A High-Performance and Fast-Recovery Scheme for Secure Non-Volatile Memory Systems

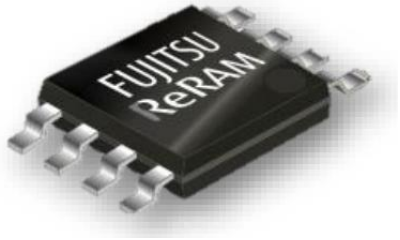
Yujie Shi, Yu Hua, Jianming Huang

Huazhong University of Science and Technology, China

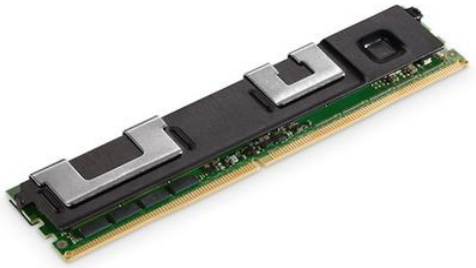
IEEE Cluster 2024

Non-Volatile memory

- Byte-addressability
- Non-volatility
- Low Latency
  - Similar to DRAM
- Large capacity
  - TB-scale



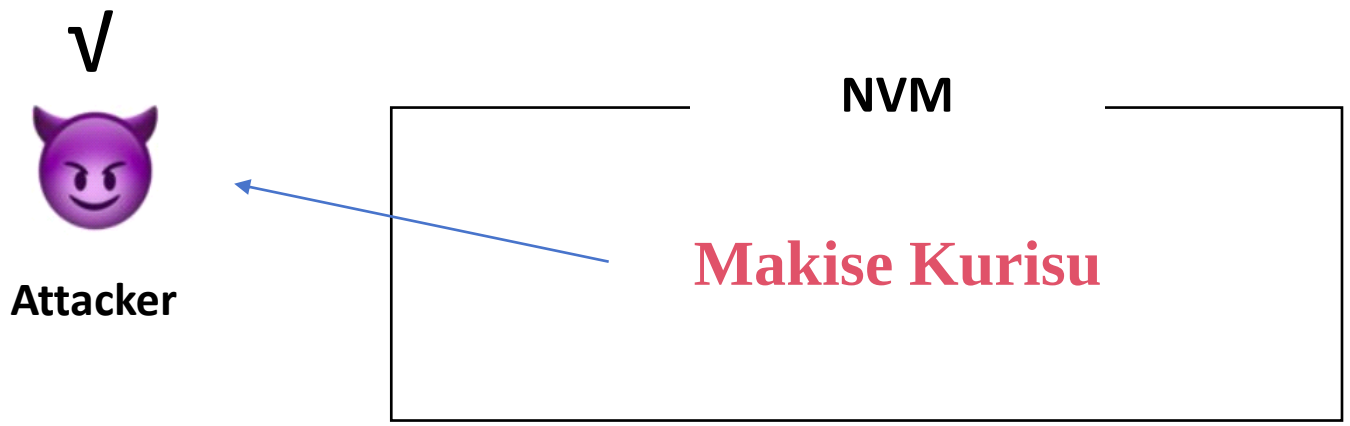
ReRAM



Intel Optane

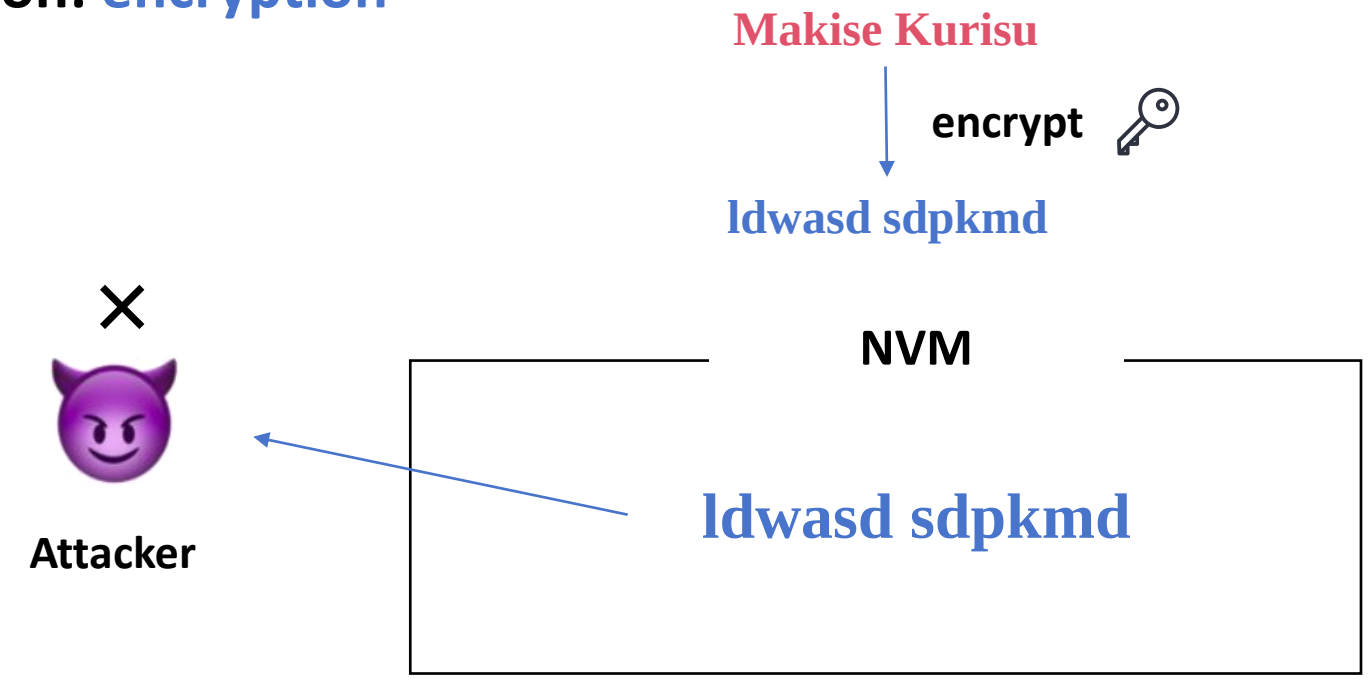
Data Security in NVM

- **Data Confidentiality**
  Attackers can steal plaintext



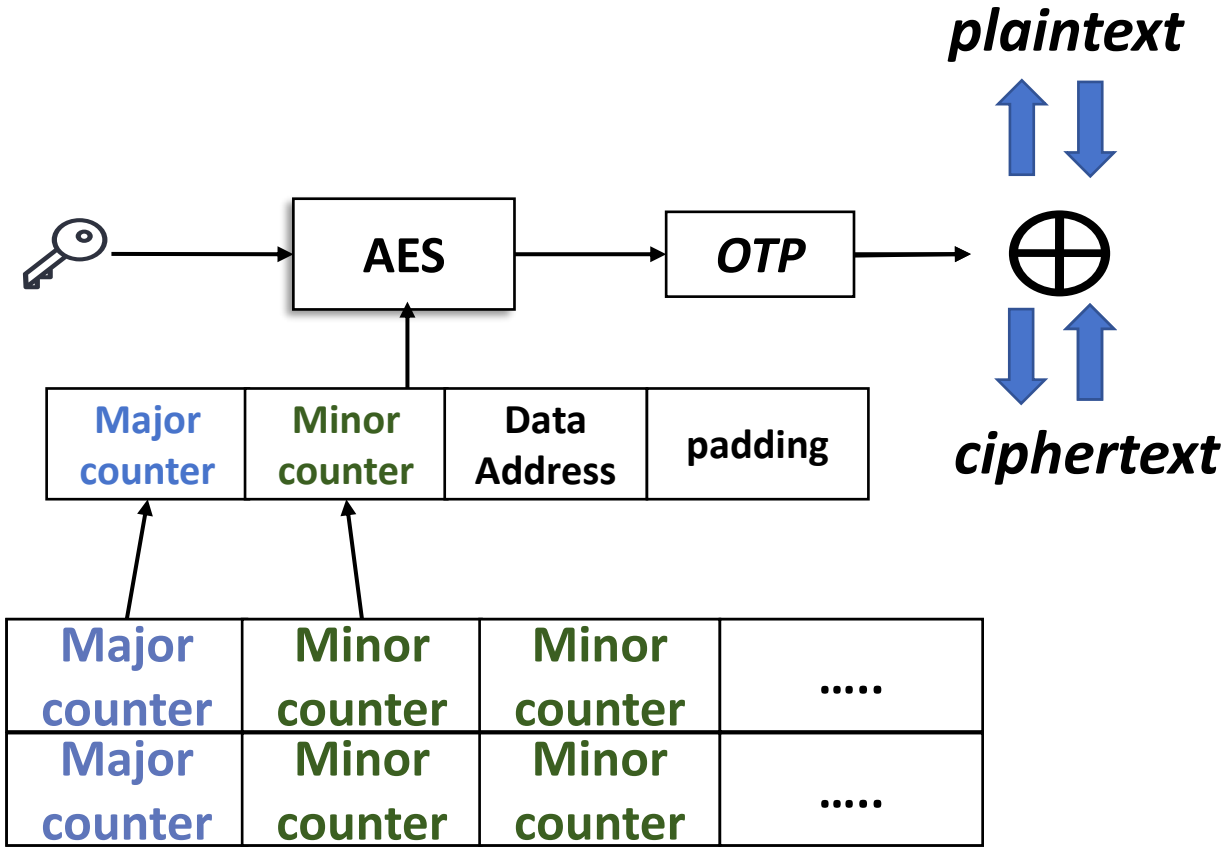
Data Security in NVM

- **Data Confidentiality**
  Attackers can steal plaintext
  Solution: encryption



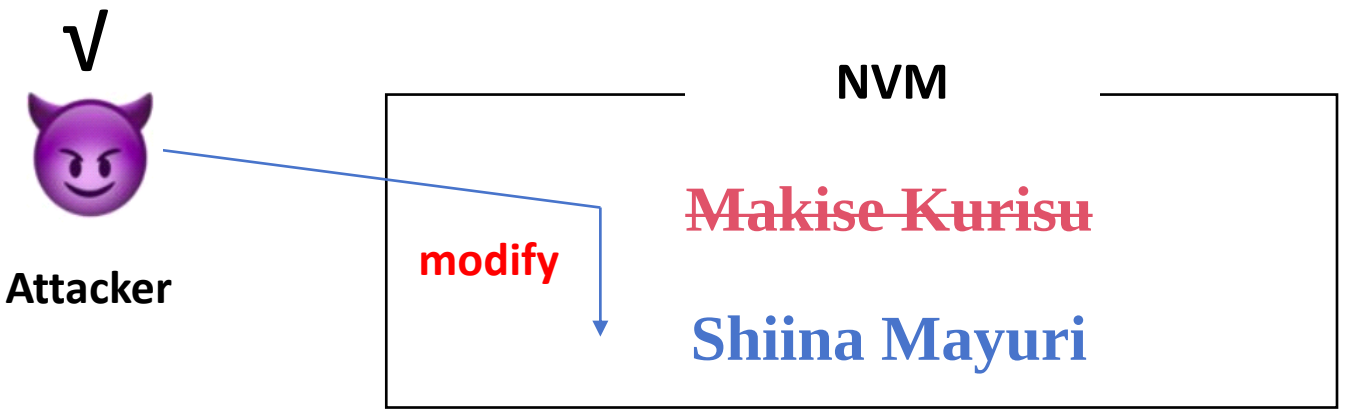
Data Security in NVM

- **Data Confidentiality**
  Counter Mode Encryption (CME)



