Aerie: Flexible File-System Interfaces to Storage-Class Memory

[APSys'13, EuroSys'14]

Haris Volos[†]

Sanketh Nalli, Sankaralingam Panneerselvam,
Venkatanathan Varadarajan, Prashant Saxena,
Michael M. Swift

[†]HP Labs

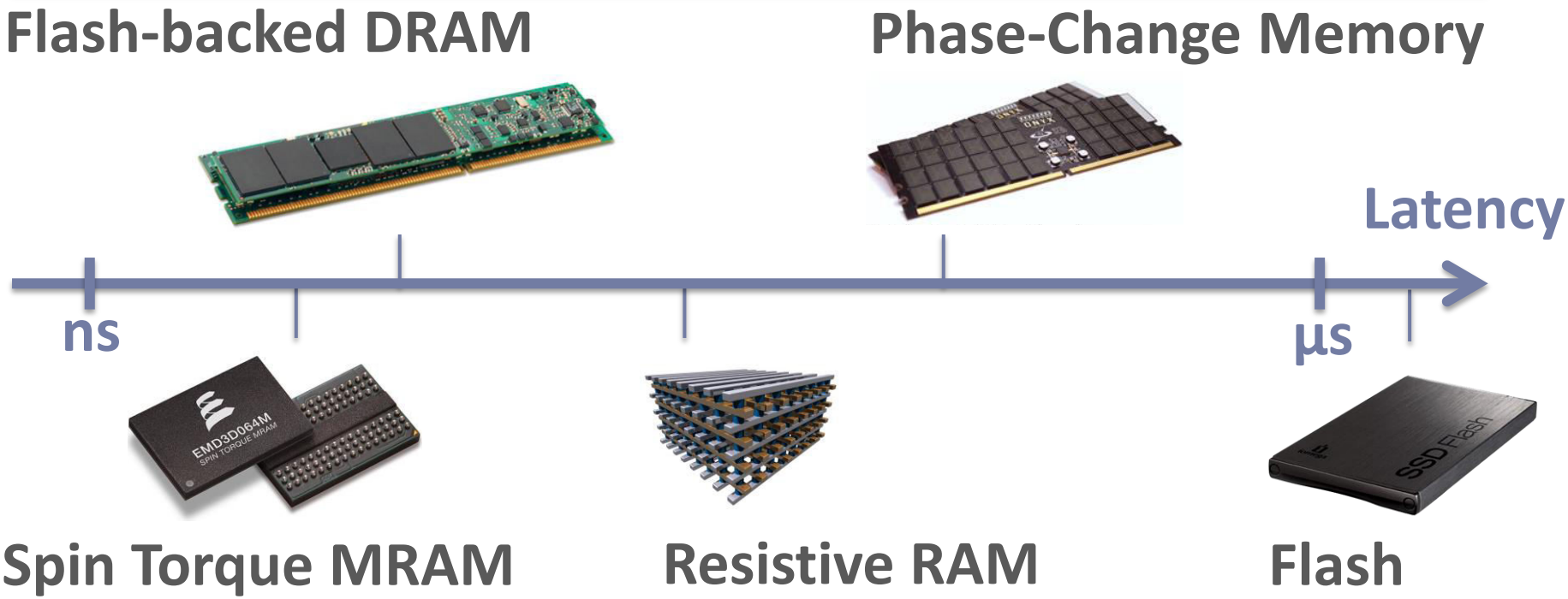
WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

WISDOM

Outline

- **Overview**
- Motivation: Interface flexibility
- Aerie: In-memory library file systems
- Evaluation

Storage-Class Memory (SCM)



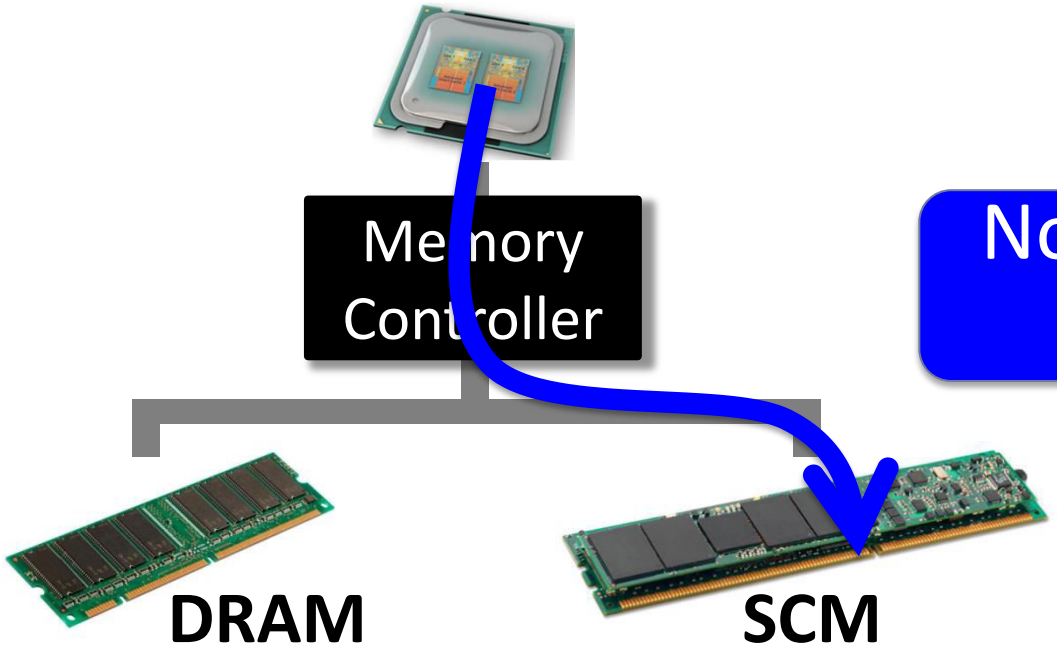
- Persistent
- Short access time

Software overhead matters

Storage-Class Memory (SCM)

