A Write-Friendly and Fast-Recovery Scheme for Security Metadata in Non-Volatile Memories

Jianming Huang, Yu Hua

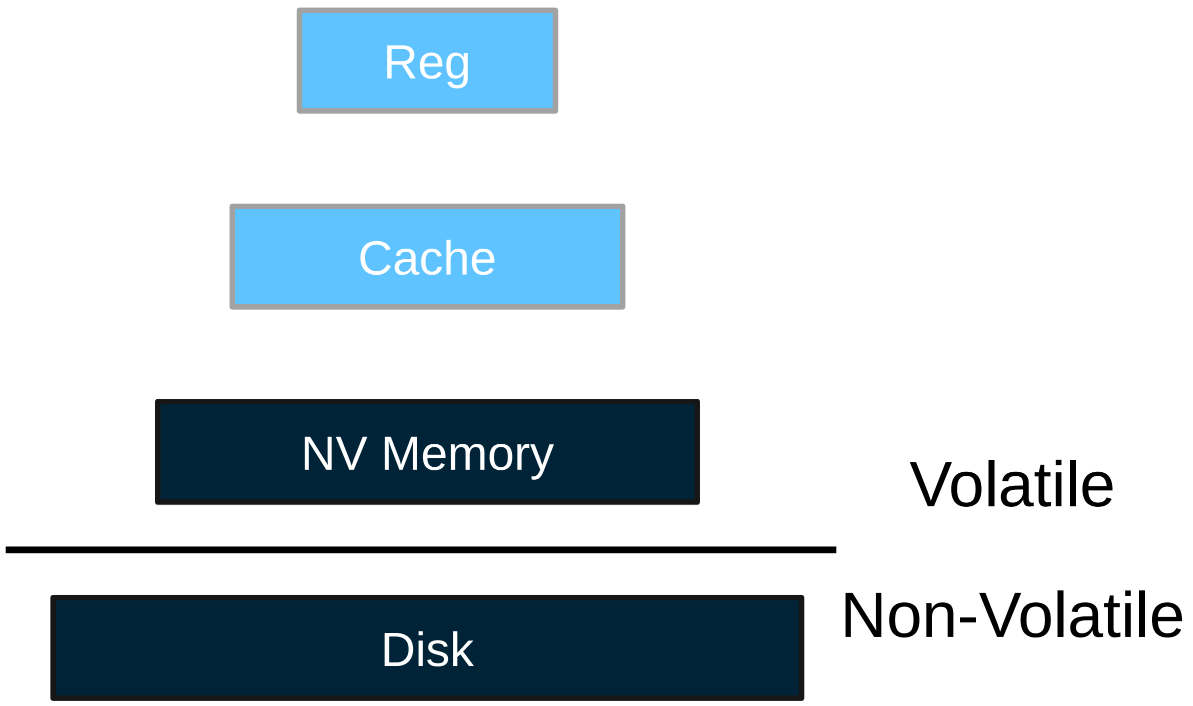
Huazhong University of Science and Technology

HPCA 2021

Outline

- Background and Motivation
- STAR Mechanism
- Evaluation
- Conclusion

Non-Volatile Memory



- Persistent memory
- Non-volatile boundary changes

NVMs need to ensure the data crash-consistency after system crashes and reboots

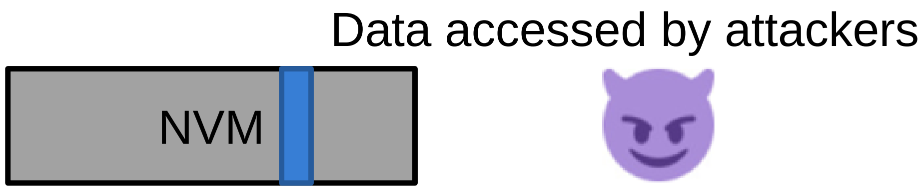
Threat Models in NVM

➢ Leaking sensitive data to attackers

- Snooping bus; Scanning memory; Stealing DIMM ……

Solution: Encryption

[Silent shredder@asplos16,Secret@DAC16…]

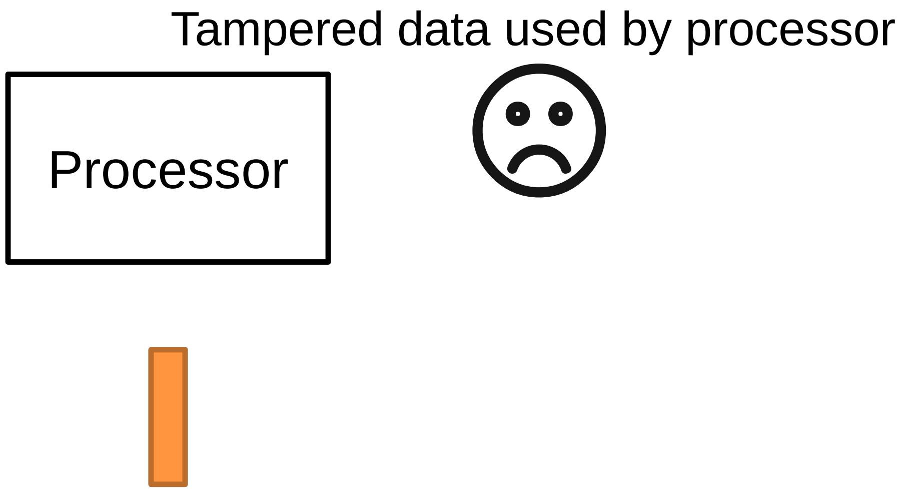


➢ Modifying data without authentication

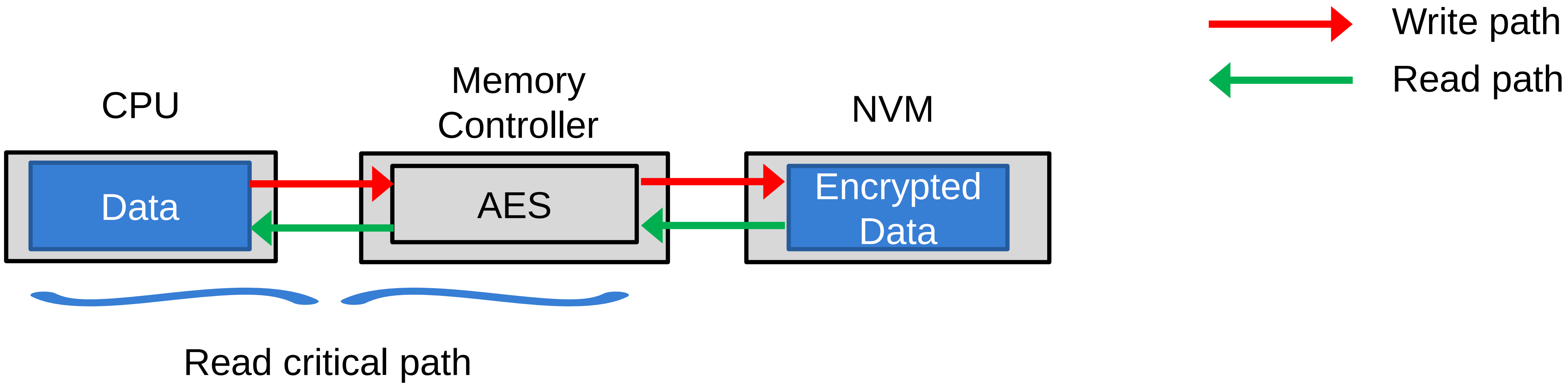
- Tampering data; Replaying data ……

Solution: Integrity Verification

[Anubis@ISCA19,Triad-NVM@ISCA19…]

