On-Demand Provisioning of CernVM File System with GlideinWMS

Namratha Urs
2020 Fermilab Computational Science Internship (FCSI)
Supervisor: Marco Mambelli
20 August 2020

Outline

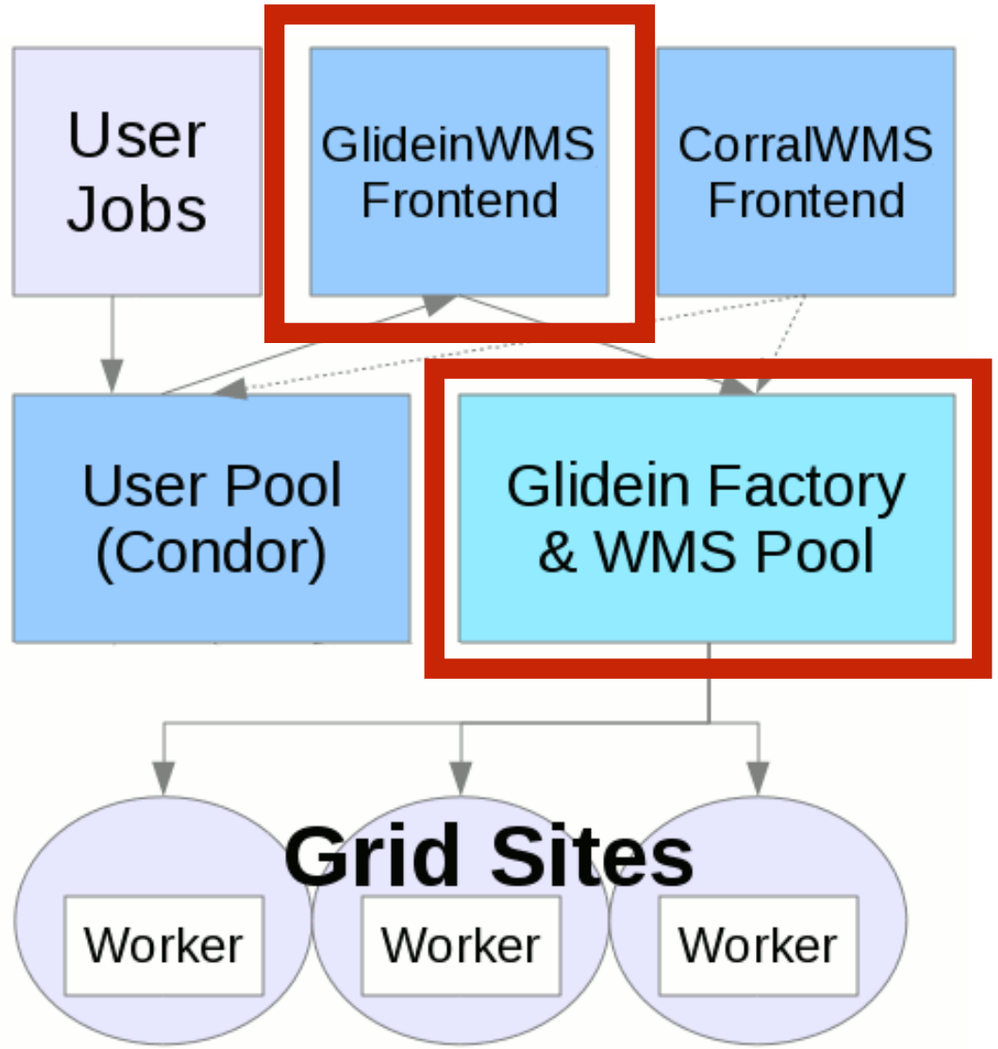
- High Throughput Computing
- GlideinWMS
- CernVM File System
- Difficult Access to Data
- Proposed Solution
- Design and Implementation
- Benefits
- Summary & Outlook

High Throughput Computing (HTC)

- Many computing resources used over long periods of time to accomplish a computational task
- Growing needs
  - Scalability
  - Accessiblity
  - Simplified management
- Trends of increased heterogeneity
  - Multiple organizations
  - Different systems
  - Less standard infrastructure
  - Different authentications
- Increasing complexity ⇒ More difficult for scientists!