- Persistent
- Short access time
- Byte addressable

Accessible via loads/stores



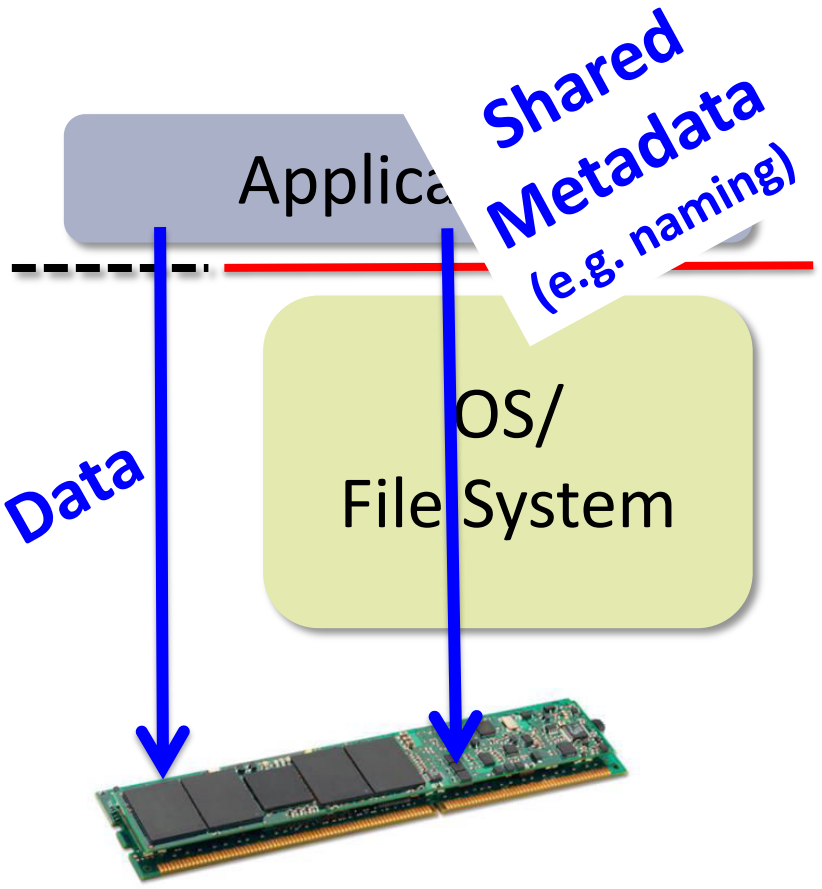
No software overhead

Accessing SCM today

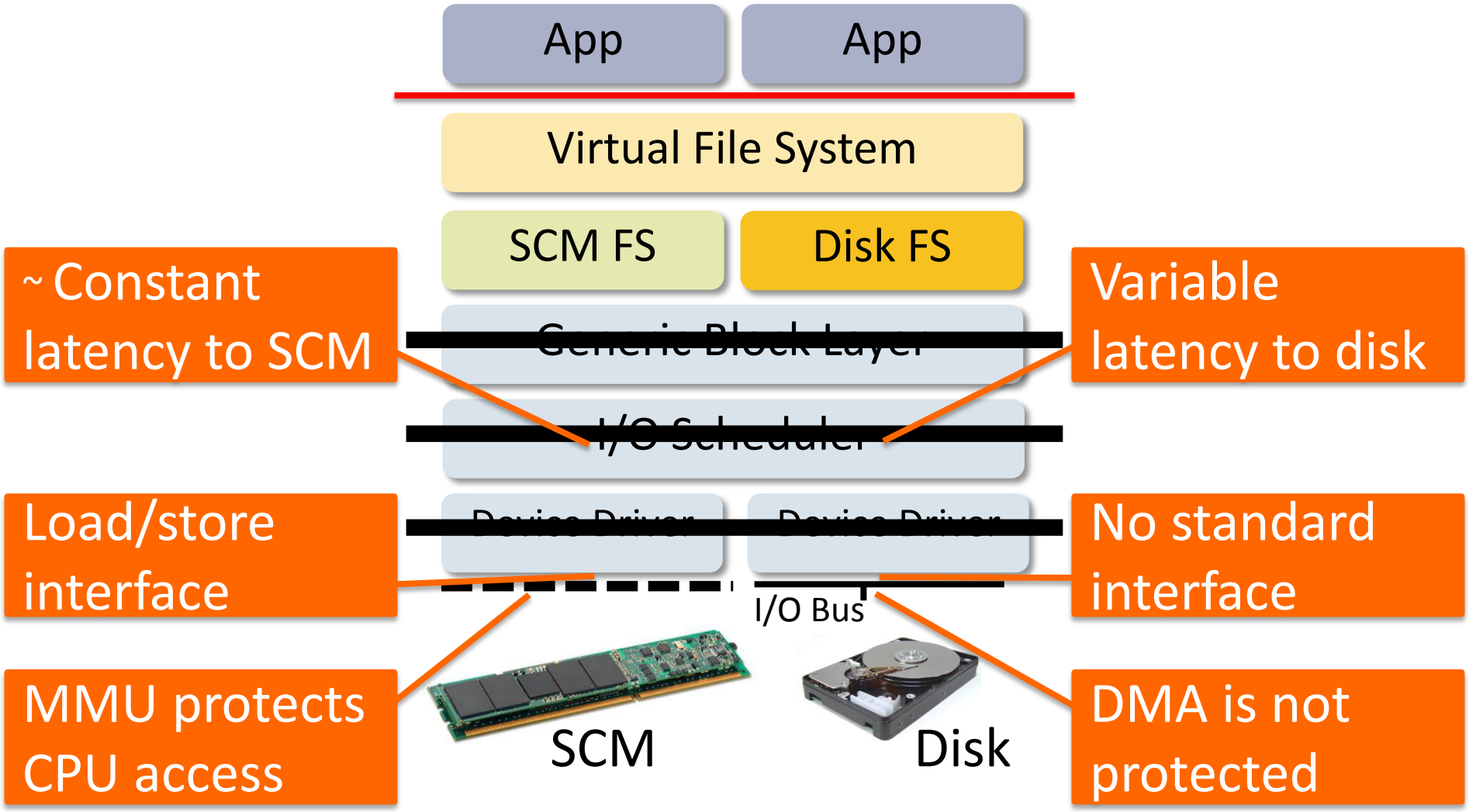
- **Direct user-mode** access for fast access to data
  - Moneta-D, PMFS, Quill, NV-Heaps, Mnemosyne

+

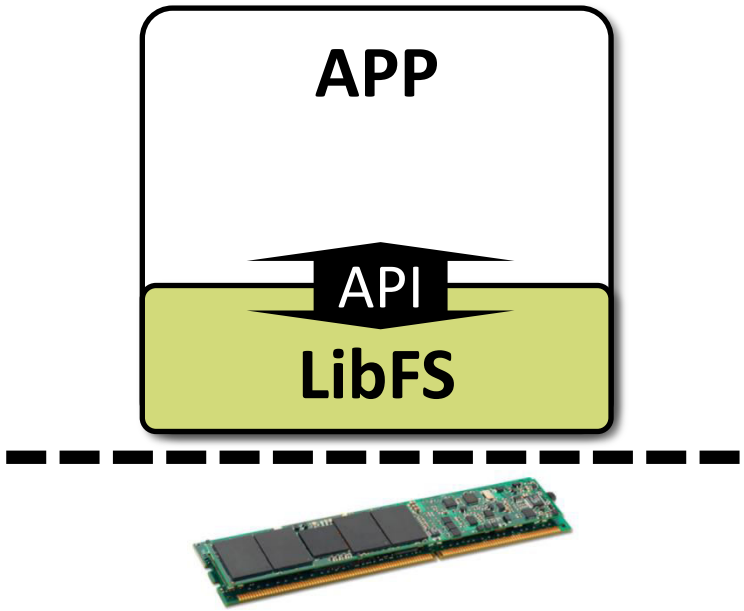
- File system for sharing
  - Shared namespace
  - Protection
  - Integrity



Does SCM need a kernel FS?



