

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



A lot to cover, we'll go fast...

... buckle up!

Highlights on the web, full details in the Manual

Highlights:

<https://htcondor.org/htcondor/release-highlights/>

Details:

<https://htcondor.readthedocs.io/en/latest/version-history/index.html>

- Documented all the new features / mechanisms that have been added at each version
- Notes about "gotchas" when upgrading from version X to Y

Administrator Upgrade Help Script

- › Besides information in the Manual about incompatibilities when upgrading versions, wouldn't it be great if a tool existed to help admin check the gotchas? We listened...
- › New tool [condor_upgrade_check](#) assists administrators upgrading an HTCondor installation between major version (i.e. V23 -> V24)
- › Intended for use by those running LTS release of HTCondor.
- › Checks current installation and setup for well known incompatibilities and informs administrator of actions to take to resolve said issues.

Feedback from the community:

"We use the manual a lot"

- could you please improve it?

"Especially the index, heavily used"

+ Dictum from Miron:

"HTCondor is a Software Suite: AP, EP, CM are separate products"

10.X Admin manual was..

Administrators' Manual

Introduction

Starting Up, Shutting Down,
Reconfiguring, and Restarting
HTCondor

Introduction to Configuration

Configuration Templates

Configuration Macros

User Priorities and Negotiation

Policy Configuration for Execute
Hosts and for Submit Hosts

Security

Networking (includes sections on
Port Usage and CCB)

DaemonCore

Hooks, Startd, Cron, and Schedd

Cron

Logging in HTCondor

Monitoring

The High Availability of Daemons

» Administrators' Manual

Administrators' Manual

- Introduction

- The Different Roles a Machine Can Play
- The HTCondor Daemons

- Starting Up, Shutting Down, Reconfiguring, and Restarting HTCondor

- Using HTCondor

- Introduction to Configuration

- HTCondor Configuration Files
- Ordered Evaluation to Set the Configuration
- Configuration File Macros
- Comments and Line Continuations
- Multi-Line Values
- Executing a Program to Produce Configuration Macros
- Including Configuration from Elsewhere

Random Topic

Random Topic

Random Topic

Random Topic

24.x Admin manual

The image shows a screenshot of the 'Administrators' Manual' website. On the left is a navigation sidebar with a tree view of the manual's contents. On the right is the main content area showing the 'Administrators' Manual' page with a table of contents. Three red callout boxes with arrows point to specific subsections: 'EP Subsection' points to 'Configuration for Execution Points', 'AP Subsection' points to 'Configuration for Access Points', and 'CM Subsection' points to 'Starting Up, Shutting Down and Reconfiguring the System'.

Administrators' Manual

- Introduction
- Starting Up, Shutting Down and Reconfiguring the System
- Introduction to Configuration
- Configuration Macros
 - Configuration for Execution Points** (EP Subsection)
 - Configuration for Access Points** (AP Subsection)
 - Configuration for Central Managers** (CM Subsection)
- Security
- Networking, Port Usage, and CCB
- Files, Directories and Logs
- Third Party/Delegated file, credential and checkpoint transfer
- ClassAds
- DAGMan Workflows
- Python Bindings

Administrators' Manual

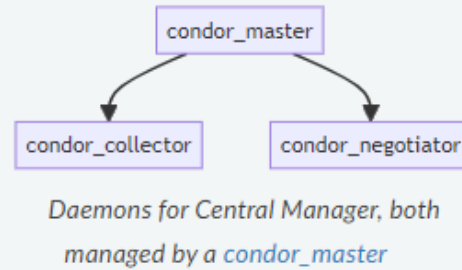
- [Introduction](#)
 - [The Different Roles a Machine Can Play](#)
 - [Putting it all together](#)
 - [The HTCondor Daemons](#)
- [Starting Up, Shutting Down and Reconfiguring the System](#) (CM Subsection)
 - [Daemons That Do Not Run as root](#)
 - [Remote Management Features](#)
 - [DaemonCore](#)
- [Introduction to Configuration](#)
 - [HTCondor Configuration Files](#)
 - [Ordered Evaluation to Set the Configuration](#)

Added diagrams to manual

Central Manager

There can be only one central manager for the pool. This machine is the collector of information, and the negotiator between resources and resource requests. These two halves of the central manager's responsibility are performed by separate daemons, so it would be possible to have different machines providing those two services. However, normally they both live on the same machine. This machine plays a very important part in the HTCondor pool and should be reliable. If this machine crashes, no further matchmaking can be performed within the HTCondor system, although all current matches remain in effect until they are broken by either party involved in the match. Therefore, choose for central manager a machine that is likely to be up and running all the time, or at least one that will be rebooted quickly if something goes wrong. The central manager will ideally have a good network connection to all the machines in the pool, since these pool machines all send updates over the network to the central manager.

