IskiOS: Lightweight Defense Against Kernel-Level Code-Reuse Attacks

Spyridoula Gravani, Mohammad Hedayati, John Criswell, Michael L. Scott

Department of Computer Science, University of Rochester

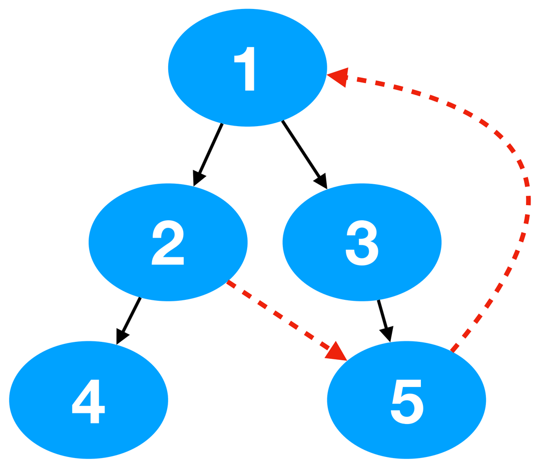
UNIVERSITY *of* ROCHESTER

Motivation

Commodity operating systems are the most trusted component in the software stack of modern computing platforms, yet, they are vulnerable to code-reuse attacks.

Code-reuse Attack

- Goal: control program behavior
- Typically:
  - Memory-safety error exploitation (e.g., buffer-overflow)
  - Corruption of return-address of a function call on stack
  - Control-flow redirection to desired code sequence



Insufficient Kernel Defenses

- Mostly label-based Control-flow Integrity (CFI) approaches
  - Over-permissive control-flow policies due to static analysis imprecision and incompleteness
  - Lack of return-address protection

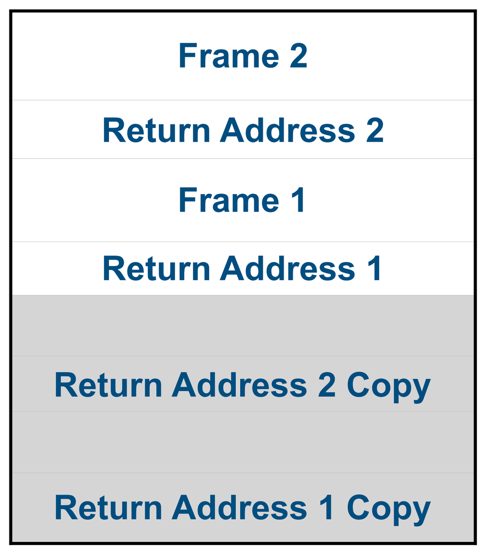
IskiOS

A Prevents Return-Address Corruption

Shadow Call Stack

- ☑ Write-protected
- ☑ Race-free
- ☑ Efficient

Prevents all *return-based* code-reuse attacks.

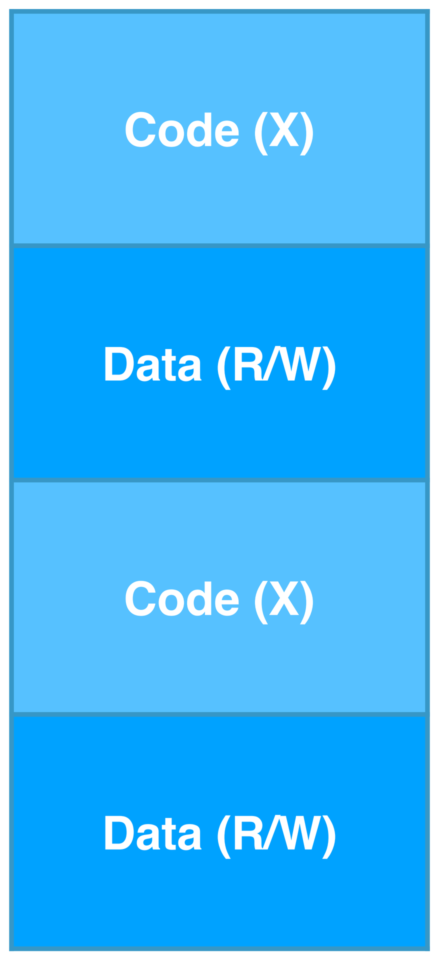


B Prevents Direct Disclosure of Code Layout

eXecute-Only Memory (XOM)

- ☑ Code cannot be read/ written
- ☑ No layout re-arrangement
- ☑ Deployed diversification entropy is maintained

Prevents all *just-in-time* attacks via direct reads.

