D-Shield: Enabling Processor-side Encryption and Integrity Verification for Secure NVMe Drives

Md Hafizul Islam Chowdhuryy
University of Central Florida

Myoungsoo Jung
KAIST

Fan Yao
University of Central Florida

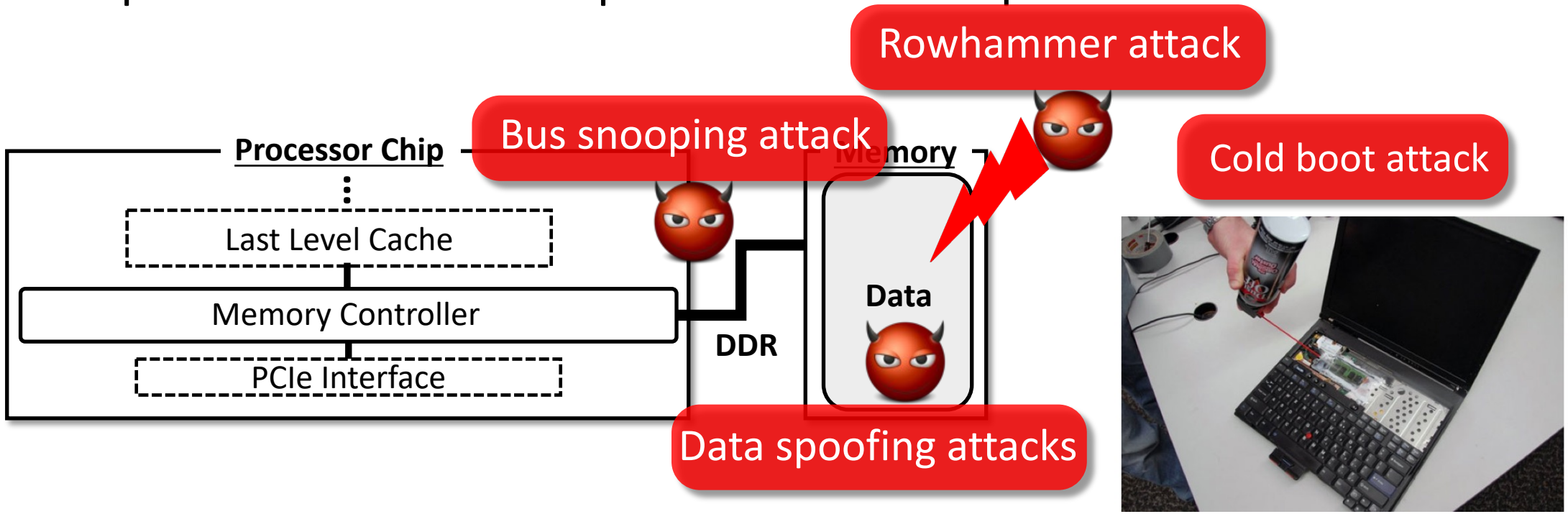
Amro Awad
NC State University

Presenter: Md Hafizul Islam Chowdhuryy
Computer Architecture and Systems Research Lab, University of Central Florida

29th IEEE International Symposium on High-Performance Computer Architecture (**HPCA**-29)
February 25th – March 1st, 2023

Hardware Security Threats to System

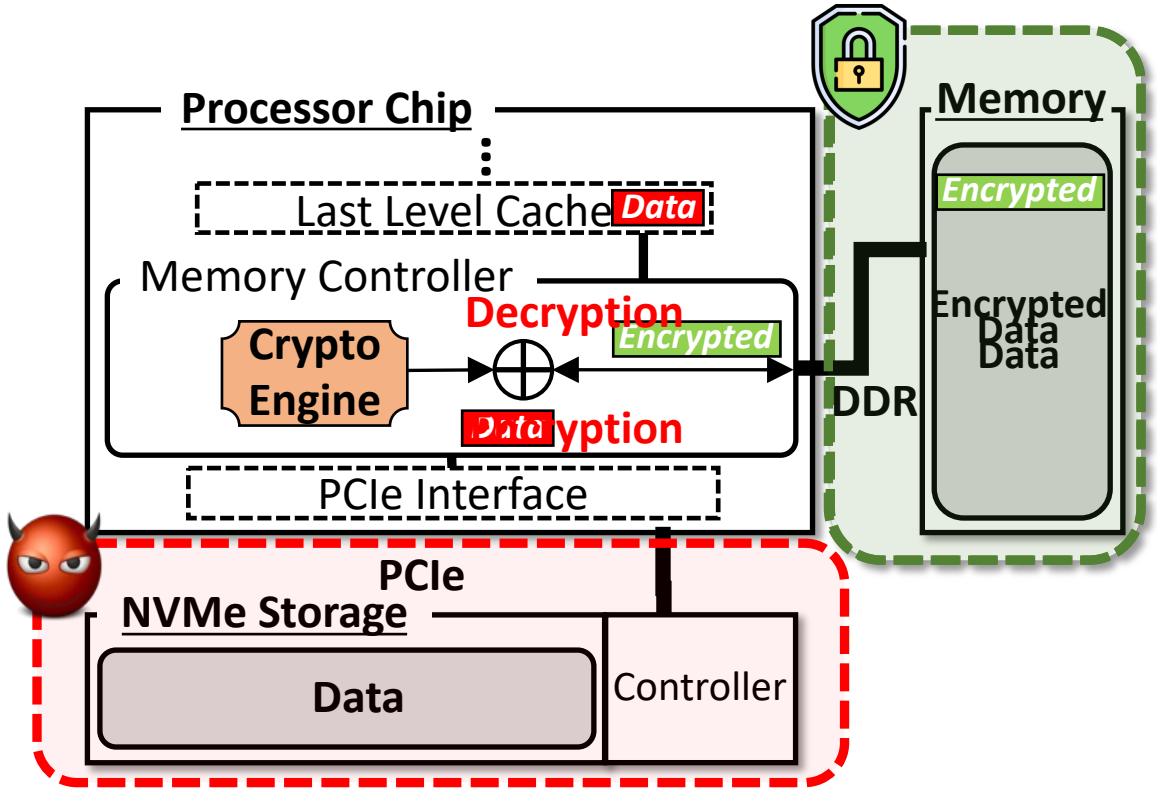
- Data is the main target of exploitation
- Multiple attack vectors are possible for off-chip data



Source: iinfosecinstitute.com

UCF

Secure Memory Architecture



Limit trust boundary to the processor chip
Protect confidentiality and integrity of **off-chip data**

Mainly focus on **memory security**

Storage security provided through **software** or **disk** itself
No processor-side support for fast storage security

UCF

Need for Architectural Support for Storage Security

- Pitfalls of existing solutions:
  - **Self-encryption disks:** encrypts data in the storage itself
    - Do not protect physical attacks (i.e., bus snooping)
  - **Software-based** disk encryption and integrity checking?

