Optimizing Crash Dump in Virtualized Environments

Haibo Chen
Assistant Processor
Parallel Processing Institute
Fudan University

http://ppi.fudan.edu.cn/haibo_chen

Collaborative work with Yijian Huang and Binyu Zang
- **Parallel Processing Institute, Fudan University**
  - Started from 1992
  - [http://ppi.fudan.edu.cn/](http://ppi.fudan.edu.cn/)

- Started research on Xen at the beginning of 2004

- Research focus
  - Using virtualization to improve the **dependability** and **scalability** of computing systems
Xen-related Projects in PPI

**Mercury**  Combining Performance with Dependability Using Self-virtualization

**Lucos**  Live Updating Operating Systems Using Virtualization

Applications

New OS
Xen-related Projects in PPI

**Chaos** Tamper-Resistant Execution in an Untrusted OS Using A Virtual Machine Monitor

![Diagram showing Cloud Applications, OS, and Attack]

- **Chaos**
- Tamper-Resistant Execution in an Untrusted OS
- Using A Virtual Machine Monitor
Xen-related Projects in PPI

Chaos Tamper-Resistant Execution in an Untrusted OS Using A Virtual Machine Monitor
**Xen-related Projects in PPI**

**Cerberus** Scaling Many-core Applications with Clustering of Commodity OSes
Cerberus: Scaling Many-core Applications with Clustering of Commodity OSes
Core Dump on System Crash

• System Crash
  – Painful, but a fact of life
  – Reboot the whole system for recovery

• Crash Dump (or Core Dump)
  – Saving System States into Persistent Storage
    • For future analysis to avoid recurring reboot

• Different level of dumps
  – Minidump, Kernel Dump and Full Dump
Core Dump and Recovery

- On system crash, We need both...
  - **Core dump** – for future diagnosis
  - **Recovery** – minimum downtime

![Diagram showing the process from System Crash to Service available again with Core Dump and System Reboot stages, with downtime indicated.](image-url)
Performance Implication

- High **latency** for full dump
- Low CPU/memory Utilization
  - by *I/O-Intensive* core dump
  - Low CPU Utilization
  - Long-term full memory reservation

System Crash → Core Dump → System Reboot → Service available again
Optimizing Opportunities in Xen

• Optimization opportunities in virtualized environments
  – Crash of a VM is not the end of the world
    • VMM software is still alive
  – Concurrent existence of multiple VMs

• Goal: minimize downtime for core dump & recovery
Optimizing Crash Dump in Xen

- Proposed optimizations
  - *Concurrent* core dump & recovery
  - *Selective* core dump by *VM introspection*
  - Disk I/O *Rate Control*
Outline

• **Design**
  • Implementation
  • Evaluation
  • Conclusion
Typical Crash Dump Tools

• Kdump
  – Based on Kexec
  – Load new kernel from reserved area on panic

• Xen
  – “xc dump-core <domid>”
• Core dump by Xend

• *Reboot-based recovery* with recovery DomainU

• The 2 DomUs *share* the file system
  – Retain persistent states
  – *Safe* sharing
Concurrent Core Dump & Recovery

• Start recovery DomU while core dump still on-the-fly

• Break full reservation of memory by core dump
  – CPU, I/O resources immediately released on crash

• Divide DomU memory into chunks
  – Core dump, reallocate memory to recovery DomainU in chunks
Core Dump and Reallocation of Memory

The Xen Hypervisor

Crashed DomU

Recovery DomU (booting)

Disk

The Xen Hypervisor

Core Dump by Xend

Reallocation by Hypervisor
Selective Dump

• Selective Dump by *VM Introspection*

1. Access Memory of Crashed DomainU
2. Extract page descriptor array
3. Identify free pages
4. Skip core-dumping free pages

• Rely on the **integrity** of guest memory management
Selective Dump

• **Introspect** the page descriptor array in guest Linux
  – *mem_map*

• **Identify** free pages
  – Whose `_count` == 0 in the page descriptor

• **Skip** core-dumping free pages
  – in *xc_domain_dumpcore_via_callback()*
Disk I/O Rate Control

