Effects of Memory Randomization, Sanitization and Page Cache on Memory Deduplication

Kuniyasu Suzaki, Kengo Iijima,
Toshiki Yagi, Cyrille Artho

Research Institute
for Secure Systems

National Institute of
Advanced Industrial Science
and Technology
AIST

EuroSec 2012 at Bern, April 10

Background 1/2

- Infrastructure as a Service(IaaS) type cloud computing offers OS hosting service and runs many virtual machines.
  - Examples: Amazon EC2, Rackspace, ...
  - Physical resources are very important. Many techniques are developed to save resources.
- We concentrate on memory.
  - ***Memory Deduplication*** is utilized to share same contents and reduces consumption of physical memory.

- Venders want to use this function, but …

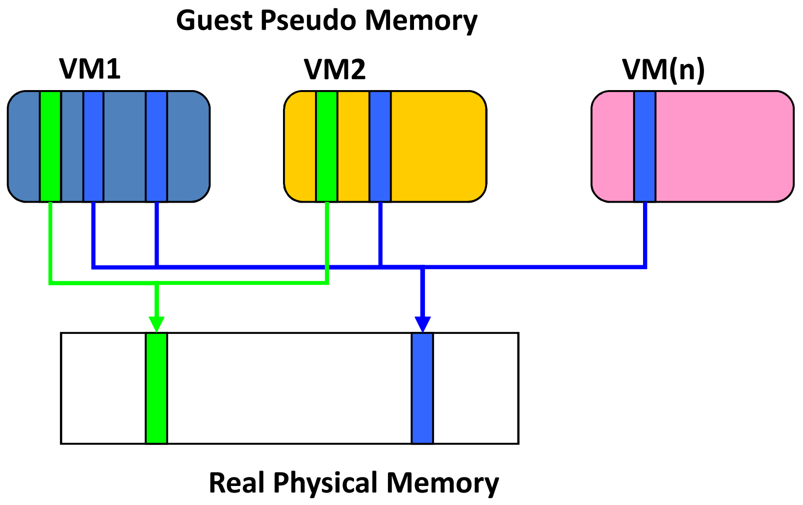
Background 2/2

- Some attacks are appeared on IaaS, and users want to increase security of Guest OS.
  - For example, Cross VM Side channel attack [CSS2009]
- Guest OSes have some choices of security function.
  - Some of them change the behavior of memory and will work to increase or decrease performance of memory deduplication.
- Our paper measures the affects of security functions on real memory deduplication (KSM with KVM).

Contents

Memory Deduplication 1/2

- Memory deduplication is a technique to share same contents page.
  - Mainly used for virtual machines.
  - Very effective when same guest OS runs on many virtual machines.
- Many virtual machine monitors include deduplication with different implementations.



Memory Deduplication 2/2

- Content-aware deduplication
  - Check contents when data is loaded from disk to memory.
    - Transparent Page Sharing on Disco [OSDI97]
    - Satori on Xen[USENIX09]
- Periodical memory-scan deduplication
  - Scan memory pages and merge when pages are same. It can treat dynamically created contents pages.
    - Content-Based Page Sharing on VMWare ESX [SOSP02]
    - Differential Engine on Xen[OSDI08]
    - KSM (Kernel Samepage Merging) [LinuxSymp09]
      - General-purpose memory deduplication for Linux.
      - Used mainly for KVM.
- Our paper uses KSM with KVM virtual machine.

KSM: Kernel Samepage Merging

- KSM has 3 states for pages.
  - Volatile : contents change frequently (not to be candidate)
  - Unshared: candidate pages for deduplication (unique at present)
  - Shared: deduplicated pages with same contents.
- Pages are scanned (default: 20msec)
  - All pages are not scanned at a trial.
  - The maximum is 25% of the available memory.
  - The time to be deduplicated depends on the situation.

