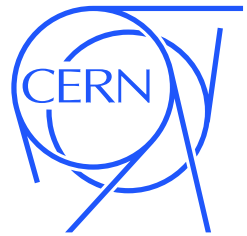


CernVM File System: Features and Future

http://cernvm.cern.ch

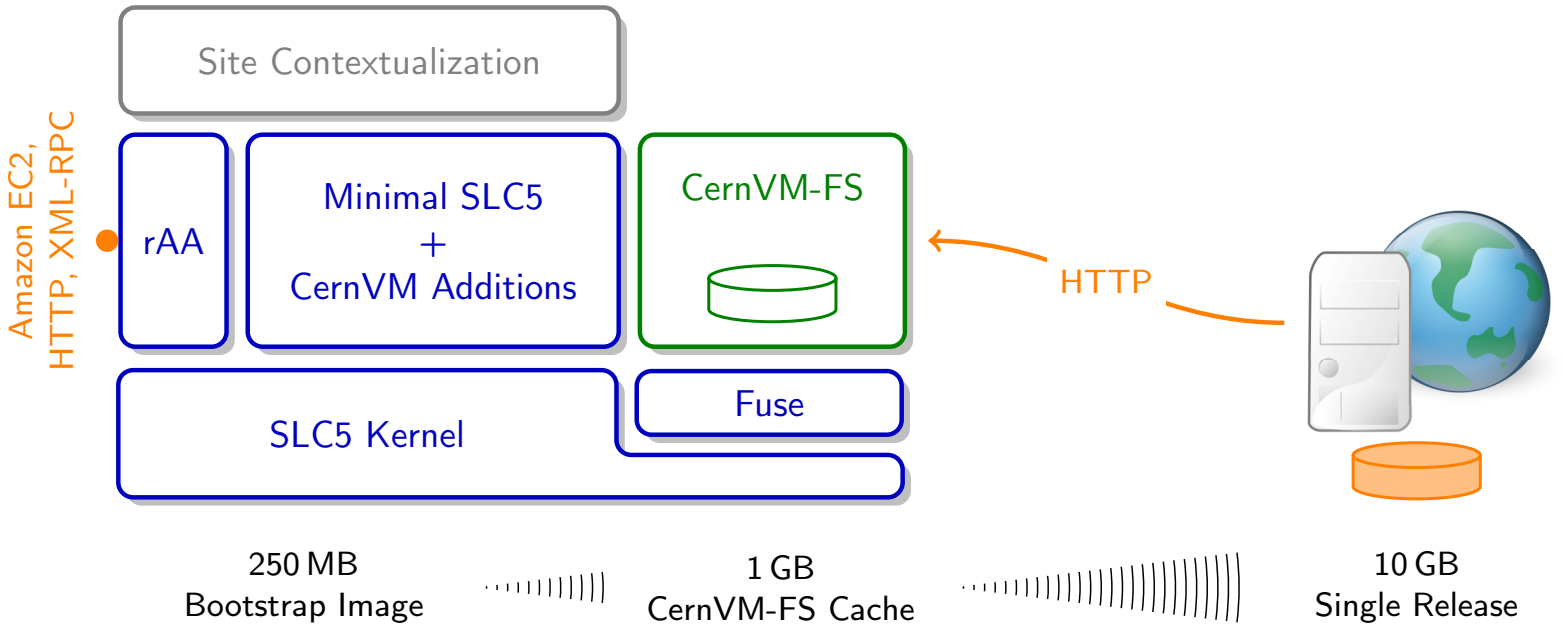
Jakob Blomer

June 2010

Outline

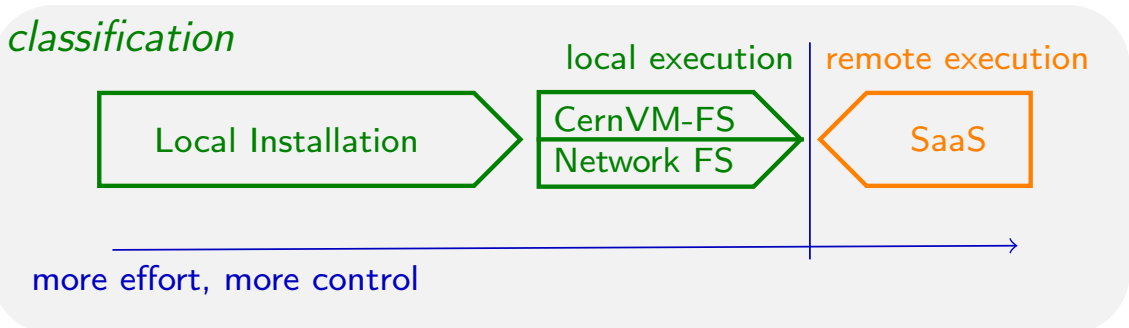
Software Distribution for the CernVM

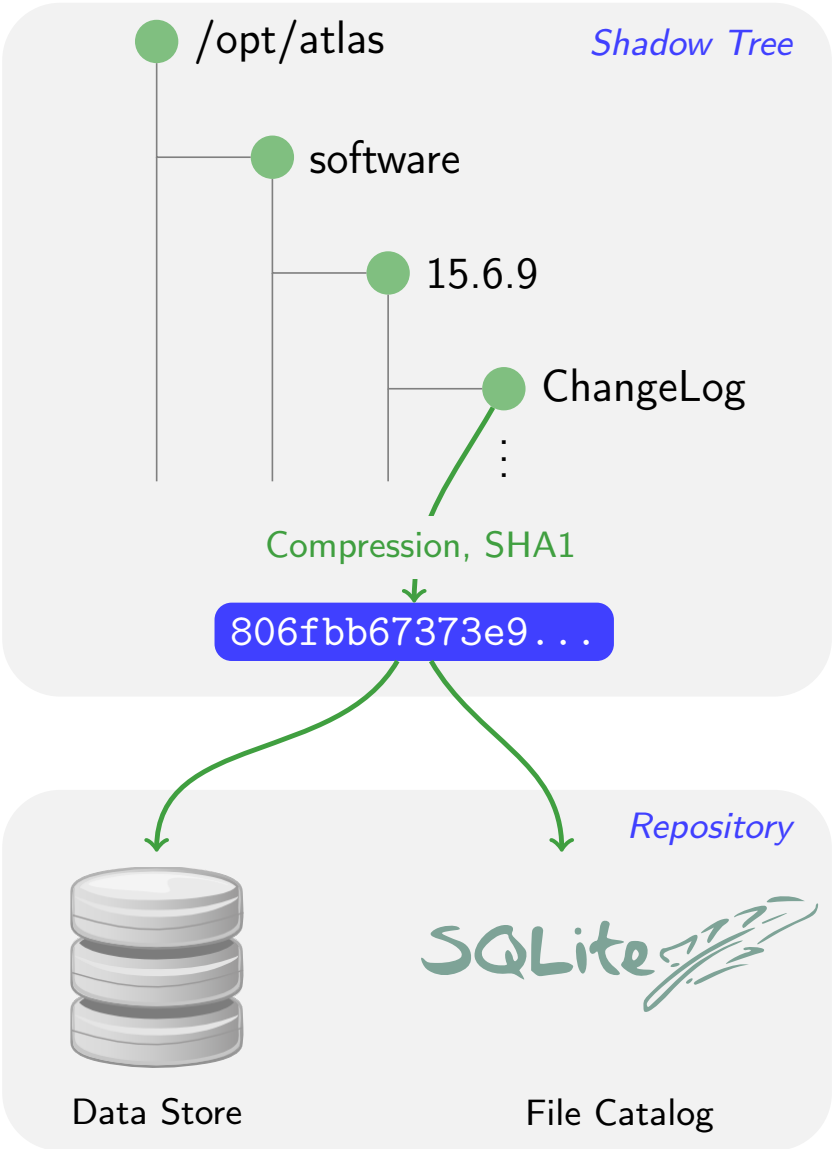
Principle: Virtual software installation by means of an HTTP File System



