Leaky Cauldron on the Dark Land: Understanding Memory Side-Channel Hazards in SGX

1,4Wenhao Wang, 2Guoxing Chen, 1Xiaorui Pan, 2Yinqian Zhang, 1Xiaofeng Wang, 3Vincent Bindschaedler, 1Haixu Tang and 3Carl A. Gunter

1Indiana University Bloomington  2The Ohio State University  
3University of Illinois Urbana-Champaign  4Institute of Information Engineering
Intel Software Guard Extensions

Processor Reserved Memory (PRM)

ELRANGE
SSN, Financial/Health Data

Application address space

PRM
Enclave Page Cache (EPC)

Physical memory
Intel Software Guard Extensions

Processor Reserved Memory (PRM)

- ELRANGE
  - SSN, Financial/Health Data

- PRM
  - Enclave Page Cache (EPC)

- Application address space
- Physical memory
Intel Software Guard Extensions

Processor Reserved Memory (PRM)

ELRANGE
SSN, Financial/Health Data

Application address space

PRM
Enclave Page Cache (EPC)

Physical memory
Intel Software Guard Extensions

Processor Reserved Memory (PRM)

**ELRANGE**
SSN, Financial/Health Data

Application address space

---

PRM
Enclave Page Cache (EPC)

Physical memory

---

Red Xs indicate access restrictions or security measures.
Intel Software Guard Extensions

Processor Reserved Memory (PRM)

- ELRANGE
  - SSN, Financial/Health Data
- Memory management
- PRM
  - Enclave Page Cache (EPC)
- Physical memory
- Application address space
Controlled-channel attacks: OS controls page tables and set traps by making pages inaccessible!
Defenses against page-fault attacks

T-SGX

Images taken from the authors’ slides
Defenses against page-fault attacks

DEJA VU

Images taken from the authors’ slides
Defenses against page-fault attacks

Deterministic multiplexing

Images taken from the authors’ slides
Defenses against page-fault attacks

T-SGX

DEJA VU

Deterministic multiplexing

Images taken from the authors’ slides
Our contributions

- A comprehensive understanding of SGX memory side channels.
  - 8 attack vectors.
Our contributions

- A comprehensive understanding of SGX memory side channels.
  - 8 attack vectors.

- Reducing AEXs induced by page level attacks.
  - A new type of attacks.
Our contributions

- A comprehensive understanding of SGX memory side channels.
  - 8 attack vectors.

- Reducing AEXs induced by page level attacks.
  - A new type of attacks.

- Achieving finer-grained (than 4 KB) spatial granularity.
  - Cache-DRAM attack.
1. Understanding Attack Surfaces

```
mov (%rax), %rbx
```

```
Address translation unit

%rax 7FFFFFFF0014E20480 F0014E20

%rbx 480

TLB hit? y n

hit caches? y n

Cache fill from physical memory

Page fault – allocate EPC page

4-level page table walk

hit page structure caches? y n

hit caches? y n

access physical memory

EPC page?

Page table entries (PTE)
1. Understanding Attack Surfaces

