Project Almanac: A Time-Traveling Solid-State Drive

Xiaohao Wang Yifan Yuan You Zhou **Chance C. Coats** Jian Huang

Systems and Platform Research Group

ILLINOIS
Electrical & Computer Engineering
COLLEGE OF ENGINEERING

Retaining Past Storage States is Important

User file operations such as multi-versioning

Retaining Past Storage States is Important

Recovering files following a system crash

Retaining Past Storage States is Important

Storage forensics and police investigations

Software-Based Approaches to Retain Past Storage States

Journaling and Logging

Software-Based Approaches to Retain Past Storage States

Journaling and Logging

Local and Cloud Backups

Software-Based Approaches to Retain Past Storage States

Journaling and Logging

Local and Cloud Backups

Snapshots and Checkpointing

Software-Based Approaches to Retain Past Storage States

Software-based approaches not only reduce storage performance, but also increase storage cost!

The Security Problem with Software-Based Approaches

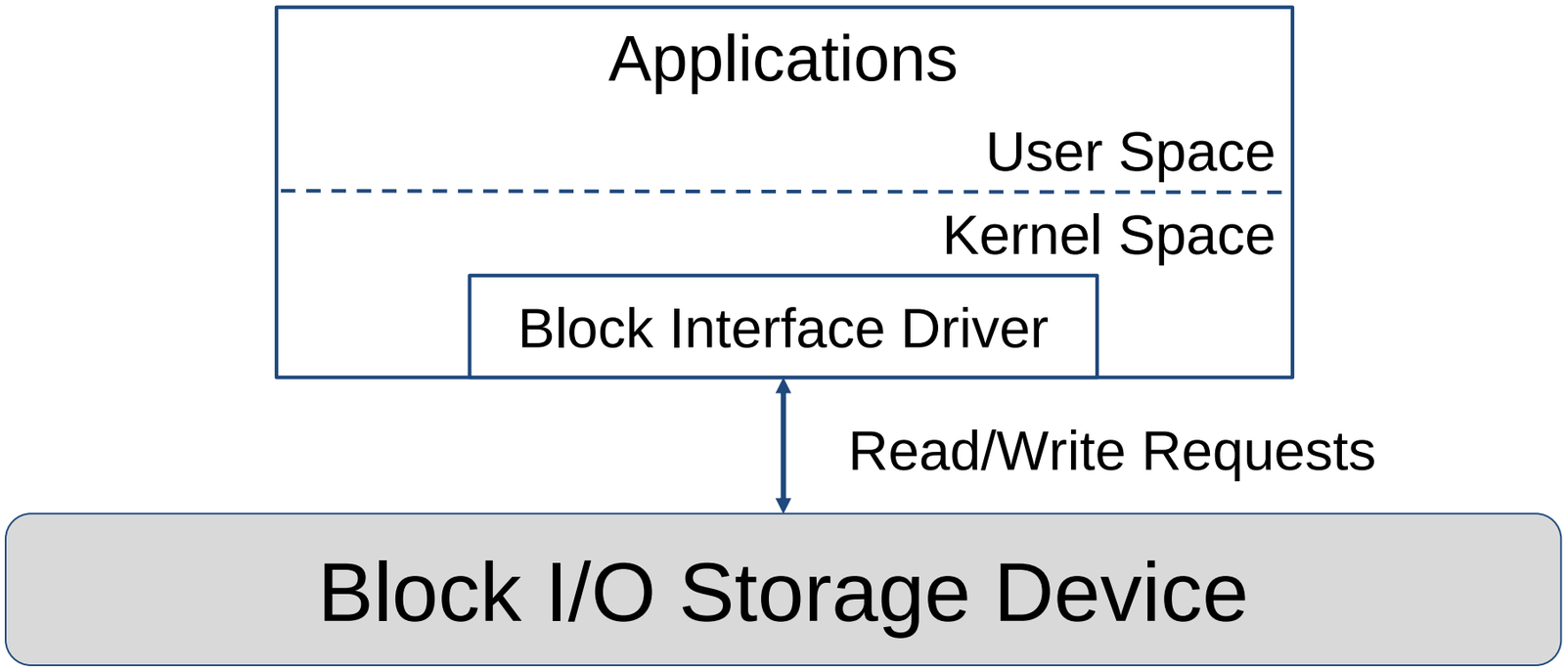
The Security Problem with Software-Based Approaches

Software systems are vulnerable to malware

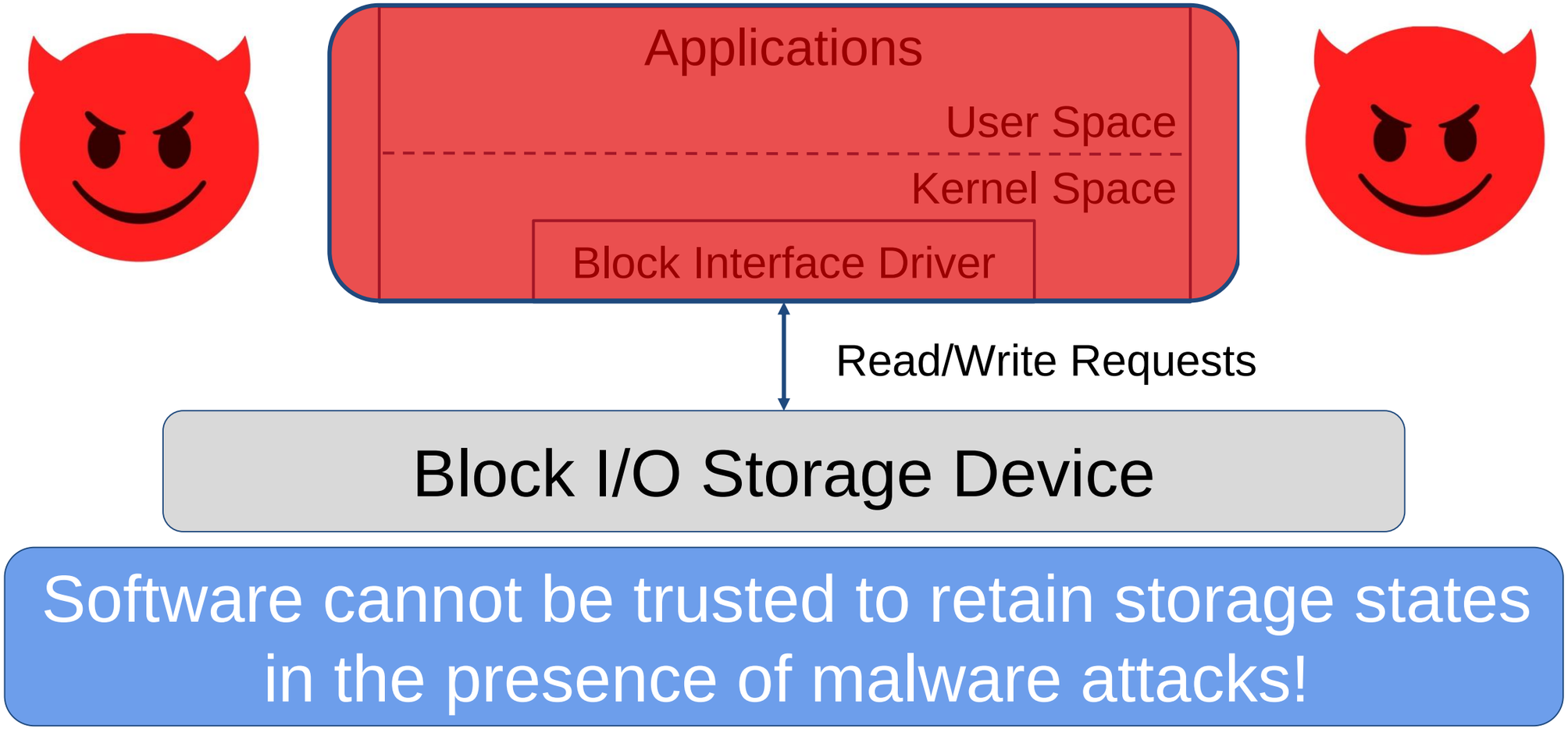
The Security Problem with Software-Based Approaches

Malware may obtain kernel privileges and stop or destroy backups and logging functions

The Underlying Problem: Threat Model Analysis

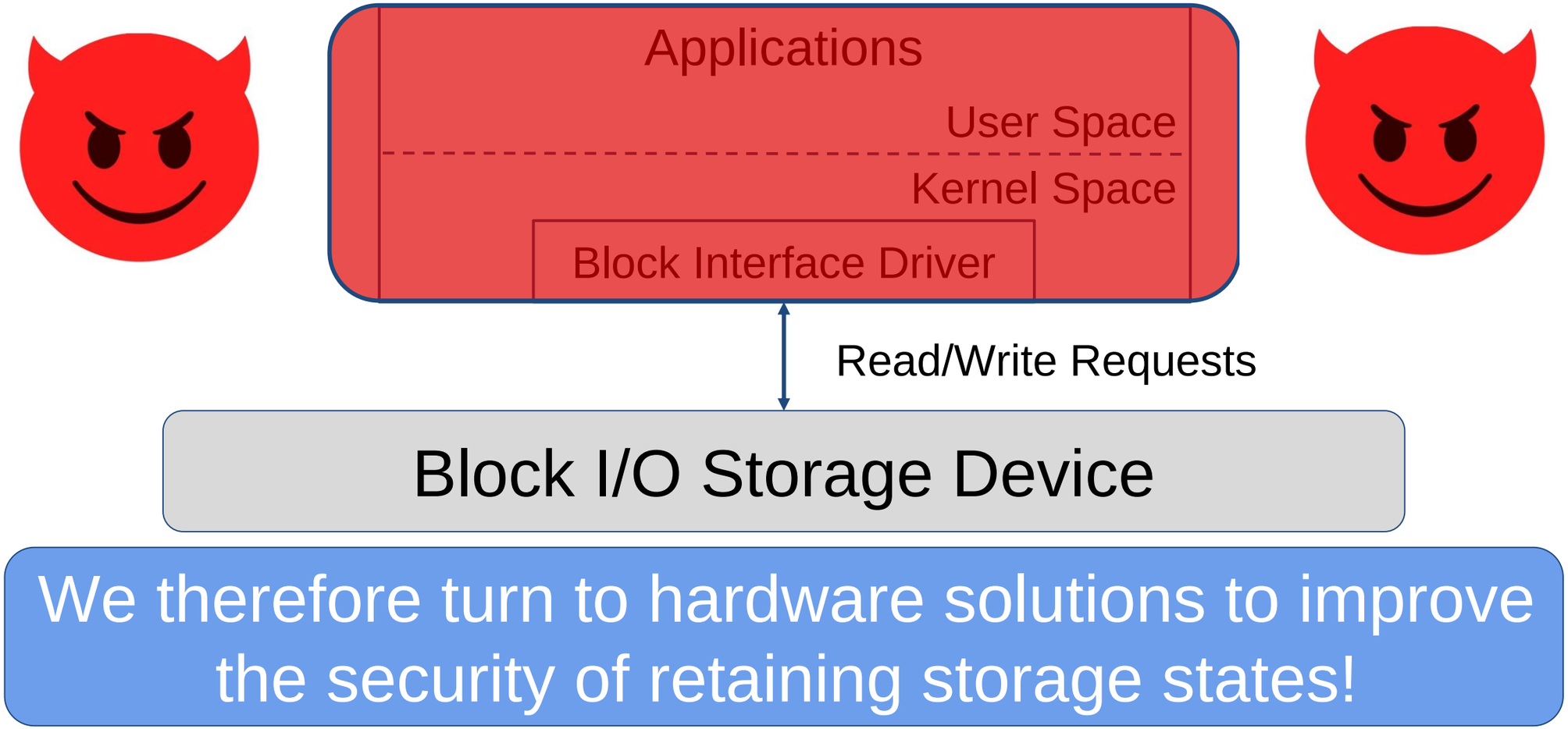


The Underlying Problem: Threat Model Analysis



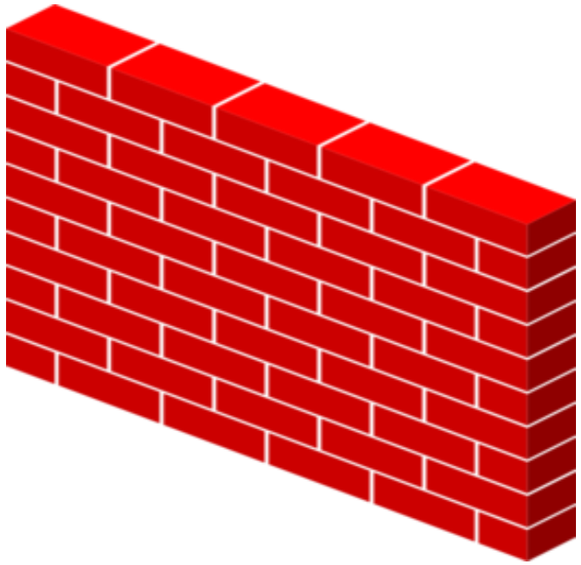
Software cannot be trusted to retain storage states in the presence of malware attacks!

The Underlying Problem: Threat Model Analysis



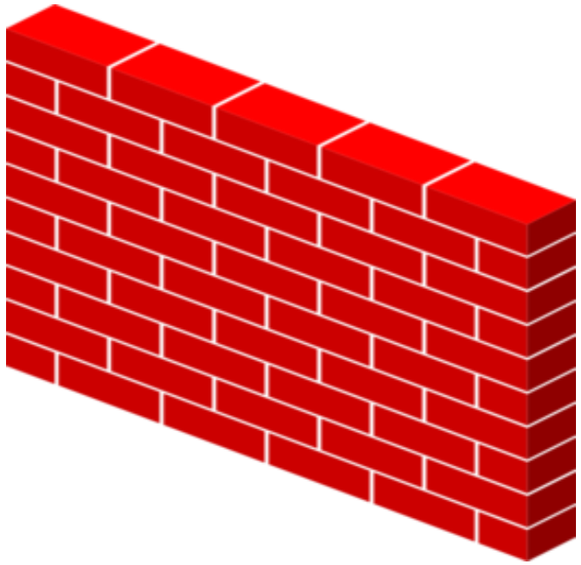
We therefore turn to hardware solutions to improve the security of retaining storage states!

Project Almanac: Our Goals



Firmware-isolated Protection

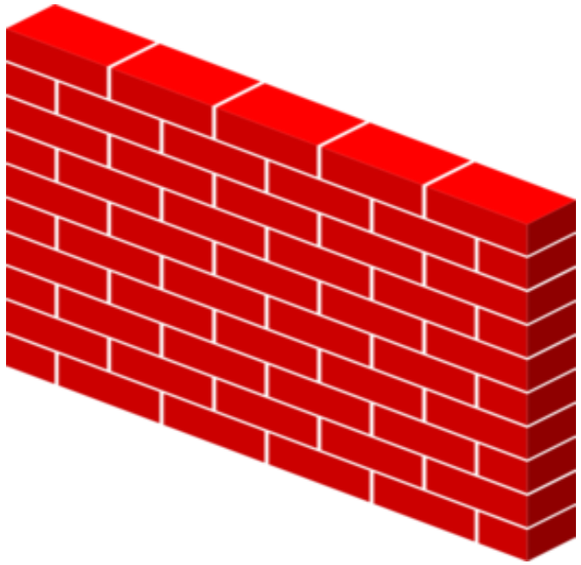
Project Almanac: Our Goals

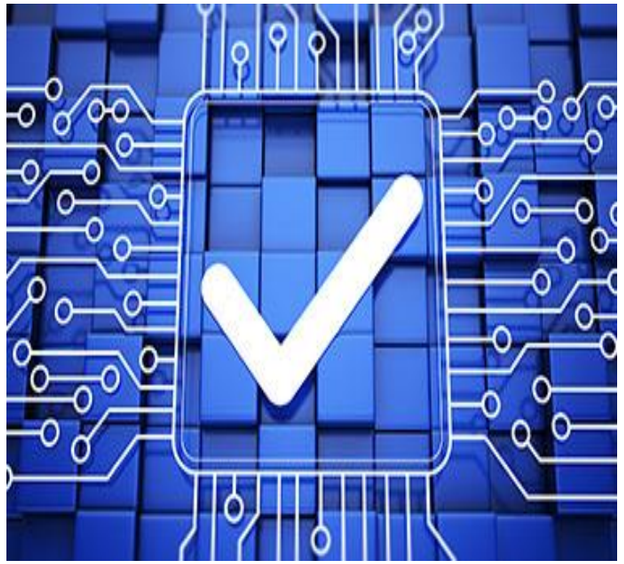


Firmware-isolated Protection

Minimal Performance Overhead

Project Almanac: Our Goals



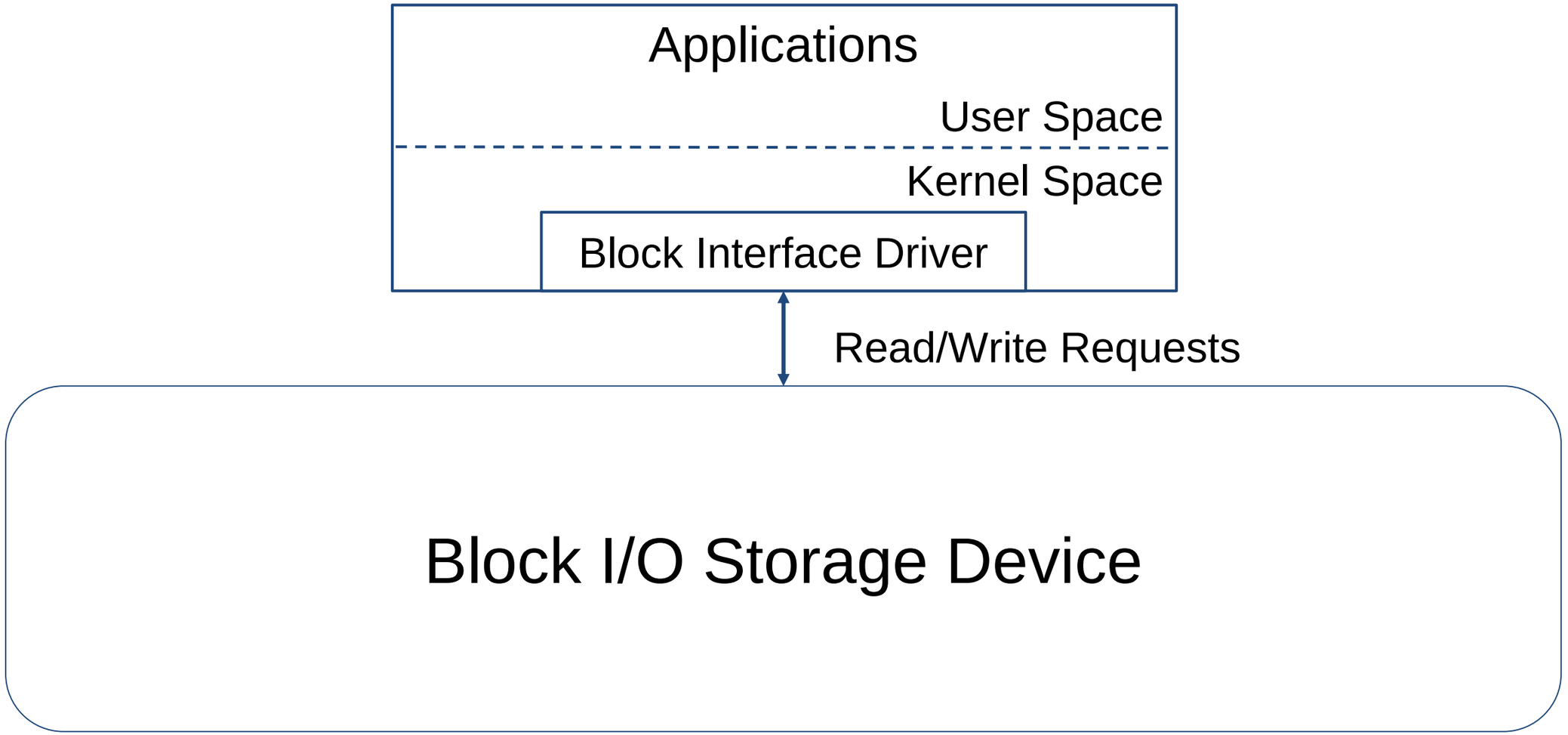


Firmware-isolated Protection

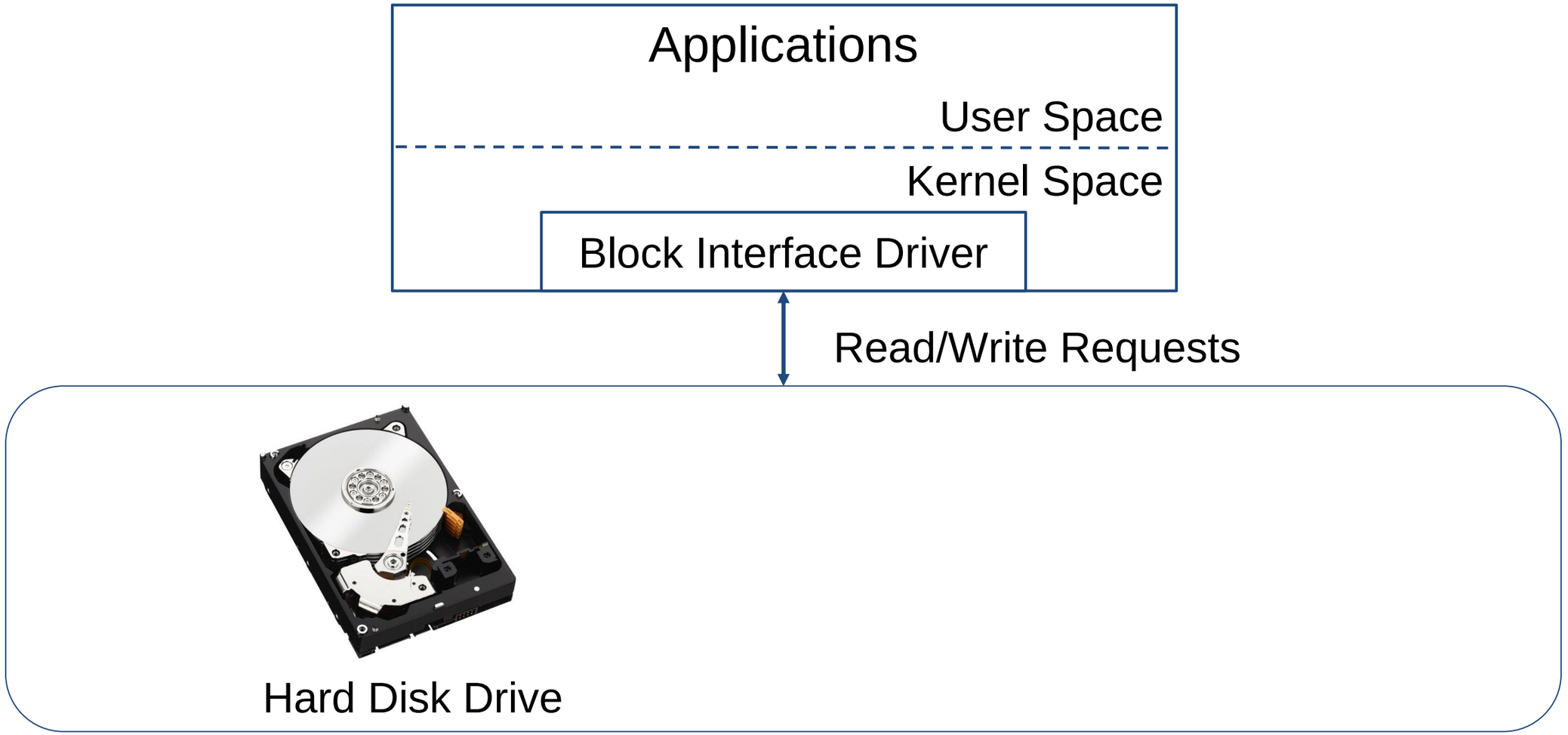
Minimal Performance Overhead

Preserving Software Functionality

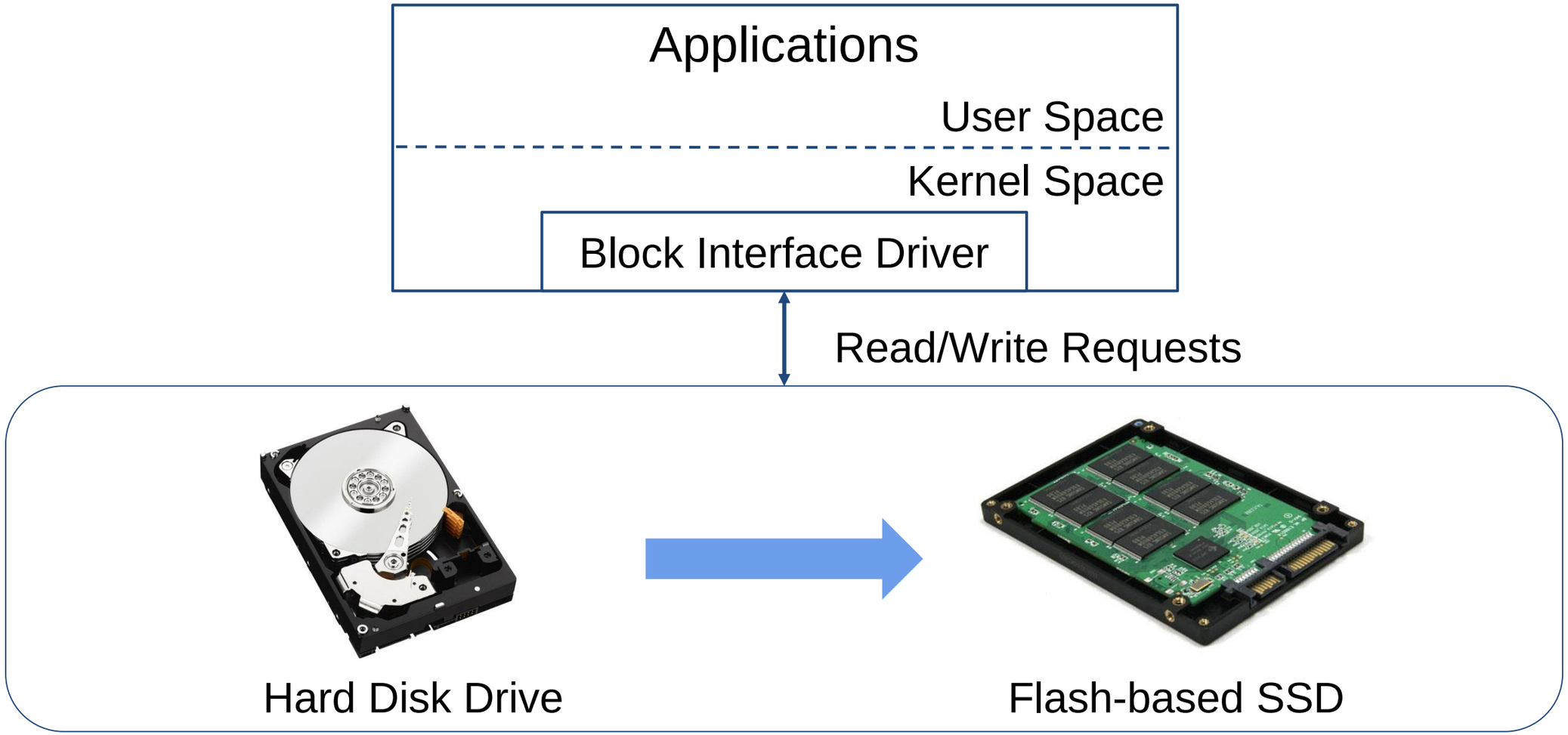
Hardware Protection Motivated by Flash Technology



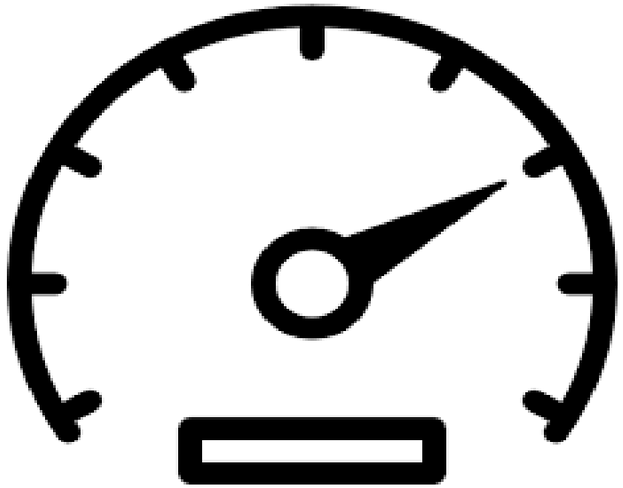
Hardware Protection Motivated by Flash Technology



Hardware Protection Motivated by Flash Technology

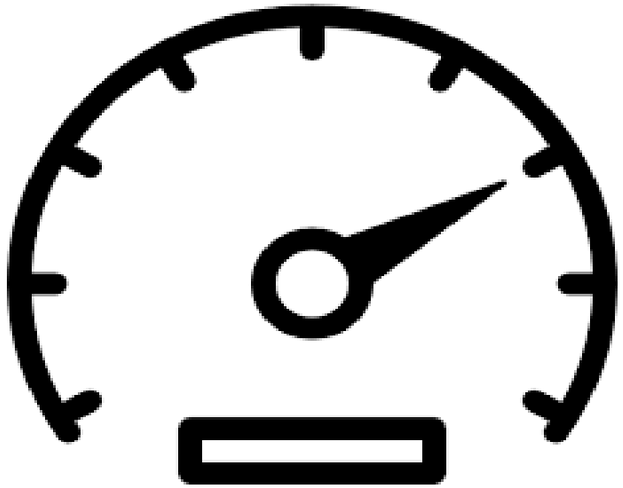


Why Solid-State Drives?



Lower Latency and
Higher Throughput

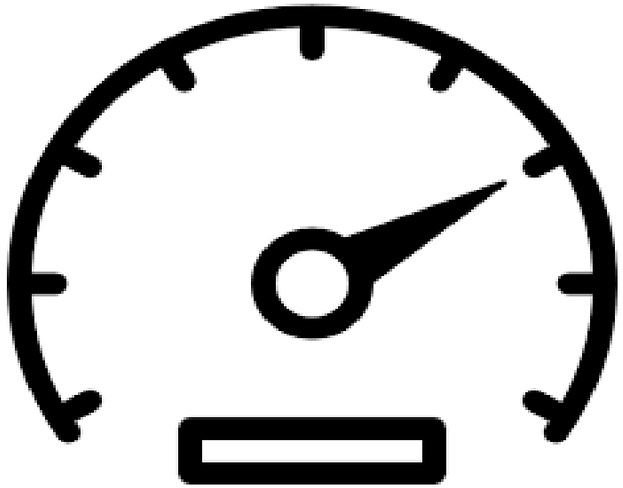
Why Solid-State Drives?



Lower Latency and Higher Throughput

Massive Parallelism

Why Solid-State Drives?



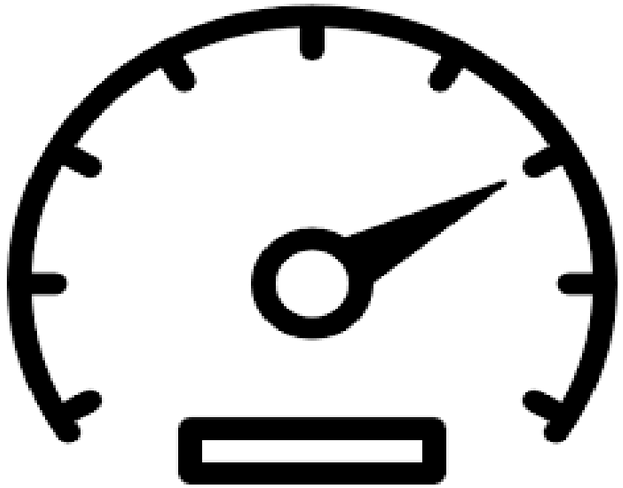
$0.20/GB

Lower Latency and Higher Throughput

Massive Parallelism

Commodity Pricing

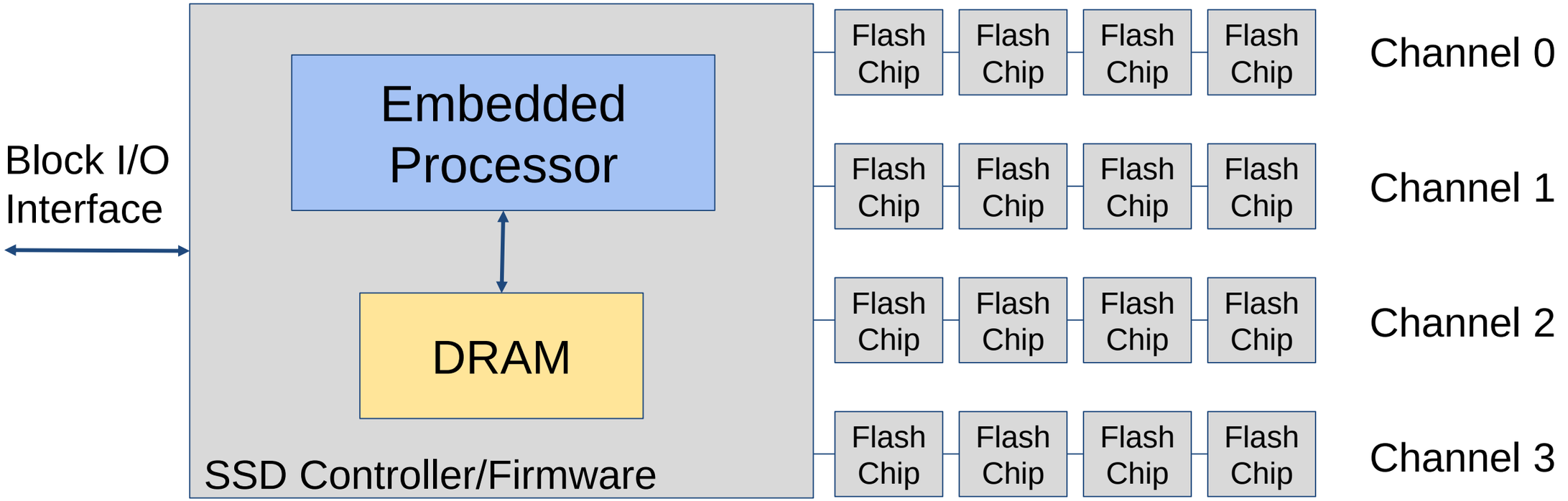
Why Solid-State Drives?



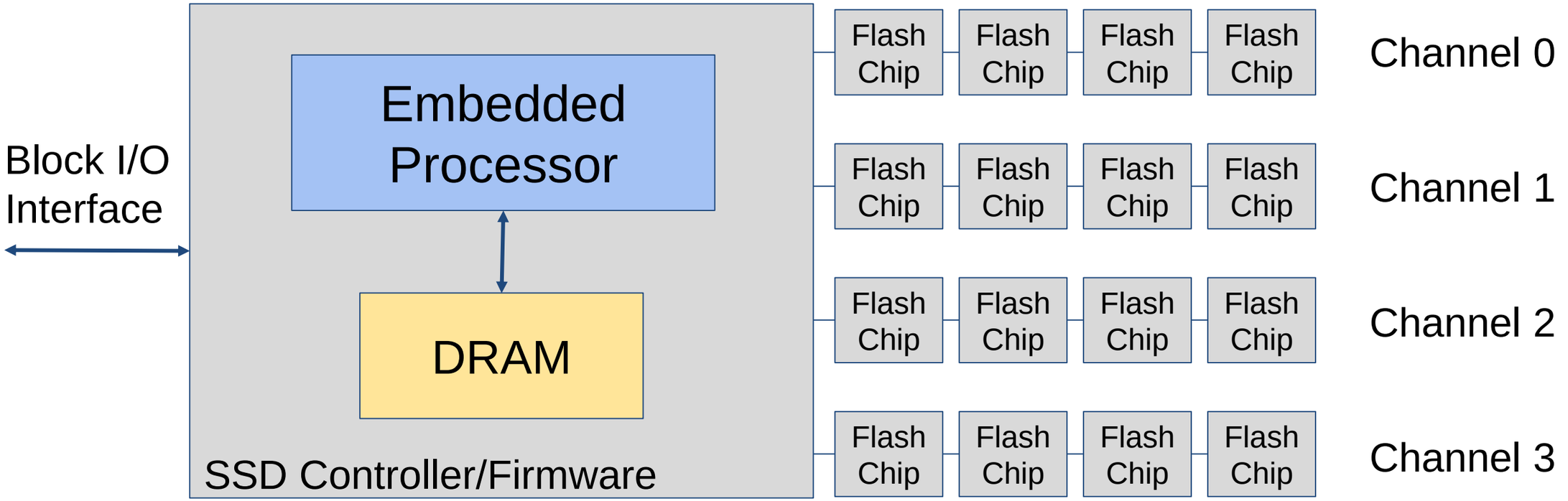
$0.20/GB

Flash is widely used in modern computing systems!