Essential Properties:

1. Defined platform
2. Read-only files
3. Public files





Data Store

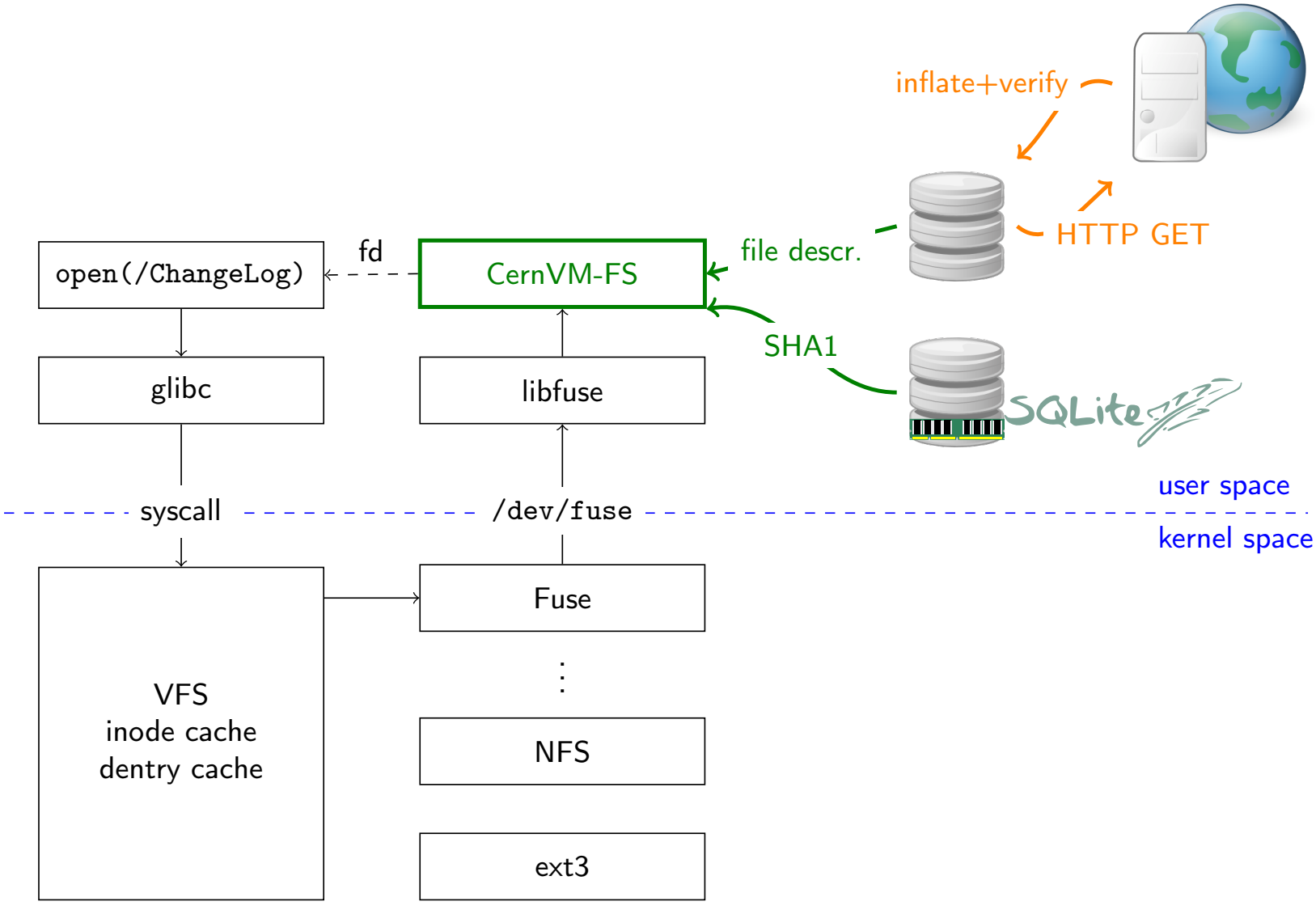
- Compressed Chunks (Files)
- Detects Duplicates
- Never Deletes

File Catalog

- Directory Structure
- Symlinks
- SHA1 of Regular Files
- Digitally Signed
- Time to Live
- Nested Catalogs

