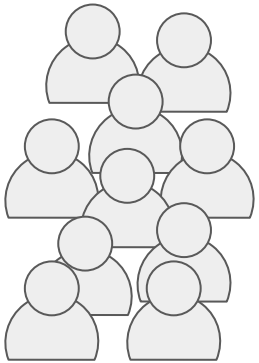
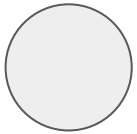
HTCondor-CE: Introduction and Overview

EGI Community Webinar Program

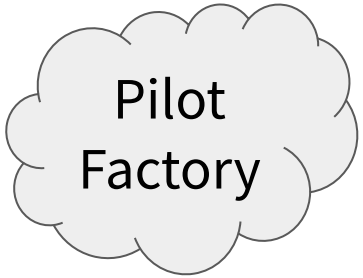
Brian Lin
University of Wisconsin–Madison

Resource Allocation Requests

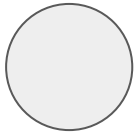




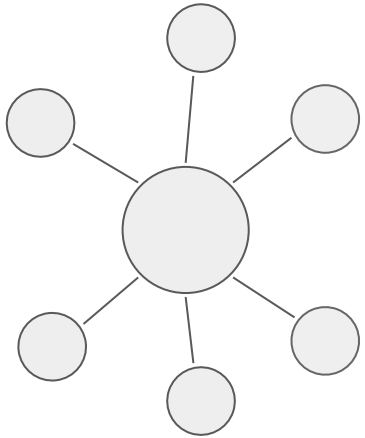
User Submit



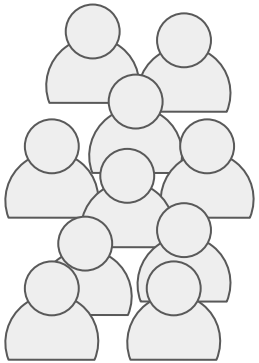
Compute
Entrypoint

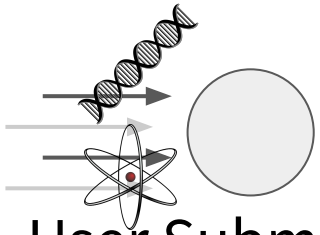


Local Batch System



Resource Allocation Requests

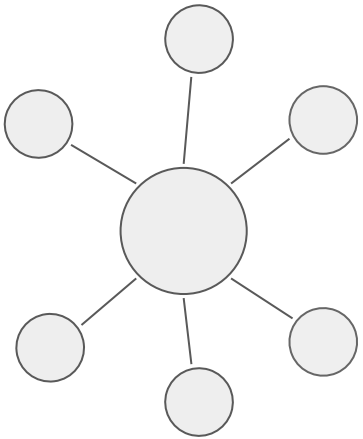




User Submit

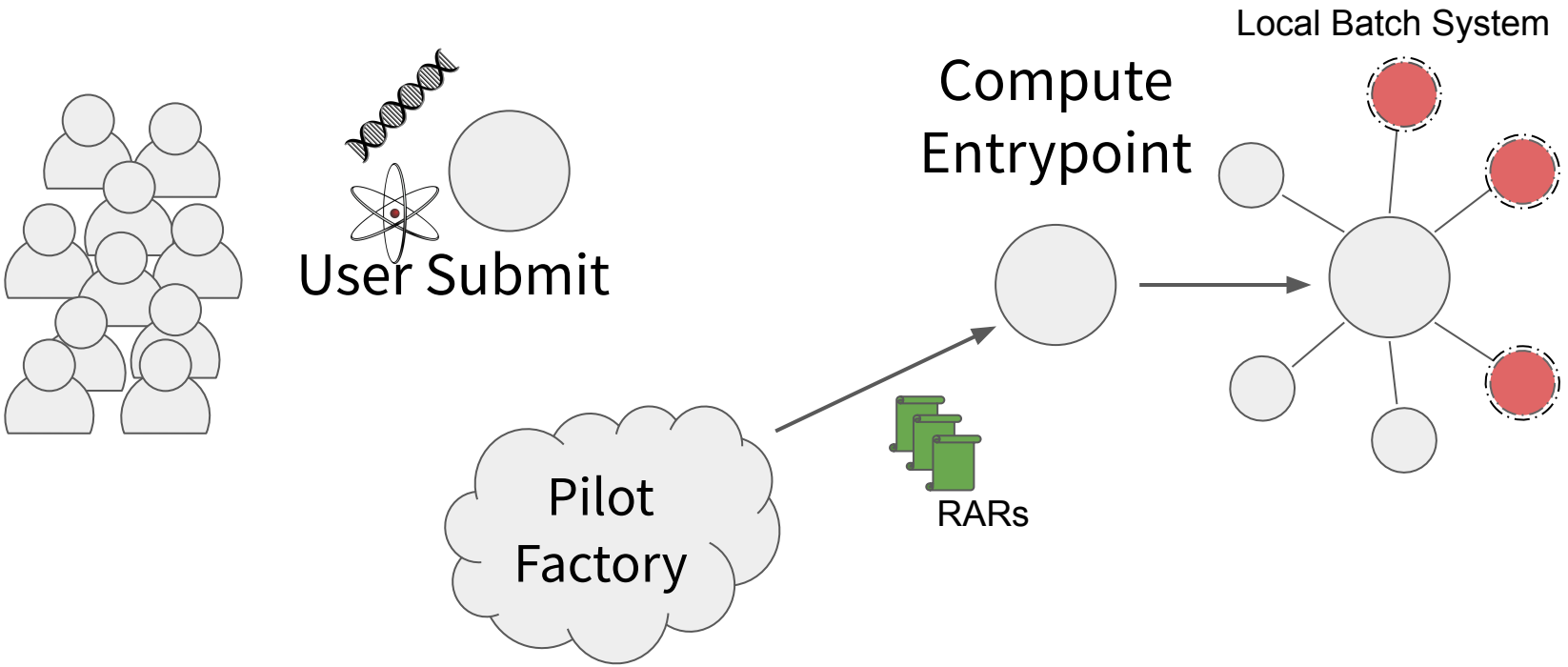
Compute Entrypoint

Local Batch System

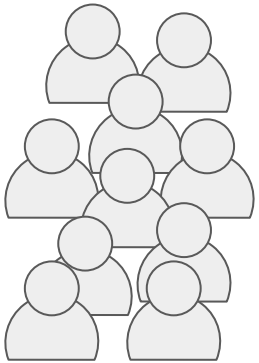


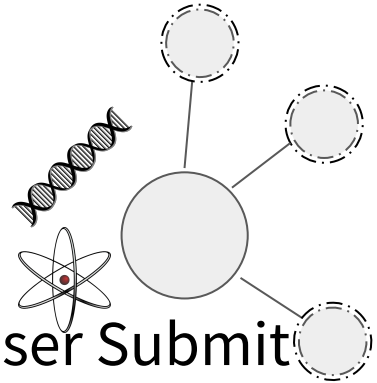
Pilot Factory

Resource Allocation Requests

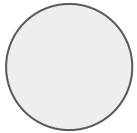


Resource Allocation Requests

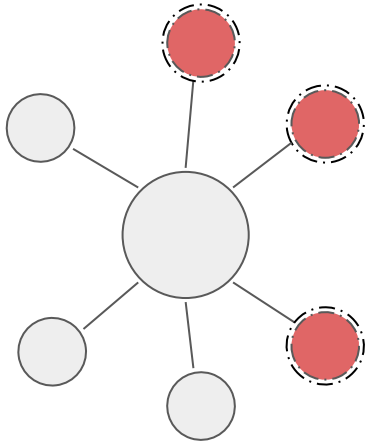




Compute Entrypoint



Local Batch System

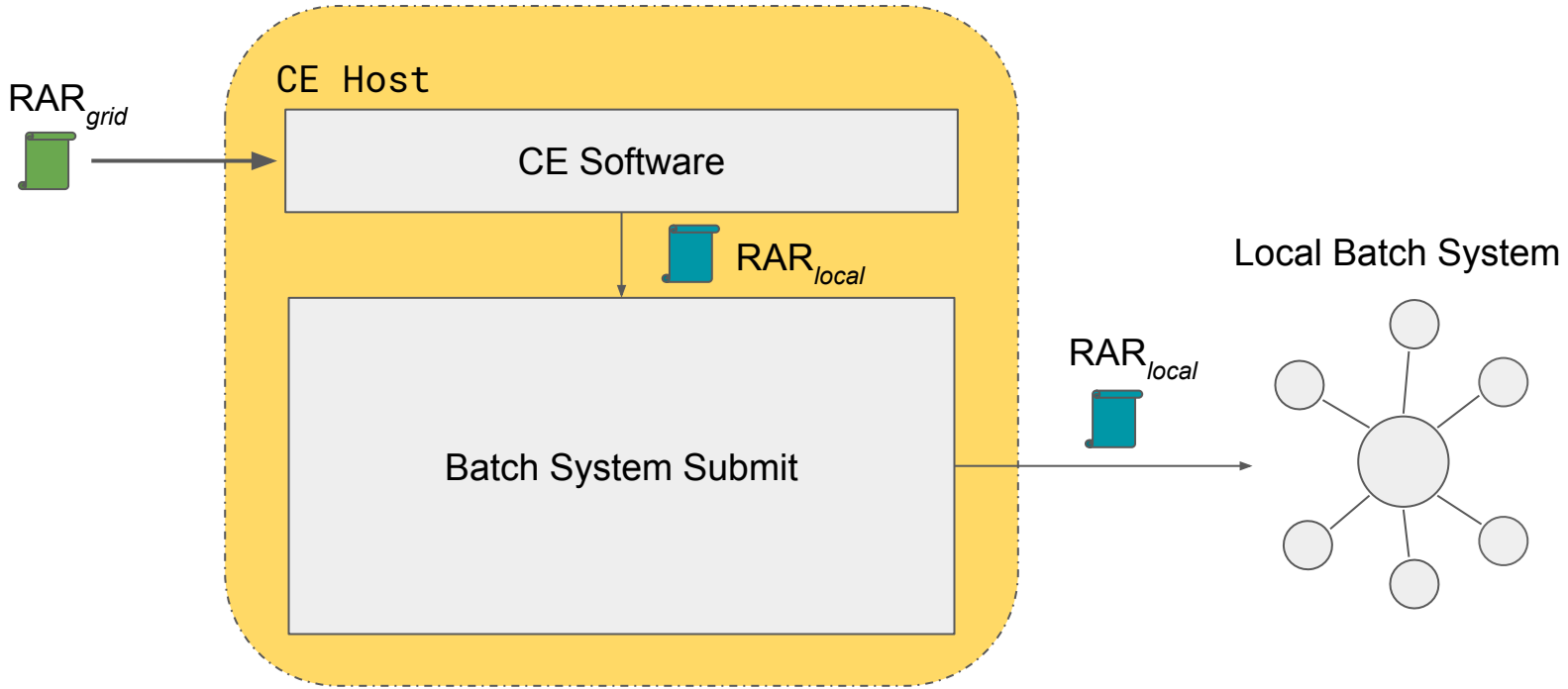


Pilot Factory

What is a CE?

- A compute entrypoint(CE) serves as the door that forwards resource allocation requests (RAR) onto your local compute resources
  - Exposes a **remote API** to accept RARs
  - Provides authentication and **authorization** of remote clients
  - Interacts with the **resource layer** (i.e. batch system)
- A CE host is made up of a thin layer of CE software installed on top of the software that submits to and manages jobs on your local batch system
- Primarily designed to support RARs (i.e., through pilot jobs) and is generally not intended for direct user submission

Compute Element Architecture



HTCondor 101

- Important HTCondor daemons:
  - Master: responsible for starting/stopping other HTCondor daemons on a host
  - SchedD: accepts jobs and stores job state information, i.e. the job queue
  - Collector: stores information about other HTCondor daemons
  - Gridmanager: submits jobs to remote SchedDs, non-HTCondor batch systems
- ClassAds are the lingua franca for describing HTCondor entities (daemons, jobs, security sessions, etc.)
  - Schema-less key/value pairs
  - Declarative language with rich expressions. Often used to compare requirements between two entities (e.g., a job and a worker node)