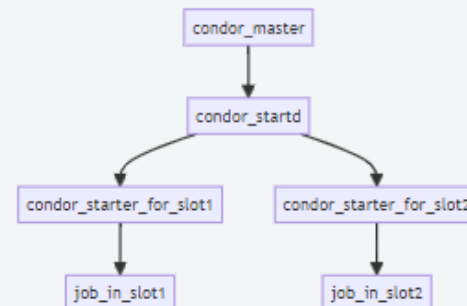
Central Manager (CM) Diagram



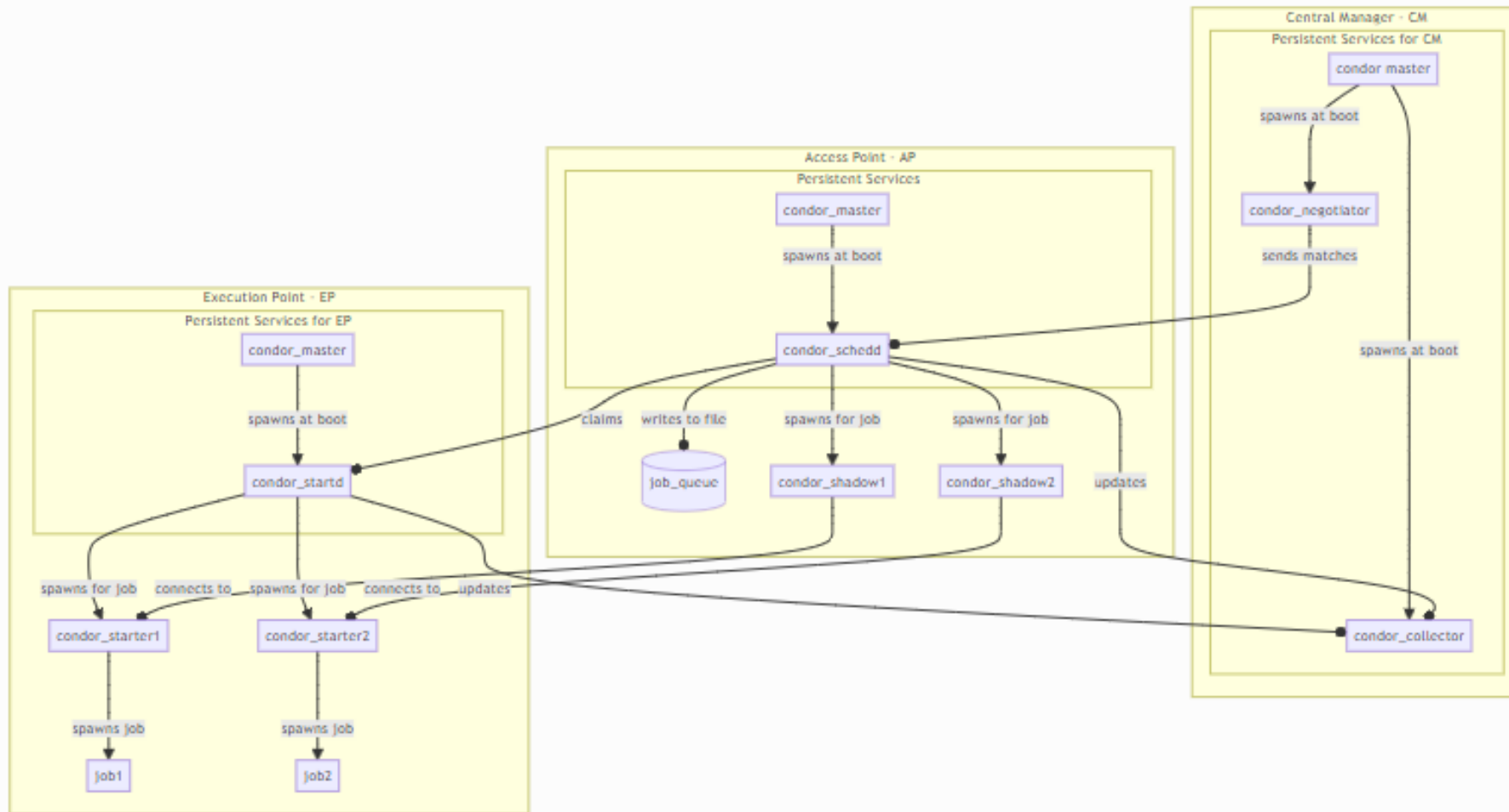
Execution Point

Any machine in the pool, including the central manager, can be configured as to whether or not it should execute HTCondor jobs. Obviously, some of the machines will have to serve this function, or the pool will not be useful. Being an execute machine does not require lots of resources. About the only resource that might matter is disk space. In general the more resources a machine has in terms of swap space, memory, number of CPUs, the larger variety of resource requests it can serve.

Execution Point (EP) Diagram



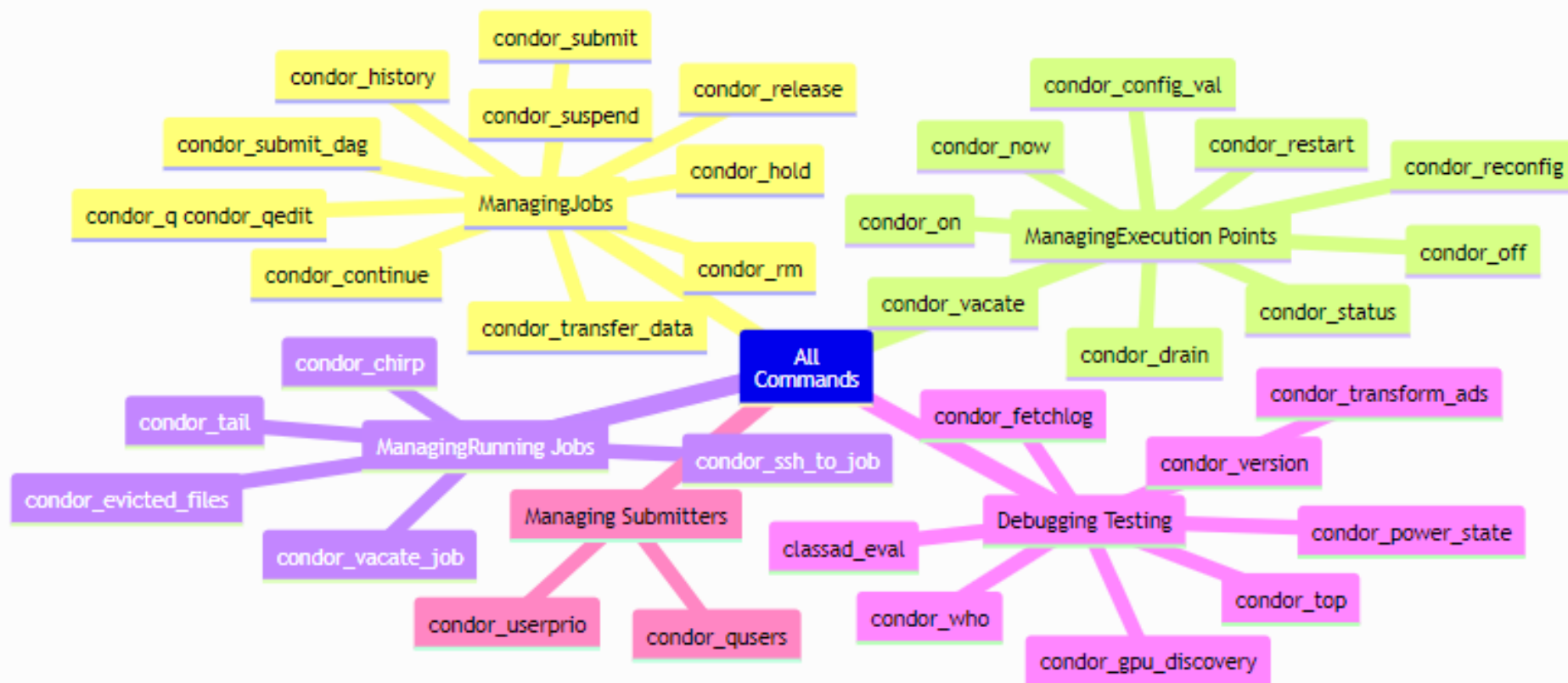
Putting it all together



HTCondor Process Architecture

Commands Reference (man pages)

