

Compressed RAM: Impacts on Live Forensics

Golden G. Richard III
Professor of Computer Science
University of New Orleans

Arcane Alloy, LLC
golden@cs.uno.edu

Irfan Ahmed
Assistant Professor of Computer Science
University of New Orleans
iahmed4@uno.edu

Who?



**Professor of Computer Science and University Research
Professor, Director, GNOCIA, University of New Orleans**

<http://www.cs.uno.edu/~golden>

Digital forensics, OS internals, reverse engineering, offensive computing,
pushing students to the brink of destruction, et al.



Founder, Arcane Alloy, LLC.

<http://www.arcanealloy.com>

Digital forensics, reverse engineering, malware analysis, security research,
tool development, training.



Co-Founder, Partner / Photographer, High ISO Music, LLC.

<http://www.highisomusic.com>

Rock stars. Heavy Metal. Earplugs.

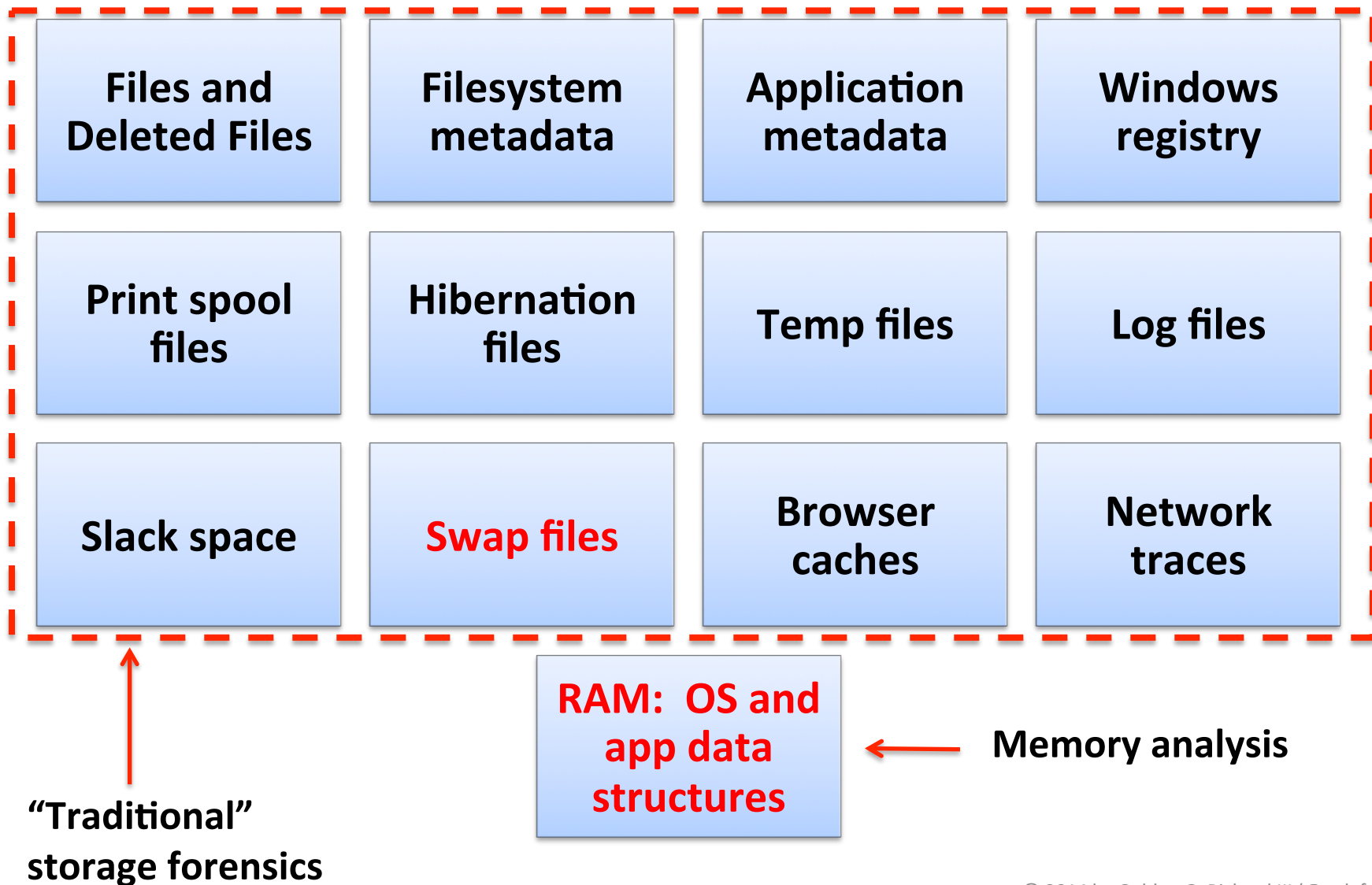
Disclosure / ACK

- The talk mentions the following commercial software, historical and current:
 - Microsoft Windows
 - Mac OS X
 - RAM Doubler
- I am a user of the first two, have experience with the third, and have no financial interest in any of them
- **ACK: Andrew Case**, who is now handling most of the Linux side of this research while I target Mac OS X

Forensics, Features, Privacy

- New OS features may positively impact privacy...
- ...but negatively impact digital forensics capabilities
- e.g.:
 - Native whole disk encryption
 - Encrypted swap
 - Secure erasure of Recycle Bin / Trash
 - Secure erasure during disk formatting
- Or the other way around
- Today's topic: **Compressed RAM**
- Claim: with some work, **positively impacts forensics**
- First some background

Where's the Evidence?



Memory Analysis on RAM Dump

- Processes (dead/alive) ← e.g., discover unauthorized programs
- Open files ← e.g., detect keystroke loggers, hidden processes
- Network connections ← e.g., find backdoors, connections to contraband sites
- Volatile registry contents
- Volatile application data ← e.g., plaintext for encrypted material, chat messages, email fragments
- Other OS structures ← e.g., analysis of memory allocators to retrieve interesting structures, filesystem cache, etc.
- Encryption keys ← e.g., keys for whole disk encryption schemes

What's missing?

Memory Analysis: Swap Files

“Traditional” forensics: capture contents of storage devices, including the swap file

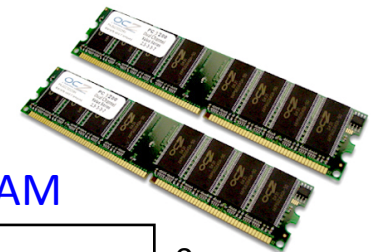
swap file
on disk



Swap file contains RAM “overflow”—if you don’t have this data, you may not have a complete memory dump

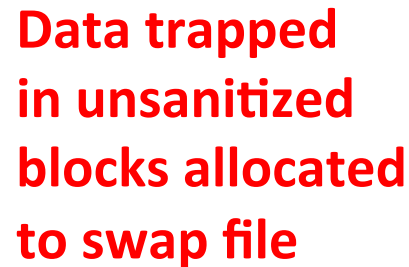
Live forensics / memory analysis: capture RAM contents

RAM



If you acquire both RAM and disk contents, memory *smearing* is likely and the swap file may be (very) out of sync with the memory dump

00000000000000000000000000000000	0
11110010010011010010010101010101	4096
01000001010010010010001000100010	...
11111111111111111111111111111111	...
.	
01010101010101010101010101010101	2GB



murder

```
void swap_crypt_ctx_initialize ( void ) {
    unsigned int i;
    if ( swap_crypt_ctx_initialized == FALSE ) {
        for ( i = 0; i < ( sizeof ( swap_crypt_key ) /
            sizeof ( swap_crypt_key [0])) ; i++) {
            swap_crypt_key [i] = random();
        }
        aes_encrypt_key (
            (const unsigned char *) swap_crypt_key ,
            SWAP_CRYPT_AES_KEY_SIZE ,
            &swap_crypt_ctx.encrypt);
        ...
        swap_crypt_ctx_initialized = TRUE ;
    }
}
```



