CernVM-FS at Extreme Scales

CHEP 2023, Norfolk, VA, USA

Speaker: Laura Promberger

Jakob Blomer[1], Laura Promberger[1], Valentin Völkl[1] and Matt Harvey[2]

May 9, 2023

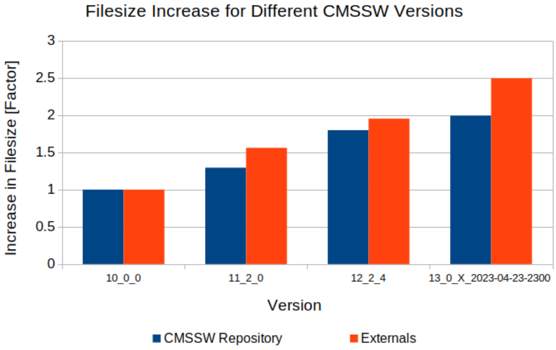
[1]CERN, Experimental Physics Department, Switzerland
[2]Jump Trading

Motivation

Expectation for HL-LHC

Increase of all CVMFS metrics by an order of magnitude

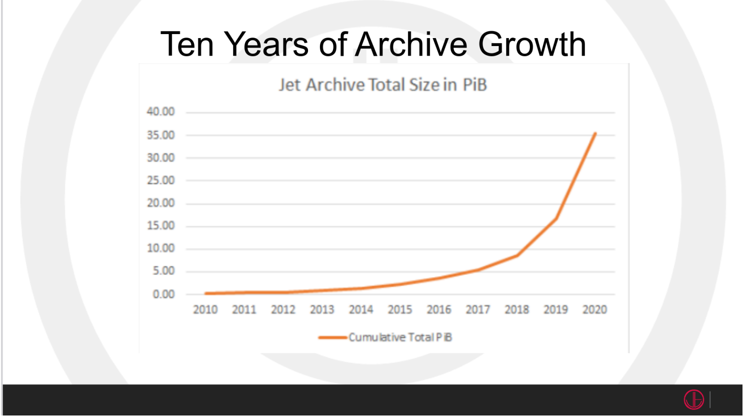
- More software
- Even more small files (e.g. Python)
- More data stored
- More users
- More containers
- More (parallel) publishing
- ...but not necessarily more repos



Each version: 20 - 55% larger

Good News: We know already it works...

Jump Trading is a “international quantitative research company” that uses CVMFS

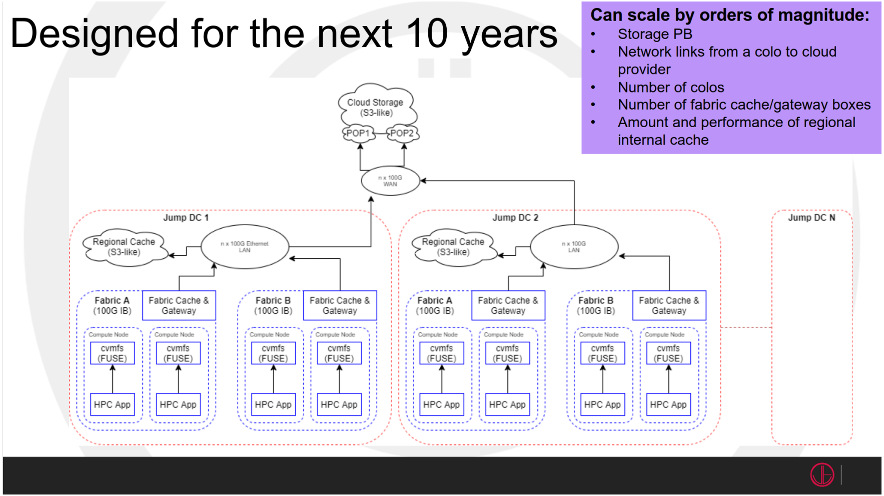


Ten Years of Archive Growth

Full presentation `https://indico.cern.ch/event/1079490/`

Jump Trading: Architecture

Good performance achieved through multiple level of caches



Designed for the next 10 years

Can scale by orders of magnitude:

- Storage PB
- Network links from a colo to cloud provider
- Number of colos
- Number of fabric cache/gateway boxes
- Amount and performance of regional internal cache

… And This is How We Improve Even Further

Performance Improvements

- 2.10 Page Cache Tracker: Much better use of kernel page cache
- 2.11 Symlink caching for fuse3 (Kernel 6.2, RedHat backporting request open)
- 2.11 `Statfs` caching
- WIP 2.11 Parallel file decompression during download
- Future Prefetching of known files clusters (Python, ROOT, etc.)
- Future `Zstd` as new compression algorithm

Rare Bugs

- 2.10 Support for in-place replacement of files without crashing long-running software that use the "old" version of these files