GlideinWMS

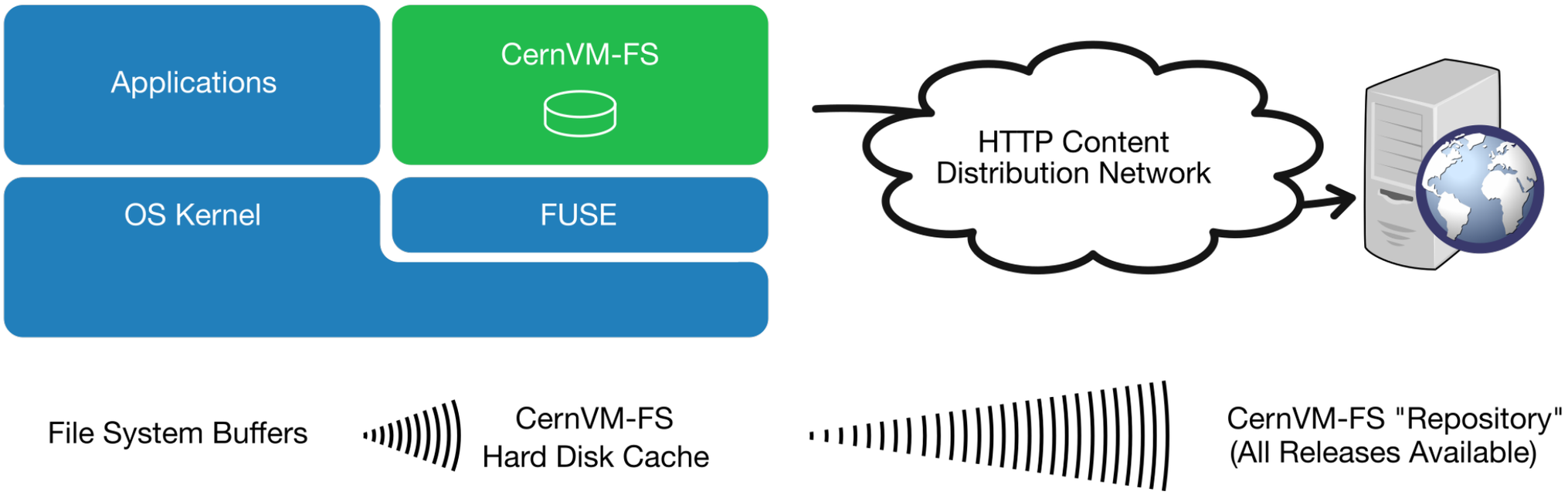
- Workflow manager that simplifies resource provisioning for distributed high throughput computing
- **Components:**
  - **Glideins (pilots)**: provide a customized execution environment for user jobs
  - **Frontend:** look for user jobs and request the Factory to provide glideins
  - **Factory:** self-advertise, listen for requests from Frontend and submit glideins
- Leverages HTCondor for scheduling and job control
- Hassle-free job execution for end-users



Graphic Credits: GlideinWMS Internal Project Documentation

CernVM File System (CernVM-FS or CVMFS)

- **Read-only**, globally distributed file system based on HTTP
- Optimized to distribute and deploy scientific software/data to nodes
- Scalable, reliable and low-maintenance software distribution service



Graphic Credits: https://cvmfs.readthedocs.io/en/stable/

Difficult Access to Data

- High-Energy Physics entails an abundance of computing resources, i.e. “sites” in the distributed computing jargon
  - Local batch farms, grid sites, private/commercial clouds, supercomputing centers
- CernVM File System (CVMFS) used in collaborations within the particle physics community
  - Distribute software stack and some data (e.g. calibrations) for experiments
  - Facilitate containerization by hosting container images and also the containerization software accessible to regular (unprivileged) users
- Some sites (HPC resources) may not provide a local installation of CVMFS
  - Minimize effort required for local installation by site admins
- **Desired outcome:**
  1. Make CVMFS available on HPC sites when a local installation is unavailable
  2. Minimize the effort required by site administrators to install CVMFS locally

My Solution

- Extend glidein functionality to install CVMFS on the worker node if not available
  - GlideinWMS sends glideins to test and setup nodes; becomes one more task
- Perform CVMFS installation in unprivileged mode
  - Required since glideins have no special privileges

The Detail!

