Pin Tumbler Lock: A Shift based Encryption Mechanism for Racetrack Memory

Hongbin Zhang, Chao Zhang, Xian Zhang, Guangyu Sun, Jiwu Shu

ASP-DAC2016
January 26, 2016
Executive Summary

Problem: Data security problem of NVM
- Data retain in the NVM after power off or stolen
- Traditional methods induce non-trivial timing overhead

Observation: Racetrack memory read and write data through shift operation, which can be used as encrypt and decrypt schematic.
- Shift operation make data encrypted, only right key can restore the data (like pin tumbler lock)

Key Ideas:
- (1) Data is encrypted by shift operation according to shift key
- (2) Keys are generated from randomizer and stored in DRAM
- (3) Encryption shift operation merger into R/W shift operation

Results: PTL get the same security strength of AES-128 with 3.1% performance overhead and 3.7% energy overhead and 1.56% storage cost and 1.6% area cost.
Outline

- Executive Summary
- Background
  - Racetrack memory
  - Problem
  - Existing proposals
  - Our proposal
- Design
- Evaluation
- Conclusion
Racetrack Memory

- Ultra-high density
- Comparable R/W latency
- Used as cache or memory
- Facing also security problems

Cell structure

Racetrack
Problem

- Data security
  - Data retain in NVM after power off
  - Data can be easy inspected after stolen

- Traditional methods
  - Existing methods try to encrypt data with software
  - Traditional methods can not protect data totally
  - Traditional methods induce non-trivial timing overhead

- How to better protect the data in racetrack memory?
Existing Proposals

- AES encryption: use AES to encrypt data before being stored in memory [DSN’ 2010]
  - Solution: protect the data in main memory using AES algorithm
  - Problem: inducing large timing overhead

- i-NVMM: encrypt main memory incrementally [ISCA’ 2011]
  - Solution: encrypt the working set of application dynamically
  - Problem: sensitive information in working set is not protected

- PAD-XOR: provide run-time protection through encrypting the main memory using sub-PAD [ICCAS’ 2013]
  - Solution: run-time protection to all data using sub-pad
  - Problem: introduces extra sub-PAD tables for encryption
Our Proposal - PTL

- Encrypt the data using shift operation
- Only right key can decrypt the data
- Shift keys are stored in volatile memory
- Keys disappear when power off and data protected

a) With no key  b) With wrong key  c) With right key  d) Unlocked

Outline

- Executive Summary
- Background
- Design
  - Shift based encryption scheme
  - Key width and security strength
  - Redundant Domain Wall
  - Random Number Generator
  - System structure
- Evaluation
- Conclusion
Shift based encryption scheme

- Data is placed vertically
- Shift operation of racetrack encrypt the data

(a) Before encryption. (b) After encryption.
Key width and security strength

- Longer key brings higher security
- Longer key induces larger key storage cost

(a) Region with 128bit shift key (key-width is 1).
(b) Region with 256bit shift key (key-width is 2).
(c) Region with 384bit shift key (key-width is 3).
Redundant Domain Wall

- Longer key brings higher security
- Longer key induces larger area cost

For the region with 128 bit key, 1 more stripe redundant bits.
For the region with 256 bit key, 3 more stripe redundant bits.
For the region with 384 bit key, 7 more stripe redundant bits.
Random Number Generator

- Using 4-stage FN to transform a data into a pseudo random number.
- In order to avoid the data with strong patterns being easily decrypted.
Structure of secure racetrack memory

Blue: read operation sequence.
Green: write operation sequence.
Outline

- Executive Summary
- Background
- Design
- Evaluation
  - Performance evaluation on methods
  - Performance evaluation on key-width
  - Energy evaluation
  - Storage cost
  - Area cost
- Conclusion
Evaluation Setup

- PTL based on CentOS
  - Compared with no-ENC, AES-ENC, Rand-Pad
- Platform
  - 4-core CPU, 32KB L1, 1MB L2, 128MB racetrack L3 cache
- Workloads
  - 13 workloads from Parsec3 benchmarks
- Metrics
  - Performance: R/W latency
  - Energy: read, write, shift and static energy
  - Storage: shift key size
  - Area: redundant domain wall size
Performance evaluation on methods

![Normalized Performance Overhead](chart)

- None Enc.
- AES-ENC
- RandPad
- PTL-ENC

Normalized Performance

- blackscholes
- bodytrack
- canneal
- dedup
- facesim
- ferret
- fluidanimate
- freqmine
- rtview
- streamcluster
- swaptions
- vips
- x264
Performance evaluation on key-width

![Performance Overhead of Different Key Width (%)](image)

- Blackscholes
- Bodytrack
- Canneal
- Dedup
- Facesim
- Ferret
- Fluidanimate
- Freqmine
- Rtview
- Streamcluster
- Swaptions
- Vips
- X264
Energy overhead evaluation

![Energy Overhead Evaluation Graph](image)

- None Enc
- AES-ENC
- RandPad
- PTL-ENC

Normalized Energy Overhead [%]

- blackscholes
- bodytrack
- canneal
- dedup
- facesim
- ferret
- fluidanimate
- freqmine
- rtview
- streamcluster
- swaptions
- vips
- x264
### Storage and area cost