• Allocating I/O bandwidth
  – between concurrent core dump and recovery

• Tune the allocation policy by user

• Trade-off between
  – Speed to release memory by core dump
  – Speed to boot recovery DomainU
• Available in enterprise virtualization products

• Scheduling in open-source Xen
  – ionice in Domain0
  – over kernel threads handling DomainU I/O in Domain0

• Alternative
  – Schedule in hypervisor, which is OS-agnostic
Outline

• Design
  • **Implementation**
• Evaluation
• Conclusion
The Prototype

• Based on Xen 3.3.0
• A utility program in Xen-tools
  – Get notified of DomainU crash by Xend
  – Initiate core dump
  – Start the recovery DomainU
• Optimized core dump routine
  – Modified \texttt{xc\_domain\_dumpcore\_via\_callback()}
• Added a new hypercall
  – Memory reallocation
Outline

• Design
• Implementation
• **Evaluation**
• Conclusion
Environment

• Xen 3.3.0; Debian Linux (kernel 2.6.28)
• TPC-W
  – The TPC-W-lycos Java Implementation
• Machine
  – Intel Core Duo 2 2.33 GHz
  – 2 GB memory
  – 320 SATA disk
Methodology

- Domain0 – 512 MB
- Crashed / Recovery DomainU – 1024 MB
- Disable ballooning
- **Event log**
  - Record moments of starting/ending of core dump/recovery
- **Downtime** = End of recovery – Start of dump
- Domain Resource consumption monitoring
Concurrent Core Dump & Recovery

Memory Consumption (KB)

Time (sec)

- Domain0
- The Virtual Machine to Crash
- The Recovery Virtual Machine
# Selective Dump

<table>
<thead>
<tr>
<th># Pages Skipped</th>
<th># Pages Core-dumped</th>
<th>Downtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>262144</td>
<td>45</td>
</tr>
<tr>
<td>192474</td>
<td>69670</td>
<td>19</td>
</tr>
</tbody>
</table>

* # Pages in Total = 262144

NOTE: not integrated with concurrent dump
Disk I/O Rate Control

• Make I/O contention occur
  – Between core dump and the recovery DomainU
  – Copy a small file when booting the recovery DomainU
    • Not memory-critical

• Case 1: recovery is given higher I/O priority
• Case 2: core dump is given higher I/O priority
Disk I/O Rate Control

**Case-1**
- Recovery has higher I/O priority
- Downtime - 31 sec
- Slow memory reallocation

**Case-2**
- Core Dump has higher I/O priority
- Downtime - 54 sec
- Fast memory reallocation
Outline

• Design
• Implementation
• Evaluation
• Conclusion
Conclusion

• Optimize core dump in Xen
  – Concurrent core dump & recovery
    • Better hardware resource utilization
  – Selective core dump by VM introspection
    • Shorter core dump latency
  – Disk I/O Rate Control
    • Improve I/O QoS

• Future work
  – Integrate the three techniques
  – More tuned QoS control
  – Compression on multi-core VM
Optimizing Crash Dump in Virtualized Environments

Haibo Chen
Assistant Processor
Parallel Processing Institute
Fudan University

http://ppi.fudan.edu.cn/haibo_chen

Collaborative work with Yijian Huang and Binyu Zang
The End

• Thank you
Selective Core Dump

- To access the page descriptor array
  - Start with the *virtual* address of the 1\textsuperscript{st} elem in array
    - Symbol info stored in kernel image
  - DomainU *virtual* address => DomainU *phys* address
    - Linux linear mapping in kernel space
  - DomainU *phys* address => *machine* address
    - Xen p2m table of DomainU, accessible by Xend
  - Map *machine* address
    - `xc_map_foreign_batch()` in Xend
Concurrent Core Dump & Recovery

1. Core dump tool grabs DomainU pages
2. Release reference by DomainU to its pages
   1. Drop reference count to 1, by relinquish_memory()
3. for each chunk:
   – Core_dump(chunk)
   – Release(chunk)
     • Drop its reference count to 0
     • Reclaimed by hypervisor
   – Add chunk to recovery DomainU
   – \textit{Start recovery DomainU} if 64 MB reclaimed in total