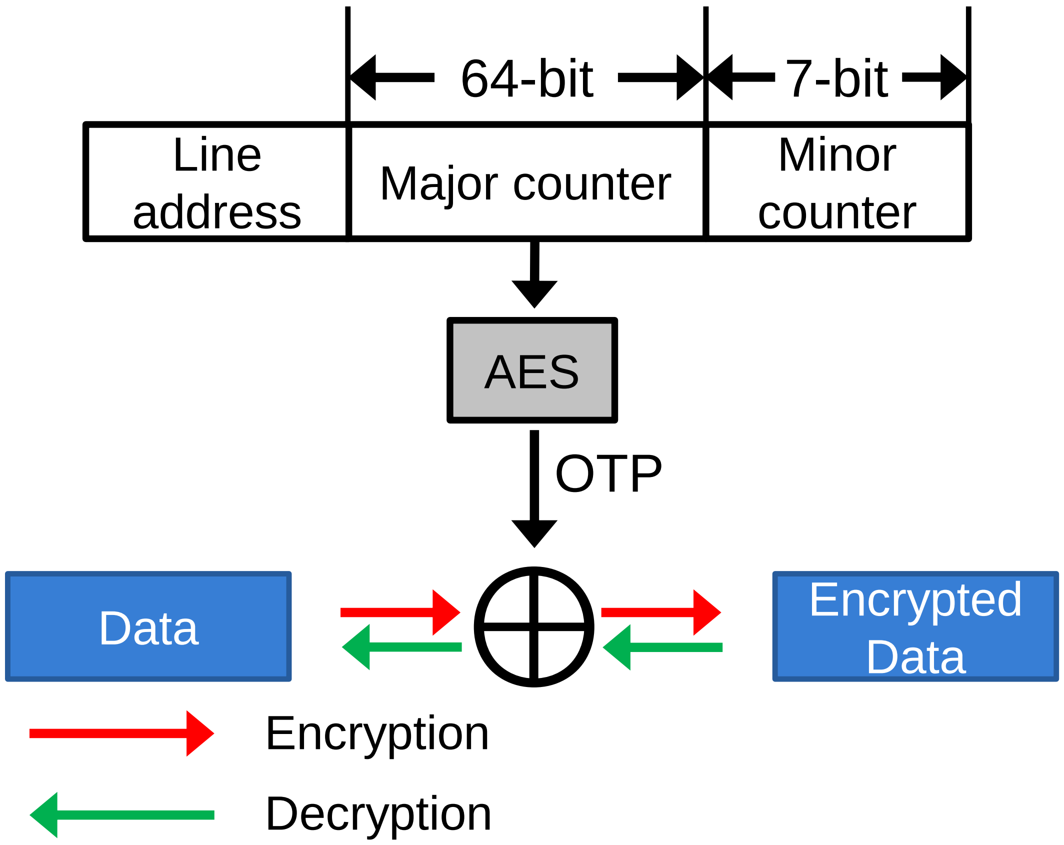


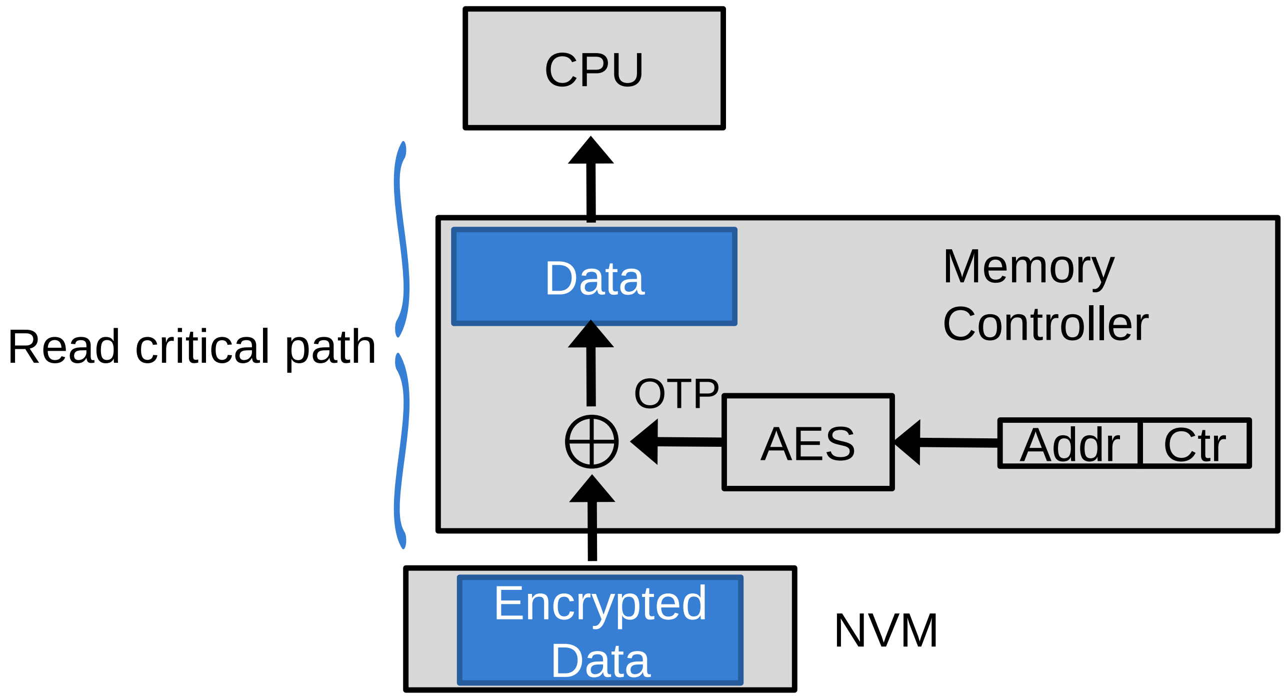
Direct Encryption



- Insecure
  - Unchanged secret key
- Low performance
  - Decryption on the read critical path

