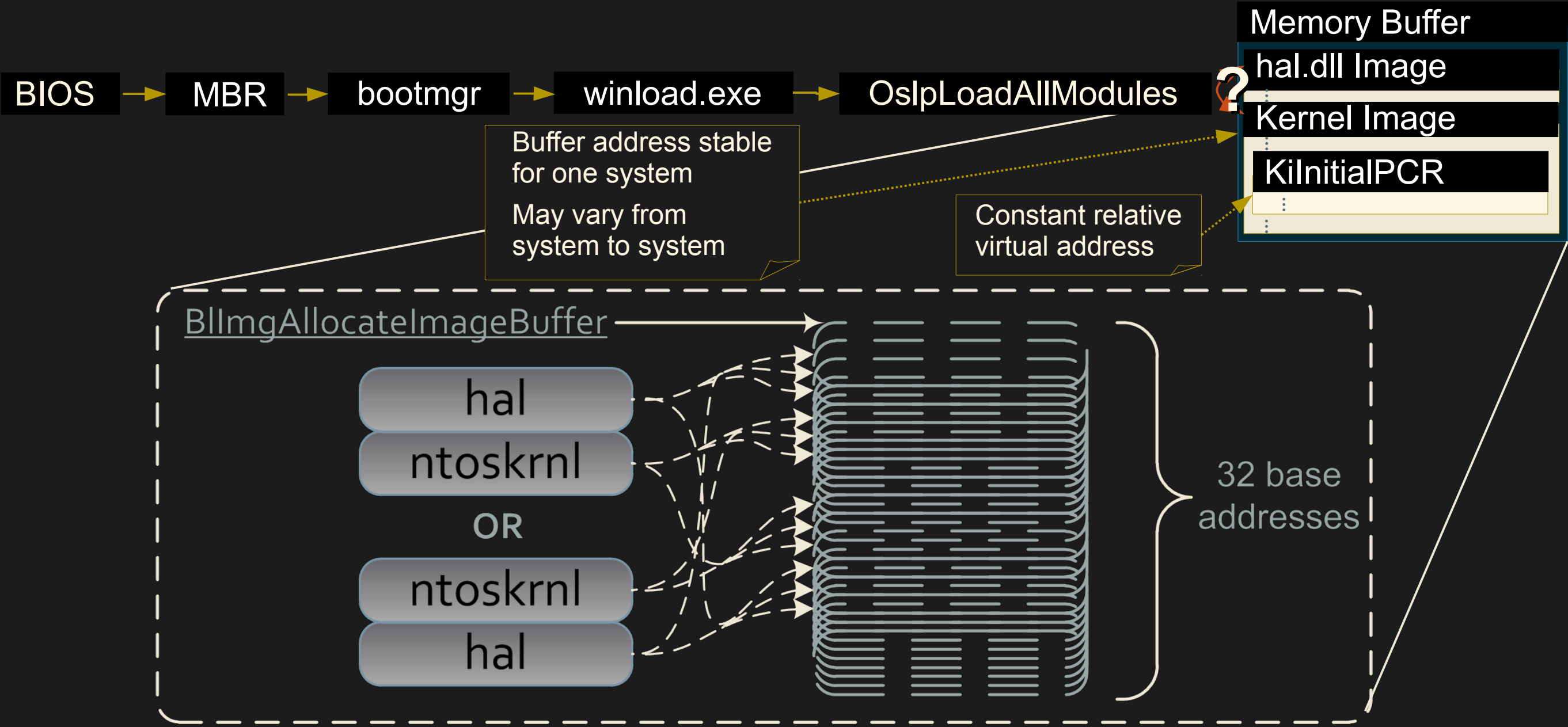
(Verified within DAGGER/DAGGER traverses page tables)

\$ No source code: IDA Pro, WinDbg, debug symbols

\$ Search path via Object Manager Namespace Directory:



[patrickx@30C3:~\$] cat 'Address Randomization'



```
[patrickx@30C3:~$] cat 'Required ME Features'
```

\$ DMA read access

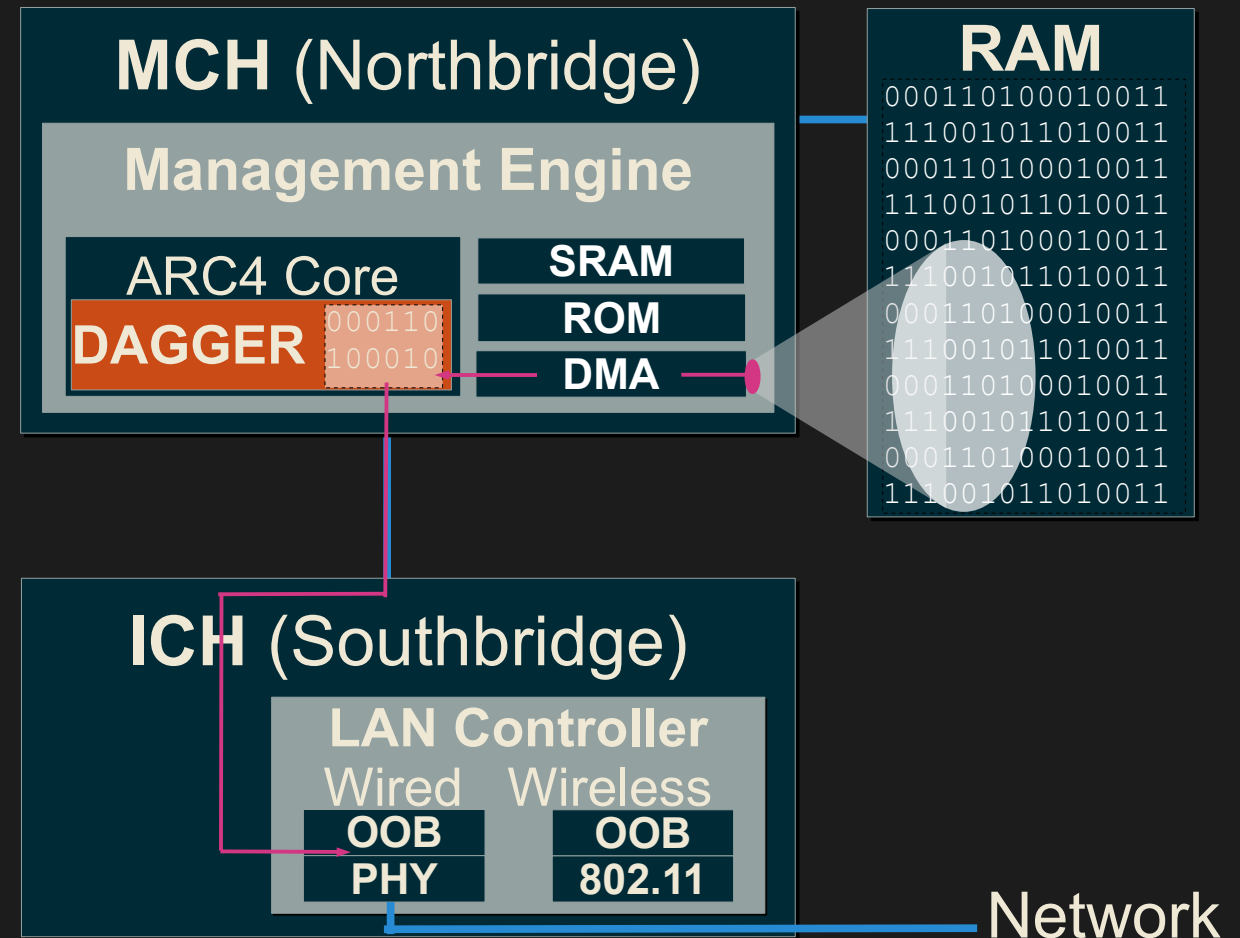
→ **easy**

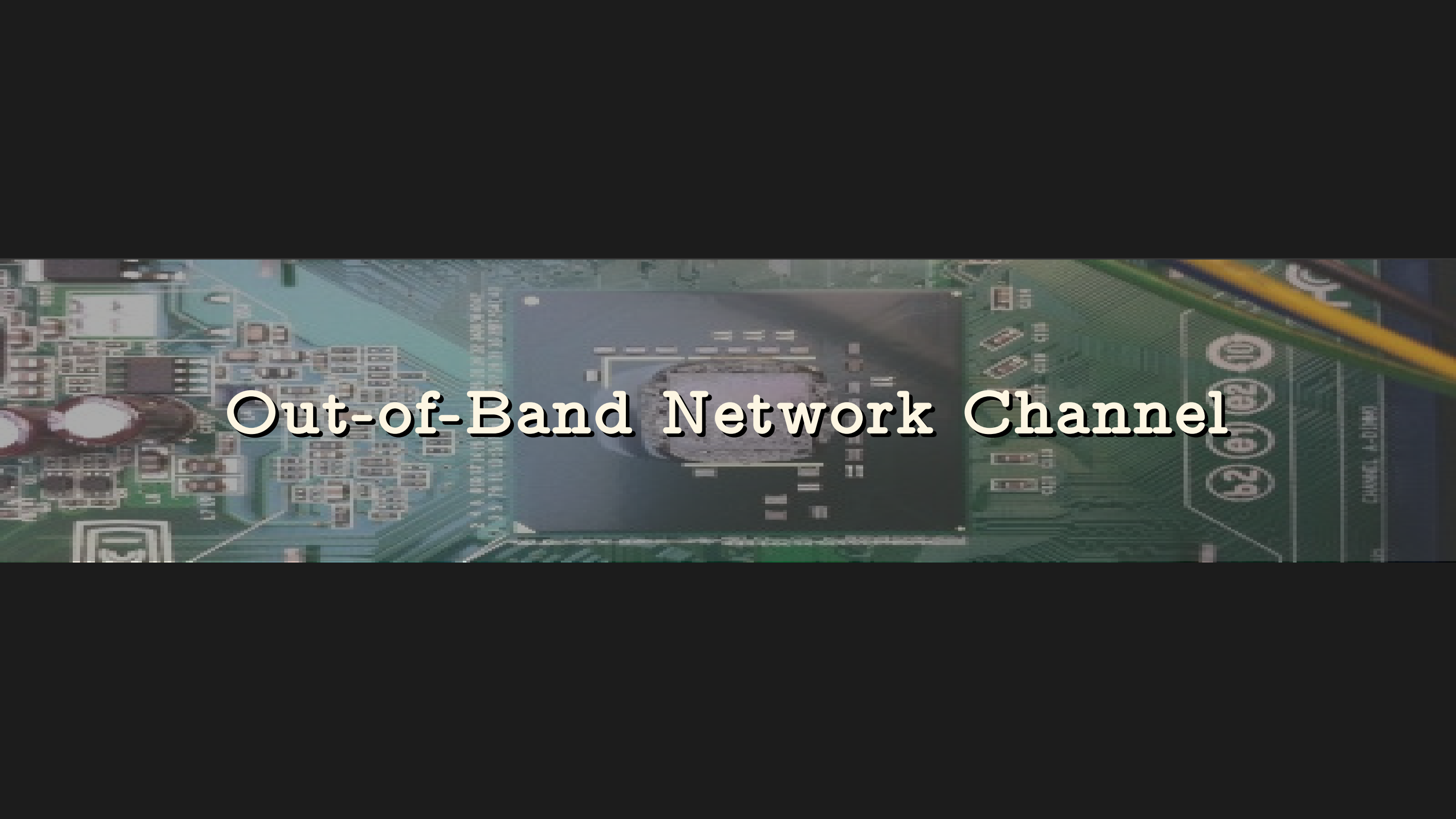
(we just changed
two bits)

\$ Stealthy network
channel

→ **challenging**

(more than
two bits :))





Out-of-Band Network Channel

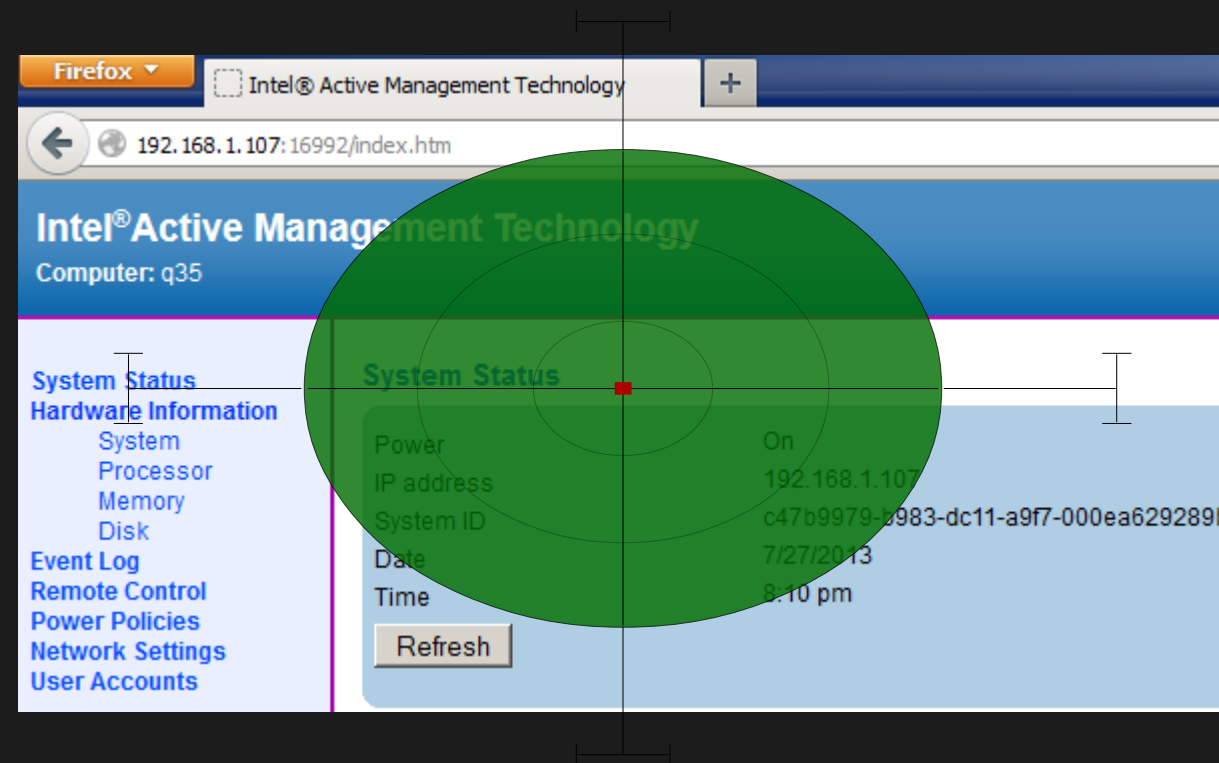
```
[patrickx@30C3:~$] cat 'Target: ME OOB'
```

\$ Needed not only to exfiltrate captured keystroke codes,
but also to download new attack code!



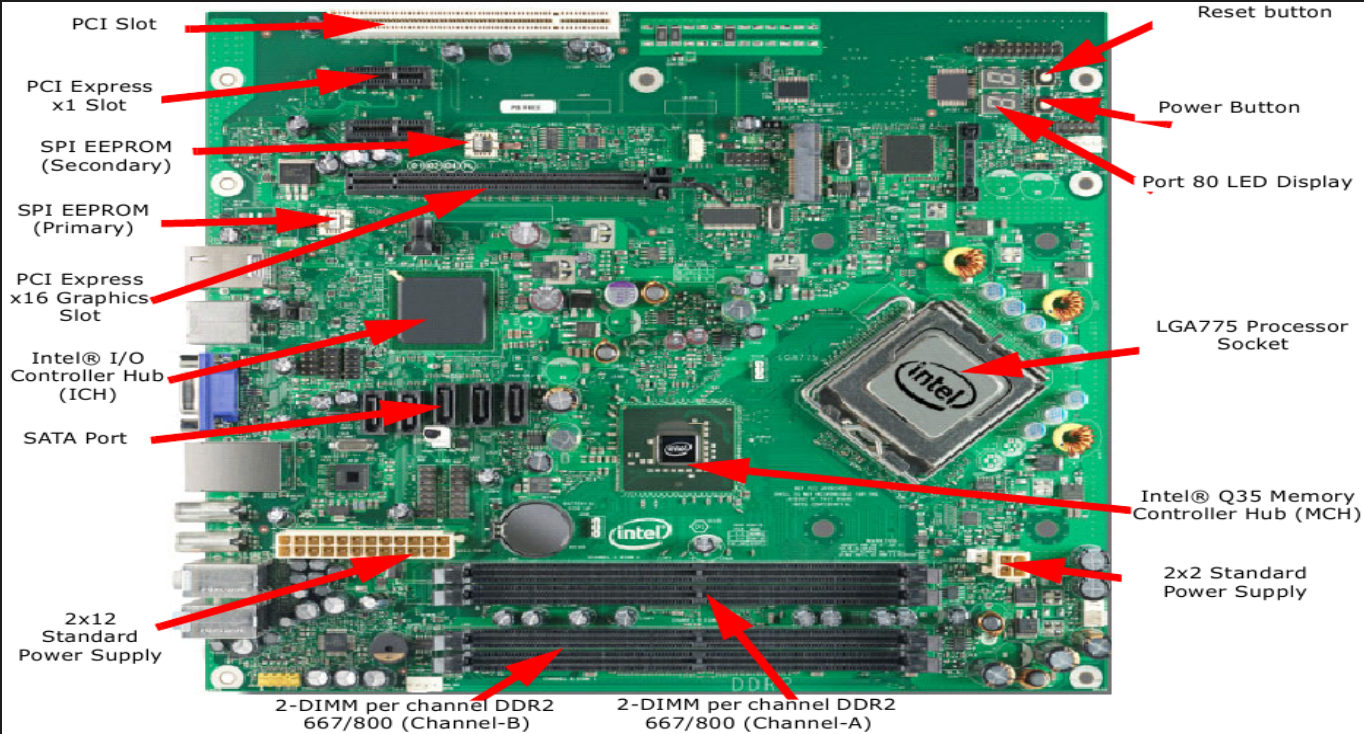
```
[patrickx@30C3:~$] cat 'Target: ME OOB'
```

\$ Needed not only to exfiltrate captured keystroke codes, but also to download new attack code!