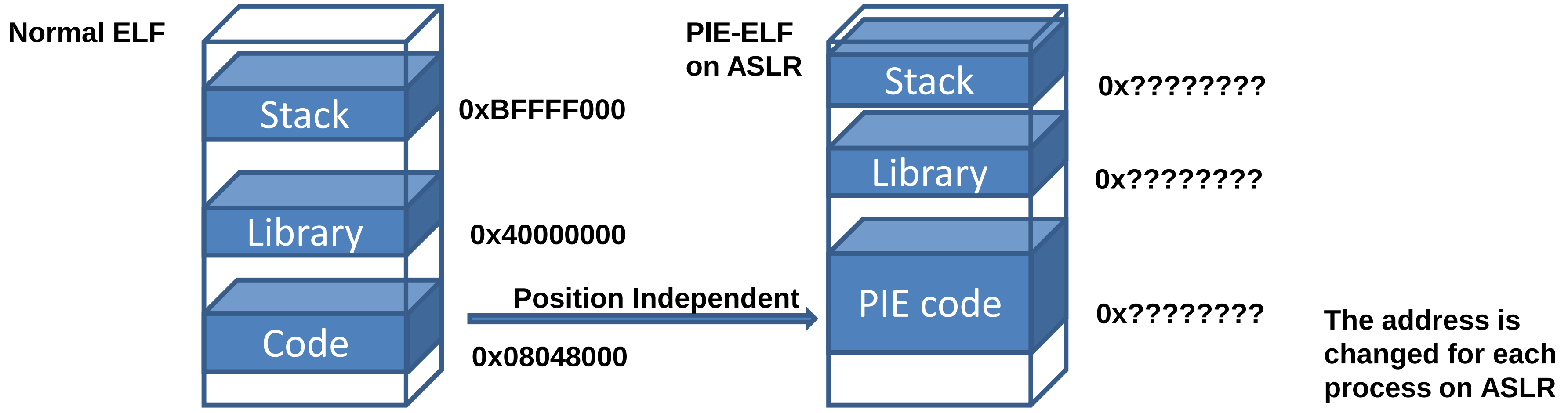
Contents

OS Security functions

- Modern OSes have security functions that modify memory contents dynamically.
  1. Address Space Layout Randomization (ASLR) with Position Independent Executable (PIE)
  2. Memory Sanitization
  3. Page Cache flashing

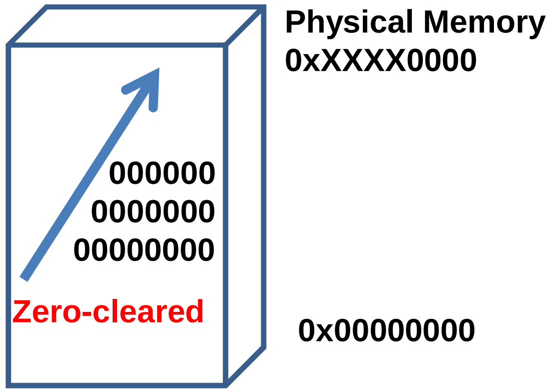
Address Space Layout Randomization(ASLR)

- Normal binary has fixed Memory layout, and has vulnerability for overflow attack.
- ASLR gives randomized offset (aligned page) to code, library, and stack.
    - Early ASLR for Linux is implemented on ***PaX*** and ***Exec Shield*** Linux.
    - Linux 2.6.12 (June 2005), Windows VISTA.
- Binaries have to be Position Independent Executable (PIE).
    - GCC has options to compile a code for PIE-ELF binary. ("-fPIE" for compiler and "–pie" for linker)
        - Disadvantage: PIE binaries become fat, because the all addressing are relocatable.
- ASLR and PIE binaries will affect memory deduplication.



Memory sanitization

- Technique for zero-clearing pages after use.
  - Sensitive data remain for long periods after use, and Chow proposed ***Secure deallocation*** [USENIX Security 04, 05] . It prevents information leak.
- Linux has ***unconditional page sanitization*** patch.
  - Increased zero-cleared pages will be deduplicated. As a result, it will reduces consumption of physical memory.
  - Disadvantage:
    - Linux's sanitization requires zero-clearing all physical pages at boot time.
    - Do not treat page cache, because cashed pages are not released from a kernel.



Page Cache flushing

- Page Cache flushing reduces possibility of information leak.
- Page Cache reduces I/O overhead when pages are accessed repeatedly. However, the page cache may include sensitive data.
- Linux kernel has ***DropCache*** for page cache flushing.
  - DropCache released all cached pages from the kernel.
    - "cron" is used to flush page cache at certain intervals (1 sec).
  - The flushed memory region is reused for other processes. As a result, it will reduce consumption of physical memory.
  - DropCache does not zero-clear the released pages. We need to enable sanitization to zero-clear the pages.
  - Disadvantage:
    - Re-loaded data from disk when the same contents are accessed again.