Counter Mode Encryption



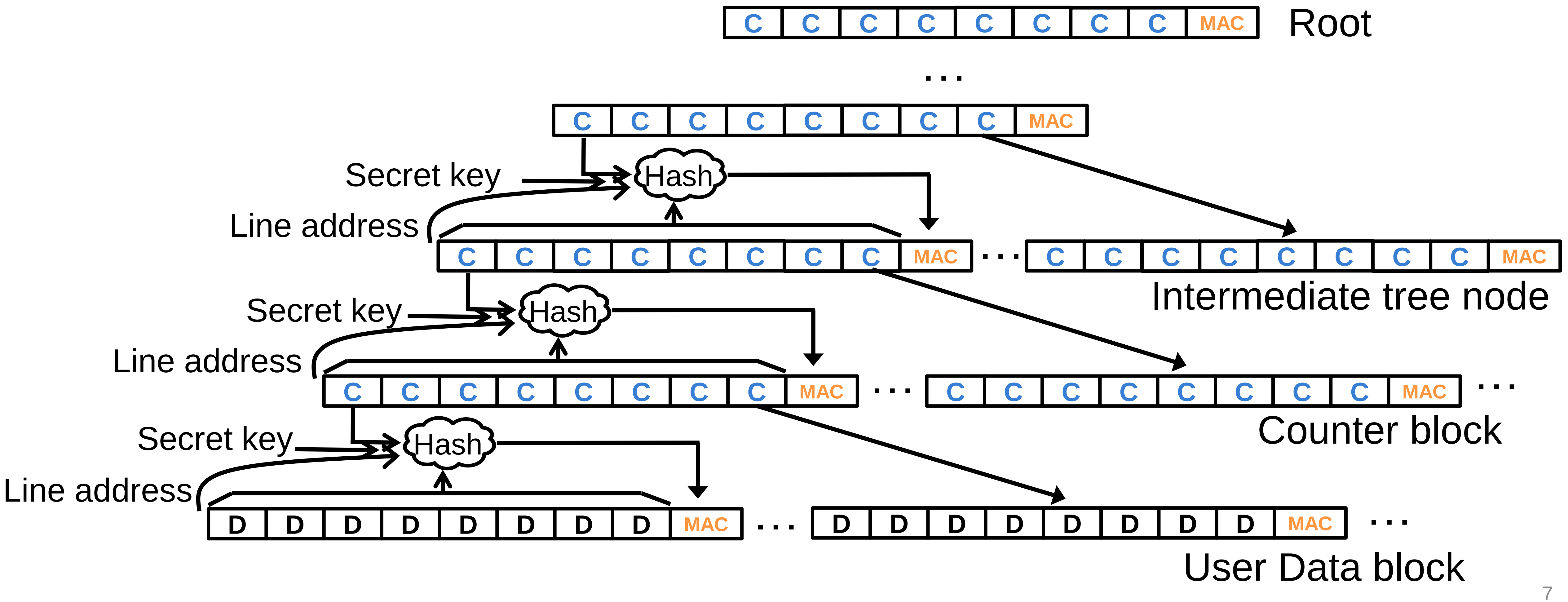


- ➢ Safer than direct AES
- ➢ Lower performance penalty than

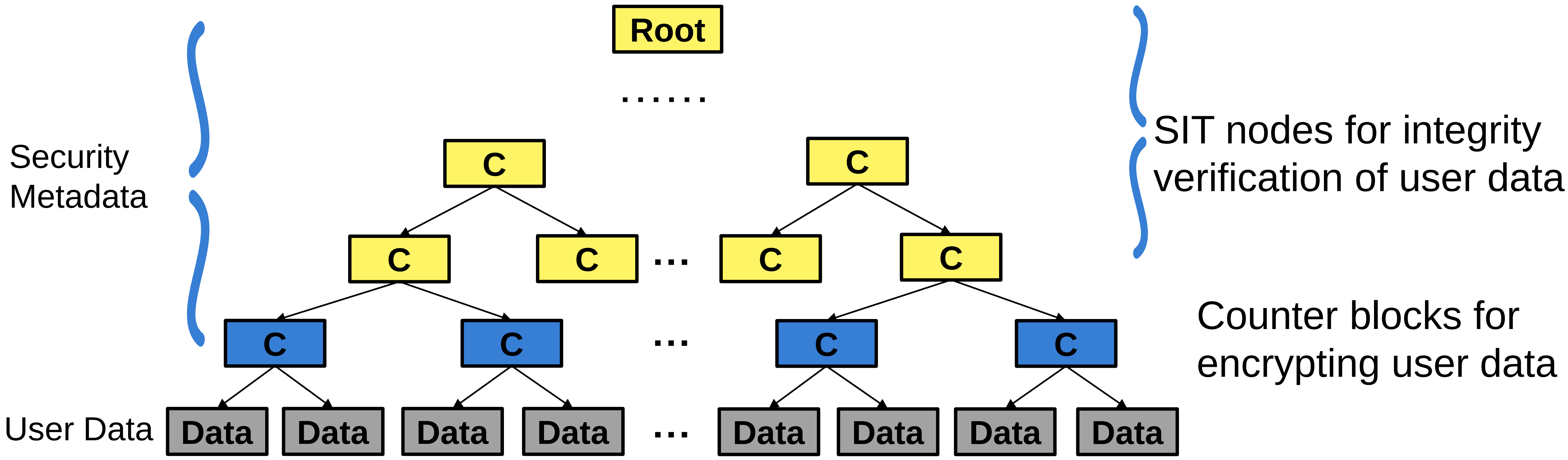
We use CME to encrypt data

Integrity Verification

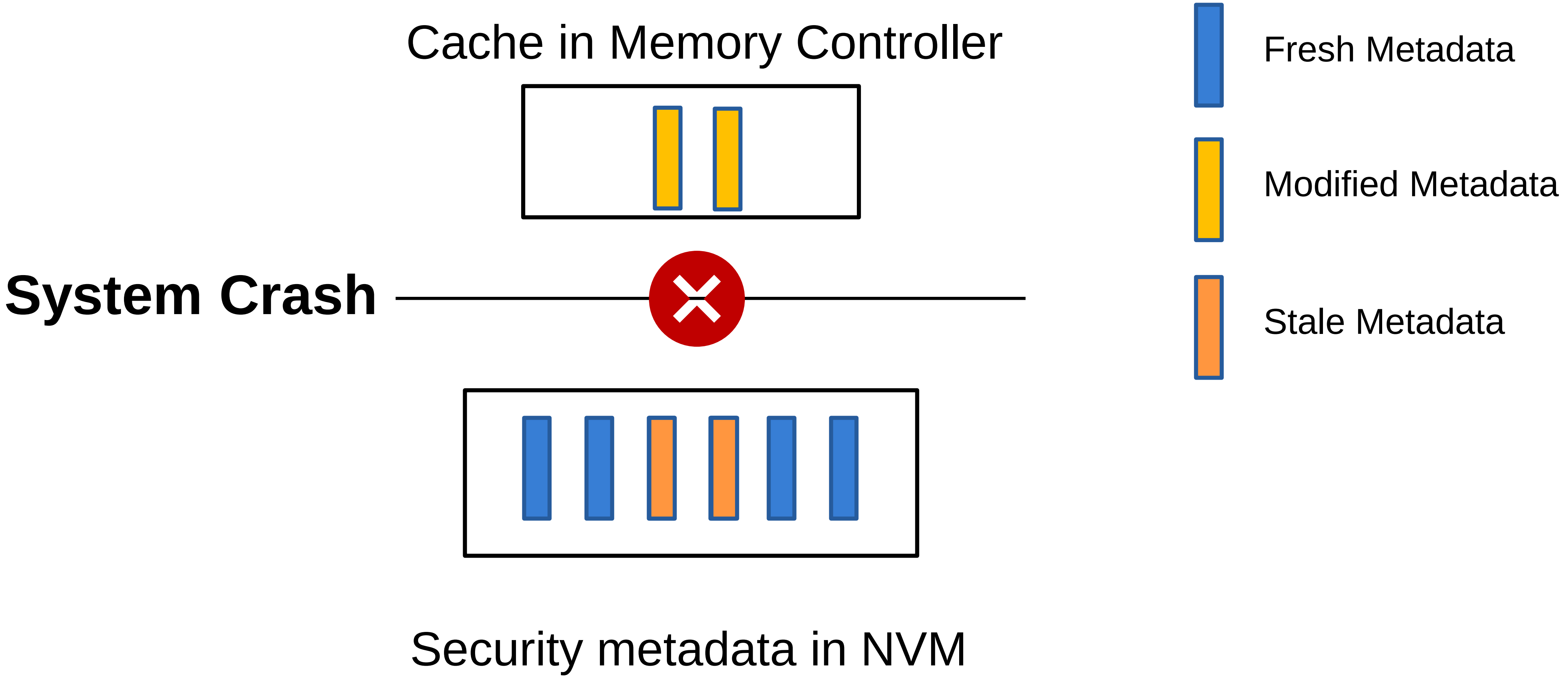
➢ SGX Integrity Tree (SIT): **Counters** and **Message Authentication Codes (MACs)**