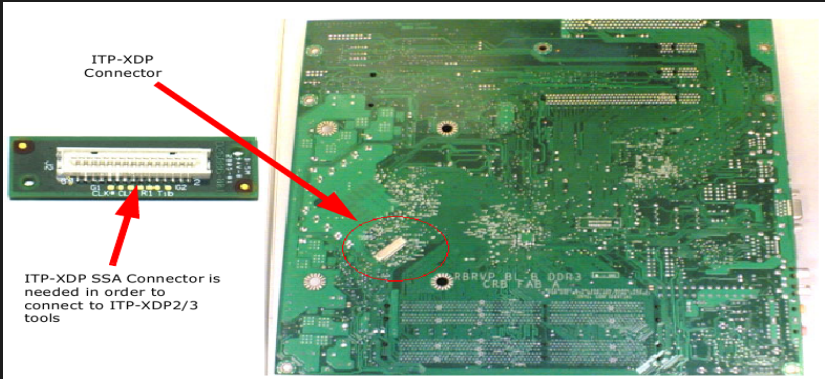


How to find firmware code responsible for webserver replies?

[patrickx@30C3:~\$] cat 'Some Tools Required'



Board Features ([Int07], p.11)



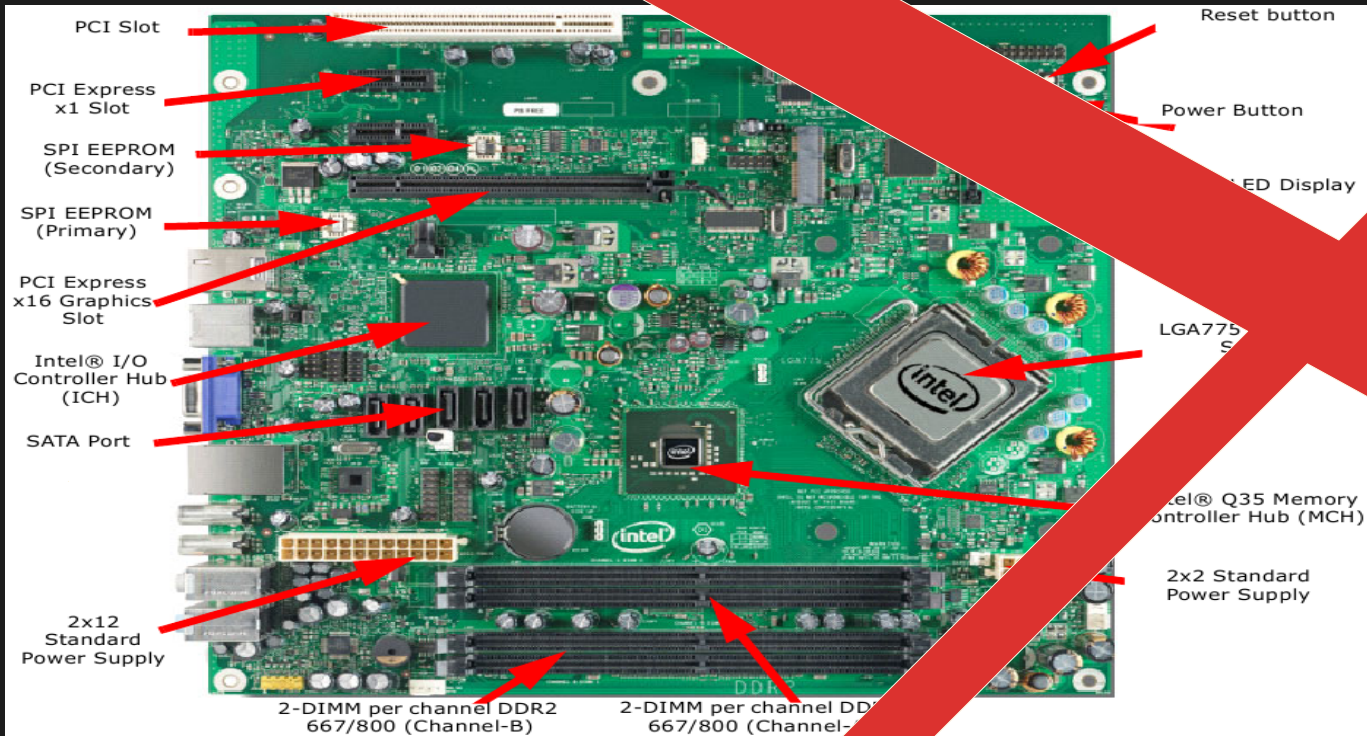
ITP-XDP Connector location (J2BC) ([Int07], p.20)

- Programming DMA hardware over JTAG port in debugger
- DMA-ing 64 bytes from system memory containing malicious VMExit handler code to internal chipset memory

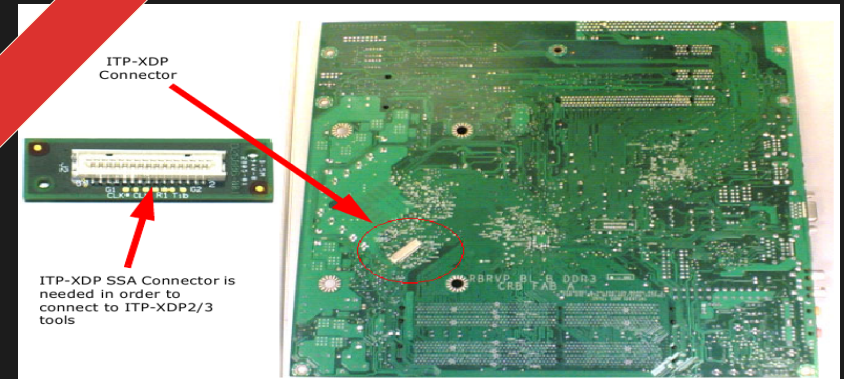
```
ARC> dump 0x20000000 00000000 00000000 00000000 :
02000000: 00000000 00000000 00000000 00000000 :
02000010: 00000000 00000000 00000000 00000000 :
02000020: 00000000 00000000 00000000 00000000 :
02000030: 00000000 00000000 00000000 00000000 :
02000040: 00000000 00000000 00000000 00000000 :
02000050: 00000000 00000000 00000000 00000000 :
02000060: 00000000 00000000 00000000 00000000 :
02000070: 00000000 00000000 00000000 00000000 :
ARC> arc:dma<1,0x73000,0,0x2000000,64>
Transferred 64 B of data from Host 0x00073000
General Status = 1
02000000: fa 55 8b ec 81 ec 84 00 00 00 89 45 b4 89 5d b8 : .U.....E..l.
02000010: 87 4d bc 89 55 c0 87 75 cc 89 7d d0 0f 20 d0 89 : .M.U..>...
02000020: 45 c4 8d 45 f8 50 68 02 44 00 00 e8 b8 08 00 00 : E.E.Ph.D....
02000030: 8d 45 d4 50 68 1e 68 00 00 e8 aa 08 00 00 8d 45 : .E.Ph.h.....E
02000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :
02000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :
02000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :
02000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :
ARC> dump 0x20000000
02000000: ec8b55fa 0084ec81 45890000 b85d89b4 : .U.....E..l.
02000010: 87bc4d89 7589c055 d07d89cc 89d0200f : .M..U..>....
```

Let's Program DMA Manually ([Bul08], p.13)

```
[patrickx@30C3:~$] cat 'Some Tools Required'
```



Board Features ([Int07], p.11)



ITP-XDP Connector location (J2BC)

Programming DMA hardware over JTAG port in debugger

Copying 64 bytes from system memory containing malicious handler code to internal chipset memory

```
02000000: 00000000 00000000 00000000 00000000 .....  
02000004: 00000000 00000000 00000000 00000000 .....  
02000008: 00000000 00000000 00000000 00000000 .....  
0200000c: 00000000 00000000 00000000 00000000 .....  
02000010: 00000000 00000000 00000000 00000000 .....  
02000014: 00000000 00000000 00000000 00000000 .....  
02000018: 00000000 00000000 00000000 00000000 .....  
0200001c: 00000000 00000000 00000000 00000000 .....  
02000020: 00000000 00000000 00000000 00000000 .....  
02000024: 00000000 00000000 00000000 00000000 .....  
02000028: 00000000 00000000 00000000 00000000 .....  
0200002c: 00000000 00000000 00000000 00000000 .....  
02000030: 00000000 00000000 00000000 00000000 .....  
02000034: 00000000 00000000 00000000 00000000 .....  
02000038: 00000000 00000000 00000000 00000000 .....  
0200003c: 00000000 00000000 00000000 00000000 .....  
02000040: 00000000 00000000 00000000 00000000 .....  
02000044: 00000000 00000000 00000000 00000000 .....  
02000048: 00000000 00000000 00000000 00000000 .....  
0200004c: 00000000 00000000 00000000 00000000 .....  
02000050: 00000000 00000000 00000000 00000000 .....  
02000054: 00000000 00000000 00000000 00000000 .....  
02000058: 00000000 00000000 00000000 00000000 .....  
0200005c: 00000000 00000000 00000000 00000000 .....  
02000060: 00000000 00000000 00000000 00000000 .....  
02000064: 00000000 00000000 00000000 00000000 .....  
02000068: 00000000 00000000 00000000 00000000 .....  
0200006c: 00000000 00000000 00000000 00000000 .....  
02000070: 00000000 00000000 00000000 00000000 .....  
02000074: 00000000 00000000 00000000 00000000 .....  
02000078: 00000000 00000000 00000000 00000000 .....  
0200007c: 00000000 00000000 00000000 00000000 .....  
arc> arc:dma<1,0x7f> 00000000 00000000 00000000 00000000  
Transferred 64 B of 00000000 00000000 00000000 00000000  
General Status=1 00000000 00000000 00000000 00000000  
02000000: fa 55 8b ec 81 00 00 89 45 b4 89 5d b8 :U..U...E..l.  
02000004: 89 4d bc 89 55 cb 00 00 89 7d d0 0f 20 d0 89 :M..U..>..  
02000008: 45 c4 8d 45 f8 50 60 02 44 00 e8 b8 08 00 00 :E..E.Ph.d..  
0200000c: 8d 45 d4 50 68 1e 68 00 00 e8 aa 08 00 00 8d 45 :E.Ph.h.....E  
02000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000014: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000018: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200001c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000024: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000028: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200002c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000034: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000038: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200003c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000044: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000048: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200004c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000054: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000058: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200005c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000064: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000068: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200006c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000074: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
02000078: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
0200007c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 :.....  
arc> dump 0x20000000 00000000 00000000 00000000  
02000000: ec8b55fa 0084ec81 45890000 b85d89b4 :..U...E..l.  
02000004: 89bc4d89 7589c055 d07d89cc 89d0200f :M..U..>..  
02000008: 00000000 00000000 00000000 00000000 :.....
```

13

7/6/2008

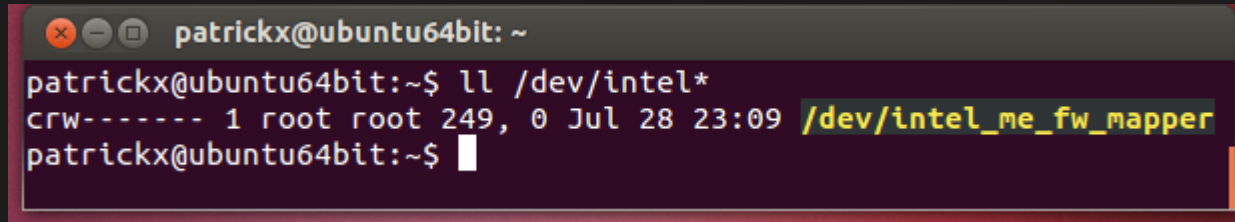
Copyright © Intel Corporation, 2006. All rights reserved. Third-party marks and brands are the property of their respective owners. All products, dates, and figures are preliminary and subject to change without notice.



Let's Program DMA Manually ([Bul08], p.13)

[patrickx@30C3:~\$] cat 'Our Research Tools'

\$ Linux:

A terminal window titled 'patrickx@ubuntu64bit: ~' with standard window controls. The prompt is 'patrickx@ubuntu64bit:~\$'. The command 'll /dev/intel*' has been entered and executed. The output is 'crw----- 1 root root 249, 0 Jul 28 23:09 /dev/intel_me_fw_mapper'. The path is highlighted in yellow. The prompt 'patrickx@ubuntu64bit:~\$' is followed by a cursor.

```
patrickx@ubuntu64bit:~$ ll /dev/intel*
crw----- 1 root root 249, 0 Jul 28 23:09 /dev/intel_me_fw_mapper
patrickx@ubuntu64bit:~$
```

[patrickx@30C3:~\$] cat 'Our Research Tools'

\$ Linux:

```
patrickx@ubuntu64bit: ~  
patrickx@ubuntu64bit:~$ ll /dev/intel*  
crw----- 1 root root 249, 0 Jul 28 23:09 /dev/intel_me_fw_mapper  
patrickx@ubuntu64bit:~$
```

```
patrickx@ubuntu64bit: ~  
patrickx@ubuntu64bit:~$ sudo xxd -l 0x50 /dev/intel_me_fw_mapper  
00000000: 244d 4f44 0000 0000 0700 0000 0300 0200 $MOD.....  
00000010: 0100 fe03 0000 0000 6814 0000 6814 0000 .....h...h...  
00000020: 0000 0001 b822 0100 5000 0000 e013 0000 .....P.....  
00000030: 4c4f 4144 4552 0000 0000 0000 0000 0000 LOADER.....  
00000040: 9b5a 7c88 2e4f a441 a6bd bf06 37bd a73b .Z|..O.A....7..  
patrickx@ubuntu64bit:~$
```

[patrickx@30C3:~\$] cat 'Our Research Tools'

\$ Linux:

```
patrickx@ubuntu64bit:~$ ll /dev/intel*
crw----- 1 root root 249, 0 Jul 28 23:09 /dev/intel_me_fw_mapper
patrickx@ubuntu64bit:~$
```

```
patrickx@ubuntu64bit:~$ sudo xxd -l 0x50 /dev/intel_me_fw_mapper
00000000: 244d 4f44 0000 0000 0700 0000 0300 0200  $MOD.....
00000010: 0100 fe03 0000 0000 6814 0000 6814 0000  .....h...h...
00000020: 0000 0001 b822 0100 5000 0000 e013 0000  ...."...P.....
00000030: 4c4f 4144 4552 0000 0000 0000 0000 0000  LOADER.....
00000040: 9b5a 7c88 2e4f a441 a6bd bf06 37bd a73b  .Z|..0.A....7..;
patrickx@ubuntu64bit:~$
```

```
patrickx@ubuntu64bit:~$
AMT Memory Monitor___
function keys: F5-> change address; F10->quit
arrow keys: left->column left; right->column right; up->line up; down->line down
other keys: page up->page up; page down->page down; home->1st line&1stcolumn; end->last line&lastcolumn
enter new address:
0088cb78: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cb94: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbcc: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbe8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cc04: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cc20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ad ff 01 00 ad ff 01
0088cc3c: 00 ad ff 01 00 ed ff 01 f0 e6 ff 01 00 00 00 00 c0 6c ff 01 c0 ac ff 01 80 a6 ff 01
0088cc58: 00 00 00 00 c0 6c ff 01 c0 6c ff 01 0d 00 01 00 00 00 00 00 00 00 00 00 02 00 00 00
0088cc74: 98 e4 7a 01 01 00 00 00 02 00 00 00 00 00 00 80 00 1c c0 14 a3 c3 00 00 00 00 00 00
0088cc90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 01 00 00 00
0088ccac: 00 00 00 00 00 00 00 00 00 00 00 00 00 02 6f 00 00 00 00 00 01 00 00 00 00 00 00 00
0088ccc8: 02 98 00 00 00 00 00 00 00 00 00 00 01 00 00 00 42 60 01 00 00 00 00 00 01 00 00 00
0088cce4: 01 00 00 00 42 62 01 00 00 00 00 00 01 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00
0088cd00: 00 00 00 00 01 00 00 00 00 44 00 01 01 00 00 00 01 00 00 00 02 00 00 00 00 00 00 00
0088cd1c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cd38: 00 00 00 00 48 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

[patrickx@30C3:~\$] cat 'Our Research Tools'

\$ Linux:

```
patrickx@ubuntu64bit:~$ ll /dev/intel*
crw----- 1 root root 249, 0 Jul 28 23:09 /dev/intel_me_fw_mapper
patrickx@ubuntu64bit:~$
```

```
patrickx@ubuntu64bit:~$ sudo xxd -l 0x50 /dev/intel_me_fw_mapper
00000000: 244d 4f44 0000 0000 0700 0000 0300 0200 $MOD.....
00000100: 0100 fe03 0000 0000 6814 0000 6814 0000 .....h...h...
00000200: 0000 0001 b822 0100 5000 0000 e013 0000 .....P.....
00000300: 4c4f 4144 4552 0000 0000 0000 0000 0000 LOADER.....
00000400: 9b5a 7c88 2e4f a441 a6bd bf06 37bd a73b .Z|..0.A....7..;
patrickx@ubuntu64bit:~$
```

```
patrickx@ubuntu64bit:~$
AMT Memory Monitor___
function keys: F5-> change address; F10->quit
arrow keys: left->column left; right->column right
other keys: page up->page up; page down->page down
enter new address:
0088cb78: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cb94: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbcc: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cbe8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cc04: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cc20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cc3c: 00 ad ff 01 00 ed ff 01 f0 e6 ff 01 00 00 00 00 00
0088cc58: 00 00 00 00 c0 6c ff 01 c0 6c ff 01 0d 00 00 00
0088cc74: 98 e4 7a 01 01 00 00 00 02 00 00 00 00 00 00 00
0088cc90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088ccac: 00 00 00 00 00 00 00 00 00 00 00 00 00 02 60 00
0088ccc8: 02 98 00 00 00 00 00 00 00 00 00 00 00 01 00 00
0088cce4: 01 00 00 00 42 62 01 00 00 00 00 00 00 01 00 00
0088cd00: 00 00 00 00 01 00 00 00 00 44 00 01 01 00 00 00
0088cd1c: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0088cd38: 00 00 00 00 48 00 00 00 00 00 00 00 00 00 00 00
```

```
patrickx@ubuntu64bit: ~/intelActiveManagementTechnology.project/subversion/patrickx.src/AMTBPHelper

=====
-* AMTBPHelper *-
=====

Press F1 and enter start and stop address of code to be disassembled.

