Transcendent Memory on Linux

Speaker: Dan Magenheimer
Oracle Corporation
Agenda

- Motivation and Challenge
- Overview of Physical Memory Management
- Transcendent Memory ("tmem") Overview
- Transcendent Memory in Action
- Status, Futures, etc.
Motivation

- **Memory** is increasingly becoming a bottleneck in **virtualized** system
- Existing mechanisms have major holes
The Virtualized Physical Memory Resource Optimization Challenge

Optimize, across time, the distribution of machine memory among a maximal set of virtual machines by:

• measuring the current and future memory need of each running VM and

• reclaiming memory from those VMs that have an excess of memory and either:
  • providing it to VMs that need more memory or
  • using it to provision additional new VMs.

• *without* suffering a significant performance penalty
The Virtualized Physical Memory Resource Optimization Challenge

Optimize, across time, the distribution of machine memory among a maximal set of virtual machines by:

• measuring the current and future memory need of each running VM and
• reclaiming memory from those VMs that have an excess of memory and either:
  • providing it to VMs that need more memory or
  • using it to provision additional new VMs.
• without suffering a significant performance penalty

…..Why is this a hard problem?
Agenda

• Motivation and Challenge
• **Overview of Physical Memory Management**
  • *in an operating system*
  • in a virtual machine monitor (Xen)
• Transcendent Memory Overview
• Transcendent Memory In Action
• Status, Futures, etc.
OS Physical Memory Management

- Operating systems are memory hogs!
OS Physical Memory Management

- Operating systems are memory hogs!

If you give an operating system more memory.....

New larger memory constraint
• Operating systems are memory hogs!

...it uses up any memory you give it!
What does an OS do with all that memory?

- Kernel code and data
- User code and data
- Page cache!
What does an OS do with all that memory?

Page cache attempts to predict future needs of pages from the disk...

sometimes it gets it right

→ “good” pages
OS Physical Memory Management

• What does an OS do with all that memory?

Page cache attempts to predict future needs of pages from the disk...

sometimes it gets it wrong → “wasted” pages
OS Physical Memory Management

- What does an OS do with all that memory?
  - ...much of the time
    - mostly page cache
  - ... some of which will be useful in the future
  - ... and some of which is wasted
Agenda

- Motivation and Challenge
- Overview of Physical Memory Management
  - in an operating system
  - *in a virtual machine monitor (Xen)*
- Transcendent Memory Overview
- Transcendent Memory In Action
- Status, Futures, etc.
VMM Physical Memory Management

- Xen partitions memory
- hypervisor memory
- dom0 memory
- guest memory
- “leftover” (in case needed later)

Dom0 is special 😊

guest

Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer
VMM Physical Memory Management

- Xen partitions memory
  - Xen memory
  - dom0 memory
  - guest 1 memory
  - guest 2 memory
  - whatever’s left over: “fallow” memory

*fallow*, adj., land left without a crop for one or more years
VMM Physical Memory Management

- Xen partitions memory
  - Xen memory
  - dom0 memory
  - guest 1 memory
  - guest 2 memory
  - whatever’s left over: “fallow” memory

fallow, adj., land left without a crop for one or more years
VMM Physical Memory Management

- Xen partitions memory among more guests
  - Xen memory
  - dom0 memory
  - guest 1 memory
  - guest 2 memory
  - guest 3…
- BUT still *fallow memory* leftover
VMM Physical Memory Management
in the presence of migration

- migration
  - requires fallow memory in the target machine
  - leaves behind fallow memory in the originating machine
VMM Physical Memory Management
in the presence of ballooning

- Use ballooning to allow guest memory size to grow?
  - Goal: fill fallow memory
VMM Physical Memory Management

in the presence of ballooning

• Look! No more fallow memory!

But…. 
VMM Physical Memory Management in the presence of ballooning

- Look! No more fallow memory!

But....
VMM Physical Memory Management
in the presence of ballooning

- Look! No more fallow memory!

But….

And but…
VMM Physical Memory Management
in the presence of ballooning

- Ballooning down to handle incoming migrations
  - NASTY issues!

To make room for a new guest, we have to starve existing guests!
Using ballooning to take memory away:
- not instantaneous (*memory inertia*)
- guest can’t predict future needs
  - good pages are evicted along with the bad
- don’t know how much/fast to balloon
  - Too much or too fast
    → thrashing or the dreaded OOM killer
The Virtualized Physical Memory Resource Optimization Challenge

Optimize, across time, the distribution of machine memory among a maximal set of virtual machines by:

- measuring the current and future memory need of each running VM and
- reclaiming memory from those VMs that have an excess of memory and either:
  - providing it to VMs that need more memory or
  - using it to provision additional new VMs.
- *without* suffering a significant performance penalty

.....*This IS a hard problem!!!*
Why this IS a hard problem!