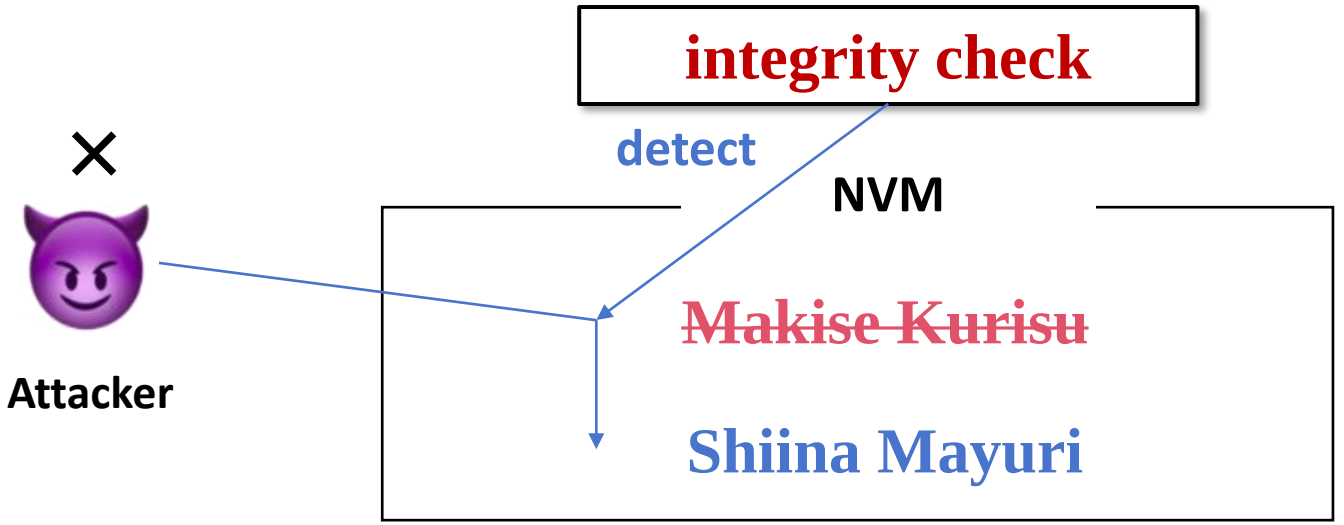
Data Security in NVM

- **Data Integrity**
  Attackers can tamper with/replay data



Data Security in NVM

- **Data Integrity**
  Attackers can tamper with/replay data
  Solution: integrity check

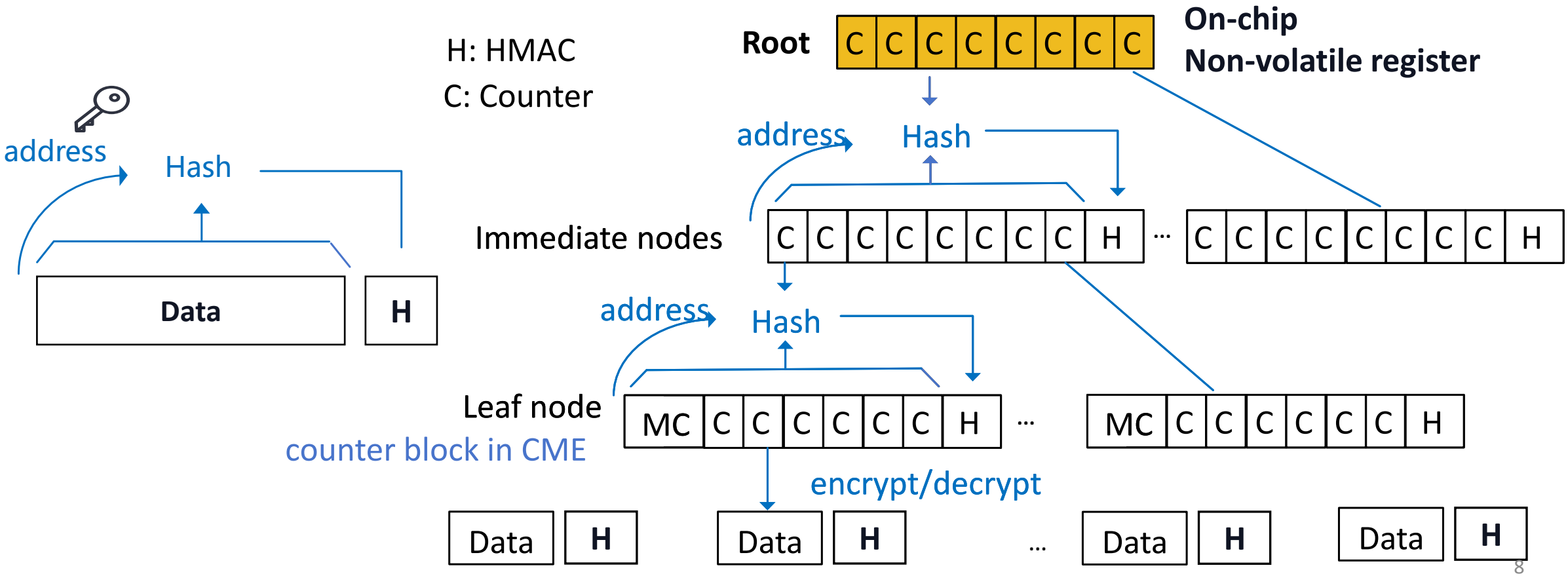


Data Security in NVM

- **Data Integrity**

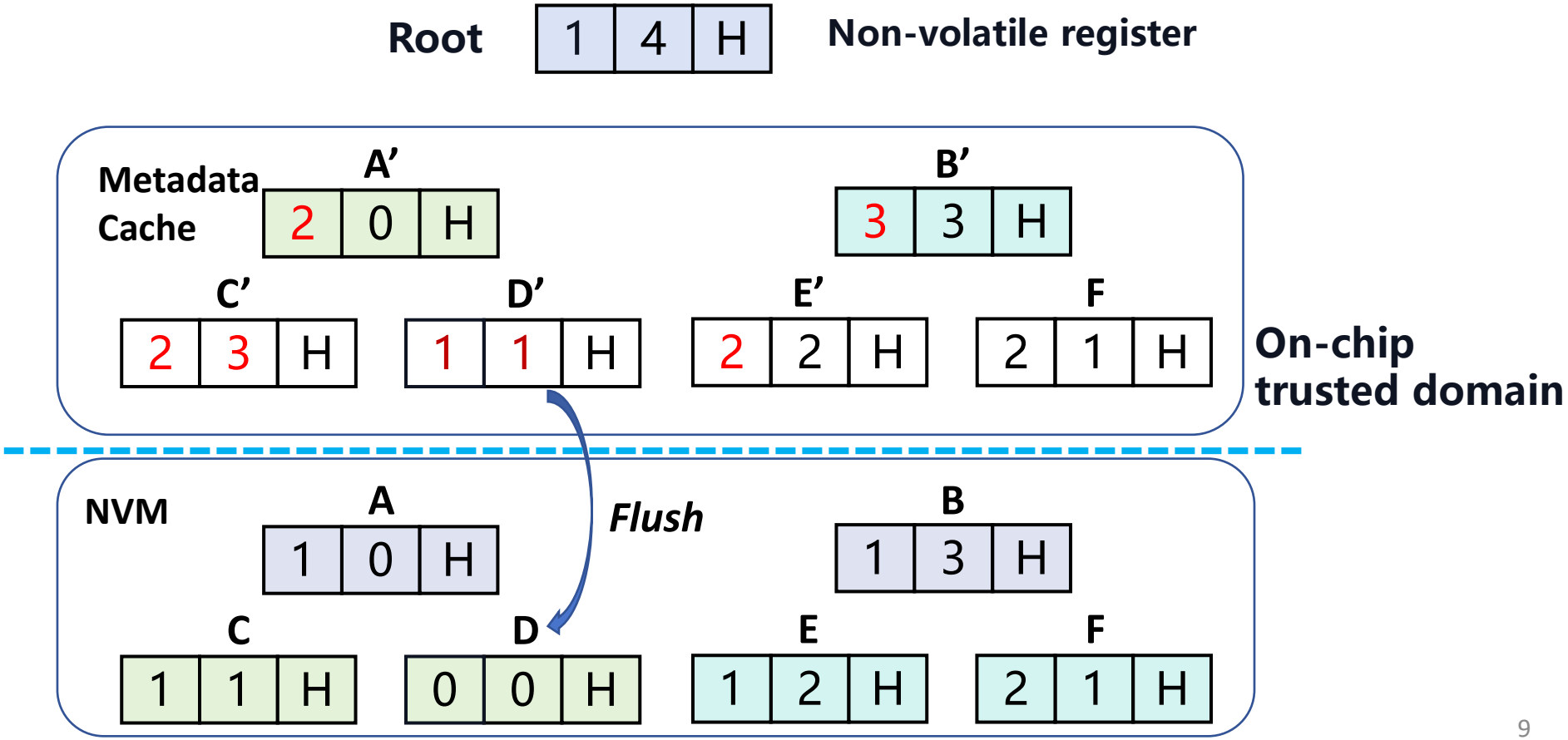
  HMAC: detect tampering attack

  Integrity tree (SGX-style integrity tree (SIT)): detect replay attack



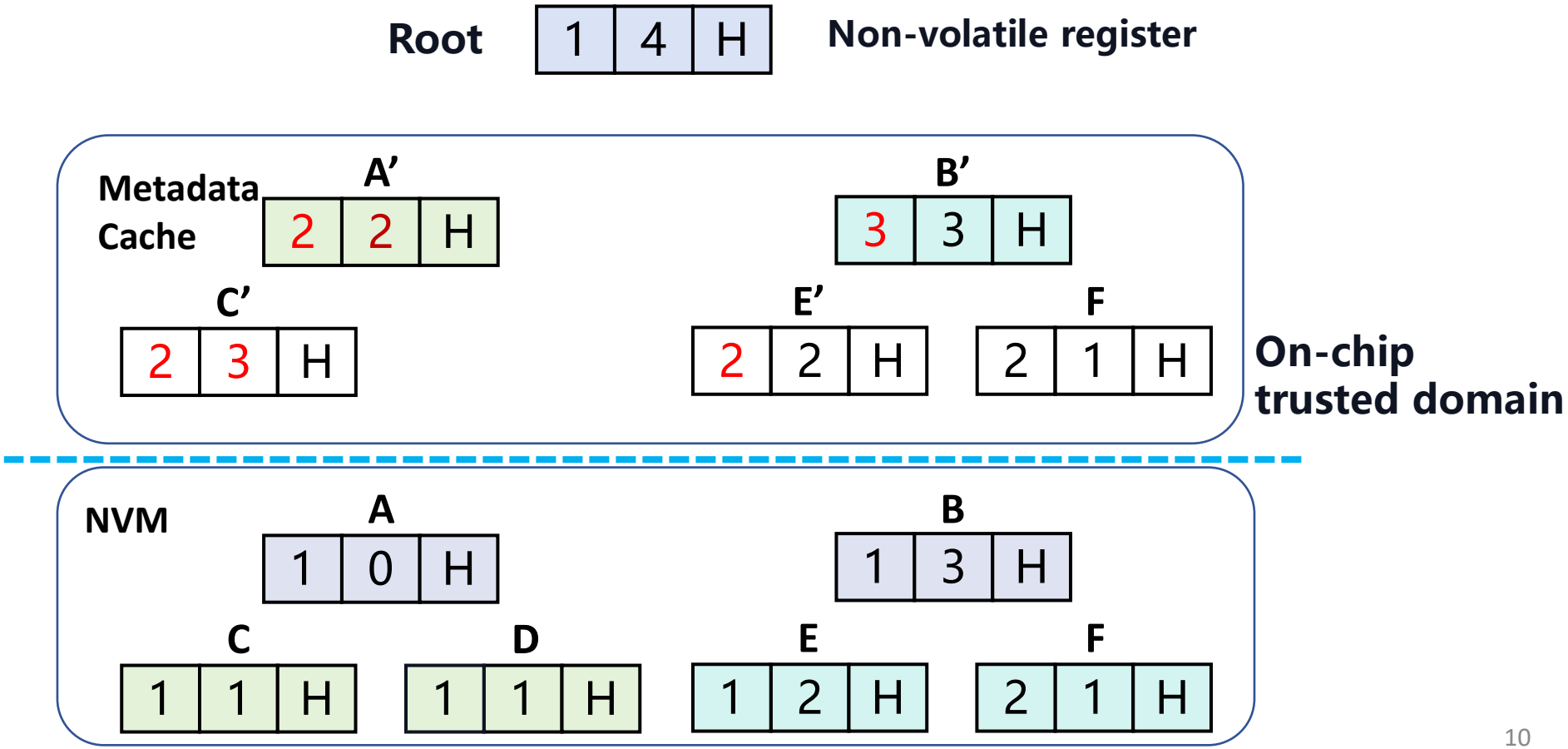
High performance NVM systems

- Security metadata are cached in the memory controller(MC)
- Lazy update scheme
  - Only update parent node when flushing child node



