CMS at TACC Frontera
(or “How to make it look like a grid site”)

Dirk Hufnagel (FNAL)

OSG AHM

4th September 2020

CMS at Frontera

- Frontera well suited for CMS workflows (8008 Intel Cascade Lake Xeon nodes, no GPU, plenty of memory, outbound network from batch nodes)

- CMS has a 500k node hour Frontera allocation (July 2020 to June 2021)
  - Continuous use from today would require ~70 nodes
  - “normal” batch queue has 100 jobs total / 50 jobs running limits
    - Need to use multi-node batch jobs

- Outbound network, so no problem for Frontier conditions access
- Biggest problem: No cvmfs available at Frontera
  - How do we access CMS software?

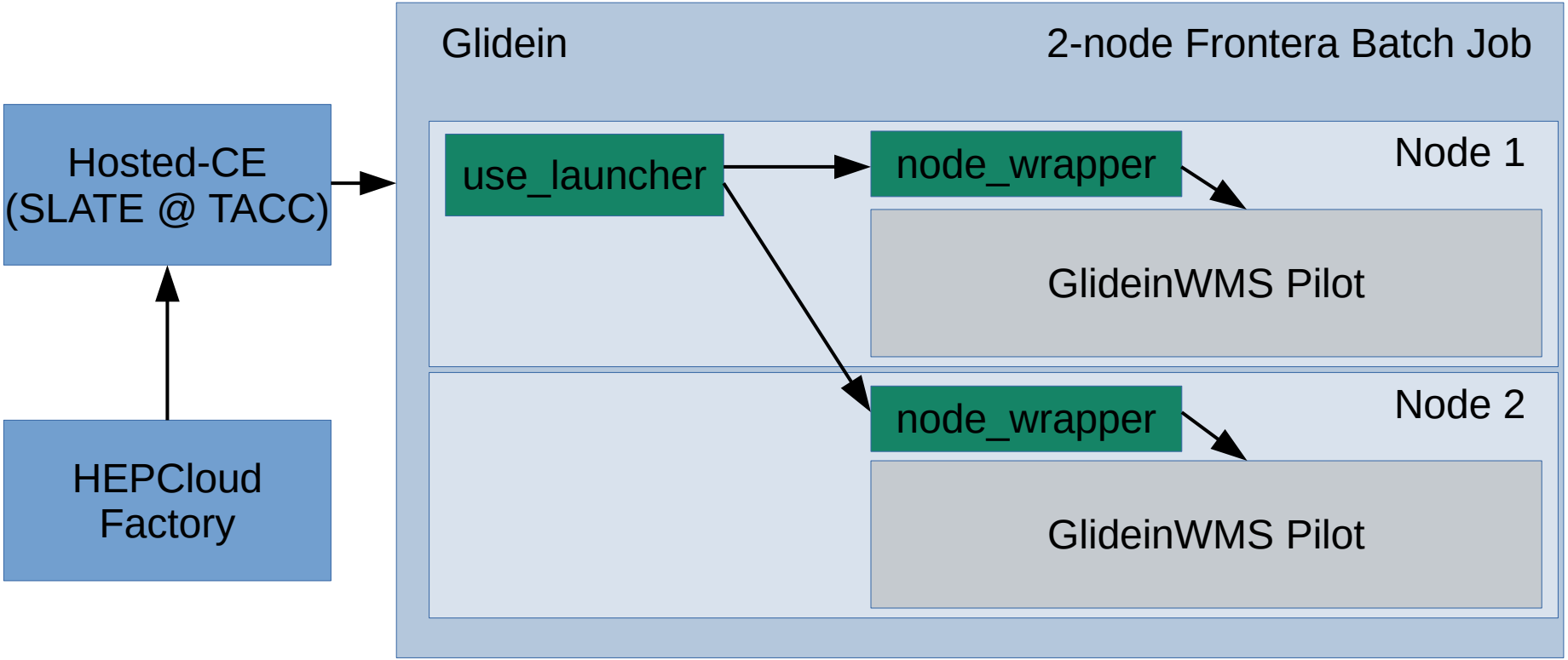
Frontier condition access

- Could use opportunistic squids at FNAL to serve Frontera batch nodes, but bandwidth use and latency make that not advisable.

- CMS has an XSEDE allocation with the same validity period as our Frontera allocation, with some hours on TACC Jetstream (Cloud).

- Installed a frontier-squid on TACC Jetstream (single VM with 40 cores).
  - Big thanks to Dave Dykstra for helping with the setup!

How to get GlideinWMS pilots onto Frontera



GlideinWMS and multi-node batch jobs

- For multi-node batch jobs usually need user supplied code to let GlideinWMS utilize all nodes

- For Frontera this code uses the TACC provided “launcher” module, which we use in our own custom use_launcher.sh script. That script is then executed directly by the Glidein.