Contents

Experiments

- Measure the effects of 3 security functions **(ALSR, memory sanitization, DropCache: page cache flushing)** on memory deduplication (KSM with KVM).
- Test environment
  - Intel Core2Quad (Q9650) 3.0GHz processor
  - Host OS: Ubuntu 9.10 (kernel: vanilla-2.6.32.1) with memory deduplication KSM and virtual machine monitor KVM
  - 2 type of Guest OSes
    - Normal Gentoo Linux (1.12.13, kernel 2.6.31) on a 32GB virtual disk (31GB ext3, 1GB swap)
    - Gentoo Linux which is built as PIE binaries
      - It utilize ALSR

Effects of Position Independent Executable (PIE)

- Gentoo Linux has 1,469 ELF binary files in /bin, /sbin, /usr/bin, and /usr/sbin.
  - On Normal Gentoo the total is 88.4MB
  - On PIE Gentoo the total is 94.6MB (7% more).
    - The biggest change: “pampop9” 5,396B -> 9,440B (75% more).
    - The smallest change: “wall” from 9,624B to 9,392B (2% less).
- From after PIE Gentoo is mainly used.

With and without ASLR, DropCache and Sanitization on 4VMs

Security function			①Peak Mem (MB) Virtual/Physical	Stable State			⑤Guest OS Boot Time (sec)
ASLR	DropCache	Sanitize		② Physical Mem (MB)	③ Sharing (MB (%))	④ Unshared+ Volatile (MB (%))	
○			574/458	234.9	106.4(45.3)	128.5(54.7)	62
○	○		431/332	206.9	70.7(34.1)	136.3(65.9)	83
○		○	2063/1661	204.6	82.1(40.1)	122.5(59.9)	61
○	○	○	2063/1616	186.5	39.4(21.1)	147.1(78.9)	83
			574/455	199.0	120.1(60.4)	78.9(39.6)	62
	○		429/316	169.5	83.1(49.0)	86.5(51.0)	82
		○	2063/1661	171.2	94.0(54.9)	77.2(45.1)	62
	○	○	2063/1161	129.9	50.4(38.8)	79.5(61.2)	85

With and without ASLR, DropCache and Sanitization on 4VMs

Security function			①Peak Mem (MB) Virtual/Physical	Stable State			
ASLR	DropCache	Sanitize		② Physical Mem (MB)	③ Sharing (MB (%))	④ [illegible] Volatile (MB (%))	[illegible]
○			574/458	234.9	106.4(45.3)	128.5(54.7)	62
○	○		431/332	206.9	70.7(34.1)	136.3(65.9)	83
○		○	2063/1661	204.6	82.1(40.1)	122.5(59.9)	61
○	○	○	2063/1616	186.5	39.4(21.1)	147.1(78.9)	83
			574/455	199.0	120.1(60.4)	78.9(39.6)	62
	○		429/316	169.5	83.1(49.0)	86.5(51.0)	82
		○	2063/1661	171.2	94.0(54.9)	77.2(45.1)	62
	○	○	2063/1161	129.9	50.4(38.8)	79.5(61.2)	85

