

# The Art of Running HTCondor as a batch system

# (10 HTCondor Features You Should Know)

Brian Bockelman  
HEPiX Spring 2014

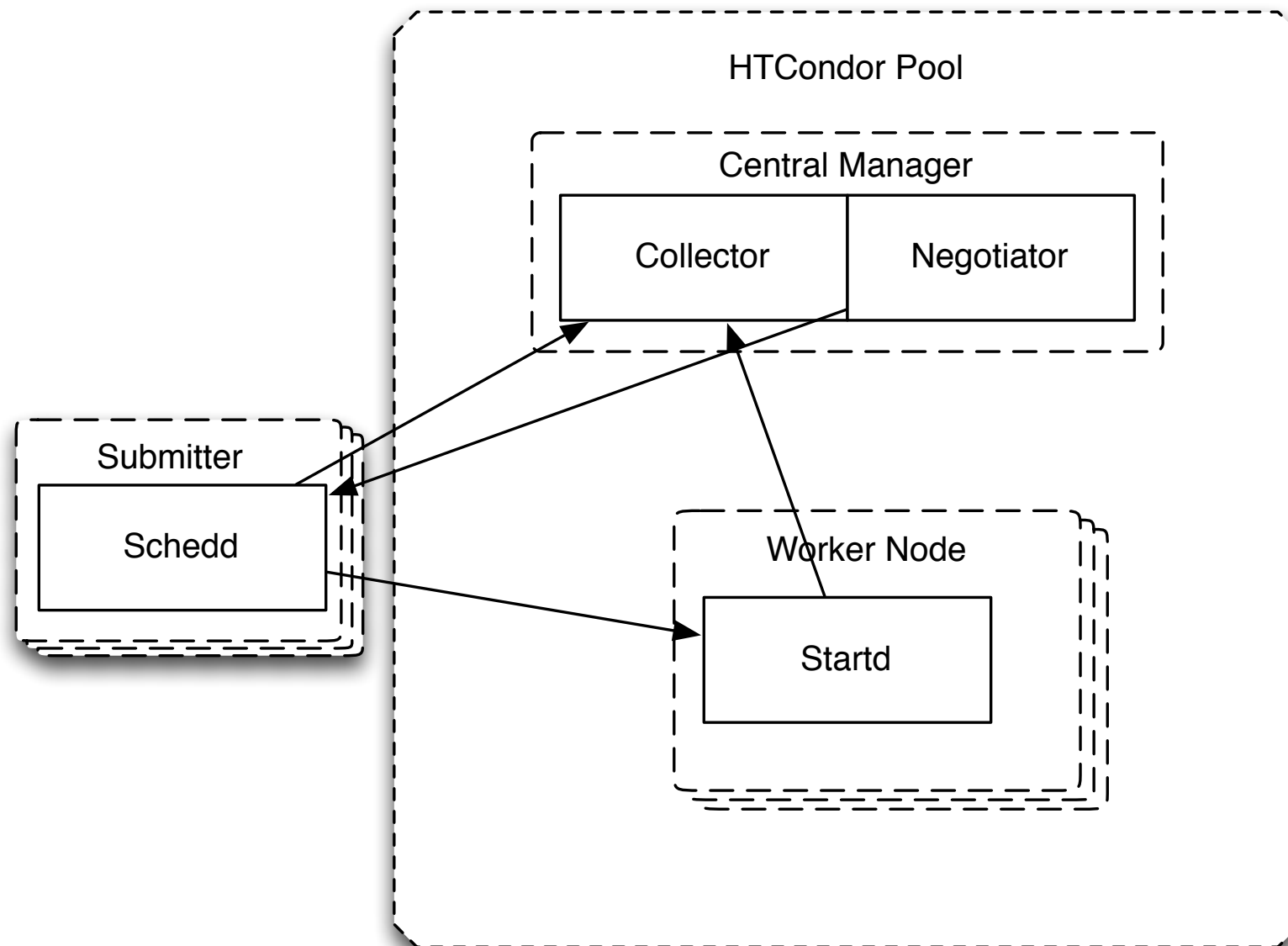


# Swiss Army HTCondor

- HTCondor really is a *platform* for high-throughput computing. The number of ways it can be used is quite immense!
- However, the bread-and-butter is still often running it as a site's batch system.
- This talk covers 10 features every sysadmin should be aware of if they run HTCondor as a batch system!
  - I start with the assumption that everyone here knows the basics of running a HTCondor system :)