This work

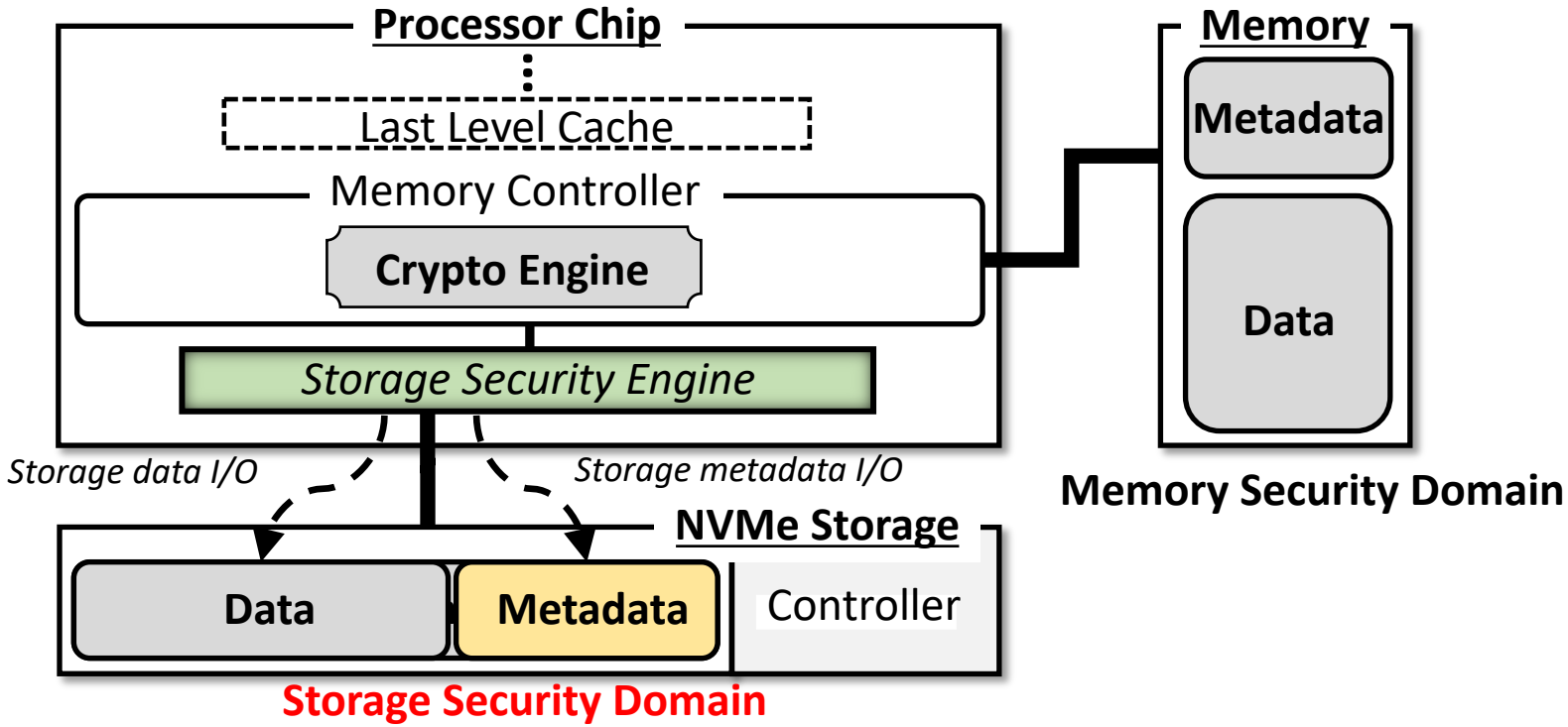
Architectural framework for *processor-side data protection* for fast storage devices.

- Emerging ultra-fast storage devices will further increase the bottleneck
  - Microseconds range access latency (e.g., Intel Optane SSD)