```asm
mov (%rax), %rbx
```

---

**Diagram:**

- `%rax` → `7FFFFFF0014E20480` → `F0014E20` → Address translation unit
- TLB hit? (Yes: `y`, No: `n`)
- `%rbx` → 480
- `%rbx` → `00882E2`
- `%rbx` → `00882E2480`
- hit caches? (Yes: `y`, No: `n`)
- EPC page? (Yes: `y`, No: `n`)
- Page fault – allocate EPC page
- 4-level page table walk
- hit page structure caches? (Yes: `y`, No: `n`)
- hit caches? (Yes: `y`, No: `n`)
- access physical memory
- page table entries (PTE)
- Cache fill from physical memory
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Summary of Attack vectors

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
Can we make the attack stealthy by reducing AEXs induced by the attack?
2. Sneaky Page Monitoring Attacks (Vector 4)

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V5. Updates of dirty flags.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
2. Sneaky Page Monitoring Attacks (Vector 4)

2. Sneaky Page Monitoring Attacks (Vector 4)


“Whenever the processor uses a paging-structure entry as part of linear-address translation, it sets the accessed flag in that entry (if it is not already set).”
Basic accessed flags monitoring attack: B-SPM

SGX hardware

OS

Page tables

enclave pages
2. Sneaky Page Monitoring Attacks (Vector 4)

Basic accessed flags monitoring attack: B-SPM

- **OS page tables**
- **SGX hardware**
- **enclave pages**

- **IPIs**
  - wait
  - A = 1? → y
  - Page Accessed. A = 0.
  - n
2. Sneaky Page Monitoring Attacks

Basic accessed flags monitoring attack: B-SPM

Evaluate on Hunspell.

<table>
<thead>
<tr>
<th>group size</th>
<th>Page-fault based</th>
<th>Accessed-flag based</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>words</td>
<td>%</td>
</tr>
<tr>
<td>1</td>
<td>51599</td>
<td>83.05</td>
</tr>
<tr>
<td>2</td>
<td>7586</td>
<td>12.21</td>
</tr>
<tr>
<td>3</td>
<td>2073</td>
<td>3.34</td>
</tr>
<tr>
<td>4</td>
<td>568</td>
<td>0.91</td>
</tr>
<tr>
<td>5</td>
<td>200</td>
<td>0.32</td>
</tr>
<tr>
<td>6</td>
<td>60</td>
<td>0.10</td>
</tr>
<tr>
<td>7</td>
<td>35</td>
<td>0.06</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0.01</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&gt; 10</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Slowdown is brought down from $1214.9 \times$ for page fault attack to $5.1 \times$ for B-SPM attack.
2. Sneaky Page Monitoring Attacks

What about if the pages that frequently accessed are to be observed?

[Diagram showing a decision point with conditions and two paths]

Path A and Path B with conditional branching to BB0 and BB1/B2/B3 respectively.
2. Sneaky Page Monitoring Attacks

Timing enhancement: T-SPM

Measure the time between two monitored pages.
2. Sneaky Page Monitoring Attacks

Timing enhancement: T-SPM

```
<table>
<thead>
<tr>
<th>α</th>
<th>β</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>α₁</td>
<td>β₁</td>
<td>t₁</td>
</tr>
<tr>
<td>α₂</td>
<td>β₂</td>
<td>t₂</td>
</tr>
<tr>
<td>α₃</td>
<td>β₃</td>
<td>t₃</td>
</tr>
<tr>
<td>α₄</td>
<td>β₄</td>
<td>t₄</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
```
2. Sneaky Page Monitoring Attacks

Timing enhancement: T-SPM

Evaluate on FreeType.

Slowdown is brought down from $252 \times$ for page fault attack to $0.16 \times$ for T-SPM attack.
2. Sneaky Page Monitoring Attacks

Can the side effect be further reduced?
2. Sneaky Page Monitoring Attacks

- V1. Shared TLB entries under HT.
- V2. Selective TLB entries flushing without HT.
- V3. Referenced PTEs are cached as data.
- V6. Triggering page faults with P/X or reserved bits.
- V7. CPU caches are shared between the enclave and non-enclave code.
- V8. The memory hierarchy, specifically the row buffers are shared.
2. Sneaky Page Monitoring Attacks

TLB flushing with HT (Vector 1): HT-SPM
2. Sneaky Page Monitoring Attacks

TLB flushing with HT (Vector 1): HT-SPM
2. Sneaky Page Monitoring Attacks

TLB flushing with HT (Vector 1): HT-SPM

- memory reference
- TLB miss
- Page table walk
- TLB priming
- TLB hit
- physical address
2. Sneaky Page Monitoring Attacks

Evaluation on EdDSA of Libgcrypt v1.7.6

