CONNECT

connect.usatlas.org

Remote Cluster Connect Factories

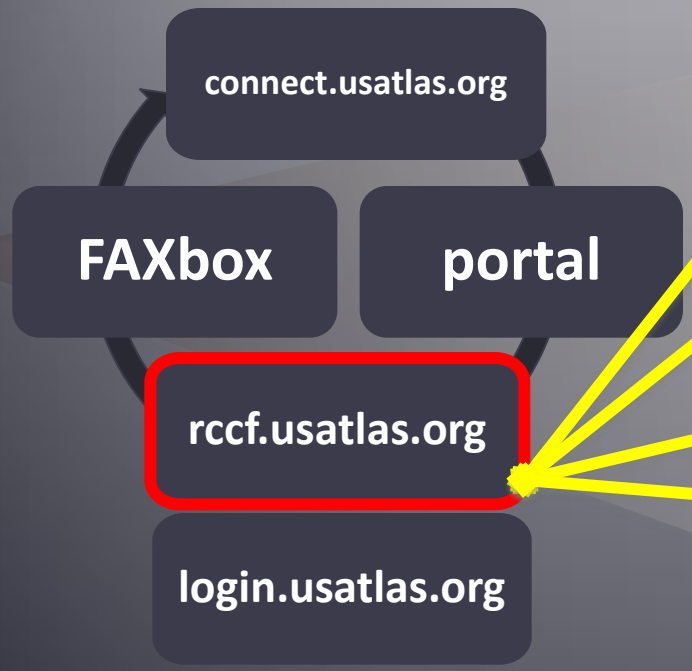
David Lesny

University of Illinois

Remote Cluster Connect Factories rccf.usatlas.org

- Central component of ATLAS Connect
  - User (user login)
  - Cluster (remote T3 flocking)
  - Panda (pilots)
- Provides Tier3 queue-like access to beyond-pledge, and off-grid resources
- Can connect off-grid clusters to Panda