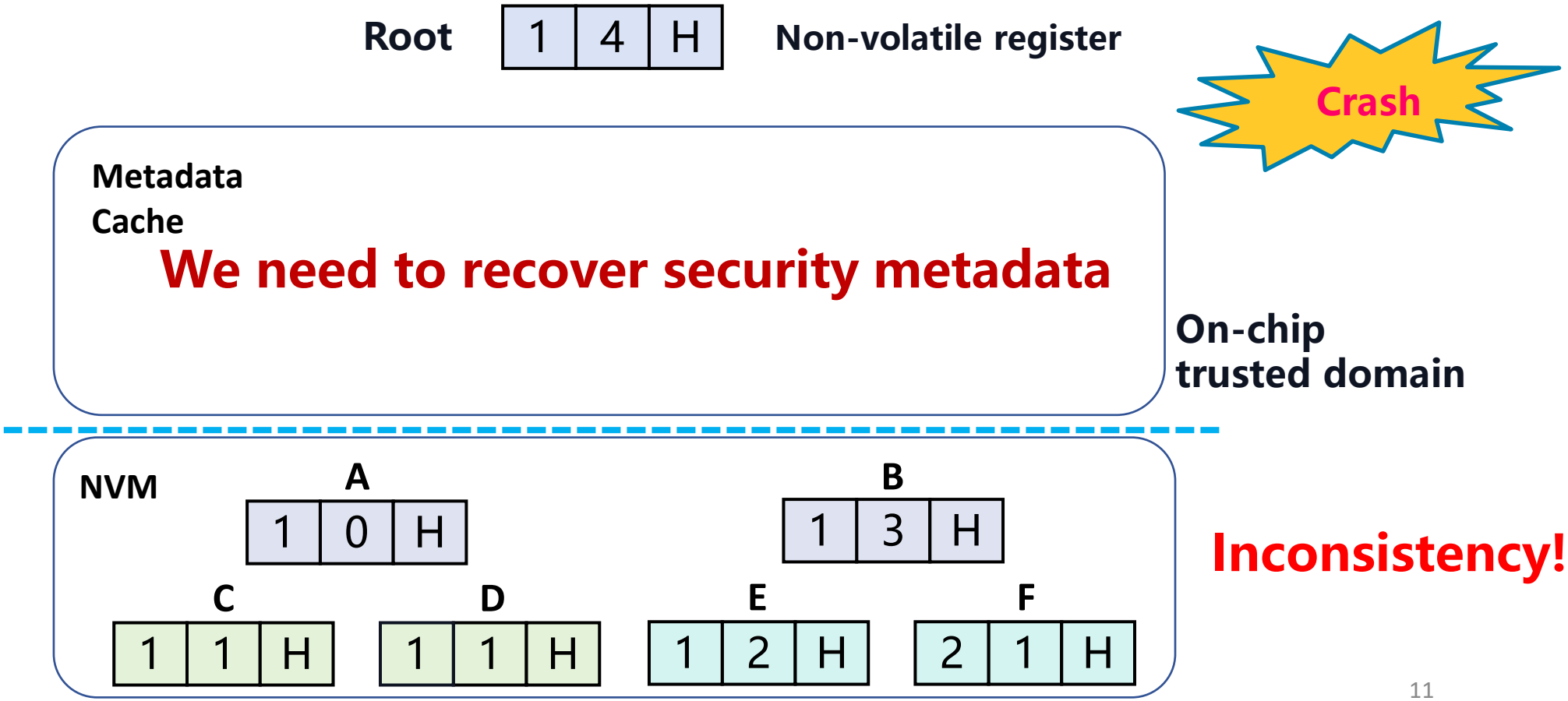
High performance NVM systems

- Security metadata are cached in the memory controller(MC)
- Lazy update scheme
  - Only update parent node when flushing child node



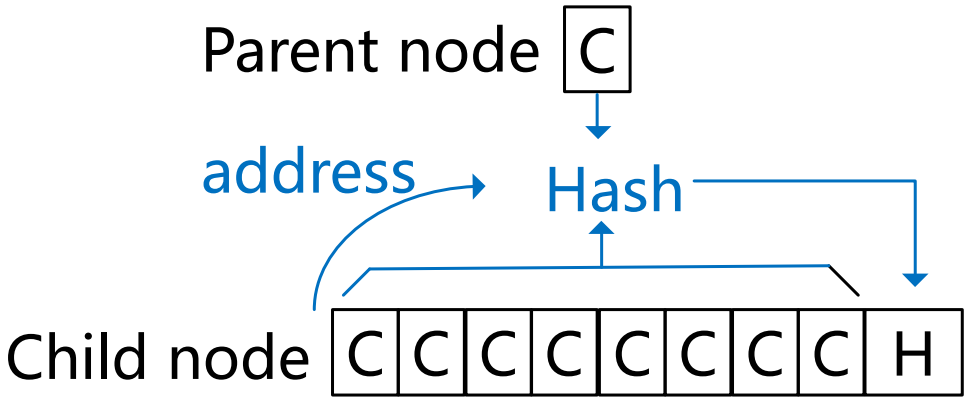
High performance NVM systems

- Security metadata are cached in the memory controller(MC)
- Lazy update scheme
  - Only update parent node when flushing child node



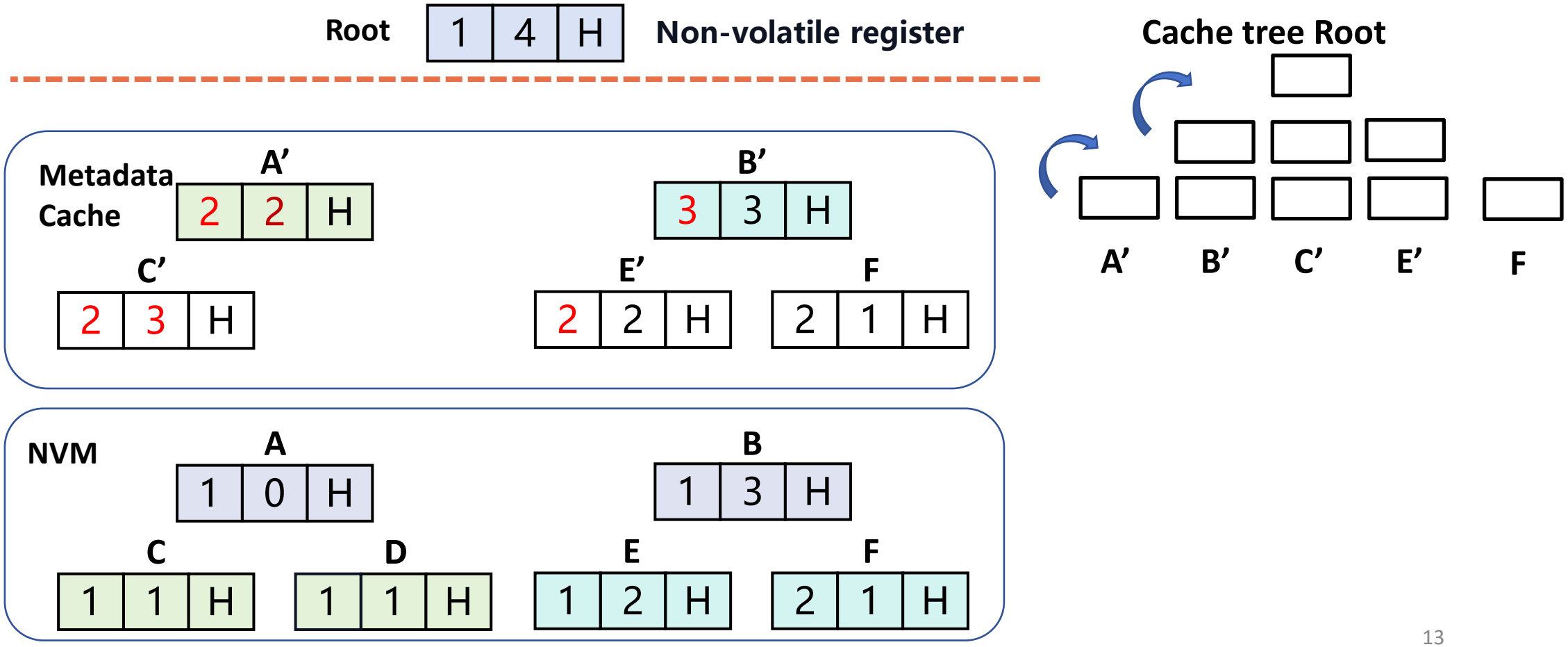
Challenge1: recover the stale metadata

- **Counter cannot be generated from child nodes**
- **HMAC calculation requires the counter of parent node as input**



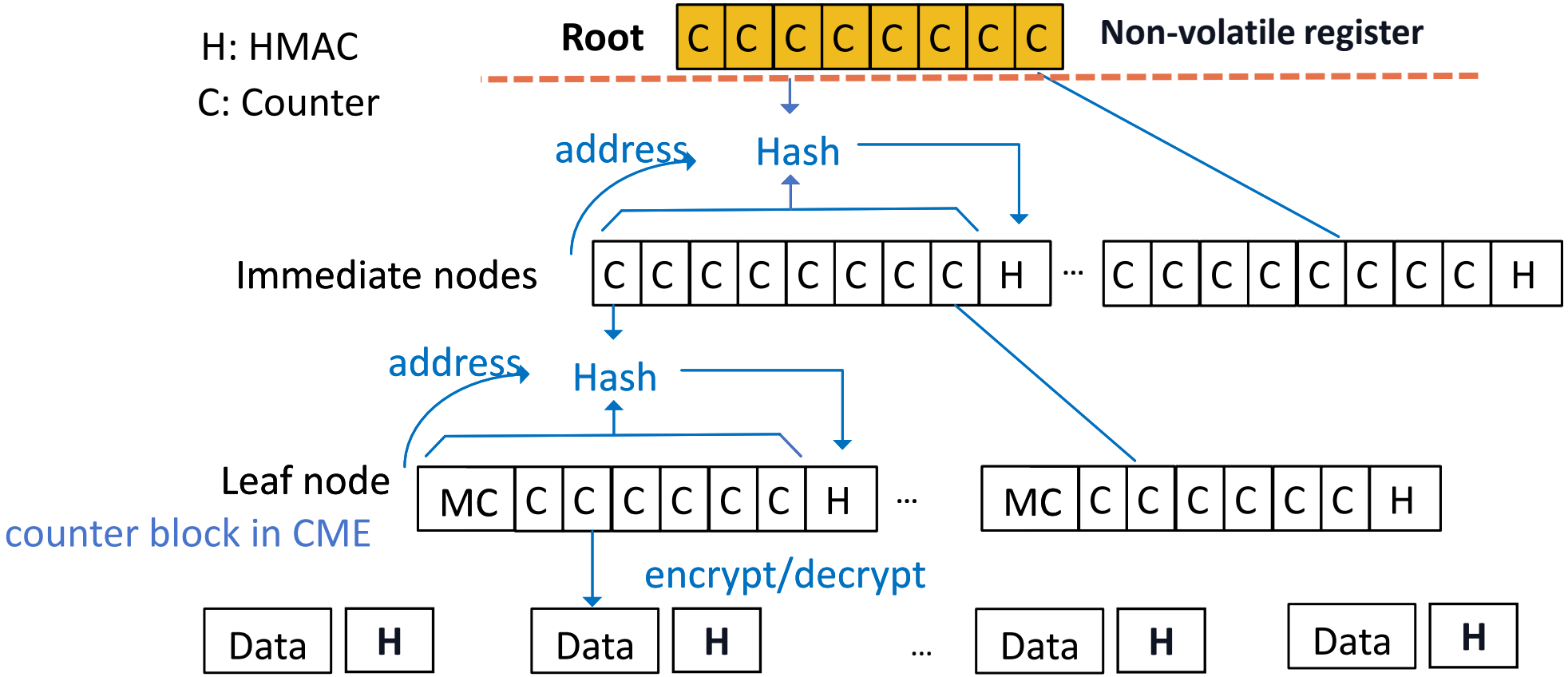
Challenge2: verify the retrieved metadata

- Root is inconsistent with the whole tree
- STAR[HPCA@21] / Anubis[ISCA@19]: cache tree (low performance)



