Moving *from* Logical Sharing of Guest OS *to* Physical Sharing of Deduplication on Virtual Machine

Kuniyasu Suzaki, Toshiki Yagi, Kengo Iijima, Nguyen Anh Quynh, Cyrille Artho

Research Center of Information Security

National Institute of Advanced Industrial Science and Technology

Research Center for Information Security

&

National Institute of Advanced Industrial Science and Technology

AIST

Yoshihito Watanebe

Alpha Systems Inc.

Contents

- Vulnerability of logical sharing (Dynamic-Link Shared Library and Symbolic Link)
- Propose replacement of logical sharing by physical sharing
  - Physical sharing
    - Deduplication on Memory and Storage
  - Self-contained binary
    - It is NOT static-Link binary.
- Experimental results
- Conclusions with discussing topics

Logical Sharing

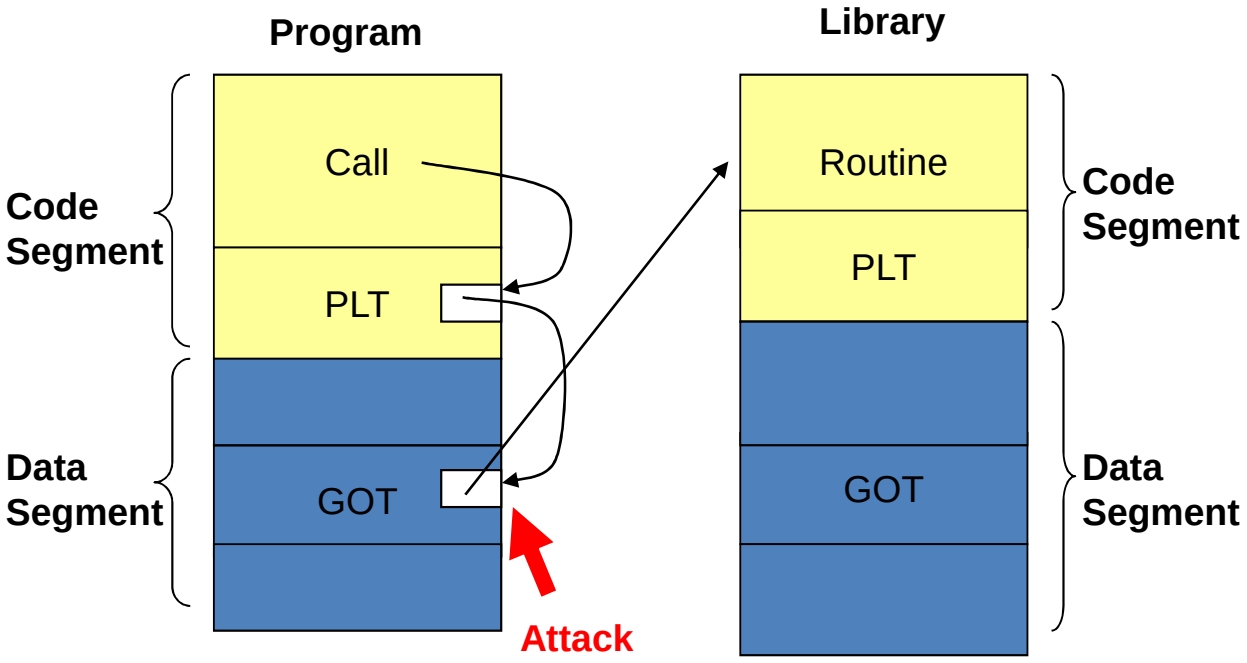
- Logical sharing is OS technique to reduce consumption of memory and storage.
  - “Dynamic-Link Shared Library” for memory and storage
  - “Symbolic Link” for storage
- Unfortunately, they include vulnerability caused by dynamic management.
  - Search Path Replacement Attack
  - GOT (Global Offset Table) overwrite attack
  - Dependency Hell
  - Etc.