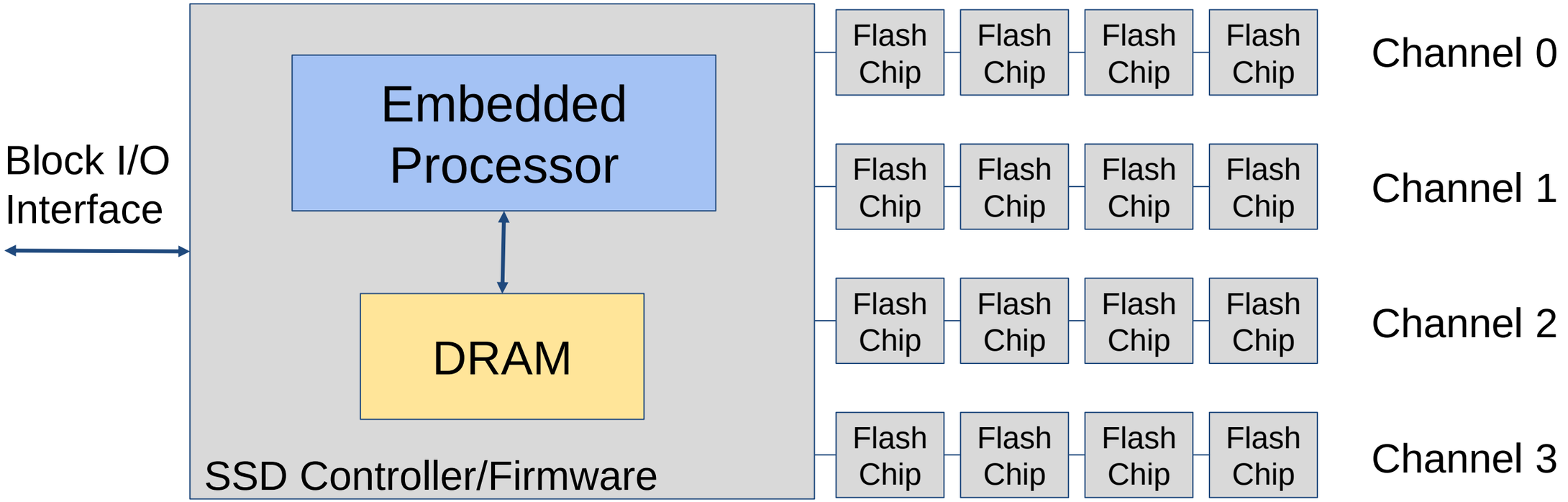
Solid-State Drive: A Perfect Candidate



Solid-State Drive: A Perfect Candidate

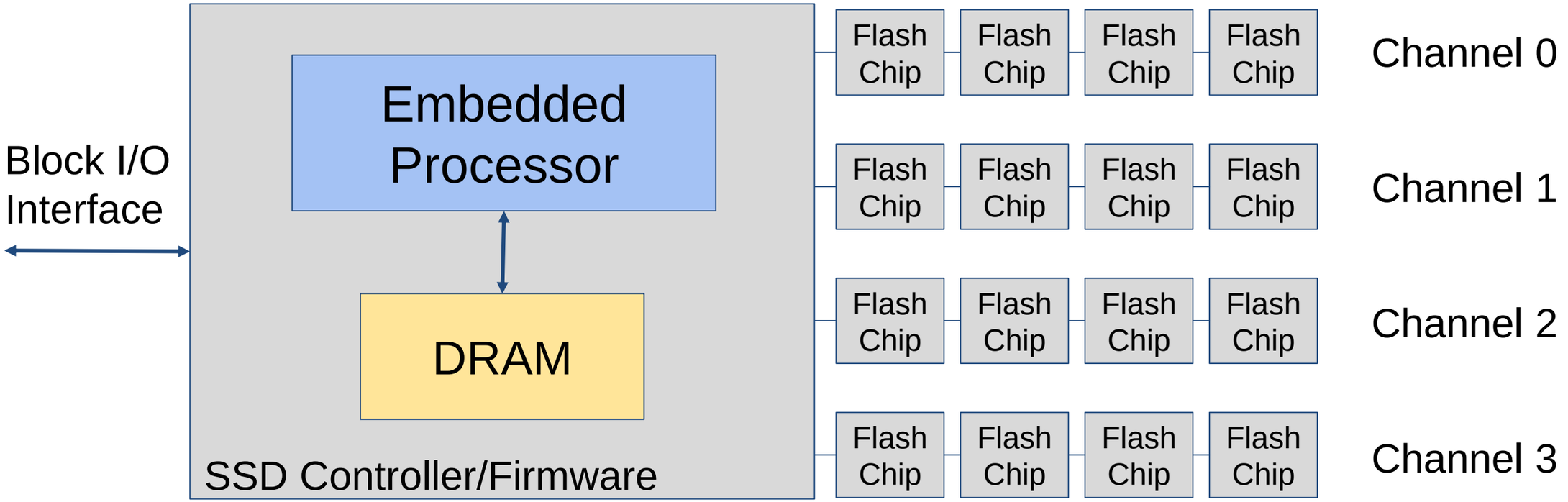


Solid-State Drive: A Perfect Candidate



Embedded Processor with DRAM

Solid-State Drive: A Perfect Candidate

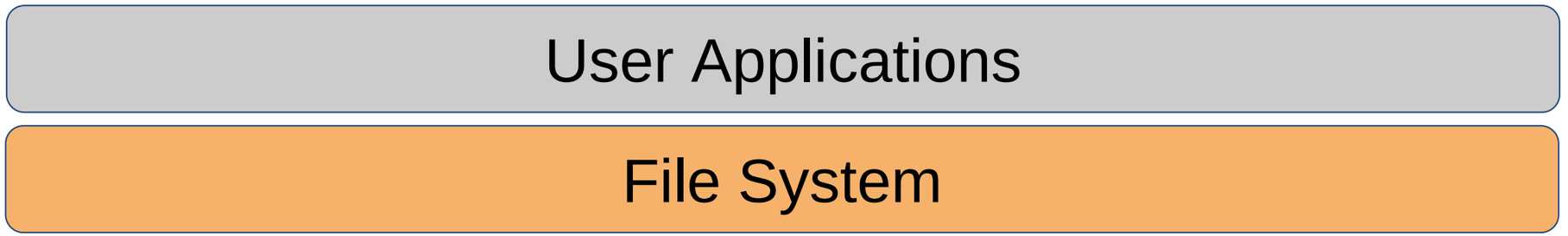


Massive Parallelism from Flash Channels

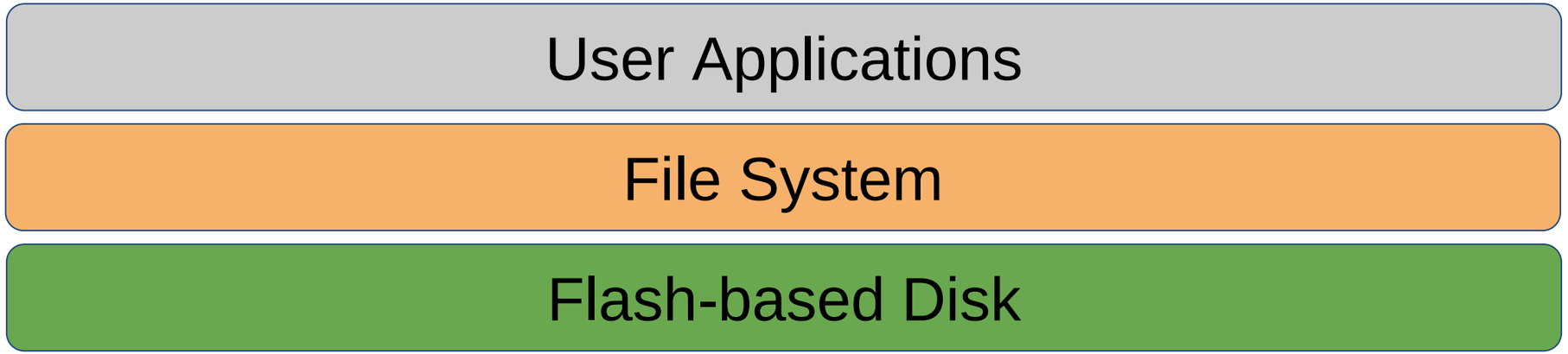
How SSDs Used in Modern Computer Systems?

User Applications

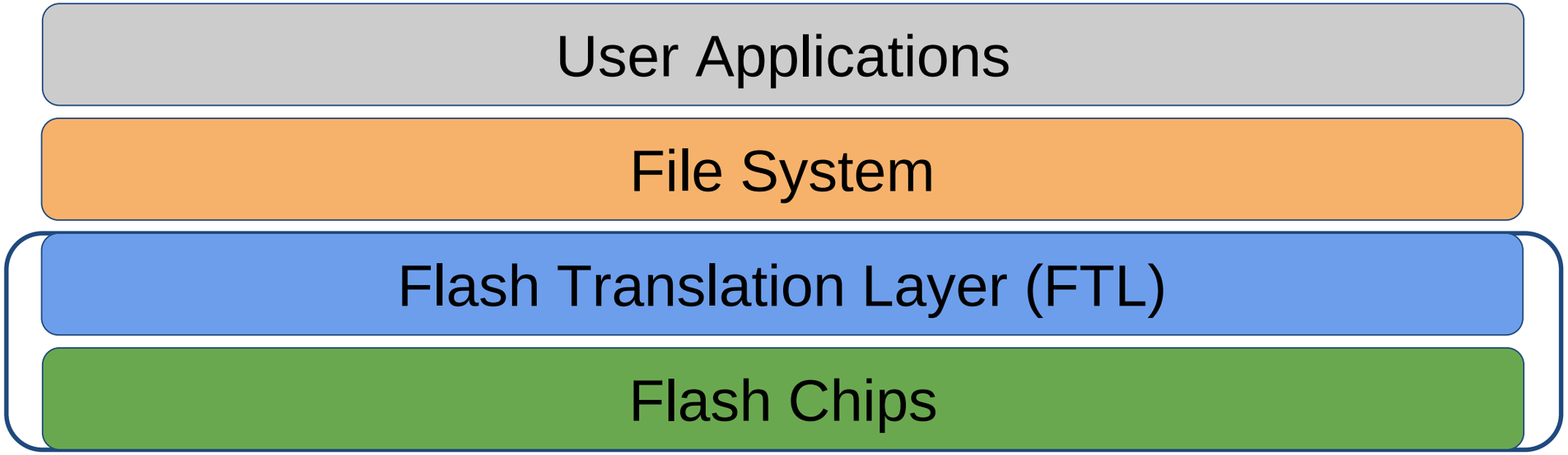
How SSDs Used in Modern Computer Systems?



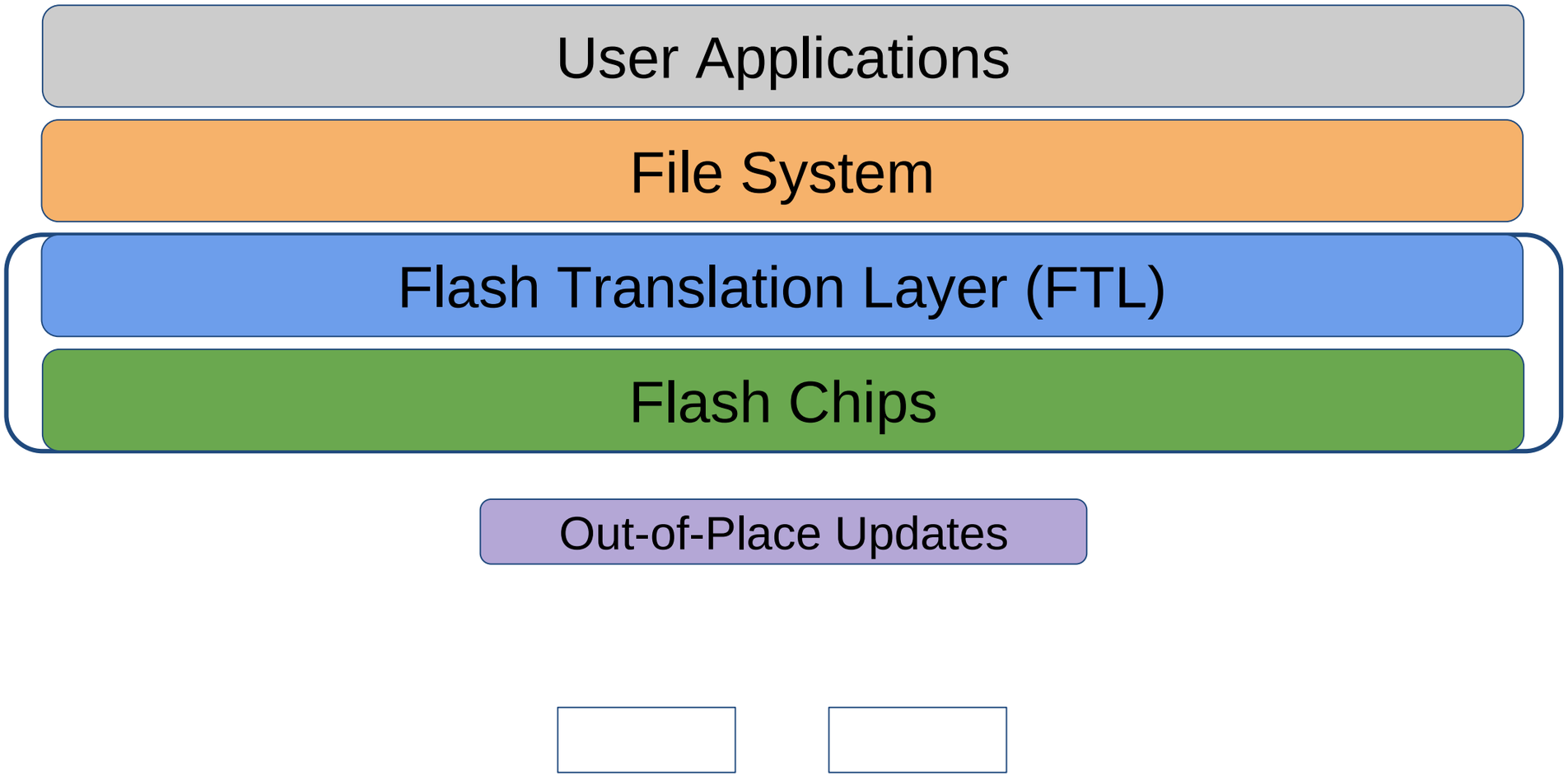
How SSDs Used in Modern Computer Systems?



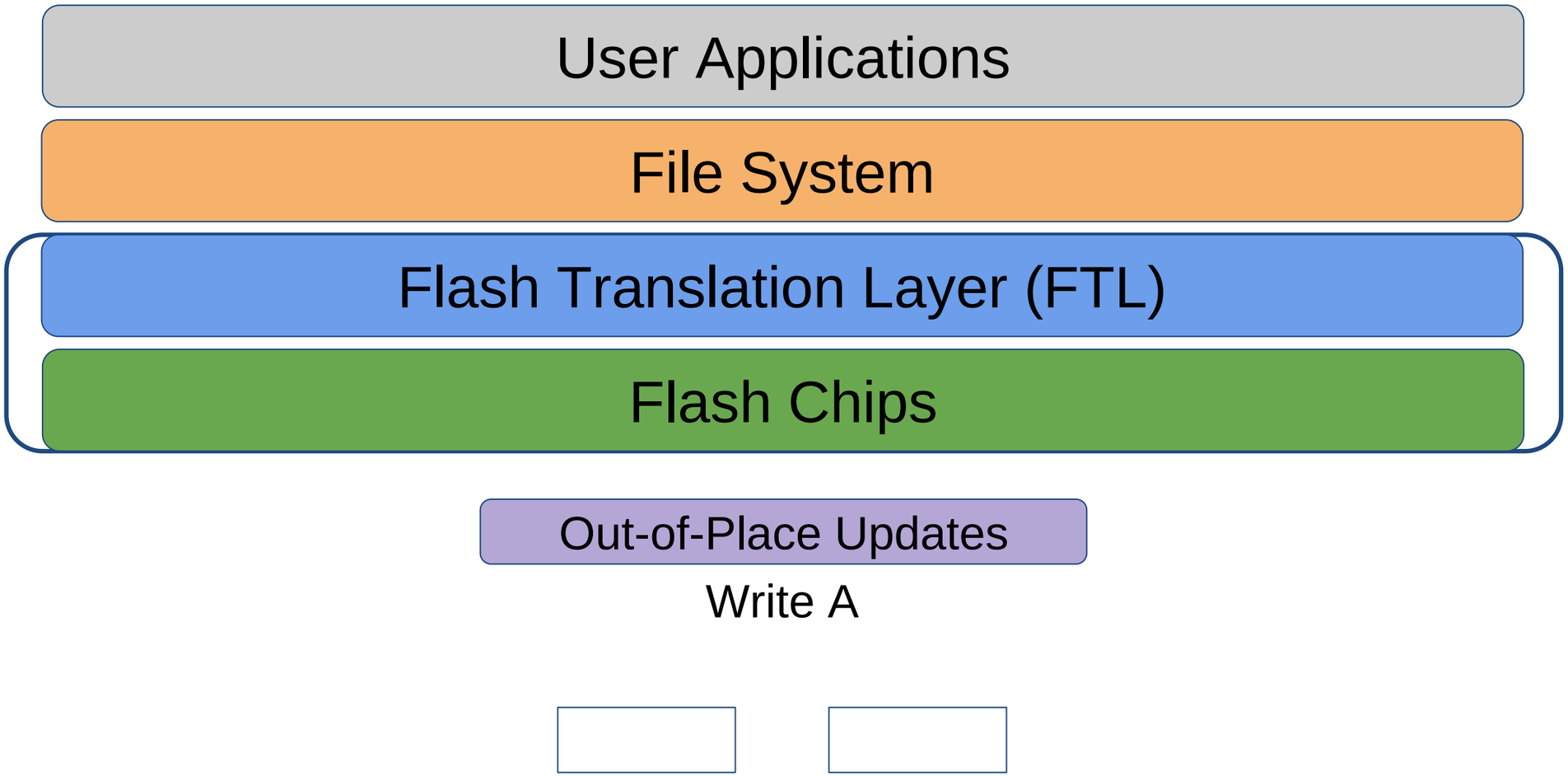
How SSDs Used in Modern Computer Systems?



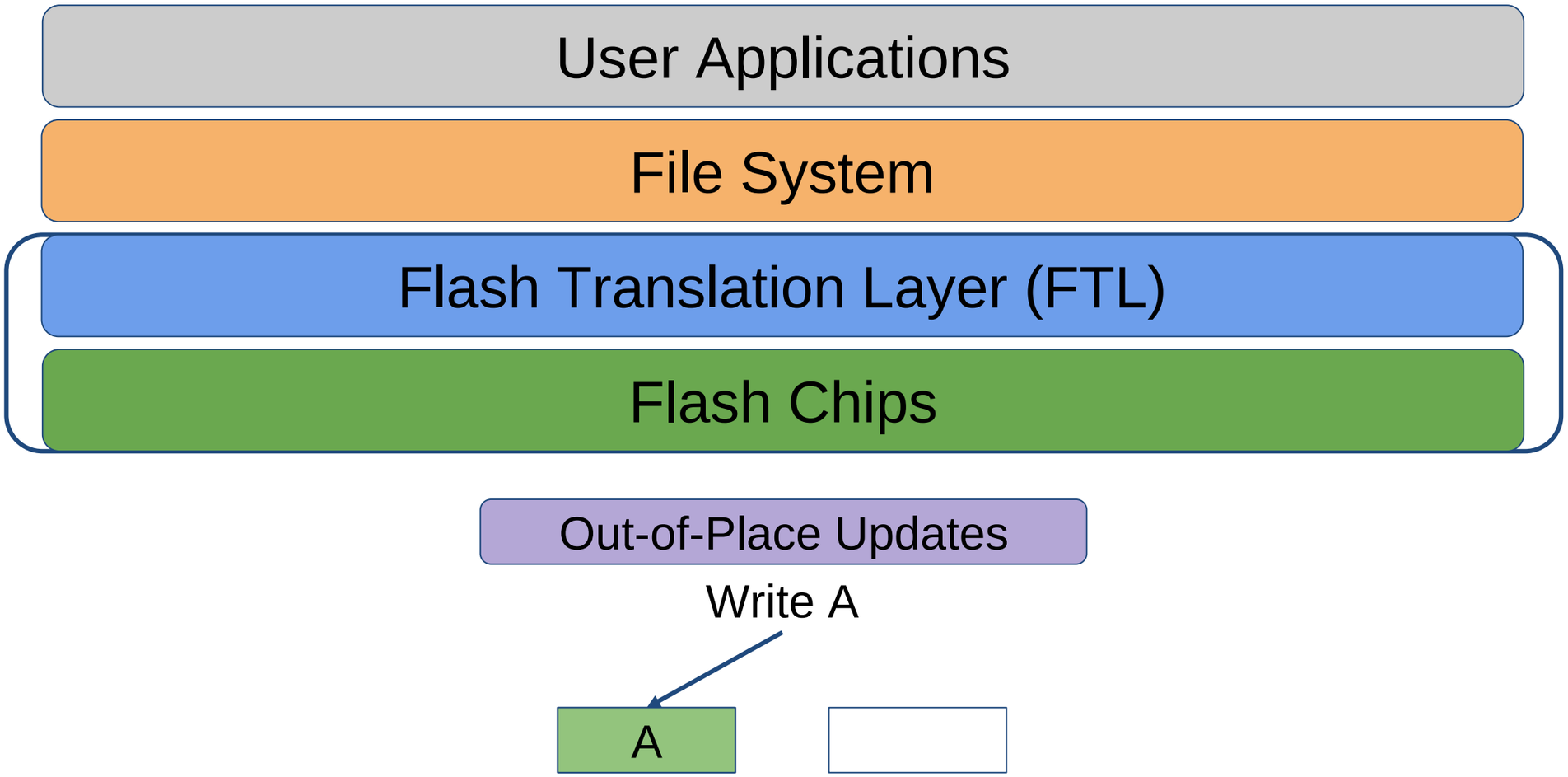
How SSDs Used in Modern Computer Systems?



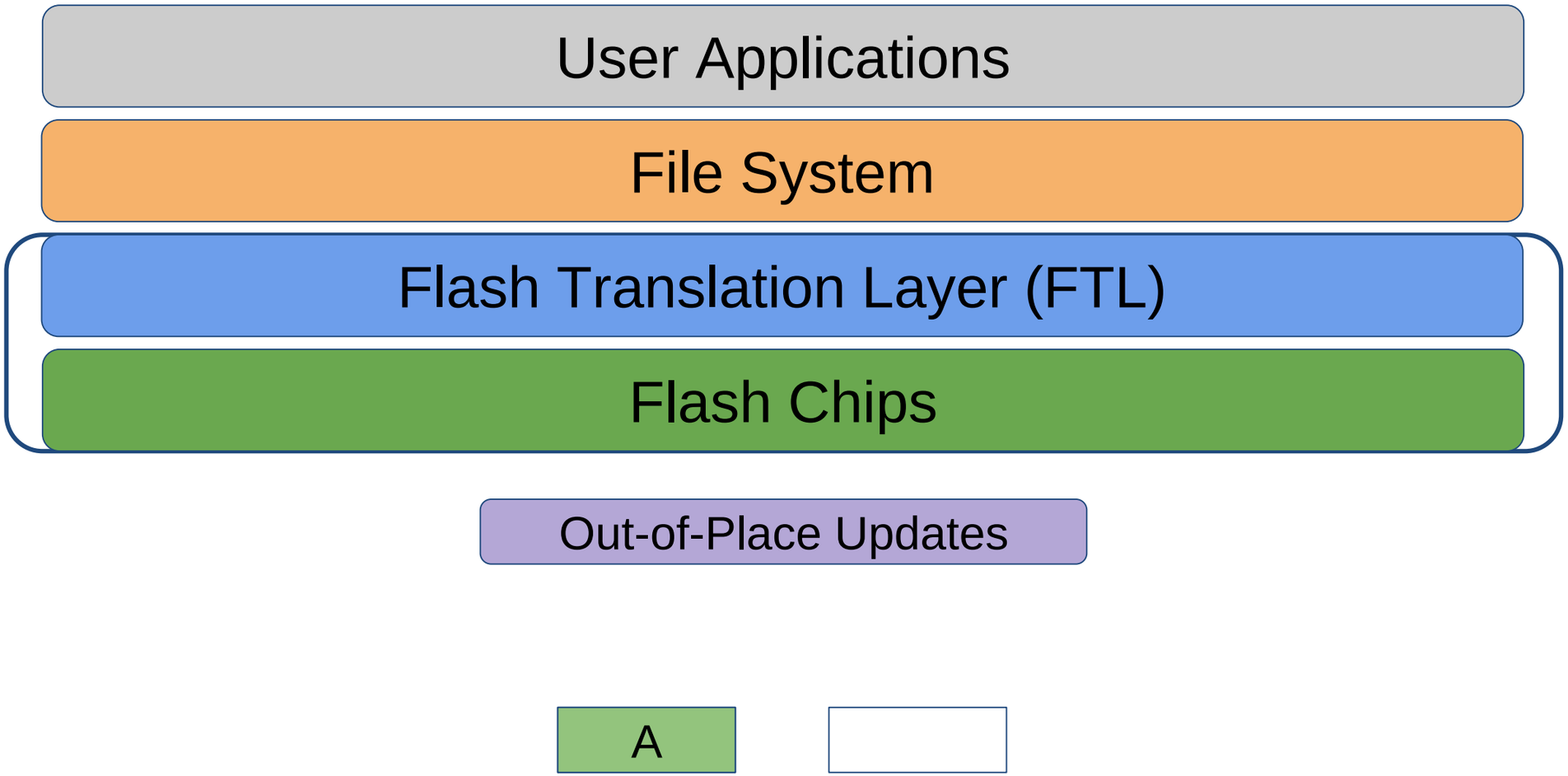
How SSDs Used in Modern Computer Systems?



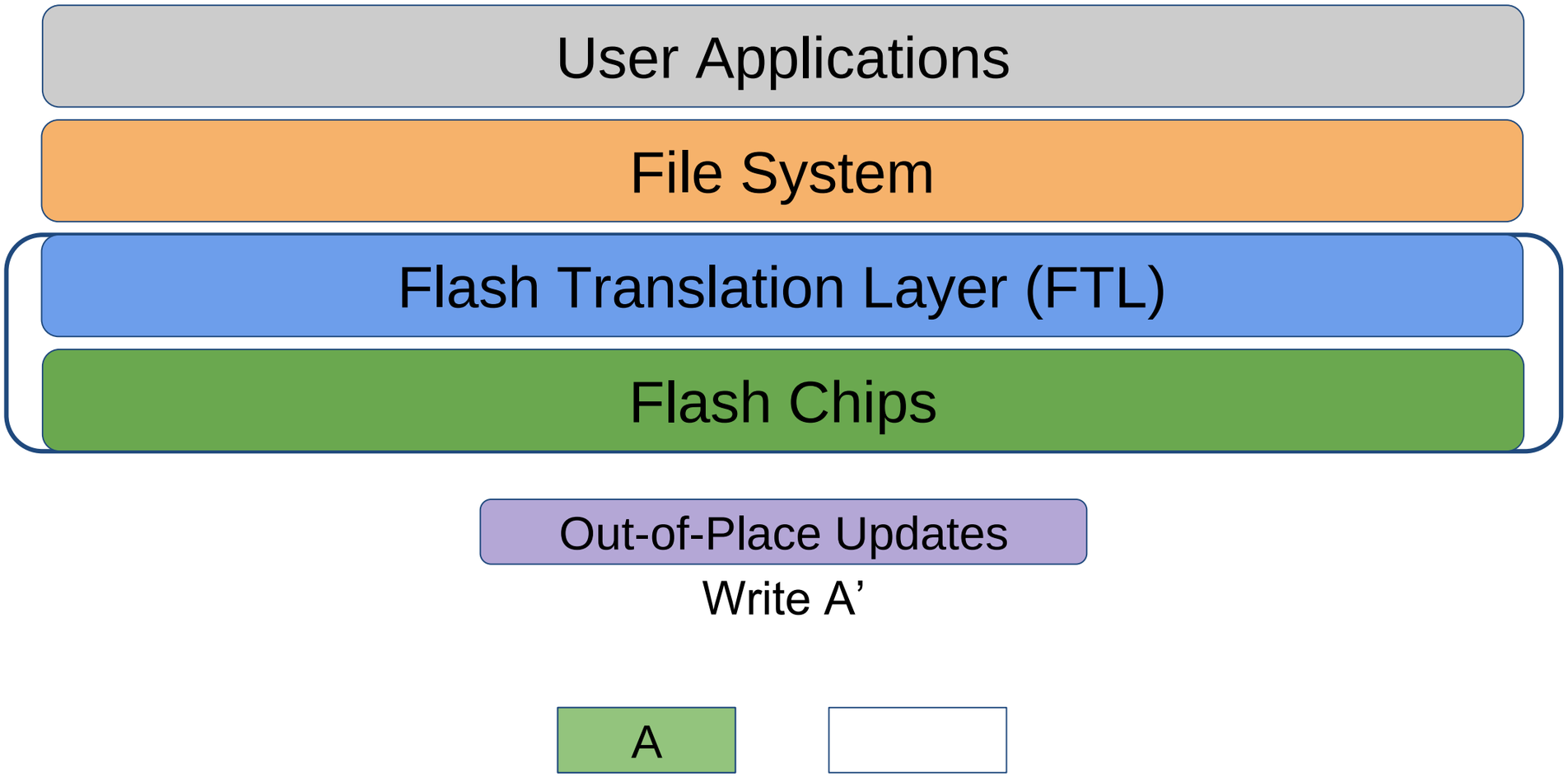
How SSDs Used in Modern Computer Systems?



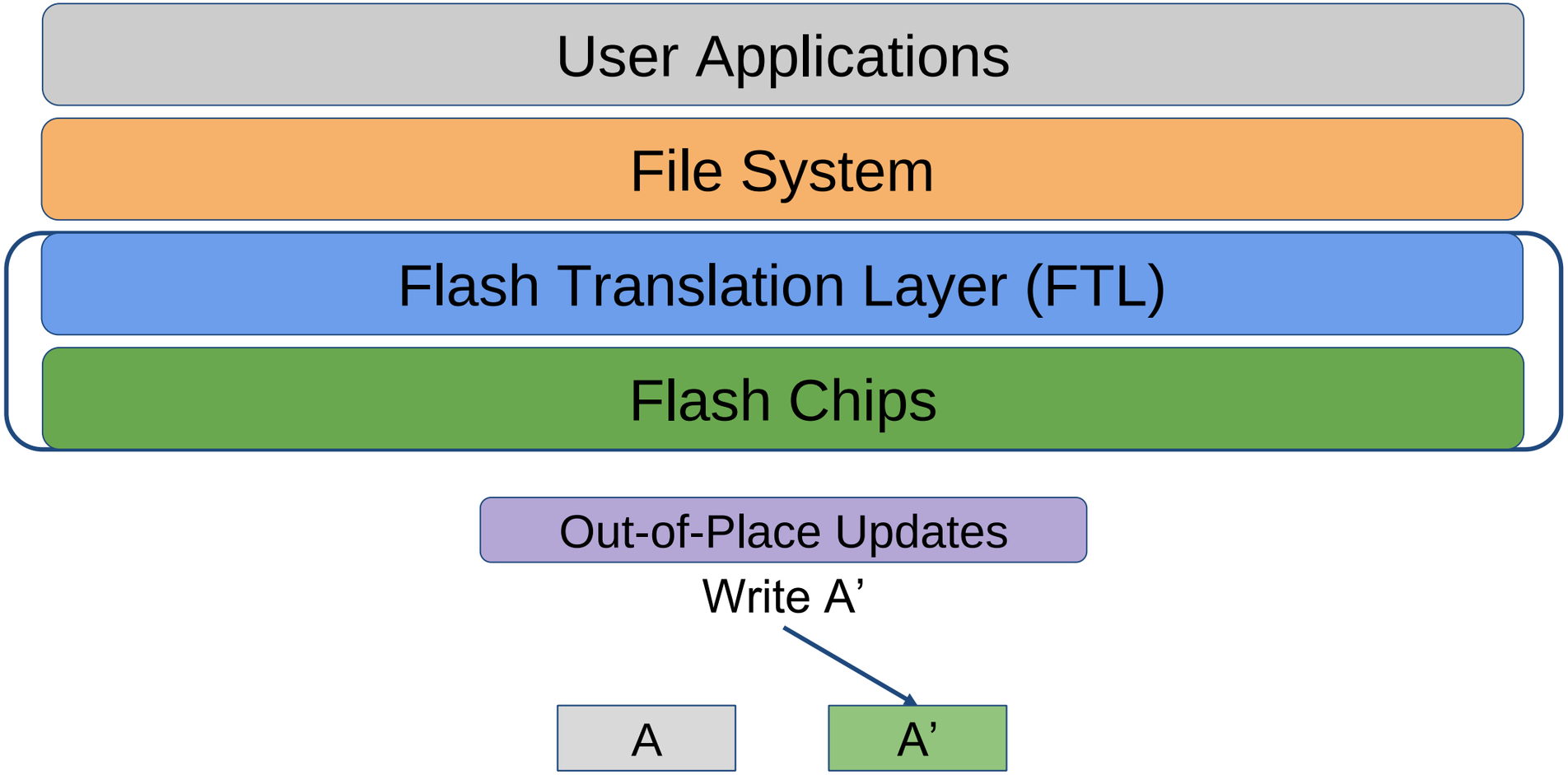
How SSDs Used in Modern Computer Systems?