Challenge3: recover the stale metadata quickly

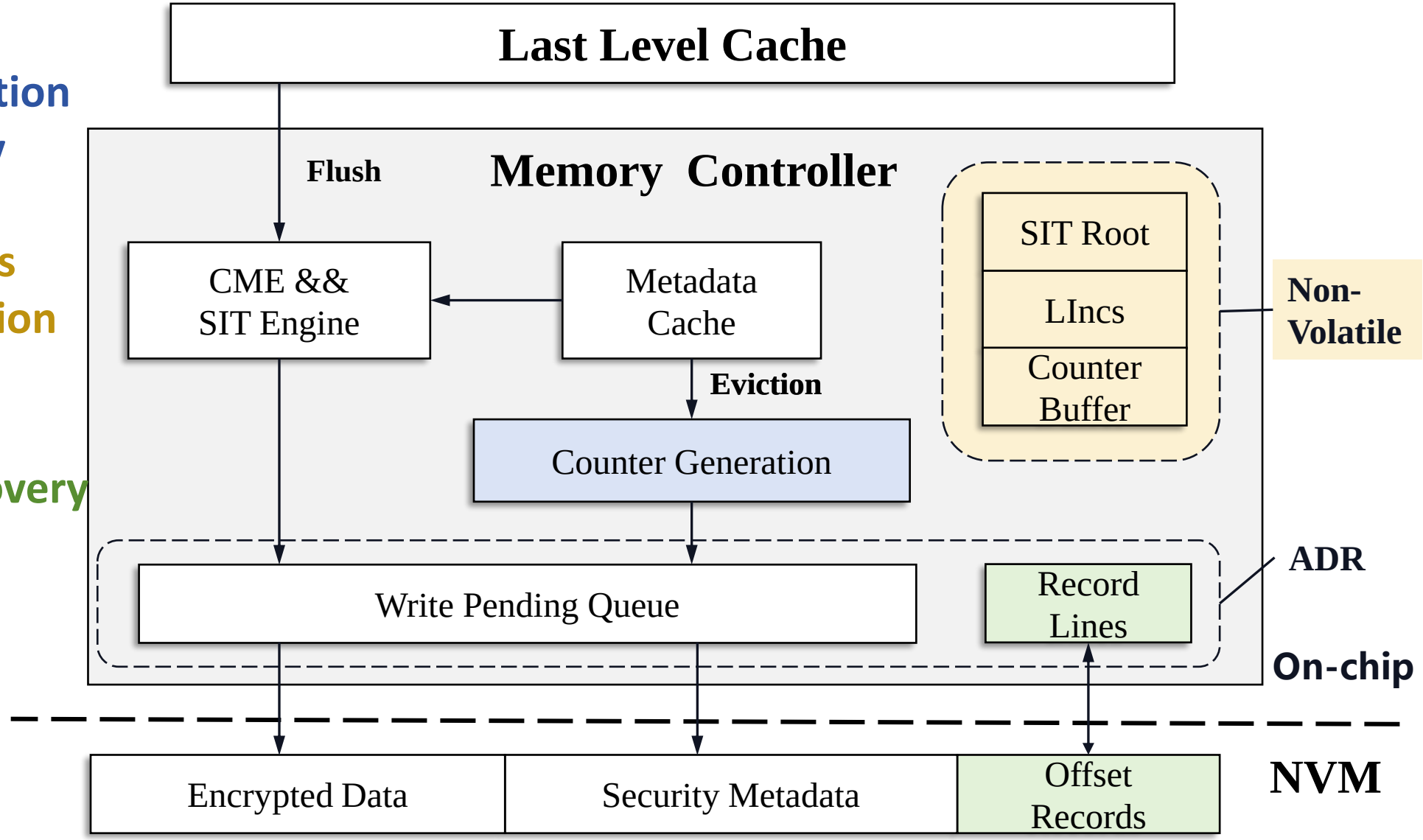
- **Recovering the whole tree requires hour-scale for TB-memory**



Bridge the gap between high performance and fast recovery

Steins Architecture

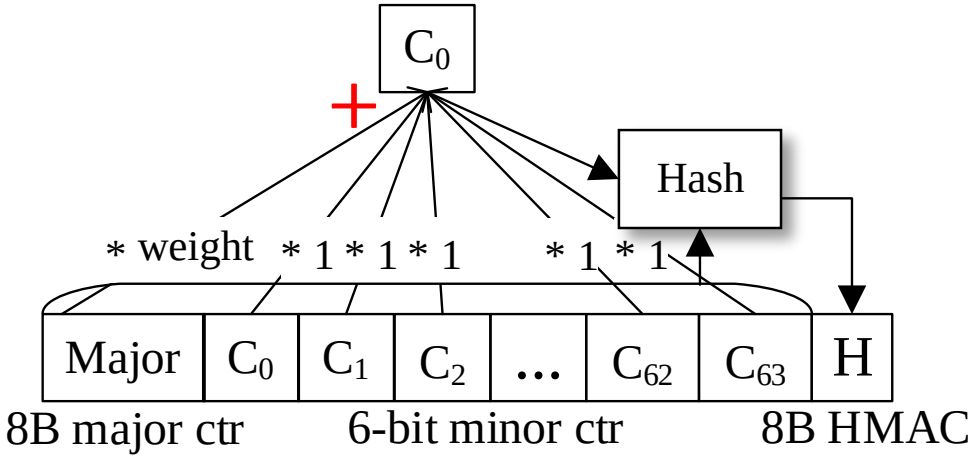
- Counter generation
  - For recovery
- Extra trust bases
  - For verification
- Data Tracking
  - For fast recovery

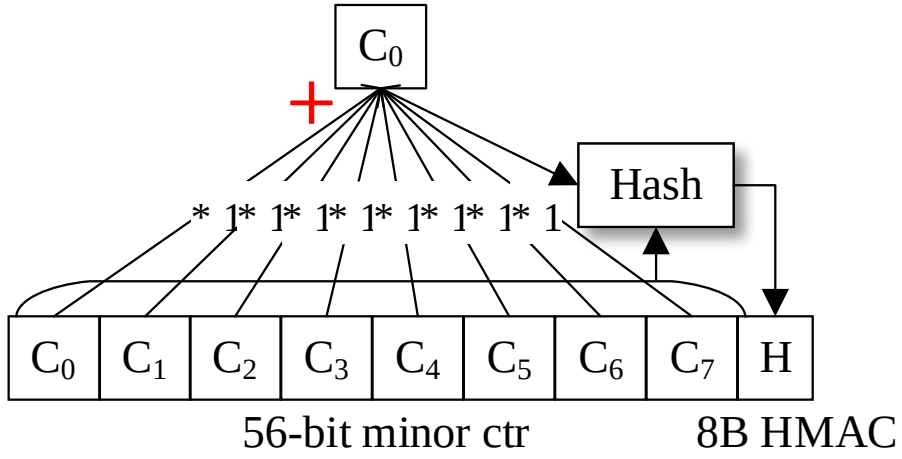


Counter generation scheme for recovery

- **Observation**
  - **Requirement for counter in SIT: never reused -> monotonically increasing**

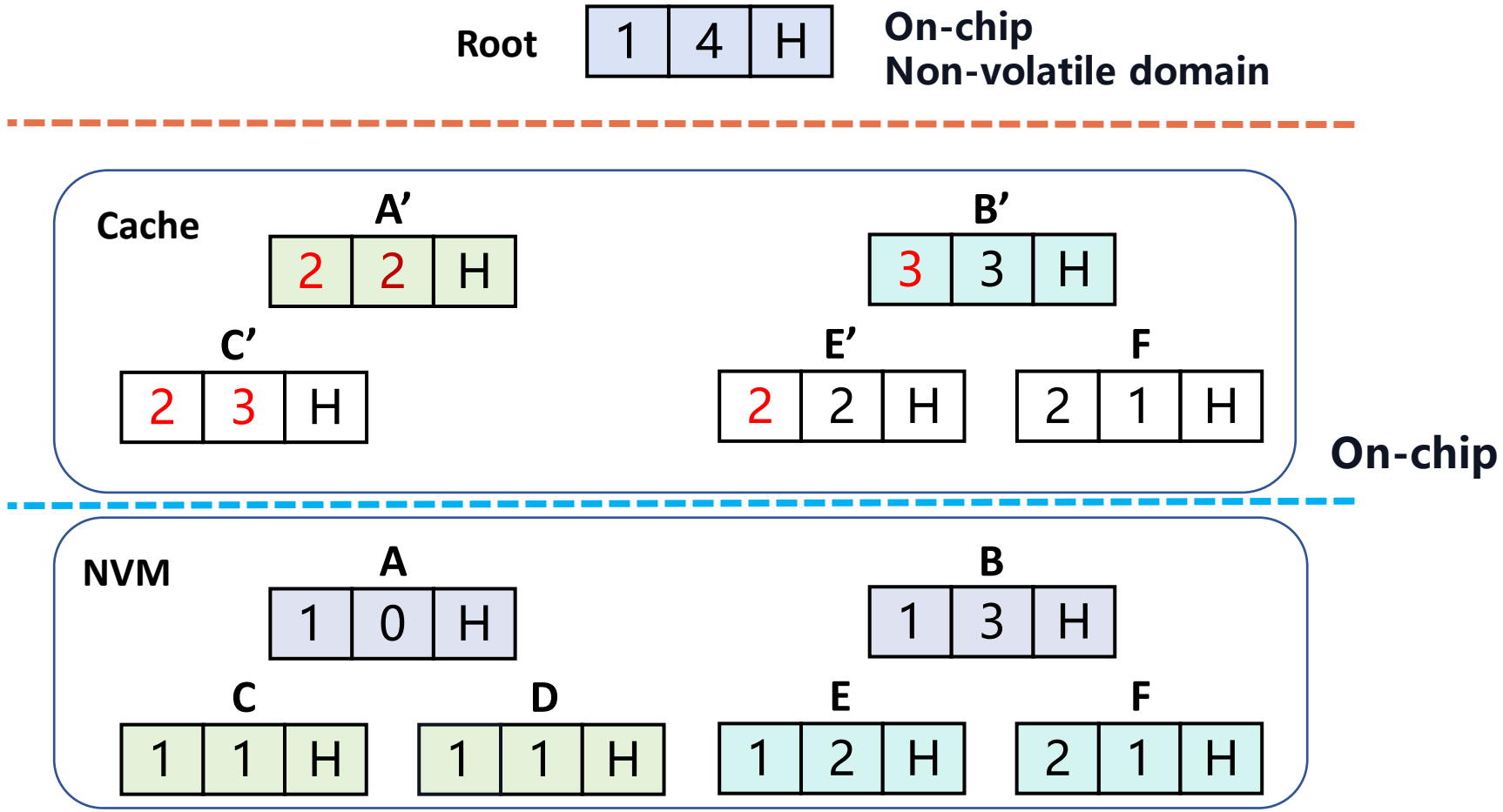
- **Generating the counter instead of using self-increasing counter**
  - **Recover from split counter scheme**
  - **Recover from general counter scheme**





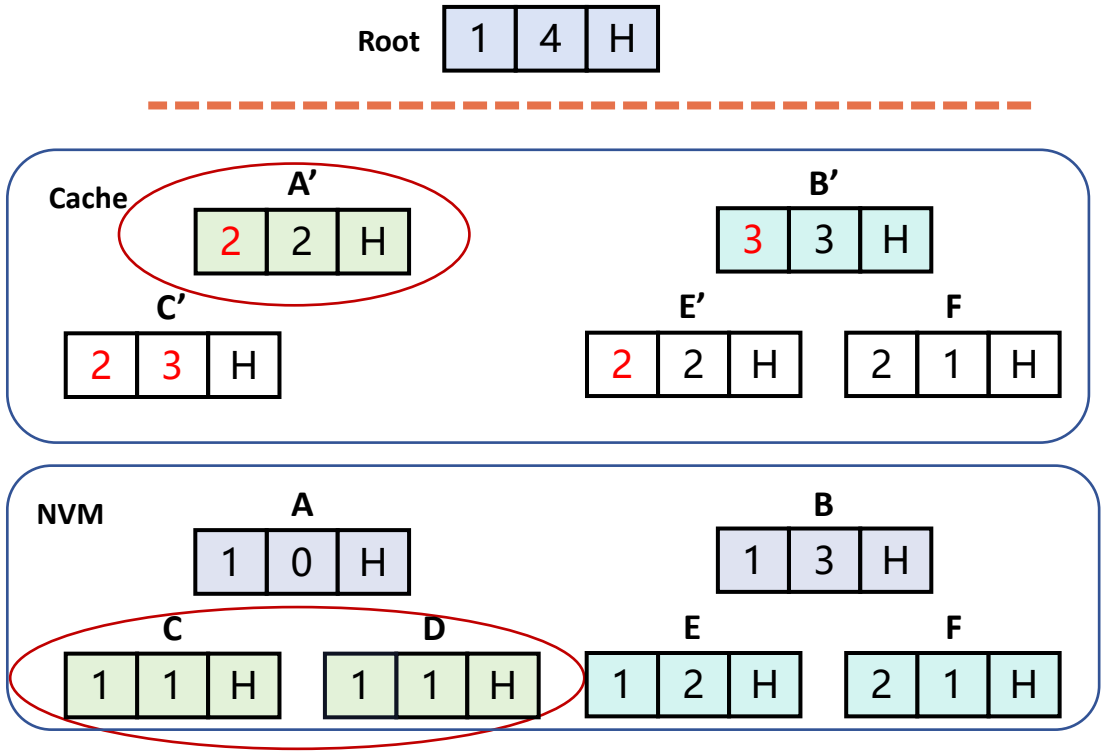
Verification during recovery

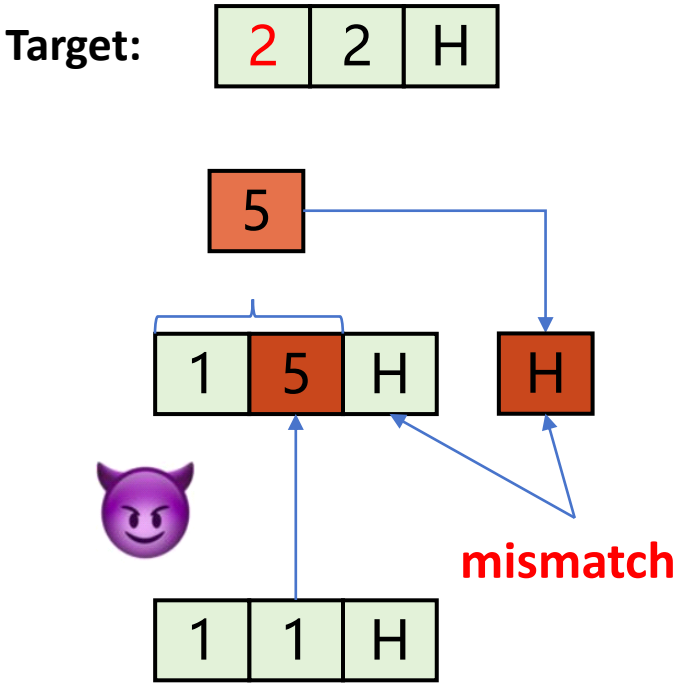
- Target
  - recover A’ B’ C’ E’ … correctly