Library file systems



[Exokernel (MIT),
Nemesis (Cambridge)]

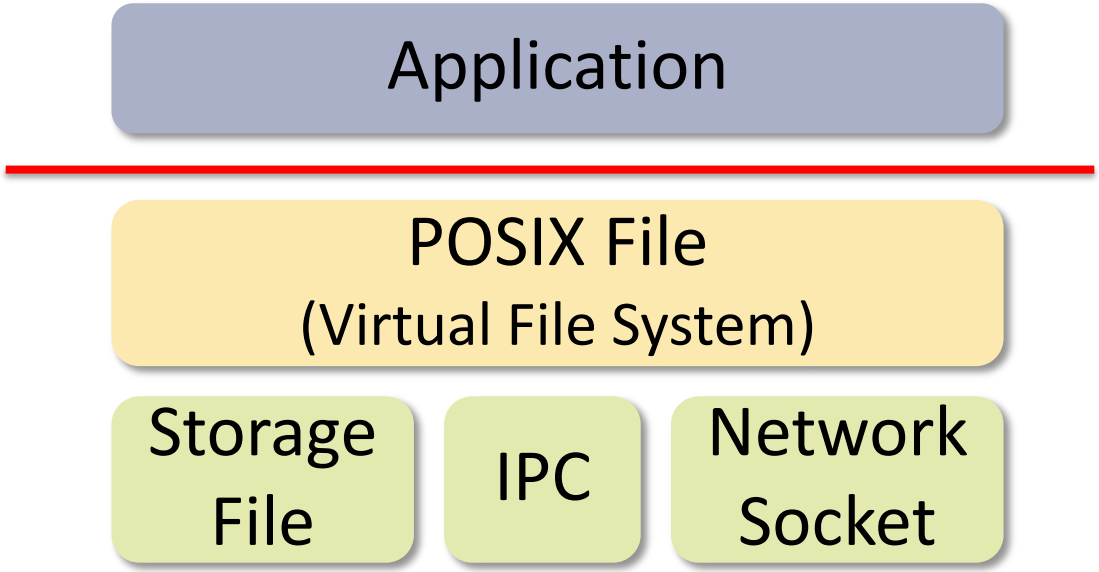
- Enable implementation **flexibility**
  - Optimize file-system interface semantics
  - Optimize operations regarding metadata

Outline

- Overview
- **Motivation: Interface flexibility**
- Aerie: In-memory library file systems
- Evaluation

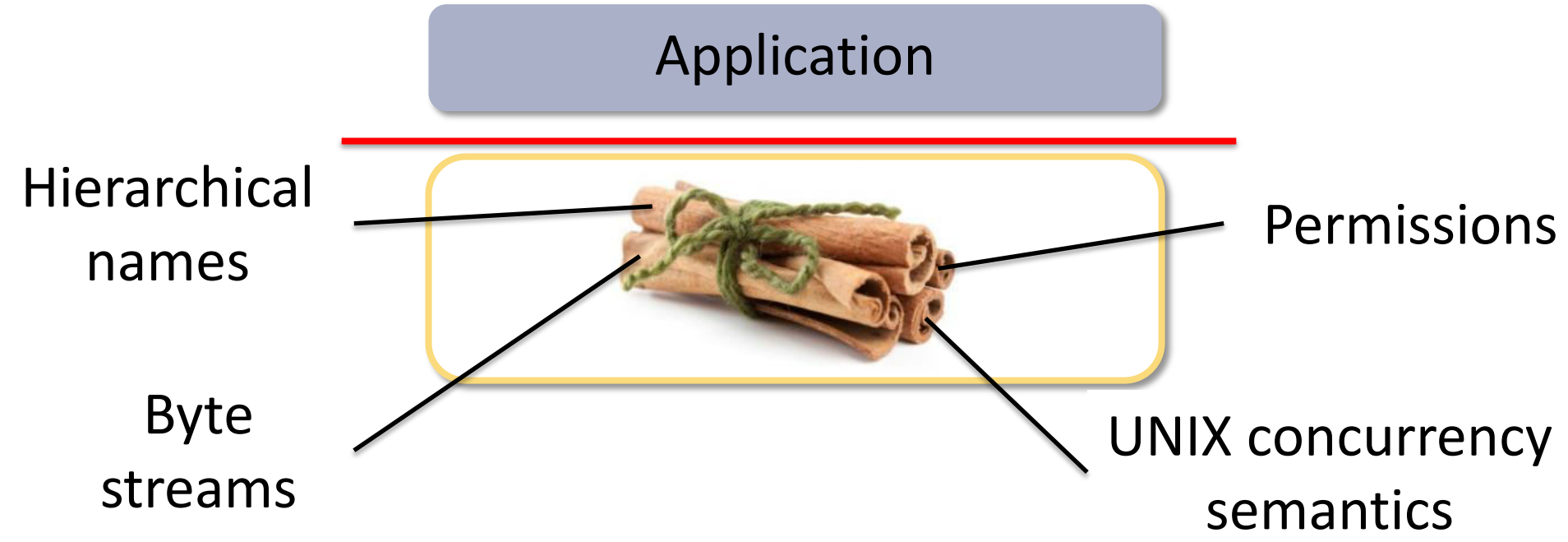
POSIX File: Expensive abstraction

- Universal abstraction: *Everything is a file*
  - Has **generic-overhead** cost



