Seeds of SEED: **R-SAW**: New Side Channels Exploiting Read Asymmetry in MLC Phase Change Memories

Md Hafizul Islam Chowdhuryy[1], Ricard Ewetz[1], Amro Awad[2], and Fan Yao[1]

[1]Department of ECE
University of Central Florida
Florida

[2]Department of ECE
North Carolina State University
North Carolina

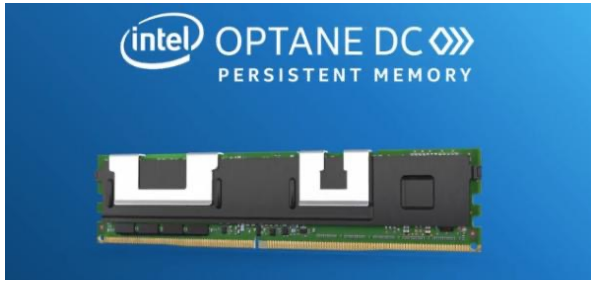
International Symposium on Secure and Private Execution Environment Design (SEED)
September 20 - 21, 2021

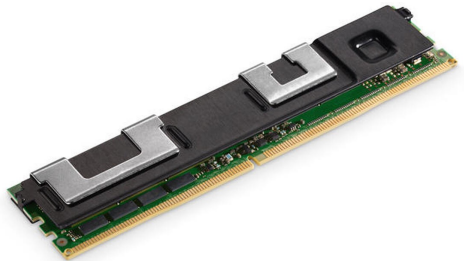
Microarchitecture Security

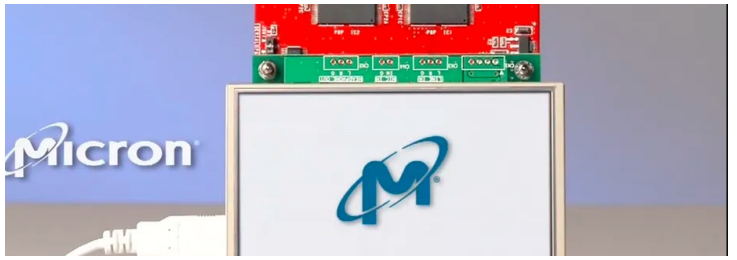
- Information security in hardware is a major concern.
  - Many microarchitectural components can be sources of leakage

Microarchitecture Security

- Information security in hardware is a major concern.
  - Many microarchitectural components can be sources of leakage
- **Emerging technologies in memory subsystems (Non-volatile memory).**
  - PCM is the major contender for future main memory
  - Prior works focus on data integrity and remanence issues
  - No prior studies of µarch timing channels in PCM

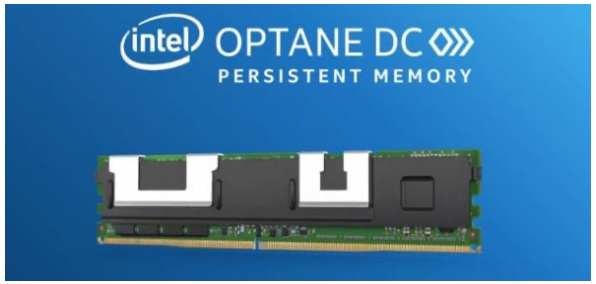


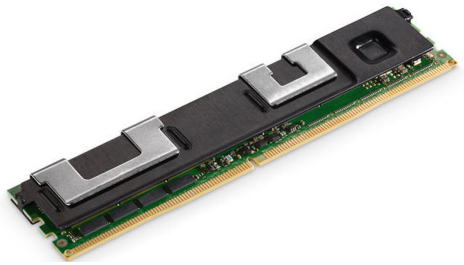


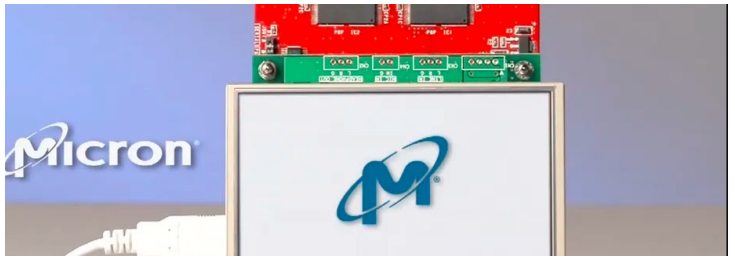


Microarchitecture Security

- Information security in hardware is a major concern.
  - Many microarchitectural components can be sources of leakage
- **Emerging technologies in memory subsystems (Non-volatile memory).**
  - PCM is the major contender for future main memory
  - Prior works focus on data integrity and remanence issues
  - No prior studies of µarch timing channels in PCM







Lessons from past: Security needs to be understood by design, instead of an afterthought.

Microarchitecture Security

- Information security in hardware is a major concern.
  - Many microarchitectural components can be sources of leakage
- **Emerging technologies in memory subsystems (Non-volatile memory).**
  - PCM is the major contender for future main memory
  - P
  - N

This work

First investigation of information leakage vulnerabilities in Multi-level Cell PCM

Lessons from past: Security needs to be understood by design, instead of an afterthought.

Background: MLC PCM

❖ PCM cells have wide range of resistance.

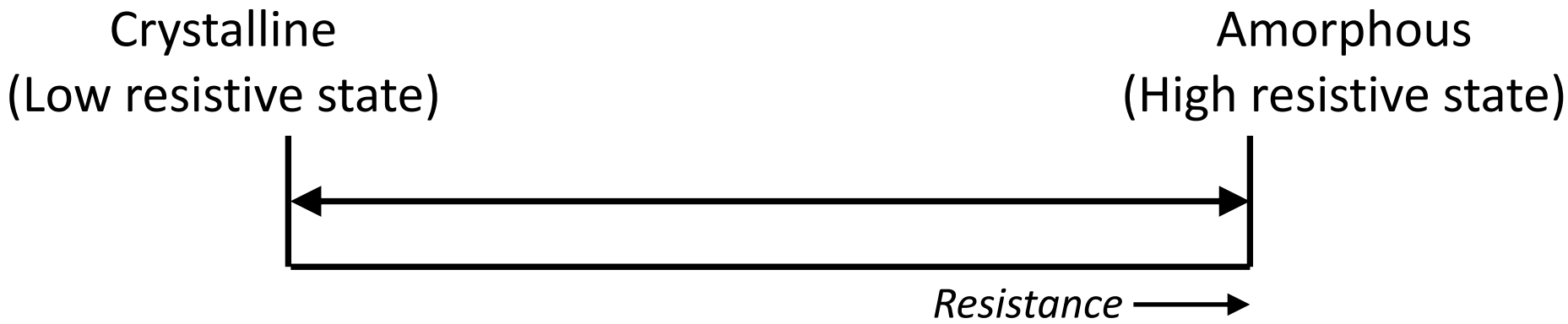


Figure: PCM cell resistance range

Background: MLC PCM

- PCM cells have wide range of resistance.
- **Single-level cell mode (SLC):** Each cell stores one bit.

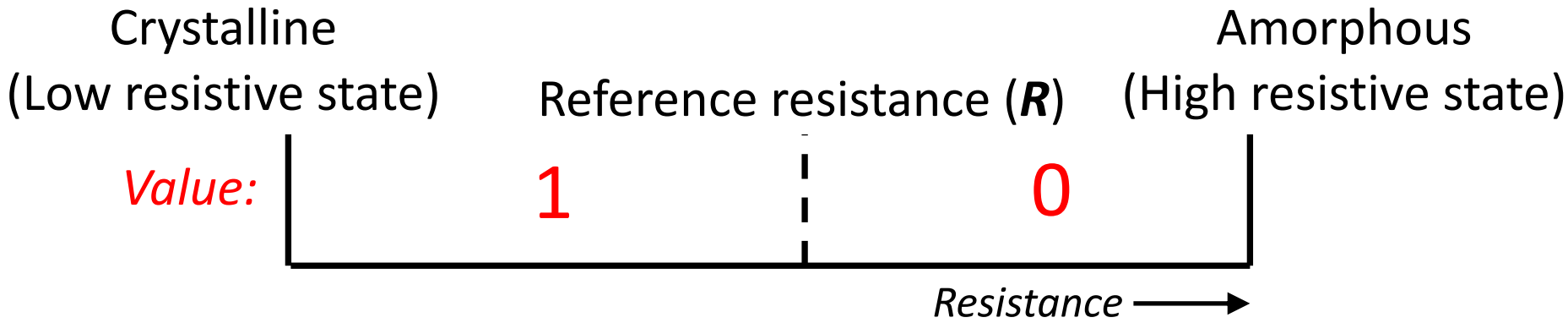


Figure: SLC PCM

Background: MLC PCM

- PCM cells have wide range of resistance.
- **Single-level cell mode (SLC):** Each cell stores one bit.
- **Multi-level cell mode (MLC):** Each cell stores two (or more) bits.