HTCondor 101

- HTCondor team maintains new feature and bug-fix versions (https://htcondor.readthedocs.io/en/latest/version-history/introduction-version-history.html) available in the ‘development’ and ‘stable’ Yum repositories, respectively:
  - New features: HTCondor 8.9 and HTCondor-CE 4
  - Bug-fix: HTCondor 8.8 and HTCondor-CE 3
- More HTCondor basics resources:
  - Center for High Throughput Computing tutorials: https://www.youtube.com/channel/UCd1UBXmZIgB4p85t2tu-gLw
  - ClassAd documentation: https://htcondor.readthedocs.io/en/stable/misc-concepts/classad-mechanism.html

HTCondor as a Compute Entrypoint

HTCondor-CE is HTCondor configured as a compute entrypoint

- Same HTCondor binaries, description language (ClassAds), and configuration language to provide the **remote API**
- Relevant HTCondor tools are wrapped to use the HTCondor-CE configuration (e.g., `condor_ce_q`, `condor_ce_status`, etc.)
- Separate `condor-ce` service

HTCondor-CE + HTCondor Batch System

- Two sets of HTCondor daemons
  - Two sets of configuration: `/etc/condor-ce/config.d/` and `/etc/condor/config.d/`
  - Two sets of logs: `/var/log/condor-ce/` and `/var/log/condor/`
- The `condor_job_router` is a quick way to identify the HTCondor-CE daemons between the two sets!

```
# pstree
[...]
  ├─condor_master─┬─condor_collector
  |               ├─condor_negotiator
  |               ├─condor_procd
  |               ├─condor_schedd
  |               ├─condor_shared_port
  |               └─condor_startd
  ├─condor_master─┬─condor_collector
  |               ├─condor_job_router
  |               ├─condor_procd
  |               ├─condor_schedd
  |               └─condor_shared_port
[...]
```

HTCondor as a Compute Entrypoint

- By default, provides GSI authentication (authN) and uses HTCondor security for **authorization** (authZ)
- HTCondor-CE 4 (available in the development repository) iterates on the default authentication model:
  - GSI authN is still supported but SciTokens/WLCG JWTs are preferred if presented by a client (and you're using HTCondor 8.9)
  - HTCondor-CE daemons authenticate with each other using local filesystem authN instead of GSI!

HTCondor as a Compute Entrypoint

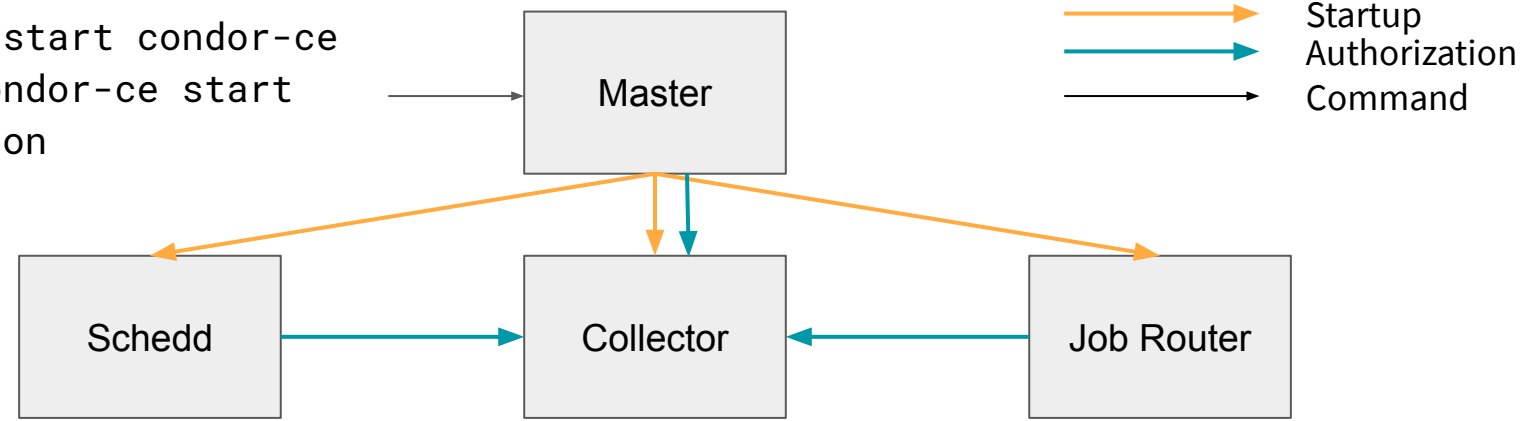
- Supports interaction with the following **resource layers**...
  - HTCondor batch systems directly
  - Slurm, PBS Pro/Torque, SGE, and LSF batch systems
  - Also with all of the above via SSH
- Non-HTCondor batch systems and SSH submission are supported via the HTCondor GridManager daemon and the Batch ASCII Language Helper Protocol (BLAHP)
  - Takes the routed job and further transforms it into your local batch's JDL
  - Specific Job ClassAd attributes result in batch system specific directives, e.g. the `BatchRuntime` attribute results in `#SBATCH --time ...` for Slurm
  - Queries the local batch system to pass along job state updates back along the job chain

Job Router Daemon

- The Job Router is responsible for taking a job, creating a copy, and changing the copy according to a set of rules
  - When running an HTCondor batch system, the copy is inserted directly into the batch SchedD. Otherwise, the copy is inserted back into the CE SchedD
  - Each chain of rules is called a “job route” and is defined by a ClassAd
  - Job routes reflect a site’s policy
- Once the copy has been created, attribute changes and state changes are propagated between the source and destination jobs