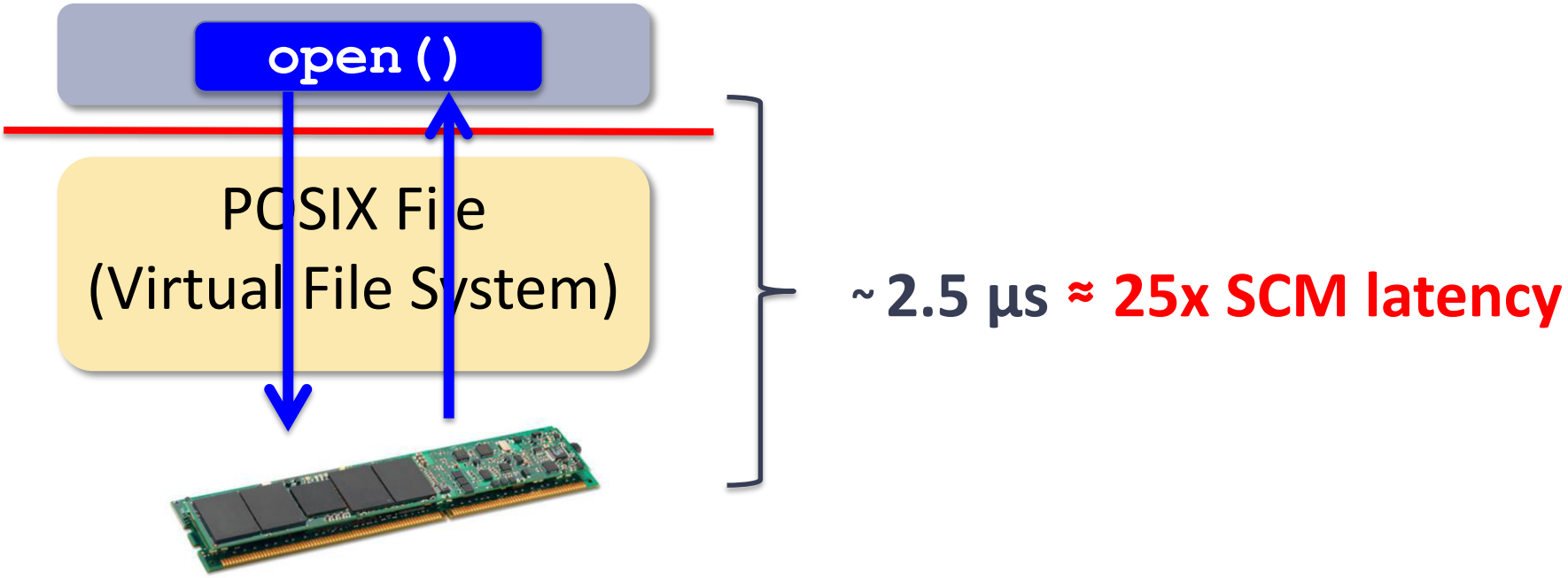
POSIX File: Expensive abstraction

- **Rigid interface** and policies
  - Has fixed components and costs
  - Hinders application-specific customization



POSIX File: Expensive abstraction

- **Generic-overhead** costs
- **Rigid interface** and policies



Customizing the file system today

- **Modify** the kernel
- **Add a layer** over existing kernel file system
- Use a **user-mode framework** such as FUSE

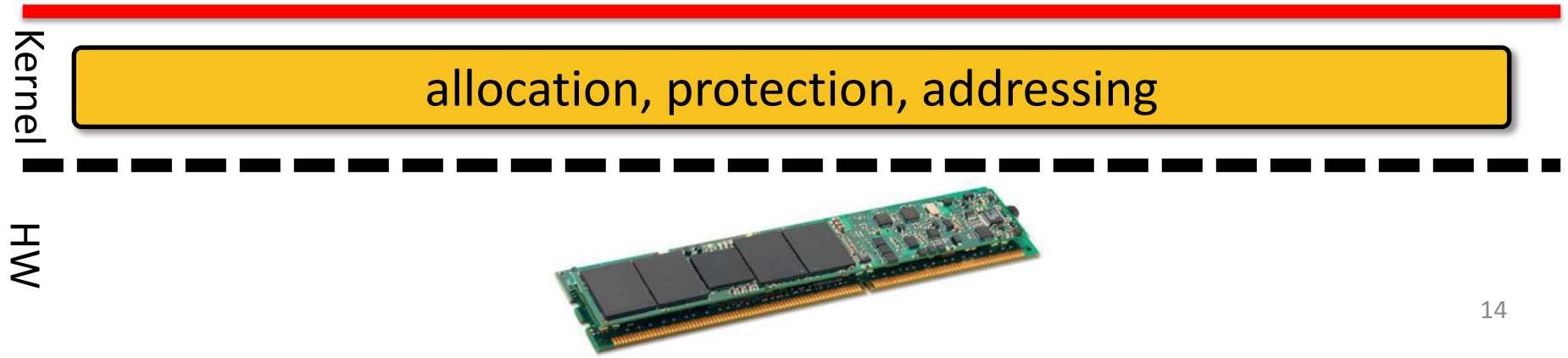
Need flexible interfaces

Outline

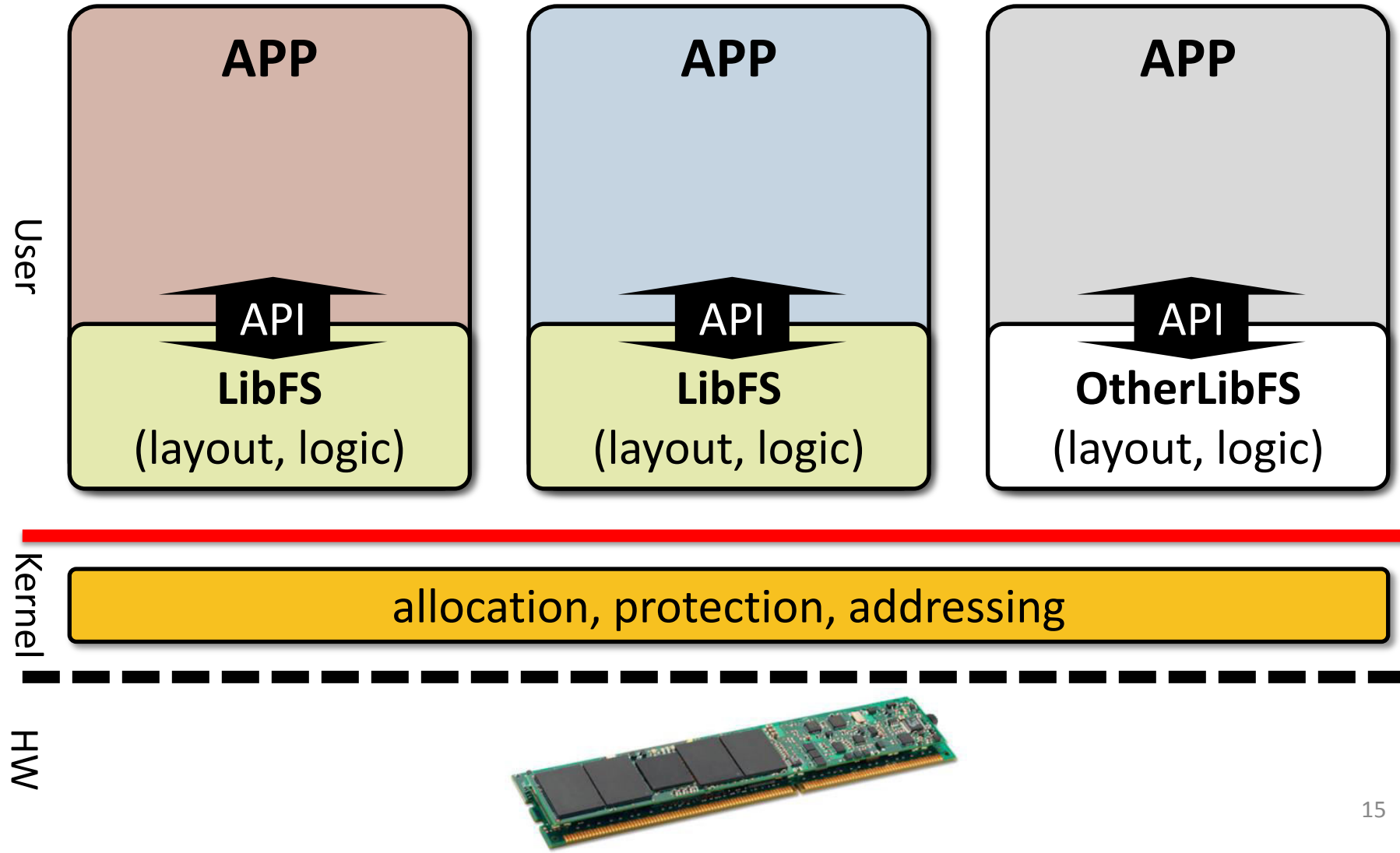
- Overview
- Motivation: Interface flexibility
- **Aerie: In-memory library file systems (libFS)**
- Evaluation

