

Exploiting Out-of-Order-Execution

Processor Side Channels to Enable
Cross VM Code Execution

Sophia D'Antoine

REcon 2015

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push [ebp+arg_1]
mov [ebp+arg_2], eax
call sub_314623
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

The Cloud

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

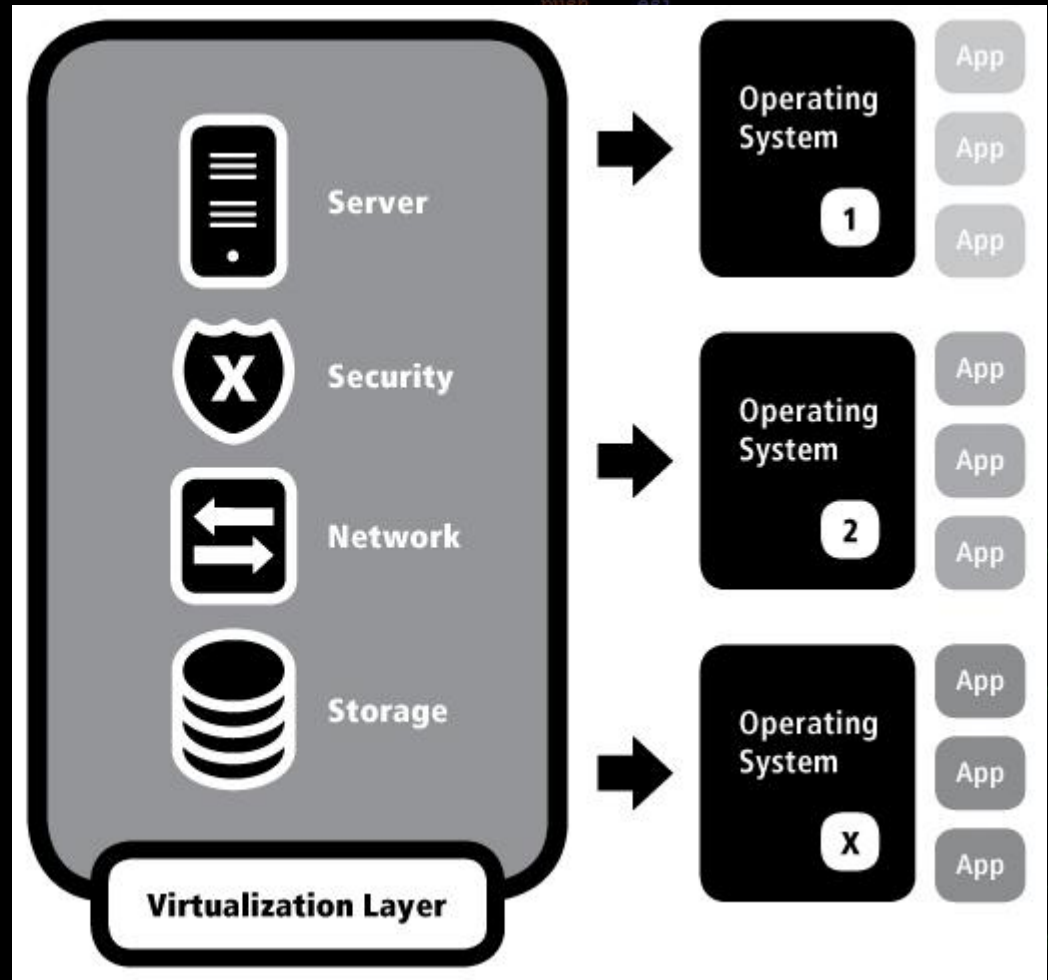
Cloud Computing (IaaS)

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
```

- Virtual instances
- Hypervisors

Dynamic allocation

=> Reduces cost



```
and eax, 0FFFFh
or eax, 80070000h
```

Everyone's Happy



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
```

```
    [ebp+arg_0], eax
sub_31486A
    eax, eax
short loc_31306D
    esi
    eax, [ebp+arg_0]
    eax
    esi, 1D0h
    esi
    [ebp+arg_4]
    edi
sub_314623
    eax, eax
short loc_31306D
    [ebp+arg_0], esi
short loc_31308F
```

```
    ; CODE XREF: sub_312FD8
    ; sub_312FD8+55
```

```
    Dh
sub_31411B
```

```
    ; CODE XREF: sub_312FD8
    ; sub_312FD8+49
```

```
sub_3140F3
    eax, eax
short loc_31307D
sub_3140F3
    jmp short loc_31308C
```

```
loc_31307D:    ; CODE XREF: sub_312FD8
```

```
    call sub_3140F3
    and eax, 0FFFFFFh
    or  eax, 80070000h
```

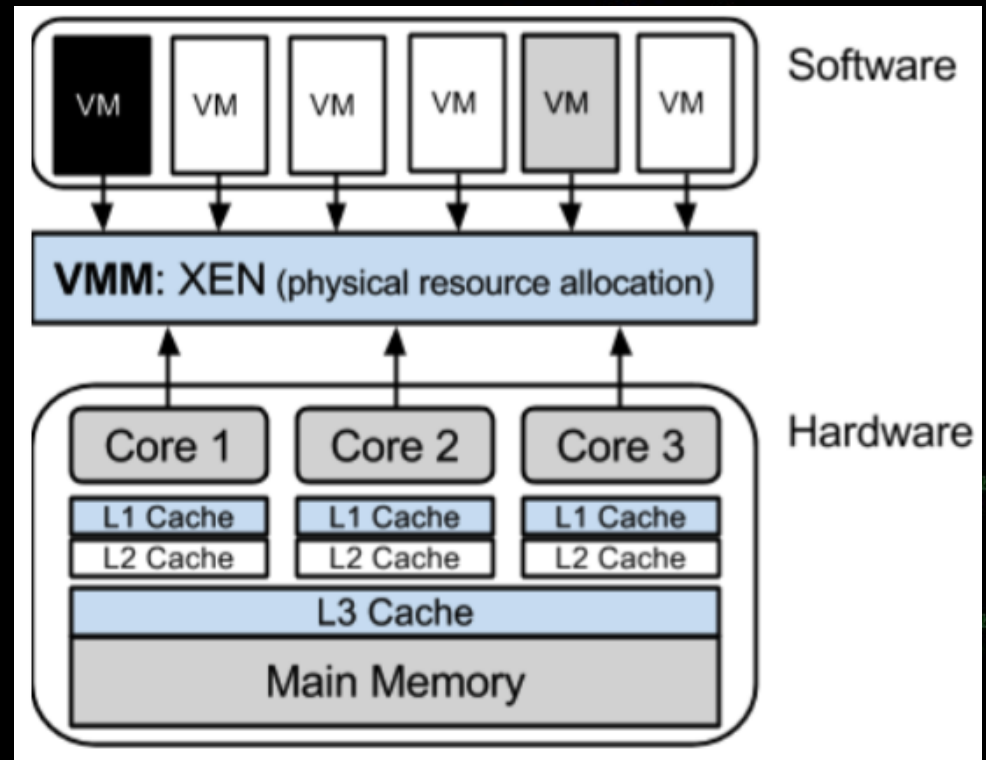
```
loc_31308C:    ; CODE XREF: sub_312FD8
```

```
    mov [ebp+var_4], eax
```

Problems with the Cloud

Security issues with cloud computing

- Sensitive data stored remotely
- Vulnerable host
- Untrusted host
- Co-located with foreign VM's



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
short loc_313066
mov eax, [ebp+var_70]
eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
```

```
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Physical co-location leads to side channel vulnerabilities.



wat

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub [ebp+var_70], 1
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

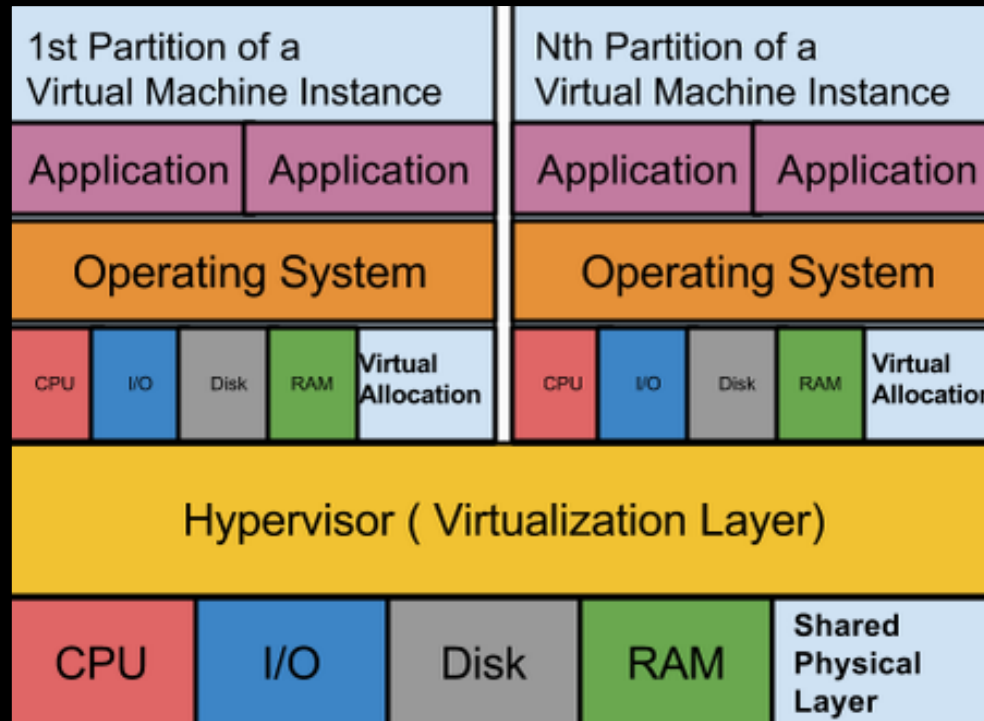
```
loc_313066: ; CODE XREF: sub_312FD8 ; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8 ; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Cloud Hardware