# If You Forgot...



# 10 Live Monitoring

- `condor_ssh_to_job` allows the user (or superuser!) to SSH directly to the job's runtime environment.

- Great for hunting down and debugging jobs.

- Does your site's security policy disallow interactive access on the WN? In that case, try `condor_tail`; this allows you to tail any file in the job's sandbox.

```
condor_startd.V6 — root@red:~ — ssh — 91x25
3187080.0  fermilab      5/19 02:09  0+00:13:05 slot1@red-d8n17.unl.edu
3187081.0  fermilab      5/19 02:09  0+00:12:38 slot1@red-d16n14.unl.edu
3187082.0  glow         5/19 02:09  0+00:05:04 slot1@red-d18n22.unl.edu
3187083.0  glow         5/19 02:09  0+00:00:02 slot1@red-d18n25.unl.edu
3187084.0  uscmsPool2555 5/19 02:13  0+00:08:05 slot1@red-d9n4.unl.edu
3187086.0  fermilab      5/19 02:17  0+00:04:30 slot1@red-d22n1.unl.edu
3187087.0  fermilab      5/19 02:17  0+00:00:47 slot1@red-d8n10.unl.edu
3187090.0  uscmsPool2559 5/19 02:21  0+00:00:45 slot1@red-d23n14.unl.edu
3187091.0  uscmsPool2558 5/19 02:21  0+00:00:45 slot1@red-d21n15.unl.edu
[root@red ~]# condor_ssh_to_job 3187090.0
Could not chdir to home directory /grid_home/uscmsPool2559: No such file or directory
Welcome to slot1@red-d23n14.unl.edu!
Your condor job is running with pid(s) 13344.
bash-4.1$ ps fux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
33559    26523  0.0  0.0  22624  1872 ?        SNs   May17    0:00 /bin/bash /var/lib/condor
33559    32588  0.0  0.0  22352  1580 ?        SN    May17    0:00 \ /bin/bash /var/lib/condor
33559     1037  0.0  0.0 102164  7964 ?        SN    May17    0:02 \ /var/lib/condor
33559     1041  0.3  0.0  26960  8996 ?        SN    May17    6:54 \ condor_pro
33559     1042  0.0  0.0 102800  8752 ?        SN    May17    1:14 \ condor_star
33559    20022  0.0  0.0 102320  8412 ?        SN    May18    0:00 \ condor_
33559    25665  0.0  0.0  22624  1872 ?        SNs   May17    0:00 /bin/bash /var/lib/condor
33559    31226  0.0  0.0  22360  1576 ?        SN    May17    0:00 \ /bin/bash /var/lib/condor
33559    32427  0.0  0.0 102016  5648 ?        SN    May17    0:01 \ /var/lib/condor
33559    32553  0.3  0.0  26944  8992 ?        SN    May17    6:52 \ condor_pro
```

# 9 Scalable Job Updates

- Every N minutes (N defaults to 5), an update is pushed out of some standard statistics (disk / CPU / memory used) from the worker node to the scheduler.
- Users can now invoke the `condor_chirp` utility to push custom attributes.
- The implementation is done in a scalable matter so users aren't able to overwhelm the system.
- Purpose is to allow for custom-purpose user monitoring.
- Initial use case is to allow CMS jobs to advertise how many events have been processed.