  - <attr name="GLIDEIN_MULTIGLIDEIN_LAUNCHALL" glidein_publish="True" job_publish="True" const="False" parameter="True" publish="True" type="string" value="/home1/05501/uscms/launcher/use_launcher.sh"/>
  - <attr name="GLIDEIN_MULTIGLIDEIN" glidein_publish="True" job_publish="True" const="False" publish="True" parameter="True" type="int" value="2"/>

Mounting cvmfs

- node_wrapper script mounts cvmfs as unprivileged user in the /cvmfs namespace and executes the GlideinWMS pilot

- It uses the cvmfsexec command from the cvmfsexec package
  (written and maintained by Dave Dykstra)

- Simple setup, script is only 7 lines of functional code

- Configured for 10GB local SSD cache on batch node
- Uses WPAD auto-discovery to find TACC Jetstream squid

cvmfsexec requirements and where CMS is using it

- Supports RHEL (latest version adds Suse support)

- cvmfsexec command requires unprivileged user namespaces
  - available on RHEL >= 7.6 (but only enabled in later versions)
  - works best with unprivileged fuse mounts
    - available on RHEL >= 7.8
    - alternatively can use fusermount
- cvmfsexec mountrepo/umountrepo also works with only fusermount

- TACC Frontera (RHEL 7.8) => tested CMS jobs
- ALCF Theta (Cray Compute Node Linux, based on Suse 15) => tested CMS jobs
- TACC Stampede2 (RHEL 7.6) => in production

Singularity, Data Input/Output

- Within the GlideinWMS pilot CMS then wraps all payloads into either an RHEL6 or RHEL7 singularity container
  - Nothing unusual, we do that at (almost) all of our sites
  - Don’t use TACC provided singularity installation, but unprivileged singularity from cvmfs (works a bit better with cvmfsexec)

- Input data is read via AAA from other CMS sites, stageout from jobs goes directly to FNAL storage

Current Status and Outlook

- Hosted-CE is still being setup, so what's presented here hasn't been tested fully end to end.
- That being said, all Frontera wrapper scripts have been tested with real CMS jobs, taking over every core on each node in a multi-node batch job.
- In addition, we have used TACC Stampede2 in similar fashion with an external Hosted-CE.
- Looking forward to full production use of Frontera!

BACKUP

use_launcher.sh

```
#!/bin/bash

module load launcher

NUMBEROFNODES=2

export LAUNCHER_WORKDIR=/home1/05501/uscms/launcher

export LAUNCHER_JOB_FILE=`mktemp -p $LAUNCHER_WORKDIR 'jobfile.XXXXXXXX'`

for i in $(seq $NUMBEROFNODES)
do
    echo "$LAUNCHER_WORKDIR/node_wrapper.sh $@" >> $LAUNCHER_JOB_FILE
done

$LAUNCHER_DIR/paramrun

rm -f $LAUNCHER_JOB_FILE
```

node_wrapper.sh

```
#!/bin/bash

# use unpriviliged singularity from cvmfs
export PATH=/cvmfs/oasis.opensciencegrid.org/mis/singularity/bin:$PATH

# want locally mounted cvmfs and local cvmfs cache
rm -rf /tmp/uscms
mkdir -p /tmp/uscms/cache
cp -a /home1/05501/uscms/launcher/cvmfsexec /tmp/uscms/cvmfsexec

# use cvmfs certs
export SINGULARITYENV_X509_CERT_DIR=/cvmfs/oasis.opensciencegrid.org/mis/certificates/

# mount cvmfs (singularity doesn't like Frontera system LD_PRELOAD libs)
env LD_PRELOAD="" /tmp/uscms/cvmfsexec/cvmfsexec cms.cern.ch unpacked.cern.ch oasis.opensciencegrid.org -- $@

# cleanup
rm -rf /tmp/uscms
```

cvmfsexec setup and config

- easy to setup

```
login3.frontera(314)$ cd /home1/05501/uscms/launcher
login3.frontera(315)$ git clone -b v4.4 https://github.com/cvmfs/cvmfsexec
login3.frontera(315)$ cd cvmfsexec
login3.frontera(315)$ ./makedist osg
```

- adjust cvmfsexec/dist/etc/cvmfs/default.local for local environment

```
login3.frontera(316)$ cat cvmfsexec/dist/etc/cvmfs/default.local
CVMFS_HTTP_PROXY="auto;DIRECT"
CVMFS_PAC_URLS="http://grid-wpad/wpad.dat;http://wpad/wpad.dat;http://cernvm-
wpad.fnal.gov/wpad.dat;http://cernvm-wpad.cern.ch/wpad.dat"
CVMFS_CACHE_BASE=/tmp/uscms/cache
CVMFS_QUOTA_LIMIT=10000
CMS_LOCAL_SITE=T3_US_TACC
```