Summary

• OS’s use as much memory as they are given
  • but cannot predict the future so often guess wrong
  • and often much memory owned by an OS is wasted

• Xen leaves large amounts of memory fallow
  • fixed partitioning results in fragmentation
  • migration requires fallow memory to succeed

• Ballooning helps but:
  • can’t predict future memory needs of guests
  • memory has inertia
  • the price of incorrect guesses can be dire

→ NEED A NEW APPROACH TO VIRTUALIZED PHYSICAL MEMORY MANAGEMENT!!
Agenda

- Motivation and Challenge
- Overview of Physical Memory Management
- Transcendent Memory Overview
- Transcendent Memory In Action
- Status, Futures, etc.
Transcendent memory
creating the transcendent memory pool

• Step 1a: reclaim fallow memory
• Step 1b: reclaim wasted guest memory (e.g. via ballooning)
• Step 1c: collect it all into a pool
Transcendent memory
creating the transcendent memory pool

• Step 2: provide *indirect* access, strictly controlled by the hypervisor and dom0
Transcendent memory API characteristics

- OS changes (small)
- narrow
- well-specified
- operations are:
  - synchronous
  - page-oriented (one page per op)
  - copy-based
  - multi-faceted
  - extensible
Transcendent Memory
Linux “tmem_ops” API

int (*new_pool)(struct tmem_pool_uuid uuid, u32 flags);

int (*put_page)(u32 pool_id, u64 object, u32 index, unsigned long pfn);

int (*get_page)(u32 pool_id, u64 object, u32 index, unsigned long pfn);

int (*flush_page)(u32 pool_id, u64 object, u32 index);

int (*flush_object)(u32 pool_id, u64 object);

int (*destroy_pool)(u32 pool_id);
Transcendent memory

Xen hypercall API

```c
int xen_tmem_op(
    u32 tmem_cmd, u32 tmem_pool, u64 object, u32 index, unsigned long gmfn, u32 tmem_offset, u32 pfn_offset, u32 len);

int xen_tmem_new_pool(
    struct tmem_pool_uuid uuid, u32 flags);
```
Transcendent memory
four different subpool types
→ four different uses

Legend:
Implemented and working today (Linux + Xen)
Now in 2.6.32 patch
Under investigation

<table>
<thead>
<tr>
<th>flags</th>
<th>ephemeral</th>
<th>persistent</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>“second-chance” clean-page cache!! → “cleancache”</td>
<td>Fast swap “device”!! → “frontswap”</td>
</tr>
<tr>
<td>shared</td>
<td>server-side cluster filesystem cache → “shared cleancache”</td>
<td>inter-guest shared memory?</td>
</tr>
</tbody>
</table>

ephemeral, adj., ... transitory, existing only briefly, short-lived (i.e. NOT persistent)
Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer

Transcendent memory
caveats

• Requirements
  • guest OS kernel must be “tmem-aware”
  • 64-bit Xen hypervisor and CPU (32-bit guests OK)

• Workload:
  • should exert memory pressure in at least one guest
  • memory pressure in multiple guests should vary across time

• For best results:
  • dom0 should be configured with a fixed memory size
  • guest should have a (virtual) swap disk configured

• Complementary to:
  • feedback-directed ballooning (e.g. “self-ballooning”)
  • transparent content-based page sharing
Transcendent memory

Linux core tmem 2.6.32 patch diffstat

Documentation/transcendent-memory.txt | 176 ++++++
include/linux/tmem.h | 88 +++
mm/Kconfig | 10
mm/Makefile | 1
mm/tmem.c | 97 ++

5 files changed, 372 insertions (+)

[PATCH 1/4] Declares the tmem_ops accessors and initializes them as no-ops (foundation layer for “cleancache” and “frontswap”)
(see http://lkml.org/lkml/2009/12/17/443 for patchset)
Agenda

- Motivation and Challenge
- Overview of Physical Memory Management
- Transcendent Memory Overview
- Transcendent Memory In Action
  - *private-ephemeral pool* → “cleancache”
  - *shared-ephemeral pool* → “shared cleancache”
  - *private-persistent pool* → “frontswap”
  - performance analysis
- Status, Future, etc.
cleancache*