*LOADER : 0x000000..0x0122b8, code: 0x000050..0x0013e0, entry: 0x000050
*KERNEL : 0x0122d0..0x28979c, code: 0x012320..0x05f068, entry: 0x031a10
*PMHWSEQ : 0x2897b0..0x28ddf0, code: 0x289800..0x28cad8, entry: 0x28a170
*QST : 0x28de00..0x2a79e8, code: 0x28de50..0x29b3f4, entry: 0x291b48
*OS : 0x2a7a00..0x88ee28, code: 0x2a7a50..0x5ada48, entry: 0x4ecc58
*ADMIN_CM : 0x88ee40..0x98ccf8, code: 0x88ee90..0x91a810, entry: 0x8b2994
*AMT_CM : 0x98cd10..0xaa35fc, code: 0x98cd60..0xa2089c, entry: 0x9bb964
*ASF_CM : 0xaa3610..0xab4dec, code: 0xaa3660..0xaad59c, entry: 0xaabc58
*INJ?CODE: 0x180000..0x250000, code: 0x??????..0x??????, entry: 0x180074

.. r0 00000000 .. ar0 00000000
.. r1 00000000 .. ar1 00000000
.. r2 00000000 .. ar2 00000000
.. r3 00000000 .. ar3 00000000
.. r4 00000000 .. ar4 00000000
.. r5 00000000 .. ar5 00000000
.. r6 00000000 .. ar6 00000000
.. r7 00000000 .. ar7 00000000
.. r8 00000000 .. ar8 00000000
.. r9 00000000 .. ar9 00000000
.. r10 00000000 .. ara 00000000
.. r11 00000000 .. arb 00000000
.. r12 00000000 .. arc 00000000
.. r13 00000000 .. ard 00000000
.. r14 00000000 .. are 00000000
.. r15 00000000 .. arf 00000000
.. r16 00000000 .. ar10 00000000
.. r17 00000000 .. ar11 00000000
.. r18 00000000 .. ar12 00000000
.. r19 00000000 .. ar13 00000000
.. r20 00000000 .. ar14 00000000
.. r21 00000000 .. ar15 00000000
.. r22 00000000 .. ar16 00000000
.. r23 00000000 .. ar17 00000000
.. r24 00000000 .. ar18 00000000
.. r25 00000000 .. ar19 00000000
.. r26 00000000 .. ar1a 00000000
.. r27 00000000 .. ar1b 00000000
.. r28 00000000 .. ar1c 00000000
.. r29 00000000 .. ar1d 00000000
.. r30 00000000 .. ar1e 00000000
.. r31 00000000 .. ar1f 00000000

BP: 0x0 StartAR: 0

AMTBPHelper started.
1RelLoad 2Set bp 3Del bp 4Continue 5Goto 6R all BP 7Fin Sess 8RelStart 9Chg StAR 10Quit
```

[patrickx@30C3:~\$] cat 'Code for Sending Packets'

\$ (un)plug network cable → one DHCP packet

```
patrickx@ubuntu64bit: ~/intelActiveManagementTechnology.project/subversion/patrickx.src/AMTBPHelper
AMTBPHelper - AMT firmware version: 3.2.1 - ARCTangent-A4 version: 0

./memory.dump:      file format binary

Disassembly of section .data:

004e2230 <.data+0x4e2230>:
4e2230: 00 7c bf 62 62bf7c00  mov     r21,0x188_cd34
4e2234: 34 cd 88 01
4e2238: 30 81 0a 08 080a8130  ld      r0,[r21,-208]
4e223c: 00 7d e0 57 57e07d00  sub.f   0,r0,0x1_000c
4e2240: 0c 00 01 00
4e2244: 02 7c 1f 60 601f7c02  mov.nz  r0,0x1_000d
4e2248: 0d 00 01 00
4e224c: 82 28 00 20 20002882  bnz     0x4e2394

4e2250: 00 7c ff 62 62ff7c00  mov     r23,0x16b_2a94
4e2254: 94 2a 6b 01
4e2258: 00 80 0b 08 080b8000  ld      r0,[r23]
4e225c: 14 00 00 08 08000014  ld      r0,[r0,20]
4e2260: 40 00 20 08 08200040  ld      r1,[r0,64]
4e2264: 00 2c cb 52 52cb2c00  sub     r22,r22,r22
4e2268: 00 2c 0b 60 600b2c00  mov     r0,r22
4e226c: 00 82 00 38 38008200  jl      [r1]
4e2270: 00 00 20 62 62200000  mov     r17,r0
4e2274: 28 81 8a 0a 0a8a8128  ld      r20,[r21,-216]
4e2278: 2c 81 0a 0a 0a0a812c  ld      r16,[r21,-212]
4e227c: 00 80 0b 08 080b8000  ld      r0,[r23]
4e2280: 14 00 00 08 08000014  ld      r0,[r0,20]
4e2284: 40 00 20 08 08200040  ld      r1,[r0,64]
4e2288: 00 a2 08 60 6008a200  mov     r0,r17
4e228c: 00 82 00 38 38008200  jl      [r1]
4e2290: 00 2c ab 61 61ab2c00  mov     r13,r22
4e2294: 00 21 ea 57 57ea2100  sub.f   0,r20,r16

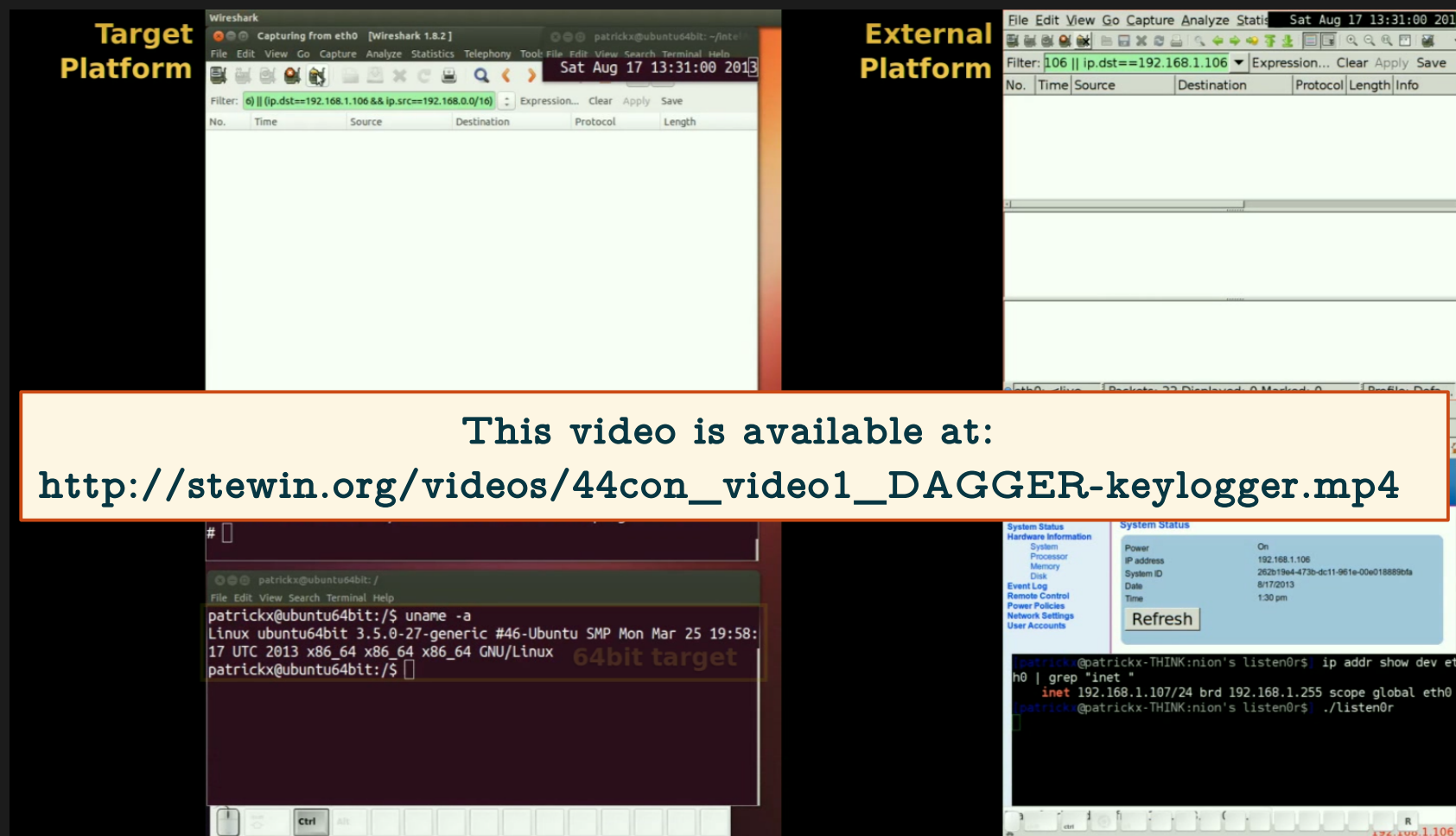
Line 6 9%                               BP: 0x0          StartA
Disassembled code from 0x4e2230 to 0x4e2300 loaded.
1Re|Load 2Set bp 3Del bp 4Continue 5Goto 6R all BP 7Fin Sess 8Re|Start 9Chg STAR 10Quit
```

```
patrickx@ubuntu64bit: ~/intelActiveManagementTechnology
AMT Memory Monitor__
function keys: F5-> change address; F10->quit
arrow keys: left->column left; right->column right; up->li
other keys: page up->page up; page down->page down; home->
_ = current address 00ffacf0 = _
00ffac80: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00ffac90: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00ffaca0: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00ffacb0: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00ffacc0: 24 4d 45 4d 00 ed ff 01 20 00 00 00 10 00 00 00
00ffacd0: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00fface0: 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f 0f
00ffacf0: 24 4d 45 4d 00 00 00 00 00 40 00 00 00 00 00 00
00ffad00: 56 81 07 00 00 80 00 00 00 00 00 00 00 00 00 00
00ffad10: ff ff ff ff ff ff 00 1c c0 14 a3 c3 08 00 45 10
00ffad20: 01 48 00 00 00 00 80 11 39 96 00 00 00 00 ff ff
00ffad30: ff ff 00 44 00 43 01 34 42 cb 01 01 06 00 aa 07
00ffad40: 25 0a 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffad50: 00 00 00 00 00 00 00 00 1c c0 14 a3 c3 00 00 00 00
00ffad60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffad70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffad80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffad90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffada0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffadb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffadc0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffadd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffade0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00ffadf0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

[patrickx@30C3:~\$]

Demo Video 1

Exfiltrating Password via OOB



```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```

AMT thread 1:
DAGGER*

keyboard buffer monitor

```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```

AMT thread 1:
DAGGER*

keyboard buffer monitor

space for new attack code

```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```

AMT thread 1:
DAGGER*

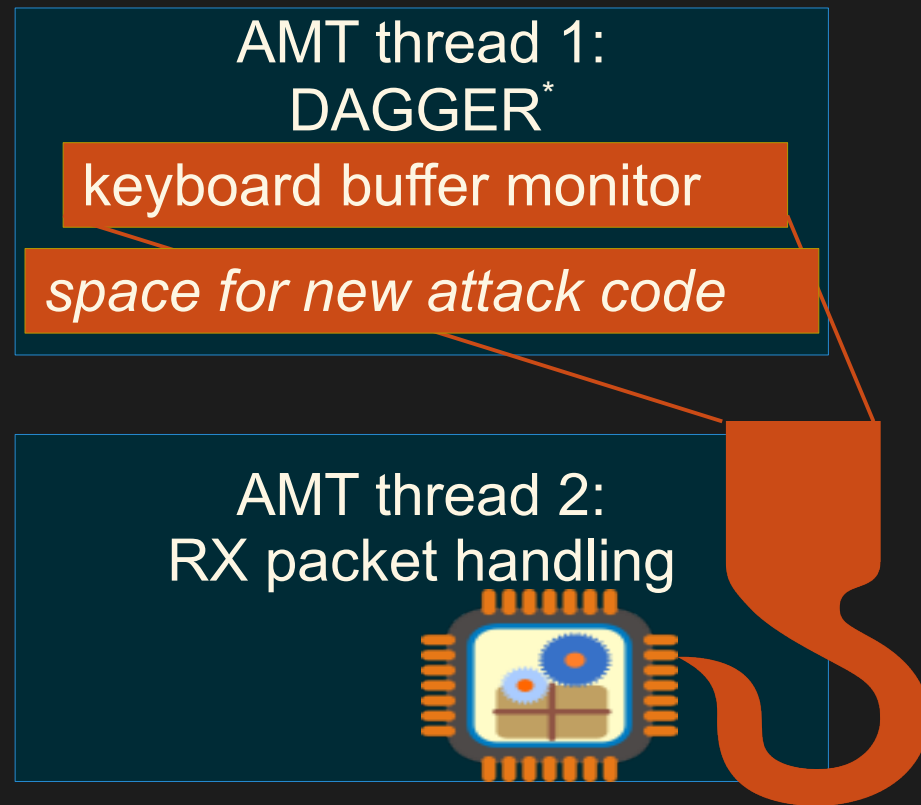
keyboard buffer monitor

space for new attack code

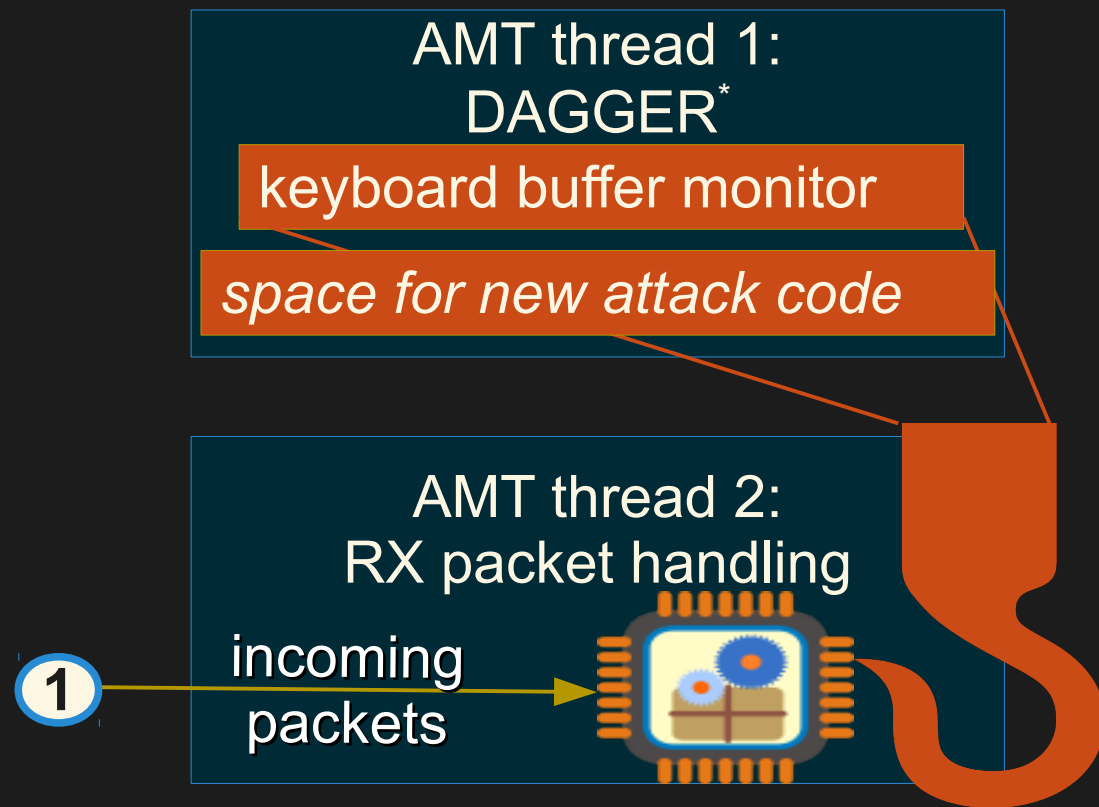
AMT thread 2:
RX packet handling



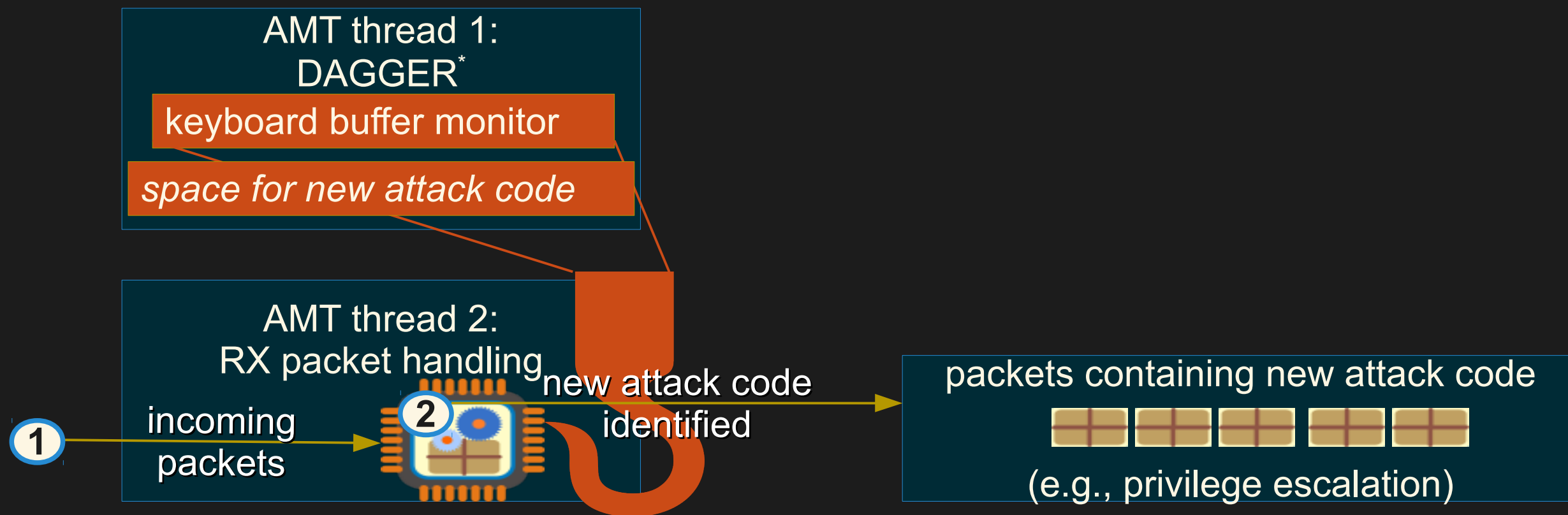
```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```