Security Metadata



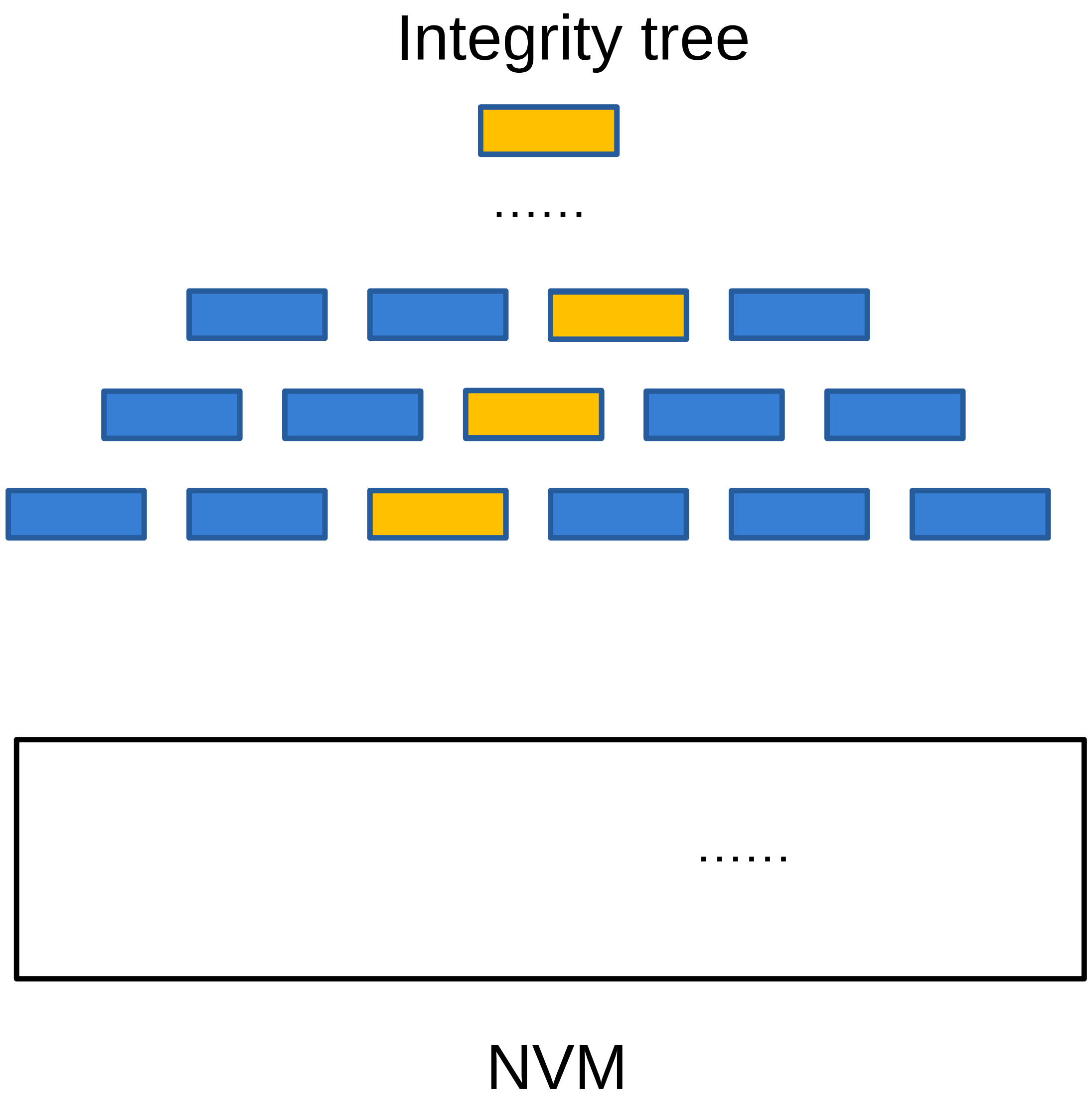
Metadata Inconsistency



Stale security metadata can't ensure the system security after reboots

Problems of Recovering Metadata

- High write overheads
  - Persisting tree nodes from leaves to root



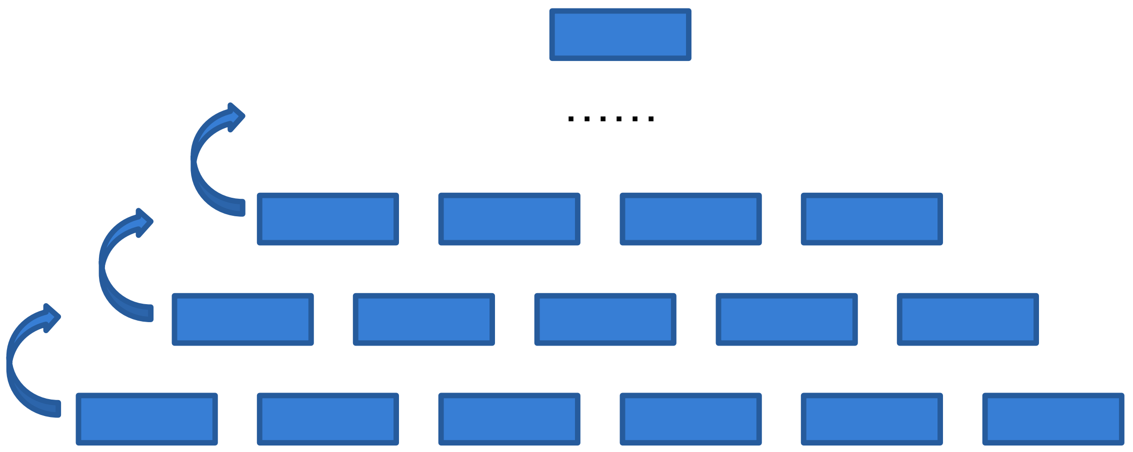
Problems of Recovering Metadata

➢ High write overheads

- Persisting tree nodes from leaves to root

➢ Long recovery time

- Reconstructing all nodes layer by layer

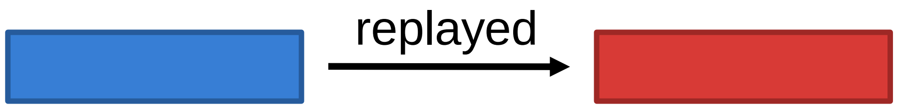


Problems of Recovering Metadata

- ➢ High write overheads
  - Persisting tree nodes from leaves to root

- ➢ Long recovery time
  - Reconstructing all nodes layer by layer

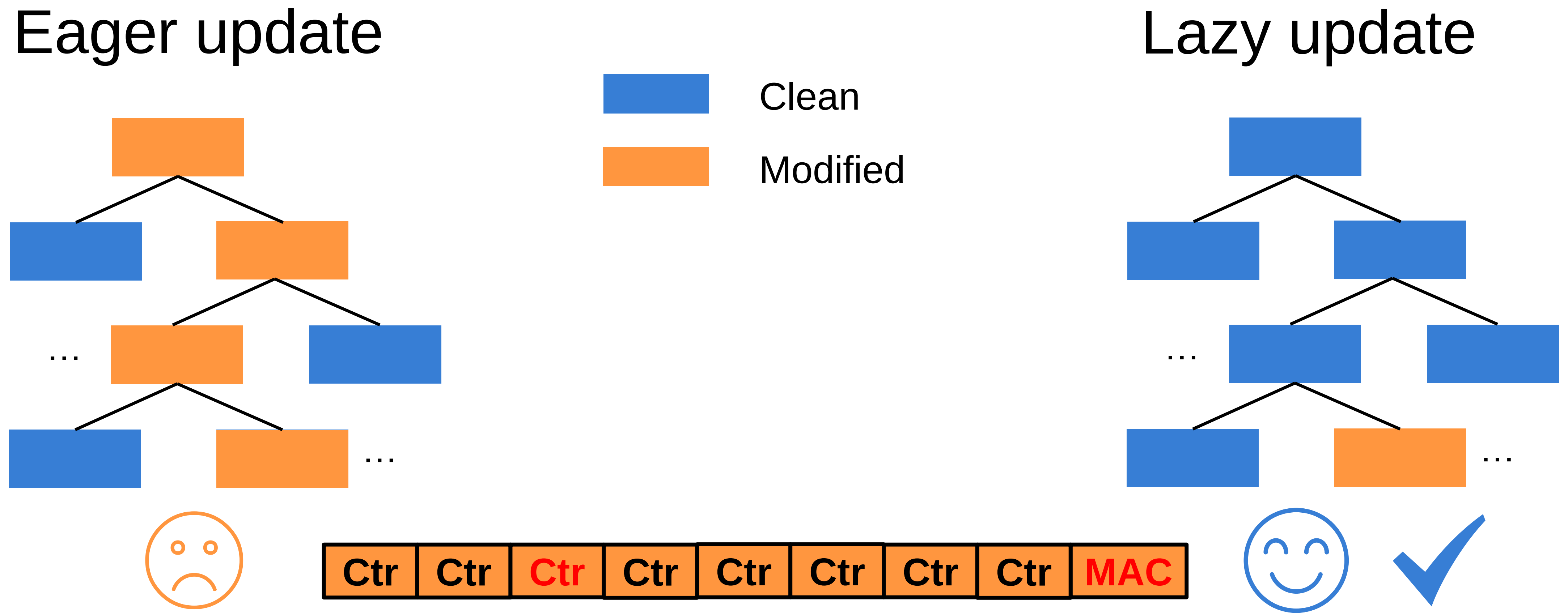
- ➢ Incorrectly recovery
  - Attacking nodes during recovery



Our design goal: correctly recover the security metadata with low write overhead and short recovery time

Observation

Modifications in the updated nodes



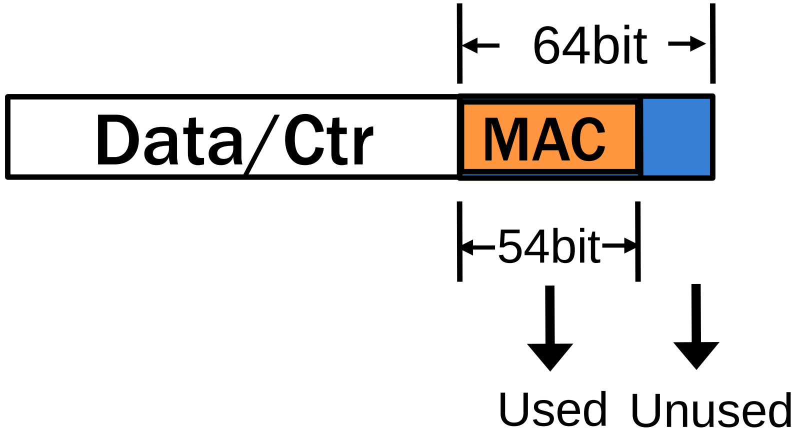
Only the corresponding counter increases by 1, and MAC is updated

Observation

Unused space in MAC (Message Authentication Code) field

64-bit MAC field in data line

- 54-bit MAC is also safe[1].



[1] Saileshwar, Gururaj and Nair, Prashant J and Ramrakhyani, Prakash and Elsasser, Wendy and Joao, Jose A and Qureshi, Moinuddin K. Morphable Counters: Enabling Compact Integrity Trees For Low-Overhead Secure Memories. MICRO18

Solutions

- Store the right metadata with low overhead
  - Recovering stale metadata using right metadata
- Identify the stale metadata
  - Only restoring the stale metadata
- Verify the recovery process
  - Detecting the attacks occurring during recovery

We propose an efficient recovery scheme STAR

Outline

- Background and Motivation
- STAR Mechanism
- Evaluation
- Conclusion

STAR Components

➢Counter-MAC synergization

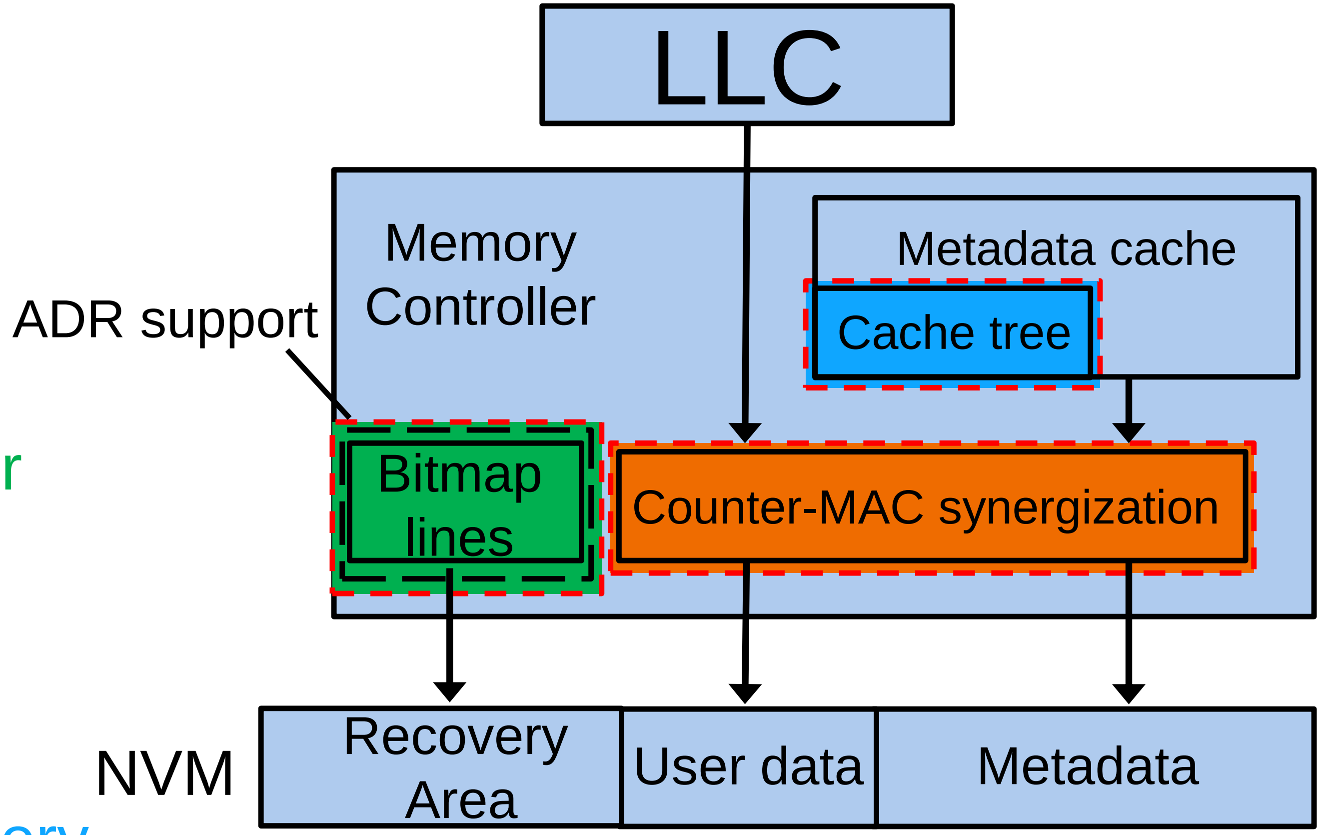
Persist the modifications w/o extra writes

