Lamassu: Storage-Efficient Host-Side Encryption

Peter Shah, Won So

Advanced Technology Group

9 July, 2015

Agenda

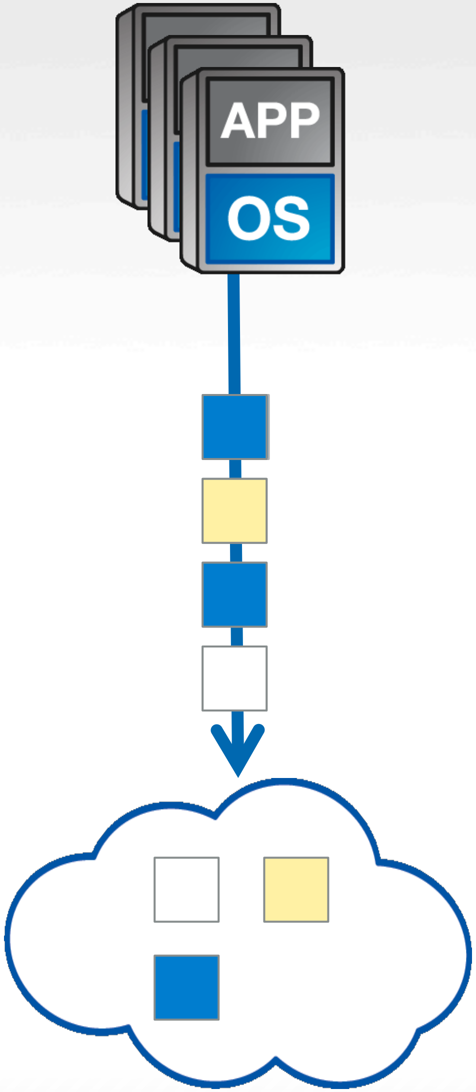
1) Overview
2) Security
3) Solution Architecture
4) Experimental Results
5) Conclusion

Overview

Architectural Goals

1) Enable external / untrusted storage

- Public Clouds, etc.



Overview

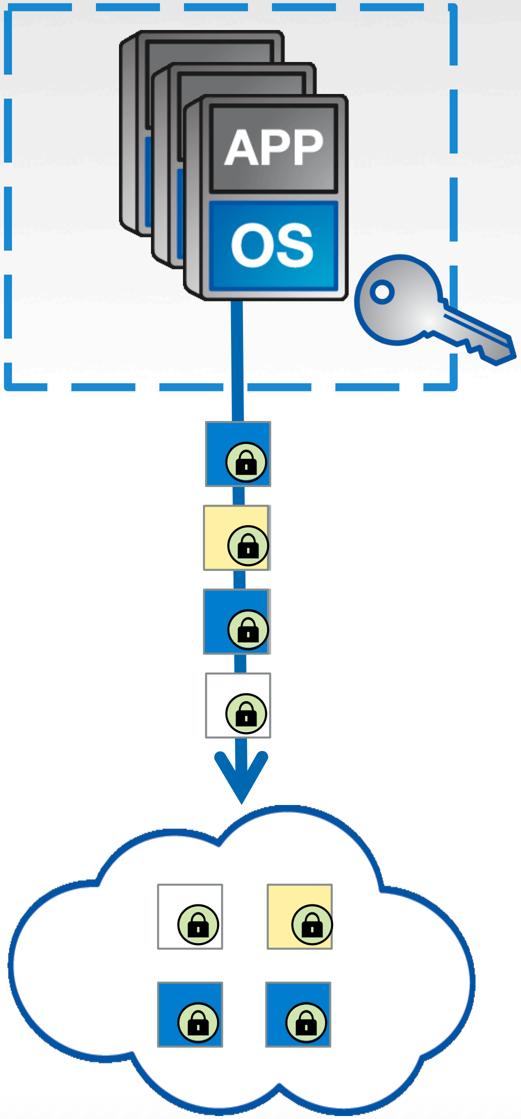
Architectural Goals

1) Enable external / untrusted storage

- Public Clouds, etc.

2) Provide data security

- Restrict trust domain



Overview

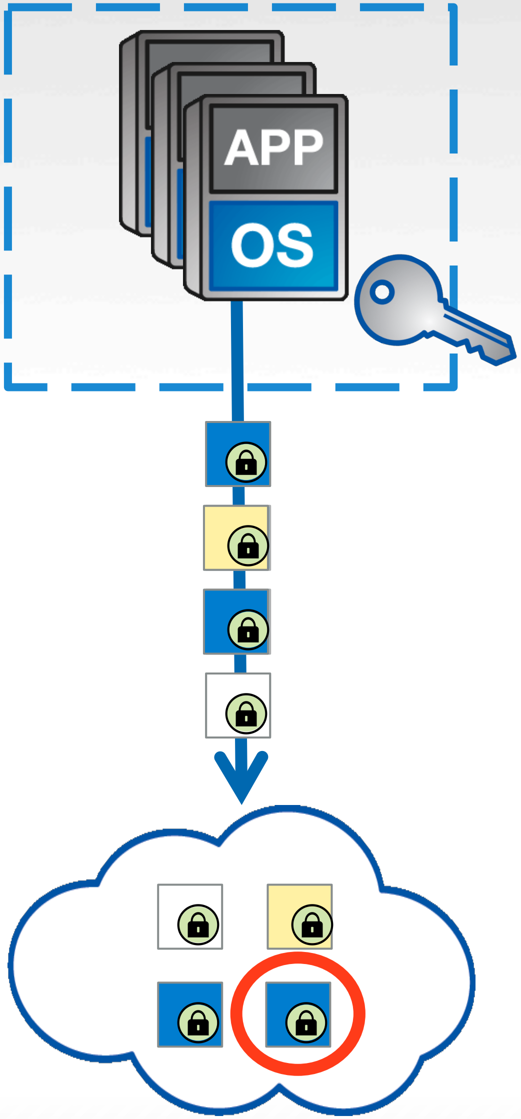
Architectural Goals

1) Enable external / untrusted storage

- Public Clouds, etc.

2) Provide data security

- Restrict trust domain



Overview

Architectural Goals

1) Enable external / untrusted storage

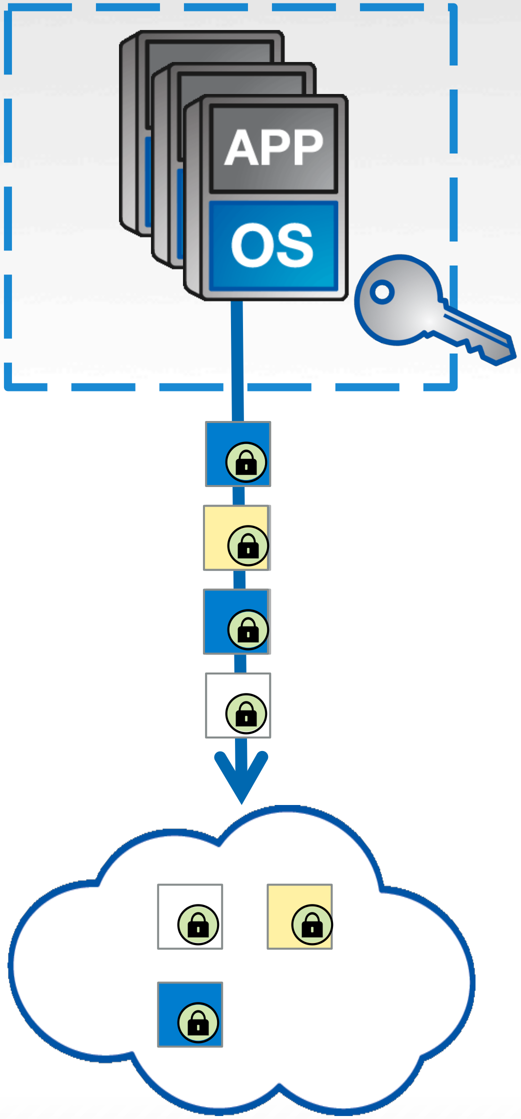
- Public Clouds, etc.

2) Provide data security

- Restrict trust domain

3) Preserve storage deduplication

- Use convergent encryption
- Focus on block-oriented deduplication



Overview

Architectural Goals

1) Enable external / untrusted storage

- Public Clouds, etc.