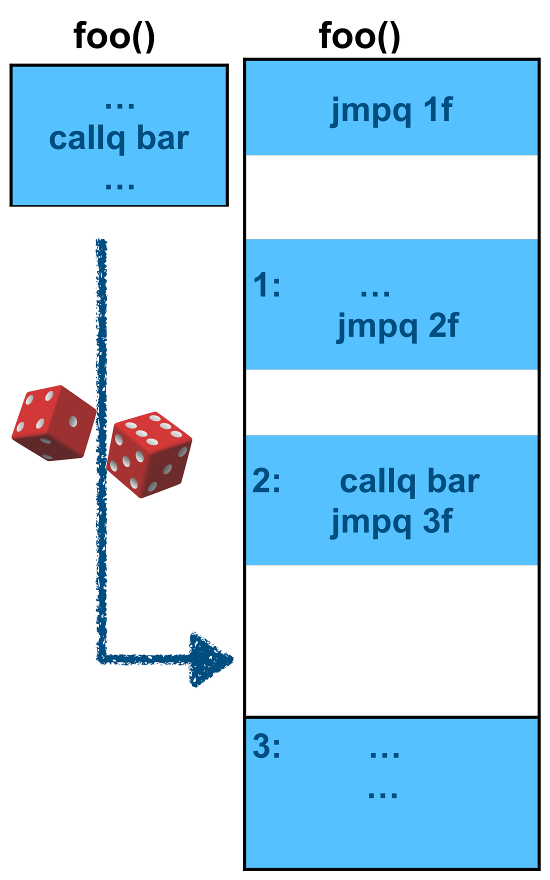


C Mitigates Indirect Memory Disclosure

Trap Padding

- ☑ Function entries and callsites are followed/ preceded by random number of traps
- ☑ Pointers in readable memory do not leak information about code layout

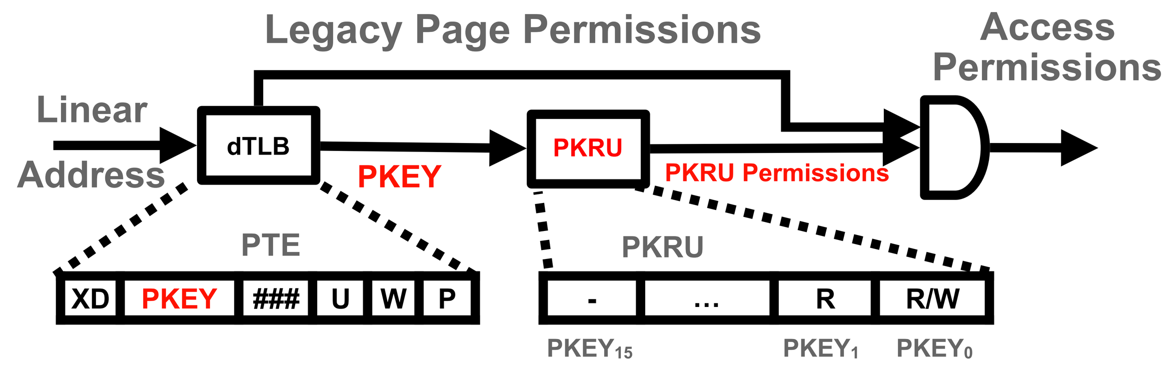
Mitigates *just-in-time* attacks via leaked code pointers.



