Hardware Contention Counters

Austin Harris

Side & Covert Channels

- A side channel occurs when a sensitive application inadvertently leaks information to a co-resident attacker through contention over shared micro-architectural resources.
- A covert channel occurs when a malicious application (e.g. document reader) intentionally sends sensitive data to a co-resident process.
- These channels can be used by attackers in the cloud (e.g. EC2) who get allocated to the same instance as the victim.

Traditional Performance Counters

- Existing counters don't differentiate events due to contention with another thread and those intrinsic to the application.
- Code instrumentation has been used to estimate contention [1] among threads in a multi-core system to eliminate false-sharing with an overhead of ~5x.

Detecting An Eavesdropper

- [2] shows an intelligent detector "Claude" that is able to detect a co-resident eavesdropper "Eve" attempting to exploit these channels using existing x86 performance counters (e.g. cache misses.)
- Claude can periodically run and check for the presence of Eve.
- Claude's behavior forces Eve to limit contention in an attempt to evade detection.
- Thus it is difficult for Claude to differentiate between events within the detection application (or within the O/S on behalf of Claude) with those due to the malicious activity of Eve.

Cache Contention Counter Prototype

- Initial prototype modifies Rocket to track L1 data cache contention for a single "secure domain" in the entire system.

Overview Of RISC-V Modifications

O.S./Toolchain:

- Addition of protected secure domain control-status register (CSR).
- System call to enable/disable secure domain mode.
- O/S support to save/restore secure domain CSR.

Rocket:

- Cache line ownership bits added to L1 tag store meta-data.
- Uarch0-2 CSRs purposed for cache counters.

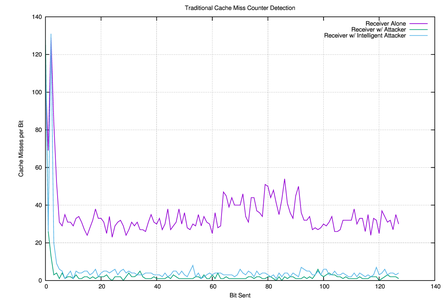
Linux Modifications

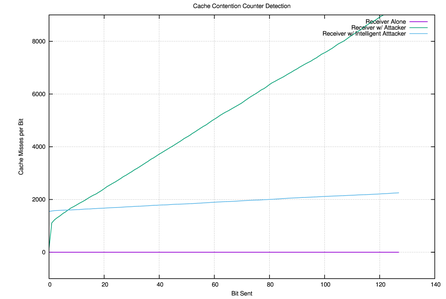
- Save/restore of secure domain CSR using pt_regs added to entry.S.
- System-call to set pt_regs->secure_domain.
- Enforces that only one alive process can be the secure-domain .

Rocket Modifications To Track L1 Contention

- L1 meta-data contains bit to track secure domain ownership
- Traditional L1 cache miss counter incremented on each MSHR allocation.
- Contention cache counter incremented when unsecure domain replaces a secure domain line.
- Alternate cache contention counter incremented when secure domain allocates a line to a set last owned by the unsecure domain.

Evaluation Of L1 Covert Channel Detection





- Evaluated on a Zedboard, configured with a 4-way, 32 set L1 D-cache.
- Easy for Eve to lower contention and hide within the noise with traditional cache miss counter.
- Contention cache counter clearly differentiates isolated execution and concurrent execution with Eve, even when Eve attempts to hide in the noise.

Future Work

- Support for tracking more than one secure domain.
- O.S. support to save/restore counters (perf, etc. support?)
- Defense mechanism when an eavesdropper is detected.
- Applications to QoS?

Contention tracking for other resources:

- Last-Level Cache, Memory Bus, DRAM System
- Branch Predictor, Functional Units

References:

[1] Qin Zhao, David Koh, Syed Raza, Derek Bruening, Weng-Fai Wong, and Saman Amarasinghe. Dynamic cache contention detection in multi-threaded applications. In Proceedings of the 7th ACM SIGPLAN/SIGOPS Inter- national Conference on Virtual Execution Environments, VEE '11, pages 27–38, New York, NY, USA, 2011. ACM. [2] C. Hunger, M. Kazdagli, A. Rawat, A. Dimakis, S. Vishwanath, and M. Tiwari. Understanding contention-based channels and using them for defense. In High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on, pages 639–650, Feb 2015.

THE UNIVERSITY OF TEXAS AT AUSTIN

SPARK RESEARCH LAB
SECURITY, PRIVACY AND COMPUTER ARCHITECTURE