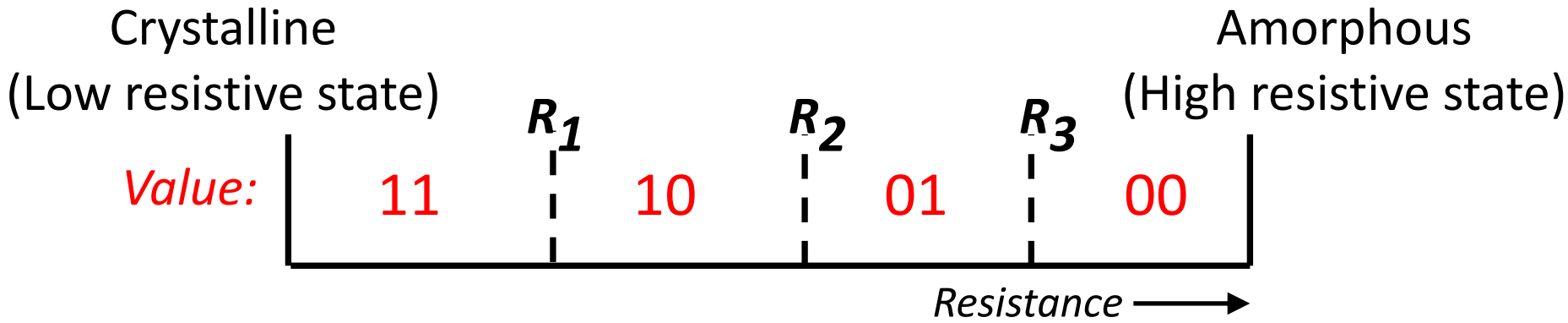
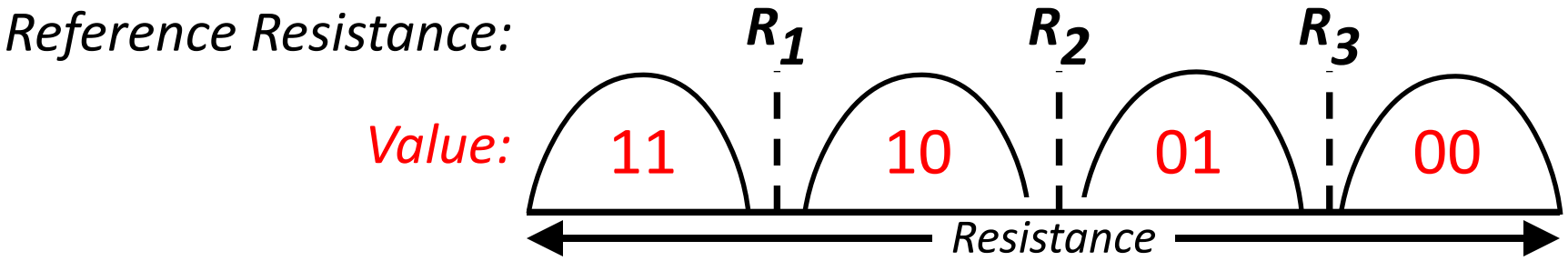
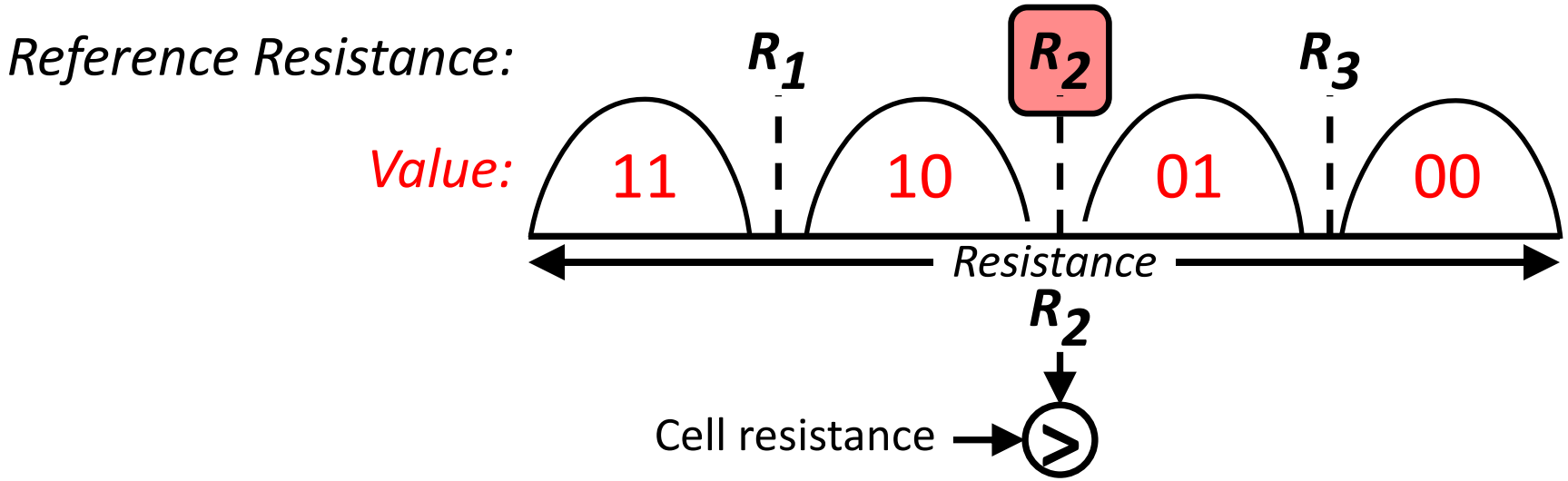


