ReplayConfusion: Detecting Cache-based Covert Channel Attacks Using Record and Replay

Mengjia Yan, Yasser Shalabi, Josep Torrellas
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

MICRO October 2016
Motivation

- Cache-based covert channel attacks
  - Communicate through cache conflicts

- Serious security threat
  - Ubiquitous attack scenario: cloud
  - Bypass security policy; no trace left

- Existing solutions unable to detect all attacks

- Contribution: *ReplayConfusion*
  - High-coverage detection mechanism
Contribution: *ReplayConfusion*

**Observations:**
1. Trojan/Spy rely on specific mapping addresses→caches
2. Attack follows a repeating pattern when transmitting

**Change mapping of addresses→caches**

**Effects:**
1. Substantially disrupt cache miss pattern
2. Retain the repeating pattern

Re-mapping is different for each process

---

![Diagram of cache and mapping](image-url)
Outline

• Background
• Attack Protocols
• ReplayConfusion
  • Observations
  • Detection Framework
• Detection Example
• Summary
Cache-based Covert Channel Attack

- Basic cache organization:
  - Slice (i.e. Bank), Set, Way
  - Cache mapping function
- Approach: Prime+Probe

Can be reverse engineered

 trojan  spy

Slice 0    Slice 1

Set 0       Set 1

Prime: 0 1 1
Probe: 0 0 0

Time
Taxonomy of Attack Protocols

- **Time**
  - Round-Robin
  - Parallel

- **Space**
  - Single Group
  - Multiple Groups

**Attack Protocols**

- **Round-robin**
  - 0
  - 1

- **Parallel**
  - 0
  - 1

**I-ACOMA Group**

ReplayConfusion detects all the attacks
Observations

Observation 1: Trojan/spy rely on a specific cache mapping function

Observation 2: Attack follows a repeating pattern when transmitting
ReplayConfusion Detection Approach

Observations:
1. Trojan/Spy rely on a specific cache mapping function
2. Attack follows a repeating pattern when transmitting

Change mapping of addresses → caches
Re-mapping is different for each process

Effects:
1. Substantially disrupt cache miss pattern
2. Retain the repeating pattern
Replay Confusion Detection Approach

Record and Replay
- Existing mature technique
  - e.g. Capo, Cyrus ...

Design new HW mapping addresses → caches
- Requirements:
  - Small impact on benign programs
  - Big impact on attacks

Analyze cache miss rate timelines
- Look for a repeating pattern in the timeline of the cache miss rate difference
Designing New Cache Mapping Functions

- Goal
  - Small impact on benign programs
  - Big impact on attacks

- Set Index Function
  - Swap or flip bits within index field
Designing New Cache Mapping Functions

- **Goal**
  - Small impact on benign programs
  - Big impact on attacks

- **Set Index Function**
  - Swap or flip bits within index field

- **Slice Selection Function**
  - Replace the bits in the function with nearby ones
Analyzing Cache Miss Rate Timelines

- Compute timeline of the **difference** in cache miss rates
  - (Recording miss rate timeline) – (Replay miss rate timeline)

<table>
<thead>
<tr>
<th></th>
<th>Benign programs</th>
<th>Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diff Value</td>
<td>Small values mostly</td>
<td>Large values when transmitting</td>
</tr>
<tr>
<td>Diff Pattern</td>
<td>No pattern</td>
<td>Repeating pattern</td>
</tr>
</tbody>
</table>

- Use auto-correlation* to detect repeating pattern in the timeline of the cache miss rate **difference**
  - Look for a fluctuating pattern in the auto-correlation

*A statistical technique that discovers repeating patterns in a signal.
## Detection Example

<table>
<thead>
<tr>
<th></th>
<th>Experiment #1: Bzip2 (co-run with h264ref)</th>
<th>Experiment #2: Spy in attack (co-run with Trojan)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss Rate Timeline in</td>
<td><img src="image1.png" alt="Graph" /></td>
<td><img src="image2.png" alt="Graph" /></td>
</tr>
<tr>
<td>Record</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miss Rate Timeline in</td>
<td><img src="image3.png" alt="Graph" /></td>
<td><img src="image4.png" alt="Graph" /></td>
</tr>
<tr>
<td>Replay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miss Rate Difference</td>
<td><img src="image5.png" alt="Graph" /></td>
<td><img src="image6.png" alt="Graph" /></td>
</tr>
<tr>
<td>Difference Timeline</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Detection Example