- On-demand provisioning of CVMFS using GlideinWMS
  - Ship with the glidein a tool to provision CVMFS and use if needed
  - Select the most reliable option to enable CVMFS given the worker node setup
- Use `cvmfsexec`: tool for making CVMFS available without a system wide installation
  - Leverage unprivileged user namespaces and FUSE interface
- 3-stage development process resulting in three module deliverables
  1. Prototype
  2. Working feature
  3. Integration with GlideinWMS

Why cvmfsexec?

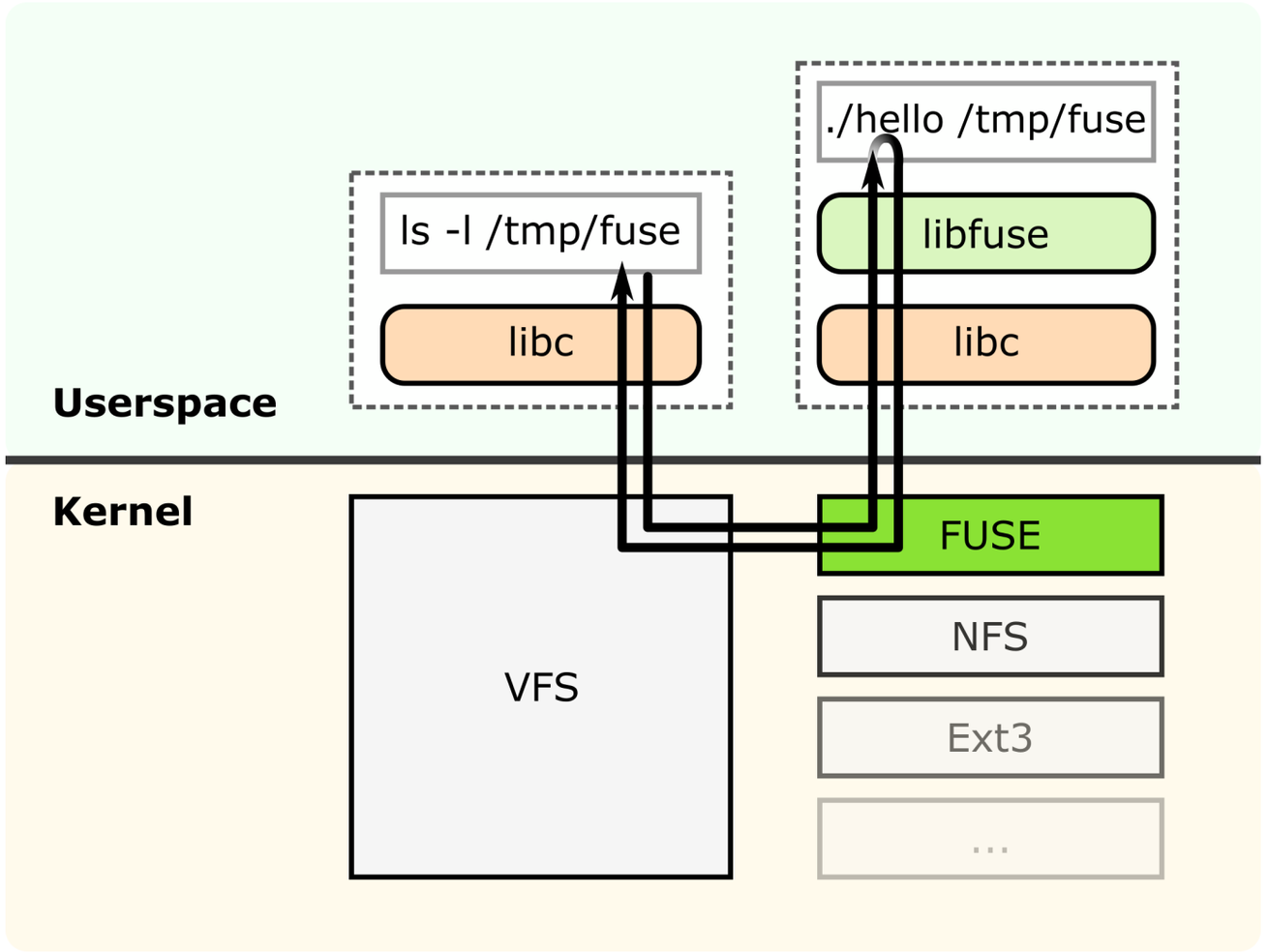
- Package support for unprivileged CVMFS
  - Relies on unprivileged user namespaces and FUSE (Filesystem in Userspace) configurations
  - Four ways to mount CVMFS as a non-root (unprivileged) user
- Creates distribution with CVMFS software and configuration
  - `osg`, `egi` or `default` parameter for latest cvmfs and configuration rpms
  - Allows custom CVMFS configuration settings
- Self-contained distribution as a single file
  - Easy sharing with other users or to many machines
  - `singcvmfs` distributions (mounts CVMFS repositories inside a container)
- Access to one of the developers led to
  - Further understanding of the software
  - Request and implementation of new features