More Background: Memory Management

- Need to allocate RAM to:
 - Operating system
 - If there is one
 - Individual applications →
 - All applications
 - If more than one can run at a time
 - Possibly fairly, or some other policy
- OS's need to manage memory efficiently to feed RAM-hungry apps
- Ever-increasing “need” for more memory

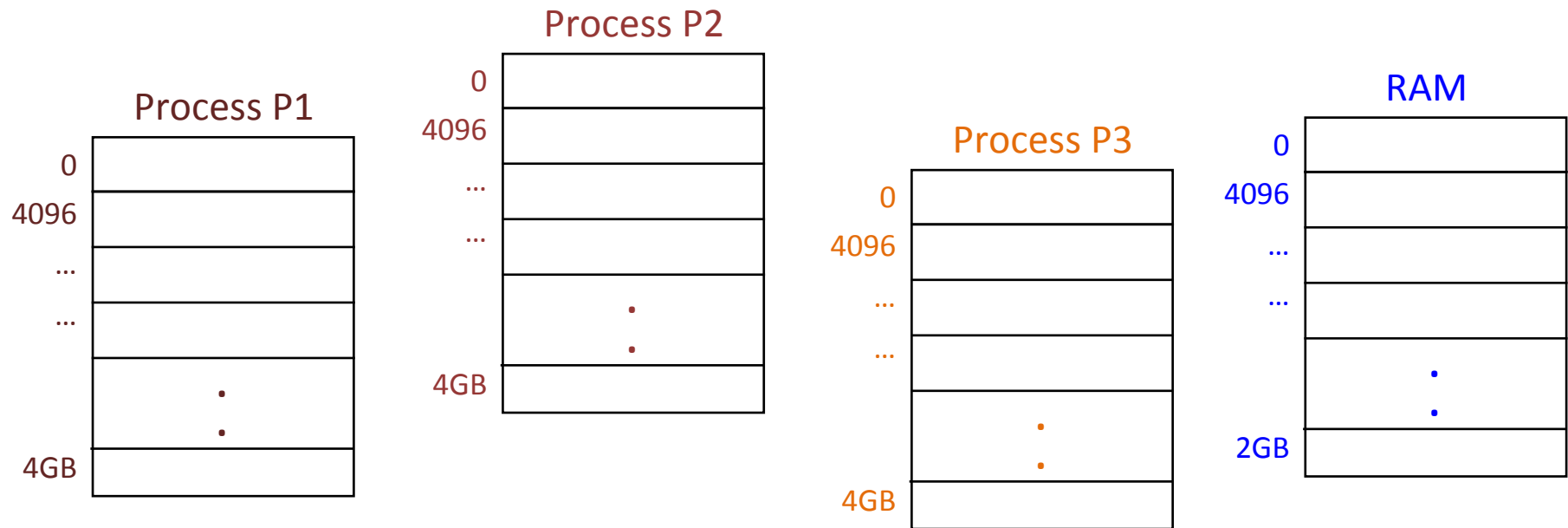


Virtual Memory

- Allows individual processes to have a contiguous, linear address space
- Virtual address space is divided into fixed-size blocks called pages
- Physical memory is divided into blocks of same size
- On x86 Intel, these are 4K
- OS keeps track of allocated and free pages of RAM
- Allocate only as many pages to a process as needed

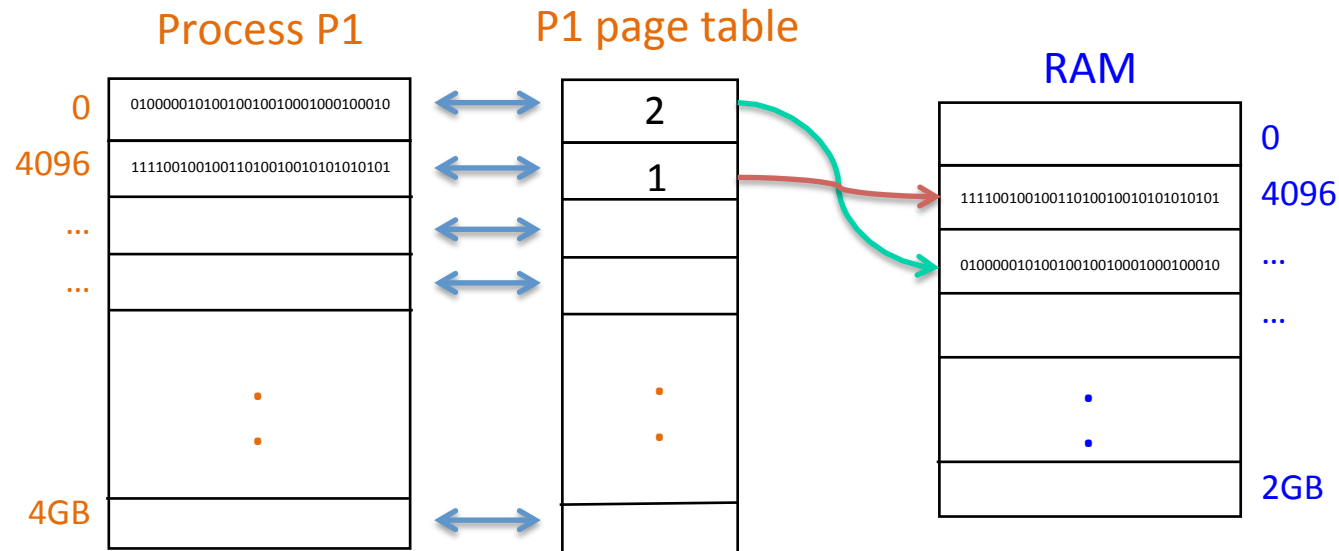
Logical vs. Physical Addresses

- Virtual address: Address whose scope is a particular process or OS kernel
- Physical address: Location in physical RAM



Paging: (Very) Simplified View

- Page tables (per process) are used to translate logical to physical addresses
- Pages for a particular process are generally not in contiguous order in RAM