HTCondor-CE Daemons

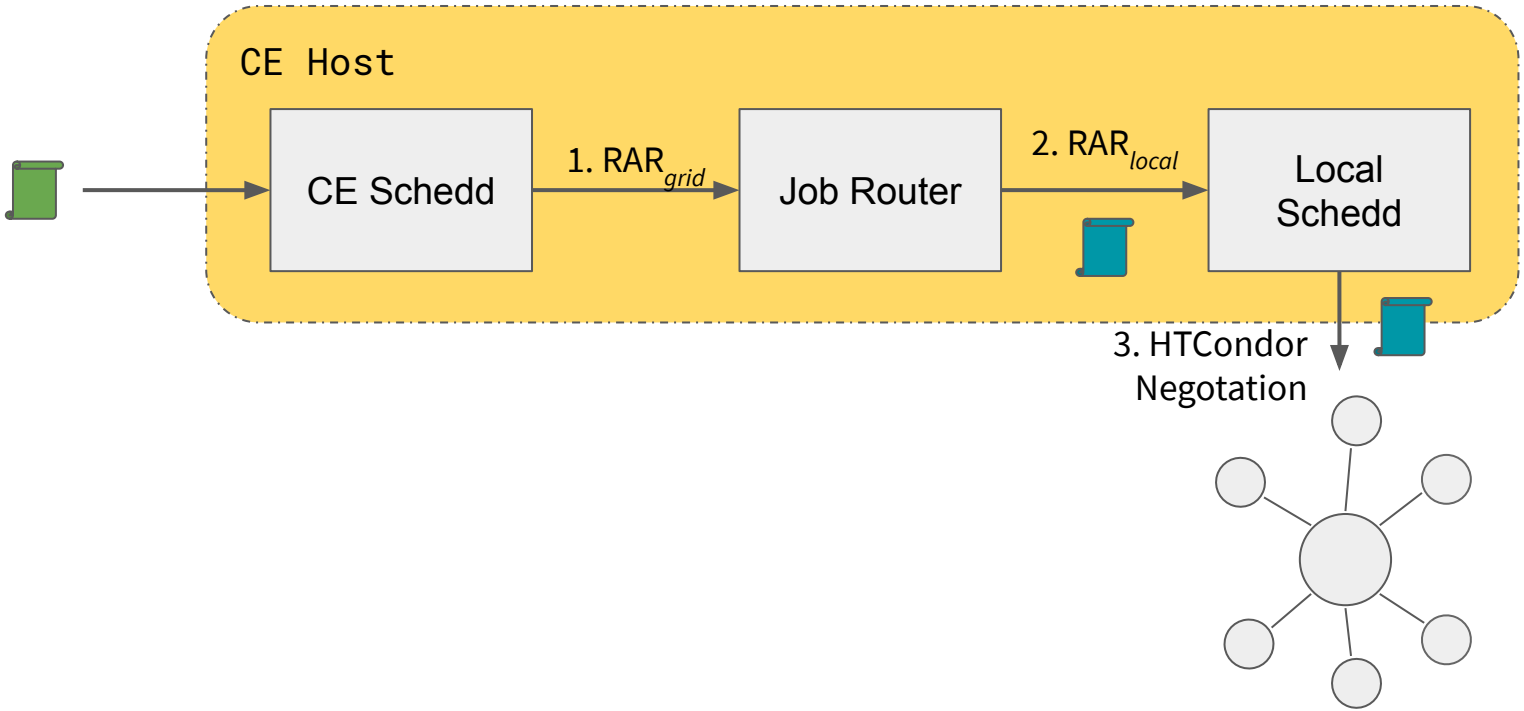
```
systemctl start condor-ce
service condor-ce start
condor_ce_on
```




```
[blin@lhcb-ce ~]$ condor_ce_status -any
MyType             TargetType         Name

Collector          None               My Pool - lhcb-ce.chtc.wisc.edu@lhcb-ce.c
Job_Router         None               htcondor-ce@lhcb-ce.chtc.wisc.edu
Scheduler          None               lhcb-ce.chtc.wisc.edu
DaemonMaster       None               lhcb-ce.chtc.wisc.edu
Submitter          None               nu_lhcb@users.htcondor.org
```

