# Silver: A Scalable, Distributed Multi-versioning, Always Growing (Ag) File System

**Michael Wei *^**, Amy Tai #^, Chris Rossbach^,
Ittai Abraham^, Udi Wieder^, Steven Swanson*, Dahlia Malkhi^

^ VMware Research

* University of California, San Diego # Princeton University

# Storage Needs Over The Years

- Early FS: Static Mapping
- Hierarchy
- Streaming – Sequential I/O is king
- Crash consistency, Journaling
- Versioning, Snapshotting, Cloning
- Dedupe, Encryption

# Distributing File Systems is hard

- Most file systems are built to span a single device
- Emerging file systems (zfs, btrfs) may span multiple devices but doesn't scale past a single machine
- Distributed file systems scale but suffer from consistency issues
    - Read/write is simple
    - Advanced features like snapshotting, versioning and cloning often require locking, if supported at all

# Redesign a distributed FS from the ground up

- A log is an ideal substrate for a FS
- Employed by many filesystems today, dating from LFS
- What if we had a efficient, distributed log?

# Distributed Log

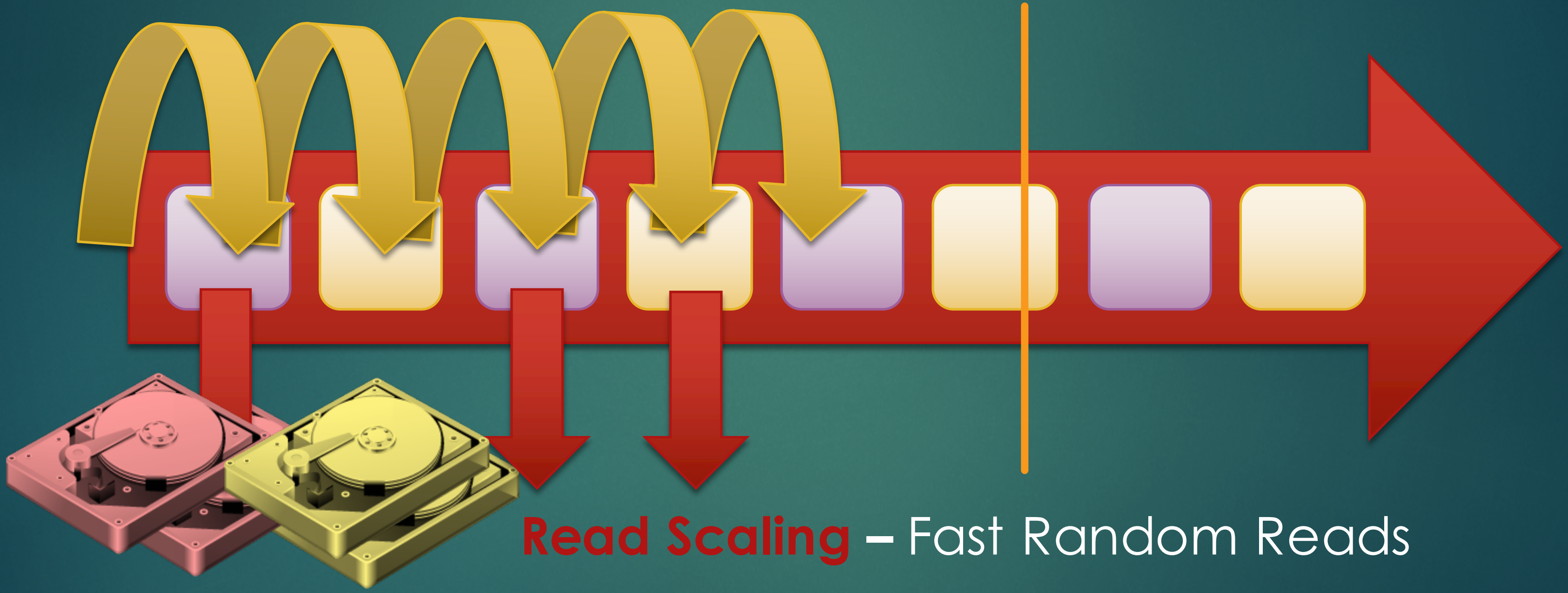
- Silver leverages a fault-tolerant, replicated distributed log
- Previously described in Corfu [NSDI'12], Tango [SOSP'13]
- Augmented with Replex [1]

- **[1] Replex: A Scalable, Highly Available Multi-Index Data Store**
  Amy Tai, Michael Wei, Michael J. Freedman, Ittai Abraham and Dahlia Malkhi

