FORESHADOW

Breaking the Virtual Memory Abstraction with Transient Out-of-Order Execution

Ofir Weisse

Joint work with

Jo Van Bulck, Marina Minkin, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, Raoul Strackx

BBC NEWS

Technology

PM

FORESHADOW

Breaking the Virtual Memory Abstraction with Transient Out-of-Order Execution

```
* Read about Ubuntu updates for L1 Terminal Fault Vulnerabilities
  (L1TF).

  - https://ubu.one/L1TF
```

'Foreshadow' attack affects Intel chips

s'

PCWorld FROM IDG

NEWS REVIEWS HOW-TO VIDEO

Privacy Encryption Antivirus

AdChoices PC and Laptop

Home / Security

UPDATED

Foreshadow attac tactics (but you're probably safe)

I'M

In e open by, you

g ve execution attack

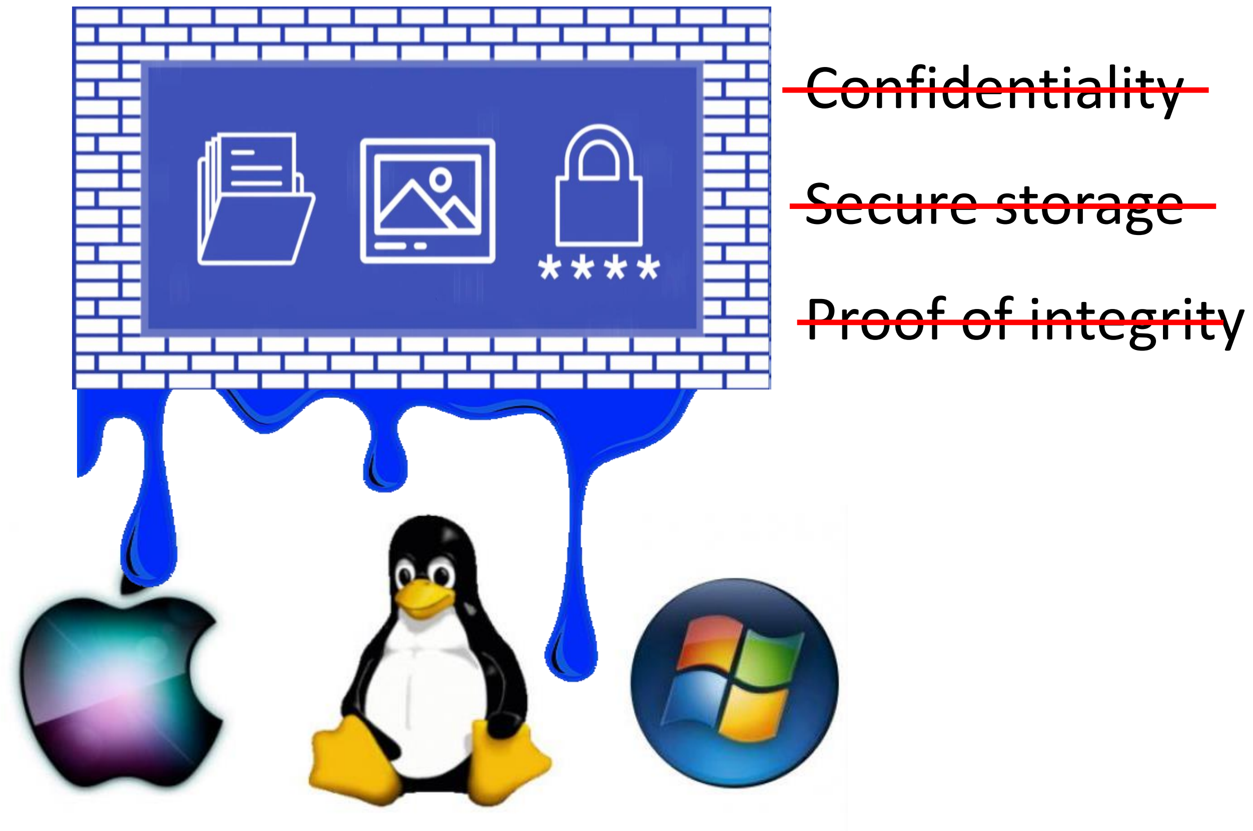
ZDNet

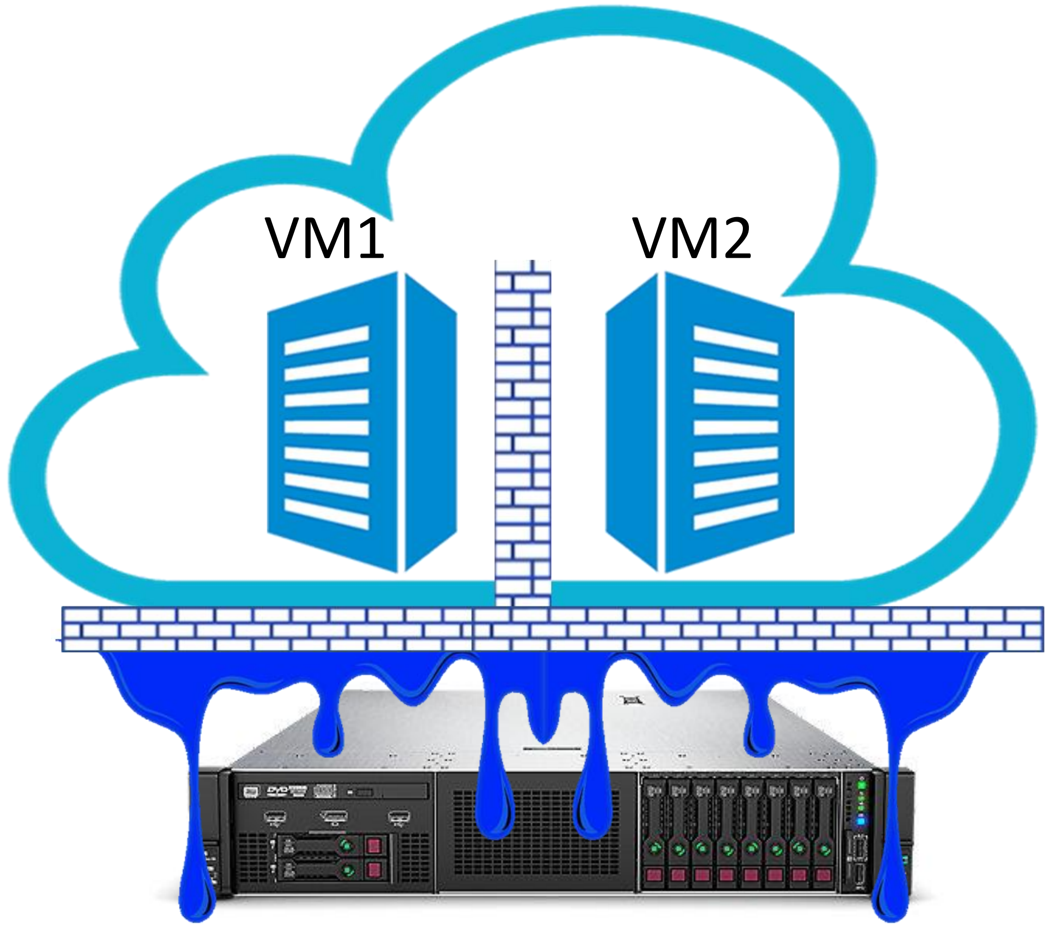
Beyond Spectre: Foreshadow, a new Intel security problem

ForeshadowAttack.com

Foreshadow (SGX)

Foreshadow-NG





Untrusted OS/VMM

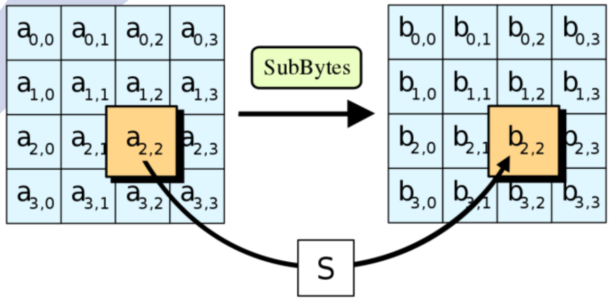
Cloud Host

Evolution of Side Channel Attacks

Foreshadow
(on others' address space)

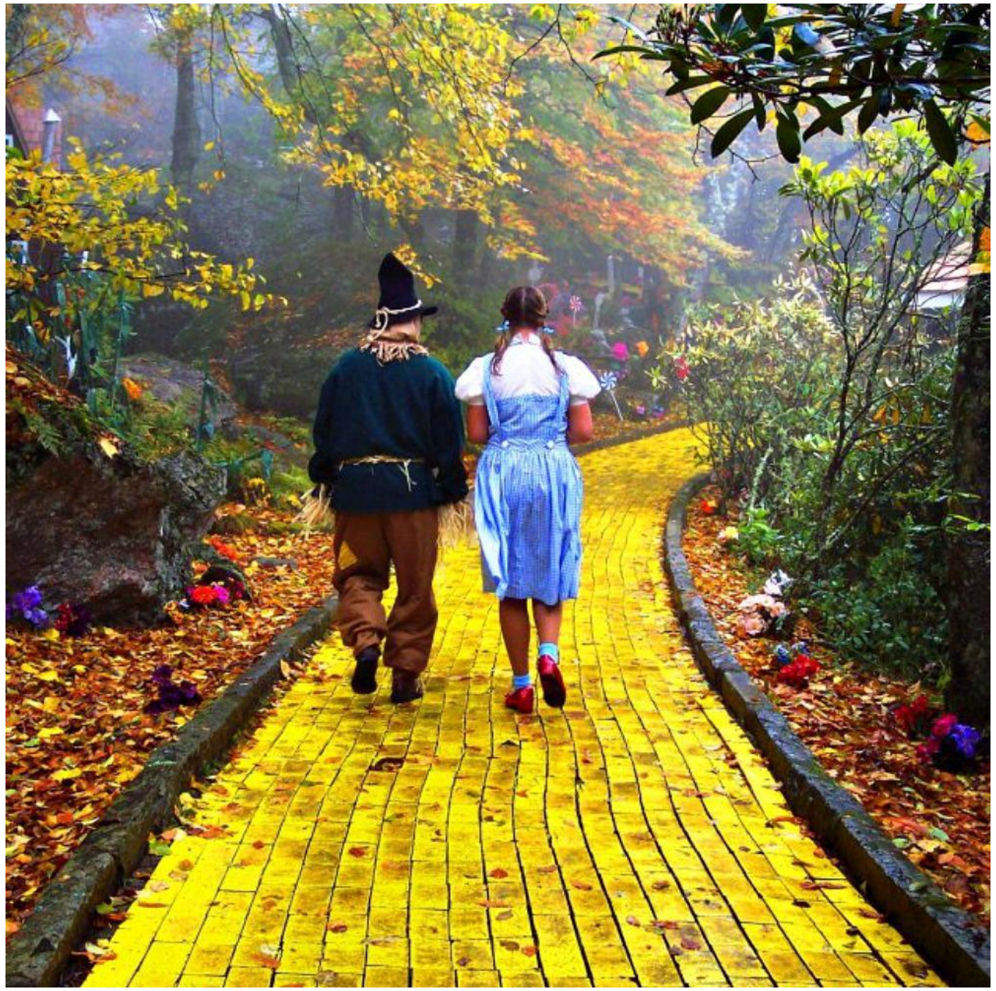
Spectre & Meltdown
(on own address space)

Classic Cache Timing
(Algorithm specific, e.g., AES)


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

Roadmap

- **Cache side channels**
- Speculative execution
- Meltdown
- SGX
- Foreshadow-SGX
- Foreshadow-NG



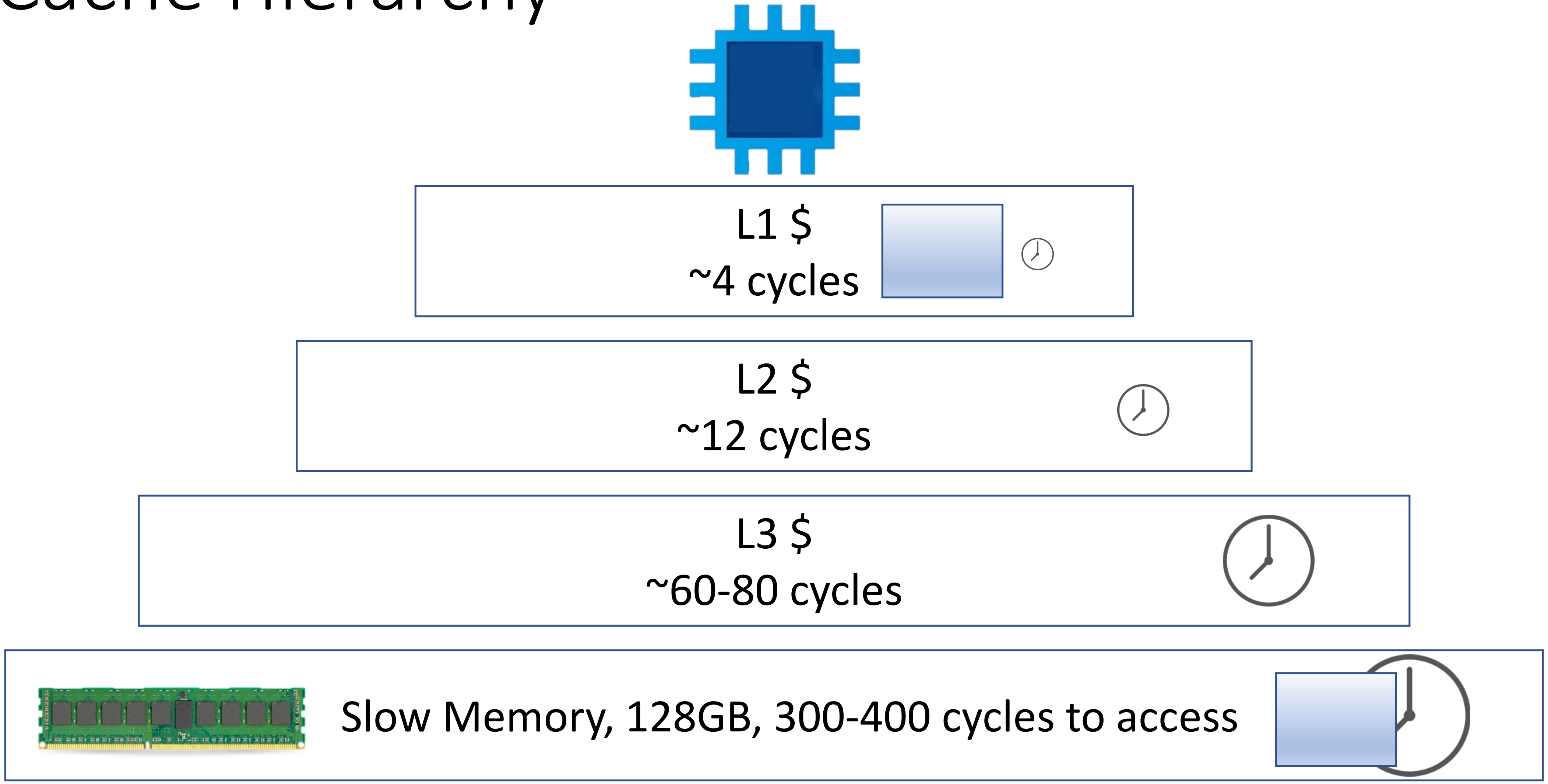
Side Channel Attacks –
Abusing Non-standard Output Channels

Cache Side Channels

intel
IA-32 Intel® Architecture Software Developer's Manual
VOLUME 1: BASIC ARCHITECTURE
intel
IA-32 Intel® Architecture Software Developer's Manual
VOLUME 2A: INSTRUCTION SET REFERENCE, A-M
intel
IA-32 Intel® Architecture Software Developer's Manual
VOLUME 2B: INSTRUCTION SET REFERENCE, N-Z
intel
IA-32 Intel® Architecture Software Developer's Manual
VOLUME 3: SYSTEM PROGRAMMING GUIDE

intel
Intel® 64 and IA-32 Architectures Software Developer's Manual
VOLUME 2B: Instruction Set Reference, N-Z
2B
intel
Intel® 64 and IA-32 Architectures Software Developer's Manual
VOLUME 3A: System Programming Guide Part 1
3A
intel
Intel® 64 and IA-32 Architectures Software Developer's Manual
VOLUME 3B: System Programming Guide Part 2
3B
intel
Intel® 64 and IA-32 Architectures Software Developer's Manual
VOLUME 1: Basic Architecture
1
intel
Intel® 64 and IA-32 Architectures Optimization Reference Manual