CONNECT user



UC3
UC RCC NEW

MWT2,
AGLT2, SWT2,
CSU Fresno

Tier1 SOON

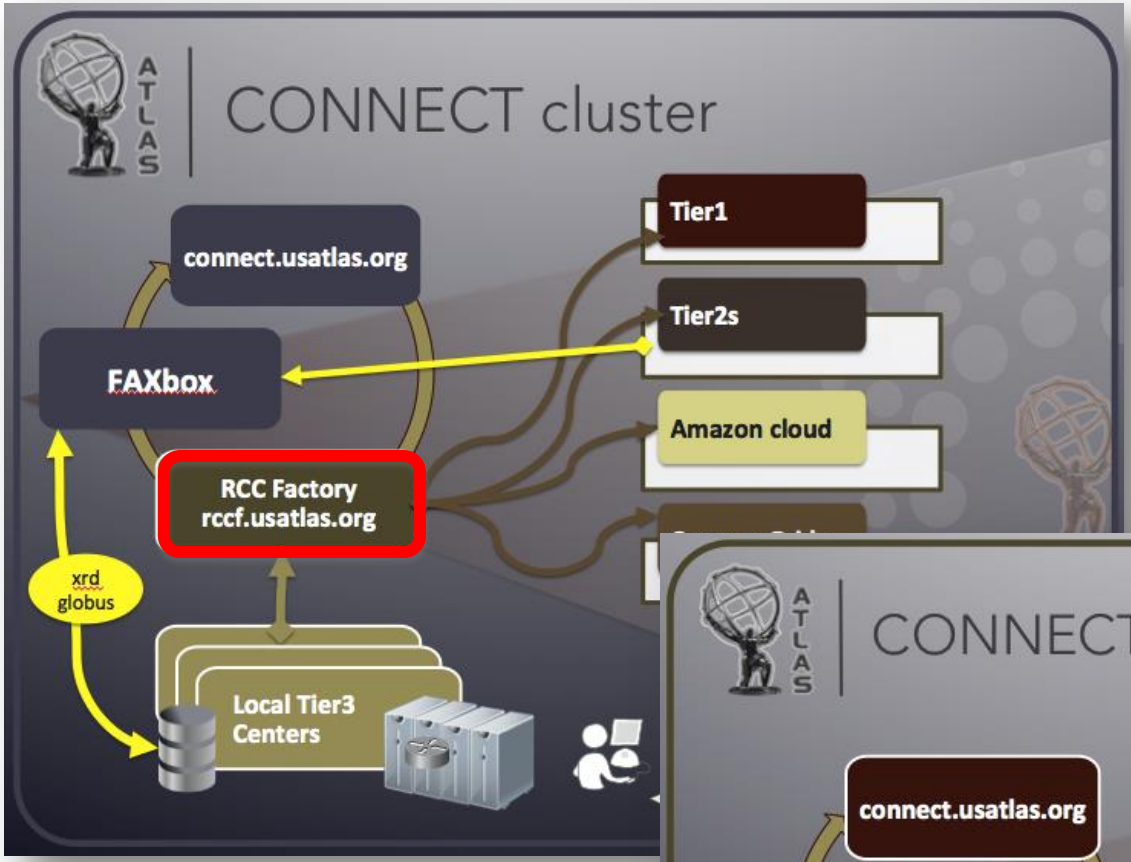
Amazon
cloud

TACC NEW
Stampede

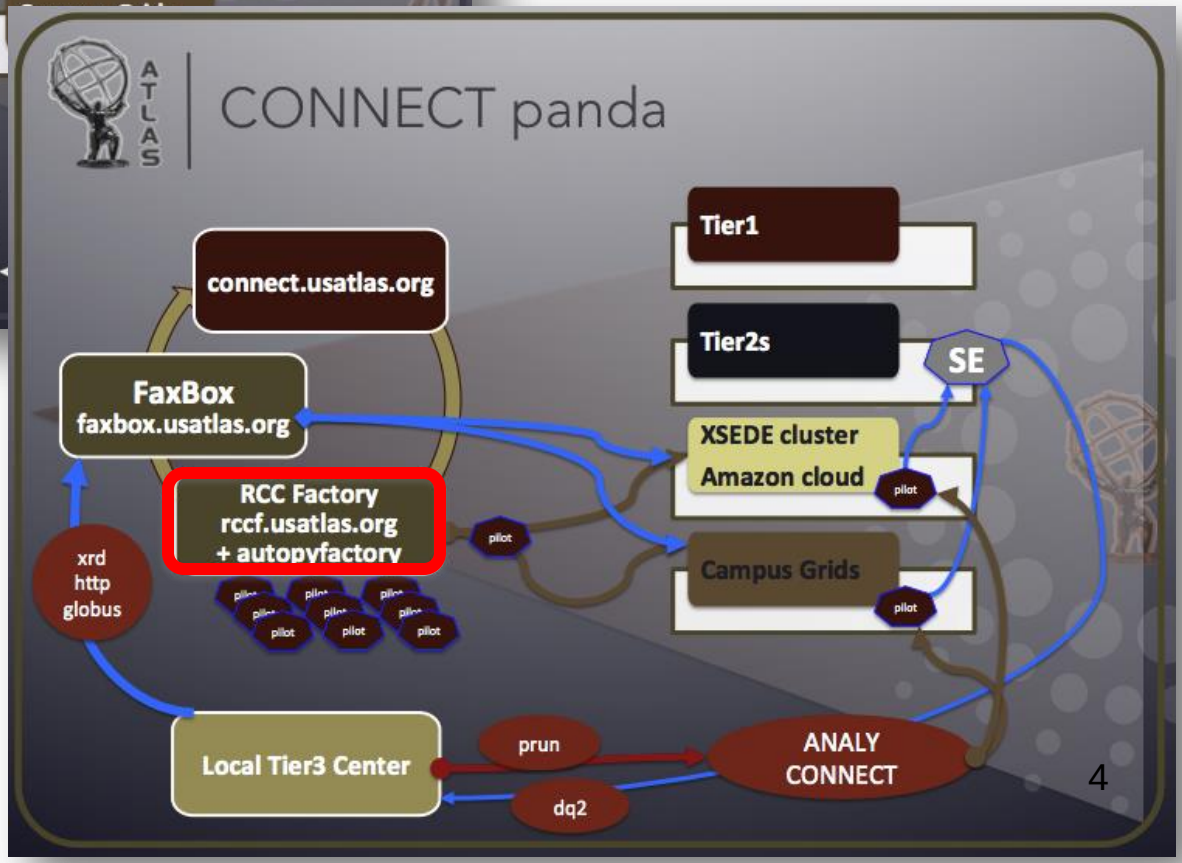
A 10k test 2/5/14

RCC Factories for
direct login service

globus ci connect



RCC Factories for
Tier 3 flocking
(implemented)
➔



RCC Factories for
Panda queues
(up next) ➔

Remote Cluster Connect

Nearly all the pieces have already been created

- Bosco is the job switch fabric used on the RCC
- ATLAS Connect login service, Faxbox, for user job and Tier 3 flocking submissions
- Parrot/CVMFS (CCTools)
- CVMFS server at MWT2
- Existing Frontier Squids

Remote Cluster Connect

Remote Cluster Connect (RCC) provides a mechanism by which an authenticated remote user either via the ATLAS Connect login service or a HTCondor flocked Tier3, can access computing resources at Tier2s (MWT2, AGLT2, SWT2), Tier1 (BNL), Tier3s, or campus clusters (Midway, Stampede, ICC, etc.).

RCC uses multiple instances of “Bosco”, called factories. Each factory is installed on a unique port and runs under a unique user account.

The factory accepts jobs from one or more HTCondor sources, such as the ATLAS Connect login host, and injects them into one or more target clusters.

Jobs are submitted using the targets batch scheduler such as HTCondor, PBS, SGE, LSF and SLURM (with PBS emulation).

Requirements on the target are an account under which the jobs are run, SSH access to a submit host with password less key for injecting the jobs, and outgoing firewall access from the compute nodes to the RCC.

RCC – Multiple Single User Bosco Instances

Remote Cluster Connect Factory (RCCF)

Single User Bosco Instance running as an RCC User on a unique SHARED_PORT

- Each RCCF is a separate Condor pool with a SCHEDD/Collector/Negotiator
- The RCCF injects glideins via SSH into a Target Scheduler