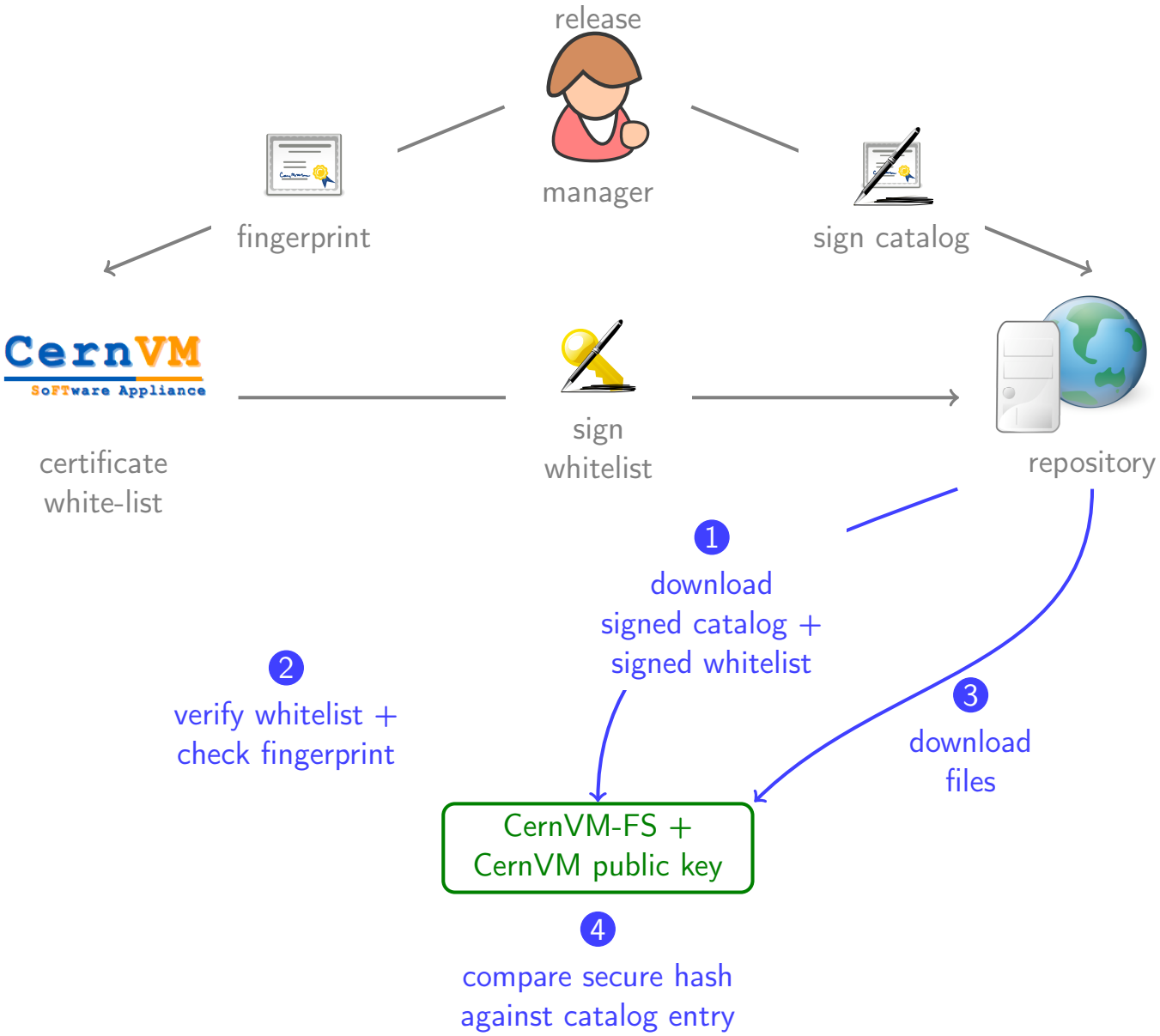
How CernVM-FS Works II

CernVM-FS Fuse Module

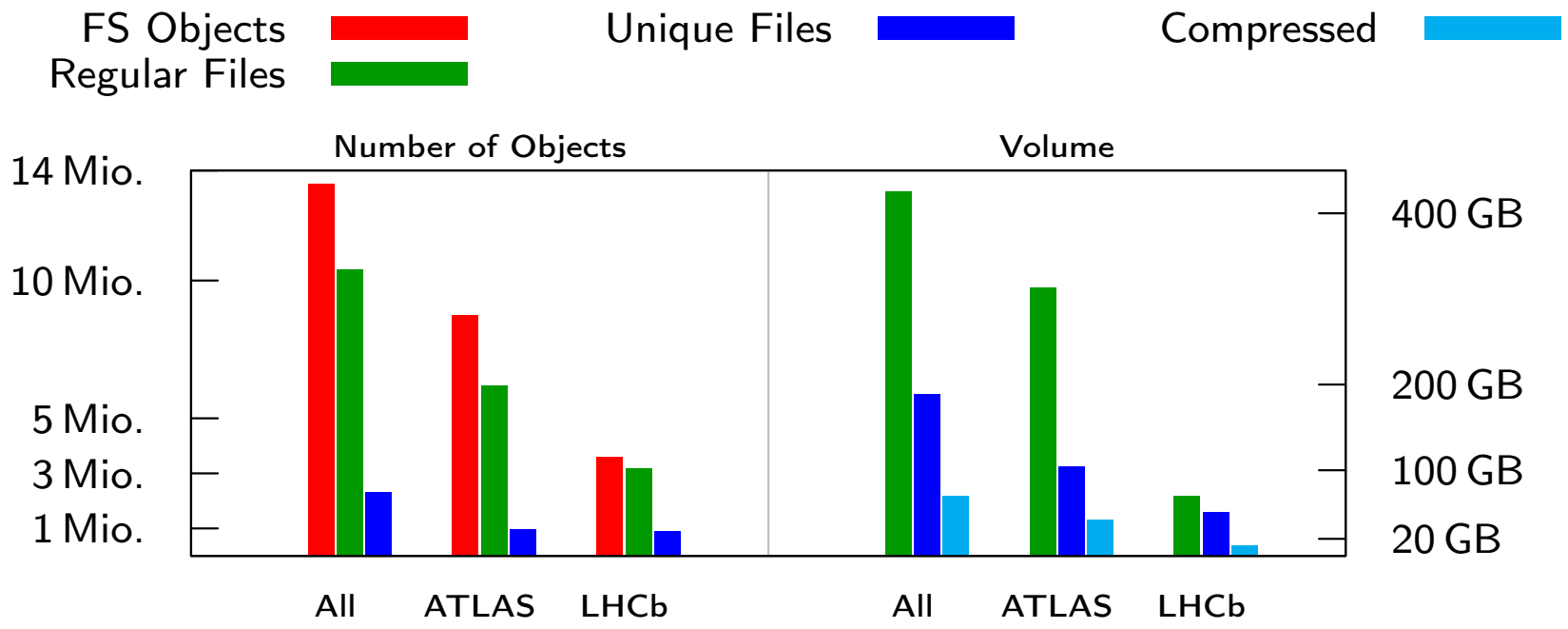


How CernVM-FS Works III

Integrity and Authenticity

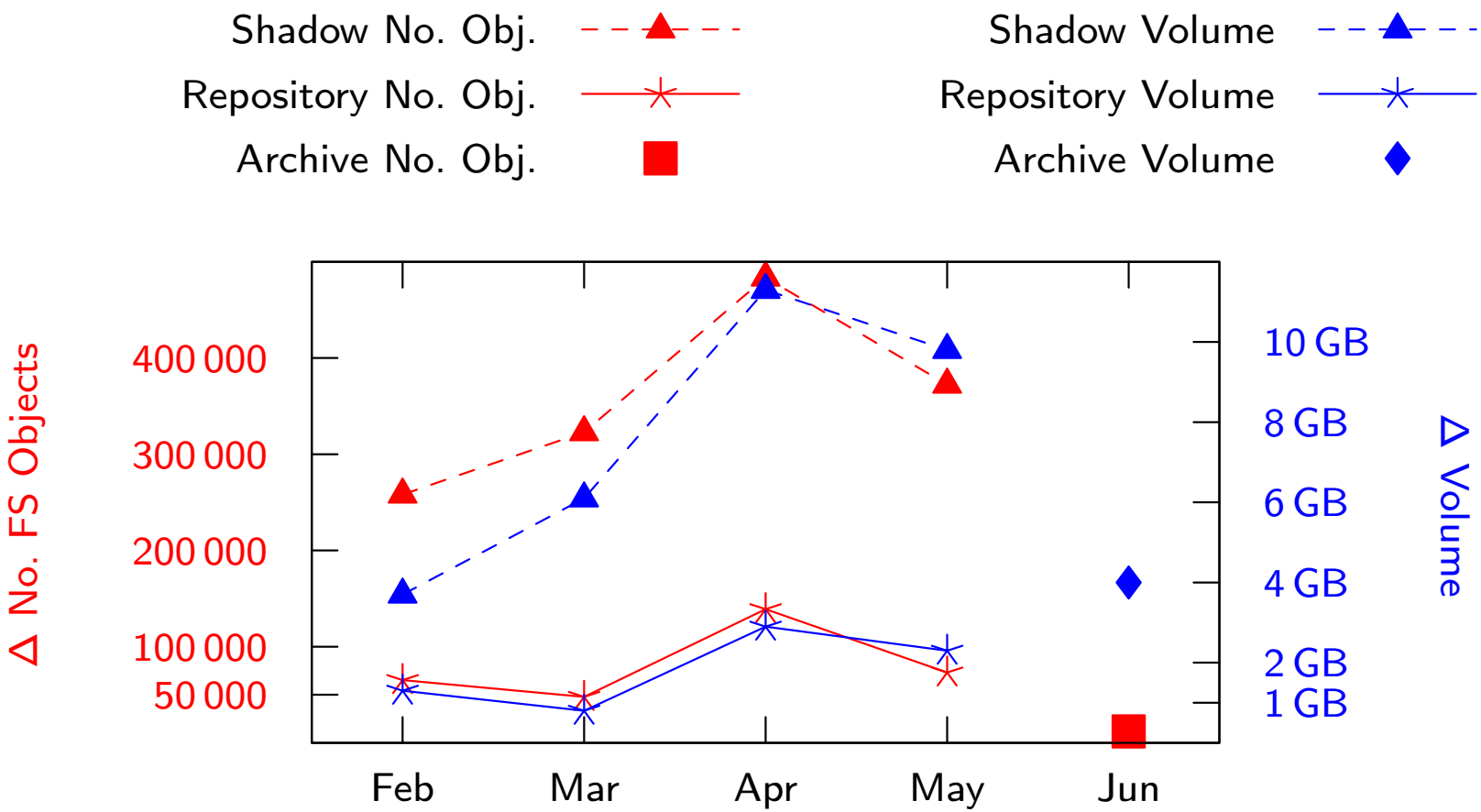


Repositories at CernVM:
ATLAS, CMS, LHCb, ALICE, LCD, NA61, H1,
Grid UI, LCG Externals
Ongoing: ATLAS Nightlies, ATLAS Conditions Database, Theory Group SW



