SecDir: A Secure Directory to Defeat Directory Side-Channel Attacks

Mengjia Yan*, Jen-Yang Wen, Christopher W. Fletcher, Josep Torrellas
University of Illinois at Urbana-Champaign
*University of Illinois at Urbana-Champaign/MIT

ISCA, June 2019
Motivation

- Cache-based side-channel attacks are serious security threats
- Directories are also vulnerable to side-channel attacks [Yan et al, S&P’19]
- It is challenging to design secure directories *inexpensively* and *scalably*
Contribution: A Secure Directory (SecDir)

- Key: Block directory interference between processes
- Main idea: Take a portion of the storage used by conventional directory and re-assign it to per-core private directory (Victim Directory)
Outline

- Background
- The Problem
- Threat Model
- SecDir Design
- Evaluation
Directory Basics

- Directory is used to keep presence information for cache lines
- A directory entry contains “sharer information”, address tag, coherence state
  - Sharer information: N presence bits, where N is # of cores in machine
- Directory is partitioned into slices like LLC using a hash function
Directories in Non-inclusive Cache Hierarchies  [Yan et al, S&P’19]

- Trend to have non-inclusive cache hierarchies
- Added **Extended Directory** to hold state for lines that are in private caches (L2)

---

![Diagram showing Extended Directory (ED) and Traditional Directory (TD) with slices of Intel Skylake-X/SP LLC and directory](image)
Directories are Vulnerable to Side-Channel Attacks  [Yan et al, S&P’19]

- Every single line in the cache hierarchy has a directory entry
- Directory conflict $\rightarrow$ Evicts victim’s directory entry $\rightarrow$ Evicts victim’s cache line
- **Root cause:** Limited per-slice directory associativity
**Defense Goal & Threat Model**

**Goal:** A secure directory to block directory interference between processes

| Attack Strategy | Co-location |  |
|-----------------|-------------|
|                 | Same-core   | Cross-core |
| Active          |             | X           |
| Passive         |             |             |

* Victim self-conflicts (e.g. in victim’s private structures) are not considered leakage*
Naïve Secure Directory Designs Are Not Scalable

- Strategy I: Substantially increase associativity of each directory slice
  - Unrealistic: Need too high associativity  (e.g. > 300 for a 22-core machine)
Naïve Secure Directory Designs Are Not Scalable

- **Strategy I:** Substantially increase associativity of each directory slice
  - Unrealistic: Need too high associativity (e.g. > 300 for a 22-core machine)

- **Strategy II:** Way-partition the directory slice (at least 1 way per security domain)
  - Unacceptable: Inflexible, low performance and limiting
Our proposal: **SecDir**

Main idea: Take part of the storage used by conventional directory and re-assign it to per-core private directories: **Victim Directories (VD)**

Slice of Intel Skylake-X directory.

Slice of SecDir.

Provide per-core isolation
Our proposal: **SecDir**

- Provides inexpensive and **scalable** isolation
- Uses modest storage

VD bank size = $\frac{1}{N} \times \text{L2 size}$

Total VD per core = $S \times \frac{1}{N} \times \text{L2 size}$

= L2 size

VD size for a core is **constant** irrespective to $N$
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
  - ED↔TD: Line location does not change; no leakage
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
  - ED↔TD: Line location does not change; *no leakage*
  - ② TD → Memory: Line is in LLC but in no L2; It is because of *L2 self-conflicts, not due to attacker*
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
  - ED ↔ TD: Line location does not change; no leakage
  - ② TD → Memory: Line is in LLC but in no L2; It is because of L2 self-conflicts, not due to attacker
  - ③ TD → VD: Line location does not change. VD of every sharer receives a copy. no leakage

![Diagram of SecDir Blocks Directory Interference](image)
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
  - ED ↔ TD: Line location does not change; *no leakage*
  - ② TD → Memory: Line is in LLC but in no L2; It is because of *L2 self-conflicts, not due to attacker*

SecDir prevents cache line evictions due to attacker induced directory interference
SecDir Optimizations

• Provides high associativity in VD
  – *VD supports Cuckoo hashing to increase effective VD associativity*
• Delivers efficient directory lookup
  – *Uses a “Early-Miss” (EM) bit* → skips many VD lookups
Experimental Setup and Benchmarks

• Configurations: two 8-core designs
  – Baseline: Use Skylake-X directory (ED associativity=12)
  – SecDir: Take 4 ways from the ED to create the VD
    • Remaining ED is as big as L2
    • Augment VD in each slice with 28.5KB \(\rightarrow\) per-core VD is as big as L2

• Benchmarks:
  – SPEC Mixes: Groups of programs running 8 threads, with different characteristics
  – PARSEC: Individual parallel programs running with 8 threads
Evaluation Results – PARSEC

- ED/TD conflicts migrate entries to VD without evicting L2 lines $\rightarrow$ fewer L2 misses
Evaluation Results – PARSEC

