Introduction to CernVM-FS

CernVM
File system

Mon 18 May 2026
EESSI Webinar

Valentin Volkl (CERN)

In collaboration with EESSI

Kenneth Hoste (UGent), et al.

EESSI
EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

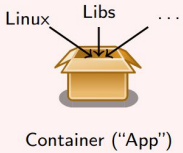
Agenda

- What is CernVM-FS?
- Quick introduction to EESSI (example CernVM-FS repository) *[hands-on]*
- CernVM-FS client installation + configuration *[hands-on]*
- CernVM-FS on large-scale systems: proxy, Stratum 1 *[hands-on]*
- Alternative access mechanisms for CernVM-FS repos *[hands-on]*
- Monitoring CernVM-FS
- Troubleshooting CernVM-FS
- Creating your own CernVM-FS repository
- Software startup performance

The problem with software distribution

Example: R in Docker

```
$ docker pull r-base
```
⟶ 1 GB image
```
$ docker run -it r-base
$ ... (fitting tutorial)
```
⟶ only 30 MB used



Working Set

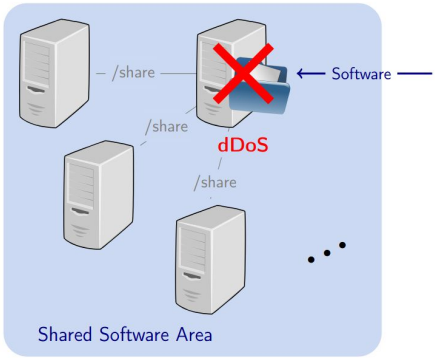
- Not more than $\mathcal{O}(100MB)$ of software requested for any task
- Very meta-data heavy:
  look for 1 000 shared libraries in 25 search paths

It's hard to scale:

iPhone App	Docker Image
20 MB	1 GB
changes every month	changes twice a week
phones update staggered	servers update synchronized

Flash Crowd Effect

- $\mathcal{O}(Mhz)$ meta data request rate
- $\mathcal{O}(khz)$ file open request rate




```
sed s/Docker/(Package Manager|VM|Tarball)/
```

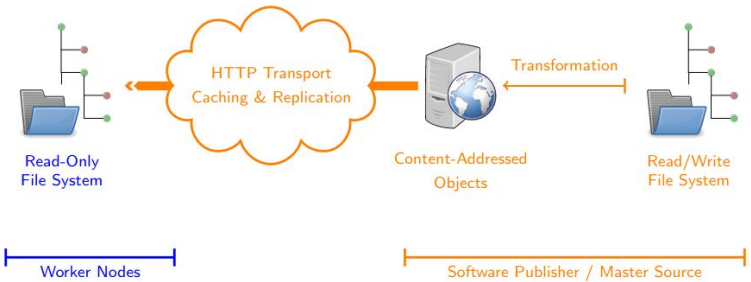
What is the CernVM FileSystem (CVMFS)?

- Global, **read-only filesystem** for **software distribution**
  - With a user experience similar to an on-demand streaming service (... but for scientific software)

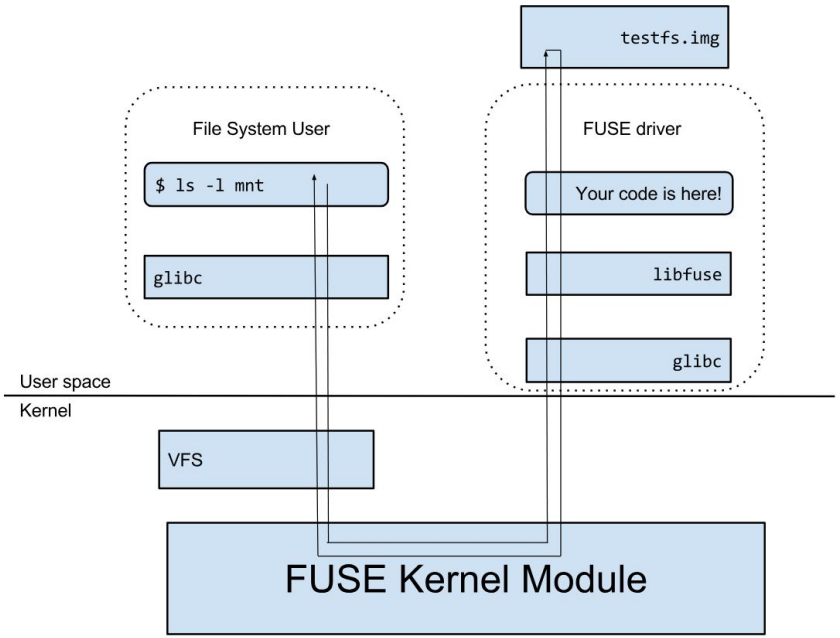
```
~$ ls /cvmfs
~$ ls /cvmfs/software.eessi.io # mounted automatically by autofs
repo
~$ ls /cvmfs/software.eessi.io
host_injections  init  README.eessi  versions
~$ cat /cvmfs/software.eessi.io/README.eessi # just-in-time download
EESSI - the European Environment for Scientific Software Installations

Getting started
---------------
```

CernVM FileSystem (CVMFS)

- Global, **read-only filesystem** for **software distribution**
  - With a user experience similar to an on-demand streaming service (... but for scientific software)
- Implemented as a filesystem in userspace, via *libfuse*
  - allows client to be installed flexibly on all workernodes
- Optimized for storing and distributing software
  - Content-adressable storage allows **de-duplication**
  - Multi-level **caching**, use of HTTP transport
  - **Compression** of data
  - Verification of data **integrity**
  - **...**



CVMFS is a Filesystem in Userspace

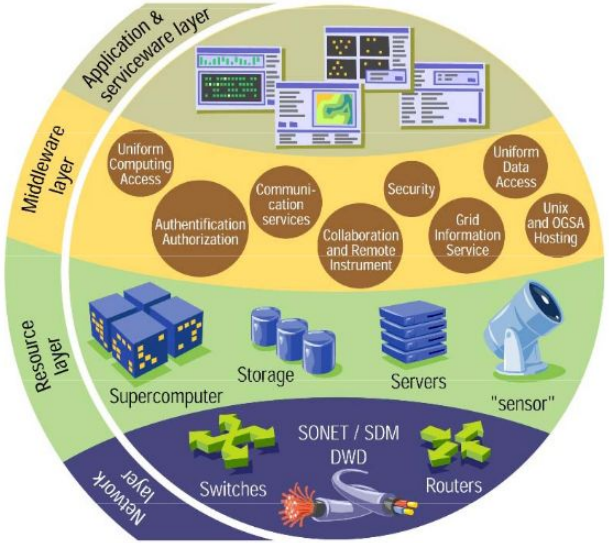


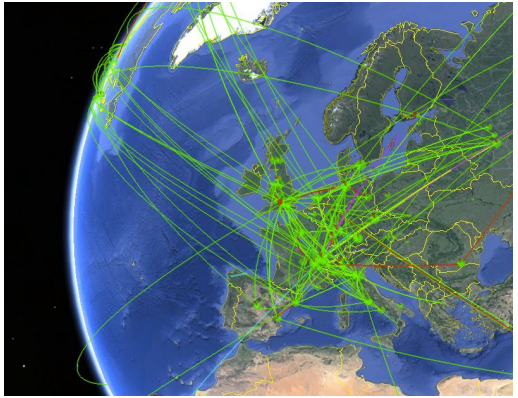
- Implements all necessary (ro) syscalls
- If file is in local cache: use that
- If file is not in local cache: download from object store and place it in local cache, use that

The Grid

CVMFS was originally developed for LHC “Computing Grid”

- Intended to provide uniform computing access to all resources pledged for the LHC



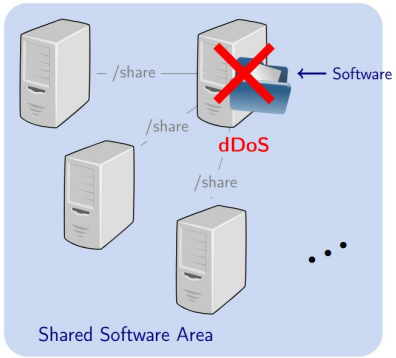


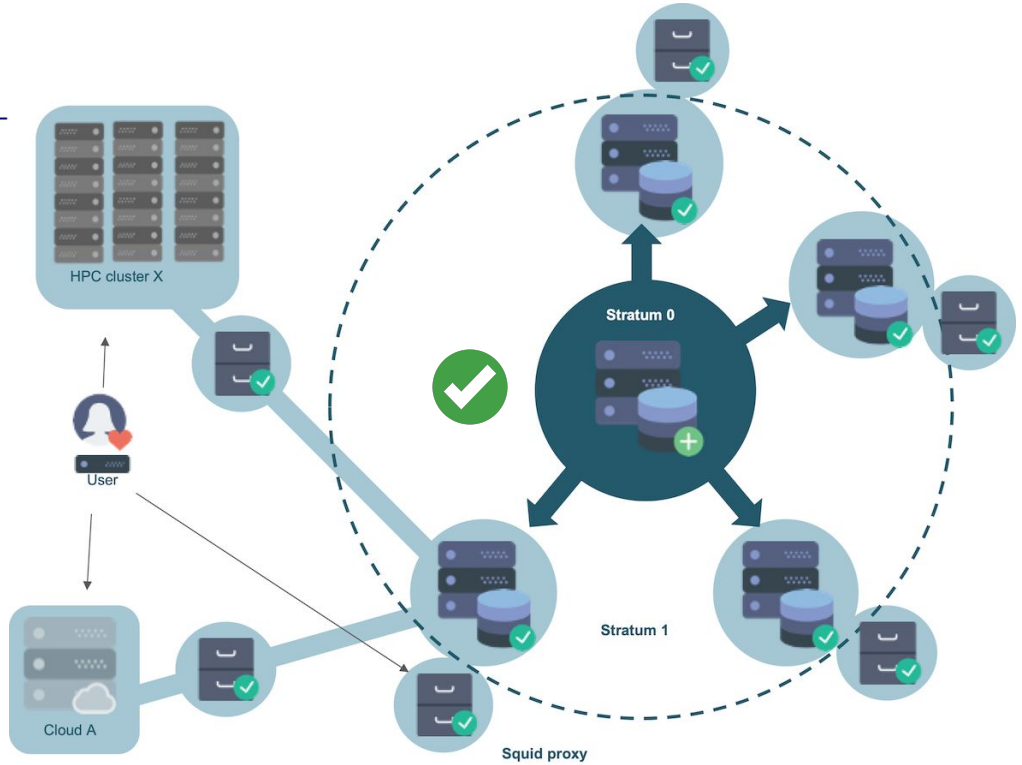
Working Set

- Not more than $\mathcal{O}(100MB)$ of software requested for any task
- Very meta-data heavy:
  look for 1 000 shared libraries in 25 search paths