➢Bitmap lines

Record the locations of stale metadata for reducing recovery time

➢Cache tree

Detect the attacks occurring during recovery



STAR Components

- ➢Counter-MAC synergization

Persist the modifications w/o extra writes

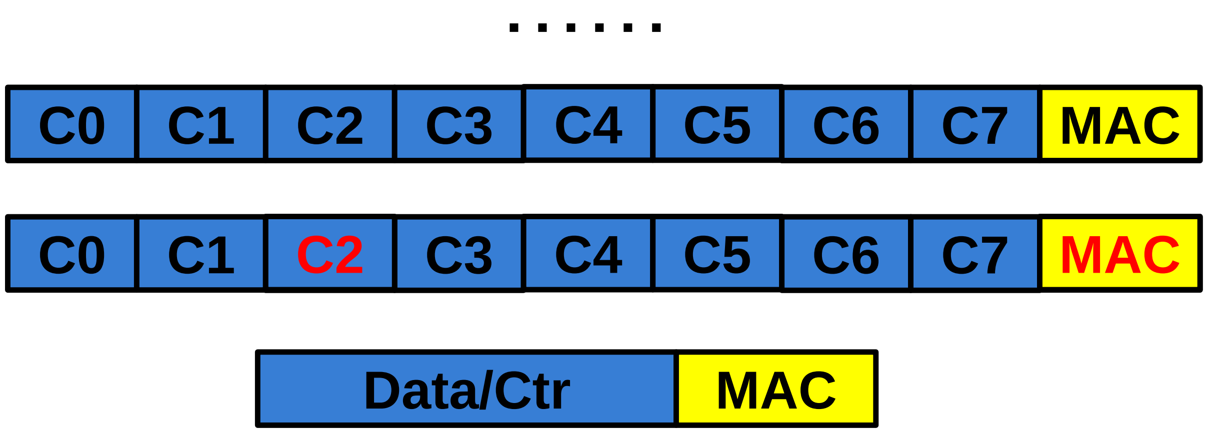
- ➢Bitmap lines

Record the locations of stale metadata for reducing recovery time

- ➢Cache tree

Detect the attacks during recovery

Counter-MAC Synergization

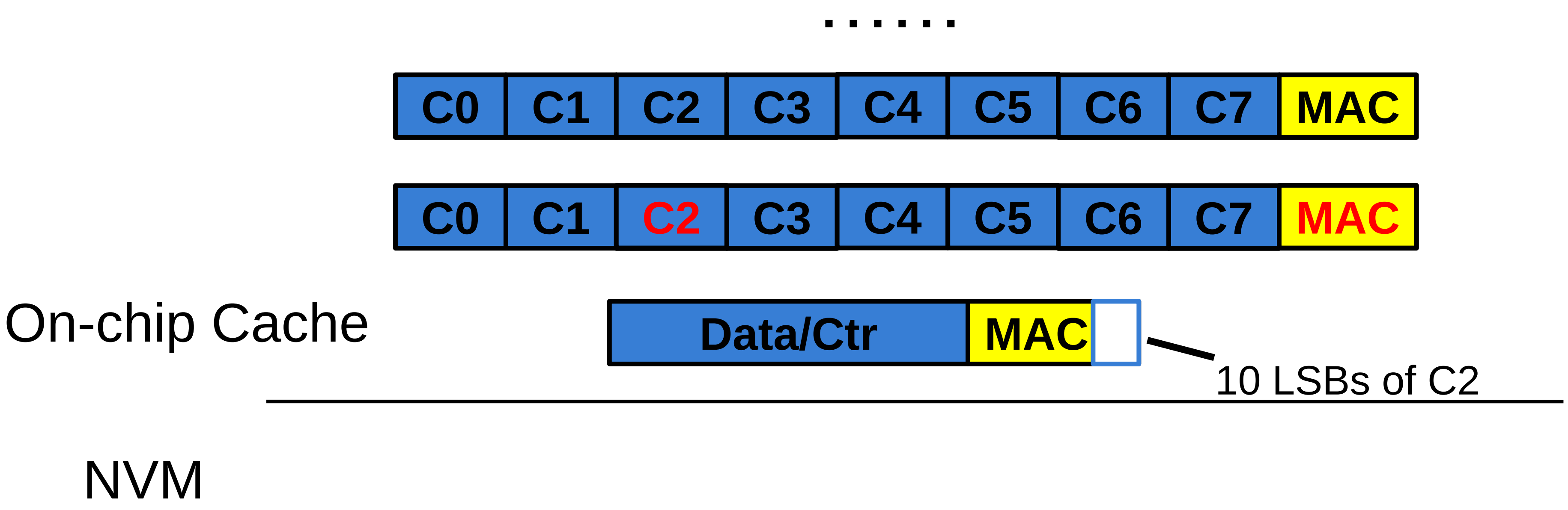


On-chip Cache

NVM

Incurring modifications in parent node via persisting child node

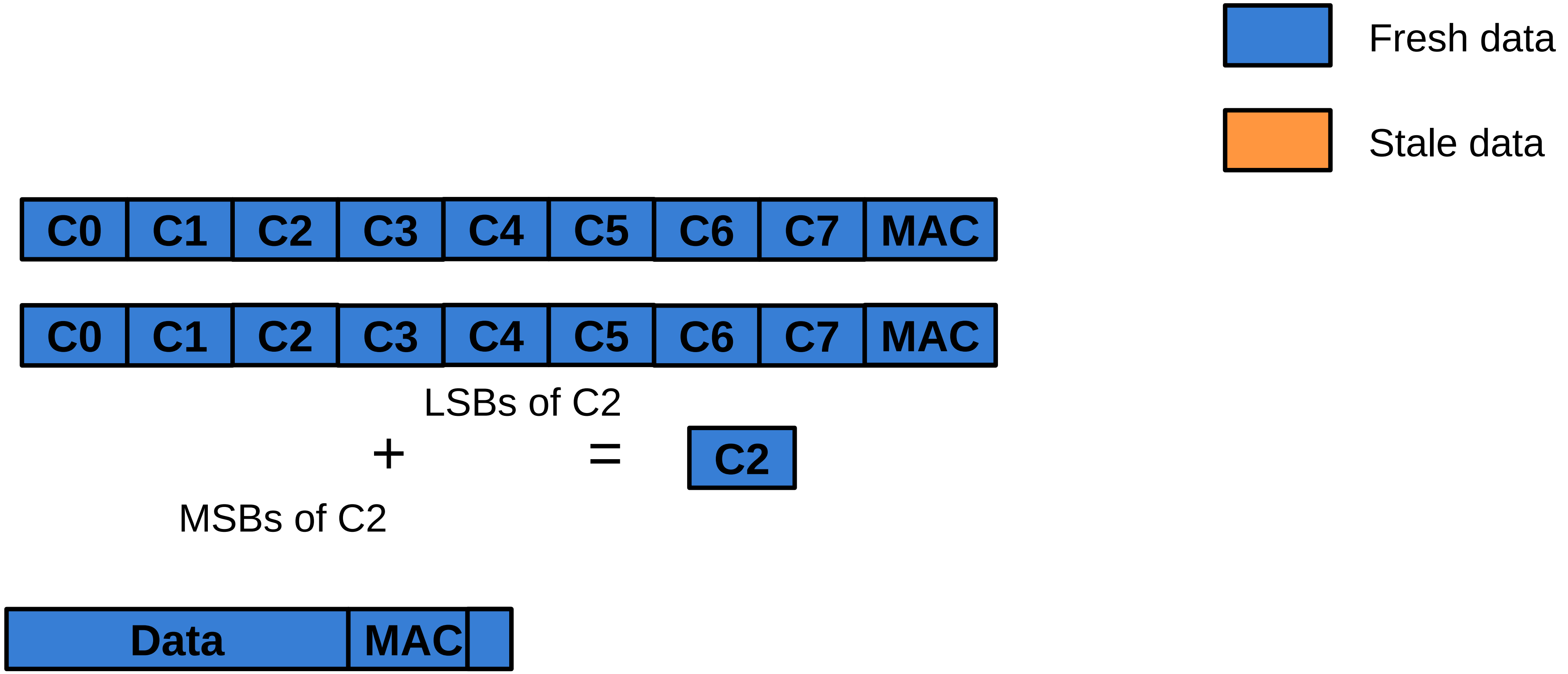
Counter-MAC Synergization



Persisting the child node and modifications in one write

Counter-MAC Synergization

- Restoring stale counter and MAC



STAR Components