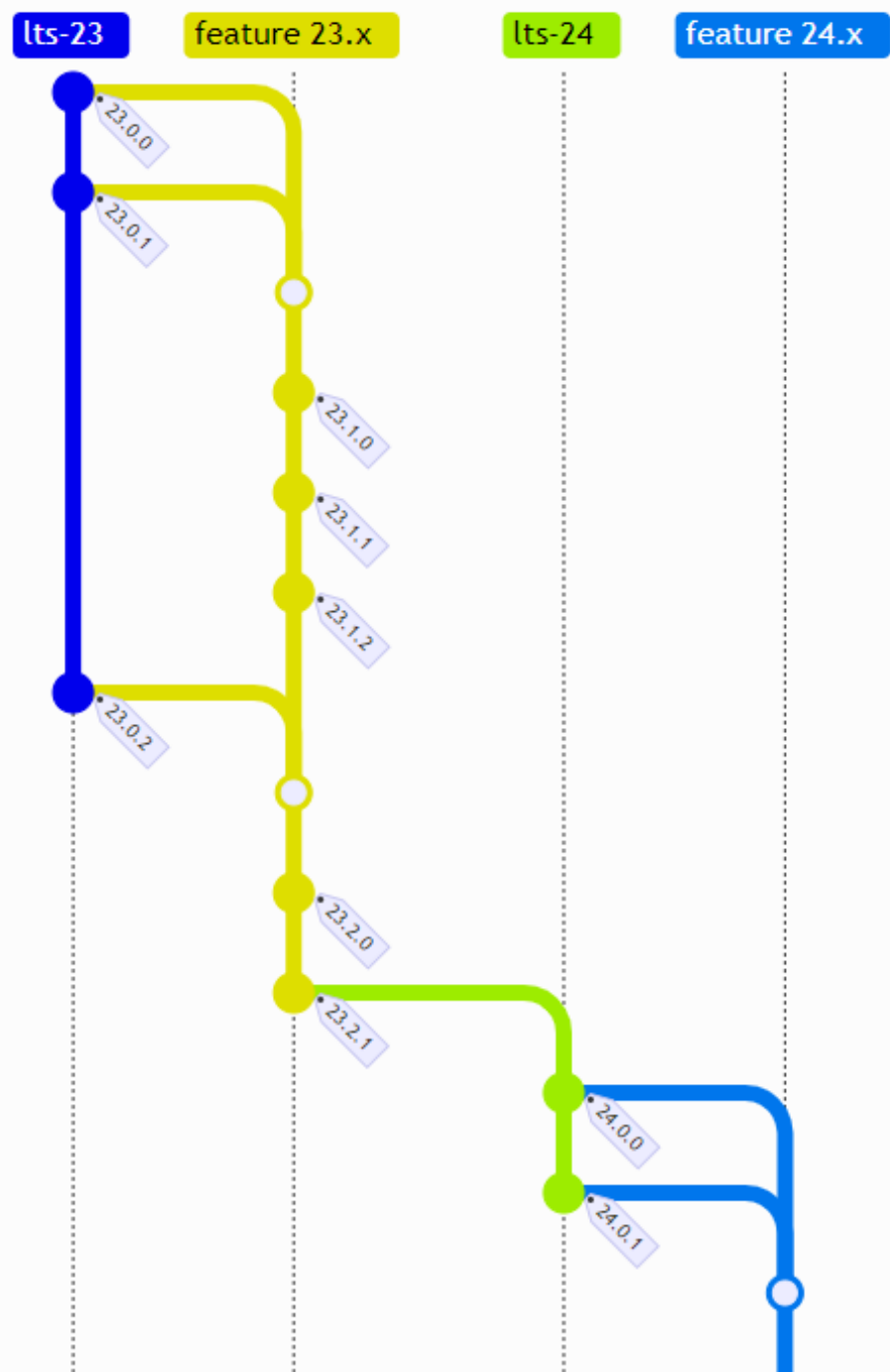
HTCondor ships with many command line tools. While the number may seem overwhelming at first, they can be divided into a few groups:



A map of all the tools

- Users' Manual
- Administrators' Manual
- ClassAds
- DAGMan Workflows
- Python Bindings
- ADDITIONAL DOCS**
- Cloud Computing
- Grid Computing
- Platform-Specific Information
- Recipes, Examples, and Other Answers

- [-] Version History and Release Notes
 - [-] Introduction to HTCondor Versions
 - [-] HTCondor Version Number Scheme
 - Types of Releases
 - Support Life Cycle
 - Repositories
 - Recommendations
 - Upgrading from an 10.0 LTS version to an 23.0 LTS version of HTCondor
 - Version 23 Feature Releases
 - Version 23.0 LTS Releases
 - Version 10 Feature Releases
 - Version 10.0 LTS Releases
- REFERENCE, GLOSSARY AND INDEX**



Moving recipes to manual

The screenshot shows a web browser window with the URL `htcondor.readthedocs.io/en/main/admin-manual/index.html`. The browser's address bar and tabs are visible at the top. Below the browser, the website's navigation sidebar is on the left, and the main content area is on the right.

HTCondor Manual

Search docs

QUICK START GUIDES

- Users' Quick Start Guide
- Downloading and Installing
- Overview

REFERENCE MANUALS

- Users' Manual
- Administrators' Manual
- ClassAds
- DAGMan Workflows
- Python Bindings

ADDITIONAL DOCS

- Cloud Computing
- Grid Computing
- Platform-Specific Information
- Recipes, Examples, and Other Answers
- Version History and Release Notes

REFERENCE, GLOSSARY AND INDEX

- Commands Reference (man pages)
- ClassAd Attributes

Administrators' Manual

Administrators' Manual

- Introduction
 - The Different Roles a Machine Can Play
 - Putting it all together
 - The HTCondor Daemons
- Starting Up, Shutting Down and Reconfiguring the System
 - Daemons That Do Not Run as root
 - Remote Management Features
 - DaemonCore
- Introduction to Configuration
 - HTCondor Configuration Files
 - Ordered Evaluation to Set the Configuration
 - Configuration File Macros
 - Comments and Line Continuations
 - Multi-Line Values
 - Executing a Program to Produce Configuration Macros
 - Including Configuration from Elsewhere
 - Reporting Errors and Warnings
 - Conditionals in Configuration
 - Function Macros in Configuration
 - Macros That Will Require a Restart When Changed
 - Pre-Defined Macros