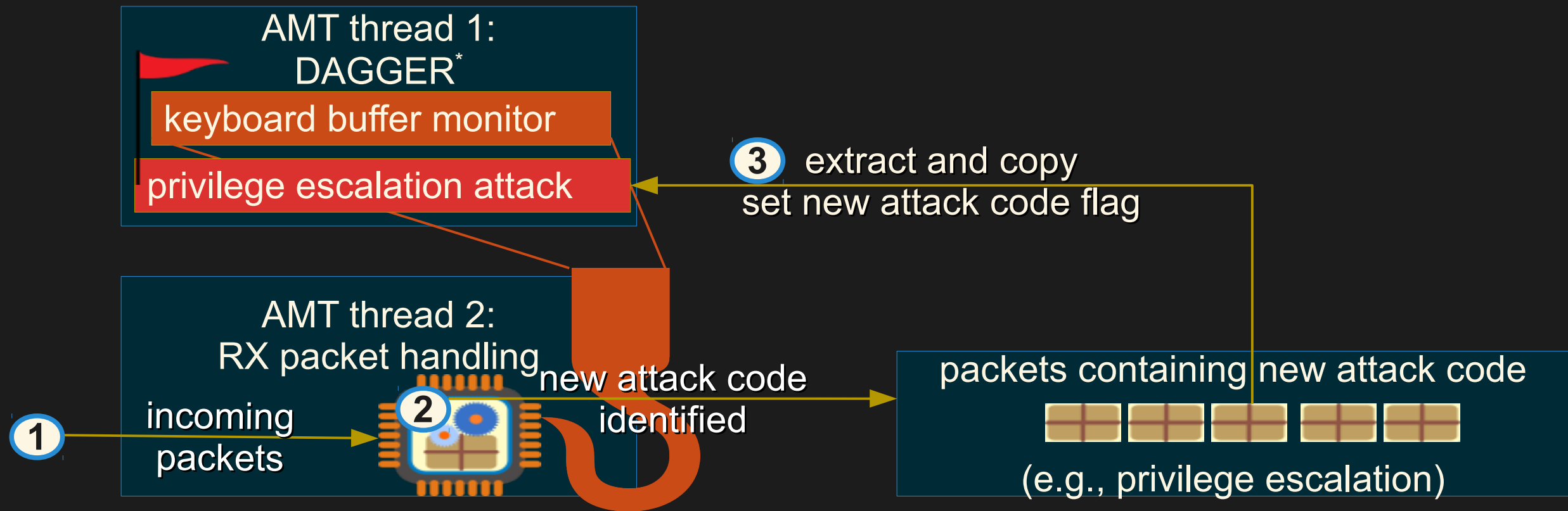
```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```



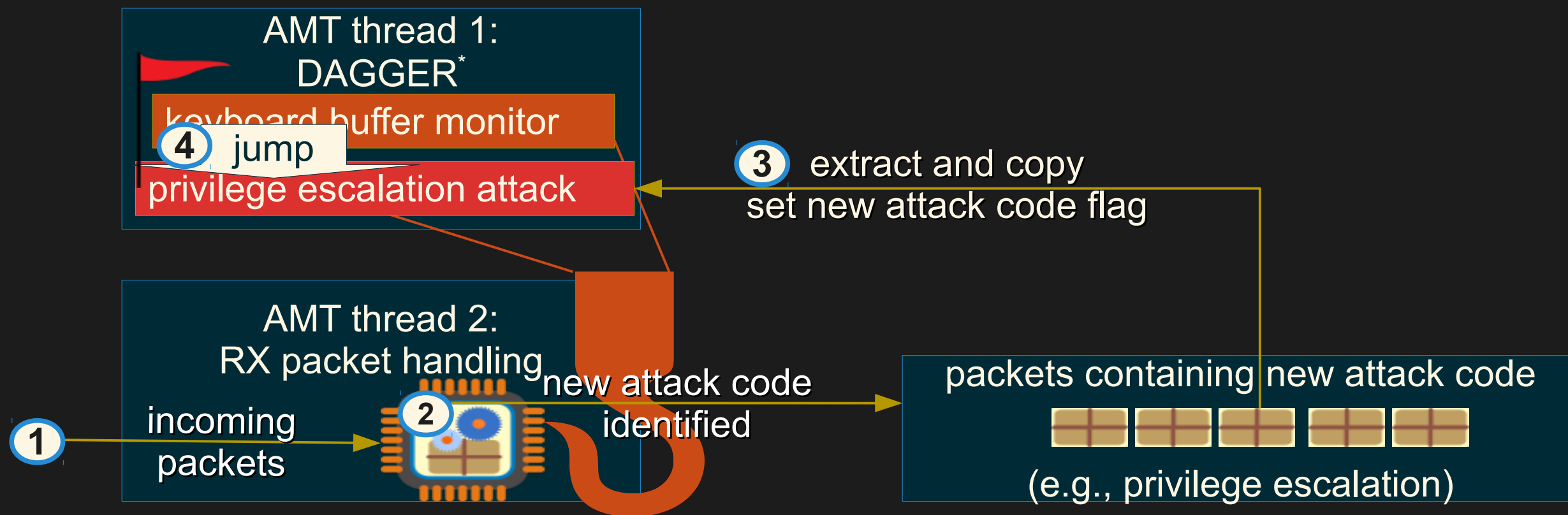
```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```



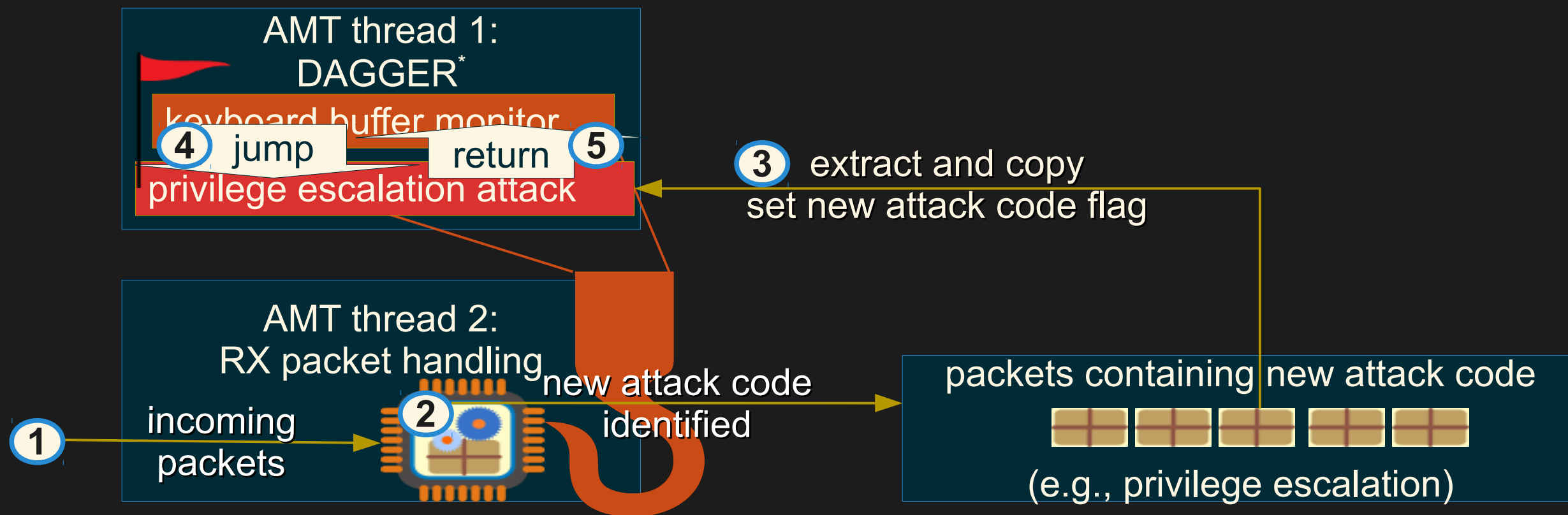
```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```



```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```

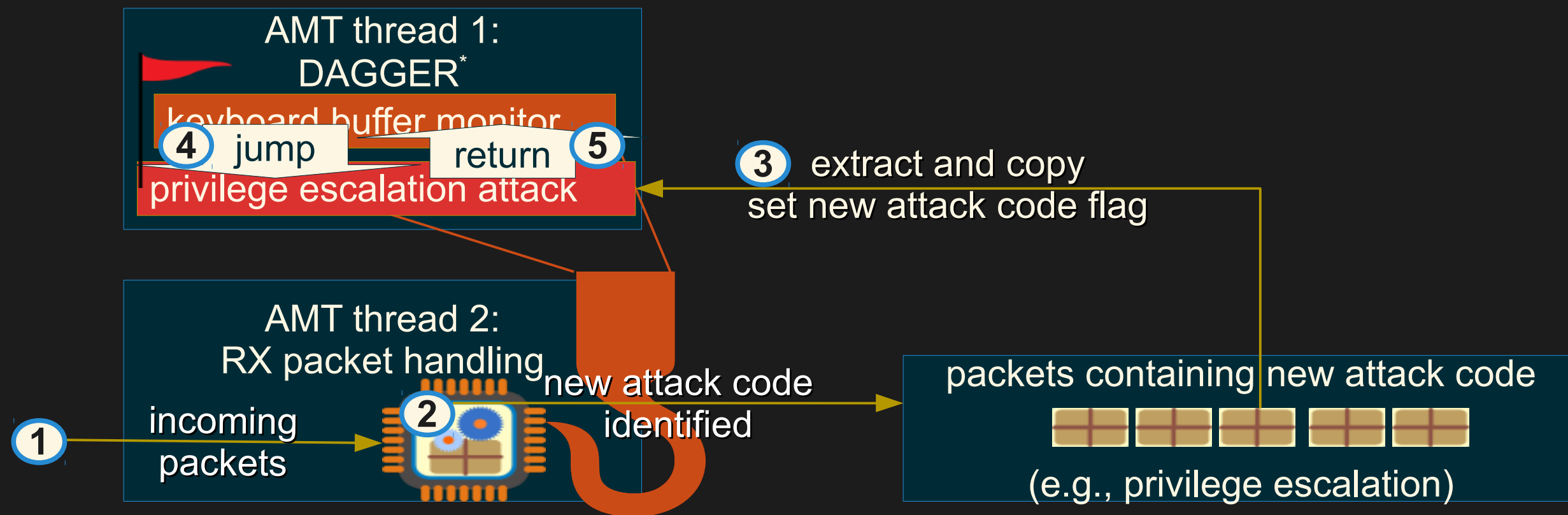


```
[patrickx@30C3:~$] cat 'DAGGER Updates'
```



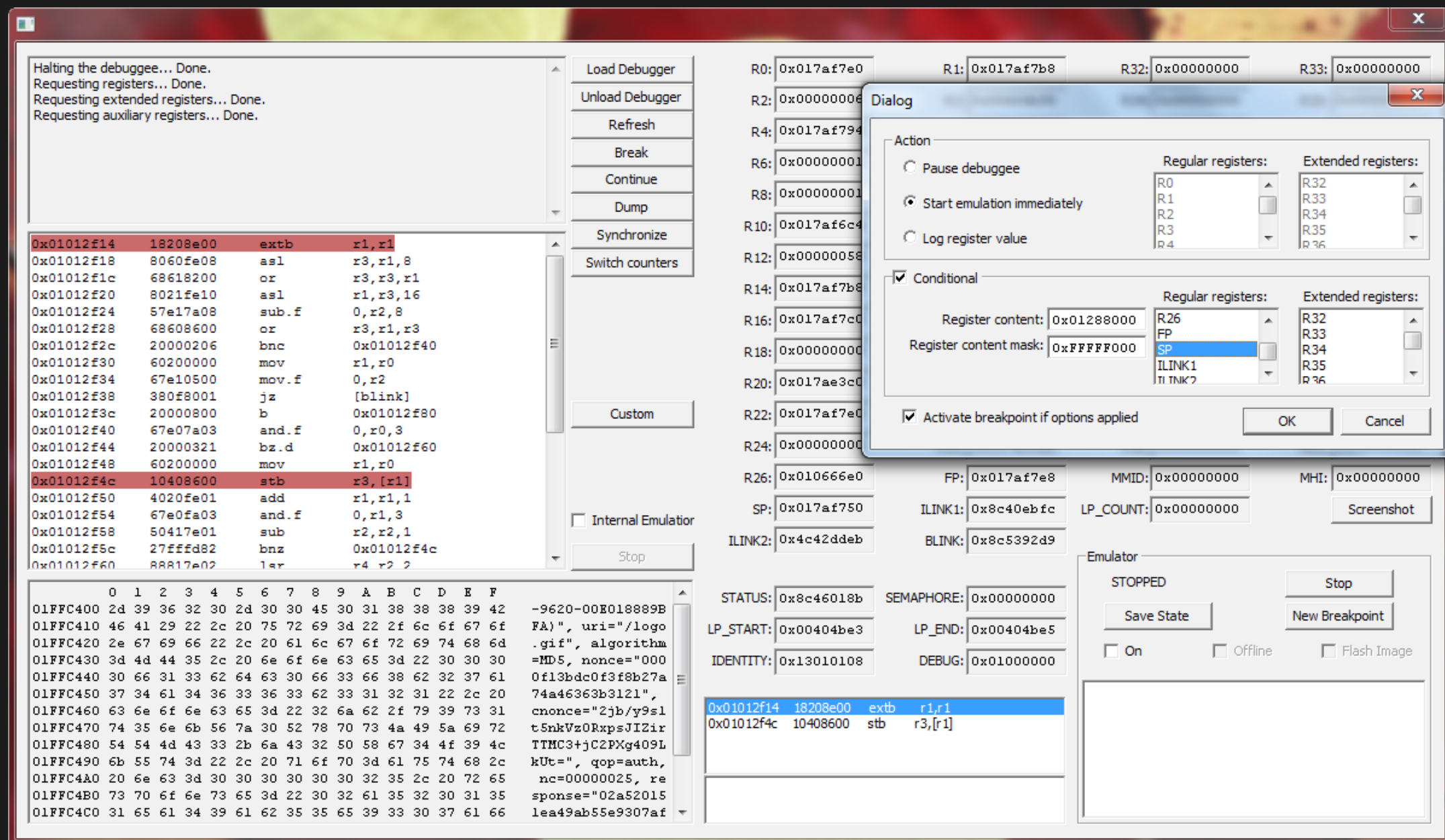
[patrickx@30C3:~\$] cat 'DAGGER Updates'

→ How to find code responsible for handling incoming network packets?