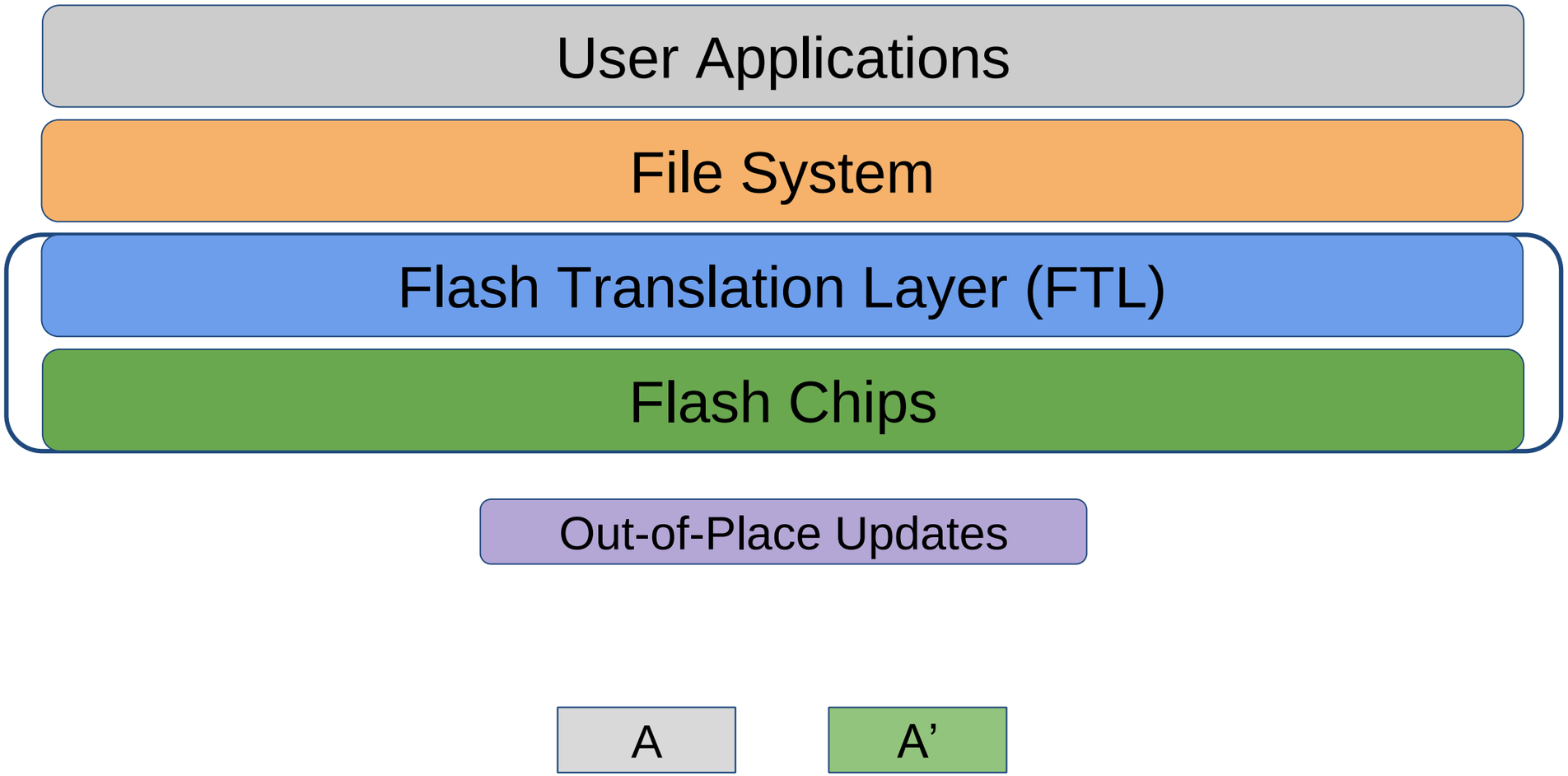
How SSDs Used in Modern Computer Systems?



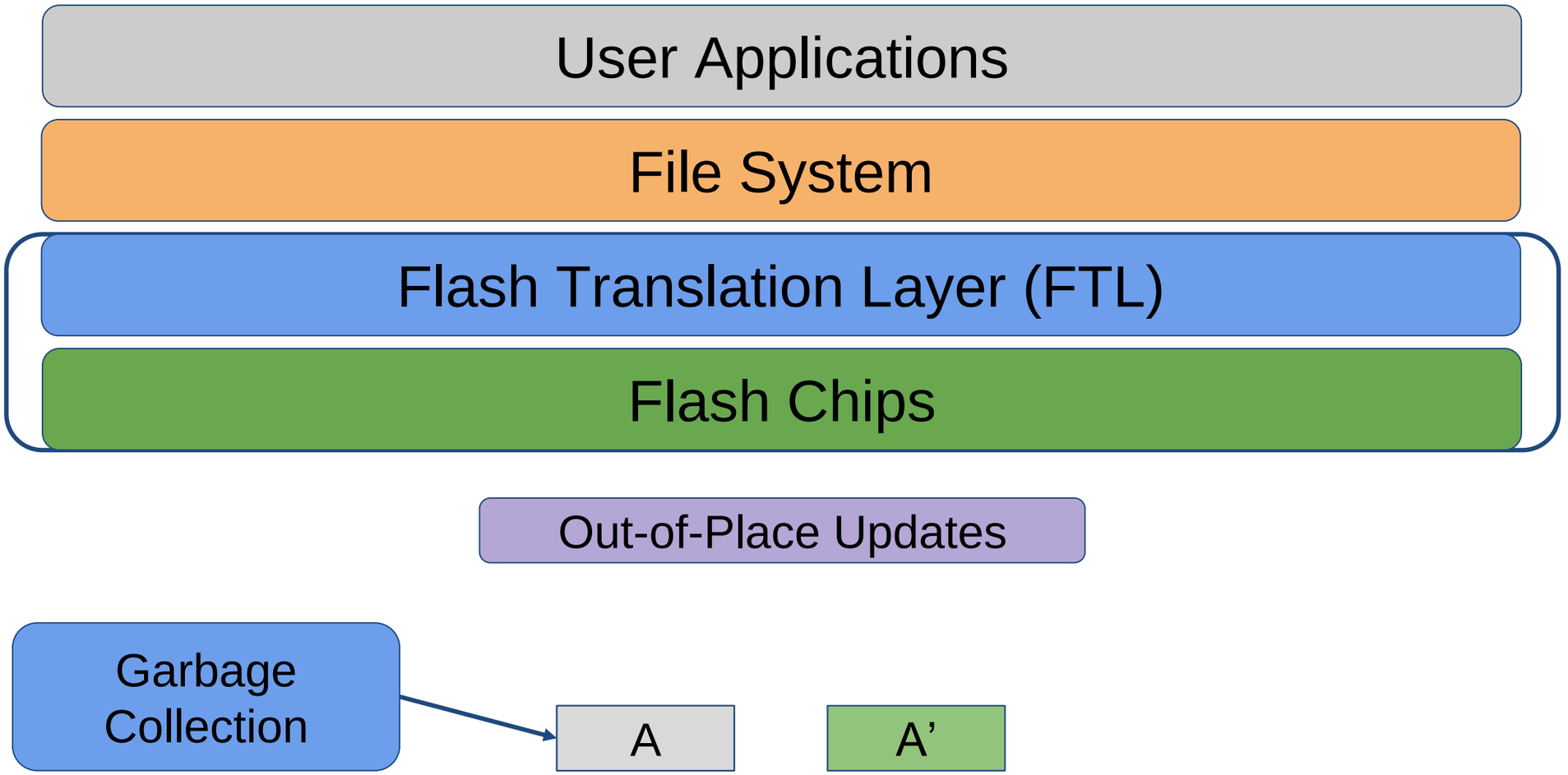
How SSDs Used in Modern Computer Systems?



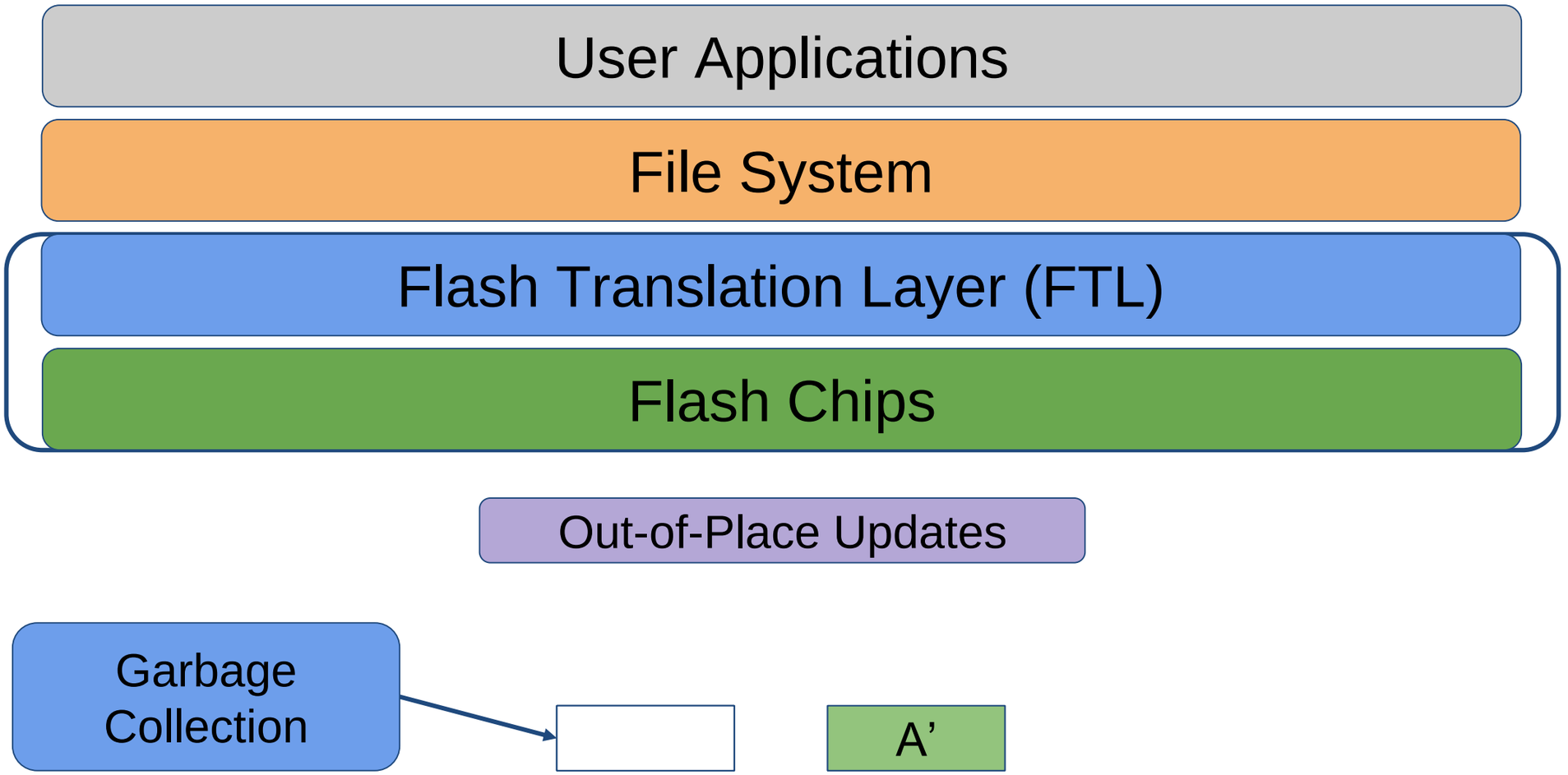
How SSDs Used in Modern Computer Systems?



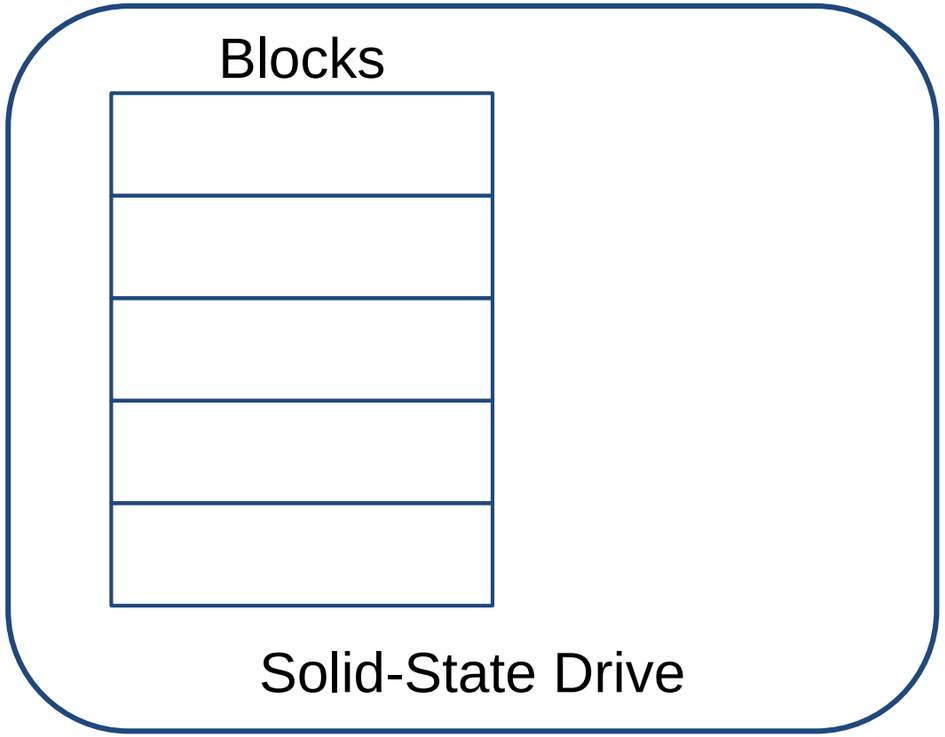
How SSDs Used in Modern Computer Systems?

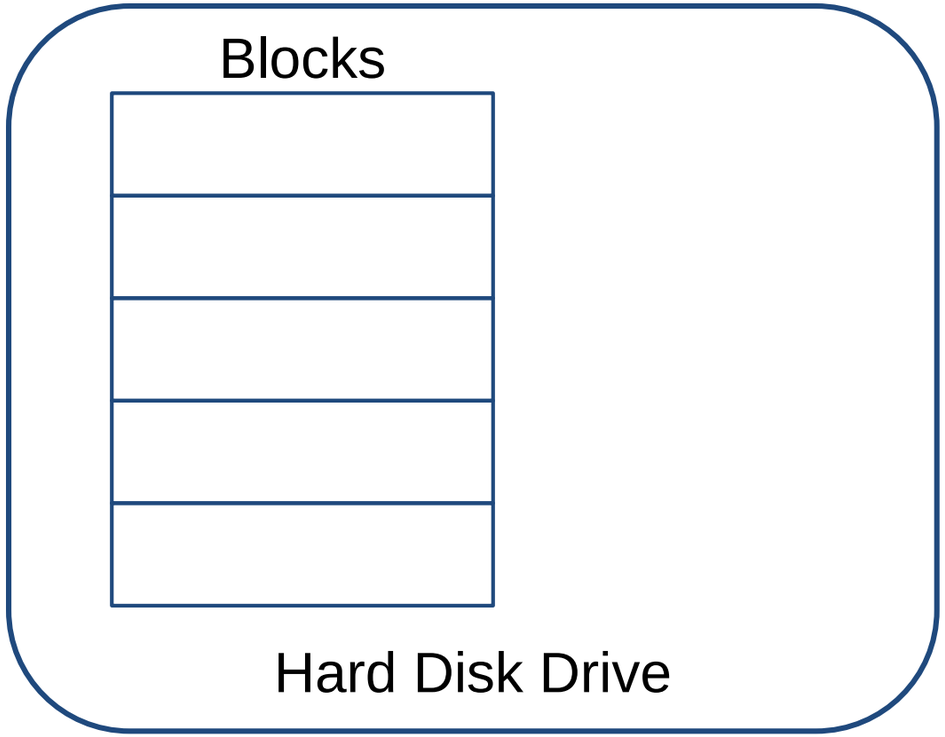


How SSDs Used in Modern Computer Systems?

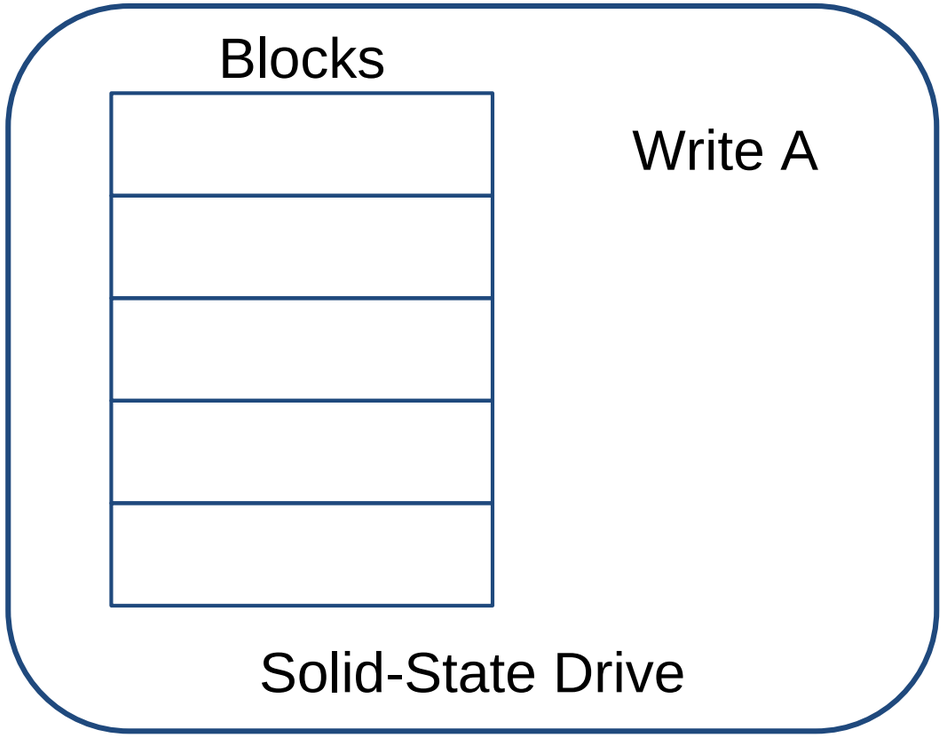


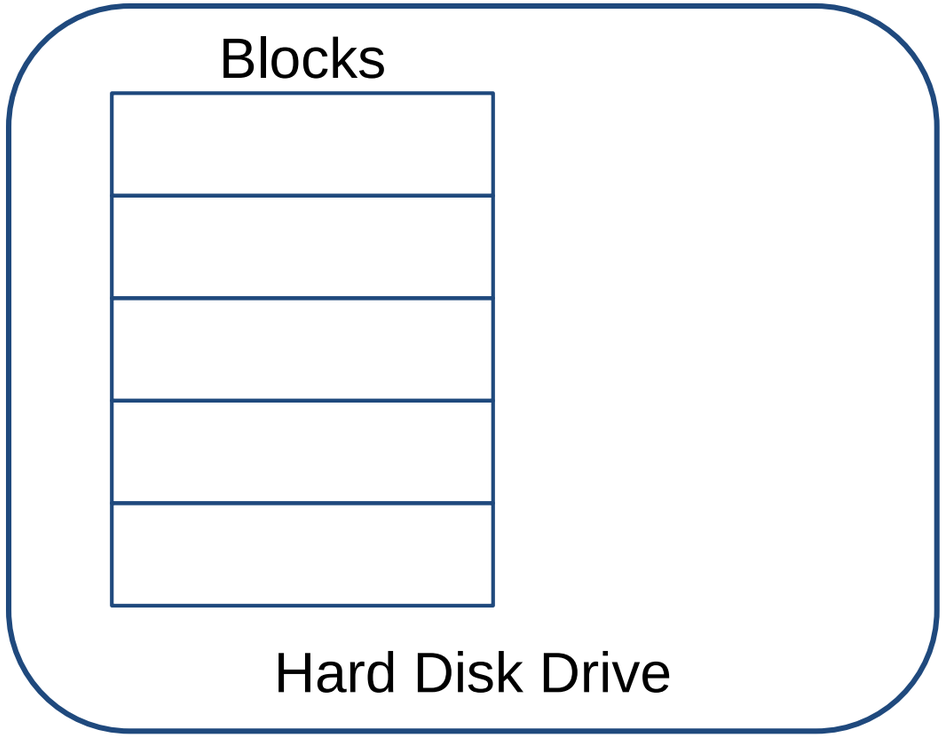
Retaining Storage-States with Out-of-Place Update



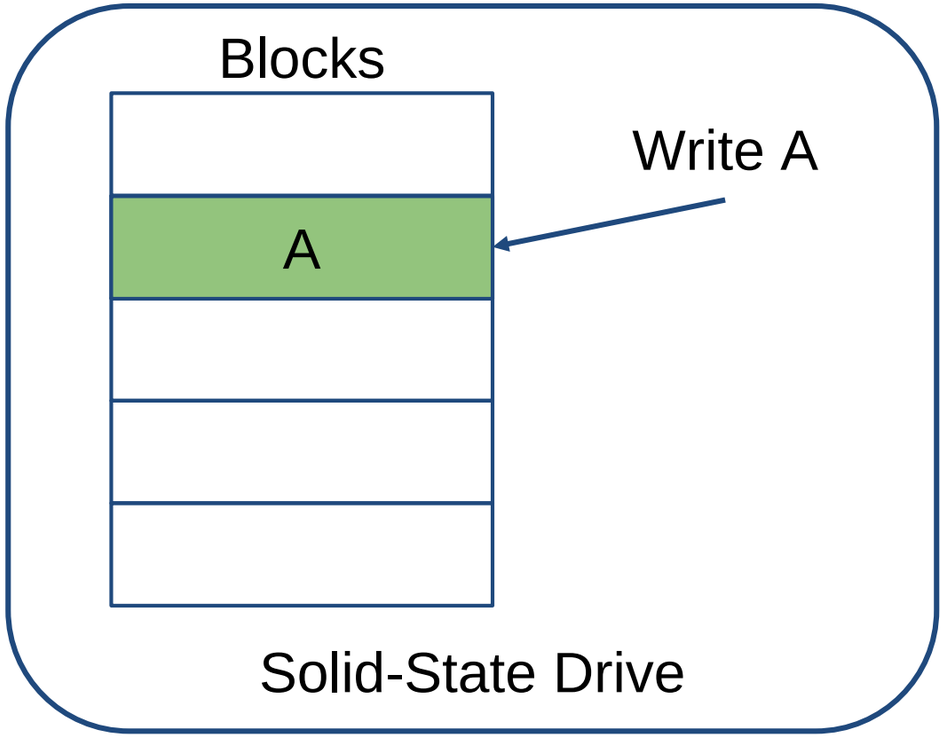


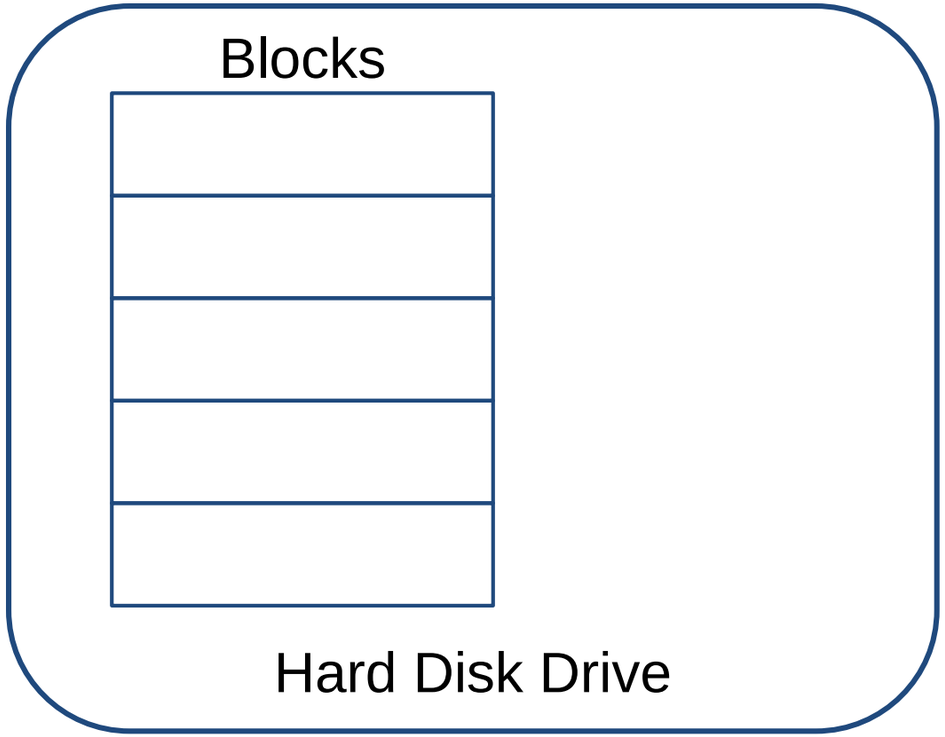
Retaining Storage-States with Out-of-Place Update



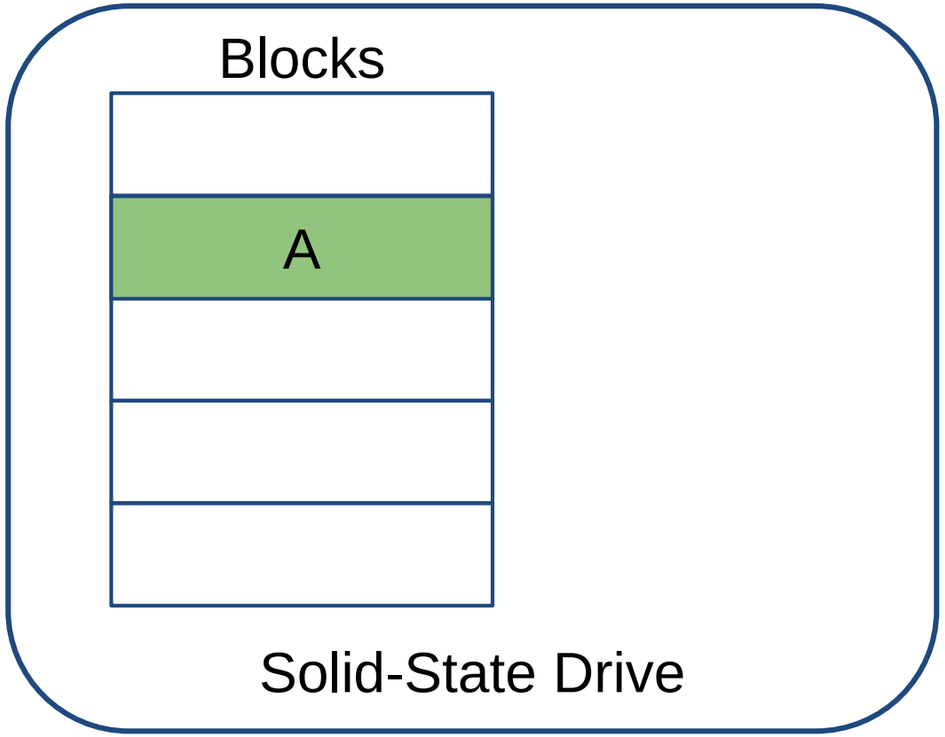


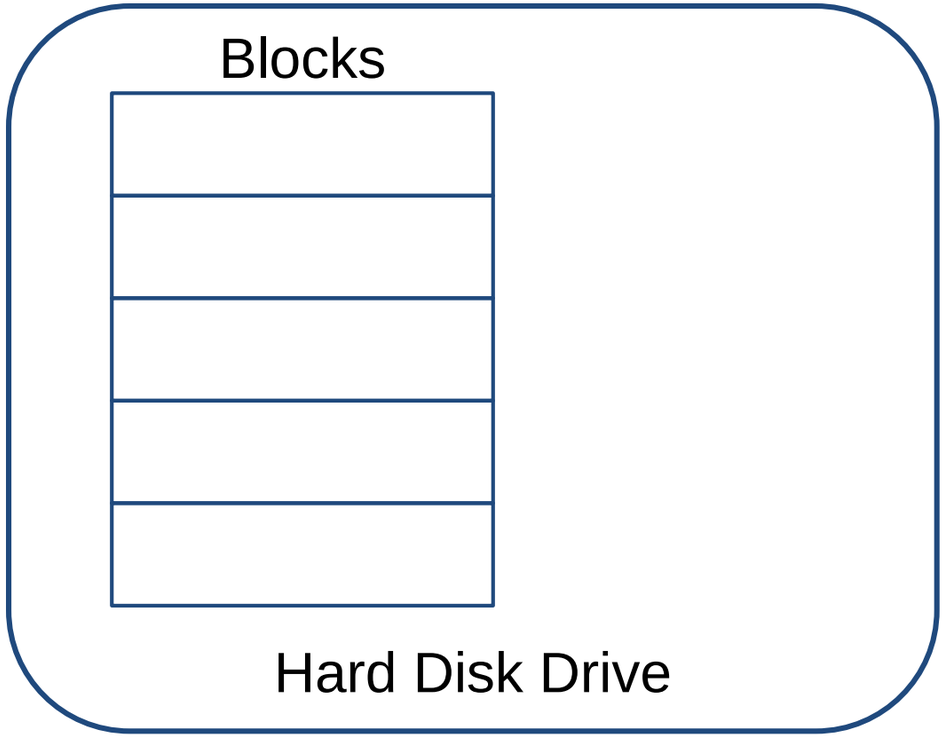
Retaining Storage-States with Out-of-Place Update



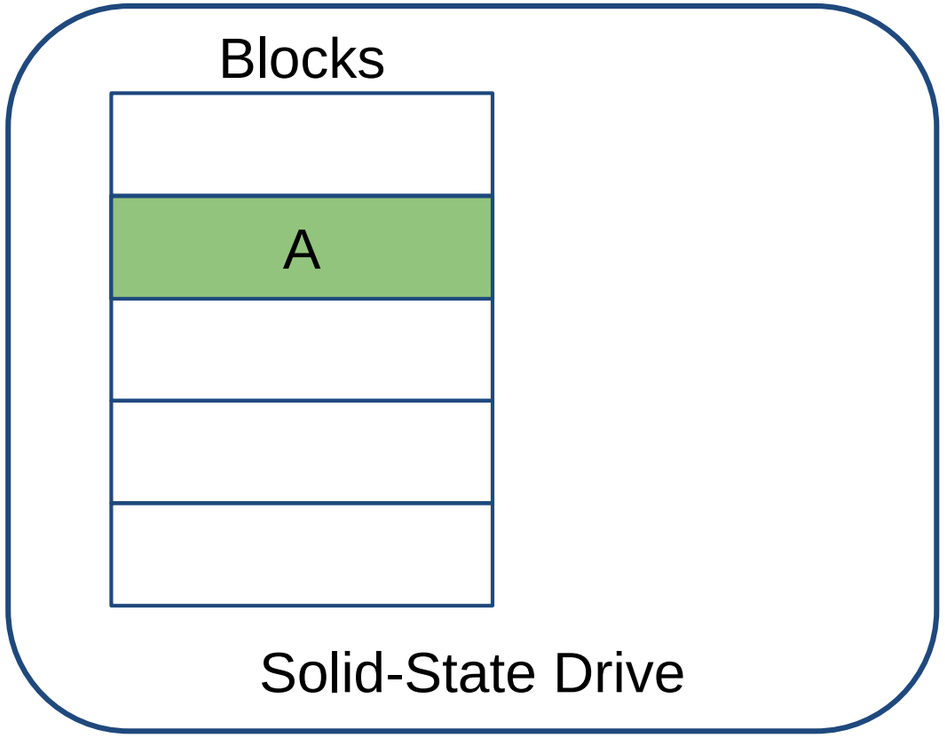


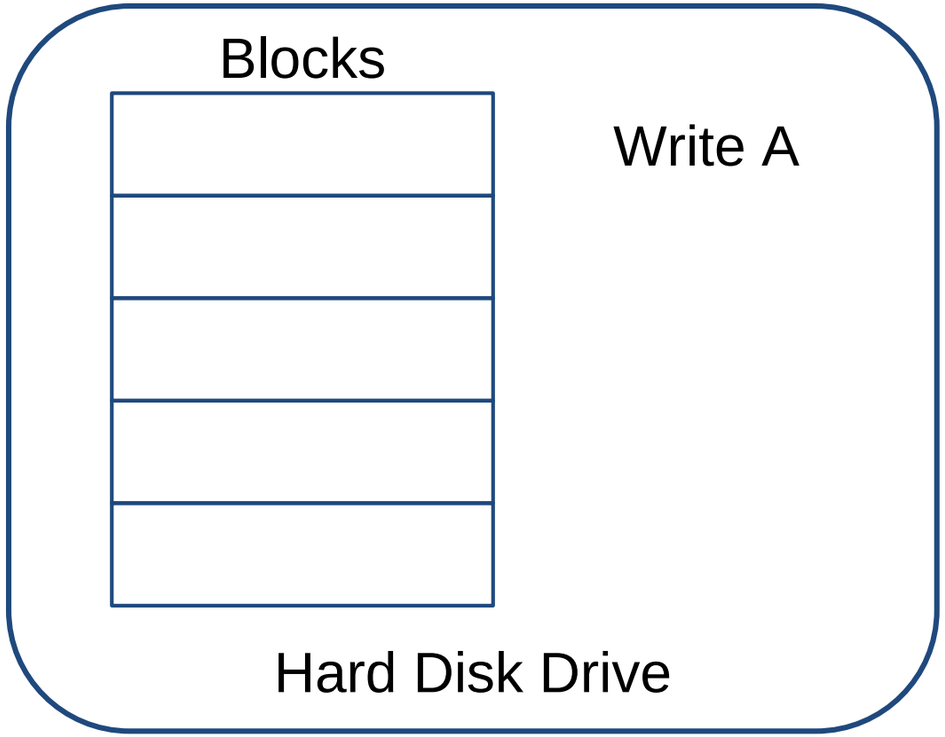
Retaining Storage-States with Out-of-Place Update



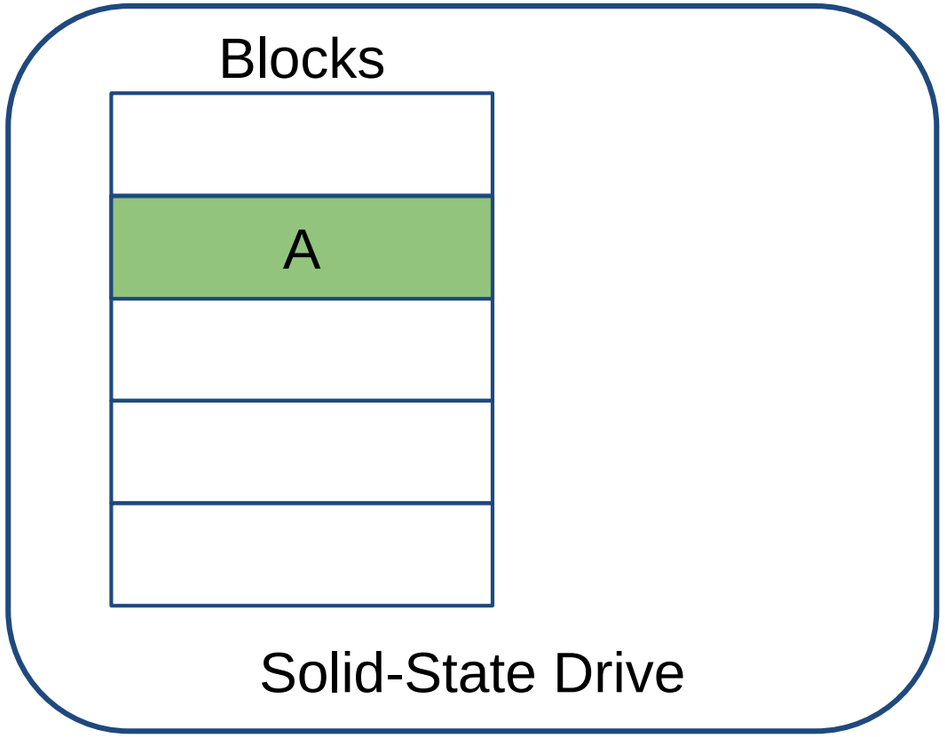


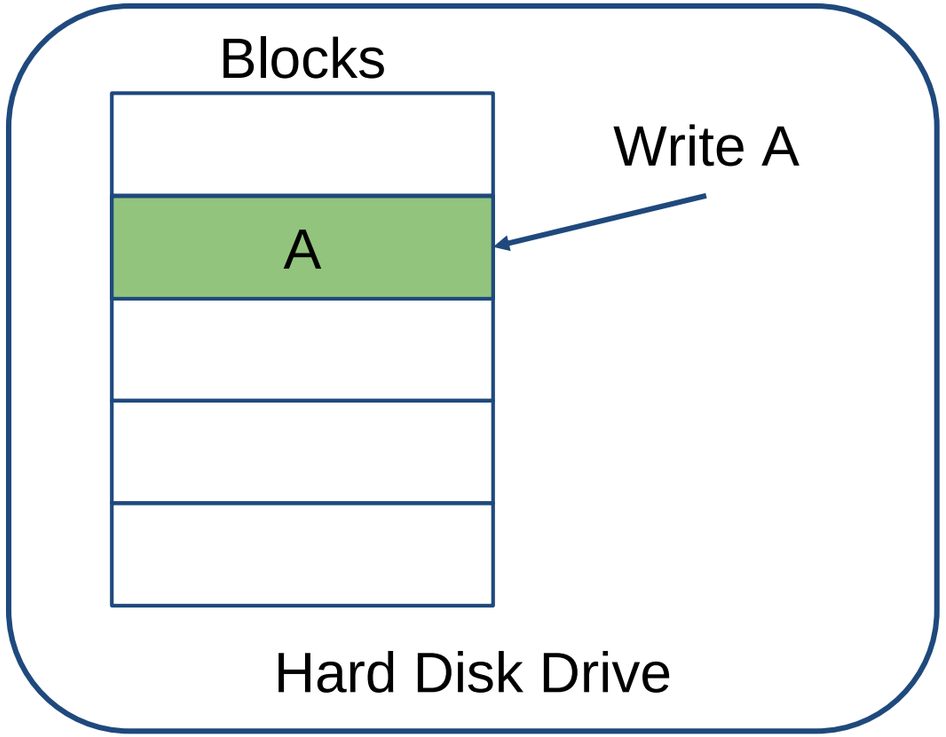
Retaining Storage-States with Out-of-Place Update