Figure: 2-bit MLC PCM

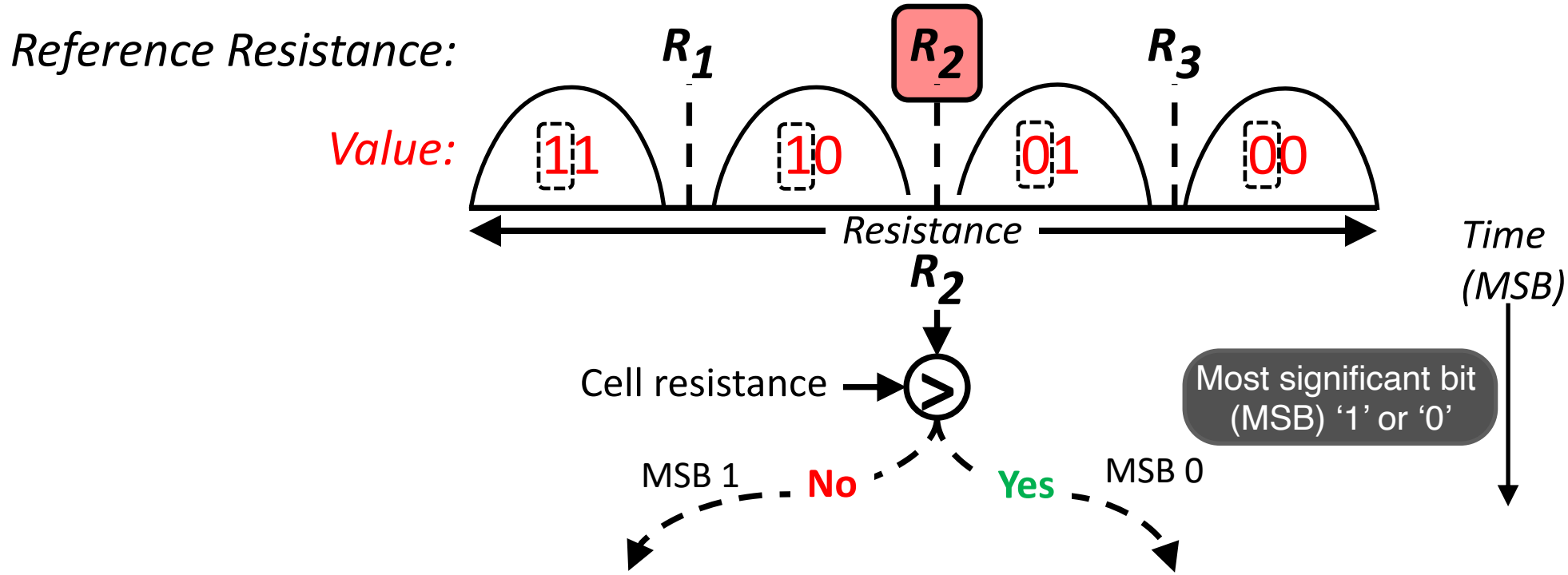
MLC PCM Read Techniques



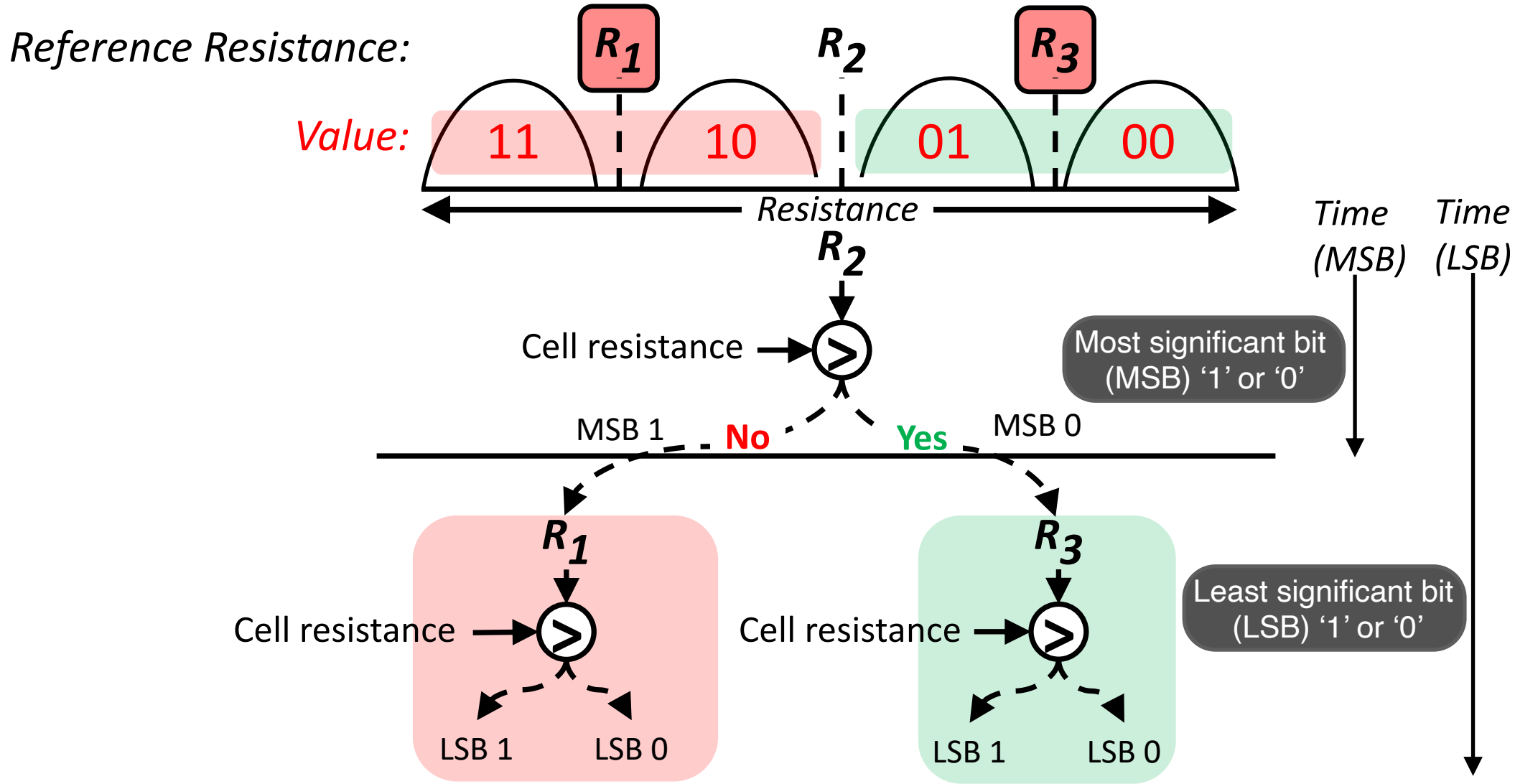
MLC PCM Read Techniques



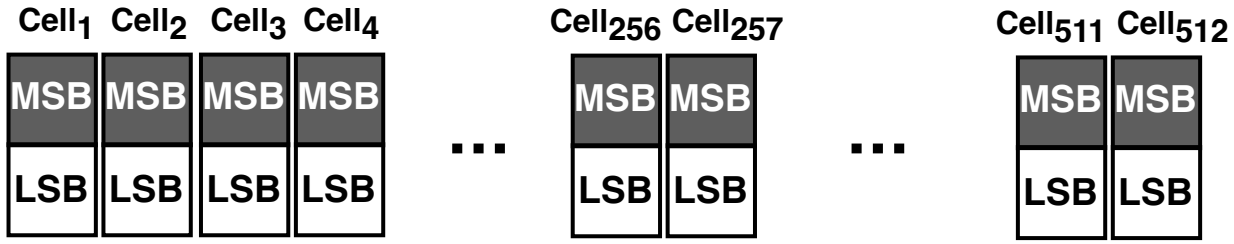
MLC PCM Read Techniques



MLC PCM Read Techniques

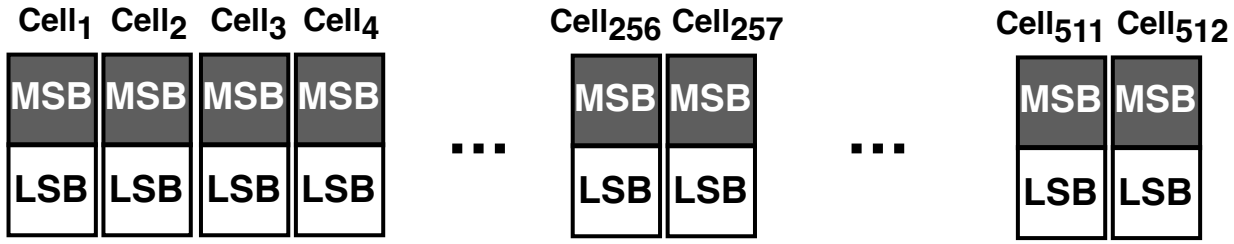


Performance Optimizations for MLC-PCM Reads



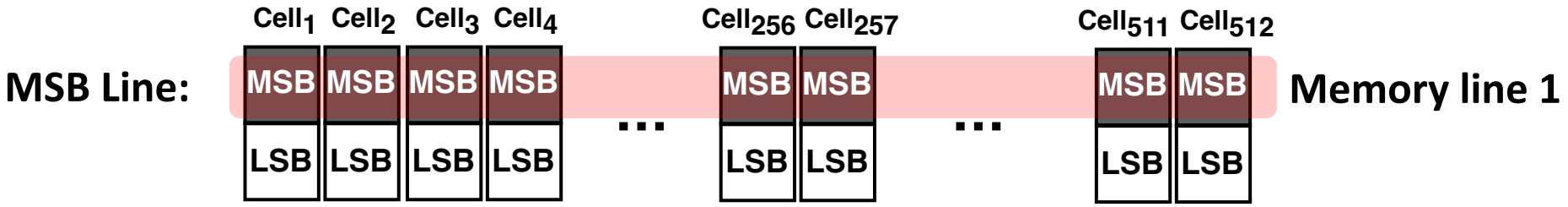
- Decouple MSB bit reads from LSB bit reads.

Performance Optimizations for MLC-PCM Reads



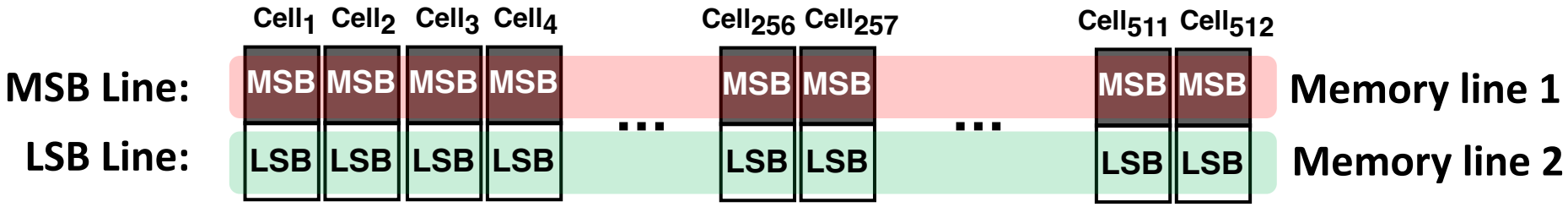
- Decouple MSB bit reads from LSB bit reads.
- **Inter-line striping:** Stripe *alternating lines* in different speed grades.