Moving recipes to manual

ADDITIONAL DOCS

Cloud Computing

Grid Computing

Platform Specific Information

Recipes, Examples, and Other Answers

Version History and Release Notes

REFERENCE, GLOSSARY AND INDEX

Commands Reference (man pages)

ClassAd Attributes

- HTCondor
- Ordered Events
- Configuration
- Comments
- Multi-Line Comments
- Executing a Job
- Including C++
- Reporting Errors
- Conditional Execution
- Function Macros
- Macros That

Many improvements to index

The image shows a screenshot of the HTCondor website. On the left is a dark navigation sidebar with the following sections:

- QUICK START GUIDES**
 - Users' Quick Start Guide
 - Downloading and Installing
 - Overview
- REFERENCE MANUALS**
 - Users' Manual
 - Administrators' Manual
 - ClassAds
 - DAGMan Workflows
 - Python Bindings
- ADDITIONAL DOCS**
 - Cloud Computing
 - Grid Computing
 - Platform-Specific Information
 - Recipes, Examples, and Other Answers
 - Version History and Release Notes
- REFERENCE, GLOSSARY AND INDEX**
 - Commands Reference (man pages)
 - ClassAd Attributes
 - Codes and Other Needed Values
 - Glossary
 - Index

At the bottom of the sidebar is a button labeled "Read the Docs" and a dropdown menu currently showing "v: main".

The main content area on the right displays a list of index entries, including:

- security
- access point
- ACCOUNTANT_DATABASE_FILE
 - NEGOTIATOR Configuration Options
- ACCOUNTANT_LOCAL_DOMAIN
 - NEGOTIATOR Configuration Options
- accounting
 - groups
- Accounting ClassAd Attribute
 - AccountingGroup
 - AccumulatedUsage
 - BeginUsageTime
 - ConfigQuota
 - IsAccountingGroup
 - LastUsageTime
 - Name
 - Priority
 - PriorityFactor
 - ResourcesUsed
 - SubmitterLimit
 - SubmitterShare
 - WeightedAccumulatedUsage
 - WeightedResourcesUsed
- Accounting (htcondor.AdTypes attribute)
- accounting groups
 - and attribute AcctGroup
 - and DAGMan
 - and negotiation
 - Submit commands
- accounting_group_user
 - and DAGMan
 - and negotiation
 - Submit commands
- AccountingGroup
 - Accounting ClassAd Attribute
- AcctGroup
 - Job ClassAd Attribute
- AcctGroupUser
 - Job ClassAd Attribute
- AccumulatedUsage
 - Accounting ClassAd Attribute
- submit Configuration Options
- APPEND_RANK_VANILLA
 - SUBMIT Configuration Options
- APPEND_REQ_VANILLA
 - SUBMIT Configuration Options
- APPEND_REQUIREMENTES
 - SUBMIT Configuration Options
- APPEND_REQUIREMENTS
 - SUBMIT Configuration Options
- APPTAINER_CACHEDIR
 - environment variables for jobs
- ARC CE
- arc_application
 - definition
 - Submit commands
- ARC_GAHP
 - GRIDMANAGER Configuration Options
- ARC_GAHP_COMMAND_LIMIT
 - GRIDMANAGER Configuration Options
- ARC_GAHP_USE_THREADS
 - GRIDMANAGER Configuration Options
- arc_resources
 - definitions
 - Submit commands
- arc_rte
 - arc_resources
 - Submit commands
- ARCH, [1]
- Arch
 - Machine ClassAd Attribute
- Args
 - Job ClassAd Attribute
 - optional attributes
- Arguments
 - Job ClassAd Attribute
- arguments
 - and DAGMan VARS
 - and jar file
 - example
 - example that fails
 - example with output
 - interactive job
 - Submit commands

case. Unless otherwise noted, configuration values that are expected to be numeric or boolean constants can be any valid ClassAd expression of operators on constants. Example:

```
MINUTE      = 60
HOUR        = (60 * $(MINUTE))
SHUTDOWN_GRACEFUL_TIMEOUT = ($(HOUR)*24)
```



Copy Button!

Cgroup v2 improvements

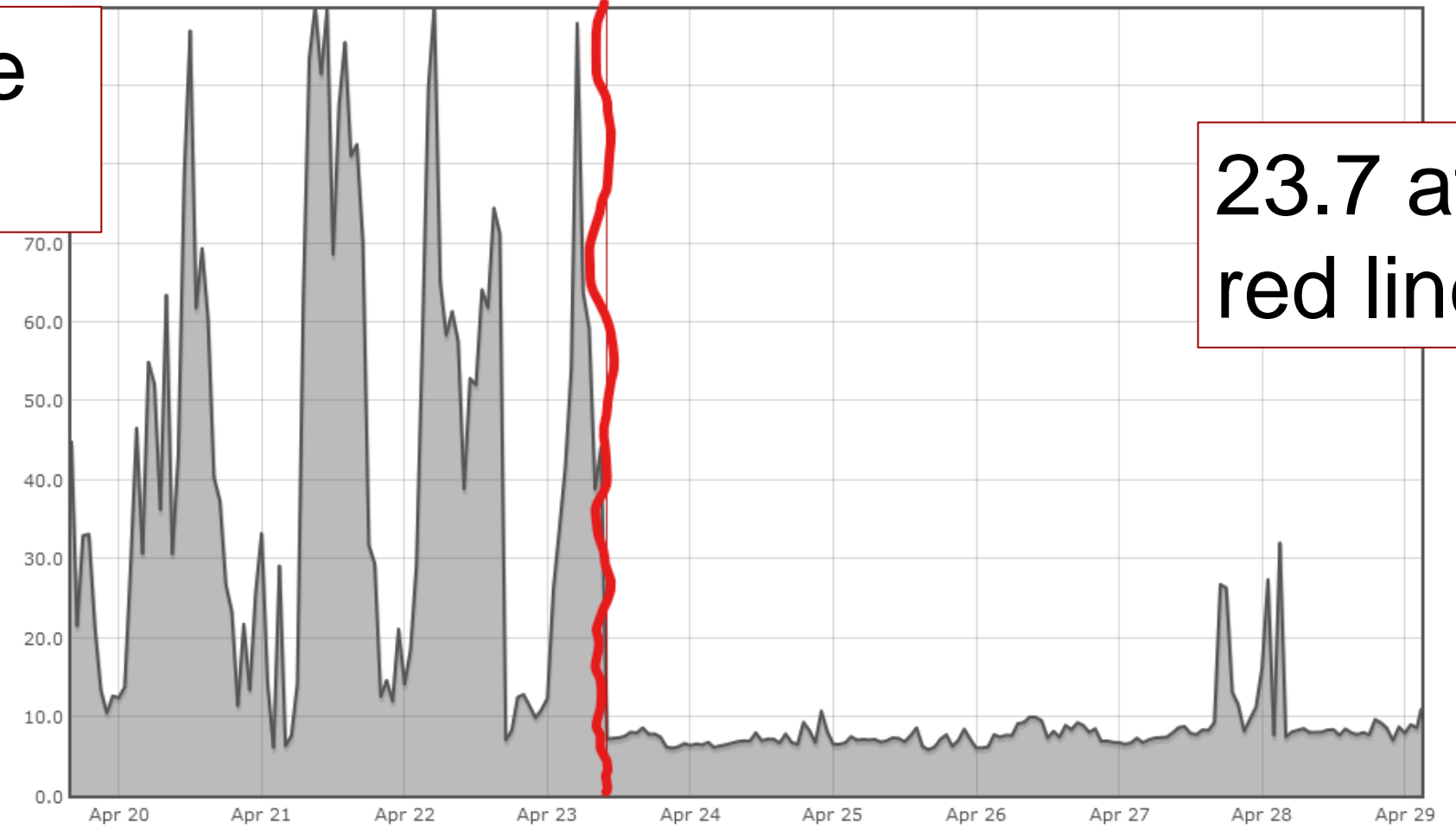
- › "cgroups" = Linux mechanism for placing a limit on resource usage of a job
- › Addition of Cgroup v2 support was Required for EL9 and Debian 12
- › HTCondor 23 gives jobs writeable "delegated" cgroups
 - > Enables "The Dream": glideins can then have better resource management!
- › Local universe jobs now in cgroups
- › Set startd MEMORY to memory limit if it starts in a cgroup
- › Cgroups directly in proc, not via the procd
 - Means we can scale > 256 slots per startd 😊