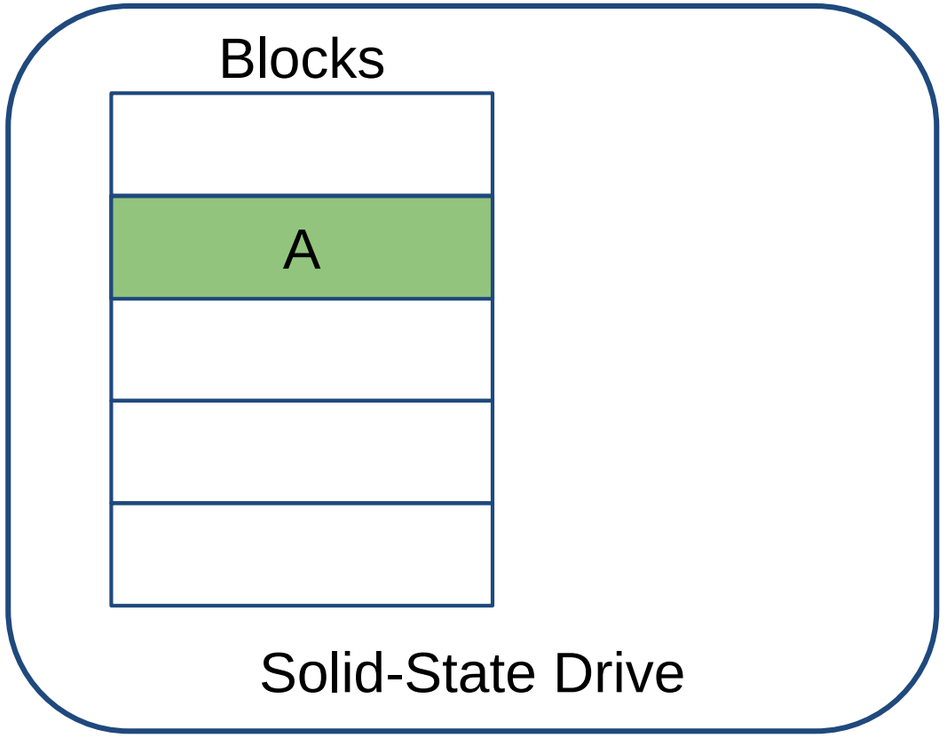


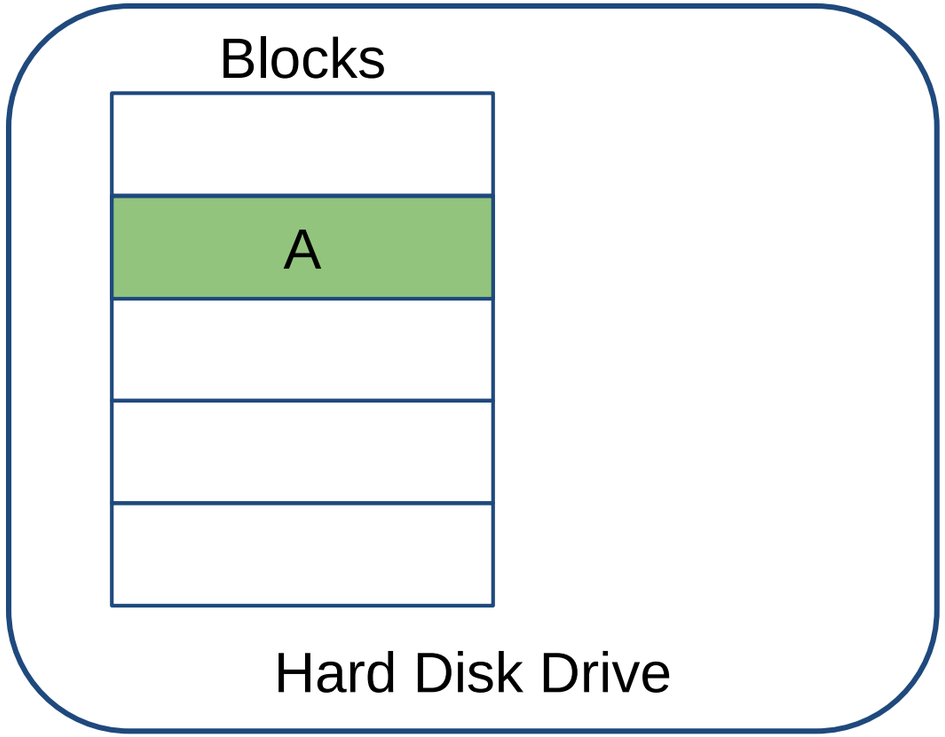
Retaining Storage-States with Out-of-Place Update



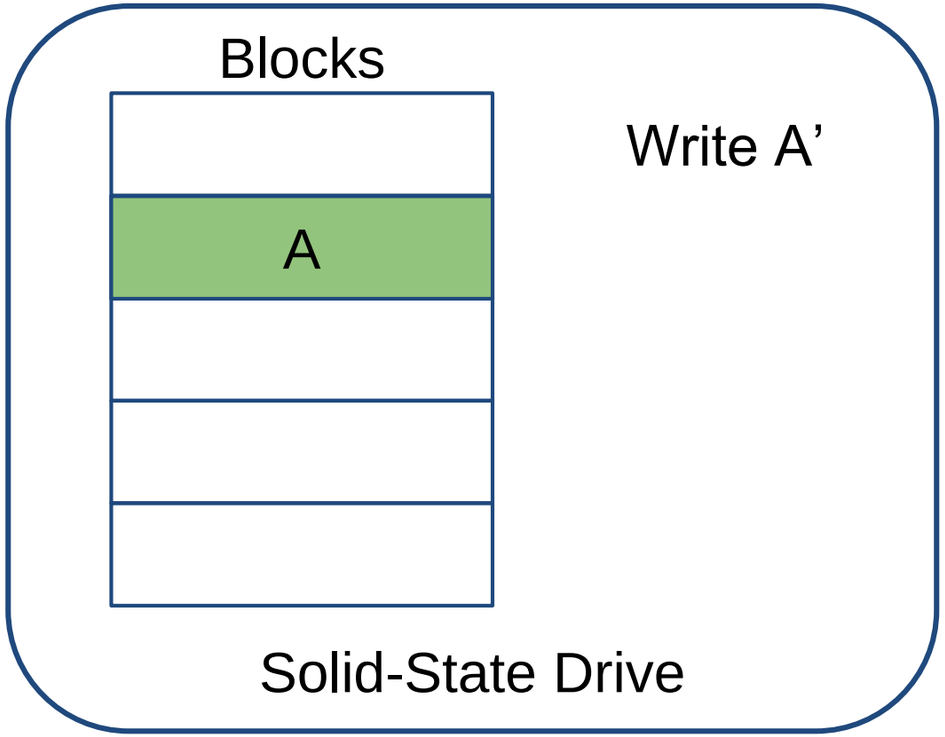


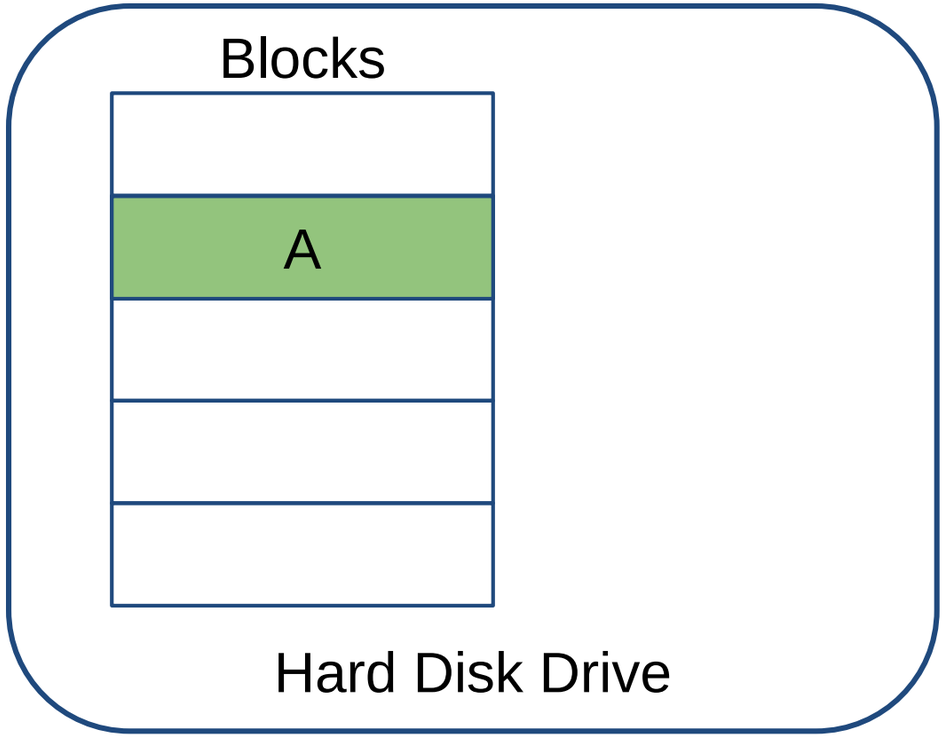
Retaining Storage-States with Out-of-Place Update



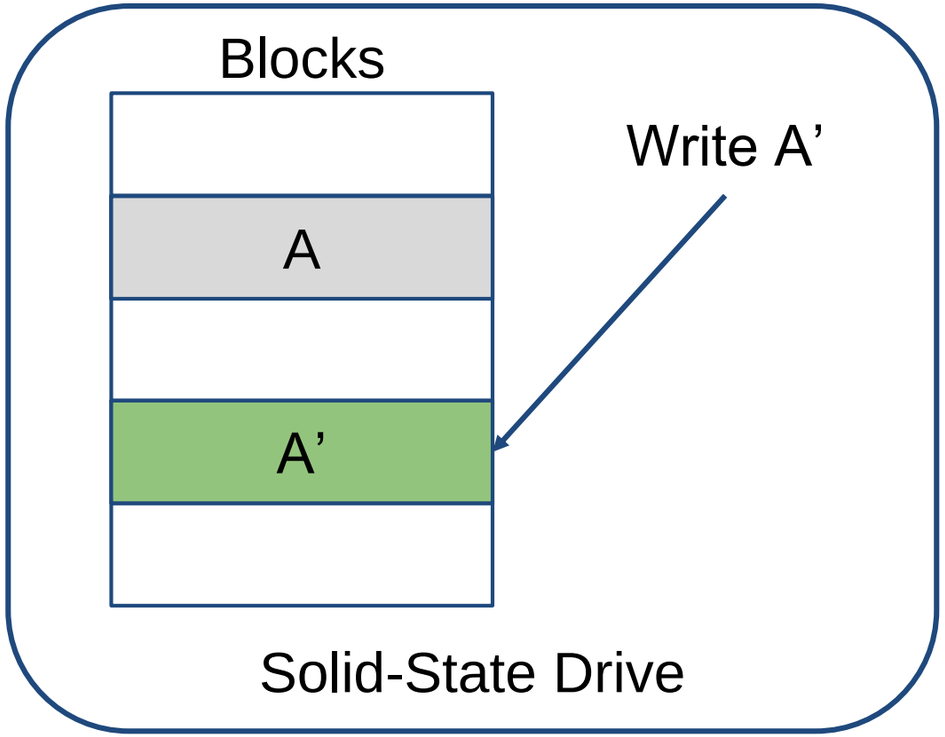


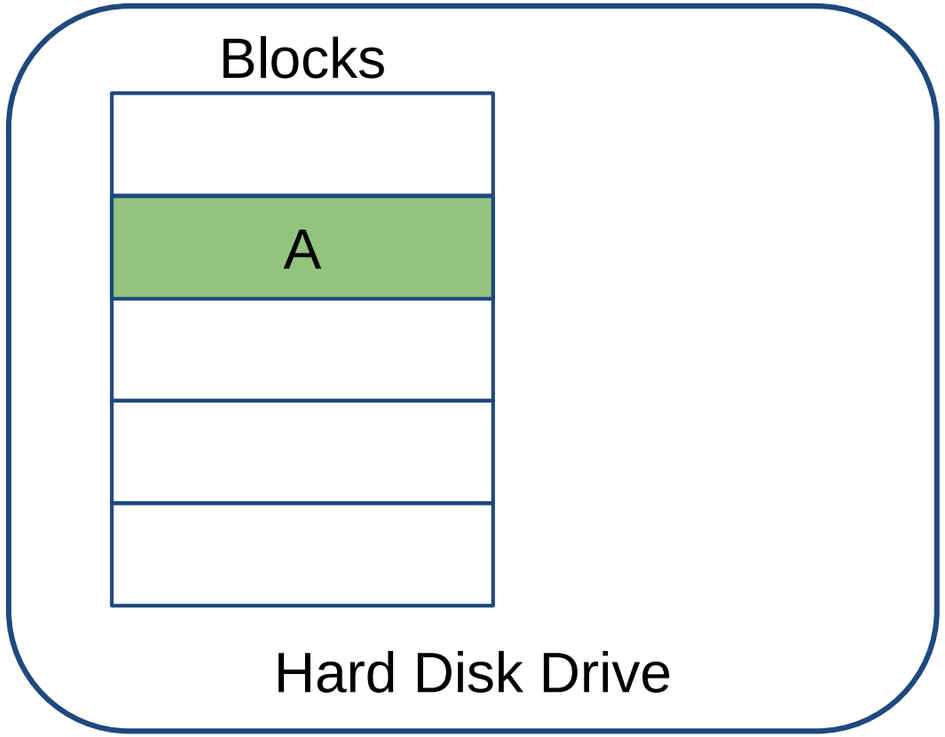
Retaining Storage-States with Out-of-Place Update



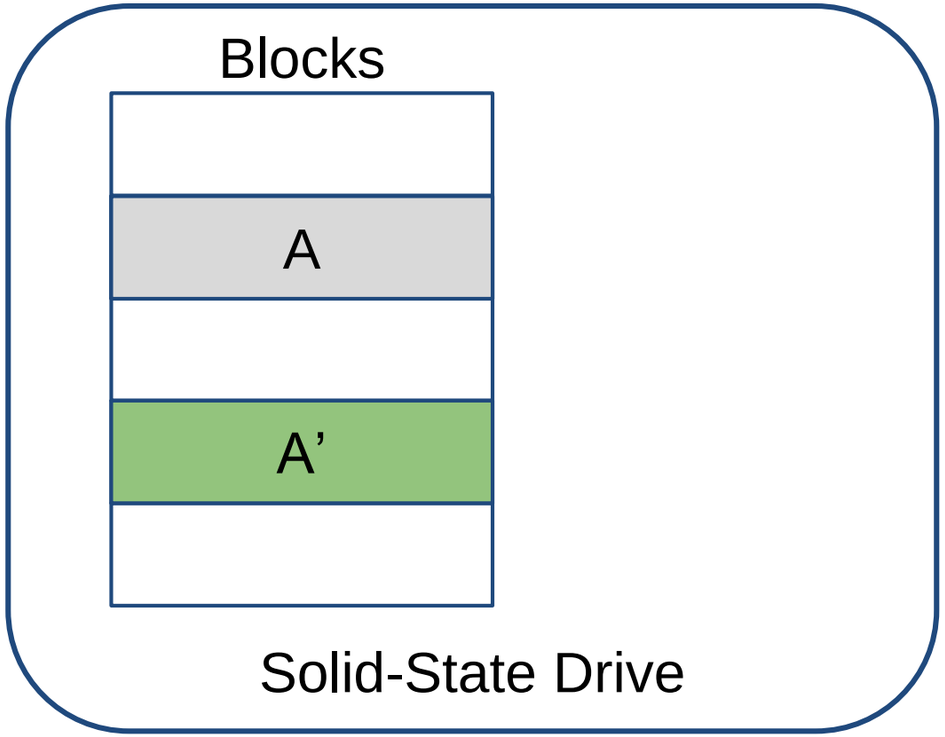


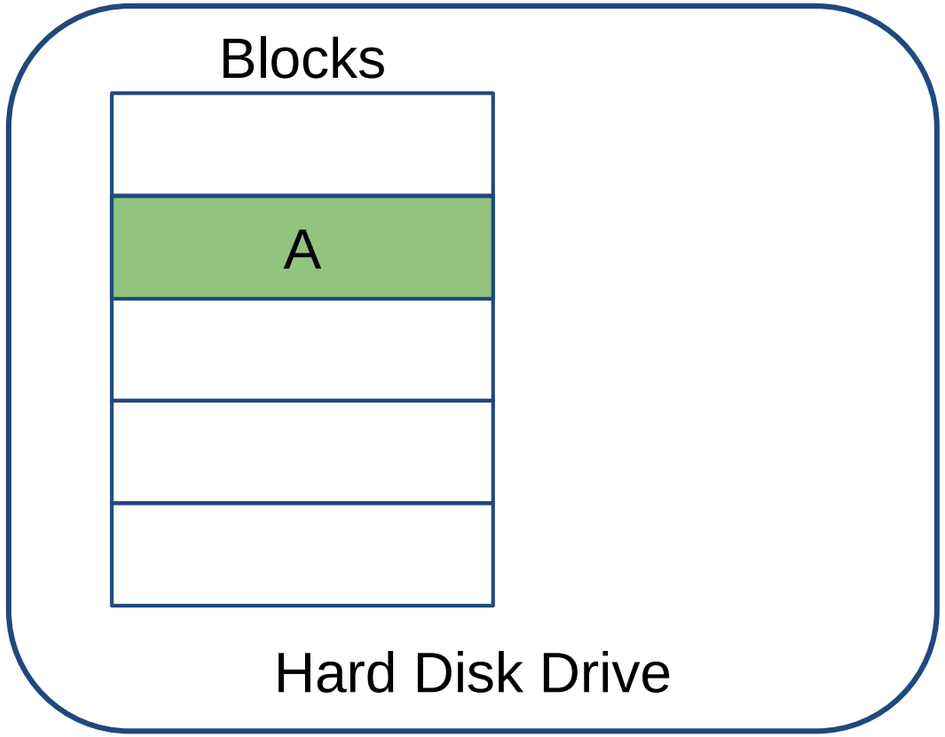
Retaining Storage-States with Out-of-Place Update



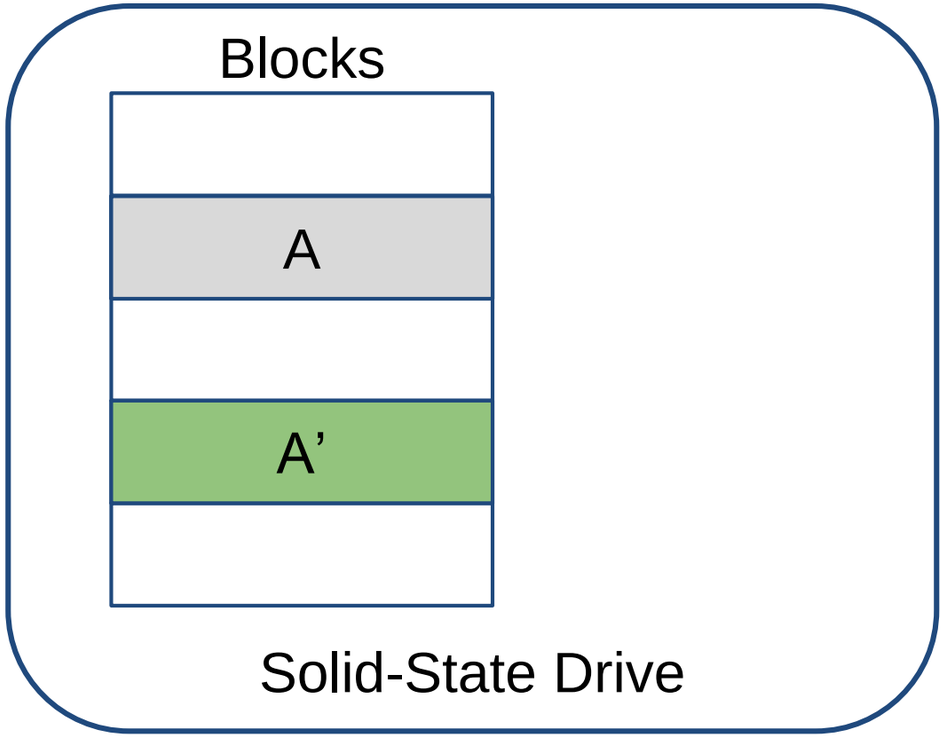


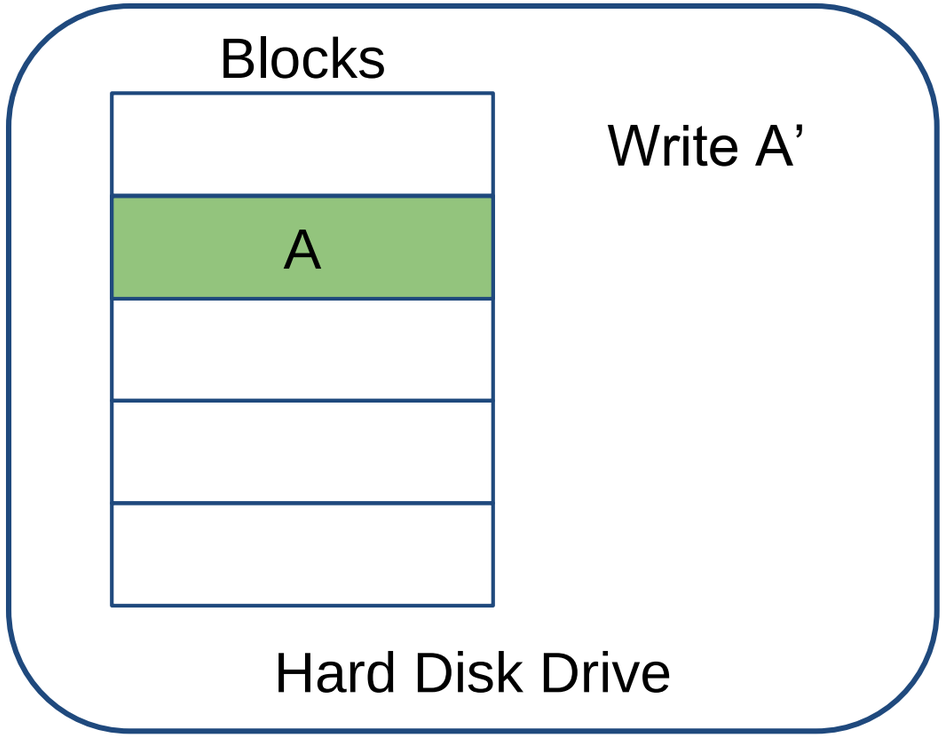
Retaining Storage-States with Out-of-Place Update



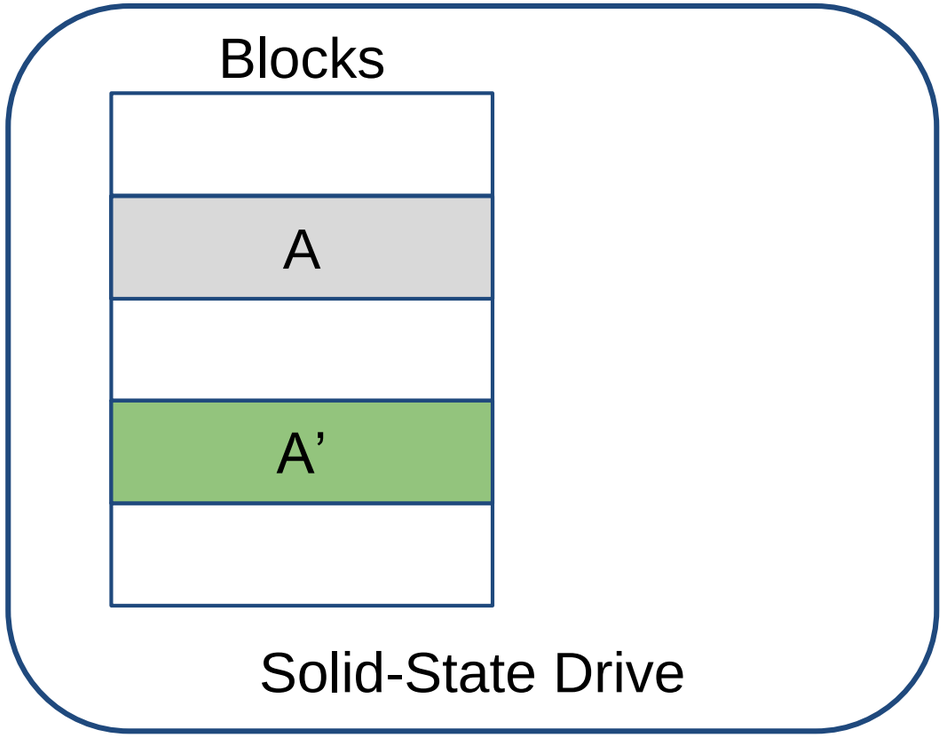


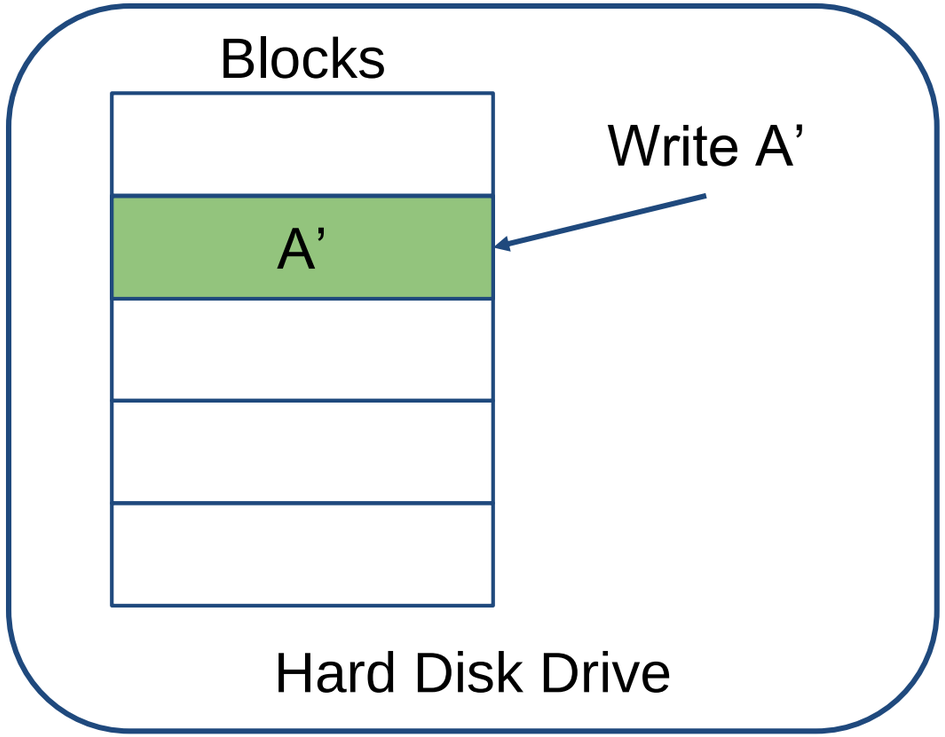
Retaining Storage-States with Out-of-Place Update



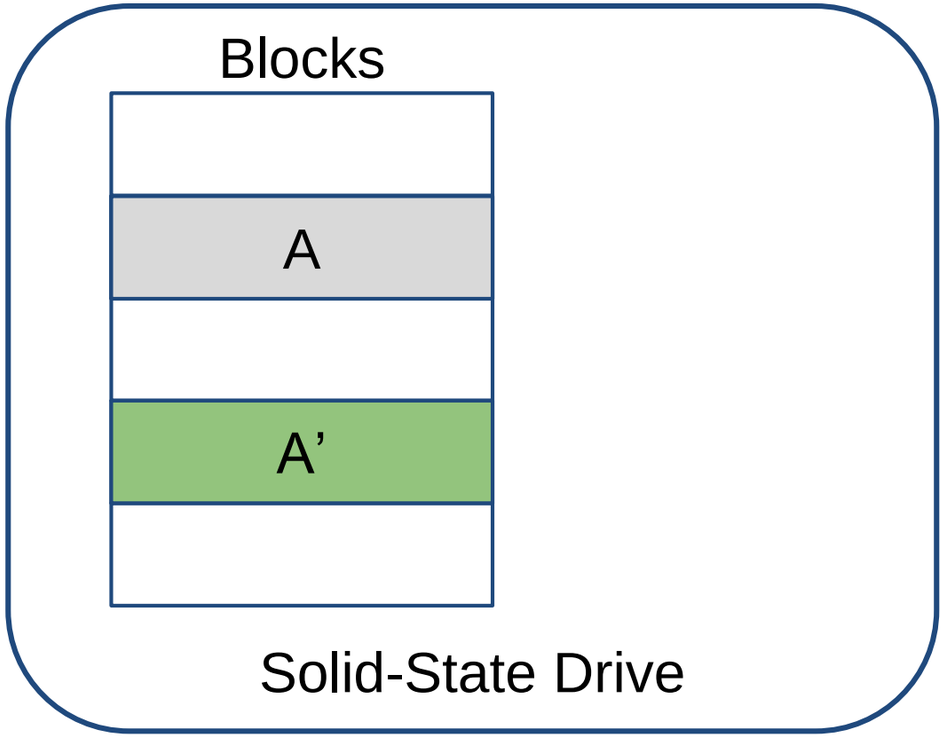


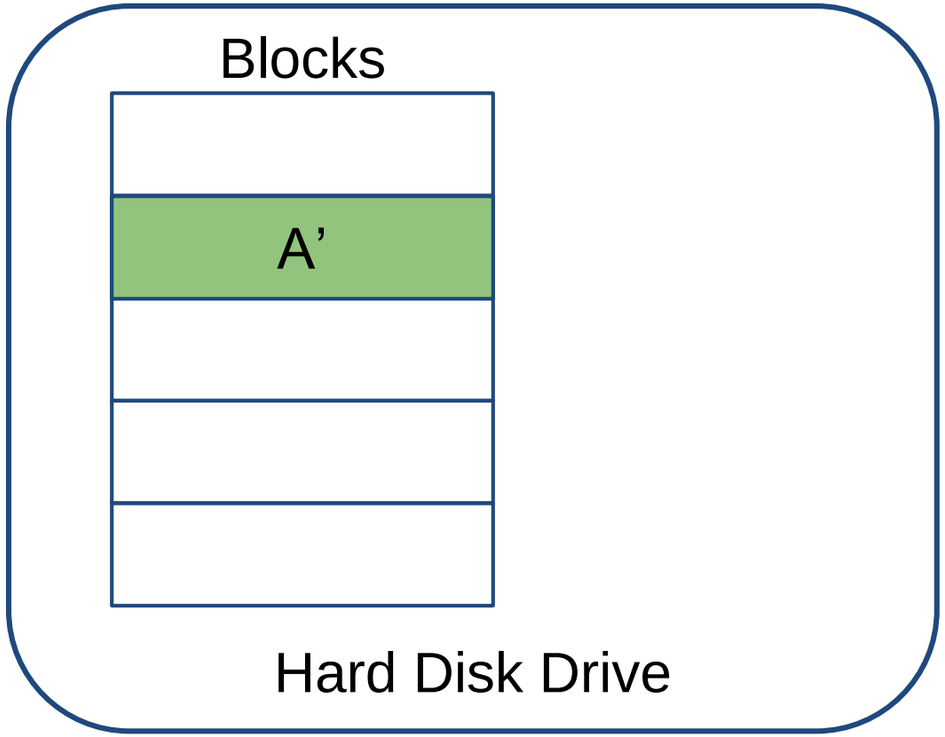
Retaining Storage-States with Out-of-Place Update



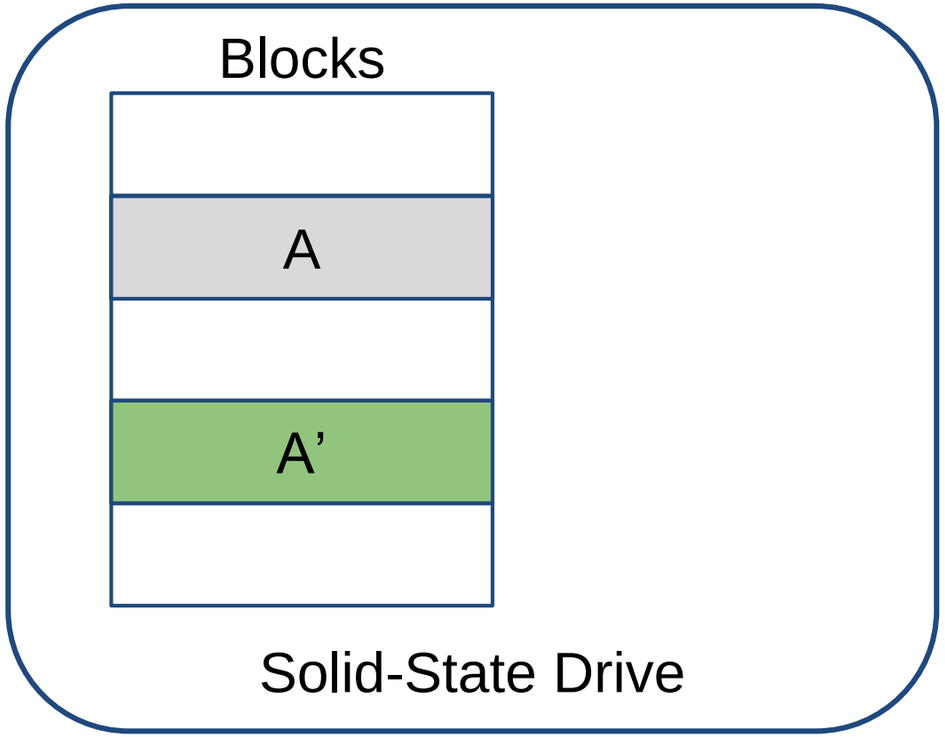


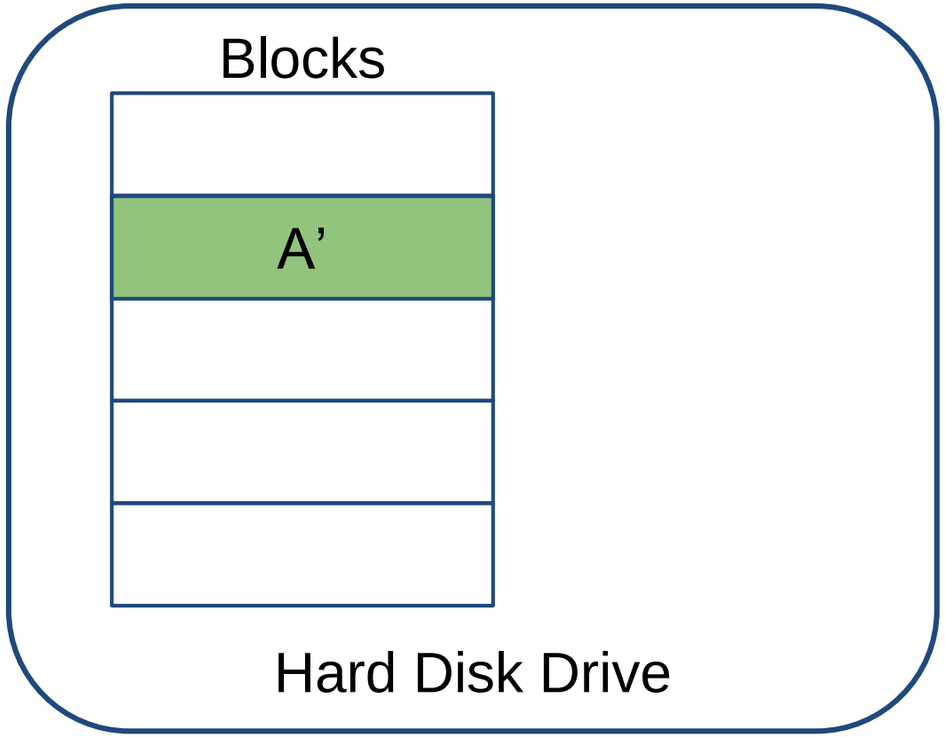
Retaining Storage-States with Out-of-Place Update





Retaining Storage-States with Out-of-Place Update



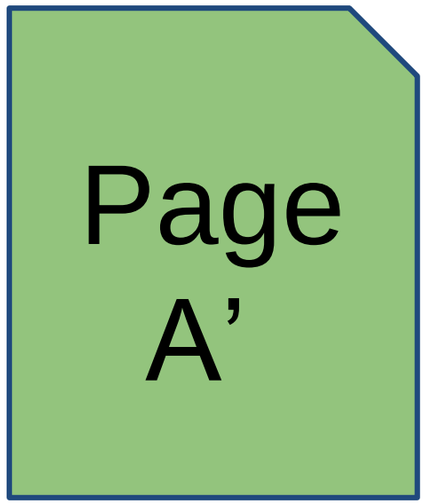


Past Storage States are Inherently Retained in Flash!