Performance Optimizations for MLC-PCM Reads



- Decouple MSB bit reads from LSB bit reads.
- **Inter-line striping:** Stripe *alternating lines* in different speed grades.
  - **MSB Lines:** Memory lines containing ***all*** MSB bits (faster)

Performance Optimizations for MLC-PCM Reads



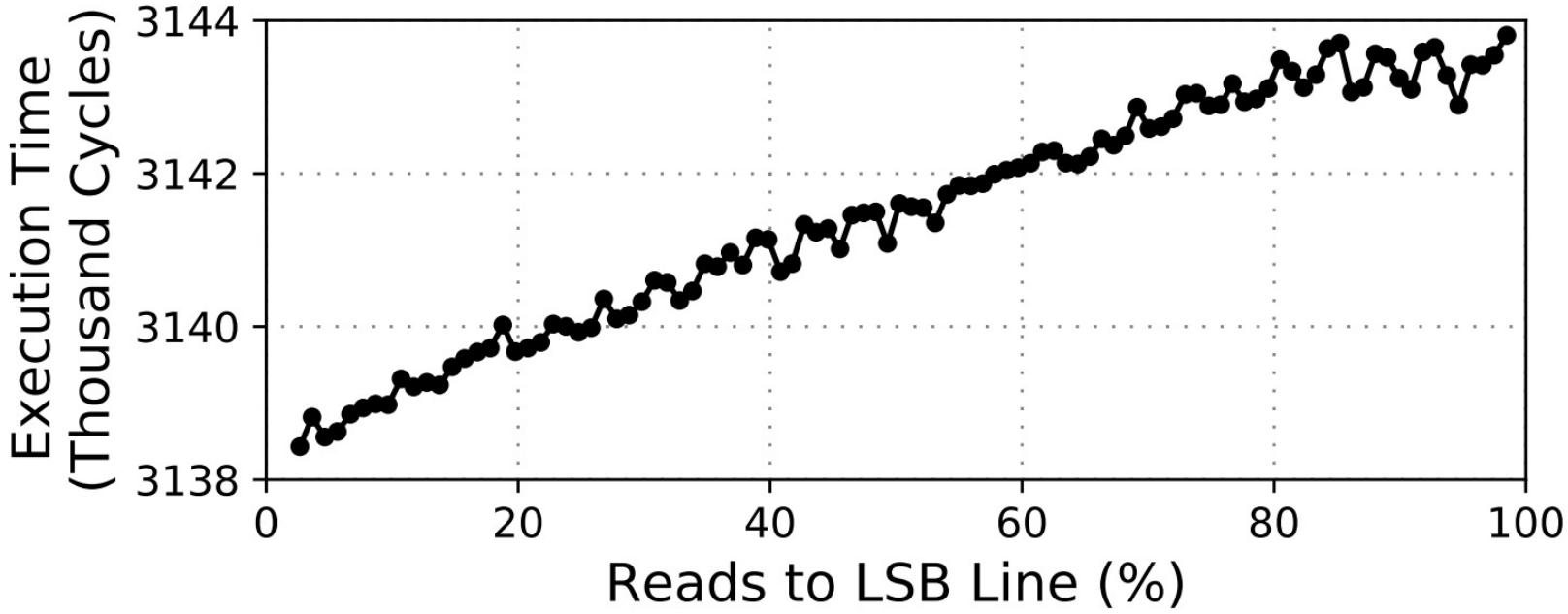
- Decouple MSB bit reads from LSB bit reads.
- **Inter-line striping:** Stripe *alternating lines* in different speed grades.
  - **MSB Lines:** Memory lines containing ***all*** MSB bits (faster)
  - **LSB Lines:** Memory lines containing ***all*** LSB bits (slower)

Are PCM Read Techniques Vulnerable to Leakage?

- **Microbenchmark:**
  - Issues fixed number of memory accesses
  - Varies MSB and LSB lines accesses

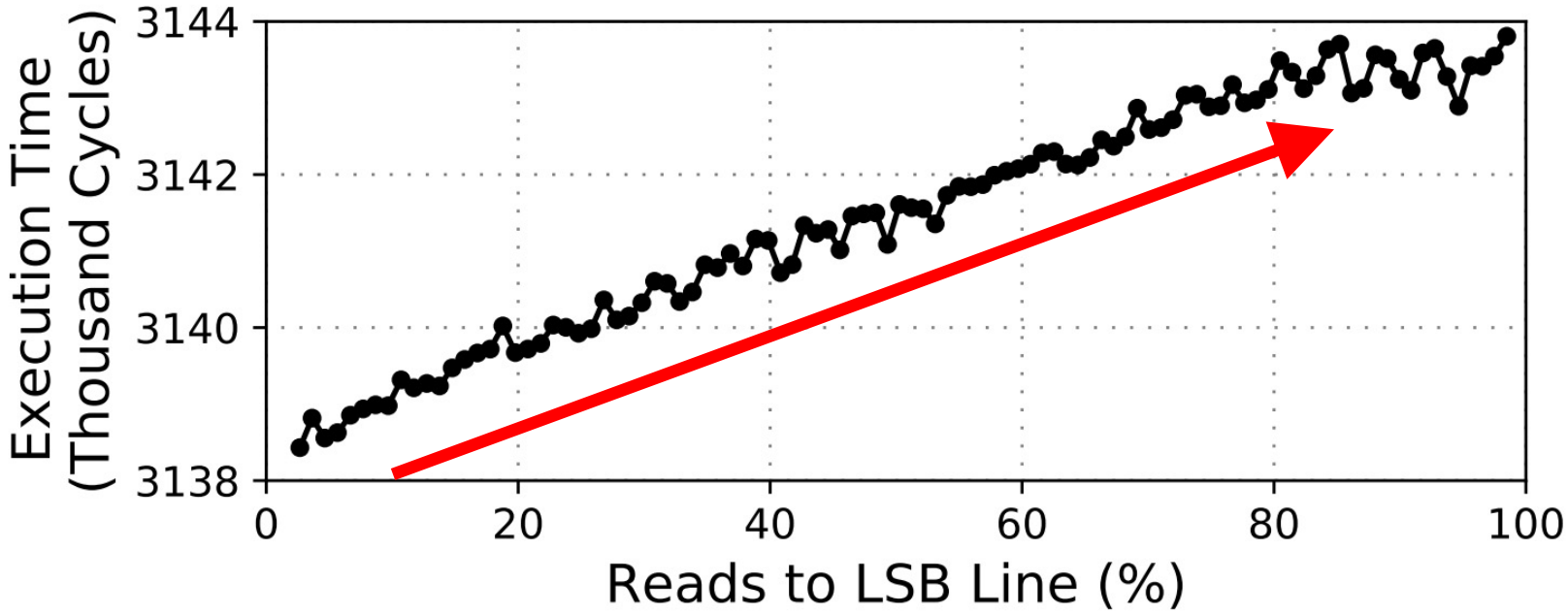
Are PCM Read Techniques Vulnerable to Leakage?

- **Microbenchmark:**
  - Issues fixed number of memory accesses
  - Varies MSB and LSB lines accesses
- **Increasing LSB ratio -> Deterministic linear increase in execution time**



Are PCM Read Techniques Vulnerable to Leakage?

- **Microbenchmark:**
  - Issues fixed number of memory accesses
  - Varies MSB and LSB lines accesses
- **Increasing LSB ratio -> Deterministic linear increase in execution time**



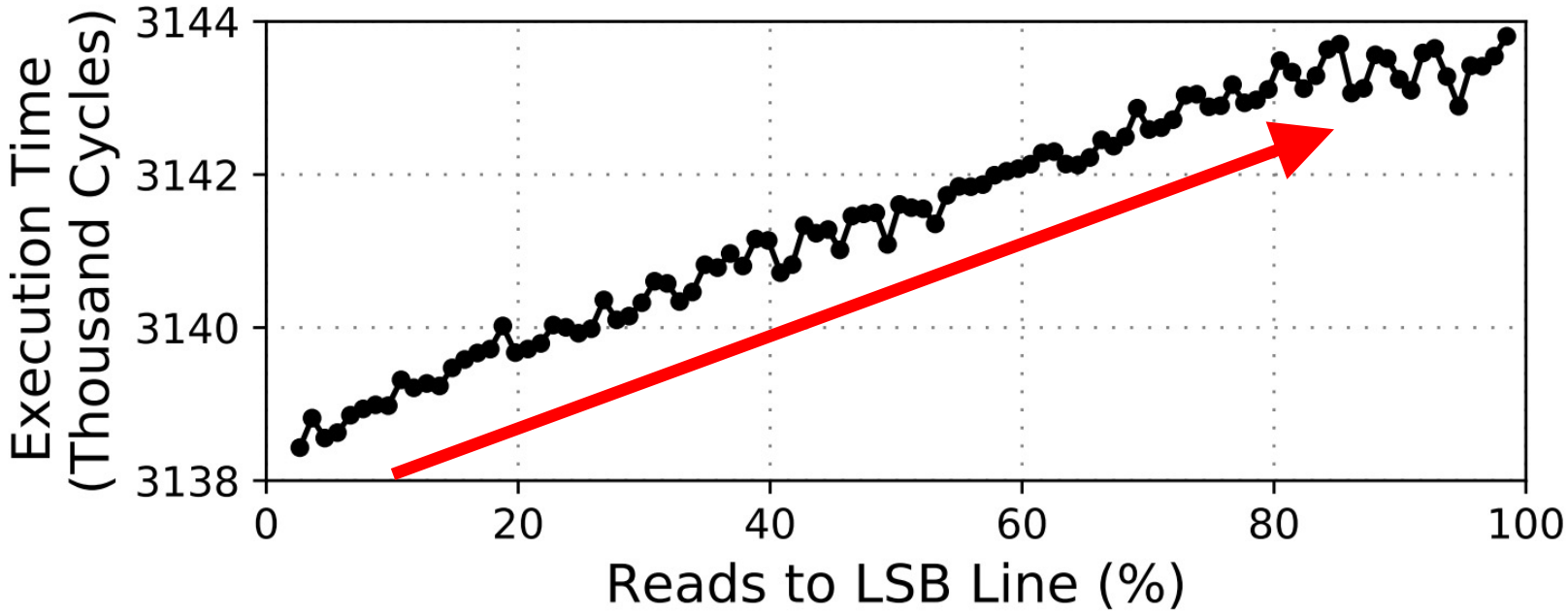
Are PCM Read Techniques Vulnerable to Leakage?