Improvement in ClassAd memory and CPU usage (OSPool, ~80k slots)

CollectorRecentDaemonCoreDutyCycle

23.6 before
red line

23.7 after
red line



Packaging Updates / New Platforms / OSes

- › Support for Debian 12 (Bookworm)
- › Support for Ubuntu 22.04 (Noble Numbat)
- › Support for **openSUSE** LEAP 15
- › HTCondor Docker images are now based on Alma Linux 9
- › HTCondor Docker images are now available for the **ARM64** CPU architecture
- › Python Bindings for **ARM64** available with PIP
- › Apptainer and Pelican included in the tarballs
- › Debian and RPM packages line up (same names and structures)

File Transfer Enhancements

- › Submit command `preserve_relative_paths = True`
`transfer_input_files = result_data/x`
ends up creating 'result_data/x' on EP job sandbox, not 'x'.
- › **OSDF** (Pelican) File Transfer Client comes with HTCSS, so can use it out of the box

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

Job Epoch History

- › Write the current Job Ad for each run instance of a job to a specified history file.
- › Enable by setting JOB_EPOCH_HISTORY=/path/to/filename
- › Query all job epochs from from history via `condor_history -epochs`
- › Query from the python bindings via `jobEpochHistory()`
- › `condor_adstash` can store job epoch ads in Elastic Search along with history ads

Persistent User records in the schedd

- › A user record must exist for each job owner
 - Created automatically by submitting a job
 - Default behavior, but can be turned off
 - Created by an admin running `condor_qusers -add`
- › New tool : `condor_qusers`
 - Add, enable, disable and edit user records
 - Disabled users cannot submit new jobs
 - Also python bindings for these commands
- › Allows the schedd to separate who owns a job from who to impersonate when running the job.

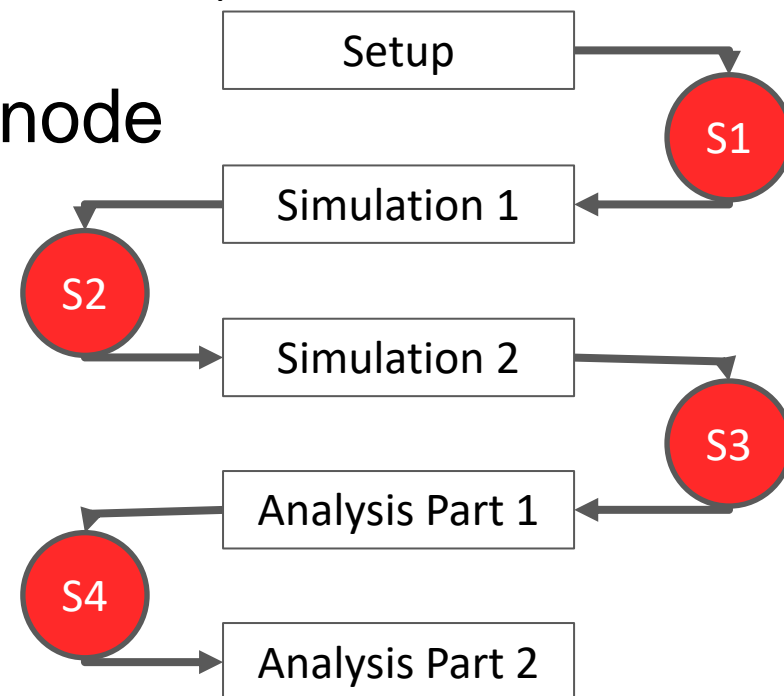
Saved DAG Progress

- › 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](#)

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).