1. Performed on Intel i7-9700K with Samsung 970 (NVMe)

UCF

Design Objectives

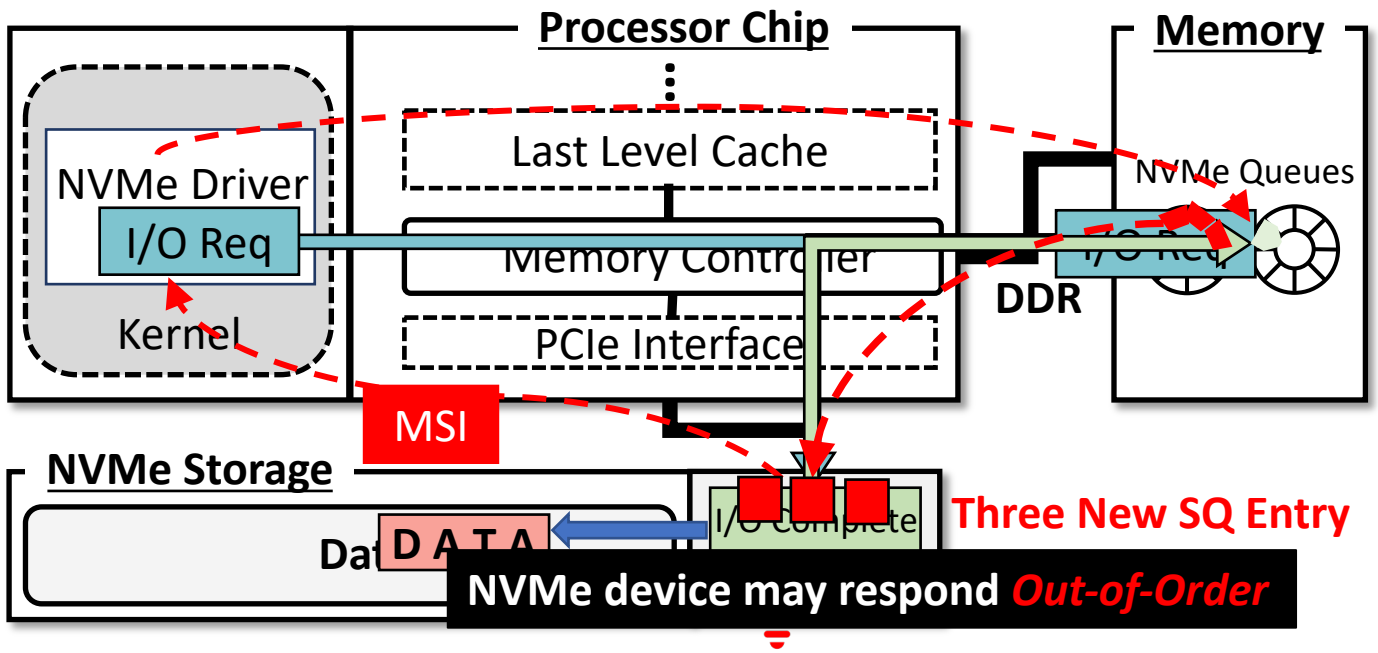


Objective 1: processor-side support (CPU as root of trust)

Objective 2: transparent to storage devices and NVMe protocols

UCF

Challenges



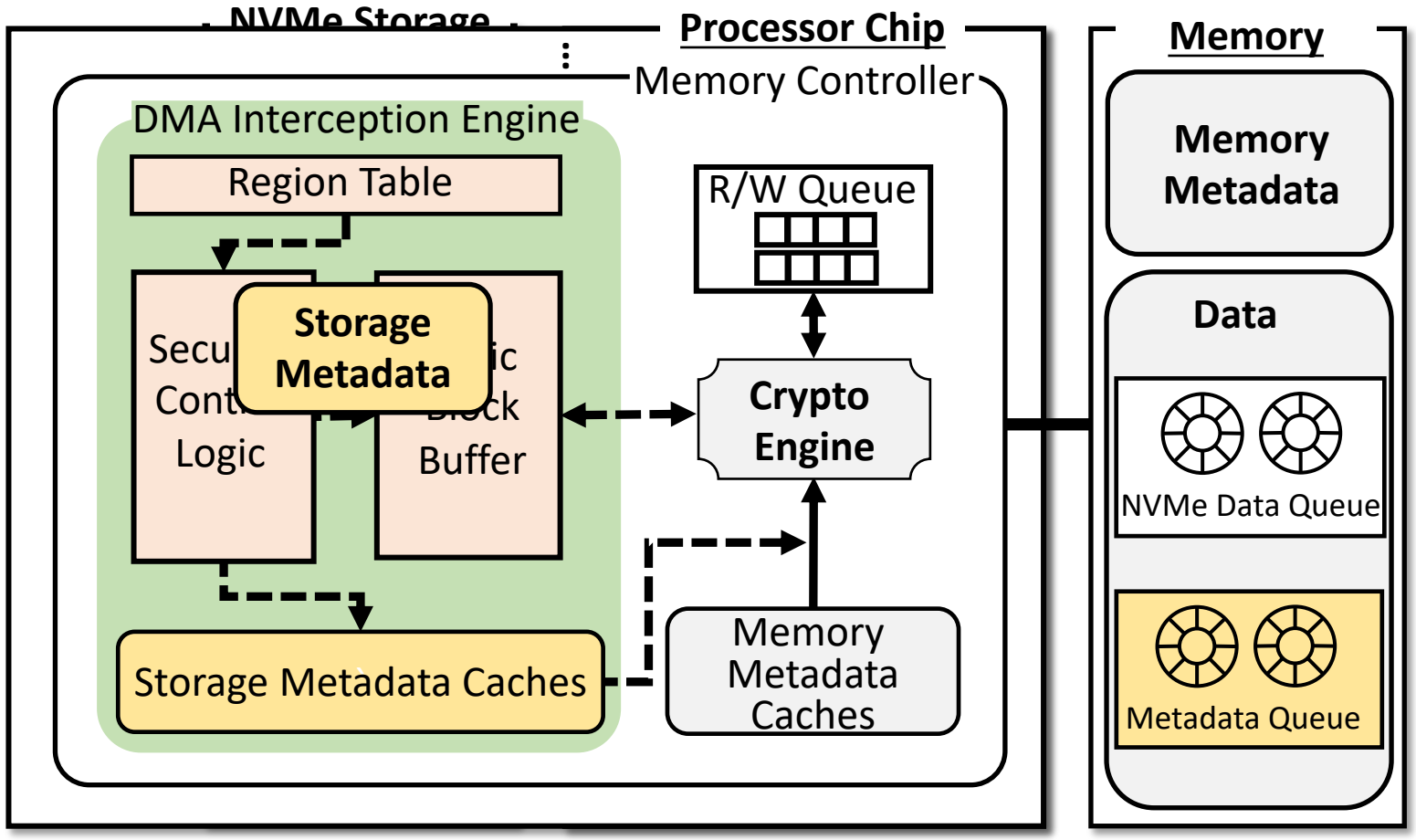
C1: Intricate SW/HW interactions → excessive software intervention can be expensive

C2: Asynchronous control/data flow → requires hardware support to identify and map DMA requests

C3: Metadata I/O overheads → efficient metadata management tailored for storage I/O characteristics