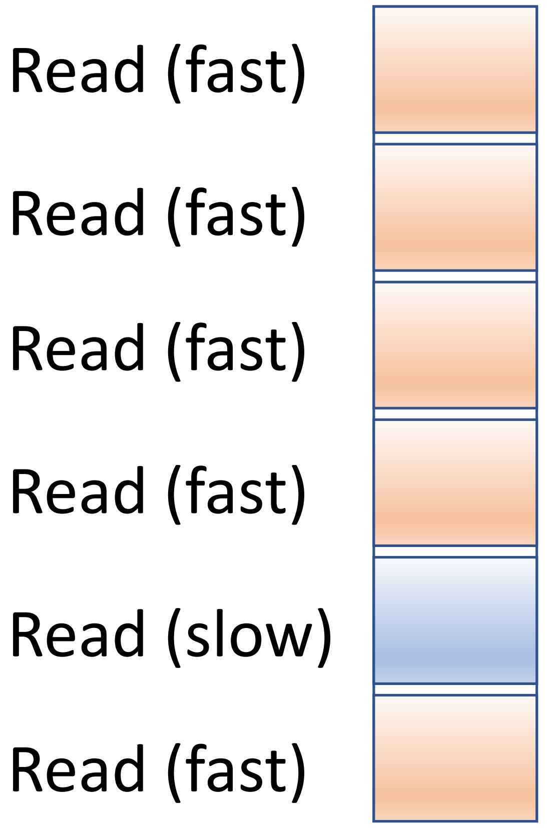
Cache Hierarchy



Background: Cache Timing Side Channel

Attacker

Cache Lines

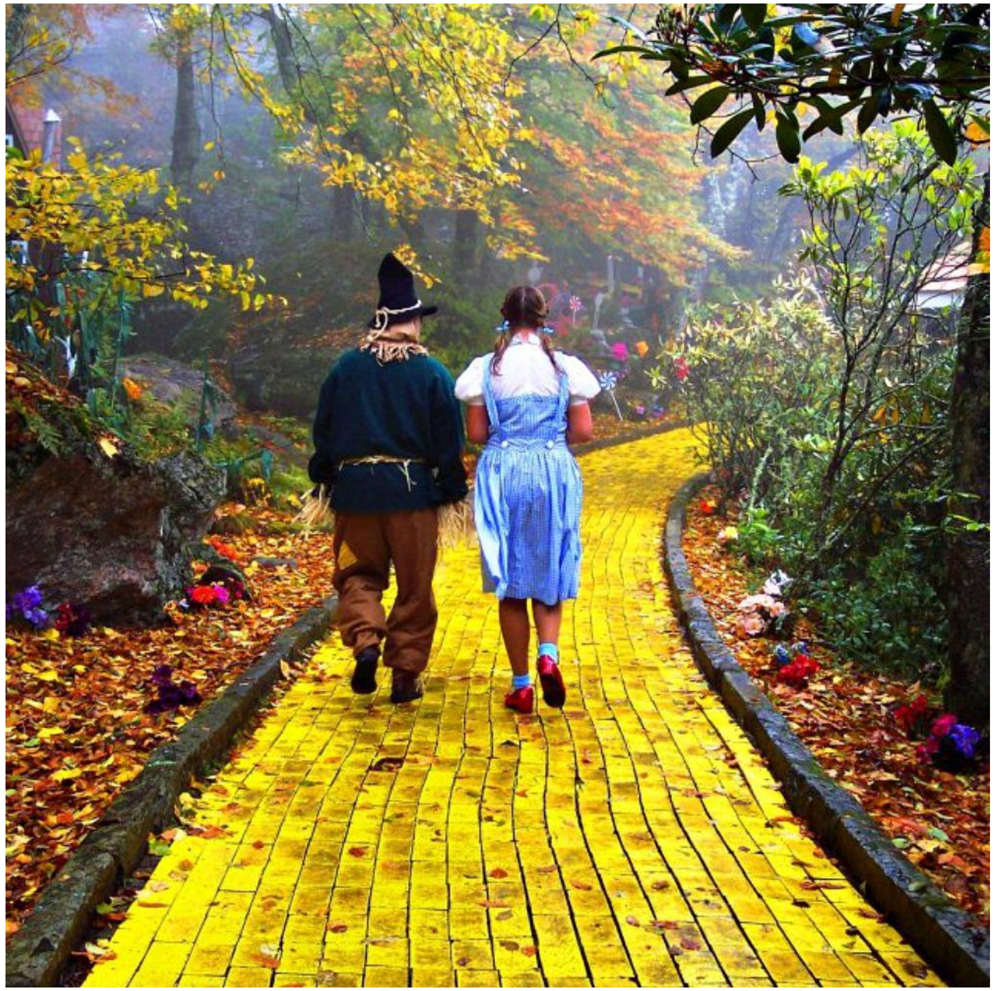


Victim

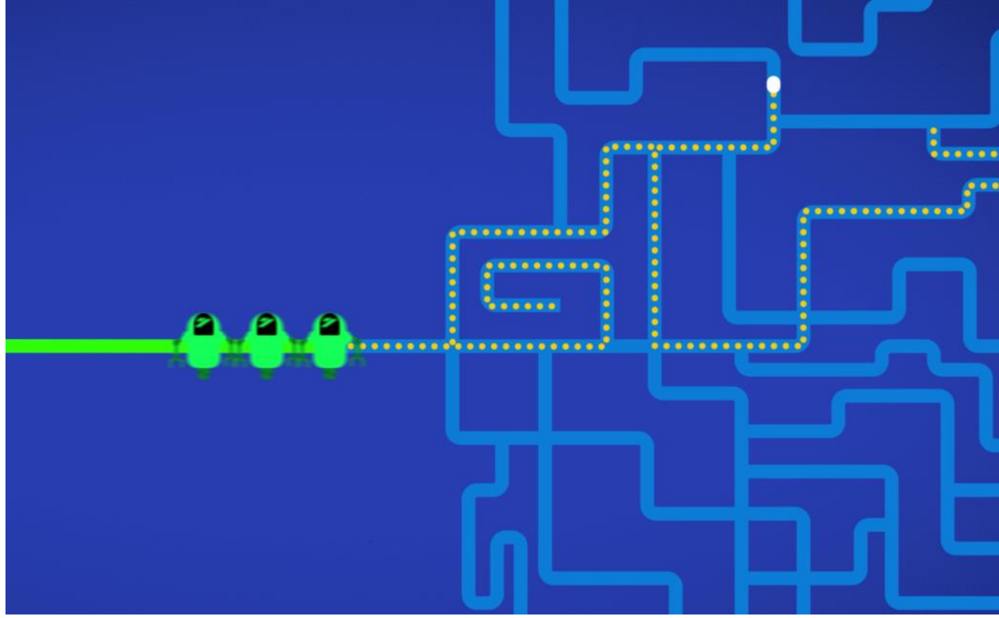
- Attacker infers victim's data access pattern
- Attack is algorithm specific

Roadmap

- Cache side channels
- **Speculative execution**
- Meltdown
- SGX
- Foreshadow-SGX
- Foreshadow-NG



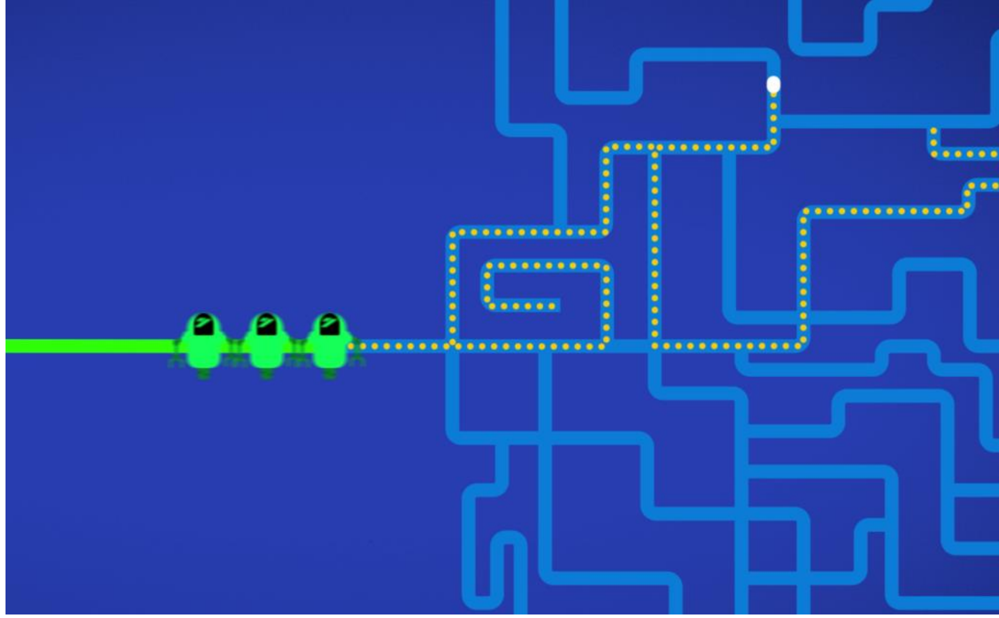
Speculative Execution

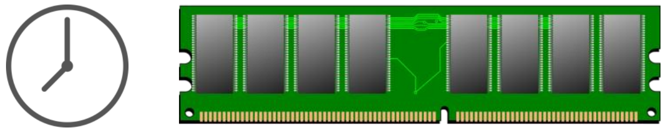

Speculating future tasks

```
data = *user_input;
res = 42 / data;
b -= res;
b++;
c[0] *=2;
d[1] += 42;
```

🟩 Retired instruction

Speculative Execution





Speculating future tasks

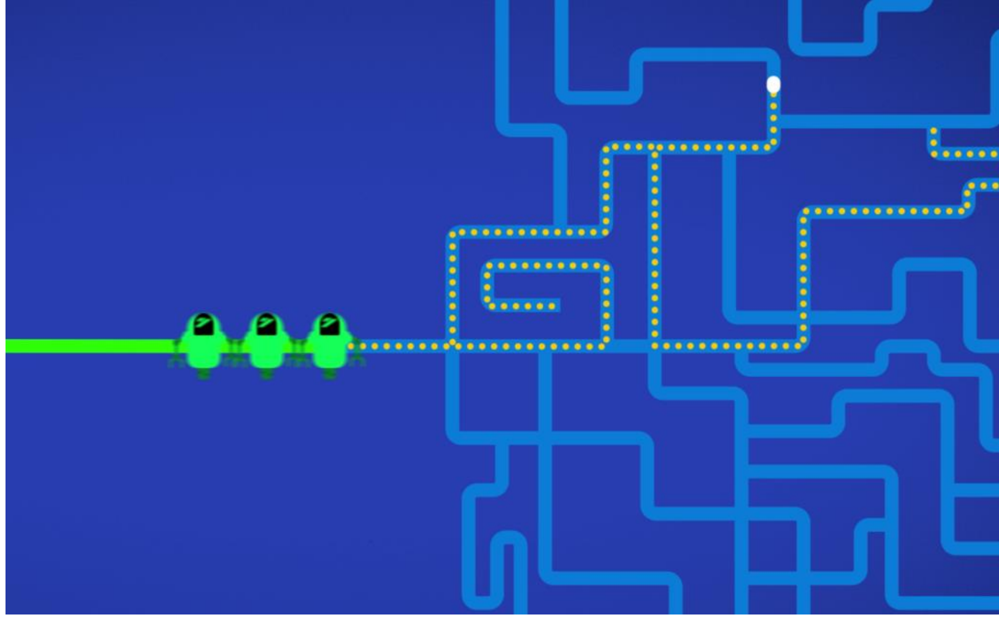
```
data = *user_input;
res = 42 / data;
b -= res;
b++;
c[0] *=2;
d[1] += 42;
```

Speculatively executed

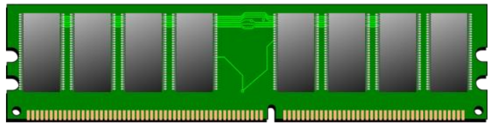
🟩 Retired instruction

🟨 Pending instruction

Speculative Execution



```
*user_input = 0;
data = *user_input;
res = 42 / data;
b -= res;
b++;
c[0] *=2;
d[1] += 42;
```



can never retire!!

Exception handler (division by 0)

