

Virtual Memory

A Project for CS854

Nick Chen
Simon Pratt
Krishna Vaidyanathan

University of Waterloo

February 25, 2016

Our proposal has 3 parts:

Our proposal has 3 parts:

- 1 Literature Review

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ Implementation

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ Implementation

Proposal: Literature Review

We wish to investigate the following operating systems:

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?
- Are there data structures for physical pages, separate from the page tables?

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?
- Are there data structures for physical pages, separate from the page tables?
- How are contiguous regions of memory managed?

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?
- Are there data structures for physical pages, separate from the page tables?
- How are contiguous regions of memory managed?
- How is memory freed?

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?
- Are there data structures for physical pages, separate from the page tables?
- How are contiguous regions of memory managed?
- How is memory freed?
 - What happens when the kernel runs out of memory?

Proposal: Literature Review

We wish to investigate the following operating systems:

- 1 Linux
- 2 NetBSD
- 3 OpenIndiana
(Previously Solaris)

For each OS, we wish to answer the following questions:

- How is physical memory managed?
- Are there data structures for physical pages, separate from the page tables?
- How are contiguous regions of memory managed?
- How is memory freed?
 - What happens when the kernel runs out of memory?
- Do they do anything special on Non-Uniform Memory Access (NUMA) architectures?

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ Implementation

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ Implementation

Proposal: Experimental Design

Proposal: Experimental Design

- Make a *testable* hypothesis based on lit. review

Proposal: Experimental Design

- Make a *testable* hypothesis based on lit. review
- Design *simple* experiments to test this hypothesis

Proposal: Experimental Design

- Make a *testable* hypothesis based on lit. review
- Design *simple* experiments to test this hypothesis
- Example:

Proposal: Experimental Design

- Make a *testable* hypothesis based on lit. review
- Design *simple* experiments to test this hypothesis
- Example:
 - Implement data structures from different VMs

Proposal: Experimental Design

- Make a *testable* hypothesis based on lit. review
- Design *simple* experiments to test this hypothesis
- Example:
 - Implement data structures from different VMs
 - Test performance

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ Implementation

Our proposal has 3 parts:

- ① Literature Review
- ② Experimental Design
- ③ **Implementation**

Proposal: Implementation

Proposal: Implementation

- Optional

Proposal: Implementation

- Optional
- Implement a memory management system for KOS

Proposal: Implementation

- Optional
- Implement a memory management system for KOS
- Use findings from:

Proposal: Implementation

- Optional
- Implement a memory management system for KOS
- Use findings from:
 - Lit review

Proposal: Implementation

- Optional
- Implement a memory management system for KOS
- Use findings from:
 - Lit review
 - Experiments

We now understand the VM design at a high-level:

We now understand the VM design at a high-level:

- Linux

High-level design

We now understand the VM design at a high-level:

- Linux
- NetBSD

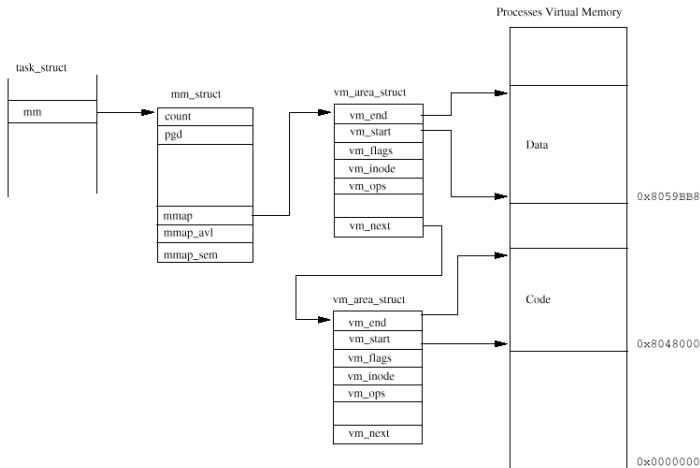
We now understand the VM design at a high-level:

- Linux
- NetBSD
- OpenIndiana

High-level: Linux

- vm_area_struct

```
44 struct vm_area_struct {
45     struct mm_struct * vm_mm;
46     unsigned long vm_start;
47     unsigned long vm_end;
48
49     /* linked list of VM areas per task, sorted by address */
50     struct vm_area_struct *vm_next;
51
52     pgprot_t vm_page_prot;
53     unsigned long vm_flags;
54
55     rb_node_t vm_rb;
56
57     struct vm_area_struct *vm_next_share;
58     struct vm_area_struct **vm_pprev_share;
59
60     /* Function pointers to deal with this struct. */
61     struct vm_operations_struct * vm_ops;
62
63     /* Information about our backing store: */
64     unsigned long vm_pgoff;
65     struct file * vm_file;
66     unsigned long vm_raend;
67     void * vm_private_data;
68 };
```