❖ Microbenchmark:

- [illegible]
- [illegible]

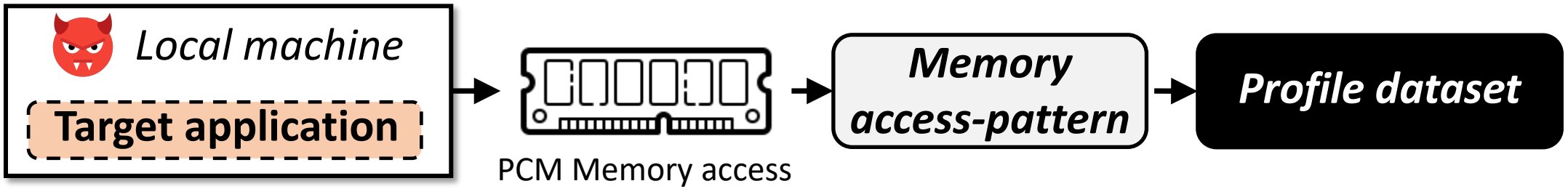
Differentiation in PCM access patterns can induce externally observable slow and fast executions.

❖ Increasing LSB ratio -> Deterministic linear increase in execution time

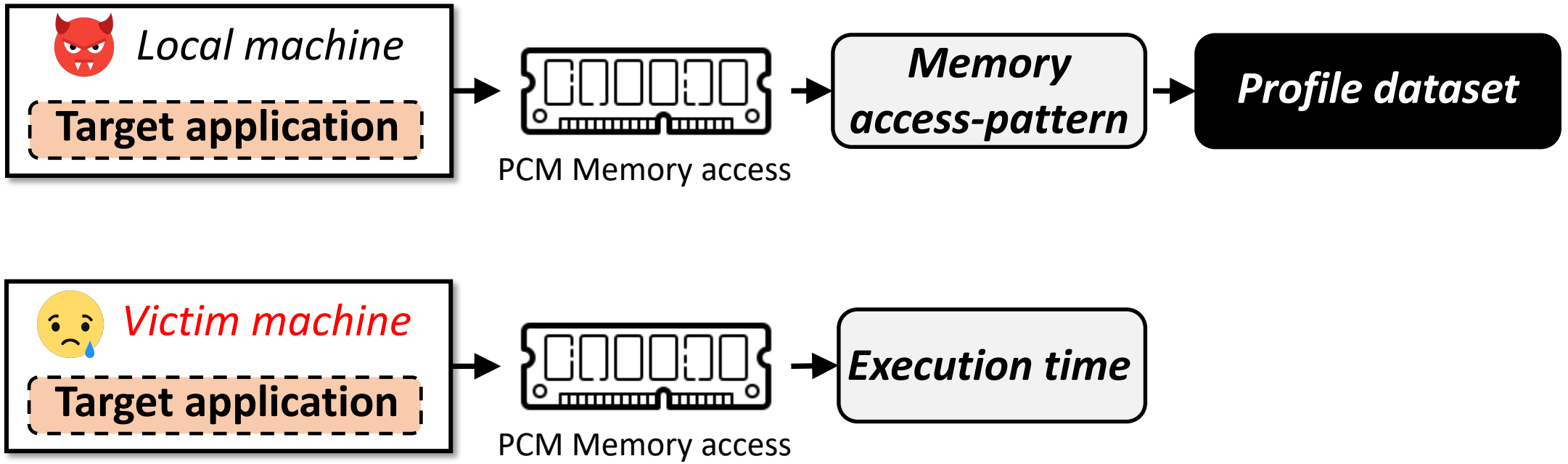


Overview of R-SAW: Exploiting Read Asymmetry

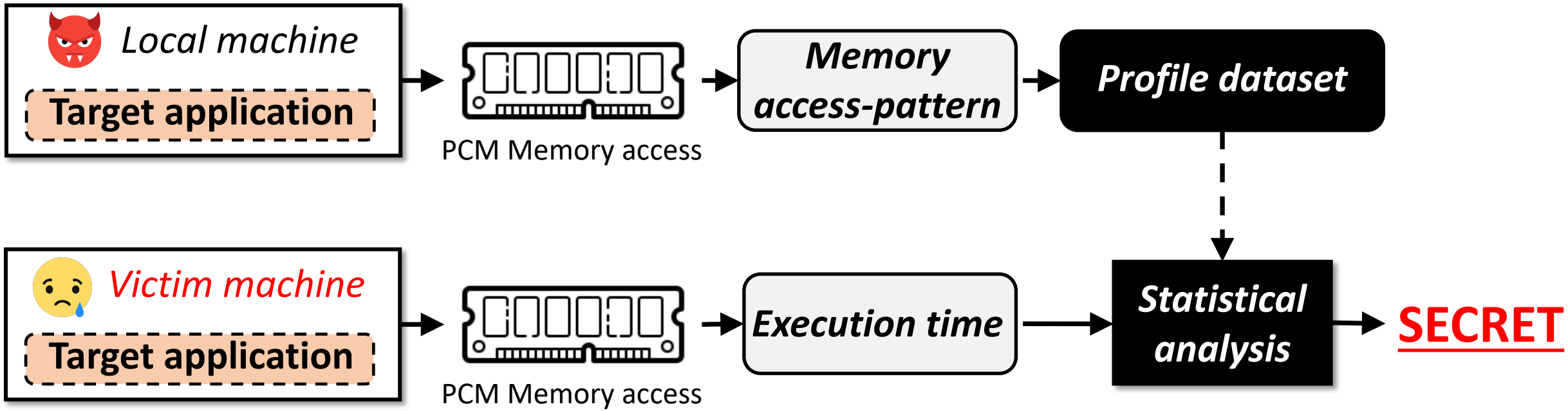
Overview of R-SAW: Exploiting Read Asymmetry



Overview of R-SAW: Exploiting Read Asymmetry

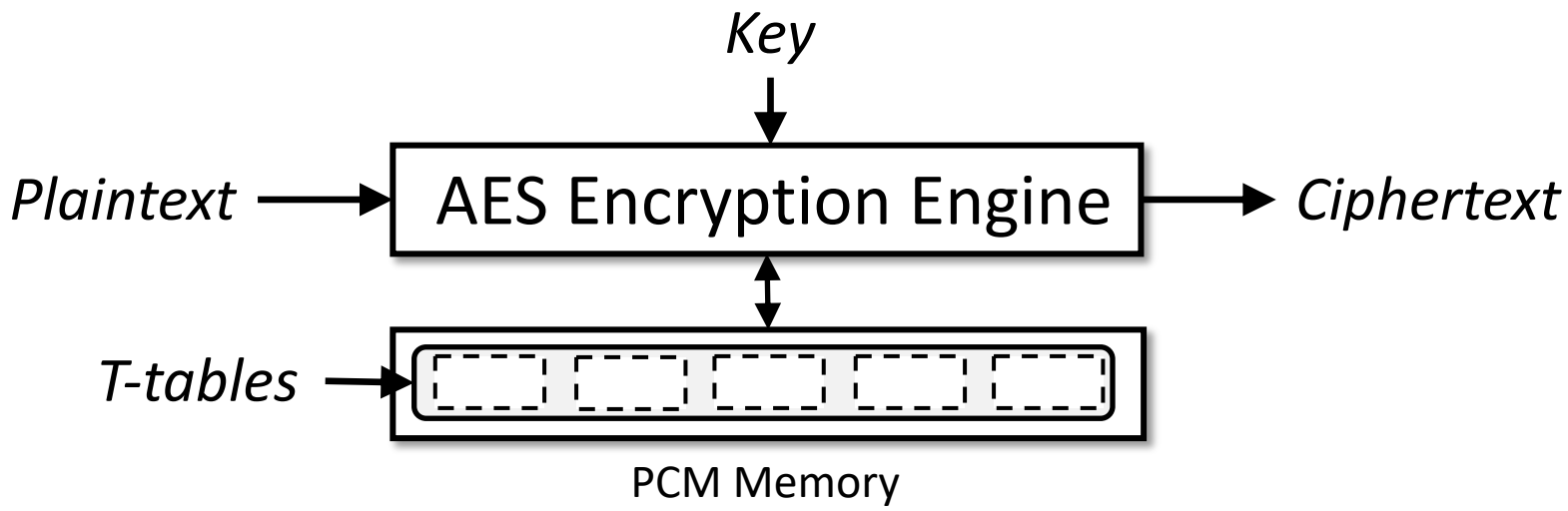


Overview of R-SAW: Exploiting Read Asymmetry



AES Encryption in OpenSSL

- AES encryption uses pre-computed values from memory (t-tables).
- PCM access patterns to these t-tables are ***secret key dependent***.



Attacking AES with R-SAW: Offline Profiling

- Collects **LSB/MSB access ratios (P)** of encryption for random plaintext (PT) and key.
- Organizes the P based on last round key byte and ciphertext (CT) byte value pair.
- For each key byte value, **MPV** stores the P corresponding to each CT byte value.