2) Provide data security

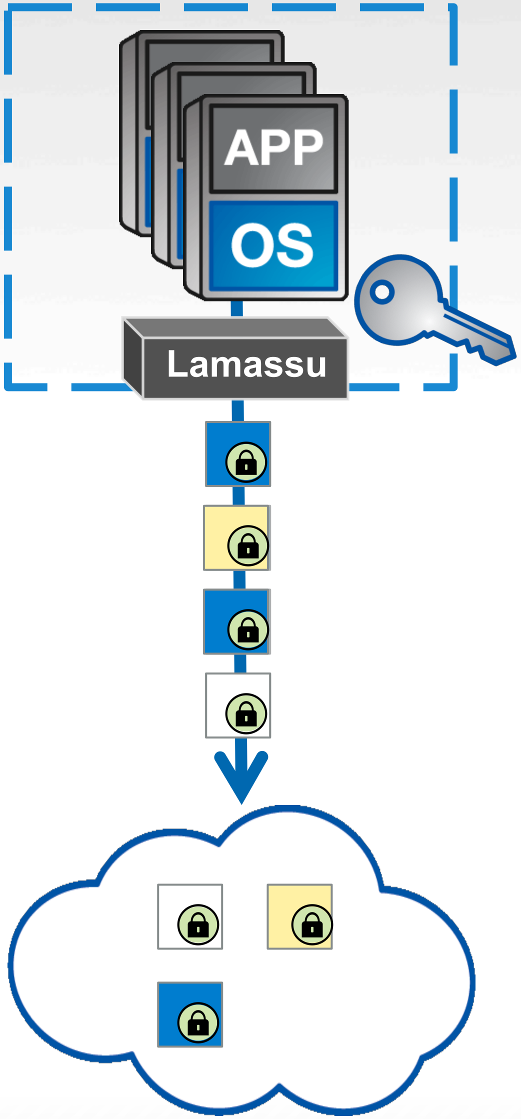
- Restrict trust domain

3) Preserve storage deduplication

- Use convergent encryption
- Focus on block-oriented deduplication

4) Work with existing applications

- Transparent addition
- No changes to app or storage systems
- Self-contained*



Security

Encryption Model

Convergent Encryption (CE)

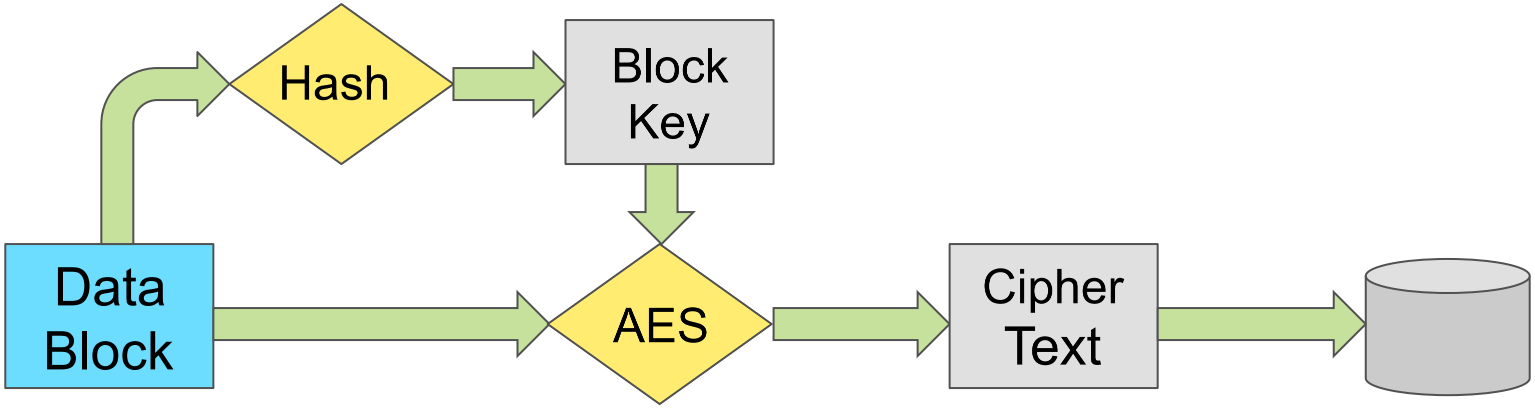
Equality-Preserving Encryption

- For any given plain text, convergent encryption will always produce the same cipher text.

Convergent Encryption

Message-Locked Encryption (MLE)

- For any given plain text, convergent encryption will always produce the same cipher text.
- Most common form: Key derived from data

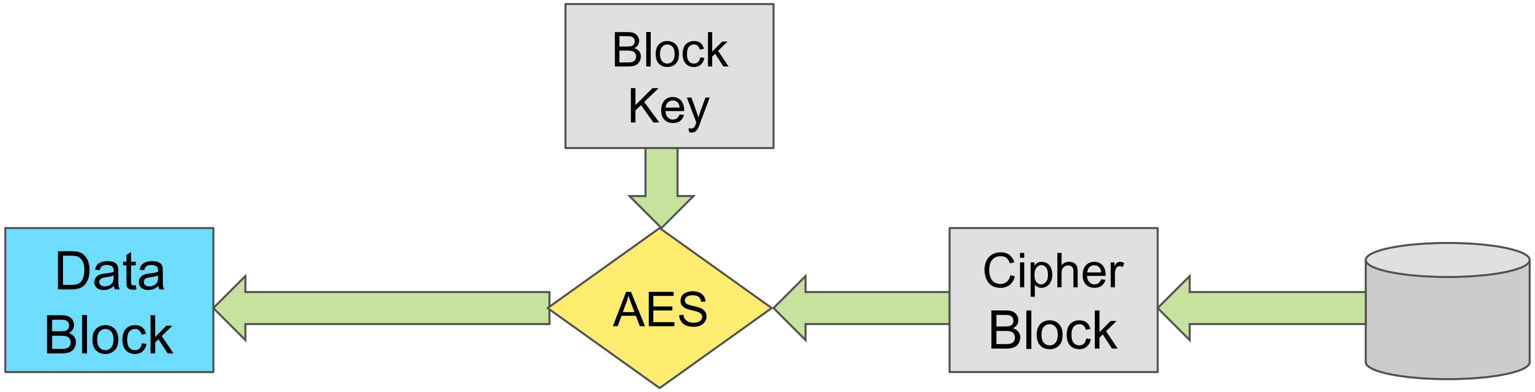


Message-locked encryption path

Convergent Encryption

Message-Locked Encryption (MLE)

- For any given plain text, convergent encryption will always produce the same cipher text.
- Most common form: Key derived from data

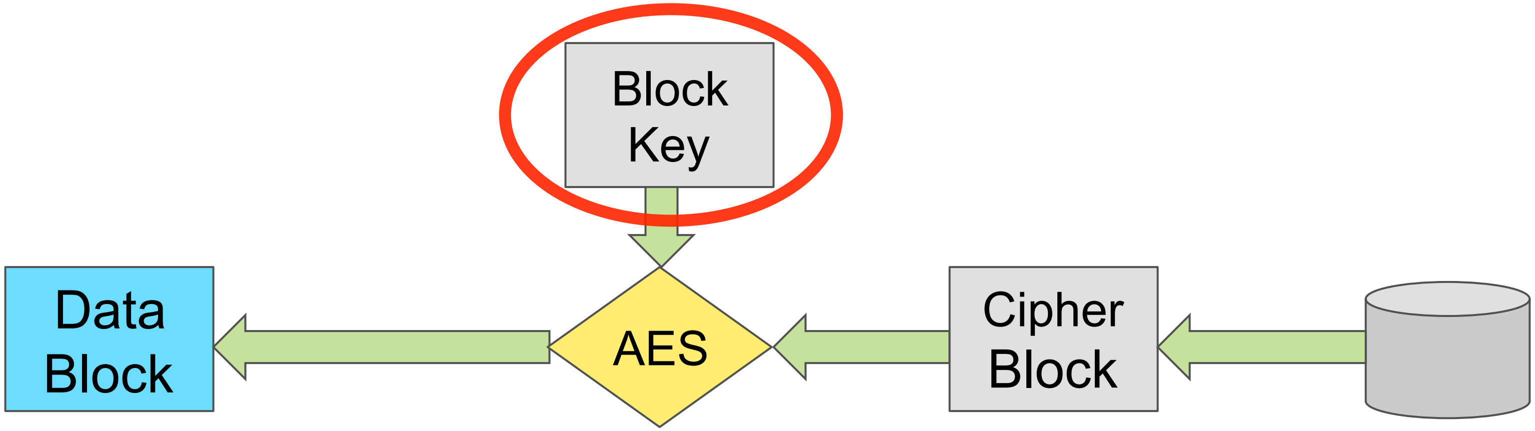


Message-locked decryption path

Convergent Encryption

Message-Locked Encryption (MLE)