… And This is How We Improve Even Further II

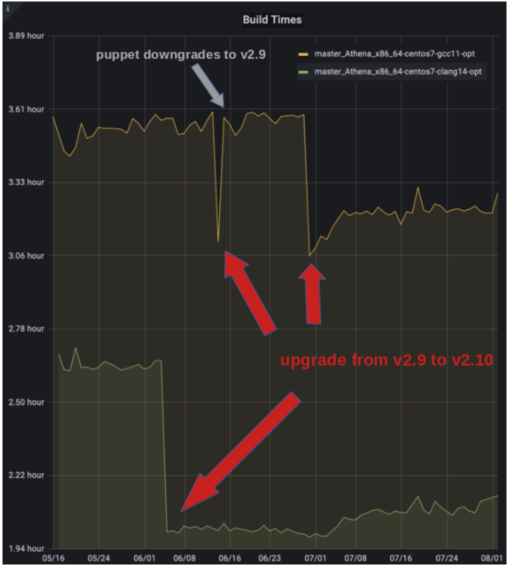
Operational Improvements

- 2.10 More extended attributes, and 2.11 protected extended attributes
- 2.11 Telemetry exposure of `internal affairs` to allow better monitoring
- 2.11 Quicker garbage collections and `cvmfs_server check`
- WIP 2.11 Proxy sharding to allow for better caching
- Future Creation of official Helm chart for cvmfs on Kubernetes

Publishing Improvements

- 2.10 Better publish failure handling on publishers
- 2.10 Support for unpacking container images through Harbor registry proxies
- Future Feature parity between remote publishers (with gateway) and local publishers

ATLAS Performance: CVMFS version 2.9 vs 2.10



Many-core compilation of ATLAS Athena with having the build tools on cvmfs

Improvements due to the page cache tracker

Some First Performance Comparison - Setup

Setup

- CVMFS client: 2x AMD EPYC 7302 16-Core, 256 GB RAM, 2 TB NVMes
- Private squid proxy: 1x Intel i7-7820X 8-Core, 64 GB RAM, 1 TB HDDs

Commands: Load software from CVMFS

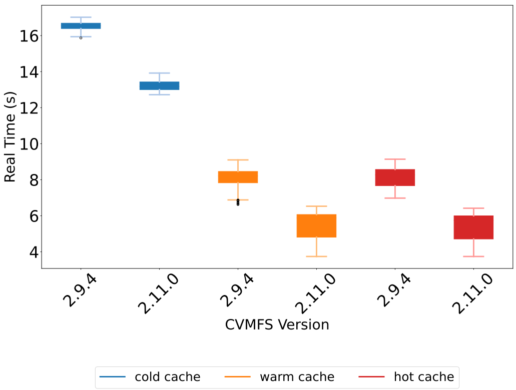
- `CMS`: Create a simulation setup script
- `DD4Hep`: Load detector description in ROOT
- `ROOT`: Load ROOT and draw a histogram
- `Tensorflow`: Load python and the modules `numpy` and `tensorflow`

Measurements

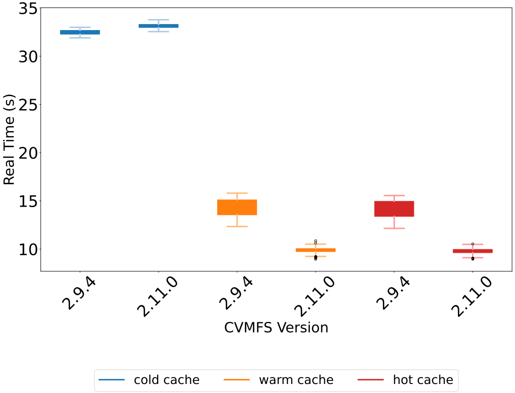
- Cold, warm, and hot cache
- `time`, `cvmfs_talk -i <repo> internal affairs`

Some First Performance Comparison - version 2.9 vs 2.11 (WIP, April 23)

(Real) run time in seconds



CMS

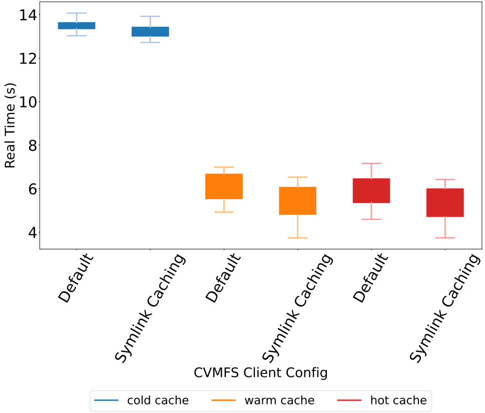


Tensorflow

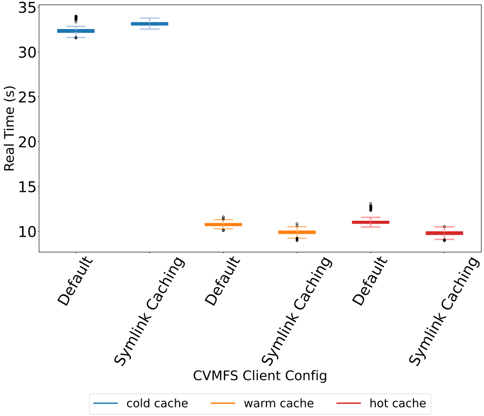
Some First Performance Comparison - Symlink Caching

