CVMFS: Software Access Anywhere

Dan Bradley dan@hep.wisc.edu

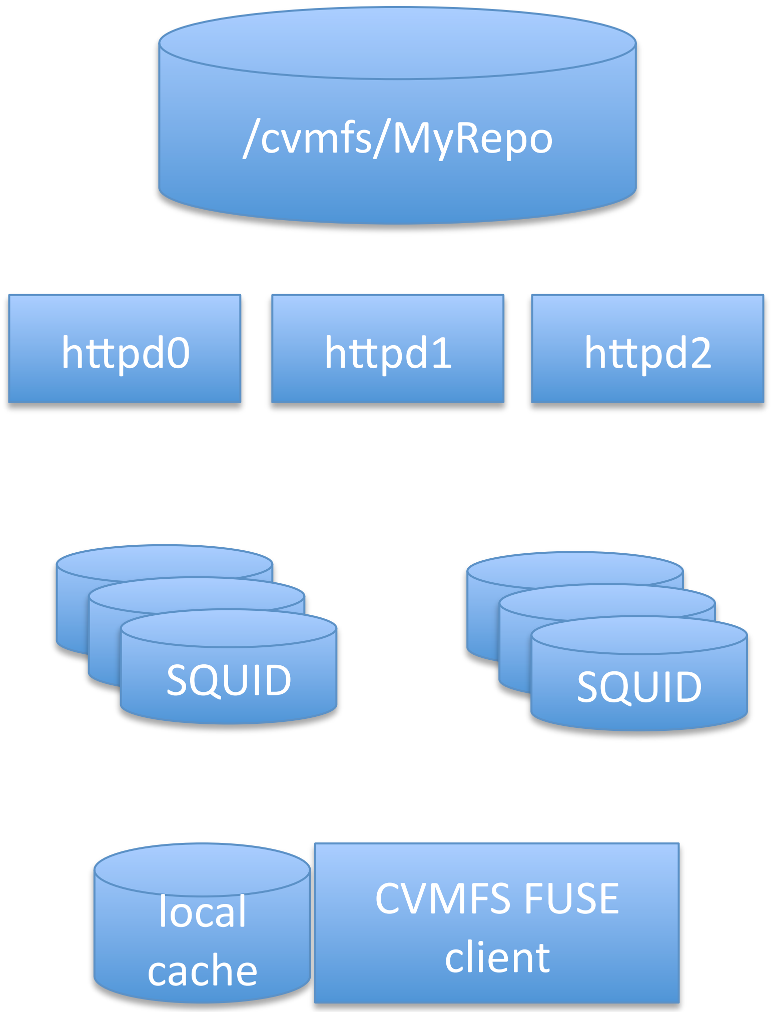
Any data, Any time, Anywhere Project

Outline

- Benefits of CVMFS to campus grid
- Installing FUSE client
- Using Parrot client (non-root)
- A glideinWMS plugin
- Existing repositories
- Hosting your own repository
- Some best practices

What is CVMFS?

- Network filesystem
- Read-only POSIX interface
  - FUSE mounted
- Fetches files via http
  - Verifies data integrity
- Aggressive caching
  - Local disk
  - Web proxies



Benefits of CVMFS to Campus Grids

- Well suited for software distribution:
  - Easily scalable
    - Local disk cache for repeated access
    - Add more web proxies as needed
  - Highly available
    - Robust error handling (failover, offline mode)
    - Add more repository mirrors as needed
  - Secure access over untrusted networks
    - Strong security mechanisms for data integrity
  - Works across administrative domains
    - Including unprivileged environments (Parrot)

Truth in Advertising

- Young project
- Active development
- Small team

- Set expectations accordingly!
  - e.g. server component rarely used outside CERN, so more rough edges than client, which is used by many LHC sites

Getting the FUSE Client

1. Install rpm
2. Tell it which http proxies to use
3. Allocate cache space
4. Enable desired repositories

Installing FUSE Client

- RPMs are available from CERN and OSG
- CERN:
  http://cernvm.cern.ch/portal/filesystem
- OSG:
  https://twiki.grid.iu.edu/bin/view/Documentation/Release3/InstallCvmfs

What if I am not root?