# 8 Flexible Accounting

- HTCondor doesn't provide a native accounting database. However, it provides a number of accounting files.
- Job history for a schedd is viewable with the `condor_history` command.
- For integrating with the site's accounting database, HTCondor can write out a single file per job in the `PER_JOB_HISTORY_DIR` on the schedd.
  - Setting `PER_JOB_HISTORY_DIR` on the worker node leaves a file per *job execution*.
  - Both directories are accessible via `condor_fetchlog` and the python bindings.
  - Having a file-per-job helps the accounting DB integration to know which jobs have already been processed.

# 7 Security Friendly

- HTCondor has a lot of flexibility for both authz and authn.
- Authentication methods: FS, CLAIMTOBE (unauthenticated), GSI, KRB5, IP-based.
  - Strong authentication methods (GSI) can be combined with IP / hostname restrictions.
  - Authentication results in a HTCondor username (such as bbockelm@unl.edu). GSI can callout to external libraries (LCMAPS) for final mapping.
- Once authenticated, site can have various policies for what the user is authorized to do. Example:
  - SCHEDD.ALLOW\_WRITE = \*@unl.edu, \$(HOSTNAME)@daemon.unl.edu



# Maybe not so friendly?

(to be fair, Nebraska is the most complex example possible)

```
condor_startd.V6 — root@red-man:/etc/puppet/environments/production/oldmodules/condor/files/c...
# Authorization settings
# These should be unnecessary, unless if we have an error below.
DENY_WRITE          = anonymous@*, unmapped@*
DENY_NEGOTIATOR     = anonymous@*, unmapped@*
DENY_ADMINISTRATOR = anonymous@*, unmapped@*
DENY_DAEMON         = anonymous@*, unmapped@*

# Defaults for HCC
FRIENDLY_DAEMONS = *@daemon.unl.edu
WORKER_NODES     = *@worker.unl.edu/172.16.1.*, *@worker.unl.edu/172.16.3.*, *@worker.unl.edu/172.16.*, *@worker.unl.edu/*
USERS            = *@unl.edu

# Authz settings for each daemon. Preferably, change the templates above
DEFAULT_WRITE = $(FRIENDLY_DAEMONS), $(HOSTNAME)@worker.unl.edu/$(FULL_HOSTNAME)
ALLOW_WRITE = $(DEFAULT_WRITE)

# Schedd is the only one accepting non-strong auth
SCHEDD.ALLOW_WRITE          = $(USERS), $(HOSTNAME)@daemon.unl.edu/$(FULL_HOSTNAME)
NEGOTIATOR.ALLOW_WRITE      = $(FRIENDLY_DAEMONS)
COLLECTOR.ALLOW_ADVERTISE_MASTER = $(FRIENDLY_DAEMONS), $(WORKER_NODES), condor@unl.edu
COLLECTOR.ALLOW_ADVERTISE_SCHEDD = $(FRIENDLY_DAEMONS)
COLLECTOR.ALLOW_ADVERTISE_STARTD = $(WORKER_NODES), $(HOSTNAME)@daemon.unl.edu/$(FULL_HOSTNAME)
STARTD.ALLOW_NEGOTIATOR     = red-condor@daemon.unl.edu/red-condor.unl.edu, $(HOSTNAME)@daemon.unl.edu/$(FULL_HOSTNAME)
SHADOW.ALLOW_WRITE         = $(DEFAULT_WRITE), $(WORKER_NODES), $(HOSTNAME)@daemon.unl.edu/$(FULL_HOSTNAME)
ALLOW_DAEMON              = $(FRIENDLY_DAEMONS), condor@unl.edu, submit-side@matchsession, $(HOSTNAME)@worker.unl.edu/$(FULL_HOSTNAME)
ALLOW_ADMINISTRATOR       = red-condor@daemon.unl.edu/red-condor.unl.edu, red-man@unl.edu/red-man.unl.edu, red-man@daemon.unl.edu/red-man.unl.edu, red-man@daemon.unl.edu/172.16.200.1, $(HOSTNAME)@daemon.unl.edu/$(FULL_HOSTNAME), $(HOSTNAME)@worker.unl.edu/$(FULL_HOSTNAME)

# Authentication settings
SEC_DEFAULT_AUTHENTICATION = REQUIRED
SEC_READ_AUTHENTICATION    = OPTIONAL
SEC_CLIENT_AUTHENTICATION  = OPTIONAL
SEC_DEFAULT_AUTHENTICATION_METHODS = GSI
SCHEDD.SEC_WRITE_AUTHENTICATION_METHODS = FS,GSI
SCHEDD.SEC_DAEMON_AUTHENTICATION_METHODS = FS,GSI
SEC_CLIENT_AUTHENTICATION_METHODS = FS,GSI
```

