

What's new in the HTCondor Software Suite (HTCSS) ? What's coming up?

HTC 23 – Madison, WI

~~Todd Tannenbaum~~

Greg Thain

Center for High Throughput Computing
University of Wisconsin-Madison

Release Channels

- › **Long-Term Support (LTS) Releases** (*formerly 'stable series'*)
 - Only bug fixes
 - vMajor.0.Update (e.g. 10.0.0, 10.0.1, 10.0.2, ...)
 - Today: HTCSS v10.0.8
- › **Feature Releases** (*formerly 'developer series'*)
 - Bug fixes plus new features
 - vMajor.Minor.Update (e.g. 10.1.0, 10.2.0, 10.3.0 ...)
 - Currently: v10.8.0 ***This is the v11.0.0 release candidate!***

Release Channels

- › **Long-Term Support (LTS) Releases** (*formerly 'stable series'*)
 - Only bug fixes
 - vMajor.0.Update (e.g. 10.0.0, 10.0.1, 10.0.2, ...)
 - Today: HTCSS v10.0.6
- › **Feature Releases** (*formerly 'developer series'*)
 - Bug fixes plus new features
 - vMajor.Minor.Update (e.g. 10.1.0, 10.2.0, 10.3.0 ...)
 - Next month: v10.8.0 ***This is the ~~v10.8.0~~ release candidate!***
v23.0.0

Release/Versioning Change

› Why?

- To synchronize support and major release cycles with OSG Software
- Allows HTCSS binaries to be the same between htcondor.org and OSG software (only build/package once)

› New Major version each summer

› Major Version number = year of release (23, 24, 25, ...)

› The two most recent LTS series will be supported e.g.

- Summer 2025: Version 25.0.0 will be released. Support for v24.0.x and v25.0.x; support for 23.0.x will end.

› HTCondor-CE version number will reflect HTCSS version.

Release/Versioning Change

> Why?

- To synchronize support with OSG Software
- Allows HTCSS binaries and OSG software (on

with OSG

condor.org

> New Major version every

> Major Version number

(4, 25, ...)

> The two most recent L

ed e.g.

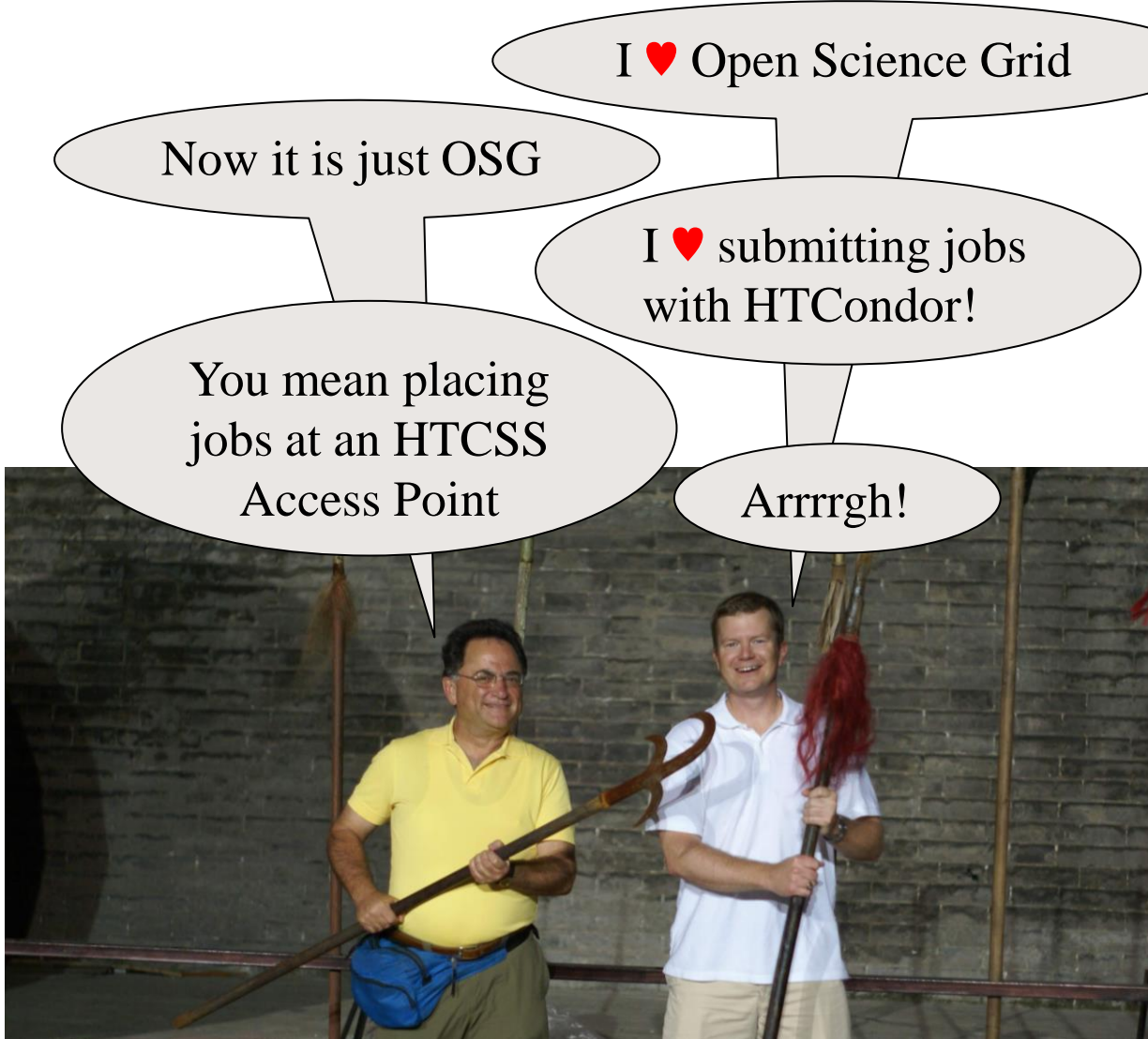
- Summer 2025: Version 25 v25.0.x; support for 23.0.x