- For any given plain text, convergent encryption will always produce the same cipher text.
- Most common form: Key derived from data

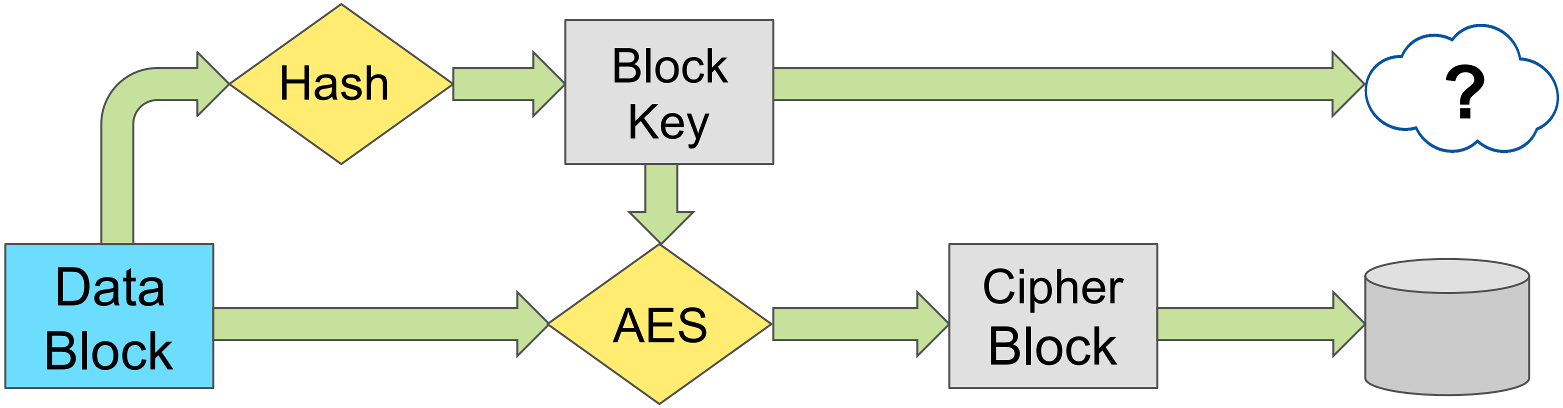


Message-locked decryption path

Convergent Encryption

Key Storage

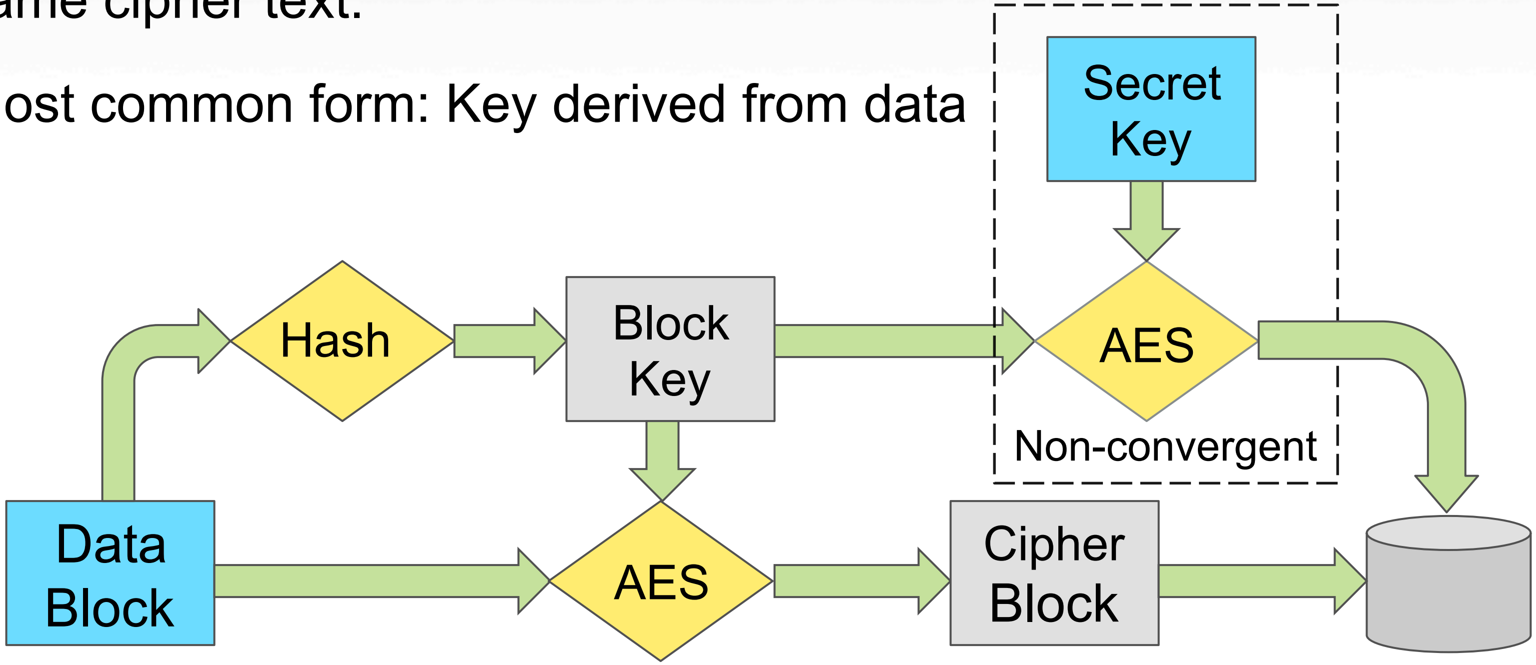
- For any given plain text, convergent encryption will always produce the same cipher text.
- Most common form: Key derived from data



Convergent Encryption

Key Storage

- For any given plain text, convergent encryption will always produce the same cipher text.
- Most common form: Key derived from data



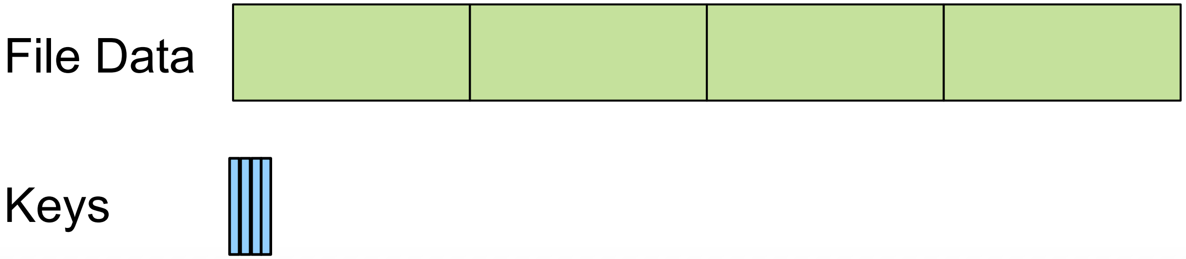
Metadata Storage

Key Storage Architecture

Keys as Metadata

Transparent Key Management

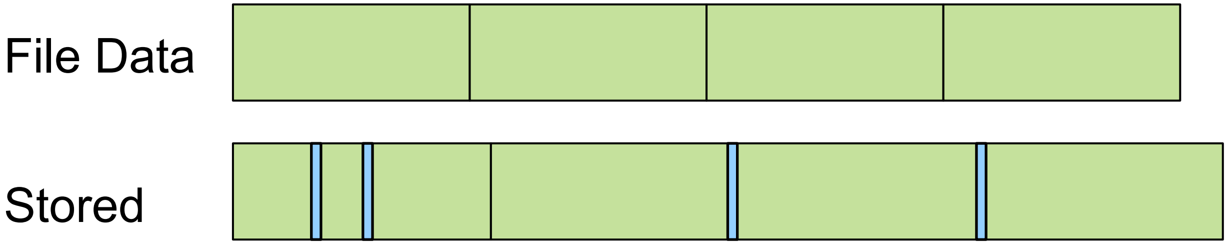
- Treat per-block hash-keys as file metadata
  - Potentially hundreds, or thousands per file