High-level: NetBSD

- Based on 386BSD, 4.4BSD-Lite

High-level: NetBSD

- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor

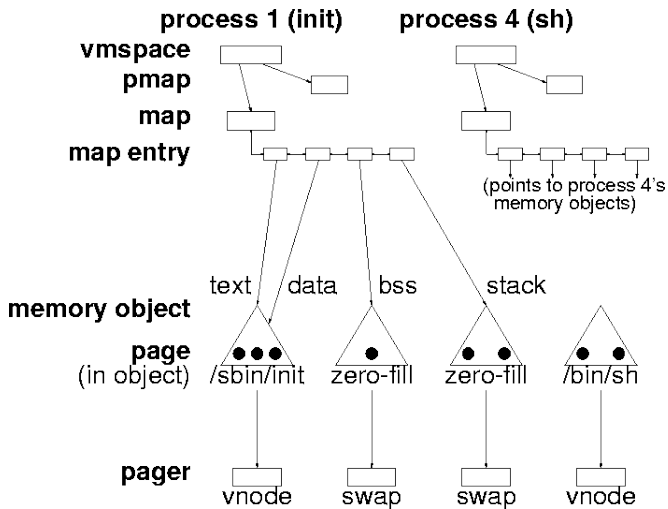
High-level: NetBSD

- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor
 - UVM (Unified(?) VM)

High-level: NetBSD

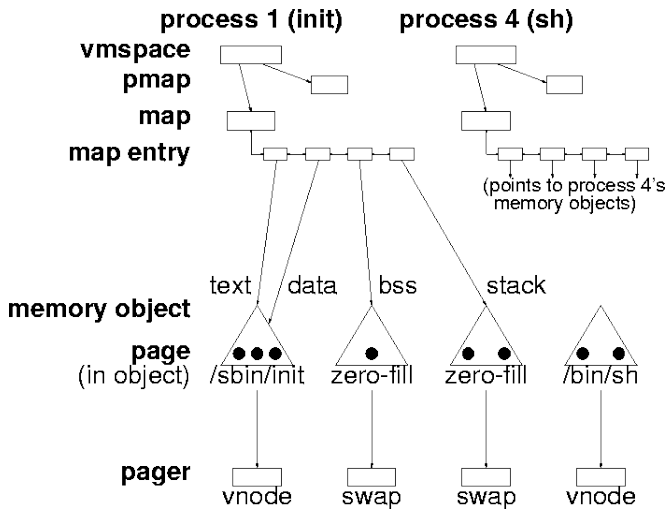
- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor
 - UVM (Unified(?) VM)
 - 13 page Usenix paper

High-level: NetBSD



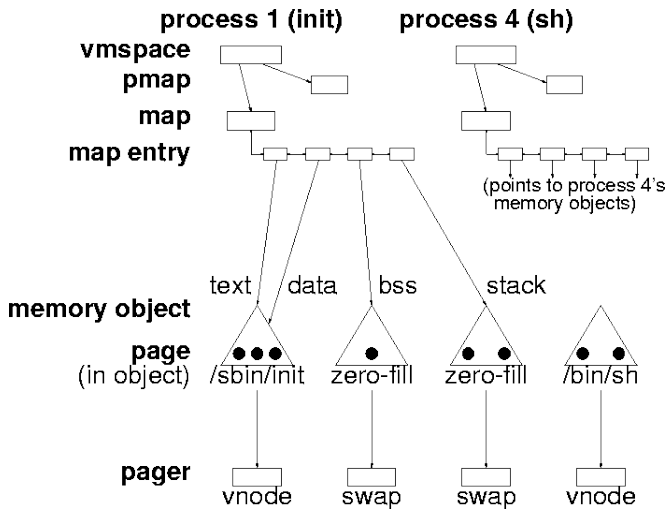
- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor
 - UVM (Unified(?) VM)
 - 13 page Usenix paper
 - 270 page PhD dissertation

High-level: NetBSD



- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor
 - UVM (Unified(?) VM)
 - 13 page Usenix paper
 - 270 page PhD dissertation
- Slightly modified in 2001 by Chuck Silvers
 - UBC (Unified Buffer Cache)
 - 5 page Usenix paper

High-level: NetBSD



- Based on 386BSD, 4.4BSD-Lite
- Rewritten in 1998 by Chuck Cranor
 - UVM (Unified(?) VM)
 - 13 page Usenix paper
 - 270 page PhD dissertation
- Slightly modified in 2001 by Chuck Silvers
 - UBC (Unified Buffer Cache)
 - 5 page Usenix paper
- Minor modifications since then

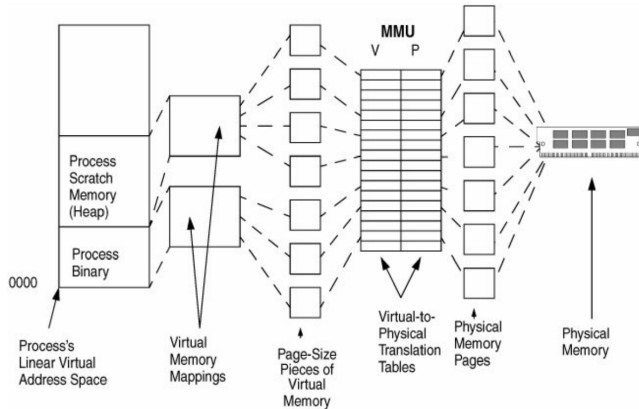
High-level: OpenIndiana

- 1 Open source fork of OpenSolaris after Oracle take over
- 2 Stewarded by the Illumos Foundation