v24.0.x and

> HTCondor-CE version number will reflect HTCSS version.



Some New Terminology



> HTCondor Software Suite (HTCSS)

- Access Point (AP)
- Execution Point (EP)
- HTCondor Pool =
Central Manager +
Execution Point(s) (EP)
- HTCondor System = AP + EP + CM
- HTCondor Compute Entrypoint (HTCondor-CE)

Some New Terminology

Software Suite

(AP)

int (EP)

ool =

ager +

oint(s) (EP)

ompute Entrypoint
(E)

Now it is just

You mean
jobs at an H
Access F



Swear Jar



**Used old
HTCondor
Terms Jar**

QUICK START GUIDES

Users' Quick Start Guide
Downloading and Installing
Overview

REFERENCE MANUALS

Users' Manual
Administrators' Manual
ClassAds
Python Bindings

ADDITIONAL DOCS

Grid Computing
Cloud Computing
Chirp – Updates from jobs
Platform-Specific Information
Frequently Asked Questions (FAQ)
Version History and Release Notes

APPENDIX AND TABLES

Command Reference Manual (man pages)
ClassAd Attributes
Codes and Other Needed Values

Glossary

Index

Read the Docs

v: latest ▾

Glossary

AP (Access Point)

An Access Point (AP) is the machine where users place jobs to be queued to be run. It usually runs the *condor_schedd* and other daemons.

Classad

A classad is a set of key value pairs. Every object in an HTCSS is described by a classad. Classad values can also be an expression, which can be evaluated in the context of another classad, in order to provide matching or ranking policy.

CM (Central Manager)

The Central Manager (CM) is the machine with the central in-memory database (*condor_collector*) of all the services, an accountant and *condor_negotiator*.

Daemon

A long-running process often operating in the background. An older term for “service”. The *condor_master*, *condor_collector*, *condor_schedd*, *condor_starter* and *condor_shadow* are some of the daemon in HTCSS.


EP (Execution Point)

The Execution Point (EP), sometimes called the worker node is where jobs run. It is managed by the *condor_startd* daemon, which is responsible for dividing all of the resources the machine into slot.

Job

Job has a very specific meaning in the HTCSS. It is the atomic unit of work in HTCSS. A job is defined by a job classad, which is usually created by *condor_submit* and a submit file. A job can have defined input files, which HTCSS will transfer to the EP. One or more operating system processes can run inside a job. Every job is a member of a cluster of jobs, which have cluster id. Each job also has a “proc id”. The job id uniquely identifies every job on an AP, the id is the cluster id followed by a dot followed by the proc id.

Sandbox



In the past year, there were 12 LTS releases and 10 Feature releases incorporating 140 enhancements and 162 bug fixes.

Since HTCondor W
there were 12 LTS re
10 Feature rele
incorporating
enhancement
and 162 bug fi

See "Detailed Notes" at
<https://htcondor.org/htcondor/release-highlights/>

HTCondor Release Highlights

Software Navigation ▾

Feature Channel

Long Term Support Channel

Feature Channel [Detailed Notes ↗](#)

Feature releases distribute HTCondor's new features and also incorporates bug fixes. Most people should use choose this channel and stay up-to-date with HTCondor's latest features.