+18%

-12%

-13%

+43%

-21%

Decrease

Increase

-15%

-14%

-35%

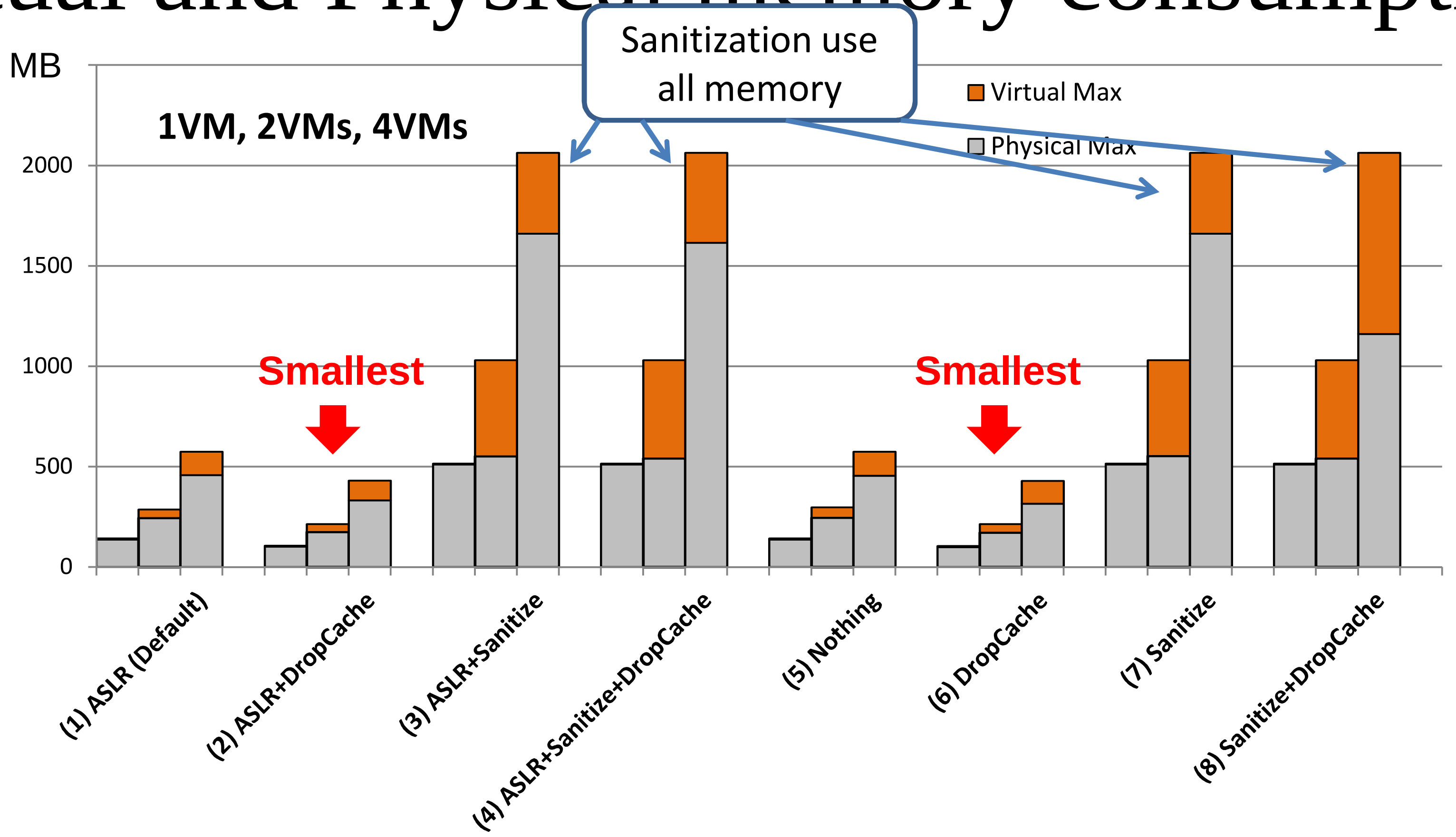
DropCache delayed the boot time.

Sanitization uses all virtual memory, but consumption of physical memory is reduced by deduplication.

ASLR increases physical memory consumption
Others (DropCache, Sanitization, and Both) decrease.

ASLR decreases deduplication and increases unique pages. It means ASLR reduces opportunities for memory deduplication.

Virtual and Physical memory consumption



- Sanitization uses all virtual memory. KSM depress the consumption of physical memory but it is still large.
- DropCache, (2) and (6), shows the smallest consumptions, because the flushed pages are reused.
  - They shows enabling DropCahe is the best from the view of memory, but DropCache affects the time performance of GuestOS.