- a **second-chance** clean page cache for a guest
  - “put” clean pages only
  - “get” only valuable pages
  - pages eventually are evicted
  - coherency managed by guest
  - exclusive cache semantics

* previously called “precache”

<table>
<thead>
<tr>
<th>Transcendent Memory Pool types</th>
<th>ephemeral</th>
<th>persistent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>private</strong></td>
<td>second-chance clean-page cache!!</td>
<td>Fast swap “device”!!</td>
</tr>
<tr>
<td></td>
<td>“cleancache” →</td>
<td>“frontswap”</td>
</tr>
<tr>
<td><strong>shared</strong></td>
<td>server-side cluster filesystem cache?</td>
<td>inter-guest shared memory?</td>
</tr>
<tr>
<td></td>
<td>“shared cleancache”</td>
<td></td>
</tr>
</tbody>
</table>

Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer
clean cache (with compression)

- Compression
  - Optional (per-guest)
  - nominally doubles available memory
  - performance-space tradeoff
  - some fragmentation issues
cleancache (multiple guests)

- second-chance page cache for *multiple* guests

Need “memory scheduler”:
- global admission/eviction policy:
  - LRU queue
  - weight balanced
shared cleancache (for clustering)

- guests sharing a clustered filesystem (ocfs2)
  - non-exclusive
  - LFU instead of LRU
  - compression optional
  - a server-side disk cache!
- security issues being worked
  - resolved for 2.6.32

<table>
<thead>
<tr>
<th>Transcendent Memory Pool types</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ephemeral</strong></td>
</tr>
<tr>
<td>private</td>
</tr>
<tr>
<td>&quot;second-chance&quot; clean-page cache!</td>
</tr>
<tr>
<td>&quot;cleancache&quot;</td>
</tr>
<tr>
<td>guest</td>
</tr>
<tr>
<td>shared</td>
</tr>
<tr>
<td>server-side cluster filesystem cache?</td>
</tr>
<tr>
<td>&quot;shared cleancache&quot;</td>
</tr>
</tbody>
</table>

Clustered filesystem

SHARED ephemeral tmem pool

guest

guest

Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer
Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer

Transcendent memory
Linux cleancache API

```c
extern void cleancache_init(struct super_block *sb);
extern int cleancache_get(struct address_space *mapping,
                           unsigned long index, struct page *empty_page);
extern int cleancache_put(struct address_space *mapping,
                           unsigned long index, struct page *page);
extern int cleancache_flush(struct address_space *mapping,
                             unsigned long index);
extern int cleancache_flush_inode(
                                   struct address_space *mapping);
extern int cleancache_flush_filesystem(
                                       struct super_block *sb);
```
Transcendent memory

Linux cleancache 2.6.32 patch diffstat

<table>
<thead>
<tr>
<th>File</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>fs/buffer.c</td>
<td>5</td>
</tr>
<tr>
<td>fs/ext3/super.c</td>
<td>2</td>
</tr>
<tr>
<td>fs/btrfs/extent_io.c</td>
<td>9 +</td>
</tr>
<tr>
<td>fs/btrfs/super.c</td>
<td>2</td>
</tr>
<tr>
<td>fs/ext4/super.c</td>
<td>2</td>
</tr>
<tr>
<td>fs/ocfs2/super.c</td>
<td>2</td>
</tr>
<tr>
<td>fs/super.c</td>
<td>5</td>
</tr>
<tr>
<td>include/linux/fs.h</td>
<td>7 +</td>
</tr>
<tr>
<td>mm/truncate.c</td>
<td>10 +</td>
</tr>
<tr>
<td>mm/filemap.c</td>
<td>11 +</td>
</tr>
<tr>
<td>mm/Kconfig</td>
<td>8 +</td>
</tr>
<tr>
<td>mm/Makefile</td>
<td>1</td>
</tr>
<tr>
<td>include/linux/cleancache.h</td>
<td>55 +++++++</td>
</tr>
<tr>
<td>mm/cleancache.c</td>
<td>183 +++++++</td>
</tr>
</tbody>
</table>

19 files changed, 310 insertions(+)

Small handful of hooks needed to utilize cleancache API

frontswap*

- over-ballooned guests experiencing unexpected memory pressure have an emergency swap disk
  - much faster than swapping
  - persistent (“dirty”) pages OK
  - prioritized higher than cleancache
  - limited by guest’s maxmem
  * previously called preswap