10.4.3 - May 9, 2023

- Fix bug than could cause the collector audit plugin to crash

10.4.2 - May 2, 2023

- Fix bug where remote submission of batch grid universe jobs fail
- Fix bug where HTCondor-CE fails to handle jobs after HTCondor restarts

10.4.1 - April 12, 2023

- Preliminary support for Ubuntu 20.04 (Focal Fossa) on PowerPC (ppc64el)

10.4.0 - April 7, 2023

- DAGMan no longer carries the entire environment into the DAGMan job
- Allows EGI CheckIn tokens to be used the with SciTokens authentication

10.3.1 - March 7, 2023

- Execution points now advertise if an sshd is available for ssh to job

10.3.0 - March 6, 2023

- Now evicts OOM killed jobs when they are under their requested memory
- HTCondor glideins can now use cgroups if one has been prepared
- Can write job information an AP history file for every execution attempt
- Can now specify a lifetime for condor_gangliad metrics
- The condor_schedd now advertises a count of unmaterialized jobs

10.2.5 - March 2, 2023

- Fix counting of unmaterialized jobs in the condor_schedd

10.2.4 - February 24, 2023

- Improved counting of unmaterialized jobs in the condor_schedd

10.2.3 - February 21, 2023

- Add a count of unmaterialized jobs to condor_schedd statistics

10.2.2 - February 7, 2023

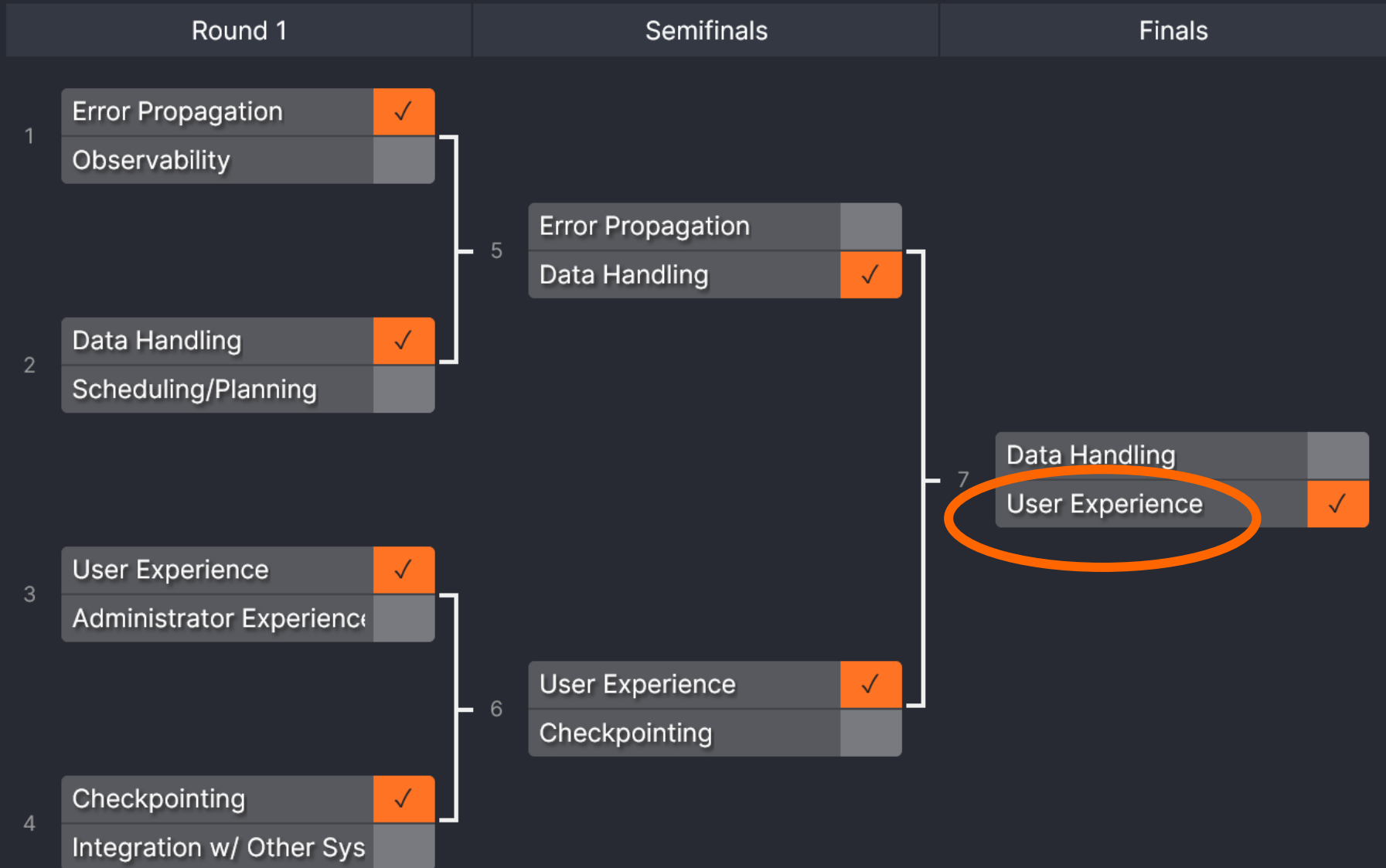
- Fixed bugs with configuration knob SINGULARITY_USE_PID_NAMESPACES

10.2.1 - January 24, 2023

- Improved condor_schedd scalability when a user runs more than 1,000 jobs

HTC23 "Hot Takes" Tournament

What enhancement(s) should HTCSS focus on next?



So, What's Cooking?

Let's start with some User Experience work at the Access Point...