- Parrot Virtual Filesystem
  - No root privileges required
  - Works as job wrapper

```
parrot_run /cvmfs/repo/MyProgram …
```

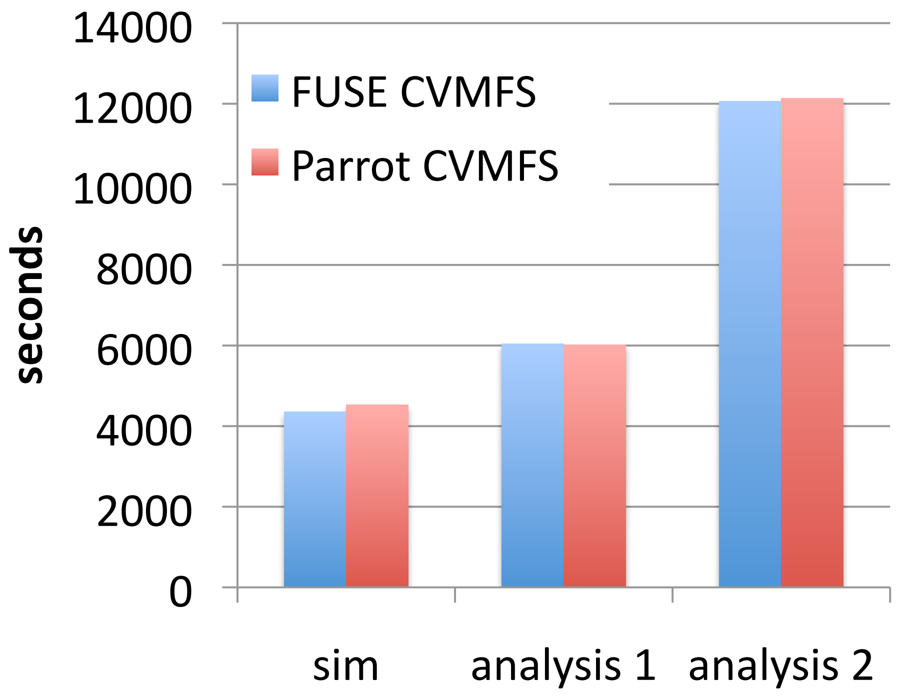
See http://www.nd.edu/~ccl/software/parrot/

Example Parrot Setup

```
$ wget http://www.nd.edu/~ccl/software/files/cctools-3.6.1-x86_64-
    redhat5.tar.gz

$ tar xzf cctools-3.6.1-x86_64-redhat5.tar.gz

$ export PATH=`pwd`/cctools-3.6.1-x86_64-redhat5/bin:$PATH

$ export HTTP_PROXY=frontier01.hep.wisc.edu:3128

$ parrot_run bash

bash-3.2$ ls /cvmfs/grid.cern.ch
3.1.22-0  3.1.39-0  3.1.41-0  3.1.45-0 3.2.11-1   default  etc     glite
```

Parrot Performance Cost

- Experience in CMS:
- For typical CMS jobs, running under Parrot is not much slower
- Your mileage may vary
  - Assume 5% performance hit until proven otherwise



Parrot Cache

- CVMFS local cache is in parrot tmp area
  - Default: /tmp/parrot.<uid>
  - Only one instance of parrot can use it at a time!
  - Override with parrot_run –t <path>
    - e.g. batch job could put it in per-job tmp dir
- Comparison to FUSE CVMFS
  - Local cache not shared between batch slots
    - So uses more bandwidth and disk space
  - If cache deleted after job runs, successive jobs in same slot must start from scratch
    - Could be a problem for short jobs (e.g. O(1) minute jobs)

Accessing Multiple Repositories

- Not efficient in current implementation
  - Considered an experimental feature
  - Disallowed by default
  - But should be ok for occasional switching from one repository to another, say < 0.1 Hz
- To enable multi-repository access in a single parrot session:

```
export PARROT_ALLOW_SWITCHING_CVMFS_REPOSITORIES=1
```

Accessing Other Repositories

- By default, Parrot knows about the CERN repositories
- Can configure Parrot to access other repositories

  export PARROT_CVMFS_REPO=*cms.hep.wisc.edu*:**pubkey**=*/path/to/cms.hep.wisc.edu.pub*,**url**=*http://cvmfs01.hep.wisc.edu/cvmfs/cms.hep.wisc.edu*

  (Or use equivalent parrot_run –r option.)