Speculatively executed

🟩 Retired instruction

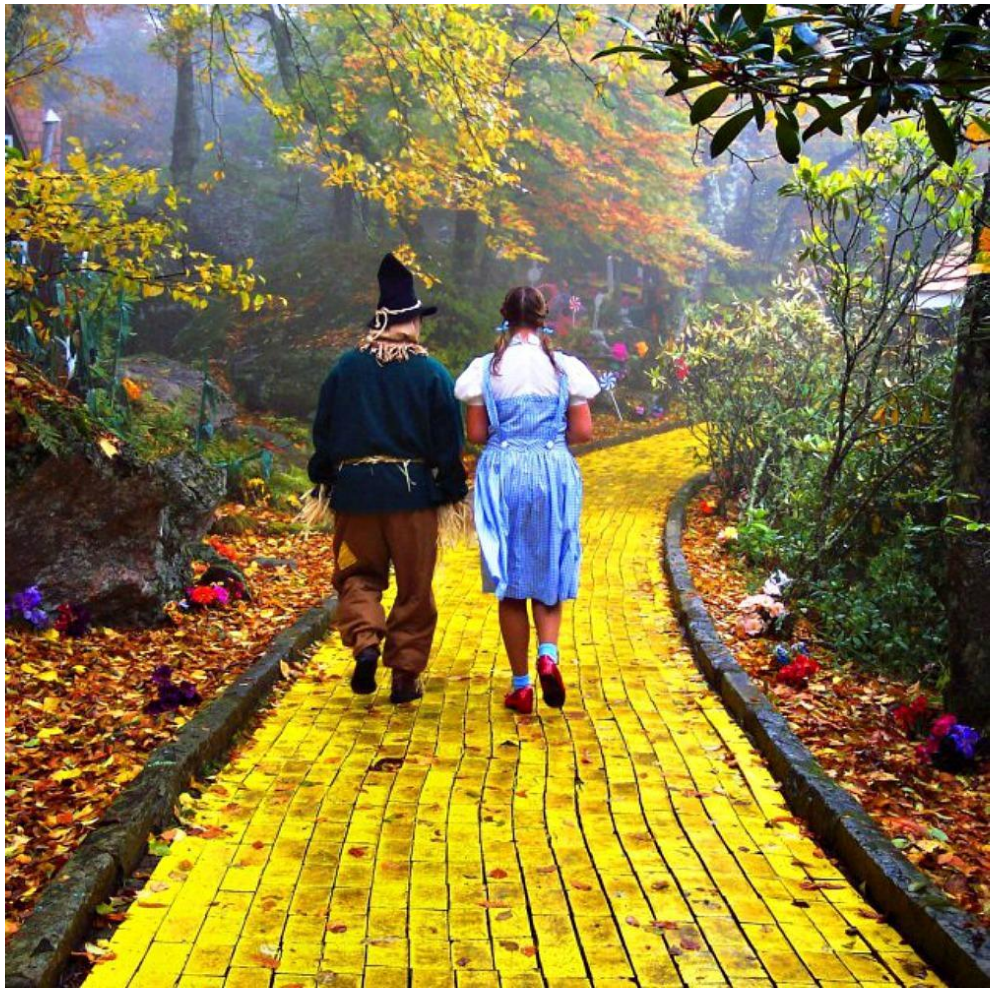
🟨 Pending instruction

⬜ Squashed instruction

Squashed instructions may leave footprints in cache

Roadmap

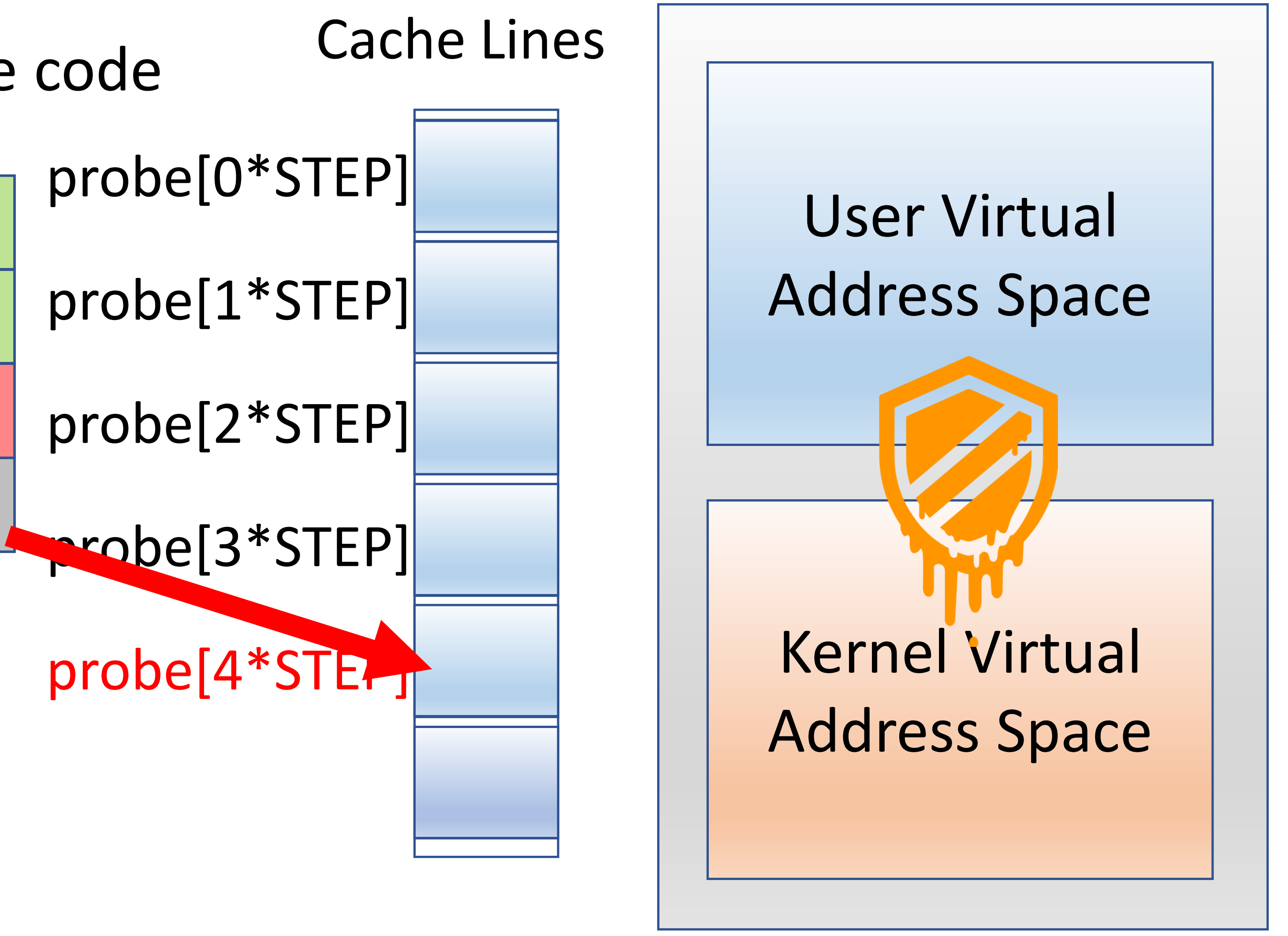
- Cache side channels
- Speculative execution
- **<u>Meltdown</u>**
- SGX
- Foreshadow-SGX
- Foreshadow-NG



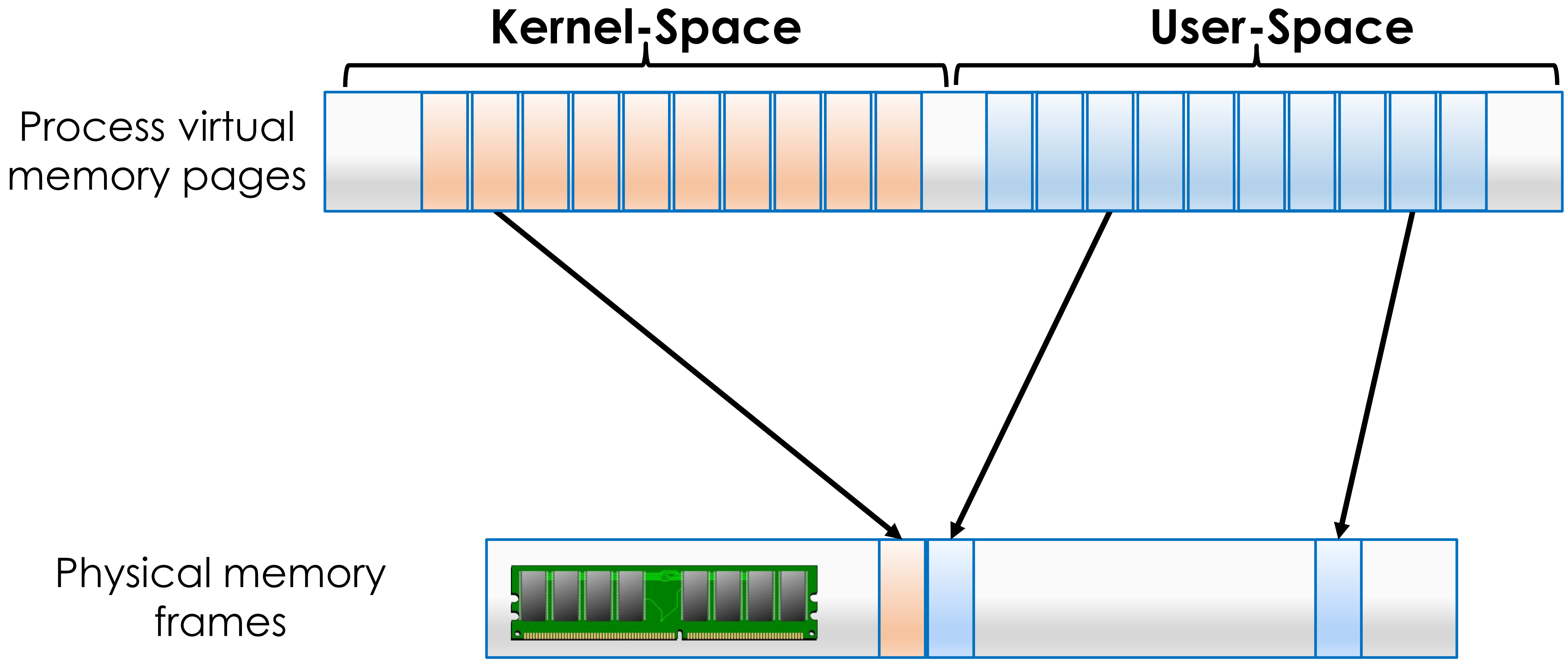
Background: Meltdown

Attacker's user-space code

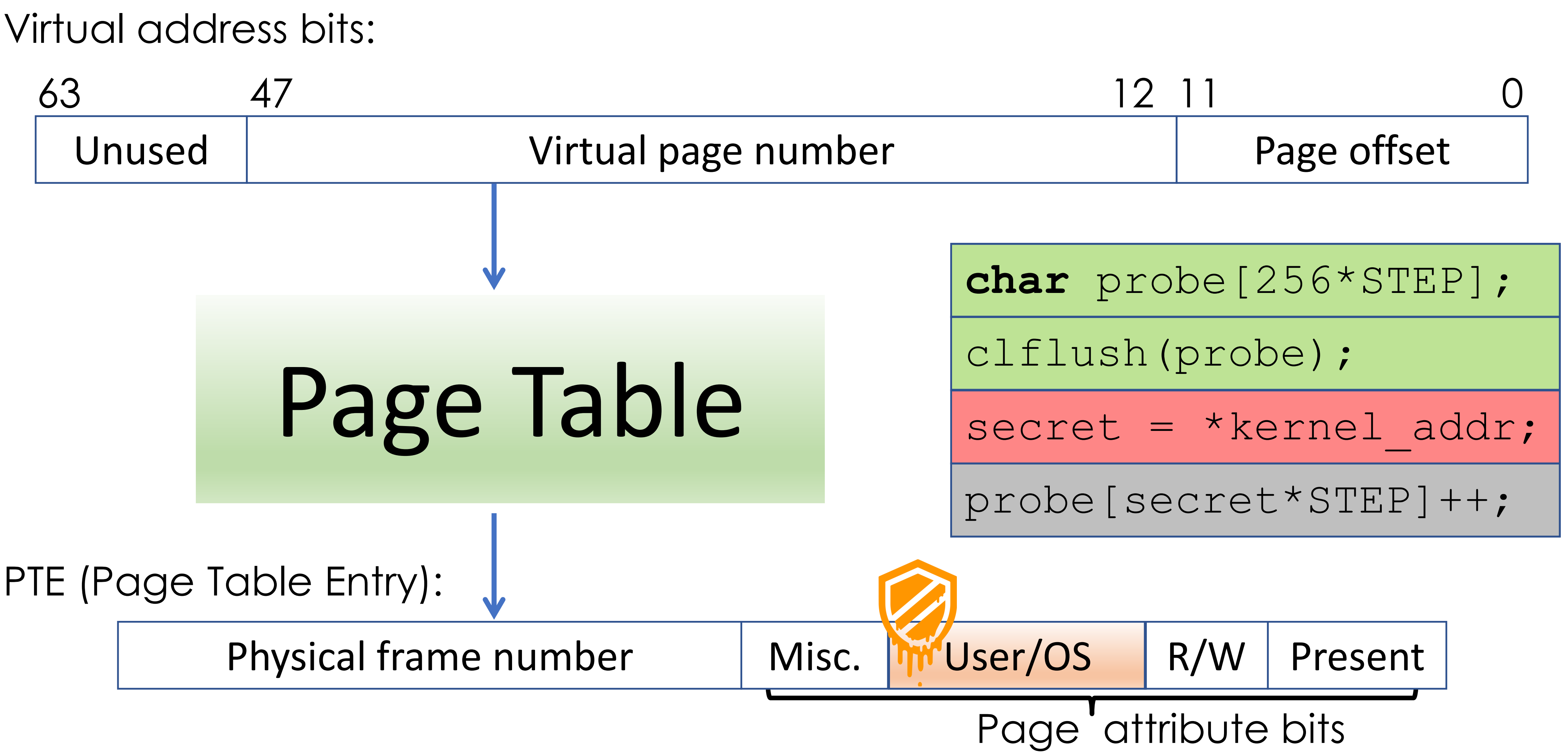
```
char probe[256*STEP];
clflush(probe);
secret = *kernel_addr;
probe[secret*STEP]++;
```



Virtual Address Space

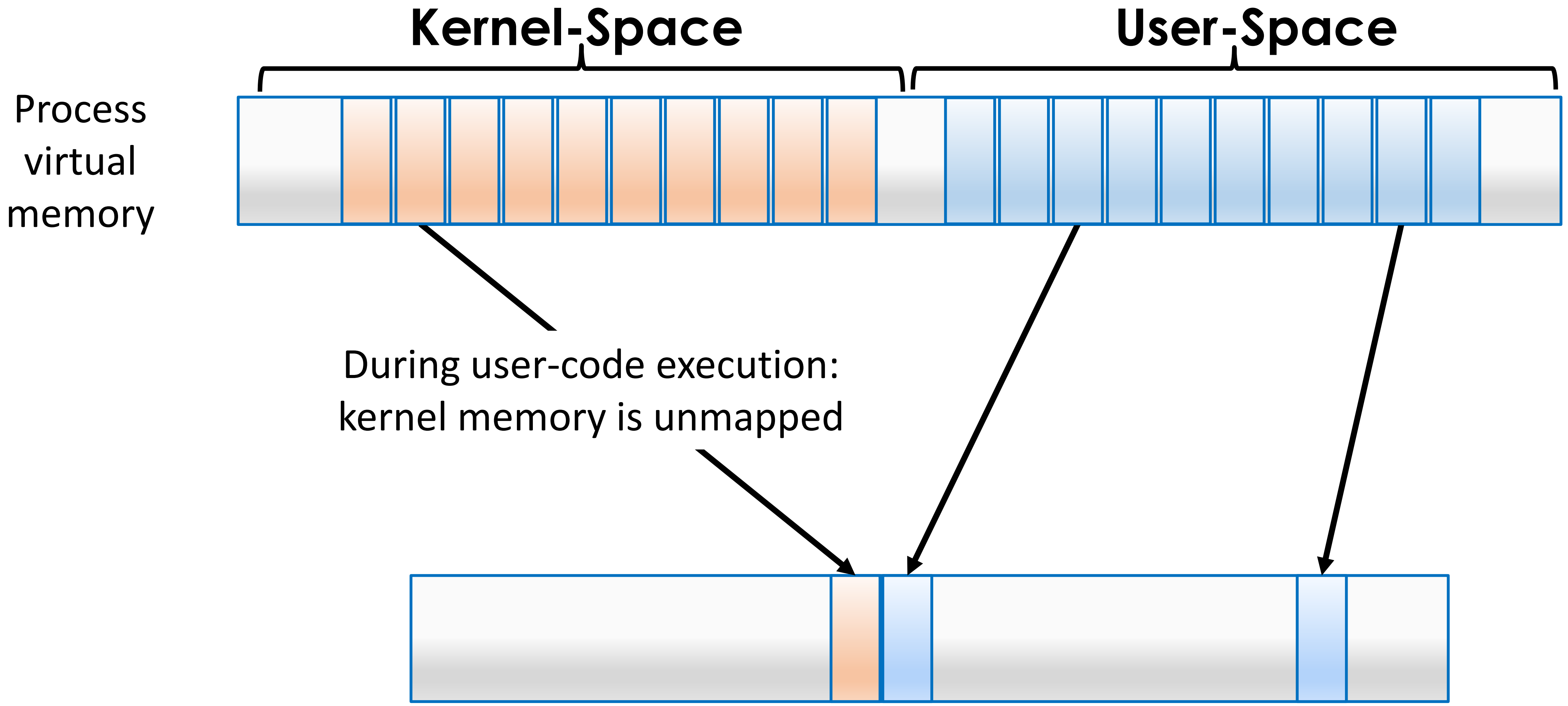


The Page Table



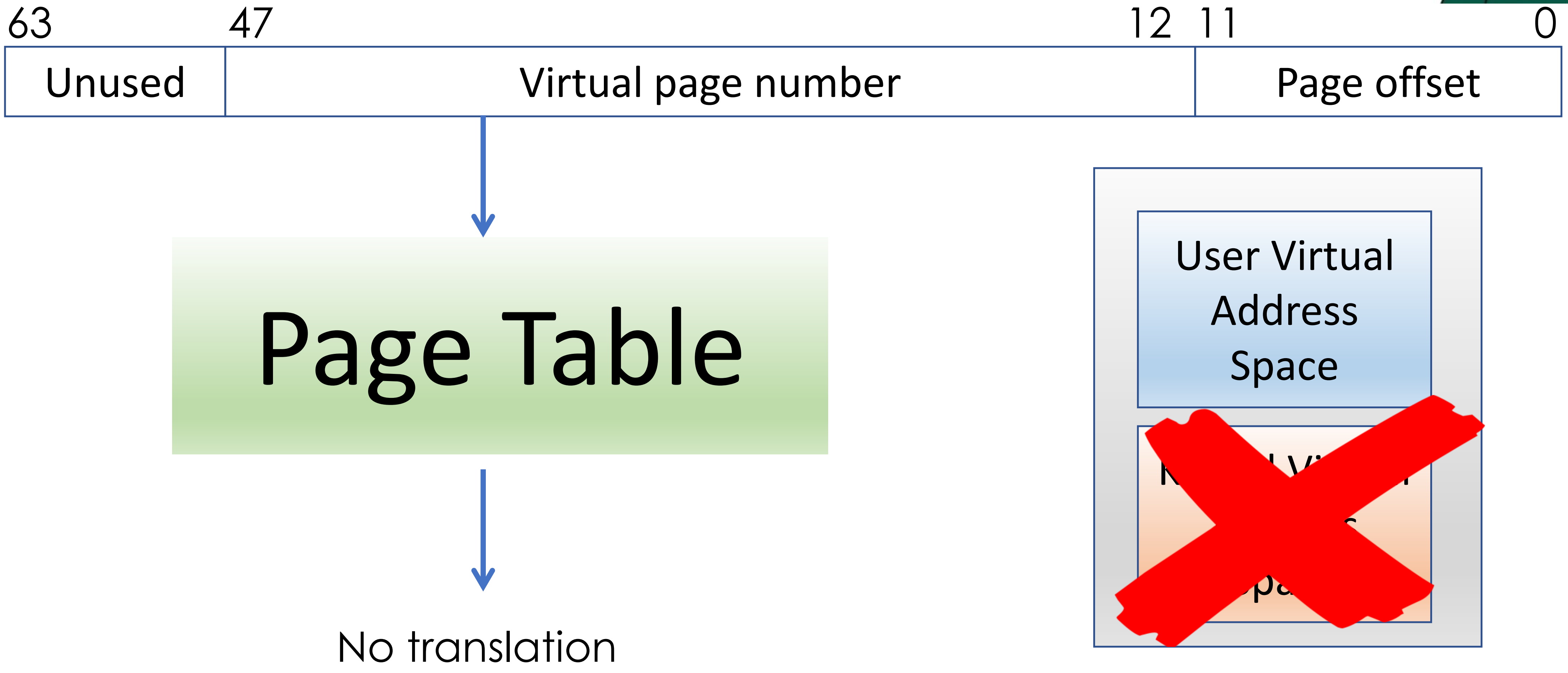
```
char probe[256*STEP];
clflush(probe);
secret = *kernel_addr;
probe[secret*STEP]++;
```