Evolving new command line user interface

› *htcondor* <noun> <verb>

- "*htcondor job submit*", "*htcondor job status*", ...
- "*htcondor dag submit*", "*htcondor dag status*", ...
- "*htcondor jobset submit*", "*htcondor jobset status*". ...
- "*htcondor annex create*", "*htcondor annex status*", ...
- "*htcondor eventlog read*"

› Legacy tools (*condor_q*, *condor_submit*, *condor_history*, ...) not going anywhere...

condor_q - looking at one running job

```
$ condor_q 123.45
```

```
-- Schedd: login04.osgconnect.net : <192.170.231.217:9618?... @ 07/12/23  
21:43:13
```

OWNER	BATCH_NAME	SUBMITTED	DONE	RUN	IDLE	TOTAL	JOB_IDS
toddt	ID: 123	7/12 21:37	_	1	_	10000	123.45

```
Total for query: 1 jobs; 0 completed, 0 removed, 0 idle, 1 running, 0  
held, 0 suspended
```


condor_q - looking at the details

```
$ condor_q -l 123.45
```

```
AccountingGroup = "group_opportunistic.EvolSims.toddt"  
AcctGroup = "EvolSims"  
AcctGroupUser = "anushd"  
AllowedExecuteDuration = 72000  
Args = "44"  
AutoClusterId = 3232  
BytesRecvd = 0.0  
BytesSent = 0.0  
ClusterId = 36371245  
Cmd = "/home/anushd/cluster_code/cwrapper.sh"  
CommittedSlotTime = 0  
CommittedSuspensionTime = 0  
CommittedTime = 0  
CondorPlatform = "$CondorPlatform: X86_64-CentOS_7.9 $"  
CondorVersion = "$CondorVersion: 10.6.0 2023-06-26 PackageID: 10.6.0-0.656423 RC $"  
CoreSize = 0  
CpusProvisioned = 1  
...
```

New Job Status

\$ htcondor job status 123.45

Job 123.45 is currently running on host exec221.chtc.wisc.edu.

It started running again 2.1 hours ago.

It was submitted 3.6 hours ago.

Its current memory usage is 2.5 GB out of 4.0 GB requested.

Its current disk usage is 3.8 GB out of 5.5 GB requested.

It has restarted 2 times.

Goodput is 80% (0.5 hours badput, 2.1 hours goodput).

New Job Status

\$ htcondor job status 123.45

Job 123.45 is currently running on host exec221.chtc.wisc.edu.

It started running again 2.1 hours ago.

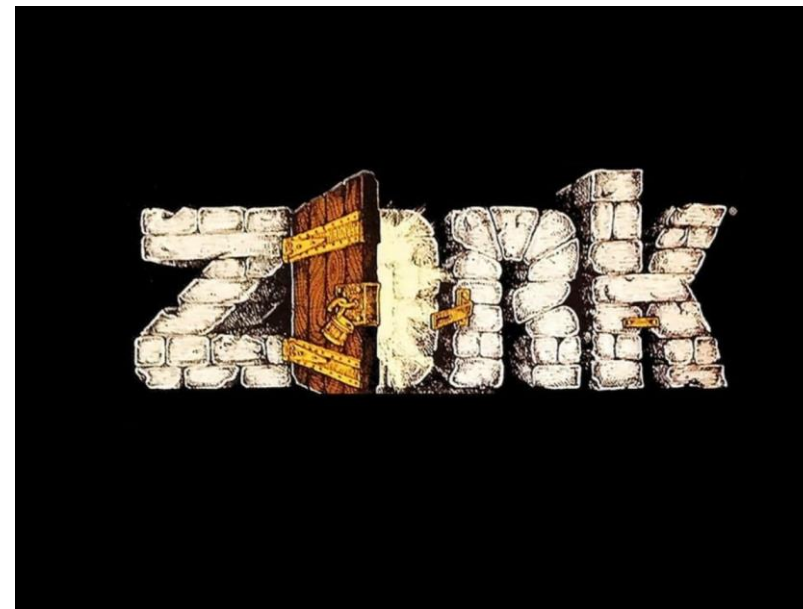
It was submitted 3.6 hours ago.

Its current memory usage is 2.5 GB out of 4.0 GB requested.

Its current disk usage is 3.8 GB out of 5.5 GB requested.

It has restarted 2 times.

Goodput is 80% (0.5 hours badput, 2.1 hours goodput).



What about a DAGMan workflow?

```
$ htcondor dag status 223
```

```
DAGMan Job 223.0 [simple.dag] has been running for 52 days 04:12:46.
```

```
DAG has submitted 382 individual job(s), of which:
```

```
45 are running.
```

```
10 are idle.
```

```
0 are held.
```

```
162 have completed successfully
```

```
DAG has failed nodes but will continue until all possible work is finished:
```

```
5 nodes failed.
```

```
10 nodes waiting to begin.
```

```
24 nodes running.
```

```
Goodput is 92%
```

```
[#####-=====-----] 34% complete.
```

What is this "eventlog" noun?