```c
void
_gcry_mpi_ec_mul_point (mpi_point_t result,
    gcry_mpi_t scalar, mpi_point_t point,
    mpi_ec_t ctx)
{
    if (ctx->model == MPI_EC_EDWARDS
        || (ctx->model == MPI_EC_WEIERSTRASS
            && mpi_is_secure (scalar))) {
        if (mpi_is_secure (scalar)) {
            /* If SCALAR is in secure memory we assume that it is the
                secret key we use constant time operation. */
            ...
        } else {
            for (j=nbits-1; j >= 0; j--) {
                _gcry_mpi_ec_dup_point (result, result, ctx);
                if (mpi_test_bit (scalar, j))
                    _gcry_mpi_ec_add_points (result, result, point, ctx);
            }
        }
    }
    return;
}
```
2. Sneaky Page Monitoring Attacks

Evaluation on EdDSA of Libgcrypt v1.7.6

```c
void _gcry_mpi_ec_mul_point (mpi_point_t result,
    gcry mpi_t scalar, mpi_point_t point,
    mpi_ec_t ctx) {
    if (ctx->model == MPI_EC_EDWARDS
        || (ctx->model == MPI_EC_WEIERSTRASS
            && mpi_is_secure (scalar))) {
        if (mpi_is_secure (scalar)) {
            /* If SCALAR is in secure memory we assume that it is the
             secret key we use constant time operation. */
            ...
        } else {
            for (j=nbits-1; j >= 0; j--) {
                _gcry_mpi_ec_dup_point (result, result, ctx);
                _gcry_test_bit (scalar, j)
            }
            _gcry_mpi_ec_add_points (result, result, point, ctx);
        }
    }
    return;
}
```

<table>
<thead>
<tr>
<th>Attacks</th>
<th>Number of AEXs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page fault attack</td>
<td>71,000</td>
</tr>
<tr>
<td>B-SPM attack</td>
<td>33,000</td>
</tr>
<tr>
<td>T-SPM attack</td>
<td>1,300</td>
</tr>
</tbody>
</table>

* HT-SPM is designed to reduce AEXs for data pages, and is not presented in the comparison.
2. Sneaky Page Monitoring Attacks

Evaluation on EdDSA of Libgcrypt v1.7.6

<table>
<thead>
<tr>
<th>Attacks</th>
<th>Number of AEXs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page fault attack</td>
<td>71,000</td>
</tr>
<tr>
<td>B-SPM attack</td>
<td>33,000</td>
</tr>
<tr>
<td>T-SPM attack</td>
<td>1,300</td>
</tr>
</tbody>
</table>

* HT-SPM is designed to reduce AEXs for data pages, and is not presented in the comparison.
3. Achieving fine-grained spatial granularity

- Cache-based attack
  - Prime+Probe: 16 KB, if 2048 cache set, 128 MB EPC
  - Flush+Reload: 64 B
3. Achieving fine-grained spatial granularity

- Cache-based attack
  - Prime+Probe: 16 KB, if 2048 cache set, 128 MB EPC
  - Flush+Reload: 64 B

- DRAMA attack
  - The program needs to have a large memory footprint, otherwise the memory reference will mostly hit the cache.
3. Achieving fine-grained spatial granularity

- **Cache-based attack**
  - Prime+Probe: 16 KB, if 2048 cache set, 128 MB EPC
  - Flush+Reload: 64 B

- **DRAMA attack**
  - The program needs to have a large memory footprint, otherwise the memory reference will mostly hit the cache.

**Cache-DRAM attack**: finer-grained attack with less noise.
3. Achieving fine-grained spatial granularity

Cache-DRAM attack

64 B granularity

