BTMMU: An Efficient and Versatile Cross-ISA Memory Virtualization
ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), 2021

Kele Huang, Fuxin Zhang, Cun Li, Gen Niu, Junrong Wu, Tianyi Liu

Institute of Computing Technology, Chinese Academy of Sciences, China
University of Chinese Academy of Sciences, China
Beijing Institute of Technology, China
University of Texas at San Antonio, USA

kele.hwang@gmail.com, https://rivalak.github.io

April 16, 2021
Outline

1. Background and Motivation
2. Framework of BTMMU
3. Experimental Results
4. Conclusion
• **GVA ⇒ GPA**: via GPT (Guest Page Table)
• **GPA ⇒ HVA**: via MMT (Memory Mapping Table)
• **HVA ⇒ HPA**: via HPT (Host Page Table)
What if we can trim down such three-level indirection to **one-level**? From GVA to HPA directly.
What if we can trim down such three-level indirection to one-level? From GVA to HPA directly.

Challenges in cross-ISA memory virtualization?
Challenges in Cross-ISA Memory Virtualization

Challenge 1: different VASs of guest and host

- **The page size**
  - 4k on x86, 8k on UltraSPARC architecture, 16k on loongson

- **The virtual address space (VAS) size**
  - 32 bits on x86_32, 48 bits on x86_64

- **The organization rule**
Challenges in Cross-ISA Memory Virtualization

Challenge 2: GVA and HVA accesses interleaved

X86 assembly

```
mov %ebx, 0(%eax + 0x4)
```

MIPS assembly

```
lw t0, &VCPU.%eax
add t0, t0, 0x4
lw t1, &VCPU.%ebx
sw t1, &VCPU.%ebx
```
Existing Approaches
Approaches for Same-ISA Memory Virtualization

- Shadow Page Table (SPT)
  - SPT stores $GVA \Rightarrow HPA$ mappings.

- Nested Paging
  - Two-level hardware address translation, including EPT and NPT.

- Logical to Real Address Translation (LRAT)
  - here logical address $== GPA$ and real address $== HPA$. 

Approaches for Same-ISA Memory Virtualization

- **Shadow Page Table (SPT)**
  - SPT stores $GVA \Rightarrow HPA$ mappings.

- **Nested Paging**
  - Two-level hardware address translation, including EPT and NPT.

- **Logical to Real Address Translation (LRAT)**
  - Here logical address $==$ GPA and real address $==$ HPA.

Even though they cannot be used for cross-ISA memory virtualization directly, some cross-ISA approaches take advantage of them for improvement.
Existing Cross-ISA Approaches
Pure Software: QEMU and Its Enhanced Edition

- Can solve all those mentioned challenges
- However, its efficiency is too slow.

The reasons:
- Fast path (SoftTLB) is not efficient enough yet
- Two-level method, for SoftTLB is an non-privileged approach
- The hit rate of SoftTLB is not high enough

Even though some methods improved it by optimizing SoftTLB, they cannot improve the essence of SoftTLB (an non-privileged cache).
Existing Cross-ISA Approaches

Hardware-Assisted: ESPT (VEE’14) and HSPT (VEE’15)

- Rationale: Utilize host hardware MMU to do acceleration by embedding the guest’s VAS into the host’s.
- HSPT share the same rationale as ESPT, but it used `mmap syscall` to avoid using host kernel module.

Both of them do shorten the indirection to one-level. But still get **limitations**:

- Guest’s VAS should be much more small than host’s
- Guest’s page size should be the same as host’s
- Privileged memory emulation is not supported
- Use signal mechanism to handle memory emulation exception
Aforementioned cross-ISA approaches have made promising results but still get limitations.
Motivation

Aforementioned cross-ISA approaches have made promising results but still get limitations.

The reason?

- Same-ISA memory virtualization has EPT, NPT and LRAT, et.
- But not designed for cross-ISA in mind.

It can make difference by utilizing dedicated hardware extension for cross-ISA memory virtualization.
Framework of BTMMU
What’s BTMMU?

A Binary Translation Memory Management Unit, consists of:

- low-cost hardware extension
- kernel module
- patched QEMU
Low-cost: Only extends two-bit MID (Machine IDentifier) of host’s VAS.