# A distributed shared log

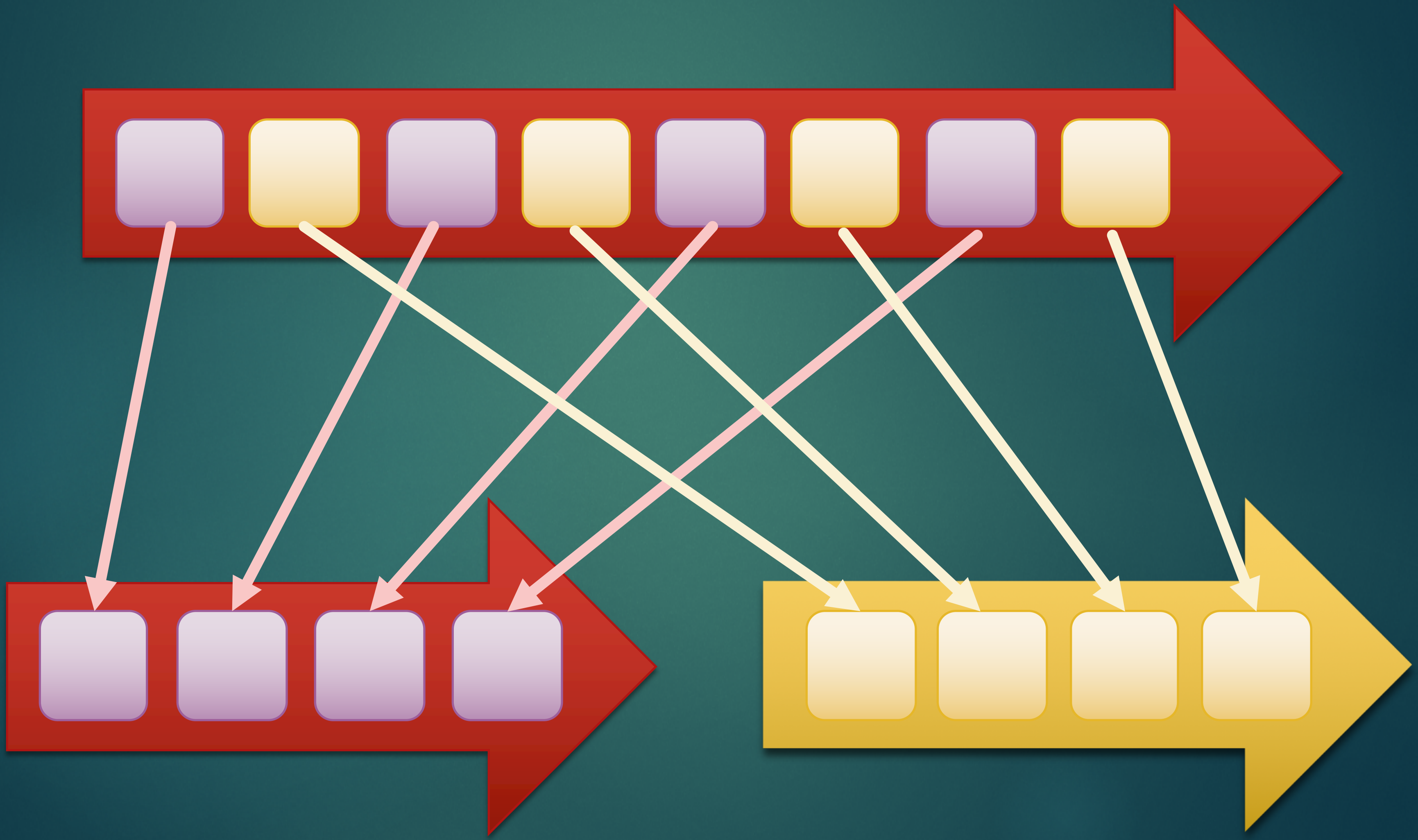
Write Scaling – Fast Updates

Strong Consistency – Snapshots

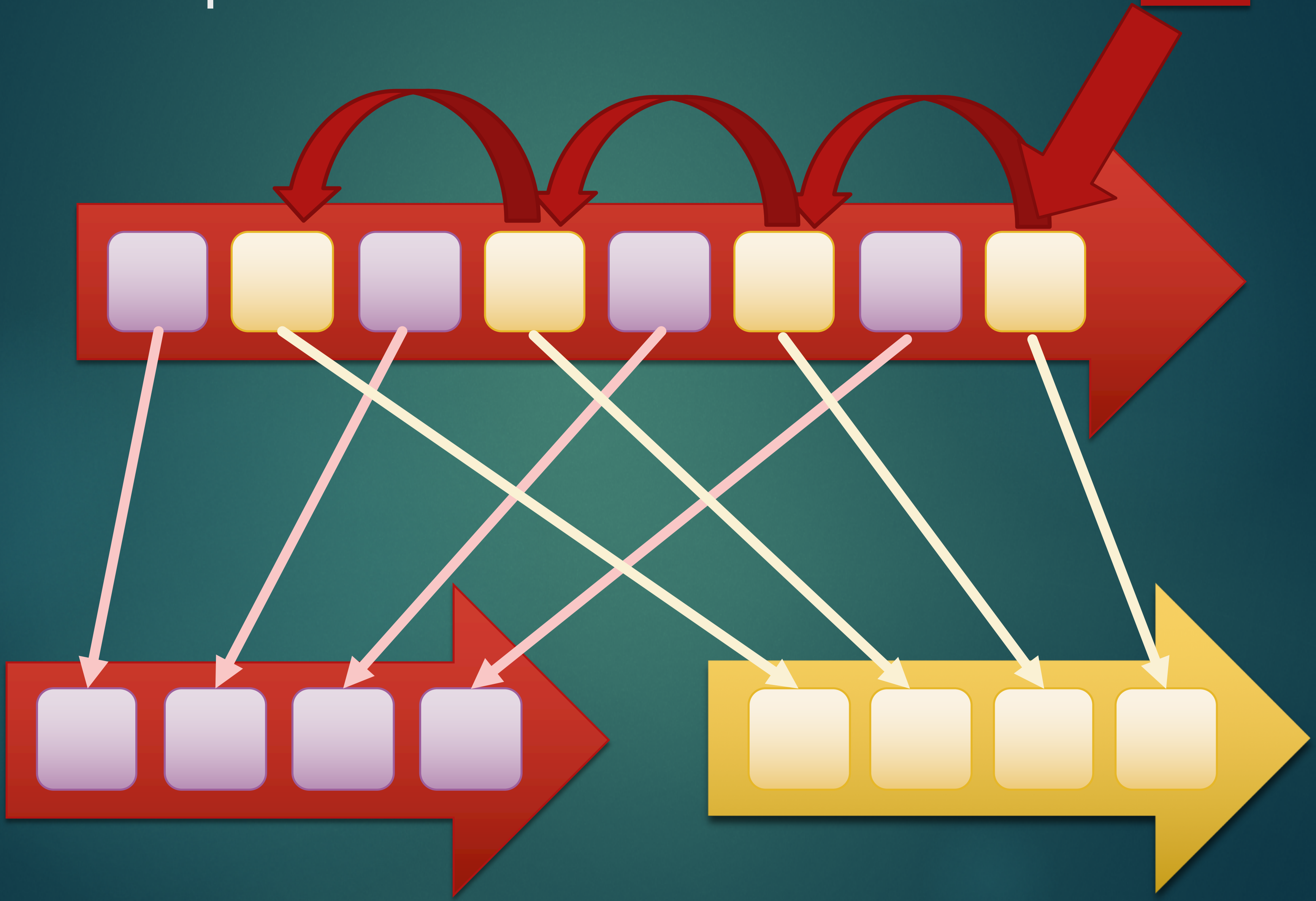


Read Scaling – Fast Random Reads

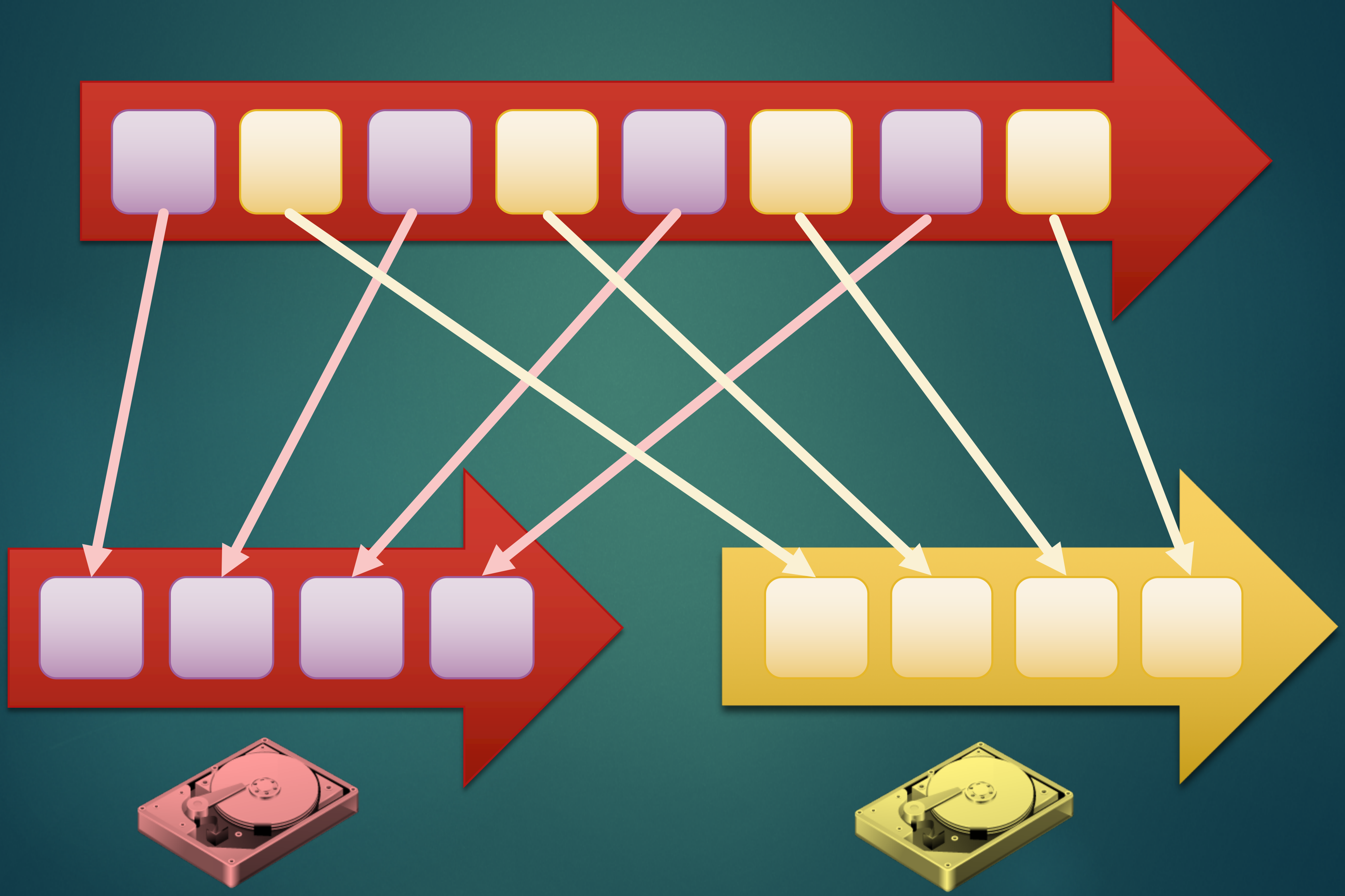
# Streams



# Backpointers



# Replex



# Log Operations

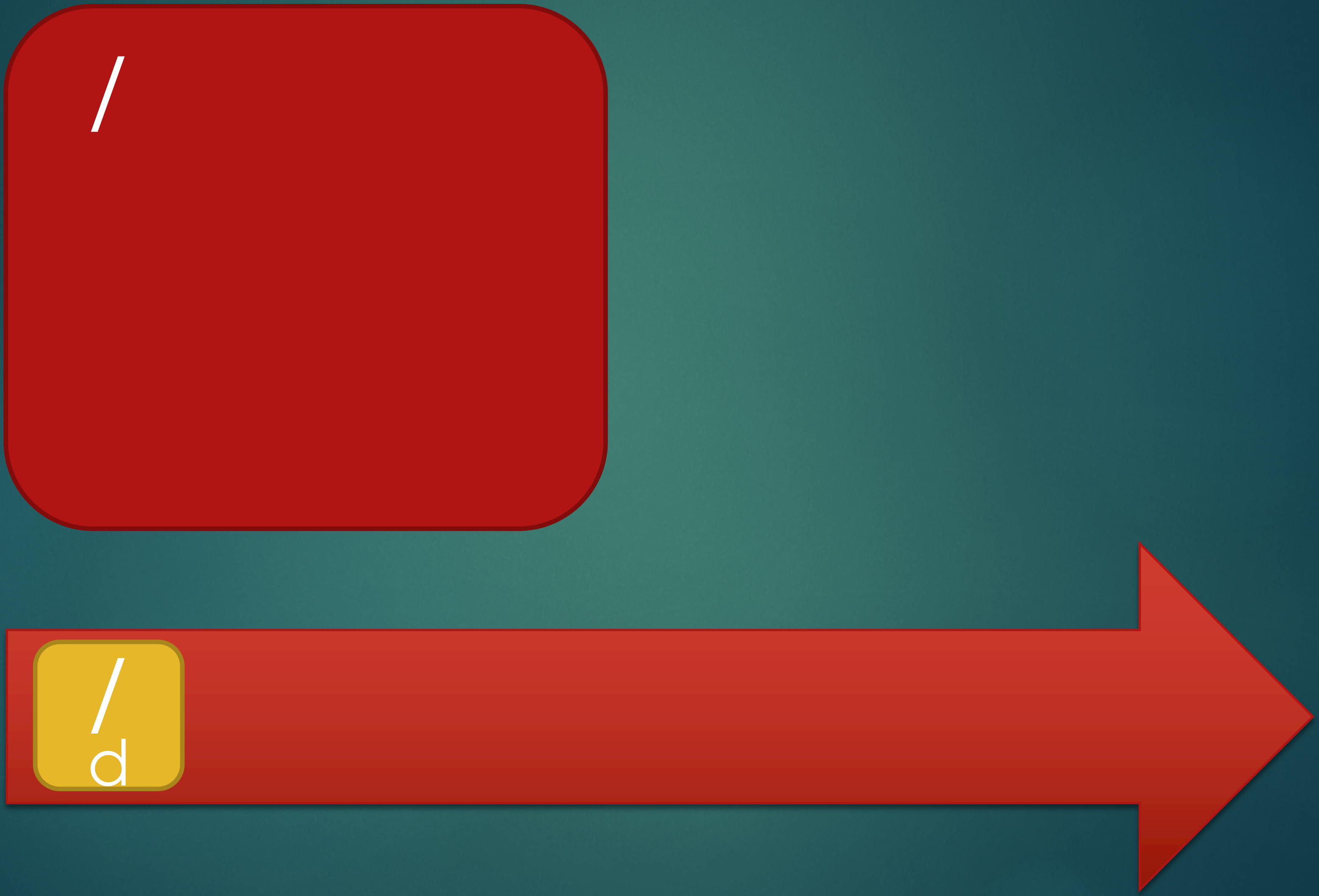
- Reads
  - Random log read given offset
  - Random stream read given offset
  - Bulk read of entire or partial stream
- Writes
  - Append to a particular stream
- Queries
  - Get last address written to a stream
- Trim
  - Releases the space used for an address
- Entries are variably sized

# Silver Architecture

- Composed of streams
  - Metadata streams, represent “files”
  - Data streams, represent file data
  - Directory streams, represent directories

- First stream is a “root” directory stream
- Each stream records deltas, or changes to that stream
  - Every ‘overwrite’ is an append of the delta