Kernel safely multiplexes SCM

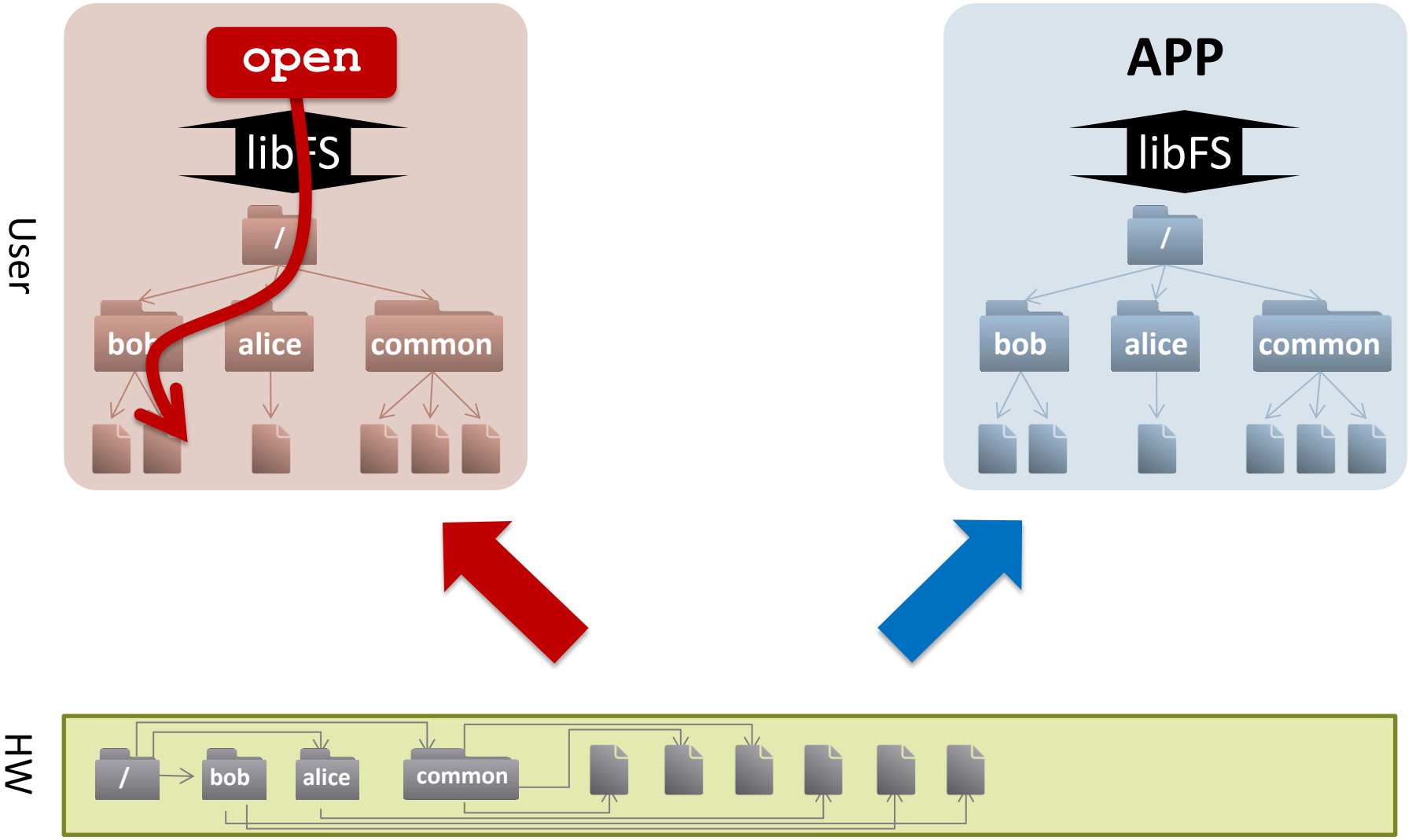
- **Allocation**: Allocates SCM regions
- **Addressing**: Memory-maps SCM regions
- **Protection**: Keeps track of region access rights