Unprivileged User Namespaces

- Namespaces isolate global system resources between independent sets of processes
  - User namespaces: created by regular users; used by unprivileged processes to access privileged capabilities
  - Limits the scope of that privilege to the user's namespace
  - Can allow a process to create namespaces and to mount a filesystem for all the processes in the same namespace
  - More modern

Filesystem In Userspace

- Framework that allows secure, non-privileged mounts without modifying kernel code
  - Bridge to the actual kernel interfaces
  - Kernel module + userspace library + mount utility (`fusermount`)
  - An opportunity to use filesystems!
  - More available



Credit: https://en.wikipedia.org/wiki/Filesystem_in_Userspace

Fermilab

Deliverable #1: Prototype

Design

- Manually executed and tested the commands
  - Detect platform (rhel7, rhel6, rhel8, centos, other)
  - Detect kernel info (2.x, 3.x, 4.x, other)
  - Detect unprivileged user namespaces supported
  - Detect unprivileged user namespaces enabled
  - Detect FUSE availability and user is in *fuse* group
  - Tie all of the above together as a helper script
- Detect if CVMFS is already mounted on the node and mount if not
  - List of CVMFS repositories can come from an environment variable or command-line argument
  - mounting CVMFS using `cvmfsexec`
  - mounting CVMFS using `mountrepo`
- Integrate the parts described above in a separate script
- Include `INFO/WARN/ERROR` messages to improve output/error/debug messages
- Test on platforms and validate against expected behavior

Deliverable #1: Prototype

Implementation

- Three shell scripts (bash-compliant)
  - `cvmfs_helper_funcs.sh` — includes helper functions for evaluating worker node configuration, performing system checks, printing system information for debugging purposes, mounting CVMFS configuration and additional repositories.
  - `cvmfs_mount.sh` — wrapper script for mounting CVMFS before user job starts
  - `cvmfs_unmount.sh` — wrapper script for unmounting CVMFS after user job ends
- Modularized code to enable code reusability and to ease testing
- Adopted standard logging mechanism for consistency and flexibility
- Bug fixes were incorporated in the scripts as and when encountered
- Inline documentation to aid code readability