<table>
<thead>
<tr>
<th>Experiment #1: Bzip2 (co-run with h264ref)</th>
<th>Experiment #2: Spy in attack (co-run with Trojan)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Miss Rate</strong></td>
<td><strong>Timeline</strong></td>
</tr>
<tr>
<td><strong>Timeline</strong></td>
<td><strong>Difference</strong></td>
</tr>
<tr>
<td><strong>Difference</strong></td>
<td><strong>Auto-correlation</strong></td>
</tr>
</tbody>
</table>

### Experiment #1: Bzip2 (co-run with h264ref)
- Miss Rate: [Graph](image1)
- Timeline: [Graph](image2)
- Difference: [Graph](image3)
- Auto-correlation: [Graph](image4)

### Experiment #2: Spy in attack (co-run with Trojan)
- Auto-correlation: [Graph](image5)
More in the Paper

- Details on the taxonomy of cache-based covert channel attacks
- More detection results
  - Attacks using different protocols
  - Attacks with background noise
  - Attacks with small group size
  - More benign programs
- Detailed discussion about robustness of ReplayConfusion
- Discussion of related works
Conclusion

• Characteristics of cache-based covert channel attacks:
  • Trojan/spy communication is tuned to mapping of addresses to caches
  • Miss rate pattern repeats when transmitting bits

• ReplayConfusion
  • Use RnR to execute the same program on machines with different mappings of addresses to caches in replay
  • Compute the timeline of the miss rate difference between record and replay
  • Detect repeating patterns → detect attack
ReplayConfusion: Detecting Cache-based Covert Channel Attacks Using Record and Replay

Mengjia Yan, Yasser Shalabi, Josep Torrellas
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

MICRO October 2016
Evaluation Result
Benign Programs

Autocorrelation vs. Lag

h264ref
sjeng
gobmk
stream
Experiment Setup

- System: Ubuntu 10.4 with 4GB memory
- 4 in-order core, 32KB private L1 cache, 2MB shared L2 cache
- L2: 8-way associative, 4 slices, 64B/block

<table>
<thead>
<tr>
<th>$f_{set}$</th>
<th>$(pa/64)%1024$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{sli}$</td>
<td>$\text{bit}_0: p18 \oplus p19 \oplus p21 \oplus p23 \oplus p25 \oplus p27 \oplus p29 \oplus p30 \oplus p31$</td>
</tr>
<tr>
<td></td>
<td>$\text{bit}_1: p17 \oplus p19 \oplus p20 \oplus p21 \oplus p22 \oplus p23 \oplus p24 \oplus p26 \oplus p28 \oplus p29 \oplus p31$</td>
</tr>
</tbody>
</table>

$F_{def}$

$F_{new}$ for replay

<table>
<thead>
<tr>
<th>$f_{set}$</th>
<th>Take $f_{set}$ from $F_{def}$ and swap the most significant and least significant 5 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{sli}$</td>
<td>$\text{bit}_0: p17 \oplus p19 \oplus p20 \oplus p22 \oplus p24 \oplus p26 \oplus p28 \oplus p30 \oplus p31$</td>
</tr>
<tr>
<td></td>
<td>$\text{bit}_1: p18 \oplus p20 \oplus p21 \oplus p22 \oplus p23 \oplus p24 \oplus p25 \oplus p27 \oplus p29 \oplus p30$</td>
</tr>
</tbody>
</table>
Example

(a) Cache miss rate timeline

(b) Cache miss rate autocorrelogram

(c) Cache miss rate difference timeline

(d) Cache miss rate difference autocorrelogram
Evaluation Result
Attacks using parallel protocols

- 2-group
- 1-group, ¼ cache, set-based
- 1-group, ¼ cache, slice-based
- 1-group, unaware
Related Work

• Defense
  • Cache Partition
  • Add noise to timer
  Either not not applicable or too much overhead
  Not work effectively

• Detection
  • Hexpad: high cache access rate
  • Chiappetta et al.: correlation between sender and receiver
  • CC-Hunter: detect alternate pattern of conflicts
  Unable to detect advanced attacks
  May have high false positives
  Only effective to attacks using a specific type of protocols
Operations of ReplayConfusion

SW

- Log
- RnR Module
- Cache Configuration Manager
- Cache Profile Manager
- Log

HW

- Cache Address Computation Unit
- Address Mapping
- Slice ID
- Set Index
- Tag
- Memory Address
- PMU

PMU

F0

F1

Fn

Fsel