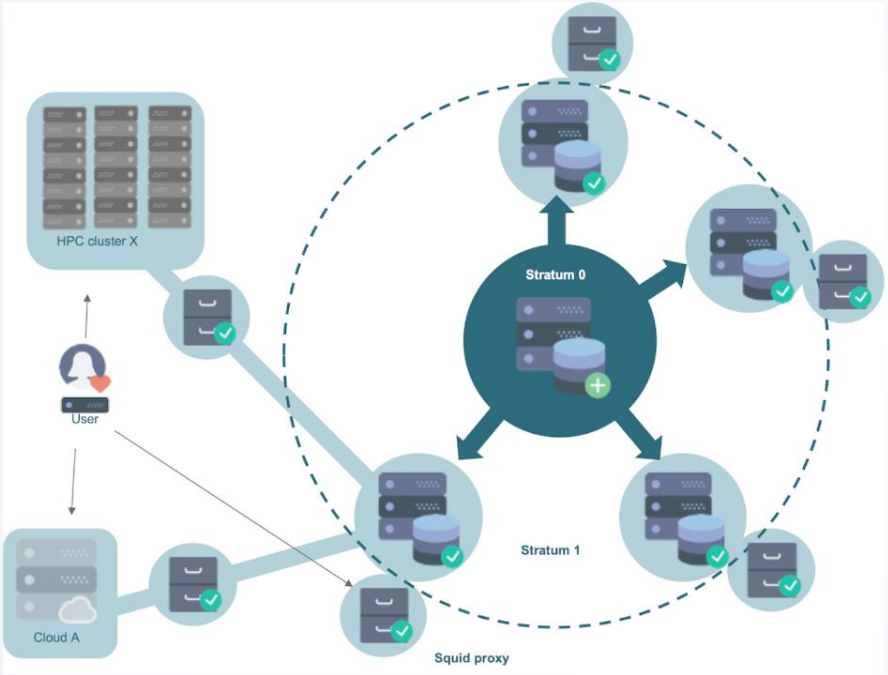
Flash Crowd Effect

- $\mathcal{O}(Mhz)$ meta data request rate
- $\mathcal{O}(khz)$ file open request rate





CVMFS — cache hierarchy



Stratum-0
Origin of the repo, owned by the librarian. Updated via publications with `cvmfs_server`

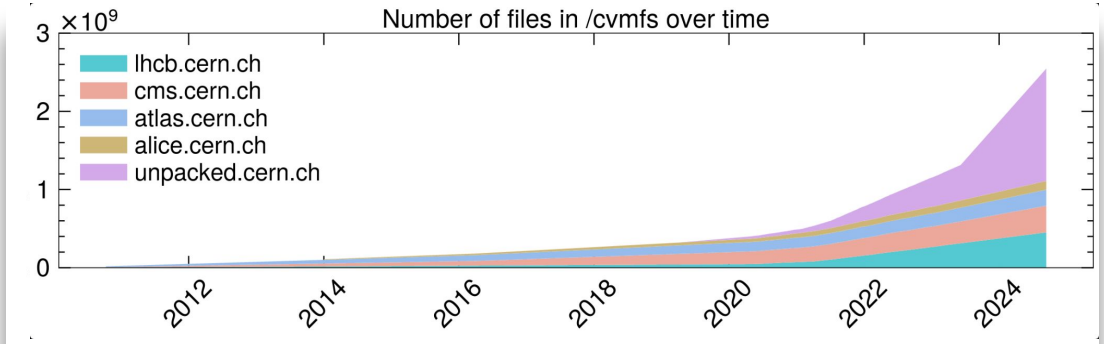
Stratum-1
Globally-distributed HTTP read-only full mirrors of the Stratum-0. Clients usetalk to the closest one.

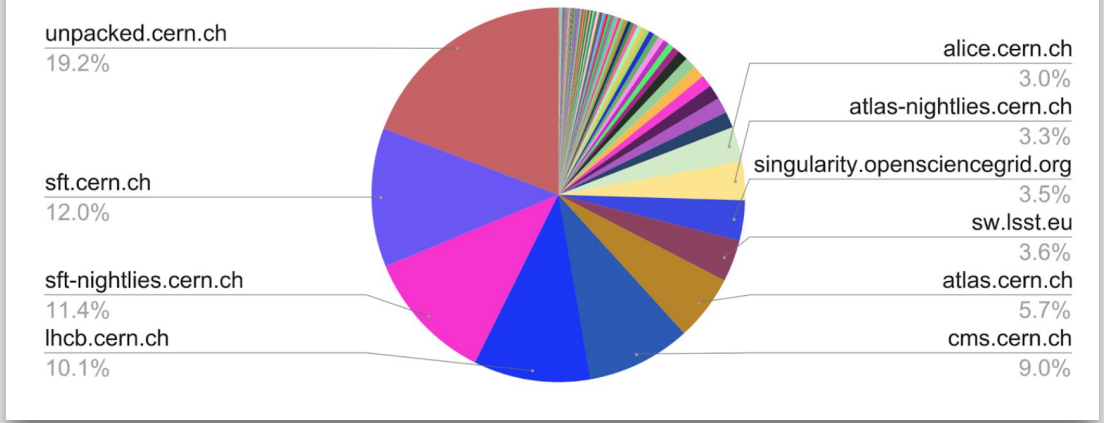
Squid proxy
Per-site forward proxy — absorbs duplicate requests when many clients on the same network start the same software.

Client disk cache
Per-node on-disk cache. First open of a file fetches it from the closest upstream tier and writes it here.

Kernel page cache
Linux page cache sits on top of the disk cache — warm reads don't hit cvmfs2 or the disk at all.

CVMFS (at CERN) in numbers

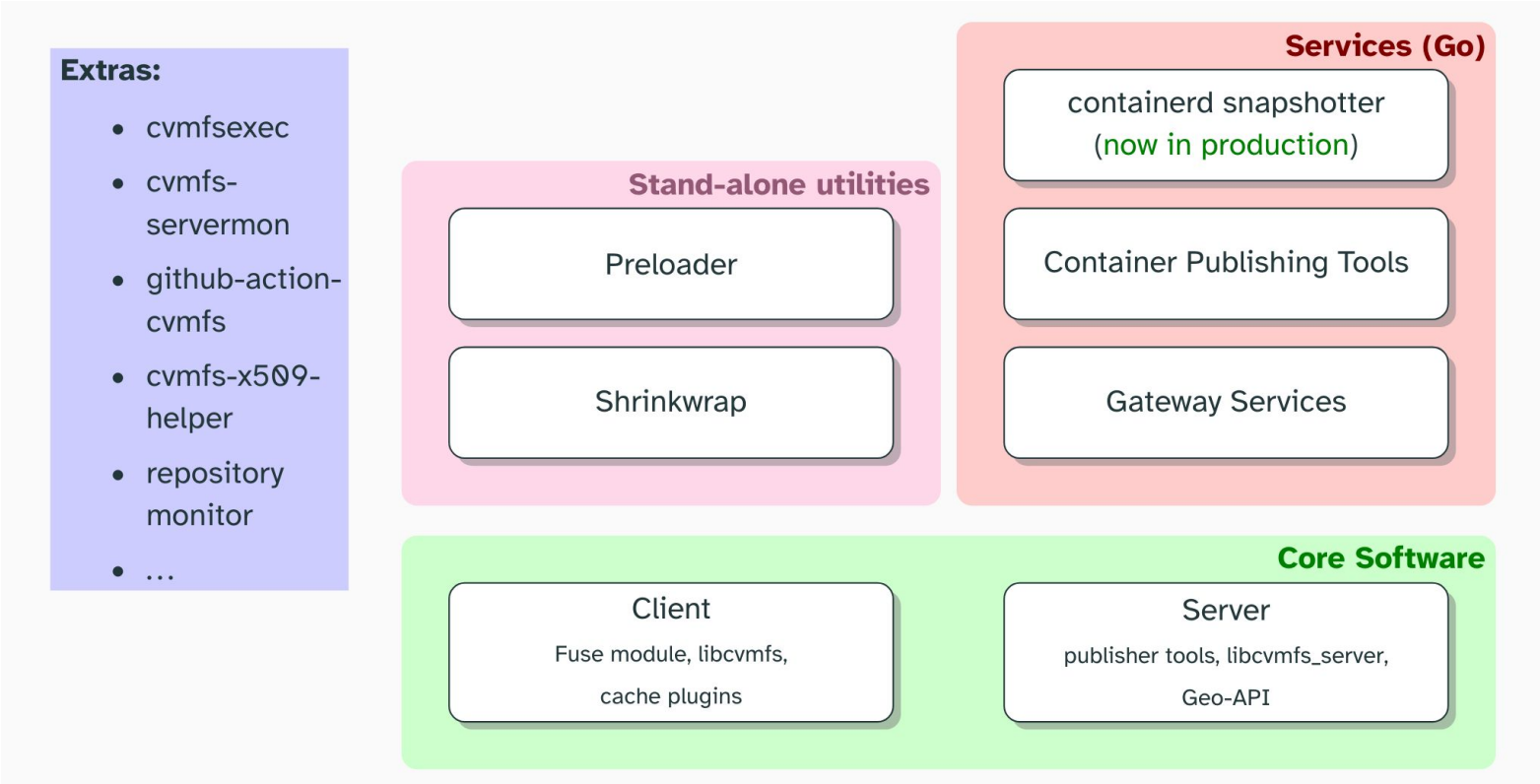




~ **15 Stratum 1s**
~ **> 33 B files** in the /cvmfs tree
~ **2 PB of data** accessible through /cvmfs
out of which ~1.5 PB in *external* files
proven to scale up to 100 PB
~ **> 8k container images**
~ ~~**260**~~ **293 repositories**