Memory-pattern vector (MPV): $\mathcal{M}(u) = \{\overline{P}^{0}_{(u)}, \overline{P}^{1}_{(u)}, \ldots, \overline{P}^{255}_{(u)}\}$

Attacking AES with R-SAW: Offline Profiling

- Collects **LSB/MSB access ratios (P)** of encryption for random plaintext (PT) and key.
- Organizes the P based on last round key byte and ciphertext (CT) byte value pair.
- For each key byte value, **MPV** stores the P corresponding to each CT byte value.

Memory-pattern vector (MPV): $\mathcal{M}(u) = \{\overline{P}^{0}_{(u)}, \overline{P}^{1}_{(u)}, \ldots, \overline{P}^{255}_{(u)}\}$

Last round key byte value, u

LSB Access ratio when <key, ciphertext> = <u, 255>

Attacking AES with R-SAW: Offline Profiling

- Collects **LSB/MSB access ratios (P)** of encryption for random plaintext (PT) and key.
- Organizes the P based on last round key byte and ciphertext (CT) byte value pair.
- For each key byte value, **MPV** stores the P corresponding to each CT byte value.

Memory-pattern vector (MPV): $\mathcal{M}(u) = \{\overline{P}^{0}_{(u)}, \overline{P}^{1}_{(u)}, \ldots, \overline{P}^{255}_{(u)}\}$

Last round key byte value, u

LSB Access ratio when <key, ciphertext> = <u, 255>

Profile dataset: **For each key byte, 256 MPVs corresponding to each value of *u***

Attacking AES with R-SAW: Runtime Monitoring

- Attacker monitors **encryption times (L)** for AES encryptions (**unknown key**).
- Organizes the L based ciphertext byte value.
- Attacker creates **ETV** by collecting L for each ciphertext byte value.

Encryption-timing vector (ETV): $\mathcal{T}(\mathbf{x}) = \{\overline{L}^{0}_{(\mathbf{x})}, \overline{L}^{1}_{(\mathbf{x})}, \ldots, \overline{L}^{255}_{(\mathbf{x})}\}$

Attacking AES with R-SAW: Runtime Monitoring

- Attacker monitors **encryption times (L)** for AES encryptions (**unknown key**).
- Organizes the L based ciphertext byte value.
- Attacker creates **ETV** by collecting L for each ciphertext byte value.

Encryption-timing vector (ETV): $\mathcal{T}(\mathbf{x}) = \{\overline{L}^{0}_{(\mathbf{x})}, \overline{L}^{1}_{(\mathbf{x})}, \cdots, \overline{L}^{255}_{(\mathbf{x})}\}$

Key byte value, **unknown**

Encryption time when
<key, ciphertext> = <**unknown**, 255>

Attacking AES with R-SAW: Runtime Monitoring

- Attacker monitors **encryption times (L)** for AES encryptions (**unknown key**).
- Organizes the L based ciphertext byte value.
- Attacker creates **ETV** by collecting L for each ciphertext byte value.

Encryption-timing vector (ETV): $\mathcal{T}(\mathbf{x}) = \{\overline{L}^{0}_{(\mathbf{x})}, \overline{L}^{1}_{(\mathbf{x})}, \cdots, \overline{L}^{255}_{(\mathbf{x})}\}$

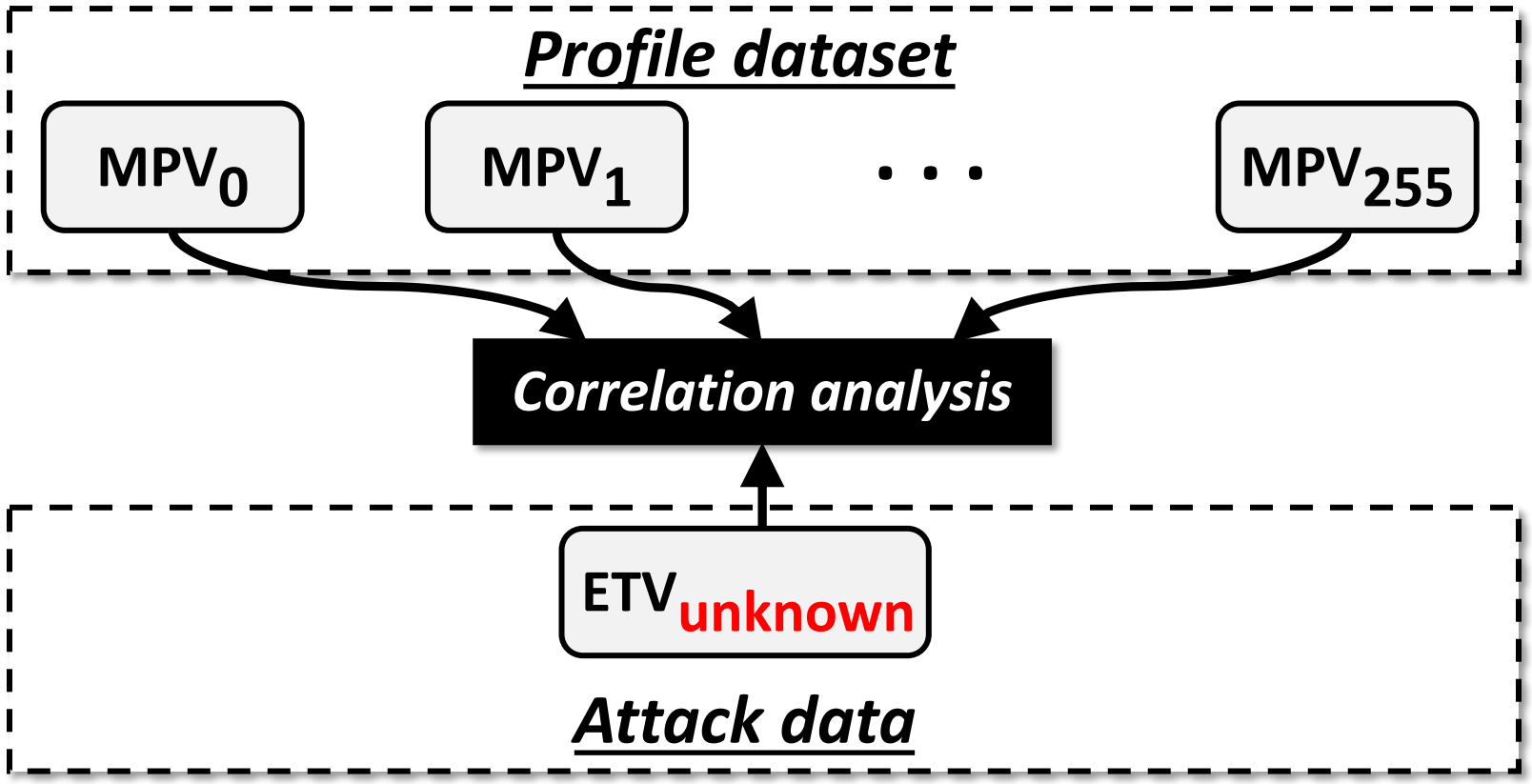
Key byte value, **unknown**

Encryption time when
<key, ciphertext> = <**unknown**, 255>

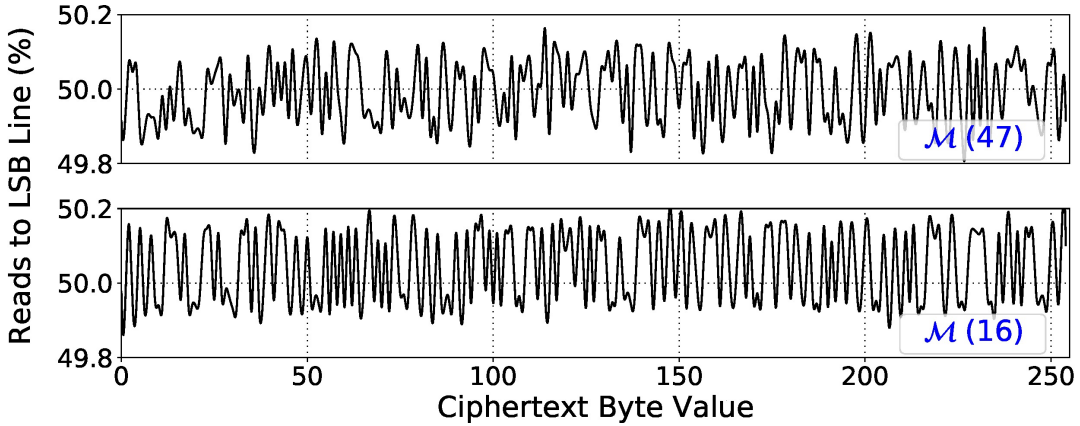
Attack data: **One ETV for the key byte value, *unknown***

Attacking AES with R-SAW: Correlation Analysis

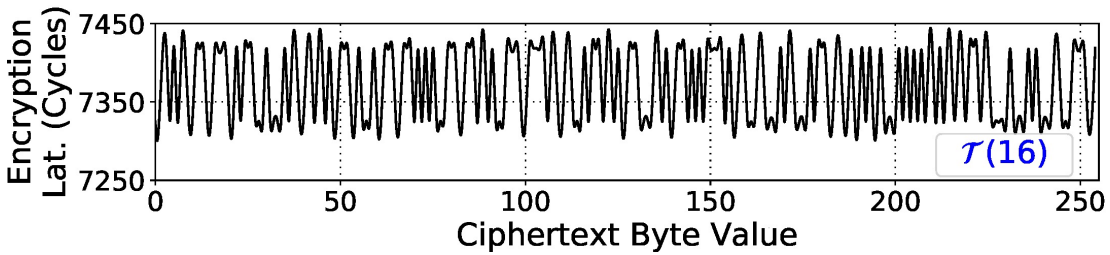
- Correlation analysis between *attack data* and *profile dataset*.
- Highest and outstanding correlation may indicate the **unknown** key value.