```

push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi

```

```

;bp+arg_0], eax
ib_31486A
ux, eax
short loc_31306D
i
i, [ebp+arg_0]
ux
i, 1D0h
i
;bp+arg_4]
li
ib_314623
ux, eax
short loc_31306D
;bp+arg_0], esi
short loc_31308F

```

```

; CODE XREF: sub_312FD8
; sub_312FD8+55

```

```

;h
ib_31411B

```

```

; CODE XREF: sub_312FD8
; sub_312FD8+49

```

```

ib_3140F3
ux, eax
short loc_31307D
call sub_3140F3
jmp short loc_31308C

```

```

loc_31307D: ; CODE XREF: sub_312FD8

```

```

call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

```

```

loc_31308C: ; CODE XREF: sub_312FD8

```

Universal Vulnerabilities

- 1) **Translation** between physical and virtual hardware based on need
- 2) Allocation causes **contention**
- 3) Private VM activities **not opaque** to co-residents

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```


Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the processor
4. Importance of memory models
5. Design of an Out-of-Order-Execution channel
6. Demo
7. Conclusion

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
mov [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Side Channel Attack

“In cryptography, a **side-channel** attack is any attack based on information gained from the physical implementation of a cryptosystem”

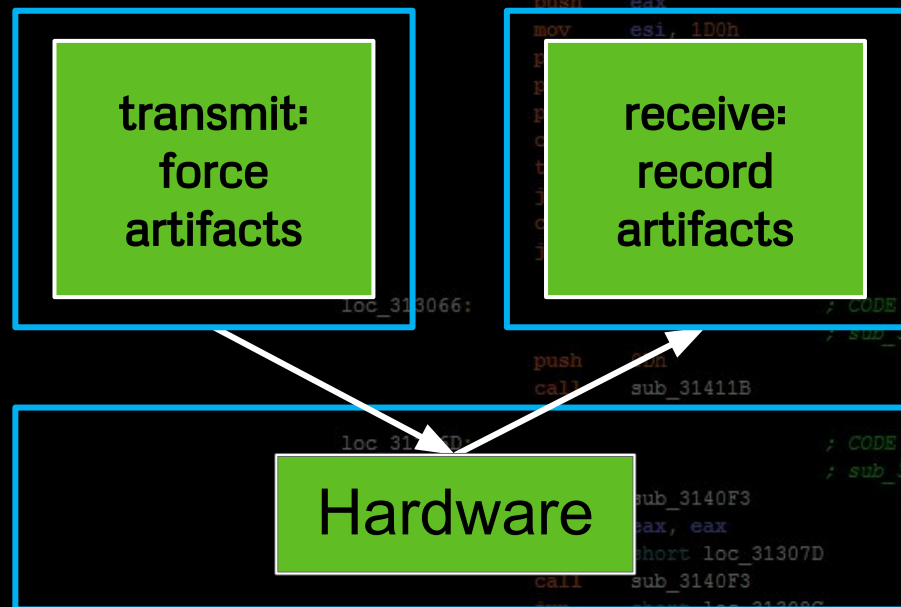
Cloud Computing

- **Hardware** side channel
- **Cross** virtual machine
- Information gained through **recordable** changes in the system

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_314623
test eax, eax
jz short loc_31306D
push esi
test eax, [ebp+arg_0]
jnz short loc_313066
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jnz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
loc_313066: ; CODE XREF: sub_312FD8+55
; sub_312FD8+55
call sub_31411B
loc_31306D: ; CODE XREF: sub_312FD8+49
; sub_312FD8+49
call sub_3140F3
test eax, eax
jnz short loc_31306D
call sub_3140F3
jmp short loc_31308C
loc_31307D: ; CODE XREF: sub_312FD8+55
; sub_312FD8+55
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C: ; CODE XREF: sub_312FD8+55
; sub_312FD8+55
mov [ebp+var_4], eax
```

Classification S/R Model

- Hardware **agnostic**
- **Two** methods of interacting
 - Transmit
 - Receive



Possible Exploits

- **Receive** (exfiltrate)
 1. crypto key theft
 2. process monitoring
 3. environment keying
 4. broadcast signal
- **Transmit** (infiltrate)
 1. DoS
 2. co-residency
- **Transmit & Receive** (network)
 1. communication (C&C)

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

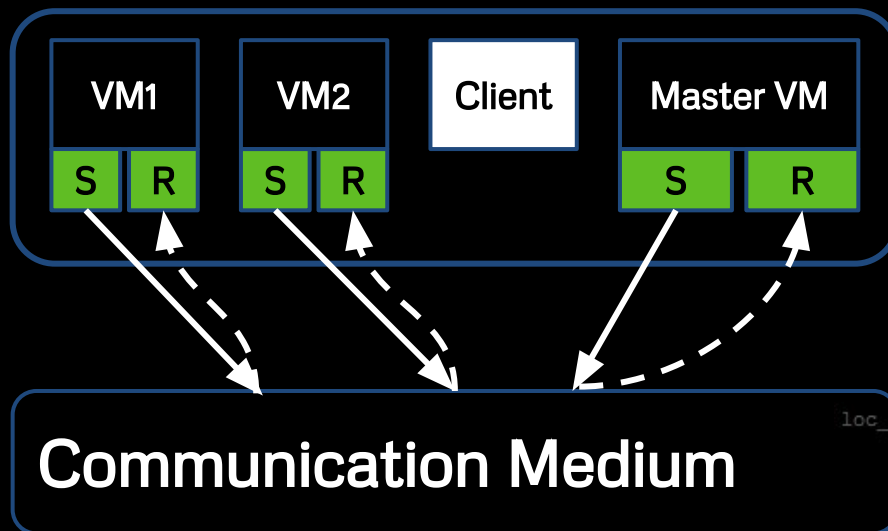
```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Communication



Virtual Allocations

Shared Hardware

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
esi
lea eax, [ebp+arg_0]
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_31306E: ; CODE XREF: sub_312FD8 ; sub_312FD8+55
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8 ; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Cache Side Channel Example [3]

Flush+Reload targets the L3 Cache Tier

- Receiving Mechanism (Adversary)
 - Flushes & queries
- Transmitting Mechanism (Victim)
 - Accesses same L3 line
- Leaked GnuPG Private Key

sophia.re/cache.pdf

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_0]
xor eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Pipeline vs Cache Channel

Benefits:

- Quiet, **covert** channel
- Not affected by **cache misses**, etc.
- Channel & noise **amplifies** in a **crowded** cloud environment

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jns short loc_313066
mov eax, [ebp+var_70]
cmpl eax, [ebp+var_84]
jnb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the pipeline
4. Importance of memory models
5. Design of an Out-of-Order-Execution channel
6. Demo
7. Conclusion

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
mov     [ebp+arg_0], esi
jz      short loc_31308F
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+55
```

```
push    0Dh
call    sub_31411B
```

```
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

```
mov     [ebp+var_4], eax
```


The Attack Vector

Side Channels which Exploit Hardware Vulnerabilities Inherent to Modern Cloud Computing Systems

Requirements:

- **Shared** hardware
- **Dynamically** allocated hardware resources
- **Co-Location** with adversarial VMs or infected VMs

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
[ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jz short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Pipeline Side Channel

We chose to target the **processor** as the hardware medium.

=> CPU's **pipeline**

=> System artifacts queried **dynamically**

- Instruction order
- Results from instruction sets

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
```

```
push esi
push eax
push edi
mov [ebp+arg_0], ebx
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
mov [ebp+arg_0], ebx
jz short loc_31306D
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jnz short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Out-of-Order-Execution

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+55
push    0Dh
call    sub_31411B

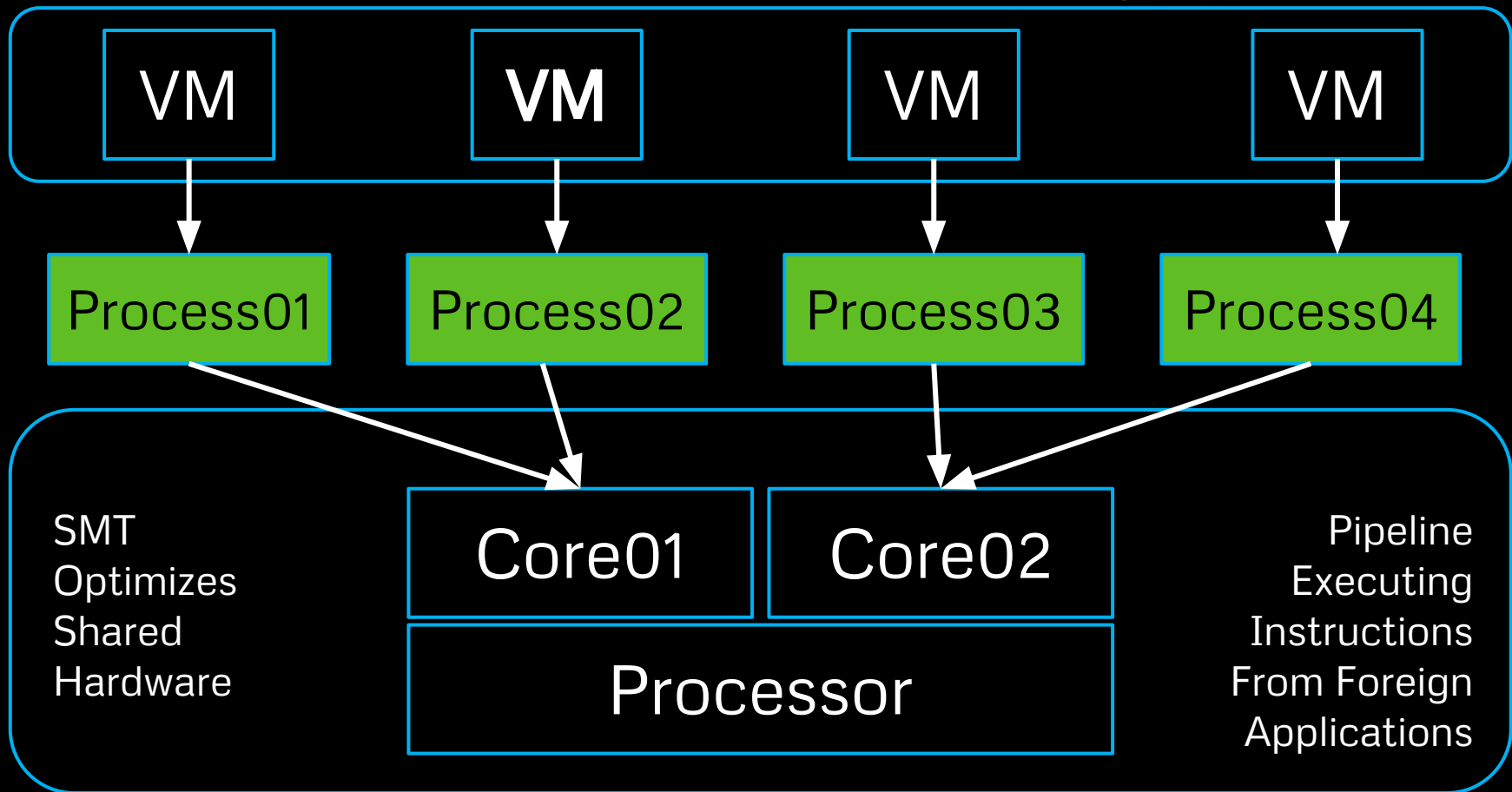
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

; -----
loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax
```

