Status and Future Perspectives of CernVM-FS

http://cernvm.cern.ch/portal/filesystem

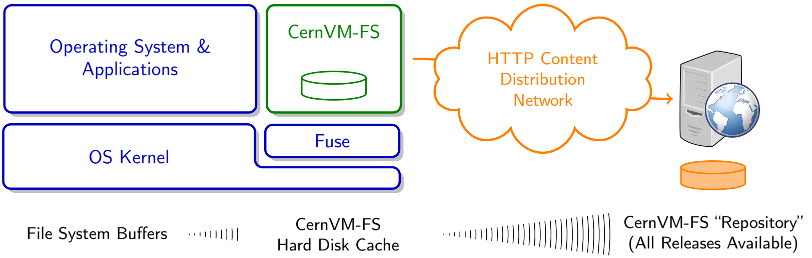
Jakob Blomer, Predrag Buncic, Ioannis Charalampidis,
Artem Harutyunyan, Dag Larsen, René Meusel

CERN PH-SFT

CHEP 2012

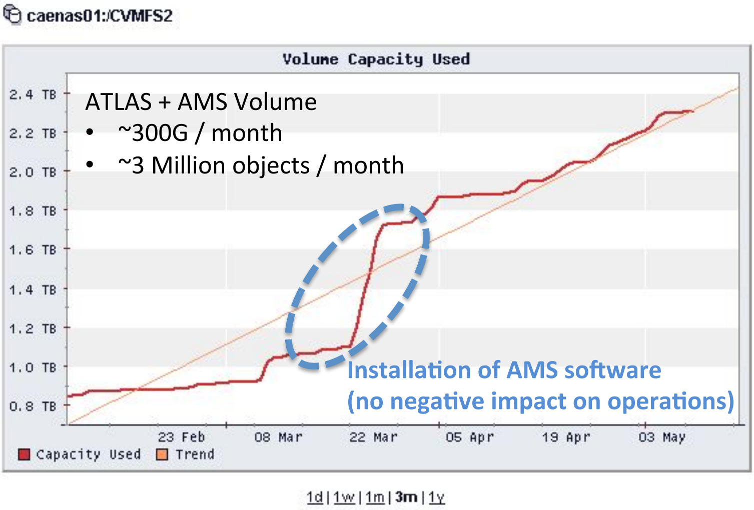
CernVM File System – Overview

Caching HTTP file system, optimized for software delivery



(Known) Users:	ATLAS (+ Conditions Data), LHCb (+ Conditions Data), CMS, NA61, NA49, BOSS, Geant4, AMS, LHC@Home 2.0
CDN:	Full replicas at CERN, RAL, BNL, ASGC, FermiLab Site-local cache servers (Frontier Squids)
Avg. Load:	Very modest, $\approx$5 MB/s, 20 requests per second on CERN Replica
Volume:	75 million objects (2010: 30 million), 5 TB (2010: 1 TB)

Caching HTTP file system, optimized for software delivery



Plot provided by Line Everaerts

CernVM-FS Client in Heterogenious Environments

In order to fully benefit from CernVM-FS, the file system has to be available on all relevant computing resources.

Range of Environments:

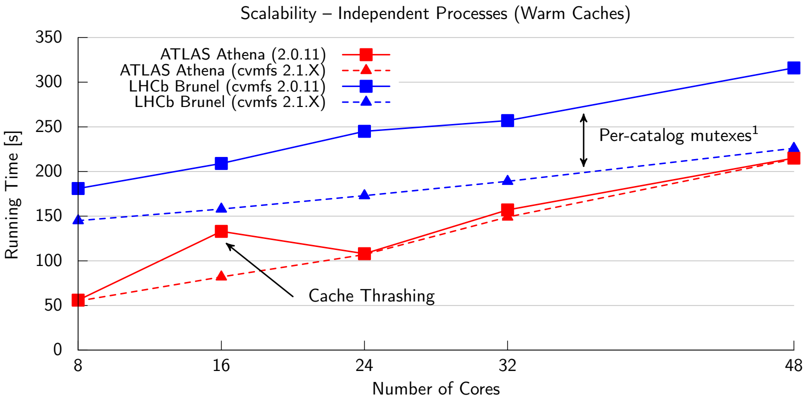
Scientific Linux, Ubuntu, SuSE, OS X
1 core to 48+ cores
1 to 10 mounted repositories
Possibly no Fuse, no local hard disk

Portability:

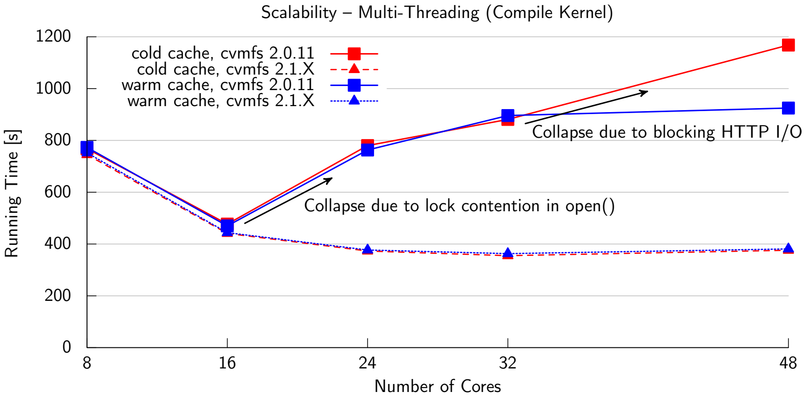
- Portable C++ / POSIX code
- Library interface, connector to *Parrot* (by Dan Bradley)

Scalability:

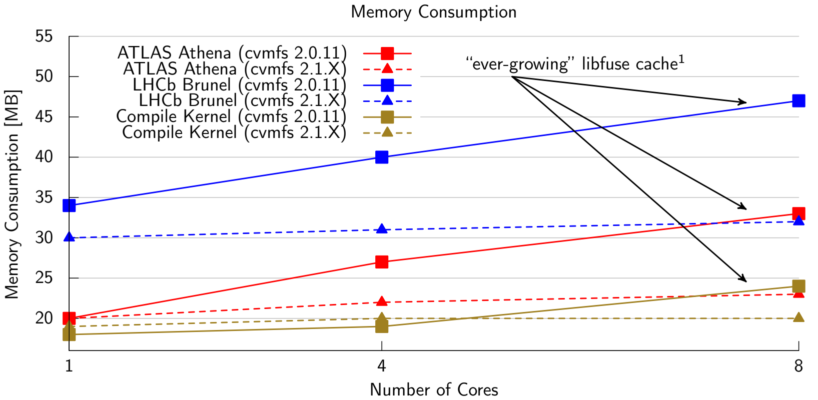
- Memory fragmentation
  open hash collision resolution $\mapsto$ linear probing
  path strings stored on the stack
- Cache thrashing
  direct mapped cache $\mapsto$ LRU cache
- Concurrent file system access
  Fine-grained locking
  Asynchronous, parallel HTTP I/O



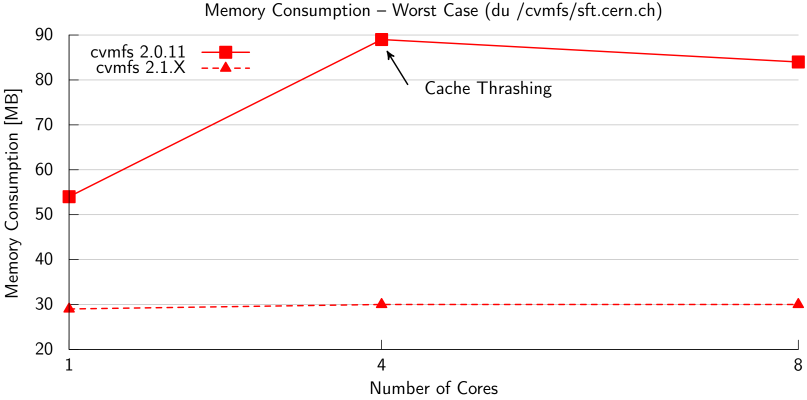
[1]LHCb uses ≈300 file catalogs: fine-grained locking pays off



Scalability of version 2.1.X limited by the scalability of the kernel build system

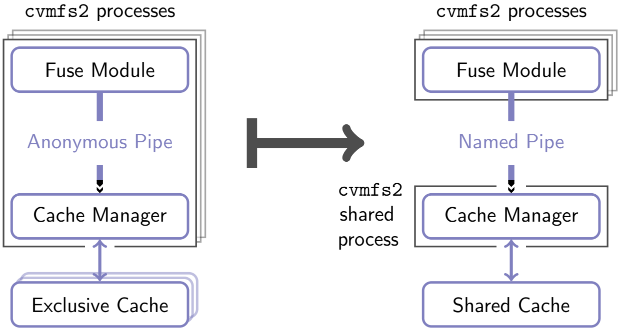


[1]libfuse cache shrinks on high memory pressure



- Worst case: recursive listing of /cvmfs/sft.cern.ch (1.5 Million entries, up to 6 000 entries per directory)
- Memory fragmentation with `std::string` becomes an issue