UCF

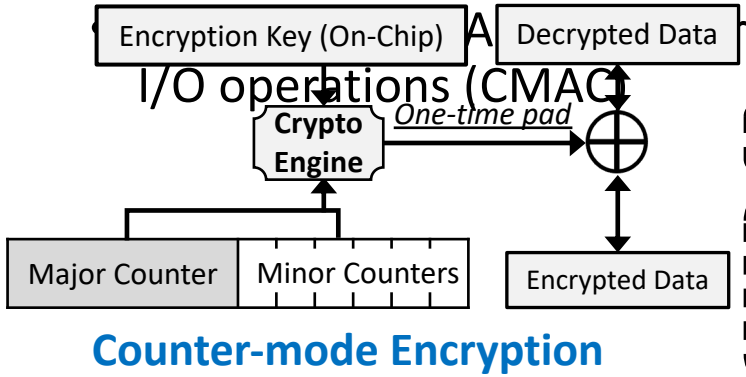
Basic D-Shield Design



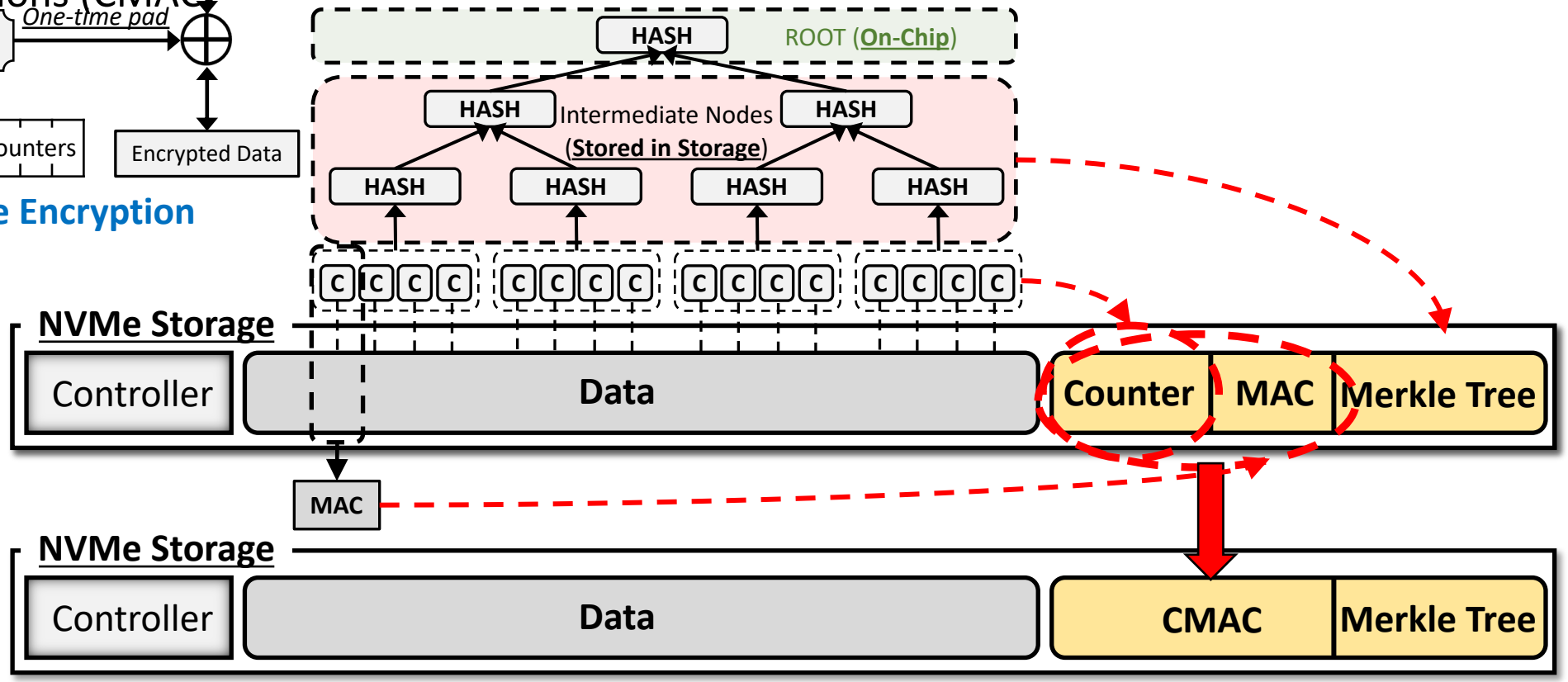
UCF

NVMe-optimized Security Metadata

- Storage metadata are stored separately in NVMe disks
- Three types of security metadata → similar to *Secure Memory*
- A[illegible]ed for each data access → store them together to reduce metadata I/O operations (CMAC)