Meltdown Mitigation - KPTI



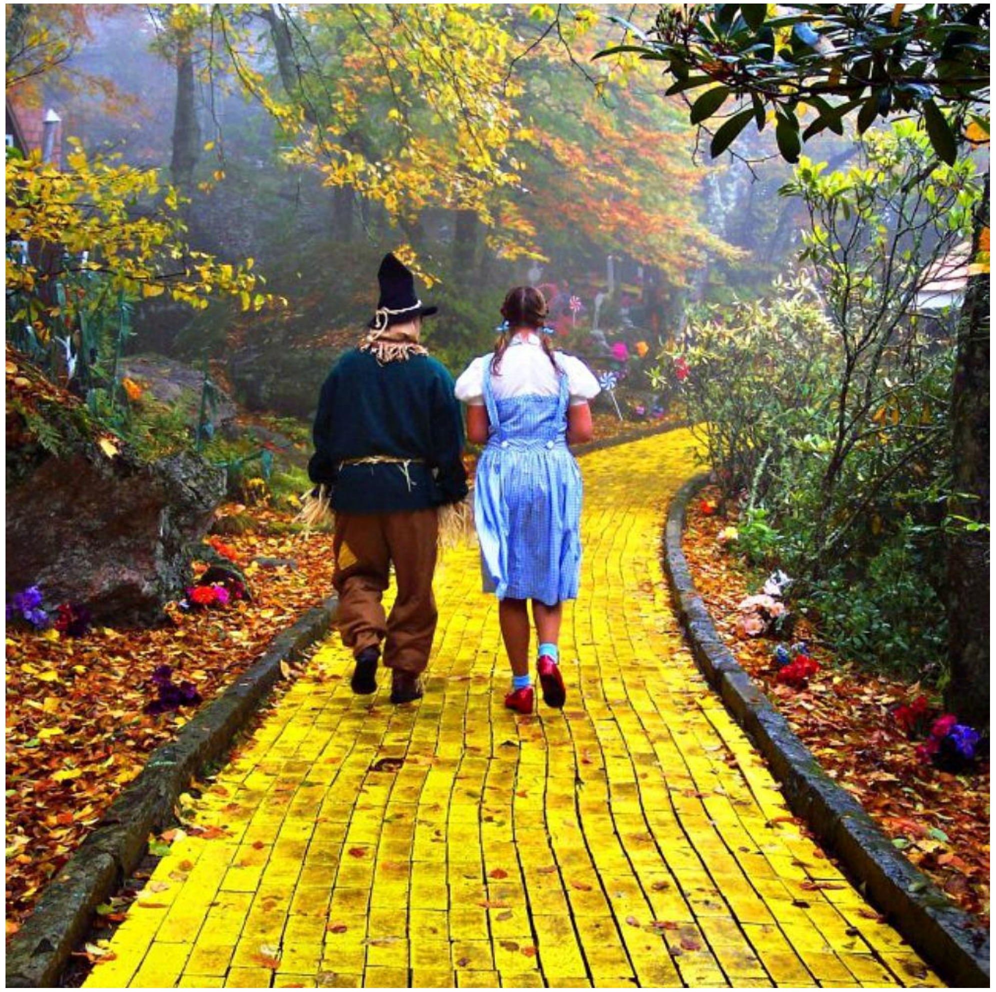
Meltdown Mitigation - KPTI

Virtual address bits:



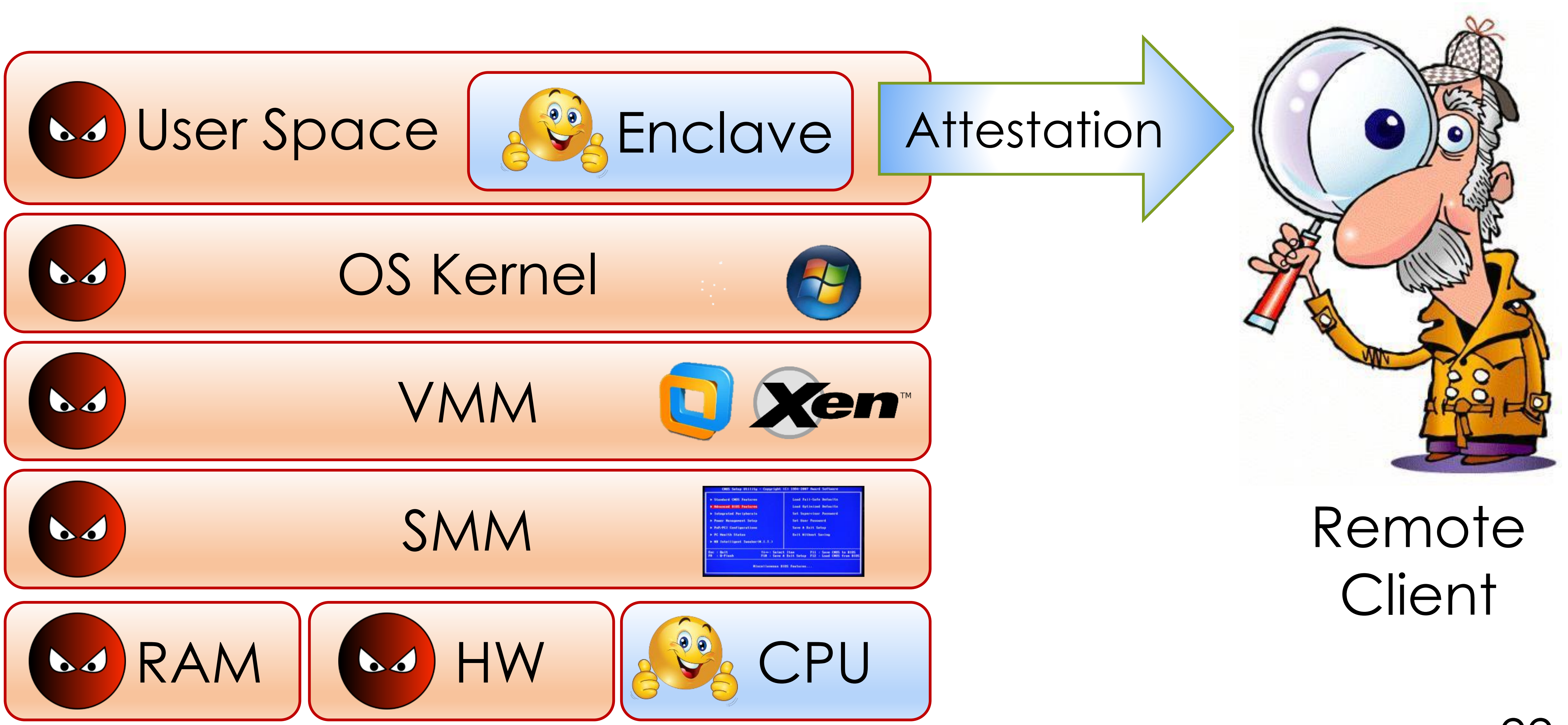
Roadmap

- Cache side channels
- Speculative execution
- Meltdown
- <u>**SGX**</u>
- Foreshadow-SGX
- Foreshadow-NG

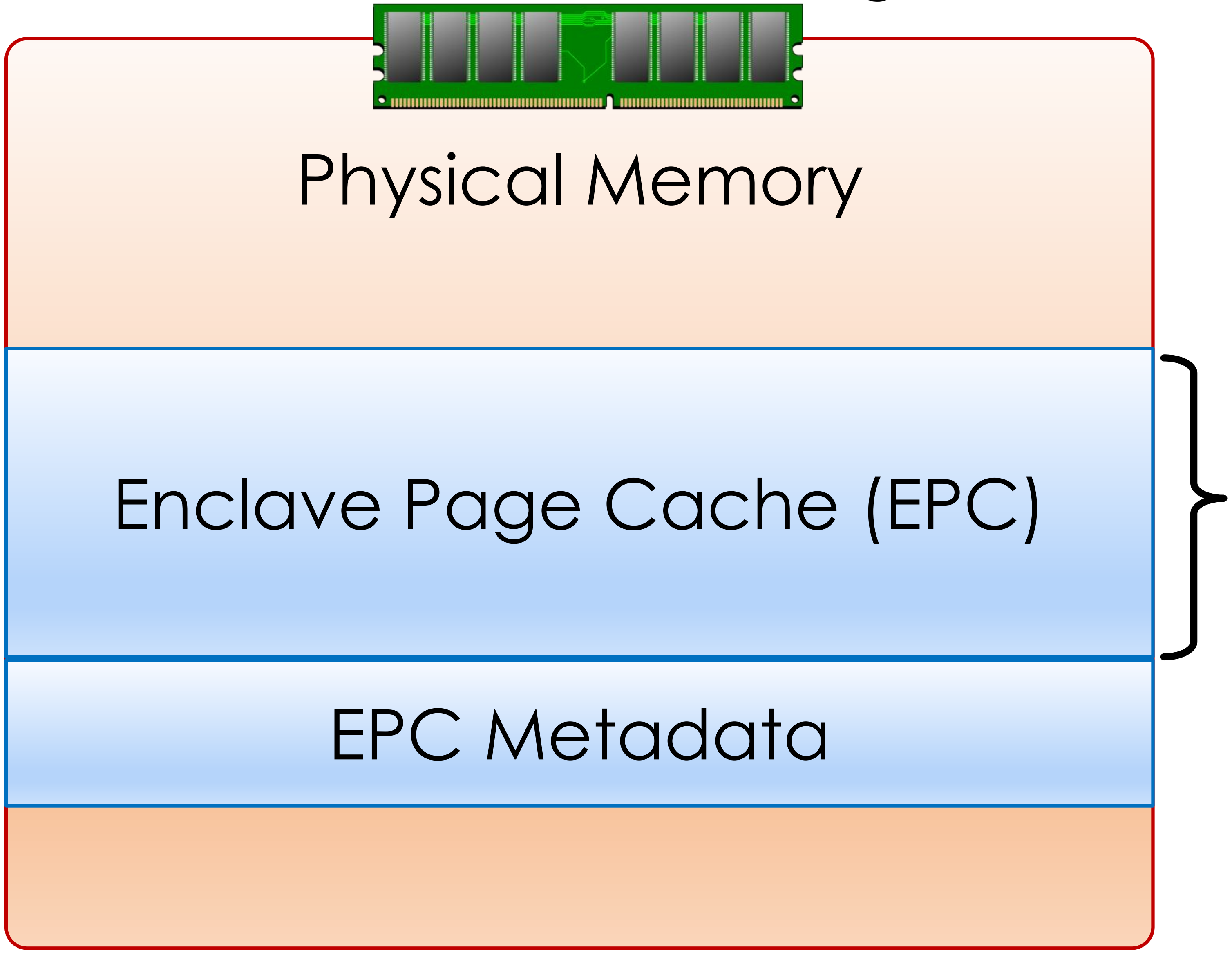


SGX (Software Guard eXtensions)

SGX in a nutshell

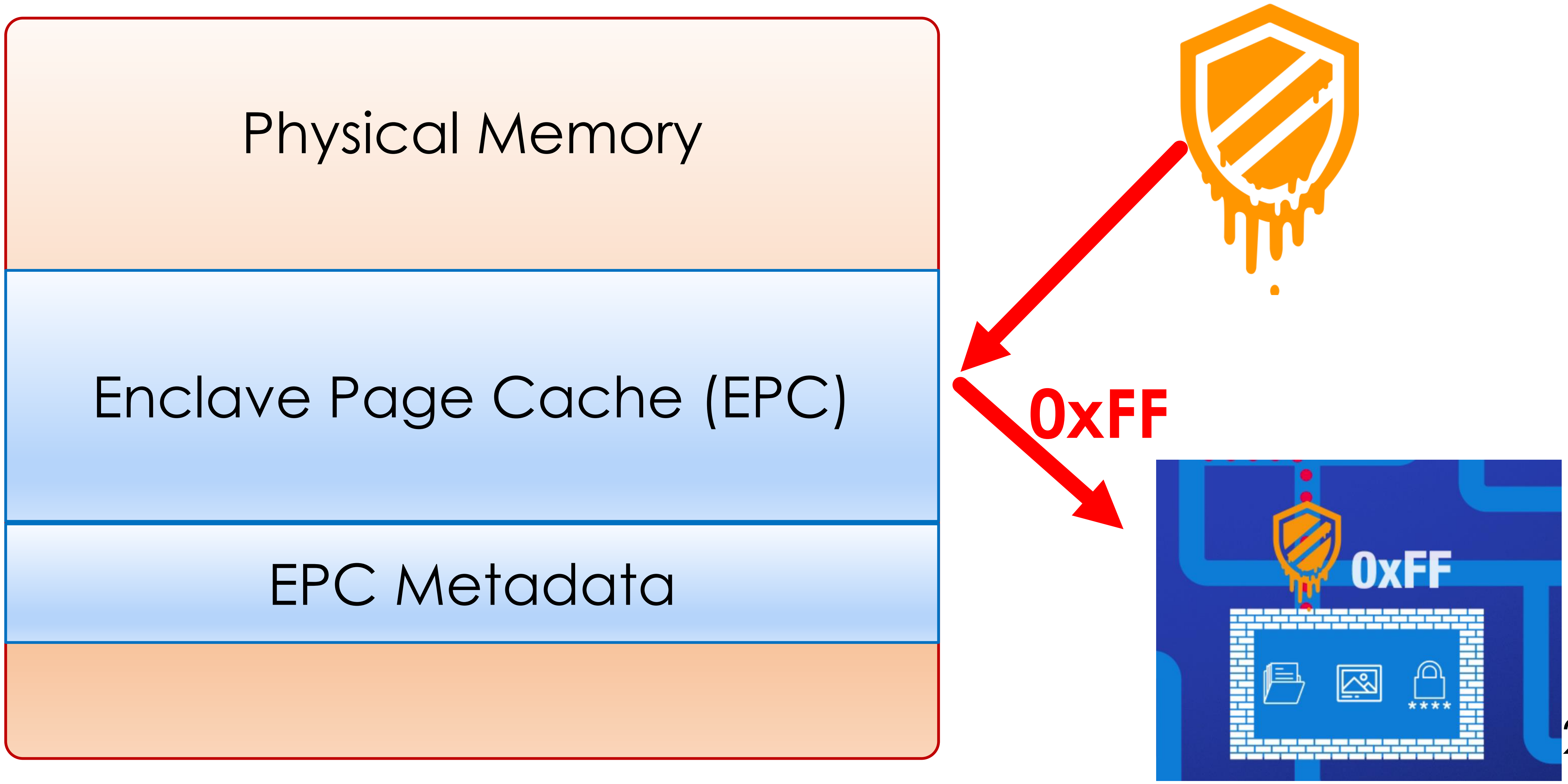


SGX – Memory Organization



Encrypted by
Memory Encryption Engine
(MEE)