Limited Storage Capacity? Compression is Key!

Differences between pages are encoded as *deltas*

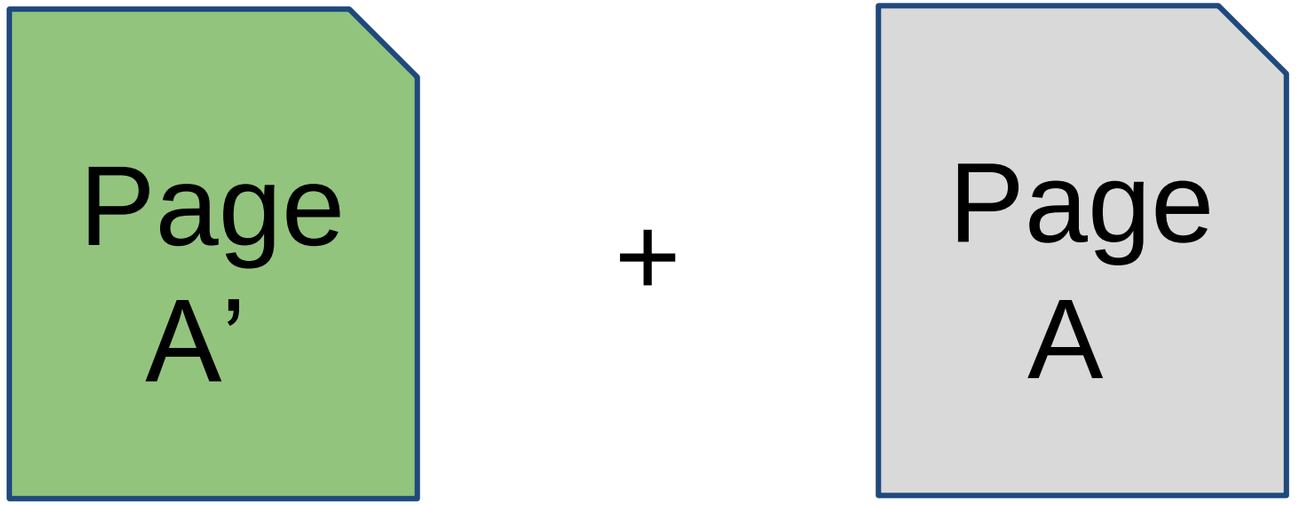
Limited Storage Capacity? Compression is Key!



Reference Flash Page

Differences between pages are encoded as *deltas*

Limited Storage Capacity? Compression is Key!

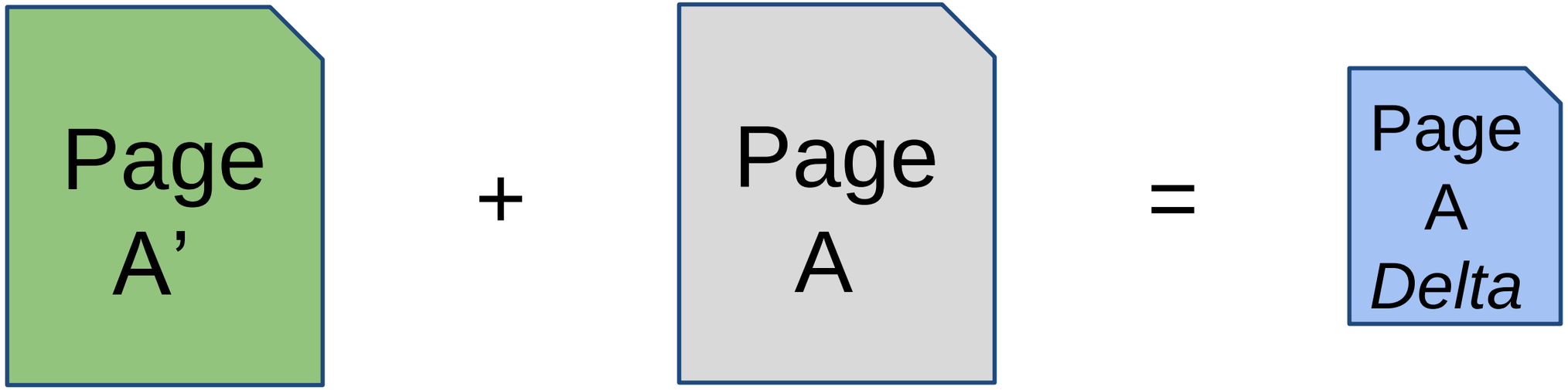


Reference Flash Page

Retained Invalid Flash Page

Differences between pages are encoded as *deltas*

Limited Storage Capacity? Compression is Key!



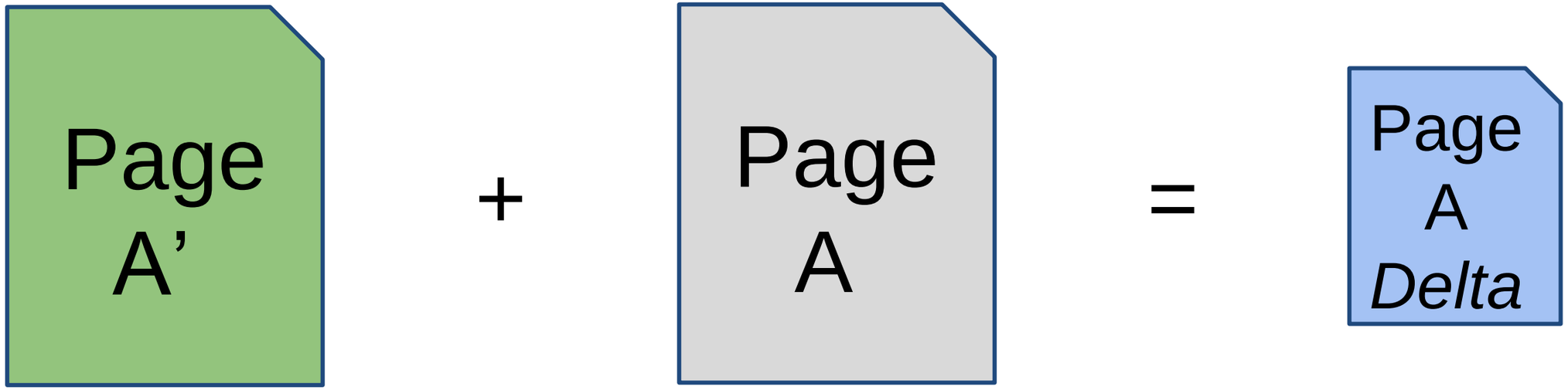
Reference Flash Page

Retained Invalid Flash Page

Compressed Delta with CR of ~0.2

Differences between pages are encoded as *deltas*

Limited Storage Capacity? Compression is Key!



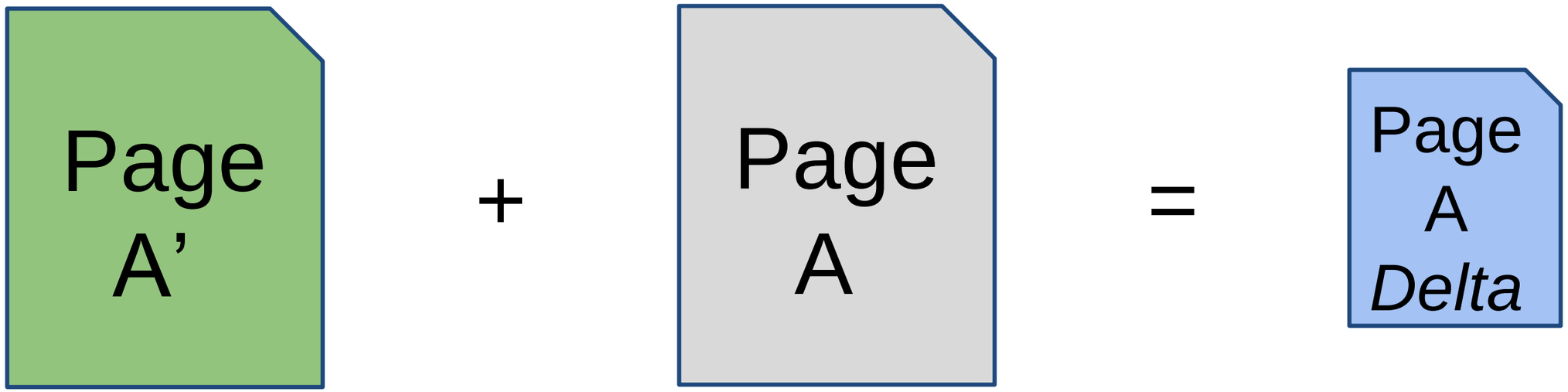
Reference Flash Page

Retained Invalid Flash Page

Compressed Delta with CR of ~0.2

Multiple deltas are coalesced into *delta blocks*

Limited Storage Capacity? Compression is Key!



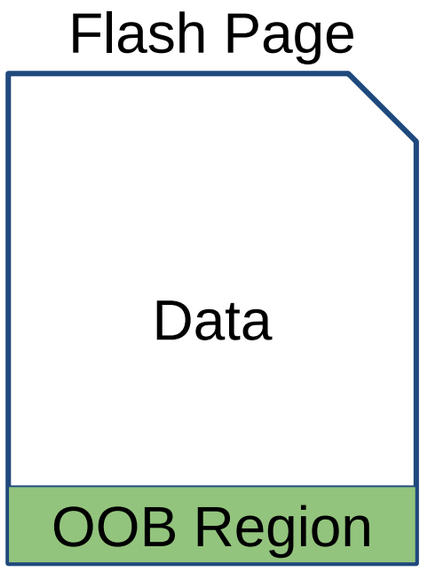
Reference Flash Page

Retained Invalid Flash Page

Compressed Delta with CR of ~0.2

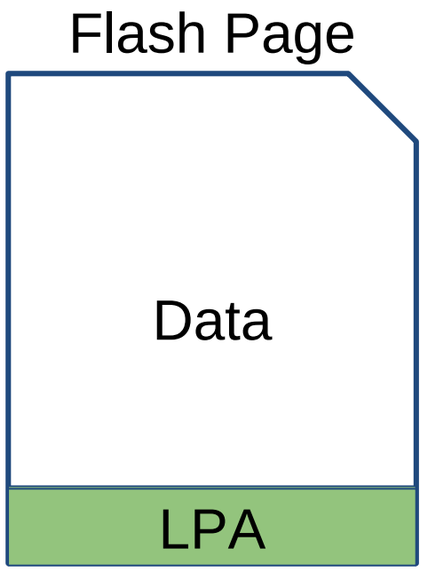
Delta Compression allows for retaining more storage state!

How to Achieve the Time-Traveling Property?



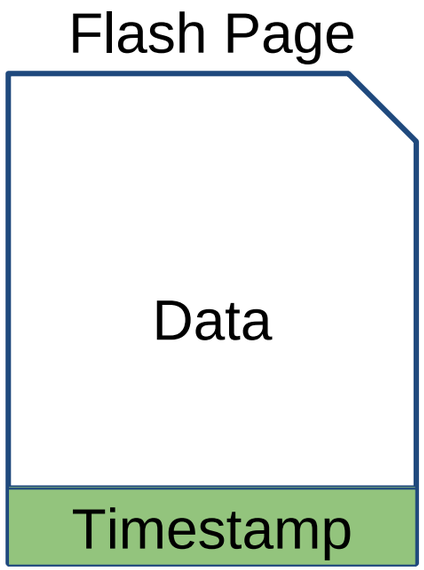
Utilize Out-of-Band Region

How to Achieve the Time-Traveling Property?



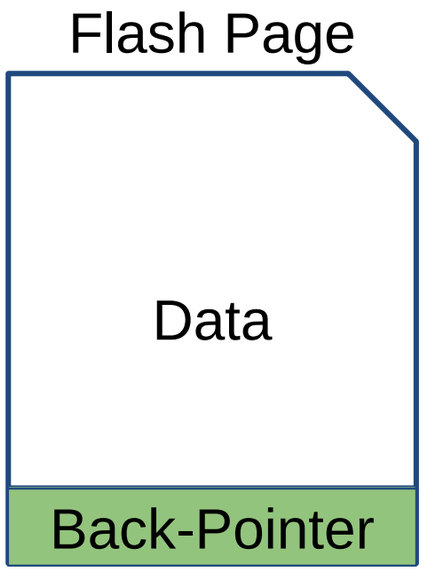
Utilize Out-of-Band Region

How to Achieve the Time-Traveling Property?



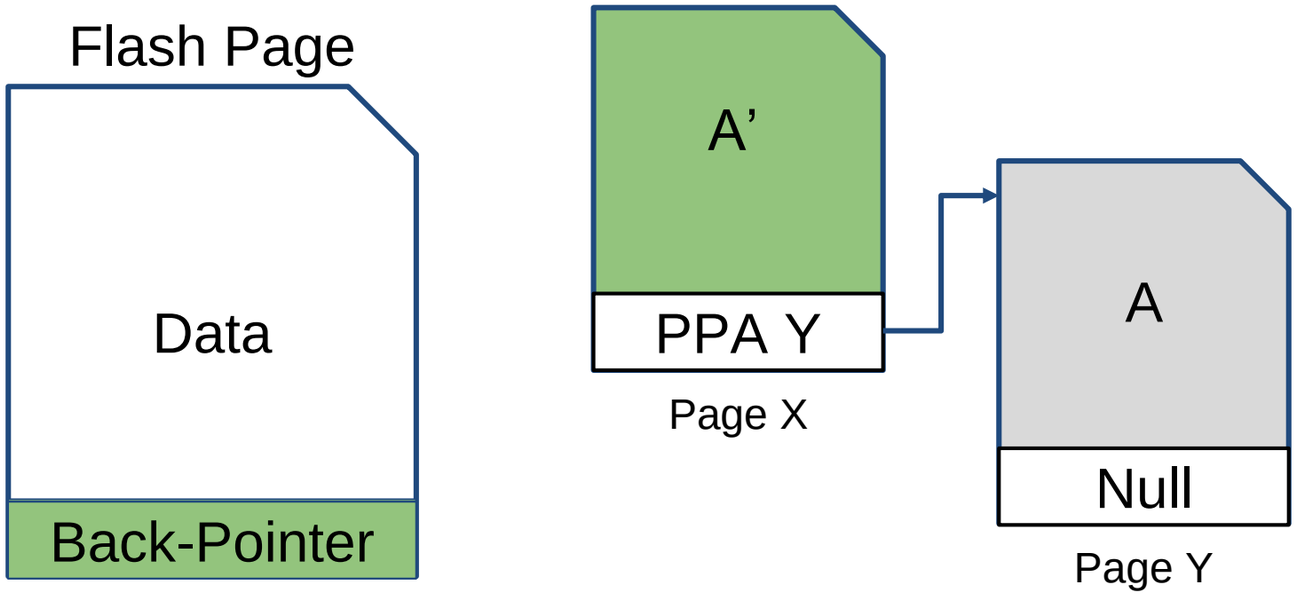
Utilize Out-of-Band Region

How to Achieve the Time-Traveling Property?



Utilize Out-of-Band Region

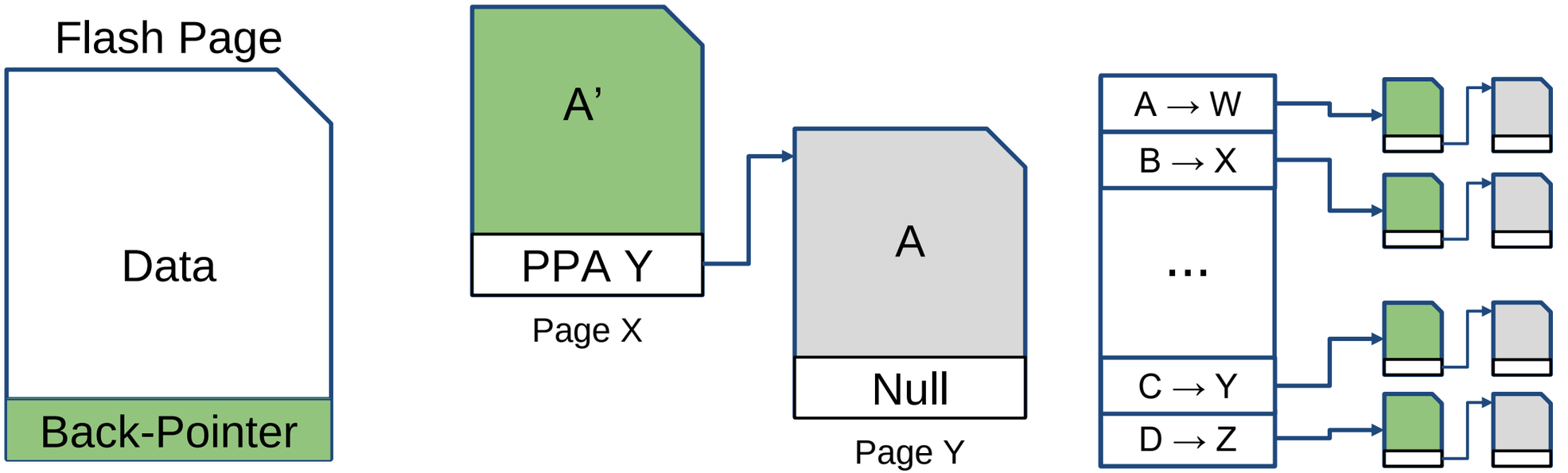
How to Achieve the Time-Traveling Property?



Utilize Out-of-Band Region

Maintain Chains of Back-Pointers

How to Achieve the Time-Traveling Property?

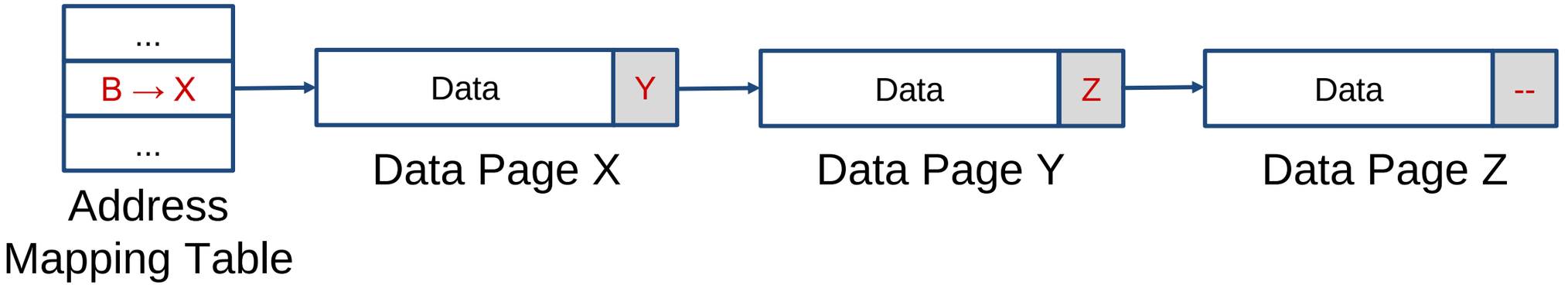


Utilize Out-of-Band Region

Maintain Chains of Back-Pointers

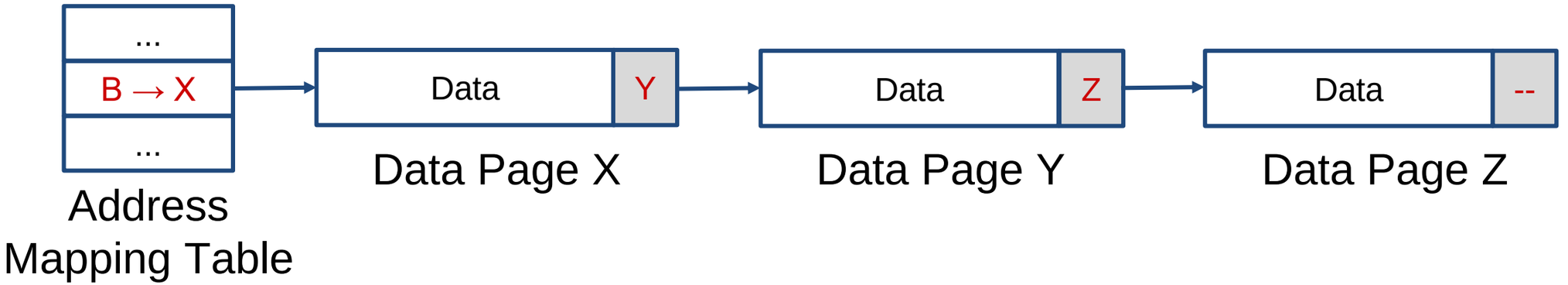
Quickly Traverse with Mapping Tables

Maintaining a Complete Chain



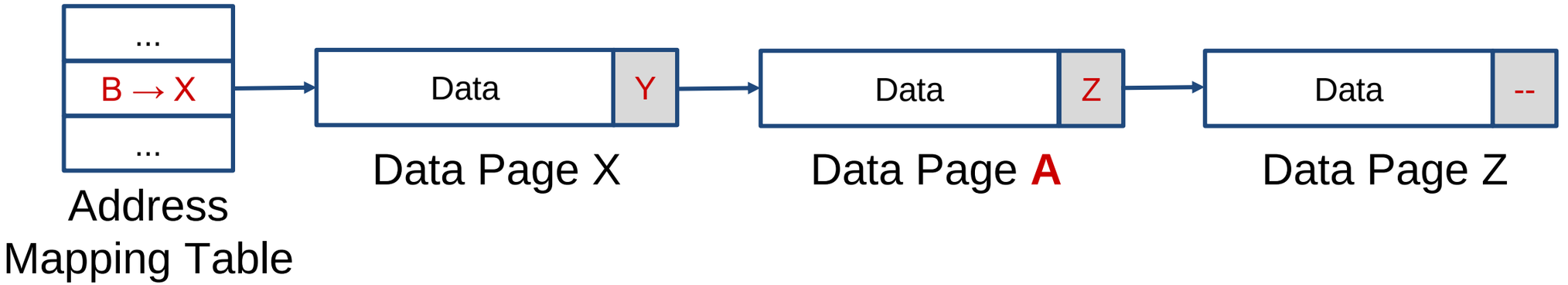
Data page chains are vulnerable to Garbage Collection

Maintaining a Complete Chain



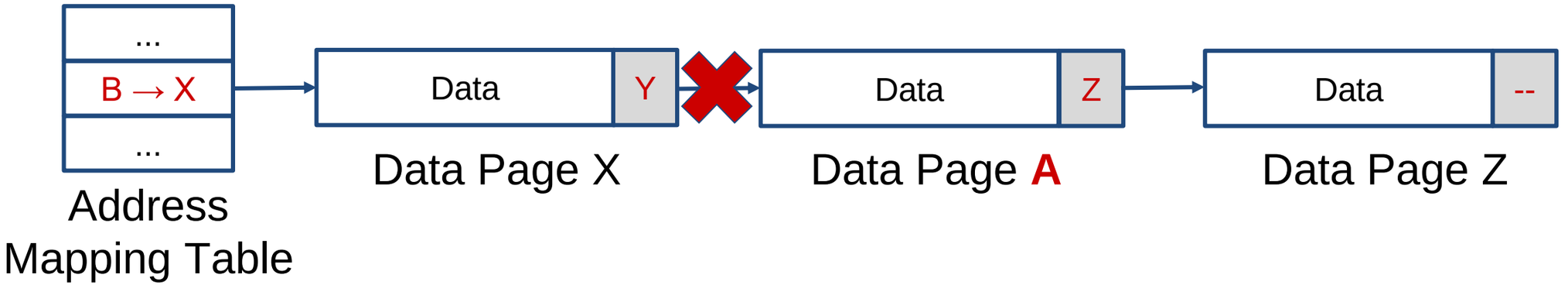
Data page Y is selected for garbage collection!

Maintaining a Complete Chain



Data page Y is selected for garbage collection!

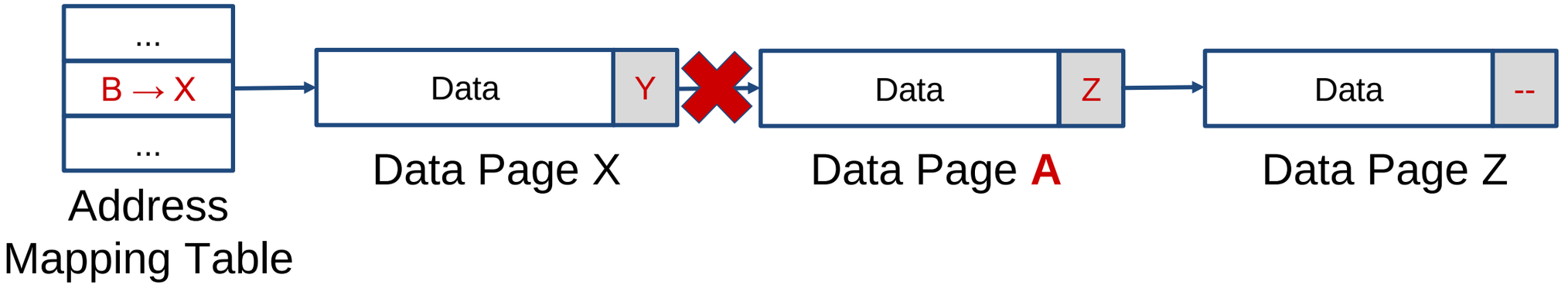
Maintaining a Complete Chain



Data page A and Z will be lost!

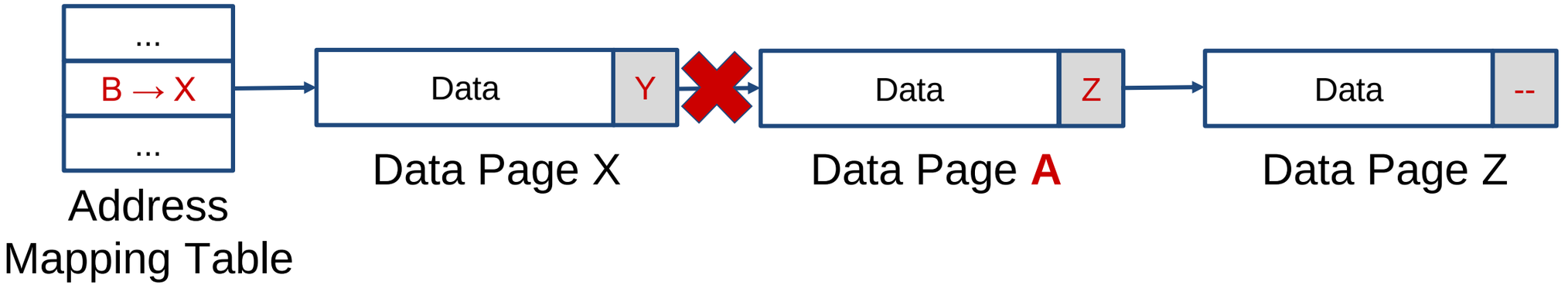
Data page Y is selected for garbage collection!

Maintaining a Complete Chain



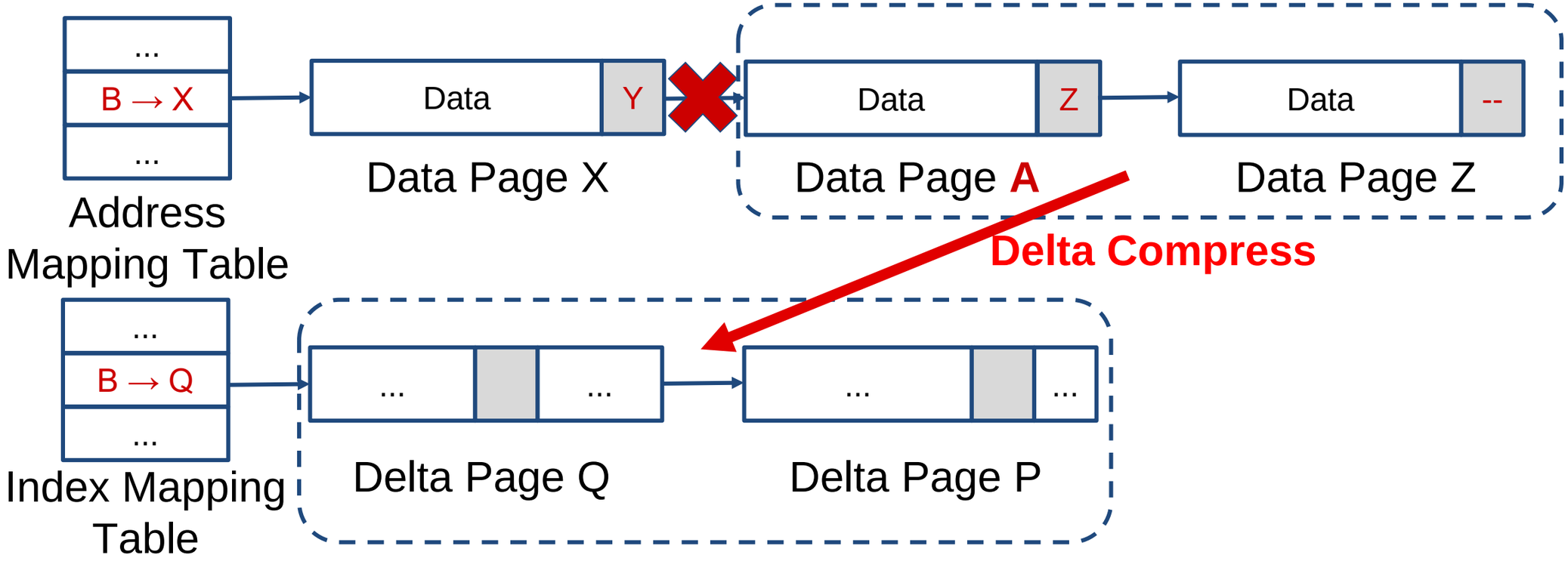
Delta page chains are NOT vulnerable to garbage collection!

Maintaining a Complete Chain



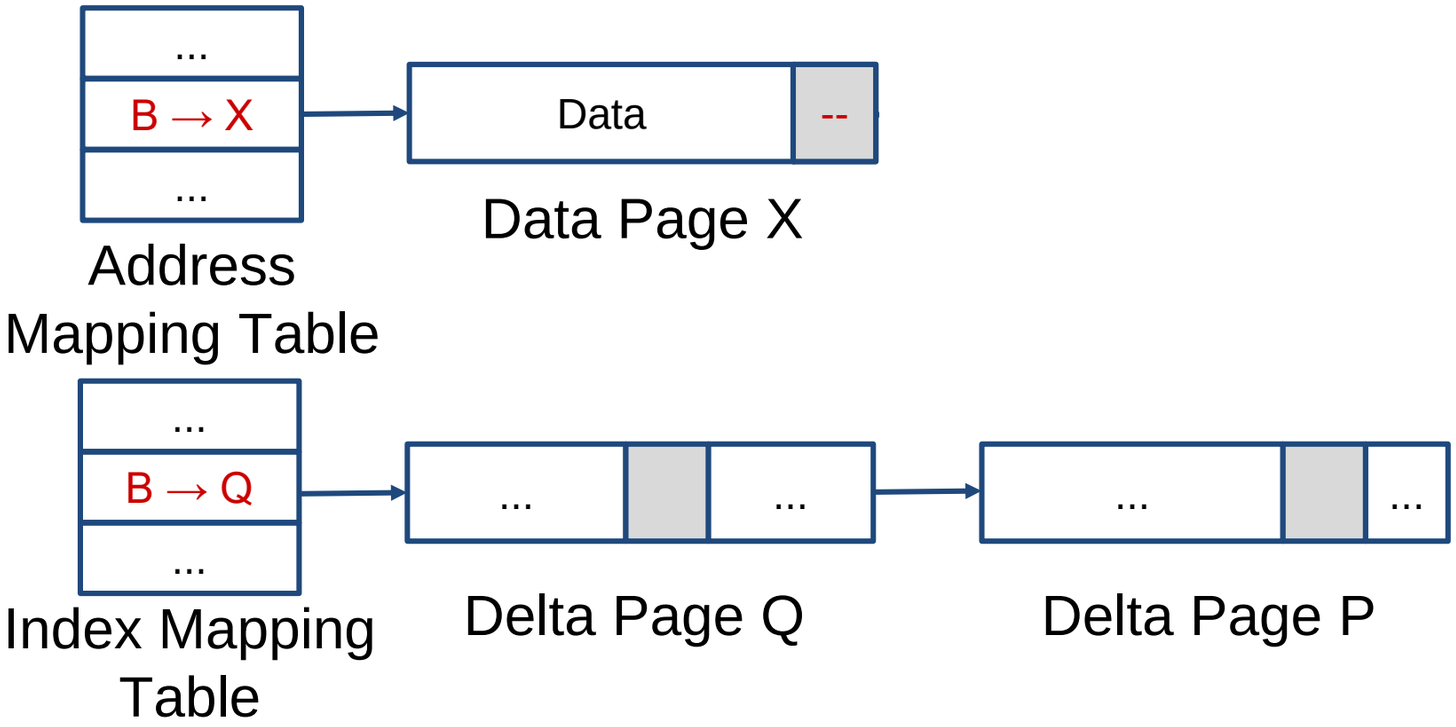
Page Y and Z are compressed to create a delta chain

Maintaining a Complete Chain



Page Y and Z are compressed to create a delta chain

Maintaining a Complete Chain

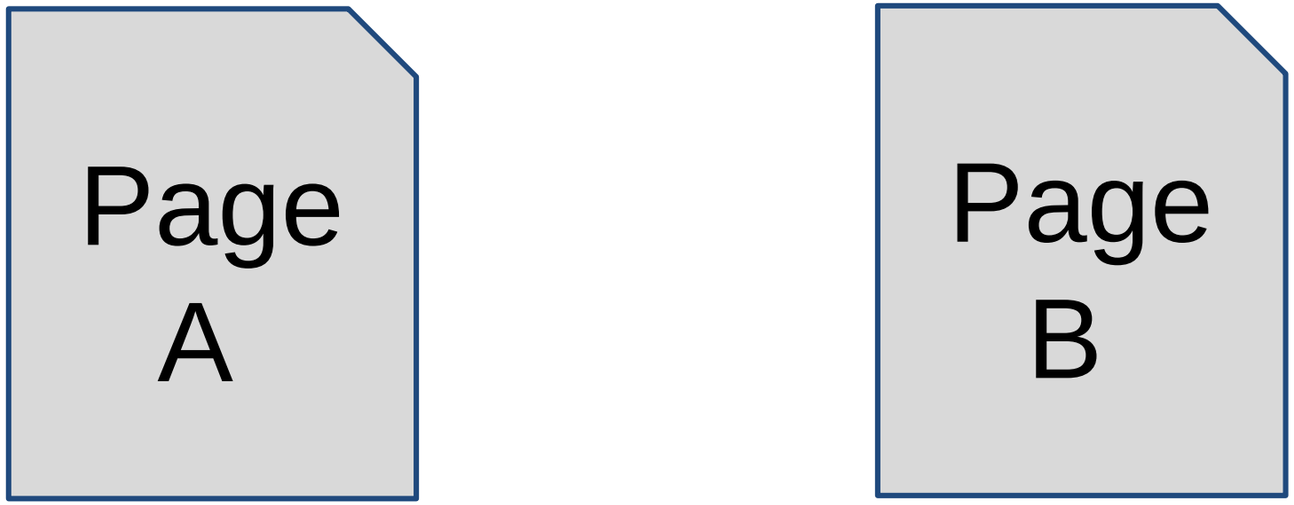


All storage states can be retained with *data* and *delta* chains!