- DRAM rows are only shared among enclaves.
- No high resolution timer inside the enclaves.
3. Achieving fine-grained spatial granularity

Cache-DRAM attack

- 64 B granularity

- DRAM rows are only shared among enclaves.
- No high resolution timer inside the enclaves.

Evaluation on a conditional branch in Gap 4.8.6.
14.6% detection, <1% false detection.
### Summary of Attack Vectors

<table>
<thead>
<tr>
<th>Vectors</th>
<th>Spatial granularity</th>
<th>AEX</th>
<th>Slow-down</th>
</tr>
</thead>
<tbody>
<tr>
<td>* i/dCache PRIME+PROBE</td>
<td>2 MB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>* L2 Cache PRIME+PROBE</td>
<td>128 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>L3 Cache PRIME+PROBE</td>
<td>16 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Page fault attack</td>
<td>4 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>B/T-SPM</td>
<td>4 KB</td>
<td>Modest</td>
<td>Modest</td>
</tr>
<tr>
<td>HT-SPM</td>
<td>4 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Cross-enclave DRAMA</td>
<td>1 KB</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>Cache-DRAM</td>
<td>64 B</td>
<td>None</td>
<td>Minimal</td>
</tr>
</tbody>
</table>

* Do not consider attacks under HT. Otherwise the AEX and slow-down will be low.
## Summary of Attack Vectors

<table>
<thead>
<tr>
<th>Vectors</th>
<th>Spatial granularity</th>
<th>AEX</th>
<th>Slow-down</th>
</tr>
</thead>
<tbody>
<tr>
<td>* i/dCache PRIME+PROBE</td>
<td>2 MB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>* L2 Cache PRIME+PROBE</td>
<td>128 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>L3 Cache PRIME+PROBE</td>
<td>16 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Page fault attack</td>
<td>4 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>B/T-SPM</td>
<td>4 KB</td>
<td>Modest</td>
<td>Modest</td>
</tr>
<tr>
<td>HT-SPM</td>
<td>4 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Cross-enclave DRAMA</td>
<td>1 KB</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>Cache-DRAM</td>
<td>64 B</td>
<td>None</td>
<td>Minimal</td>
</tr>
</tbody>
</table>

* Do not consider attacks under HT. Otherwise the AEX and slow-down will be low.
<table>
<thead>
<tr>
<th>Vectors</th>
<th>Spatial granularity</th>
<th>AEX</th>
<th>Slow-down</th>
</tr>
</thead>
<tbody>
<tr>
<td>* i/dCache PRIME+PROBE</td>
<td>2 MB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>* L2 Cache PRIME+PROBE</td>
<td>128 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>L3 Cache PRIME+PROBE</td>
<td>16 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Page fault attack</td>
<td>4 KB</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>B/T-SPM</td>
<td>4 KB</td>
<td>Modest</td>
<td>Modest</td>
</tr>
<tr>
<td>HT-SPM</td>
<td>4 KB</td>
<td>None</td>
<td>Modest</td>
</tr>
<tr>
<td>Cross-enclave DRAMA</td>
<td>1 KB</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>Cache-DRAM</td>
<td>64 B</td>
<td>None</td>
<td>Minimal</td>
</tr>
</tbody>
</table>

* Do not consider attacks under HT. Otherwise the AEX and slow-down will be low.
Conclusions

- We identified 8 attack vectors in SGX memory management.
Looking again at the attack surfaces