- The glidein creates a virtual job slot from Target Scheduler to the RCCF
- Any jobs which are in that RCCF then run in that virtual job slot
- Jobs are submitted to the RCCF by flocking from a Source SCHEDD
- The RCCF can inject glideins to multiple Target Scheduler hosts
- The RCCF can accept flocked jobs from multiple Source SCHEDD hosts
- Must have open bidirectional access to at least one port on Target Scheduler
- Firewalls can create problems – SHARED_PORT makes it easier (single port)
- Scale testing to a level of 5k jobs so far

Source is always HTCondor

User submitting jobs always uses HTCondor submit files regardless of the target

User does not have to know what scheduler is used at the target

```
Universe                        = Vanilla
Requirements                    = ( IS_RCC ) && ((Arch == "X86_64") || (Arch == "INTEL"))
+ProjectName                    = ”atlas-org-illinois”
Executable                      = gensherpa.sh
Should_Transfer_Files           = IF_Needed
When_To_Transfer_Output         = ON_Exit
Transfer_Output                 = True
Transfer_Input_Files            = 126894_sherpa_input.tar.gz,
    MC12.126894.Sherpa_CT10_llll_ZZ.py
Transfer_Output_Files           = EVNT.pool.root
Transfer_Output_Remaps          =
    "EVNT.pool.root=output/EVNT_$(Cluster)_$(Process).pool.root”
Arguments                       = "$(Process) 750”
Log                             = logs/$(Cluster)_$(Process).log
Output                          = logs/$(Cluster)_$(Process).out
Error                           = logs/$(Cluster)_$(Process).err
Notification                    = Never
Queue 100
```

AtlasTier1,2 vs. Campus Clusters

Tier1,2 targets are known and defined

- CVMFS is installed and working
- Atlas repositories are configured and available
- Required Atlas RPMs are installed on all compute nodes

