31st IEEE Symposium on Security & Privacy, Oakland CA, May 16-19 2010

HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity

Zhi Wang, Xuxian Jiang
North Carolina State University

Computer Science
NC STATE UNIVERSITY

Outline

- Motivation
- Design
- Implementation & Evaluation
- Related Work
- Summary

Outline

- Motivation
- Design
- Implementation & Evaluation
- Related Work
- Summary

Virtualization Adoption

- Rapidly growing in industry
  - 16% server workloads on virtual machines now
  - 50% by 2012[1]
- Widely applied to security problems
  - Guest integrity monitoring

    ReVirt (Dunlap et al, OSDI '02), Livewire (Garfinkel et al, NDSS '03), VMwatcher (Jiang et al, CCS '07), Lares (Payne et al, Oakland '08), SIM (Sharif et al, CCS '09)…
  - Guest integrity protection

    SecVisor (Seshadri et al, SOSP '07), NICKLE (Riley et al, RAID '08), HookSafe (Wang et al, CCS '09)…
  - System software analysis

    AfterSight (Chow et al, USENIX ATC '08), K-Tracer (Lanzi et al, NDSS '09), PoKeR (Riley et al, EuroSys '09) …
  - …

[1]:Gartner Symposium/ITxpo 2009

Common Assumption

A Trustworthy Hypervisor!

Bloated TCB of Type I Hypervisors

Hypervisor	Hypervisor SLOC	TCB
Xen-4.0	194K	Xen, Dom0
VMware ESXi[1]	200K	VM Kernel
Hyper-V[1]	100K	Hyper-V, Windows 2008 Server
BitVisor	194K	BitVisor

1. NOVA: A Microhypervisor-Based Secure Virtualization Architecture (Udo Steinberg et al, EuroSys '10)

Vulnerabilities & Attacks

- Common Vulnerabilities and Exposures (CVE)
  - Xen - 26, VMware ESX - 18 (til 11/2009)
- VM escape attacks
  - Xen 0wning Trilogy (Invisible Things Lab, Blackhat '08)
  - Cloudburst: A VMware Guest to Host Escape (Kostya Kortchinsky, Blackhat '09)
- Hypervisor based rootkits
  - SubVirt (King et al, Oakland '06), Blue Pill (Invisible Things Lab, Blackhat '06), Virtiol (Dino A. Dai Zovi, Blackhat '06)

Existing Solutions

- Reduce TCB
  - TrustVisor (McCune et al, Oakland '10), NOVA (Steinberg et al, EuroSys '10) , Improving Xen Security through Disaggregation (Murray et al, VEE '08), …
- Formal verification
  - seL4 (Klein et al, SOSP '09), …

Our goal is to enable self-protection of commodity type-I (bare-metal) hypervisors!