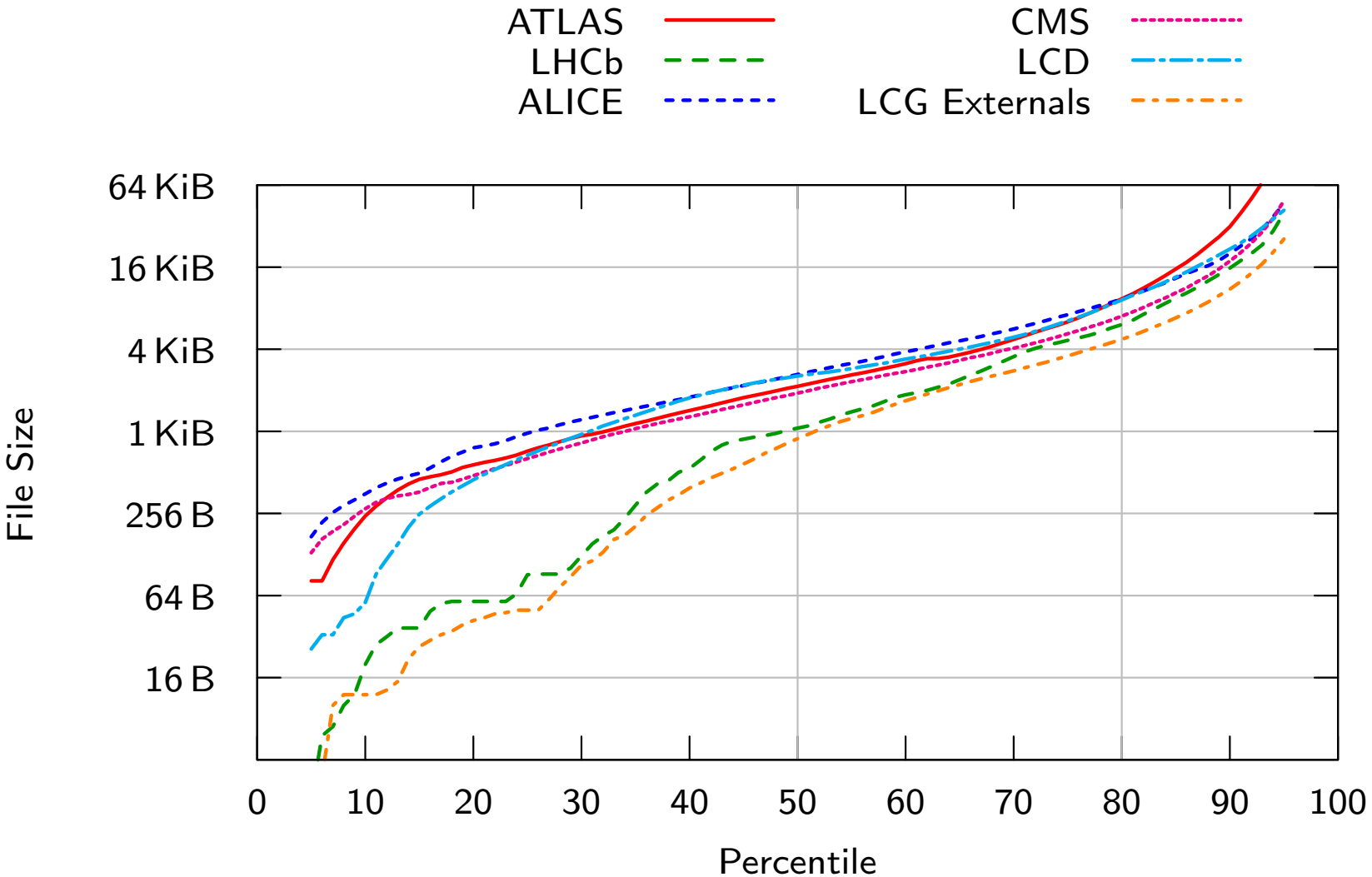
Overall: 420 GB, 13.5 Mio. File System Objects
Repository Core: 91 GB (17%), 2.4 Mio. File System Objects (17%)
(+ Archive Data)

- LHCb growth of file system objects and volume in 2010
- Unreferenced data chunks in LHCb repository by June 2010 (*Archive Data*)