# 6 Firewall Friendly

- A HTCondor cluster involves many daemons - all of which must communicate with each other over TCP.
  - All but collector default to a randomly-selected port, which traditionally made the firewall configuration a big headache.
- If you enable the `condor_shared_port` daemon, it will, using socket passing, aggregate all communication through a single TCP port (9618).
  - Greatly simplifies the firewall configuration!
- In the 8.3.0 / 8.3.1 timeframe, we hope to make `shared_port` enabled by default.

# 5 Customizable output formats

- Hate the condor\_status or condor\_q output formats?
- Starting in 8.1.6, the sysadmin can customize the default output formats.
- Provide a format file as specified at <https://htcondor-wiki.cs.wisc.edu/index.cgi/wiki?p=ExperimentalCustomPrintFormats>; uses a SQL-like syntax.
- Alternately, user can specify their own file.

# condor\_status - default

```
condor_startd.V6 — bbockelm@hcc-briantest:~ — ssh — 118x41

[bbockelm@hcc-briantest ~]$ condor_status
Name                               OpSys      Arch      State      Activity LoadAv Mem      ActvtyTime
slot10@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:15
slot11@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:16
slot12@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:17
slot13@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:18
slot14@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:19
slot15@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:20
slot16@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:13
slot17@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:14
slot18@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:15
slot19@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:16
slot1@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+11:37:47
slot20@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:17
slot21@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:18
slot22@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:19
slot23@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:20
slot24@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:13
slot25@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:14
slot26@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:15
slot27@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:16
slot28@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:17
slot29@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:18
slot2@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.730  262    2+15:43:15
slot30@hcc-briante                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:20
slot3@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:16
slot4@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:17
slot5@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:18
slot6@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:19
slot7@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:20
slot8@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:13
slot9@hcc-briantes                 LINUX      X86_64    Unclaimed  Idle      0.000  262    2+15:43:14
Total Owner Claimed Unclaimed Matched Preempting Backfill
X86_64/LINUX      30      0      0      30      0      0      0
Total      30      0      0      30      0      0      0
[bbockelm@hcc-briantest ~]$
```