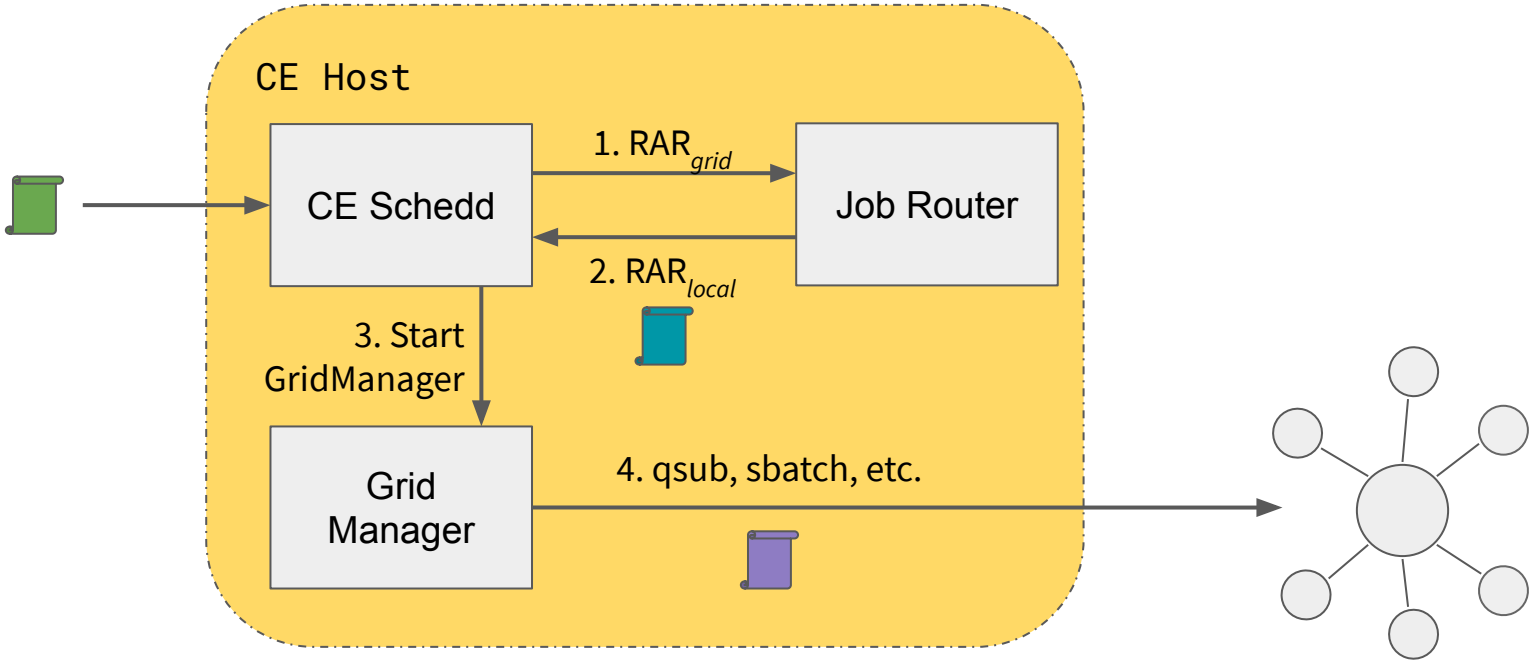
HTCondor-CE + HTCondor Batch System



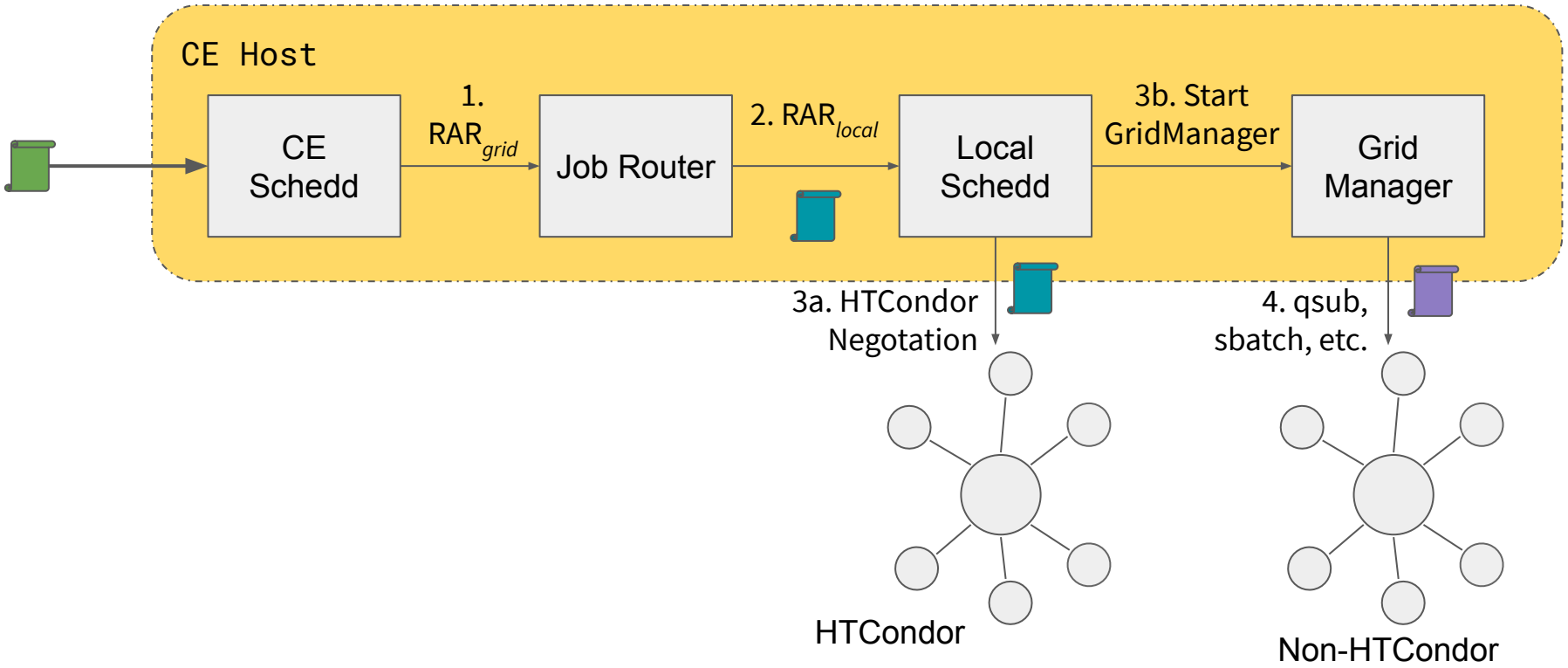
HTCondor-CE + Non-HTCondor Batch System

- Since there is no local batch system SchedD, jobs are routed back into the CE SchedD as “Grid Universe” jobs
- Grid Universe jobs spawn a Gridmanager daemon per user with log files: `/var/log/condor-ce/GridmanagerLog.<user>`
- Requires a shared filesystem across the cluster for pilot job file transfers