Processor Pipeline Contention

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_74]
mov eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
```



```
and eax, 0FFFFh
or eax, 80070000h
```

```
loc_31308C:
mov [ebp+var_4], eax
; CODE XREF: sub_312FD8
```

RECEIVER

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+55
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C

; -----
loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax

```

Record Out of Order Execution [6]

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jra short loc_313066
mov eax, [ebp+var_70]
call eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
sub_314623
```

8.2.3.4 Loads May Be Reordered with Earlier Stores to Different Locations

The Intel-64 memory-ordering model allows a load to be reordered with an earlier store to a different location. However, loads are not reordered with stores to the same location.

The fact that a load may be reordered with an earlier store to a different location is illustrated by the following example:

Example 8-3. Loads May be Reordered with Older Stores

Processor 0	Processor 1
mov [_x], 1	mov [_y], 1
mov r1, [_y]	mov r2, [_x]
Initially x = y = 0	
r1 = 0 and r2 = 0 is allowed	

```
loc_31306D: ; CODE XREF: sub_312FD8 ; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

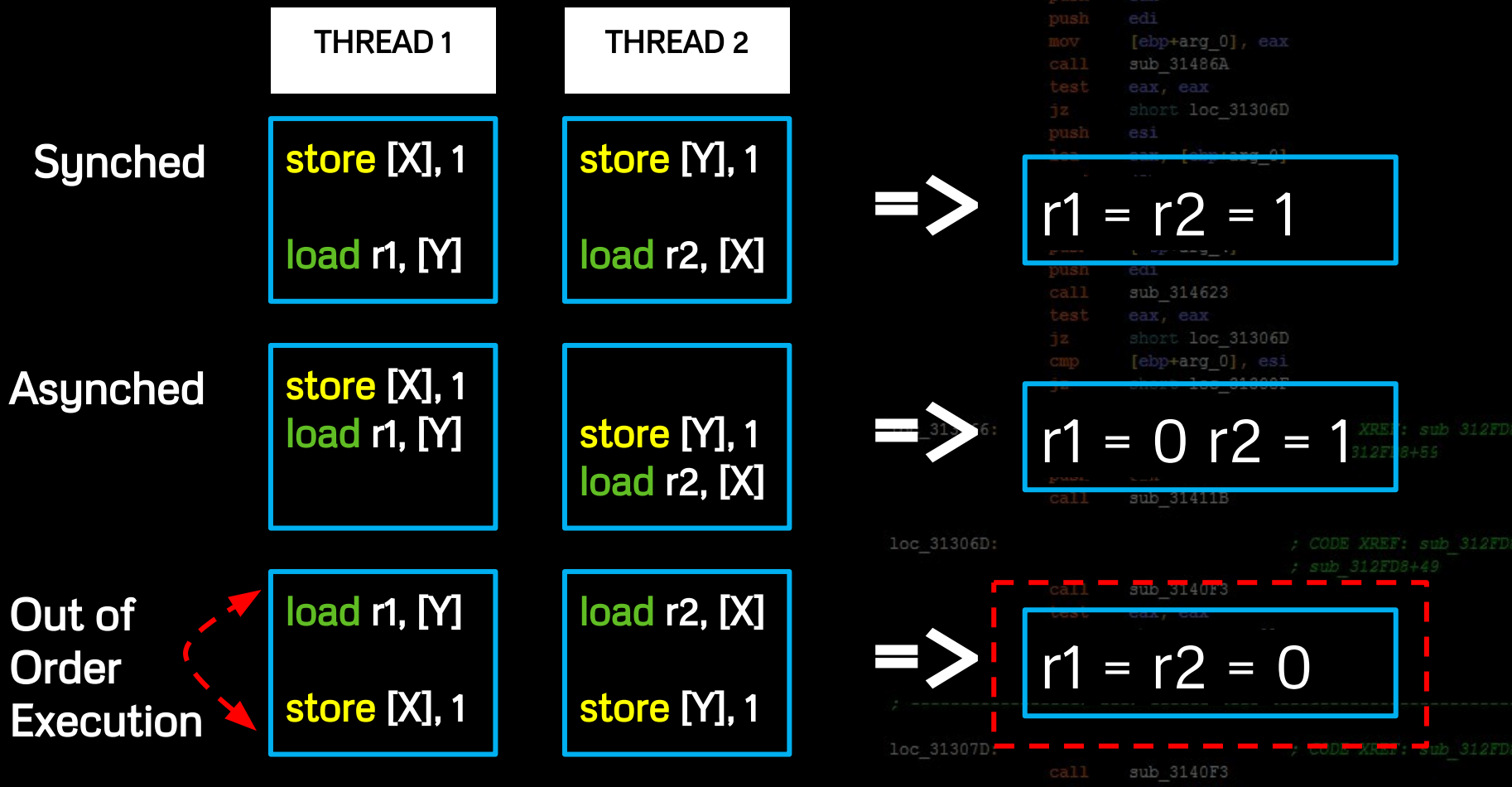
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Record Out of Order Execution

```

push  edi
call  sub_314623
test  eax, eax
jz    short loc_31306D
cmp   [ebp+arg_0], ebx
jra   short loc_313066
mov   eax, [ebp+var_70]
call  sub_314623
jz    short loc_313066
jb    short loc_313066
sub   eax, [ebp+var_84]
push  esi
push  esi
push  eax
push  edi
mov   [ebp+arg_0], eax
call  sub_31486A
test  eax, eax
jz    short loc_31306D
push  esi
lea   eax, [ebp+arg_0]

```



```

push  edi
call  sub_314623
test  eax, eax
jz    short loc_31306D
cmp   [ebp+arg_0], esi
jz    short loc_31306F
push  esi
call  sub_31411B
loc_31306D:
; CODE XREF: sub_312FD8
; sub_312FD8+49
call  sub_3140F3
loc_31307D:
; CODE XREF: sub_312FD8
call  sub_3140F3
and   eax, 0FFFFFFh
or    eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8
mov   [ebp+var_4], eax

```

Record Out of Order Execution

```
int X,Y,count_OoOE;
```

```
....initialize semaphores Sema1 & Sema2...
```

```
pthread_t thread1, thread2;
```

```
pthread_create(&threadN, NULL, threadNFunc, NULL);
```

```
for (int iterations = 1; ; iterations++)
```

```
    X,Y = 0;
```

```
    sem_post(beginSema1 & beginSema2);
```

```
    sem_wait(endSema1 & endSema2);
```

```
    if (r1 == 0 && r2 == 0)
```

```
        count_OoOE ++;
```

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+arg_70]
mov     esi, [ebp+arg_74]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea    eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

```
loc_31306D: ; CODE XREF: sub_312FD8+49
```

Averages matter

```
loc_31306D: ; CODE XREF: sub_312FD8+49
; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8+49
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8+49
```

```
mov     [ebp+var_4], eax
```


TRANSMITTER

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Force Out of Order Execution

Memory Fences

Mfence:

- x86 instruction full memory barrier **prevents** memory reordering of any kind
- order of **100 cycles** per operation

```
push    edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], ebx
jnz   short loc_313066
mov    eax, [ebp+var_10]
cmp    [ebp+var_14]
jb     short loc_313066
sub    eax, [ebp+var_84]
push   esi
push   esi
push   eax
push   edi
mov    [ebp+arg_0], eax
call   sub_31486A
test   eax, eax
jz     short loc_31306D
push   esi
lea   eax, [ebp+arg_0]
push   eax
mov    esi, 1D0h
push   esi
push   [ebp+arg_4]
push   edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], esi
jz     short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
... mov dword ptr [_spin1], 0
... mfence
```

```
XREF: sub_312FD8
312FD8+49
```

```
... mov dword ptr [_spin2], 0
... mfence
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call   sub_3140F3
and    eax, 0FFFFFFh
or     eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov    [ebp+var_4], eax
```

Force Out of Order Execution

THE PIPELINE



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
inc ebx
mov eax, [ebp+var_10]
cmp [ebp+var_14], eax
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
```

```
loc_31306D: ; CODE XREF: sub_312FD8 ; sub_312FD8+49
call sub_31411B
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the processor
4. Importance of memory models
5. Design of an Out-of-Order-Execution channel
6. Demo
7. Conclusion