Verification during recovery

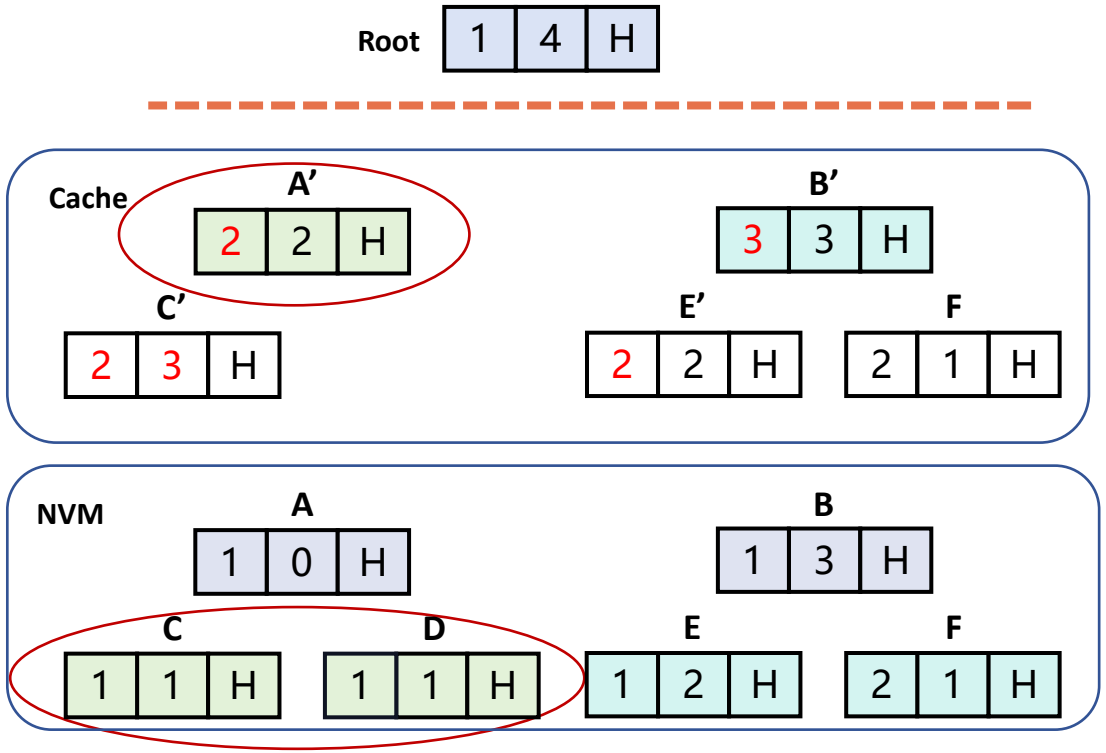
- Integrity attacks
  - Tampering attacks on child nodes: detected by HMAC

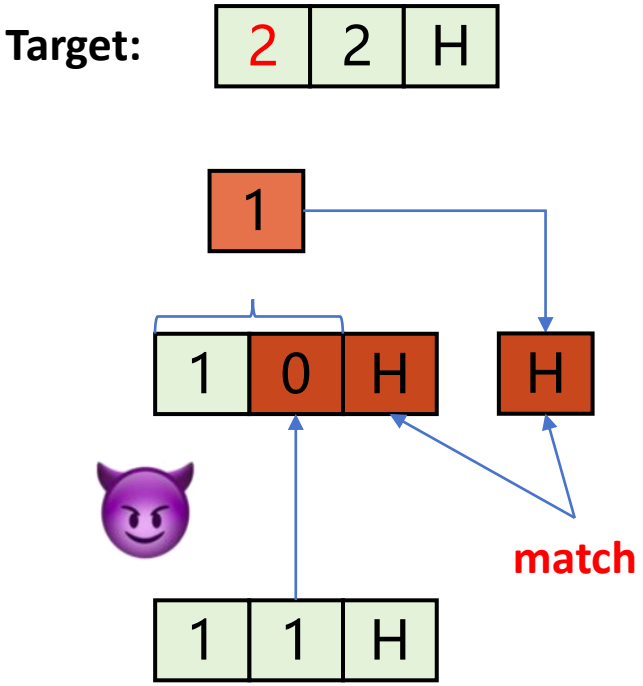




Verification during recovery

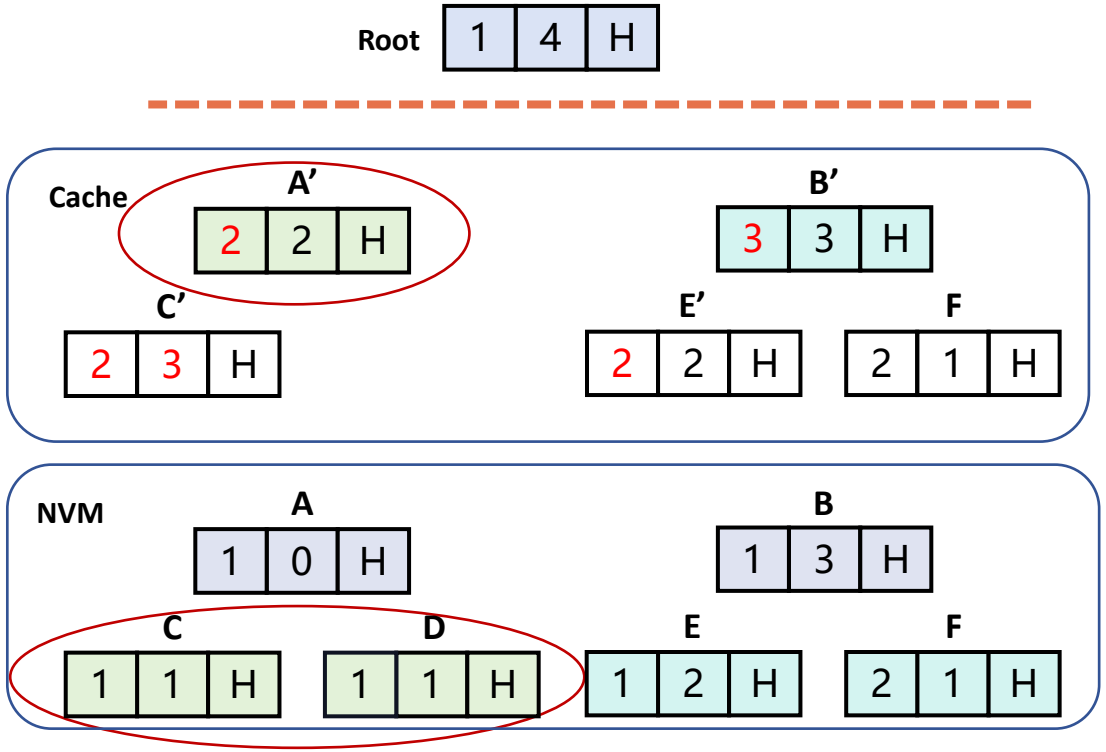
- Integrity attacks
  - Tampering attacks on child nodes: detected by HMAC
  - Attackers are limited to replaying child nodes

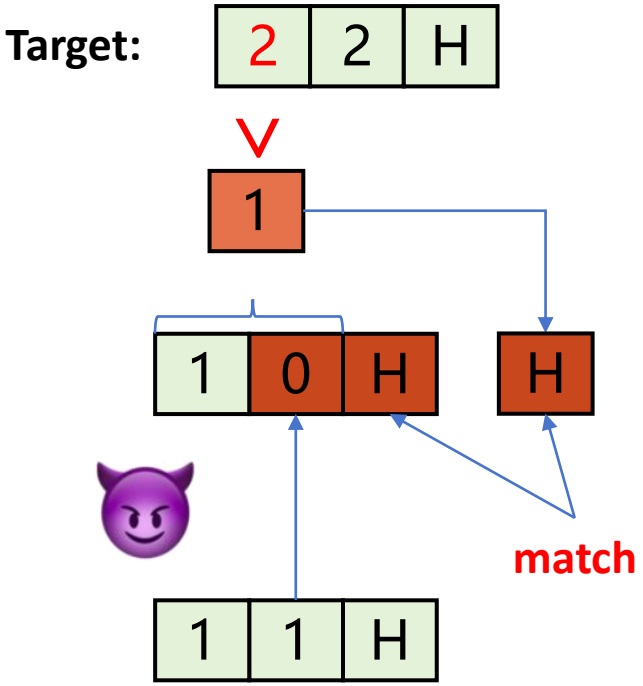




Detect replay attacks

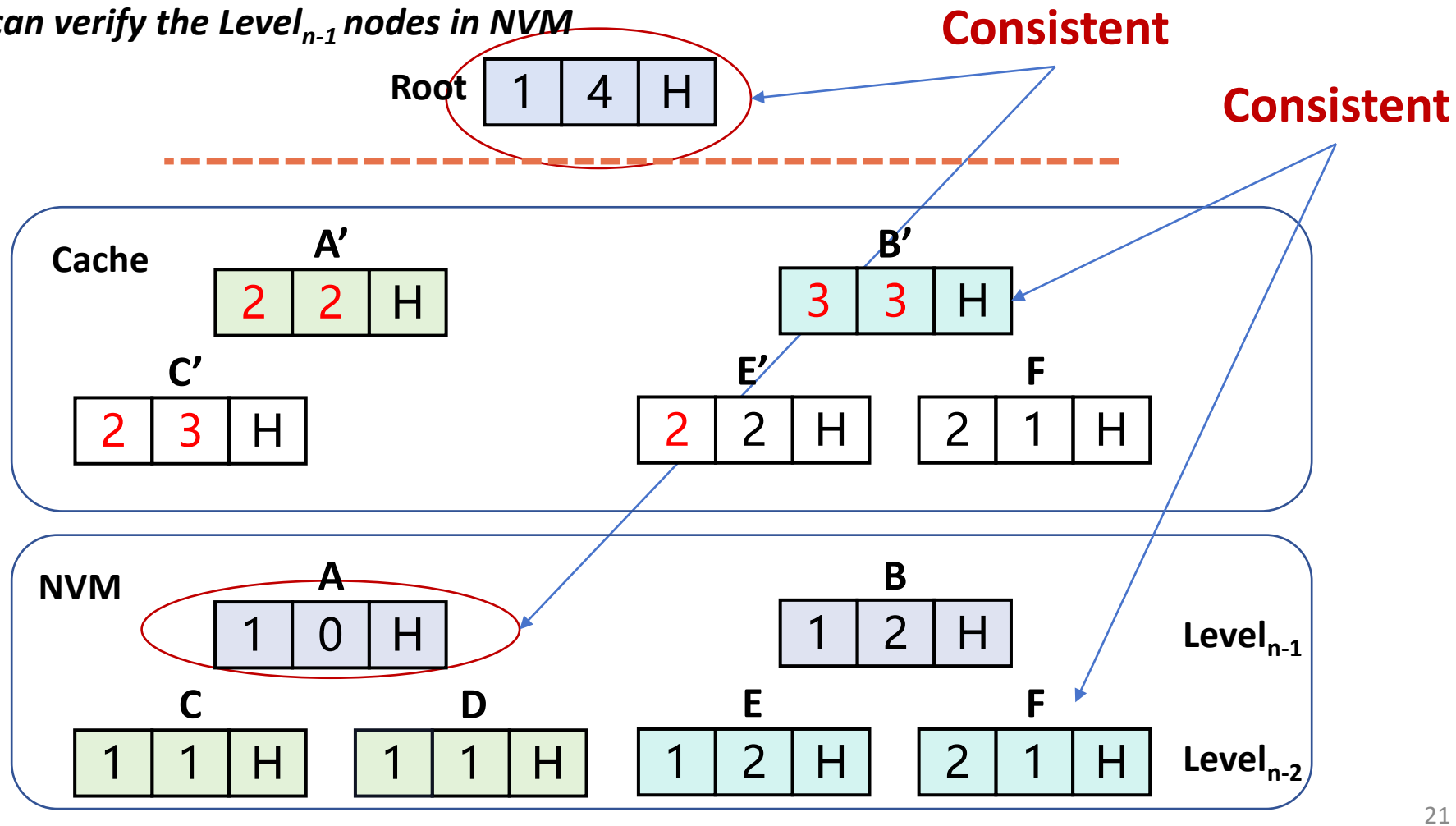
- Observation
  - Retrieved counter is smaller than the newest counter
    - The counter is monotonically increasing