# Silver Example

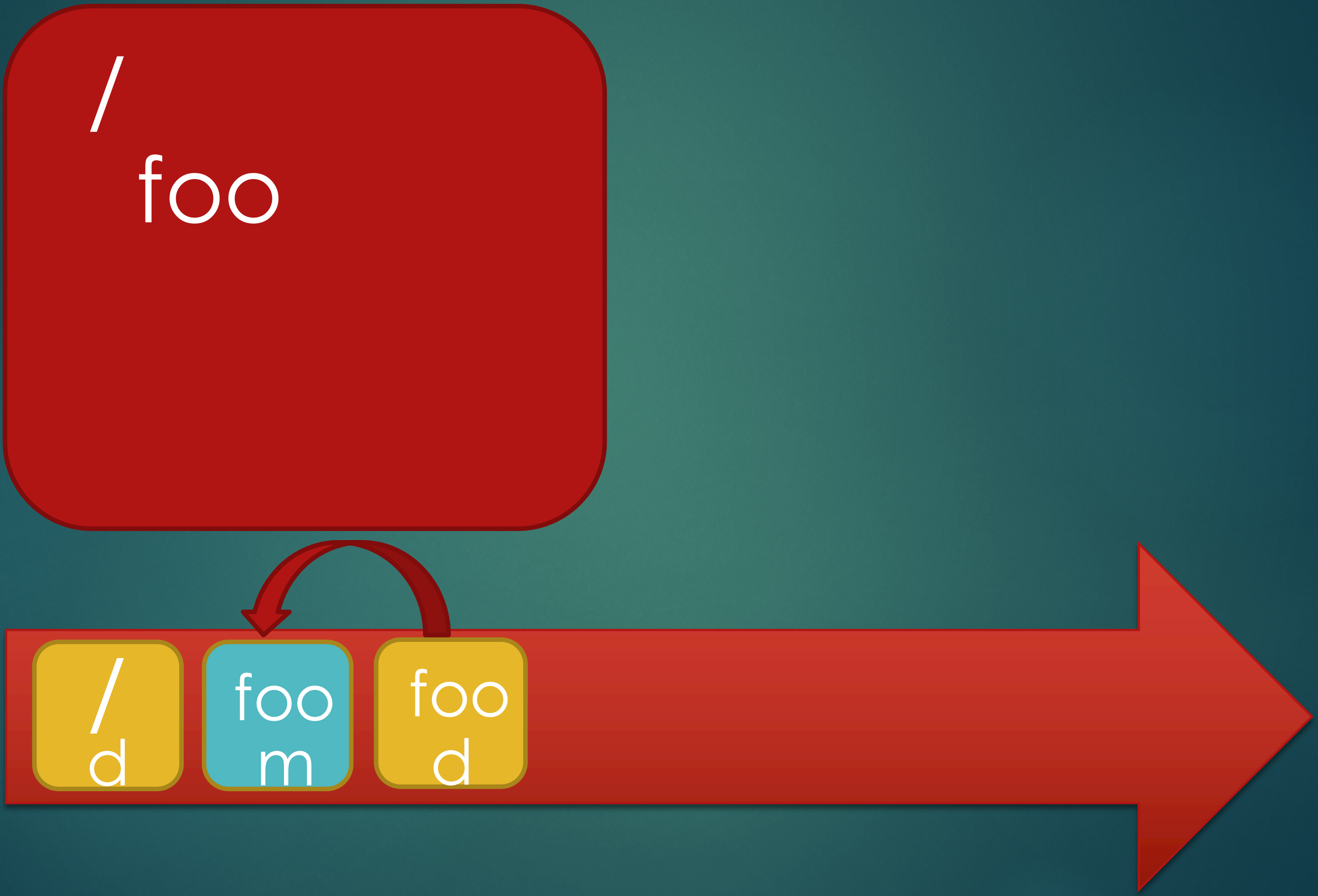


/

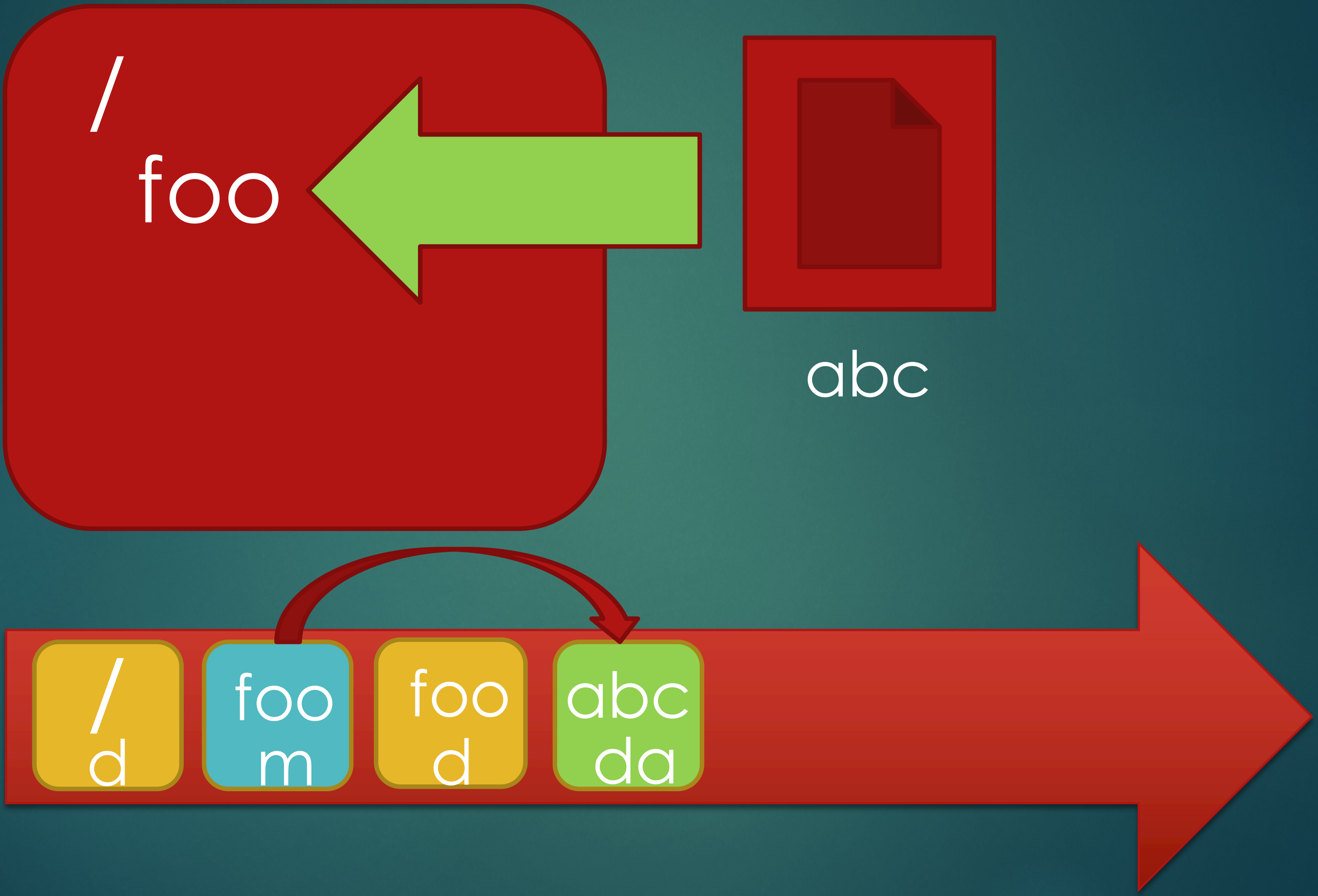
/

d

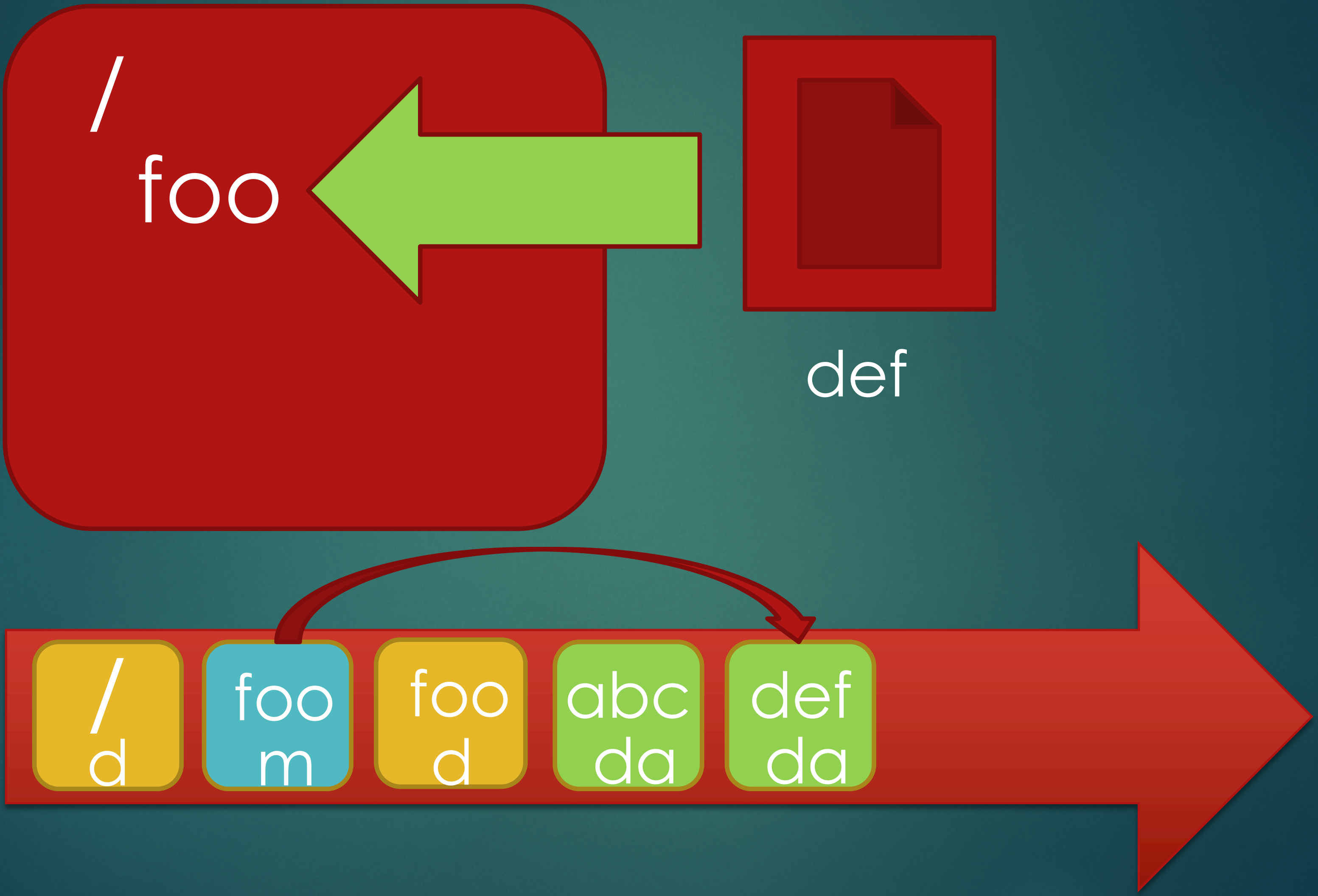