- Backed by S3(+CEPH) or local storage

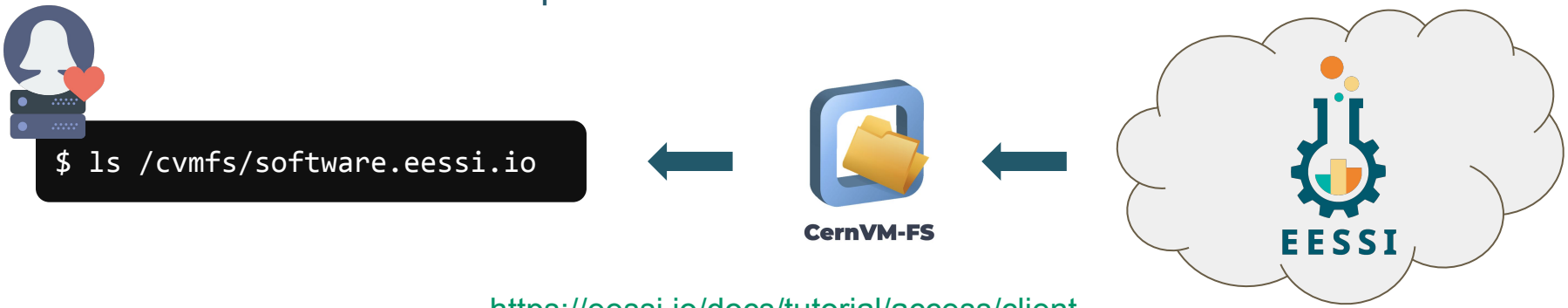
CernVM-FS Code and components



CernVM-FS client installation + configuration

To access a CernVM-FS repository, you need to:

1. Install the CernVM-FS client software
2. Configure it to:
   - Be aware of the CernVM-FS repositories you want to use
   - Specify how to connect to the CernVM-FS servers for those repos
   - Control max. disk space to use for CernVM-FS client cache



https://eessi.io/docs/tutorial/access/client

CernVM-FS client installation + configuration

1. Install the CernVM-FS client software

Packages are available for various CPU families & Linux distros (+ macOS), see https://cernvm.cern.ch/fs

On a RHEL-based operating system:

```
# install cvmfs-release package to add RPM repository
pkg=cvmfs-release-latest.noarch.rpm
pkg_url=https://ecsft.cern.ch/dist/cvmfs/cvmfs-release/$pkg
sudo dnf install -y $pkg_url

# install CernVM-FS client package
sudo dnf install -y cvmfs

# complete the setup
sudo cvmfs_config setup
```

https://eessi.io/docs/tutorial/access/client

CernVM-FS client installation + configuration

2. Configure CernVM-FS client: config repository

By default, CernVM-FS is aware of several repositories

via the CernVM-FS config repository (`cvmfs-config.cern.ch`).

Contents are managed via https://github.com/cvmfs-contrib/config-repo

This includes the necessary configuration to access the EESSI repositories:

```
$ ls /cvmfs/cvmfs-config.cern.ch/etc/cvmfs/domain.d | grep eessi
eessi.io.conf
$ ls /cvmfs/cvmfs-config.cern.ch/etc/cvmfs/keys/eessi.io
eessi.io.pub
```

https://eessi.io/docs/tutorial/access/client

CernVM-FS client installation + configuration

2. Configure CernVM-FS client: specific repositories

To make CernVM-FS aware of repositories, it needs to know:

- Hostnames/IPs of the **Stratum 1 servers** (in `/etc/cvmfs/domain.d/`)
- **Public key** for those repositories (in `/etc/cvmfs/keys/`)

For EESSI, you can simply install our `cvmfs-config-eessi` package:

```
$ rpm -q --filesbypkg cvmfs-config-eessi
cvmfs-config-eessi       /etc/cvmfs/domain.d/eessi.io.conf
cvmfs-config-eessi       /etc/cvmfs/keys/eessi.io/eessi.io.pub
```

It can be beneficial to not rely on CernVM-FS configuration repository…

https://eessi.io/docs/tutorial/access/client

CernVM-FS client installation + configuration

2. Configure CernVM-FS client: local configuration (proxy, cache, …)

Additional configuration files can be created under `/etc/cvmfs`, to specify:

- **Proxy server** (+ private Stratum 1) that this client should use
- Amount of **disk space** that can be used for the CernVM-FS **client cache**
- The **path** for the client cache (default: `/var/lib/cvmfs`)

See also https://cvmfs.readthedocs.io/en/stable/cpt-configure.html

```
$ cat /etc/cvmfs/default.local
CVMFS_HTTP_PROXY="http://my-proxy.example.com:3128"  # proxy server to use
CVMFS_QUOTA_LIMIT=10000      # max. 10GB for CernVM-FS client cache
CVMFS_CACHE_BASE=/ssd/cvmfs  # custom path for CernVM-FS client cache
```

https://eessi.io/docs/tutorial/access/client

CernVM-FS client installation + configuration

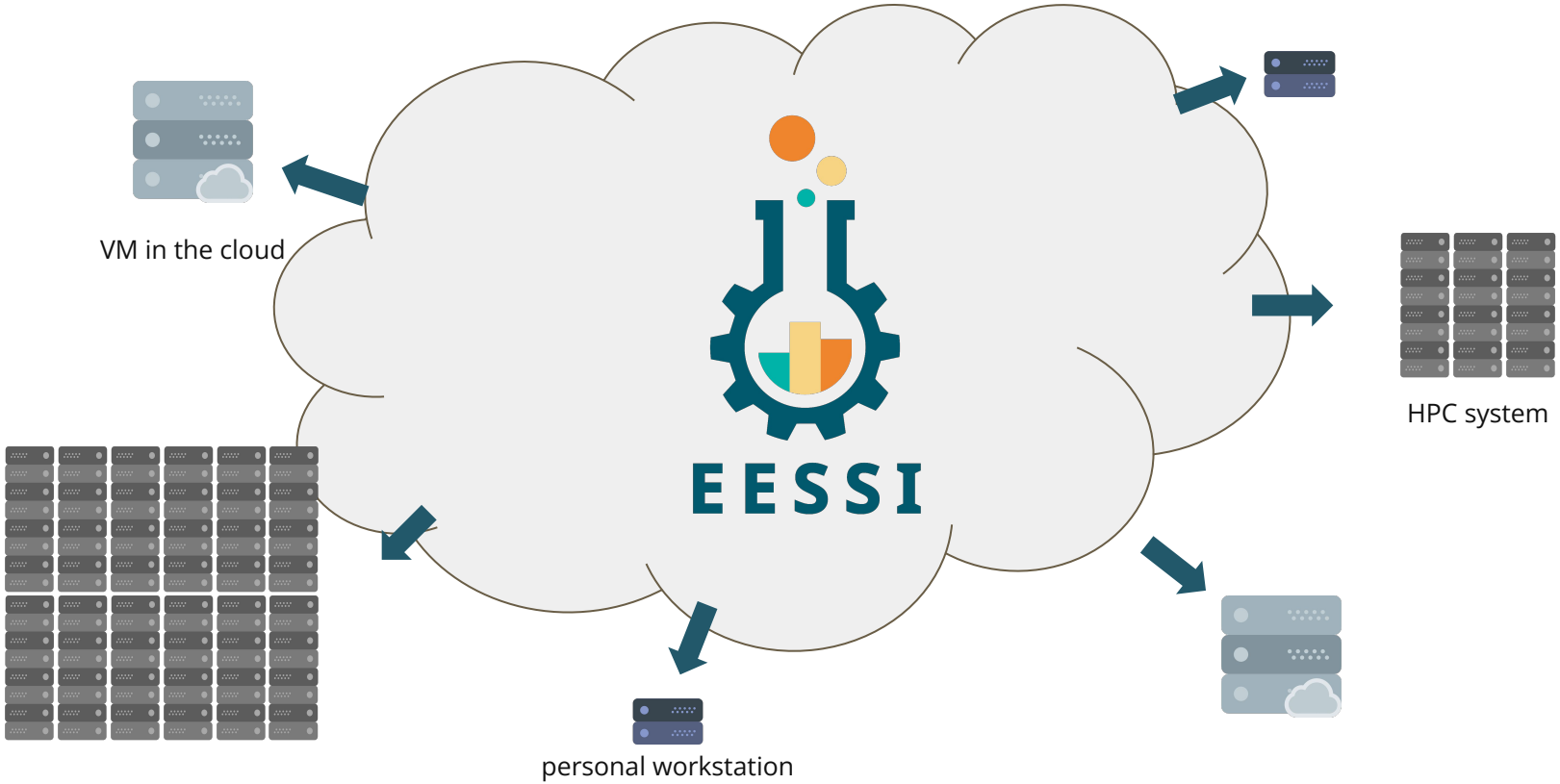
Hands-on demo: installing configuration for accessing EESSI repositories

```
# install package that provides EESSI configuration for CernVM-FS
pkg=cvmfs-config-eessi-latest.noarch.rpm
pkg_url=https://github.com/EESSI/filesystem-layer/releases/download/latest/$pkg
sudo dnf install -y $pkg_url

# create client configuration file for CernVM-FS
# (no squid proxy, 10GB local CernVM-FS client cache)
sudo bash -c "echo 'CVMFS_CLIENT_PROFILE="single"' > /etc/cvmfs/default.local"
sudo bash -c "echo 'CVMFS_QUOTA_LIMIT=10000' >> /etc/cvmfs/default.local"

# reload CernVM-FS client configuration
sudo cvmfs_config reload
```