# condor\_status - custom

```
condor_startd.V6 — bbockelm@hcc-briantest:~ — ssh — 118x41

[bbockelm@hcc-briantest ~]$ condor_status -pr /tmp/testy2.cpf
Machine             Slot      Platform  CPU Mem  Status      StatusTime JobId  J/Min
hcc-briantest.unl.edu slot10    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:15 no   0.00
hcc-briantest.unl.edu slot11    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:16 no   0.00
hcc-briantest.unl.edu slot12    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:17 no   0.00
hcc-briantest.unl.edu slot13    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:18 no   0.00
hcc-briantest.unl.edu slot14    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:19 no   0.00
hcc-briantest.unl.edu slot15    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:20 no   0.00
hcc-briantest.unl.edu slot16    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:13 no   0.00
hcc-briantest.unl.edu slot17    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:14 no   0.00
hcc-briantest.unl.edu slot18    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:15 no   0.00
hcc-briantest.unl.edu slot19    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:16 no   0.00
hcc-briantest.unl.edu slot1     X86_64_SL6  1  262 Unclaimed/Idle 2+11:42:47 no   0.00
hcc-briantest.unl.edu slot20    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:17 no   0.00
hcc-briantest.unl.edu slot21    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:18 no   0.00
hcc-briantest.unl.edu slot22    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:19 no   0.00
hcc-briantest.unl.edu slot23    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:20 no   0.00
hcc-briantest.unl.edu slot24    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:13 no   0.00
hcc-briantest.unl.edu slot25    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:14 no   0.00
hcc-briantest.unl.edu slot26    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:15 no   0.00
hcc-briantest.unl.edu slot27    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:16 no   0.00
hcc-briantest.unl.edu slot28    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:17 no   0.00
hcc-briantest.unl.edu slot29    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:18 no   0.00
hcc-briantest.unl.edu slot2     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:15 no   0.00
hcc-briantest.unl.edu slot30    X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:19 no   0.00
hcc-briantest.unl.edu slot3     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:16 no   0.00
hcc-briantest.unl.edu slot4     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:17 no   0.00
hcc-briantest.unl.edu slot5     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:18 no   0.00
hcc-briantest.unl.edu slot6     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:19 no   0.00
hcc-briantest.unl.edu slot7     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:20 no   0.00
hcc-briantest.unl.edu slot8     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:13 no   0.00
hcc-briantest.unl.edu slot9     X86_64_SL6  1  262 Unclaimed/Idle 2+15:48:14 no   0.00
Total Owner Claimed Unclaimed Matched Preempting Backfill
X86_64/LINUX 30 0 0 30 0 0 0
Total 30 0 0 30 0 0 0
[bbockelm@hcc-briantest ~]$
```

