Whispers in the Hyper-space: High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu*, Zhang Xu, Haining Wang
The College of William and Mary

* Now affiliated with NEC Laboratories America Inc.

Virtualization and Cloud Computing

- **Server Virtualization**
  - Consolitates workload
  - Simplifies resource management
  - Enabling Utility-based Cloud Computing

- **Server Virtualization Technologies**
  - Goal: Computing consolidation
  - Design: Logically-separate but physically-shared
  - NOT equivalent to physically separated machines
  - Non-negligible differences expected, e.g. covert channels

The Threat of Covert Channels

- **Covert Channels**
  - Exploit imperfection in the isolation of shared resources
  - Enable communication bypassing mandatory access controls (i.e. data exfiltration)

- **On non-virtualized systems**
  - File system objects, network stack, shared processor cache, input device, etc.

- **On virtualized systems**
  - No practical exploit to date

Related Work

- **Precautionary Research**
  - Hey, you, get off of my cloud (Ristenpart et al., CCS'09)
    - Bandwidth 0.02bps << 1bps by TCSEC
  - Follow up research (Xu et al., CCSW'11)
    - Improved bandwidth 3.2bps (±10% error rate)
    - Conclusion: can do very limited harm in the cloud

- **Industry Response**
  - Amazon EC2 Dedicated Instances
    - Significant extra-cost (Targeting the "paranoid high-end?"
Whispers in the Hyper-space

High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang

Project Outline

- Understand Existing Cache Covert Channel
  - Why it doesn’t work in the Cloud

- Design a New Covert Channel
  - Targeting cross-VM communication

- Realistic Evaluation
  - In-house and on Amazon EC2

Struggles of Cache Covert Channel

- Classic Cache Channel
  - A hybrid timing & storage channel

  ![Cache Channel Diagram]

- Classic Cache Channels
  - Works very well on hyper-threaded processor
  - Bandwidth ~400 kilo bytes/sec
  - ~3,000,000 bits/sec
  - Performs very poorly on virtualized environment
  - Bandwidth ~1 bit/sec

- Reasons for Struggling
  - Addressing Uncertainty
  - Scheduling Uncertainty
  - Cache Physical Limitation
Whispers in the Hyper-space

High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang

---

**Struggle #1: Addressing Uncertainty**

- Virtualization
  - Additional Layer of Indirection (or, complication)

---

**Struggle #2: Scheduling Uncertainty**

- Cache “Round-trip” Requirement
- Strict Round-Robin Scheduling

---

**Struggle #3: Physical Limitation**

- Require stably shared cache, however
  - Non-participating workload destabilize the cache
  - Cores between physical processors do not share cache!
**Project Outline**

- Understand Existing Cache Covert Channel
  - Why it doesn’t work in the Cloud

- Design a New Covert Channel
  - Targeting cross-VM communication

- Realistic Evaluation
  - In-house and on Amazon EC2

---

**Redesigning Data Transmission**

- Change of Data Modulation Technique
  - Cache region based encoding no longer suitable
  - State Patterns \(\rightarrow\) Timing Patterns

![Timing Patterns](image)

---

**Redesigning Data Transmission**

- Change of Data Modulation Technique
  - Cache region based encoding no longer suitable
  - State Patterns \(\rightarrow\) Timing Patterns

- Change of Scheduling Requirement
  - Signal generation and detection are instantaneous

![Timing Diagram](image)
Whispers in the Hyper-space

High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang

Demonstration
- L2 cache channel using new transmission scheme
  - Intel Core2, Hyper-V, Windows Guests

![Graph showing access latency over time with 39 bits in 200μs equating to 190 kilo bits / sec]

Progress So Far...
- Addressing Uncertainty
- Scheduling Uncertainty
- Cache Physical Limitation

An Insurmountable Limitation
- A shared cache is not guaranteed
  - Virtual processor migration results in frequent interruption

![Diagram of processor cores and cache levels with sender and receiver symbols]

The Memory Bus: Fibers of the Hyper-space
The Bus Contention Channel

- Look beyond the processor cache...
- ...we see the memory bus

Exploiting Atomic Instructions

- Atomic Instructions
  - Special x86 memory manipulations
  - Ensures operation atomicity
  - Very useful for thread synchronization...
  - ...and covert channels

Exploiting Atomic Instructions

- Theory vs. Reality
  - Theoretical
    - Performance impact is localized
  - Pre-Pentium Pro
    - Atomicity is implemented using bus lock signal
Exploiting Atomic Instructions

Theory vs. Reality

- Theoretical (Not Exploitable)
  - Performance impact is localized
- Pre-Pentium Pro (Exploitable)
  - Atomicity is implemented using bus lock signal
  - Pentium Pro – Pre-Nehalem
  - Bus lock is reduced, but not eliminated

Exploiting Atomic Instructions

When target data can be put on a cache line
- Cache line locking is performed

But when target data spans two cache lines...
- Reverts to old bus locking behavior

- Coordinated flush of all in-flight memory operations

Exploiting Atomic Instructions

- Theory vs. Reality
  - Theoretical (Not Exploitable)
    - Performance impact is localized
  - Pre-Pentium Pro (Exploitable)
    - Atomicity is implemented using bus lock signal
  - Pentium Pro – Pre-Nehalem (Exploitable)
    - Bus lock is reduced, but not eliminated
  - Nehalem ~ ?
    - No shared memory bus!
Exploiting Atomic Instructions

- **Theory vs. Reality**
  - Theoretical (Not Exploitable)
    - Performance impact is localized
  - Pre-Pentium Pro (Exploitable)
    - Atomicity is implemented using bus lock signal
  - Pentium Pro – Pre-Nehalem (Exploitable)
    - Bus lock is reduced, but not eliminated
  - Nehalem – ? (Still Exploitable)
    - No shared memory bus!