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

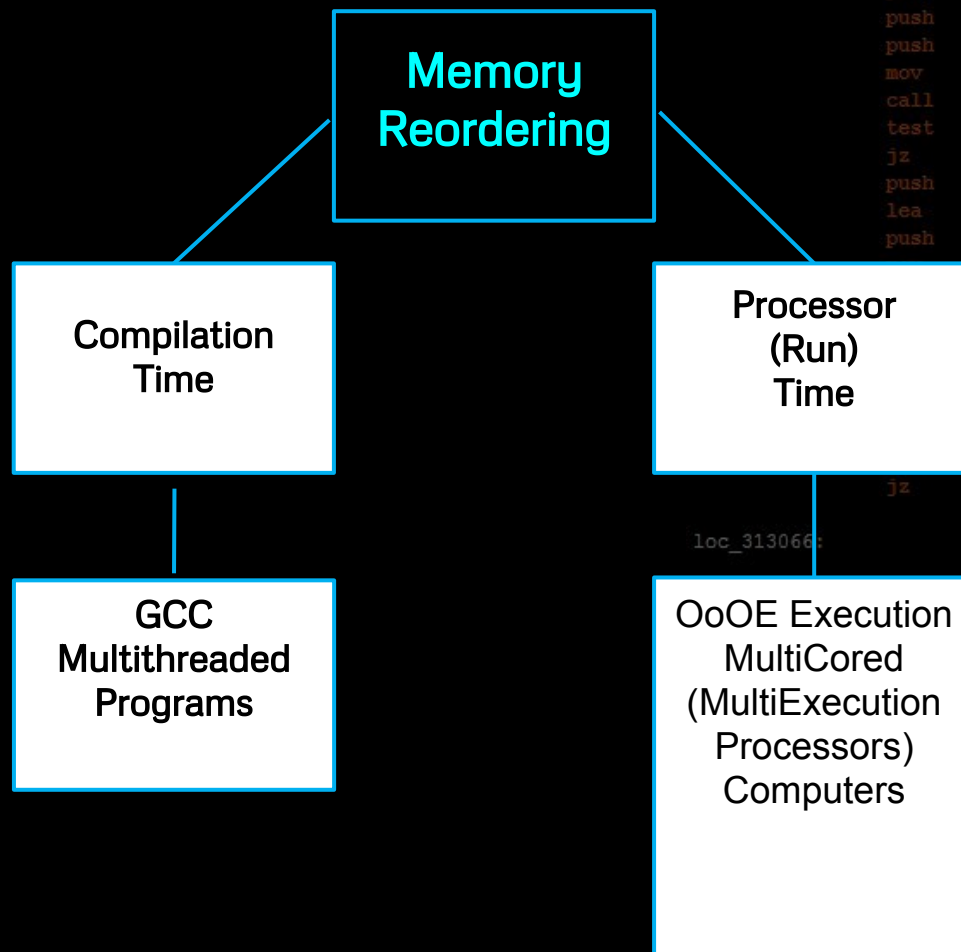
```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Types of Memory Reordering



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
mov [ebp+var_80], eax
cmp [ebp+var_84], eax
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
esi, 1D0h
esi
[ebp+arg_4]
edi
sub_314623
eax, eax
short loc_31306D
[ebp+arg_0], esi
short loc_31308F
jz
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
0Dh
sub_31411B
; CODE XREF: sub_312FD8
; sub_312FD8+49
sub_3140F3
eax, eax
short loc_31307D
sub_3140F3
short loc_31308C
; CODE XREF: sub_312FD8
sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Types of Memory Reordering

Dynamic side channel artifacts

Processor
(Run)
Time

OoOE Execution
MultiCored
(MultiExecution
Processors)
Computers

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
jmp [ebp+arg_0], ebx
mov ebx, 10_313066
eax, [ebp+var_80]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Weak Memory Models [7]

```

push  edi
call  sub_314623
test  eax, eax
jz    short loc_31306D
cmp   [ebp+arg_0], ebx
jnz   short loc_313066
mov   eax, [ebp+var_70]
cmp   eax, [ebp+var_84]
jb    short loc_313066
sub   eax, [ebp+var_84]
push  esi
push  esi
push  eax
push  edi
    
```



```

XREF: sub_312FD8
312FD8+55
XREF: sub_312FD8
312FD8+49
    
```

```

loc_31307D:                                ; CODE XREF: sub_312FD8
call  sub_3140F3
and   eax, 0FFFFFFh
or    eax, 80070000h

loc_31308C:                                ; CODE XREF: sub_312FD8
mov   [ebp+var_4], eax
    
```

Types of Memory Reordering

4 types of run time reordering barriers



[4, 5]

- Instruction A visible to all processes before B occurs
- #StoreLoad most expensive operation

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+var_80], ebx
jbe short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
sub [ebp+var_84], eax
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
push [ebp+arg_0], esi
push short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```


Force Out of Order Execution

Memory Barrier

- ‘Lock-free programming’ on SMT multiprocessors
- #StoreLoad unique prevents r1=r2=0
- x86: **mfence** (effects the pipeline)

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
inc ebx
mov eax, [ebp+var_10]
cmp [ebp+var_14], eax
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
mov eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
loc_313166: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Channel Transmitter (Victim)

- Force Out-of-Order-Execution patterns
- Affect the order of stores and loads
- Time frame dependant
- x86: mfence



Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the processor
4. Importance of memory models
5. Design of an Out of Order Execution channel
6. Demo
7. Conclusion

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_313066
mov [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Lab Model

Scheduler **Xen** hypervisor

- Popular commercial IaaS platforms

Xeon Processors

Shared **multi-core/ multi-processor** hardware

- **8** logical CPU's/ **4** cores
- **6** virtual machines (VM's)
- Parallel Processing/ Simultaneous Multi-Threading
On (**SMT**)

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
esi, 1D0h
push esi
[ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
```

```
REF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

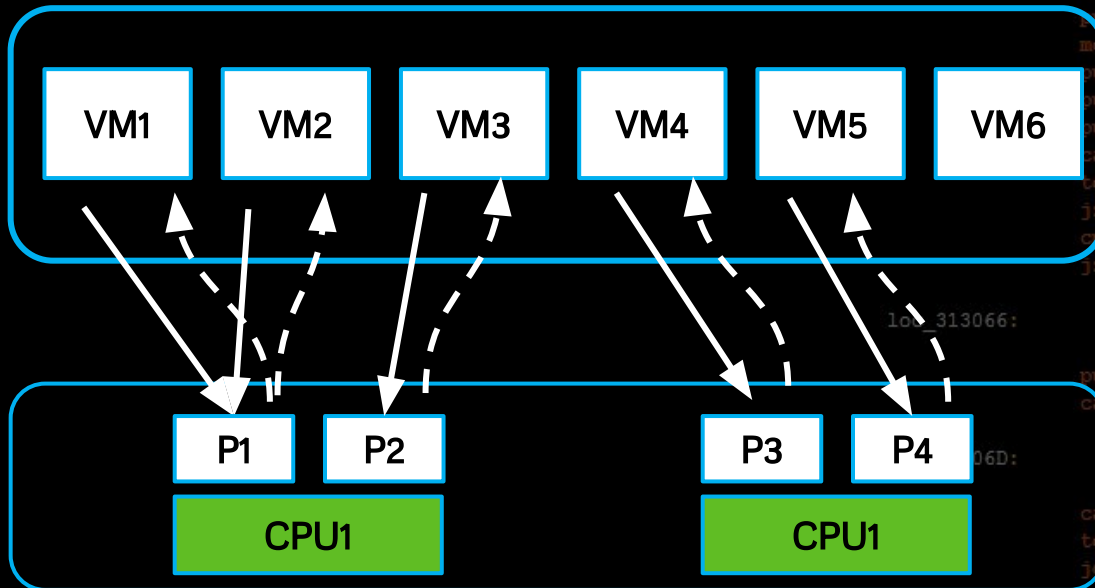
```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

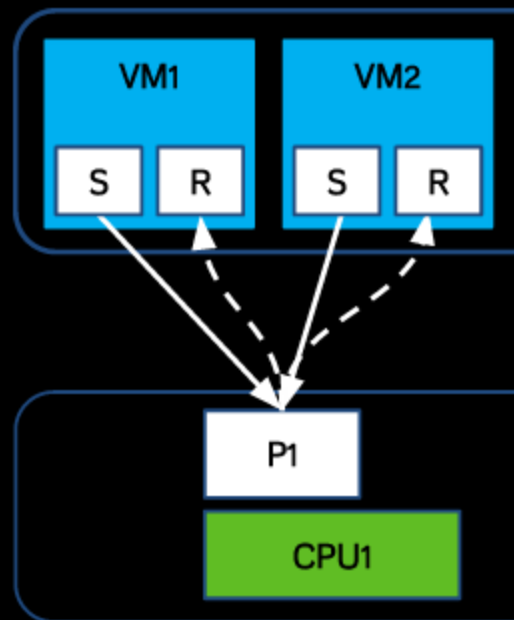
Virtual Machines

- 6 Windows 7 VM's



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
ja short loc_31306D
cp [ebp+arg_0], esi
jz short loc_31308F
loc_313066:
push 0Dh
call sub_31411B
loc_31306D:
call sub_3140F3
test eax, eax
j short loc_31307D
call sub_3140F3
jmp short loc_31308C
; CODE XREF: sub_312FD8
; sub_312FD8+55
; CODE XREF: sub_312FD8
; sub_312FD8+49
; CODE XREF: sub_312FD8
; sub_312FD8+49
```

Virtual Machine S/R



```

push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

```

```

13066: ; CODE XREF: sub_312FD8
; sub_312FD8+55

```

```

push 0Dh
call sub_31411B

```

```

1306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49

```

```

call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

```

```

loc_31307D: ; CODE XREF: sub_312FD8

```

```

call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

```

```

loc_31308C: ; CODE XREF: sub_312FD8

```

```

mov [ebp+var_4], eax

```

Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the processor
4. Importance of memory models
5. Design of an Out-of-Order-Execution channel
6. Demo
7. Conclusion

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
mov [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Demo Links

sophia.re/sender.py

sophia.re/receiver.py

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```


Overview

1. Introduction
2. Cloud exploitation techniques
3. Targeting the processor
4. Importance of memory models
5. Design of an Out-of-Order-Execution channel
6. Demo
7. Conclusion