› *htcondor* <noun> <verb>

- "*htcondor job submit*", "*htcondor job status*", ...
- "*htcondor dag submit*", "*htcondor dag status*", ...
- "*htcondor jobset submit*", "*htcondor jobset status*", ...
- "*htcondor annex create*", "*htcondor annex status*", ...
- "*htcondor eventlog read*"

› Legacy tools (condor_q, condor_submit, condor_history...) not going anywhere...

What is this "eventlog" noun?

> *htcondor* <noun> <verb>

- "*htcondor job submit*", "*htcondor job status*", ...
- "*htcondor dag submit*", "*htcondor dag status*", ...
- "*htcondor jobset submit*", "*htcondor jobset status*", ...
- "*htcondor annex create*", "*htcondor annex status*", ...
- "*htcondor eventlog read*"

> Legacy tools (*condor_q*, *condor_submit*, *condor_history*...) not going anywhere...

See demo by Todd Miller
<https://agenda.hep.wisc.edu/event/1733/contributions/25505/attachments/8274/9664/go>

What is this "eventlog" noun?

› *htcondor* <noun> <verb>

- "*htcondor job submit*", "*htcondor job status*", ...
- "*htcondor dag submit*", "*htcondor dag status*", ...
- "*htcondor jobset submit*", "*htcondor jobset status*", ...
- "*htcondor annex create*", "*htcondor annex status*", ...
- "*htcondor eventlog read*"

› Legacy tools (condor_q, condor_submit, condor_history...) not going anywhere...

Lots of info users/admins can get from AP and EP job history files

```
$ condor_history
```

```
$ condor_history -file startd-hist
```

```
$ condor_adstash
```

Different info in event logs

```
000 (076.000.000) 2023-07-06 16:29:23 Job submitted
from host: <...>.
001 (076.000.000) 2023-07-06 16:29:23 Job executing on
host: <...>...

005 (076.000.000) 2023-07-06 16:30:19 Job terminated.
    (1) Normal termination (return value 0)
        Usr 0 00:00:00, Sys 0 00:00:00
        Usr 0 00:00:00, Sys 0 00:00:00
    0 - Total Bytes Sent By Job
    0 - Total Bytes Received By Job
```

NEW info in event logs w/ v10.6

```
000 (076.000.000) 2023-07-06 16:29:23 Job submitted
from host: <...>.
001 (076.000.000) 2023-07-06 16:29:23 Job executing on
host: <...>...
```

```
SlotName: slot1@example.com
```

```
CondorScratchDir = "/condor/execute/dir_123"
```

```
Cpus = 1
```

```
Disk = 27093557
```

```
Memory = 128
```


THERE'S A KNOB FOR THAT™

```
uilog_execute_attrs = GLIDEIN_Site
```

e.g. GLIDEIN info in event logs

```
000 (076.000.000) 2023-07-06 16:29:23 Job submitted
from host: <...>.
001 (076.000.000) 2023-07-06 16:29:23 Job executing on
host: <...>...
  SlotName: slot1@example.com
  CondorScratchDir = "/condor/execute/dir_123"
  Cpus = 1
  Disk = 27093557
  Memory = 128
GLIDEIN_Site = "BestOSPoolSite"
```

Appears in all 3 event logs

1. Your per-job (workflow?)
set by the `log = some_log_file` in the submit file
2. For a DAGMan job, in the DAGman nodes.log
3. If configured, the global AP events file which has events for all jobs from all users on that AP

Using the most constrained file you need gives you efficiencies
–faster to only query your own workflow log rather than global

Enter htcondor eventlog

Work in progress to read these with htcondor

e.g.

```
$ htcondor eventlog read my_log_file
```

Note new noun "eventlog" in cli

Works on all three kinds of eventlog files

htcondor eventlog read ...

```
$ htcondor eventlog read my_log_file
```

Job	Host	Start Time	Evict Time	Evictions	Wall Time	CPU Usage
79.0	foo	7/6 17:24	7/6 17:25	1	0+00:01:00	0+00:00:52
80.0	foo	7/6 17:35	7/6 17:36	1	0+00:01:00	0+00:00:52
81.0	foo	7/6 17:43	7/6 17:44	1	0+00:01:00	0+00:00:53

And with group-by

```
$ htcondor eventlog read -group-by GLIDEN_Site my_log_file
```

Site	CPU Usage	Job Starts	Job Successes	Job Failures
MWT2	0+01:13:00	59	57	2
NWICG_NDCMS	0+00:50:20	2	1	0
UConn-HPC	0+00:05:00	2	1	0
UColorado_HEP	0+00:51:11	2	1	0
NMSU-Discovery-CE	0+04:24:01	7	5	0

What other nouns should we have?