Flexibility: dedicated exception entrance for memory emulation exception handling.
### Instruction Translation

<table>
<thead>
<tr>
<th>Guest Instruction</th>
<th>Host Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Non-privileged GVA Memory Access</strong></td>
<td></td>
</tr>
<tr>
<td>E.g. MOV %EBX, 0(%EAX); ; GVA in %EAX</td>
<td>SETMID 2;</td>
</tr>
<tr>
<td></td>
<td>Load T1, 0(T0); ; GVA in T0</td>
</tr>
<tr>
<td><strong>Privileged GVA Memory Access</strong></td>
<td></td>
</tr>
<tr>
<td>E.g. MOV 0(%EBX), %EAX; ; GVA in %EBX</td>
<td>SETMID 3;</td>
</tr>
<tr>
<td></td>
<td>Store T0, 0(T1); ; GVA in T1</td>
</tr>
</tbody>
</table>
Software Implementation

- Regular exception handler
- Fast exception handler
- Optimizations for real deployment
  - Multiple SPT
  - MMIO-retranslation
Experimental Results
Methodology

- Platforms
  - System TL630-V001 (Loongson 3A4000)
  - System LENOVO 81J0 (Intel® Core™ i7-8565U)

- Workloads
  - SPEC CINT2006 benchmarks
  - Four real-world applications
  - IOZone

- Virtual Machine: QEMU version 4.2.1
BTMMU outperforms QEMU-i386-softmmu by average 1.40x.

BTMMU outperforms QEMU-x86_64-softmmu by average 1.36x.
Compared to reproduced-HSPT

- On x86 host: to show our proper implementation.
- Use 32-bit guest: HSPT can only support this.
- Guest shares same page size as host: also because of HSPT’s limitation.

(a) Comparison of relative speedup of SPEC CINT2006 (with train input); higher is better.

(b) Comparison of absolute runtime of SPEC CINT2006 (with train input) in seconds; lower is better.
Table: Memory emulation exception in baseline QEMU, HSPT and BTMMU

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Exception Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>QEMU</td>
<td>SoftTLB miss</td>
</tr>
<tr>
<td>HSPT</td>
<td>Segmentation fault</td>
</tr>
<tr>
<td>BTMMU</td>
<td>GTLB invalid exception</td>
</tr>
</tbody>
</table>
In-depth Analysis

**Table: Memory exception rates comparison**

<table>
<thead>
<tr>
<th></th>
<th>Old-QEMU</th>
<th>Base-QEMU</th>
<th>HSPT</th>
<th>BTMMU</th>
</tr>
</thead>
<tbody>
<tr>
<td>445</td>
<td>3.17%</td>
<td>0.0407%</td>
<td>0.0168%</td>
<td>0.0028%</td>
</tr>
<tr>
<td>456</td>
<td>2.20%</td>
<td>0.3759%</td>
<td>0.0034%</td>
<td>0.0012%</td>
</tr>
<tr>
<td>458</td>
<td>4.90%</td>
<td>0.0393%</td>
<td>0.0112%</td>
<td>0.0024%</td>
</tr>
<tr>
<td>464</td>
<td>1.62%</td>
<td>0.0366%</td>
<td>0.0053%</td>
<td>0.0015%</td>
</tr>
<tr>
<td>471</td>
<td>2.24%</td>
<td>1.8128%</td>
<td>0.0158%</td>
<td>0.0038%</td>
</tr>
<tr>
<td>483</td>
<td>3.84%</td>
<td>0.4720%</td>
<td>0.0239%</td>
<td>0.0061%</td>
</tr>
<tr>
<td>Mean</td>
<td>2.79%</td>
<td><strong>0.1630%</strong></td>
<td><strong>0.0104%</strong></td>
<td><strong>0.0026%</strong></td>
</tr>
</tbody>
</table>

The exception rate of BTMMU is the lowest. BTMMU performs better.
Conclusion

- BTMMU: identifies the challenges faced in cross-ISA memory virtualization and proposes an efficient and versatile solution named BTMMU to address them.
  - Dual-TLB
  - Effective software implementation
  - Optimizations for real deployment

- Evaluation:
  - 1.40x speedup on IA32-to-MIPS64
  - 1.36x speedup on X86_64-to-MIPS64
  - Better performance than HSPT
BTMMU: An Efficient and Versatile Cross-ISA Memory Virtualization

ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), 2021

Kele Huang, Fuxin Zhang, Cun Li, Gen Niu, Junrong Wu, Tianyi Liu

Institute of Computing Technology, Chinese Academy of Sciences, China
University of Chinese Academy of Sciences, China
Beijing Institute of Technology, China
University of Texas at San Antonio, USA

kele.hwang@gmail.com, https://rivalak.github.io

April 16, 2021