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*"

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

```
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

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 (Pelican) File Transfer Client comes with HTCSS, so can use it out of the box

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

EP Disk Reservation and Enforcement



Allows HTCondor to utilize the Linux Logical Volume Manager to provide isolated work environments and strong disk usage enforcement for jobs executing on EP's.

- No LVM ? EP will create a loopback filesystem on the fly!
- Introduced in v23.5.2
- Requires an HTCondor EP running on Linux as root

Look in the Manual for config knob **STARTD_ENFORCE_DISK_LIMITS**

HTCondor Python Bindings Version 2

- The version 1 bindings depend on an unsupported library (boost.python), so we needed to do something to make sure the bindings would remain available.
- Bindings are intended to be generally compatible; `import htcondor2 as htcondor` should mostly just work.
 - However: We removed some of the things marked as deprecated in the version 1 bindings, and will be deprecating a few other APIs that are not widely used
- Win for users: Compared to version 1, the new version 2 bindings will be much less picky about the specific version of Python interpreter being used 😊

"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 runtimes and the container image to determine if errors are the system's fault or the job's 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
```

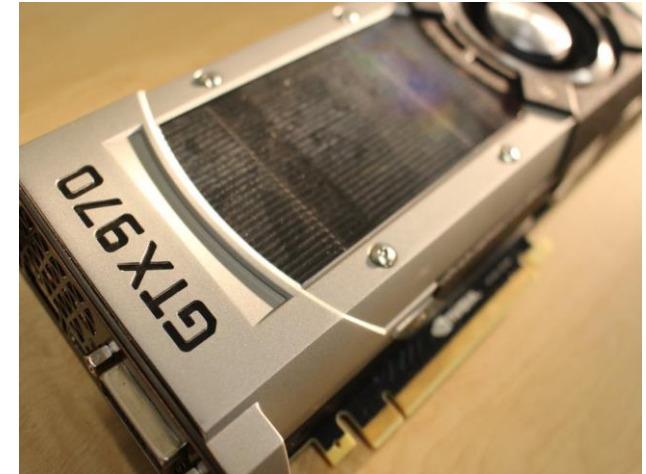


Containers, cont.

- › First class EP support for a default container image to use
 - EP can be configured by the administrator to always run jobs inside a specified container (unless the job brings along its own container)
- › Apptainer is included/packed into HTCSS EP Product
 - Allows an EP to run containers without requiring the admin to install any additional software
- › Todo: Deal with Docker Hub banning

GPU Scheduling Activities

- › HTCondor has long been able to
 - detect GPU devices and schedule GPU jobs
 - monitor/report job GPU utilization
- › **New: Support for heterogenous GPUs in one server**
 - E.g. a server with two different models of GPU cards
 - NVIDIA Multi-Instance GPU (MIG) partitioning
- › **New: Jobs can only see the GPU devices assigned by the EP by using cgroup device namespaces**
- › **New: Support for AMD ROCm**
- › **More first-class keywords in submit file**



Submit File Example:

```
Executable = foo.exe
RequestGPUs = 1
gpus_minimum_capability = 7.0
gpus_minimum_memory = 16GB
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"

CE Improvements

- › CE Dashboard (for hosted CE's at first...)
- › condor_ce_test_token: new tool to generate short lived SciTokens to more easily test resource provisioning requests to the CE
- › condor_ce_upgrade_check: new tool helps look for compatibility problems before upgrading your CE
- › condor_ce_trace: can now test batch system integration without needing to have a SciToken
- › GPUS: Ensure that jobs requesting GPUs land on HTCCondor EPs with GPUs

CE Upgrade to v24

- › There is a new, more powerful syntax for expressing job routes on your CE
 - Must use the new syntax in v24
 - Will include a tool to help you change your route syntax to the new style
 - Details at <https://htcondor.com/htcondor-ce/v6/configuration/job-router-overview/#route-syntaxes>

New defaults – might be surprising

- › Dropping support for multiple queue statements in a single submit file
(Use queue foreach, etc.)
- › Partitionable Slots will be enabled by default (instead of static partitioning)
- › The job's executable will no longer be renamed to 'condor_exec.exe'
- › GPU discovery is enabled on all Execution Points by default

What's Cooking in the Kitchen...



Tight Integration with Pelican (OSDF)

Self-Checkpointing Applications

- Self-checkpointing jobs have been a success, especially on the OS Pool, where runtimes are shorter and frequently interrupted.
- By default, self-checkpointing jobs store their checkpoints on the AP at which they were placed. This adds a lot of file - transfer and -storage load on the AP.
- We've therefore been working on the ability to store checkpoints to third-party services including **Pelican (OSDF)**. We are current testing the initial implementation with OSPool users.

Pelican Local Cache on the EP

- › Cache files coming from OSDF (Pelican) on the EP
 - `use feature: PelicanCache`
 - or `use feature: PelicanCache (size of cache)`
- › Especially thinking about caching Apptainer container images (SIF files)! (as a viable alternative to CVMFS?...)
- › Up next: AP control over which files are cached on the AP or not

Propagation of Errors Pelican->HTCSS Really Bad...

Transfer input files failure at execution point slot1_1@node2228.cluster.lidas.cit while receiving files from access point condor-f5. Details: Error from slot1_1@node2228.cluster.lidas.cit: FILETRANSFER:1:non-zero exit (11) from /usr/libexec/condor/stash_plugin. |Error:

```
(...Path...)P(...Path...)e(...Path...)l(...Path...)i(...Path...)c(...Path...)a(...Path...)n(...Path...) (...Path...)C(...Path...)l(...Path...)i(...Path...)e(...Path...)n(...Path...)t(...Path...)
(...Path...)E(...Path...)r(...Path...)r(...Path...)o(...Path...)r(...Path...):( ...Path...)P(...Path...)r(...Path...)o(...Path...)b(...Path...)l(...Path...)e(...Path...)m(...Path...)
(...Path...)i(...Path...)n(...Path...)i(...Path...)t(...Path...)i(...Path...)a(...Path...)l(...Path...)i(...Path...)z(...Path...)i(...Path...)n(...Path...)g(...Path...)
(...Path...)t(...Path...)h(...Path...)e(...Path...) (...Path...)P(...Path...)e(...Path...)l(...Path...)i(...Path...)c(...Path...)a(...Path...)n(...Path...)
(...Path...)C(...Path...)l(...Path...)i(...Path...)e(...Path...)n(...Path...)t(...Path...)
(...Path...)c(...Path...)o(...Path...)n(...Path...)f(...Path...)i(...Path...)g(...Path...)u(...Path...)r(...Path...)a(...Path...)t(...Path...)i(...Path...)o(...Path...)n(...Path...):( ...Path...)
(...Path...)I(...Path...)n(...Path...)v(...Path...)a(...Path...)l(...Path...)i(...Path...)d(...Path...)
(...Path...)f(...Path...)e(...Path...)d(...Path...)e(...Path...)r(...Path...)a(...Path...)t(...Path...)i(...Path...)o(...Path...)n(...Path...)
(...Path...)v(...Path...)a(...Path...)l(...Path...)u(...Path...)e(...Path...) (...Path...)o(...Path...)s(...Path...)g(...Path...)-
(...Path...)h(...Path...)t(...Path...)c(...Path...).( ...Path...)o(...Path...)r(...Path...)g(...Path...):( ...Path...):( ...Path...)
(...Path...)T(...Path...)i(...Path...)m(...Path...)e(...Path...)o(...Path...)u(...Path...)t(...Path...) (...Path...)w(...Path...)h(...Path...)e(...Path...)n(...Path...)
(...Path...)q(...Path...)u(...Path...)e(...Path...)r(...Path...)y(...Path...)i(...Path...)n(...Path...)g(...Path...)
(...Path...)m(...Path...)e(...Path...)t(...Path...)a(...Path...)d(...Path...)a(...Path...)t(...Path...)a(...Path...):( ...Path...) (...Path...)G(...Path...)e(...Path...)t(...Path...)
(...Path...)h(...Path...)t(...Path...)c(...Path...).( ...Path...)o(...Path...)r(...Path...)g(...Path...)/( ...Path...).( ...Path...)w(...Path...)e(...Path...)l(...Path...)l(...Path...)-
(...Path...)k(...Path...)n(...Path...)o(...Path...)w(...Path...)n(...Path...)/( ...Path...)p(...Path...)e(...Path...)l(...Path...)i(...Path...)c(...Path...)a(...Path...)n(...Path...)-
(...Path...)c(...Path...)o(...Path...)n(...Path...)f(...Path...)i(...Path...)g(...Path...)u(...Path...)r(...Path...)a(...Path...)t(...Path...)i(...Path...)o(...Path...)n(...Path...)h(...Path...):( ...Path...)
(...Path...)c(...Path...)o(...Path...)n(...Path...)t(...Path...)e(...Path...)x(...Path...)t(...Path...)
(...Path...)d(...Path...)e(...Path...)a(...Path...)d(...Path...)l(...Path...)i(...Path...)n(...Path...)e(...Path...)
(...Path...)e(...Path...)x(...Path...)c(...Path...)e(...Path...)e(...Path...)d(...Path...)e(...Path...)d(...Path...)
(...Path...)((...Path...)C(...Path...)l(...Path...)i(...Path...)e(...Path...)n(...Path...)t(...Path...).( ...Path...)T(...Path...)i(...Path...)m(...Path...)e(...Path...)o(...Path...)u(...Path...)t(...Path...)
(...Path...)e(...Path...)x(...Path...)c(...Path...)e(...Path...)e(...Path...)d(...Path...)e(...Path...)d(...Path...) (...Path...)w(...Path...)h(...Path...)i(...Path...)l(...Path...)e(...Path...)