Search Path Replacement Attack

- Dynamic-link searches a shared library at run time using a search path.
  - Search path is defined by environment variables.
    - Example: “LD_LIBRARY_PATH”
  - It allows us to change shared libraries in any directories.
- Unfortunately, the search path is easily replaced by an attacker and leads to malicious shared libraries.
  - Caller program has no methods to certify libraries.
- Static-link solves this problem but it wastes memory and storage.

GOT Overwrite Attack

- ELF format has GOT (Global Offset Table) to locate position-independent function address of shared library. The value of GOT is assigned at run time.
  - GOT is created on Data Segment and vulnerable for overwrite attack.
- Static link solves this problem but it wastes memory and storage.



Dependency Hell (DLL Hell in Windows)

- Dependency Hell is a management problem of shared libraries.
  - Package manager maintains versions of libraries. However, the version mismatch may occur, when a user updates a library without package manager.
  - Caller program has no methods to certify libraries.
- Dependency Hell is escalated by symbolic-link, because most shared libraries use symbolic-link to manage minor updates.
  - /lib/libc.so.6 -> libc-2.10.1.so
  - # ln –s libc-2.11.1.so libc.so.6
- Static link solves this problem but it wastes memory and storage.

Solution, and further problems

- The problems are solved by static-link, but it increase consumption of memory and storage.
  - Fortunately, the increased consumption is mitigated by new technique, **deduplication**.
  - SLINY[USENIX'05] developed deduplication in Linux kernel.
  - It looks the problems are solved …
- Two trends
  - Current applications assume dynamic-link and are not re-compiled as static-link easily .
  - Current virtualization offers us deduplication.
    - SLINKY uses special Linux kernel. It is not applied on any OSes.
    - Using virtualization, guest OS only has to consider the solution without regard to physical consumption.

Static-Link is not easy

- Current applications deeply depend on dynamic-link shared libraries for flexibility and for avoiding license contamination problems.

- We tried to re-compile /bin, /sbin, /usr/bin, and /usr/sbin dynamic-linked binaries (1,162) with static-link on Gentoo.
  - 185 (15.9%) binaries are re-compiled with static-link.

- Binary packages make it difficult to re-compile, because they are not easy to get all source code.
  - Commercial applications make problem more difficult.

Self-Contained Binaries

- Self-contained binary translator
  - It is developed to bring a binary to another machine.
  - **It integrates shared libraries into an ELF binary file.**
  - Advantage
    - Prevent Search Path Replacement Attack and Dependency Hell, because it integrates all libraries.
    - Mitigate GOT Overwrite Attack, because the addresses are prefixed for each execution.
  - Disadvantage
    - **Consume more memory and storage than static-link**
- Tools
  - Statifier, Autopacage, Ermine for Linux
  - VMWare “ThinApps(was Thinstall)” for Windows

Statifier (1/2)

- Statifier includes shared library into an ELF binary.
- On Normal binary
  ① `_dl_start()` of `ld-linux.so`
    - Reallocate dynamic link libraries and map them
  ② `_dl_start_user()` of `ld-linux.so`
    - Call initialization functions in libraries
- Statifier creates self-contained binary
  – Take snapshot before `_dl_start_user()` and analyze relocation information of functions of libraries from `/proc/PID/maps`.
  – The libraries and relocation information are embedded into the binary.

Statifier (2/2)

- Self-Contained Binary
  - Relocation information and shared libraries are loaded by the `starter` of statifier.
    - Includes special libraries: `linux-gate.so`, `ld-linux.so`
  - The ELF binary has no `INTERP` segment to call `ld-linux.so`
  - `ldd` command shows no dynamic-link shared libraries