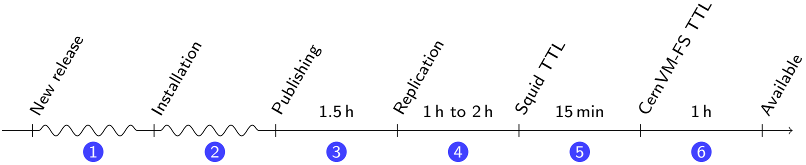
Issue: Enforce shared *quota*, coordinated bookkeeping required
Idea: Turn the cache manager thread into a shared process



- No extra service: automatically spawned by first cvmfs mount point, automatically terminated by last unmount
- Named pipe can be turned into a network socket:
  Foundation for distributed shared memory cache

Benchmark: Distribute new ATLAS release
400 000 files and directories, 10 GB compressed, 20 % new data

Necessary steps and delays with current version:



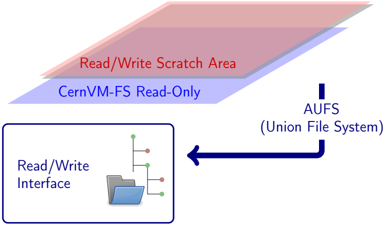
(Compared to "Grid Installation Jobs": delay reduced from days to 4 h to 5 h)

Challenges: POSIX read/write interface required
Bulk write of many small files

Goal: Overall delay less than 1 h

CernVM-FS Publish Interface using Overlay-FS

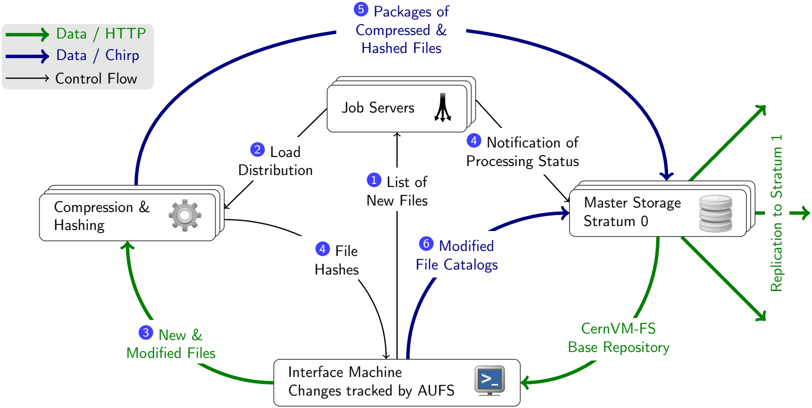
- AUFS part of Scientific Linux
- $< 5\,\%$ performance loss (untar)



Improvements compared to a
separate read-write copy of the repository:

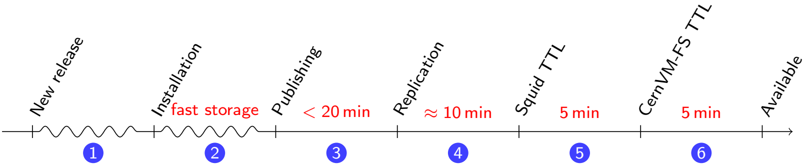
- Authoritative repository storage benefits from de-duplication
  Storage savings for ATLAS: from 22 Million to 1.8 Million objects
- Encapsulated change set in scratch area
- Snapshots provided by CernVM-FS

Scaling of the Publish Process



Roles:	File System Interface, Worker Node, Job Manager, Master Storage (+ Stratum 0 Webserver, Signing Server)
Protocols:	Chirp, HTTP
Storage Interface:	Put, Get, Rename/Commit (on Stratum 0)

New ATLAS release, 400 000 files and directories:



Details by step:

- ② Encapsulated scratch area allows for fast local storage / ramdisk
- ③ Distributed prototype reduces processing of the changeset to <20 min
- ④ Immediate replication at 4 MB/s: $<$ 10 min
- ⑤ Can be reduced to 5 min
- ⑥ Can be reduced to 5 min to 15 min

$\Longrightarrow$ Overall delay from 210 min to 270 min to 30 min to 50 min

CernVM-FS Client

- On the way to support *all* relevant HEP computing resources

Publisher's End

- Persistent storage entirely in CernVM-FS format
- Time to publish a new release can be reduced to $< 1\,\mathrm{h}$

CernVM-FS has the potential to be used as exclusive software distribution system with low maintenance on both reader's and publisher's end.