[patrickx@30C3:~\$] cat 'Our Research Tools'

\$ Windows



[patrickx@30C3:~\$] cat 'Trace Log'

\$ Windows

```
emutrace2.parsed.log
9172 0x014e4f30 624a2800 mov r18,r20 R18: 0x01736eba STATUS: 0x8c5393cd
9173 0x014e4f34 09ad81e8 ld r13,[fp,-24] R13: 0x01ffc400 STATUS: 0x8c5393ce
9174 0x014e4f38 0a6d81ec ld r19,[fp,-20] R19: 0x00000042 STATUS: 0x8c5393cf
9175 0x014e4f3c 67e82100 mov.f 0,r16 STATUS: 0x8c5393d0
9176 0x014e4f40 20000701 bz 0x014e4f7c STATUS: 0x8c5393d1
9177 0x014e4f7c 57e9fa64 sub.f 0,r19,100 STATUS: 0x6c5393e0
9178 0x014e4f80 20000405 bc 0x014e4fa4 STATUS: 0x6c5393e1
9179 0x014e4fa4 081f0000 ld r0,[0x16b_2a94] R0: 0x0106a2b8 STATUS: 0x6c5393ea
9180 0x014e4fac 08000004 ld r0,[r0,4] R0: 0x01068644 STATUS: 0x6c5393ec
9181 0x014e4fb0 08600008 ld r3,[r0,8] R3: 0x00404b98 STATUS: 0x6c5393ed
9182 0x014e4fb4 60092400 mov r0,r18 R0: 0x01736eba STATUS: 0x6c5393ee
9183 0x014e4fb8 60269a00 mov r1,r13 R1: 0x01ffc400 STATUS: 0x6c5393ef
9184 0x014e4fbc 6049a600 mov r2,r19 R2: 0x00000042 STATUS: 0x6c5393f0
9185 0x014e4fc0 38018200 jl [r3] BLINK: 0x6c5393f1 STATUS: 0x6c5393f1 {
9186 0x01012e60 381f0000 j 0x01180644 STATUS: 0x6c404b99
9187 0x01180644 538e7eb4 sub sp,sp,180 SP: 0x017af6d4 STATUS: 0x6c460192
9188 0x01180648 100e3e84 st blink,[sp,132] STATUS: 0x6c460193
9189 0x0118064c 100e7cac st 0x46_019a,[sp,172] STATUS: 0x6c460194
9190 0x01180654 2fff4380 bl 0x01180074 BLINK: 0x6c460196 STATUS: 0x6c460196 {
9242 0x01180658 28038b00 bl 0x011822b4 BLINK: 0x6c460197 STATUS: 0x6c460197 {
9256 0x0118065c 2fff5c00 bl 0x01180140 BLINK: 0x8c460198 STATUS: 0x8c460198 {
9304 0x01180660 0bee0084 ld blink,[sp,132] BLINK: 0x6c5393f1 STATUS: 0x6c460199
9305 0x01180664 0b8e0078 ld sp,[sp,120] SP: 0x017af788 STATUS: 0x6c46019a
9306 0x01180668 67810500 mov.f lp_count,r2 STATUS: 0x2c46019b
9307 0x0118066c 68800200 or r4,r0,r1 R4: 0x01ffeeba LP_COUNT: 0x00000042 STATUS: 0x2c46019c
9308 0x01180670 381f0000 j 0x01012e68 STATUS: 0x2c46019d
9309 0x01012e68 380f8101 jz.f [blink] STATUS: 0x2c404b9b
9310 0x01012e6c 67e27a03 and.f 0,r4,3 STATUS: 0x2c404b9c
9311 0x01012e70 88817e03 lsr r4,r2,3 R4: 0x00000008 STATUS: 0x2c404b9d
9312 0x01012e74 20001082 bnz 0x01012efc STATUS: 0x2c404b9e
9313 0x01012efc 5020fe01 sub r1,r1,1 R1: 0x01ffc3ff STATUS: 0x2c404bc0
9314 0x01012f00 50607e01 sub r3,r0,1 R3: 0x01736eb9 STATUS: 0x2c404bc1
9315 0x01012f04 30000100 lp 0x01012f10 STATUS: 0x2c404bc2 LP_START: 0x00404bc2 LP_END: 0x00404bc4
9316 0x01012f08 08809401 ldb.a r4,[r1,1] R1: 0x01ffc400 R4: 0x00000000 LP_COUNT: 0x00000041 STATUS: 0x2c404bc3
9317 0x01012f0c 11418801 stb.a r4,[r3,1] R3: 0x01736eba STATUS: 0x2c404bc2
9318 0x01012f08 08809401 ldb.a r4,[r1,1] R1: 0x01ffc401 R4: 0x00000019 LP_COUNT: 0x00000040 STATUS: 0x2c404bc3
9319 0x01012f0c 11418801 stb.a r4,[r3,1] R3: 0x01736ebb STATUS: 0x2c404bc2
9320 0x01012f08 08809401 ldb.a r4,[r1,1] R1: 0x01ffc402 R4: 0x000000d1 LP_COUNT: 0x0000003f STATUS: 0x2c404bc3
9321 0x01012f0c 11418801 stb.a r4,[r3,1] R3: 0x01736ebc STATUS: 0x2c404bc2
9322 0x01012f08 08809401 ldb.a r4,[r1,1] R1: 0x01ffc403 R4: 0x0000009f LP_COUNT: 0x0000003e STATUS: 0x2c404bc3
9323 0x01012f0c 11418801 stb.a r4,[r3,1] R3: 0x01736ebd STATUS: 0x2c404bc2
9324 0x01012f08 08809401 ldb.a r4,[r1,1] R1: 0x01ffc404 R4: 0x0000006f LP_COUNT: 0x0000003d STATUS: 0x2c404bc3
9325 0x01012f0c 11418801 stb.a r4,[r3,1] R3: 0x01736ebe STATUS: 0x2c404bc2
```

[patrickx@30C3:~\$] cat 'Trace Log'

mov	r0,r18	R0: 0x01736eba	STATUS: 0x6c5393ee
mov	r1,r13	R1: 0x01ffc400	STATUS: 0x6c5393ef
mov	r2,r19	R2: 0x00000042	STATUS: 0x6c5393f0
il	[r3]	BLINK: 0x6c5393f1	STATUS: 0x6c5393f1 {
i	0x01180644	STATUS: 0x6c404b99	
sub	sp,sp,180	SP: 0x017af6d4	STATUS: 0x6c460192
st	blink,[sp,132]	STATUS: 0x6c460193	
st	0x46_019a,[sp,172]	STATUS: 0x6c460194	
bl	0x01180074	BLINK: 0x6c460196	STATUS: 0x6c
bl	0x011822b4	BLINK: 0x6c460197	STATUS: 0x6c
bl	0x01180140	BLINK: 0x8c460198	STATUS: 0x8c
ld	blink,[sp,132]	BLINK: 0x6c5393f1	STATUS:
ld	sp,[sp,120]	SP: 0x017af788	STATUS: 0x6c460
mov.f	lp_count,r2	STATUS: 0x2c46019b	
or	r4,r0,r1	R4: 0x01ffeeba	LP_COUNT: 0x00000004
i	0x01012e68	STATUS: 0x2c46019d	
iz.f	[blink]	STATUS: 0x2c404b9b	
and.f	0,r4,3	STATUS: 0x2c404b9c	
lsr	r4,r2,3	R4: 0x00000008	STATUS: 0x2c404b9d
bnz	0x01012efc	STATUS: 0x2c404b9e	
sub	r1,r1,1	R1: 0x01ffc3ff	STATUS: 0x2c404bc0
sub	r3,r0,1	R3: 0x01736eb9	STATUS: 0x2c404bc1
lp	0x01012f10	STATUS: 0x2c404bc2	LP_START: 0
ldb.a	r4,[r1,1]	R1: 0x01ffc400	R4: 0x00000000 LP_
stb.a	r4,[r3,1]	R3: 0x01736eba	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc401	R4: 0x00000019 LP_
stb.a	r4,[r3,1]	R3: 0x01736ebb	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc402	R4: 0x000000d1 LP_
stb.a	r4,[r3,1]	R3: 0x01736ebc	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc403	R4: 0x0000009f LP_
stb.a	r4,[r3,1]	R3: 0x01736ebd	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc404	R4: 0x0000006f LP_

[patrickx@30C3:~\$] cat 'Trace Log'

```
mov      r0,r18      R0: 0x01736eba  STATUS: 0x6c5393ee
mov      r1,r13      R1: 0x01ffc400  STATUS: 0x6c5393ef
mov      r2,r19      R2: 0x00000042  STATUS: 0x6c5393f0
il       [r3]        BLINK: 0x6c5393f1  STATUS: 0x6c5393f1  {
    i      0x01180644      STATUS: 0x6c404b99
    sub    sp,sp,180      SP: 0x017af6d4  STATUS: 0x6c460192
    st     blink,[sp,132]  STATUS: 0x6c460193
    st     0x46_019a,[sp,172]  STATUS: 0x6c460194
    bl     0x01180074      BLINK: 0x6c460196  STATUS: 0x6c
    bl     0x011822b4      BLINK: 0x6c460197  STATUS: 0x6c
    bl     0x01180140      BLINK: 0x8c460198  STATUS: 0x8c
    ld     blink,[sp,132]  BLINK: 0x6c5393f1  STATUS:
    ld     sp,[sp,120]     SP: 0x017af788  STATUS: 0x6c460
mov.f    lp_count,r2     STATUS: 0x2c46019b
or       r4,r0,r1      R4: 0x01ffeeba  LP_COUNT: 0x00000004
    i      0x01012e68      STATUS: 0x2c46019d
    iz.f   [blink]        STATUS: 0x2c404b9b
    and.f  0,r4,3         STATUS: 0x2c404b9c
    lsr    r4,r2,3        R4: 0x00000008  STATUS: 0x2c404b9d
    bnz    0x01012efc     STATUS: 0x2c404b9e
    sub    r1,r1,1        R1: 0x01ffc3ff  STATUS: 0x2c404bc0
    sub    r3,r0,1        R3: 0x01736eb9  STATUS: 0x2c404bc1
    lp     0x01012f10     STATUS: 0x2c404bc2  LP_START: 0
    ldb.a  r4,[r1,1]      R1: 0x01ffc400  R4: 0x00000000  LP_
    stb.a  r4,[r3,1]      R3: 0x01736eba  STATUS: 0x2c404bc2
    ldb.a  r4,[r1,1]      R1: 0x01ffc401  R4: 0x00000019  LP_
    stb.a  r4,[r3,1]      R3: 0x01736ebb  STATUS: 0x2c404bc2
    ldb.a  r4,[r1,1]      R1: 0x01ffc402  R4: 0x000000d1  LP_
    stb.a  r4,[r3,1]      R3: 0x01736ebc  STATUS: 0x2c404bc2
    ldb.a  r4,[r1,1]      R1: 0x01ffc403  R4: 0x0000009f  LP_
    stb.a  r4,[r3,1]      R3: 0x01736ebd  STATUS: 0x2c404bc2
    ldb.a  r4,[r1,1]      R1: 0x01ffc404  R4: 0x0000006f  LP_
```

}memcpy call

[patrickx@30C3:~\$] cat 'Trace Log'

mov	r0,r18	R0: 0x01736eba	STATUS: 0x6c5393ee
mov	r1,r13	R1: 0x01ffc400	STATUS: 0x6c5393ef
mov	r2,r19	R2: 0x00000042	STATUS: 0x6c5393f0
<u>il</u>	[r3]	BLINK: 0x6c5393f1	STATUS: 0x6c5393f1 {
<u>i</u>	0x01180644	STATUS: 0x6c404b99	
sub	sp,sp,180	SP: 0x017af6d4	STATUS: 0x6c460192
st	blink,[sp,132]	STATUS: 0x6c460193	
st	0x46_019a,[sp,172]	STATUS: 0x6c460194	
<u>bl</u>	0x01180074	BLINK: 0x6c460196	STATUS: 0x6c
<u>bl</u>	0x011822b4	BLINK: 0x6c460197	STATUS: 0x6c
<u>bl</u>	0x01180140	BLINK: 0x8c460198	STATUS: 0x8c
ld	blink,[sp,132]	BLINK: 0x6c5393f1	STATUS:
ld	sp,[sp,120]	SP: 0x017af788	STATUS: 0x6c460
mov.f	lp_count,r2	STATUS: 0x2c46019b	
or	r4,r0,r1	R4: 0x01ffeeba	LP_COUNT: 0x00000004
<u>i</u>	0x01012e68	STATUS: 0x2c46019d	
<u>iz.f</u>	[blink]	STATUS: 0x2c404b9b	
and.f	0,r4,3	STATUS: 0x2c404b9c	
lsr	r4,r2,3	R4: 0x00000008	STATUS: 0x2c404b9d
<u>bnz</u>	0x01012efc	STATUS: 0x2c404b9e	
sub	r1,r1,1	R1: 0x01ffc3ff	STATUS: 0x2c404bc0
sub	r3,r0,1	R3: 0x01736eb9	STATUS: 0x2c404bc1
lp	0x01012f10	STATUS: 0x2c404bc2	LP_START: 0
<u>ldb.a</u>	r4,[r1,1]	R1: 0x01ffc400	R4: 0x00000000 LP_
<u>stb.a</u>	r4,[r3,1]	R3: 0x01736eba	STATUS: 0x2c404bc2
<u>ldb.a</u>	r4,[r1,1]	R1: 0x01ffc401	R4: 0x00000019 LP_
<u>stb.a</u>	r4,[r3,1]	R3: 0x01736ebb	STATUS: 0x2c404bc2
<u>ldb.a</u>	r4,[r1,1]	R1: 0x01ffc402	R4: 0x000000d1 LP_
<u>stb.a</u>	r4,[r3,1]	R3: 0x01736ebc	STATUS: 0x2c404bc2
<u>ldb.a</u>	r4,[r1,1]	R1: 0x01ffc403	R4: 0x0000009f LP_
<u>stb.a</u>	r4,[r3,1]	R3: 0x01736ebd	STATUS: 0x2c404bc2
<u>ldb.a</u>	r4,[r1,1]	R1: 0x01ffc404	R4: 0x0000006f LP_

}memcpy parameter
}memcpy call

[patrickx@30C3:~\$] cat 'Trace Log'

```
mov      r0,r18      R0: 0x01736eba  STATUS: 0x6c5393ee
mov      r1,r13      R1: 0x01ffc400  STATUS: 0x6c5393ef
mov      r2,r19      R2: 0x00000042  STATUS: 0x6c5393f0
il       [r3]        BLINK: 0x6c5393f1  STATUS: 0x6c5393f1  {
  i       0x01180644      STATUS: 0x6c404b99
  sub     sp,sp,180      SP: 0x017af6d4  STATUS: 0x6c460192
  st      blink,[sp,132]  STATUS: 0x6c460193
  st      0x46_019a,[sp,172]  STATUS: 0x6c460194
  bl      0x01180074      BLINK: 0x6c460196  STATUS: 0x6c
  bl      0x011822b4      BLINK: 0x6c460197  STATUS: 0x6c
  bl      0x01180140      BLINK: 0x8c460198  STATUS: 0x8c
  ld      blink,[sp,132]  BLINK: 0x6c5393f1  STATUS:
  ld      sp,[sp,120]     SP: 0x017af788  STATUS: 0x6c460
mov.f    lp_count,r2     STATUS: 0x2c46019b
or       r4,r0,r1      R4: 0x01ffeeba  LP_COUNT: 0x00000004
  i       0x01012e68      STATUS: 0x2c46019d
  iz.f    [blink]        STATUS: 0x2c404b9b
  and.f   0,r4,3         STATUS: 0x2c404b9c
  lsr     r4,r2,3        R4: 0x00000008  STATUS: 0x2c404b9d
  bnz     0x01012efc     STATUS: 0x2c404b9e
  sub     r1,r1,1        R1: 0x01ffc3ff  STATUS: 0x2c404bc0
  sub     r3,r0,1        R3: 0x01736eb9  STATUS: 0x2c404bc1
  lp      0x01012f10     STATUS: 0x2c404bc2  LP_START: 0
  ldb.a   r4,[r1,1]     R1: 0x01ffc400  R4: 0x00000000  LP_
  stb.a   r4,[r3,1]     R3: 0x01736eba  STATUS: 0x2c404bc2
  ldb.a   r4,[r1,1]     R1: 0x01ffc401  R4: 0x000000019  LP_
  stb.a   r4,[r3,1]     R3: 0x01736ebb  STATUS: 0x2c404bc2
  ldb.a   r4,[r1,1]     R1: 0x01ffc402  R4: 0x0000000d1  LP_
  stb.a   r4,[r3,1]     R3: 0x01736ebc  STATUS: 0x2c404bc2
  ldb.a   r4,[r1,1]     R1: 0x01ffc403  R4: 0x00000009f  LP_
  stb.a   r4,[r3,1]     R3: 0x01736ebd  STATUS: 0x2c404bc2
  ldb.a   r4,[r1,1]     R1: 0x01ffc404  R4: 0x00000006f  LP_
```