Reclaiming Garbage Data in Time Order

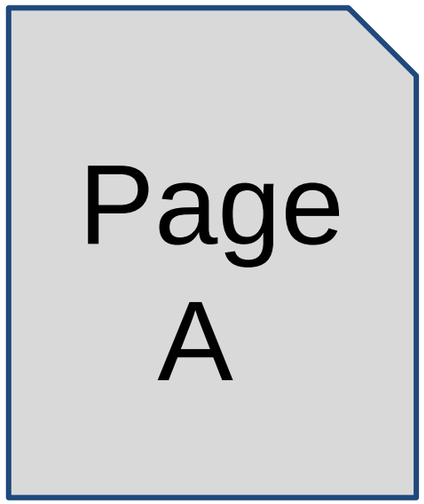
Garbage collection must quickly find the state of invalid pages

Reclaiming Garbage Data in Time Order

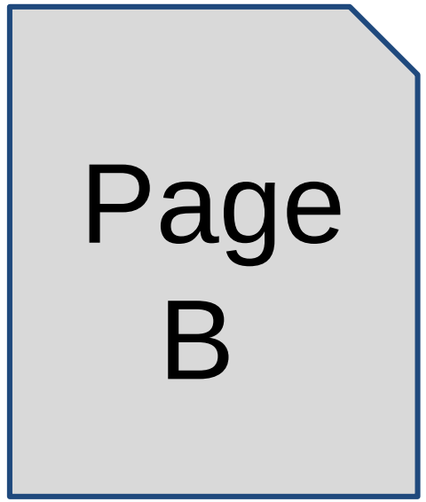


Garbage collection must quickly find the state of invalid pages

Reclaiming Garbage Data in Time Order

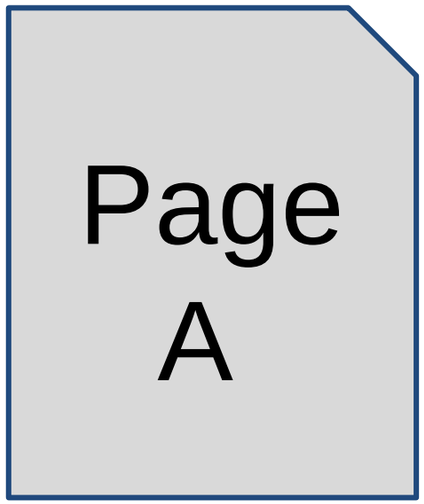


Retained Invalid Page → Keep

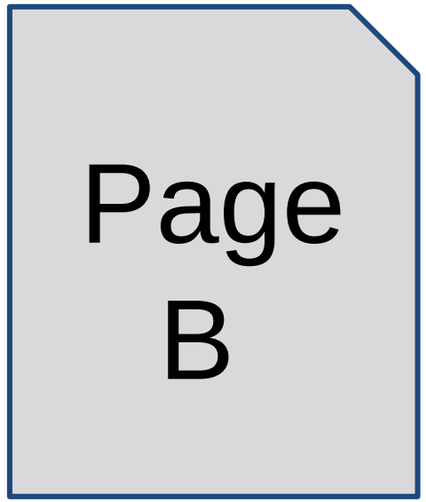


Garbage collection must quickly find the state of invalid pages

Reclaiming Garbage Data in Time Order



Retained Invalid Page → Keep



Expired Page → Reclaimable

Garbage collection must quickly find the state of invalid pages

Reclaiming Garbage Data in Time Order

Physical Page Address	Invalidation Time
0	T_0
1	T_{11}
...	
N-2	T_2
N-1	T_5

A table could be used to track page invalidation timestamps

Reclaiming Garbage Data in Time Order

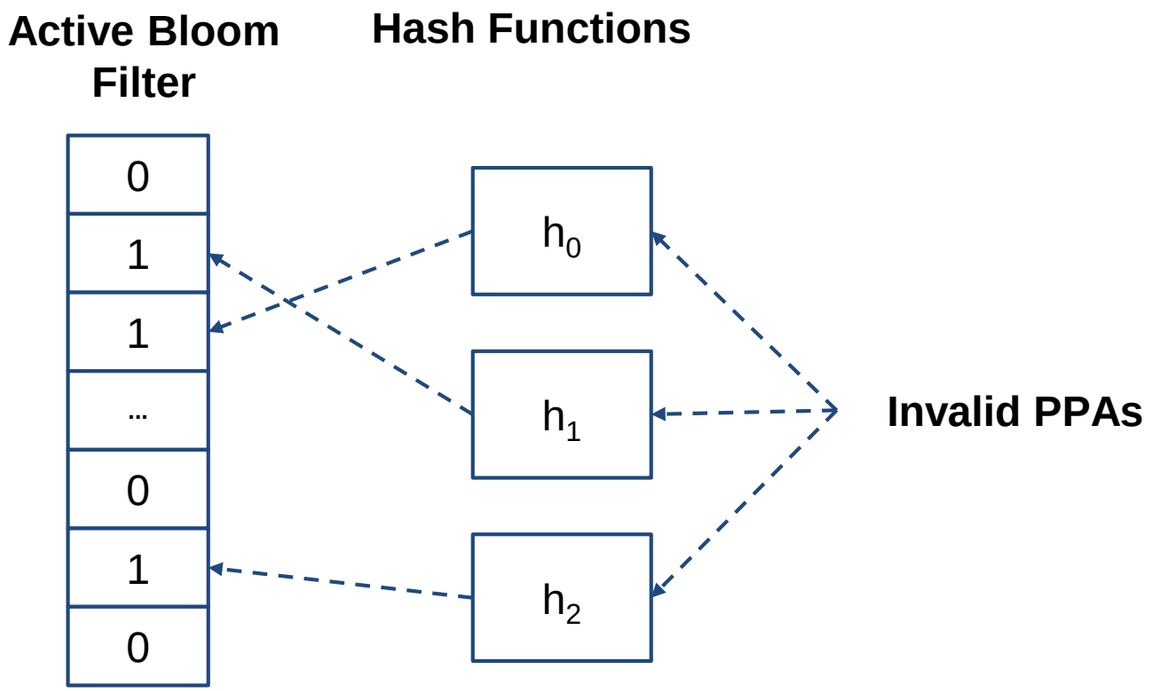
Physical Page Address	Invalidation Time
0	T_0
1	T_{11}
...	
N-2	T_2
N-1	T_5

Even for reasonable SSD sizes, this table is too large to cache

Reclaiming Garbage Data in Time Order

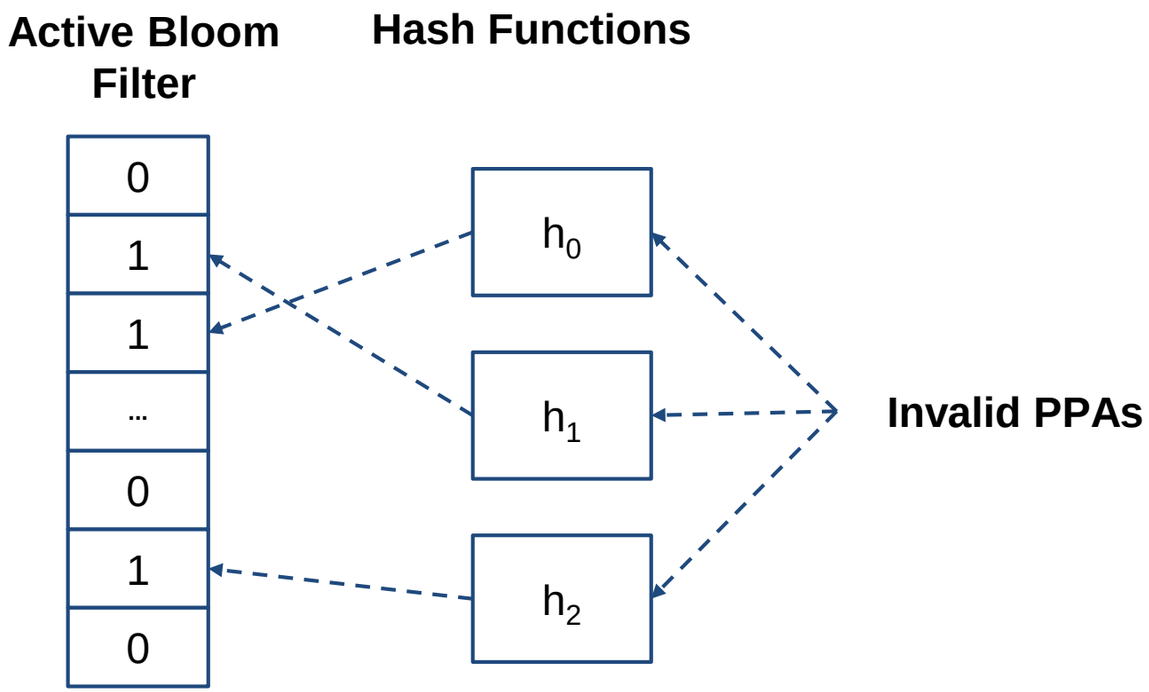
We need a *space-efficient* solution!

Reclaiming Garbage Data in Time Order



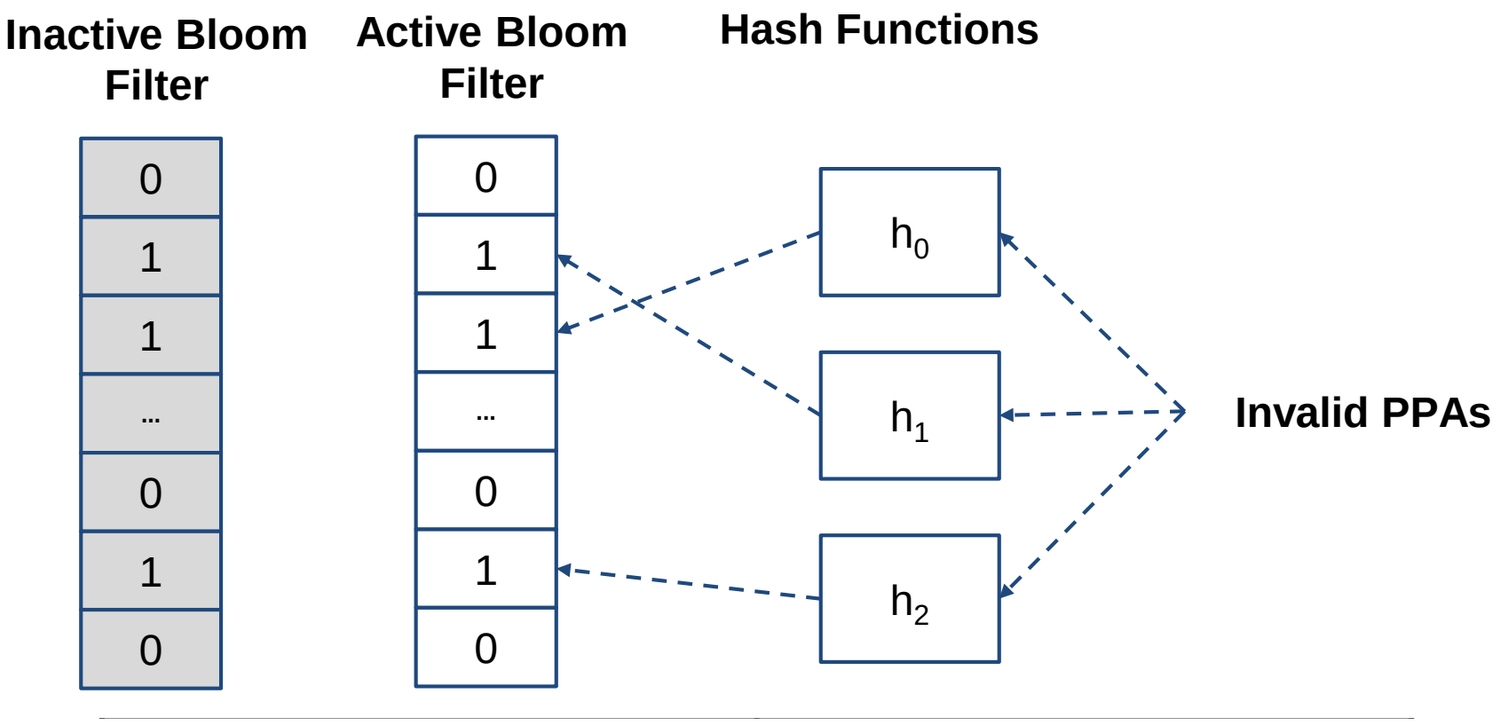
We use *Bloom filters* to track invalid pages efficiently

Reclaiming Garbage Data in Time Order



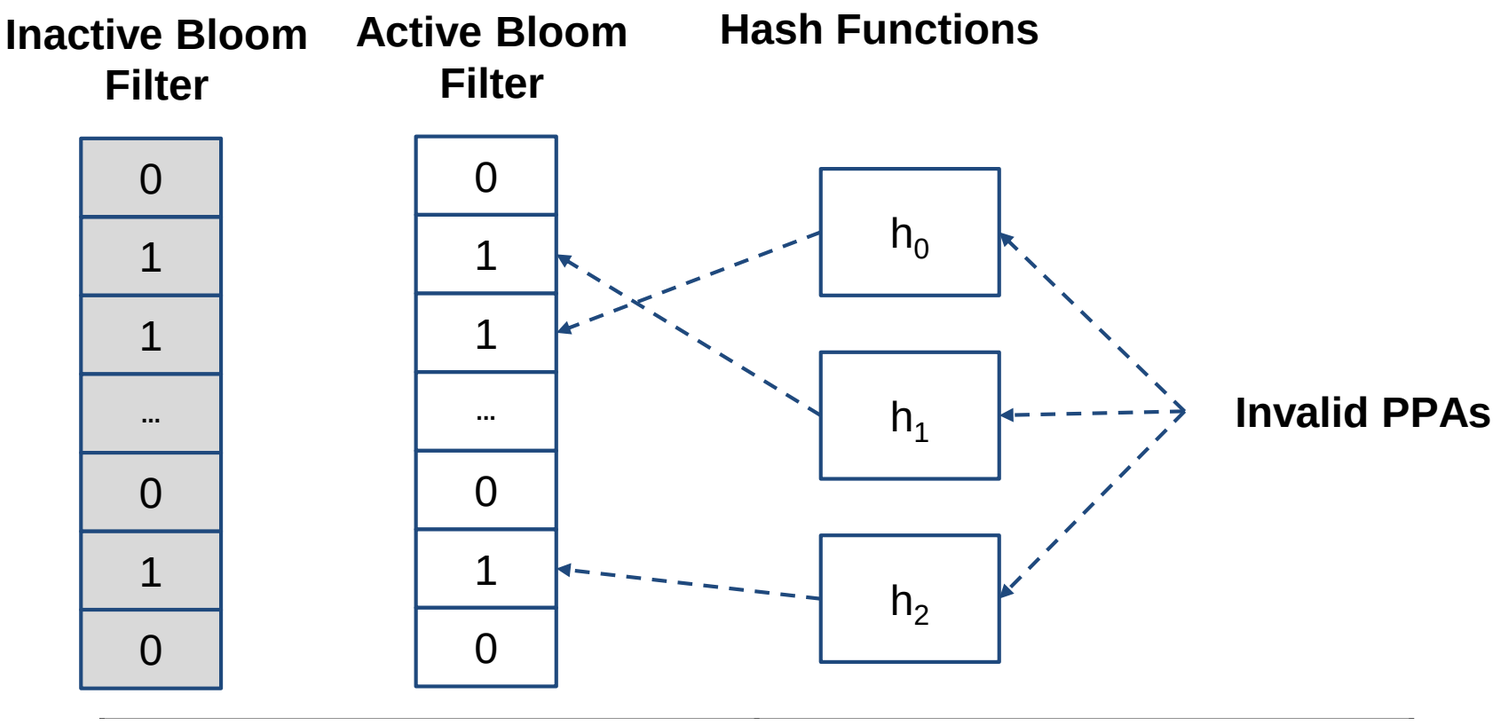
Invalid pages are added to the active Bloom filter

Reclaiming Garbage Data in Time Order



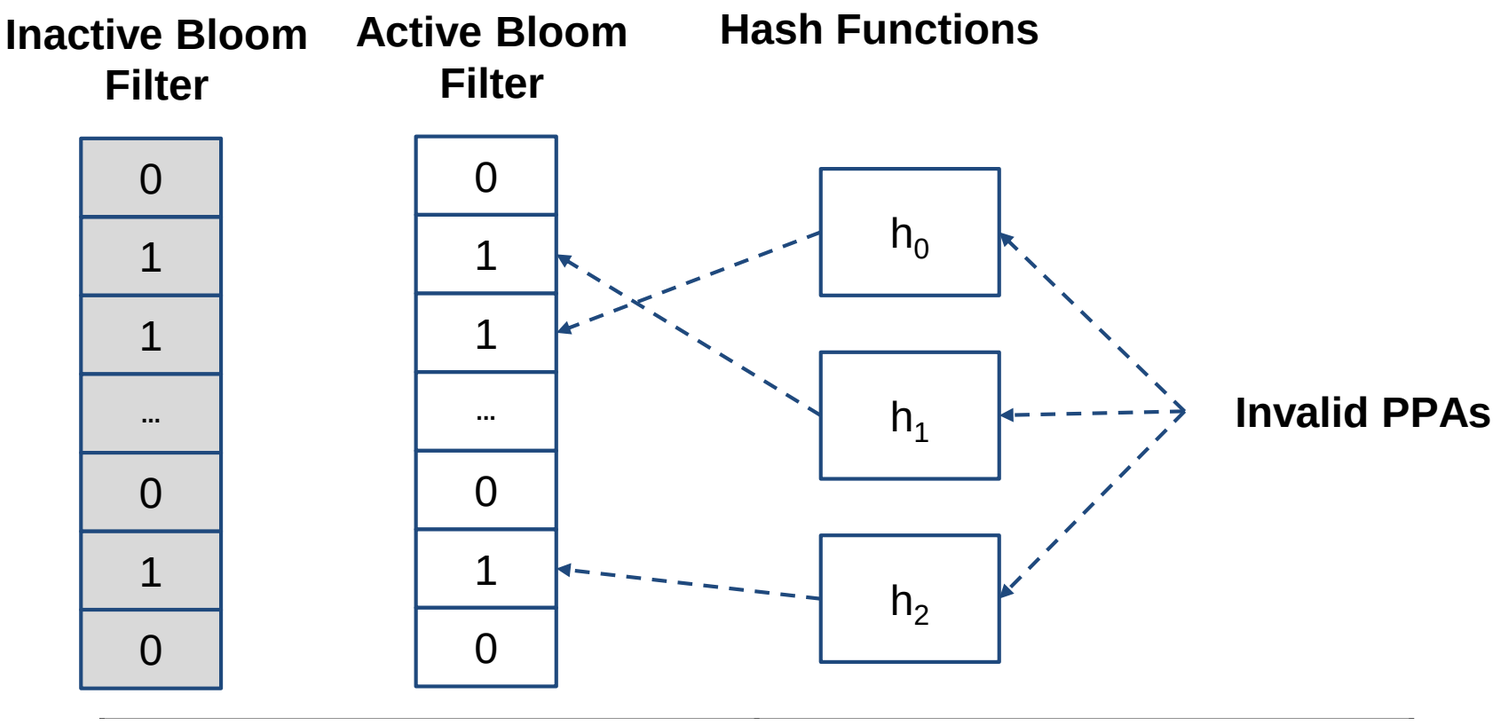
More Bloom filters are created as pages are invalidated

Reclaiming Garbage Data in Time Order



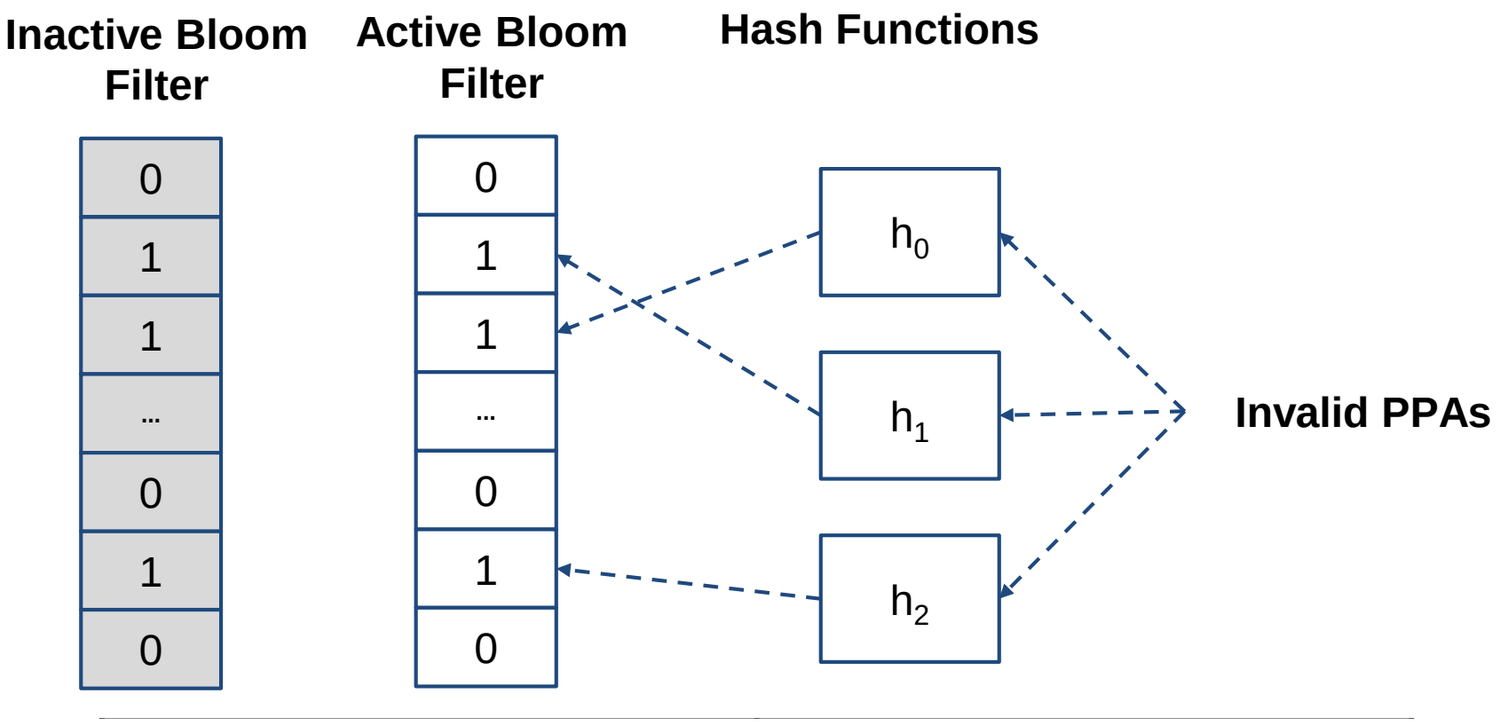
All filters are checked for hits during garbage collection

Reclaiming Garbage Data in Time Order



Any pages which hit in the filters must be retained

Reclaiming Garbage Data in Time Order

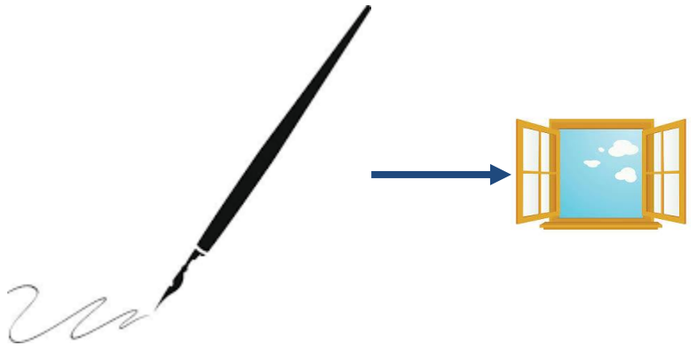


Bloom filters quickly identify expired data and save space!

Workload Variations: Keeping an Adaptive Window

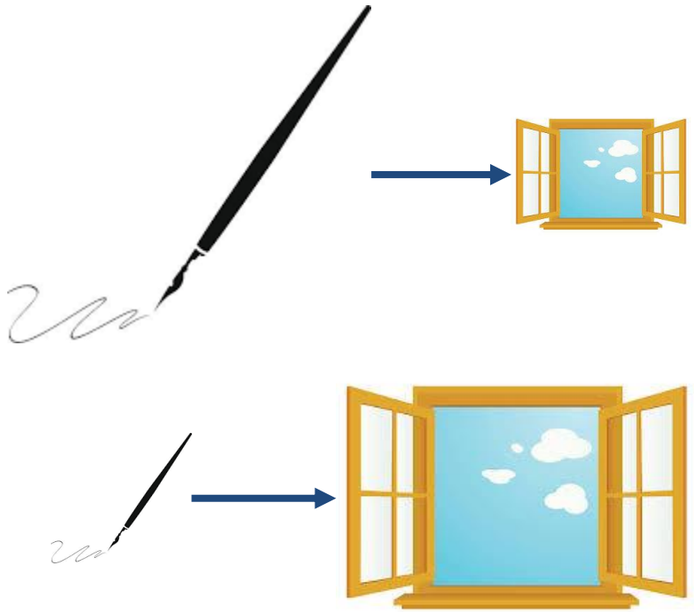
Trade-off Between Performance and Retention Time

Workload Variations: Keeping an Adaptive Window



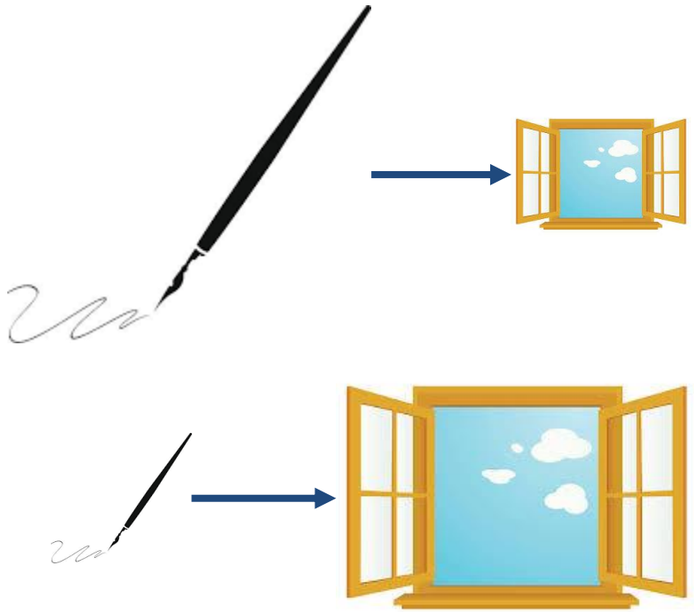
Trade-off Between Performance and Retention Time

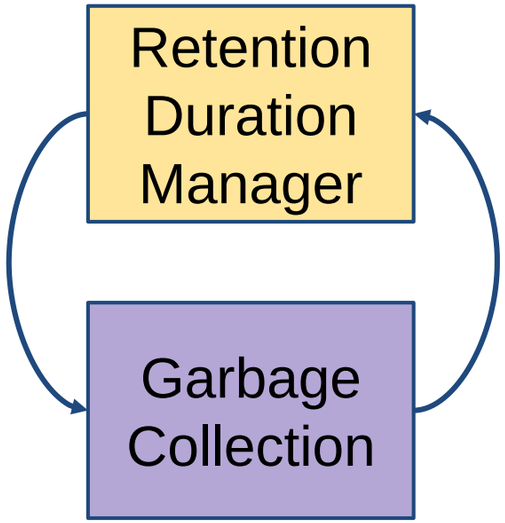
Workload Variations: Keeping an Adaptive Window



Trade-off Between Performance and Retention Time

Workload Variations: Keeping an Adaptive Window

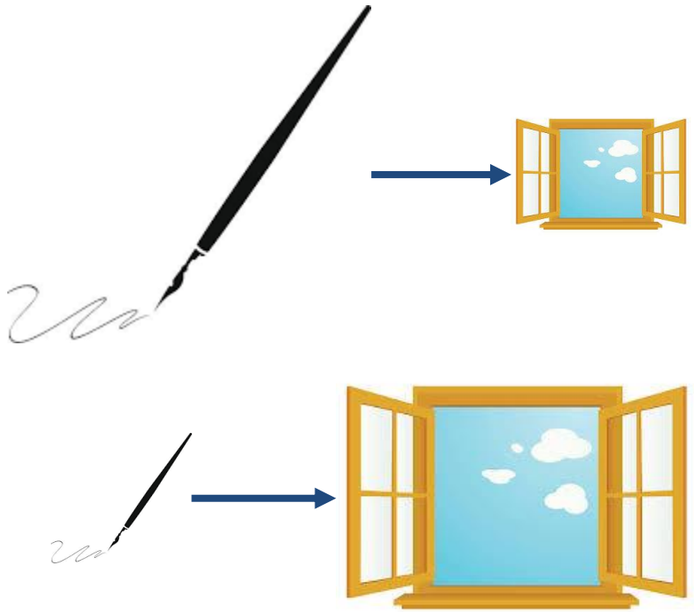


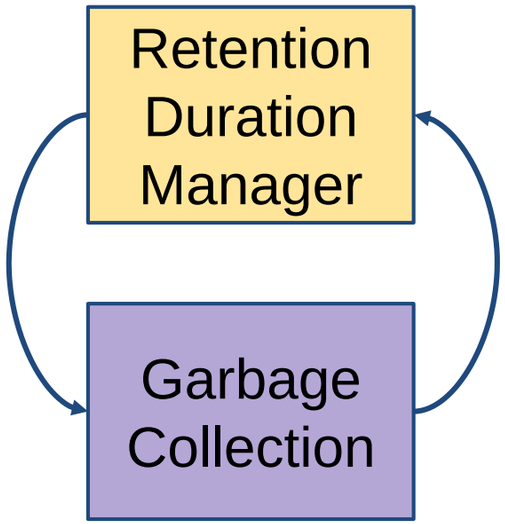


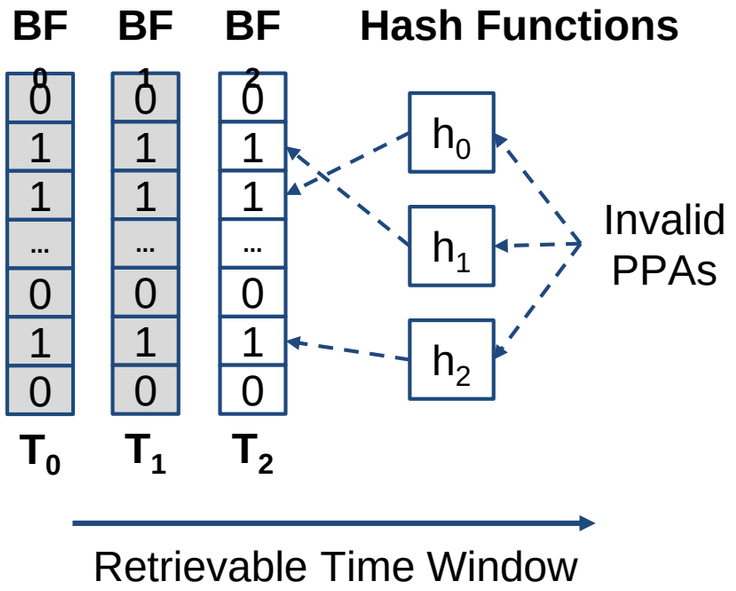
Trade-off Between Performance and Retention Time

Garbage Collection Feedback

Workload Variations: Keeping an Adaptive Window





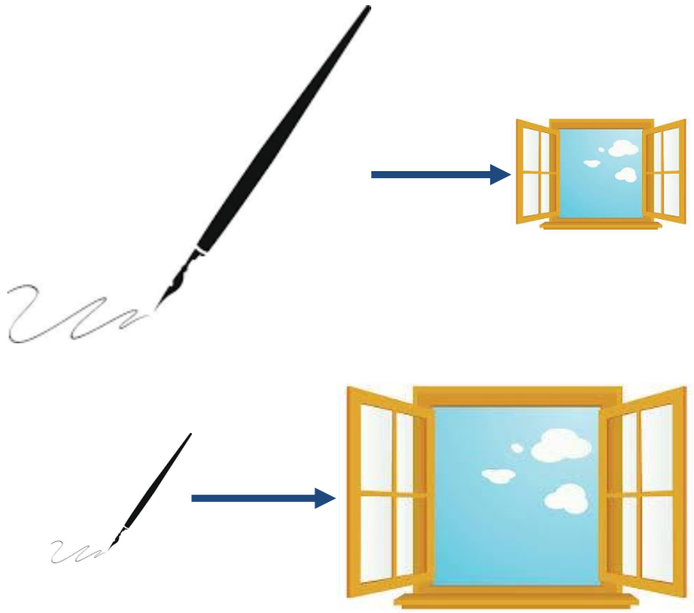


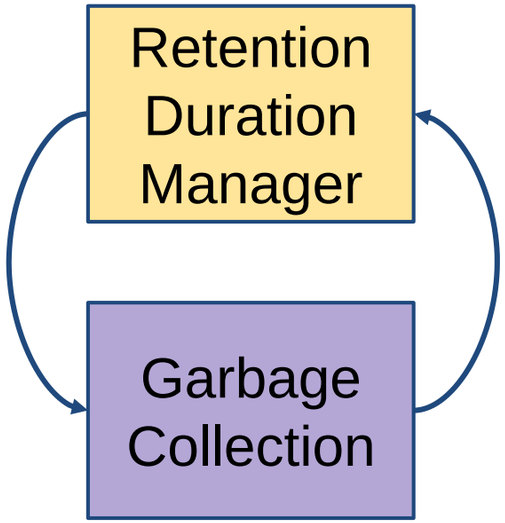
Trade-off Between Performance and Retention Time