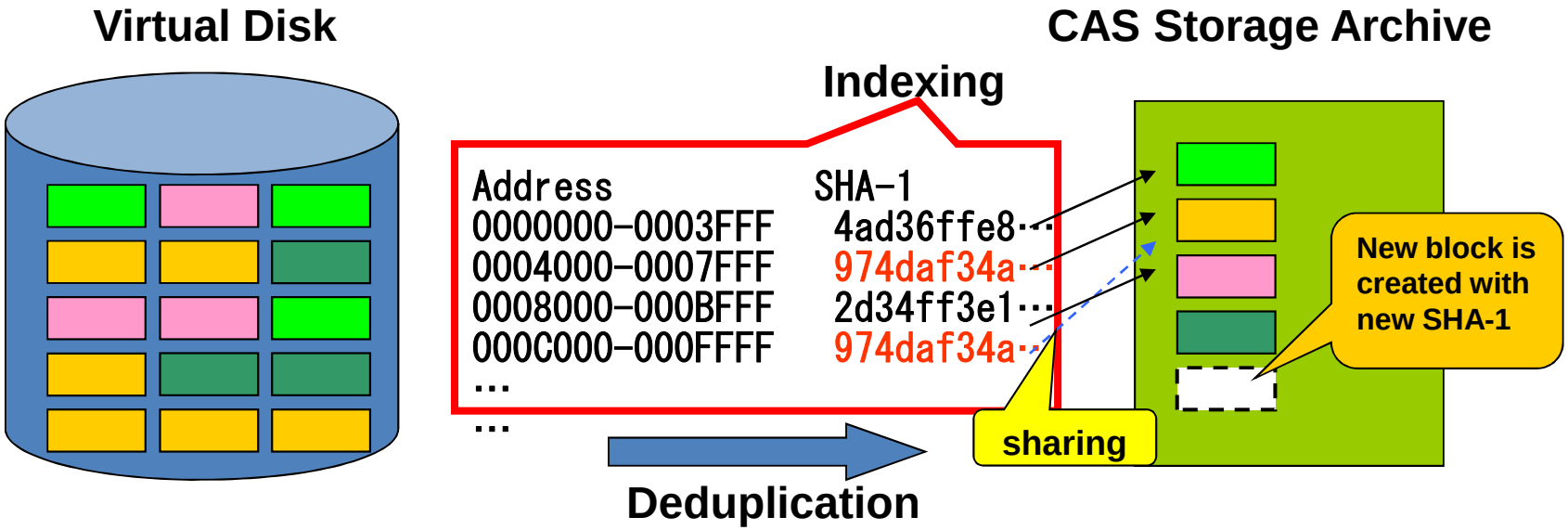
- However, Statifier makes a larger binary than static link.

Deduplication

- Technique to share same-content chunks at block level (memory and storage).
- Same-content chunks are shared by indirect link.
  - It is easy to implement when a virtual layer exists to access a block device.
  - Some virtualizations include deduplication mechanism.

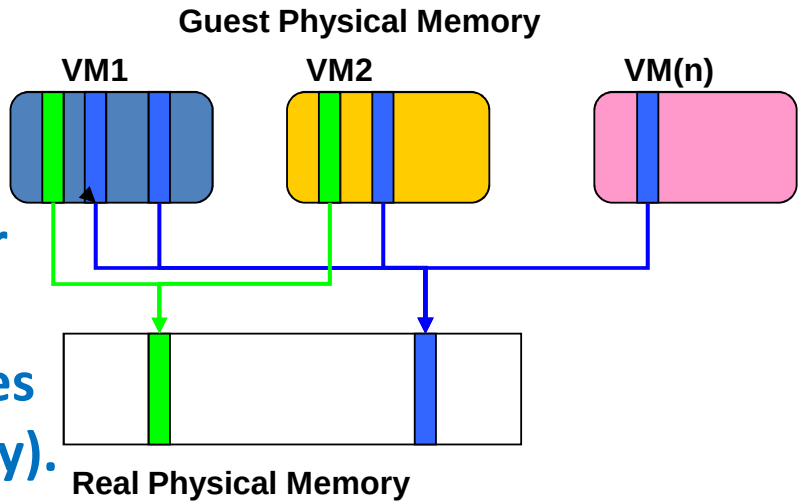
Storage Deduplication

- Used by CAS (Content addressable Storage)
  - data is not addressed by its physical location. Data is addressed by a unique name derived from the content (a secure hash is used as a unique name usually)
  - Same contents are expressed by one original content (same hash) and addressed by indirected link.
    - Plan9 has Venti [USENIX FAST02]
    - NetApp Deduplication (Data Domain) [USENIX FAST08]
    - LBCAS (Loopback Content Addressable Storage) [LinuxSymp09]