HTCondor-CE + Non-HTCondor Batch System



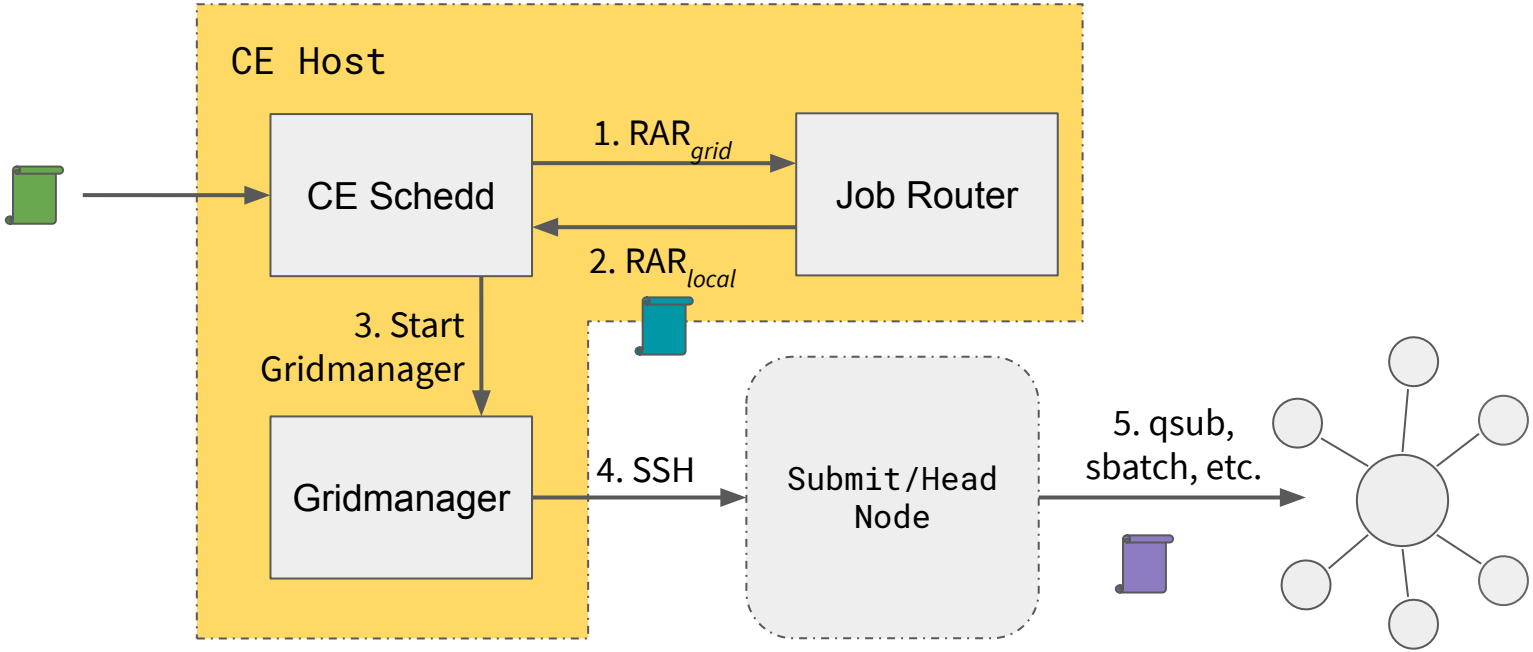
HTCondor-CE + HTCondor + Non-HTCondor



HTCondor-CE + SSH

- Using BOSCO (https://osg-bosco.github.io/docs/), HTCondor-CE can be configured to submit jobs over SSH
  - Requires SSH key-based access to an account on a node that can submit and manage jobs on the local batch system
  - Requires shared home directories across the cluster for pilot job file transfer
- The Open Science Grid (OSG) uses HTCondor-CE over SSH to offer HTCondor-CE as a Service (a.k.a. Hosted CE) for small sites
- Can support up to ~10k jobs concurrently

HTCondor-CE + SSH



HTCondor-CE Requirements

- Open port (TCP) 9619
- Shared filesystem for non-HTCondor batch systems for pilot job file transfer
- CA certificates and CRLs installed in `/etc/grid-security/certificates/`
  VO information installed in `/etc/grid-security/vomsdir/`
- Ensure mapped users exist on the CE (and across the cluster)
- Minimal hardware requirements
  - Handful of cores
  - HTCondor backends should plan on ~½ MB RAM per job
- For example, our Hosted CEs run on 2 vCPUs and 2GB RAM

Configuring HTCondor-CE

Authentication and Authorization

- Authentication can be configured via the HTCondor-CE unified mapfile `/etc/condor-ce/condor_mapfile`
  - One mapping per line with the following format:
    <AUTH METHOD> <AUTH NAME> <HTCONDOR PRINCIPLE>
  - Auth names supports perl-compatible regular expressions
  - Selected mapping is determined by first-match
- HTCondor principles (`<USERNAME>@<DOMAIN>`) determine authorization level
  - <hostname>@daemon.htcondor.org: authorized as a daemon
  - .*@users.htcondor.org: authorized to submit jobs
  - GSS_ASSIST_GRIDMAP: a special value telling HTCondor-CE to call out to another service for user mapping, e.g. LCMAPS, Argus
- https://htcondor-ce.readthedocs.io/en/latest/installation/htcondor-ce/#configuring-authentication

Batch System Configuration

- For HTCondor batch systems, specify the locations of your local batch SchedD, Collector, and SPOOL directory
- For non-HTCondor batch systems, configure the BLAHP and configure how you will share the CE SPOOL directory across your batch system
- https://htcondor-ce.readthedocs.io/en/latest/installation/htcondor-ce/#configuring-the-batch-system

Job Router Configuration

- Declare your site policy
- Job routes specify which jobs to consider and how to transform them
- Each route is described with ClassAds
- Job routes are constructed by combining each entry in `JOB_ROUTER_ENTRIES` with the `JOB_ROUTER_DEFAULTS`
- https://htcondor-ce.readthedocs.io/en/latest/batch-system-integration/

```
$ condor_ce_job_router_info
-config
Route 1
Name         : "Local_Condor"
Universe     : 5
MaxJobs      : 10000
MaxIdleJobs  : 2000
GridResource :
Requirements : true
ClassAd      :
    [
[...]
```

Example Job Routes

```
# condor_ce_config_val -name ce1.opensciencegrid.org -pool ce1.opensciencegrid.org:9619 JOB_ROUTER_ENTRIES

[
Name = "COVID19_Jobs";
TargetUniverse = 5;
Requirements = (IsCOVID19 =?= True);
set_ProjectName = "COVID19_WeNMR";
]
[
Name = "Non_COVID19_Jobs";
TargetUniverse = 5;
set_ProjectName = "WeNMR";
]
```

Job Router Matching

- By default, each job is compared to each job route's requirements expression (`Requirements = True` by default) in the order specified by `JOB_ROUTER_ROUTE_NAMES`
- To use round-robin matching behavior, set the following in your configuration (not within the routes):
  `JOB_ROUTER_ROUND_ROBIN_SELECTION = True`

Job Router Transformations

Special job route functions are used to transform jobs, evaluated in the following order.