Campus Clusters are typically not “ATLAS-ready”

- CVMFS most likely not installed
- No Atlas repositories and thus no atlas software
- Very unlikely every RPM installed

We could “ask” that these pieces be added, but we prefer to be unobtrusive

Provide CVMFS via Parrot/CVMFS

Parrot/CVMFS (CCTools) has the ability to get all these missing elements

- CCTools, job wrapper and environment variables in a single tarball
- Tarball uploaded and unpacked on target as part of virtual slot creation
- Package only used on sites without CVMFS (Campus Clusters)

Totally transparent to the end user

- The wrapper executes the users job in the Parrot/CVMFS environment
- Atlas CVMFS repositories are available then available to the job
- With CVMFS we can also access the MWT2 CVMFS Server

CVMFS Wrapper Script

The CVMFS Wrapper Script is the glue that binds

- Defines Frontier Squids (site dependent list) for CVMFS
- Sets up access to MWT2 CVMFS repository
- Runs the users jobs in the Parrot/CVMFS environment

One missing piece remains to run Atlas jobs – Compatibility Libraries

Atlas Compatibility Libraries

Atlas requires a vary large number of RPMS not normally installed on CC.

These include compatibility libraries and the 32 bit libraries

List of “required” RPMs are dependencies in the HEP_OSlibs_SL6 RPM

We could “ask” the the CC to installed this RPM but maybe another way

Provide all libraries via a CVMFS repository

HEP_OSlibs_SL6

Dumped all dependencies listed in HEP_OSlibs_SL6 1.0.15-1

Fetch all RPMS from Scientific Linux server

Many of these are not relocatable RPMs so used cpio to unpack

```
rpm2cpio $RPM| cpio --quiet --extract --make-directories --unconditional
```

Also added a few other RPMs not currently part of HEP_OSlibs

This creates a structure which looks like

```
drwxr-xr-x  2 ddl mwt2 4096 Feb 17 22:58 bin
drwxr-xr-x 15 ddl mwt2 4096 Feb 17 22:58 etc.
drwxr-xr-x  6 ddl mwt2 4096 Feb 17 22:58 lib
drwxr-xr-x  4 ddl mwt2 4096 Feb 17 22:58 lib64
drwxr-xr-x  2 ddl mwt2 4096 Feb 17 22:58 sbin
drwxr-xr-x  9 ddl mwt2 4096 Feb 17 22:57 usr
drwxr-xr-x  4 ddl mwt2 4096 Feb 17 22:57 var
```

Deploy via MWT2 CVMFS Server

Put this “dump” on the MWT2 CVMFS Server

```
[root@uct2-cvmfs ~]# ll /cvmfs/osg.mwt2.org/atlas/sw/HEP_OSlibs_SL6/1.0.15-0
total 28
drwxr-xr-x  2 root root 4096 Feb 18 11:38 bin
drwxr-xr-x 15 root root 4096 Feb 18 11:38 etc.
drwxr-xr-x  6 root root 4096 Feb 18 11:38 lib
drwxr-xr-x  4 root root 4096 Feb 18 11:38 lib64
drwxr-xr-x  2 root root 4096 Feb 18 11:38 sbin
drwxr-xr-x  9 root root 4096 Feb 18 11:38 usr
drwxr-xr-x  4 root root 4096 Feb 18 11:38 var
```

Link the users jobs to this libraries in the Parrot/CVMFS wrapper

```
myPATH=/cvmfs/osg.mwt2.org/atlas/sw/HEP_OSlibs_SL6/1.0.15-1
export LD_LIBRARY_PATH="$myPATH/usr/lib64:$myPATH/lib64:$myPATH/usr/lib:$myPATH/lib:$LD_LIBRARY_PATH"
```

Success to Stampede and Midway

Two test cases have successfully run on Midway and Stampede Clusters

Peter Onyisi provided both a Reco_trf and Sherpa jobs to test

Jobs can now run transparently on any cluster available which are

- MWT2
- AGLT2
- SWT2
- FresnoState (Tier3)
- Midway

Coming soon

- BNL (Tier1)
- Stampede (need change for 16 jobs in one submission)