Demonstration

- **Memory Bus Channels**
  - Intel Core2, Hyper-V, Windows VMs
  - Intel Xeon (Nehalem), Xen Linux VMs

So Far, So Good... And Then?

- **Addressing Uncertainty**
- **Scheduling Uncertainty**
- **Cache Physical Limitation**

Space Communication is Hard!
Reliable Communication

- The memory bus channel is quite noisy

![Access Latency Graph](image)

- Receiving Confirmation
  - No Explicit Send-and-Ack
  - Simultaneous receiver detection
    - Leveraging the system-wide observable bus contention

- Clock Synchronization
  - Self-clock coding – Differential Manchester Coding
    - Standard network coding used in token ring network

- Error Correction
  - Again, no Send-and-Ack
  - Forward Error Correction – Reed-Solomon Coding
    - Use for optical discs, satellite communication, etc.

Reliable Communication

- Data Framing
  - Send data in small fixed-length packets
  - Improves resilience to interruptions and errors

- Overview of Complete Protocol

![Data Framing Diagram](image)
Whispers in the Hyper-space

High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang

Project Outline

- Understand Existing Cache Covert Channel
  - Why it doesn’t work in the Cloud

- Design a New Covert Channel
  - Targeting cross-VM communication

- Realistic Evaluation
  - In-house and on Amazon EC2

In-house Experiment

- Realistic Setup
  - Intel Xeon (Nehalem), Xen, Linux Guests
  - Spawn two VMs, single vCPU, 512MB memory
  - Sender and receiver runs as unprivileged user programs

- Protocol Parameters
  - 32 bit data frame (8 bit preamble, 24 bit payload)
  - FEC Parity : Data = 4 : 4

- Channel Capacity
  - Bandwidth: 781.6 ± 13.6 bps
  - Error rate: 0.27%

Amazon EC2 Experiments

- Preparation
  - Spawn two physical co-resident VMs
    - /proc/cpuinfo: Intel Xeon (Yorkfield) (share memory bus)

- Initial Trial:
  - Protocol Parameters
    - 24 bit data frame (8 bit preamble, 16 bit payload)
    - FEC Parity : Data = 4 : 4
  - Best Channel Capacity
    - Bandwidth: 343.5 ± 66.1 bps
    - Error rate: 0.39%
    - But degrades over time in an interesting way

Amazon EC2 Experiments

- Three Staged Performance
  - Due to Xen scheduler preemption
Whispers in the Hyper-space

High-speed Covert Channel Attacks in the Cloud

Zhenyu Wu, Zhang Xu, and Haining Wang

Amazon EC2 Experiments

❖ Adapting to “Offensive” scheduling:
  ◦ Protocol Parameters
    ▪ Slow down transmission rate by 20%
    ▪ Increase FEC Parity: Data = 4 : 2
  ◦ Stable Channel Capacity
    ▪ Bandwidth: 107.9 ± 39.9 bps
    ▪ Error rate: 0.75%

Amazon EC2 Experiments

❖ Protocol Versatility
  ◦ Automatic speed adjustment according to channel quality

Mitigation Avenues

❖ Tenants
  ◦ Limited Options – Performance Anomaly Dentations
    ▪ Defender continuously monitors memory access latency
    ▪ High overhead, high false positives

❖ Cloud Providers
  ◦ Preventative Measures
    ▪ Dedicated Instance (Amazon EC2) – Effective but costly
    ▪ Alternative Low-Cost Placement Strategies
      ◦ Controlled and deterministic sharing
      ◦ Makes arbitrary attack difficult
Whispers in the Hyper-space

Mitigation Avenues

- Cloud Providers
  - Detective Measures
    - Leverage access to Hypervisor, and rich server resource
    - Discover performance anomalies by hardware counters
    - Low overhead, low false positive penalties
  - Hardware Manufactures
    - Isolation Improvements (Preventative Measures)
    - Trap instead of handling exotic conditions
    - Tag request by VM, and limit performance impact scope
    - Costly (one-time), but effective and efficient

Conclusion

- Covert Channel Attacks in the Cloud are Practical
  - We uncovered a high speed covert channel medium
  - We designed a reliable transmission protocol
  - We evaluated the performance in a real cloud environment

- Call for Effective and Efficient Mitigations
  - Three avenues of approaches:
    - Tenant
    - Cloud Provider
    - Hardware Manufacture

Q & A

Thank you for listening!