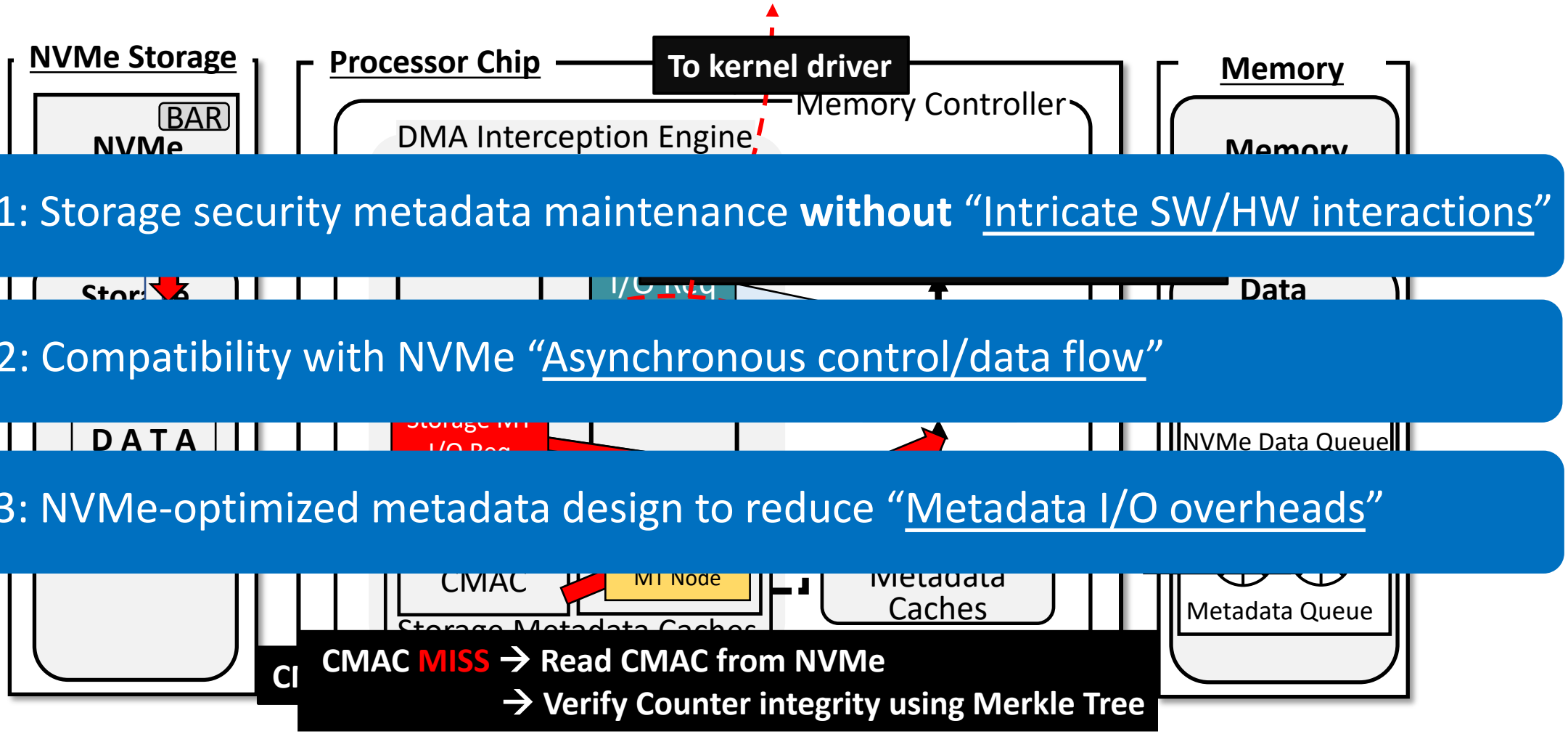


Counter-mode Encryption



UCF

D-Shield Operation (Read)



C1: Storage security metadata maintenance **without** "Intricate SW/HW interactions"

C2: Compatibility with NVMe "Asynchronous control/data flow"

C3: NVMe-optimized metadata design to reduce "Metadata I/O overheads"

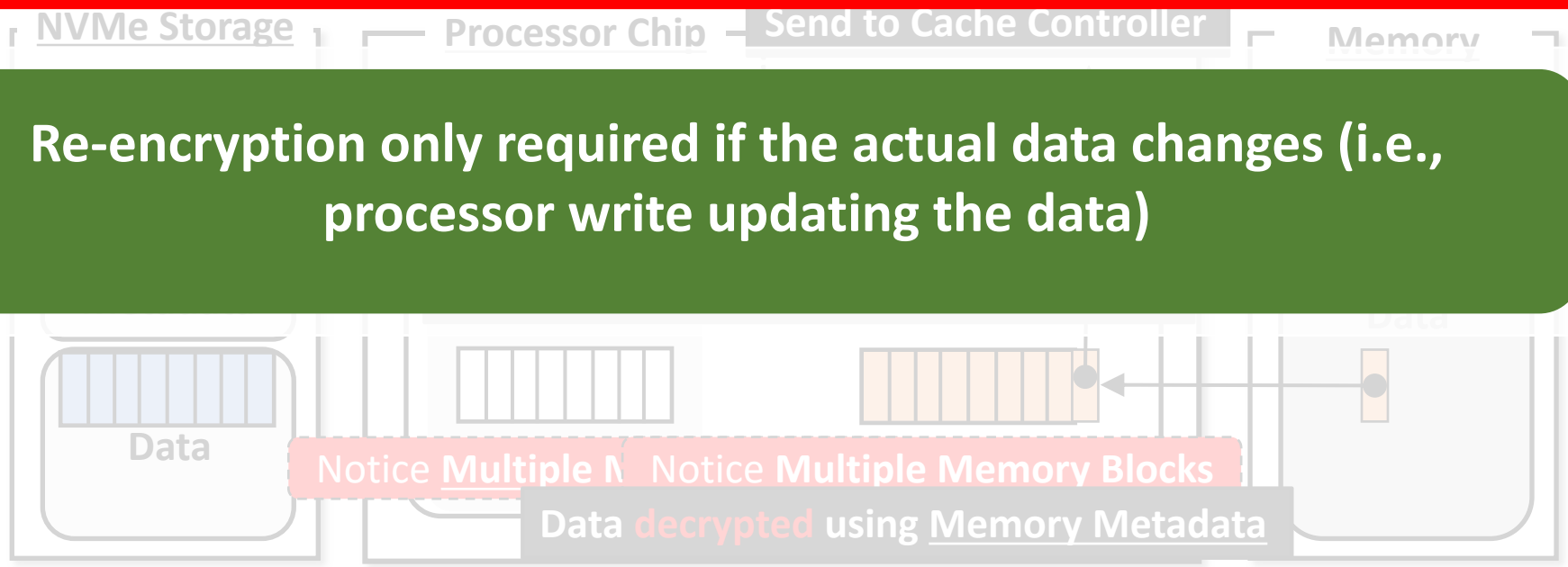
UCF

D-Shield-Hyb: Cross Domain Access Optimization

- Basic D-Shield provides proper off-chip data security with standalone protection for memory and storage