```
push    edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], ebx
jnz   short loc_313066
mov    eax, [ebp+var_70]
cmp    eax, [ebp+var_84]
jb     short loc_313066
sub    eax, [ebp+var_84]
push   esi
push   esi
push   eax
push   edi
mov    [ebp+arg_0], eax
call   sub_31486A
test   eax, eax
jz     short loc_31306D
push   esi
lea   eax, [ebp+arg_0]
push   eax
mov    esi, 1D0h
push   esi
push   [ebp+arg_4]
push   edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
mov    [ebp+arg_1], esi
jz     short loc_31308F
```

```
loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+55
```

```
push   0Dh
call   sub_31411B
```

```
loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49
```

```
call   sub_3140F3
test   eax, eax
jg     short loc_31307D
call   sub_3140F3
jmp    short loc_31308C
```

```
loc_31307D:                                     ; CODE XREF: sub_312FD8
```

```
call   sub_3140F3
and    eax, 0FFFFFFh
or     eax, 80070000h
```

```
loc_31308C:                                     ; CODE XREF: sub_312FD8
```

```
mov    [ebp+var_4], eax
```

Potential Channel Mitigation

Protected Resource Ownership

- Isolating VM's
- Turn off hyperthreading
- Blacklisting resources for concurrent threads
- Downside: cloud benefits

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
eax, [ebp+var_70]
eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
lea eax, [ebp+var_23]
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

In Conclusion...

Contribution:

We demonstrate a novel **Out of Order Execution** side channel.

- **Dynamic** querying/ forcing method
- **Application** to cloud computing
- **Mitigation** techniques

```
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F
```

```
313066:                                     ; CODE XREF: sub_312FD8
                                     ; sub_312FD8+55
```

```
push    0Dh
call    sub_31411B
```

```
loc_31306D:                             ; CODE XREF: sub_312FD8
                                     ; sub_312FD8+49
```

```
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
```

```
loc_31307D:                             ; CODE XREF: sub_312FD8
```

```
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h
```

```
loc_31308C:                             ; CODE XREF: sub_312FD8
```

```
mov     [ebp+var_4], eax
```

Acknowledgements

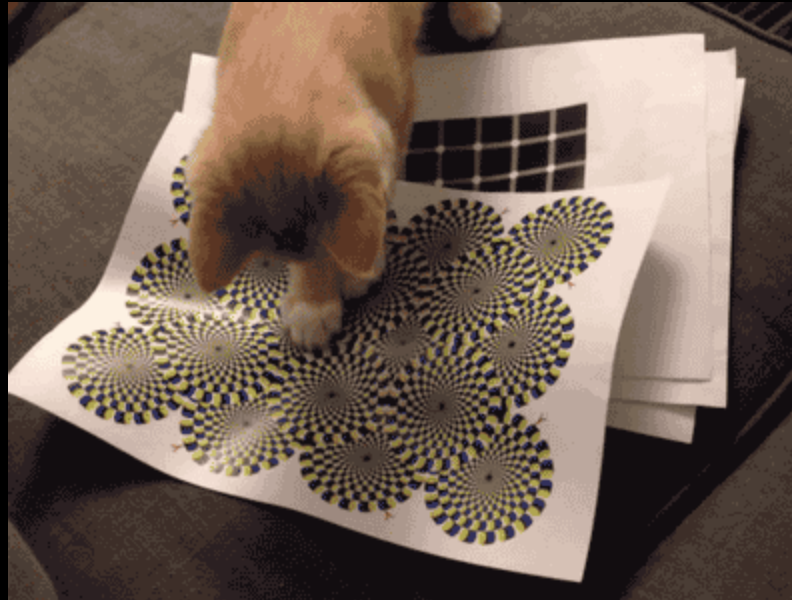
- Jeremy Blackthorne
- RPISEC
- Trail of Bits

The logo for Trail of Bits, featuring the words "TRAIL" and "BITS" in a large, bold, white sans-serif font with a slight 3D effect and shadows. The word "OF" is smaller and positioned between "TRAIL" and "BITS".

64 6f 6f 6d 2e 6c 79 6e 78 6a 65 72
RPISEC
2e 71 75 65 6e 64 2e 55 6e 69 78 2d

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push esi
mov [ebp+var_70], eax
push esi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jz short loc_31308C
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Any Questions?



IRC: quend (#rπισec, #pwning)
email: sophia@trailofbits.com
thesis link: sophia.re/thesis.pdf

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

loc_31307D: call sub_3140F3 ; CODE XREF: sub_312FD8

```
and eax, 0FFFFFFh
or eax, 80070000h
```

loc_31308C: mov [ebp+var_4], eax ; CODE XREF: sub_312FD8

References

- [1] <http://www.thewhir.com/web-hosting-news/aws-to-reach-24-billion-in-revenue-by-2022-morgan-stanley>
- [2] <http://www.forbes.com/sites/louiscolombus/2015/01/24/roundup-of-cloud-computing-forecasts-and-market-estimates-2015/>
- [3] <https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-yarom.pdf>
- [4] <http://bartoszmilewski.com/2008/11/05/who-ordered-memory-fences-on-an-x86/>
- [5] <http://preshing.com/20120913/acquire-and-release-semantics/>
- [6] http://www.intel.com/Assets/en_US/PDF/manual/253668.pdf
- [7] <http://preshing.com/20120930/weak-vs-strong-memory-models/>
- [8] http://en.wikipedia.org/wiki/Memory_barrier#An_illustrative_example
- [9] <http://preshing.com/20120710/memory-barriers-are-like-source-control-operations/>

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
test eax, eax
jz short loc_31306D
push esi
eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

; -----
loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

EXTRA SLIDES

```

push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], ebx
jnz     short loc_313066
mov     eax, [ebp+var_70]
cmp     eax, [ebp+var_84]
jb      short loc_313066
sub     eax, [ebp+var_84]
push    esi
push    esi
push    eax
push    edi
mov     [ebp+arg_0], eax
call    sub_31486A
test    eax, eax
jz      short loc_31306D
push    esi
lea     eax, [ebp+arg_0]
push    eax
mov     esi, 1D0h
push    esi
push    [ebp+arg_4]
push    edi
call    sub_314623
test    eax, eax
jz      short loc_31306D
cmp     [ebp+arg_0], esi
jz      short loc_31308F

loc_313066:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+55
push    0Dh
call    sub_31411B

loc_31306D:                                     ; CODE XREF: sub_312FD8
                                                ; sub_312FD8+49
call    sub_3140F3
test    eax, eax
jg      short loc_31307D
call    sub_3140F3
jmp     short loc_31308C
; -----

loc_31307D:                                     ; CODE XREF: sub_312FD8
call    sub_3140F3
and     eax, 0FFFFFFh
or      eax, 80070000h

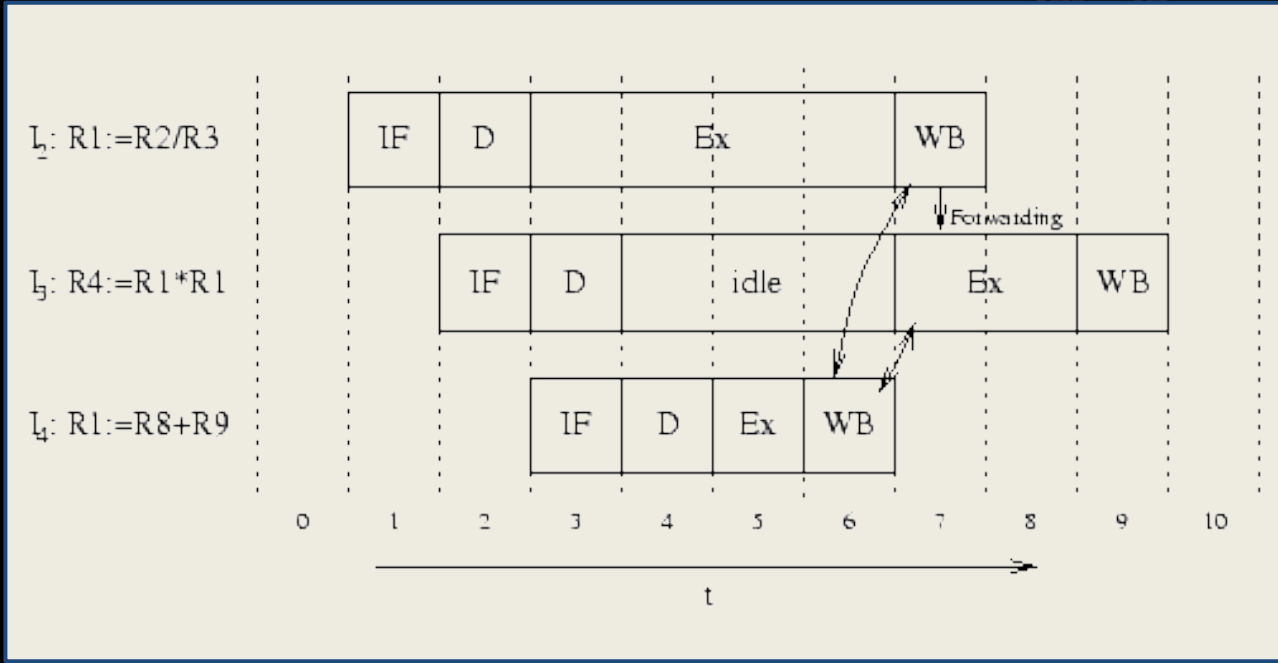
loc_31308C:                                     ; CODE XREF: sub_312FD8
mov     [ebp+var_4], eax

```

```

push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi

```



```

eax
006D
arg_0]
006D
esi
008F
; CODE XREF: sub_312FD8
; sub_312FD8+55
; CODE XREF: sub_312FD8
; sub_312FD8+49

```