- For 128MB cache

<table>
<thead>
<tr>
<th>Region Size</th>
<th>Key Num.</th>
<th>Key width</th>
<th>Key length</th>
<th>Key Storage</th>
<th>Storage Cost(%)</th>
<th>Redun. Area</th>
<th>Area Cost(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1KB</td>
<td>128K</td>
<td>1</td>
<td>128</td>
<td>2MB</td>
<td>1.56</td>
<td>2MB</td>
<td>1.6</td>
</tr>
<tr>
<td>2KB</td>
<td>64K</td>
<td>1</td>
<td>256</td>
<td>2MB</td>
<td>1.56</td>
<td>2MB</td>
<td>1.6</td>
</tr>
<tr>
<td>4KB</td>
<td>32K</td>
<td>1</td>
<td>512</td>
<td>2MB</td>
<td>1.56</td>
<td>2MB</td>
<td>1.6</td>
</tr>
<tr>
<td>1KB</td>
<td>128K</td>
<td>2</td>
<td>256</td>
<td>4MB</td>
<td>3.12</td>
<td>6MB</td>
<td>4.7</td>
</tr>
<tr>
<td>2KB</td>
<td>64K</td>
<td>2</td>
<td>512</td>
<td>4MB</td>
<td>3.12</td>
<td>6MB</td>
<td>4.7</td>
</tr>
<tr>
<td>4KB</td>
<td>32K</td>
<td>2</td>
<td>1024</td>
<td>4MB</td>
<td>3.12</td>
<td>6MB</td>
<td>4.7</td>
</tr>
<tr>
<td>1KB</td>
<td>128K</td>
<td>3</td>
<td>384</td>
<td>6MB</td>
<td>4.68</td>
<td>14MB</td>
<td>10.9</td>
</tr>
<tr>
<td>2KB</td>
<td>64K</td>
<td>3</td>
<td>768</td>
<td>6MB</td>
<td>4.68</td>
<td>14MB</td>
<td>10.9</td>
</tr>
<tr>
<td>4KB</td>
<td>32K</td>
<td>3</td>
<td>1536</td>
<td>6MB</td>
<td>4.68</td>
<td>14MB</td>
<td>10.9</td>
</tr>
</tbody>
</table>
### Storage and area cost

- For 4GB main memory

<table>
<thead>
<tr>
<th>Region Size</th>
<th>Key Num.</th>
<th>Key width</th>
<th>Key length</th>
<th>Key Storage</th>
<th>Storage Cost(%)</th>
<th>Redun. Area</th>
<th>Area Cost(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1KB</td>
<td>4M</td>
<td>1</td>
<td>128</td>
<td>64MB</td>
<td>1.56</td>
<td>64MB</td>
<td>1.6</td>
</tr>
<tr>
<td>2KB</td>
<td>2M</td>
<td>1</td>
<td>256</td>
<td>64MB</td>
<td>1.56</td>
<td>64MB</td>
<td>1.6</td>
</tr>
<tr>
<td>4KB</td>
<td>1M</td>
<td>1</td>
<td>512</td>
<td>64MB</td>
<td>1.56</td>
<td>64MB</td>
<td>1.6</td>
</tr>
<tr>
<td>1KB</td>
<td>4M</td>
<td>2</td>
<td>256</td>
<td>128MB</td>
<td>3.12</td>
<td>192MB</td>
<td>4.7</td>
</tr>
<tr>
<td>2KB</td>
<td>2M</td>
<td>2</td>
<td>512</td>
<td>128MB</td>
<td>3.12</td>
<td>192MB</td>
<td>4.7</td>
</tr>
<tr>
<td>4KB</td>
<td>1M</td>
<td>2</td>
<td>1024</td>
<td>128MB</td>
<td>3.12</td>
<td>192MB</td>
<td>4.7</td>
</tr>
<tr>
<td>1KB</td>
<td>4M</td>
<td>3</td>
<td>384</td>
<td>192MB</td>
<td>4.68</td>
<td>448MB</td>
<td>10.9</td>
</tr>
<tr>
<td>2KB</td>
<td>2M</td>
<td>3</td>
<td>768</td>
<td>192MB</td>
<td>4.68</td>
<td>448MB</td>
<td>10.9</td>
</tr>
<tr>
<td>4KB</td>
<td>1M</td>
<td>3</td>
<td>1536</td>
<td>192MB</td>
<td>4.68</td>
<td>448MB</td>
<td>10.9</td>
</tr>
</tbody>
</table>
Conclusion

- Security is one of the problems of NVM
  - Data retained in NVM when power off
  - Difficult to provide run time protecting
- Shift based PTL schematic solved this problem
  - Achieving the same security strength as AES
  - With less performance cost and energy consuming
  - With less storage and area cost
- PTL achieves the same security strength of AES-128 with 3.1% performance overhead and 3.7% energy overhead and 1.56% storage cost and 1.6% area cost.
Major contribution

- The first work that leverages the RM structure and shifting operations for NVM data encryption.
- Present a scheme achieving the same or higher security strength as prior works using AES.
- Our encryption mechanism is compatible with RM design for different levels of a memory hierarchy.
- Our work achieves less performance and energy overhead than existing approaches.
Pin Tumbler Lock: A Shift based Encryption Mechanism for Racetrack Memory

Hongbin Zhang, Chao Zhang, Xian Zhang, Guangyu Sun, Jiwu Shu