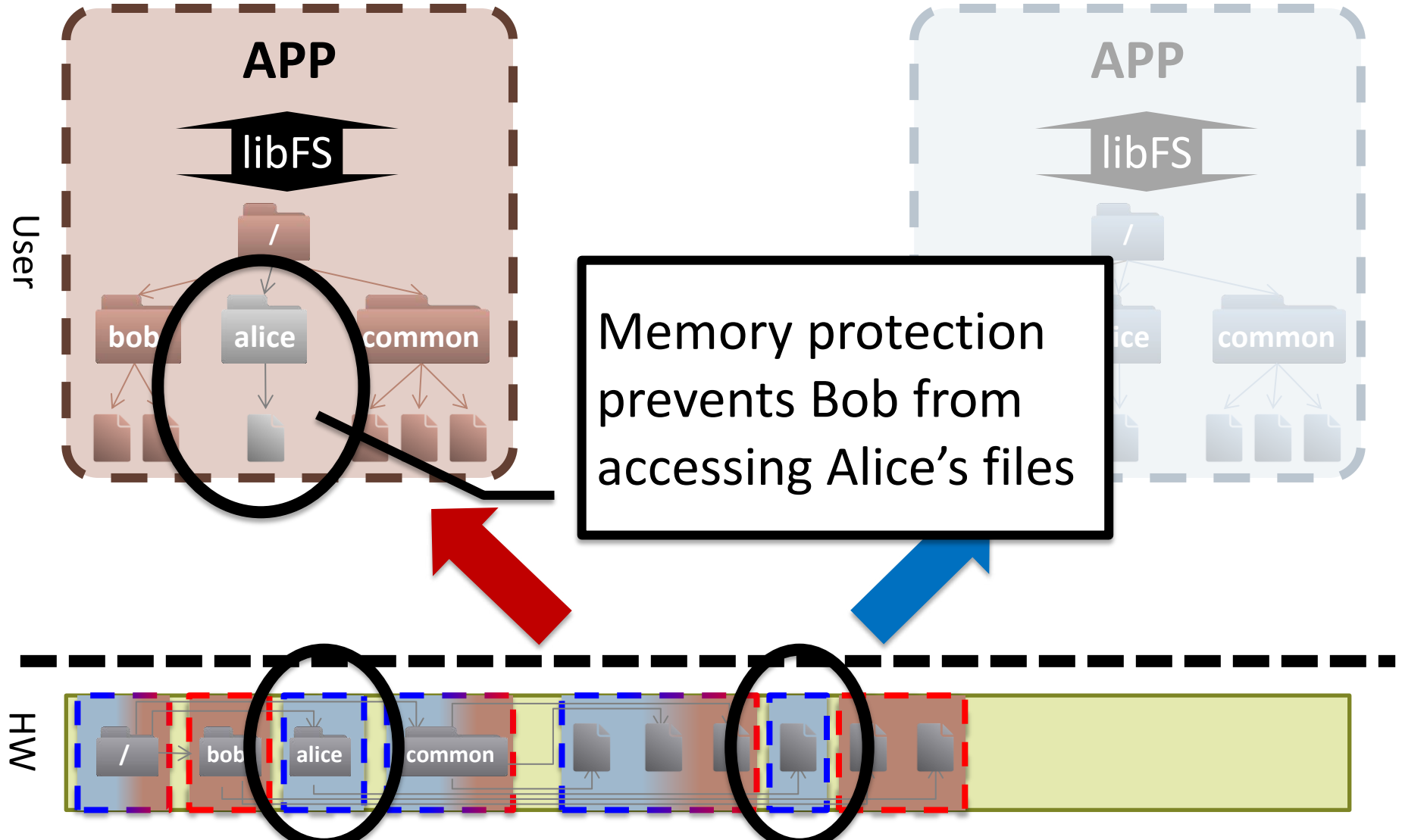
Library implements functionality



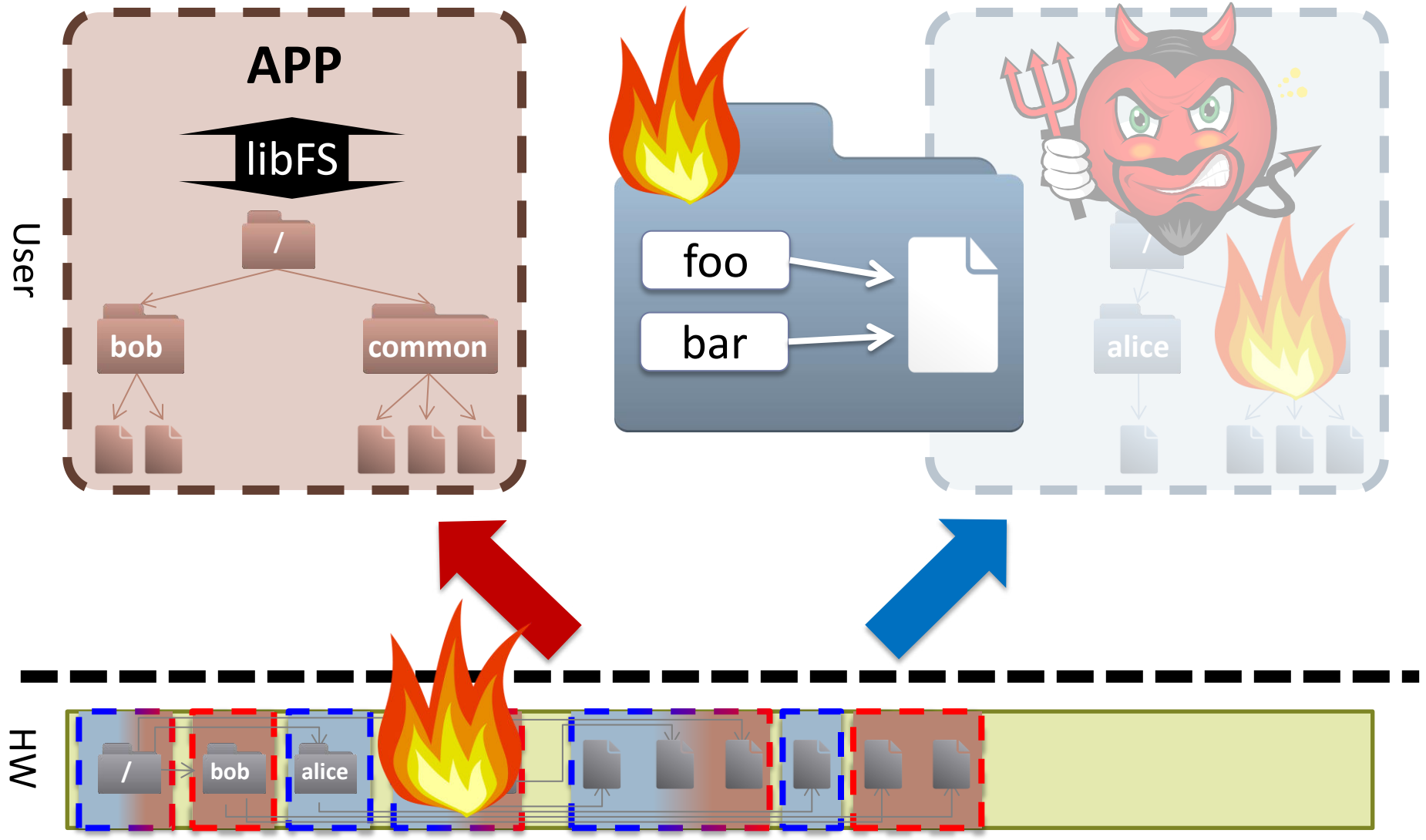
Shared namespace



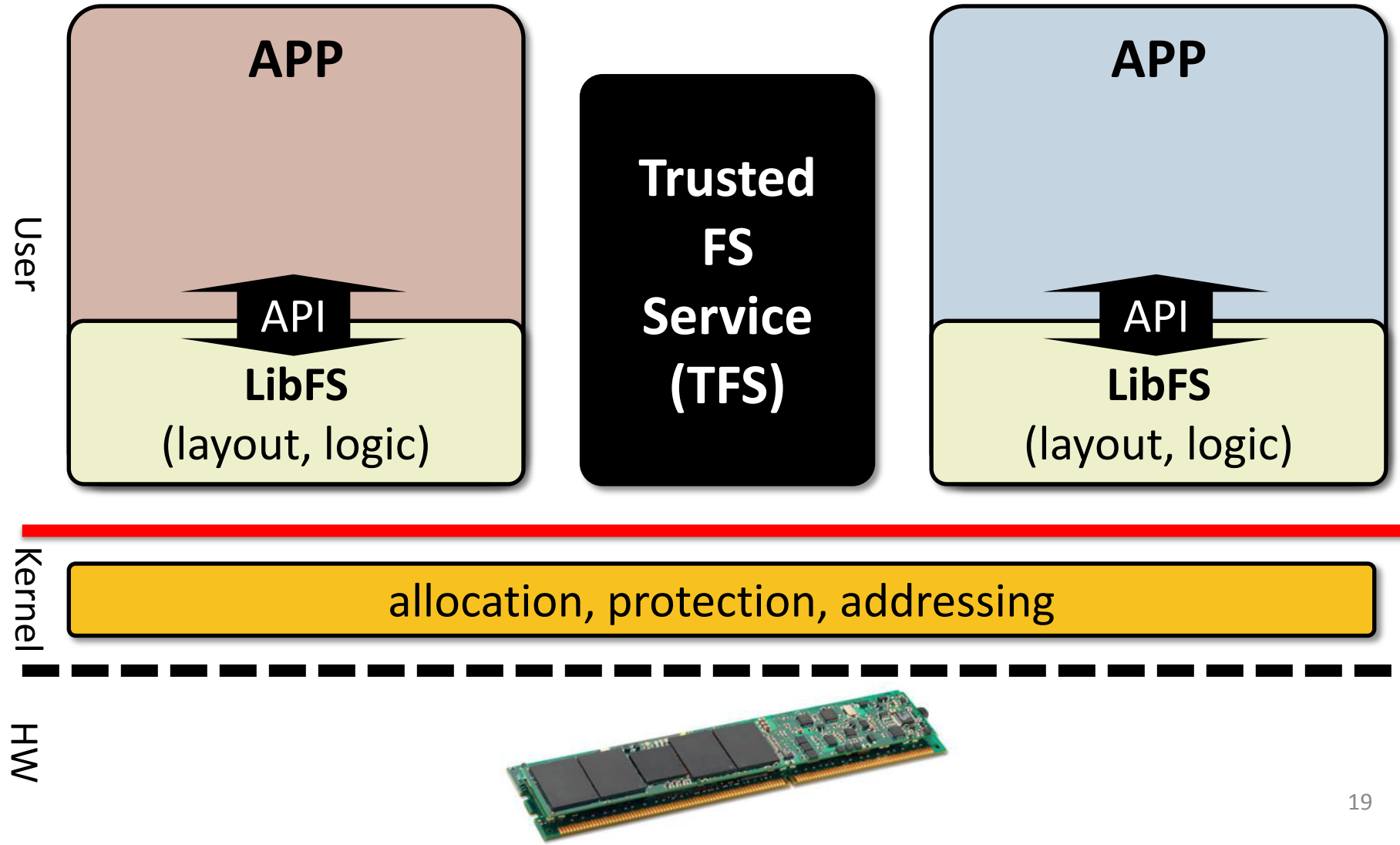
Hardware-enforced permissions



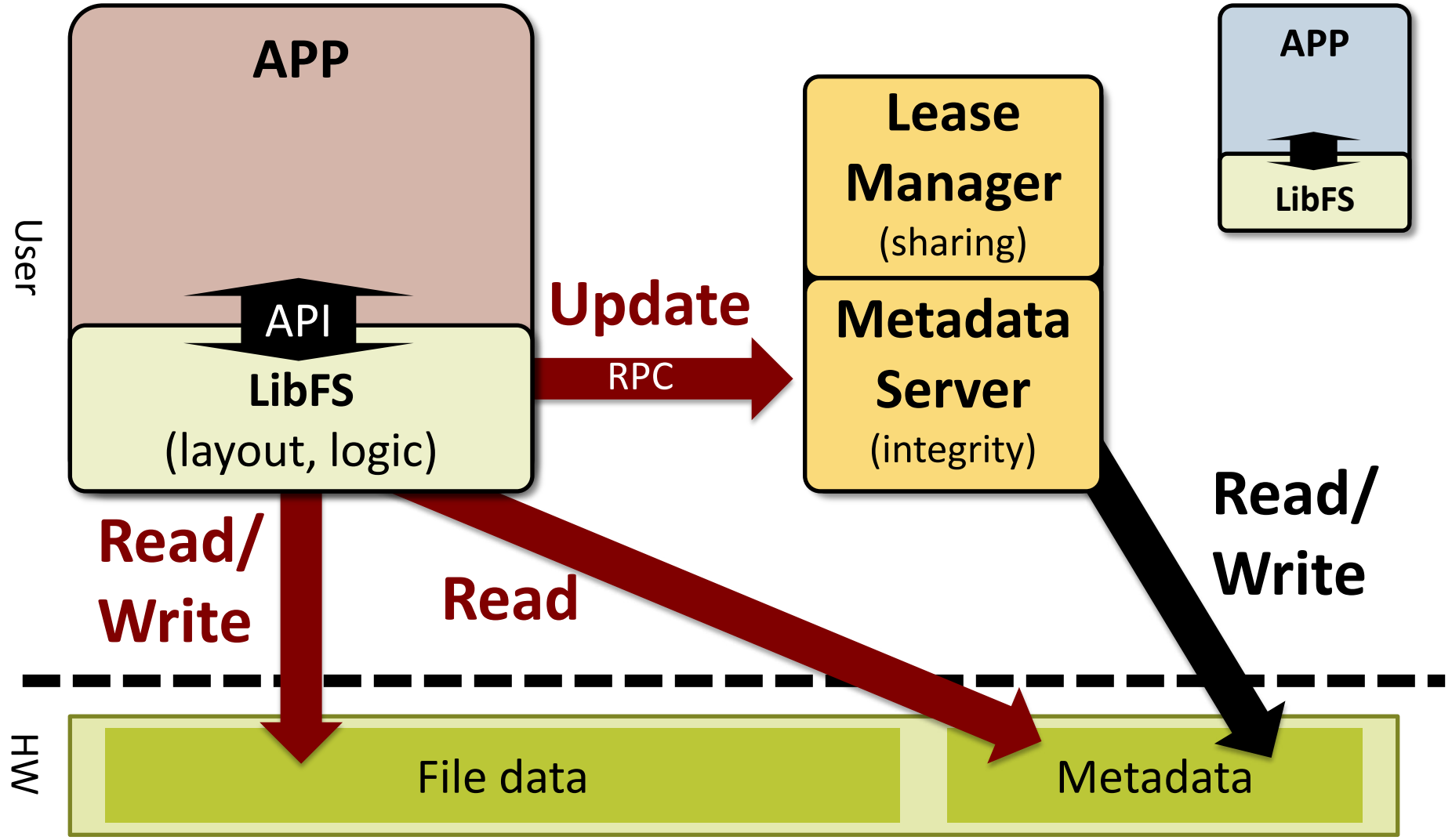
Hardware protection cannot guarantee integrity



Integrity via Trusted File Service



Decentralized architecture



Outline

- Overview
- Motivation: Interface flexibility
- Aerie: In-memory library file systems (libFS)
- **Evaluation**

File Systems

Functionality: PXFS

- POSIX interface: **open/read/write/unlink**
- Hierarchical namespace
- POSIX concurrency semantics
- File byte streams