```

call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

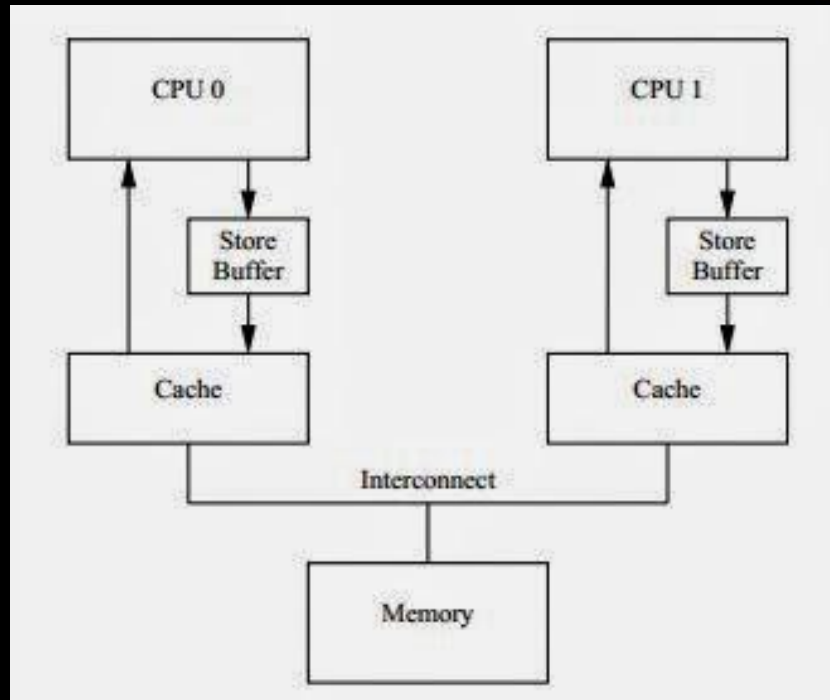
```

```

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax

```

```

push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
sh edi
v [ebp+arg_0], eax
ll sub_31486A
st eax, eax
sh short loc_31306D
a esi
sh eax, [ebp+arg_0]
sh eax
v esi, 1D0h
sh esi
sh [ebp+arg_4]
sh edi
ll sub_314623
st eax, eax
p short loc_31306D
[ebp+arg_0], esi
short loc_31308F
; CODE XREF: sub_312FD8
; sub_312FD8+55
sh 0Dh
ll sub_31411B
; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C

```

```

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax

```

OoOE vs Other Channel

Applicability:

- Subversive **applications** show potential
- **Detection difficult** by an “intelligent” hypervisor
- Interference (**eavesdropping**) sufficiently **mutilates** channel

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
mov [ebp+arg_0], esi
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

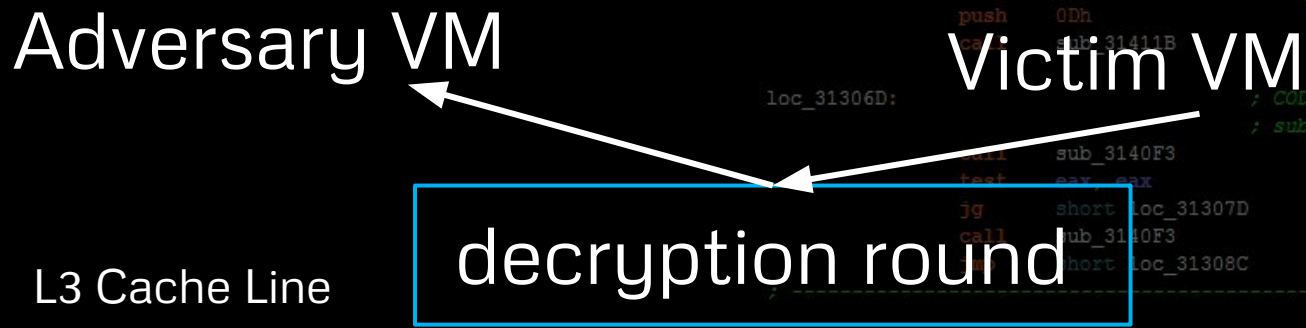
```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Cache Side Channel Example [3]

- Successfully leaked the private key from the **GnuPG**
- Leaked **96.7%** bits of the secret key



```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_84]
jnb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F

loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B

loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jnz short loc_31308C

loc_31307D: ; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h

loc_31308C: ; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Classification of Each Unit

```

push    edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], ebx
jnc    short loc_313066
mov    eax, [ebp+var_70]
cmp    eax, [ebp+var_84]
jb     short loc_313066
sub    eax, [ebp+var_84]
push   esi
push   esi
push   eax
    
```

Hardware Medium	Transmitting Mechanism	Reception Mechanism
Processor	Processor Register and Functional Unit Resources Contention	Time Compared Against Threshold
Cache Tier	Prime-Probe, Shared Cache Functionality	Time Compared Against Threshold
System Bus	System Bus Restricted Access Contention	Measurement of Memory Access Capabilities
Main Memory	Prime-Probe, Shared Main Memory Storage	Measurement of Memory Access Capabilities
Hard Disk Drive	Prime-Probe, Shared Disk Drive Data Access	Time Compared Against Threshold

```

sub_312FD8
55
    
```

```

sub_312FD8
49
    
```

```

loc_31307D: ; CODE XREF: sub_312FD8
    
```

```

call   sub_3140F3
and    eax, 0FFFFFFh
or     eax, 80070000h
    
```

```

loc_31308C: ; CODE XREF: sub_312FD8
    
```

```

mov    [ebp+var_4], eax
    
```

List of Physically Shared Units

- Processors (CPU/ GPU)
- Cache Tiers
- System Buses
- Main Memory
- Hard Disk Drive

Literature demonstrates exploits across each hardware unit.

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
short loc_313066
mov eax, [ebp+var_70]
mov [ebp+var_84], [ebp+var_70]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jz short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

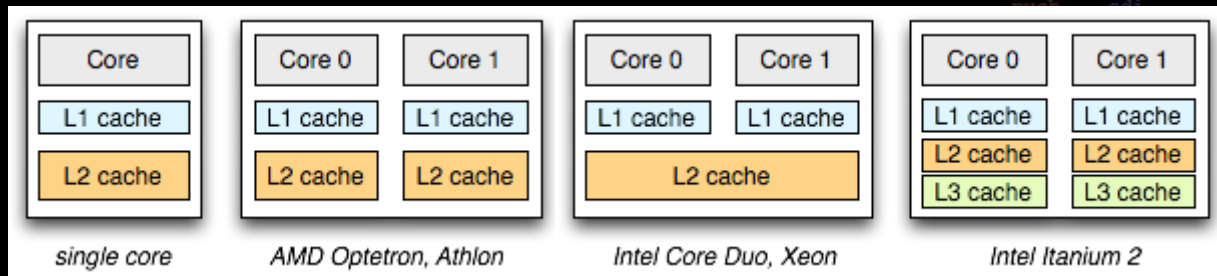
```
loc_31307D: ; CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Hardware Architectures (1)

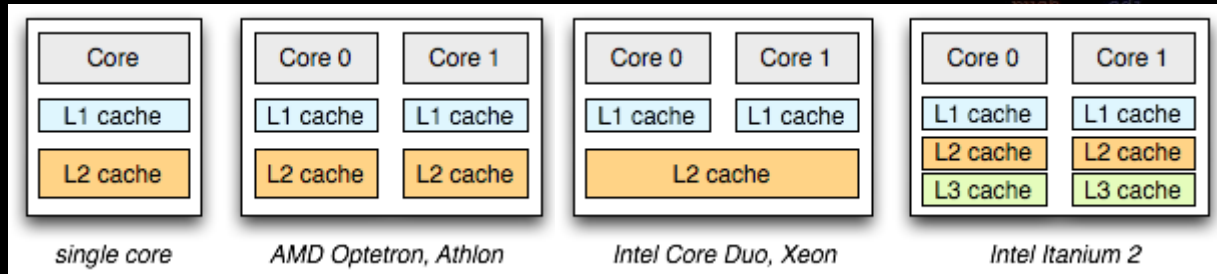


Intel's Core Duo, Xeon Architecture

1. Each processor has two cores
2. The Xen hypervisor schedules between all processors on a server
3. Each core then allocates processes on its pipeline