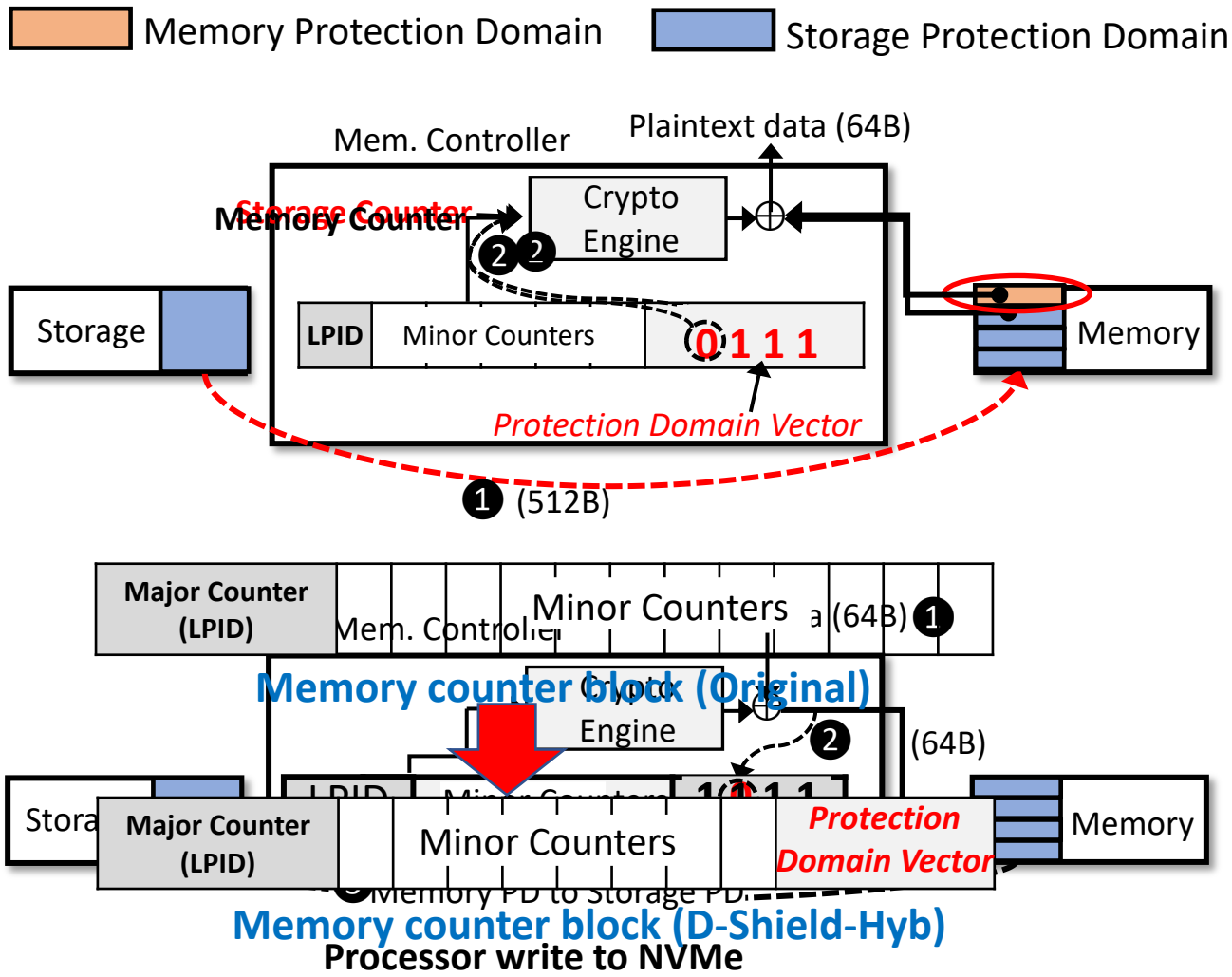
Moving data between protection domains (i.e., memory and storage) can be *expensive*
Additional utilization of the cryptographic engine
Prolonged NVMe data path

Re-encryption only required if the actual data changes (i.e., processor write updating the data)



UCF

D-Shield-Hyb: Cross Domain Access Optimization

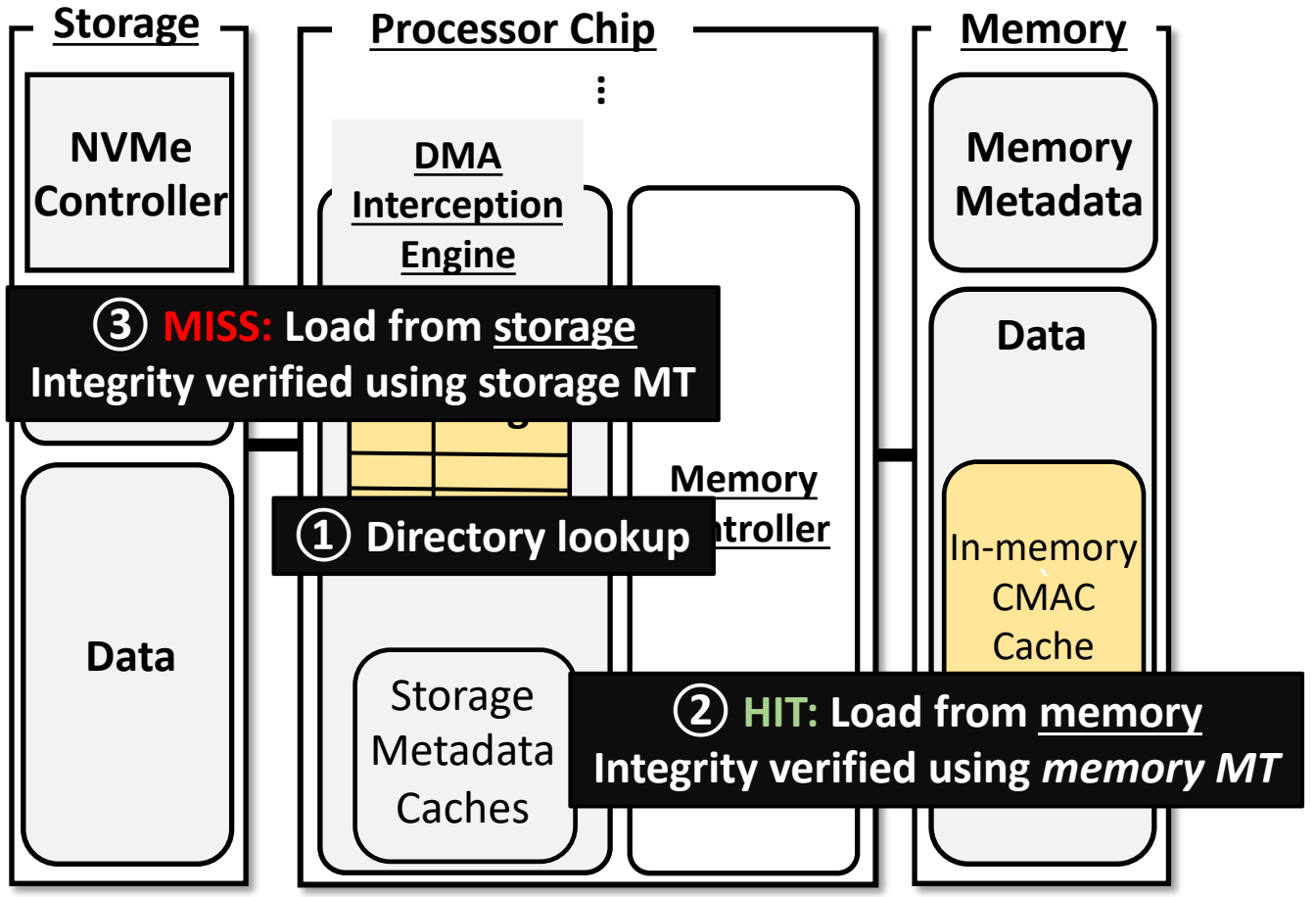


- Bookkeep the ownership of logic blocks **in memory**
- Track the security domain for transferred data block
- Performs only one iteration of decryption/encryption *on-demand*