# Silver Example



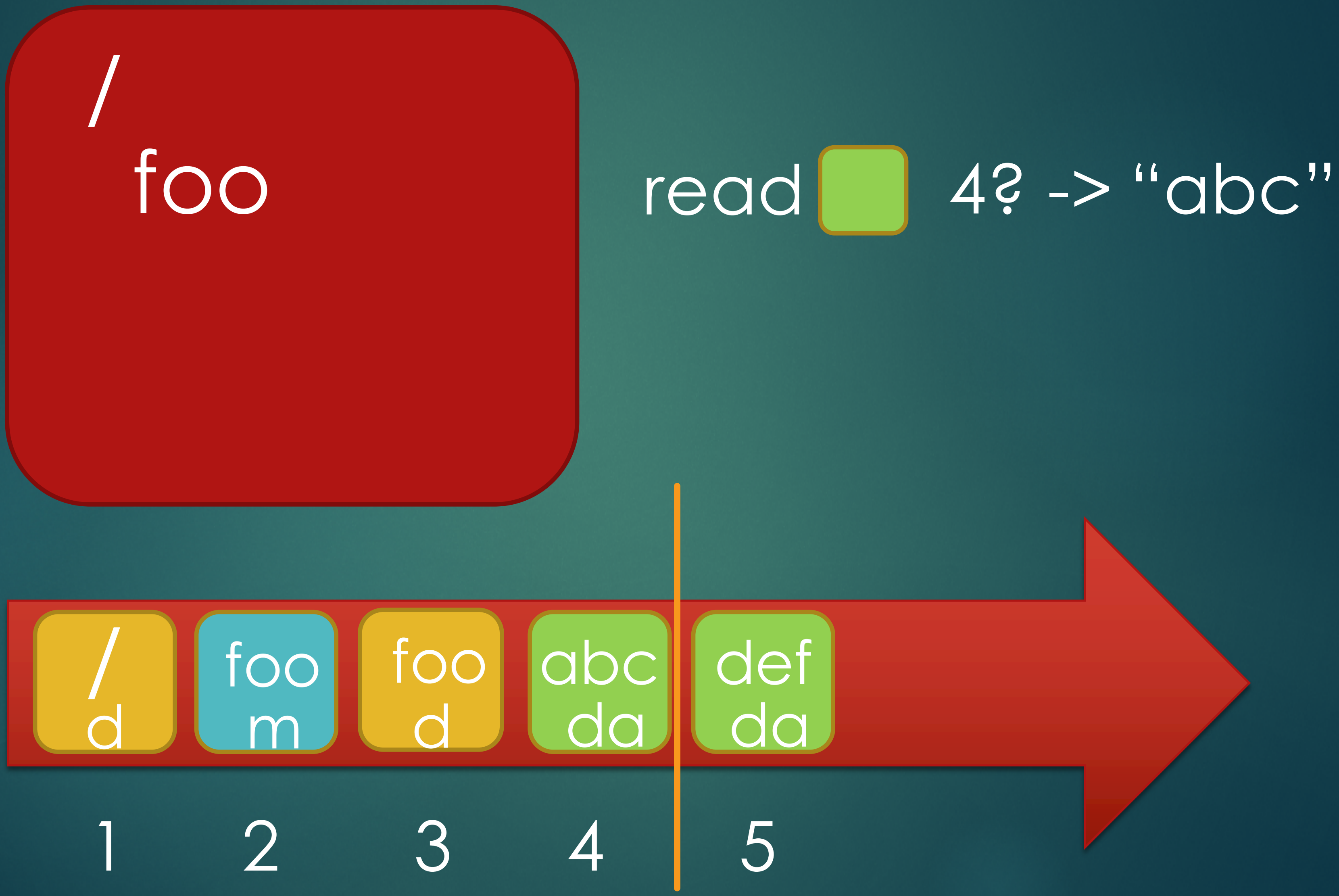
# Silver Example



# Silver Example



# Silver Example



/
foo

read 🟩 4? -> “abc”