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push esi
; eax
loc_31306D:
; CODE XREF: sub_312FD8+55
; sub_312FD8+55
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
; CODE XREF: sub_312FD8+49
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D:
; CODE XREF: sub_312FD8+49
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8+49
mov [ebp+var_4], eax
```

Hardware Architectures (2)



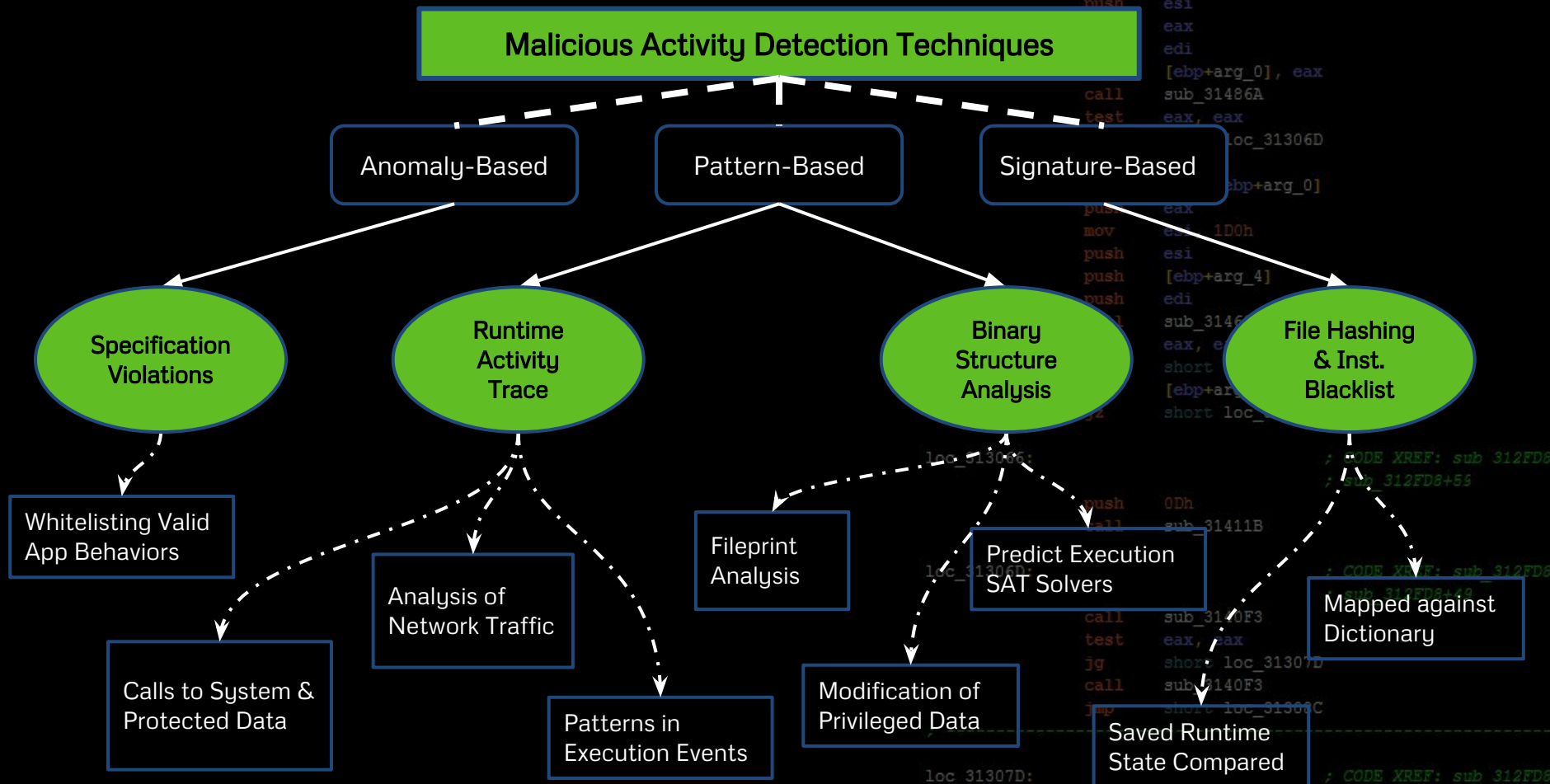
Modern Computation

=> Multiple processes run on a pipeline (**SMT**)

=> Relaxed memory model

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp var_84, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push esi
; eax
loc_31306D:
; eax
arg_0]
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
mov esi, [ebp+arg_0]
jz short loc_31308F
loc_313066:
; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
loc_31306D:
; CODE XREF: sub_312FD8
; sub_312FD8+49
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
; -----
loc_31307D:
; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```

Classifying Channel Detection

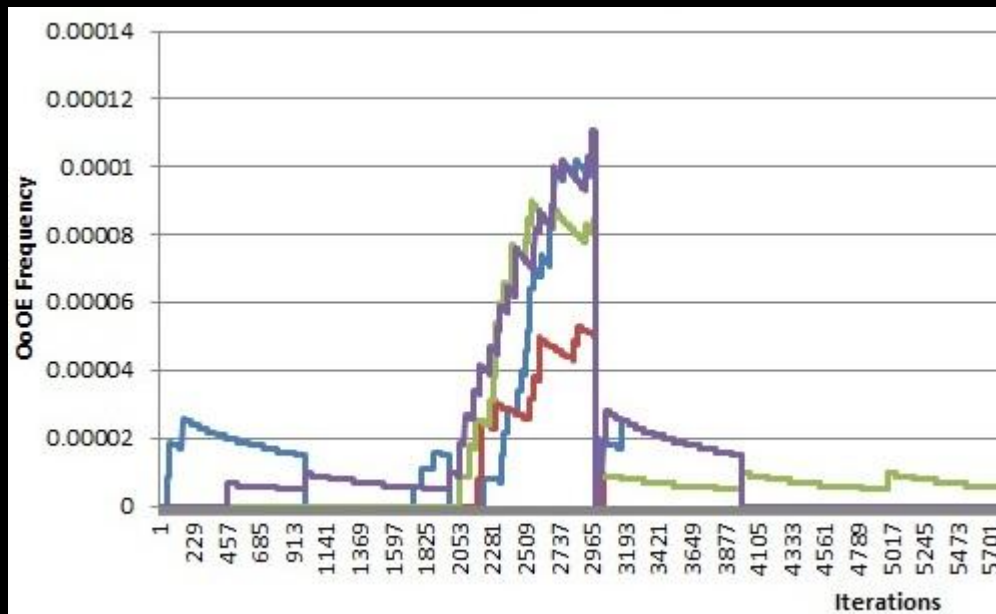


```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
mov [ebp+var_84], ebx
mov eax, [ebp+var_84]
mov [ebp+var_84], eax
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
eax
edi
[ebp+arg_0], eax
call sub_31486A
test eax, eax
loc_31306D
[ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
sub_3146
eax, e
short
[ebp+arg_
short loc_
loc_313066:
; CODE XREF: sub_312FD8
; sub_312FD8+55
push 0Dh
call sub_31411B
loc_31306D:
; CODE XREF: sub_312FD8
; sub_312FD8+22
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31306C
loc_31307D:
; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C:
; CODE XREF: sub_312FD8
mov [ebp+var_4], eax
```


Potential Channel Detection (1)

Signature

- Changes in the **signature** of a hardware unit **over time**



```
loc_31306D:
    push    edi
    call   sub_314623
    test   eax, eax
    jz     short loc_31306D
    cmp    [ebp+arg_0], ebx
    mov    eax, [ebp+var_70]
    jb     short loc_313066
    sub    eax, [ebp+var_84]
    push   esi
    push   esi
    push   eax
    push   edi
    mov    [ebp+arg_0], eax
    call  sub_31486A
    test   eax, eax
    jz     short loc_31306D
    push   esi
    mov    esi, 1D0h
    push   esi
    push   [ebp+arg_4]
    push   edi
    call  sub_314623
    test   eax, eax
    jz     short loc_31306D
    cmp    [ebp+arg_0], esi
    jz     short loc_31308F
    loc_31306F:
    push   0Dh
    call  sub_31411B
    loc_31306D:
    call  sub_3140F3
    test   eax, eax
    jg     short loc_31307D
    call  sub_3140F3
    jmp   short loc_31308C
;
loc_31307D:
    call  sub_3140F3
    and   eax, 0FFFFFFh
    or    eax, 80070000h
loc_31308C:
    mov   [ebp+var_4], eax
```

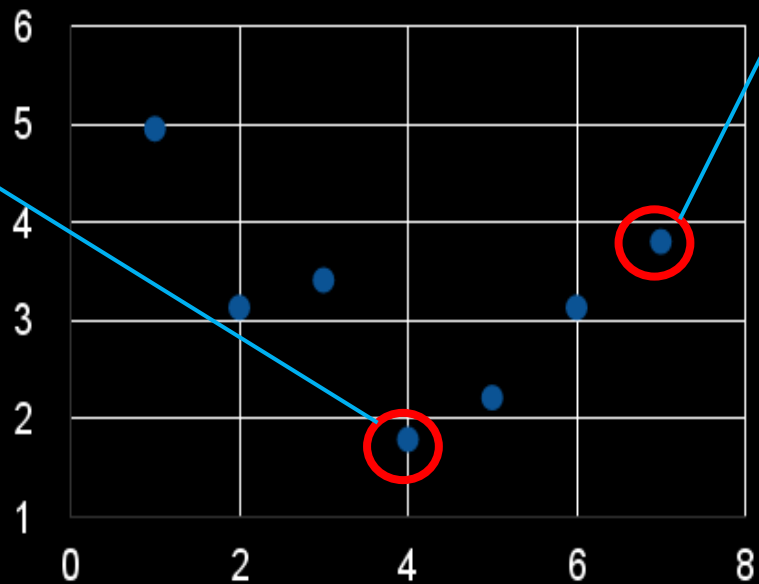
Classification of Intent

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push eax
mov [ebp+var_0], eax
xor [ebp+var_4], eax
xor [ebp+var_4], eax
loc_31306D:
mov [ebp+arg_0], esi
loc_31308F:
; CODE XREF: sub_312FD8
; sub_312FD8+55
111B:
; CODE XREF: sub_312FD8
; sub_312FD8+49
10F3:
xor [ebp+var_0], esi
loc_31307D:
10F3:
loc_31308C:
loc_31307D:
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
loc_31308C:
mov [ebp+var_4], eax
```

broadcast
signal

sec

Avg Time per 4bits



bi-way
communication

attack implementation

Potential Channel Detection (2)

Anomaly

- Specification
- Pattern recognition
- Records average OoOE patterns
- Predicts what to expect

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
mov eax, [ebp+var_70]
jg short loc_313066
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
mov [ebp+arg_0], eax
call sub_31486A
test eax, eax
jz short loc_31306D
push esi
lea eax, [ebp+arg_0]
push eax
mov esi, 1D0h
push esi
push [ebp+arg_4]
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], esi
jz short loc_31308F
```

```
loc_313066: ; CODE XREF: sub_312FD8
; sub_312FD8+55
```

```
push 0Dh
call sub_31411B
```

```
loc_31306D: ; CODE XREF: sub_312FD8
; sub_312FD8+49
```

```
call sub_3140F3
test eax, eax
jg short loc_31307D
call sub_3140F3
jmp short loc_31308C
```