- Under benign conditions, the performance overhead is negligible
  + Fewer L2 misses
  - VD accesses add 5-10 cycles

Summary: Secure and little performance impact
More in the paper & Discussion

• More performance results for SPECMIX
• Security discussion
  – VD timing issues
• Performance evaluation
  – Effects of the two optimizations: cuckoo hashing and Early-Miss bits
  – Storage and area overhead
Conclusion

• Directories are vulnerable to side-channel attacks [Yan et al, S&P’19]
• Naïve solutions are not effective
• Contribution: **SecDir**
  – Main idea: Take a portion of the storage used by conventional directory and re-assign it to per-core private directory (Victim Directory)
  – Provides isolation inexpensively and scalably
  – Uses moderate storage
Q&A
SecDir Blocks Directory Interference

- Consider each directory transition and its security implications
  - ED ↔ TD: Line location does not change; *no leakage*
  - ② TD → Memory: Line is in LLC but in no L2; *L2 self-conflicts*
  - ③ TD → VD: Line location does not change. VD of every sharer receives a copy. *no leakage*
  - ④ VD → TD: L2 wants to write back the cache line to LLC; *L2 self-conflict*
Evaluation of SPECMIX

- Under benign conditions, the performance overhead is negligible
  - ED/TD conflicts migrate entries to VD: do not evict L2 lines $\rightarrow$ fewer L2 misses
  - VD accesses add 5-10 cycles

Summary: Secure and little performance impact
Evaluation of SPECMIX

- SecDir has fewer L2 misses because fewer directory conflicts
- No VD hits (since no shared data) → VD accesses add to a DRAM latency
Minimizing VD Self-Conflicts

- Organize VD as Cuckoo Directory
- Performance: Longer lookup/insert latency
- Security:
  - Reduce VD self-conflicts
  - Obscures victim self-conflict patterns

VD bank

\[ \begin{array}{cc}
1 & a \\
2 & c \\
3 & e \\
4 & g \\
\end{array} \]

\[ h_1(x) \]
\[ h_2(x) \]

SecDir – ISCA’19
Example: VD Offers High Associativity

- Example: insert x into an almost full VD

(a) Before inserting item x
(b) After item x inserted
Early Detection of VD Misses

- Under benign conditions: VD will be highly underutilized
- Want to quickly detect when a VD access will miss → save E
  - Add an Empty Bit (EB) per set and bank
  - If all the entries in that set of that bank are Invalid → EB is set
SecDir Uses Low Area

- VD does not store “sharing information”
- More cores \(\rightarrow\) More bits of sharing information “saved”

**Baseline:** Skylake-X directory (W\(_{ED}\)=12).

**SecDir:** Take some ED ways for VD. For example, keep W\(_{ED}\)=8 (such that ED can hold as many lines as L2).

Summary: by stealing 4 ways of ED, we quickly attain a per-core VD that has as many entries as L2 lines.

Comparing the number of per-core VD entries machine-wide to the number of L2 lines. Values above 1 mean that the per-core VD has more entries than lines in L2.
Directories are Vulnerable to Attacks

- As the victim re-accesses the data → directory entry reloaded
- Attacker can observe the directory changing

![Diagram showing the vulnerability of directories to attacks](image)

[Yan S&P’19]
SecDir Properties

- Provides inexpensive and scalable isolation
- Provides high associativity
- Uses low storage
- Delivers efficient directory lookup
Benchmarks

- **SPEC Mixes**
  Profile applications on baseline to classify them into
  CCF (core cache fit); LLCF (LLC fit); LLCT (LLC

<table>
<thead>
<tr>
<th>Categories</th>
<th>Name</th>
<th>Applications</th>
<th>Name</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCF, CCF</td>
<td>mix0</td>
<td>gobmk, sjeng</td>
<td>mix1</td>
<td>hmer, gamess</td>
</tr>
<tr>
<td>LLCF, LLCF</td>
<td>mix2</td>
<td>bzip2, omnetpp</td>
<td>mix3</td>
<td>gromacs, zeusmp</td>
</tr>
<tr>
<td>LLCT, LLCT</td>
<td>mix4</td>
<td>libquantum, lbm</td>
<td>mix5</td>
<td>bwaves, sphinx3</td>
</tr>
<tr>
<td>CCF, LLCF</td>
<td>mix6</td>
<td>sjeng, omnetpp</td>
<td>mix7</td>
<td>h264ref, zeusmp</td>
</tr>
<tr>
<td>CCF, LLCT</td>
<td>mix8</td>
<td>gobmk, libquantum</td>
<td>mix9</td>
<td>namd, bwaves</td>
</tr>
<tr>
<td>LLCF, LLCT</td>
<td>mix10</td>
<td>omnetpp, bwaves</td>
<td>mix11</td>
<td>zeusmp, lbm</td>
</tr>
</tbody>
</table>

Table 5: SPEC workload mixes.

- **PARSEC**