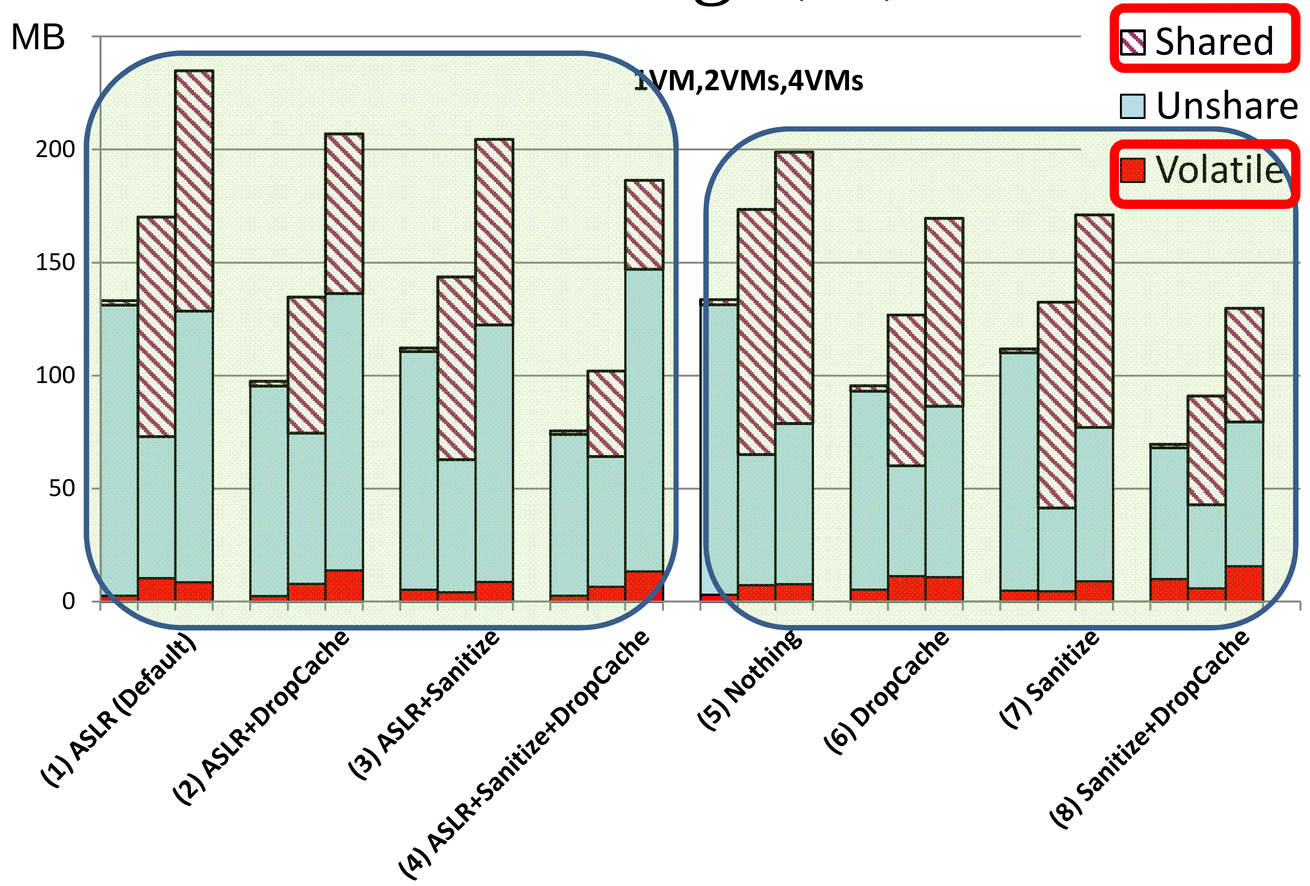
DropCache: Page Flushing

- DropCache was issued every second. It takes 10 - 20 microseconds (1% or 2%), thus it does not affect performance severely.
- However, the boot time with DropCache increases by about 20 seconds (30% more).
- The other boot procedures read 65MB of data from disk. DropCache increases this to 99MB (52% more).

	PIE Gentoo (seconds)
No DropCache	61–62
DropCache	82–85

	Disk Read (MB)
Normal	65
DropCache	99
Sanitize	65

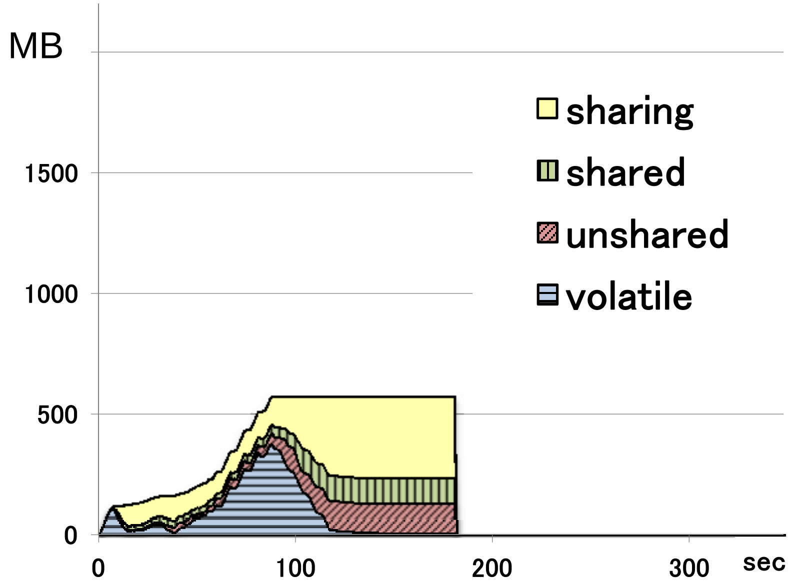
Memory Usage in KSM: PIE Gentoo at a stable state after booting 1, 2, and 4 VMs