SGX Abort Page Semantics



SGX Abort Page Semantics

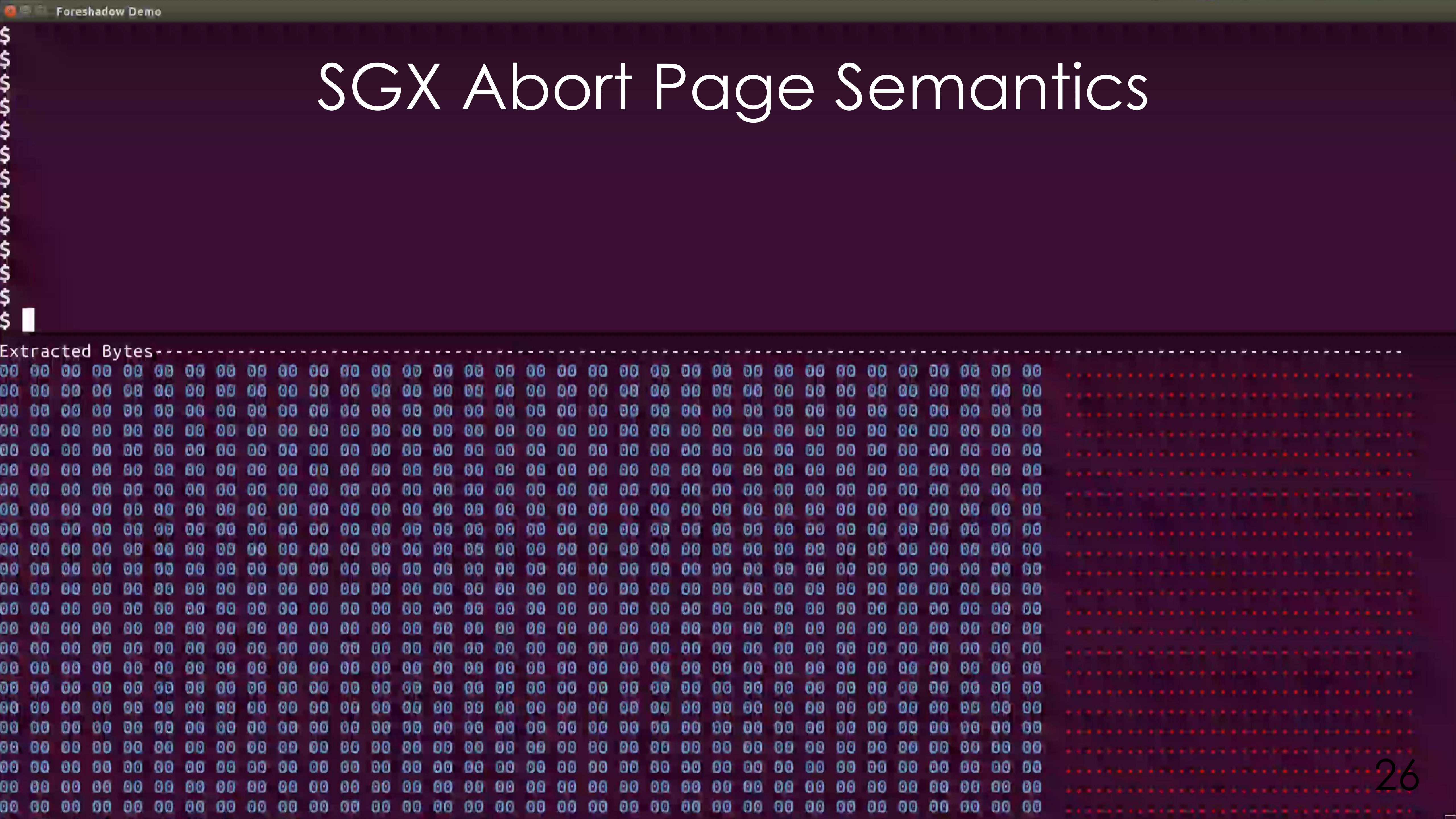
- No exception raised
- Writes are ignored
- Reads return 0xFF

```
char probe[256*STEP];
clflush(probe)
secret = *enclave_addr;
probe[secret*STEP]++
```

→ 0xFF

0xFF

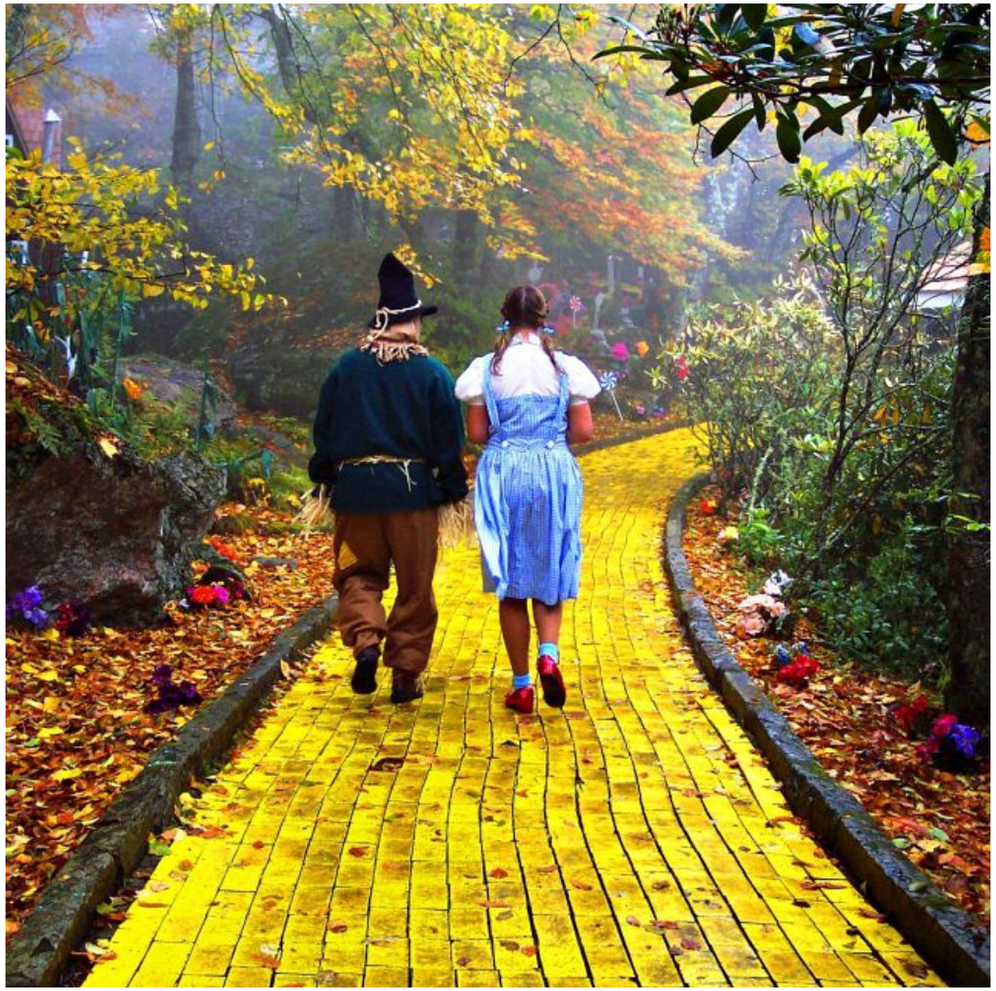
SGX Abort Page Semantics



```
$
$
$
$
$
$
$
$
$
$
$
$
$
Extracted Bytes---------------------------------------------------------------------------------------------------------------
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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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   ..................................
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 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 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 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 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 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 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 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 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 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 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   ..................................
```

Roadmap

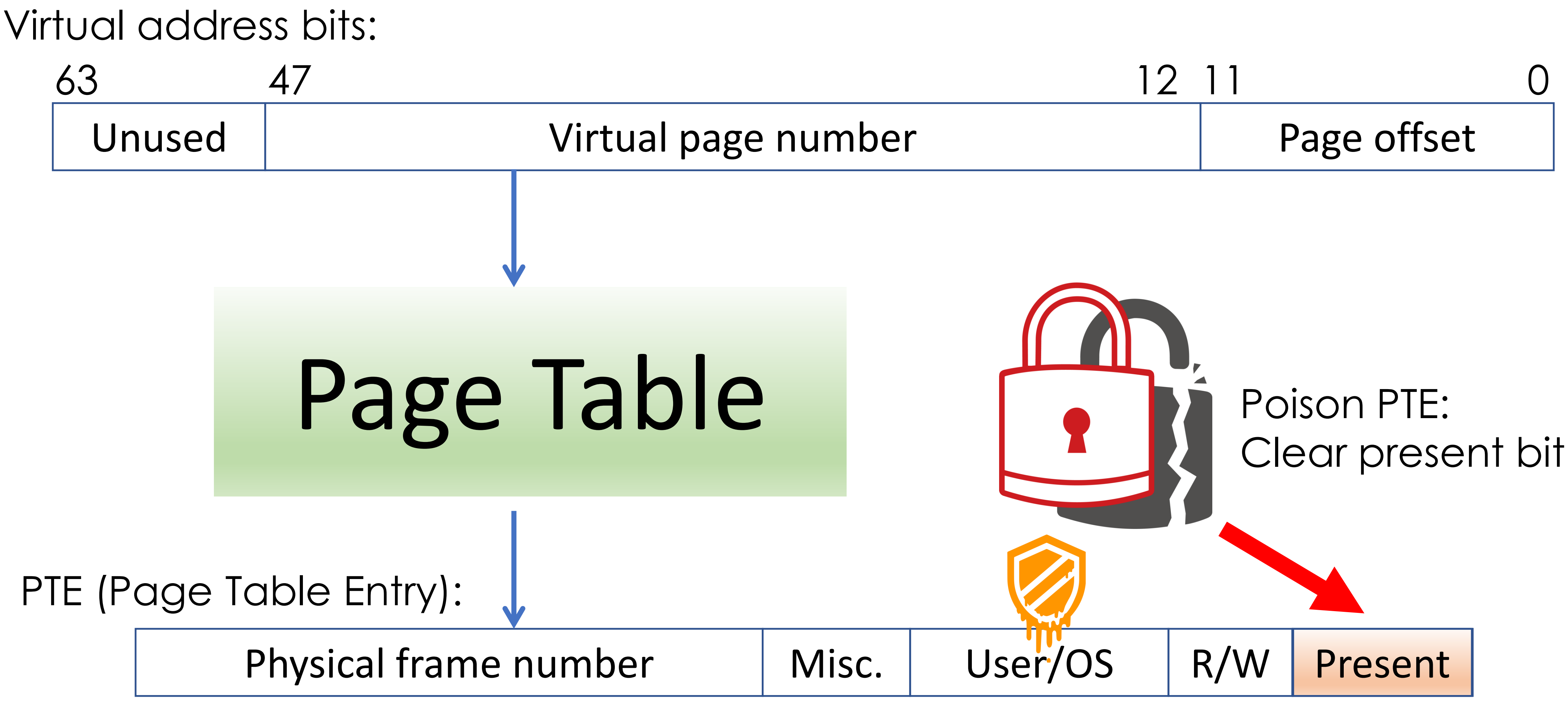
- Cache side channels
- Speculative execution
- Meltdown
- SGX
- **<u>Foreshadow-SGX</u>**
- Foreshadow-NG



Foreshadow –
Causing a Translation Terminal Fault

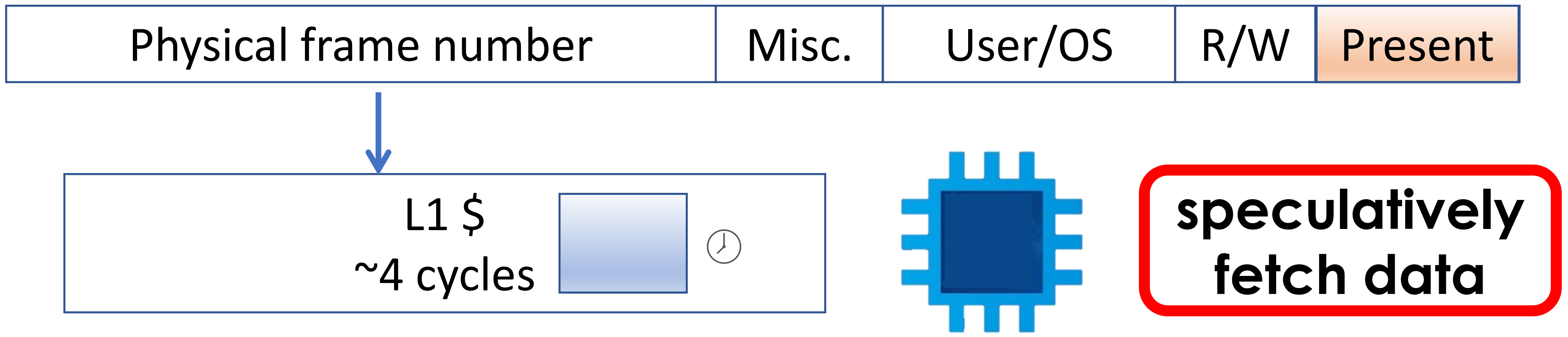
- Variant 1: Invalid PTE (Page Table Entry)
- Variant 2: Enclave to Enclave (E2E) rogue mapping

Foreshadow –
Causing a Translation Terminal Fault



What happens when the translation faults?

Faulty PTE (Page Table Entry):

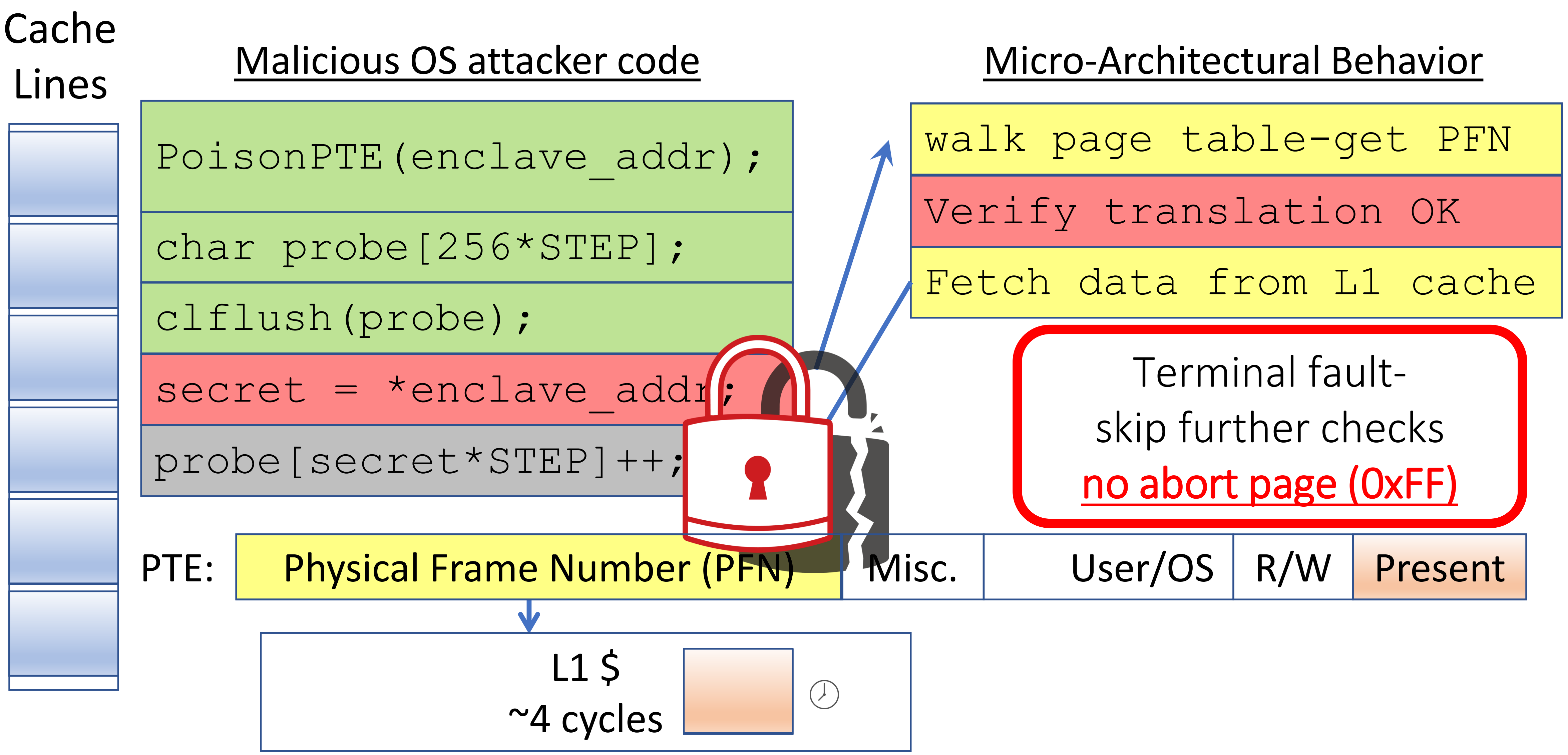


Following a terminal fault (from Intel's report):

- SGX memory checks are skipped (no 0xFF)
- Boundaries between VM and host are ignored
- System Management Mode (SMM) checks are skipped

Source: https://software.intel.com/security-software-guidance/insights/deep-dive-intel-analysis-l1-terminal-fault

Foreshadow Attack


Cache
Lines
Malicious OS attacker code
Micro-Architectural Behavior
PTE:
Physical Frame Number (PFN)
Misc.
User/OS
R/W
Present
L1 $
~4 cycles