Detect replay attacks

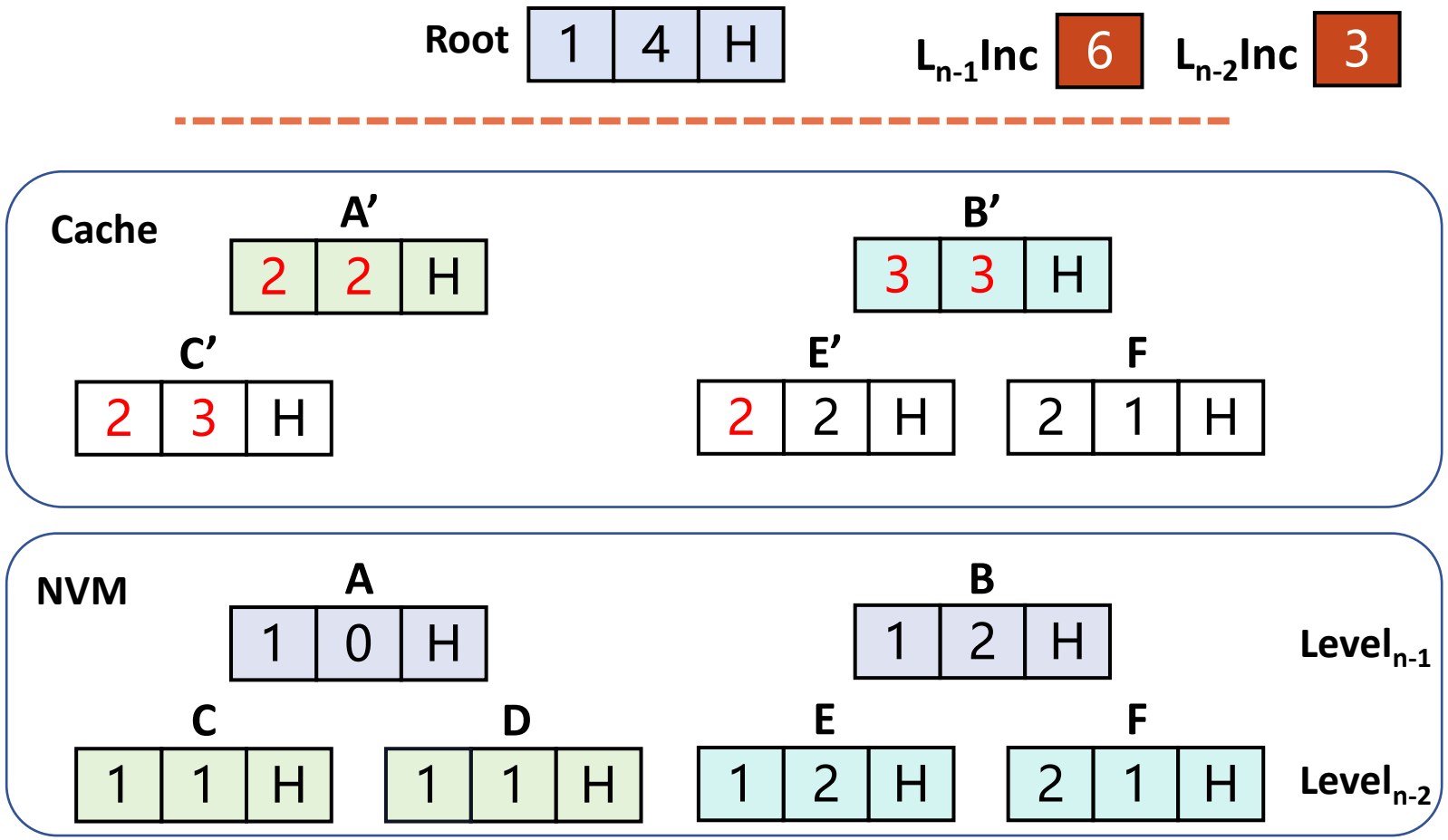
- **Observation**
  - **Retrieved counter is smaller than the newest counter**
  - **Node is consistent with its child nodes**
    - ***The root can verify the Level$_{n-1}$ nodes in NVM***



Detect replay attacks

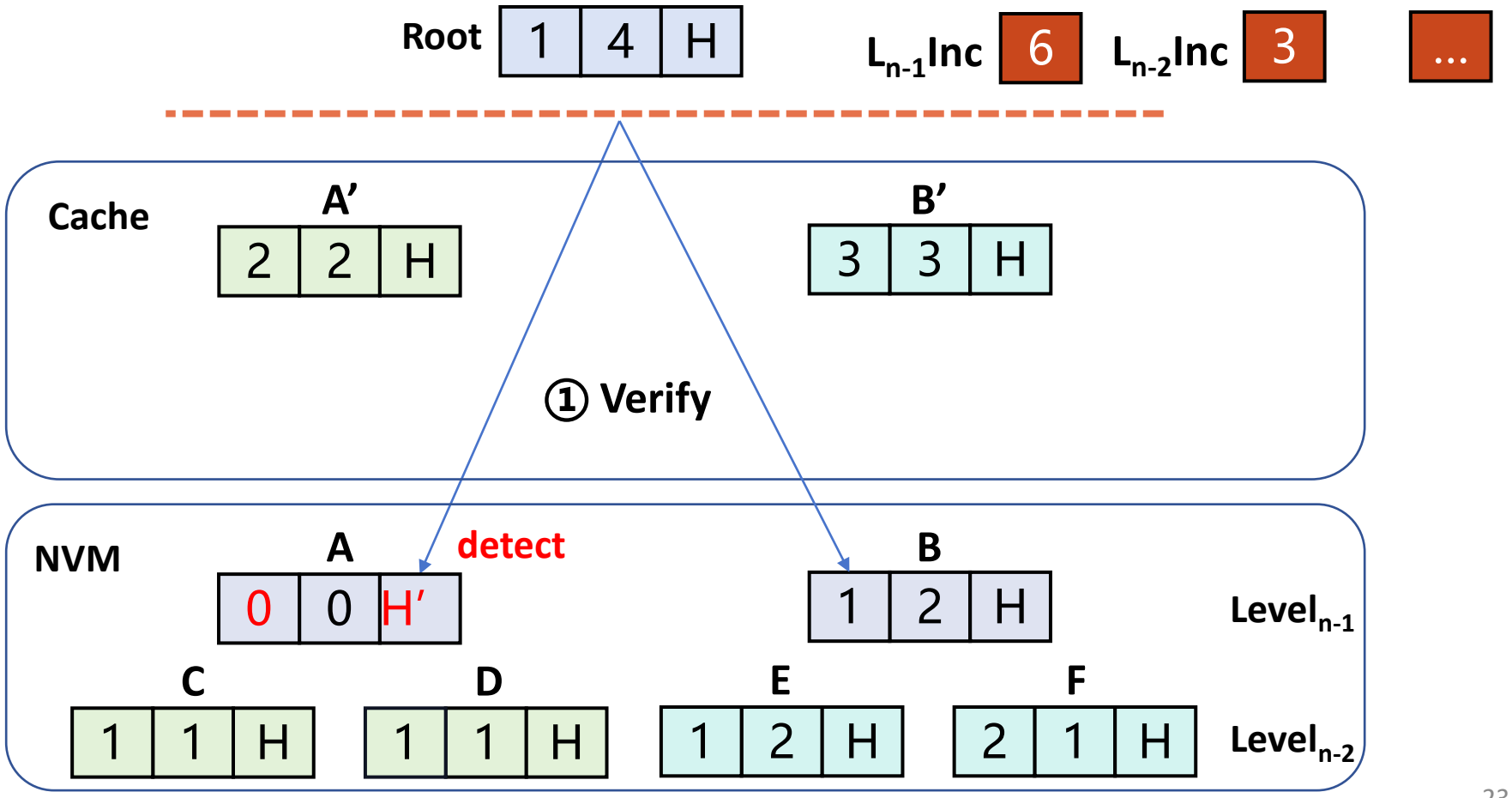
- Appropriate Trust Bases
  - Level Increases(LIncs)

The total increase of cached counters of dirty nodes over their counterparts in NVM for each level