Source code: `https://github.com/cvmfs/cvmfs`
Nightly builds: `http://ecsft.cern.ch/dist/cvmfs`
Mailing lists: `cvmfs-talk@cern.ch`, `cvmfs-devel@cern.ch`
Next major release planned for August 2012

Do not forget to visit the

CERN PH/SFT Group Booth

in Kimmel Center (right in front of coffee table on 4th floor)

To learn more about the **CERN Virtual Machine**
Poster 134: *Managing Virtual Machine Lifecycle in CernVM Project*
Poster 135: *Long-term preservation of analysis software environment*

To learn more about **CernVM Co-Pilot**
"CernVM Co-Pilot: an Extensible Framework for Building Scalable Cloud Computing Infrastructures"
(by A Harutyunyan)

5 Backup Slides

New meta-data memory cache:

Memory Cache in 2.0.X

- inode $\mapsto$ {path, meta-data} cache by libfuse:
  size controled by memory pressure
- Hash map with chaining as collision resolution
  (vulnerable to memory fragmentation)
- path $\mapsto$ meta-data cache:
  direct-mapped / 2-way-associative hybrid cache
  (vulnerable to cache thrashing)

Memory Cache in 2.1.X

- All caches: CernVM-FS least-recently-used (LRU) data structure
- LRU: $\mathcal{O}(1)$, hash map with linear resolving + list
- static memory pool pre-allocated

CernVM-FS Versions:	2.0.11, 2.1.0 preview (git-86806d060e5) default installation
Machines:	Intel Xeon E5345 (8 cores), 8 GB RAM, SLC5
	AMD Opteron 6164 HE (48 cores), 96 GB RAM, SLC6
RTT to Web Proxy:	100 μs to 200 μs
Repository Revisions:	ATLAS – 526 (SHA-1 eb3d939bdc7af12882383d905e52571772a946ec)
	LHCb – 2151 (SHA-1 85c4d9e7ccd3bd7db5bb9b60cb57fd0c0a567cdd)
	SFT – 125 (SHA-1 a271e80cce9947b2c21c9bc0f5850bc551600bb6)

Benchmark Scripts:

```
. /cvmfs/lhcb.cern.ch/etc/login.sh
. SetupProject.sh Brunel
gaudirun.py $BRUNELSYSROOT/tests/options/testBrunel.py
```

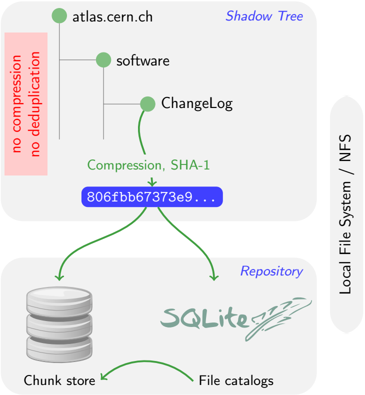
```
. $ATLAS_LOCAL_ROOT_BASE/user/atlasLocalSetup.sh
. $AtlasSetup/scripts/asetup.sh -cmtconfig=x86_64-slc5-gcc43-opt 17.4.0
athena.py AthExHelloWorld/HelloWorldOptions.py
```

```
cd /cvmfs/sft.cern.ch/lcg/external/experimental/linux
./compileKernel.sh 2.6.32.57
```

```
du -ch -max-depth=3 /cvmfs/sft.cern.ch
```

Discarding the "Shadow Tree"

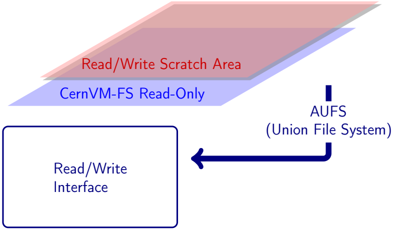
Current Backend



$$\text{Repository}_{r+1} = f(\text{Repository}_r, \text{Shadow Tree}, \text{ChangeLog})$$

- 2 data copies
- ChangeLog requires kernel module

New Backend



$$\text{Repository}_{r+1} = g(\text{CernVM-FS}_r, \text{Scratch Area})$$

- Standard components
- $< 5\,\%$ performance loss (untar)
- Snapshots provided by CernVM-FS

(Not shown: interface to storage)
Storage savings for ATLAS:
From 22 Million to 1.8 Million file system objects

Status of Envisaged Features

Mac OS X support	
From sources	ready for testing
Packaging (by Manuel Giffels)	under development
NFS Export	
For immutable mount points	ready for testing
Including automatic catalog reload	under investigation
Shared local hard disk cache	ready for testing
Encrypted repository / ownership support	planned
Distributed storage backend	under development
Connector to Parrot (by Dan Bradley)	to be ported from 2.0 branch
Distributed Shared Memory Cache	
Automatic peer discovery	ready for testing
Support for "μ file catalogs"	to be ported from 2.0 branch
Support for file chunking	to be implemented
Remote cache access	to be implemented