}memcpy parameter
}memcpy call

*our main hook
is also traced into*

[patrickx@30C3:~\$] cat 'Trace Log'

```
mov      r0,r18      R0: 0x01736eba  STATUS: 0x6c5393ee
mov      r1,r13      R1: 0x01ffc400  STATUS: 0x6c5393ef
mov      r2,r19      R2: 0x00000042  STATUS: 0x6c5393f0
il       [r3]        BLINK: 0x6c5393f1  STATUS: 0x6c5393f1  {
i        0x01180644    STATUS: 0x6c404b99
sub      sp,sp,180    SP: 0x017af6d4  STATUS: 0x6c460192
st       blink,[sp,132]  STATUS: 0x6c460193
st       0x46_019a,[sp,172]  STATUS: 0x6c460194
bl       0x01180074    BLINK: 0x6c460196  STATUS: 0x6c
bl       0x011822b4    BLINK: 0x6c460197  STATUS: 0x6c
bl       0x01180140    BLINK: 0x8c460198  STATUS: 0x8c
ld       blink,[sp,132]  BLINK: 0x6c5393f1  STATUS:
ld       sp,[sp,120]    SP: 0x017af788  STATUS: 0x6c460
mov.f    lp_count,r2    STATUS: 0x2c46019b
or       r4,r0,r1      R4: 0x01ffeeba  LP_COUNT: 0x00000004
i        0x01012e68    STATUS: 0x2c46019d
iz.f     [blink]       STATUS: 0x2c404b9b
and.f    0,r4,3        STATUS: 0x2c404b9c
lsr      r4,r2,3       R4: 0x00000008  STATUS: 0x2c404b9d
bnz      0x01012efc    STATUS: 0x2c404b9e
sub      r1,r1,1       R1: 0x01ffc3ff  STATUS: 0x2c404bc0
sub      r3,r0,1       R3: 0x01736eb9  STATUS: 0x2c404bc1
lp       0x01012f10    STATUS: 0x2c404bc2  LP_START: 0
ldb.a    r4,[r1,1]     R1: 0x01ffc400  R4: 0x00000000 LP
stb.a    r4,[r3,1]     R3: 0x01736eba  STATUS: 0x2c404bc2
ldb.a    r4,[r1,1]     R1: 0x01ffc401  R4: 0x00000019 LP
stb.a    r4,[r3,1]     R3: 0x01736ebb  STATUS: 0x2c404bc2
ldb.a    r4,[r1,1]     R1: 0x01ffc402  R4: 0x000000d1 LP
stb.a    r4,[r3,1]     R3: 0x01736ebc  STATUS: 0x2c404bc2
ldb.a    r4,[r1,1]     R1: 0x01ffc403  R4: 0x0000009f LP
stb.a    r4,[r3,1]     R3: 0x01736ebd  STATUS: 0x2c404bc2
ldb.a    r4,[r1,1]     R1: 0x01ffc404  R4: 0x0000006f LP
```

}memcpy parameter
}memcpy call

our main hook
is also traced into

first bytes of
an incoming
packet

[patrickx@30C3:~\$] cat 'Trace Log'

mov	r0,r18	R0: 0x01736eba	STATUS: 0x6c5393ee
mov	r1,r13	R1: 0x01ffc400	STATUS: 0x6c5393ef
mov	r2,r10	R2: 0x01736eb9	STATUS: 0x6c5393f0
il	[r3]	BLINK: 0x6c5393f1	STATUS: 0x6c5393f1 {
i	0x01180644	STATUS: 0x6c404b99	
sub	sp,sp,180	SP: 0x017af6d4	STATUS: 0x6c460192
st	blink,[sp,132]	STATUS: 0x6c460193	
st	0x46_019a,[sp,172]	STATUS: 0x6c460194	
bl	0x01180074	BLINK: 0x6c460196	STATUS: 0x6c460196
bl	0x011822b4	BLINK: 0x6c460197	STATUS: 0x6c460197
bl	0x01180140	BLINK: 0x8c460198	STATUS: 0x8c460198
ld	blink,[sp,132]	BLINK: 0x6c5393f1	STATUS: 0x6c5393f1
ld	sp,[sp,120]	SP: 0x017af788	STATUS: 0x6c460199
mov.f	lp_count,r2	STATUS: 0x2c46019b	
or	r4,r0,r1	R4: 0x01ffeeba	LP_COUNT: 0x00000004
i	0x01012e68	STATUS: 0x2c46019d	
iz.f	[blink]	STATUS: 0x2c404b9b	
and.f	0,r4,3	STATUS: 0x2c404b9c	
lsr	r4,r2,3	R4: 0x00000008	STATUS: 0x2c404b9d
bnz	0x01012efc	STATUS: 0x2c404b9e	
sub	r1,r1,1	R1: 0x01ffc3ff	STATUS: 0x2c404bc0
sub	r3,r0,1	R3: 0x01736eb9	STATUS: 0x2c404bc1
lp	0x01012f10	STATUS: 0x2c404bc2	LP_START: 0
ldb.a	r4,[r1,1]	R1: 0x01ffc400	R4: 0x00000000 LP_
stb.a	r4,[r3,1]	R3: 0x01736eba	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc401	R4: 0x00000019 LP_
stb.a	r4,[r3,1]	R3: 0x01736ebb	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc402	R4: 0x000000d1 LP_
stb.a	r4,[r3,1]	R3: 0x01736ebc	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc403	R4: 0x0000009f LP_
stb.a	r4,[r3,1]	R3: 0x01736ebd	STATUS: 0x2c404bc2
ldb.a	r4,[r1,1]	R1: 0x01ffc404	R4: 0x0000006f LP_

hook to intercept incoming packets

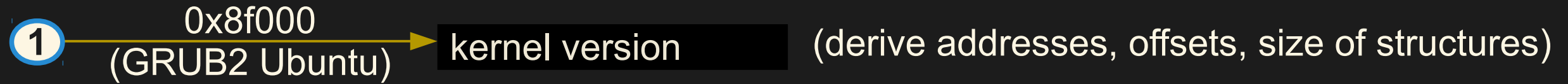
}memcpy parameter
}memcpy call

our main hook
is also traced into

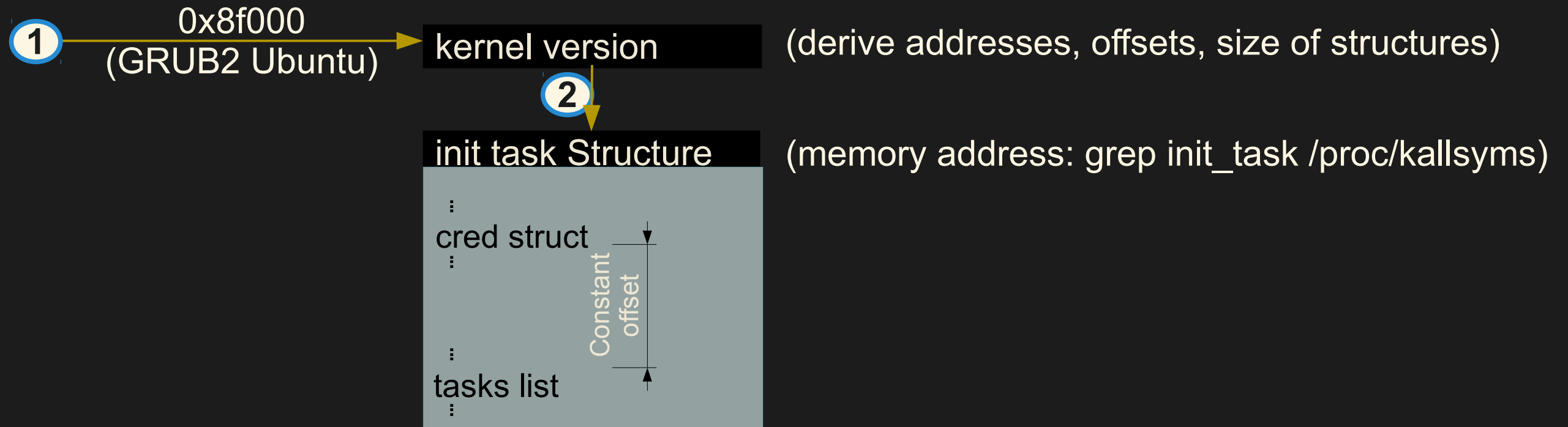
first bytes of
an incoming
packet

```
[patrickx@30C3:~$] cat 'Privilege Escalation'
```

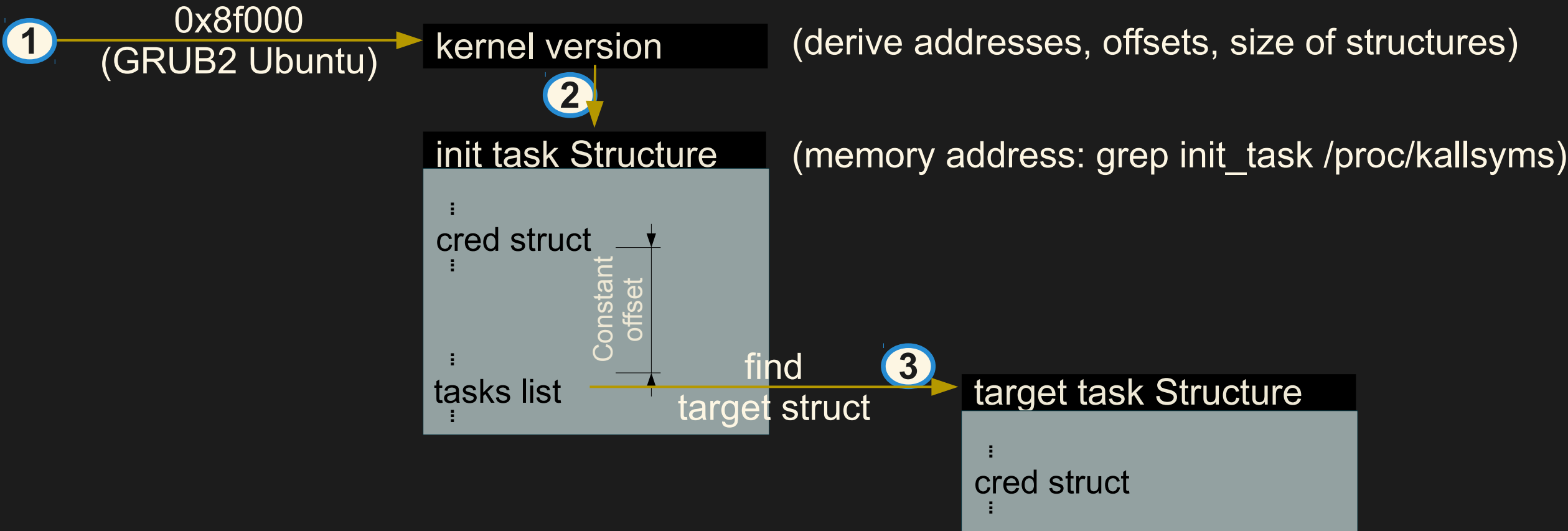
```
[patrickx@30C3:~$] cat 'Privilege Escalation'
```



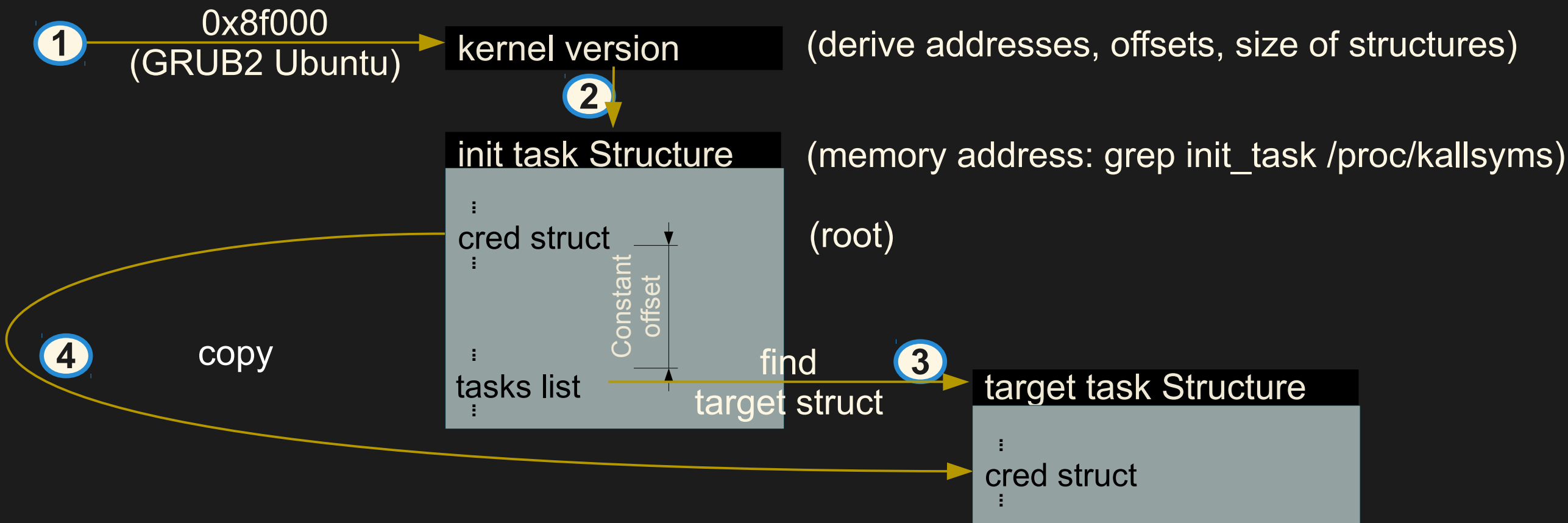
[patrickx@30C3:~\$] cat 'Privilege Escalation'



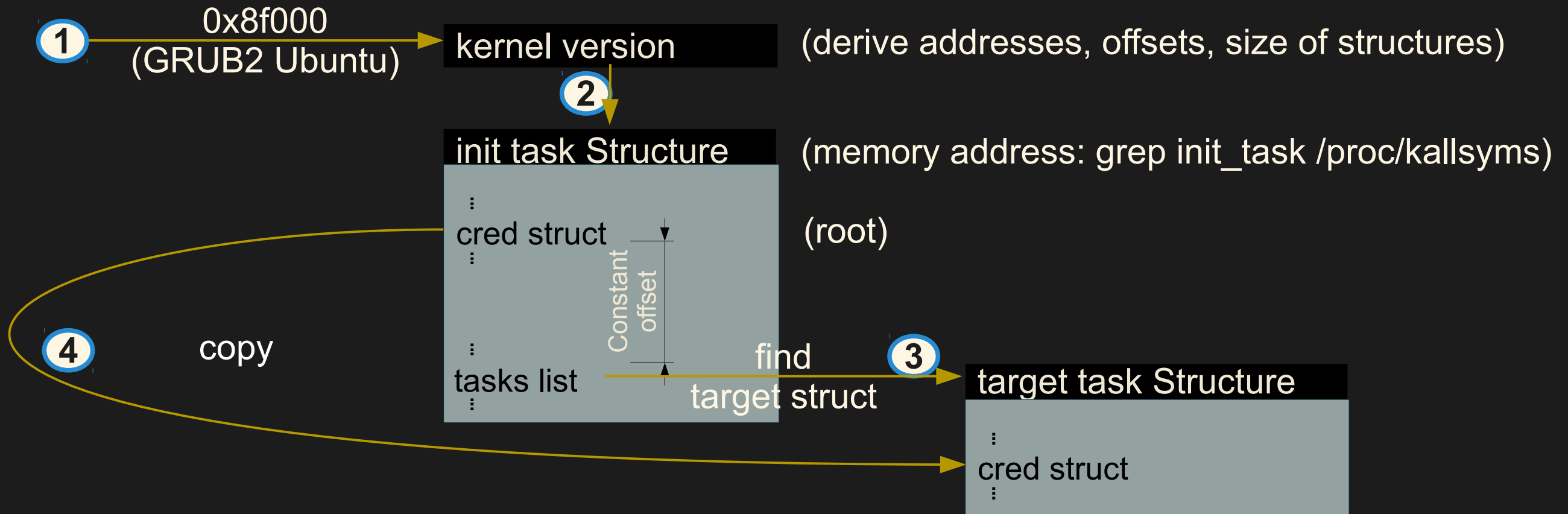
[patrickx@30C3:~\$] cat 'Privilege Escalation'



[patrickx@30C3:~\$] cat 'Privilege Escalation'



[patrickx@30C3:~\$] cat 'Privilege Escalation'



\$ Binary: DMA_poc_remote_privilege_escalation.arc4.elf

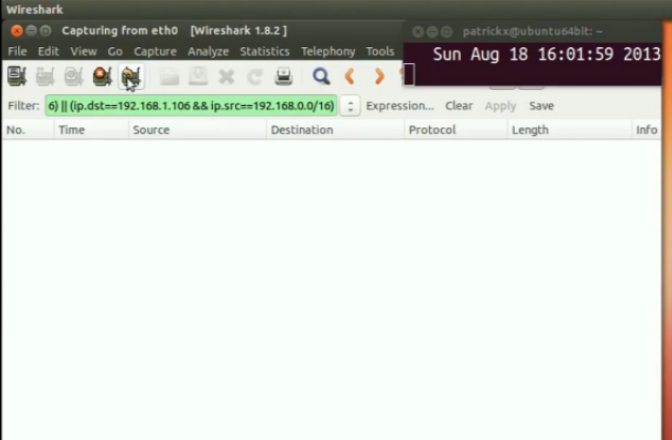
\$ Sent via hping3

man hping3 "[...] send (almost) arbitrary TCP/IP packets to network hosts [...]"