https://eessi.io/docs/tutorial/access/client

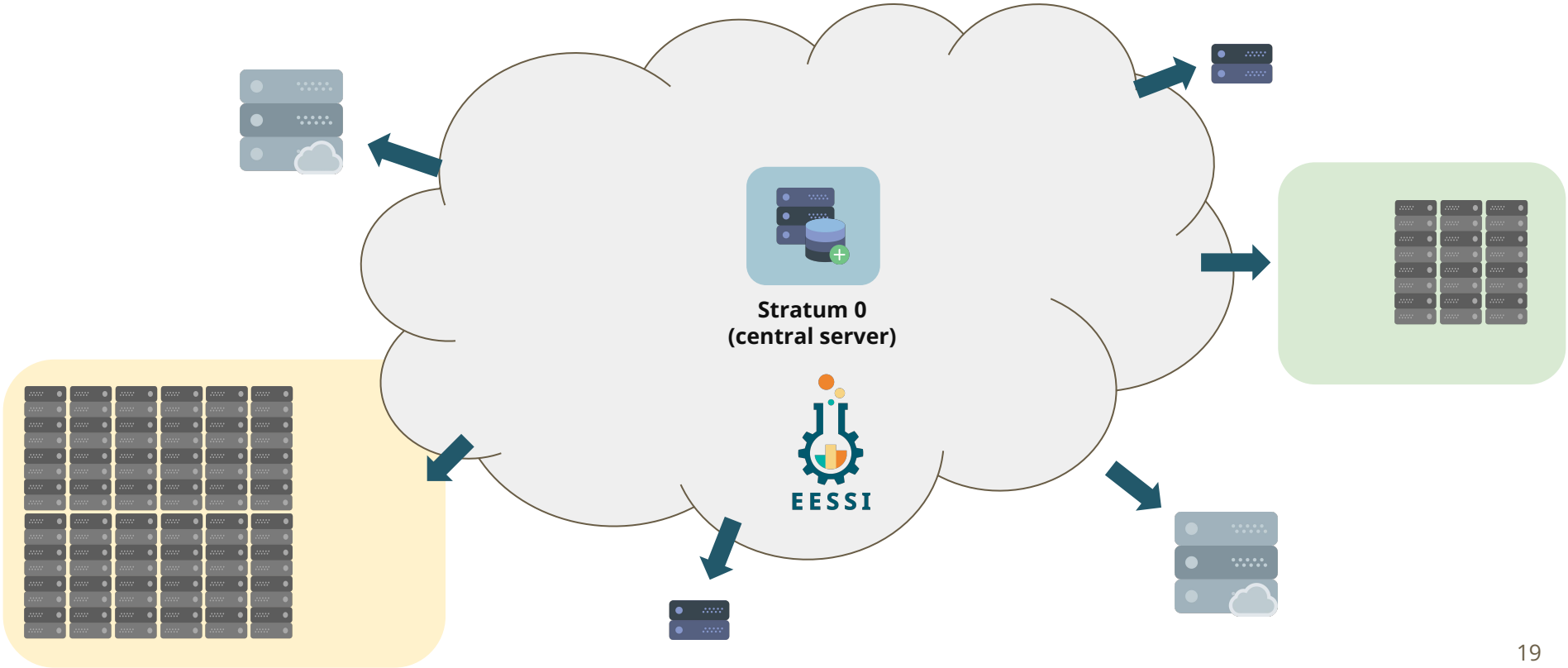
CernVM-FS on large scale systems

More effort is required when you go beyond a handful of clients…



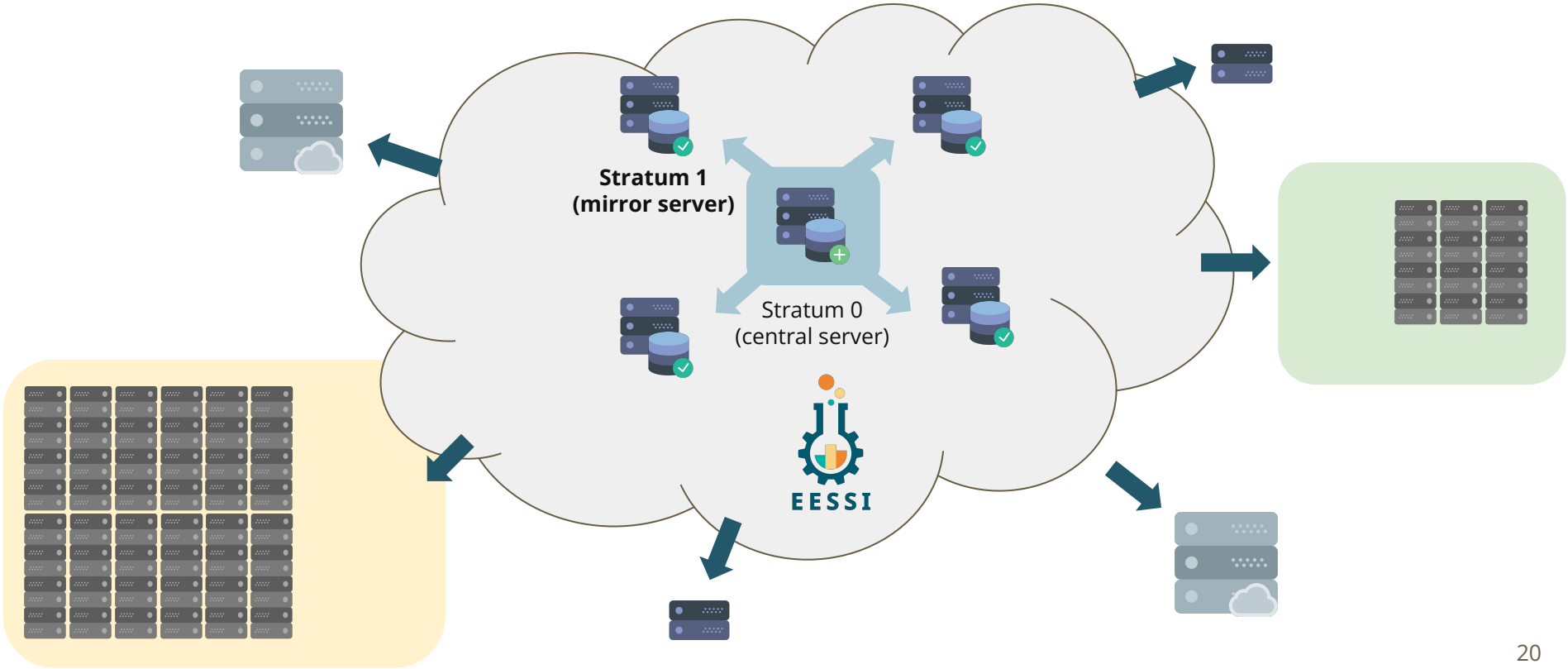
CernVM-FS on large scale systems

Network of public Stratum 1 “mirror” and proxy servers (+ private ones)



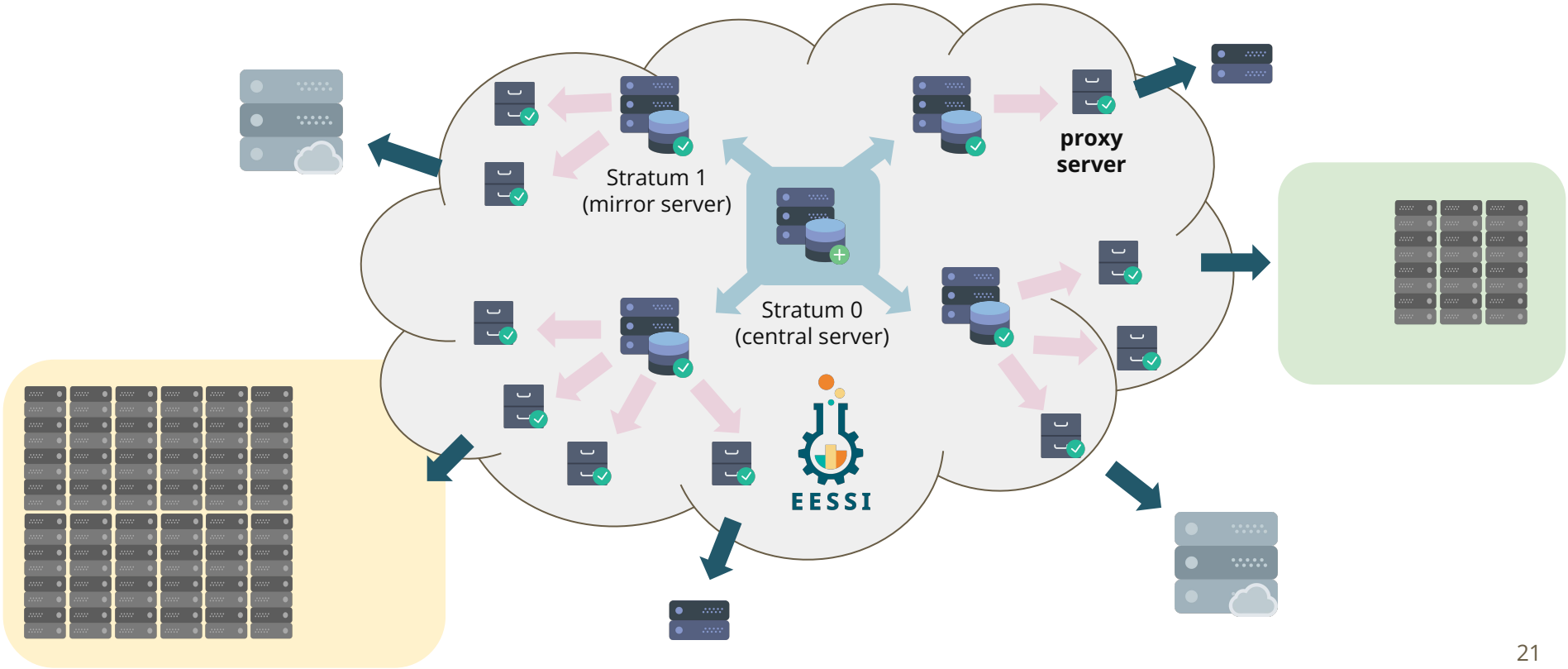
CernVM-FS on large scale systems

Network of public Stratum 1 “mirror” and proxy servers (+ private ones)