- › *htcondor* <noun> <verb>
 - "*htcondor ap status*", "*htcondor cm status*", ???
 - "*htcondor wallet add*", "*htcondor wallet remove*", ???
- › Others?
- › What about web interface?
 - Open OnDemand for job placement?
 - Command-line noun verb exercise still useful

Submit File Enhancements

- › Can make "+CustomAttributes" first class with knobs **EXTENDED_SUBMIT_[COMMANDS|HELPPFILE]:**

Instead of:

```
+SiteJobType = "analysis"  
queue
```

Just:

```
SiteJobType = analysis  
queue
```

- › Define job submit templates:

```
use template: Matlab( mymatrix.m )  
queue
```

File Transfer Error Propagation

› Hold Reason and Reason Codes now useful

12 [TransferOutputError]	An error occurred while transferring job output files or checkpoint files.	The Unix errno number.
13 [TransferInputError]	An error occurred while transferring job input files.	The Unix errno number.

Example of "condor_q -hold" (output directory missing on the Access Point)

Old Message:

Error from slot1@TODDS480S: STARTER at 127.0.0.1 failed to send file(s) to <127.0.0.1:50288>; SHADOW at 127.0.0.1 failed to write to file C:\condor\test\not_there\blah: (errno 2) No such file or directory

New Message:

Transfer output files failure at access point SUBMIT1 while receiving files from execution point slot3@NODE5. Details: writing to file C:\condor\test\not_there\blah: (errno 2) No such file or directory

File Transfer Enhancements

- › Submit macro `preserve_relative_paths = True`

```
transfer_input_files = result_data/x
```

ends up creating 'result_data/x' on EP job sandbox, not 'x'.

- › OSDF File Transfer Client comes with HTCSS, so can have out of the box

```
transfer_input_files = osdf:///foo/xxx/yyy
```

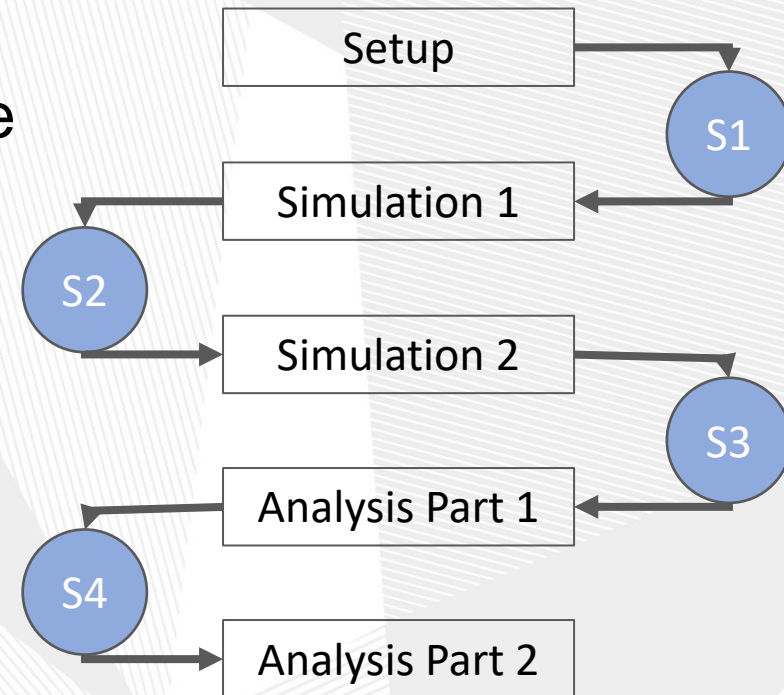
Saved DAG Progress

- Added new saved progress file for a DAG in V10.5.0 that is kind of like a video game save
 - File is similar too a rescue file
 - Written at the first start of a specified node

sample.dag

```
...  
SAVE_POINT_FILE S1  
SAVE_POINT_FILE S2 post_simulation1.save  
SAVE_POINT_FILE S3 ./post_simulation2.save  
SAVE_POINT_FILE S4 ../../foo/mid_analysis.save  
...
```