Attacking AES with R-SAW: Evaluation of Attack

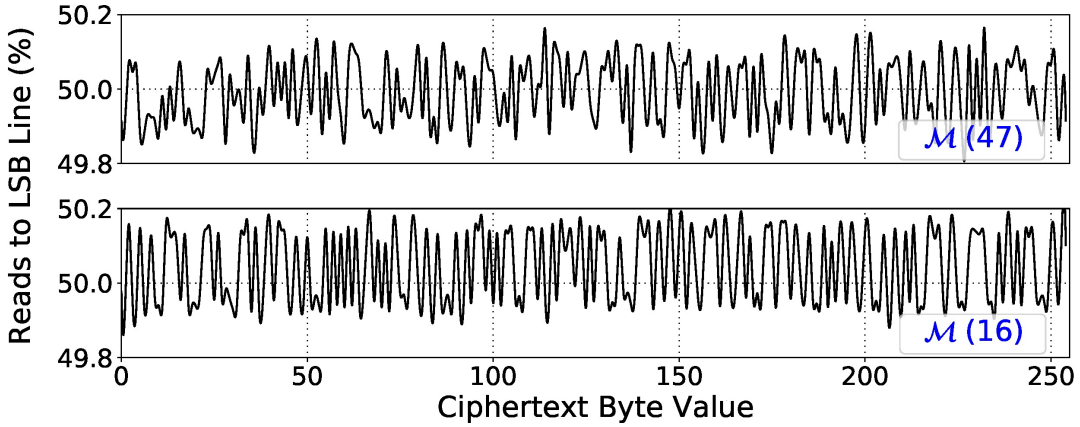


Two example traces of MPV from local system

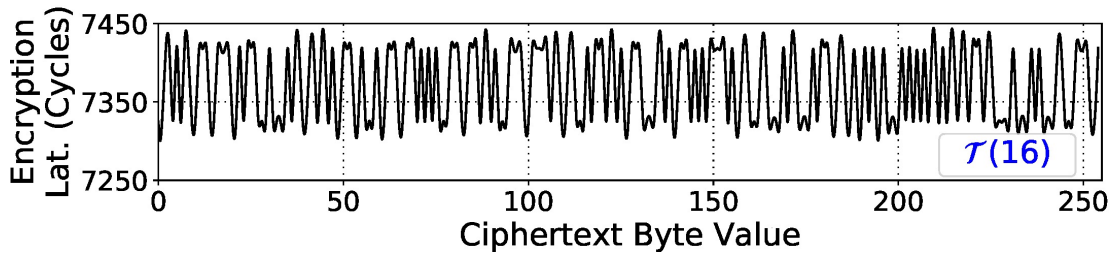


Example ETV from Victim system

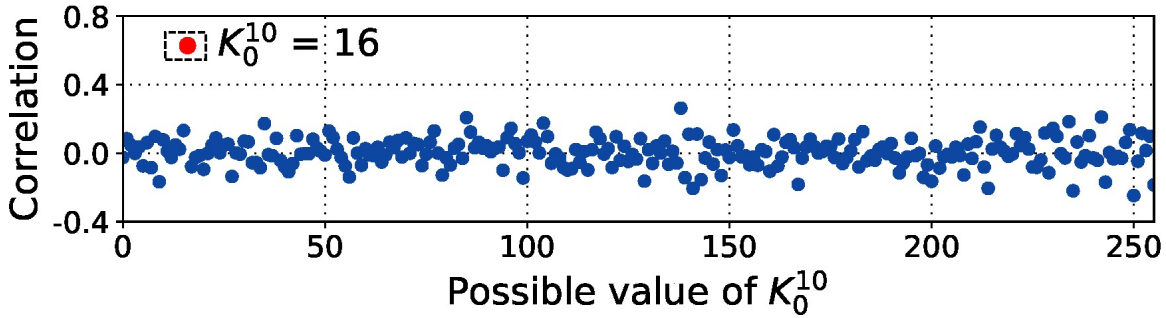
Attacking AES with R-SAW: Evaluation of Attack



Two example traces of MPV from local system

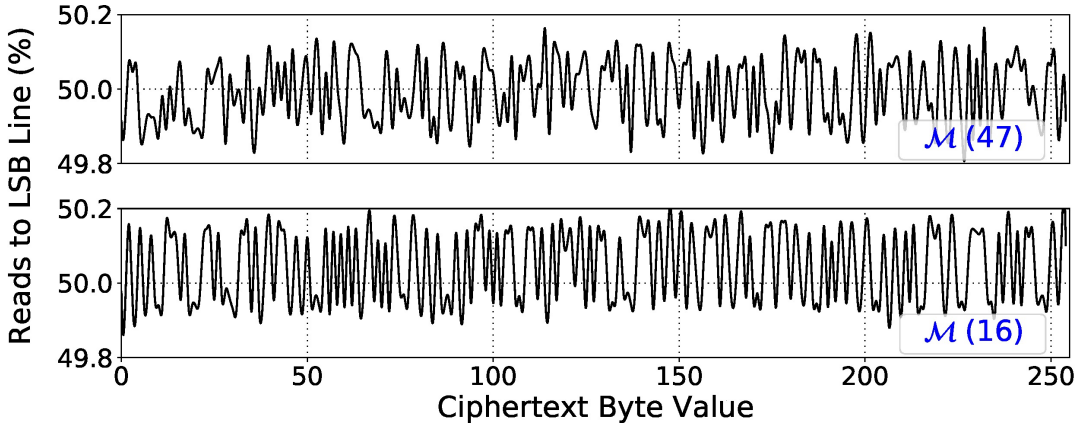


Example ETV from Victim system

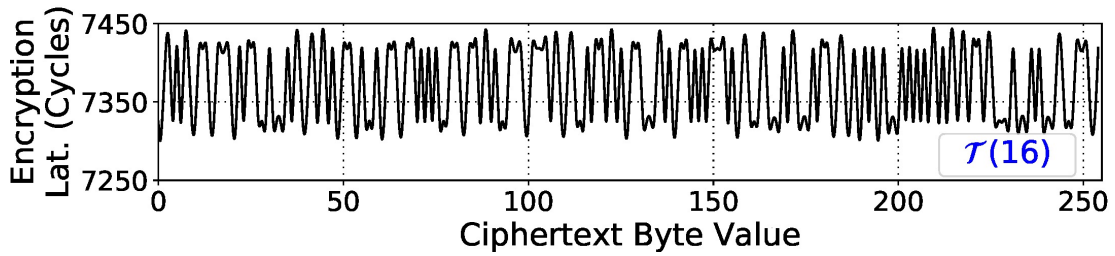


Correlation analysis

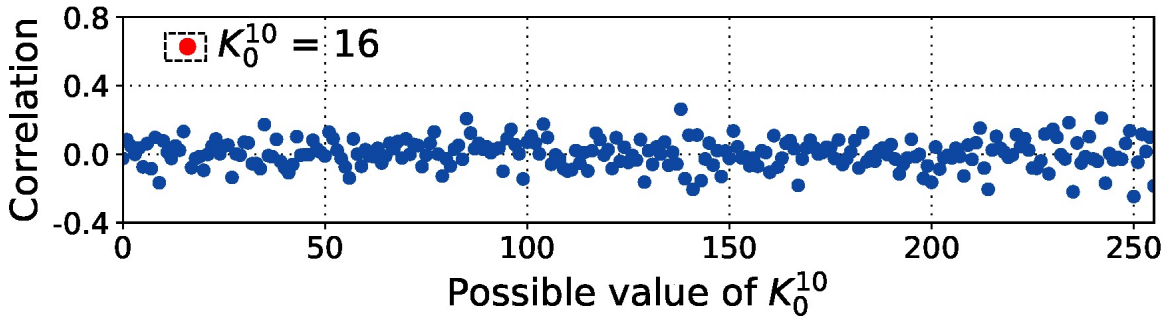
Attacking AES with R-SAW: Evaluation of Attack