Keys as Metadata

Transparent Key Management

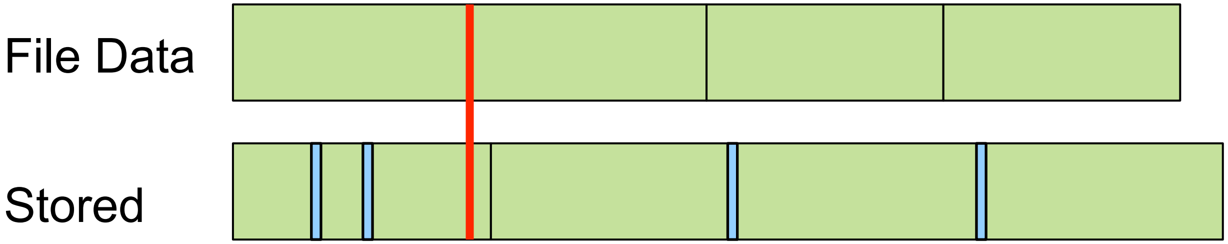
- Treat per-block hash-keys as file metadata
  - Potentially hundreds, or thousands per file
- Store keys inside each file
  - Preserve transparency
  - Allow external storage to copy, rename, etc.



Keys as Metadata

Transparent Key Management

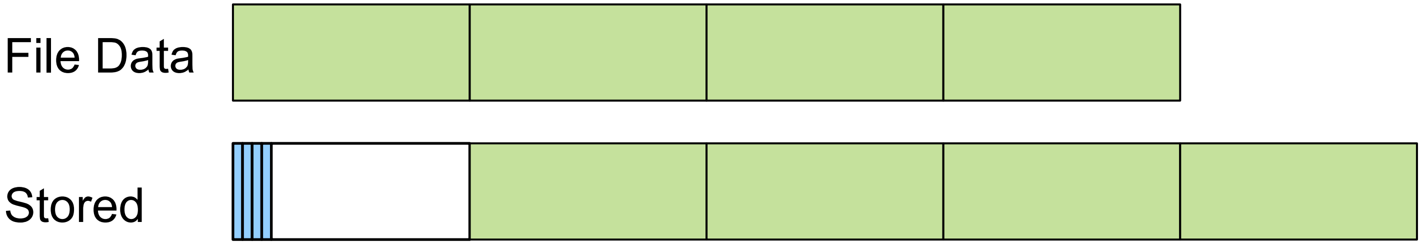
- Treat per-block hash-keys as file metadata
  - Potentially hundreds, or thousands per file
- Store keys inside each file
  - Preserve transparency
  - Allow external storage to copy, rename, etc.
- Separate data from metadata
  - Keep keys from polluting duplicate blocks
  - Keep added data from breaking block alignment



Keys as Metadata

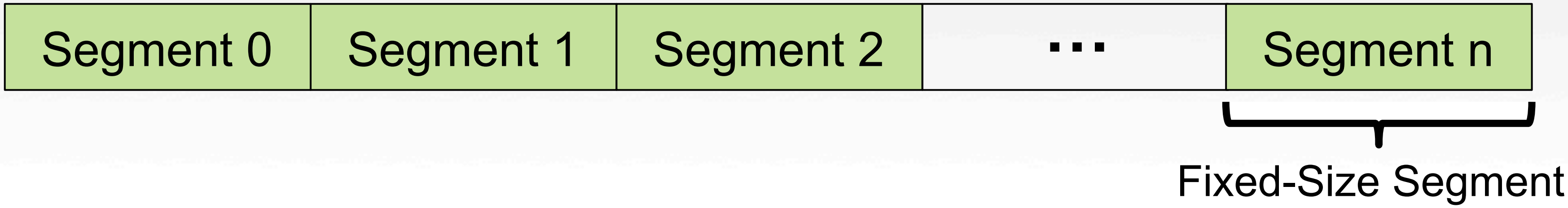
Transparent Key Management

- Treat per-block hash-keys as file metadata
  - Potentially hundreds, or thousands per file
- Store keys inside each file
  - Preserve transparency
  - Allow external storage to copy, rename, etc.
- Separate data from metadata
  - Keep keys from polluting duplicate blocks
  - Keep added data from breaking block alignment