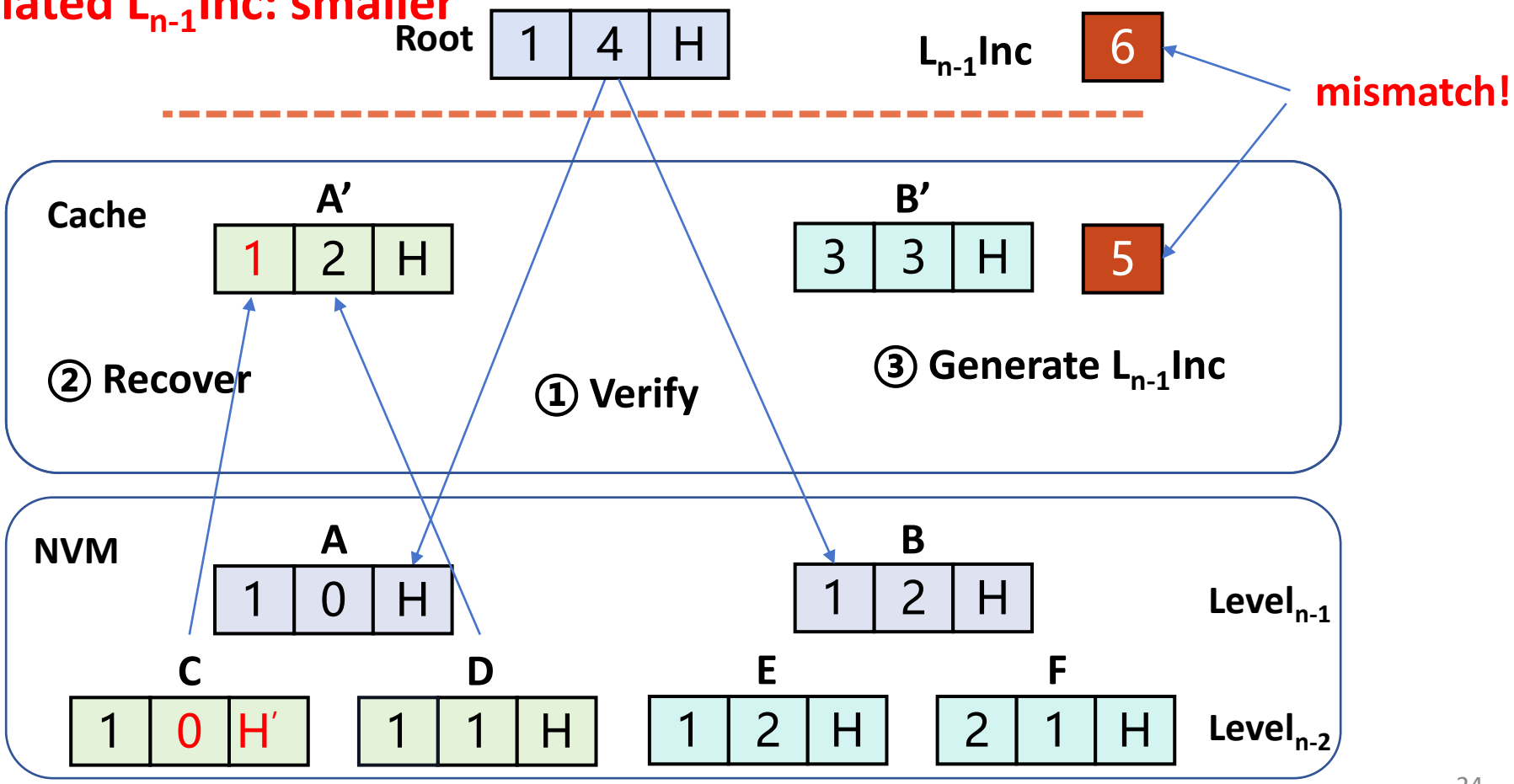
Detect replay attacks

- Recovery Processes(From root to leaf)
  - Replay A: detected by root



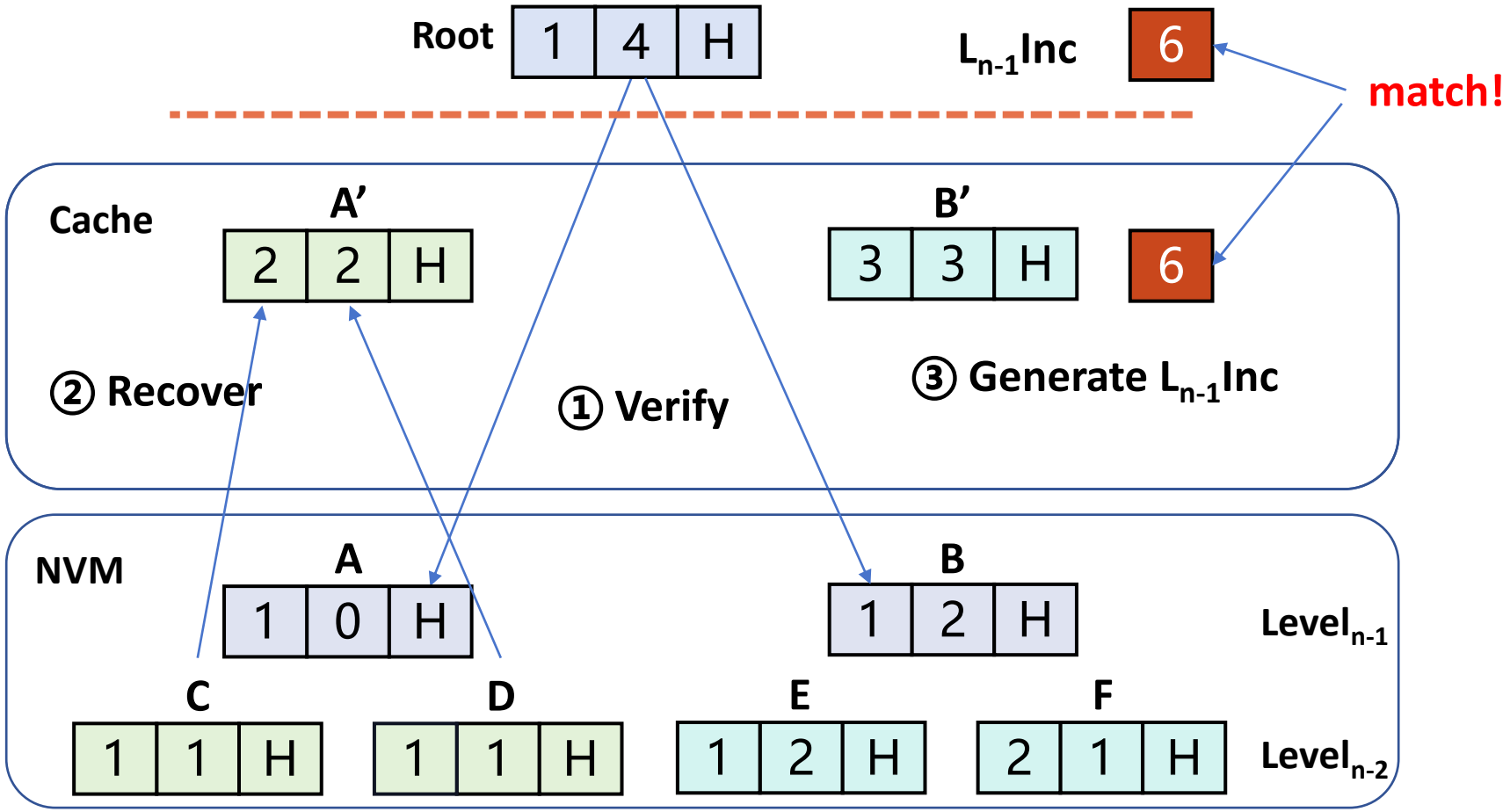
Detect replay attacks

- Recovery Processes(From root to leaf)
  - Replay A: detected by root
  - Replay C: A' becomes smaller
    - Recalculated L_{n-1}Inc: smaller



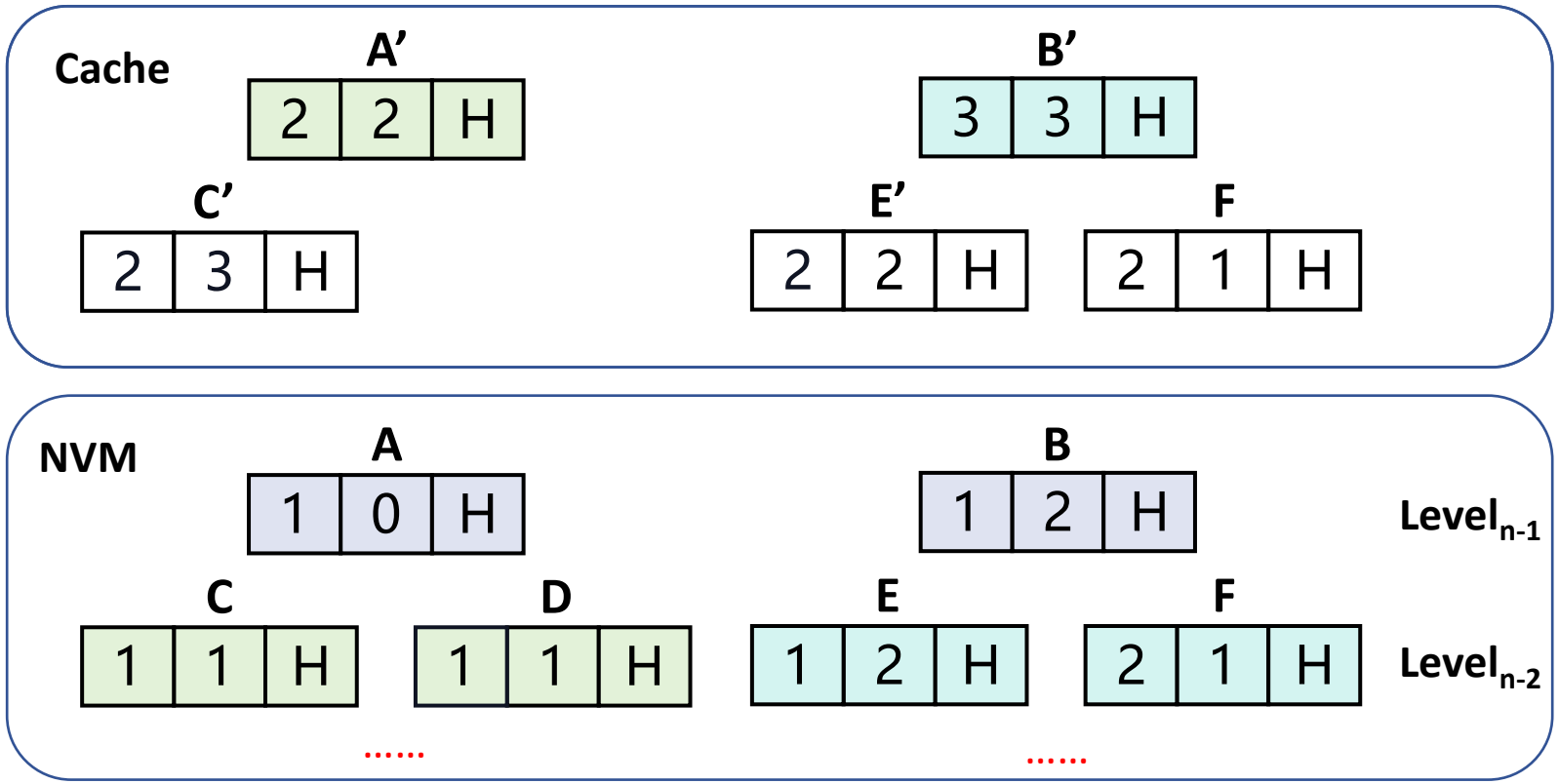
Detect replay attacks

- Recovery Processes(From root to leaf)
  - No attack:
    - Retrieved A’ and B’ can be trusted

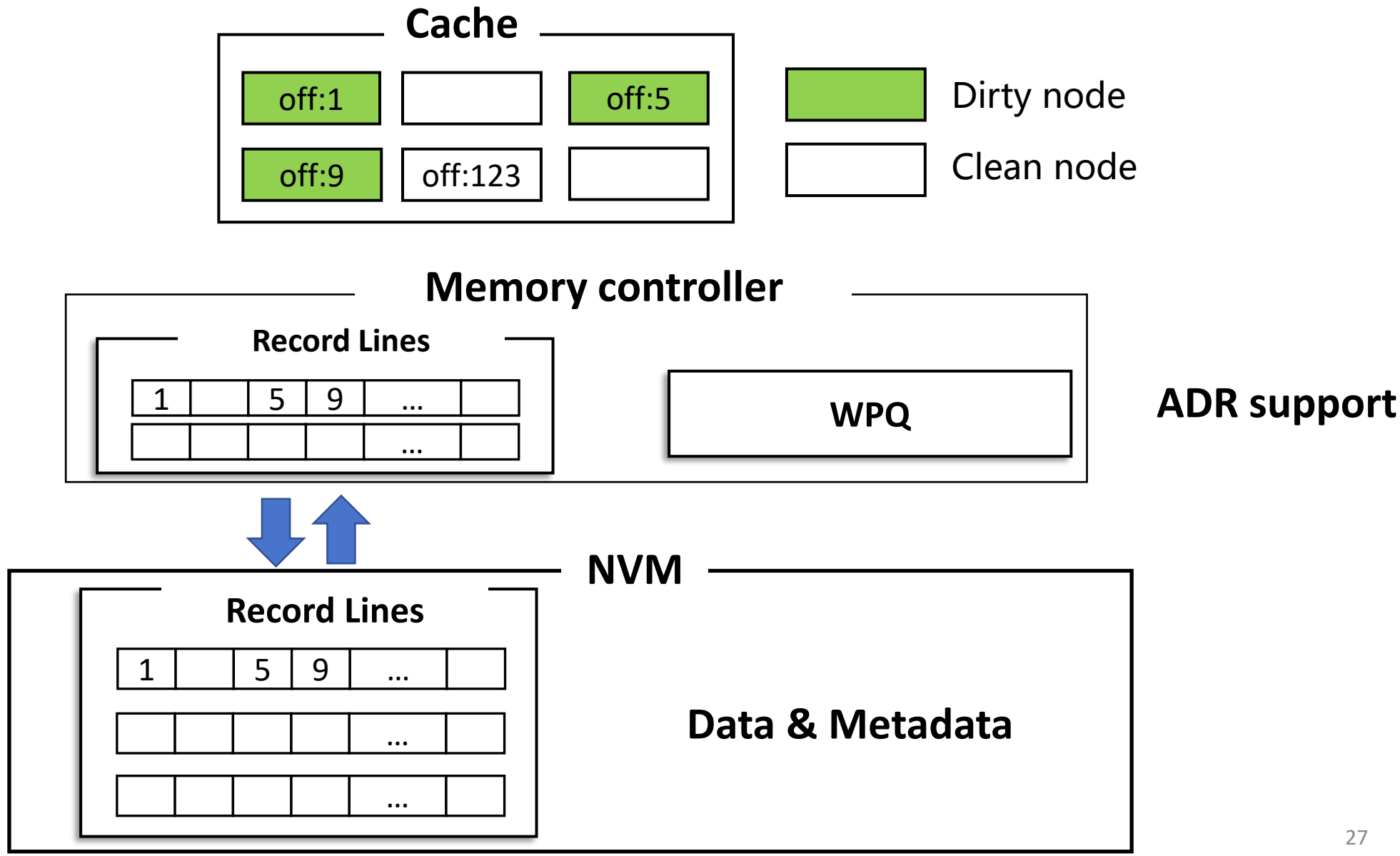


Detect replay attacks

- Recovery Processes(From root to leaf)
  - No attack:
    - Retrieved A’ and B’ can be trusted to verify
    - Recover and verify L_{n-2} nodes (C’ E’)
    - …..
    - Recover and verify leaf nodes