64-bit Mode: 4-level Page Tables: 4K Pages

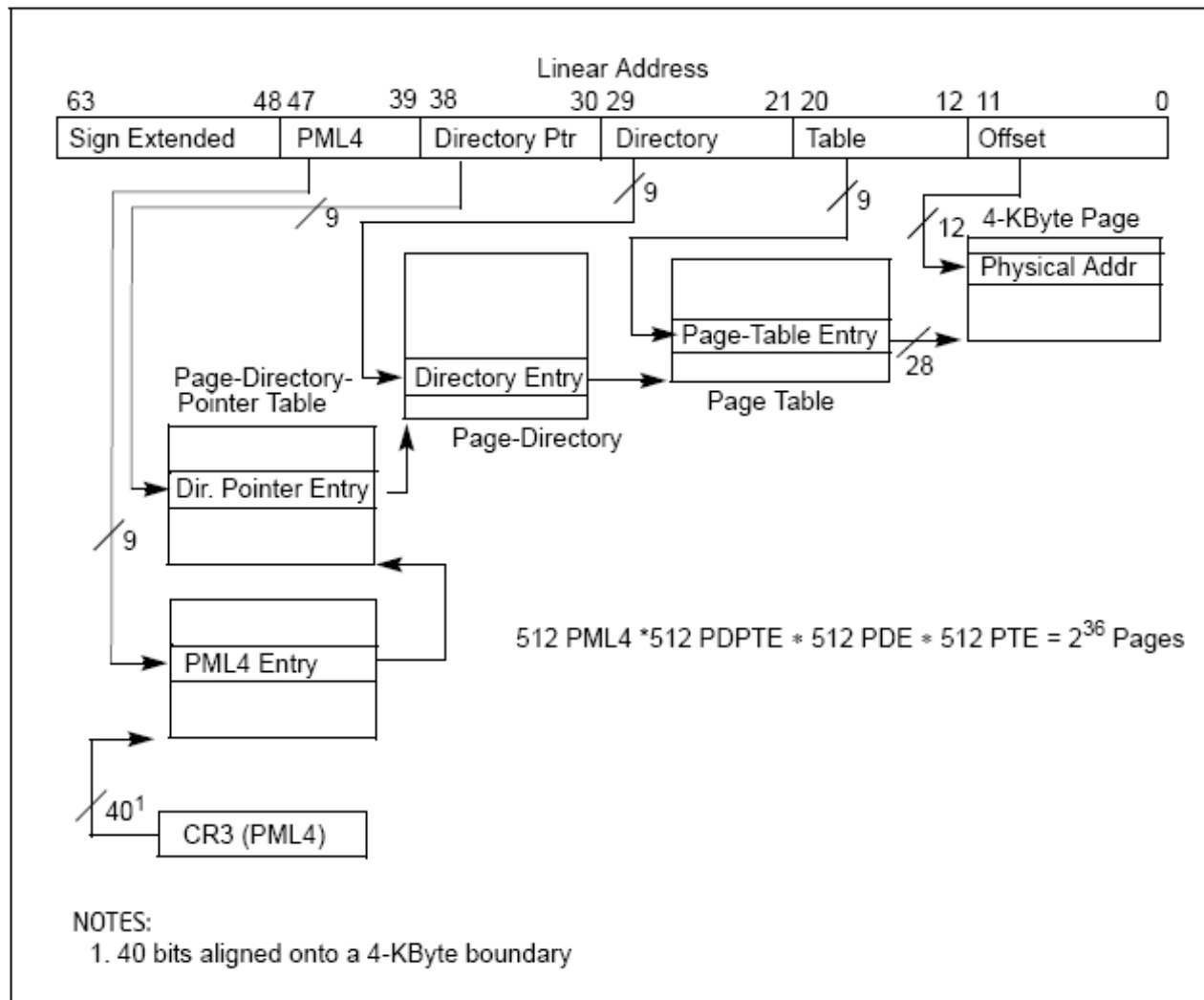


Figure 3-24. IA-32e Mode Paging Structures (4-KByte Pages)

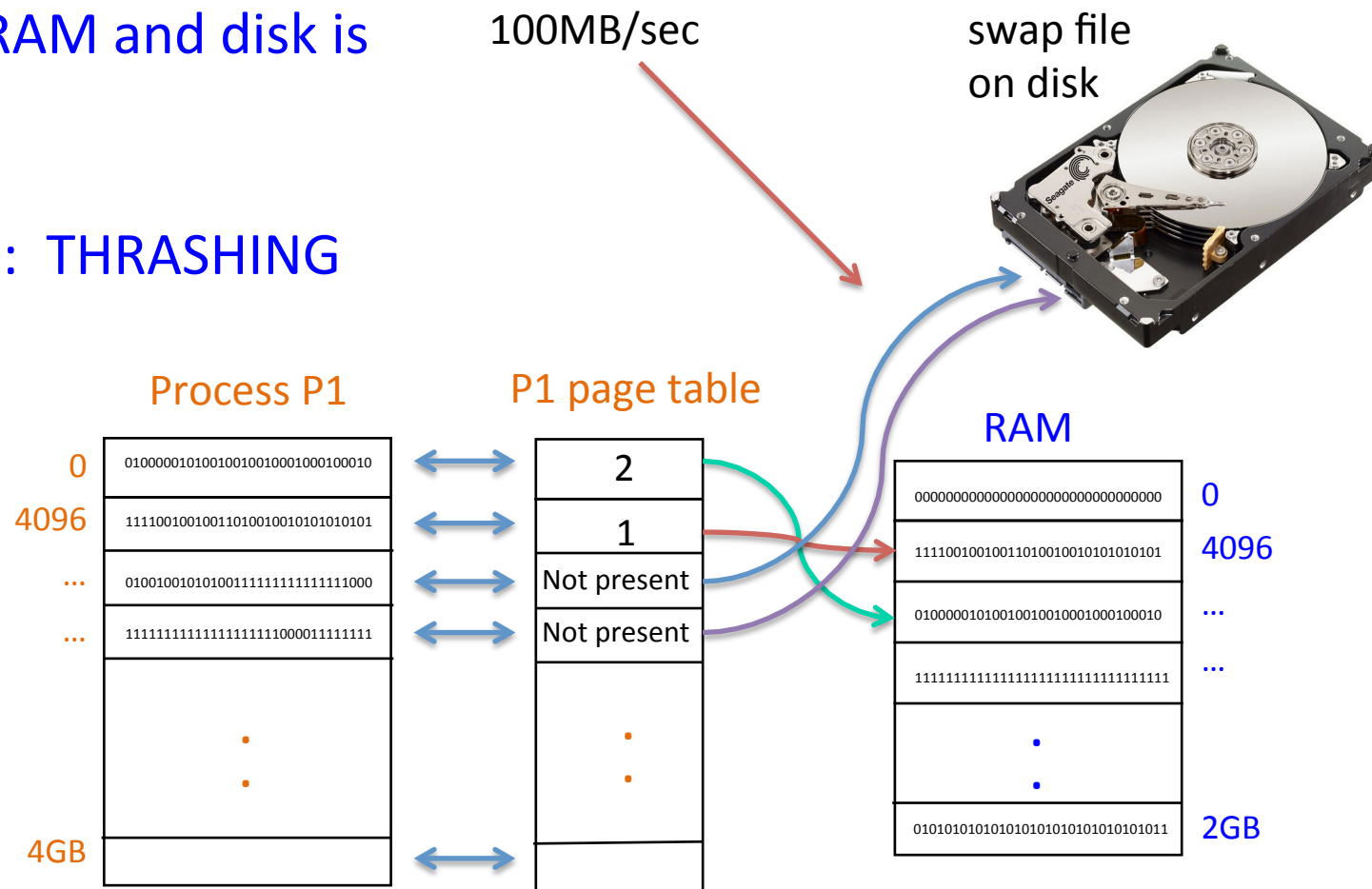
Stretching RAM

- Want to allocate more memory to processes than total amount of physical RAM
- Extend physical RAM by swapping pages to swap file on disk and retrieving as necessary
- Allows more (and bigger) processes to execute
- Workable, because:
 - Not all executing processes are active at once
 - Processes typically have “working sets” of pages, e.g., memory they’re actively using “now”