➢Counter-MAC synergization

Persist the modifications w/o extra writes

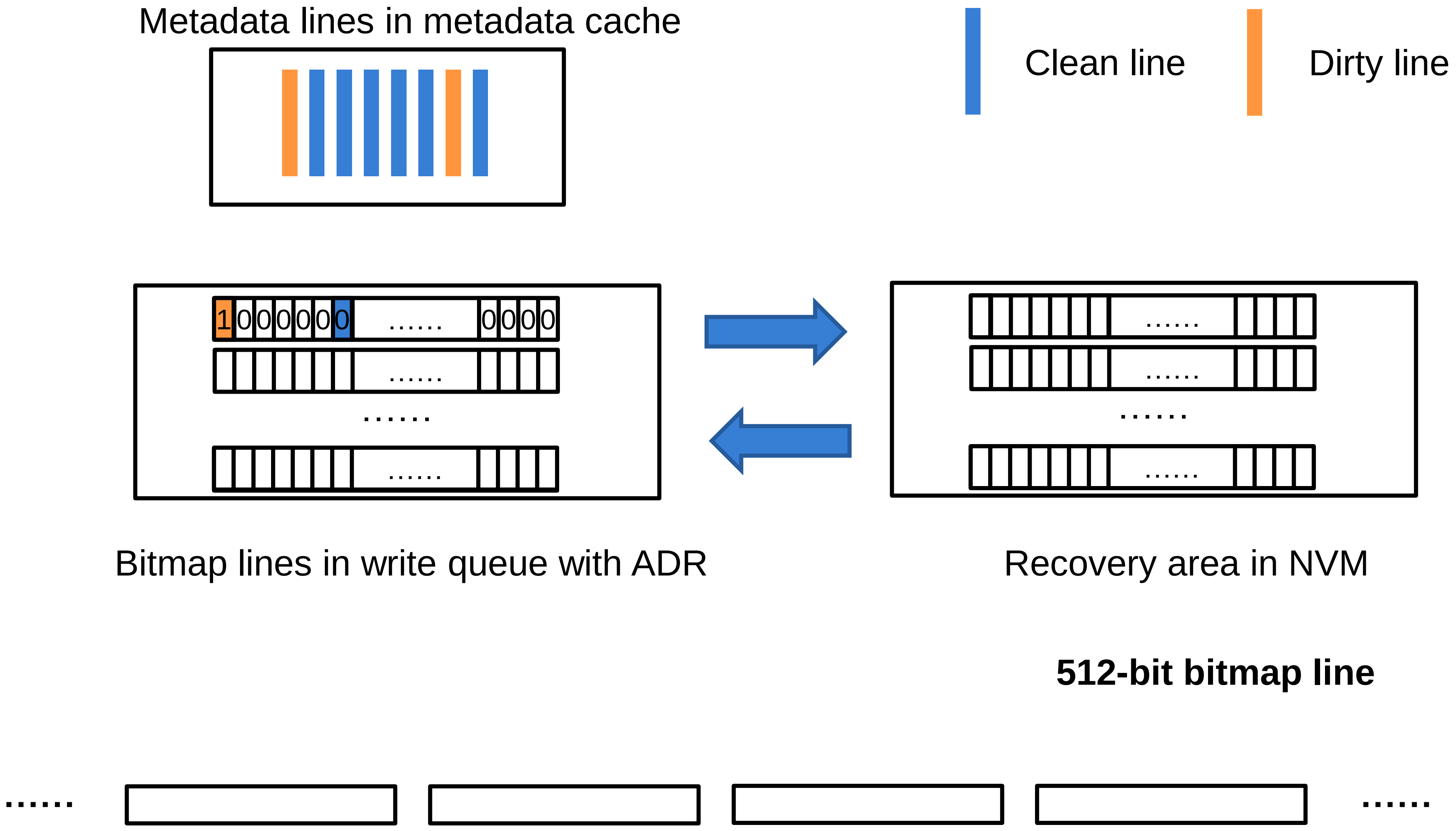
➢Bitmap lines

Record the locations of stale metadata for reducing recovery time

➢Cache tree

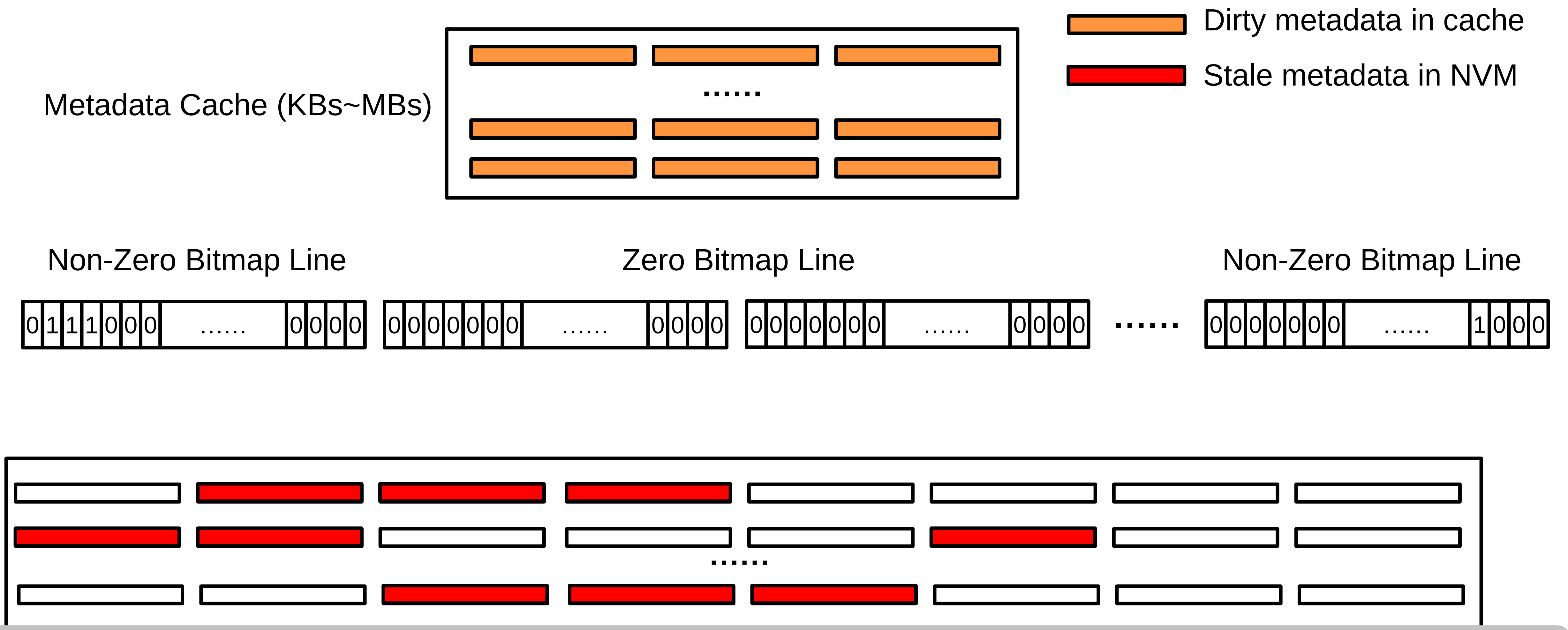
Detect the attacks during recovery

Bitmap Lines



32KB continuous metadata lines (512x64B=32KB)

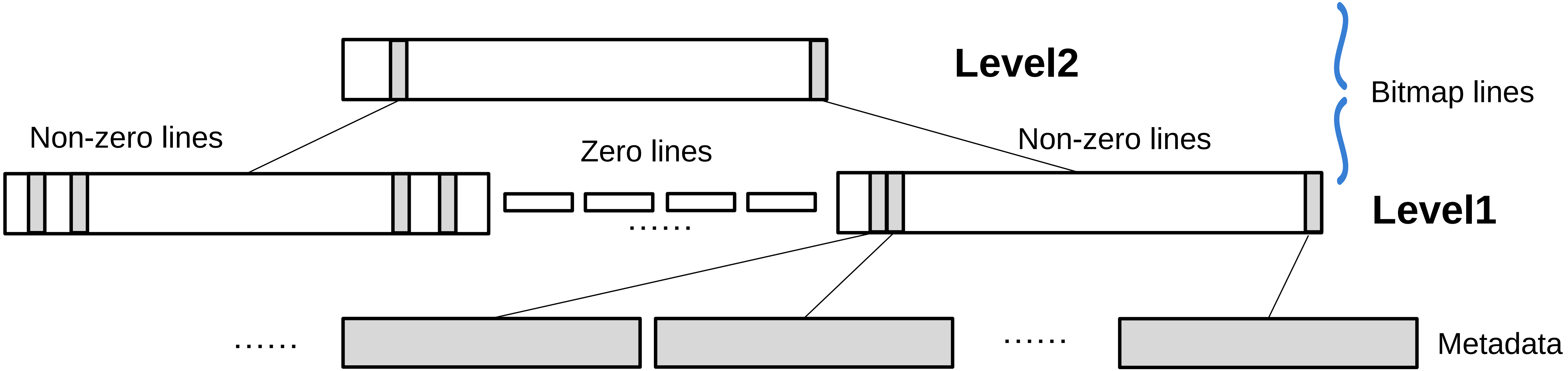
Bitmap Lines



Reading zero bitmap lines is useless to locate stale metadata

Multi-layer Index

Only reading non-zero bitmap lines



Indicating the non-zero bitmap lines and stale metadata

STAR Components

- Counter-MAC synergization

Persist the modifications w/o extra writes

- Bitmap lines

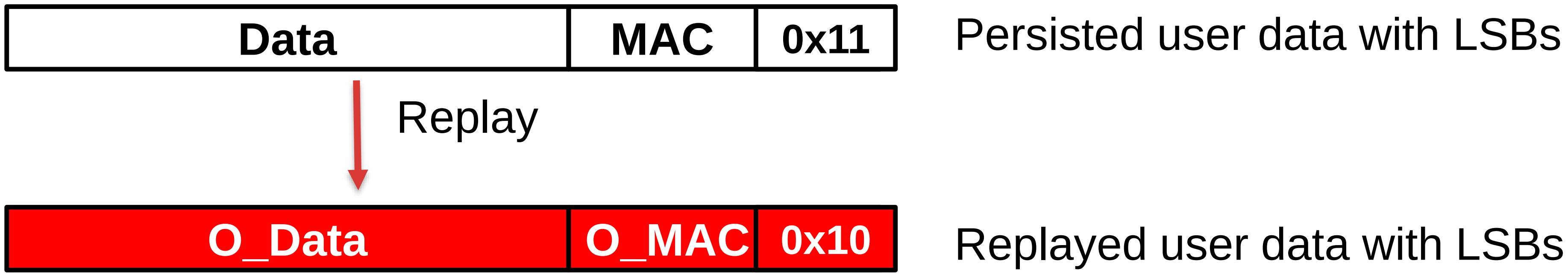
Record the locations of stale metadata for reducing recovery time