CVMFS v2.11 (WIP, April 23) with and without symlink caching

(Default Client Config: Statfs Caching, Kernel Caching)



CMS



Tensorflow

Future: A first exploration of using Zstd

Compressing CVMFS cache file chunks

Library	uncompressed	zpipe	zstd
#Files	1004	1004	1004
Size (MB)	2300	999	866
Time (min)	-	1:36	0:15
Compression Ratio	-	2.30	2.66

Zstd saves 15% in space and is 6x faster than zpipe

Note:

- zpipe is CVMFS calling zlib
- Due to benchmark setup the exact factor of speed-up of zstd is not representative

Summary

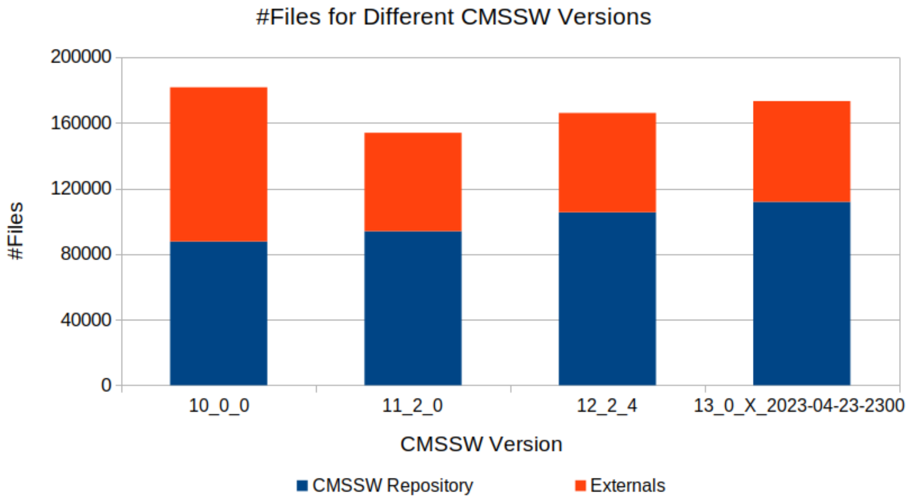
- CVMFS expects an order of magnitude growth in all metrics for HL-LHC
- Confident that the current design sustains the expected scale
- Rich set of performance and operational improvements underway to ensure proper quality of service at HL-LHC scales

- **Performance Improvements**
  - Symlink and `statfs` caching
  - Parallel decompression
  - Prefetching of known file clusters
  - `Zstd` compression
- **Operational Improvements**
  - Official cvmfs Helm chart
  - Proxy sharding

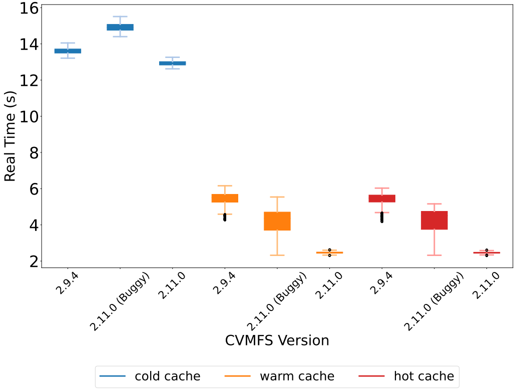
- **Publishing Improvements**
  - Feature parity between remote publishers and local publishers
- **General Improvements**
  - Housekeeping
  - Better documentation

Questions?

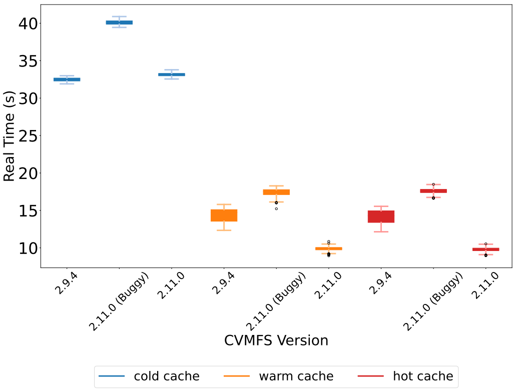
CMSSW: Increase of Number of Files



Some First Performance Comparison - Finding bugs



DD4hep



Tensorflow