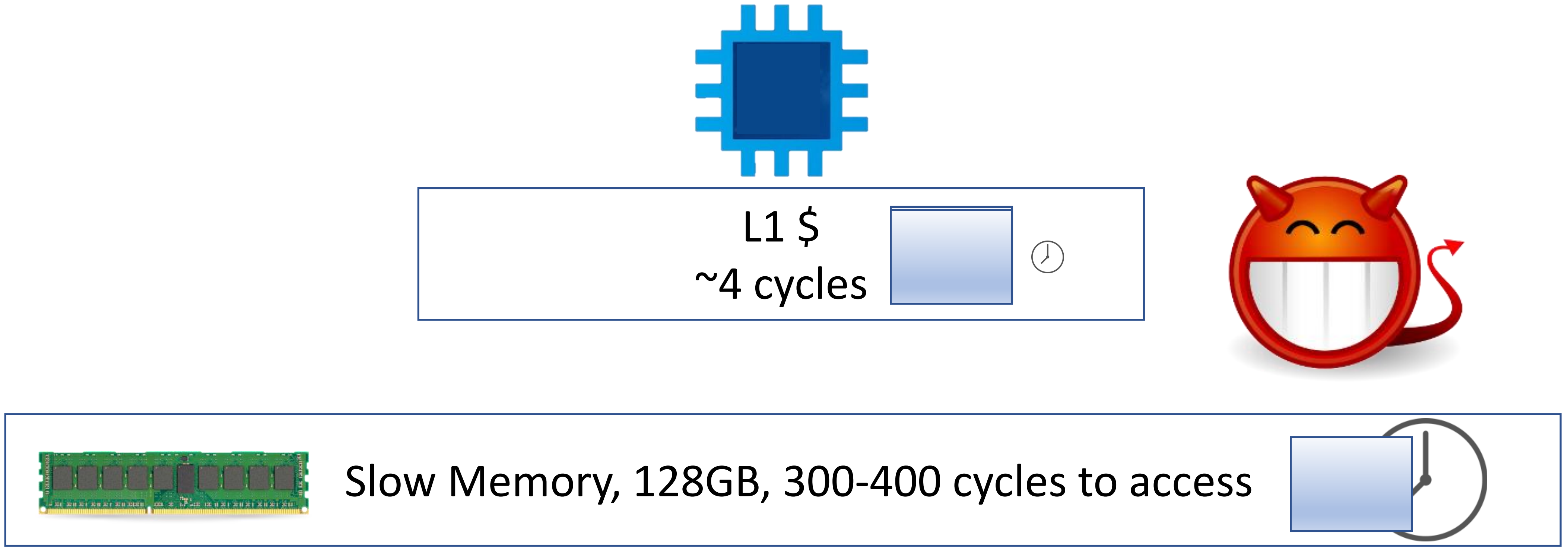
```
PoisonPTE(enclave_addr);
char probe[256*STEP];
clflush(probe);
secret = *enclave_addr;
probe[secret*STEP]++;
```

```
walk page table-get PFN
Verify translation OK
Fetch data from L1 cache
```

Terminal fault-
skip further checks
no abort page (0xFF)

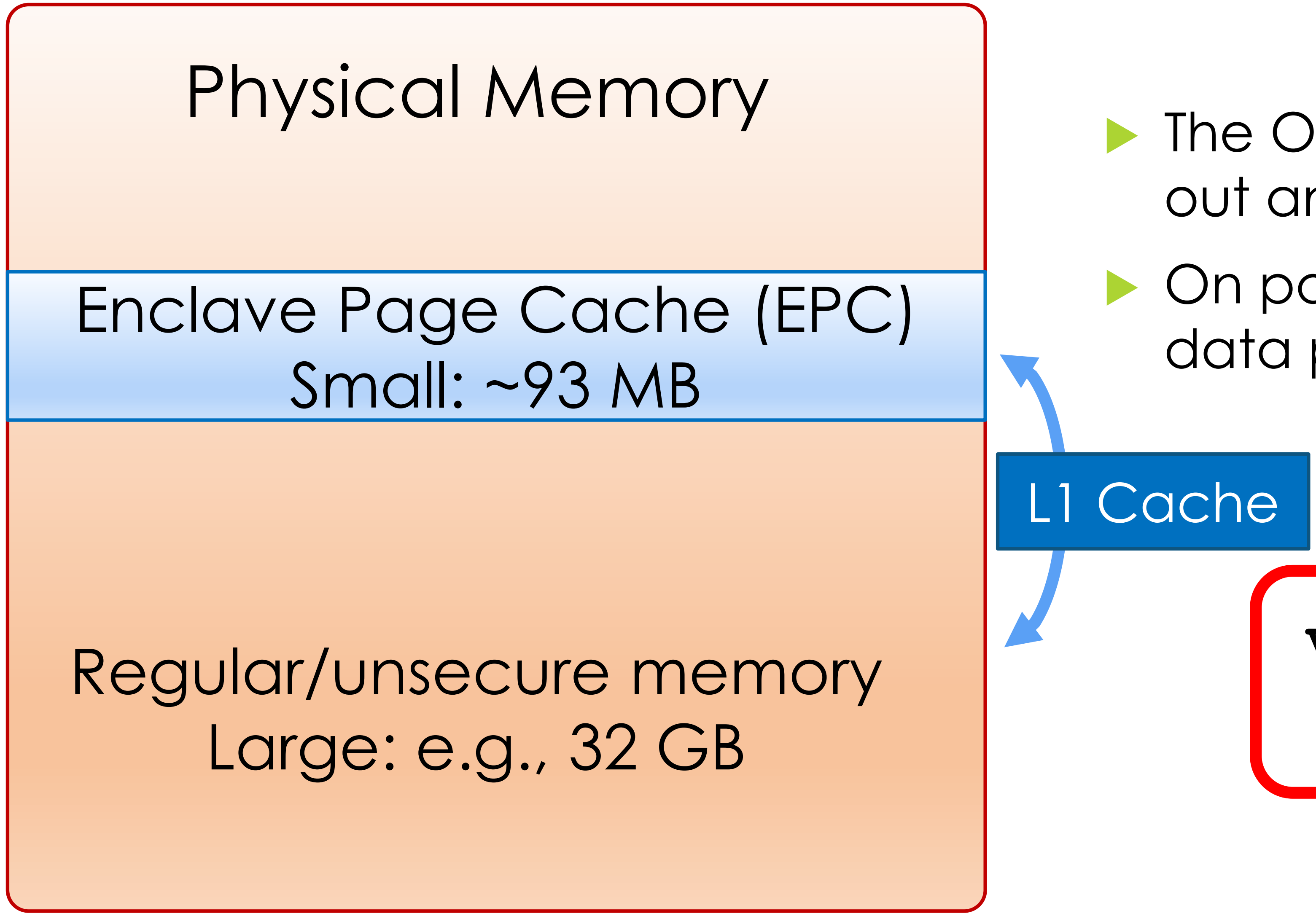
Only Data in L1 Cache is Exposed

- Following a “terminal fault” only data in L1 cache may be fetched



But what if the attacker can bring data into L1 cache?

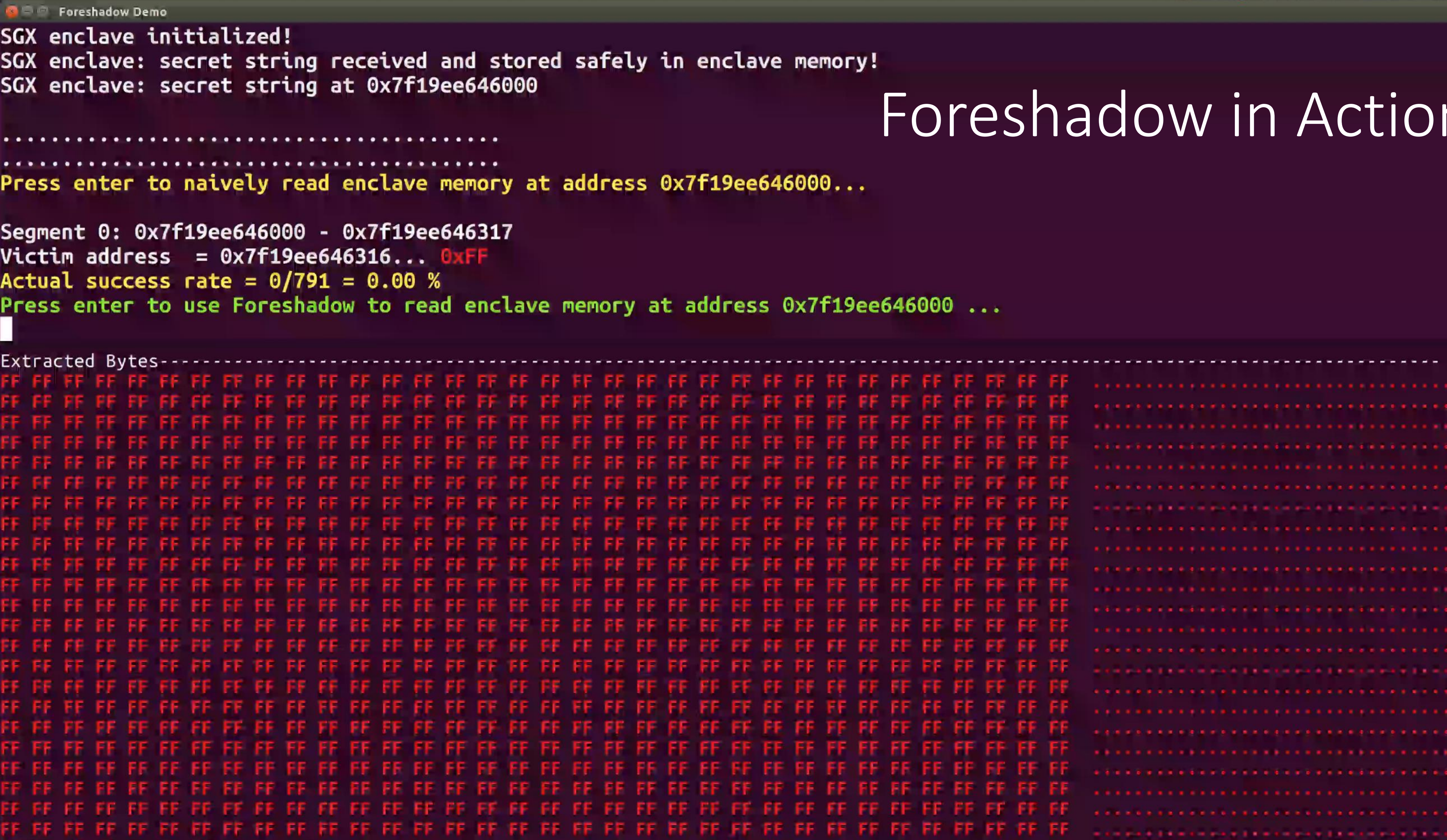
Maliciously Fetching Into L1 Cache



- The OS can "securely" page-out and page-in SGX pages
- On page-in - the decrypted data passes through L1 cache

Victim doesn't need to run!!

Foreshadow in Action



```
SGX enclave initialized!
SGX enclave: secret string received and stored safely in enclave memory!
SGX enclave: secret string at 0x7f19ee646000

................................................
................................................
Press enter to naively read enclave memory at address 0x7f19ee646000...

Segment 0: 0x7f19ee646000 - 0x7f19ee646317
Victim address  = 0x7f19ee646316... 0xFF
Actual success rate = 0/791 = 0.00 %
Press enter to use Foreshadow to read enclave memory at address 0x7f19ee646000 ...


Extracted Bytes----------------------------------------------------------------------------------------------------------------------------
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ..................................
```

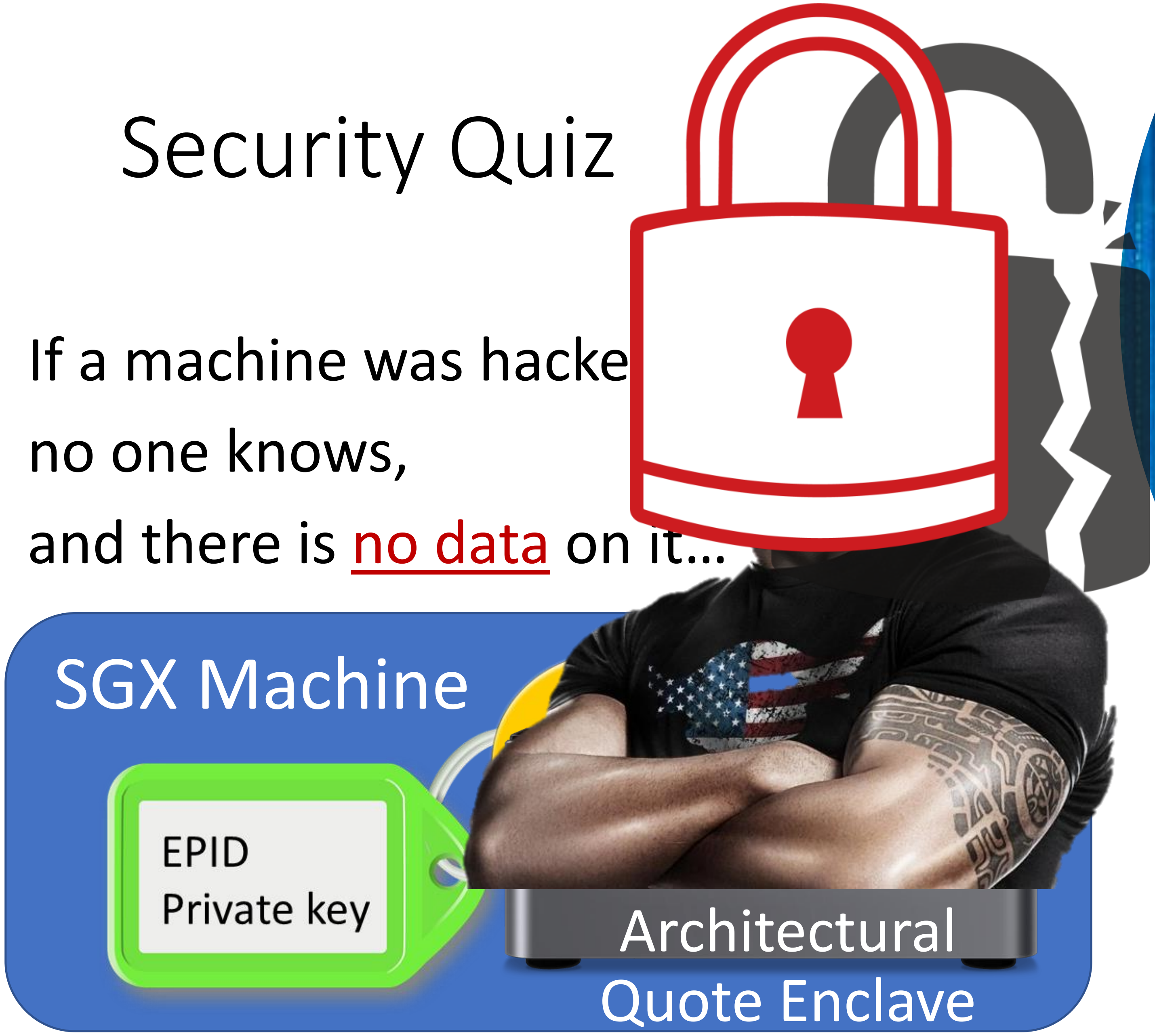
Implications on SGX Enclaves and Ecosystem

- Confidentiality is completely gone:
  Foreshadow can dump entire enclaves
- At any given time, without the enclave running
- Secure storage is not safe:
  Foreshadow can extract SGX sealing (secure storage) keys
- Proof of integrity (attestation) can be forged:
  Foreshadow can extract secrets from
  - Intel Launch Enclave
  - Intel Quote Enclave

Ramification: a collapse of the attestation ecosystem

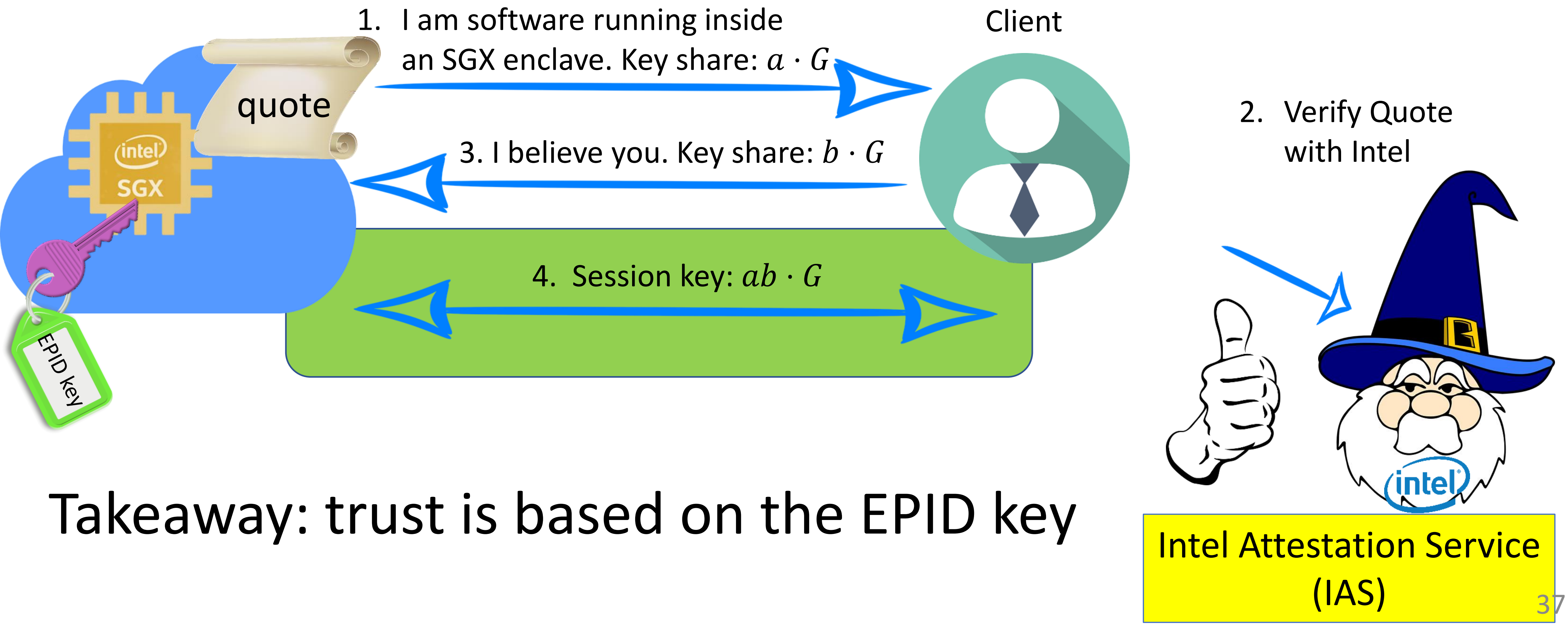
Security Quiz

If a machine was hacke
no one knows,
and there is no data on it…



Should we care?