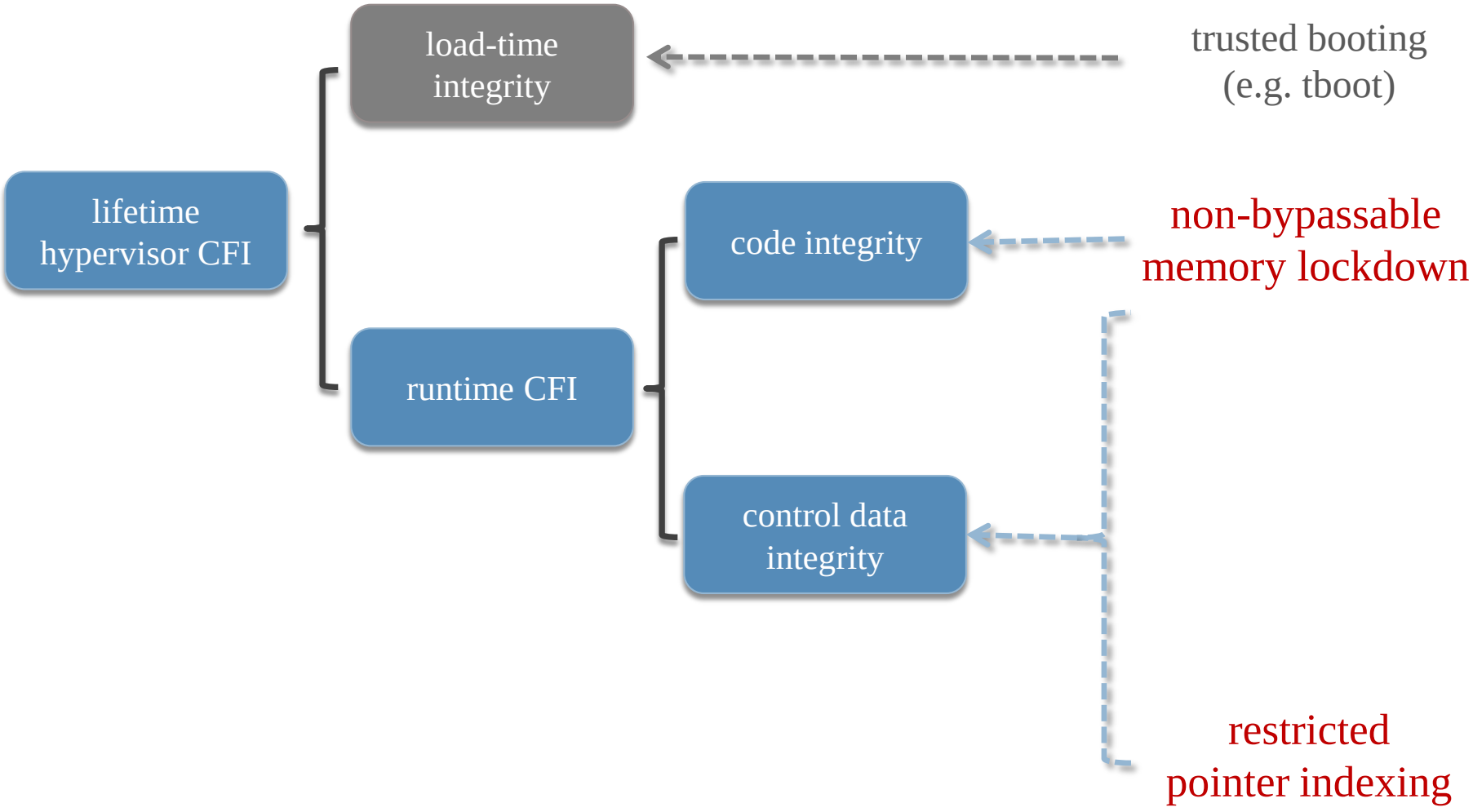
Outline

- Motivation
- Design
- Implementation & Evaluation
- Related Work
- Summary

Assumptions

- Trustworthy (x86) hardware
  - IOMMU to prevent malicious DMA transactions
  - Trusted System Management Mode (SMM)
- Software bugs in the hypervisor

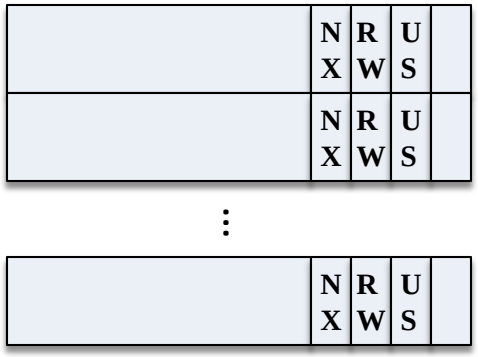
Our Approach: HyperSafe



Non-bypassable Memory Lockdown

x86 Paging Mode

- Page tables determine memory properties
  - Permissions in a page table entry:
    - NX – Non-executable
    - R/W – Read-only or Writable
    - U/S – User or supervisor page
  - W ⊕ X: a page can be either writable or executable, but not both



- All memory accesses by software are translated and controlled by page tables
  - Including reads/writes of page tables

HyperSafe's Memory Lockdown

- Pitfalls in existing W ⊕ X
  - Mixed code and data
    - → Mixed code and data are prohibited
  - Double mapping with conflicting attributes
    - → Double mapping must have conforming attributes
  - Writable page tables
    - → Read-only page tables

No code can modify the write-protected hypervisor code and data!

Challenge

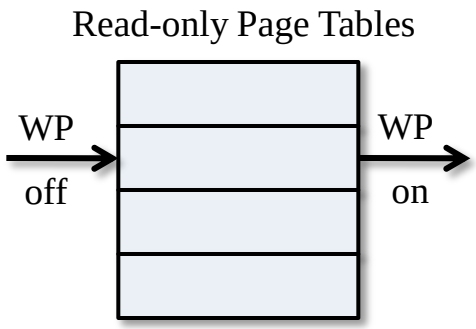
How to safely allow benign page table updates???

Hardware Feature to the Rescue!