/
d

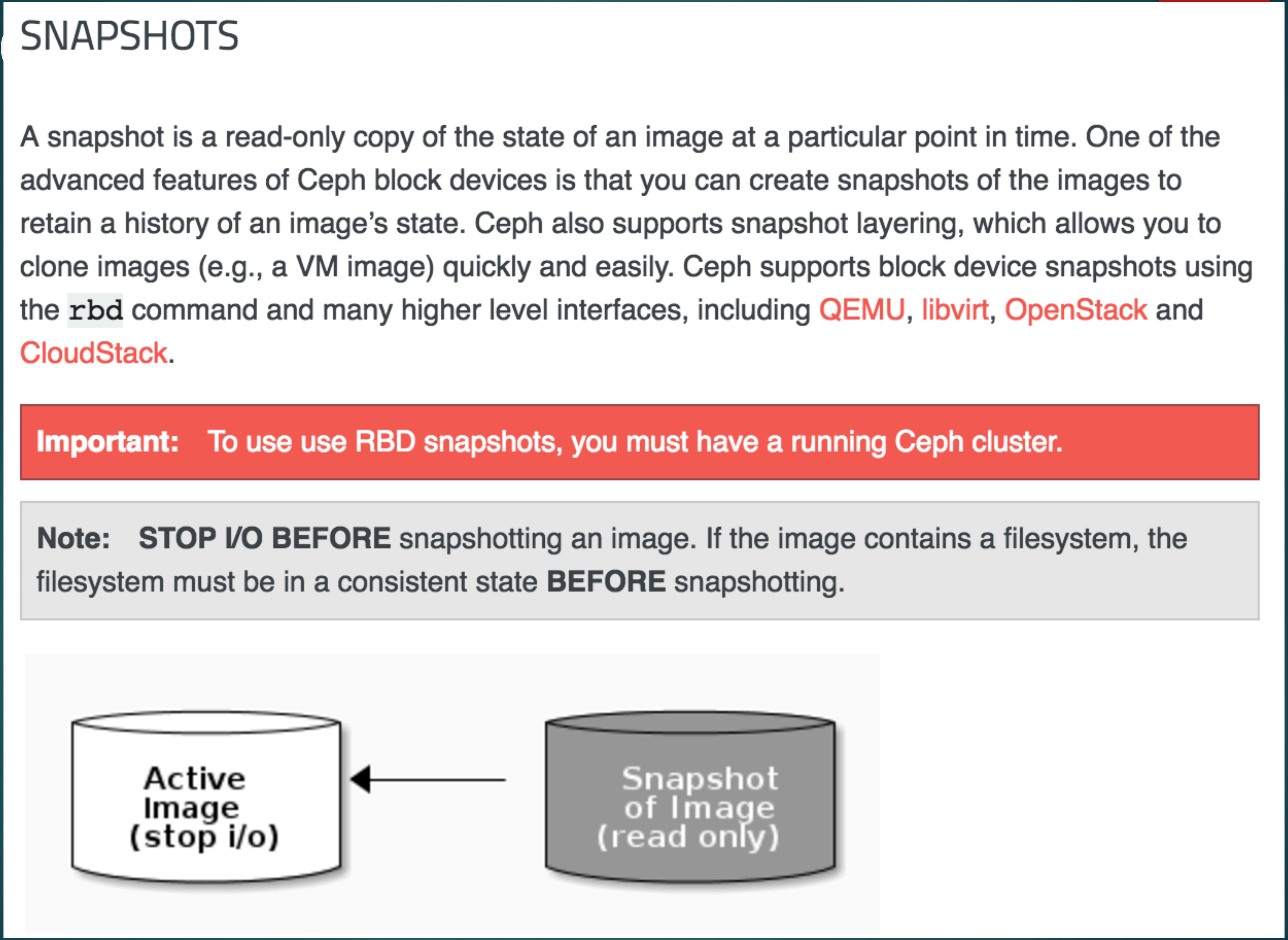
foo
m

foo
d

abc
da

def
da

1 2 3 4 5



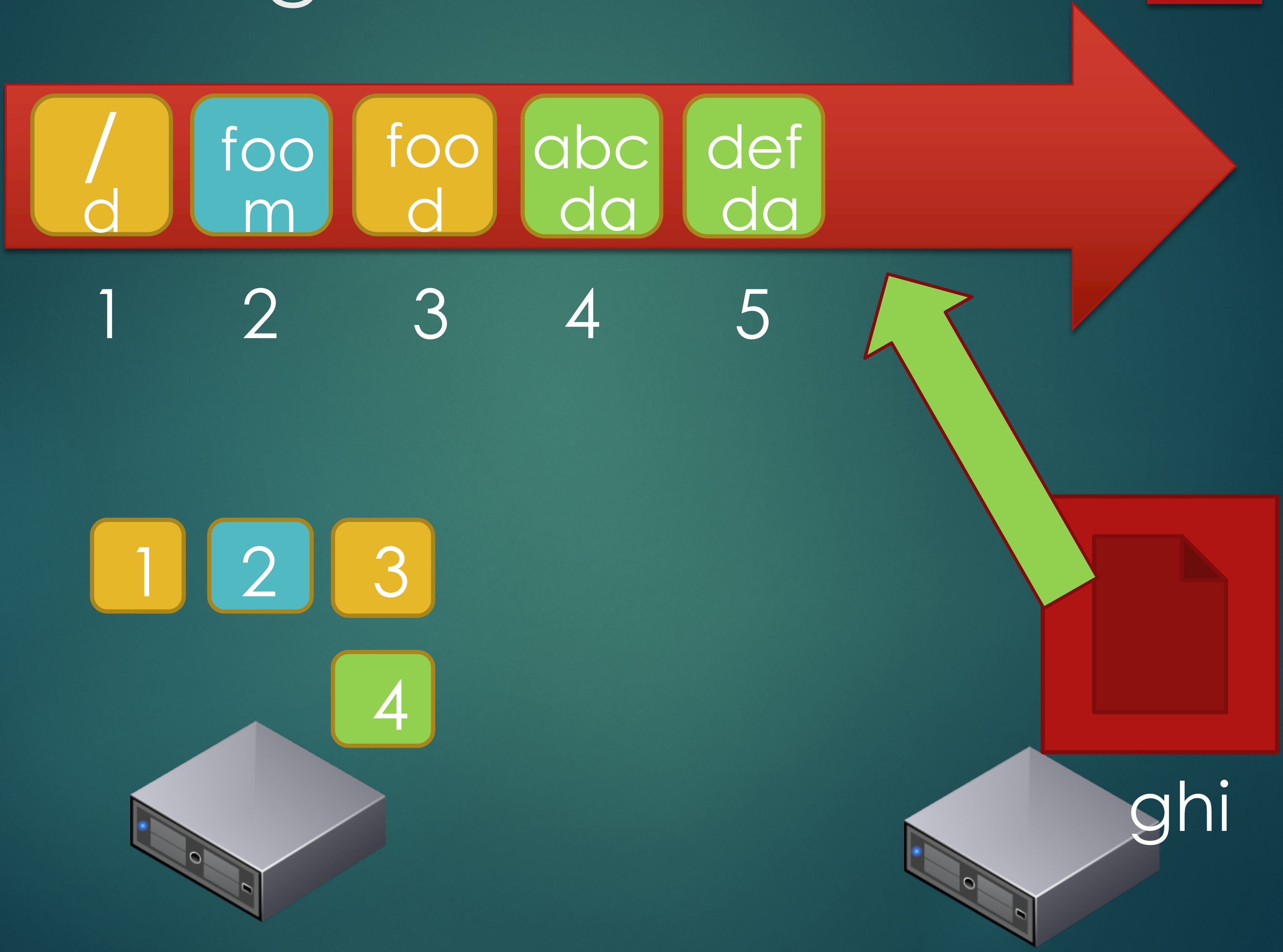
# SNAPSHOTS

A snapshot is a read-only copy of the state of an image at a particular point in time. One of the advanced features of Ceph block devices is that you can create snapshots of the images to retain a history of an image's state. Ceph also supports snapshot layering, which allows you to clone images (e.g., a VM image) quickly and easily. Ceph supports block device snapshots using the `rbd` command and many higher level interfaces, including QEMU, libvirt, OpenStack and CloudStack.

**Important:** To use use RBD snapshots, you must have a running Ceph cluster.

**Note:** **STOP I/O BEFORE** snapshotting an image. If the image contains a filesystem, the filesystem must be in a consistent state **BEFORE** snapshotting.