[patrickx@30C3:~\$]

Demo Video 2

Privilege Escalation via OOB

Target Platform


Wireshark 1.8.2 interface showing a capture from eth0. The filter is set to `(ip.dst==192.168.1.106 && ip.src==192.168.0.0/16)`. The packet list shows a single packet from 192.168.1.106 to 255.255.255.255 via UDP.

External Platform


Terminal output from the external platform showing a netcat listener and a netcat client connection:

```
[patrickx@patrickx-THINK:nion's listen0r$] ./listen0r
r
r

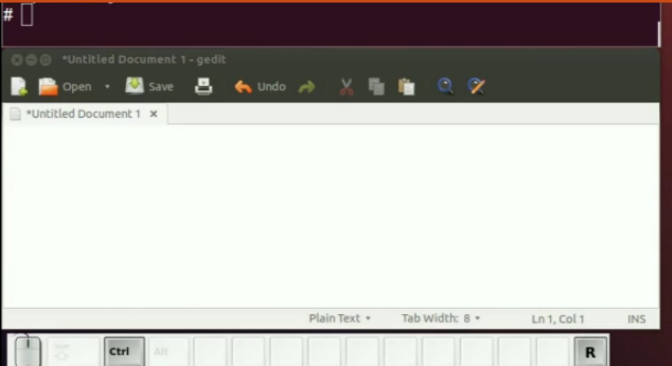
[patrickx@patrickx-THINK:net_convert$] sudo hping3 -i u10000 -c 76 -I eth0 -S 192.168.1.106 -p 16992 -d 1402 -E ./DMA poc remote privilege escalation.arc4.e
```

System status window showing hardware information:

System Status	
Hardware Information	
System	On
Processor	192.168.1.106
Memory	262b19e4-473b-dc11-961e-00e018889bfa
Disk	8/18/2013
Event Log	4:00 pm
Remote Control	

Terminal output showing an SSH connection to the target platform:

```
[patrickx@patrickx-THINK:~$] ssh 192.168.1.106
```



Gedit editor window showing an empty file named "Untitled Document 1".

This video is available at:

http://stewin.org/videos/44con_video2_DAGGER-privilege-escalation.mp4



Covert Network Channel

[patrickx@30C3:~\$] cat "Trick Non-host Monitors"

\$ JitterBug based, see "Keyboards and Covert Channels" [Sha06]:

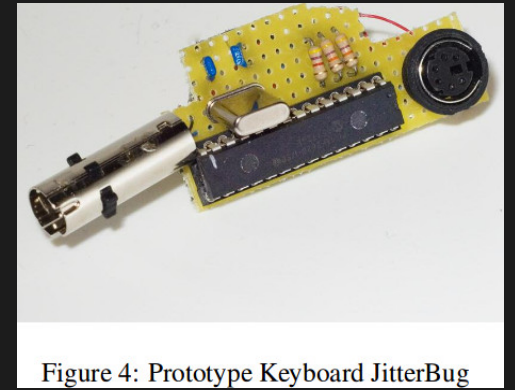
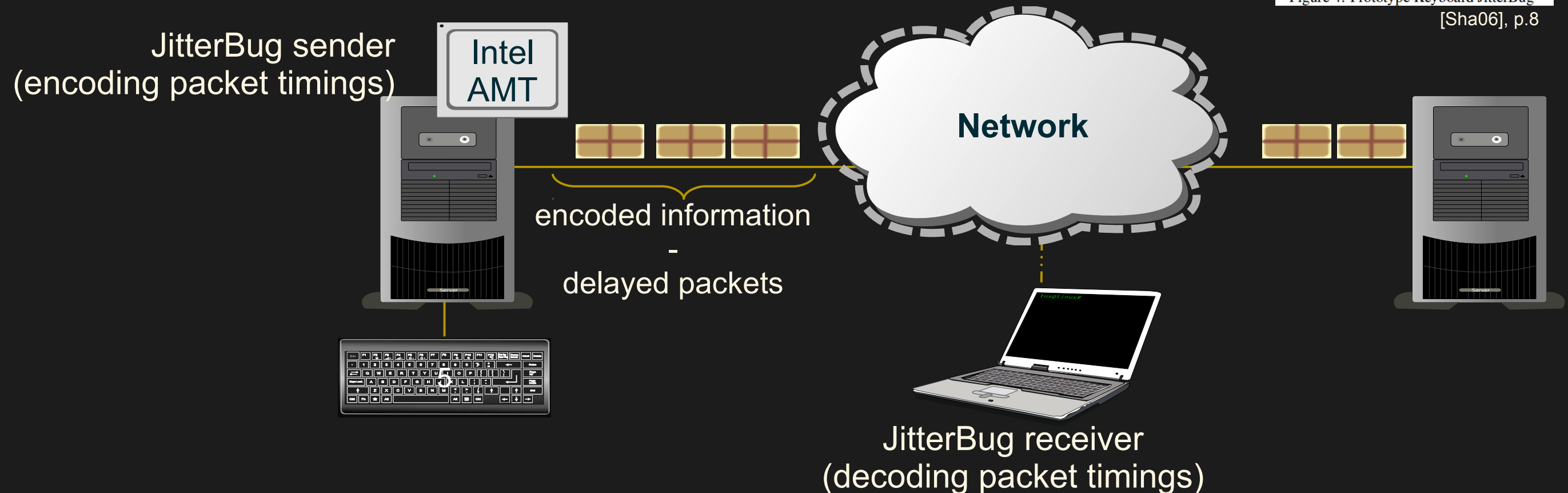


Figure 4: Prototype Keyboard JitterBug
[Sha06], p.8



[patrickx@30C3:~\$] cat 'More ME Features'

\$ Outgoing packet interception ✓

\$ Measure time!

AMT peripheral (timer) access:

```
lr r0,[0x8011]
```

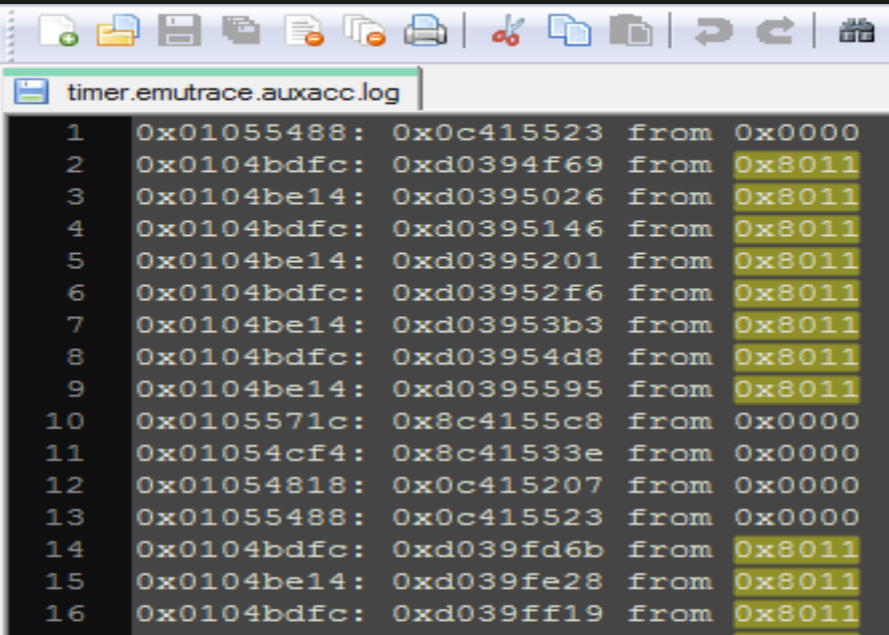
→ Read timer register:

resolution
~ 996500 Hz

\$ Packets
to delay

Wireshark log of an
AMT TCP session

No.	Time	Protocol Info
1	0.000000	TCP amt-soap-http > 7512 [SYN, ACK]
2	0.001725	TCP amt-soap-http > 7512 [ACK]
3	0.002169	TCP amt-soap-http > 7512 [ACK]
4	0.207100	TCP amt-soap-http > 7512 [PSH,ACK]
5	0.209416	TCP amt-soap-http > 7512 [PSH,ACK]
6	0.214836	TCP amt-soap-http > 7512 [PSH,ACK]
7	13.125414	TCP amt-soap-http > 7512 [FIN,PSH,ACK]



```
[patrickx@30C3:~$] cat 'Execution Stages'
```

No.	Description	Duration	Overhead
1.	Find keyboard buffer	100-110 ms	AMT irresponsive
2.	Log sensitive information (e.g., detect keystrokes following a login name)	determined by user input	insignificant
3.	Leak sensitive information (encode into legitimate packet delays)	unlimited, continuous replay	low, but detectable

[patrickx@30C3:~\$]

Demo Video 3

JitterBug

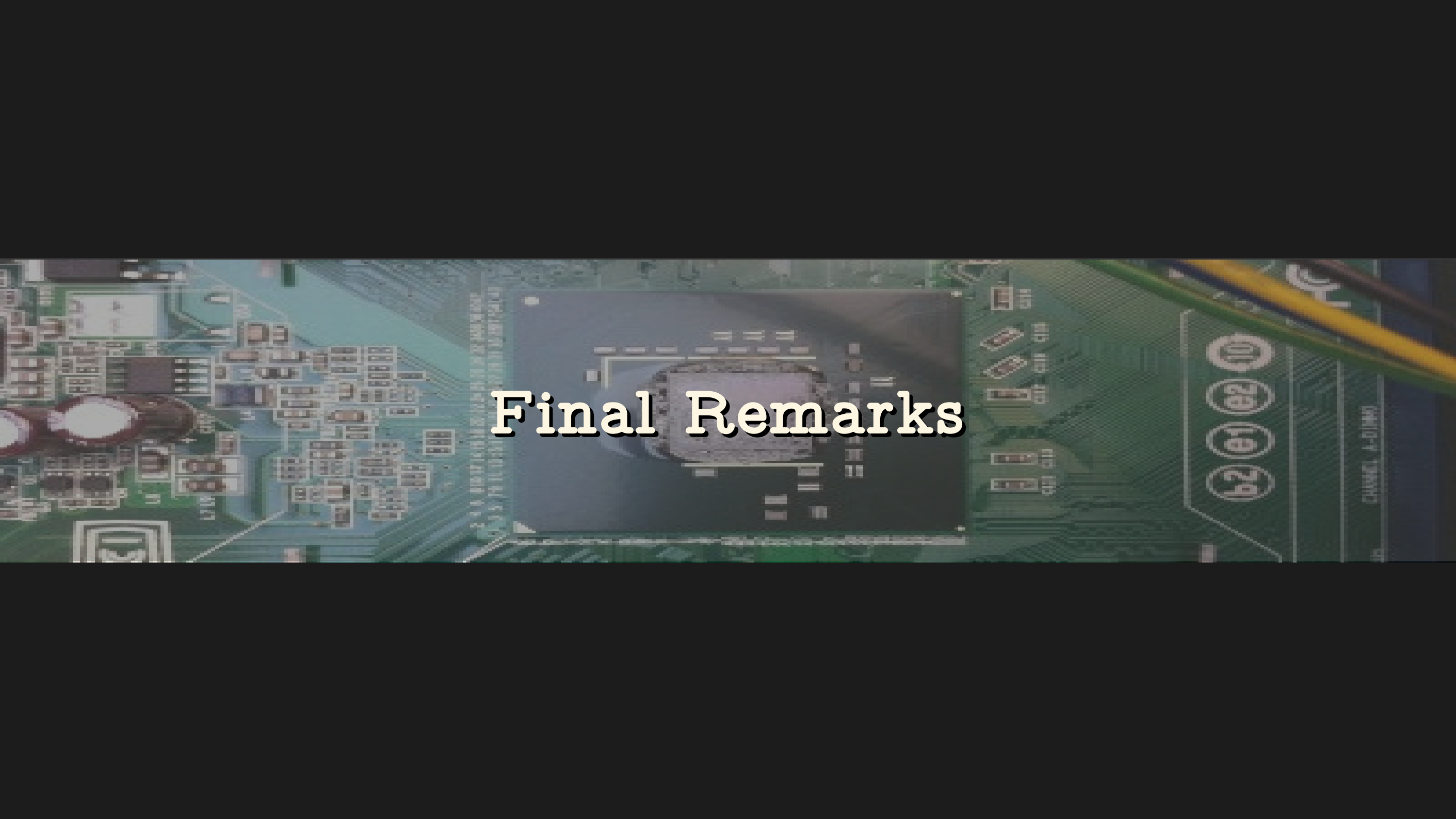
The screenshot displays the JitterBug malware interface. At the top, a pink banner reads "Attac". Below it, a console window shows a table of statistics:

Nr.	Bit	Time	delta	Full	delta	Err
1	0		6513	6513		
2	x		11197	51197		
3	x		12223	12223		
4	1		16107	16107		
5	0		6733	46733		
6	x		9893	89893		
7	1		14074	34074		
8	0		3525	3923525		

A speech bubble points to this table: "Some statistics in a boring console". To the right, a "Remote administrator" window shows an "Intel Active Management Technology" login screen with a "Log On" button. Below it, an "Authentication Required" dialog box is visible. A speech bubble points to the console output: "Decoded bit".

A yellow banner in the center states: "This video is available at: http://stewin.org/videos/44con_video3-DAGGER-jutterbug.avi".

The bottom section of the interface features a circular gauge on the left, a scatter plot on the right, and a "Frame start sequence" box displaying "10100111". A speech bubble points to the gauge: "Leaked data". Another speech bubble points to the "Frame start sequence" box: "Frame start sequence".



Final Remarks

```
[patrickx@30C3:~$] cat 'Countermeasures'
```

\$ Virtualization Technology for
Directed I/O (I/OMMU, [Abr06])

\$ Attacks: [San10], [Woj09], [Woj11a],
[Woj11b]

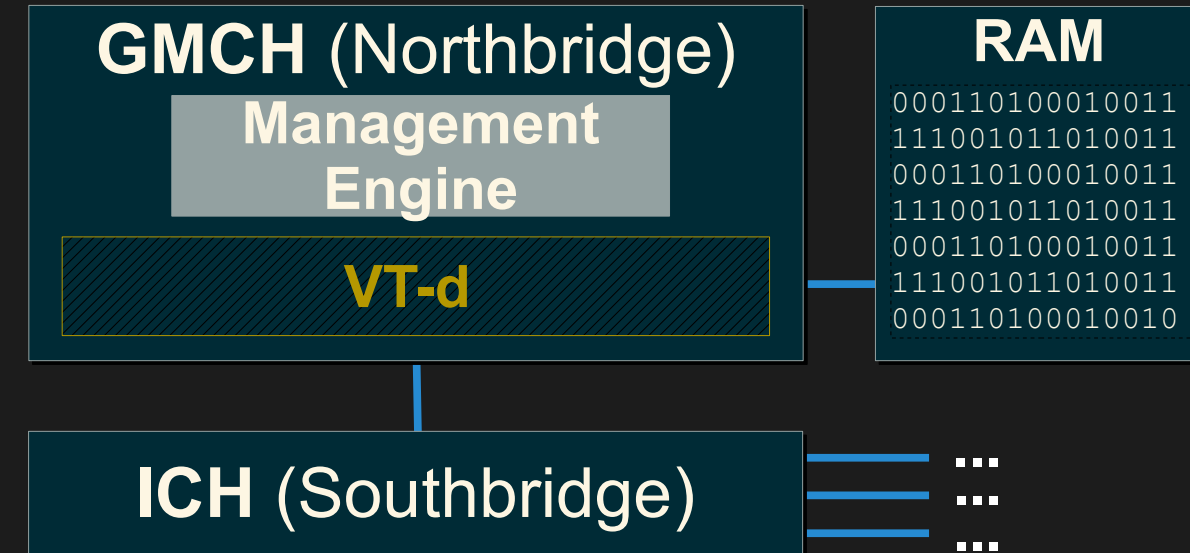
\$ No driver for Windows (including 8)