- Cache tree

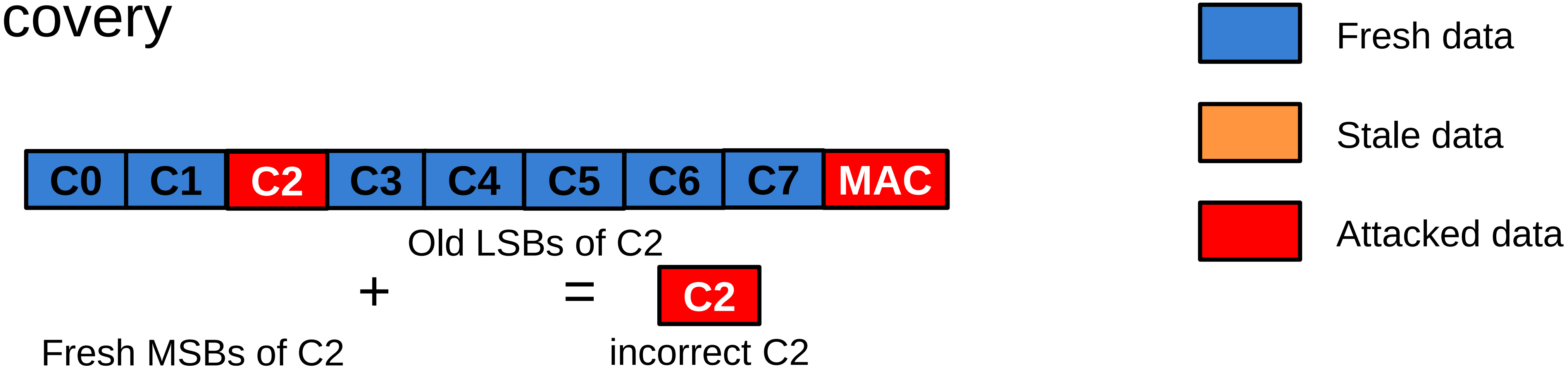
Detect the attacks during recovery

Attacks during recovery

- Replay attack

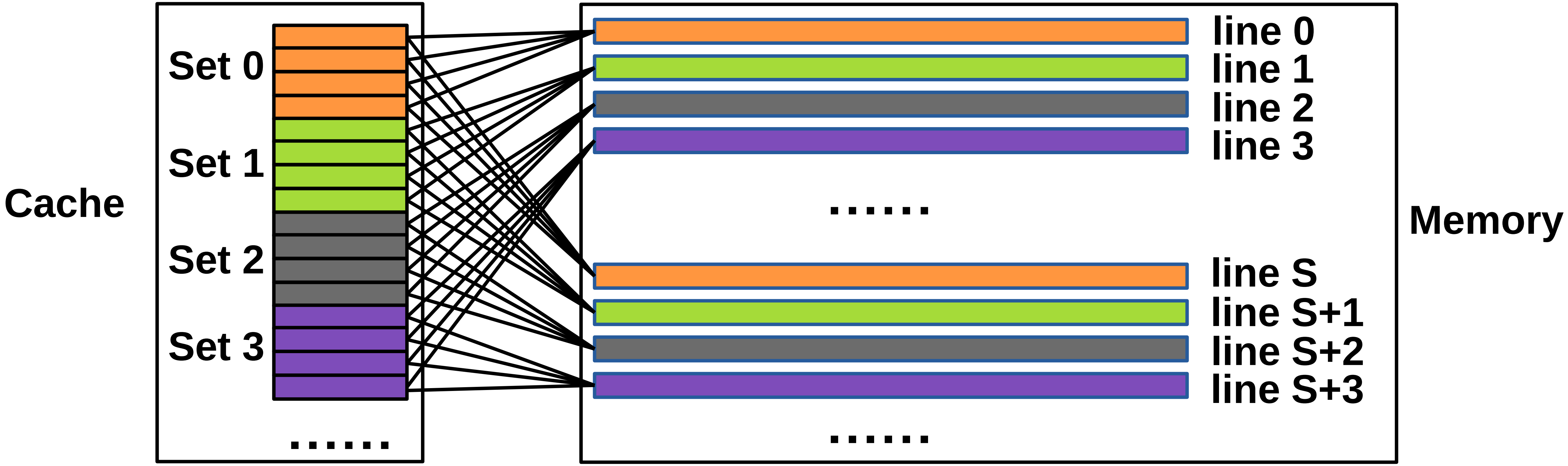


- Incorrect recovery



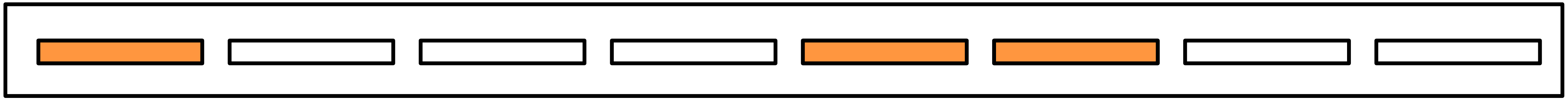
Attacks can't be detected in traditional integrity verification schemes

Cache Structure

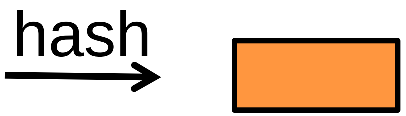


Cache tree

Set-Way cache: 8 ways in a set



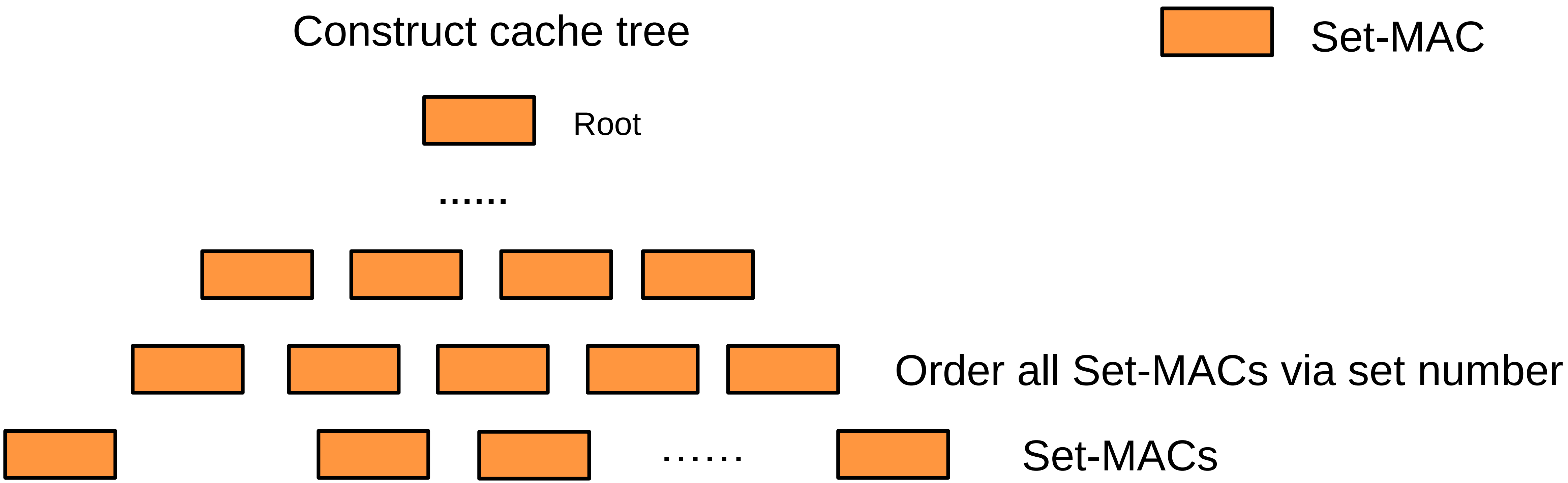
Logically order the dirty lines via the ascending addresses



Set-MAC

Generate the Set-MAC via dirty lines in the set

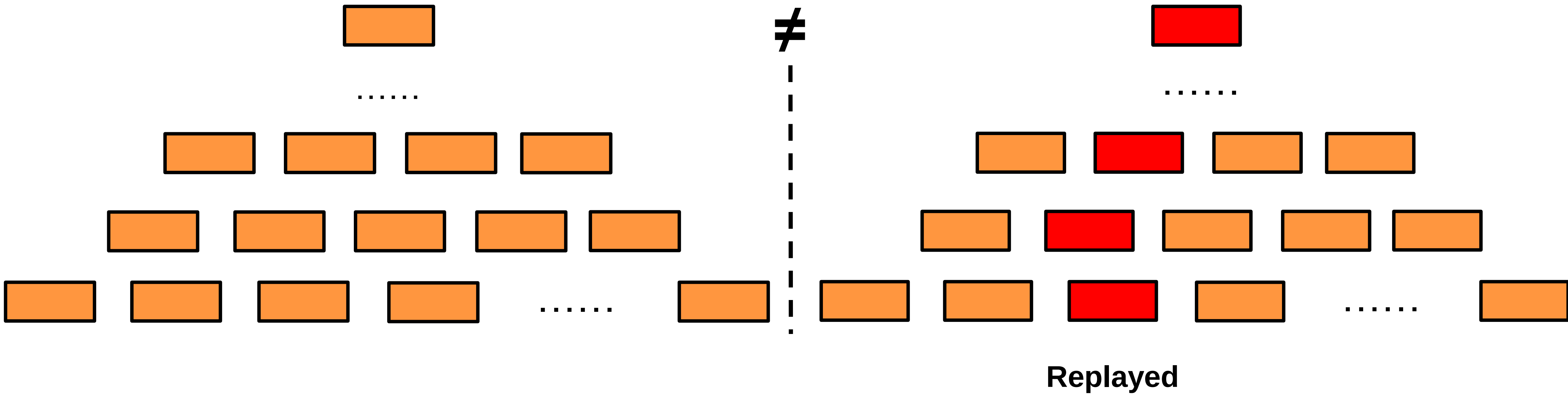
Cache tree



Construct cache tree
Set-MAC
Root
......
Order all Set-MACs via set number
......
Set-MACs