File Structure

Logical File Layout



Segment 0

Segment 1

Segment 2

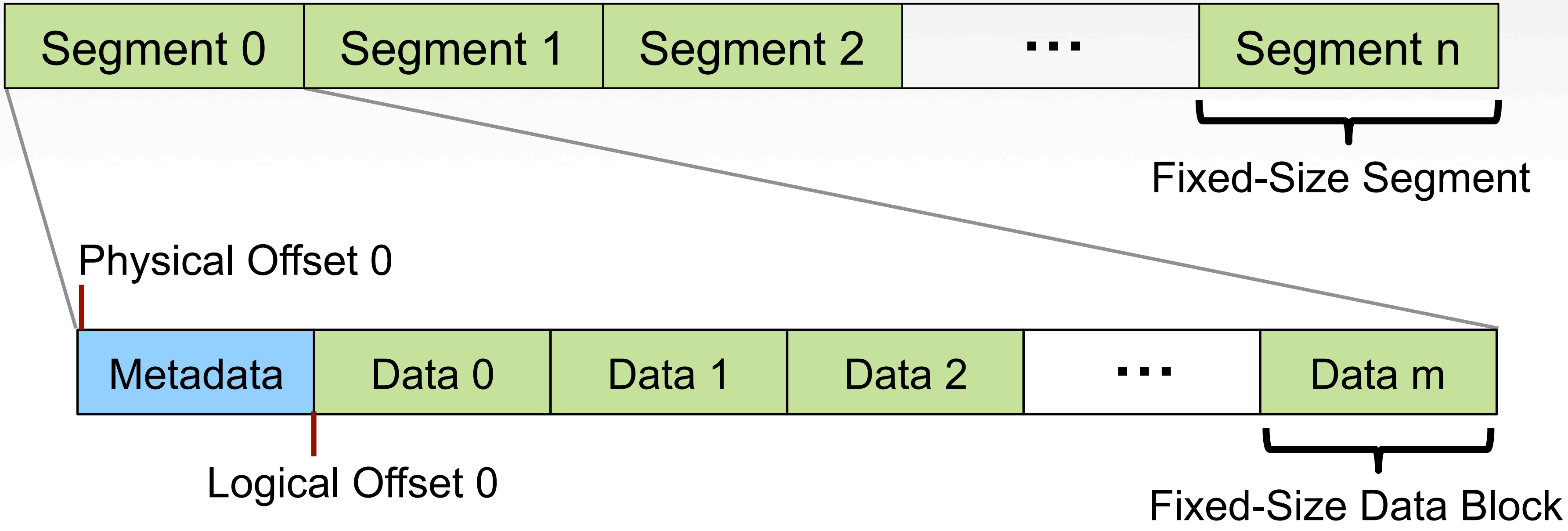
…

Segment n

Fixed-Size Segment

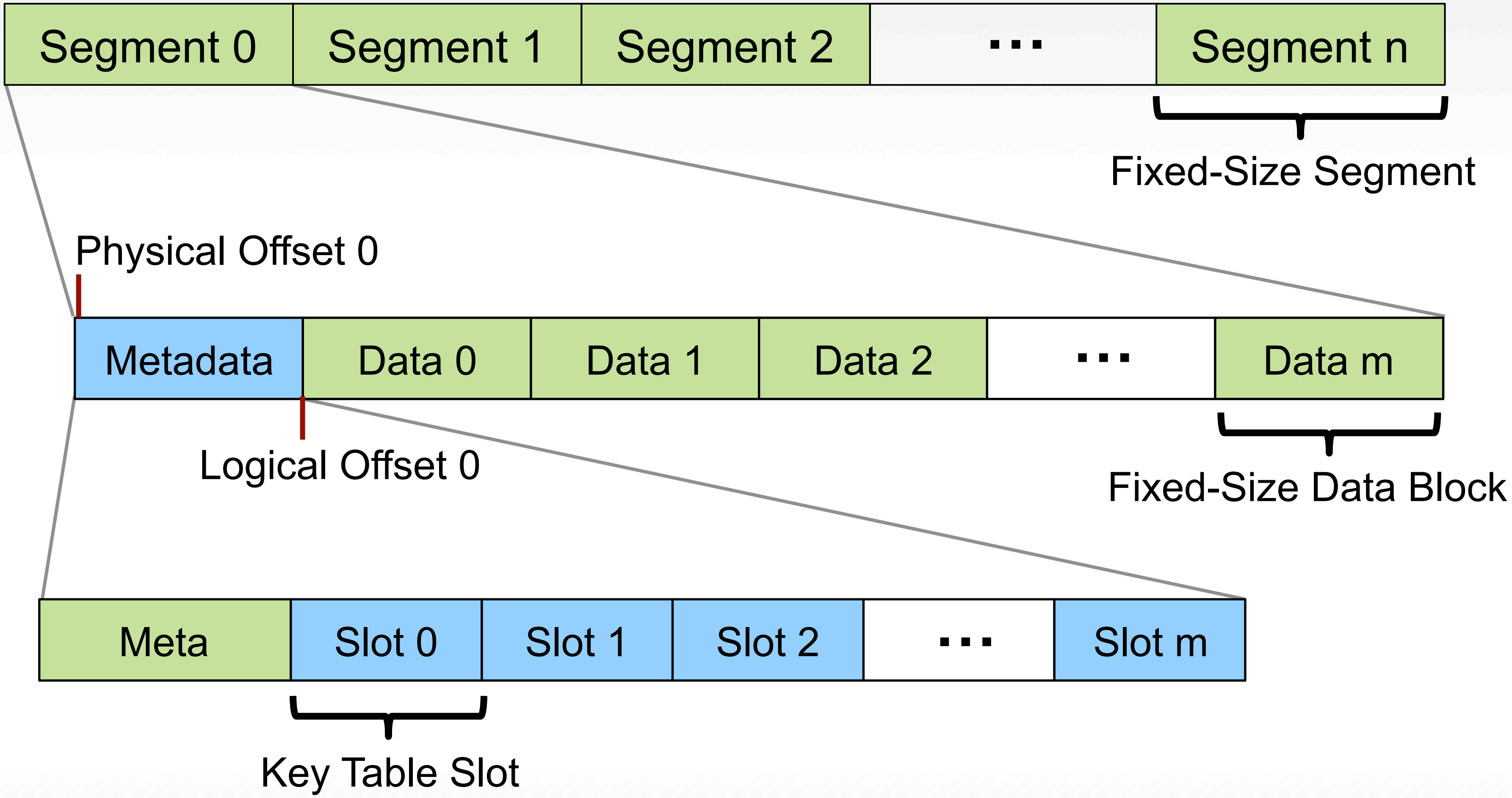
File Structure

Logical File Layout



File Structure

Logical File Layout



Metadata Consistency

Crash Detection and Recovery

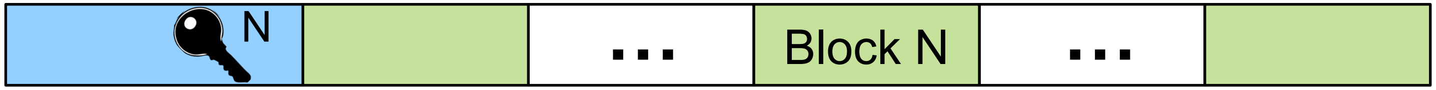
- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes

Metadata Consistency

Crash Detection and Recovery

- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes

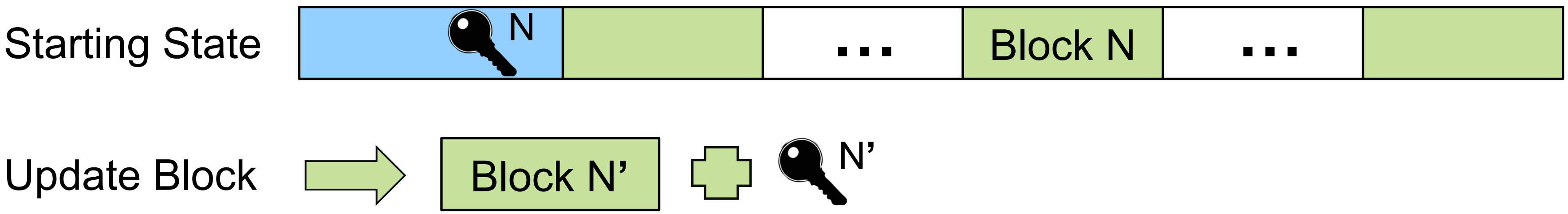
Starting State



Metadata Consistency

Crash Detection and Recovery

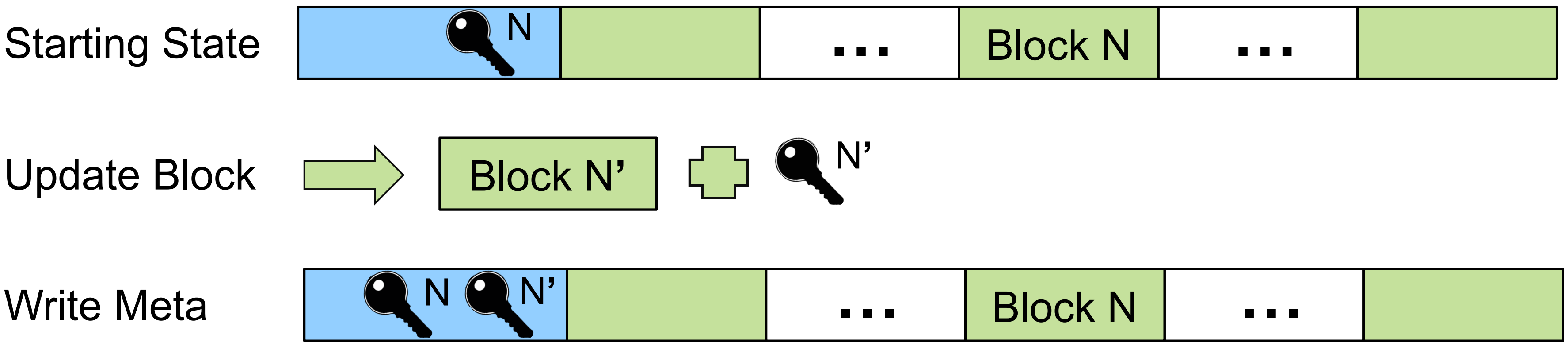
- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes



Metadata Consistency

Crash Detection and Recovery

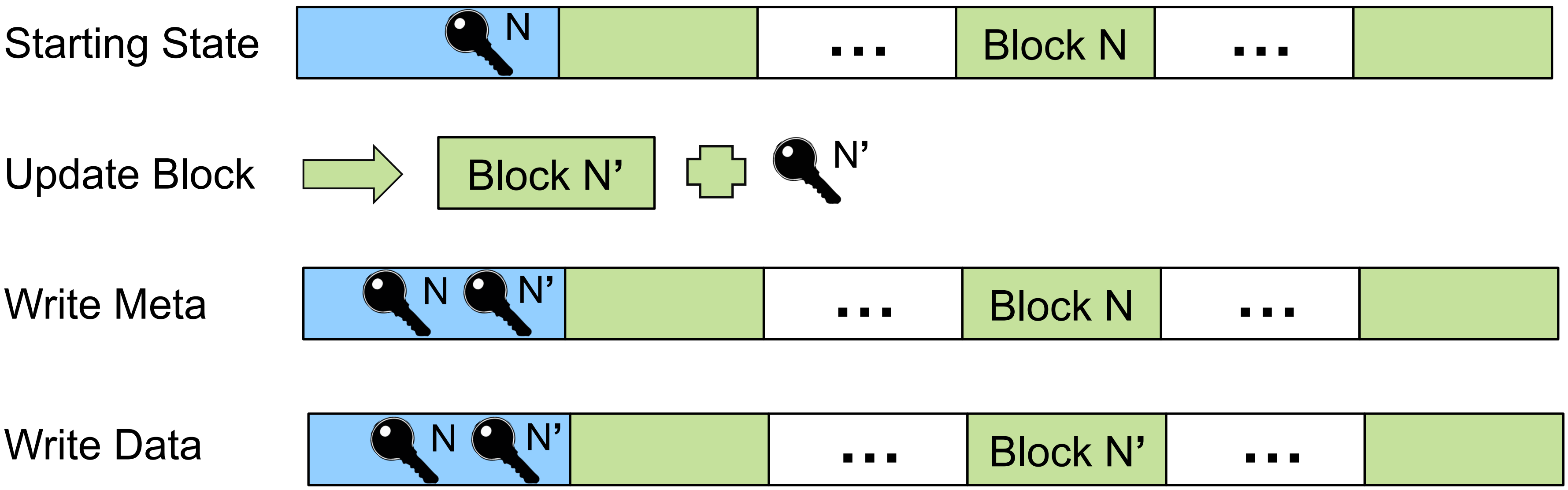
- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes