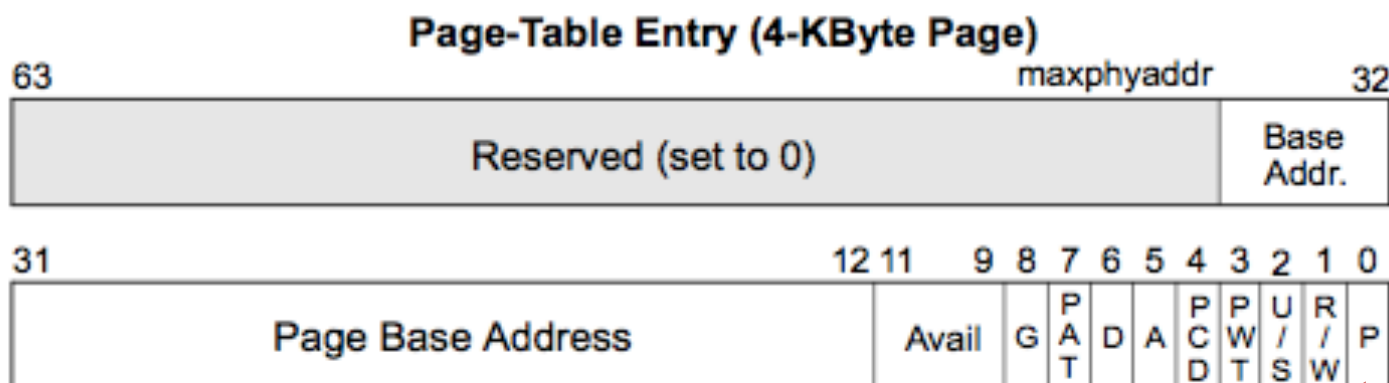
Virtual Memory with Swapping

For good performance,
can't overdo it! The pipe
between RAM and disk is
small

Otherwise: THRASHING



Swapping Explicitly Supported by HW



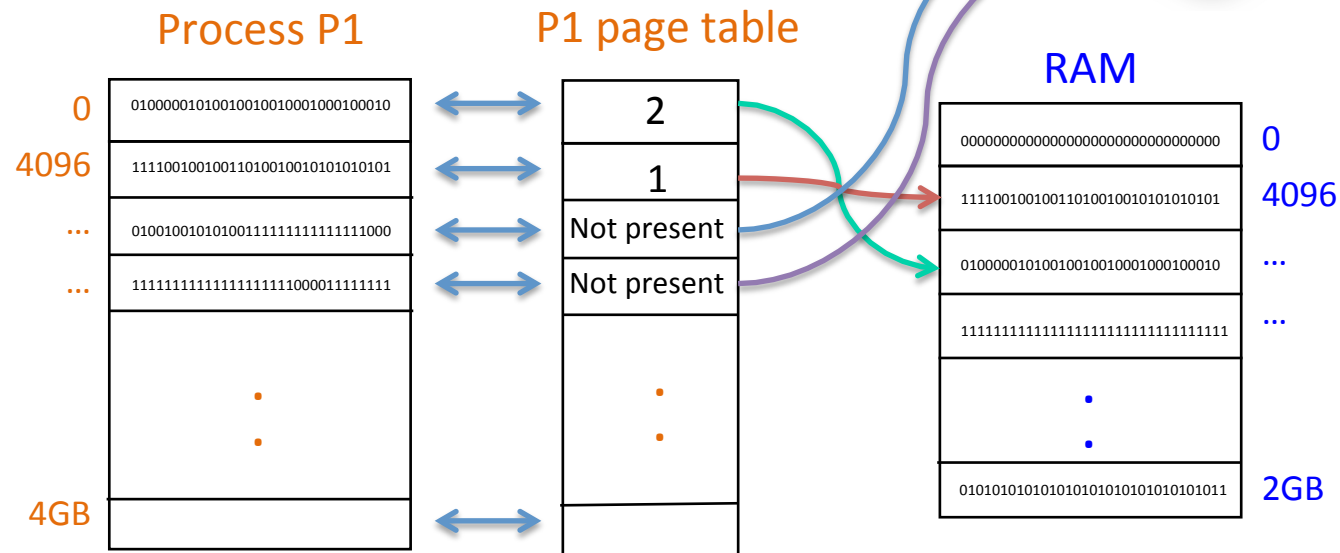
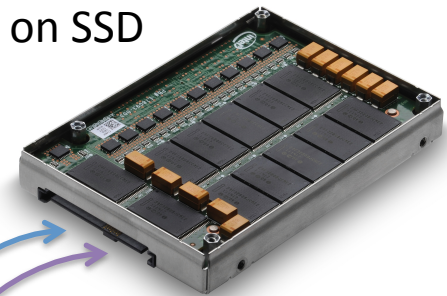
**Present bit in
page table entry (PTE)**

Virtual Memory with Swapping

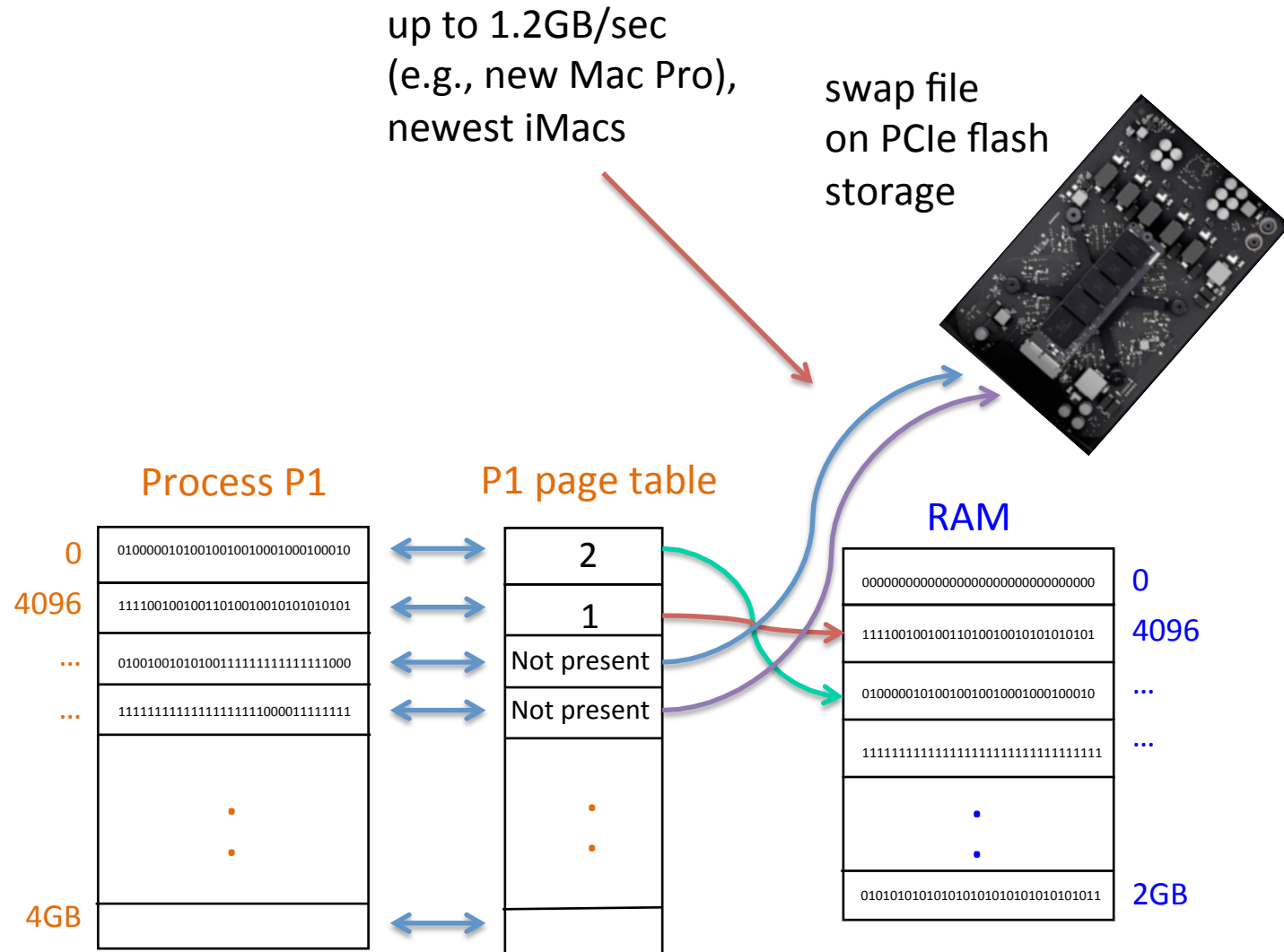
SSDs substantially increase
size of pipe