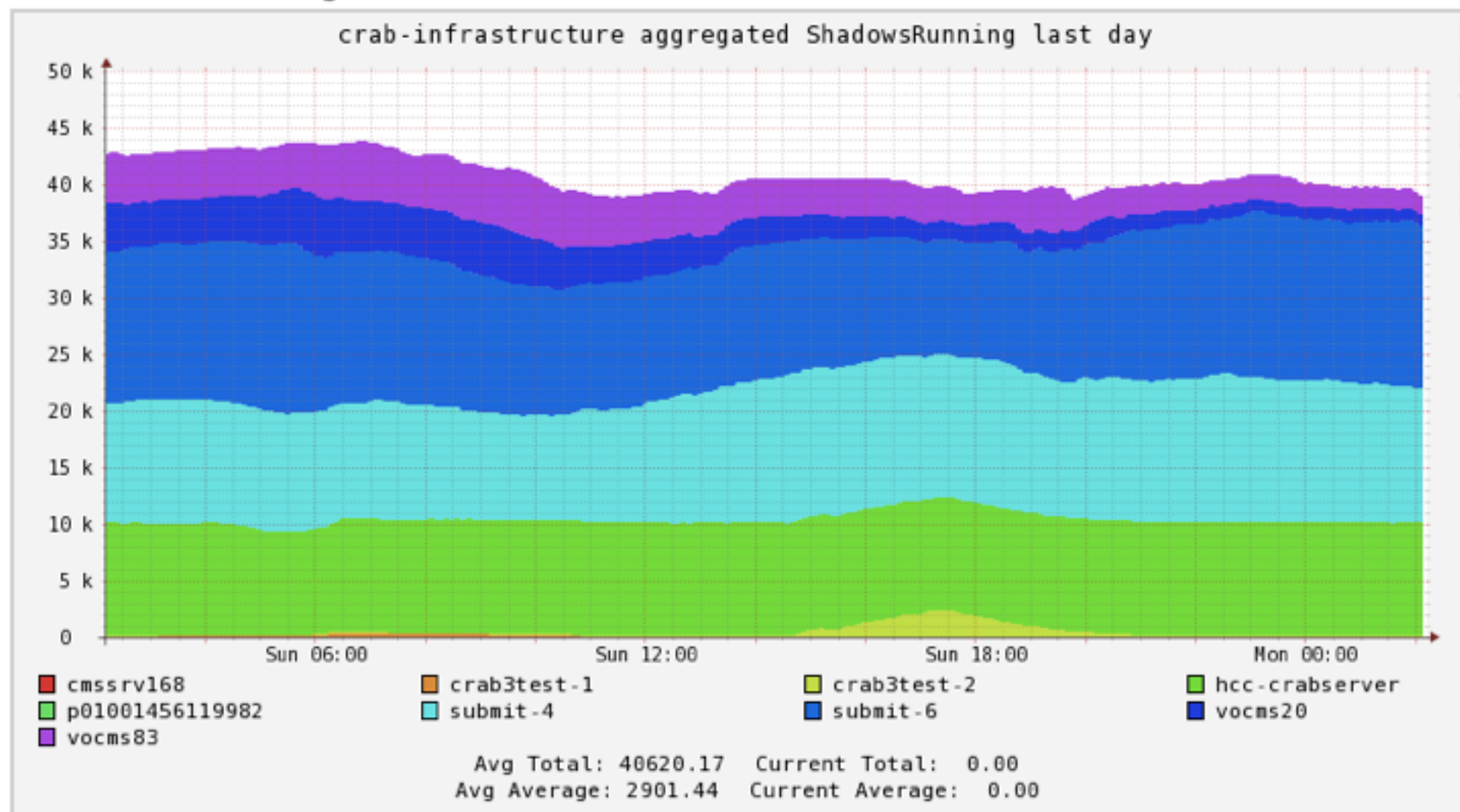
# 4 Hooks

- HTCondor can invoke sysadmin-provide scripts at various points of the job's lifecycle - pre/post and periodically while the job runs.
  - This can be used to provide a custom monitoring script or setup/cleanup the job environment.
- The `STARTD_CRON` mechanism causes the startd to periodically execute a script and publish the results in the machine's ClassAd; great for integrating health monitoring into HTCondor.
- The `BENCHMARKS` mechanism runs a script at node startup; useful for normalizing CPU power for accounting purposes.

# 3 Ganglia Integration

- Starting in 8.1.0, HTCondor ships with the `condor_gangliad`; this daemon polls the collector for various statistics and pushes them into Ganglia.
- We ship with a set of sane default metrics; the admin can customize any metric through the configuration file.
  - Ganglia will not beat a hand-written, heavily-tweaked monitoring system; we're hoping this will cover 80% of the need though!
- Don't use Ganglia? If you provide a script that is command-line compatible with `gmetric`, you can push these to any arbitrary monitoring system.

In	Now: 6.9M	Min: 2.6M	Avg: 7.7M	Max: 18.0M
Out	Now: 68.0M	Min: 31.2M	Avg: 87.8M	Max: 290.7M

crab-infrastructure **ShadowsRunning** last day sorted by name

Metric

ShadowsRunning

Show Hosts Scaled:

Auto

Same

None

Size

small

## Columns


4

(0 = metric + reports)