Design

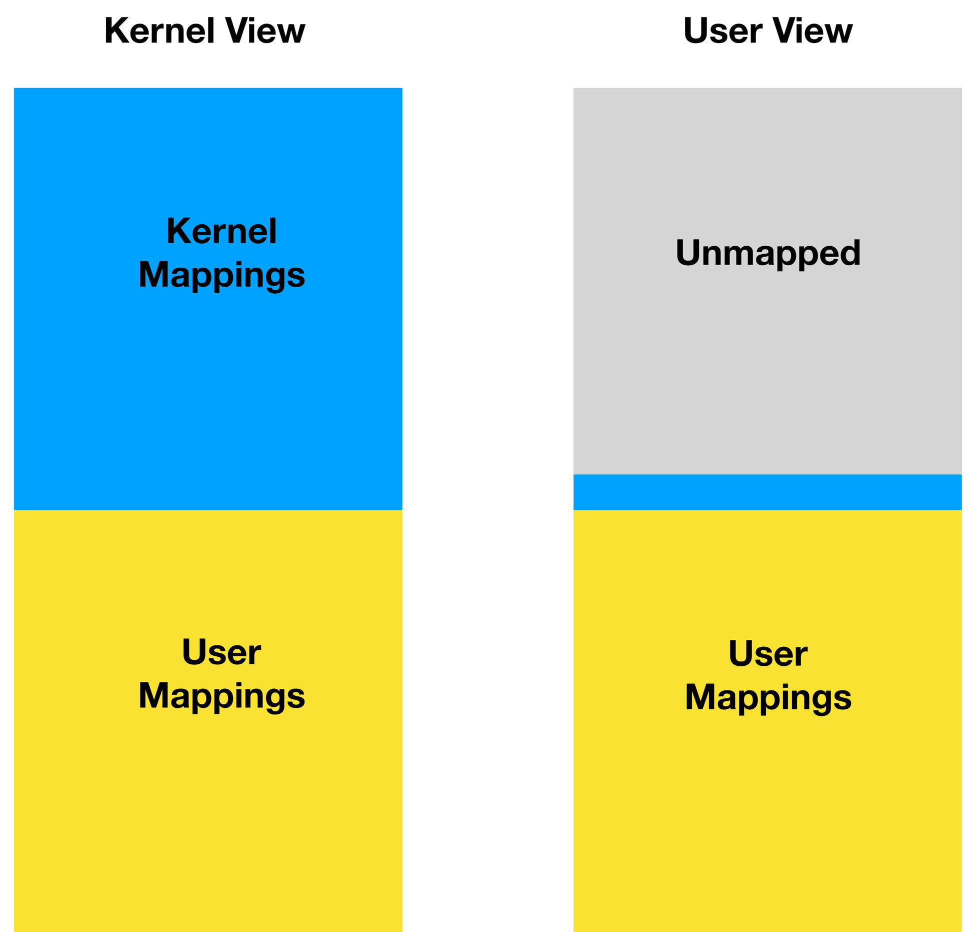
Intel Protection Keys for Userspace (PKU)

- **PKRU:** 32-bit user-accessible register
- `wrpkru/rdpkru` instructions
- Up to **16 protection domains** within a singe address space
- **Applies to PTEs with U/S bit set (i.e., user memory only)**



Kernel Page Table Isolation (KPTI)

- Software-only mitigation for Meltdown attack
- Separate page tables for user/kernel isolation
- U/S bit redundant
- **Key idea:** mark ***all*** memory as user (**U/S clear**) and rely solely on KPTI for isolation
- Enables ***Protection Keys for Kernel (PKK)***



Evaluation

Benchmark	vanilla		pti	pti+xom	pti+xom+cph	pti+xom+cph+ss-lfo-swo
Apache	30131.13	*req/s*	1.99%	7.93%	31.34%	58.58%
Kbuild	56.93	*sec*	1.48%	∼ 0%	2.89%	7.20%
GnuPG	15.30	*sec*	∼ 0%	∼ 0%	1.18%	3.91%
OpenSSL	3814.23	*sign/s*	∼ 0%	∼ 0%	∼ 0%	∼ 0%
PyBench	1789	*msec*	∼ 0%	∼ 0%	∼ 0%	∼ 0%
PHPBench	477859	(*score*)	∼ 0%	∼ 0%	∼ 0%	∼ 0%
PostMark	5210	*trans/s*	8.91%	8.29%	19.63%	56.90%
SQLite	549.33	*sec*	3.87%*	10.45%*	5.66%	3.03%
Redis	2.16*M*	*gets/s*	4.39%	4.62%	6.19%	20.72%
Nginx	34193.45	*req/s*	7.28%	9.33%	28.70%	56.09%
Memcached	106973.37	*gets/s*	9.72%	7.33%	19.60%	49.97%
Geomean			0.92%	0.83%	1.58%	3.92%

* Indicates that the relative standard deviation in performance among test runs is between 3.5% and 12.8%.

TABLE II: IskiOS runtime overhead (% over `vanilla` Linux) on the Phoronix Test Suite.

vanilla	pti+xom		pti+xom+cph		pti+xom+cph+ss	
22.55 *MB*	∼ 22.55 *MB*	∼ 0%	88.04 *MB*	292%	90.02 *MB*	299%

TABLE III: IskiOS code size for different configurations and overhead over `vanilla` Linux.

This work was supported in part by NSF grants CCF-1717712, CCF-1422649, and CNS-1618213, and by a Google Faculty Research award.

The 4th NSF Secure and Trustworthy Cyberspace Principal Investigator Meeting (2019 SaTC PI Meeting)
October 28-29, 2019 | Alexandria, Virginia