Transcendent Memory Pool types

<table>
<thead>
<tr>
<th></th>
<th>ephemeral</th>
<th>persistent</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>&quot;second-chance&quot; clean-page cache!! → “cleancache”</td>
<td>Fast swap “device”!! → “frontswap”</td>
</tr>
<tr>
<td>shared</td>
<td>server-side cluster filesystem cache? → “shared cleancache”</td>
<td>inter-guest shared memory?</td>
</tr>
</tbody>
</table>

OOM
Transcendent memory
Linux frontswap API

extern void frontswap_init(unsigned type);
extern int frontswap_put(struct page *page);
extern int frontswap_get(struct page *page);
extern void frontswap_flush(unsigned type,
   unsigned long offset);
extern void frontswap_flush_area(unsigned type);
extern void frontswap_shrink(unsigned long target_pages);
extern int frontswap_test(struct swap_info_struct *sis,
   unsigned long offset);

(Note: “type” is Linux term for which swap disk, an index into swap_info_struct array)
Transcendent Memory

Linux frontswap 2.6.32 patch diffstat

8 files changed, 441 insertions(+), 6 deletions(-)

[PATCH 3/4] Somewhat more invasive than cleancache due to new frontswap_map array and “partial swapoff”

(see http://lkml.org/lkml/2009/12/17/443 for patchset)
Transcendent memory
Linux tmem-xen glue 2.6.32 patch diffstat

```
arch/x86/include/asm/xen/hypercall.h|   8 +
drivers/xen/Makefile                |    1
drivers/xen/tmem.c                  |  97 ++++++++++++ 
include/xen/interface/tmem.h        |   43 ++++
include/xen/interface/xen.h         |   22 ++
5 files changed, 171 insertions(+) 
```

[PATCH 4/4] Changes in Xen-specific code to layer
Linux tmem API on xen hypercall

(see http://lkml.org/lkml/2009/12/17/443 for patchset)
Transcendent Memory

Linux 2.6.32 tmem patch diffstat

showing changed existing files only

<table>
<thead>
<tr>
<th>File</th>
<th>Insertions (+)</th>
<th>Deletions (-)</th>
</tr>
</thead>
<tbody>
<tr>
<td>fs/buffer.c</td>
<td>5 ++</td>
<td></td>
</tr>
<tr>
<td>fs/ext3/super.c</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>fs/btrfs/extent_io.c</td>
<td>9 +++</td>
<td></td>
</tr>
<tr>
<td>fs/btrfs/super.c</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>fs/ext4/super.c</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>fs/ocfs2/super.c</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>fs/mpage.c</td>
<td>8 ++</td>
<td></td>
</tr>
<tr>
<td>fs/super.c</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>mm/filemap.c</td>
<td>11 ++++</td>
<td></td>
</tr>
<tr>
<td>mm/page_io.c</td>
<td>12 ++++</td>
<td></td>
</tr>
<tr>
<td>mm/swapfile.c</td>
<td>43 ++++++++++++</td>
<td></td>
</tr>
<tr>
<td>mm/truncate.c</td>
<td>10 +++</td>
<td></td>
</tr>
<tr>
<td>kernel/sysctl.c</td>
<td>12 +++</td>
<td></td>
</tr>
<tr>
<td>include/linux/fs.h</td>
<td>7 ++</td>
<td></td>
</tr>
<tr>
<td>include/linux/swap.h</td>
<td>51 ++++++++++++</td>
<td></td>
</tr>
<tr>
<td>include/linux/sysctl.h</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>mm/Kconfig</td>
<td>26 +++++</td>
<td></td>
</tr>
<tr>
<td>mm/Makefile</td>
<td>3 +</td>
<td></td>
</tr>
</tbody>
</table>

18 files changed, 204 insertions (+), 7 deletions (-)

Agenda

- Motivation and Challenge
- Overview of Physical Memory Management
- Transcendent Memory Overview
- Transcendent Memory In Action
  - private-ephemeral pool → “cleancache”
  - shared-ephemeral pool → “shared cleancache”
  - private-persistent pool → “frontswap”
- performance analysis
- Status, Future, etc.
performance analysis: screenshot
performance analysis: workload

Eight VM’s, each running:

```
# service xenballoond start
# cd src/linux-2.6.30
# while true; do
   make clean; make -j4;
   done
```
performance analysis: data collection
performance analysis: data overview

Quad-core physical CPU, single socket, dual-thread

8 guests with 384MB and 2 vcpus each

Staggered starts, some running for nearly a virtual hour, some just started

Transcendent Memory on Linux (linux.conf.au Wellington 2010) - Dan Magenheimer
performance analysis: memory pressure