# Caching



/
d

foo
m

foo
d

abc
da

def
da

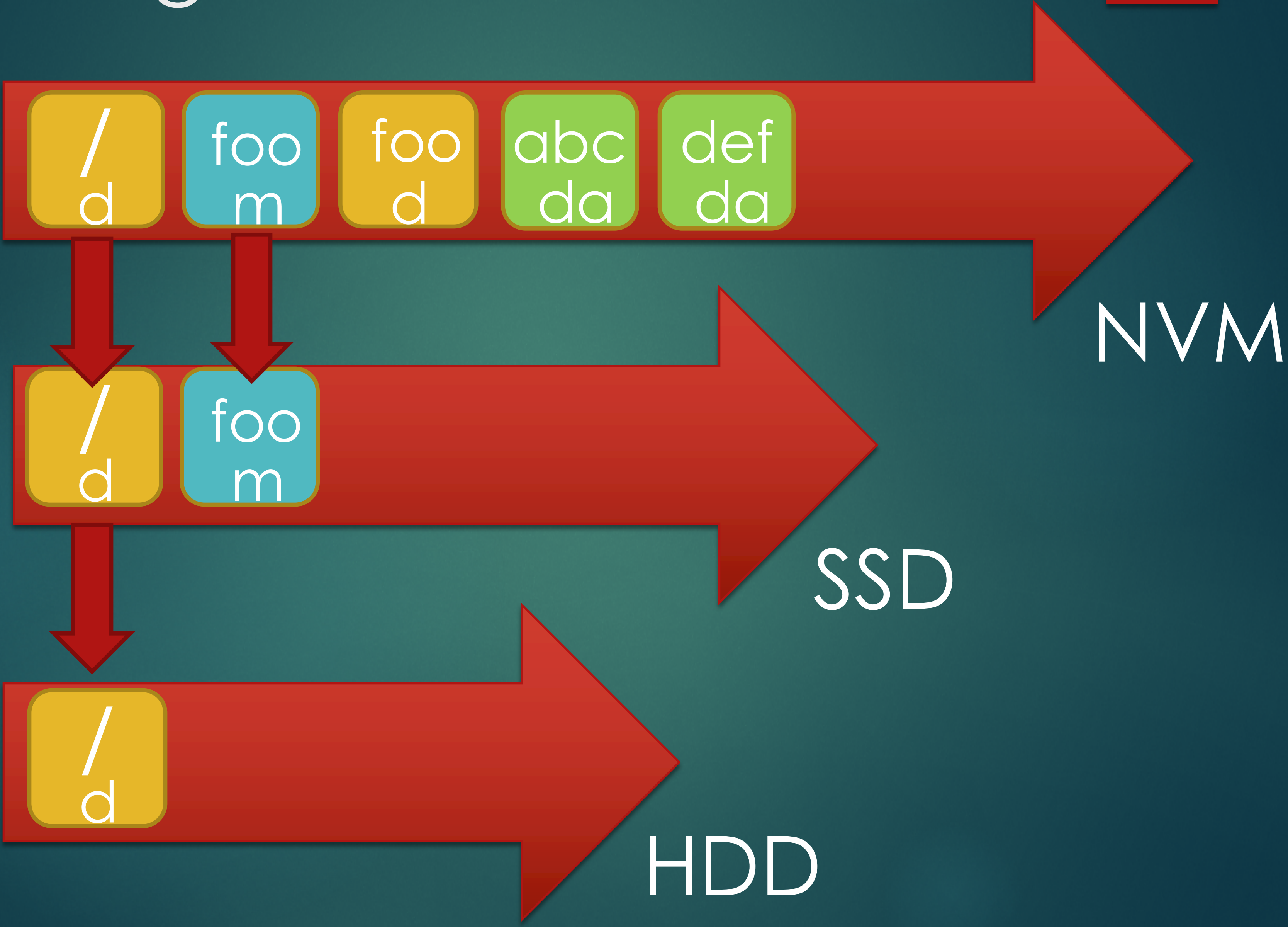
1 2 3 4 5

1 2 3

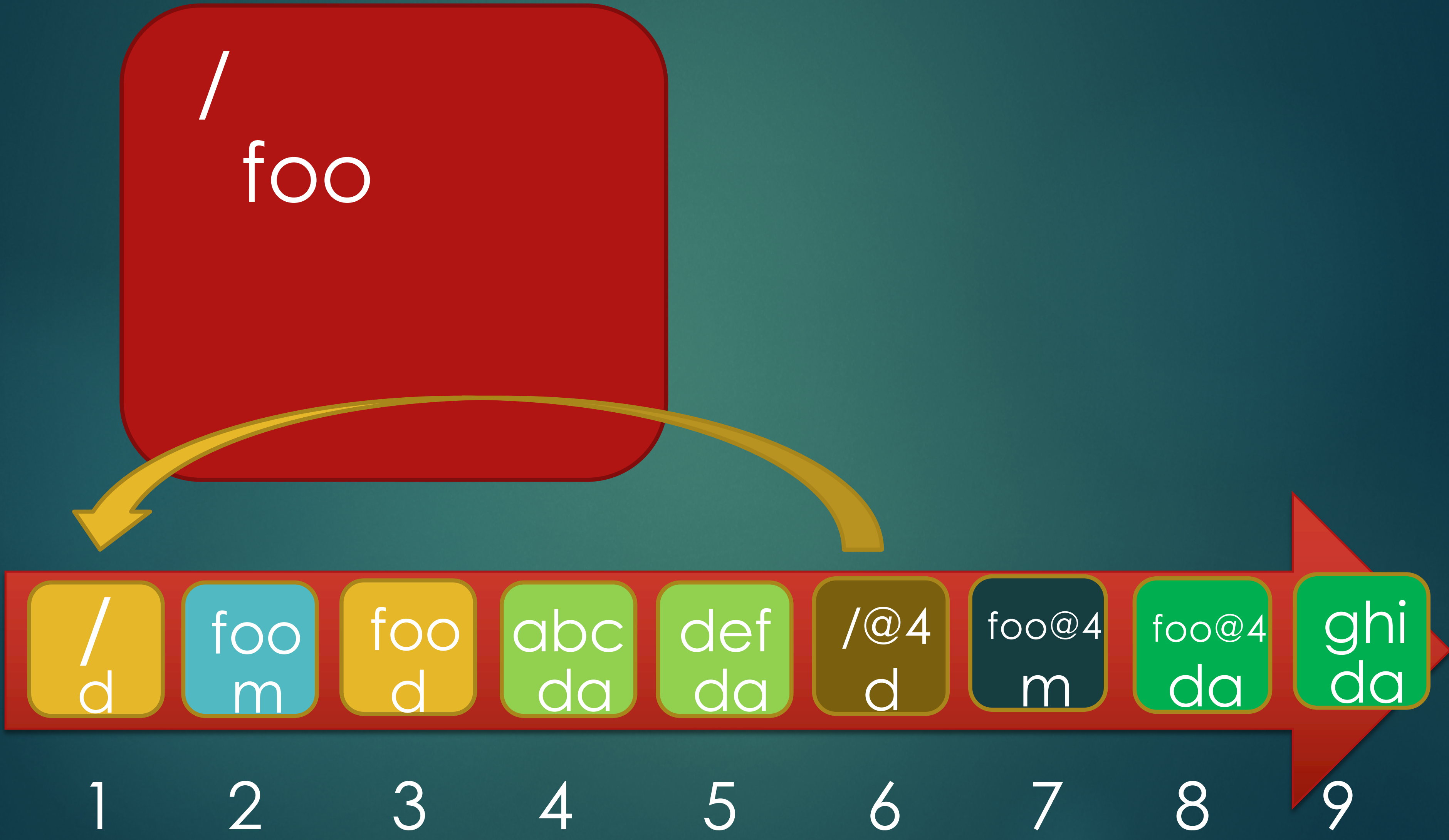
4

ghi

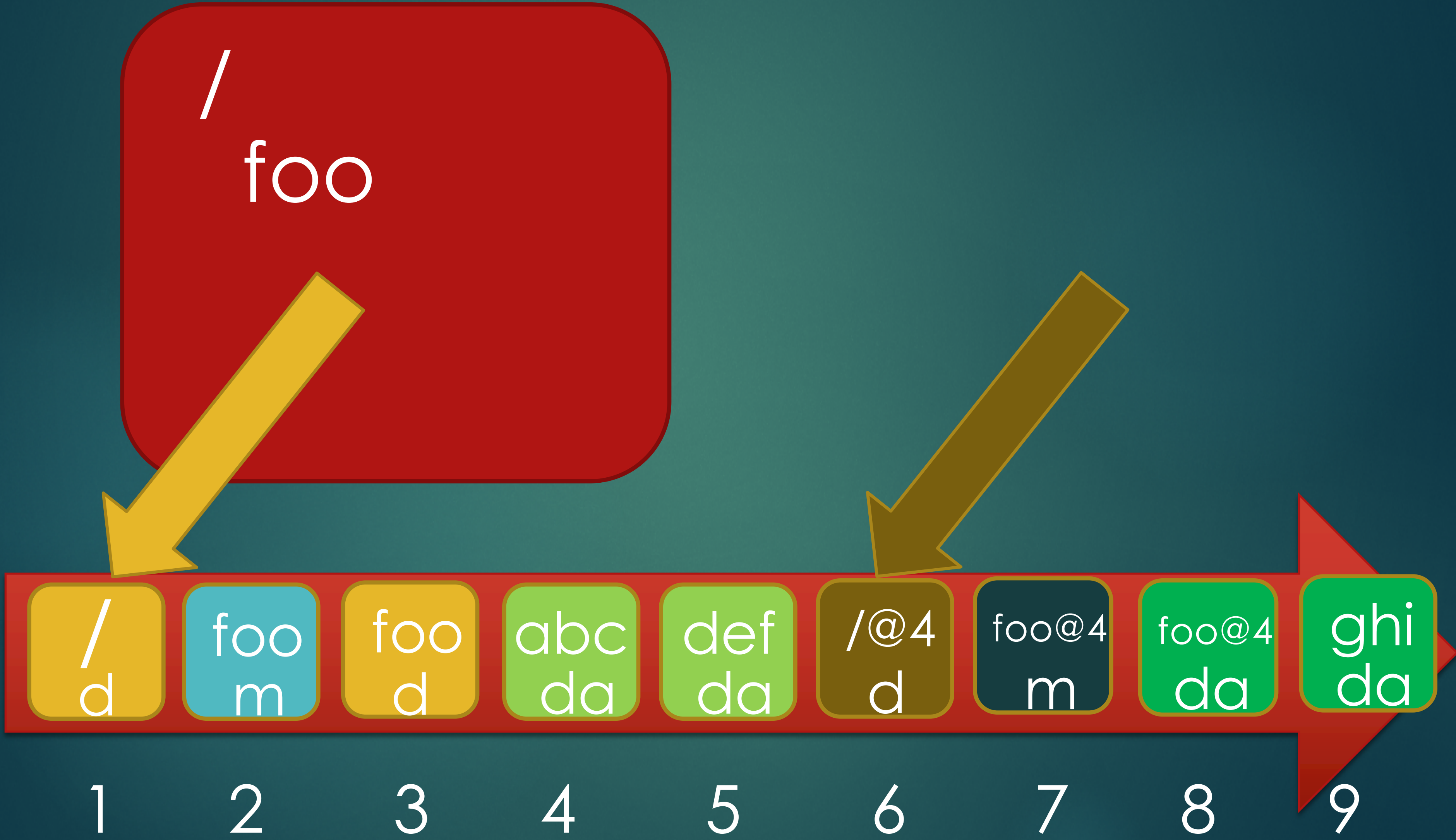
# Tiering



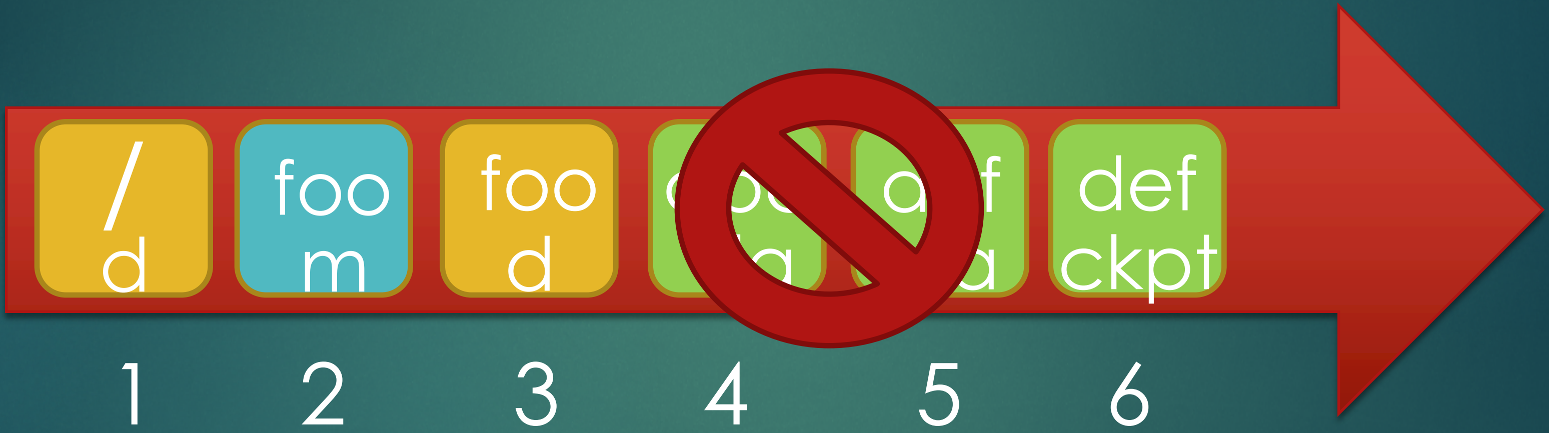
# Clones (CoW)



# Clones (CoW)



# Checkpointing



# Evaluation

- Corfu log built in Java
- FUSE prototype over JNR
- Simple: ~4,000 SLOC
- Java limits performance measurements
- Log microbenchmarks:
  - 60K appends/s, ~100k streams
  - 50ms to read a stream with 200 entries in a system with 100k streams
    (compared to 200ms+ with backpointers).

# Evaluation

- Basic Ops:
  - Cloning any part of FS: <1ms
  - Accessing clones: ~.5ms overhead
  - Snapshot access: ~2ms to access typical snapshot

# Future Work

- Merge metadata streams into directory streams
- Leverage transactional interface of Corfu
- Performance tuning: C/C++ implementation
- Comparison against HDFS, Ceph, CalvinFS

# Conclusion

- Silver is a file system architected from the ground up to take advantage of a efficient, distributed log
- Distributed logs make it easy to support advanced operations such as multi-versioning, CoW clones, distributed caching and tiering while maintaining consistency
- In future work, we hope to take our Java design, which has enabled a very rapid prototype to be built and translate it into a performant native design