Repository Statistics III

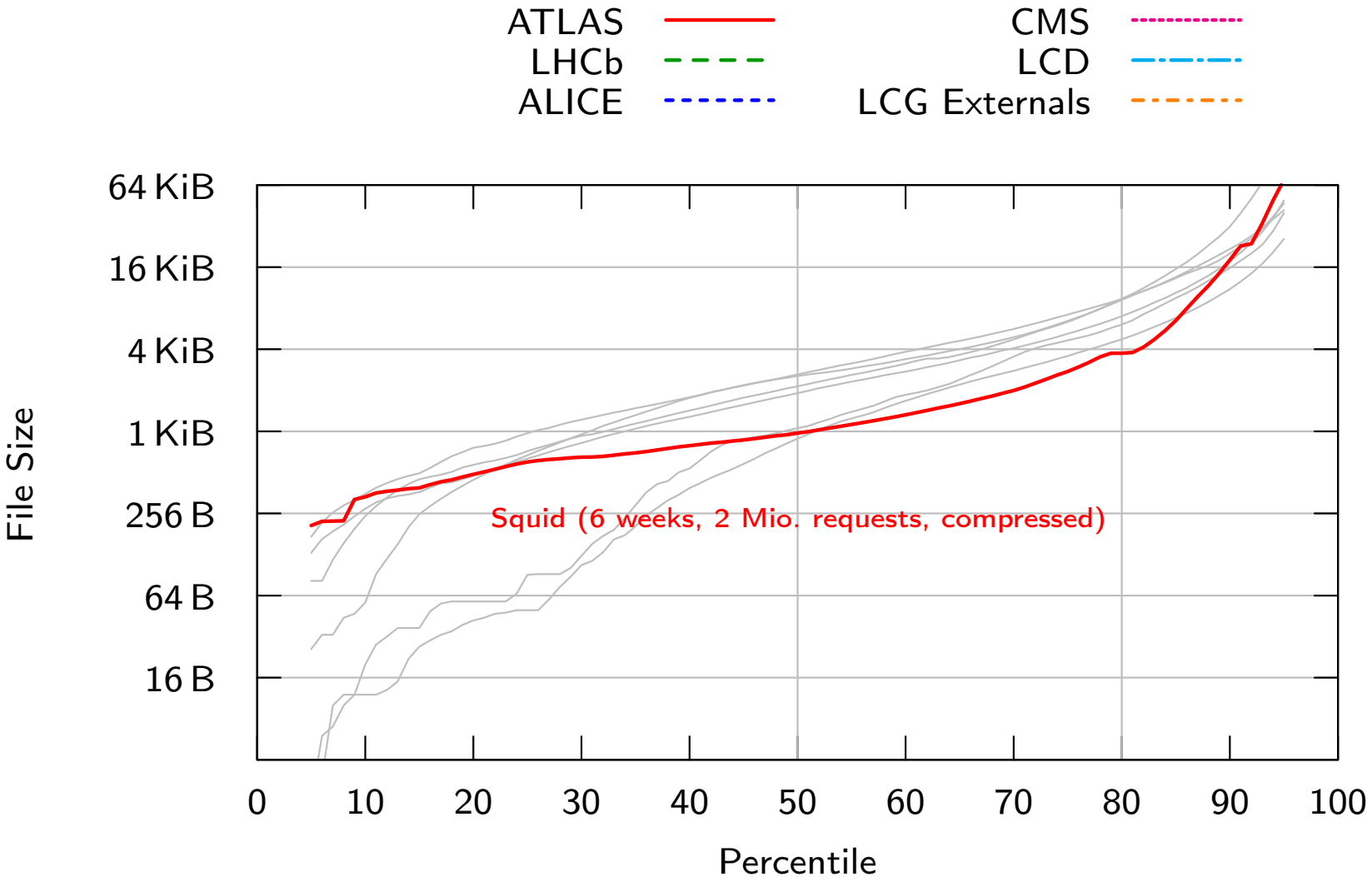
File Size Distribution of Shadow Tree



No need for file chunking, latency is key issue

Repository Statistics III

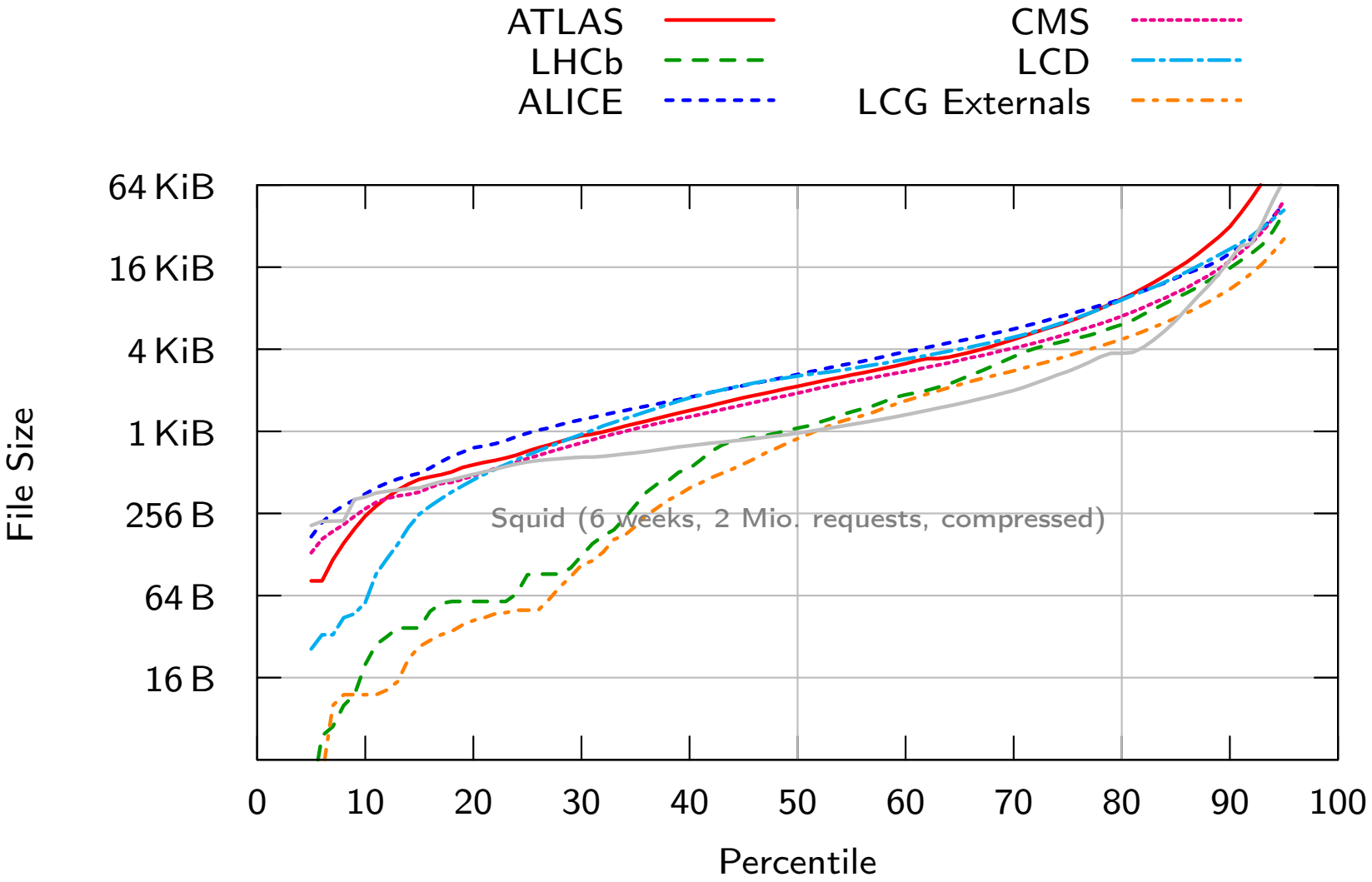
File Size Distribution of Shadow Tree



No need for file chunking, latency is key issue

Repository Statistics III

File Size Distribution of Shadow Tree



No need for file chunking, latency is key issue

Scalability and Performance

CernVM-FS distributes static, (almost) immutable web content

- Backed up by SimpleCDN content delivery network
- Piggy-back on Frontier Squids
- CernVM Squid appliance for Tier3 clusters (in development)

Cold Cache Performance:

Compile and run ROOT stressHepix, baseline running time 3 m

Extra Running Time by Latency

WAN Extra Running Time (RTT 100 ms)

Scalability and Performance

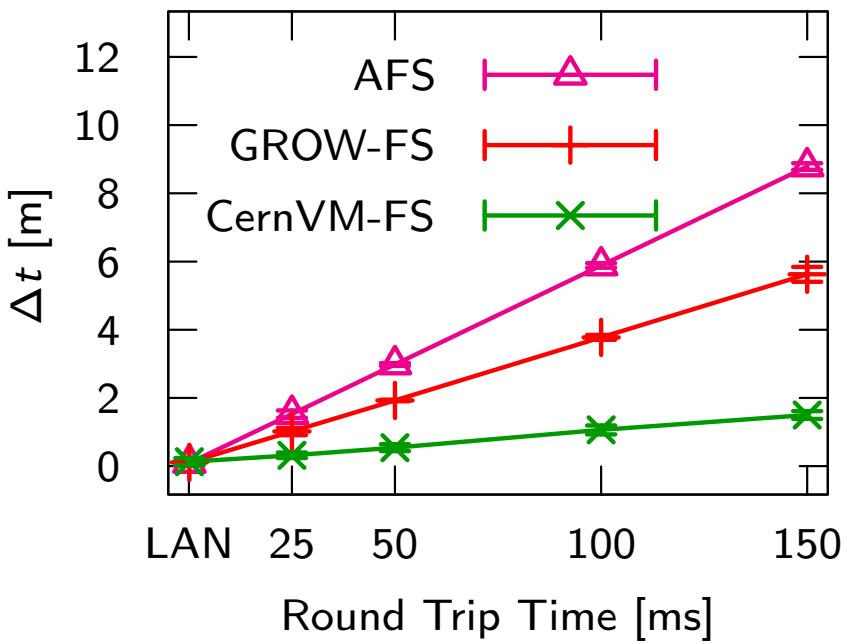
CernVM-FS distributes static, (almost) immutable web content

- Backed up by SimpleCDN content delivery network
- Piggy-back on Frontier Squids
- CernVM Squid appliance for Tier3 clusters (in development)

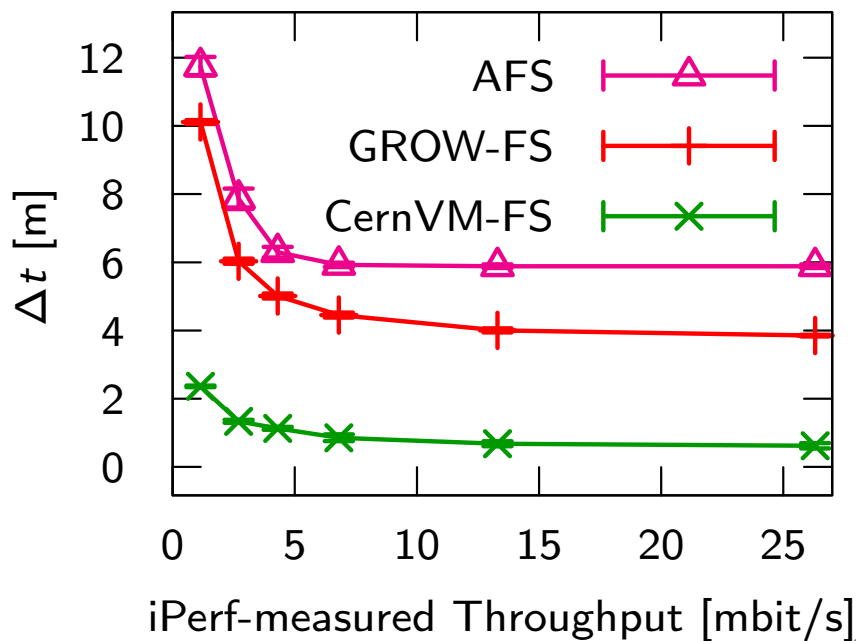
Cold Cache Performance:

Compile and run ROOT stressHepix, baseline running time 3 m

Extra Running Time by Latency



WAN Extra Running Time (RTT 100 ms)



Cache Layers: Linux Kernel, CernVM-FS direct-mapped catalog cache

Benchmark \ Syscall	stat()		open()		read()	
	all	uniq	all	uniq	all	uniq
Kernel Comp.	438.8	4.2	426.9	2.4	426.2	2.4
Kernel Cache Hitrate	98%		—		99%	
Catalog Cache Hitrate	56%		95%		—	
ATLAS Examples Comp.	4 987.7	43.5	111.1	2.3	119.5	2.3
Kernel Cache Hitrate	91%		—		96%	
Catalog Cache Hitrate	4%		94%		—	
LHCb Analysis	75.6	11.0	5.8	1.2	12.3	1.2
Kernel Cache Hitrate	81%		—		41%	
Catalog Cache Hitrate	13%		96%		—	

Syscall Numbers in Thousands

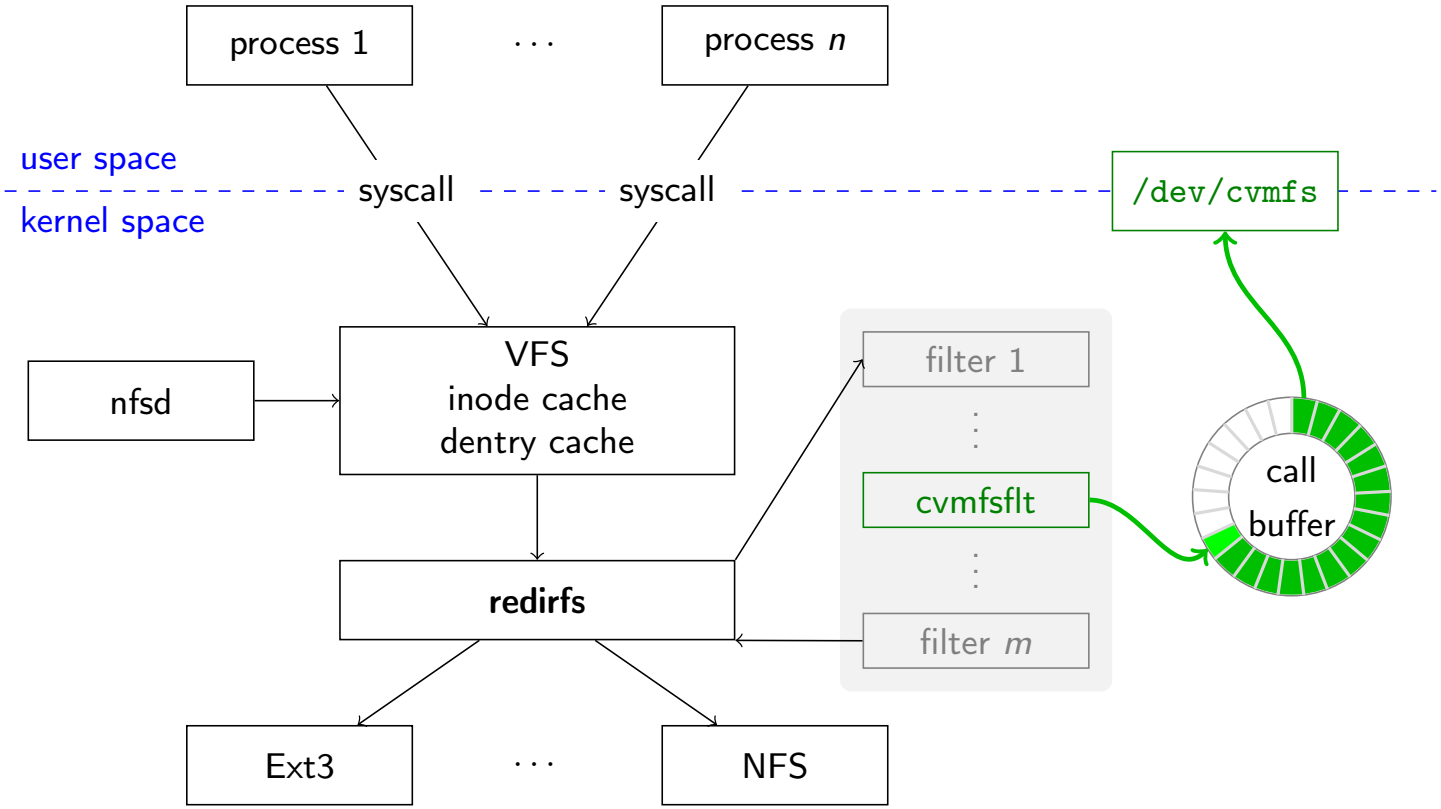
- Many syscalls on few path names
- For kernel cache hits there is no difference to local file system
- In case of running and compiling software: Fuse overhead neglegible

Developments in Progress:

1. **Revision Control**
   Missing piece: preserve file catalog revisions
   Store file catalogs as file chunks, store revision history in file catalogs
   With 2 releases per week: < 10 GB and 800 files per month
2. **Failover-Mirror of the Repository**
   CernVM-FS already supports automatic host failover
   Mirror maintained by CernVM-customized Squid / rsync software appliance
3. **Repository Updates & Publishing**

Problem	Synchronize shadow tree and repository Requires efficient calculation of file system change set
Syscall Hooks	Not the right interface, e. g. VFS calls from NFS kernel daemon
Fuse Module[1]	Without direct IO: very slow on write With direct IO: no `mmap()`
inotify	Not scalable, requires watchpoint for every directory Kernel event buffer can overrun
Kernel Module	Hook in VFS calls inside kernel General kernel-level framework *redirfs* available

[1] See http://xtreemfs.blogspot.com/2008/08/fuse-performance.html



Redirfs filters activated based on path prefix:

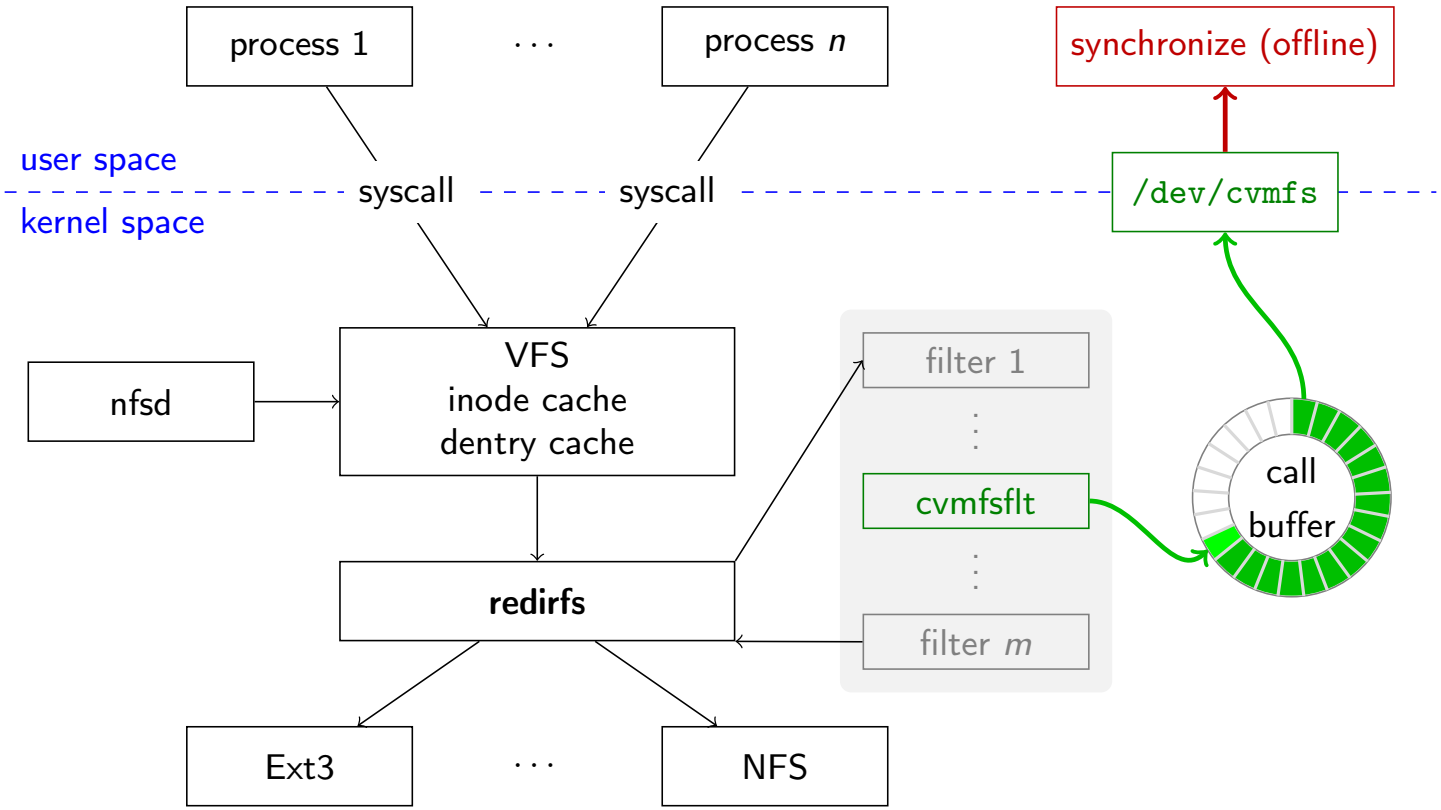
Normal Operation Capture writing VFS calls to character device

Call Buffer Full Block writing VFS calls

Synchronizing Repository Forbid writing VFS calls

Repository Maintenance II

VFS Filter based on Redirfs



Redirfs filters activated based on path prefix:

Normal Operation Capture writing VFS calls to character device

Call Buffer Full Block writing VFS calls

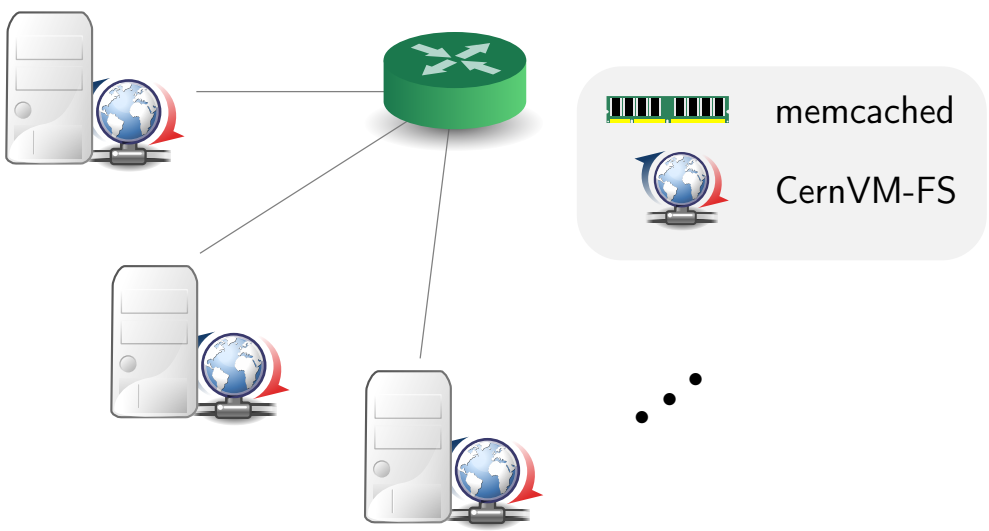
Synchronizing Repository Forbid writing VFS calls

Use Case Local cluster of virtualized worker nodes

Requirements

- Relieve central cache entity
- Deal with potentially high peer churn
- ZeroConf

Idea Customized DHT algorithm for memcached[1], Auto-configuration by multicast IP



memcached

- LRU Memcache
- Slab allocator
- Steering via TCP/UDP

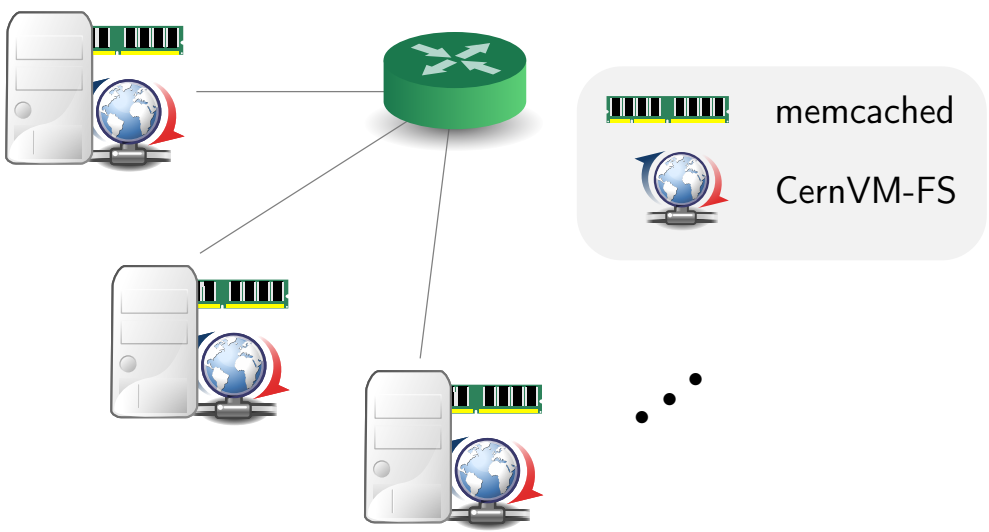
[1] Blomer, Fuhrmann, *A Fully Decentralized File System Cache for the CernVM-FS*, GridPeer 2010

Use Case Local cluster of virtualized worker nodes

Requirements

- Relieve central cache entity
- Deal with potentially high peer churn
- ZeroConf