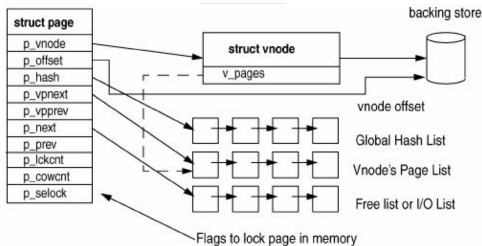
Virtual memory management unit

- 1 Solaris kernel breaks up virtual address space into mappings for each type of memory (eg., heap, stack)
- 2 Hardware MMU maps pages to physical memory using platform-specific translation tables
- 3 Memory management to manage pages is basically swapping and demand paging

Solaris 10 Virtual to Physical Memory Management



Solaris 10 page table structure



- Page table structure different from x86 hardware page table structure

We have found some significant differences so far:

We have found some significant differences so far:

- What happens when the kernel runs out of memory?

We have found some significant differences so far:

- What happens when the kernel runs out of memory?
- How does the kernel access user memory?

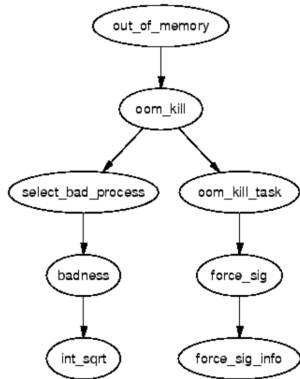
We have found some significant differences so far:

- What happens when the kernel runs out of memory?
- How does the kernel access user memory?
- What are the copy-on-write mechanisms?

What happens when the kernel runs out of memory?

Linux:

- Start killing processes



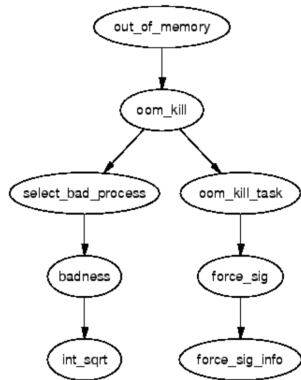
What happens when the kernel runs out of memory?

Linux:

- Start killing processes

NetBSD:

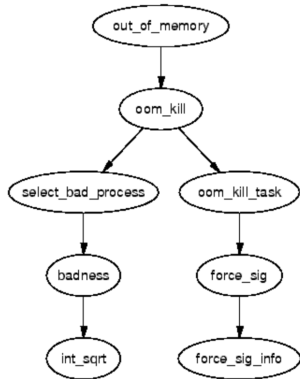
- Panic!



What happens when the kernel runs out of memory?

Linux:

- Start killing processes



NetBSD:

- Panic!

OpenIndiana:

- Periodically checks kernel space, and "snaps" data to user space if kernel space is low
- If kernel runs out of memory, crashes as far as I can tell

How does the kernel access user memory?

Linux:

- Map all of physical memory into kernel's virtual address space

How does the kernel access user memory?

Linux:

- Map all of physical memory into kernel's virtual address space

NetBSD:

- Copy data from user space to kernel space

How does the kernel access user memory?

Linux:

- Map all of physical memory into kernel's virtual address space

NetBSD:

- Copy data from user space to kernel space

OpenIndiana:

- ???

What are the copy-on-write mechanisms?

Linux:

- Page-based copy

What are the copy-on-write mechanisms?

Linux:

- Page-based copy

OpenIndiana:

- Anonymous maps

What are the copy-on-write mechanisms?

Linux:

- Page-based copy

OpenIndiana:

- Anonymous maps

NetBSD:

- Copied SunOS/Solaris

Summary

- ① Literature Review
 - High-level design
 - Differences
- ② Experimental Design
- ③ Implementation

References

- UVM dissertation:
<http://vorpai.math.drexel.edu/course/opsys2/uvm-project/uvm.pdf>
- UVM paper:
https://www.usenix.org/legacy/event/usenix99/full_papers/cranor/cranor.pdf
- UBC paper:
<https://www.usenix.org/legacy/publications/library/proceedings/usenix2000/freenix/silvers.html>
- *Understanding the Linux Virtual Memory Manager*
<https://www.kernel.org/doc/gorman/html/understand/index.html>
- McDougall, Richard, and Jim Mauro. Solaris internals: Solaris 10 and OpenSolaris kernel architecture. Pearson Education, 2006.

- NetBSD data structure diagram from:

http://usenix.org/legacy/publications/library/proceedings/usenix99/full_papers/cranor/cranor_html/index.html

- Linux vm_area_struct source from:

???

- Linux data structures diagram from:

???

- Linux OOM diagram from:

???

- Solaris VM diagram:

McDougall, Richard, and Jim Mauro. Solaris internals: Solaris 10 and OpenSolaris kernel architecture. Pearson Education, 2006.

- These slides are distributed under the creative commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0).
- See <http://creativecommons.org/licenses/by-sa/4.0/> for details.