CernVM-FS on large scale systems

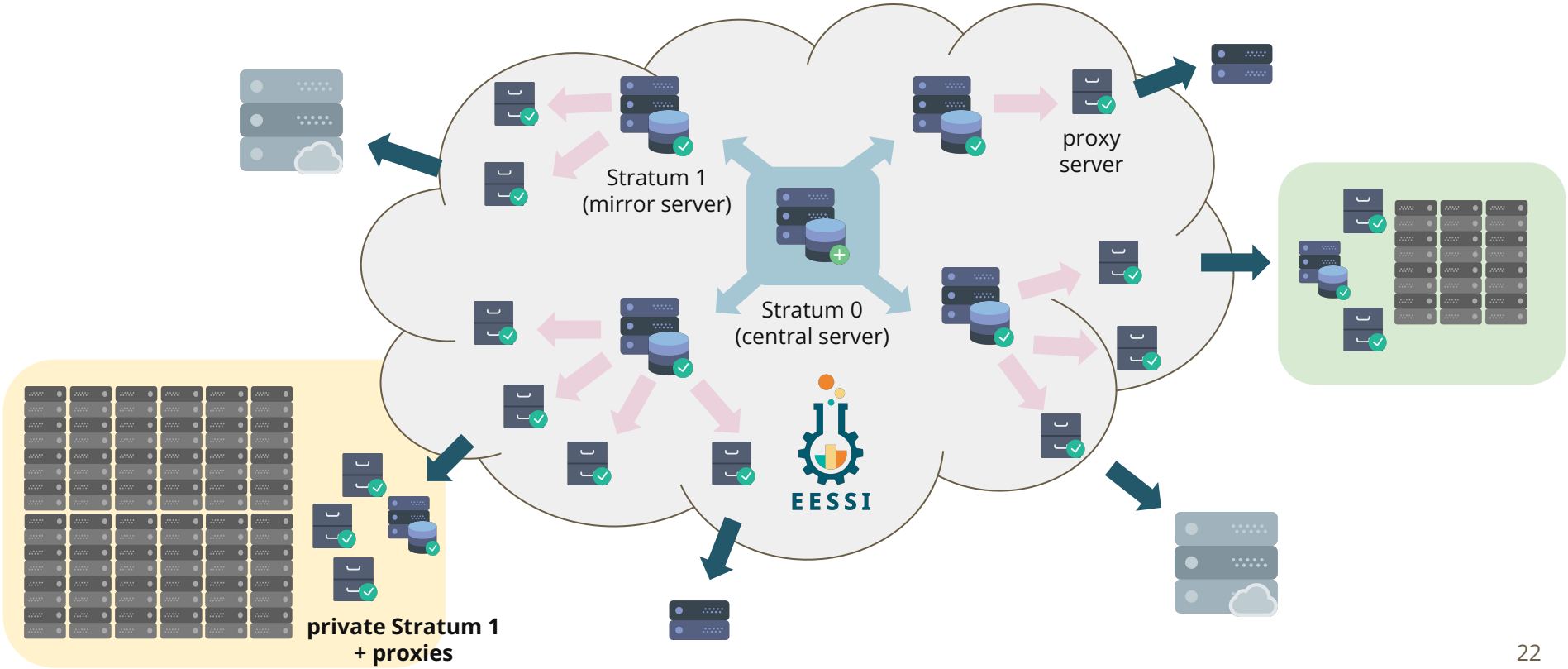
Network of public Stratum 1 “mirror” and proxy servers (+ private ones)



CernVM-FS on large scale systems

CernVM-FS

Network of public Stratum 1 “mirror” and proxy servers (+ private ones)



CernVM-FS on large scale systems

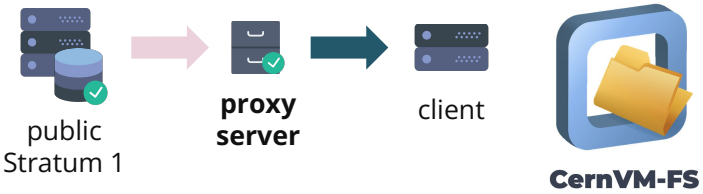
Proxy servers:

- Partial cache of CernVM-FS repositories being used on clients
- Close to the clients (ideally), helps with reducing network latency
- To offload the Stratum 1 mirror servers
- Usually Squid proxy, but alternatives are available (Varnish, …)

Stratum 1 mirror servers

- Full copy of contents of selected CernVM-FS repositories
- Close to the proxy servers, reduce network latency + protect against internet disconnect
- Periodic sync with Stratum 0 central server for selected CernVM-FS repositories (replicate)
- For EESSI: today ~360GB of disk space required, growing with ~1GB/day
    - Other CernVM-FS repositories can be *a lot* larger, see repositories @ CERN…

CernVM-FS on large scale systems



https://eessi.io/docs/tutorial/access/proxy

Installing and configuring a (Squid) proxy server

```
sudo dnf install -y squid       # install Squid proxy software
sudo vim /etc/squid/squid.conf  # configure Squid (network ACLs, disk space, ...)
sudo systemctl start squid      # start Squid service
```

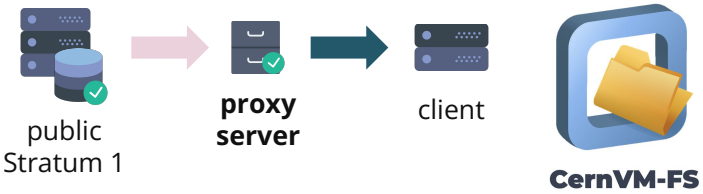
Re-configuring client to use a proxy server

```
$ grep PROXY /etc/cvmfs/default.local   # check proxy setting in client config
CVMFS_HTTP_PROXY="http://my-proxy.example.com:3128"
$ sudo cvmfs_config reload              # reload CernVM-FS client config
```

Requirements:

- Fast disk (SSD/NVME) + sufficient disk space (at least ~100GB for EESSI is OK)
- Fast and reliable network connection to clients
- Port on which proxy server is running must be accessible (default: 3128)

CernVM-FS on large scale systems



https://eessi.io/docs/tutorial/access/proxy

Example configuration for Squid proxy (`/etc/squid/squid.conf`)

```
acl local_nodes src YOUR_CLIENT_IPS          # IPs (or IP ranges) that can access proxy
# destination domains that are allowed (Stratum 1 servers)
acl stratum_ones dstdomain .cern.ch .opensciencegrid.org .eessi.science
http_port 3128                               # port for Squid proxy (must be accessible!)
http_access deny !stratum_ones               # deny all except Stratum 1 servers
http_access allow local_nodes                # allow local nodes
http_access allow localhost                  # allow local nodes
http_access deny all
minimum_expiry_time 0
maximum_object_size 1024 MB
cache_mem 1024 MB                            # proxy memory cache of 1GB
maximum_object_size_in_memory 128 KB
cache_dir ufs /var/spool/squid 100000 16 256  # 100 GB disk cache
```

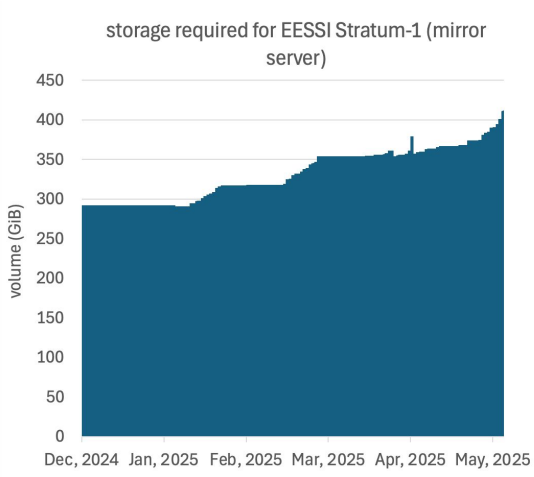
CernVM-FS on large scale systems

private
Stratum 1

https://eessi.io/docs/tutorial/access/stratum1

Installing a (private) Stratum 1 mirror server

```
# install cvmfs-release package to add RPM repository
pkg=cvmfs-release-latest.noarch.rpm
pkg_url=https://ecsft.cern.ch/dist/cvmfs/cvmfs-release/$pkg
sudo dnf install -y $pkg_url

# install CernVM-FS server package
sudo dnf install -y cvmfs-server

# install mod_wsgi Apache adapter module
sudo yum install -y python3-mod_wsgi
```



Requirements:

- Fast disk (SSD/NVME) + a lot of disk space (today: ~360GB for EESSI, grows by ~1GB/day)
- Fast and reliable network connection to proxy servers/clients
- Port 80 (HTTP) must be accessible

CernVM-FS on large scale systems

private
Stratum 1

CernVM-FS

https://eessi.io/docs/tutorial/access/stratum1

Adding repositories to a (private) Stratum 1 mirror server

```
# add public key for CernVM-FS repositories to mirror (in this case EESSI)
sudo mkdir -p /etc/cvmfs/keys/eessi.io/
sudo cp eessi.io.pub /etc/cvmfs/keys/eessi.io/

# disable GeoAPI service (not relevant for private Stratum 1)
echo 'CVMFS_GEO_DB_FILE=NONE' | sudo tee -a /etc/cvmfs/server.local

# create replica of a CernVM-FS repository
sync_server='aws-eu-west-s1-sync.eessi.science'
repo='software.eessi.io'; key_dir='/etc/cvmfs/keys/eessi.io'
sudo cvmfs_server add-replica -o $USER http://${sync_server}/cvmfs/${repo} ${key_dir}

# initial synchronisation of replica of a CernVM-FS repository
cvmfs_server snapshot software.eessi.io   # this will take several hours...
```

CernVM-FS on large scale systems

https://eessi.io/docs/tutorial/access/stratum1

CernVM-FS

Re-configuring client to use a private Stratum 1 without proxy *(not recommended!)*

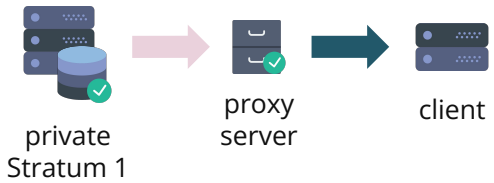
private
Stratum 1

```
# check Stratum-1 servers for eessi.io
$ grep SERVER /etc/cvmfs/domain.d/eessi.io.local
CVMFS_SERVER_URL="http://my-stratum1.example.com/cvmfs/@fqrn@"

# reload CernVM-FS client config
$ sudo cvmfs_config reload
```