500GB/sec or so

swap file
on SSD



Virtual Memory with Swapping



RAM Compression, Not Swapping

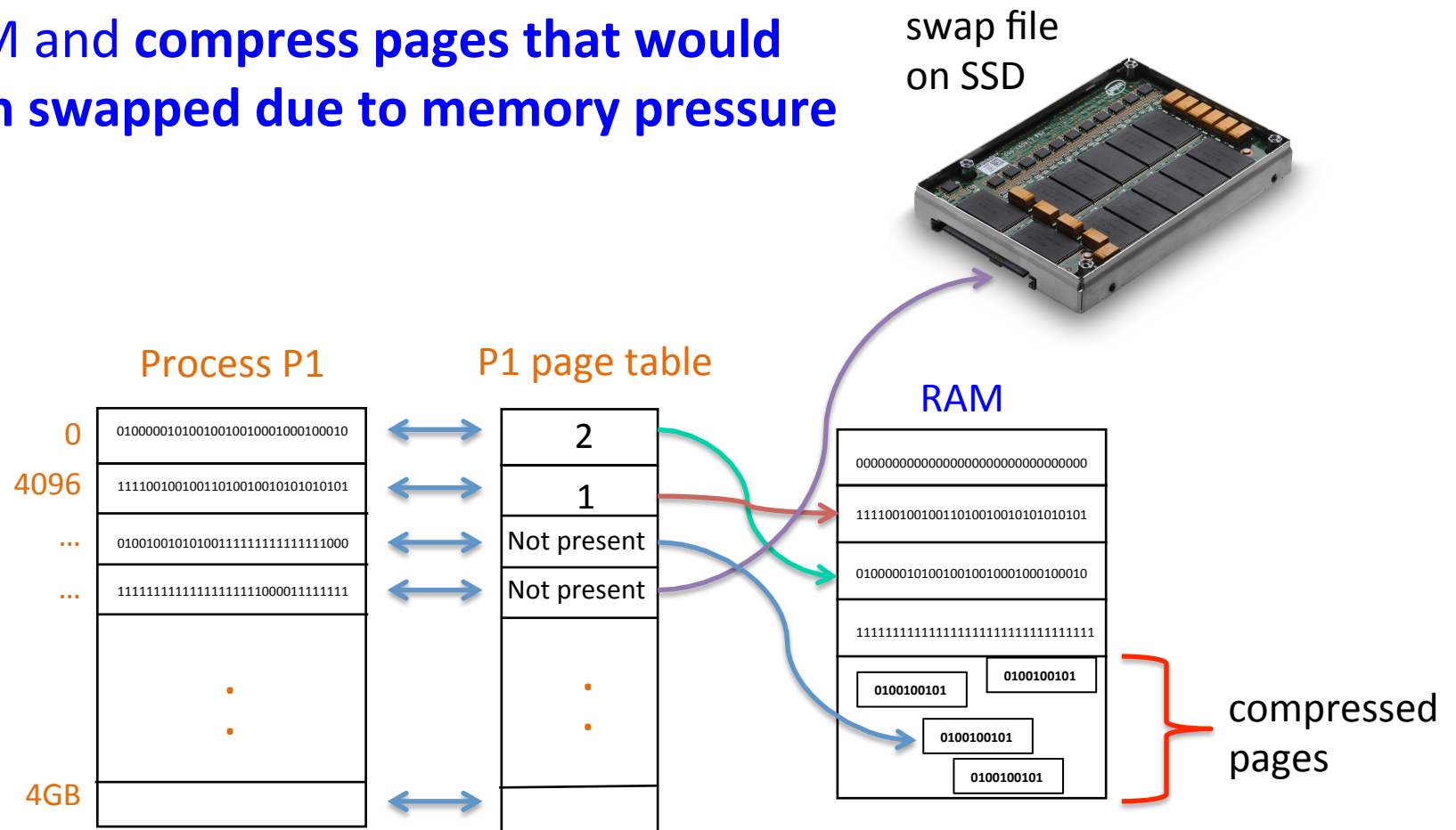
- Optimization:
 - “Hoard” some RAM, not available to processes
 - When physical memory is running low, compress 4K pages in memory instead of swapping to disk
- Why bother?
- SSDs are fast
- PCIe solid state storage is crazy fast (1.2GB/sec+)
- Memory is cheap[er]
- So...why?

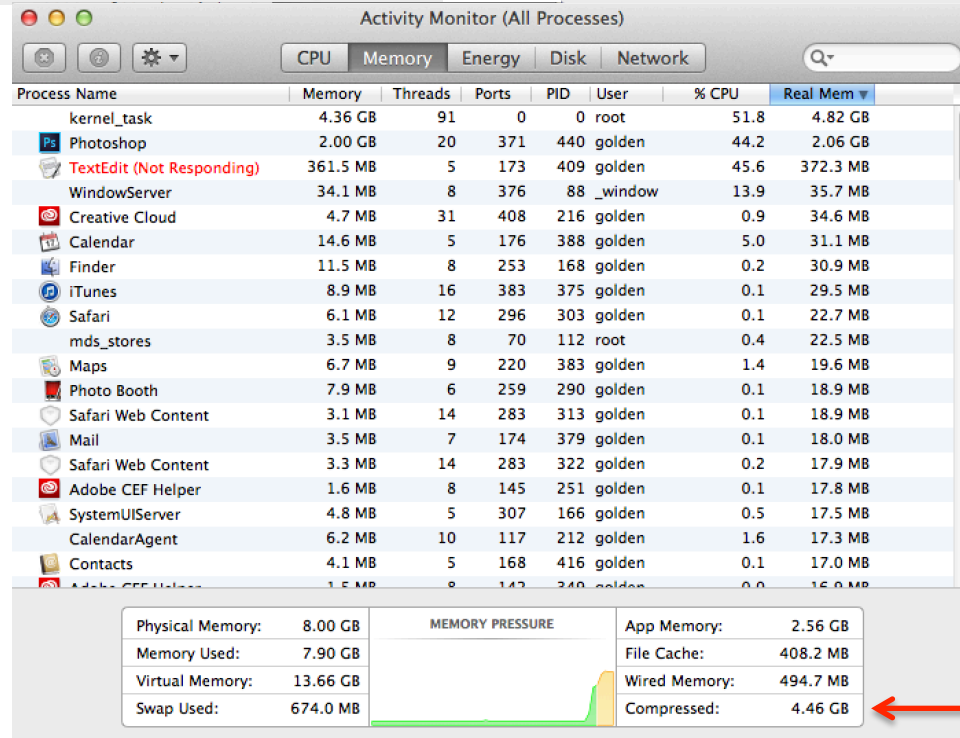
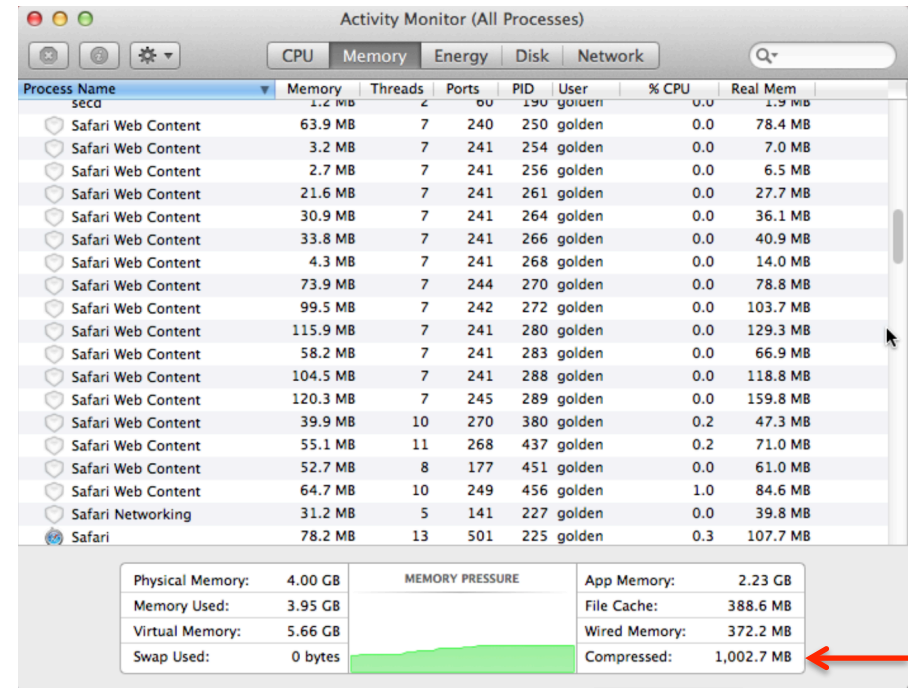
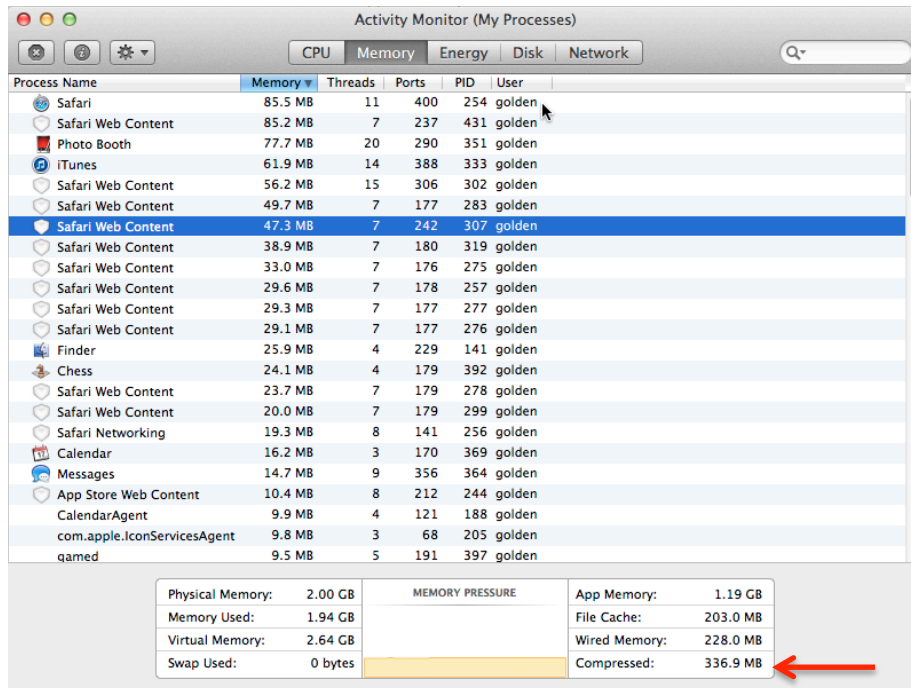
It's Worth It