- See Parrot user's manual for more cvmfs options
  - e.g. local cache quota

Use-case: FUSE CVMFS at home, glidein+Parrot abroad

- Idea:
  - Job can expect uniform CVMFS access wherever it lands
  - No need to modify job code for different environments
    - Campus machines we administer
    - OSG machines we don't administer

A glideinWMS Job Wrapper

- If job says it requires CVMFS
  - Wraps job in parrot
  - Uses site squid, if possible
  - Otherwise, need a friendly squid at home
    - May limit scalability
    - Access control?
- See https://github.com/dcbradley/parrot_glidein_wrapper

glideinWMS CVMFS local cache

- Two cases:
  - Using glexec
    - Each job has its own disk cache
    - Deleted when job exits
  - Not using glexec
    - Cache is saved for lifespan of glidein
    - May improve efficiency for very short jobs
- Do we need glexec?
  - Wrapper uses Parrot's identity boxing feature
    - Provides privilege separation between job and glidein
    - But cannot be 100% trusted yet due to wrapper running in user-controlled environment – work in progress

glideinWMS parrot_cfg

```
# configure parrot cvmfs options
# Here we just set the local cache quota
# Only default (CERN) repositories are enabled here
PARROT_CVMFS_REPO="<default-repositories>:quota_limit=4000,quota_threshold=2000"

# central proxies to use for CVMFS if the local site proxy cannot be used
CVMFS_PROXIES="http://cache01.example.edu:8001|http://cache02.example:8001"

# CVMFS repository to use to test site web proxy
CVMFS_TEST_REPO="http://cvmfs-stratum-one.cern.ch/opt/cms"

# path to test to validate cvmfs access
CVMFS_TEST_PATH=/cvmfs/cms.cern.ch

# If true and parrot can't access CVMFS_TEST_PATH, abort glidein startup.
GlideinRequiresParrotCVMFS=false

# If true, all jobs are wrapped with parrot, regardless of job's RequireCVMFS attribute.
GlideinAlwaysUseParrotWrapper=false
```

Example glideinWMS job

```
# tell glidein to wrap the job in parrot
# (only relevant if glidein config makes this feature optional)
+RequiresCVMFS = True

Executable = my_program
Output = stdout
Error = stderr
Queue
```

Existing Repositories

- CERN repositories
  http://cernvm.cern.ch/portal/cvmfs/examples
  grid.cern.ch, cms.cern.ch, atlas.cern.ch, etc.
- OASIS
  - OSG project under development
  - VOs may publish files in repository hosted by OSG
  - Alternative to maintaining files in OSG_APP at all target sites
- Wisconsin OSG_APP (GLOW OSG site)
  - VOs write to it like any other OSG_APP

CVMFS Server

- Only needed if you wish to create your own repository
- Lightweight service
  - Kernel module to detect updates
  - Program to prepare published files
  - httpd to serve files
  - Most I/O done by proxies
- May also want a mirror server
  - httpd + periodic sync of repository files