Track dirty nodes for fast recovery



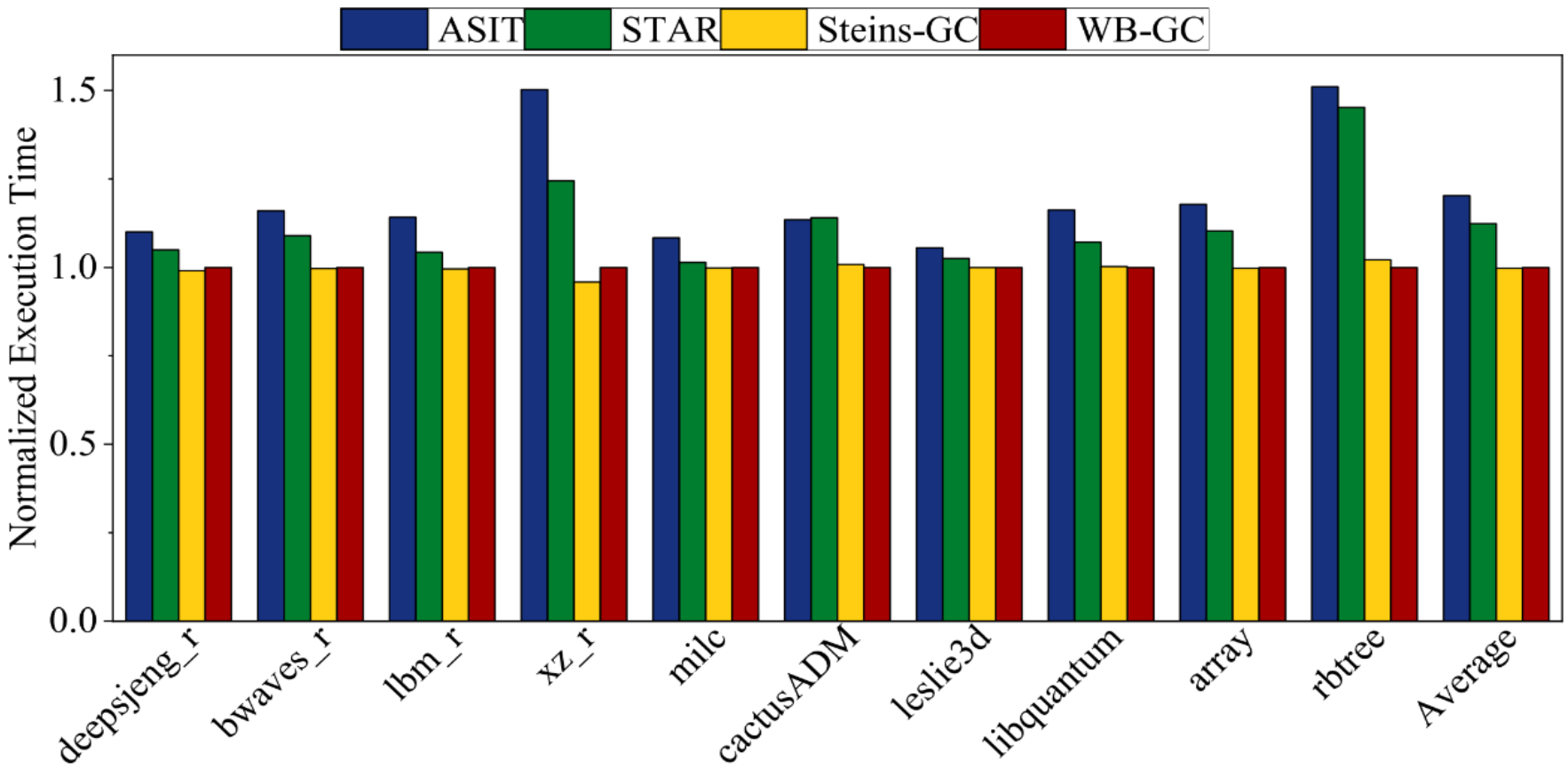
Performance Evaluation
Gem5 + NVMain

Processor	8 cores(**2 GHz**); L1(**32 KB**), L2(**512 KB**), L3(**2 MB**) Caches
Memory Controller	Security Metadata Cache(**256 KB**) Record lines (**16KB**)
NVM	16 GB
SIT	8/9 levels

Comparisons

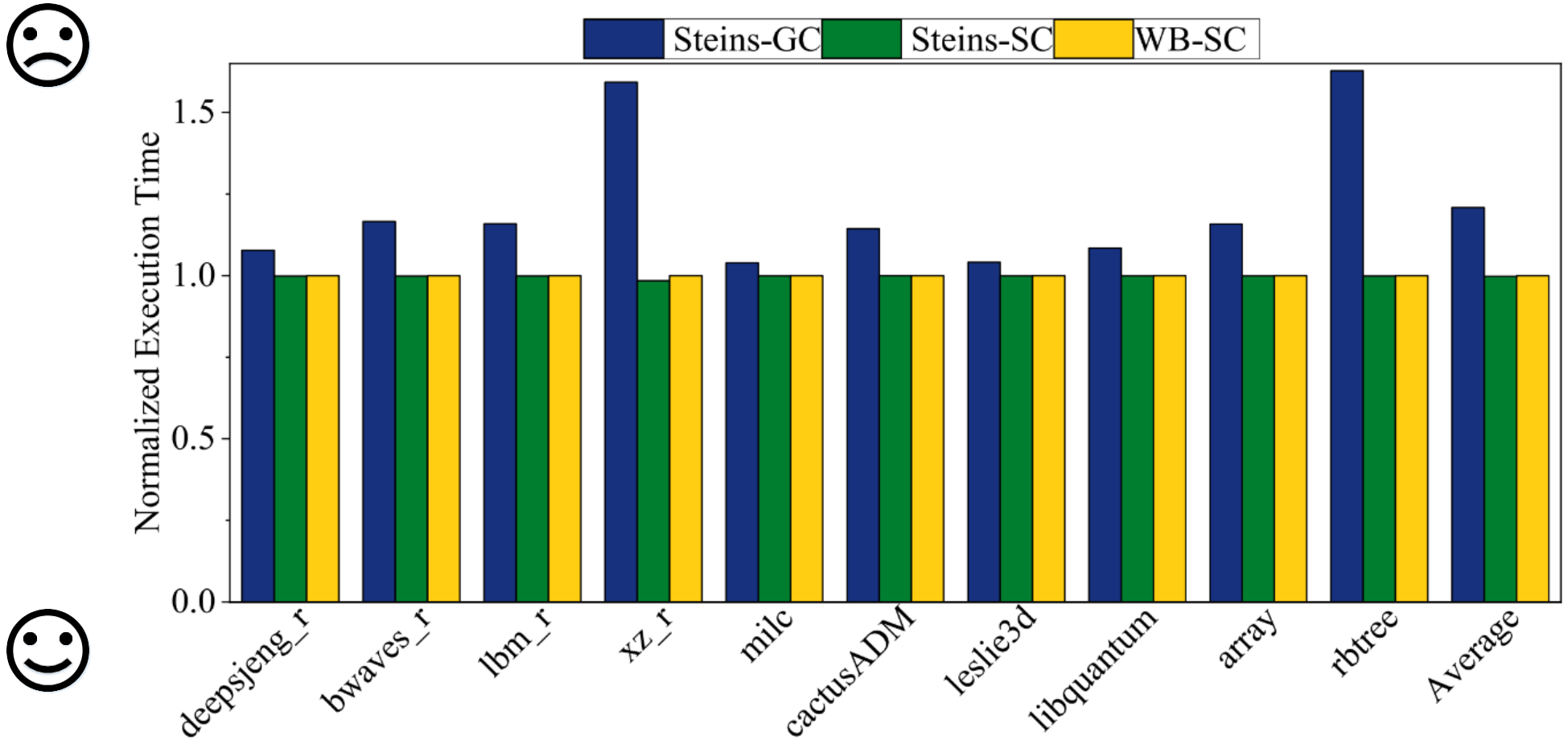
- Anubis[ISCA'19]
- STAR [HPCA'21]
- **Our Steins(Steins-GC / Steins-SC)**
- Write-Back (WB-GC / WB-SC)

Execution time



- Similar to the baseline which does not support recovery

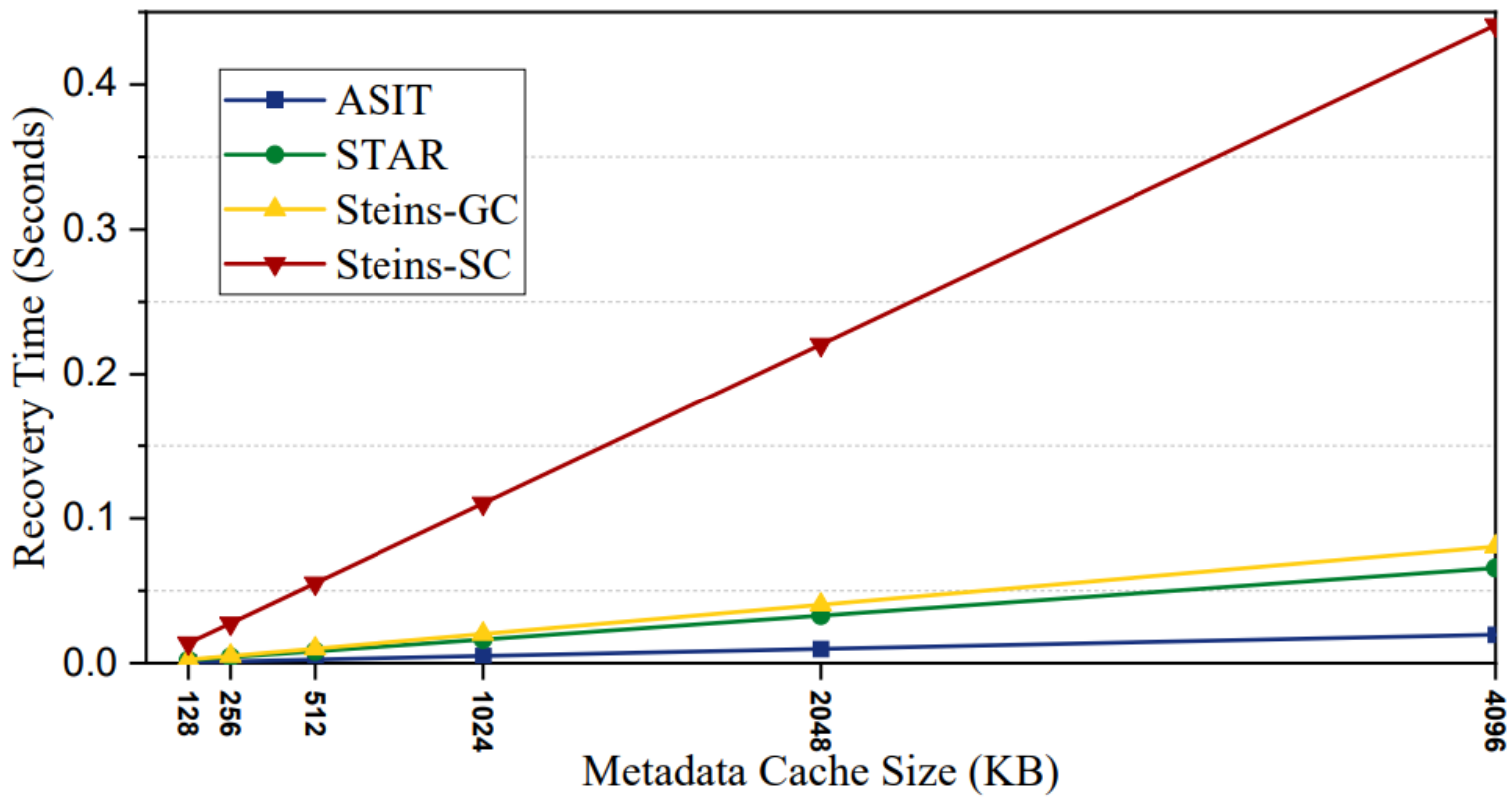
Execution time



- Steins-SC performs much better than Steins-GC

Recovery time

☹



☺

- Steins-SC: 0.44s
- Steins-GC: 0.08s

Conclusion

- **Design goal**
  - **Bridge the gap between fast recovery and high performance in secure NVM systems.**

- **We propose cost-efficient Steins, supporting the fast recovery of SIT while guaranteeing high performance.**
  - **Efficient counter generation scheme for recovery.**
  - **Offset-based tracking for fast location.**
  - **Appropriate trust bases for fast verification.**