```
loc_31307D: ; CODE XREF: sub_312FD8
```

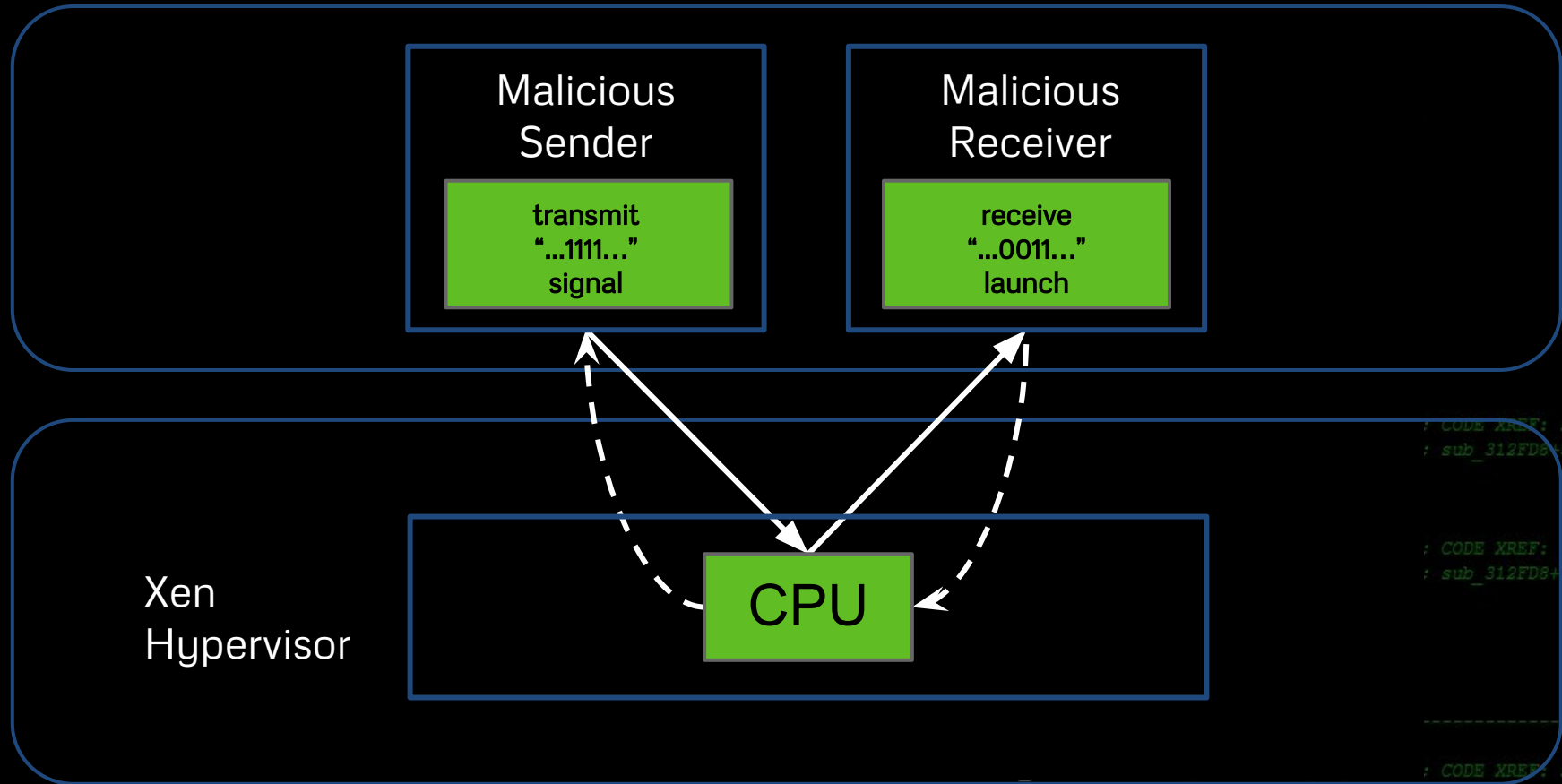
```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C: ; CODE XREF: sub_312FD8
```

```
mov [ebp+var_4], eax
```

Communication of a Malicious Process

```
push edi
call sub_314623
xor eax, eax
mov eax, loc_313066
cmp [ebp+var_84], eax
jnz short loc_313066
mov eax, [ebp+var_70]
cmp eax, [ebp+var_84]
jb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
```



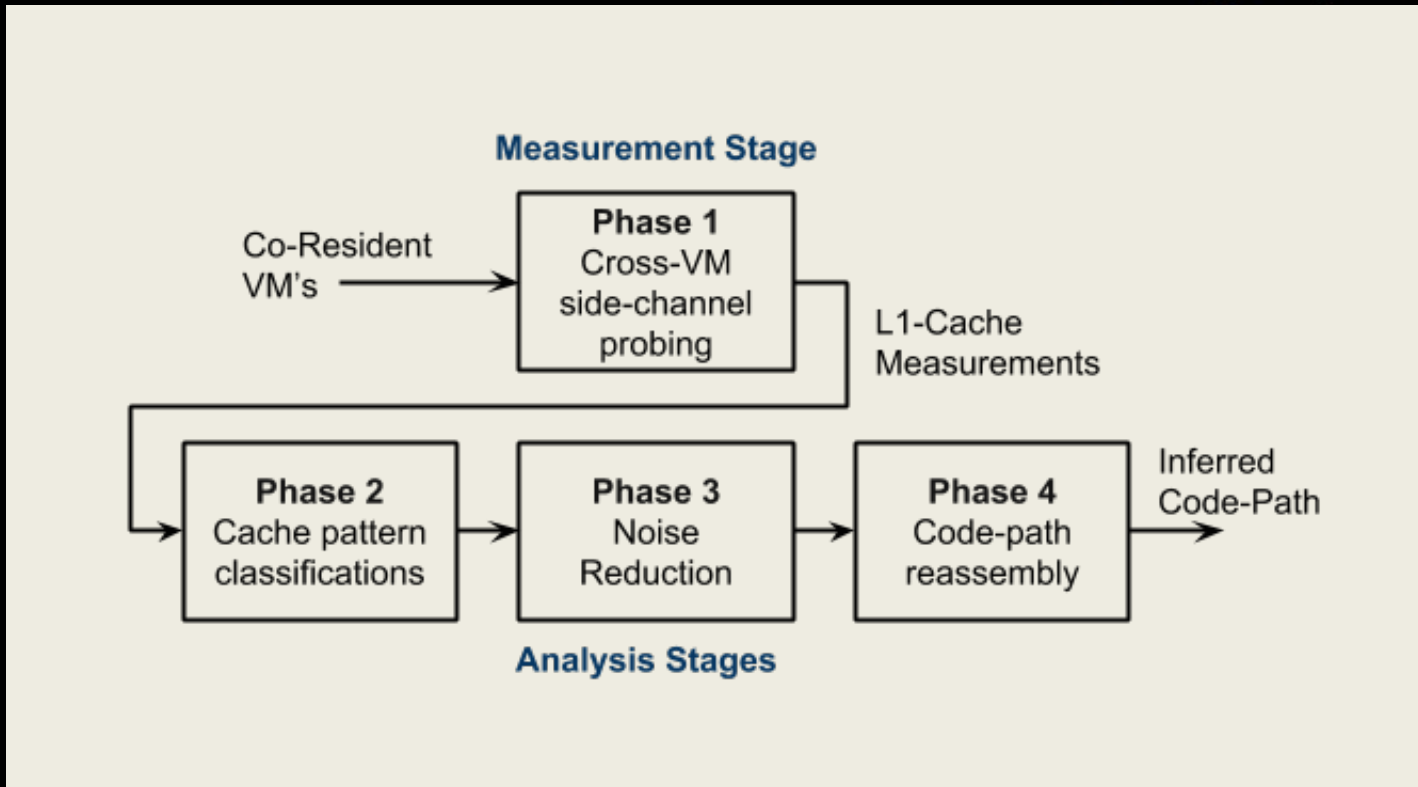
```
: CODE XREF: sub_312FD8
: sub_312FD8+55
: CODE XREF: sub_312FD8
: sub_312FD8+49
: CODE XREF: sub_312FD8
```

```
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
```

```
loc_31308C:
mov [ebp+var_4], eax
: CODE XREF: sub_312FD8
```

Stages of Side Channel Attack

```
push edi
call sub_314623
test eax, eax
jz short loc_31306D
cmp [ebp+arg_0], ebx
jnb short loc_313066
mov eax, [ebp+var_70]
leap eax, [ebp+var_4]
jnb short loc_313066
sub eax, [ebp+var_84]
push esi
push esi
push eax
push edi
```



```
CODE XREF: sub_312FD8
sub_312FD8+55
CODE XREF: sub_312FD8
sub_312FD8+49
```

Example applied to L1 cache side channel

```
loc_31308C:
; CODE XREF: sub_312FD8
call sub_3140F3
and eax, 0FFFFFFh
or eax, 80070000h
mov [ebp+var_4], eax
```

Dynamic Differences

- **Dynamic** allocations of physical resources
- Force artifacts on the **shared hardware**
- **Reception** of these artifacts
 - Querying the specific hardware unit
 - **Difficulty/ reliability** unique to each hardware unit.

```
push    edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], ebx
jnz   short loc_313066
mov    eax, [ebp+var_70]
cmp    eax, [ebp+var_84]
jb     short loc_313066
sub    eax, [ebp+var_84]
push   esi
push   esi
push   eax
push   esi
call   sub_31486A
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], esi
jnz   short loc_313066
push   eax
mov    esi, 1D0h
push   esi
push   [ebp+arg_4]
push   edi
call   sub_314623
test   eax, eax
jz     short loc_31306D
cmp    [ebp+arg_0], esi
jz     short loc_31308F
loc_313066:                                     ; CODE XREF: sub_312FD8
; sub_312FD8+55
push   0Dh
call   sub_31411B
loc_31306D:                                     ; CODE XREF: sub_312FD8
; sub_312FD8+49
call   sub_3140F3
test   eax, eax
jg     short loc_31307D
call   sub_3140F3
jmp    short loc_31308C
; -----
loc_31307D:                                     ; CODE XREF: sub_312FD8
call   sub_3140F3
and    eax, 0FFFFFFh
or     eax, 80070000h
loc_31308C:                                     ; CODE XREF: sub_312FD8
mov    [ebp+var_4], eax
```