- Aggressive self-ballooning
  - launched as an initd service
  - feedback-driven using \textit{CommittedAS} from /proc/meminfo
  - considering moving in-kernel

See: Magenheimer, D., \textit{Memory Overcommit… without the Commitment}, Xen Summit 2008
performance analysis: kernel data

normal /proc/vmstat data from each VM
performance analysis: tmem data

Important per-VM tmem statistics
performance analysis: cleancache

- cleancache benefit
  - reduced page-in’s from external storage:
    - by 30-40%
    - by 200-250/sec

(note, cleancache does not reduce page-out’s… pgout shown only for reference)
performance analysis: frontswap

- frontswap benefit
- eliminated all physical swapping
- reduced I/Os to external storage:
  - by 128K writes
  - by 80K reads
  - read+write: ~70/sec
**performance analysis: CPU cost**

- **Tmem total cost**
  - instrumented in hypervisor
  - approximation only
  - CPU overhead:
    - 0.08%-0.15%

<table>
<thead>
<tr>
<th>pgout</th>
<th>PRCCHin</th>
<th>TMEMsec</th>
</tr>
</thead>
<tbody>
<tr>
<td>3884505</td>
<td>699707</td>
<td>4.8</td>
</tr>
<tr>
<td>3794961</td>
<td>706921</td>
<td>4.7</td>
</tr>
<tr>
<td>3362377</td>
<td>583137</td>
<td>4.3</td>
</tr>
<tr>
<td>2489733</td>
<td>353064</td>
<td>3.1</td>
</tr>
<tr>
<td>2426817</td>
<td>319615</td>
<td>3.4</td>
</tr>
<tr>
<td>1837041</td>
<td>271033</td>
<td>2.5</td>
</tr>
<tr>
<td>776258</td>
<td>68403</td>
<td>0.9</td>
</tr>
<tr>
<td>32289</td>
<td>9</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Time(s)**
- 3332.9
- 3220.5
- 3167.7
- 2617.4
- 2617.0
- 1798.4
- 1188.0
- 30.0
performance analysis: summary

• Bottom line:

~0.1% of one core
saves ~300 io/sec

(depending on workload, your mileage may vary, objects appear smaller in the mirror, etc., etc.)
Agenda

• Motivation and Challenge
• Overview of Physical Memory Management
• Transcendent Memory Overview
• Transcendent Memory In Action
• **Status, Future, etc.**
Current Status

- Cleancache, frontswap, and shared cleancache patches for 2.6.32 posted to LKML
  - now includes support for btrfs and ext4
  - most useful statistics available via sysfs
- Tmem to be released in Xen 4.0 (*target release Feb 2010*)
  - Linux tmem patch checked-in in 2.6.18-xen tree
  - save/restore and live migration support fully functional
- Tmem *already* released in Oracle VM 2.2 (*Oct 2009*)
  - “technology preview”
Future Work

• possible integration with ramzswap and FS-cache
  (*Nitin Gupta investigating*)
• combine tmem with page-sharing
• in-kernel self-ballooning
• shared-persistent pool investigation
  • inter-guest communication?
• real world performance measurement/analysis
  • identify any tuning opportunities, rinse, repeat
• tmem for:
  • linux containers?
  • lguest? KVM?
  • fully-virtualized guests?
  • embedded?
Acknowledgements

- Chris Mason (Oracle)
  - Linux vfs changes for cleancache
  - Btrfs help
- Zhigang Wang (Oracle)
  - Xen tools (xm + libxc) code
- Kurt Hackel, Dave McCracken (Oracle)
  - design and debugging help
- Sunil Mushran, Joel Becker (Oracle)
  - ocfs2 help for shared cleancache
- Jeremy Fitzhardinge, Ian Pratt, Keir Fraser, Jan Beulich
  - hard questions and good advice
- Excellent feedback on lkml and xen-devel
With apologies to the late, great Bob Hope…

Thanks for the memory…
I really could use more
My throughput’s on the floor
My balloon is flat, my swap disk’s fat
I’ve O-O-M’s in store
Overcommitted so much…
And Thank YOU so much!

FOR MORE INFORMATION:
http://oss.oracle.com/projects/tmem
Transcendent Memory on Linux

http://oss.oracle.com/projects/tmem

Speaker: Dan Magenheimer
Oracle Corporation

dan.magenheimer@oracle.com