CernVM-FS on large scale systems

https://eessi.io/docs/tutorial/access/stratum1

Re-configuring proxy + client to use a private Stratum 1 + proxy



Reconfigure proxy to serve as proxy for private Stratum 1:

```
# check ACL for Stratum 1 severs in proxy configuration
$ sudo grep 'acl stratum_ones' /etc/squid/squid.conf
acl stratum_ones dstdomain .cern.ch .opensciencegrid.org .eessi.science my-stratum1.example.com
# reload Squid proxy to pick up changes to configuration
$ sudo systemctl reload squid
```

Make sure client is configured to use proxy server:

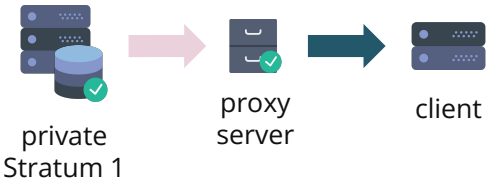
```
$ grep PROXY /etc/cvmfs/default.local   # check proxy setting in client config
CVMFS_HTTP_PROXY="http://my-proxy.example.com:3128"
$ sudo cvmfs_config reload              # reload CernVM-FS client config
```

CernVM-FS on large scale systems

https://eessi.io/docs/tutorial/access/stratum1

CernVM-FS

Verify setup and get some usage info via `cvmfs_config stat`




```
$ cvmfs_config stat -v software.eessi.io
Version: 2.12.7.0
PID: 24262
Uptime: 0 minutes
Memory Usage: 38372k
File Catalog Revision: 5348 (expires in 3 minutes)
File Catalog ID: c01b8e2895c593131ddeb1dbd7894a4744b28960
No. Active File Catalogs: 55
Cache Usage: 951227k / 10240001k
File Descriptor Usage: 0 / 130560
No. Open Directories: 0
No. IO Errors: 0
Connection: http://<S1_IP>/cvmfs/software.eessi.io through proxy http://<PROXY_IP>:3128 (online)
Usage: 4875 open() calls (hitrate 26.441%), 533 opendir() calls
Transfer Statistics: 271939k read, avg. speed: 29587k/s
```

Alternative access mechanisms for CernVM-FS repos

On some systems, using CernVM-FS can be a bit more challenging

- Offline or diskless workernodes in HPC clusters
- CernVM-FS client can not be installed

Various **alternative mechanisms** to access CernVM-FS repositories,
or to use the software they provide, are available…

Being creative with location of CernVM-FS client cache, squashfs images,
using a container image that includes CernVM-FS client, cvmfsexec tool, …

Going beyond the recommended setup comes with tradeoffs.

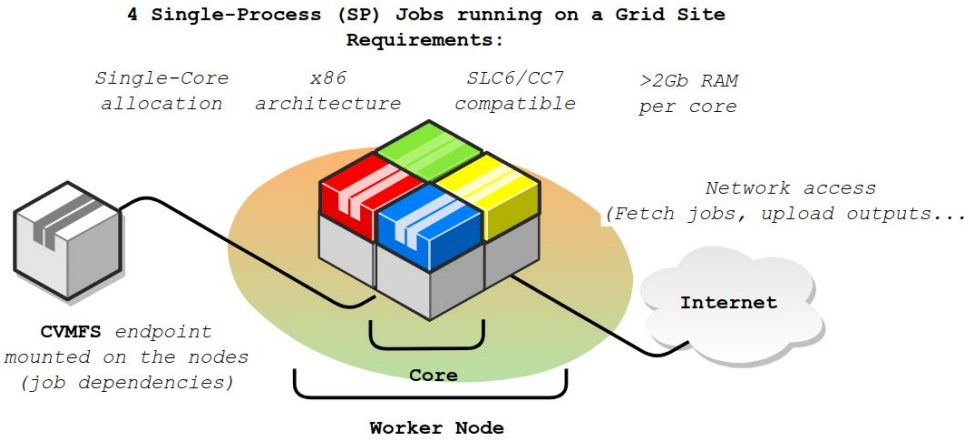
https://eessi.io/docs/tutorial/configuration_hpc

Accessing CernVM-FS repositories on HPC systems

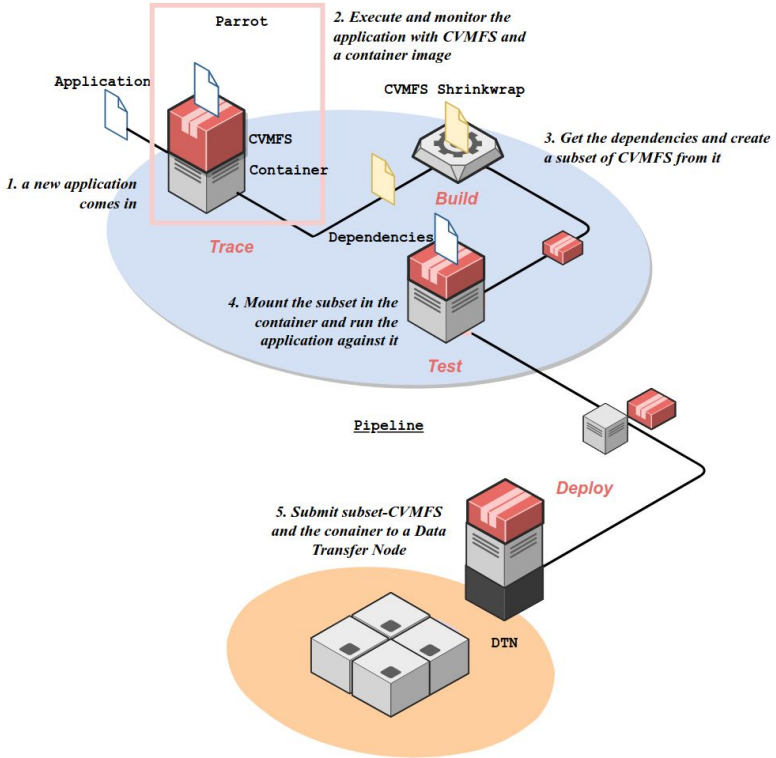
HPC sites may impose many restrictions.

Workarounds for many configurations exist, but come at different levels of cost.

Best case:



Worst case:

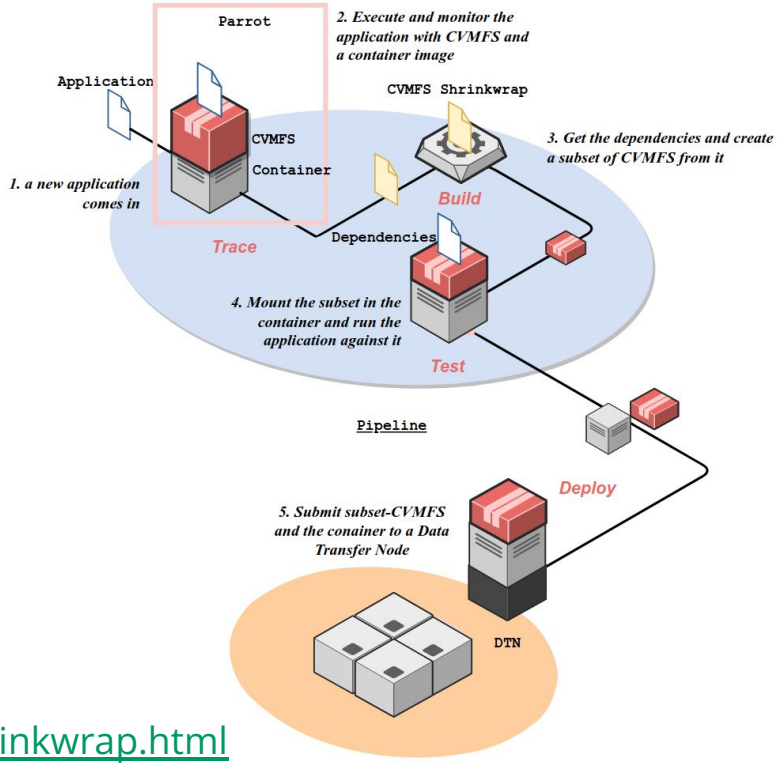


Alexandre F Boyer [CHEP 2023]

CVMFS Shrinkwrap

For when you have no network connection, or no CernVM-FS client on the nodes!

- Copies contents of CernVM-FS repo to a local file system
  - rsync, but more efficient (keeps deduplication)
- Some LHC experiments have built a whole framework to trace the file accesses of specific jobs and shrinkwrap only that
- Similar: `cvmfs_preload`
- 👎 Labor intensive



https://cvmfs.readthedocs.io/en/stable/cpt-shrinkwrap.html

Advanced cache configurations

For when you have no local disks!

- Use Loopback file system on cache
  - One file per repository
  - Easier on the metadata servers of cluster file system
- Use RAM cache or Tiered Cache
  - Example: https://cvmfs.readthedocs.io/en/stable/cpt-configure.html#example
- Workarounds no longer recommended:
  - NFS exports
    - Some HPC sites have tried running the cvmfs client on just one server and exporting to worker nodes over NFS. These installations can be made to work, but they are very inefficient, and often run into operational problems.
  - Parrot Connector

https://eessi.io/docs/tutorial/configuration_hpc

cvmfsexec

For when you have no admin privileges!

Tool by Dave Dykstra for mounting CernVM-FS repositories as an unprivileged user, without the cvmfs package being installed by a system administrator.

- 4 modes, depending on availability of certain features on the host
  - Fusermount
  - Unprivileged namespace fuse mounts
  - Setuid installation of Singularity/Apptainer >= 3.4
- 👎 No shared cache on system

https://github.com/cvmfs/cvmfsexec

Quick demo with cvmfsexec

https://www.eessi.io/docs/blog/2024/06/28/espresso-portable-test-run-eurohpc

Step 1: Installing cvmfsexec *(does not require admin privileges!)*

```
cd ~
git clone https://github.com/cvmfs/cvmfsexec.git
cd cvmfsexec
./makedist default
```

Step 2: access CernVM-FS repository

(requires user namespace support, or other supported mechanism)

```
cvmfsexec software.eessi.io -- /bin/bash -l
```

OK for single-node workloads, feasible but tricky for multi-node workloads…

Monitoring CernVM-FS

A good monitoring setup makes sure the system is running smoothly,
and lets you tweak performance.