- Write-protect (WP) bit in CR0 controls interaction of supervisor and read-only pages
  - WP = 1: Read-only pages are protected even from supervisor
  - WP = 0: Supervisor can write into read-only pages

Benign Page Table Updates

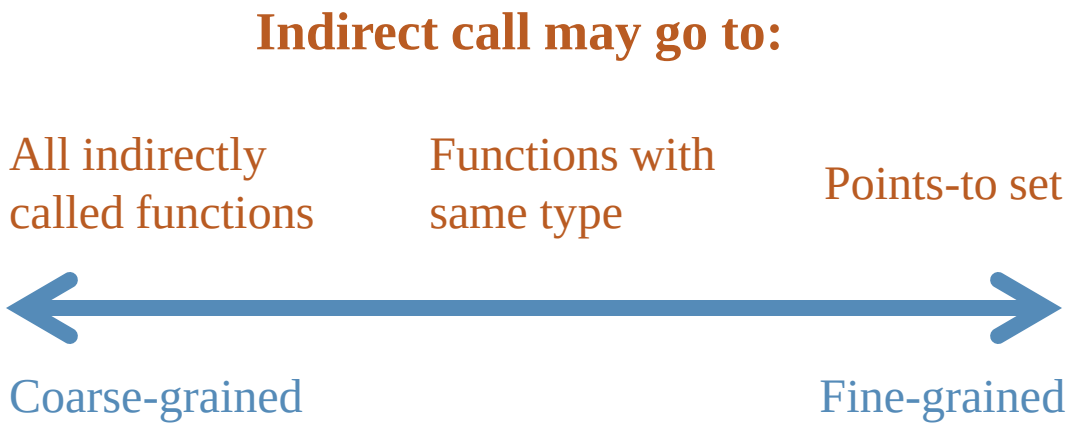
- WP = 1 by default to lock down memory
- Update page table atomically
  1. Disable interrupt
  2. WP = 0
  3. Verify proposed change
  4. Update read-only page table
  5. WP = 1
  6. Enable interrupt



Restricted Pointer Indexing (RPI)

Control Flow Integrity (CFI)

- CFI: runtime execution paths must follow control flow graph (CFG)
- CFG may have different granularities



CFG Construction in HyperSafe

- Points-to analysis required
- Manual analysis to handle domain knowledge / assembly code in prototype
  - e.g. assembly code to access per-cpu data (function pointers) in gs segments

Enforce Control Flow Integrity

- Restricted Pointer Indexing
  - Collect control data into tables (protected by memory lockdown)
  - Replace control data with the indexes to the table
  - Convert the index back to transfer control

Only legitimate control data in the table can be used for control flow transfer!