```
mov (%rax), %rbx
```

```
%rax 7FFFFFFF0014E20480 F0014E20
%rbx 480 00882E2480
```

TLB hit?

- Address translation unit
  - 4-level page table walk
  - hit page structure caches?
  - hit caches?
  - EPC page?
  - Page fault – allocate EPC page
    - hit caches?
    - EPC page?
    - page table entries (PTE)
    - access physical memory

Cache fill from physical memory
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
- New attacks that induce few AEXs, that bypass existing defenses
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
- New attacks that induce few AEXs, that bypass existing defenses
  - Interrupts are not necessary to attack the enclave.
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
- New attacks that induce few AEXs, that bypass existing defenses
  - Interrupts are not necessary to attack the enclave.
- Attacks can achieve finer-grained spatial granularity.
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
- New attacks that induce few AEXs, that bypass existing defenses
  - Interrupts are not necessary to attack the enclave.
- Attacks can achieve finer-grained spatial granularity.
- Attack vectors can be combined to be more effective
  - TLB flushing + SPM, Cache + DRAM, Page monitoring + timing
  - Others?
Conclusions

- We identified 8 attack vectors in SGX memory management.
  - There can be more.
- New attacks that induce few AEXs, that bypass existing defenses
  - Interrupts are not necessary to attack the enclave.
- Attacks can achieve finer-grained spatial granularity.
- Attack vectors can be combined to be more effective
  - TLB flushing + SPM, Cache + DRAM, Page monitoring + timing
  - Others?
- Defenses?
Thanks! Any questions?

www31@indiana.edu
Backup Slides
Characterizing memory vectors

Spatial granularity

The smallest unit of information directly observable to the adversary.

Temporal observability

The ability for the adversary to measure the timing signals generated during the execution of the target program.

Side effects

Observable anomalies caused by an attack, which could be employed to detect the attack, such as AEX.
Intel Software Guard Extensions

- Life cycle of an enclave thread
Related work on Security’17

- Vector 3, 4
1. Understanding Attack Surfaces

mov (%rax), %rbx
1. Understanding Attack Surfaces

mov (%rax), %rbx

%rax → 7FFFFFFF0014E20480 → F0014E20

%rbx → 480

virtual page number

page offset
1. Understanding Attack Surfaces

```c
mov (%rax), %rbx
```

Diagram:
- `%rax` points to `7FFFD0014E20480`
- `%rbx` points to `480`
- Address translation unit: `F0014E20`
- Physical page number: `00882E2`
- Page offset: `480`
- Physical address: `00882E2480`

TLB hit? [Y]
1. Understanding Attack Surfaces

```
mov (%rax), %rbx
```

---

**%rax**
- 7FFFFFF0014E20480
- Address translation unit
- TLB hit?
- physical page number
- physical address
- 4-level page table walk
- hit page structure caches?
- hit caches?
- access physical memory
- EPC page?
- page table entries (PTE)

**%rbx**
- 480
- page offset
1. Understanding Attack Surfaces

mov (%rax), %rbx

%rax \rightarrow 7FFFFFFF014E20480 \rightarrow F0014E20 \rightarrow Address translation unit

TLB hit?  
  \( y \)  
  4-level page table walk
  \( y \)  
  hit page structure caches?  
  \( y \)  
  access physical memory

\( n \)  
Page fault – allocate EPC page

\( y \)  
EPC page?

physical page number

physical address

virtual page number

page offset
1. Understanding Attack Surfaces

```
mov (%rax), %rbx
```

virtual page number

<table>
<thead>
<tr>
<th>%rax</th>
<th>7FFFFFF0014E20480</th>
<th>%rax</th>
</tr>
</thead>
<tbody>
<tr>
<td>%rbx</td>
<td>480</td>
<td>%rbx</td>
</tr>
</tbody>
</table>

physical page number

physical address

page offset

Address translation unit

TLB hit?

4-level page table walk

hit page structure caches?

hit caches?

Page fault – allocate EPC page

access physical memory

EPC page?

page table entries (PTE)
1. Understanding Attack Surfaces

```
mov (%rax), %rbx
```

- `%rax` ➔ `7FFFFFF0014E20480` ➔ `F0014E20`
- `%rbx` ➔ `480`

**Address translation unit:**

- Virtual page number ➔ `00882E2`
- Page offset ➔ `00882E2480`

**TLB hit?**

- `y` ➔ `4-level page table walk`
- `n` ➔ `hit caches?`

**hit caches?**

- `y` ➔ `Page fault – allocate EPC page`
- `n` ➔ `access physical memory`

**Cache fill from physical memory**

**EPC page?**

- `y` ➔ `hit page structure caches?`
- `n` ➔ `page table entries (PTE)`