- Swapping can cause serious performance issues
- PCIe storage is fast...memory is much faster
- New Mac Pro memory bandwidth: **60GB/sec**
- Max swap bandwidth: **1.2GB/sec**
- Lots of cores—compress and decompress very efficiently
- Ultraportable laptops still RAM limited
- e.g., current Macbook Air: Base 4GB, Max 8GB RAM
- Virtualization, even on laptops
- Malware analysts, VMWare Workstation, Fusion, etc.
- Photoshop, large documents, etc.

Virtual Memory with Compressed RAM

In Mac OS X Mavericks and Linux, set aside some RAM and **compress pages that would have been swapped due to memory pressure**





It's Back

- Compressed RAM idea isn't new
- RAM Doubler -- ~20 years ago
- Historically, not very effective
- Processors too slow to offset compress/decompress costs
- Plus poor integration with OS
- Old academic paper (2003) on compression schemes for RAM:
 - M. Simpson, R. Barua, S. Biswas, "Analysis of Compression Algorithms for Program Data"
 - <http://terpconnect.umd.edu/~barua/matt-compress-tr.pdf>
- Compression scheme in Mavericks (WKdm) was invented in 1999
- Now, super fast CPUs, lots of cores
- Tight integration

Compressed RAM: Forensic Impact

Currently, “simultaneous” capture results in lots of smearing

With compressed RAM, may be little or no data swapped out—capturing RAM captures (some) “swap”!

No support in current tools—compressed data opaque or ignored

Our tools enable analysis of compressed RAM

swap file on disk



PHYSICAL RAM

00000000000000000000000000000000	0
11110010010011010010010101010101	4096
01000001010010010010001000100010	...
11111111111111111111111111111111	...
.	
01010101010101010101010101010101	2GB