UCF

D-Shield-Pro: In-Memory Caching

- Storage metadata cache misses have high overheads
- Miss in CMAC block is more expensive since it may require additional metadata access (i.e., for Merkle tree blocks)
- **Idea**: *In-memory CMAC block caching* to increase the CMAC block hit ratio



UCF

Experimental Setup

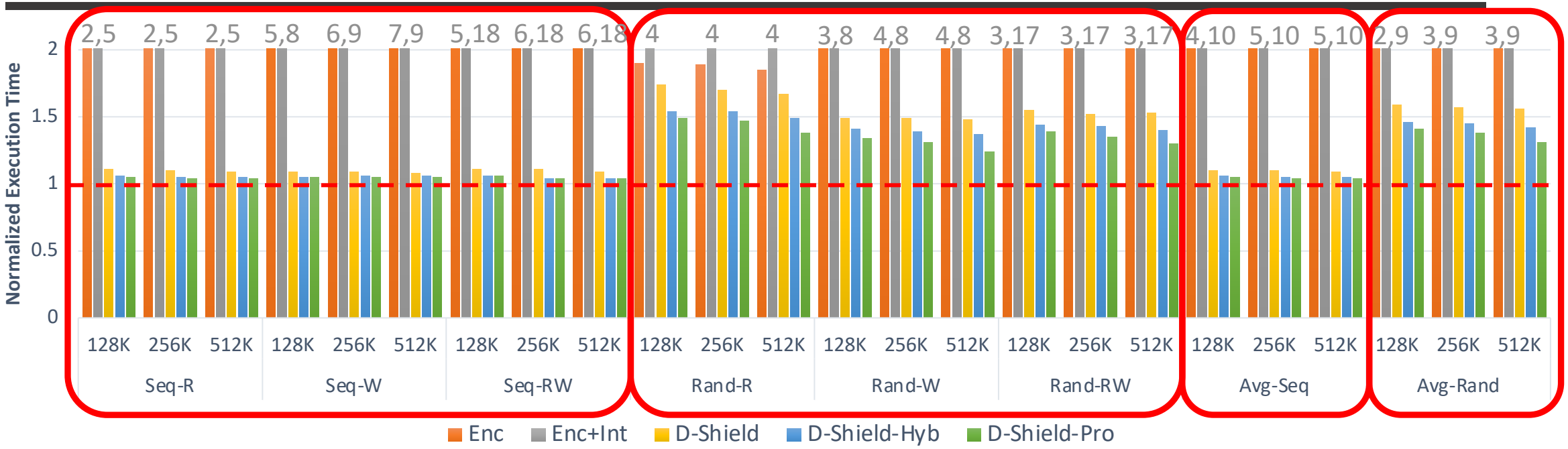
- **Simulator:** Gem5-based full-system simulation (SimpleSSD)
- **OS:** Ubuntu 18.04; **Kernel:** Linux 4.9

Hardware	***Configurations***
Processor	4-core, 3.0 GHz in-order, x86
L1 I/D-cache	Private, 64KB, 4-way
L2 cache	Shared, 16MB, 16-way
Main memory	DDR4 based 16GB
Cryptographic Engine	
Encryption/Hash operation (64B)	40 cycles
DMA Interception Engine	
Metadata cache	256KB 8-way each
Hash operation (512B)	320 cycles
NVMe Disk	
Capacity	512GB
Cell model	Z-NAND based MLC PCM
Avg. random access latency(µS)	READ: 10.5, WRITE: 9

Evaluation Methodology

- **Workloads:**
  - **I/O intensive applications:** Flexible I/O
  - **Server-class applications:** database, document storage system
  - **Graph algorithms:** YCSB suite (twitter follow network)
- **Baselines (All variants include *secure memory*):**
  - **Insecure:** Default NVMe storage system without security mechanism
  - **Enc:** dm-crypt-based encryption for NVMe storage system
  - **Enc+Int:** dm-crypt with dm-integrity for NVMe disk encryption and integrity checking

Evaluations: D-Shield Performance



# of transactions (from left to right): **128K, 256K, 512K**

Runtime (normalized to Insecure) of FIO benchmark

Sequential workloads: 4.4% overhead compared to *Insecure* (Avg)

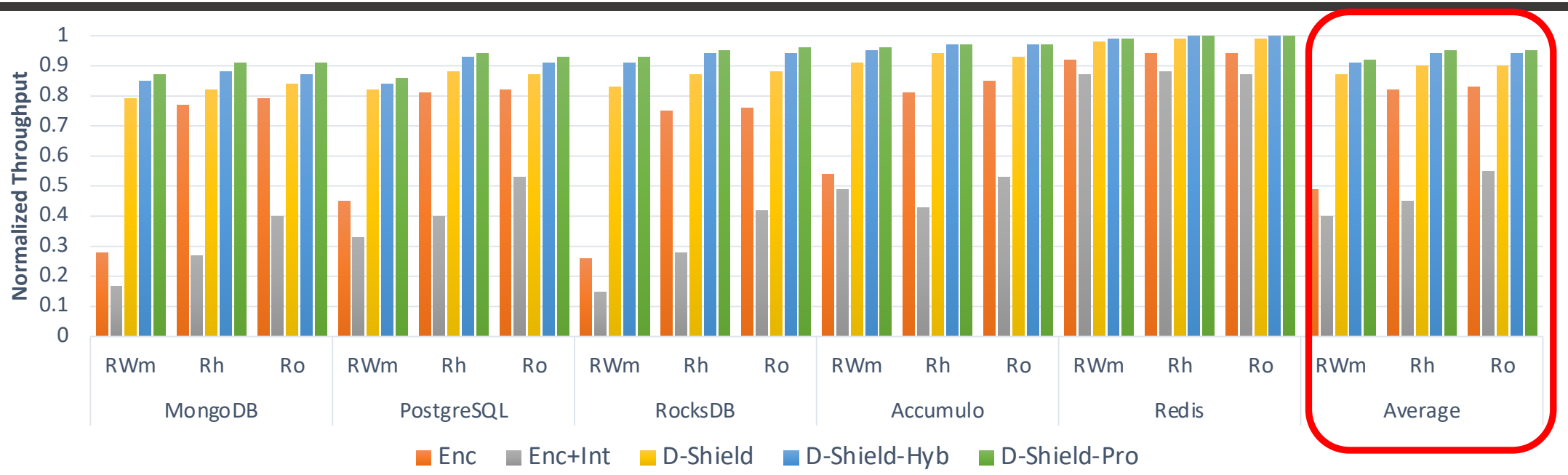
4.1x less compared to **Enc**, **10x less** compared to **Enc+Int**

