Introduction to SGX (Software Guard Extensions) and SGX Virtualization

Kai Huang, Jun Nakajima (Speaker)
July 12, 2017
Agenda

- SGX Introduction
- Xen SGX Virtualization Support
- Backup
SGX: Reduce TCB to “HW + Enclave”

**Enclave:**
- A protected container in App’s address space (ring 3).
- Even privileged SW cannot access enclave directly.
- Reduce TCB to “HW + Enclave”

⇒ *App gets its own capability of protection*
SGX: Prevent Memory Snooping Attacks

- Security perimeter is the CPU package boundary
- Data and code unencrypted inside CPU package
- Data outside CPU package is encrypted and/or integrity checked
- External memory reads and bus snooping only see encrypted data

*MEE: SGX Memory Encryption Engine*
SGX Enclave

- **Enclave**
  - Trusted Execution Environment embedded in application
  - Provides confidentiality and/or integrity
  - With its own code/data.
  - With controlled entry points
  - Multiple threads supported

- **EPC (Enclave Page Cache)**
  - Trusted Memory to commit enclave (via page table)
  - With additional access check
  - Typically reserved by BIOS as Processor Reserved Memory
  - Along with EPCM with limited size (ex, 32M, 64M, 128M)

- **New SGX instructions to manage/access Enclave**
  - ENCLS, ENCLU.

*EPCM (Enclave Page Cache Map)*:
Used by HW to track EPC status (not-visible to SW)
Instruction Behavior Changes in Enclave

• Invalid Instructions

<table>
<thead>
<tr>
<th>Instructions</th>
<th>Result</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPUID, GETSEC, RDPMC, SGDT, SIDT, SLDT, STR, VMCALL, VMFUNC</td>
<td>UD</td>
<td>Might cause VM exit.</td>
</tr>
<tr>
<td>IN, INS/INSB/INSW/INSD, OUT, OUTS/OUTSB/OUTSW/OUTSD</td>
<td>UD</td>
<td>I/O fault may not safely recover. May require emulation.</td>
</tr>
<tr>
<td>SMSW</td>
<td>UD</td>
<td>Might provide access to kernel information.</td>
</tr>
<tr>
<td>ENCLU[ENTER], ENCLU[ERESUME]</td>
<td>GP</td>
<td>Cannot enter an enclave from within an enclave.</td>
</tr>
</tbody>
</table>

• Behavior Changes
  – RDTSC, RDTSCP: Only legal when SGX2 is available.
  – RDRAND, RDSEED, PAUSE: May cause VMEXIT
  – INVD: #UD in enclave
  – INT3
SGX Application Flow

1. Define and partition App to untrusted and trusted part.
2. App creates enclave
3. Trusted function is called;
5. Trusted function returns.
6. App continues normal execution.
Agenda

• SGX Introduction
• Xen SGX Virtualization Support
• Backup
Virtualizing SGX

• General Enabling – Pretty Straightforward
  – Discover SGX and Manage EPC in Hypervisor
  – Expose part of the host EPC to guest
    – Size: configurable from user
    – Base: calculated internally
  – SGX CPUID/MSR emulation
  – Setup EPT mapping for guest EPC and host EPC.
    – ENCLS and ENCLU runs perfectly in non-root mode.

• EPC Virtualization Approaches
  – Static Partitioning
  – Oversubscription
  – Ballooning
SGX Interaction with VMX

- New ENCLS VMEXIT
  - New bit in secondary exec control to enable ENCLS VMEXIT
  - New 64-bit bitmap to control which ENCLS leaves will trigger VMEXIT

- New bits to indicate whether VMEXIT (any) is from Enclave
  - Bit 27 in exit_reason
  - Bit 4 in GUEST_INTERRUPTIBILITY_INFO.
**Xen SGX Virtualization Support**

* ENCLS emulation & EPT violation may not needed, depending on implementation of EPC virtualization.
## EPC Virtualization Approaches (3)

<table>
<thead>
<tr>
<th></th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
</table>
| Static Partitioning  | • Easy implementation (no ENCLS trapping/emulation, No EPT violation)  
|                      | • No hypervisor overhead                                            | Potential inefficient use of EPC               |
| Ballooning           | • Pros of “static partitioning”                                    | Require ballooning driver in guest             |
|                      | • More efficient use of EPC                                        |                                                 |
| Oversubscription     | More efficient use of EPC                                          | • Complicated implementation                  |
|                      |                                                                     | • Higher hypervisor overhead                   |

- We have preliminary patches on github with “static partitioning” implemented.
- Oversubscription vs Ballooning?
Questions?