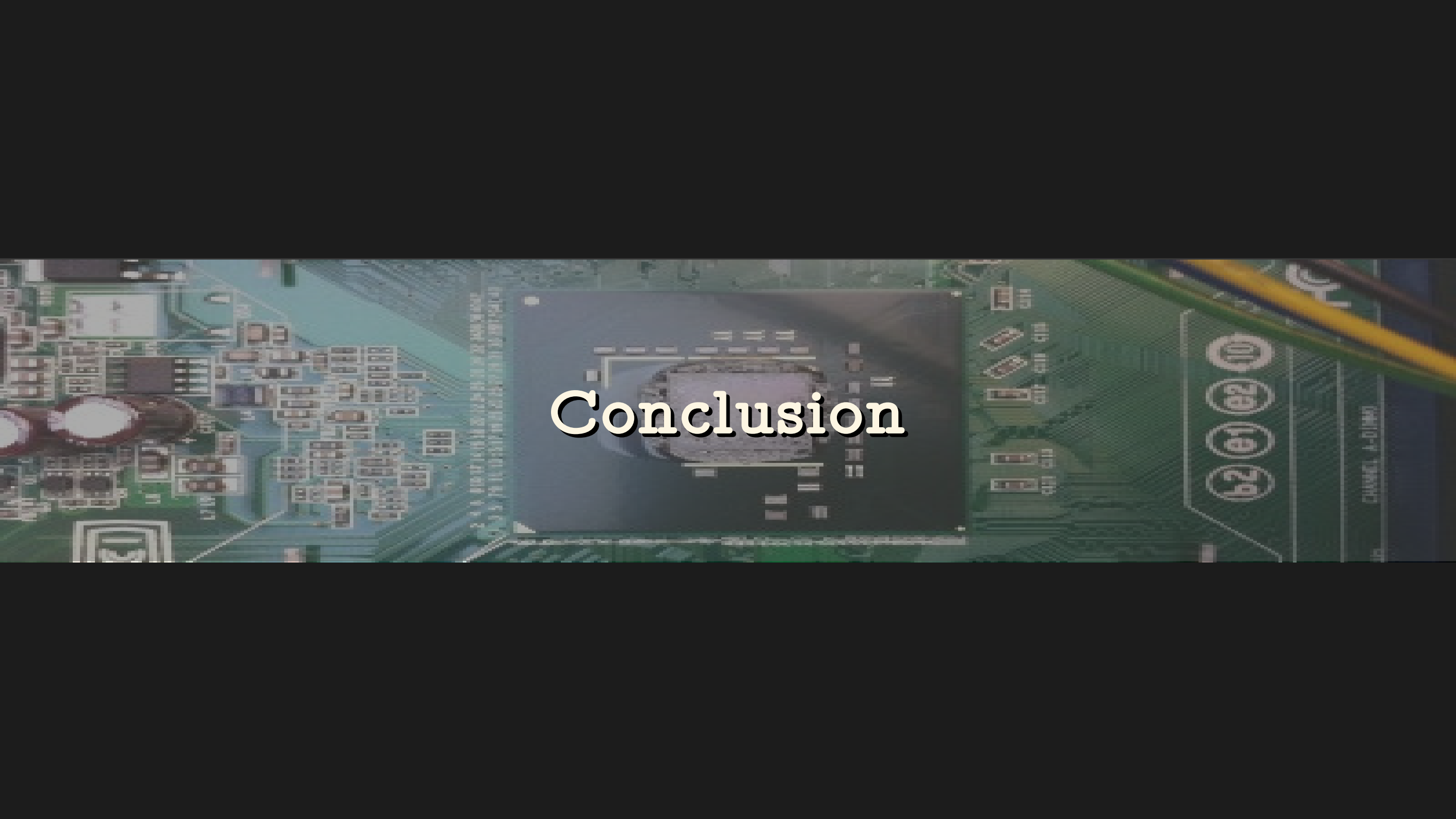
\$ Academic:

\$ VIPER [Li11] / NAVIS [Duf11] / BARM [Ste13]

\$ 30C3:

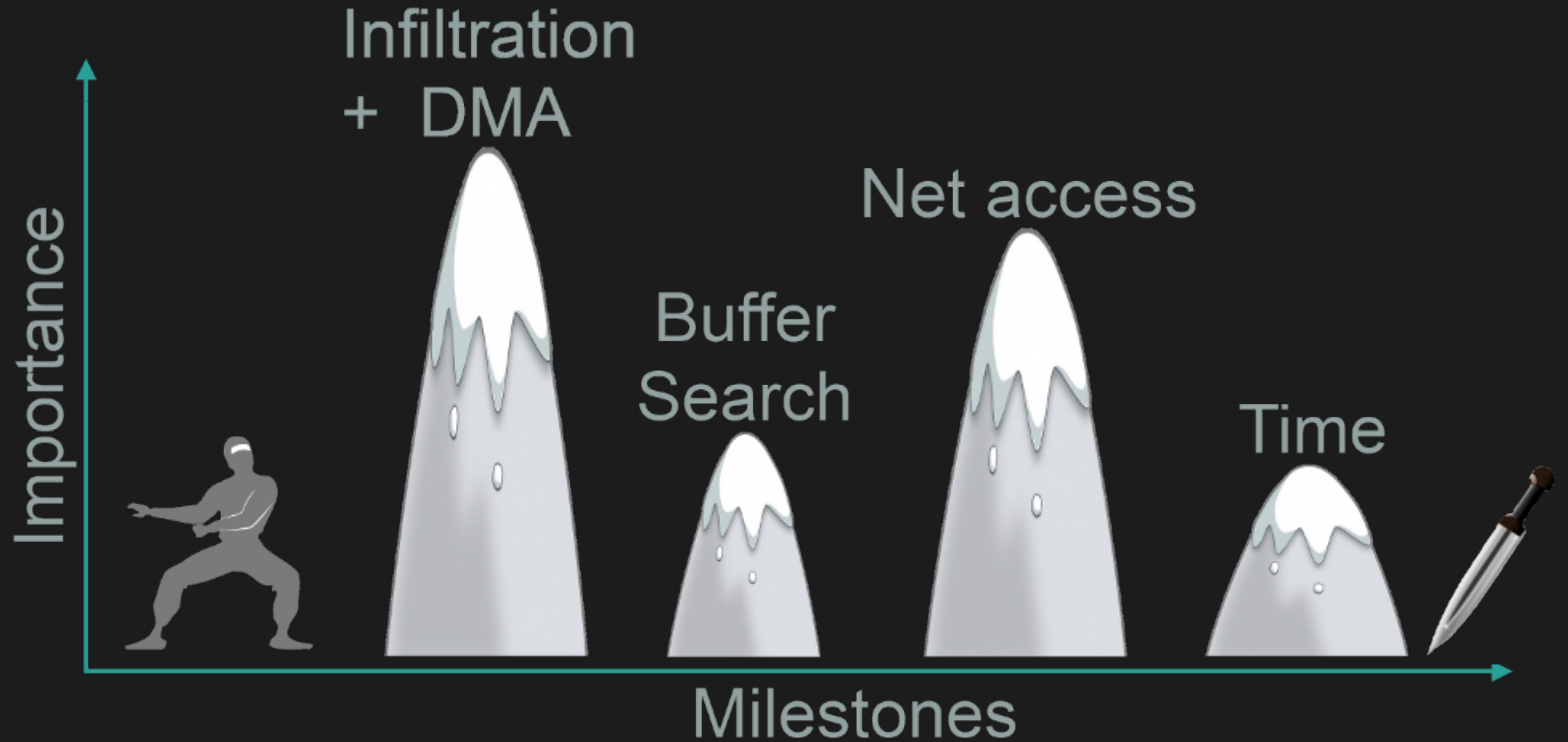
\$ Hardening hardware and choosing a #goodBIOS [Stu13]





Conclusion

```
[patrickx@30C3:~$] cat 'Conclusion'
```



Persistent, Stealthy, Remote-controlled Dedicated Hardware Malware

Patrick Stewin and Iurii Bystrov

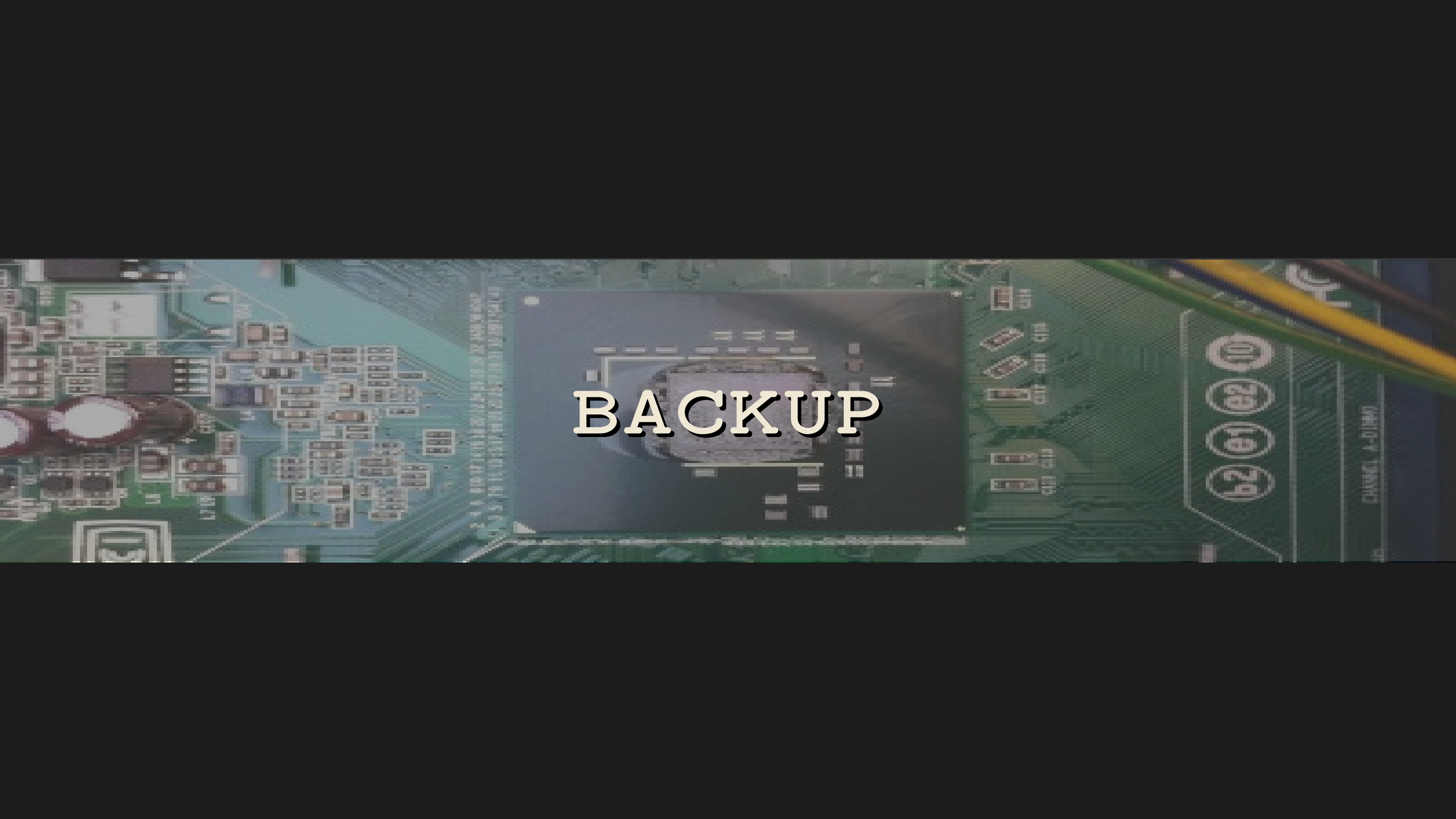
Security in Telecommunications (SecT)

TU Berlin

patrickx@sec.t-labs.tu-berlin.de

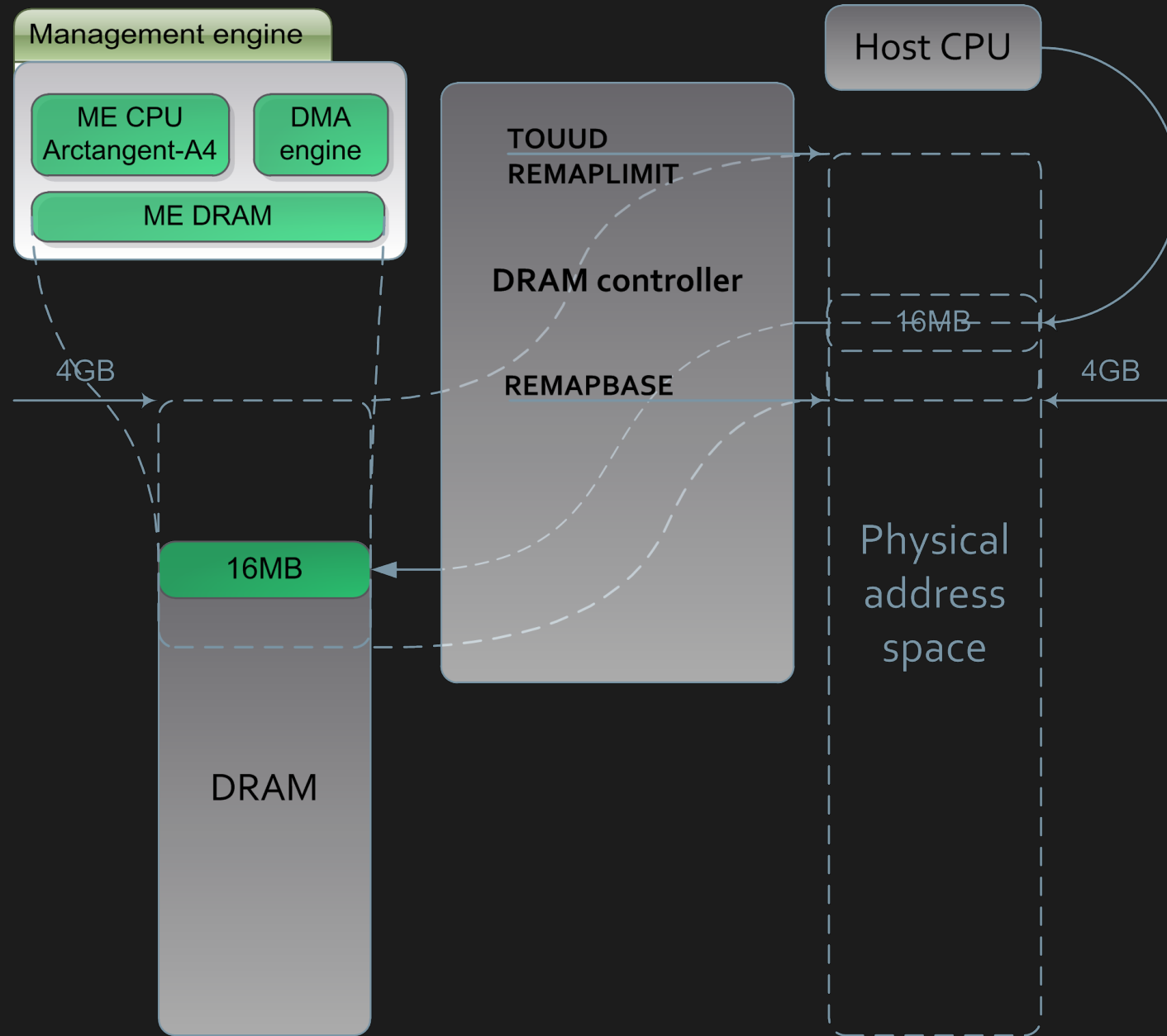
30C3, Hamburg, Germany





BACKUP

[patrickx@30C3:~\$] cat 'Memory Reclaiming'



[patrickx@30C3:~\$] cat 'References/Related Work'

- [Abr06] D. Abramson, J. Jackson, S. Muthrasanallur, G. Neiger, G. Regnier, R. Sankaran, I. Schoinas, R. Uhlig, B. Vembu, and J. Wiegert: Intel Virtualization Technology for Directed I/O
- [Aum10] D. Aumaitre and C. Devine: Subverting Windows 7 x64 Kernel with DMA attacks
- [Boi06] A. Boileau: Hit by a Bus: Physical Access Attacks with Firewire.
- [Bul08] Y. Bulygin: Chipset based Approach to detect Virtualization Malware.
- [Del10] G. Delugre: Closer to metal: Reverse engineering the Broadcom NetExtreme's firmware
- [Del11] G. Delugre. How to develop a rootkit for Broadcom NetExtreme network cards
- [Dor04] M. Dornseif: Owned by an iPod - hacking by Firewire.
- [Dor05] M. Dornseif, M. Becher, and C. N. Klein: FireWire - all your memory are belong to us
- [Duf10] L. Duflot, Y.-A. Perez, G. Valadon, and O. Levillain: Can you still trust your network card?

```
[patrickx@30C3:~$] cat 'References/Related Work'
```

[Duf11] L. Duflot, Y.-A. Perez, and B. Morin: What if you can't trust your network card?

[Int07] Intel Corporation: Intel Core 2 Duo Processor and Intel Q35 Express Chipset Development Kit

[Kum09] A. Kumar, P. Goel and Y. Saint-Hilaire: Active Platform Management Demystified – Unleashing the power of Intel vPro Technology

[Li11] Y. Li, J. M. McCune, and A. Perrig: VIPER: Verifying the integrity of peripherals' firmware

[May05] D. Maynor: DMA: Skeleton key of computing && selected soap box rants

[San10] F. Sang, E. Lacombe, V. Nicomette, and Y. Deswarte: Exploiting an I/OMMU vulnerability

[Sha06] G. Shah, A. Molina and M. Blaze: Keyboards and Covert Channels

[Sko12] I. Skochinsky: Rootkit in your laptop: Hidden code in your chipset and how to discover what exactly it does

```
[patrickx@30C3:~$] cat 'References/Related Work'
```

[Ste12] P. Stewin and I. Bystrov. Understanding DMA Malware

[Ste13] P. Stewin: A Primitive for Revealing Stealthy Peripheral-Based Attacks on the Computing Platform's Main Memory

[Stu13] P. Stuge: Hardening hardware and choosing a #goodBIOS

[Ter09] A. Tereshkin and R. Wojtczuk: Introducing Ring -3 Rootkits

[Tri08] A. Triulzi: Project Moux Mk.II.

[Tri10] A. Triulzi: The Jedi Packet Trick takes over the Deathstar

[Woj09] R. Wojtczuk, J. Rutkowska, and A. Tereshkin: Another Way to Circumvent Intel Trusted Execution Technology

[Woj11a] R. Wojtczuk,, and J. Rutkowska: Attacking Intel TXT via SINIT code execution hijacking

[Woj11b] R. Wojtczuk, and J. Rutkowska: Following the White Rabbit: Software attacks against Intel VT-d technology