Architectural Attacks and their Mitigation by Binary Transformation

Eran Tromer, MIT

Joint work with

Thomas Ristenpart
Hovav Shacham
Stefan Savage
(attacks)

and

Saman Amarasinghe
Austin Chu
Ronald Rivest
Qin Zhao
(mitigation)
Security of virtualization in cloud computing

What if someone running on the shared hardware is malicious?
Virtualization

20 mattresses

Virtual machine manager

Virtual memory

Hardware
Cross-talk through architectural channels
Cross-talk through architectural channels

- Contention for shared hardware resources
Cross-talk through architectural channels

- Contention for shared hardware resources
- Example: contention for CPU data cache
Cross-talk through architectural channels

- Contention for shared hardware resources
- Example: contention for CPU data cache
Cross-talk through architectural channels

- Contention for shared hardware resources
- Example: contention for CPU data cache
Cross-talk through architectural channels

- Contention for shared hardware resources
- Example: contention for CPU data cache leaks memory access patterns (timing + address)
**Cross-talk through architectural channels**

- Contention for shared hardware resources
- Example: contention for CPU data cache leaks memory access patterns (timing + address)
- This is sensitive information!
- Example: Steal encryption keys in 65ms from OS kernel

[Osvik Shamir Tromer 05] (non-virtualized process vs. kernel)
Demonstrated, using Amazon EC2 as a study case:

- **Cloud cartography**
  Mapping the structure of the “cloud” and locating a target on the map.

- **Placement vulnerabilities**
  An attacker can place his VM on the same physical machine as a target VM (40% success for a few dollars).

- **Cross-VM exfiltration**
  Once VMs are co-resident, information can be exfiltrated across VM boundary.
  → covert channels
  → keystroke timing eavesdropping
  → password theft  [Song Wagner Tian 01]

*All via standard customer capabilities, using our own VMs to simulate targets. We believe these vulnerabilities are general and apply to most vendors.*
Countermeasures?

Secure

[Naq09] [MR04] [GO95]
OS mode
bitslice
[Inte105]
No cache
No sharing

Efficient

[WL07] [ZZLP04]
Ignore it

Generic
DynamoREA
Dynamic Runtime Enforcement of Abstraction

Approach:
**Dynamic binary rewriting**

Transform x86 instructions on-the-fly to eliminate information flow through architectural effects.

Supports common apps on COTS platforms (Linux x86).

Tool: VMware’s DynamoRIO. Observe and modify:
- instructions
- memory management
- I/O
- system calls
DynamoREA transformations

**Example:**
Degrade observation of timing

**Example:**
Inject noise/delays to hide leakage signal

**General:**
Make execution a deterministic function of what the process knows anyway
→ indistinguishable from a **leak-free system**
→ attacker learns nothing
Goal:
Securely run existing apps on leaky platforms.

Methodology:
• **Secure by default.**
• Optimize handling of common cases for efficiency.

Currently:
Proof-of-concept prototype. Keep posted!

Contact: tromer@mit.edu