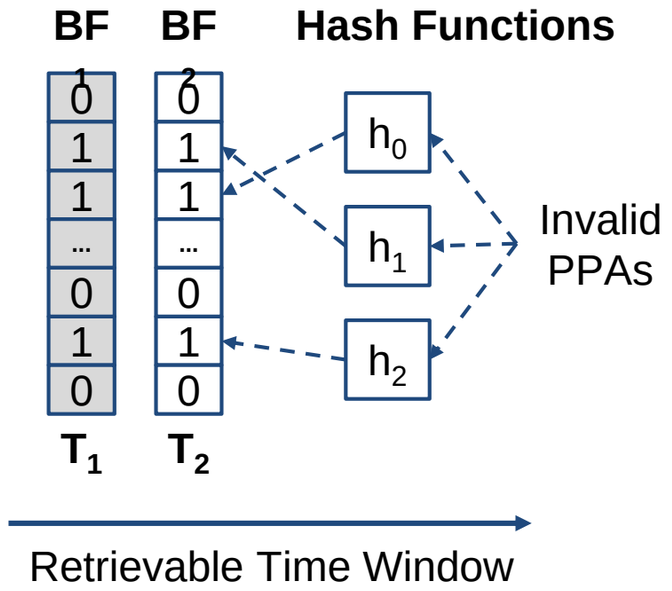
Garbage Collection Feedback

Bloom Filters Hold PPAs in Order

Workload Variations: Keeping an Adaptive Window





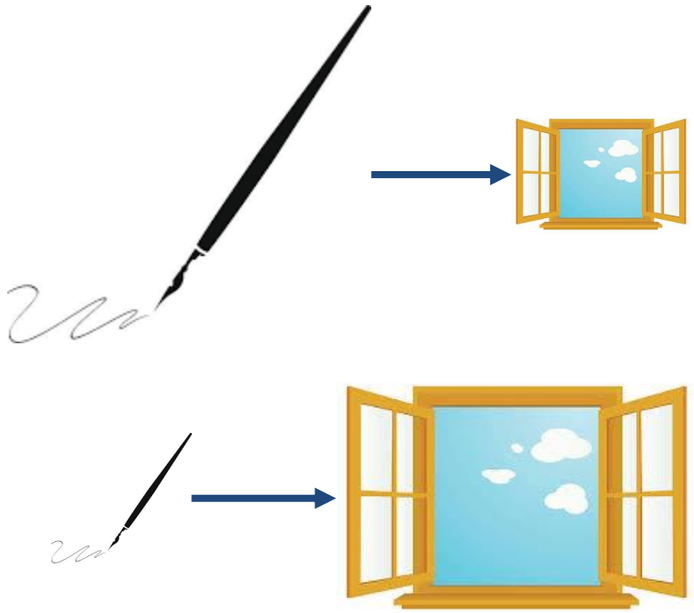


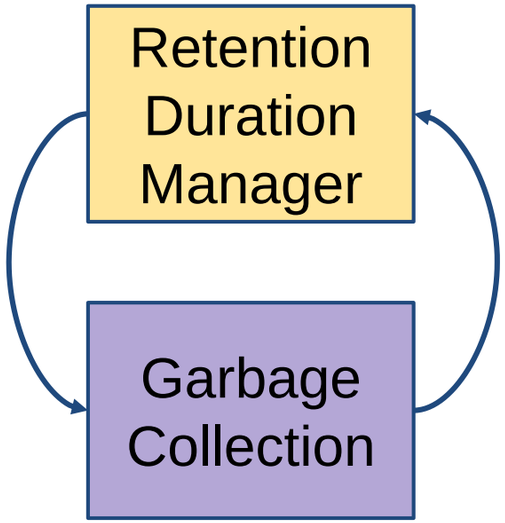
Trade-off Between Performance and Retention Time

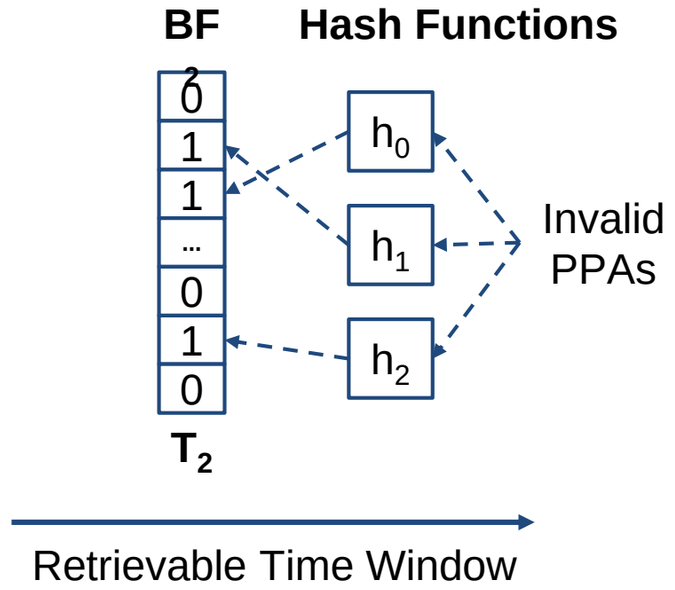
Garbage Collection Feedback

Bloom Filters Hold PPAs in Order

Workload Variations: Keeping an Adaptive Window





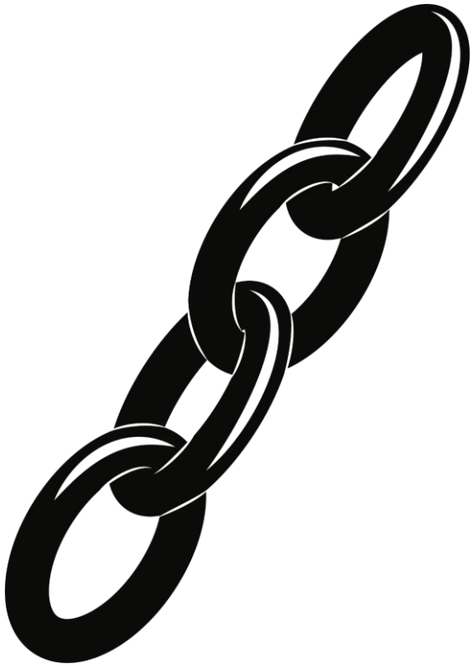


Trade-off Between Performance and Retention Time

Garbage Collection Feedback

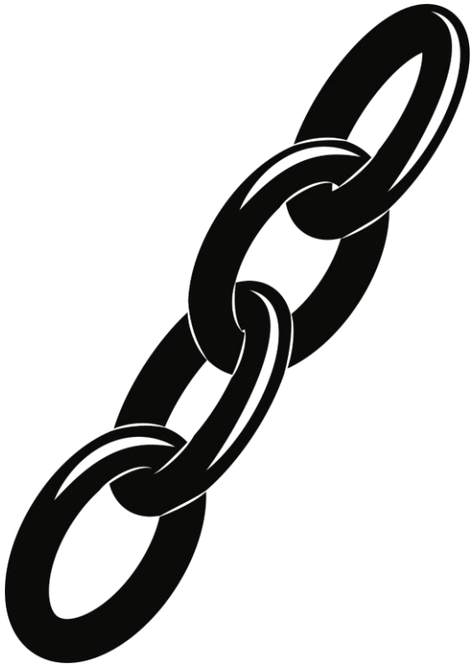
Bloom Filters Hold PPAs in Order

TimeKits: State Query



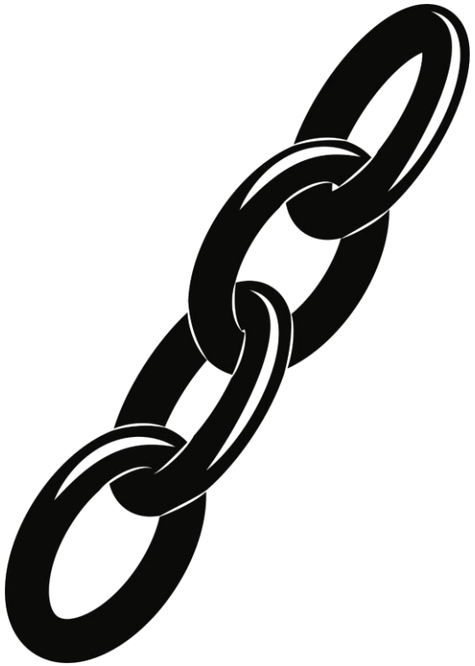
Perform *fast* state queries by leveraging back-pointer chains

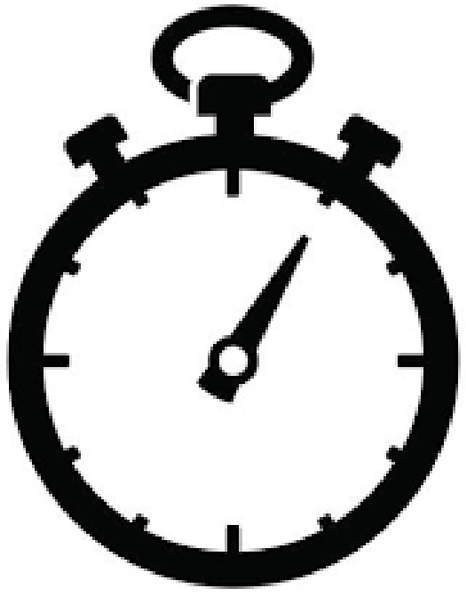
TimeKits: State Query



Per-address state queries retrieve the history of a given LPA

TimeKits: State Query



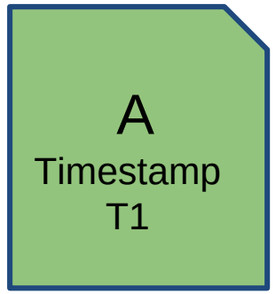


Time-range state queries retrieve all LPA changes in a range of time

TimeKits: State Rollback

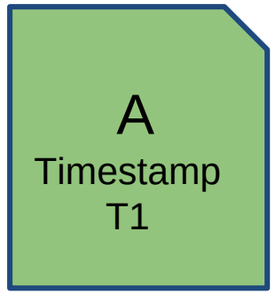
Possible to rollback any address to a previous state

TimeKits: State Rollback



Possible to rollback any address to a previous state

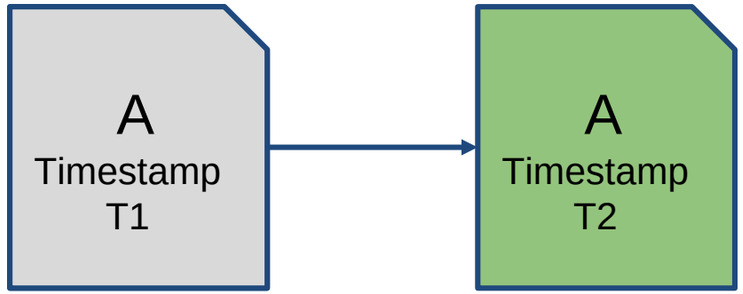
TimeKits: State Rollback



Update to A

Possible to rollback any address to a previous state

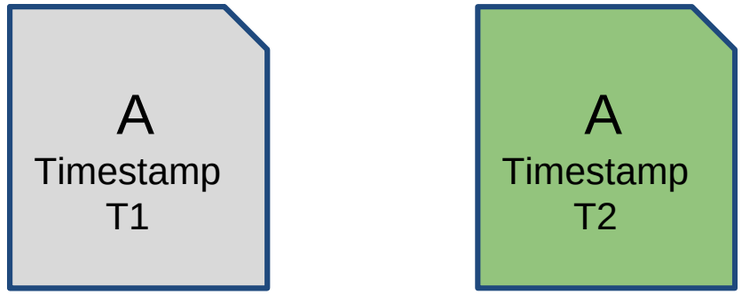
TimeKits: State Rollback



Update to A

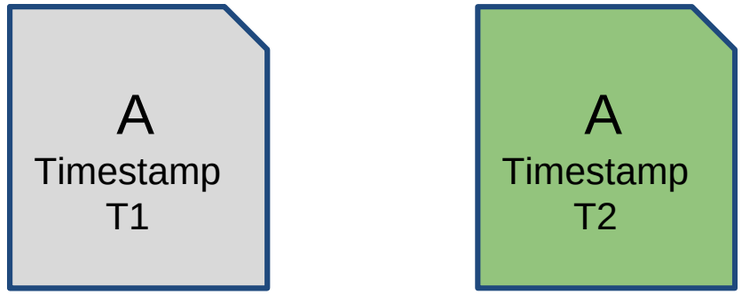
Possible to rollback any address to a previous state

TimeKits: State Rollback



Possible to rollback any address to a previous state

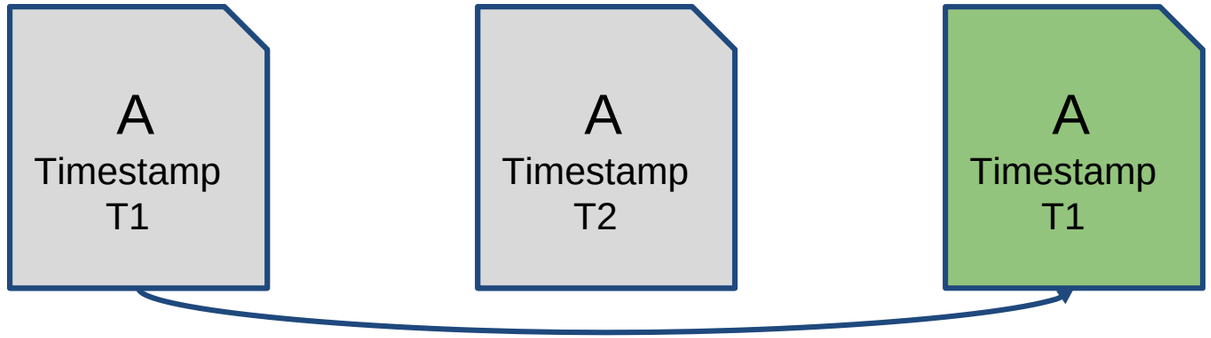
TimeKits: State Rollback



Rollback A to previous timestamp

Possible to rollback any address to a previous state

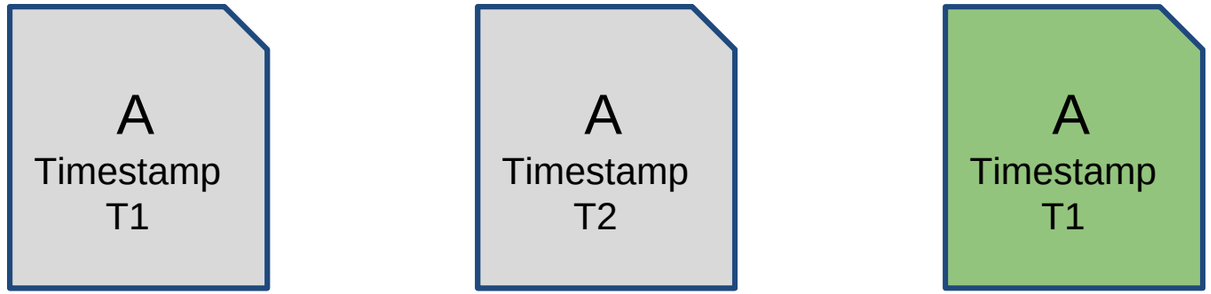
TimeKits: State Rollback



Rollback A to previous timestamp

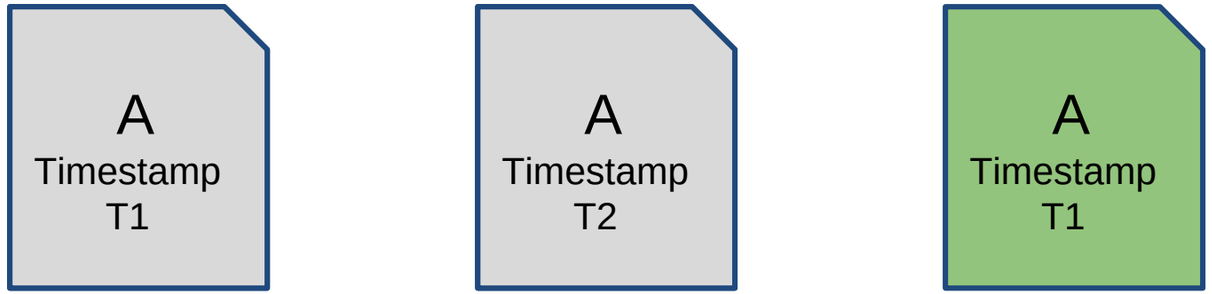
Possible to rollback any address to a previous state

TimeKits: State Rollback



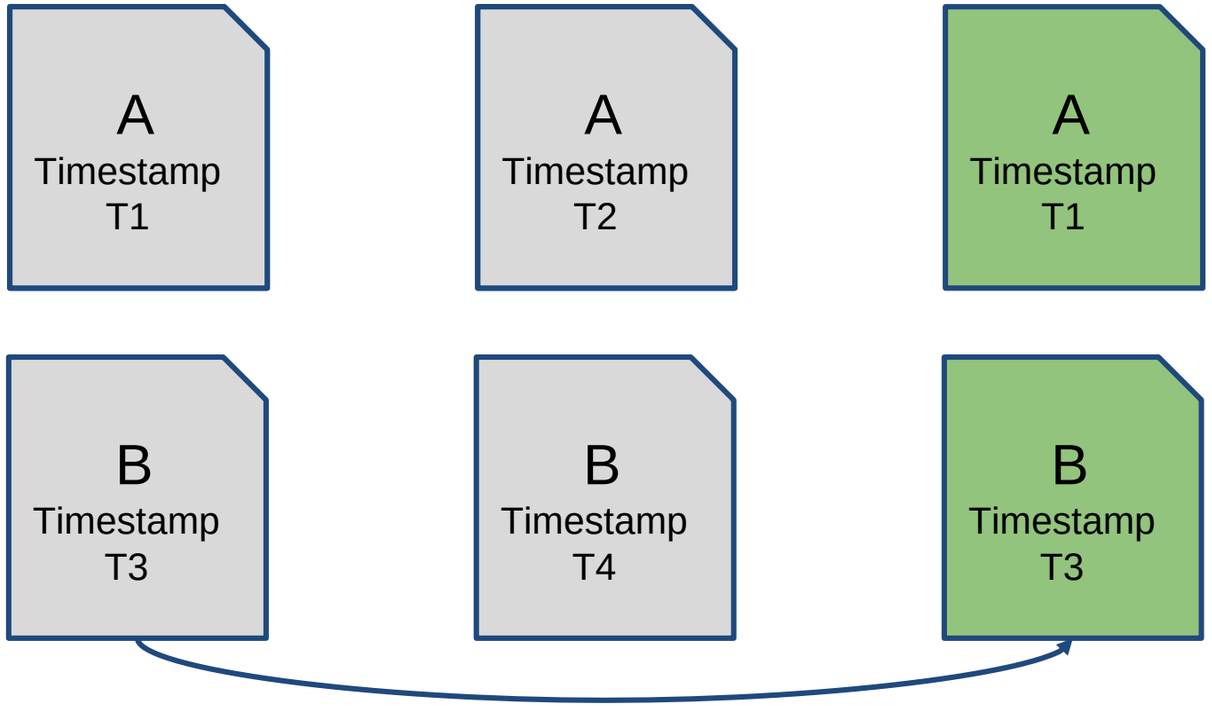
Possible to rollback any address to a previous state

TimeKits: State Rollback



Channel parallelism allows *fast* rollback of multiple addresses

TimeKits: State Rollback



Channel parallelism allows *fast* rollback of multiple addresses

Experiment Setup

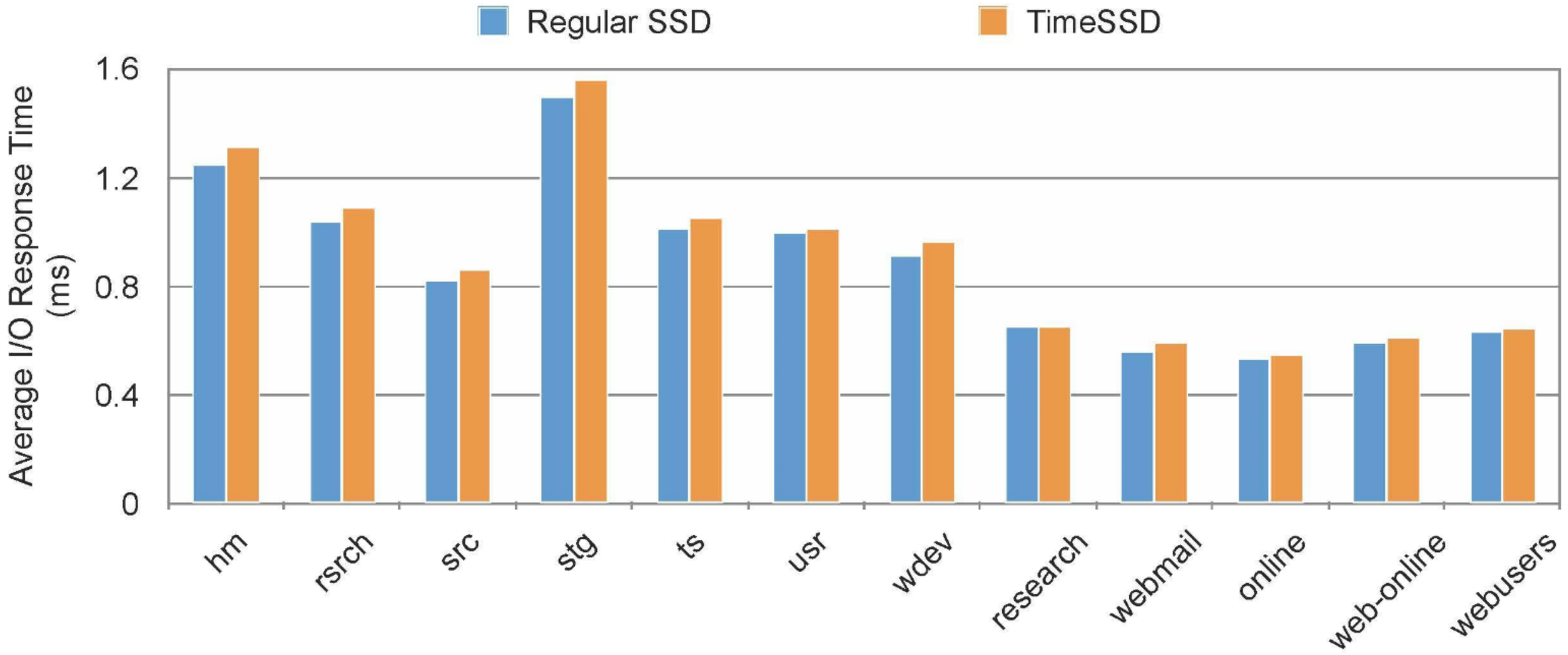
HW Platform

Cosmos+ OpenSSD FPGA
development Board
1TB SSD, 4KB page
with 12B OOB data

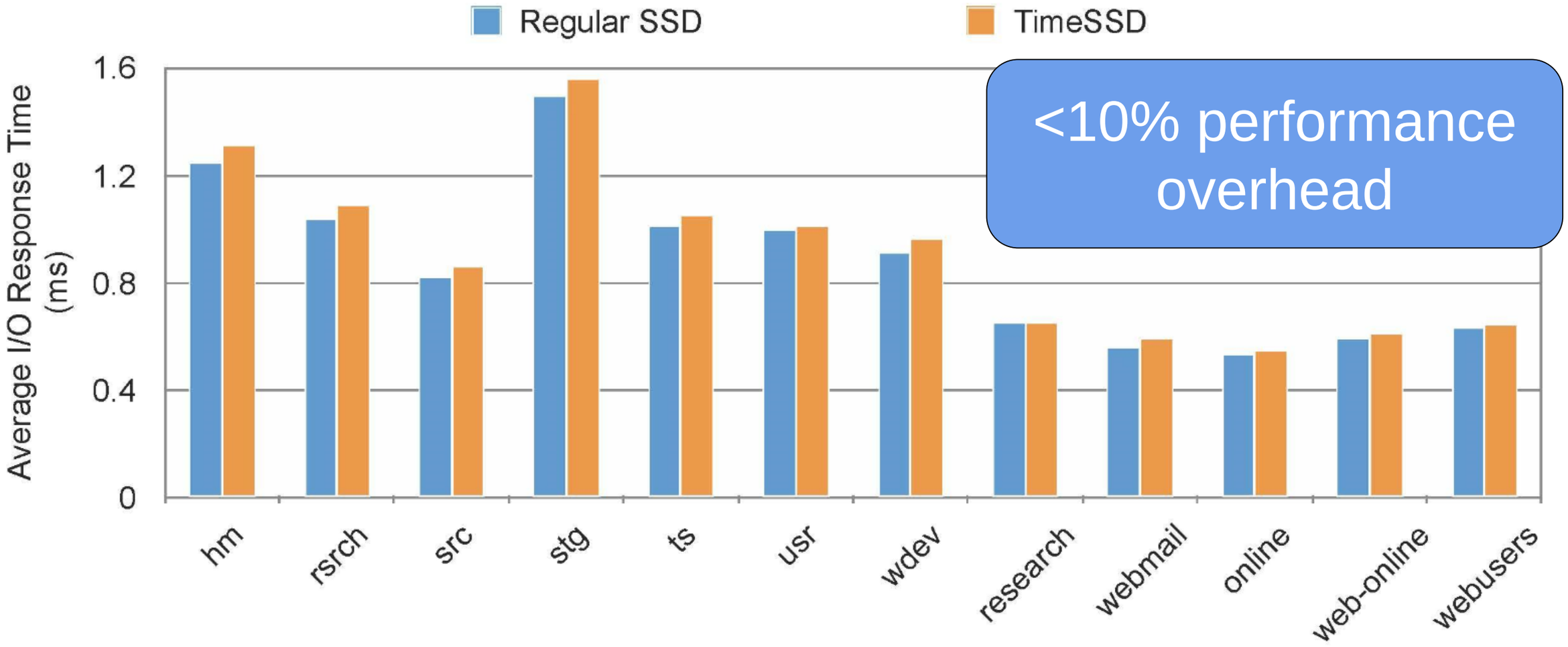
Benchmarks

Storage traces from MSR and FIU
IOZone benchmarks
PostMark benchmark
OLTP database engine
Ransomware malware samples

Performance: TimeSSD vs. Regular SSDs

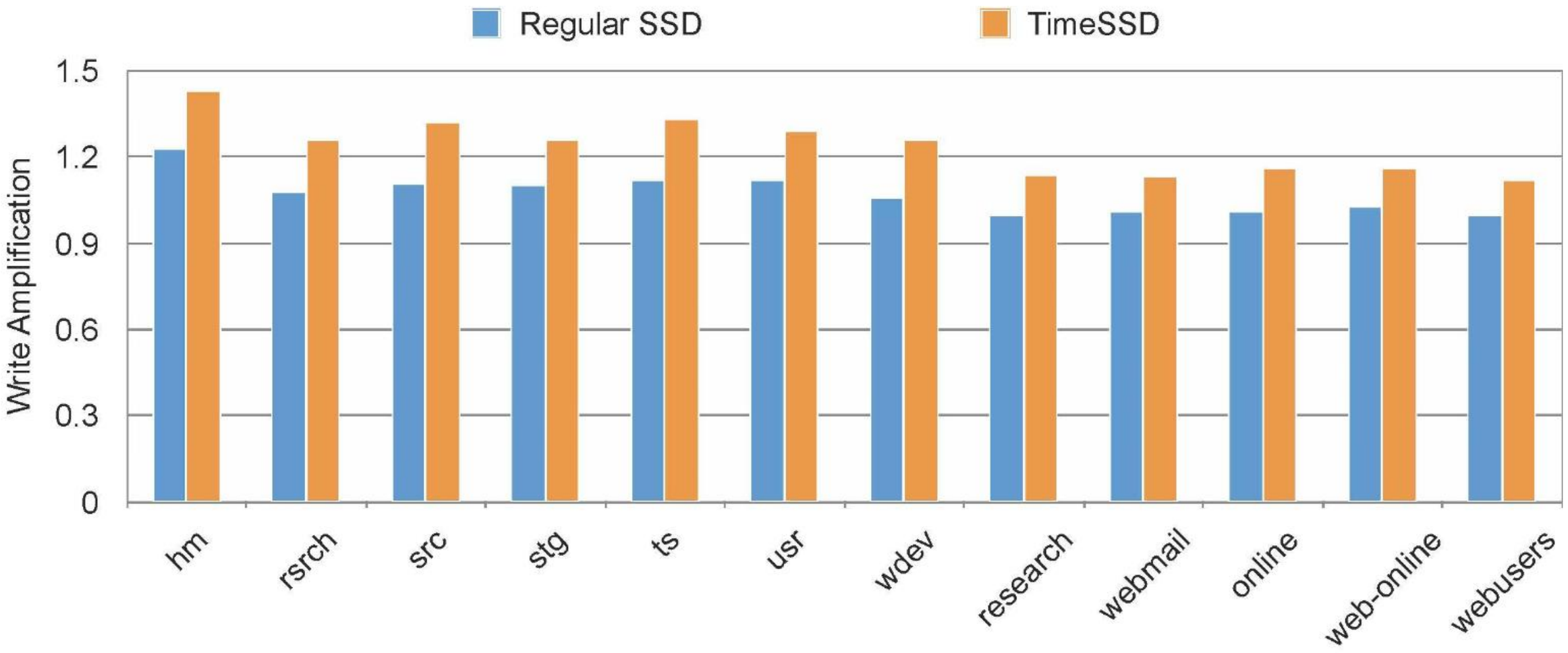


Performance: TimeSSD vs. Regular SSDs

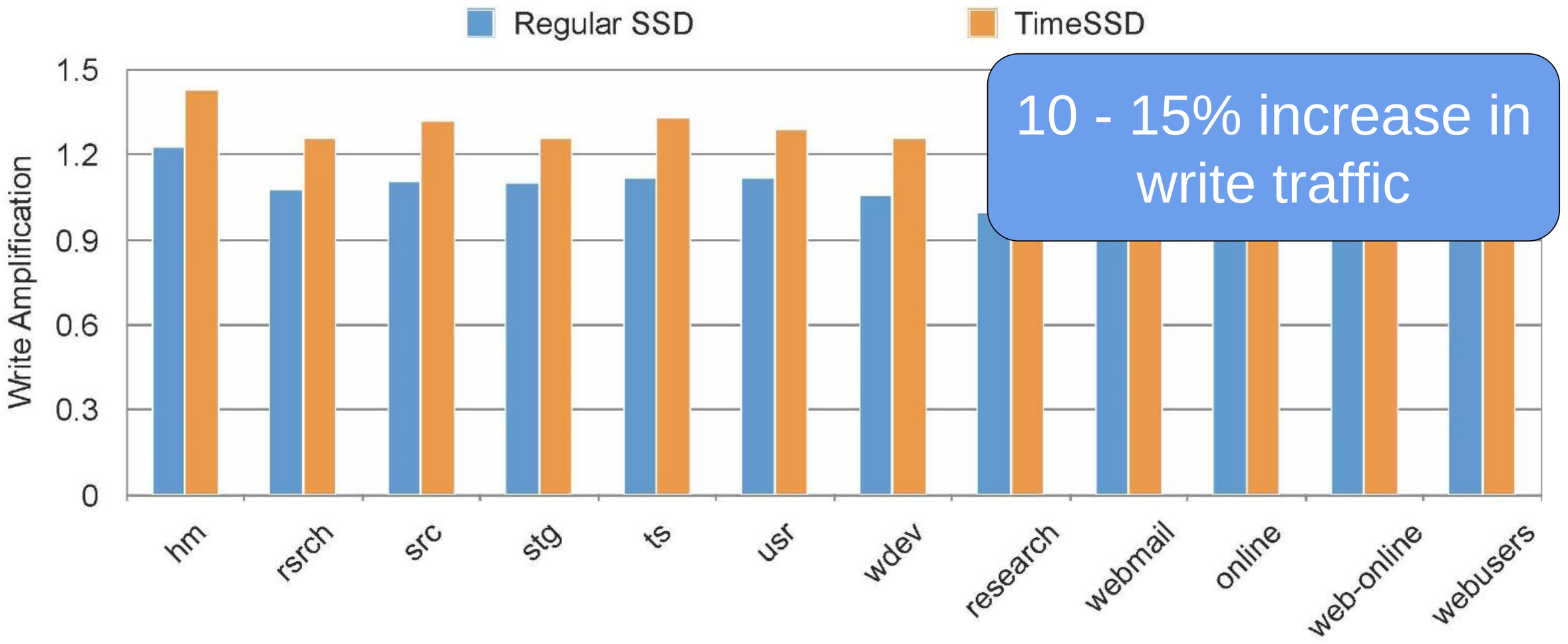


<10% performance overhead

Device Lifetime: TimeSSD vs. Regular SSDs



Device Lifetime: TimeSSD vs. Regular SSDs

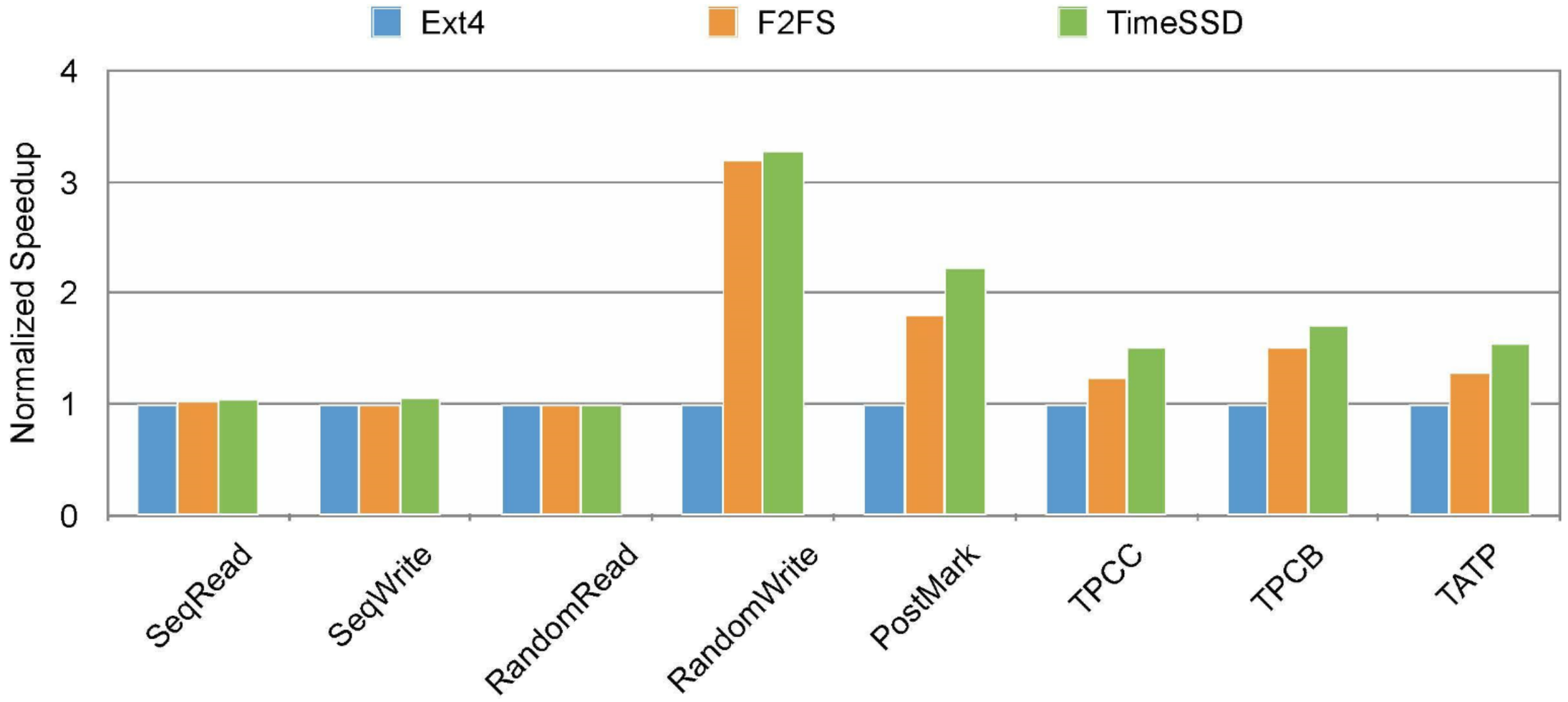


10 - 15% increase in write traffic

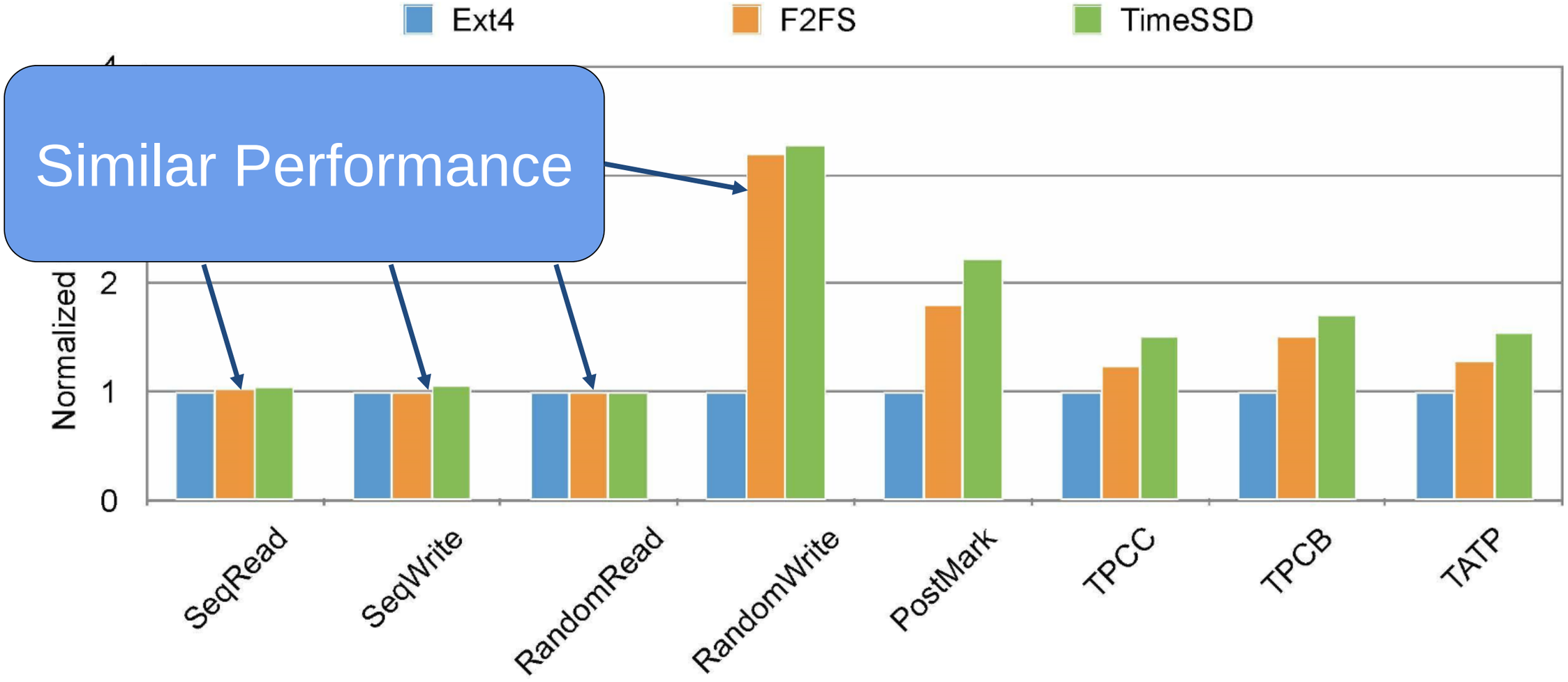
Device Lifetime: TimeSSD vs. Regular SSDs

SW solutions have WA factors of 2.5-6!

