Updates from the RISC-V TEE Group

Nick Kossifidis <mick@ics.forth.gr>
Security-related RISC-V Task Groups
About the TEE Task Group

- One of the most popular groups (112 registered members)
- Regular conference calls / mailing list
- Its mission is:
  - To define an architecture specification for supporting Trusted Execution Environments on RISC-V processors
  - To provide necessary implementation guidelines and/or recommendations in order to assist developers to realize the specification
  - To enable the development of necessary components (hardware and software) to support the specification
Work in progress

• **On the hardware side**
  • Modifications on the Physical Memory Protection (PMP) mechanism
  • Proposal for an I/O Physical Memory Protection (IOPMP) block
  • Proposal for a Control Flow Integrity (CFI) extension

• **On the software side**
  • Secure Monitor architecture

• **TODO**
  • Secure Boot
  • ...

...
Physical Memory Protection on RISC-V

- Part of the Machine ISA (Privilege Spec)
- Per-hart firewall for physical memory access
- 32bit addresses for RV32, 56bit for RV64
- 4 address matching modes
- R/W/X permission handling

<table>
<thead>
<tr>
<th>A</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OFF</td>
<td>Null region (disabled)</td>
</tr>
<tr>
<td>1</td>
<td>TOR</td>
<td>Top of range</td>
</tr>
<tr>
<td>2</td>
<td>NA4</td>
<td>Naturally aligned four-byte region</td>
</tr>
<tr>
<td>3</td>
<td>NAPOT</td>
<td>Naturally aligned power-of-two region, ≥8 bytes</td>
</tr>
</tbody>
</table>

Table 3.8: Encoding of A field in PMP configuration registers.

<table>
<thead>
<tr>
<th>pmaddr</th>
<th>pmcfg.A</th>
<th>Match type and size</th>
</tr>
</thead>
<tbody>
<tr>
<td>yyyy</td>
<td>yyyy</td>
<td>NA4 4-byte NAPOT range</td>
</tr>
<tr>
<td>yyyy</td>
<td>yyyy0</td>
<td>NAPOT 8-byte NAPOT range</td>
</tr>
<tr>
<td>yyyy</td>
<td>yyyy01</td>
<td>NAPOT 16-byte NAPOT range</td>
</tr>
<tr>
<td>yyyy</td>
<td>yyyy011</td>
<td>NAPOT 32-byte NAPOT range</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>2(\times\text{LEN}) byte NAPOT range</td>
</tr>
<tr>
<td>...</td>
<td>0y01</td>
<td>NAPOT 2(\times\text{LEN})+1-byte NAPOT range</td>
</tr>
<tr>
<td>...</td>
<td>y011</td>
<td>NAPOT 2(\times\text{LEN})+2-byte NAPOT range</td>
</tr>
<tr>
<td>...</td>
<td>0111</td>
<td>NAPOT 2(\times\text{LEN})+3-byte NAPOT range</td>
</tr>
<tr>
<td>...</td>
<td>1111</td>
<td>NAPOT 2(\times\text{LEN})+4-byte NAPOT range</td>
</tr>
</tbody>
</table>

Figure 3.26: PMP address register format, RV32.

Figure 3.27: PMP address register format, RV64.

Figure 3.28: PMP configuration register format.
Physical Memory Protection on RISC-V
Virtual memory protection on RISC-V

- Part of the Supervisor ISA (Privilege Spec)
- 32bit virtual addresses for RV32, 39/48bit for RV64
- Radix-tree page table, 4KiB pages with support for 4MiB (RV32) and 2MiB (RV64) “megapages”, 1GiB “gigapages” and 512GiB “terapages” (RV64)
- Each table entry handles R/W/X permissions and the U permission that allows access to that entry from U mode (else it’s S mode only)
- The sstatus.SUM bit allows Supervisor to R/W User mode pages (SMAP) (execution of User mode memory from Supervisor is always denied)
- The sstatus.MXR bit allows executable only pages to also be treated as readable
Proposed PMP modifications

- Currently the only way to limit M mode's access is to use Locked entries, however locked entries are permanent until a hart reset is performed + are also enforced on S/U modes which doesn't make sense since S/U modes can't modify PMP settings anyway (so locking an entry only makes sense for M mode)

- We want to prevent M mode from accessing memory that belongs to S/U modes, to provide the equivalent of S mode's sstatus.SUM bit

- We want to have locked rules that are only enforced on M mode but not on S/U modes (e.g. to allow M mode to only have execute permission, without also allowing S/U to have the same privilege)

- Say hello to Machine Mode Isolation bit on mstatus (mstatus.MMI)!

<table>
<thead>
<tr>
<th>L bit on pmpcfg</th>
<th>mstatus.MMI</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Temporary entry; R/W/X enforced on sub-M modes; M-mode succeeds</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Temporary entry; R/W/X enforced on sub-M modes; M-mode fails</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Locked entry; R/W/X enforced on all modes</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Locked entry; R/W/X enforced on M-mode; sub-M modes fails</td>
</tr>
</tbody>
</table>
I/O PMP Block proposal
Control Flow Integrity extension proposal
Secure Monitor’s architecture

Current implementations from group members

- MultiZone from HexFive (https://hex-five.com/products/)
- Keystone from UC Berkeley (https://keystone-enclave.org/)

A lot of work to be done!

- Define APIs between TEEs and between TEEs and the rest of the world (we need to work together with the upcoming platform specification task group e.g. for the SBI part)
- Define a memory isolation scheme using PMP (there is a draft proposal on that)
- Define a memory isolation scheme for I/O PMP
- Define mechanisms for handling multiple harts
- Define mechanisms for interrupt handling / delegation
- Define common format for TEE binaries (e.g. ELF with extras)
- Write code for all of the above and test it
- Provide an SDK
- ...
Questions ?
Thank you!