Outline

- Motivation
- Design
- Implementation & Evaluation
- Related Work
- Summary

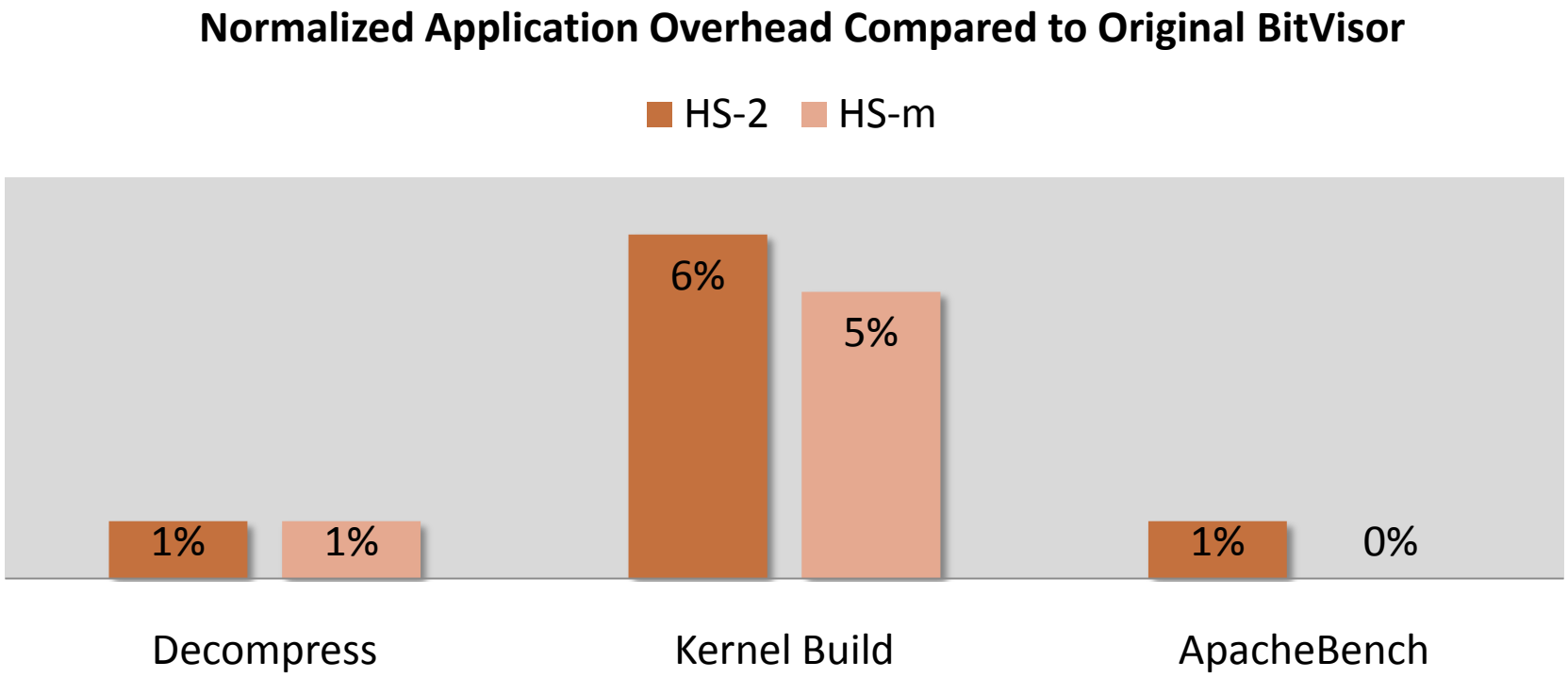
Implementation

- Implementing techniques:
  - Memory lockdown: modify hypervisor's memory management code
  - Restricted Pointer Indexing: extend LLVM compiler to instrument related instructions
- Prototypes of HyperSafe:
  - Full support for BitVisor
  - Partial support for Xen, additional engineering needed

Security Analysis

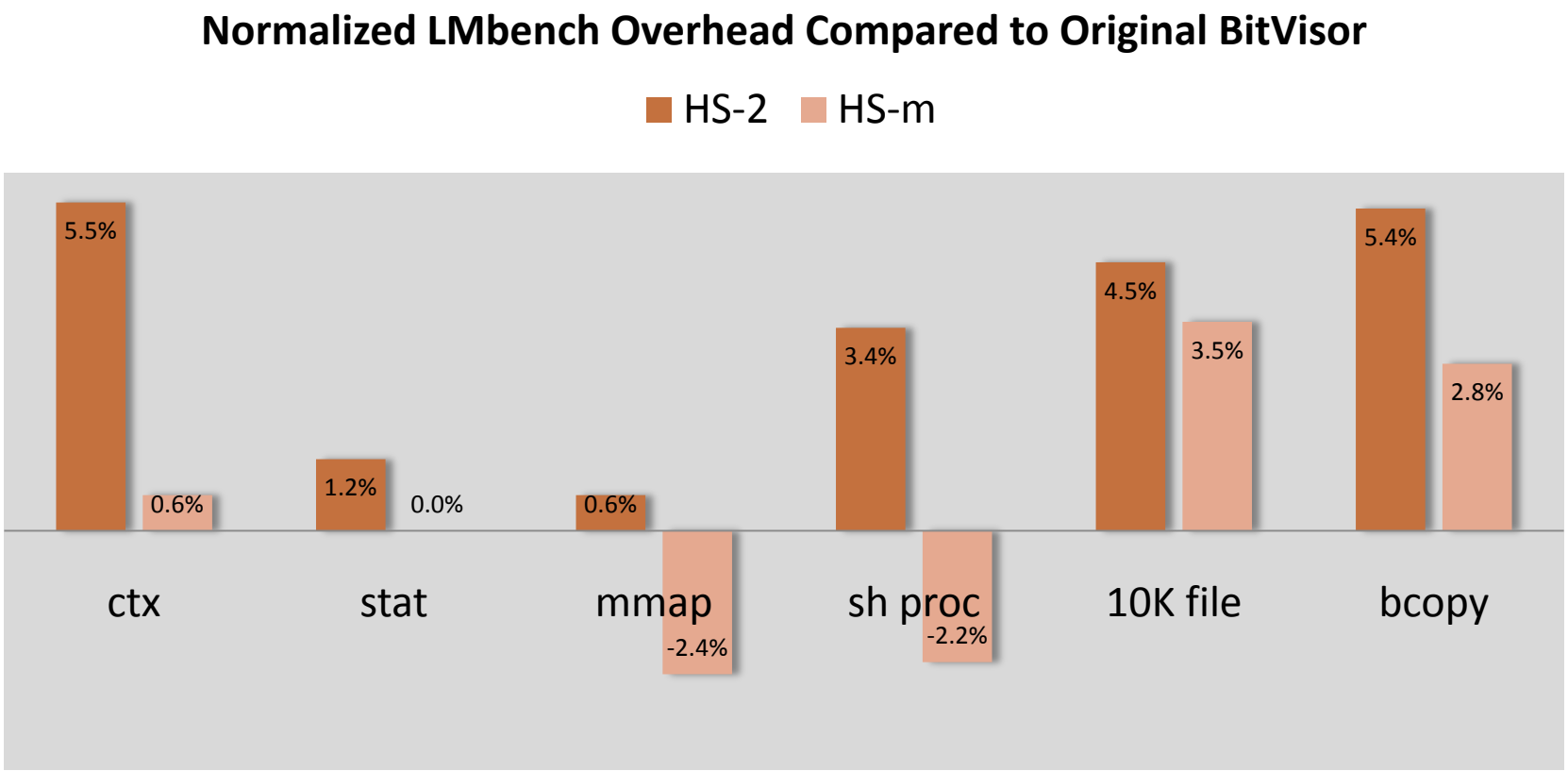
- Disable WP bit
  - Misuse page table update function ← RPI
- Subvert page table
  - Misuse page table update function ← RPI
  - Map hypervisor memory to a compromised guest VM ← Memory lockdown
- Return-oriented programming ← Memory lockdown, RPI