(...Path...)a(...Path...)w(...Path...)a(...Path...)i(...Path...)t(...Path...)i(...Path...)n(...Path...)g(...Path...)
(...Path...)h(...Path...)e(...Path...)a(...Path...)d(...Path...)e(...Path...)r(...Path...)s(...Path...))( ...Path...) (Version: 7.7.4) ( URL file = )
```

Better... but not there yet...

Transfer input files failure at execution point slot1_1@node2228.cluster.ldas.cit while receiving files from access point condor-f5. Details: Pelican Client Error: Attempt #3: from dtn-pas.cinc.nrp.internet2.edu:8443: read tcp 172.16.33.4:45582->163.253.29.17:8443: read: connection reset by peer (0s elapsed, 0s since start); Attempt #2: from its-condor-xrootd1.syr.edu:8443: read tcp 172.16.33.4:39255->128.230.247.232:8443: read: connection reset by peer (0s elapsed, 0s since start); Attempt #1: from osg-new-york-stashcache.nrp.internet2.edu:8443: read tcp 172.16.33.4:37508->163.253.72.2:8443: read: connection reset by peer (0s since start) (Version: 7.9.2; Site: VU-AUGIE)

Simpler User Experience for jobs w/ Tokens:

**Let the AP fetch tokens
from issuers on behalf of
the user**

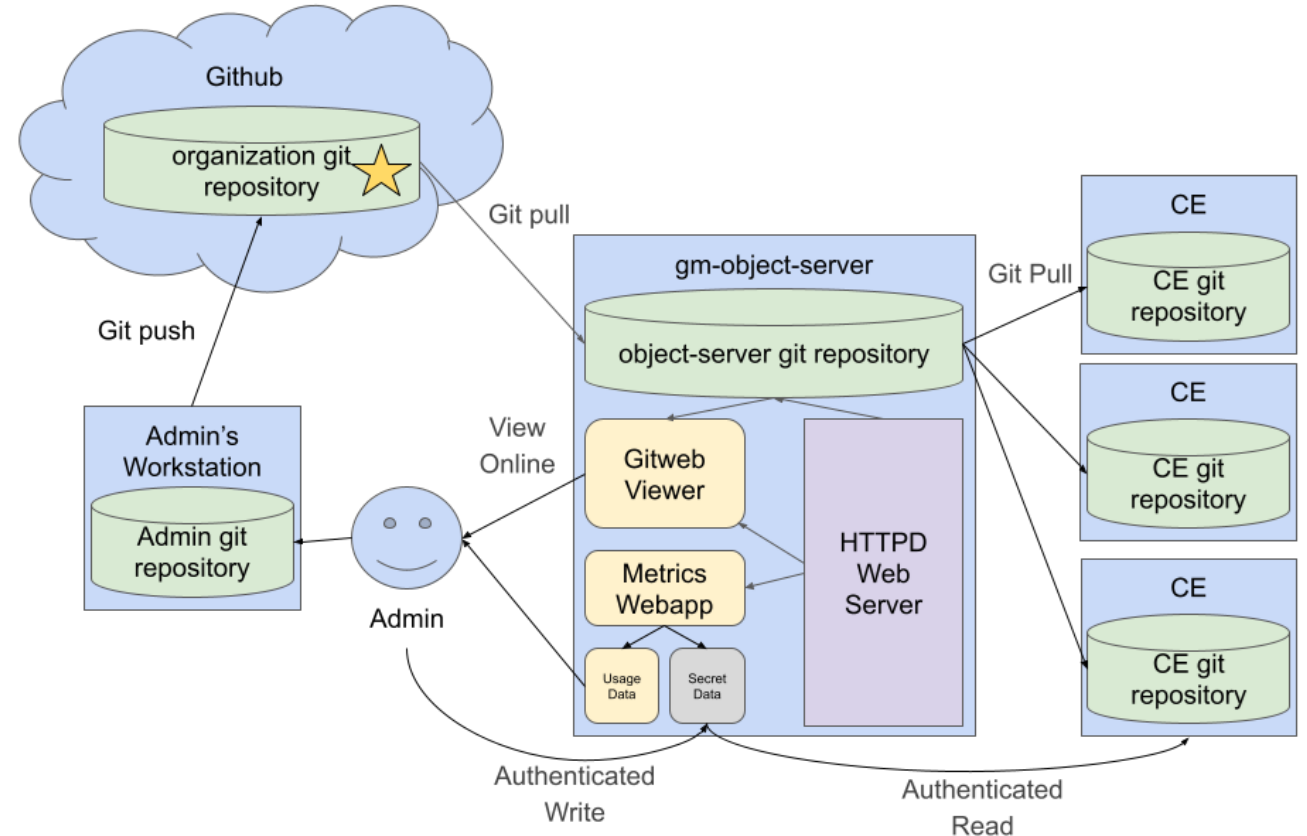
The "Glidein Manager" Activity

The Glidein Manager - Problem Statement

- Capacity in the OSPool is provisioned through input provided by multiple stakeholders
 - Centrally managed Glidein Factory specifies HTCondor and GlideinWMS binaries used across the entire pool
 - Research Organizations apply configuration to Glideins at runtime via the GlideinWMS Frontend
- This leads to complications in system observability and updatability
- The Glidein Manager will provide organizations a central location to specify both binaries and configuration for their Glideins

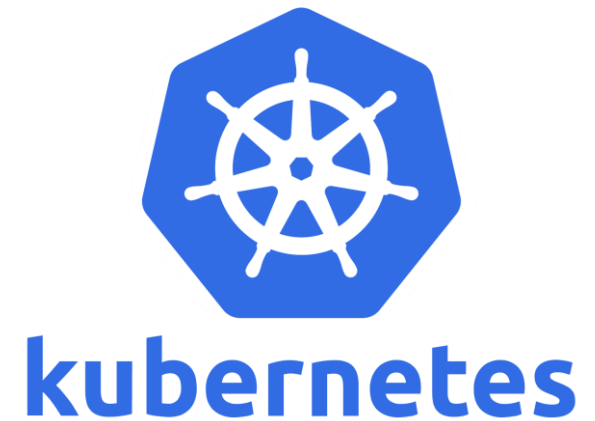
The Glidein Manager

- New service that provides Organizations a central location to specify executable versions and config for their Glideins
- Git Server that mirrors an upstream repository
 - Utilize popular GitOps workflows
- Hosts config files and executable specifications as structured text
- Allows executables and config to be versioned in lockstep via Git commits
- Collects usage metrics and generates reports



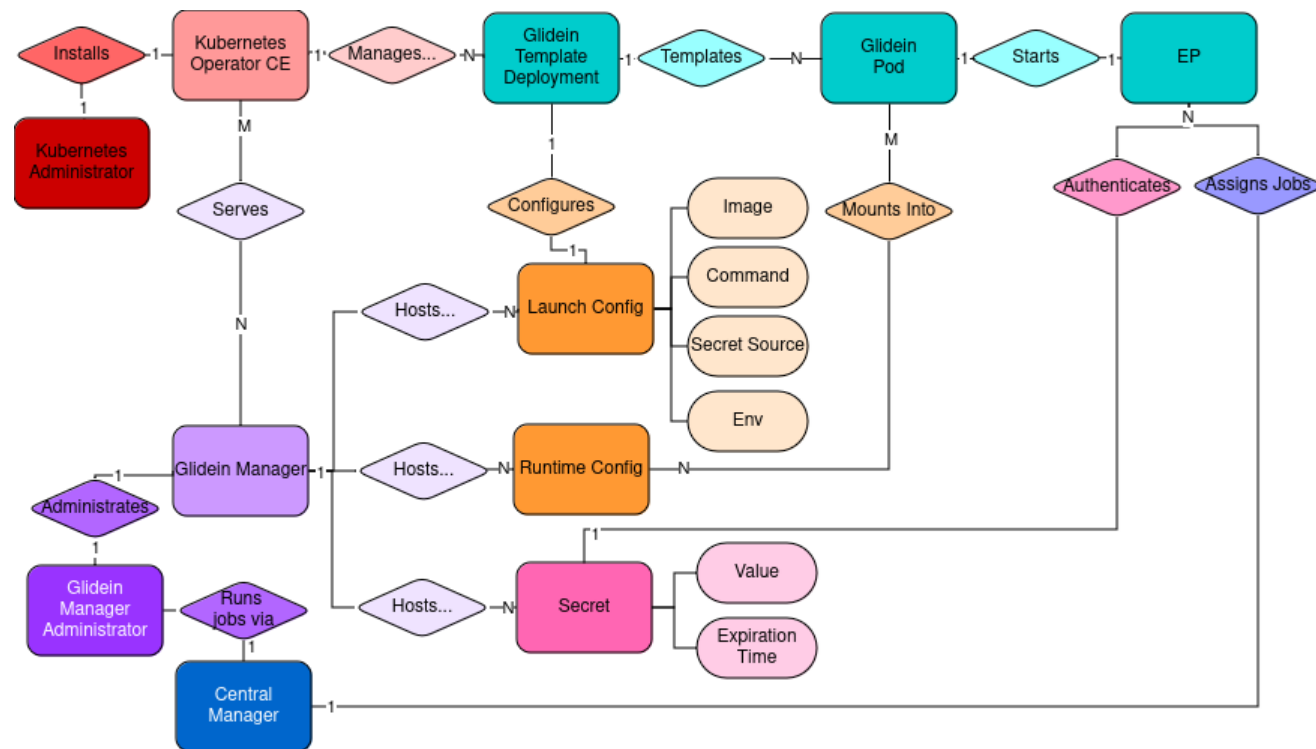
Side Lesson: Kubernetes Operators

- Kubernetes - Popular container orchestration service
- Kubernetes Operator - Privileged container within a cluster that programmatically manages other containers
 - Built in Go via several well-supported Open Source SDKs
- Glideins can be run as containers within a Kubernetes Cluster
- A Kubernetes Operator that creates Glidein containers can act as a Compute Entrypoint



The Glidein Manager: Kubernetes Operator CE Edition

- Kubernetes Operator that acts as a Compute Entrypoint for a cluster,
- Translates versioned config data from a Glidein Manager into Glidein-running resources in its cluster
 - launching Glideins via autoscaling deployment
- Updates resources within its cluster when new Git commits are pushed to the Glidein Manager
- Lets cluster owners allocate resources to research organizations, and research organizations configure how those resources are used



Thank you!

<https://htcondor.org>

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