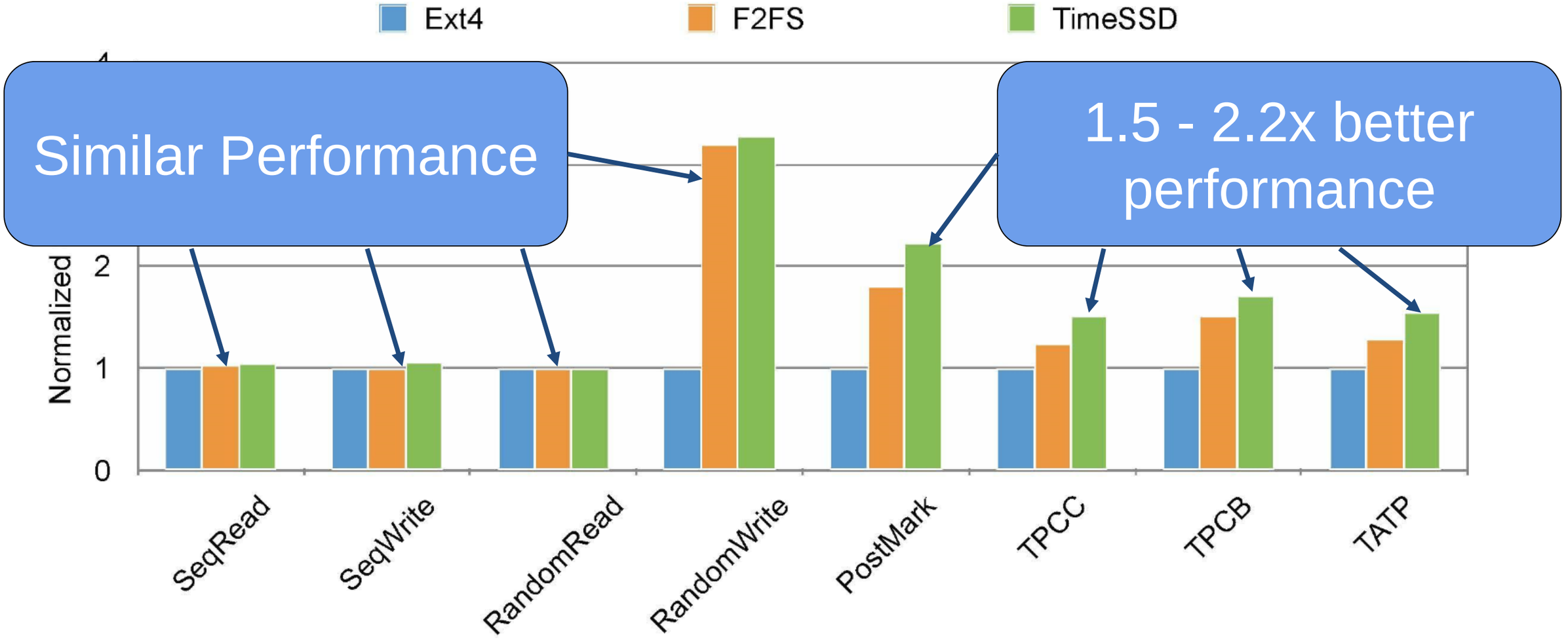
TimeSSD vs. Software-Based Solutions



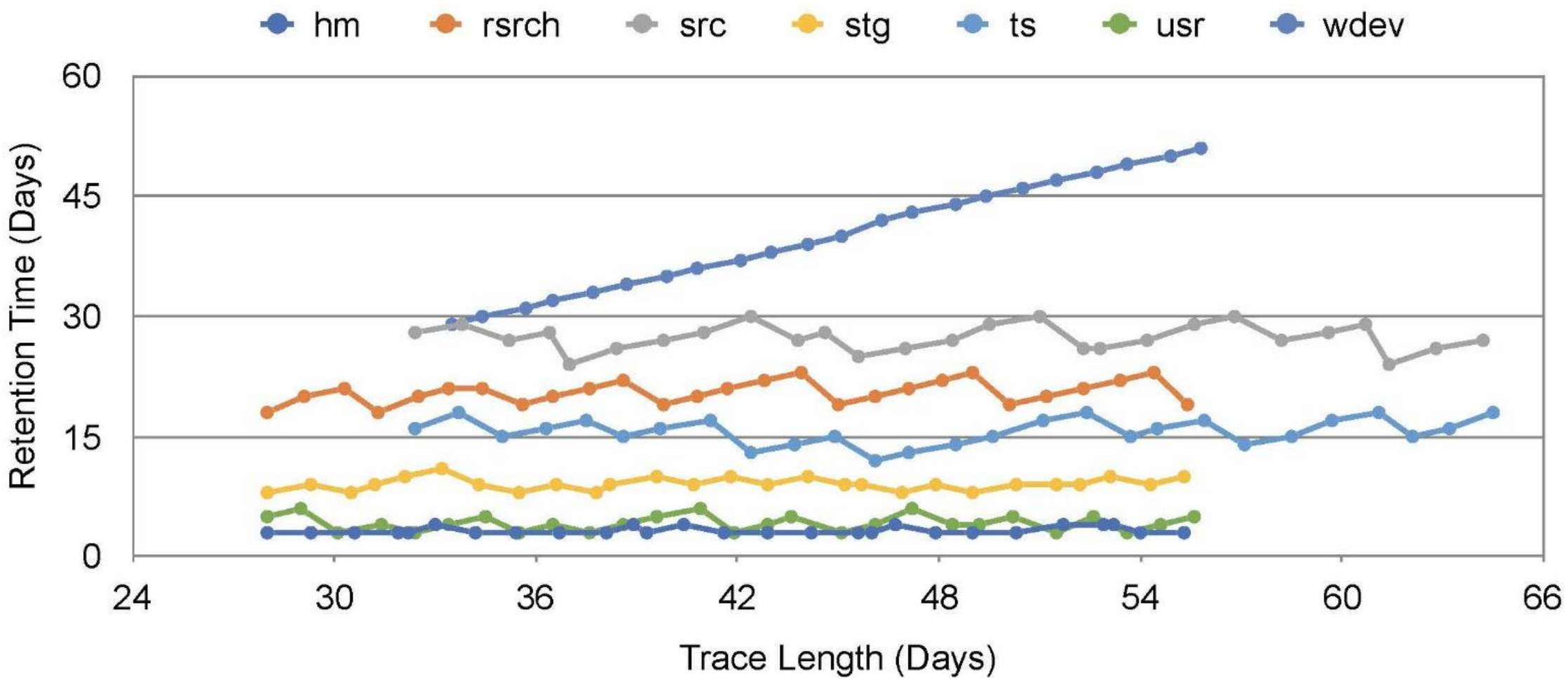
TimeSSD vs. Software-Based Solutions



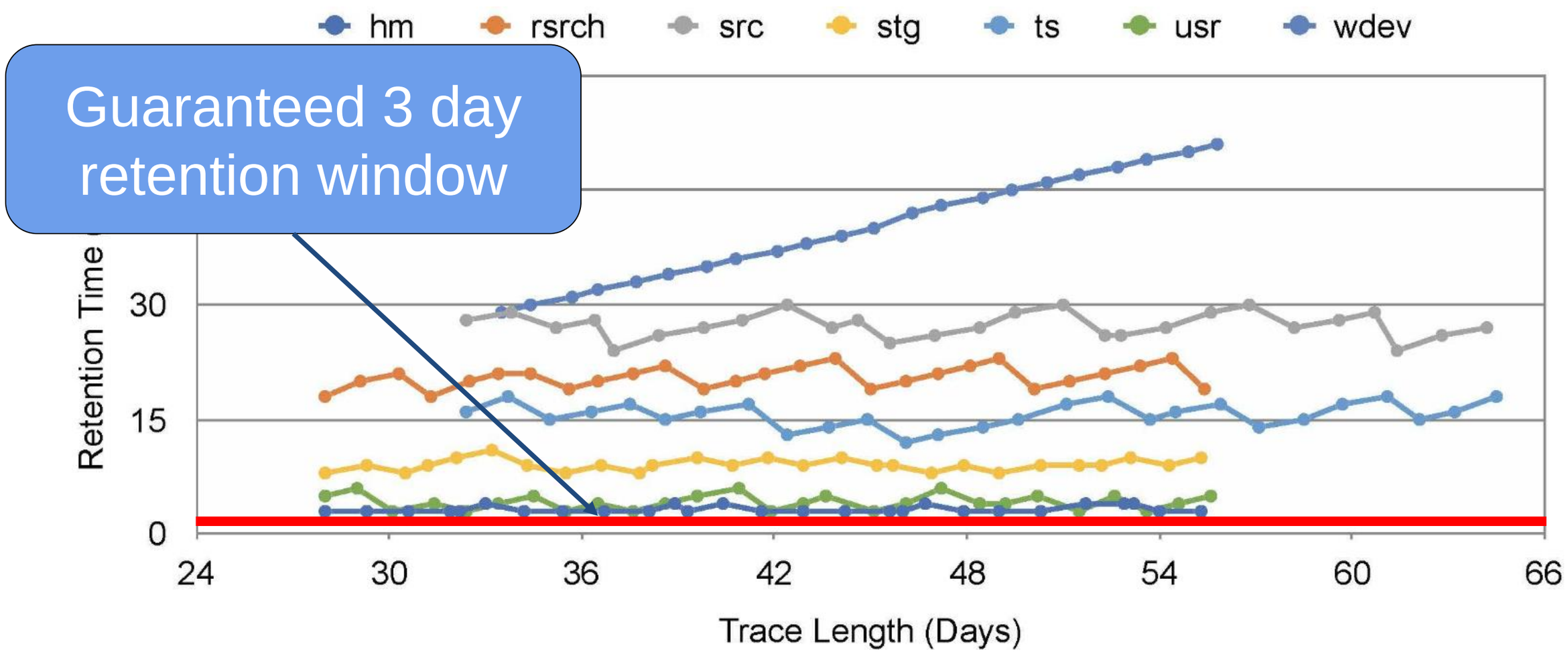
TimeSSD vs. Software-Based Solutions



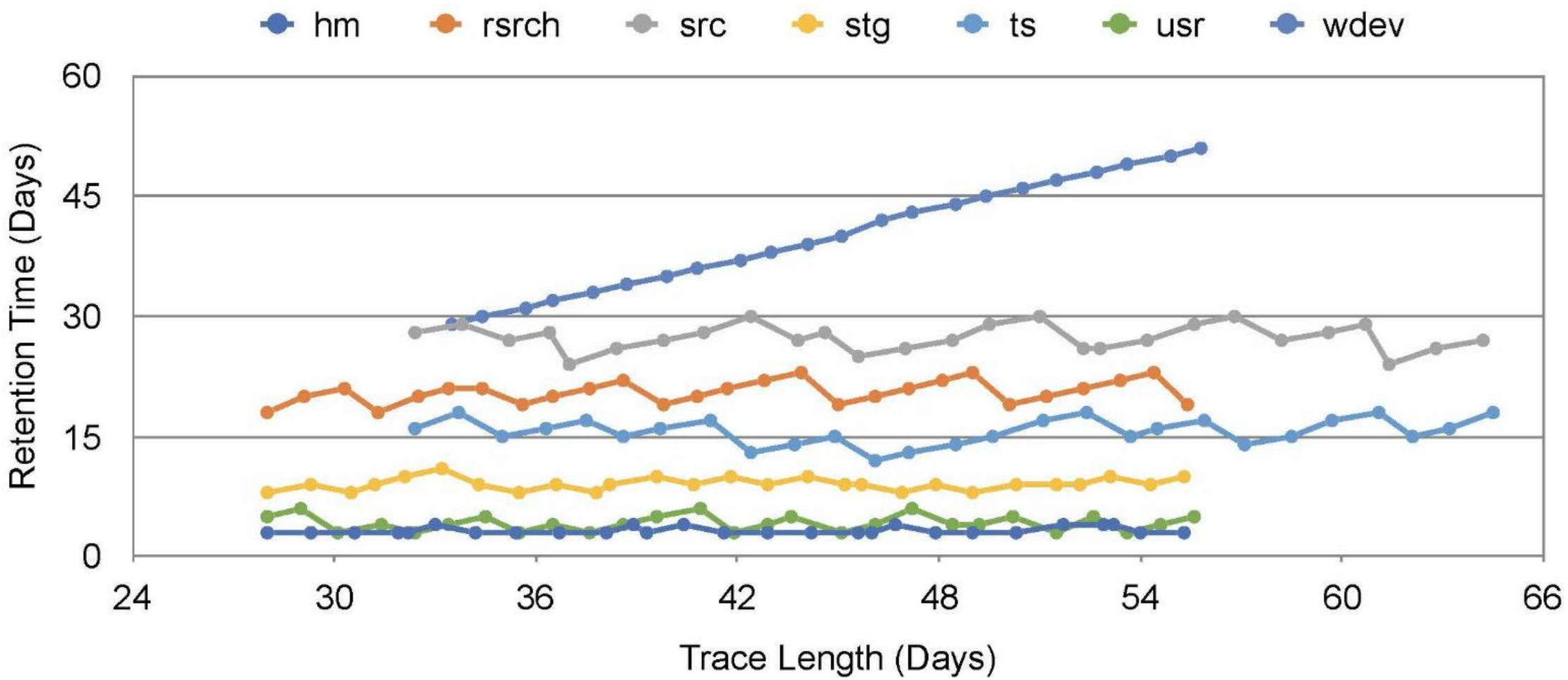
Workload-Adaptive State Retention



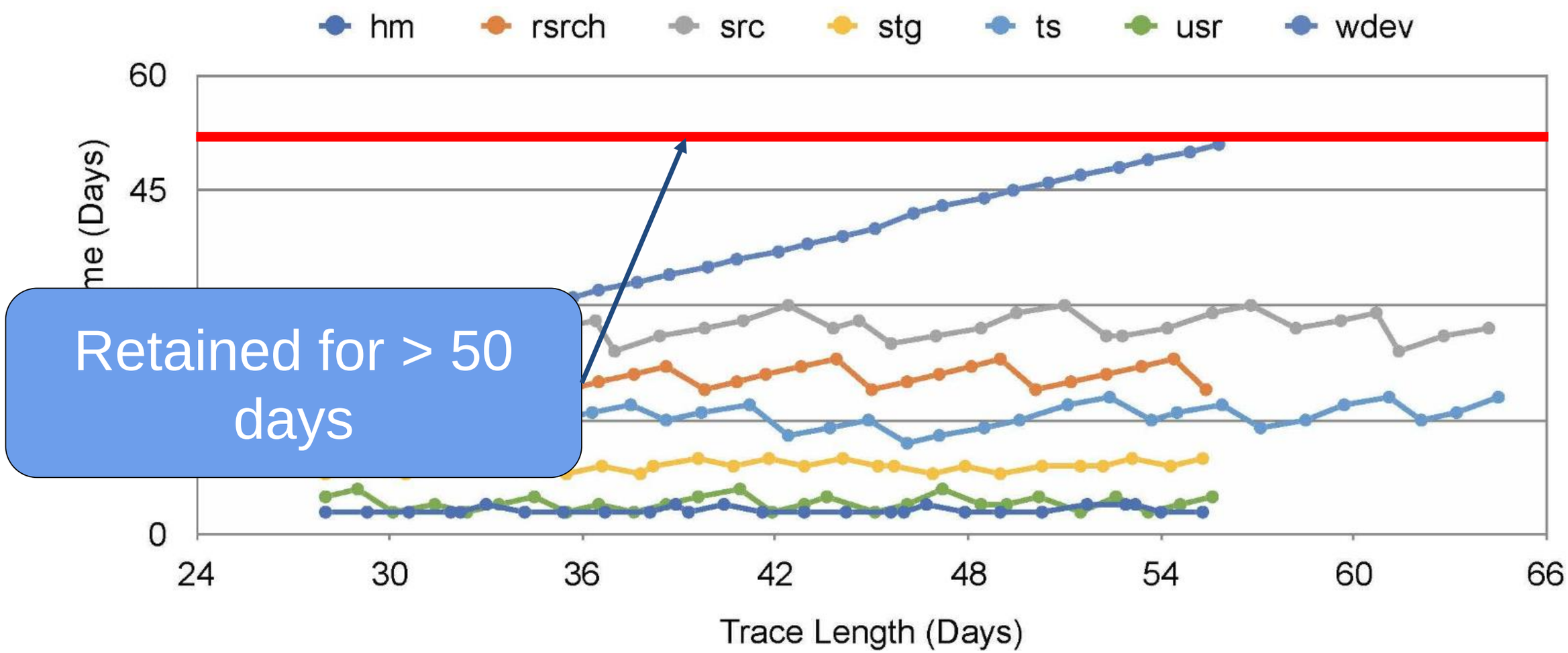
Workload-Adaptive State Retention



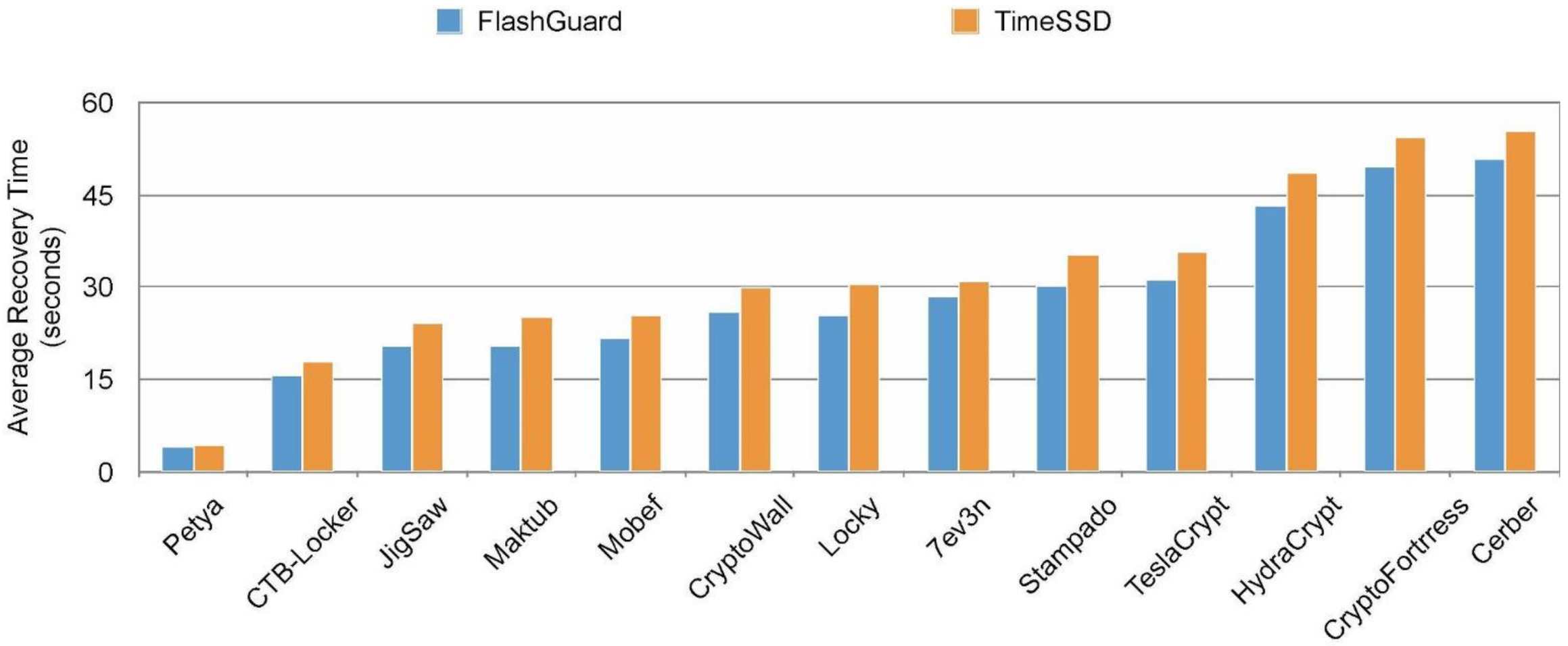
Workload-Adaptive State Retention



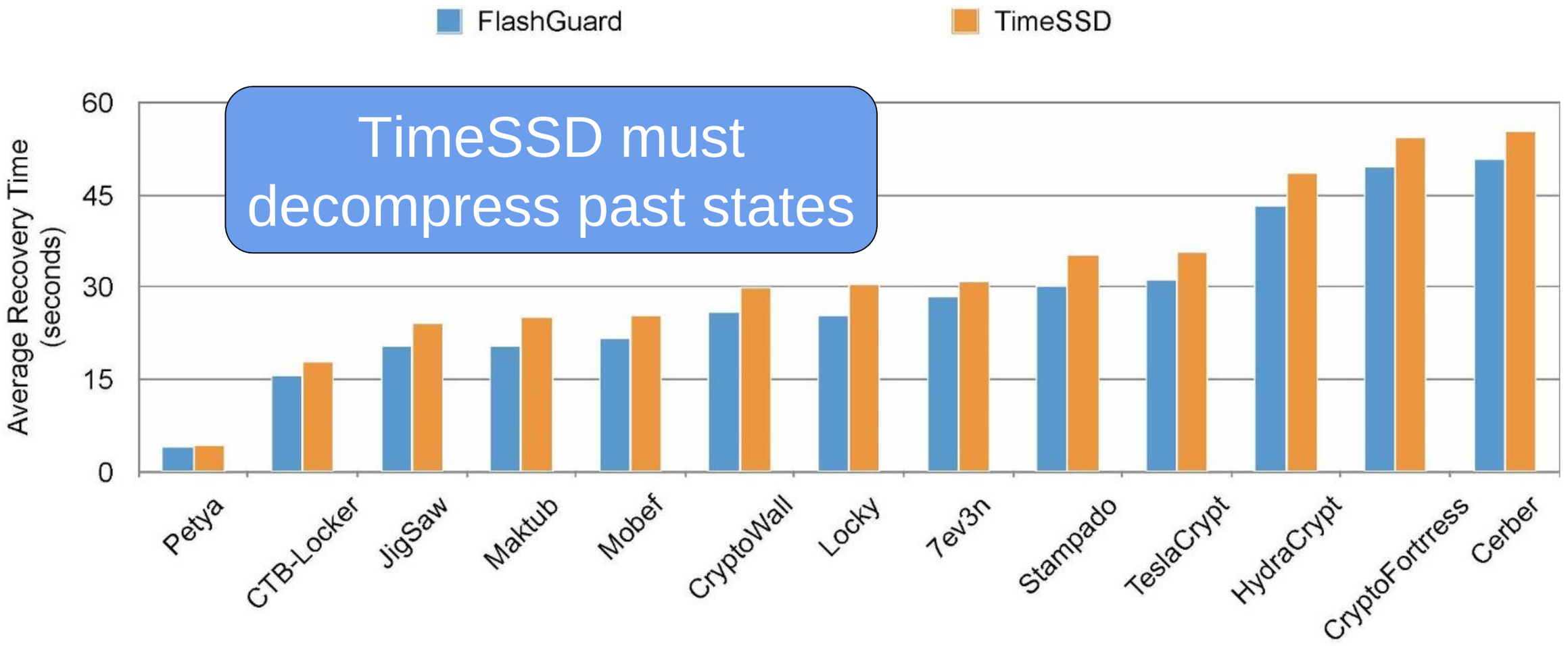
Workload-Adaptive State Retention



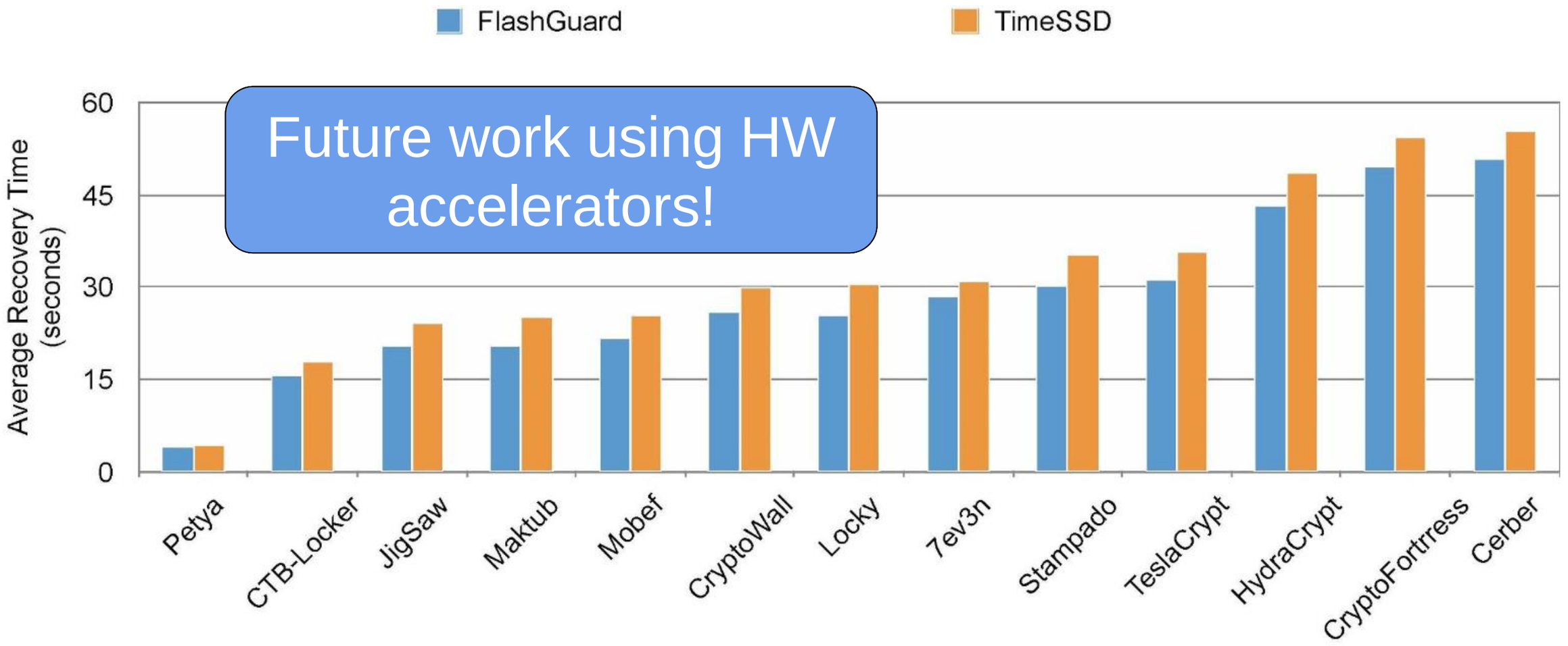
Data Recovery Time After Ransomware Attacks



Data Recovery Time After Ransomware Attacks

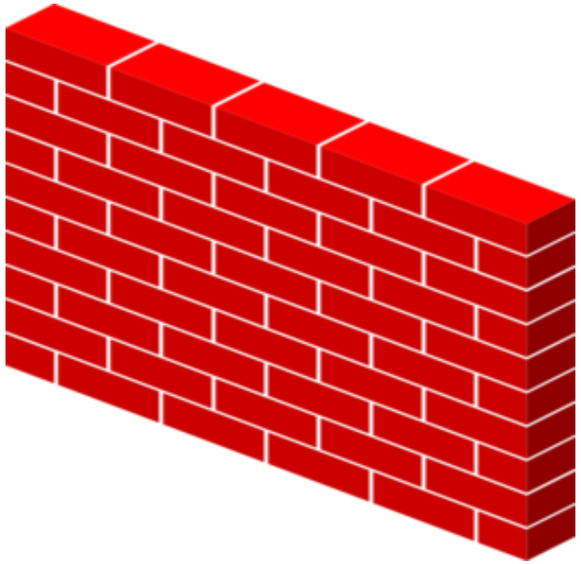


Data Recovery Time After Ransomware Attacks



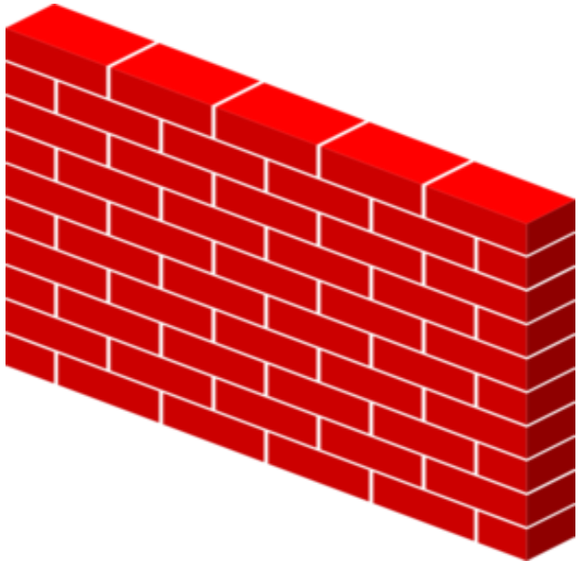
Future work using HW accelerators!

Project Almanac Summary



Firmware Isolation
Increased Security

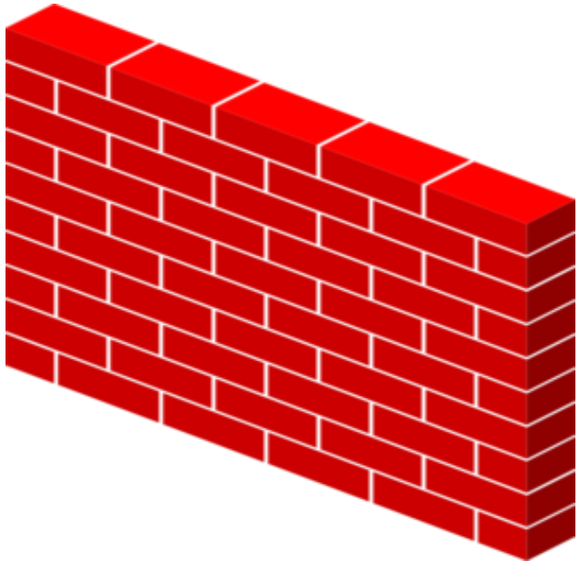
Project Almanac Summary

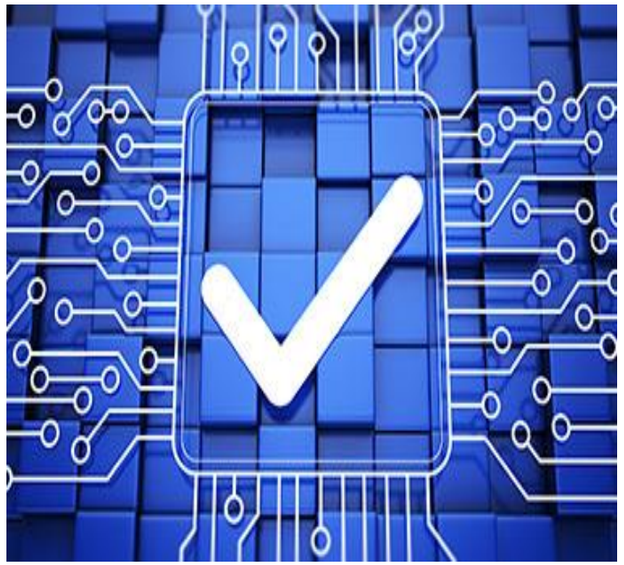


Firmware Isolation
Increased Security

Minimal Impact on
Performance and
Lifetime

Project Almanac Summary





Firmware Isolation Increased Security

Minimal Impact on Performance and Lifetime

Achieved Software Functionality

Thanks!

Xiaohao Wang Yifan Yuan You Zhou **Chance C. Coats** Jian Huang

Systems and Platform Research Group

ILLINOIS

Q&A