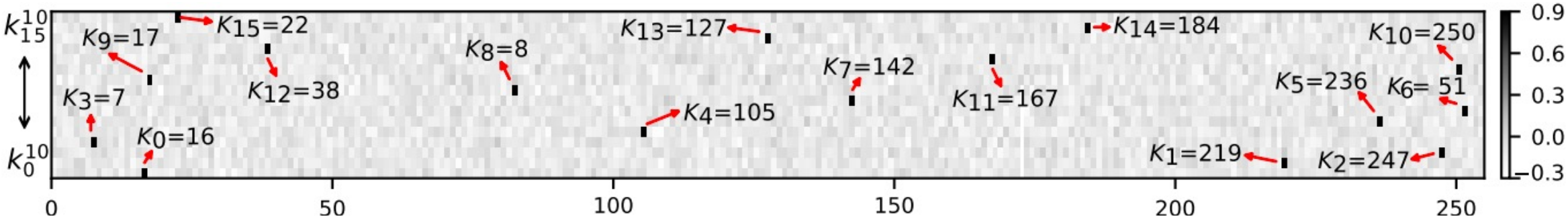
Two example traces of MPV from local system



Example ETV from Victim system

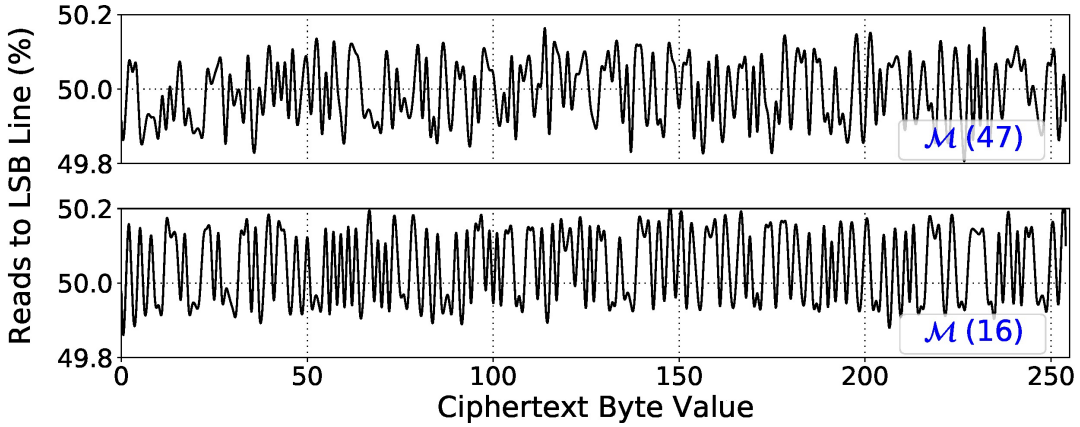


Correlation analysis

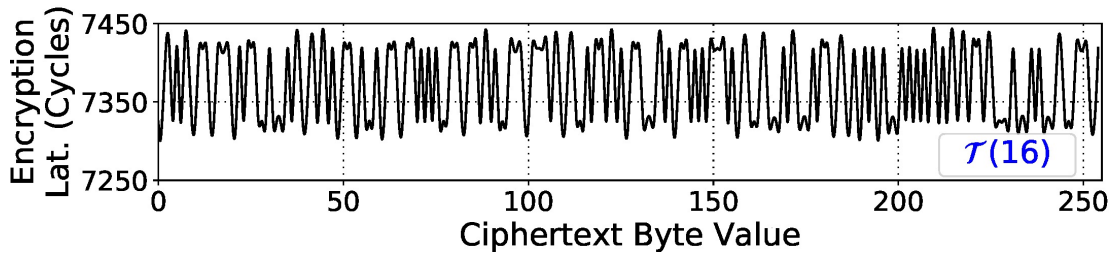


Complete 16-byte AES Key recovery using R-SAW

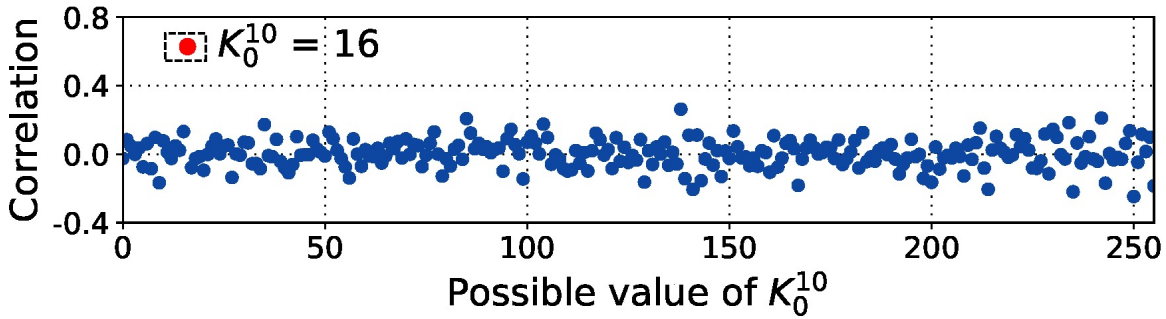
Attacking AES with R-SAW: Evaluation of Attack



Two example traces of MPV from local system

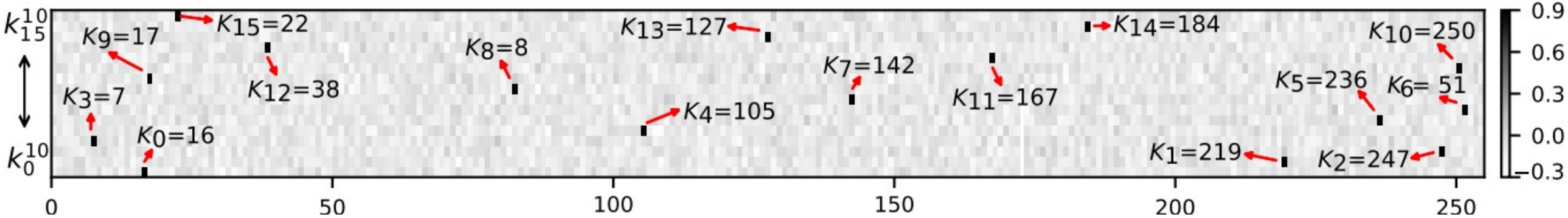


Example ETV from Victim system



Correlation analysis

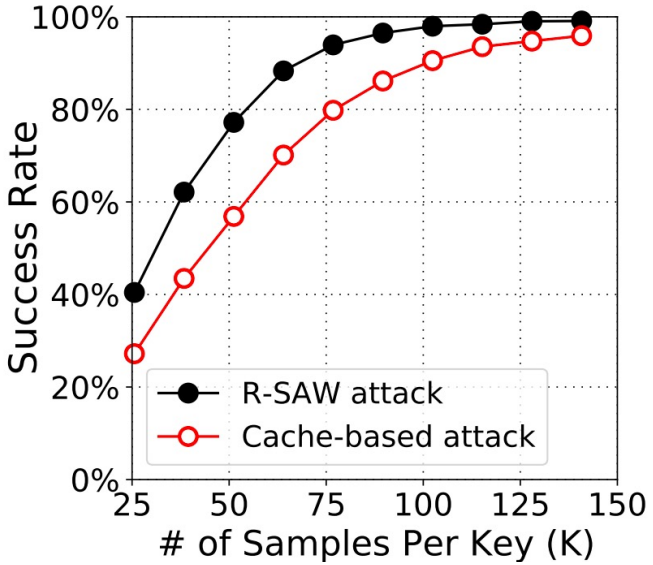
98.5% accuracy of recovering 100 keys.



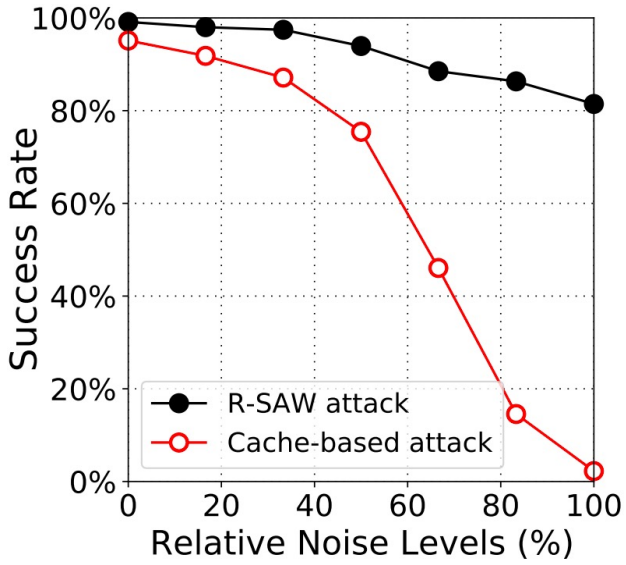
Complete 16-byte AES Key recovery using R-SAW

More on paper

- Comparison of R-SAW with state-of-the-art cache-based attacks.
  - Resiliency of R-SAW against system noise
  - Feasibility of R-SAW with small number of attack samples
- Discussions on potential mitigations for R-SAW.



Sample-size study



Noise resiliency

Thanks! Questions?

Md Hafizul Islam Chowdhuryy
Email: reyad@knights.ucf.edu