Remote Attestation: Establishing Trust with Remote Enclaves



1. I am software running inside an SGX enclave. Key share: $a \cdot G$
Client
quote
intel
SGX
3. I believe you. Key share: $b \cdot G$
2. Verify Quote with Intel
4. Session key: $ab \cdot G$
EPID key
intel
Intel Attestation Service (IAS)

Takeaway: trust is based on the EPID key

EPID - Enhanced Privacy ID

- EPID mega feature – awesome privacy
- Millions of signatures are unlinkable
- No one knows who signed what

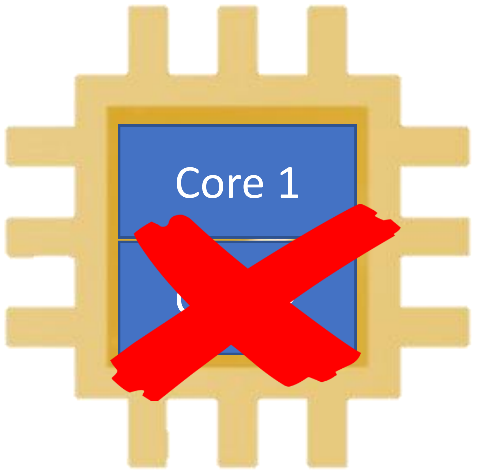
EPID failure – abusing privacy

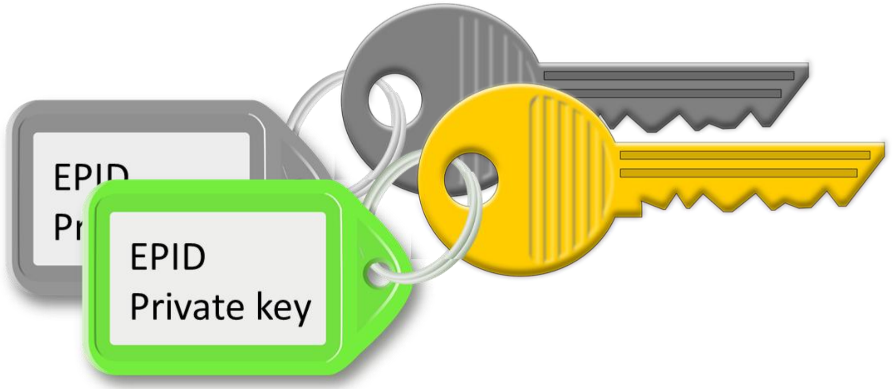
A single extracted EPID key can be used to sign millions of unlinkable signatures

@ForeshadowAaaS

Foreshadow-SGX Mitigations

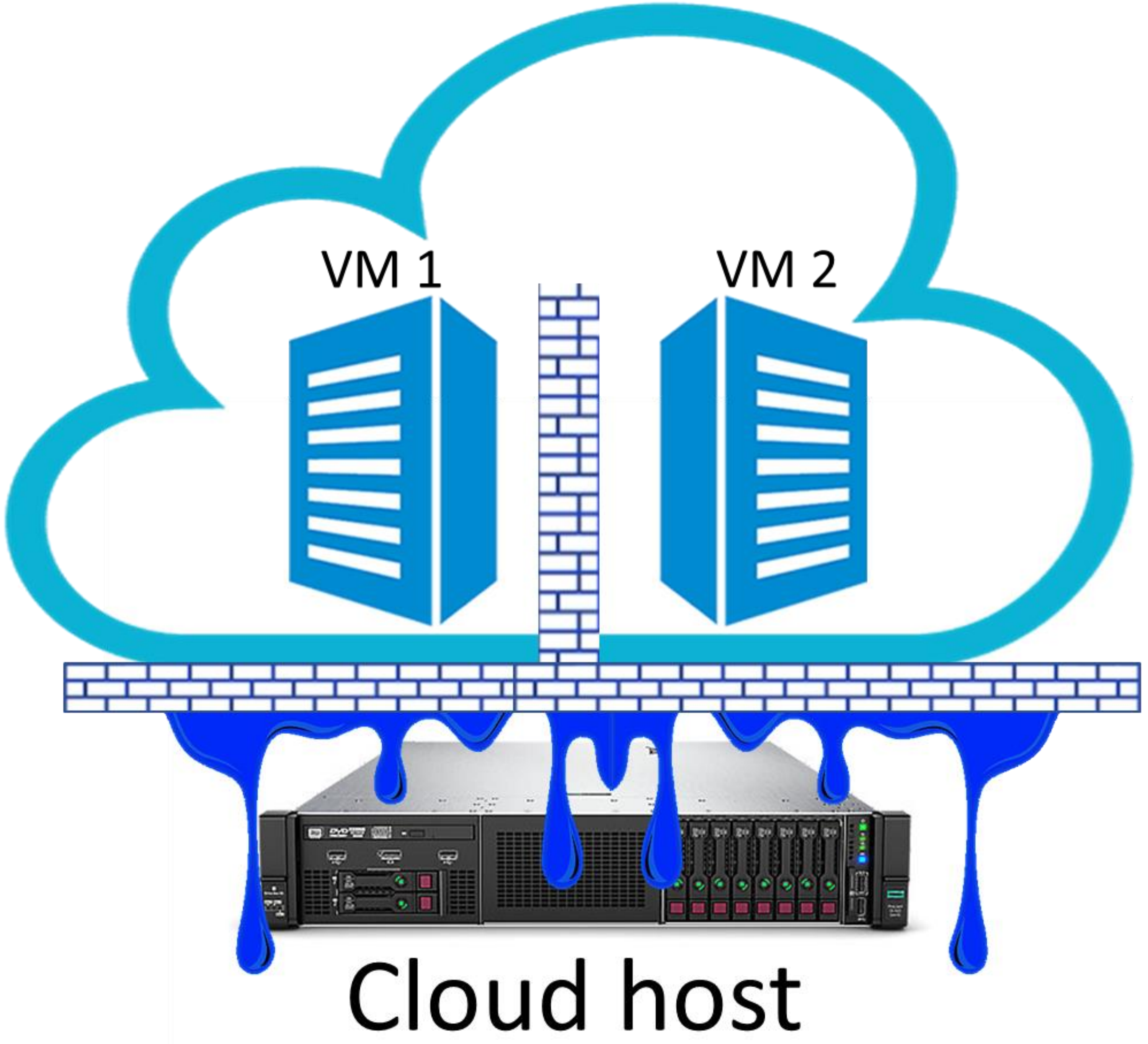
- Flush L1 Cache after enclave exits and “page-in/out” operations
  - New L1 flush “instruction” added
- Disable HyperThreading
- Have two sets of Attestation/Sealing keys
  - For HyperThreading On/Off



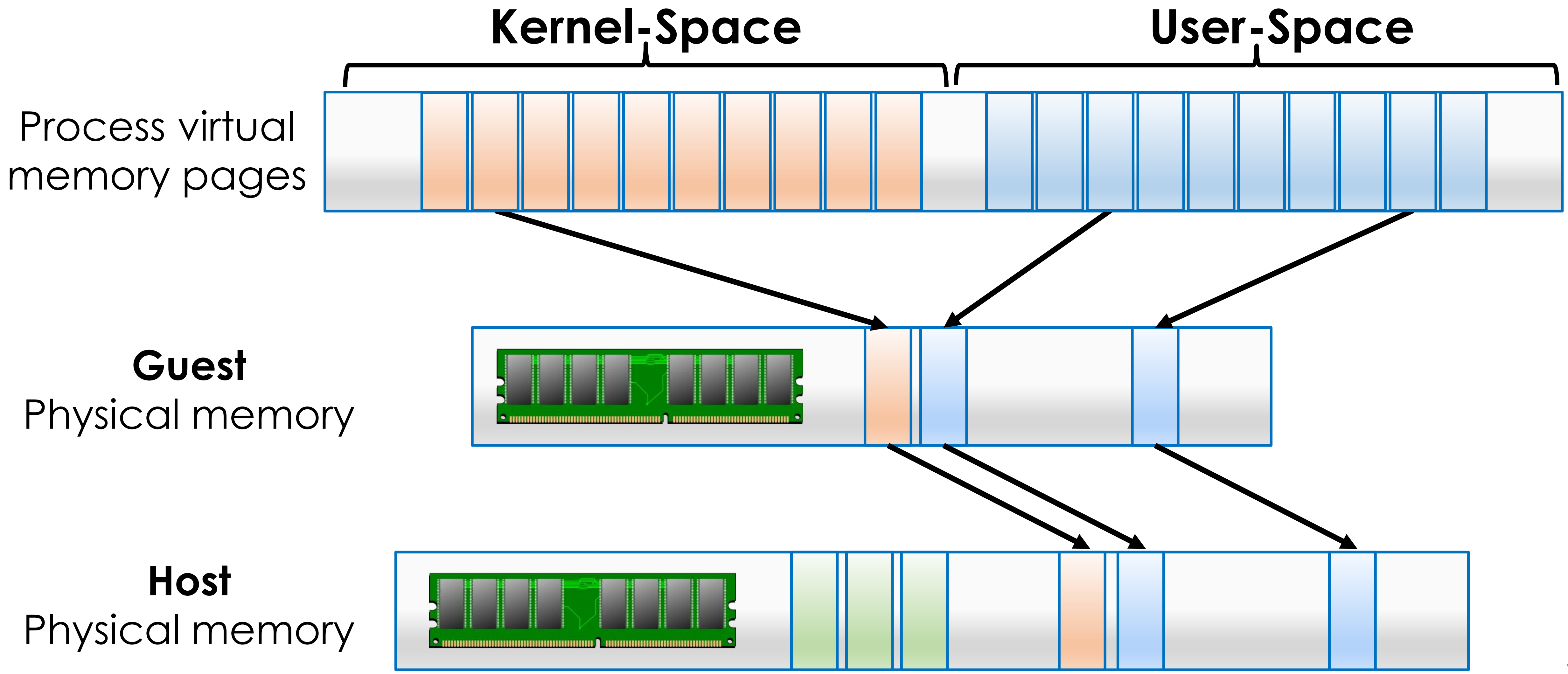


Roadmap

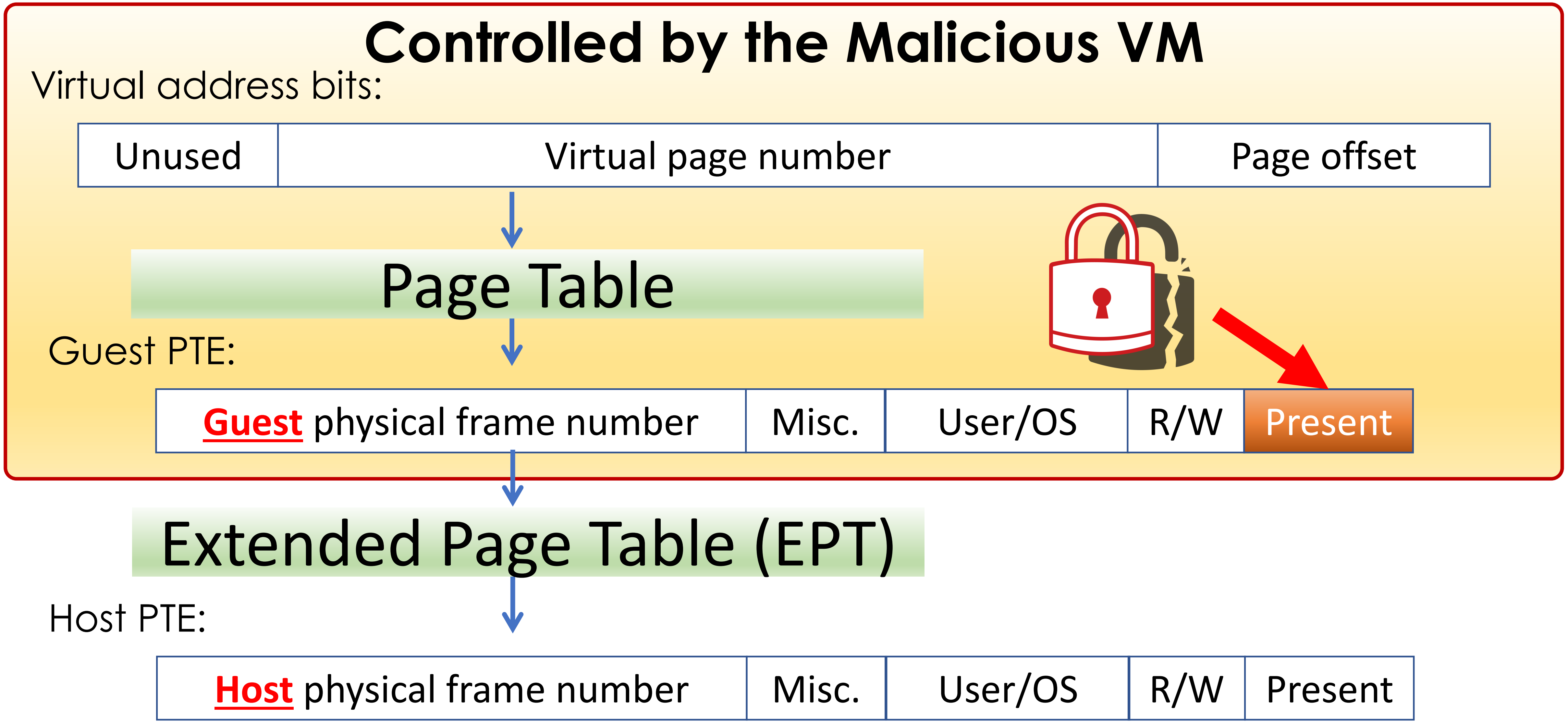
- Cache side channels
- Speculative execution
- Meltdown
- SGX
- Foreshadow-SGX
- **Foreshadow-NG**
  - User-space to kernel
  - Reading SMM memory
  - VM-to-VM/M