Some recommendations for monitoring:

- Health Checks and Syslog warnings
- Frequency of cache cleanups: If too frequent (< length of typical job),
  your cache is probably too small.
  # cache cleanups in the last 24 hours can be accessed via:

```
cvmfs_talk -i example.repo.org ncleanup24
```

- Current revision (to see possible lags to stratum-1)

https://eessi.io/docs/tutorial/monitoring

Monitoring CernVM-FS - Existing solutions

- Prometheus Exporter for CernVM-FS Clients:
  - https://github.com/cvmfs-contrib/prometheus-cvmfs
- Collectd Plugin to Monitor CernVM-FS Clients :
  - https://github.com/cvmfs/collectd-cvmfs
- CernVM File System Server Monitoring API
  - https://github.com/cvmfs-contrib/cvmfs-servermon
- Nagios check_cvmfs.sh script (builtin)
  - https://github.com/cvmfs/cvmfs/blob/devel/add-ons/check_cvmfs.sh
- Influx Telemetry Aggregator (builtin)
  - https://cvmfs.readthedocs.io/en/stable/cpt-telemetry.html

https://eessi.io/docs/tutorial/monitoring

Troubleshooting CernVM-FS

https://eessi.io/docs/tutorial/troubleshooting

Troubleshooting CernVM-FS

When using a CernVM-FS repository doesn't work, you should check for:

- Problems with the CernVM-FS client installation or **configuration**
  - `ls /cvmfs`, `sudo cvmfs_config reload`, `cvmfs_config showconfig`, `sudo cvmfs_talk -i`, `cvmfs_config probe`, check in `/etc/cvmfs/*`
- **Connectivity issues**: firewall, network ACLs, ports (80 for HTTP, 3128 for proxy), …
  - Output of `sudo cvmfs_talk -i`, use standard tools (`nc`, `telnet`, `curl`, `iperf3`, …)
- **Mounting problems**: `autofs` (auto-unmount), debug via manual mounting
- **Insufficient resources**: memory, disk space, network latency/bandwidth, …
- Problems with CernVM-FS client **cache** (out of disk space, corruption, …)
  `sudo cvmfs_config wipecache`
- **Log messages** in syslog (`/var/log/messages`), extended attributes on `/cvmfs/<repo>`

https://eessi.io/docs/tutorial/troubleshooting

CernVM-FS error message interpretation

Since it is a filesystem, CernVM-FS is limited in how it can report errors

- More details can (sometimes) be found in the syslog!

- **“Too many levels of symbolic links”**
  - Attempts to access a mount point within a namespace without shared/rshared
  - Usually a container! => Use `-v /cvmfs/cvmfs:shared`
- **“No such file or directory” (on a path you’d expect to be there)**
  - This can happen when new CernVM-FS catalogs (where metadata is stored) can not be loaded into the cache. Check the syslog!

https://eessi.io/docs/tutorial/troubleshooting

CernVM-FS error message interpretation

Since it is a filesystem, CernVM-FS is limited in how it can report errors

- More details can (sometimes) be found in the syslog!

- **“Software caused connection abort”**
  - The FUSE connection has been cut by the kernel (this usually only happens by admin intervention). Need to remount.
- **“Transport endpoint is not connected”**
  - The `cvmfs2` process has died, and the repository needs to be unmounted and remounted. Usually the “watchdog” process tries to do this automatically, but in this scenario it crashed too.

https://eessi.io/docs/tutorial/troubleshooting

Creating a CernVM-FS repository

You can create your own CernVM-FS repository

- To host the central software stack for your HPC cluster(s)
- To host software built on top of existing CernVM-FS repositories (like EESSI)

You will need to:

- Have a central Stratum 0 server to host your repository on
- Replicate your repository on Stratum 1 mirror server(s) + update proxy configurations
- Update the CernVM-FS configuration on clients to be aware of your repository
- Publish updates of the repository contents,
  and make sure that Stratum 1 mirror servers synchronize regularly

https://eessi.io/docs/tutorial/creating-repo
https://cvmfs.readthedocs.io/en/stable/cpt-repo.html

Creating a CernVM-FS repository

To create your repository on your Stratum 0 server:

```
cvmfs_server mkfs example.domain.tld
```

Adding contents is done through publishing, which is transactional:

```
cvmfs_server transaction
# make changes to repo (add software)
cvmfs_server publish
```

For large repositories, you should look into creating nested catalogs, garbage collection, …

https://eessi.io/docs/tutorial/creating-repo
https://cvmfs.readthedocs.io/en/stable/cpt-repo.html

Software startup performance

- CernVM-FS was designed for software installations
  (but can also be used for data)
- This is clearly reflected in the startup performance of software,
  when comparing with other filesystems often used to host software installations

Let's take a look at startup performance of TensorFlow provided by EESSI:

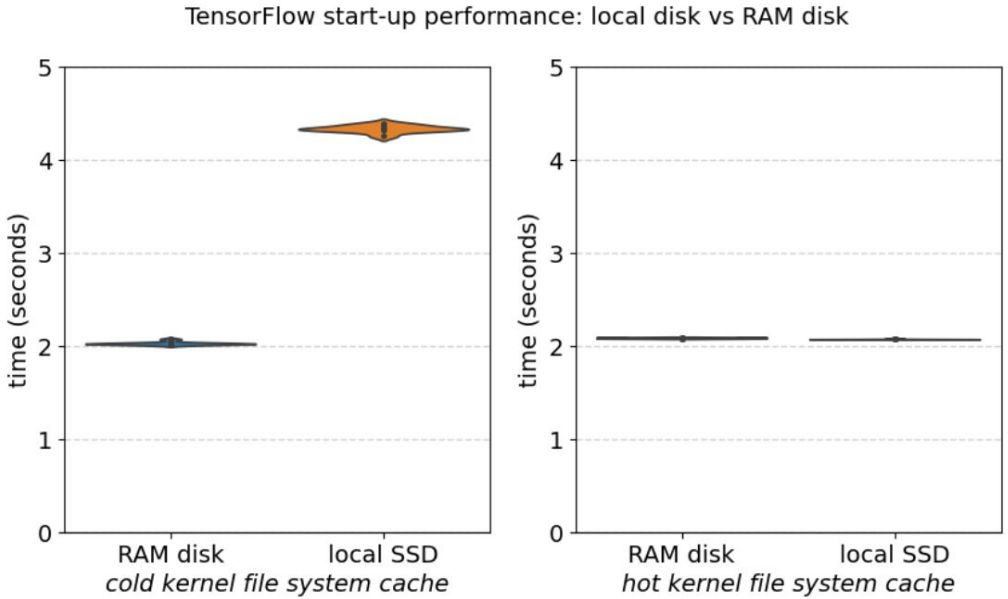
- `module load TensorFlow/2.13.0-foss-2023a`
  `python -c 'import tensorflow'`
- Requires ~3,500 files, totalling ~1.1GB of data (based on CernVM-FS stats)
- Mostly `*.pyc` files, but also binaries, shared libraries, …

https://eessi.io/docs/tutorial/performance

Software startup performance

Point of reference: full software stack installed on ramdisk or local disk (SSD)

(not a realistic scenario in practice)

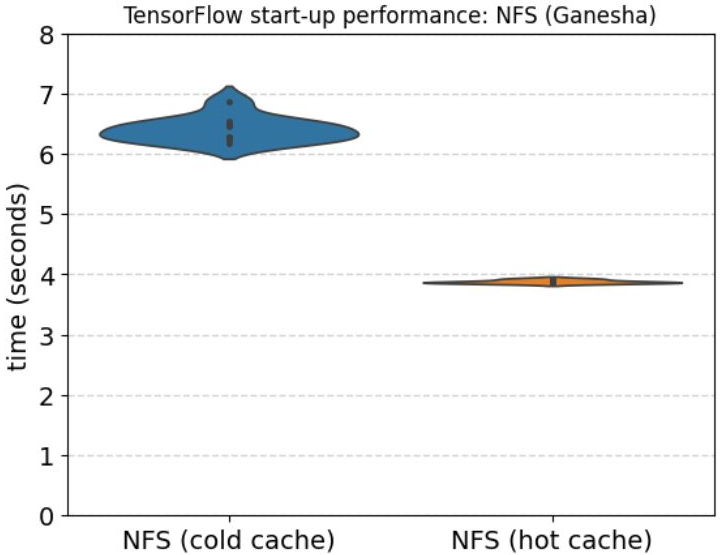


- At least 2 seconds required to complete import of `tensorflow` Python package
- Twice as long when necessary files are not in kernel filesystem cache yet

https://eessi.io/docs/tutorial/performance

Software startup performance

NFS filesystem, mostly unloaded

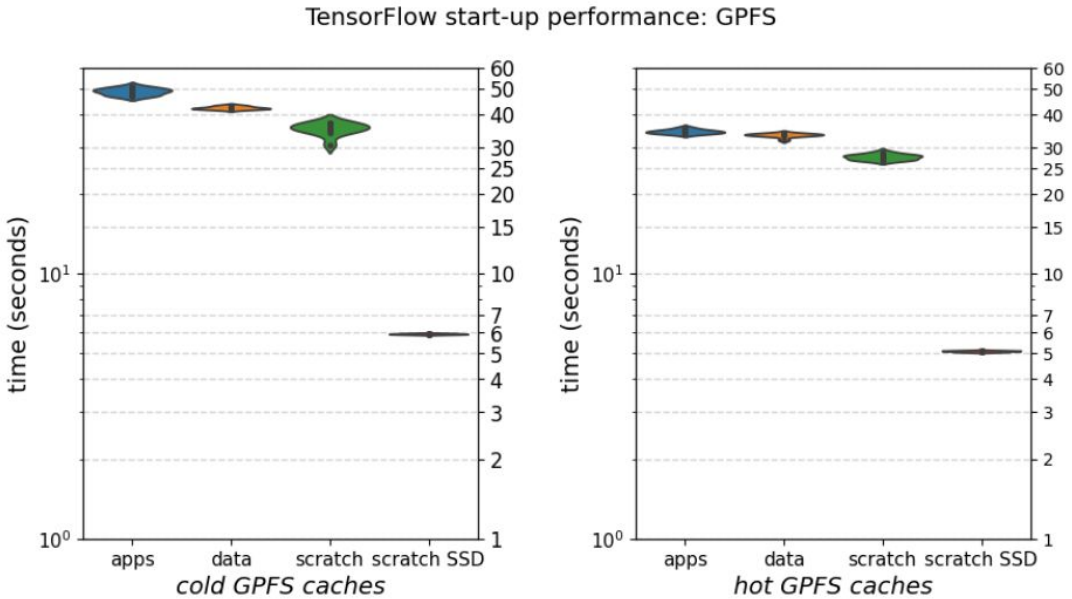


https://eessi.io/docs/tutorial/performance

Software startup performance

IBM Storage Scale (a.k.a GPFS) and Lustre filesystems

are typically not well tuned for workloads that involve accessing lots of small files