Mavericks VM: Too Much Detail

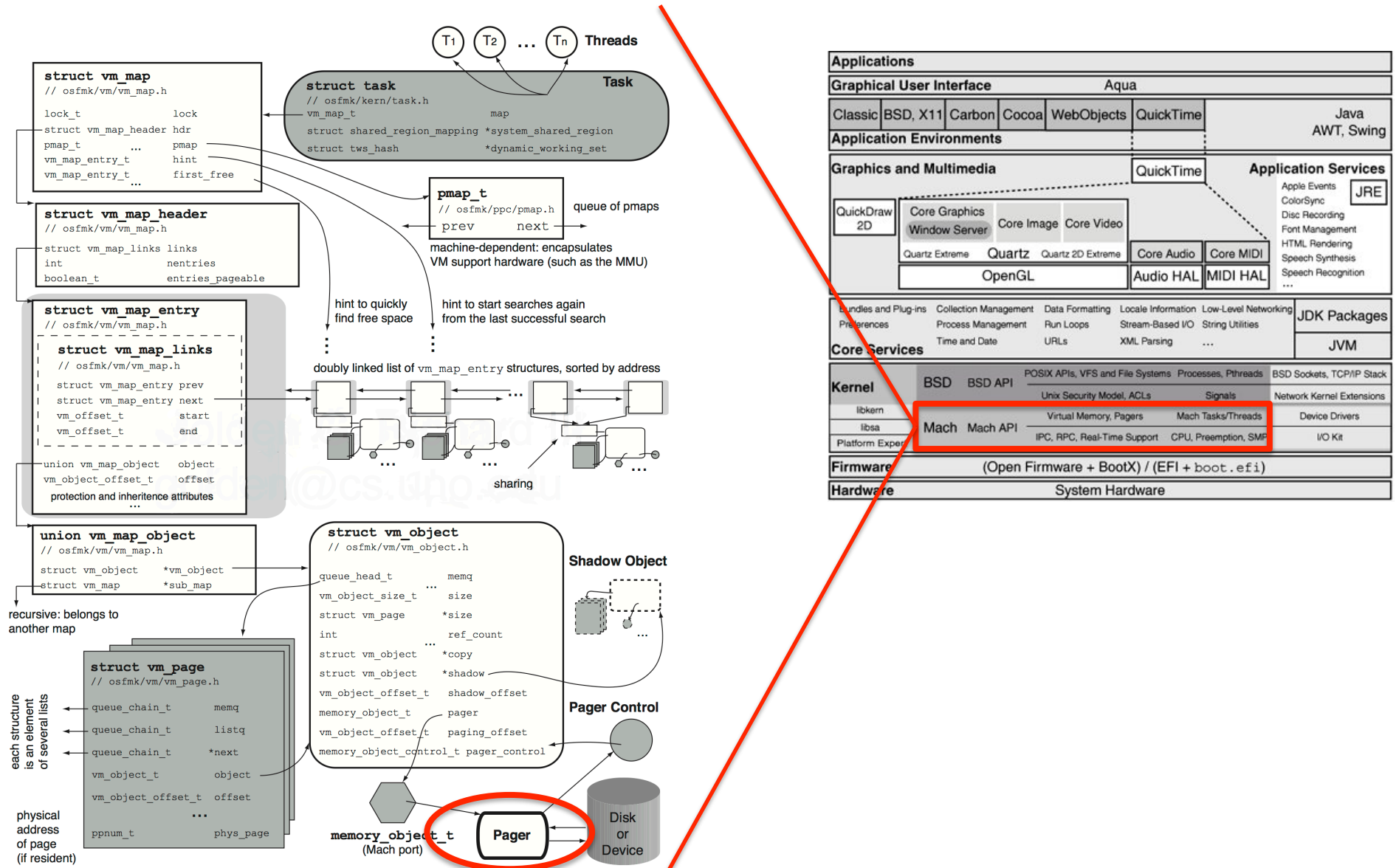


FIGURE 8-6 Details of the Mac OS X Mach VM architecture

Mac OS X Mavericks Implementation

- Implemented as a pager
- Fits nicely into the standard Mach virtual memory architecture
- Compressor pager hoards a modest amount of memory
- Page in / Page out requests all go through the compressor pager
- On page fault, pages actually swapped to disk are swapped in, decompressed, and returned
- Otherwise, just decompressed and returned

Mavericks Implementation (2)

- Compressor pager monitors “fullness”
- Schedules pages to be swapped out to disk as necessary
- Implementation in C (like most of Mach and BSD) except for optimized version of WKdm
- Pages are compressed and decompressed using WKdm:
 - Paul R. **Wilson**, Scott F. **Kaplan**, and Yannis Smaragdakis, The Case for Compressed Caching in Virtual Memory Systems, *Proceedings of the USENIX Annual Technical Conference*, 1999.
- Mavericks: Hand-optimized 64-bit assembler version of Kaplan’s original C code

Mach VM: C + asm

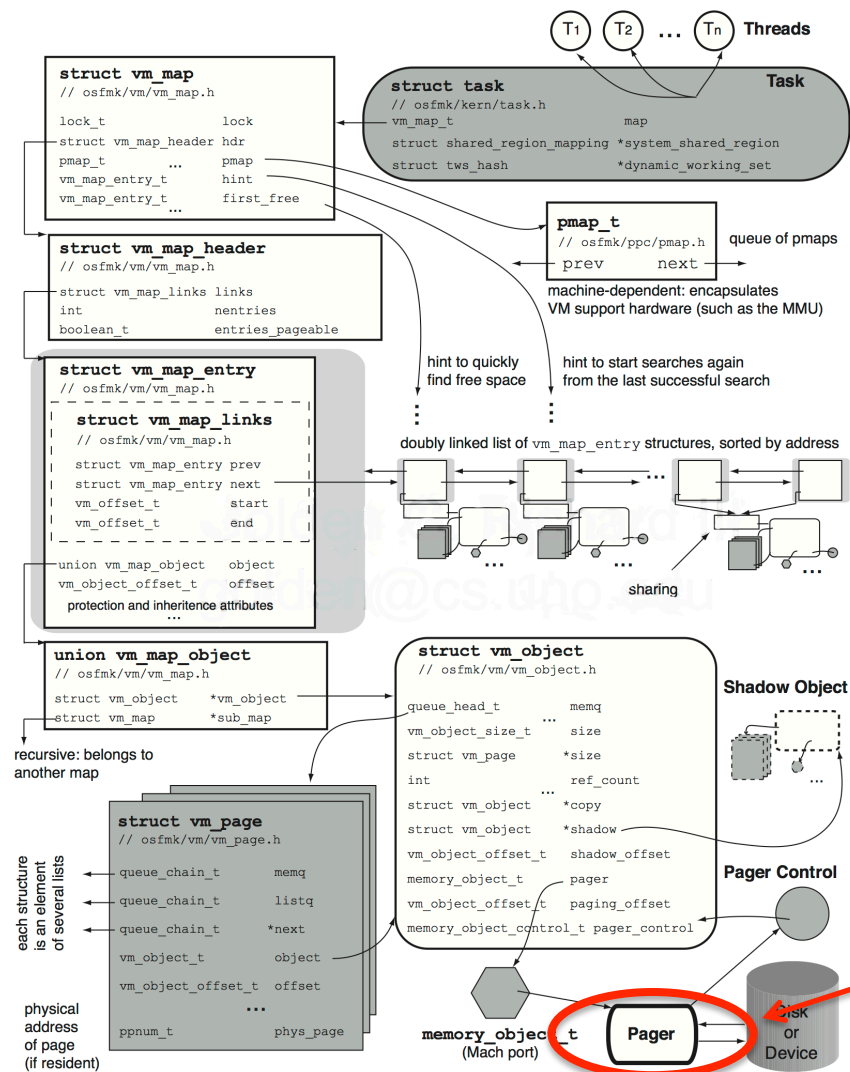


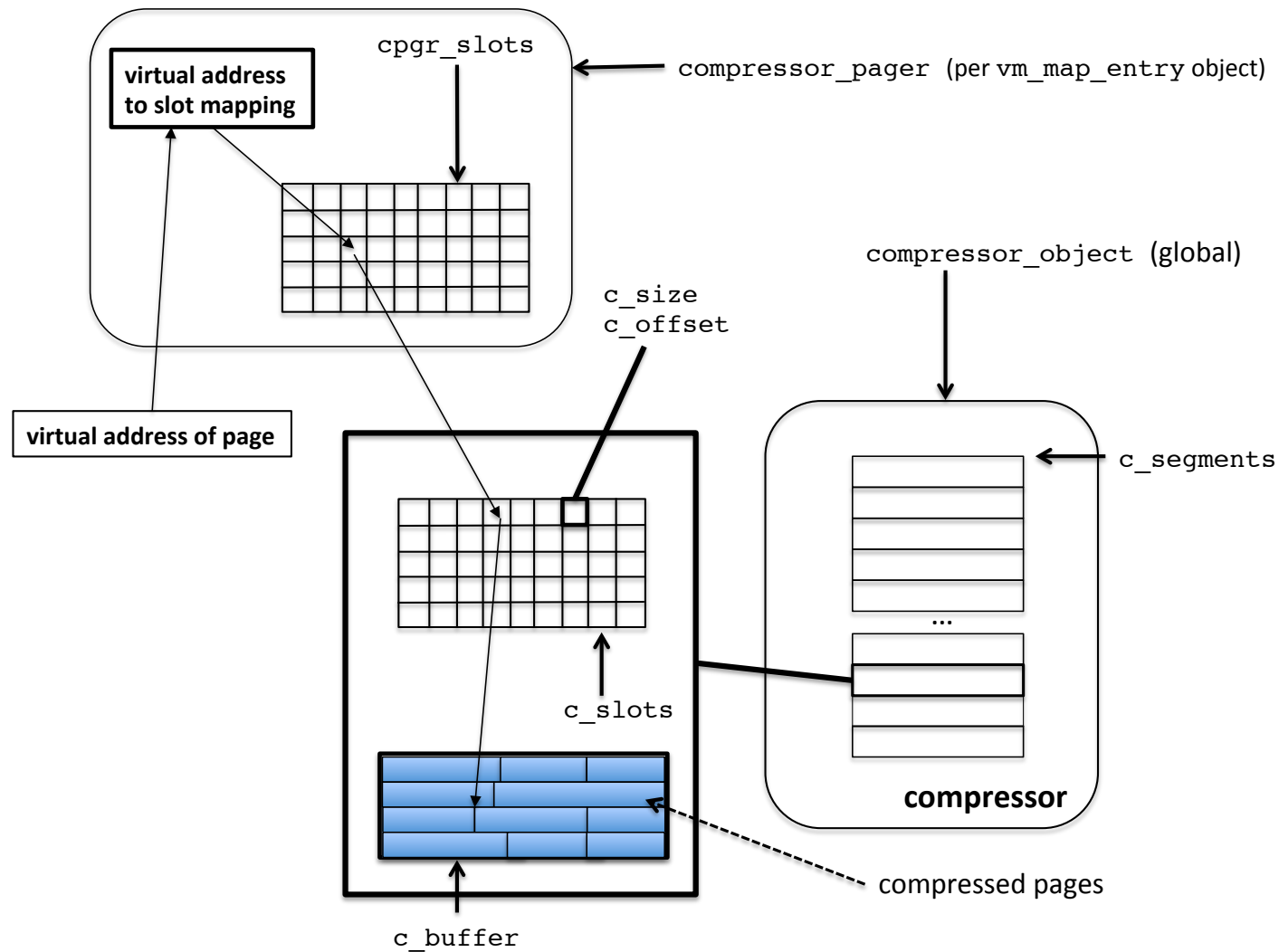
FIGURE 8-6 Details of the Mac OS X Mach VM architecture