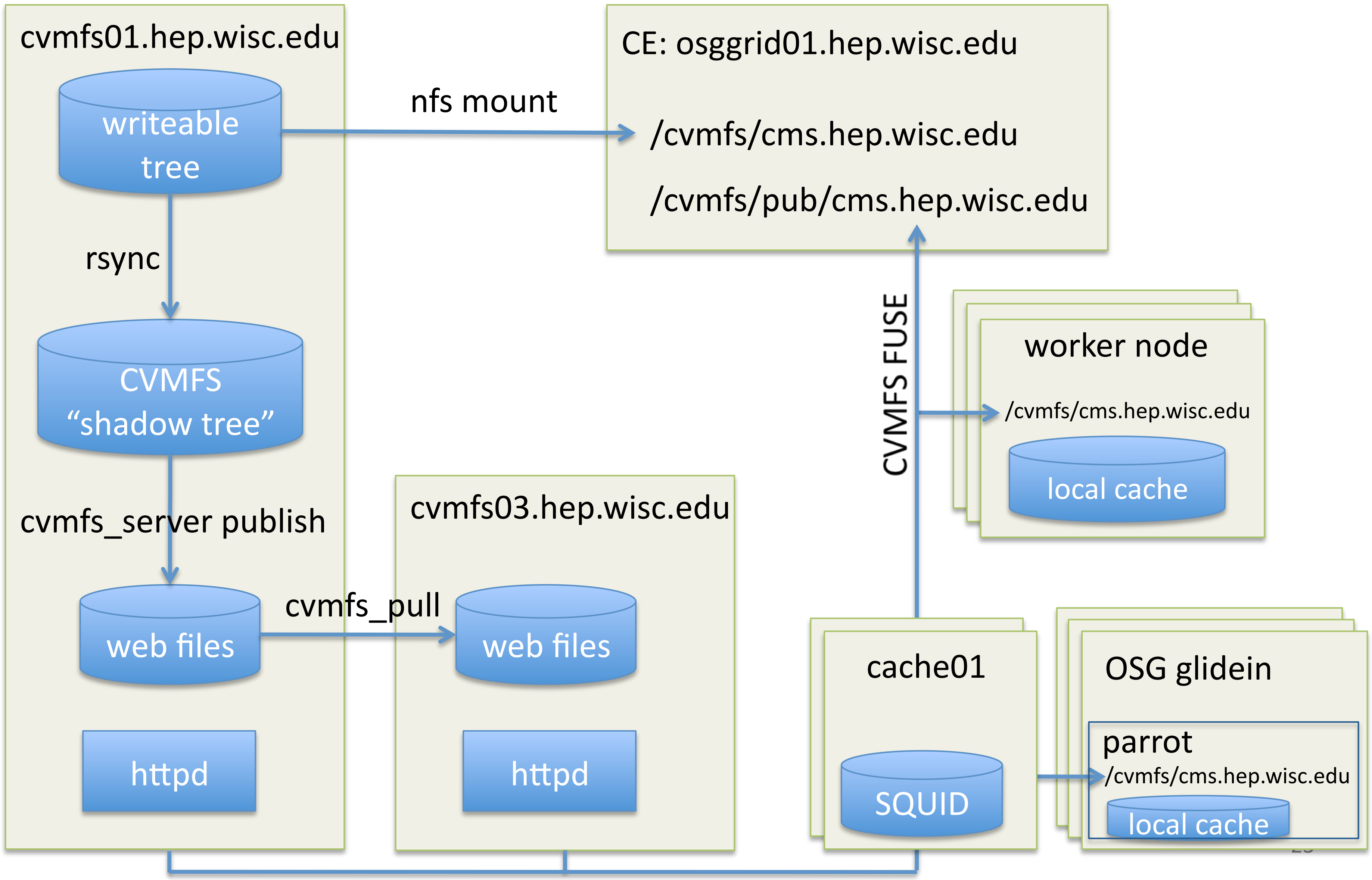
Managing the Repository

- Simple case: one software maintainer (cvmfs user)
  - Updates software tree
  - Triggers cvmfs publication step
  - New files show up on clients an hour later (or less)

Managing the Repository

- More complicated scenario: implementing OSG_APP with CVMFS
  - There are many software maintainers
  - We don't want them to have to trigger publication
- Tried periodically running publication
  - Caused long delays and/or write errors to software maintainers operating at time of publication
- Instead, using periodic rsync from user-maintained tree into cvmfs software tree
  - Then publish to cvmfs
  - Software maintainers are never blocked

Wisconsin OSG_APP Repository



Some CVMFS Best Practices

- Following examples are for HTCondor
  - Ideas are more general

Integrating with HTCondor: health check

- Problem: job runs and fails on machine with broken CVMFS
  - e.g. cache is on broken/full filesystem
- How to avoid such black holes:
  - startd cron job tests for working cvmfs
  - Publishes MyRepo_CVMFS_Exists = True
    - Actual expression: ifThenElse(isUndefined (LastHeardFrom),CurrentTime,LastHeardFrom) - 1352866188 < 3600
    - True until test expires in 1 hour
  - Job requires TARGET.MyRepo_CVMFS_Exists == True

check_cvmfs startd cron job

- See https://github.com/dcbradley/startd_cron
  - Basic functional test
  - Monitor cache space
    - Important if cache does not have its own dedicated partition
  - Advertise current CVMFS catalog version

Integration with HTCondor: stale FS

- Problem: job runs and fails on machine that does not yet see latest cvmfs contents
- How to avoid this race condition:
  - startd cron job publishes catalog version: MyRepo_CVMFS_Revision = 4162
  - Job should require execute node revision >= submit node revision
    - For OSG jobs, we do this in condor.pm

Example Job

```
# set the following to output of command:
# attr -q -g revision /cvmfs/myrepo
+SubmitNodeCVMFSRevision = 1234

Requirements = TARGET.MyRepo_CVMFS_Exists
   && TARGET.MyRepo_CVMFS_Revision >=
   SubmitNodeCVMFSRevison
```

Links

- CVMFS website: http://cernvm.cern.ch/portal/filesystem
- Parrot website: http://www.nd.edu/~ccl/software/parrot/
- Parrot CVMFS job wrapper for glideinWMS https://github.com/dcbradley/parrot_glidein_wrapper
- CVMFS OSG_APP implementation https://github.com/dcbradley/cvmfs_osg_app
- HTCondor cvmfs startd cron script https://github.com/dcbradley/startd_cron