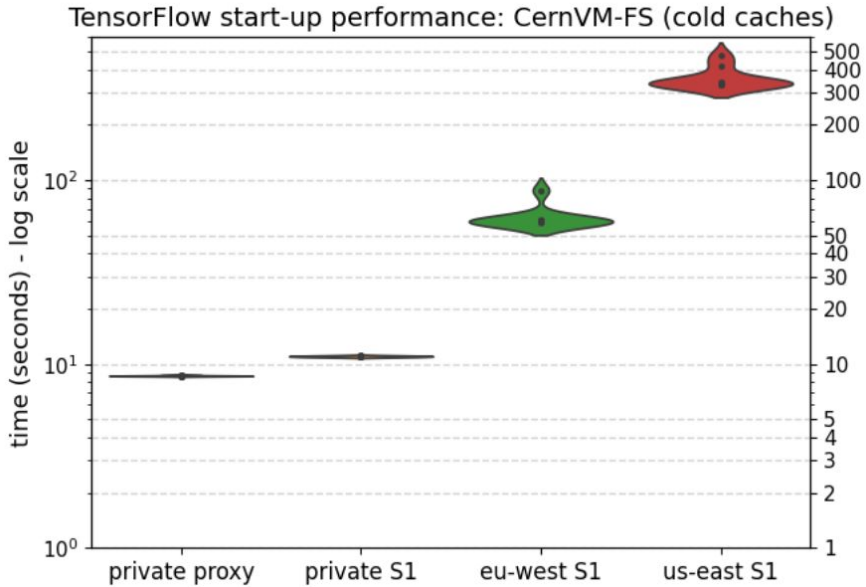


- 30-60 seconds required for importing tensorflow Python package,
  even when GPFS pagepool is “hot”
- Significantly better when backed by SSDs

https://eessi.io/docs/tutorial/performance

Software startup performance

Different CernVM-FS configurations, with cold CernVM-FS client cache (worst-case scenario)



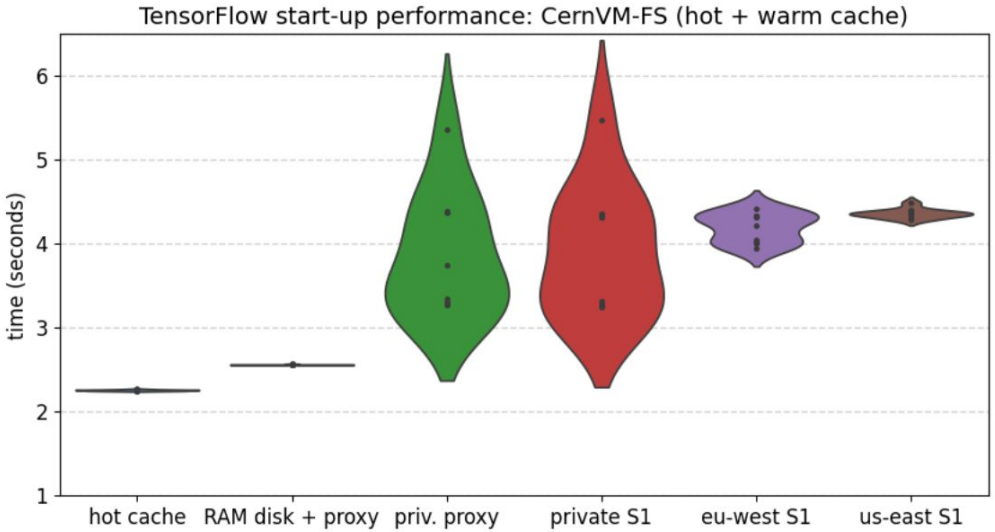
- Dramatically better when Stratum 1 mirror server is close (in network)
- Can be a lot worse if Stratum 1 is far away within no proxies closeby (due to network latency + limited bandwidth)

https://eessi.io/docs/tutorial/performance

Software startup performance

Different CernVM-FS configurations, different scenarios:

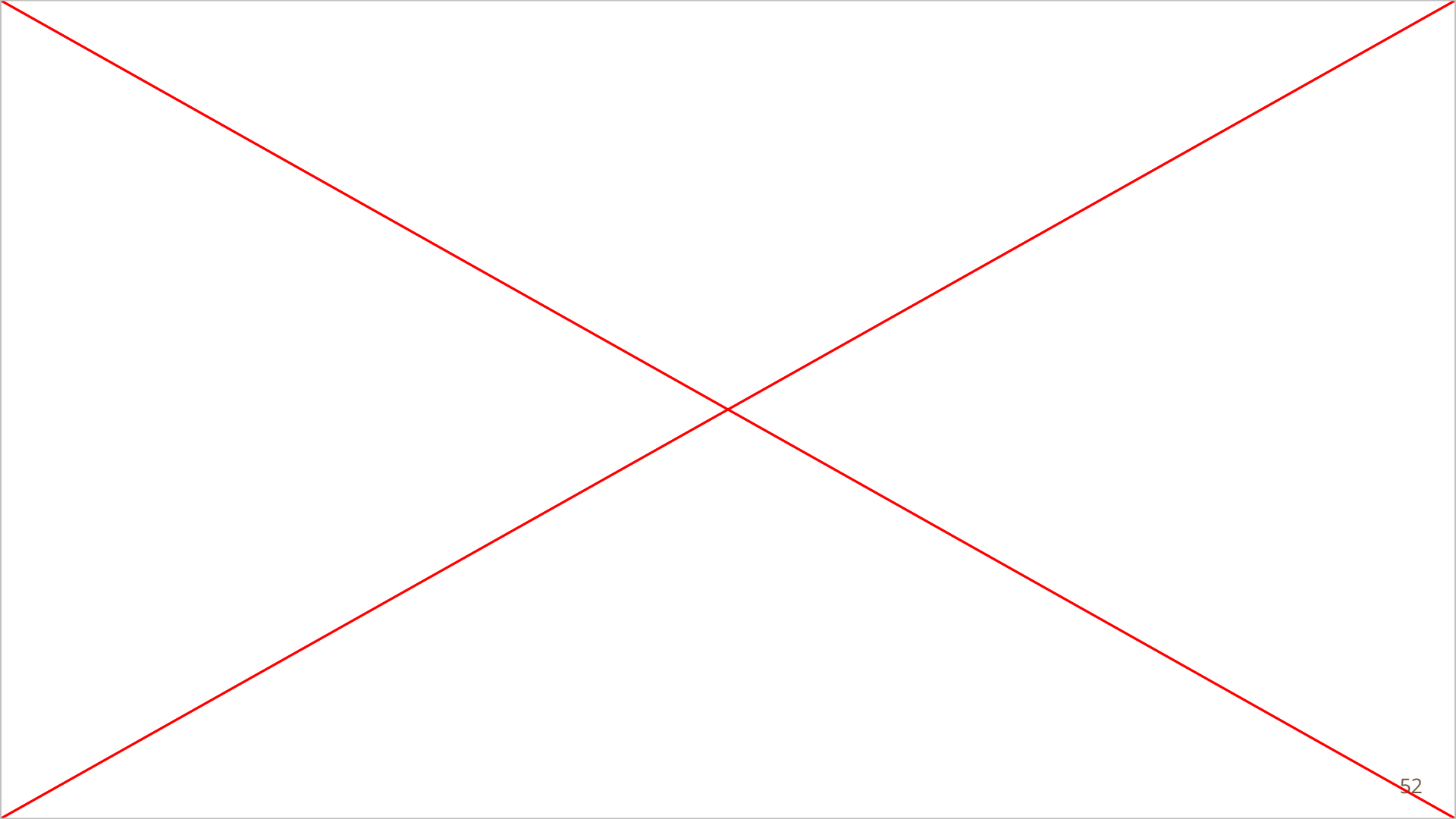
- hot cache (kernel filesystem cache)
- warm cache (CernVM-FS client cache)



- CernVM-FS client cache dramatically improves software startup performance
- Even when Stratum 1 mirror servers are far away

https://eessi.io/docs/tutorial/performance

CVMFS and container images



Outlook on new Developements

All currently being worked on

- Kubernetes Deployments: Helm Charts for Server Deployments
- Varnish cache configurations
- High Availability Gateway Groups
- File bundles (reduce latency in interactive use)
- Partial Stratum-1 replication

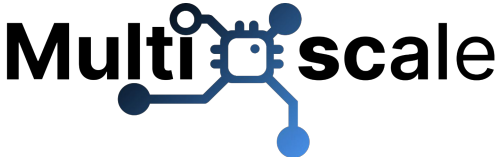
Acknowledgements

CernVM
File system

EESSI
EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

Kenneth Hoste (HPC-UGent)
Thomas Röblitz (Univ. of Bergen)
Bob Dröge (Univ. of Groningen)

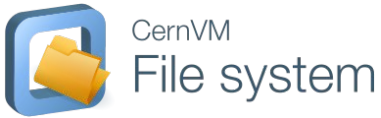
Valentin Völkl (CERN)
Georgios Christodoulis (CERN)
Jakob Blomer (CERN)



EuroHPC
Joint Undertaking

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and countries participating in the project under grant agreement No 101093169.

Questions?



cernvm.cern.ch/fs | docs | GitHub | forum | Mattermost (chat)

eessi.io | docs | GitHub | Slack | YouTube | support | Twitter/X

www.multixscale.eu | YouTube | Twitter/X | LinkedIn