Show only nodes matching

File

May graphs to show



Conto

secondly

deconding

### References

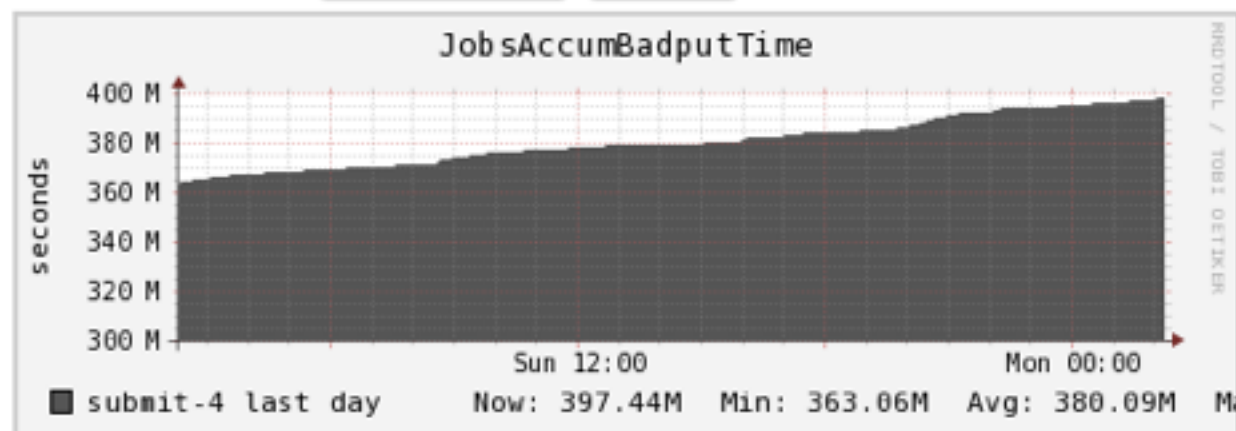


JobsAccumBadputTime

CSV JSON Inspect Trend

Hide/Show Events

Timeshift

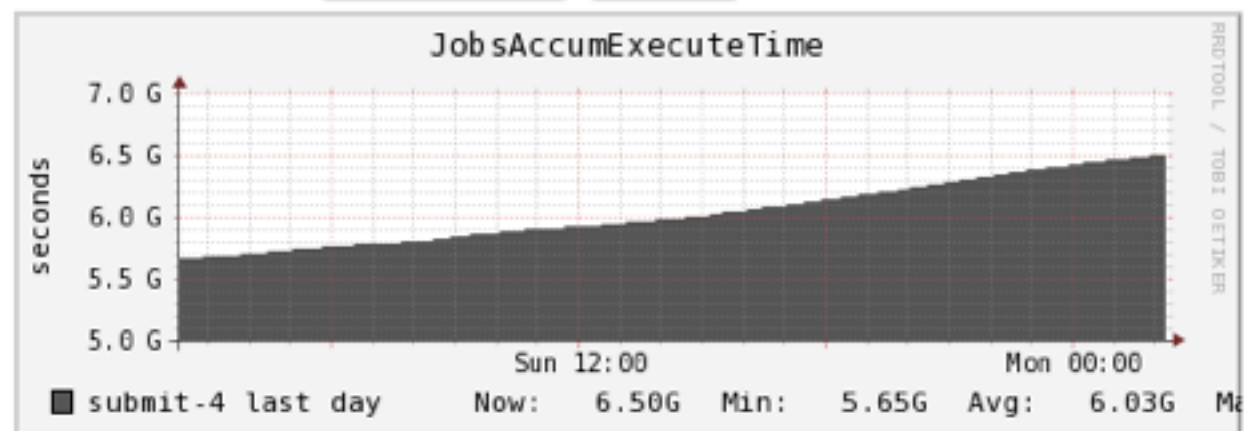


JobsAccumExecuteTime

CSV JSON Inspect Trend

Hide/Show Events

Timeshift