Cache tree

During recovery, we reconstruct the cache tree to detect the attacks



Outline

- Background and Motivation
- STAR Mechanism
- **Evaluation**
- Conclusion

Experimental Setup

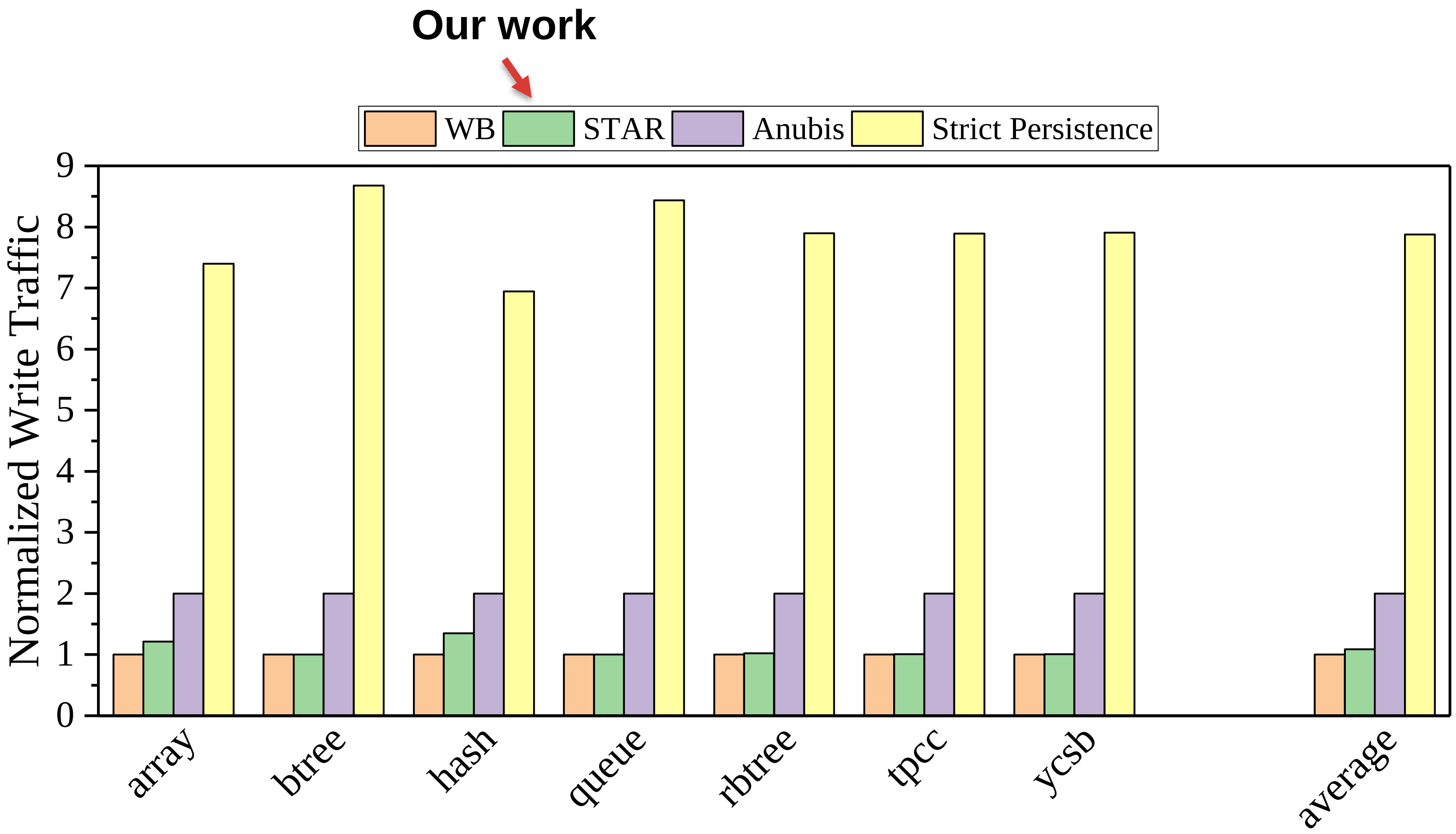
Gem5 + NVMain

Processor	8 cores(**2 GHz**); L1(**64 KB**), L2(**512 KB**), L3(**4 MB**) Caches
Memory Controller	Security Metadata Cache(**512 KB**); Bitmap Lines(**16 lines, 1 KB**)
NVM	16 GB; tRCD/tCL/tCWD/tFAW/tWTR/tWR =48/15/13/50/7.5/300 ns
Secure Parameters	SIT (**9 levels**); Cache Tree (**4 levels**)

Comparisons

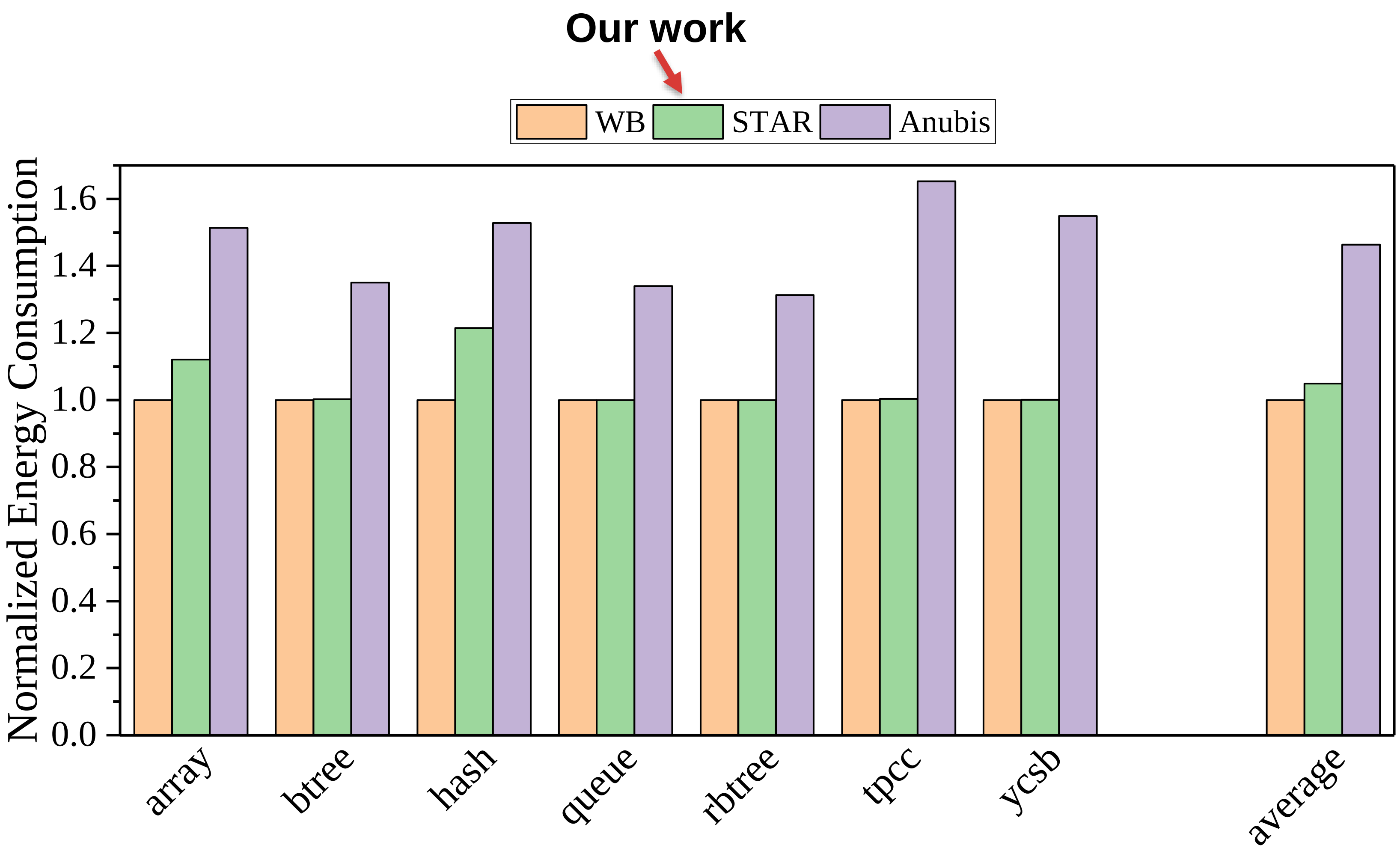
Write-back cache (WB)	Baseline, fail to recover system
STAR	**Our work**
Anubis[ISCA19]	1x extra memory writes
Strict Persistence	Persist all nodes in a branch of tree

Write traffic



SATR reduces 92% extra writes than Anubis

Energy Consumption



Compared with Anubis, SATR reduces 42% energy overheads

Outline

- Background and Motivation
- STAR Mechanism
- Evaluation
- **Conclusion**

Conclusion

Design Goal

- Correctly recovering the stale security metadata with low write overhead and short recovery time

Key Idea

- STAR disaggregates the persistence of modifications and addresses of metadata and provides recovery verification
  - Counter-MAC synergization : reduce memory writes
  - Bitmap lines: locate the stale metadata
  - Cache tree: verify the recovery process

Result

- STAR reduces 92% extra writes than Anubis and fast recovers the security metadata

Thanks! Q&A

Email: jmhuang@hust.edu.cn