Memory Deduplication

- Memory deduplication is mainly used for virtual machines.
- Very effective when same guest OS runs on several virtual machines.
- On Virtual Machine Monitor
  - Disco[OSDI97] has Transparent Page Sharing
  - VMWare ESX has Content-Based Page Sharing [SOSP02]
  - Xen has Satori[USENIX09] and Differential Engine[OSDI08]
- On Kernel
  - Linux has KSM (Kernel Samepage Merging) from 2.6.32 [LinuxSymp09]
    - Memory of Process(es) are deduplicated
    - KVM uses this mechanism
- **These targets are virtual machines, but our proposal uses memory deduplication on a single OS image, which increase same pages with copy of libraries (self-contained binary).**



Evaluation

- Evaluate the effect of moving form logical sharing to physical sharing.
  - Effect of Statifier
    - Applied on binaries under /bin,/sbin,/usr/bin,/usr/sbin of Gentoo (installed on 32GB virtual disk for KVM virtual machine)
  - Memory Deduplication
    - KSM (Kernel Samepage merging) of Linux with KVM virtual machine (758MB).
  - Storage Deduplication
    - LBCAS (Loopback Content Addressable Storage)

Static Analysis of Statifier

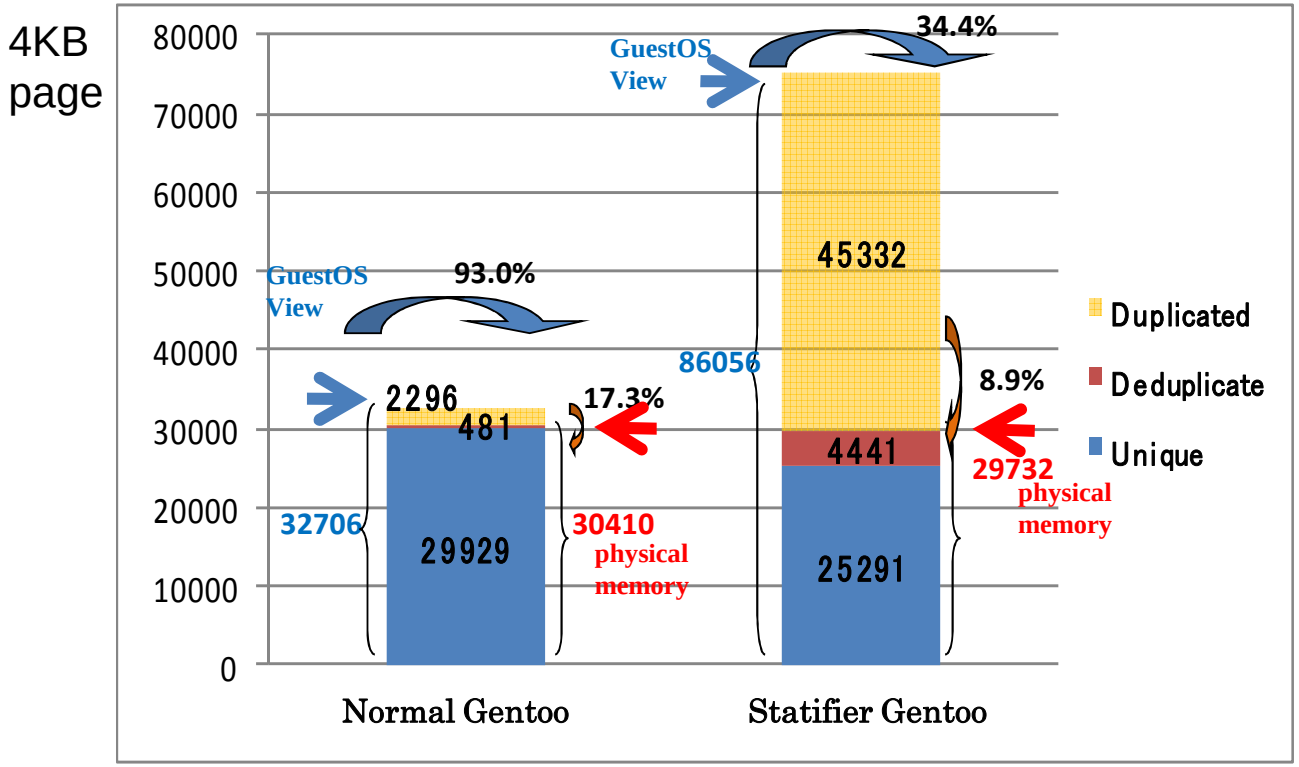
- Gentoo was customized by statifier.
  - The ELF (1,162) binaries under /bin (82 files), /sbin (74), /usr/bin (912), /usr/sbin (94) were customized by statifier.