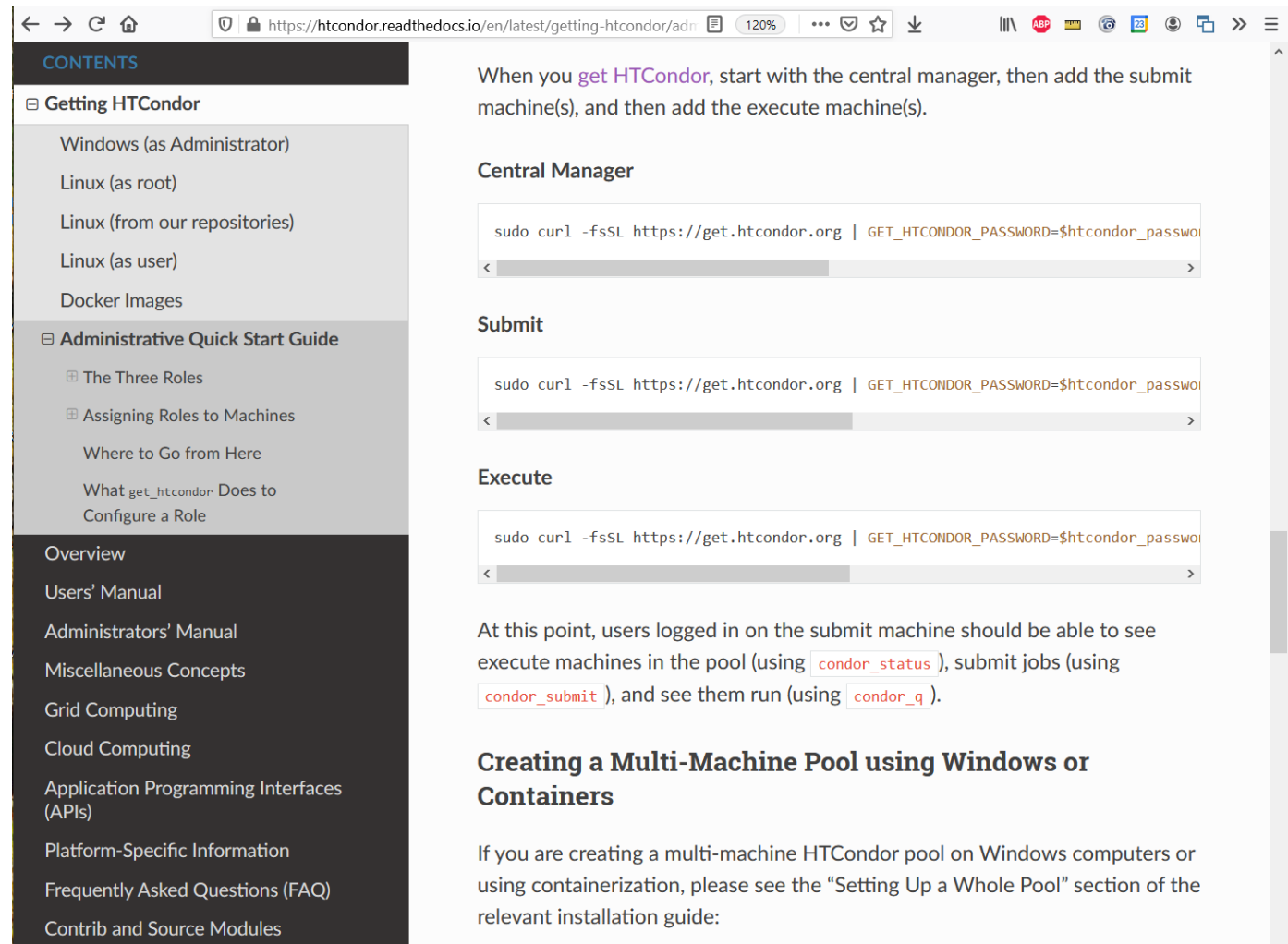
Example Workflow Visualized



Link to [DAGMan Save Point File Documentation](#)

Improve life for administrators as well...

- › Administrative Quick-Start Guide
 - Installs all needed packages
 - Configures a secure HTCondor pool using ID tokens
- › Fast "one-line" install and configure
- › Install an up-to-date secure system including CM, AP, and EP in < 5 min



The screenshot shows a web browser displaying the HTCondor documentation. The left sidebar contains a navigation menu with the following items: CONTENTS, Getting HTCondor (with sub-items: Windows (as Administrator), Linux (as root), Linux (from our repositories), Linux (as user), Docker Images), Administrative Quick Start Guide (with sub-items: The Three Roles, Assigning Roles to Machines, Where to Go from Here, What get_htcondor Does to Configure a Role), Overview, Users' Manual, Administrators' Manual, Miscellaneous Concepts, Grid Computing, Cloud Computing, Application Programming Interfaces (APIs), Platform-Specific Information, Frequently Asked Questions (FAQ), and Contrib and Source Modules. The main content area is titled 'When you get HTCondor, start with the central manager, then add the submit machine(s), and then add the execute machine(s)'. It includes three sections: 'Central Manager' with a terminal command `sudo curl -fsSL https://get.htcondor.org | GET_HTCONDOR_PASSWORD=$htcondor_passwo`, 'Submit' with the same command, and 'Execute' with the same command. Below the commands, it states: 'At this point, users logged in on the submit machine should be able to see execute machines in the pool (using `condor_status`), submit jobs (using `condor_submit`), and see them run (using `condor_q`).' The final section is 'Creating a Multi-Machine Pool using Windows or Containers', which begins with: 'If you are creating a multi-machine HTCondor pool on Windows computers or using containerization, please see the "Setting Up a Whole Pool" section of the relevant installation guide:'.