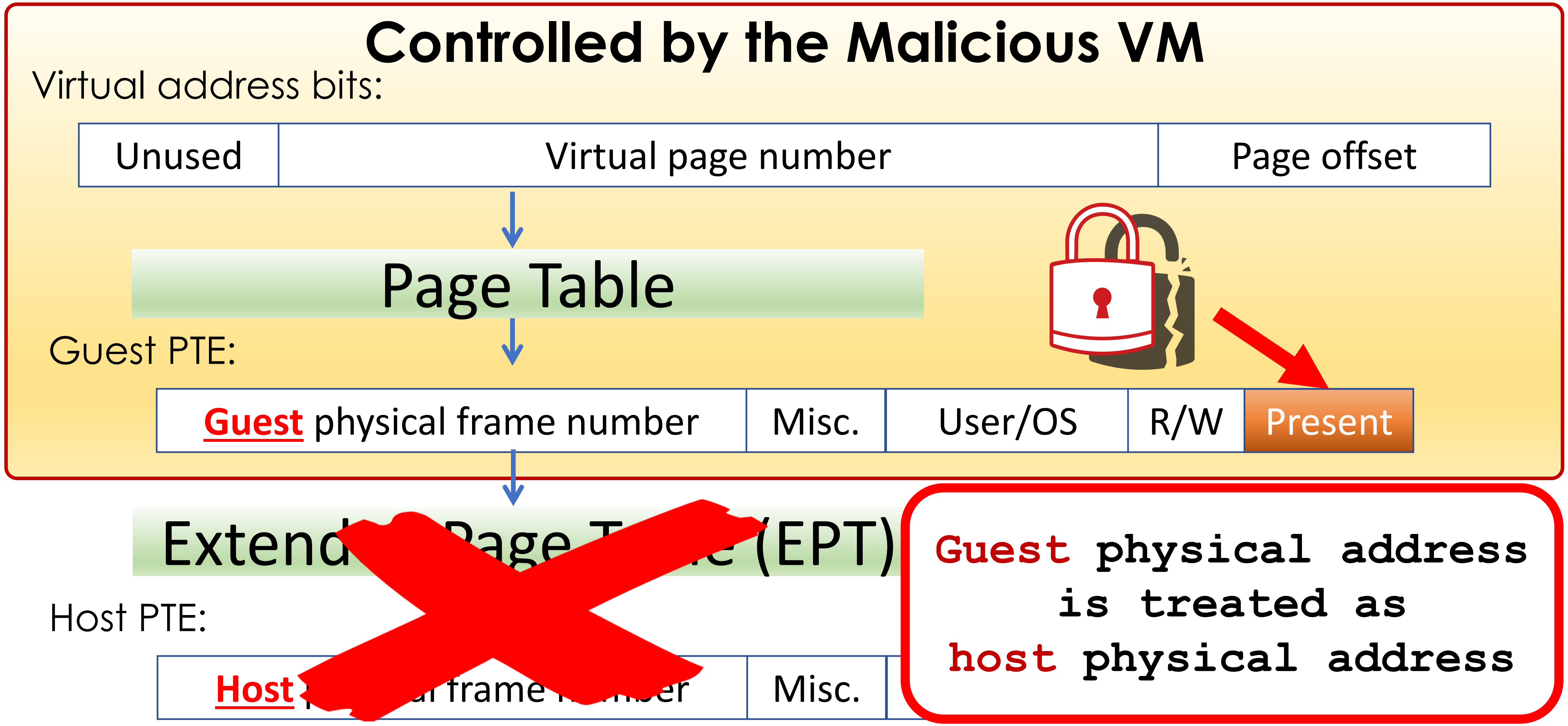
Nested Virtual Address Space



The Extended Page Table & Foreshadow

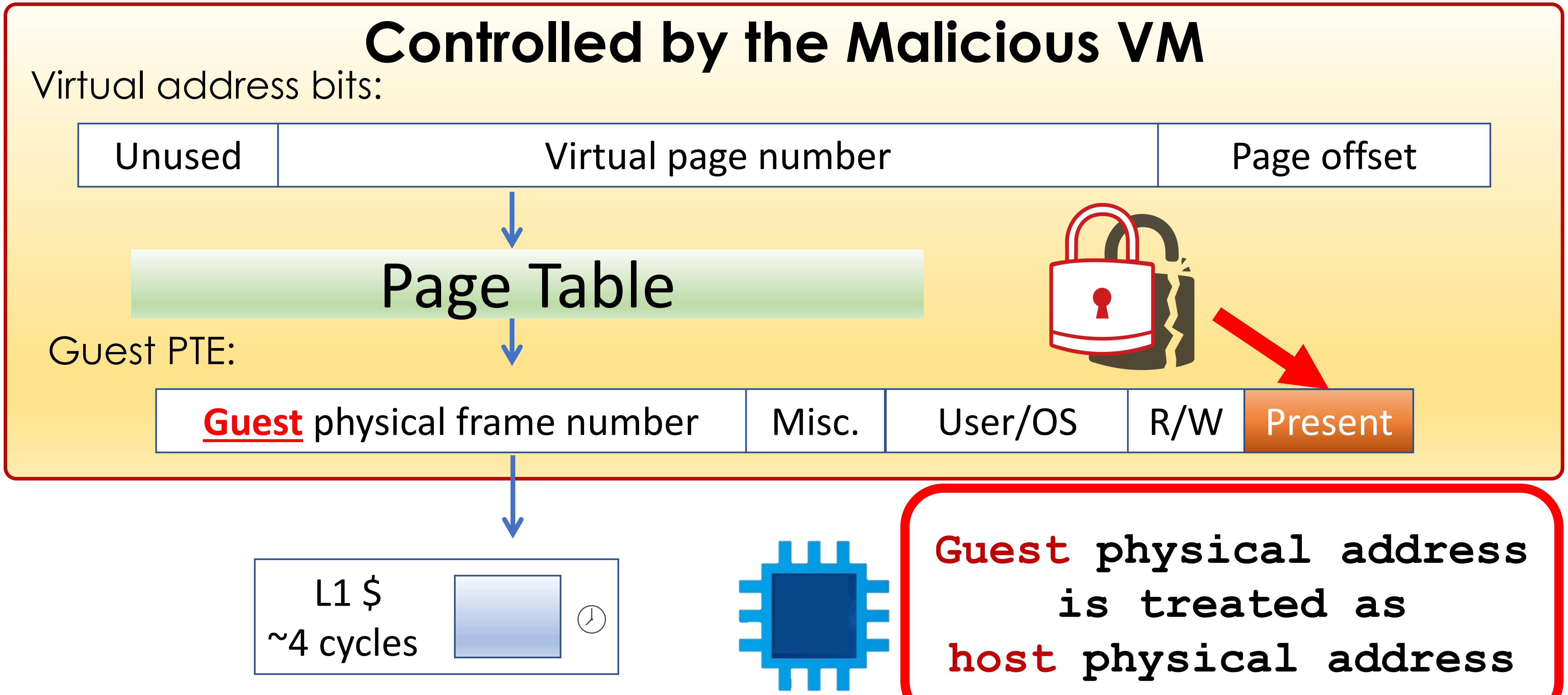


The Extended Page Table & Foreshadow



Guest physical address is treated as host physical address

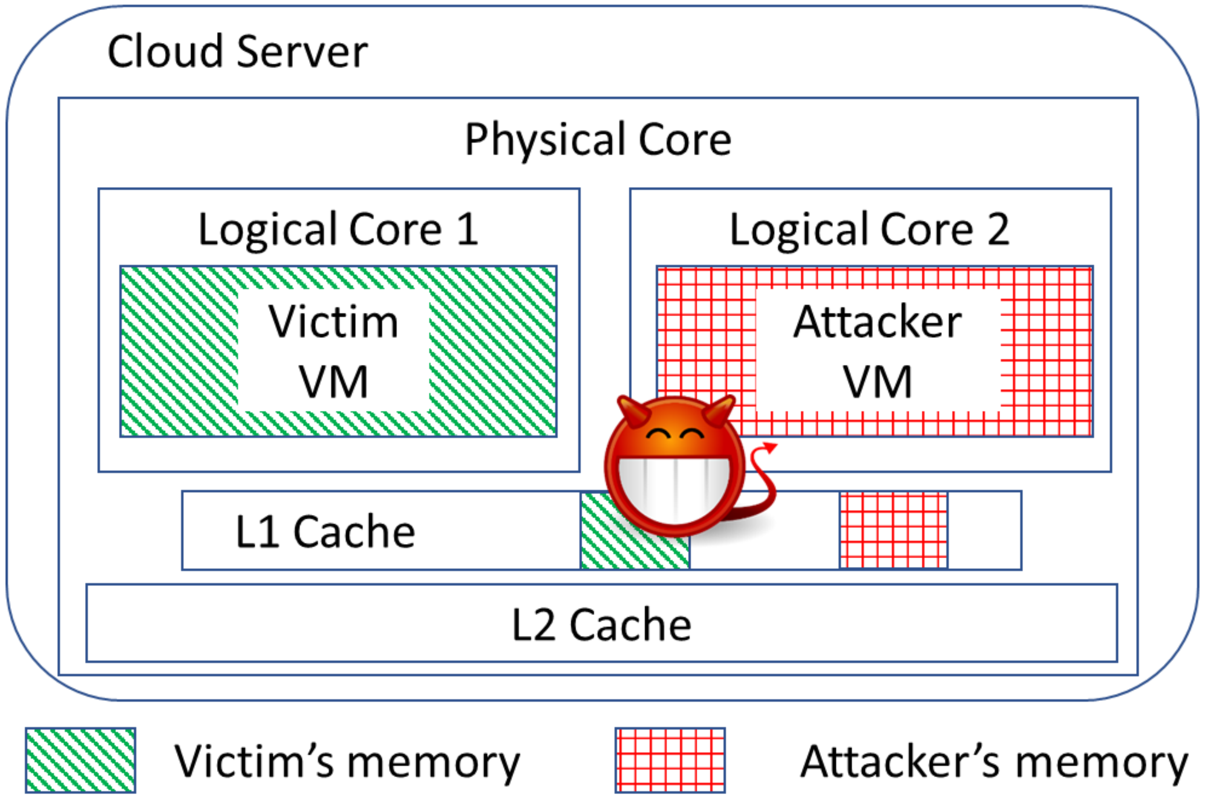
The Extended Page Table & Foreshadow



Guest physical address is treated as host physical address

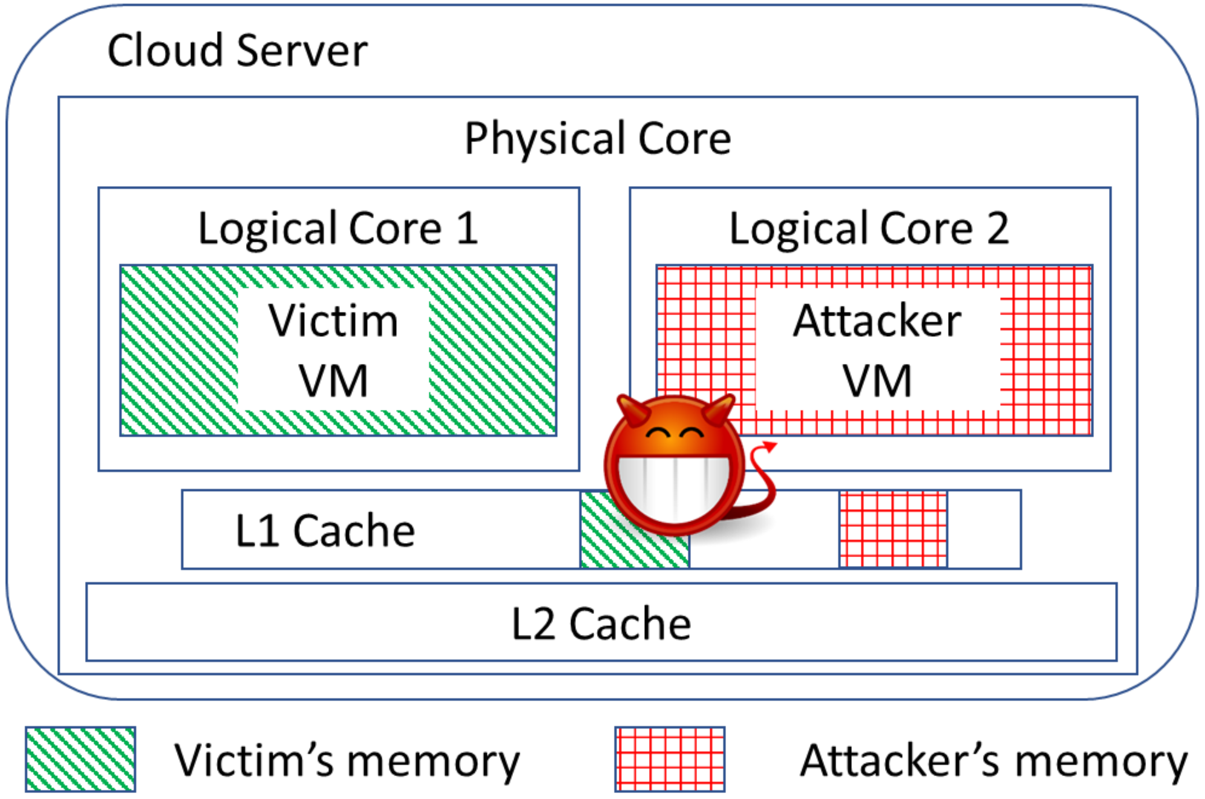
Implications

- VM boundary is broken
- A malicious VM can read data from a neighboring VM or the VMM



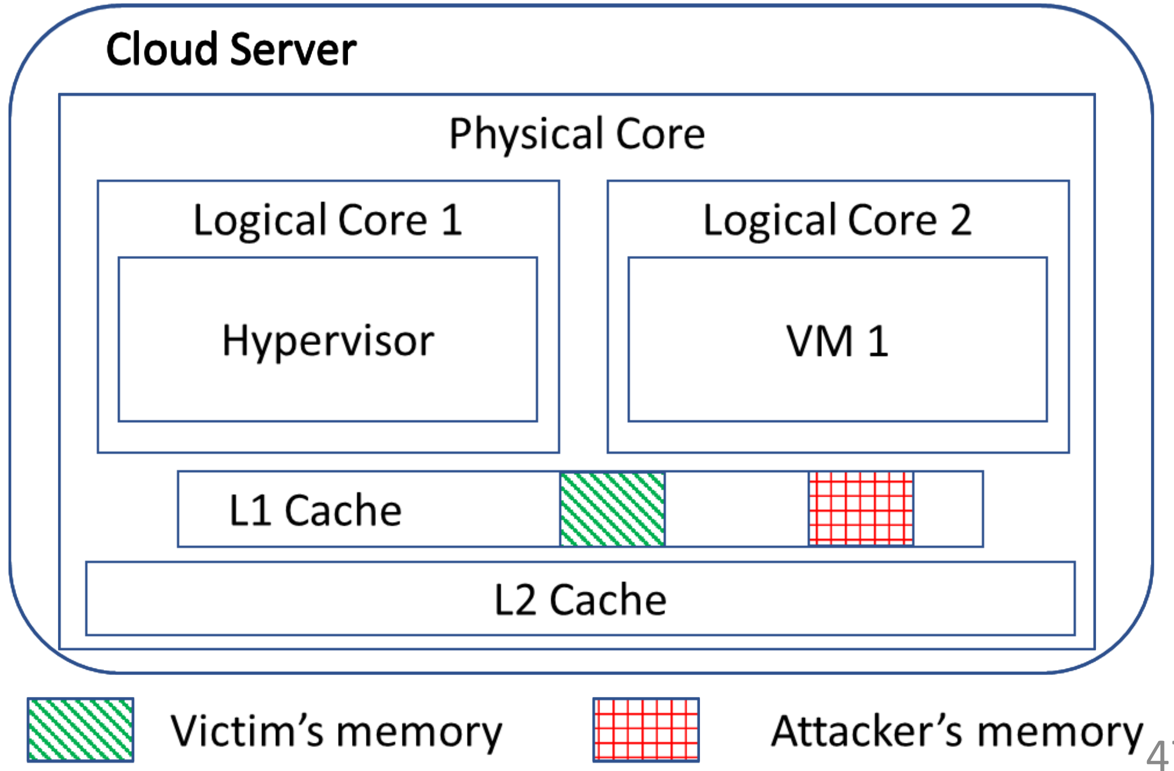
Attack Limitations

- Data needs to reside in L1 cache (unlike the SGX attack)
- Attacker needs to guess/know physical address
- no know attacks in the wild



Mitigating Foreshadow-NG

- Disabling HyperThreading is devastating for performance
  - So what can we do?
- Never run two VMs on the same physical core
  - May impact performance
- Flush L1 cache on VMENTER
- On VMEXIT to hypervisor – make sure other sibling core is trusted



Conclusions

- Foreshadow-SGX: a complete break of SGX, including
  - Confidentiality
  - Secure storage
  - Attestation

Patch your machine!

- Privacy-preserving protocols can backfire (e.g., EPID)
- Foreshadow-NG: VM boundary is cracked
- Mitigations come at a performance cost

ForeshadowAttack.com