1. Copy an attribute from the original job ad to the routed job ad:

```
copy_foo = "original_foo";
```

2. Delete an attribute from the original job ad from the routed job ad:

```
delete_foo = True;
```

3. Set an attribute in the routed job ad to a value or expression

```
set_requirements = (OpSys == "LINUX");
```

4. Set an attribute in the routed job ad to value that is evaluated in the context of the original job ad.

```
eval_set_Experiment = strcat("cms.", Owner);
```

Grid Service Integration

Pilot Factories

- Production HTCondor-CEs in the US have been proven to work with Dirac, GlideinWMS, and Harvester
  - NOTE: Dirac pilots are left in the job queue for up to 30 days. HTCondor-CE 4.4.0 adds the optional `COMPLETED_JOB_EXPIRATION` configuration so that you can control how many days completed jobs may remain in the queue
- SciToken and WLCG JWT based pilot submission have been tested by GlideinWMS and Harvester developers with HTCondor-CE
- User payload job auditing is available for pilots that report back to the HTCondor-CE Collector

APEL Accounting

- The `htcondor-ce-apel` RPM contains configuration, scripts, and services for generating APEL batch and blah records
- Scripts key off of configuration on each worker node for scaling factor information
- Then write batch and blah records to `APEL_OUTPUT_DIR` (default: `/var/lib/condor-ce/apel/`) with `batch-` and `blah-` prefixes, respectively
- Only supports HTCondor-CE with an HTCondor batch system
- https://htcondor-ce.readthedocs.io/en/latest/installation/htcondor-ce/#uploading-accounting-records-to-apel

BDII Integration

- The `htcondor-ce-bdii` package contains a script that generates LDIF output for all HTCondor-CEs at a site as well as an underlying HTCondor batch system
- Only supports HTCondor batch systems
- https://htcondor-ce.readthedocs.io/en/latest/installation/htcondor-ce/#enabling-bdii-integration

HTCondor-CE Central Collector

- HTCondor-CE offers a simple information service using the built-in HTCondor View feature to report useful grid information
  - Contact information (hostname/port)
  - Access policy (authorized virtual organizations)
  - What resources can be accessed?
  - Debugging info (site batch system, site name, versions) for humans
- Each HTCondor-CE in a grid can be configured to report information to one or more HTCondor-CE Central Collectors
- New install documentation! https://htcondor-ce.readthedocs.io/en/latest/installation/central-collector/

HTCondor-CE Central Collector

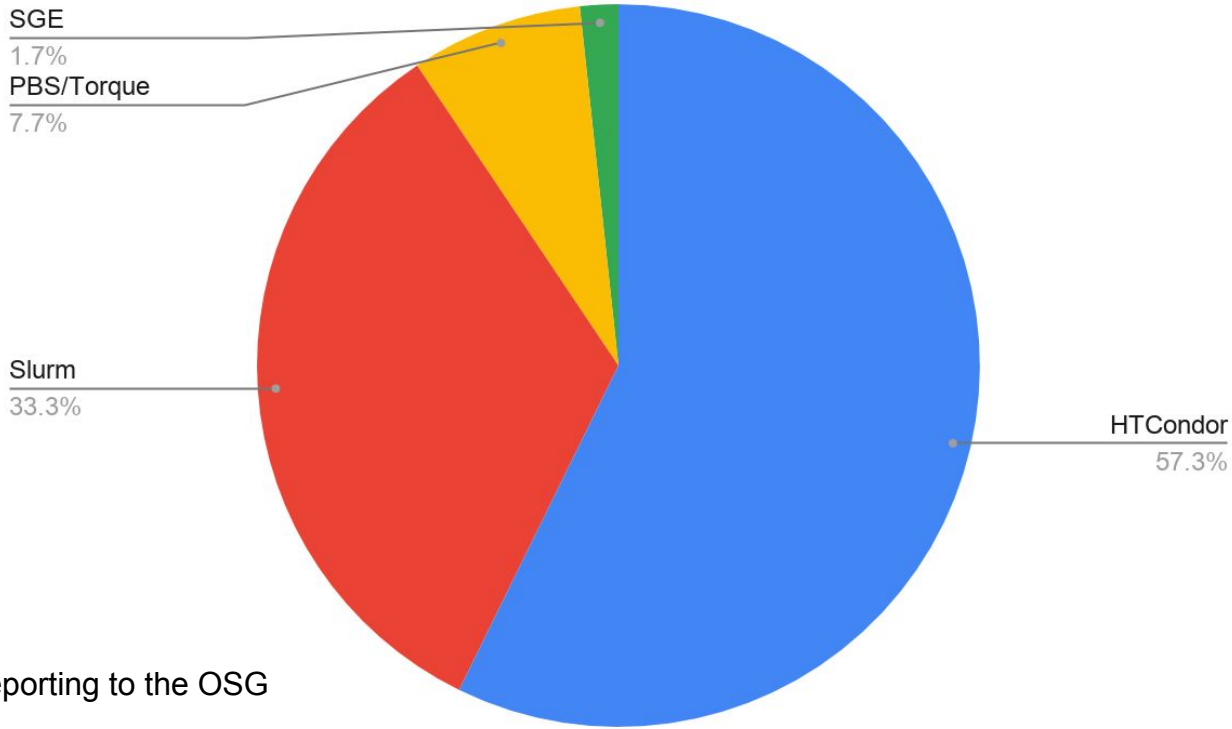
```
# condor_ce_status -schedd -pool collector.opensciencegrid.org:9619
Name                  Resource              Batch   CEVer CondorVer Uptime       Resource

249cc.yeg.cybera.c OSG_CA_CYBERA_EDMO Condor   4.2.1 8.8.8       54+05:37:42 condor 249cc.yeg.cybera.ca 249cc.yeg.cybera.ca:9619
CE01.CMSAF.MIT.EDU MIT_CMS             Condor    3.2.1 8.8.8       11+05:16:27 condor CE01.CMSAF.MIT.EDU CE01.CMSAF.MIT.EDU:9619
CE02.CMSAF.MIT.EDU MIT_CMS_2           Condor    3.2.1 8.8.8       11+04:25:14 condor CE02.CMSAF.MIT.EDU CE02.CMSAF.MIT.EDU:9619
CE03.CMSAF.MIT.EDU MIT_CMS_3           Condor    3.2.0 8.8.8       1+07:31:23 condor CE03.CMSAF.MIT.EDU CE03.CMSAF.MIT.EDU:9619
atlas-ce.bu.edu    NET2                SGE      3.2.1 8.6.13      35+09:19:47 condor atlas-ce.bu.edu atlas-ce.bu.edu:9619
bgk01.sdcc.bnl.gov BNL_BELLE_II_CE_1   Condor    3.2.2 8.8.8       55+07:20:48 condor bgk01.sdcc.bnl.gov bgk01.sdcc.bnl.gov:9619
bgk02.sdcc.bnl.gov BNL_BELLE_II_CE_2   Condor    3.2.2 8.8.8       55+07:39:08 condor bgk02.sdcc.bnl.gov bgk02.sdcc.bnl.gov:9619
brown-osg.rcac.pur Purdue-Brown        SLURM     4.1.0 8.8.8       48+08:14:37 condor brown-osg.rcac.purdue.edu
brown-osg.rcac.purdue.edu:9619
[...]
```