Execution Point Developments

P(artitionable) Slots Today

- › pslot is never claimed, which constrains AP planning.
- › AP claims a series of independent dslots made from the pslot
- › AP is dependent on CM for managing pslot resources
 - CM can take resources away and offer them to other APs

Pslot Claiming

- › AP can now claim entire pslots
 - For a set period of time
 - Minus resources still in use by previous claimant
- › This will allow improved resource management
 - AP can manage pslot resources without aid or hinderance of a pool CM
 - Intelligent draining for different sized jobs
 - Less work for CM negotiator
 - CM can be unaware of users at the AP

"Container Universe"

- › EP advertises container runtimes available, and uses whichever one can get the job done
- › Now EP does a lot of testing of container runtime and the container image to determine if errors are the systems fault or the jobs fault



- › New world order:

```
container_image = /cvmfs/my/image/dir/  
# Or container_image = docker://Debian  
# Or container_image = myImage.sif  
# Or container_image = http://xxx/image.sif
```



Containers, cont.

- › First class EP support for a default container image to use
- › Deal with Docker Hub banning
- › Package Apptainer into HTCSS EP Product

GPU Scheduling Activities

- › HTCondor has long been able to
 - detect GPU devices and schedule GPU jobs
 - monitor/report job GPU processor utilization
 - monitor/report job GPU memory utilization
- › Now also support for heterogenous GPUs in one server
 - E.g. a server with two different models of GPU cards
 - NVIDIA Multi-Instance GPU (MIG) partitioning
- › **Currently on the radar:**
 - Working on concurrent jobs on one device
 - Dealing with locked-up GPUs (dead slots?)
 - Controlling exposure of GPU devices w/ device namespaces



Submit File Example:

```
Executable = foo.exe  
RequestGPUs = 1  
RequireGPUs =  
    Capability > 7.0  
Queue
```

First-class Backfill pslots

- › Motivating Scenario: I want my GPU rich server to give priority to GPU jobs, but backfill with CPU-only jobs
- › A p-slot provisioned from a shadow set of resources that tracks contention with the primary set of resources
 - `SLOT_TYPE_<N>_BACKFILL = TRUE`
- › BACKFILL slots have special attributes
 - `BackfillSlot = true`
 - `ResourceConflict = "Memory, GPUs, GPU-aabbccdd"`
- › `ResourceConflict` given the names of the resources that contend with an active primary slot

Ex: Backfill the CPUs on a GPU node

```
# make a TYPE_1 primary P-slot and give it all of the resources
#
use FEATURE : GPUs
use FEATURE : PartitionableSlot(1, 100%)
SLOT_TYPE_1_START = TARGET.RequestGpus > 0

# make a TYPE_2 backfill P-slot with 90% of the shadow resources and no GPUs
#
SLOT_TYPE_2_BACKFILL = true
use FEATURE : PartitionableSlot(2, 90%, GPUs=0)
SLOT_TYPE_2_PREEMPT = size(ResourceConflict?:"") > 0

# The backfill slot should only run jobs that opt in as BackfillJob
SLOT_TYPE_2_START = TARGET.BackfillJob
```

Will likely expose  as "use policy: PreferGPUJobs"

Better Job Disk Space Management

- › Config knob **STARTD_ENFORCE_DISK_LIMITS = True**
 - Create an ephemeral filesystem for the job sandbox on the EP.
 - Uses LVM or makes a loop-back file system
 - Reserves space on disk for the job
 - Enables EP to perform fast usage queries and cleanup

Couple other bits of goodness

- › No longer "rename" job executable to condor_exec.exe w/ file transfer
- › Better OOM Killer handling

HTCondor-CE Activities

- › HTCondor-CE Dashboard. Admin can point their browser at the CE and see how their site is being used, e.g.:
 - Current number of active and idle glideins
 - Usage by project
- › Added ability to distinguish when the target batch system is unreachable or not functioning.
 - When this is the case, HTCondor marks the resource as unavailable instead of putting impacted jobs on hold.
 - Grid ads in the collector give details about why the remote scheduling system is considered unavailable.
GridResourceUnavailableTimeReason, GridResourceUnavailableTimeReasonCode

Plus even more...

- › Data Enhancements
- › Working on managing SPOOL usage with LOTMAN
 - Protection for delegated transfers
 - Checkpoint to OSDF
 - Pelican Integration in the months to come
- › Security Enhancements
 - Plugin capability for token mapping
 - SSL proxy cert support without GSI
 - Need to improve token UX beyond condor_submit (e.g. dagman)

Recent OS and Architecture Additions

› Enterprise Linux 8

- Processors:
 - Intel/AMD ("x86_64")
 - ARM ("aarch64")
 - Power PC ("ppc64le")

› Enterprise Linux 9 ☺ ☺ ☺

- Required work to support cgroups v2 and OpenSSL 3
- Processors:
 - Intel/AMD ("x86_64")
 - ARM ("aarch64")

Future of Enterprise Linux rebuilds (aka Alma, Rocky) uncertain... ☹

Thank you!

Follow us on Twitter!
<https://twitter.com/HTCondor>



This work is supported by NSF under Cooperative Agreement OAC-2030508 as part of the PATh Project. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF

PATh PARTNERSHIP to ADVANCE
THROUGHPUT
COMPUTING