Idea Customized DHT algorithm for memcached[1], Auto-configuration by multicast IP



memcached

- LRU Memcache
- Slab allocator
- Steering via TCP/UDP

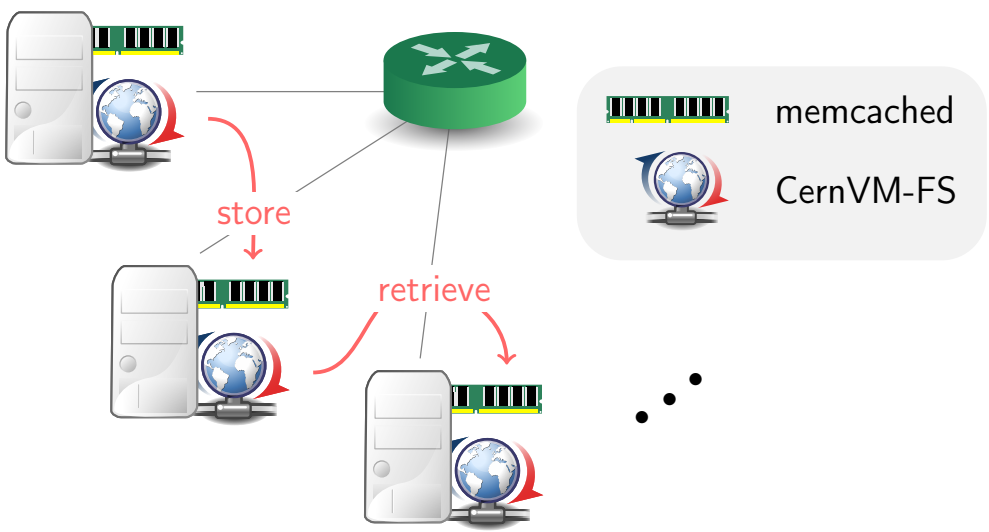
[1] Blomer, Fuhrmann, *A Fully Decentralized File System Cache for the CernVM-FS*, GridPeer 2010

Use Case Local cluster of virtualized worker nodes

Requirements

- Relieve central cache entity
- Deal with potentially high peer churn
- ZeroConf

Idea Customized DHT algorithm for memcached[1], Auto-configuration by multicast IP



memcached

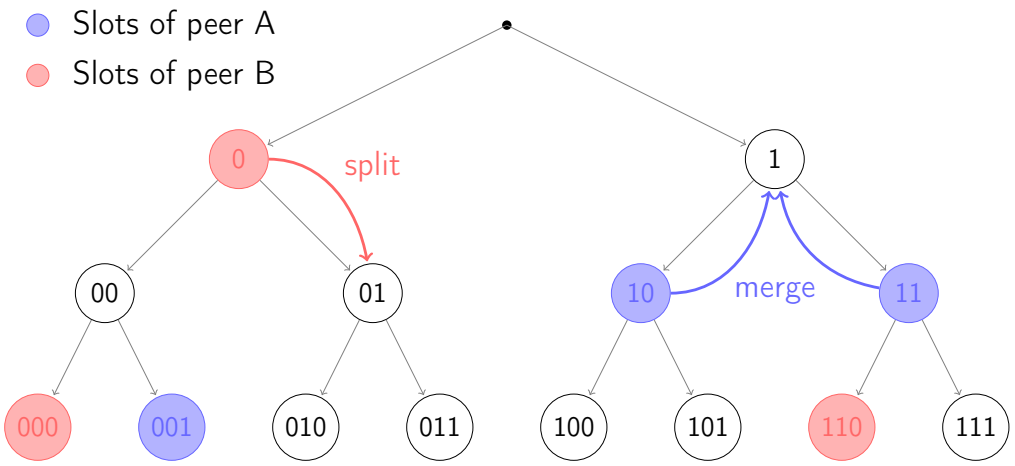
- LRU Memcache
- Slab allocator
- Steering via TCP/UDP

[1] Blomer, Fuhrmann, *A Fully Decentralized File System Cache for the CernVM-FS*, GridPeer 2010

Which peer's memory cache contains a certain data chunk?

Classic DHT: Peer ID determines hash space responsibility
Our Algorithm: Peers flow freely in hash space

Every peer maintains a small number of slots:



requires $< 1\,\mathrm{KB}$ / peer

split

Drop responsibility for left/right subtree on

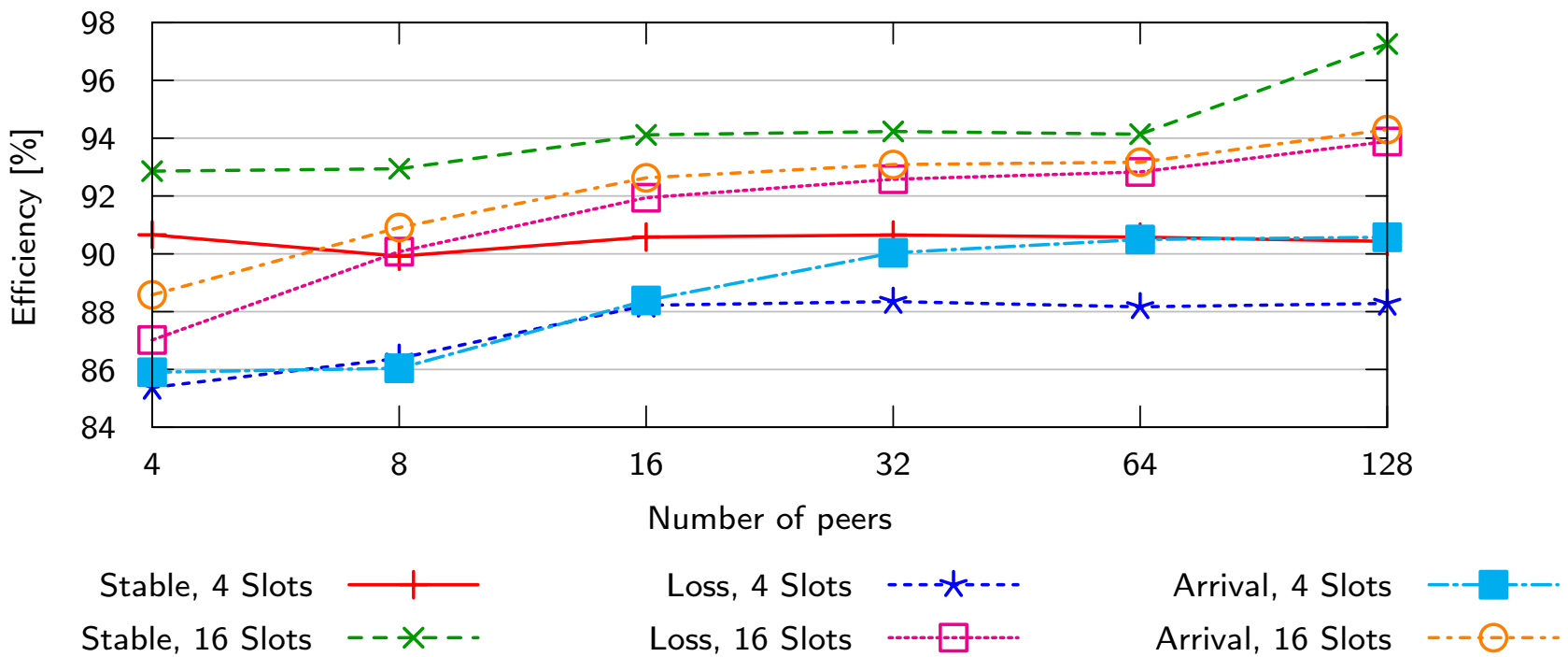
- too many cache misses
- too many requests

merge

Make a free slot by merging existing ones with largest common prefix

Discrete event simulator fed with traces of ATLAS examples compilation

- 110 000 `open()` calls of 1 100 distinct files
- Not equally distributed over time, opened in hot spots
- Distribution of requests: 10%-40% deviation from mean
- Efficiency in case of peer churn:



Current State of CernVM-FS

- Specialized network file system for software repositories
- Serves 420 GB and 13.5 Million files and directories
- Outperforms NFS and AFS
- In practice, warm cache speed comparable to local file system
- Scalable infrastructure
- Delivered auto-configured by CernVM or independently as rpm/yum package[1] or source tarball[2]

Developments in Progress

- Better support for (virtualized) Tier3s and clusters
- Simple repository installation for small VOs
- Extend to conditions databases

---

[1] http://cvmrepo.web.cern.ch/cvmrepo/yum

[2] https://cernvm.cern.ch/project/trac/cernvm/downloads