xnu-2422.1.72/osfmk/vm:
~73K lines of C

xnu-2422.1.72/osfmk/x86_64/WKdm*.s:
~1000 lines of 64-bit assembler for
WKdm_compress_new() /
WKdm_decompress_new()

Just vm_compressor /
vm_compressor_pager:
~4K lines of C + the assembler, above

Compressor / Compressor Pager Internals



Volatility

- Most popular memory analysis framework
- Open source
- Portable, written in Python 😊
- Supports analysis of Windows, Linux, Mac
- Plugins add new functionality
- Our contribution: New plugins for compressed RAM analysis
 - Mac implementation: Mostly me (@nolaforensix)
 - Linux implementation: Mostly Andrew Case (@attrc)

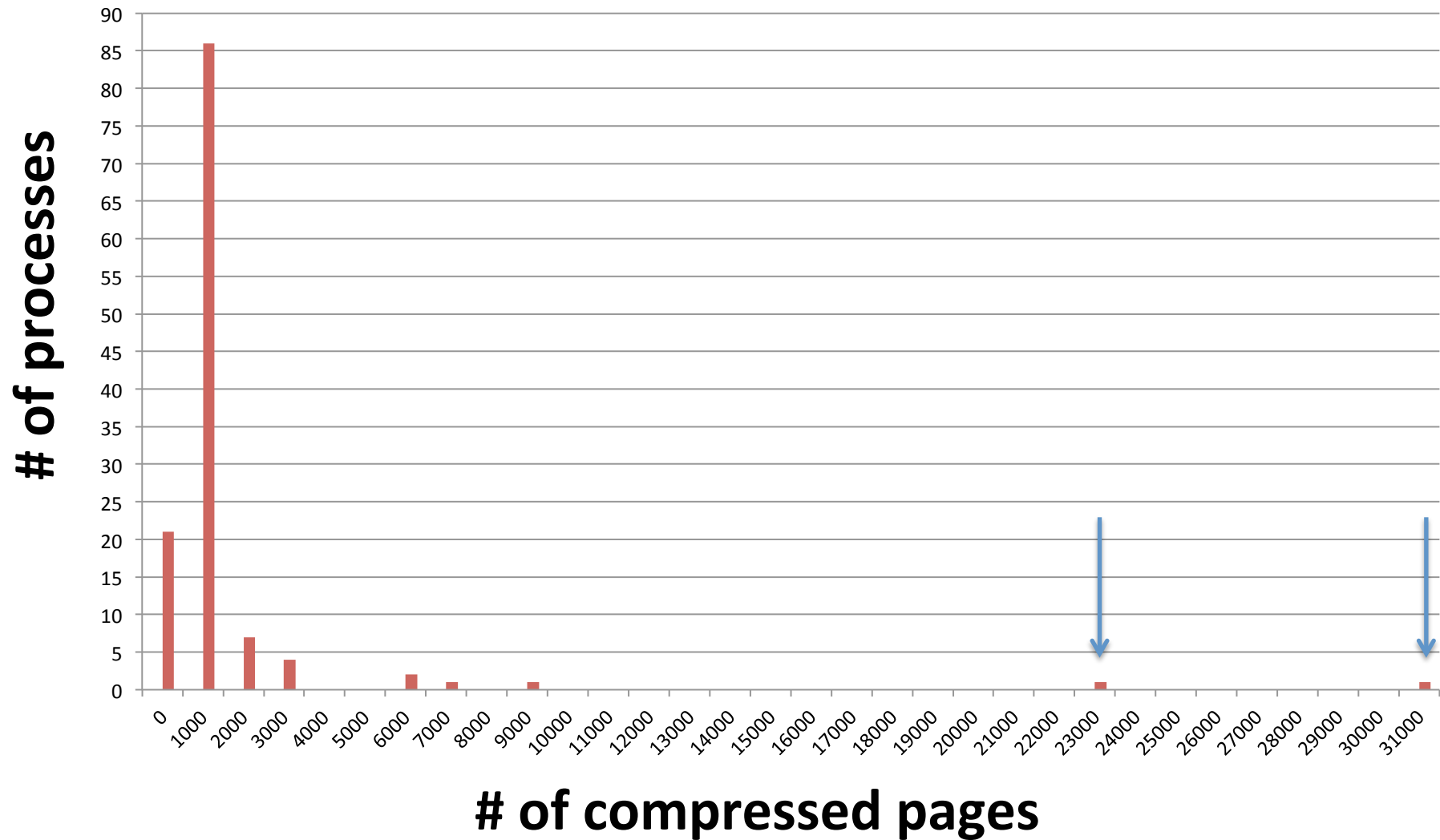
New Plugins for Volatility

- **mac_compressed_swap (/ linux_compressed_swap)**
 - Decompress and dump all compressed pages
 - Emits compressor stats:
 - Compressor memory used : 19167408 bytes
 - Available uncompressed memory : 466462 pages
 - Available memory : 471251 pages
 - ...
- “Easiest” of the new plugins to implement
 - Required Python implementation of WKdm decompression (for Mac)
 - Required detailed analysis of compressor pager and emulation of portions in Python
 - But not entire page fault → decompression path

New / Modified Plugins for Volatility

- **mac_dump_maps (/ linux_dump_maps)**
 - **Dump address space for all (or individual) processes**
 - Previous implementation used Volatility's standard mechanism for emitting 4K pages in address space
 - Simply skipped pages marked "not present" (swapped)
 - Now, pages that are marked not present compared against compressor pager holdings
 - Decompressed if possible and made available
- Involves deeper analysis of Mach / Linux VM systems
- Much more complex (not your problem)
- We ❤️ OS internals

Representative Distribution of Compressed Pages



Mac OS X 10.9, 2GB RAM, 124 processes, moderate memory pressure, 300MB compressed

Compressed RAM: Conclusions

- Significantly reduces swapping to disk
- Used very aggressively in Mac OS X Mavericks
- Soothes RAM $\leftrightarrow$ swap file consistency problems for memory analysis
- Makes more evidence available
- Essential for more “complete picture” in memory analysis
- Our analysis tools integrated into Volatility
- Side effect of research: (more) affirmation that Python is too slow for most security tool development
- That’s for another talk or over a beer

Questions?

Benchmarks (Sorry, Python)