Performance: Applications



- HS-2 implements coarse-grained RPI with two target tables (return instructions and indirect calls)
- HS-m implements fine-grained RPI with one target table per function and indirect call

Performance: LMbench

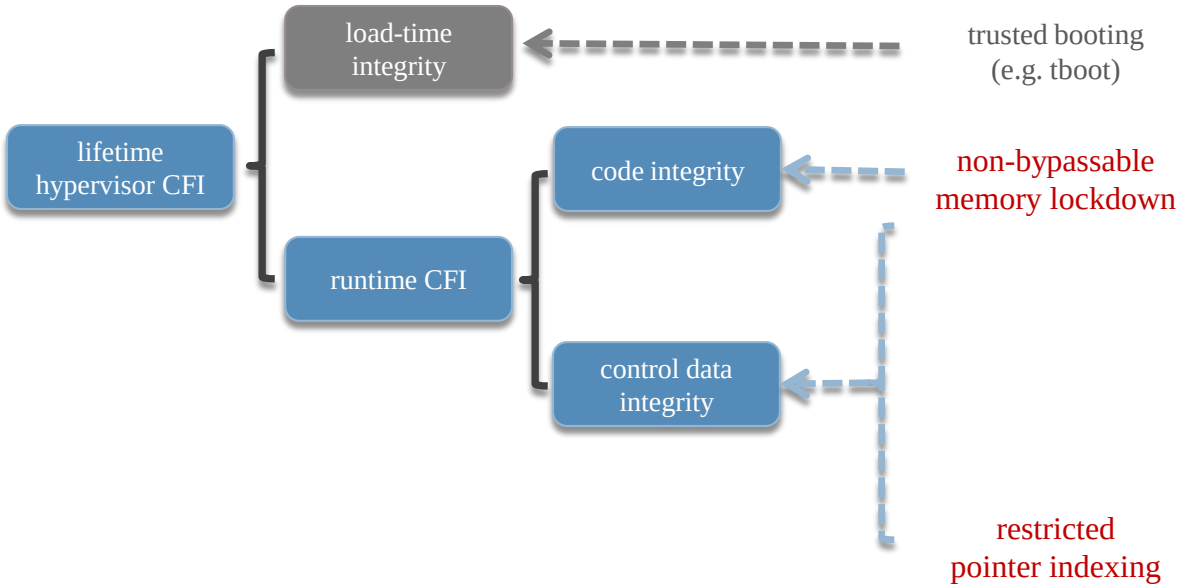


Related Work

- Program Analysis and Formal Proof
  - seL4 (Klein et al, SOSP ‘09), WIT (Akritidis et al, SOSP ‘08), KLEE (Cadar et al, OSDI ‘08), …
- Guest Integrity Monitoring or Protection
  - SIM (Sharif et al, CCS ‘09), SecVisor (Seshadri et al, SOSP ‘07), SBCFI (Petroni et al, CCS ‘07), …
- Trusted Computing
  - TrustVisor (McCune et al, Oakland ‘10), Flicker (McCune et al, EuroSys ‘08), Pioneer (Seshadri et al, SOSP ’05), …

Summary

- HyperSafe is a lightweight approach to provide lifetime control-flow integrity for commodity Type-I hypervisors.



Thanks, Questions?