Random workloads: 39% overhead compared to *Insecure* (Avg)

2.05x less compared to **Enc**, **7x less** compared to **Enc+Int**

UCF

Evaluations: D-Shield Performance



Throughput of D-Shield on real-world server applications

D-Shield can maintain 94% (Avg) throughput compared to *Insecure*

24% higher compared to **Enc** and **49%** higher compared to **Enc+Int**

UCF

Evaluations: Hardware Overhead

- NVMe storage overhead: **3.14%** for **Security Metadata**
- On-chip storage overhead: **2x256KB** for **Storage Metadata Caches**

  552 Bytes for **Region Table**
- In-memory storage: **128MB** for **In-memory Cache** (D-Shield-Pro only)
- D-Shield on-chip logic:
  - Implemented using Verilog
  - Synthesized with Synopsis DC with 45nm

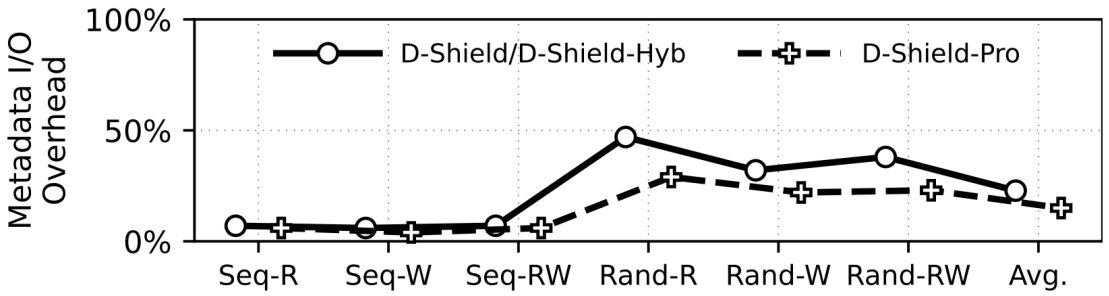
Module	**Area (mm^2)**
Logic Block Buffer	0.23
Security Control Logic	0.08

Conclusion

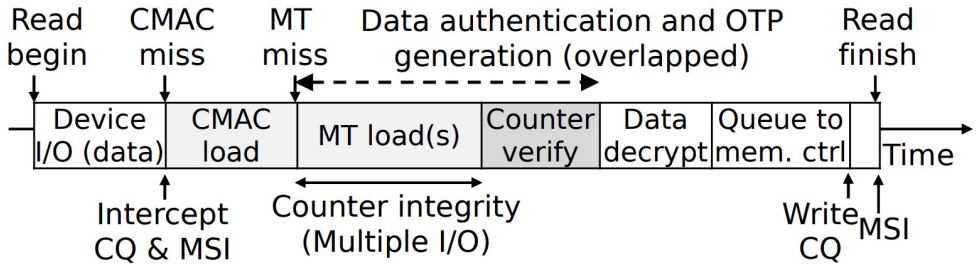
- **Existing storage protection offers limited security and impose high overhead**
- **D-Shield offers architectural framework for processor-side storage security**
- **D-Shield-Hybrid optimizes cross-domain data transfer substantially**
- **D-Shield-Pro reduces metadata overheads through in-memory caching**
- **Modest performance overheads in real-world workloads**
  - While providing state-of-the-art data security

More on Paper

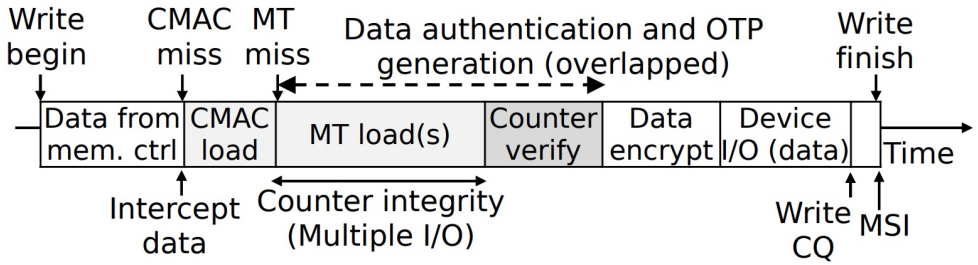
- **Architectural design space explorations**
- **Additional details on D-Shield designs:**
  - Complete R/W paths
  - Metadata arrangements and maintenance
- **D-Shield overhead analysis:**
  - Additional I/O overheads
  - Logic and storage overheads
- **Sensitivity analysis of D-Shield schemes**
- **And more…**



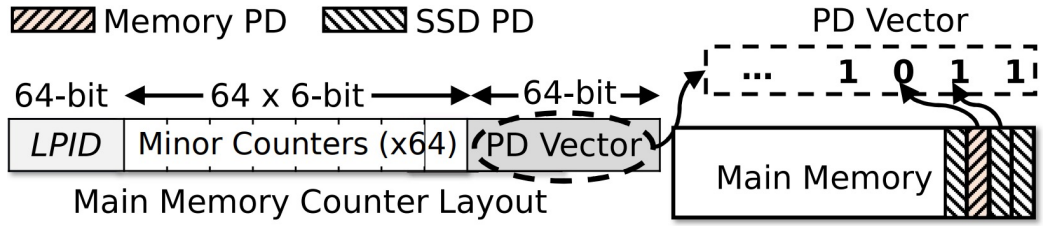
D-Shield I/O overheads



(a) DMA Interception Engine Events for Read operation



NVMe Read/Write path in D-Shield



D-Shield-Hybrid metadata storage

UCF

Thanks! Questions?

Md Hafizul Islam Chowdhuryy
CASR Lab (https://casr.ece.ucf.edu)
Email: reyad@knights.ucf.edu
Email: fan.yao@knights.ucf.edu