Metadata Consistency

Crash Detection and Recovery

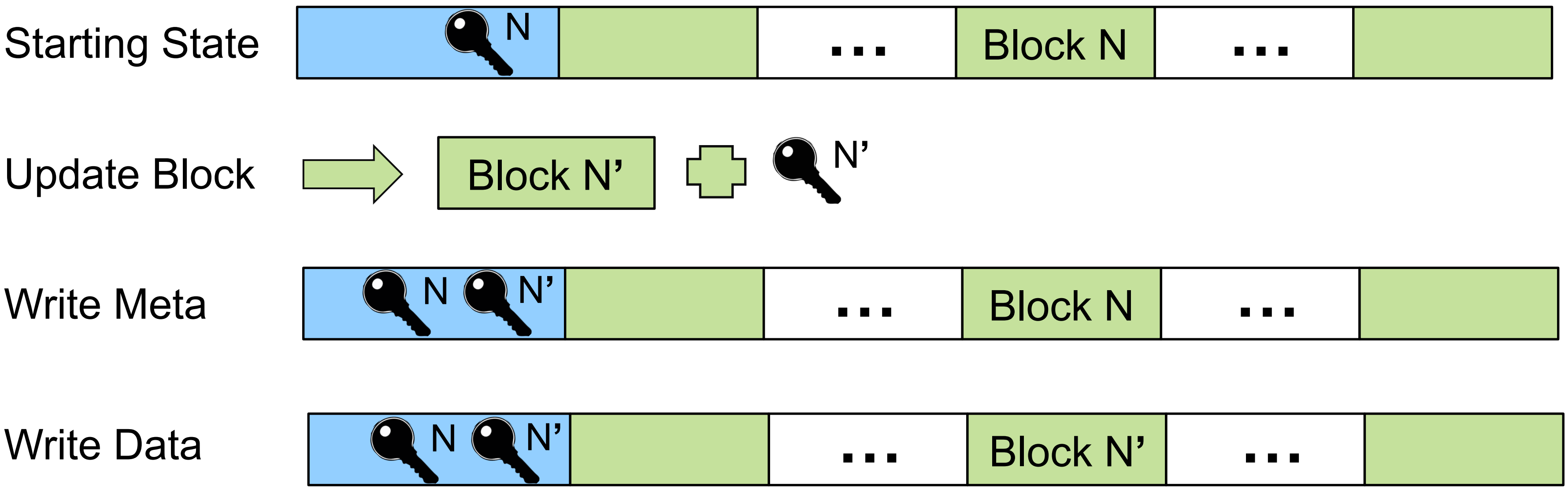
- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes



Metadata Consistency

Crash Detection and Recovery

- Data and metadata must be in sync
  - Depends on underlying storage to prevent partial writes



Starting State: N | | ... | Block N | ... |

Update Block: Block N' + N'

Write Meta: N N' | | ... | Block N | ... |

Write Data: N N' | | ... | Block N' | ... |

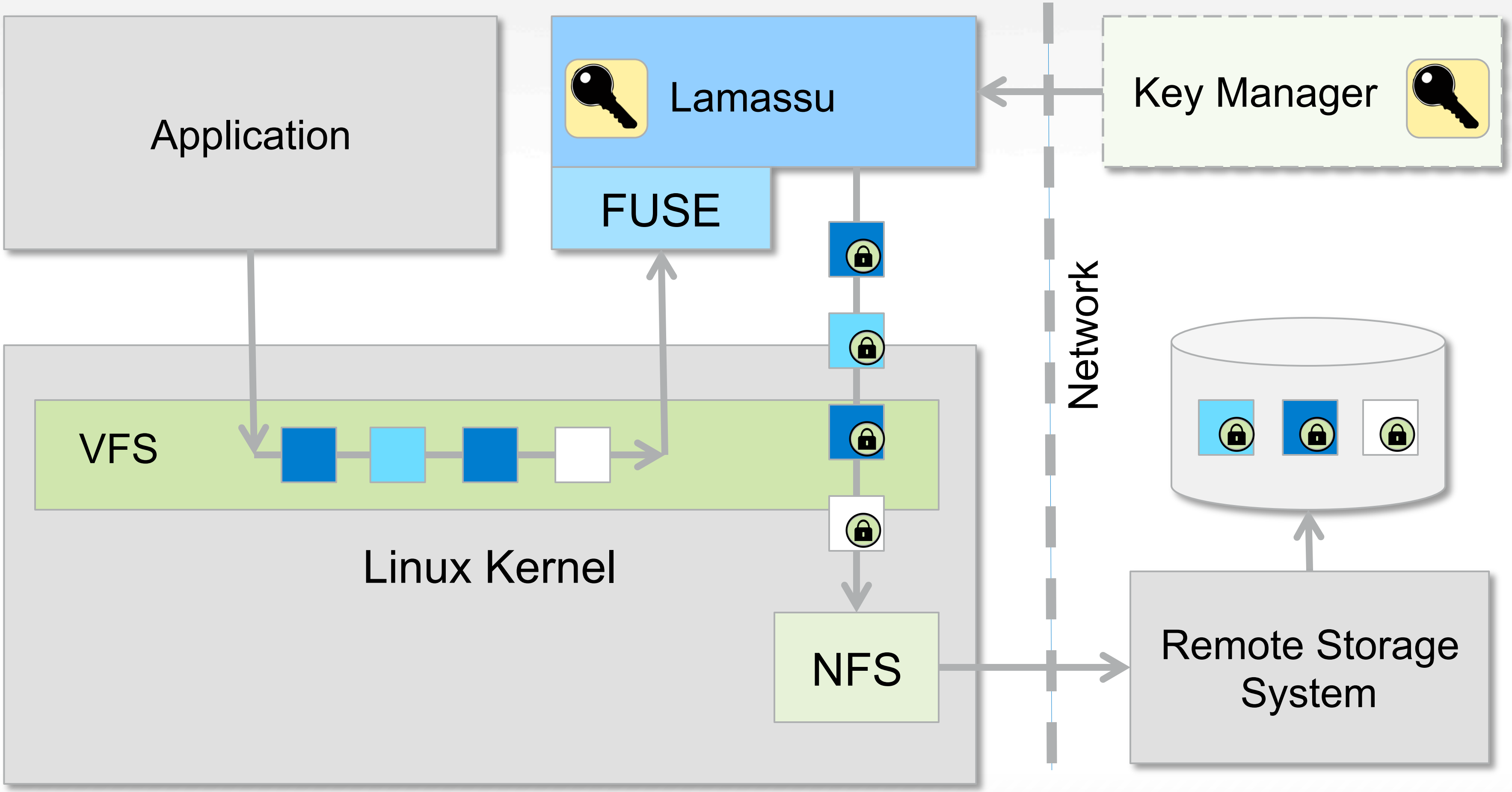
- Stale keys are cleaned up during subsequent metadata updates

Results

Storage Efficiency & Performance

Overview

Prototype Implementation



Comparison with other Systems

Benchmarking Strategy

1) PlainFS

- FUSE-based (pass-through)

Comparison with other Systems

Benchmarking Strategy

1) PlainFS

- FUSE-based (pass-through)

2) EncFS

- FUSE-based
- Provides AES encryption

Comparison with other Systems

Benchmarking Strategy

1) PlainFS

- FUSE-based (pass-through)