	Original (Dynamic-link)	Statifier	Increase
Total	87,865,480	3,572,936,704	40.66
Average	75,615	3,074,816	40.66
Max (gnome-open)	5,400	8,732,672	1617.16
Min (qmake)	3,426,340	6,094,848	1.78

- The disk image (includes non-statifiered files) was expnaded from 3.75GB to 7.08GB (1.88 times).

Effect of Memory Deduplication

- Memory usage at the end of login
- Statifier expanded memory consumption from the view of GuestOS,
- but Deduplication reduced physical memory consumption.

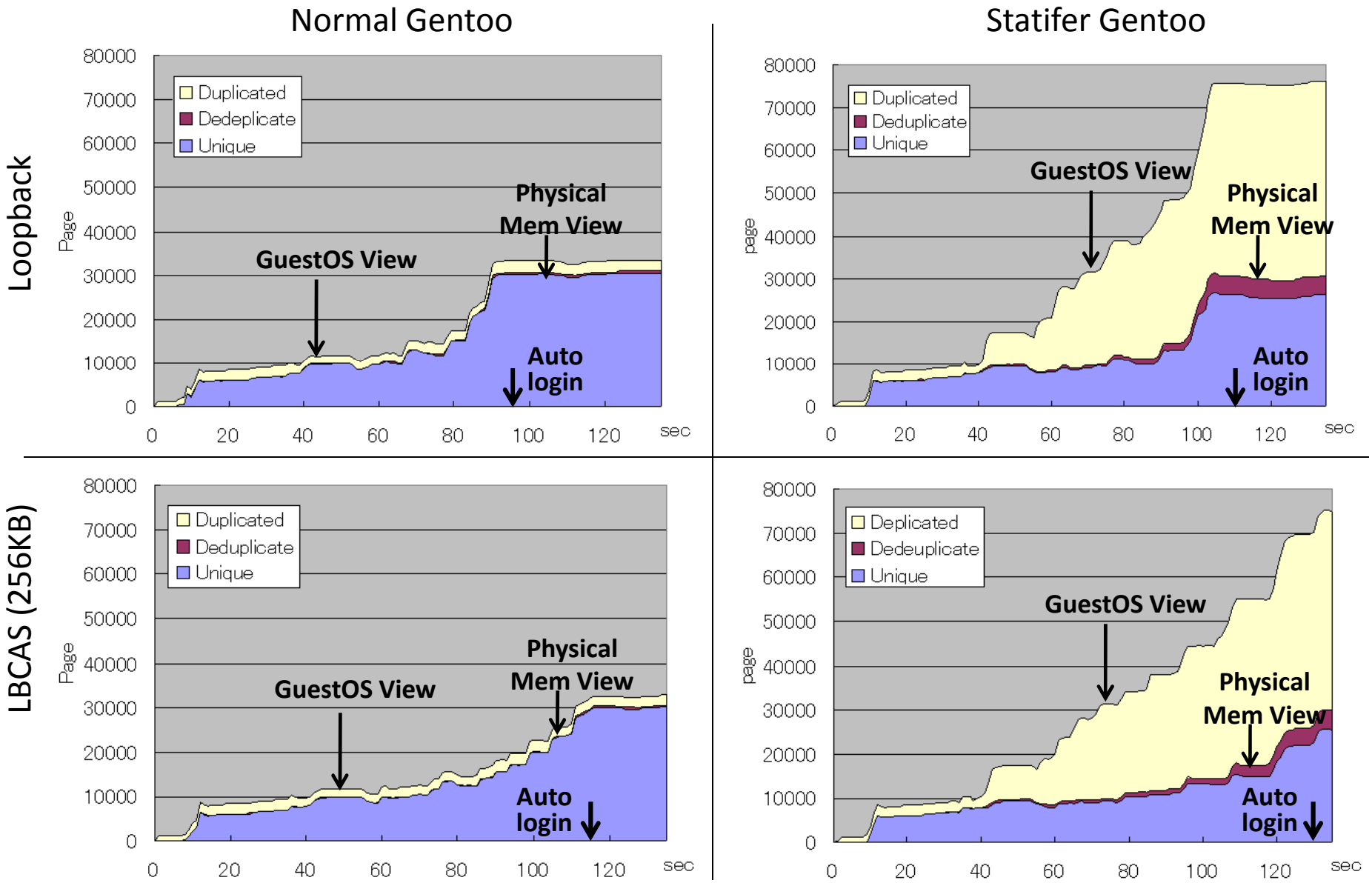


Effect of Storage Deduplication

- Storage usage (static) and total read data at boot (dynamic) .
- Statifier expanded storage consumption from the view of GuestOS on both cases, but Deduplication reduced physical storage consumption in static and dynamic.
- Smaller chunk is easy to be deduplicated but time overhead is large.

	Static		Dynamic (boot)	
	normal	statifier	normal	statifier
On Loopback (Guest OS View)	3,754MB	7,075MB (1.88)	151.7MB	341.0MB (2.25)
LBCAS 16KB	268,454 [4195MB]	4352MB [278,499] (1.04)	---	----
LBCAS 64KB	74,679 [4667MB]	83,863 [5241MB] (1.12)	218MB [3,481]	304MB [4,866] (1.40)
LBCAS 256KB	22,806 [5701MB]	6723MB [26,892] (1.18)	390MB [1,560]	505MB [2,019] (1.29)

Trace of memory consumption



Time overhead at boot

- Statifier reduced the boot time, because it eliminated dynamic reallocation overhead.
- Deduplication increased the boot time. The overhead of KSM and LBCAS was less than 37%.
  - The overhead is a penalty to remove the vulnerabilities of logical sharing.

	Without KSM		With KSM	
	Normal	Statifier	Normal	Statifier
Loopback	95s	84s	95s	105s
LBCAS (256KB)	107s	108s	115s	130s

Conclusion & Discussion (1/2)

- Self-Contained binaries strengthen OS security.
  - Prevent Search Path Replacement Attack, GOT (Global Offset Table) overwrite attack, Dependency Hell
  - Easy to apply on normal OS. It does not require source code and re-compile.
  - Increase consumption of memory and storage.
- Deduplication mitigates the consumption of memory and storage caused by self-contained binary.
  - Encourage moving from Logical sharing to Physical Sharing
- Deduplication is utilized to increase security on single OS.

Conclusion & Discussion (2/2)

- Deduplication will be mainly used on IaaS type (multi-tenants) Cloud Computing.
- Two directions of research
  - Increase code sharing
    - "Return-Oriented Programming" style becomes popular?
      - » Tools: Return Oriented Rootkit [USENIX Security 09]
  - Keep security
    - Code sharing will increase a chance to attack
    - *Attack for deduplication* will be presented in **Rump Session** of USENIX Security.