HTCondor-CE Central Collector

```
$ condor_ce_status -schedd -pool collector.opensciencegrid.org:9619 -json
[
{
  "AddressV1": "{[ p=\"primary\"; a=\"18.12.1.31\"; port=9619; n=\"Internet\"; spid=\"323298_41ac_3\"; noUDP=true; ], [
p=\"IPv4\"; a=\"18.12.1.31\"; port=9619; n=\"Internet\"; spid=\"323298_41ac_3\"; noUDP=true; ]}",
  "AuthenticatedIdentity": "ce01.cmsaf.mit.edu@daemon.opensciencegrid.org",
  "AuthenticationMethod": "GSI",
  "Autoclusters": 0,
  "CollectorHost": "CE01.CMSAF.MIT.EDU:9619",
  "CondorPlatform": "$CondorPlatform: X86_64-CentOS_7.5 $",
  "CondorVersion": "$CondorVersion: 8.6.13 Oct 30 2018 $",
  "CurbMatchmaking": false,
  "DaemonCoreDutyCycle": 0.04549036158372677,
  "DaemonStartTime": 1569321031,
  "DetectedCpus": 16,
  "DetectedMemory": 24094,
  "FileTransferDownloadBytes": 0.0,
[...]
```

HTCondor-CE Central Collector



Data from 117 CEs reporting to the OSG Central Collector

Why Use HTCondor-CE

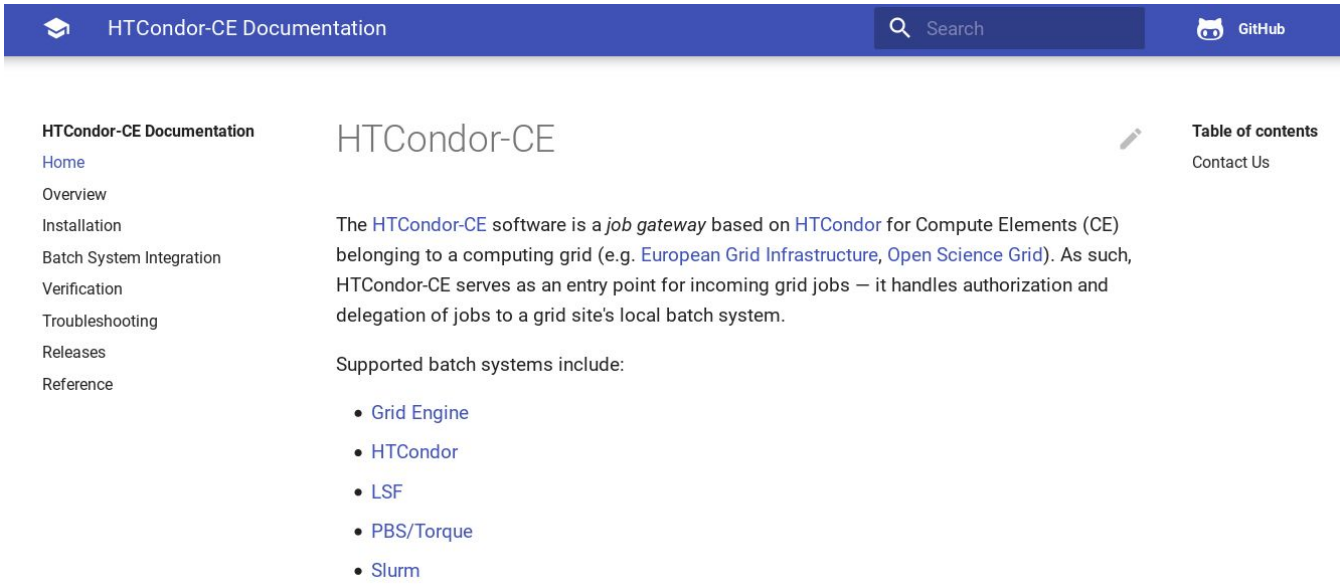
- If you are using HTCondor for batch:
  - One less software provider - same thing all the way down the stack.
  - HTCondor has an extensive feature set - easy to take advantage of it (e.g., Docker universe).
- Regardless, a few advantages:
  - Can scale well (up to at least 16k jobs; maybe higher).
  - Declarative ClassAd-based language.
- But disadvantages exist:
  - Non-HTCondor backends are finicky outside of PBS and Slurm.
  - Declarative ClassAd-based language.

What's Next?

- Features
  - HTCondor-CE Registry: a Central Collector service that facilitates token exchange between site HTCondor-CEs and pilot factories to eliminate the need for site HTCondor-CE host certificates
  - Simplified Job Route configuration language
  - Containers, Helm Charts?
- Events
  - July HTCondor-CE office hours; date and time TBD but will be announced via http://www.htcondor.org and mailing lists: https://research.cs.wisc.edu/htcondor/mail-lists/
  - European HTCondor Week 7-11 September 2020

Getting Started with HTCondor-CE

- Available as RPMs via HTCondor (and OSG) Yum repositories
- Start installation with documentation available via http://htcondor-ce.org



In Conclusion

- Special thanks to EGI for the opportunity to talk; especially Catalin Condurache and Giuseppe La Rocca for all their help!
- The HTCondor team is happy to discuss anything related to HTCondor-CE through our community mailing list: htcondor-users@cs.wisc.edu
- Or contact the HTCondor team directly: htcondor-admin@cs.wisc.edu
- Questions?