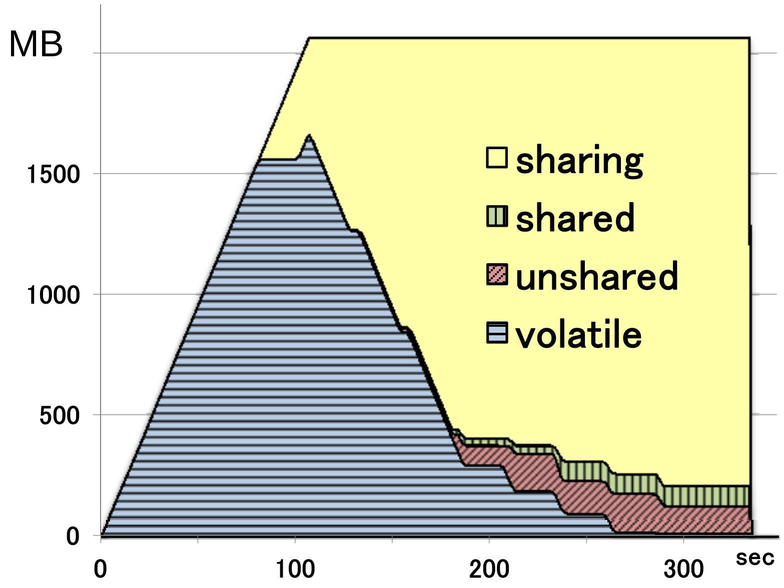
- Total memory(volatile+unshared+sharing) with ASLR are larger than without ASLR.
- Volatile and Sharing pages are almost the same. ASLR increases the number of unshared pages. It means that ASLR increases pages with different contents.
- However, it is strange because ASLR dons not change the contests. The reason is not well analyzed.

Trace of Memory usage on KSM with 4VMs

- Sanitization sets memory contents to zero on all memory pages at boot time. KSM could not catch up.
- However, the performance of Guest OS is not effected, because KSM limits the number of pages at a trial, and postpone deduplication.
  - After booting, KSM deduplicates memory pages.



Without Sanitization(PIE Gentoo on ASLR)



With Sanitization

Contents

Discussions

- Co-Design of GuestOS and VMM
  - Some VMMs offers zero-cleared pseudo memory to a VM. If memory sanitization recognizes it, the zero-clearing process may be removed.
  - If VMM control page flushing (balance page cache and I/O), it can achieve global optimization among many VMs.
    - It will be a counter part of ***ballon memory*** of VMM.
- Consider Scalability on IaaS
  - DropCache reduces the consumption of memory, but it also reduce time performance. If a server (IaaS) hosts OSes which don’t care time performance, DropCache is effective.

Related Works

- Memory deduplication has potential to change structure of OS.
  - Memory deduplication is used to increase security
    - SLINKY[USENIX ATC 2005] encourages static-linking, because memory deduplication reduces the increase memory.
    - Instead of static-linking, Self-contained ELF binaries (binaries which include shared libraries) is used for generalization [HotSec 2010]
  - Attack on memory deduplication
    - Memory disclosure attack for Copy-On-Write [EuroSec 2011]
    - OS Fingerprint [IPCCC2011]
- Memory deduplication is improved.
    - KSM++ [RESoLVE 2012]
    - HICAMP [APSOLS 2012] Hardware memory deduplication. The size is very small (16-256 Byte).

Conclusions

- Our paper measures the affects of security functions (ASLR, sanitization, page cache flushing) on real memory deduplication (KSM with KVM).
  - ASLR increases consumption of physical memory, which is more than anticipated.
  - Memory sanitization has to initialize all memory, but the CPU overhead for GuestOS is low, because memory merging is postponed at heavy load.
  - Page cache flushing (DropCache) reduces consumption of physical memory by reusing released pages. However, the overhead is large, which is caused by re-loading data from a disk.
- We should consider co-design of VMM and security function of guestOS and increase the scalability of IaaS.