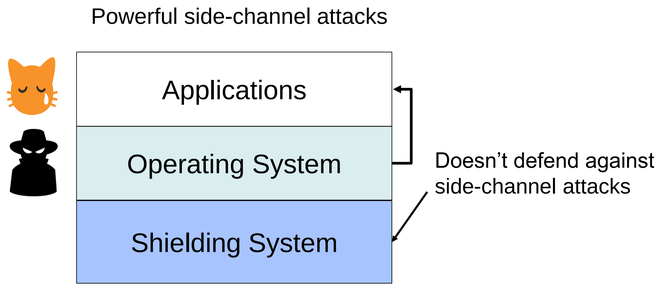
Shielding Software From Privileged Side-Channel Attacks

Xiaowan Dong[1], Zhuojia Shen[1], John Criswell[1], Alan L. Cox[2], Sandhya Dwarkadas[1]

[1]University of Rochester, [2]Rice University

OS-Launched Side-Channel Attacks

A compromised OS can launch side-channel attacks to steal confidential application data



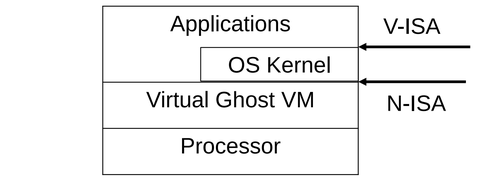
Compromised OS infers the victim application's memory access behavior via

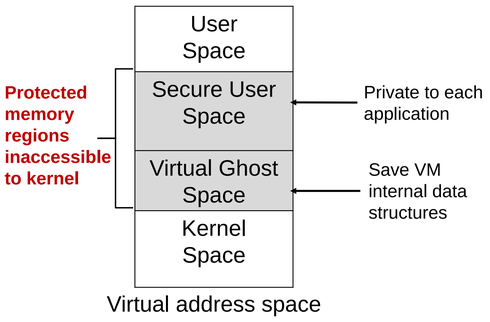
- ***Page table side channels***
  Trace page faults, page table updates
- ***Cache side channels***
  Time accesses to shared caches

Virtual Ghost

A compiler-based virtual machine (VM) that protects confidential application data from OS

- *Software fault isolation (SFI)* instruments every kernel load and store
- OS has to invoke *SVA-OS instructions* to perform privileged operations





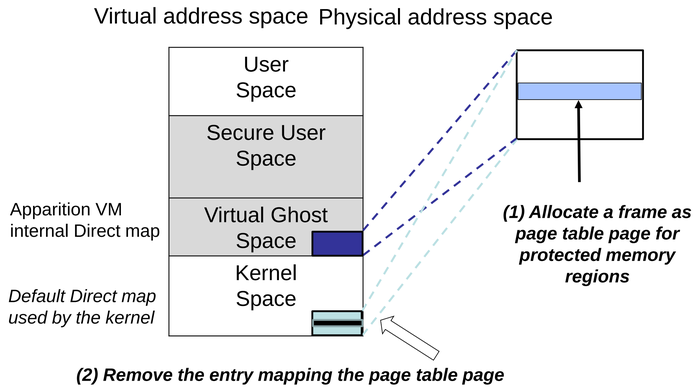
Apparition

Defends against page table and last-level cache side-channel attacks launched by OS

Apparition = Virtual Ghost + defenses

- Prevents OS from reading or writing
  - *Secure user space*
  - *Page table pages mapping secure user space*
  - *LLC lines of secure user space*
- Controls native code generation of the kernel

Page Table Side-channel Defenses



- *Direct map*: a range of virtual memory mapping the entire physical memory as a single block
- Page table pages accessed via direct map
- Prevent OS from reading or writing the page table of the protected memory regions

➡ Remove the entry mapping the page table page from the kernel's direct map

- Apparition VM manages secure user space memory allocation instead of OS
- Map physical frames upon allocation rather than at access time (disabling lazy memory allocation)

LLC Side-channel Defenses

Partition the LLC using Intel cache allocation technology (CAT)

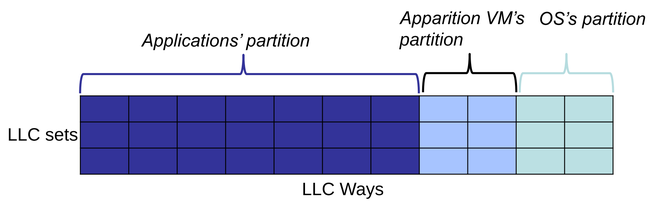
- Assign different partitions to applications needing protection, OS and Apparition VM

Apparition VM

- Configures cache partitioning at boot time
- Prevents the OS from reconfiguring the partitions via its V-ISA
- Switches to the corresponding partition based on the code running (application, Apparition VM, and OS)

Each application has a *private* partition

- Flush the cache over context switch when multiple applications share the same partition



Spectre and Meltdown Attacks

Apparition helps prevent LLC side-channel leaks

Our HASP paper [1] presents *Spectre-resistant (variant 1) SFI*

- Bit-masking instructions add data dependence between memory load and bounds check
- Bit-masking SFI is faster than *lfence*
  - Multiple bounds check can run in parallel
- Suggested enhancements to SFI using Intel MPX

[1] X. Dong, Z. Shen, J. Criswell, A. Cox, and S. Dwarkadas. **Spectres, Virtual Ghosts, and Hardware Support.** In HASP '18.

Evaluation

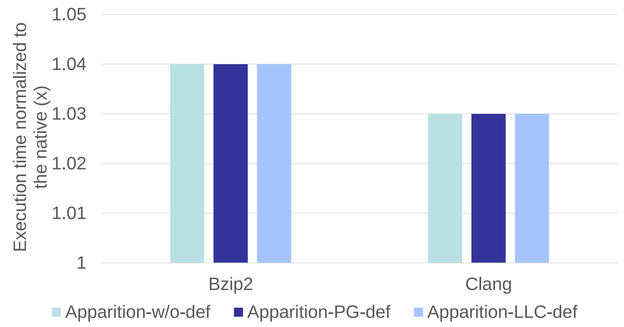
Apparition defends against page table and LLC side-channel attacks with low overhead (1% to 18% compared to Native FreeBSD)

The overhead of page table side-channel defenses

- Mapping more frames than necessary due to disabling lazy memory allocation

The overhead of LLC side-channel defenses:

- Cache partition switching
- Smaller space on LLC



File Size	Apparition-w/o-def	Apparition-PG-def	Apparition-LLC-def
1 KB	9.5 ms	23.7 ms	12.1 ms
2 KB	9.5 ms	23.8 ms	12.1 ms
…	x ms	(x + 14) ms	…
16 MB	386.2 ms	400.1 ms	394.6 ms
32 MB	761.8 ms	776.1 ms	776.6 ms

Acknowledgement

The authors thank the anonymous reviewers for their insightful feedback. This work was supported by NSF Awards CNS-1319353, CNS-1618497, CNS-1618588, CNS-1629770, and CNS-1652280