File Systems

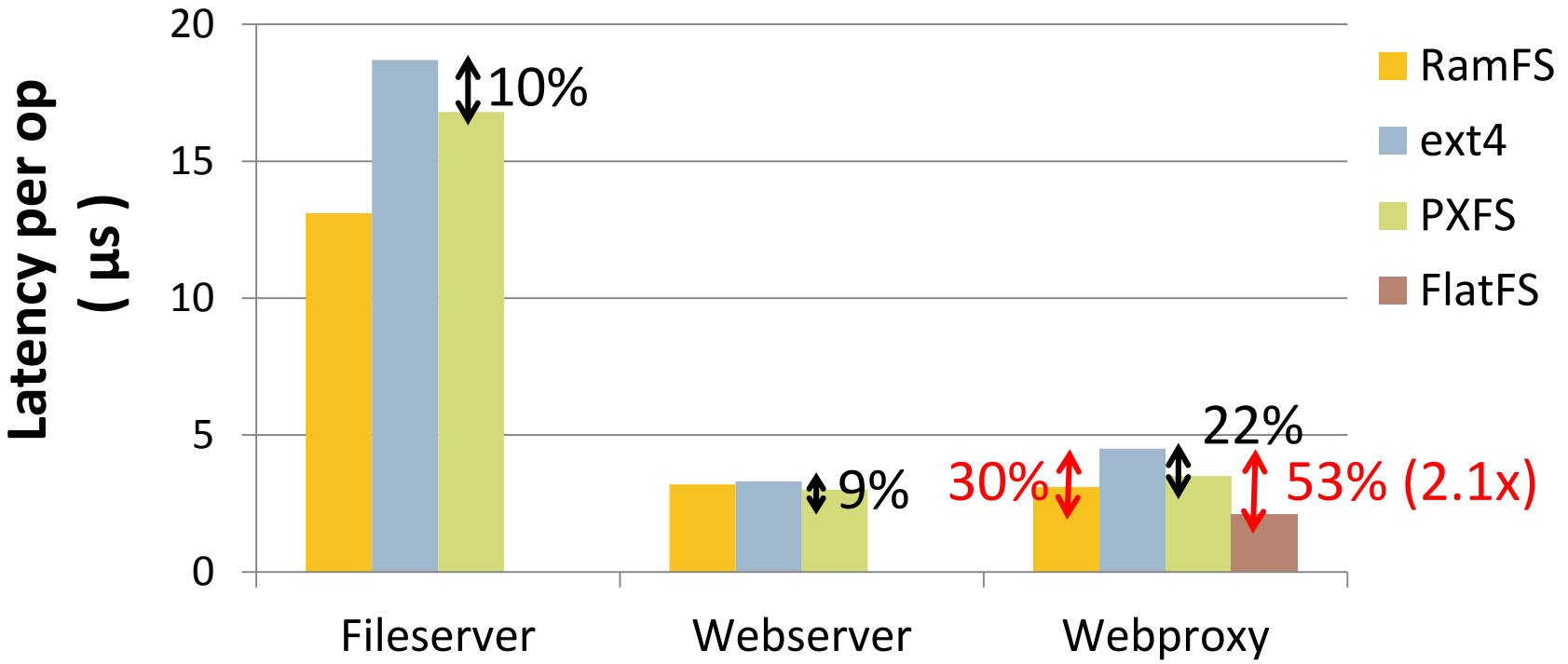
Functionality: PXFS

- POSIX interface: **open/read/write/unlink**
- Hierarchical namespace
- POSIX concurrency semantics
- File byte streams

Optimization: FlatFS

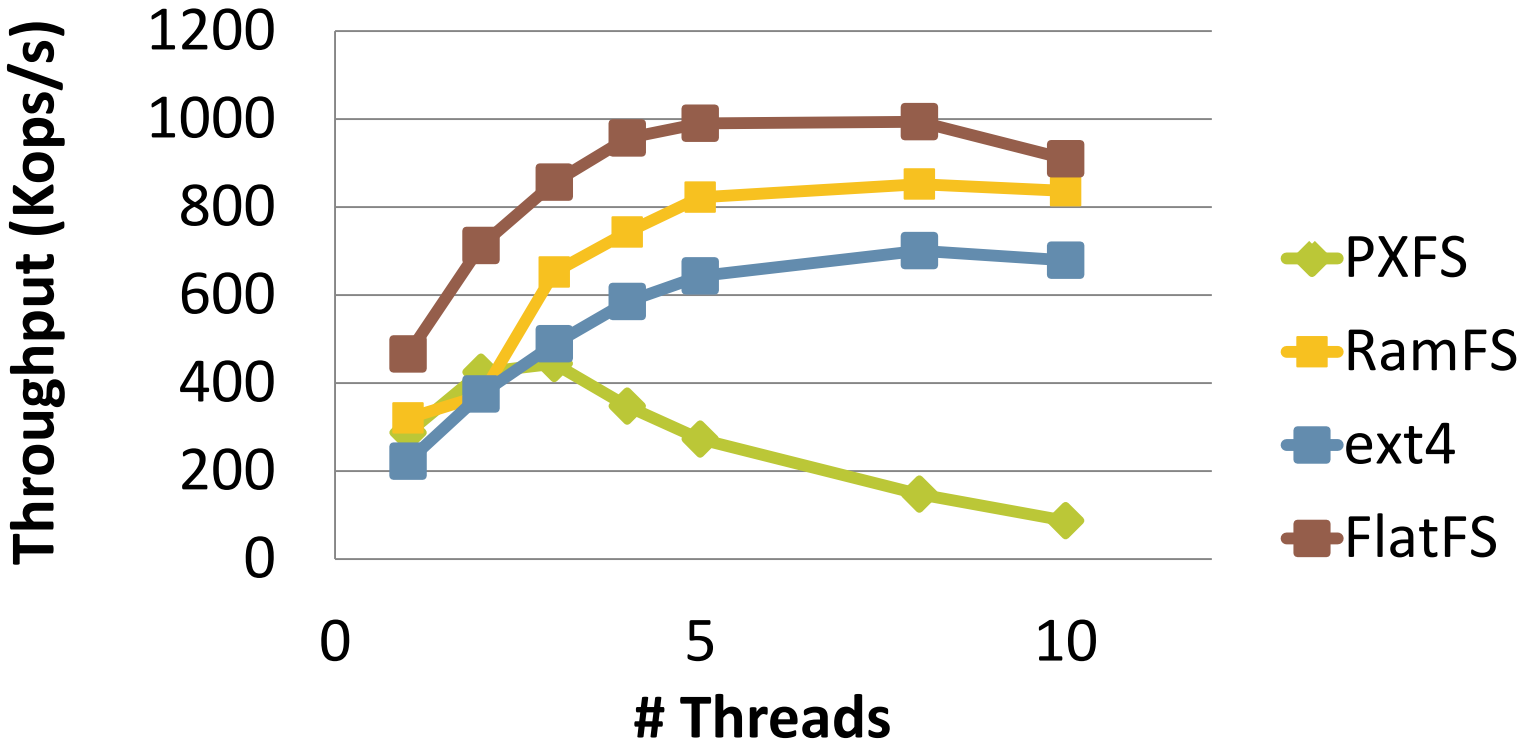
- Key-value interface: **put/get/erase**
- Flat namespace
- KV-store concurrency semantics
- Short, immutable files

Application-workload performance



- PXFS performs better than kernel-mode FS
- FlatFS exploits app semantics to improve performance

Scalability: Webproxy



- FlatFS retains its benefits over kernel-mode file systems

Conclusion

- Software interface overheads handicap fast SCM
- Flexible interface is a must for fast SCM
- Aerie: Library file systems help remove generic overheads for higher performance
  - FlatFS improves performance by up to 110%

Thank you! Questions?