2) EncFS

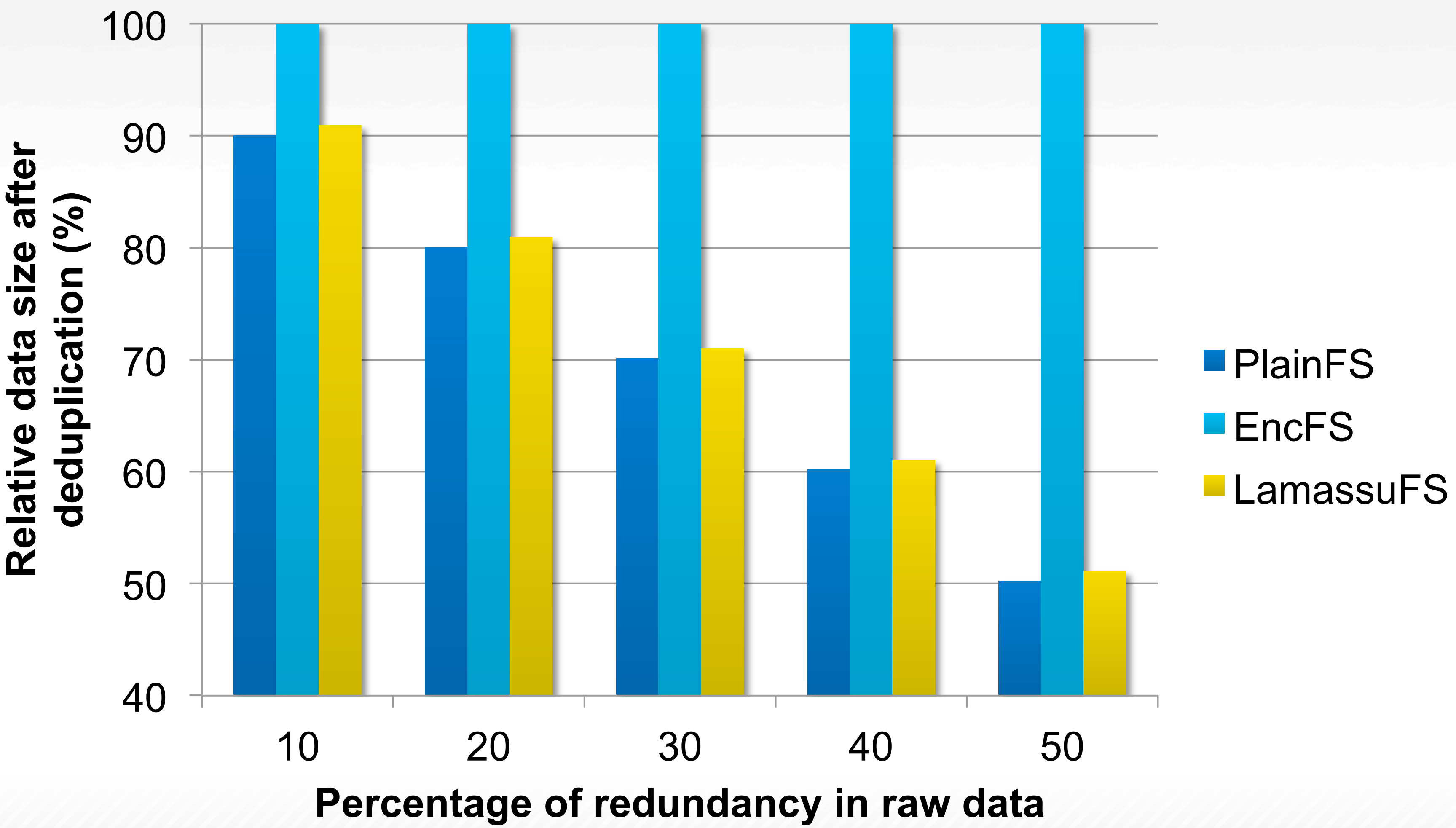
- FUSE-based
- Provides AES encryption

3) LamassuFS

- FUSE-based
- Provides AES encryption
- Provides convergent encryption

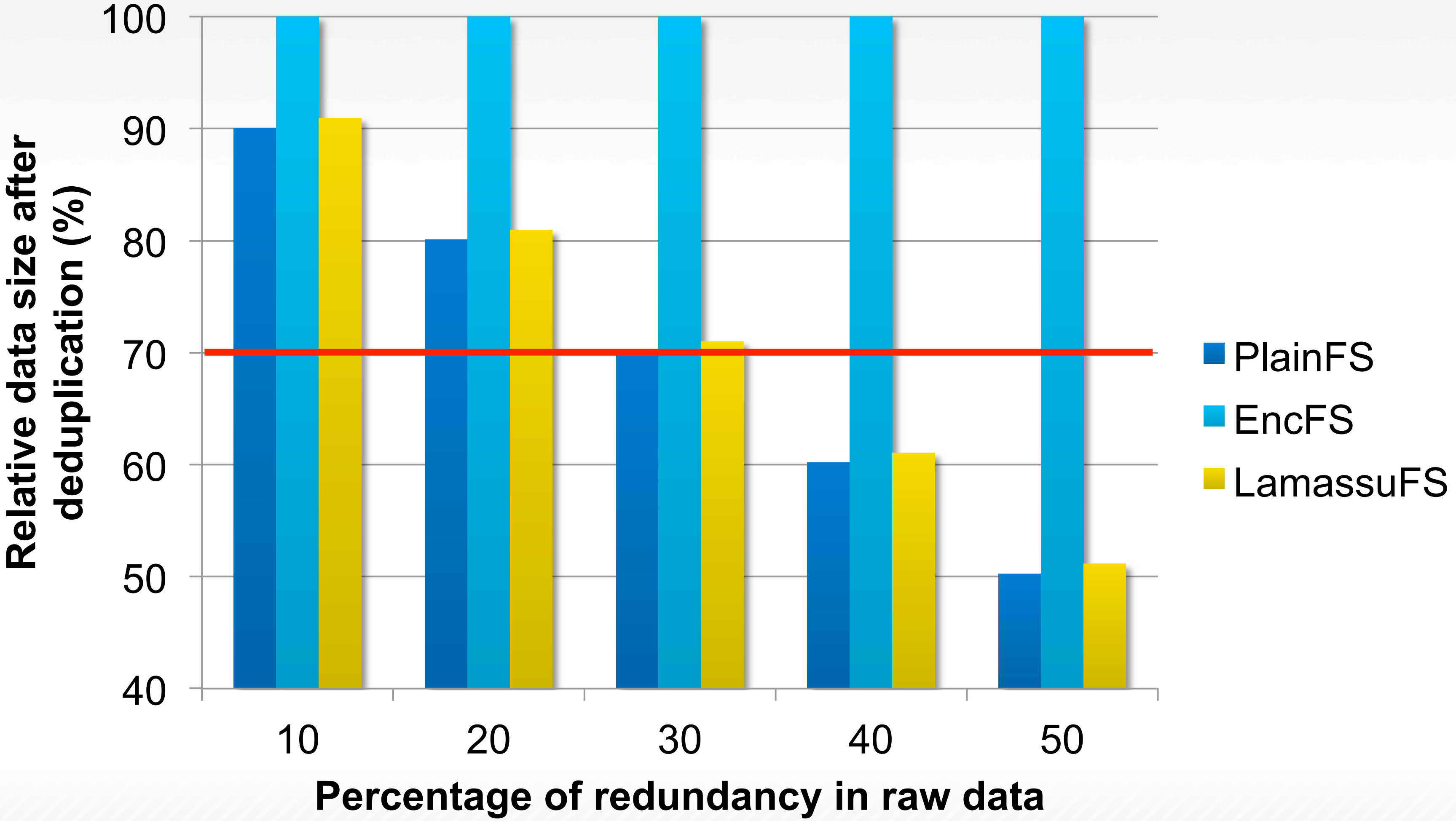
Deduplication Results

Comparison of Deduplication Ratios



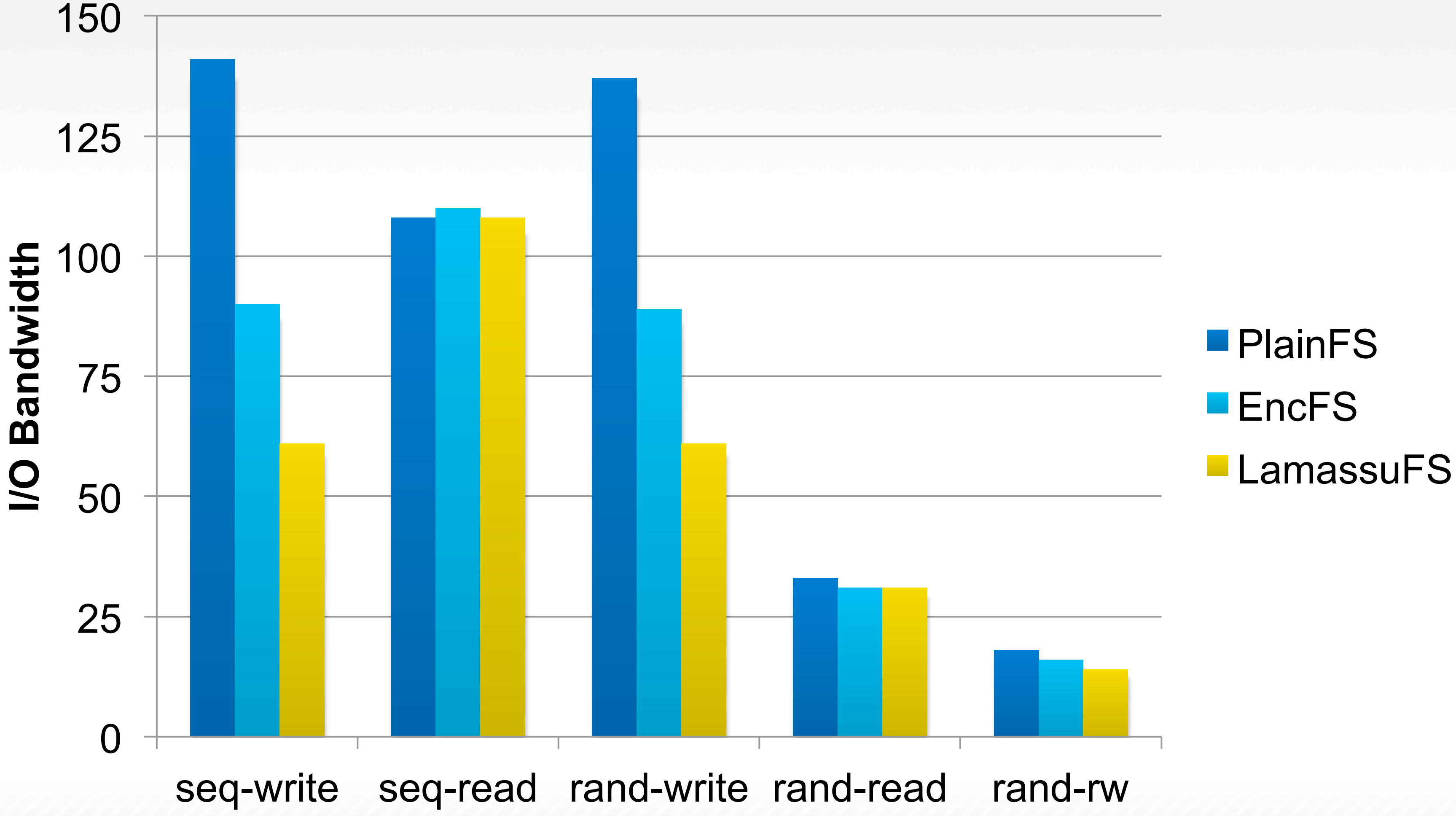
Deduplication Results

Comparison of Deduplication Ratios



Singe File I/O Throughput

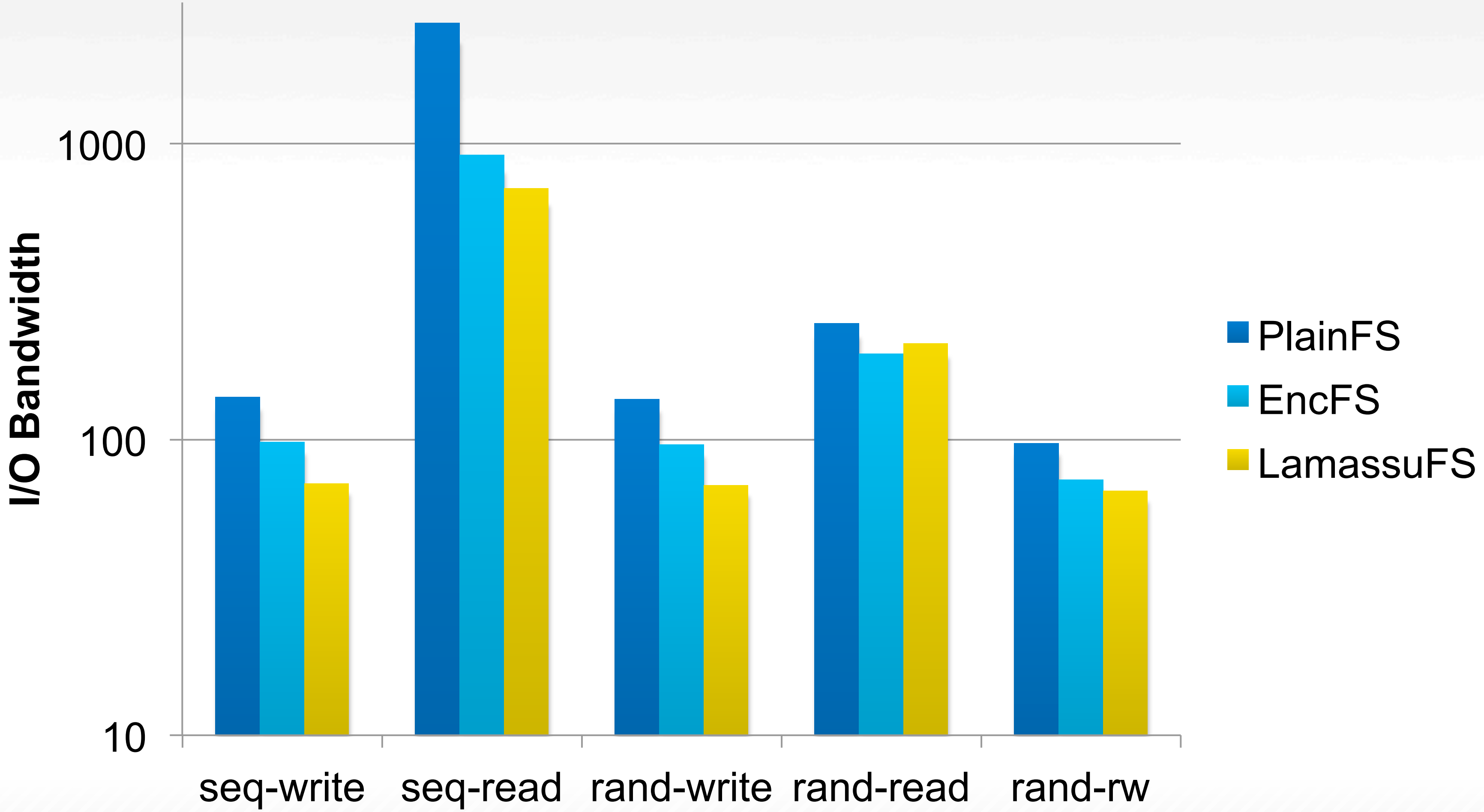
Comparison with other FUSE systems using remote NFS storage



NetApp®

Single File I/O Throughput

Comparison with other FUSE systems using local DRAM storage



Conclusions

Recap and Observations

- Strong security on shared storage
  - Uses standard encryption techniques
- Preserves storage-based deduplication
- Transparent to both application and storage
  - Easy to deploy
- Flexible user-mode architecture
  - Can integrate with other host-side technologies

Conclusions

Recap and Observations

- Strong security on shared storage
  - Uses standard encryption techniques
- Preserves storage-based deduplication
- Transparent to both application and storage
  - Easy to deploy
- Flexible user-mode architecture
  - Can integrate with other host-side technologies

Questions?

Special Thanks

James Kelley

Thank You