Deliverable #1: Prototype

Testing

CVMFS Testing Matrix

unprivileged user namespaces supported? (via sysctl)	unprivileged user namespaces enabled? (via unshare)	fuse installed?	fusermount available?	user in 'fuse' group?	cvmfsexec works?	mountrepo/umountrepo works?	test remarks for mountrepo usage
Yes	Yes	No	No	No	Yes	No	requires fuse/fusermount
Yes	Yes	No	No	Yes	Yes	No	requires fuse/fusermount
Yes	Yes	No	Yes	No	Yes	No	inconsistent FUSE config
Yes	Yes	No	Yes	Yes	Yes	No	inconsistent FUSE config
Yes	Yes	Yes	No	No	Yes	No	inconsistent FUSE config
Yes	Yes	Yes	No	Yes	Yes	No	inconsistent FUSE config
Yes	Yes	Yes	Yes	No	Yes	Yes	works even though the user is not in fuse group
Yes	Yes	Yes	Yes	Yes	Yes	Yes	works
Yes	No	No	No	No	No	No	neither cvmfsexec nor mountrepo works (error related to unshare)
Yes	No	No	No	Yes	No	No	neither cvmfsexec nor mountrepo works (error related to unshare)
Yes	No	No	Yes	No	No	No	inconsistent FUSE config
Yes	No	No	Yes	Yes	No	No	inconsistent FUSE config
Yes	No	Yes	No	No	No	No	inconsistent FUSE config
Yes	No	Yes	No	Yes	No	No	inconsistent FUSE config
Yes	No	Yes	Yes	No	No	Yes	mountrepo works (even though the user is not in fuse group)
Yes	No	Yes	Yes	Yes	No	Yes	mountrepo works
No	No	No	No	No	No	No	neither cvmfsexec nor mountrepo works (failed to exec fusermount)
No	No	No	No	Yes	No	No	neither cvmfsexec nor mountrepo works (failed to exec fusermount)
No	No	No	Yes	No	No	No	inconsistent FUSE config
No	No	No	Yes	Yes	No	No	inconsistent FUSE config
No	No	Yes	No	No	No	No	neither cvmfsexec nor mountrepo works (permission denied error)
No	No	Yes	No	Yes	No	No	inconsistent FUSE config
No	No	Yes	Yes	No	No	No	inconsistent FUSE config
No	No	Yes	Yes	Yes	Yes	Yes	mountrepo works

Deliverable #2: Working Feature

- Configured a custom script using the parameters in the glidein config file
  - `cvmfs_test.sh` — imports helper functions and invokes the CVMFS mount script
- Manually created tarball containing auxiliary files
  - Utilities to mount/unmount CVMFS: platform- and architecture-specific distributions
  - Main scripts: (a) helper functions, (b) mount script and (c) unmount script
- Added wrapper script + tarball to the Factory configuration file to send to the glidein-customized node
- Large number of distributions created for various combinations of platform- and architecture-specifications
  - Extra level of granularity with `osg` and `egi` configuration repositories for CVMFS
- Addition of unit tests using BATS to ensure code quality

Deliverable #3: Integration with GlideinWMS

- GlideinWMS code is modified to add the custom script and the tarball to the default list of uploads
- GlideinWMS code is modified to automate tarball preparation at the time of factory reconfiguration/upgrade
  - Created a custom shell script that automatically generates the distribution based on the specifics of the worker node
    - Reduces the number of distributions that are shipped as artifacts (ONE versus many)

Benefits

- Lower overhead for site administrators
  - Less software to install!
- Easy code and data access for scientists
  - End-user jobs run in desired container image and access software/data distributed by CVMFS
- Improved flexibility to use new resources (e.g. HPC resources)
  - Allows GlideinWMS pilots to support HPC

References

- GlideinWMS Documentation
- https://glideinwms.fnal.gov/presentations/intro/GlideinWMS.pdf
- CernVM File System Docs
- cvmfsexec - https://www.github.com/cvmfs/cvmfsexec
- Unprivileged User Namespaces - https://lwn.net/Articles/532593/
- FUSE - https://www.kernel.org/doc/html/latest/filesystems/fuse.html
- Project codebase - https://www.github.com/namrathaurs/gwms-cvmfs

Acknowledgements

This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

Thank You!

Marco Mambelli

Dave Dykstra

GlideinWMS team

Saba Sehrish

Lorena Lobato

Krista Larson

Cara Brown

Judy Nunez, Sandra Charles

...

Summary & Outlook

- Caters to provisioning CVMFS on-demand based on the worker node setup
  - Added ability to use CVMFS on sites where local installation is absent
  - CVMFS installation by non-privileged users
  - Ships necessary utilities and installs CVMFS by assessing the worker node
- Adaptation of GlideinWMS for use with HPC sites
- Towards the roadmap for GlideinWMS behavior regarding Singularity
  - Necessary step to allow GlideinWMS to also start Singularity on resources where neither CVMFS nor Singularity are available
    - CVMFS provides both the Singularity binary and the containers images
    - Glidein starts singularity in unprivileged mode using the container of choice and make CVMFS available