CRIU On PowerPC

LPC 2015

Laurent Dufour
The PowerPC Family

- Linux kernel supports from Power 3 to current Power 8
- Large CPU family
- Embedded and Server
- 32bits and 64bits
- 32bits and embedded not in the picture
Big Endian and Little Endian

- Power 8 fully support Big Endian (BE) and Little Endian (LE)
- Most of Linux distributions are supporting LE mode
- Some Linux distributions are supporting BE & LE mode
- CRIU support is currently done for LE mode only
- BE support in progress
- Both architecture (ppc64 and ppc64le) should share most of the code in CRIU
  - New directory criu/arch/ppc64
2 Application Binary Interfaces

- Little Endian uses the new ABIv2 (July 2014)
- Big Endian still uses ABIv1 (July 2004)
- ABIv2 is not compatible with ABIv1
- Should have impact on the parasite code's entry points
- Currently only supporting ABIv2 and LE mode
Linux kernel impacts

- Enabling `kcmp()` system call on PowerPC (available in 4.2)
- VDSO remapping tracking (available in 4.2)
- Soft dirty tracking support (in progress)
VDSO remap support

- On PowerPC, kernel keep the vDSO base address per process
- Used to create the signal return trampoline
- The signal return is done through the vDSO
- Calling vDSO's service `__kernel_sigtramp_rt64`
- Address computed from the base of the vDSO
  
  ```
  regs->link = current->mm->context.vdso_base + vdso64_rt_sigtramp;
  ```
- Need kernel patch to catch vDSO remapping operation
VDSO remap support

- New `arch_remap()` hook
- Called by `move_vma()`
- On PowerPC, handling vDSO unmapping and remapping
- Partial unmapping or remapping is not supported
  - May lead to process core dump
  - Not a real issue for CRIU
- Should be applicable to the ARM architecture as well
  - `setup_return()` in `arch/arm64/kernel/signal.c` does
    - `sigtramp = VDSO_SYMBOL(current->mm->context.vdso, sigtramp);`
- PowerPC support provided in kernel 4.2
VDSO trampoline

- Used when restarting over a new vDSO
- Need to jump from checkpointed vDSO to the new one
- X86 code does something like
  - Move immediate64 into ra
    - Jump to (ra)
- That cannot be done in 2 instructions on PowerPC (RISC)
- Some vDSO calls are small
  - __kernel_get_tbfreq as 6 assembly instructions
- The vDSO trampoline's code has to be smaller than that
- The vDSO trampoline is done in 2 steps on PowerPc
VDSO trampoline

Old entry point 1
Old entry point 2
vdso_trampoline

New entry point 2
New entry point 1

Old vDSO

New vDSO
VDSO issues

- What happened if the checkpointed process is running in the vDSO when the checkpoint is done?
  - Danger if the vDSO is changed at restart time
  - Risk if the vDSO manipulate kernel's data

- Code merging against the multiple architectures
  - arch/*/vdso.c
  - arch/*/vdso-pie.c
## Vector registers

- **2 set of vector registers**
  - ALTIVEC: 32 x 128bit registers
  - VSX: 64 x 128bit registers
- **VSX registers overlap ALTIVEC and FPU registers**

<table>
<thead>
<tr>
<th>VSR[n]</th>
<th>VSR doubleword 0</th>
<th>VSR doubleword 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSR[0]</td>
<td>FPR[0]</td>
<td></td>
</tr>
<tr>
<td>VSR[1]</td>
<td>FPR[1]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSR[31]</td>
<td>FPR[31]</td>
<td></td>
</tr>
<tr>
<td>VSR[32]</td>
<td></td>
<td>VR[0]</td>
</tr>
<tr>
<td>VSR[33]</td>
<td></td>
<td>VR[1]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSR[63]</td>
<td></td>
<td>VR[31]</td>
</tr>
</tbody>
</table>
Vector registers

- PowerPc kernel tracks process use of vector registers using internal per thread variables not exposed to user space
- Current implementation in CRIU forces saving/restore of Vector registers even if process didn't touch them
- Working but not fully compliant with signal handling in rare case
- Need kernel patch to retrieve vector registers thread usage.
- In the todo list.
Todo List

- Vector registers full support
- Soft Dirty Tracking
- Docker support
- Transactional Memory
- CPU features checking
Soft Dirty Tracking

- Requiring dealing with free bit in the PTE
- Work in progress
- Current work based on the new swap PTE encoding
  - Commit 2ac7a1a9641c powerpc/mm: Change the swap encoding in pte.
- Targetted to kernel 4.3
- No impact identified on CRIU
Docker Support

- Should be transparent to the architecture
- Currently having issue building docker on PowerPc
Transactional Memory Support

- Power 8 introduced Transactional Memory (TM)
- Checkpoint done during the transaction may abort the transaction
- Current ptrace implementation is not TM-aware
  - https://lkml.org/lkml/2015/5/19/700
- At restart the process should run back into the tbegin with a transaction failure code (TM_CAUSE_RESCHED for instance)
- Process will be expected to run again the transaction once the restart is done
CPU features checking

- AT_HWCAP and AT_HWCAP2 expose various CPU feature
  - PPC_FEATURE_HAS_ALTIVEC
  - PPC_FEATURE_HAS_ALTIVEC
  - PPC_FEATURE2_TAR (TM support)
  - ...

- Implement arc/ppc64/cpu.c backend
Questions?

Laurent Dufour
laurent.dufour@fr.ibm.com
Legal Statement

This work represents the view of the authors and does not necessarily represent the view of IBM. IBM is a registered trademark of International Business Machines Corporation in the United States and/or other countries. Linux is a registered trademark of Linus Torvalds. Other company, product, and service names may be trademarks or service marks of others.