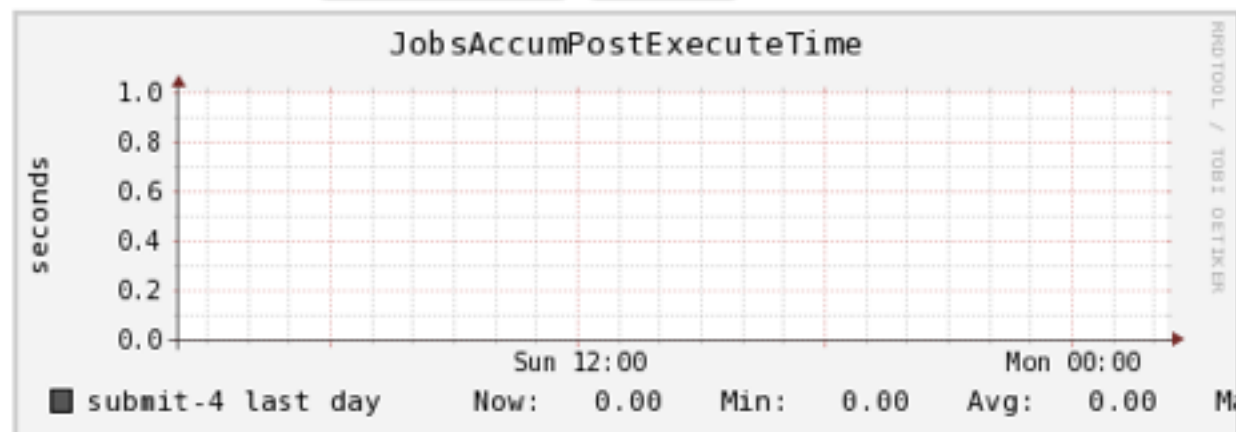


JobsAccumPostExecuteTime

CSV JSON Inspect Trend

Hide/Show Events

Timeshift

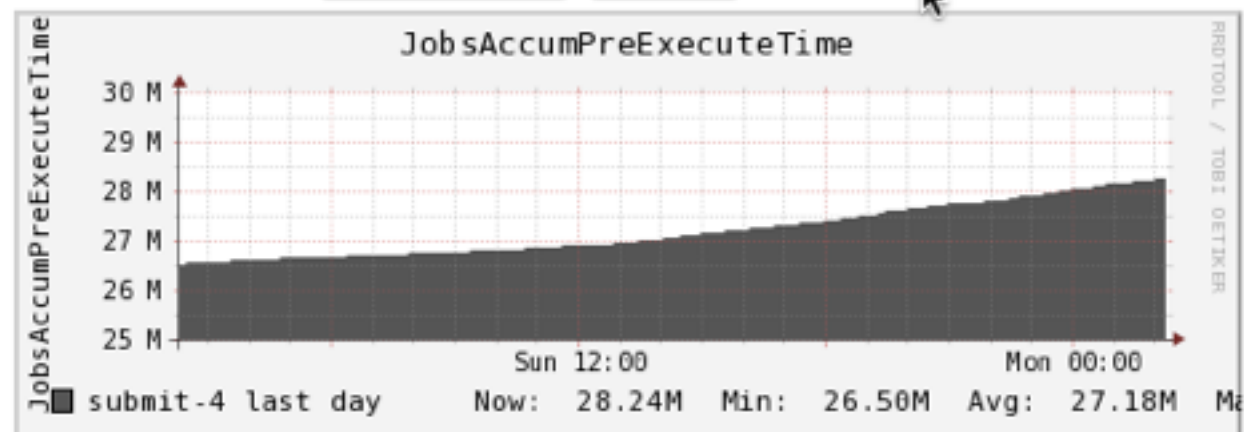


JobsAccumPreExecuteTime

CSV JSON Inspect Trend

Hide/Show Events

Timeshift

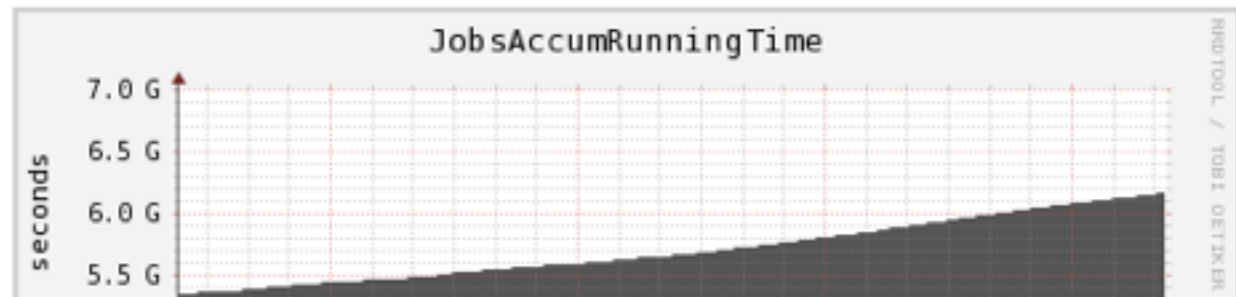


JobsAccumRunningTime

CSV JSON Inspect Trend

Hide/Show Events

Timeshift

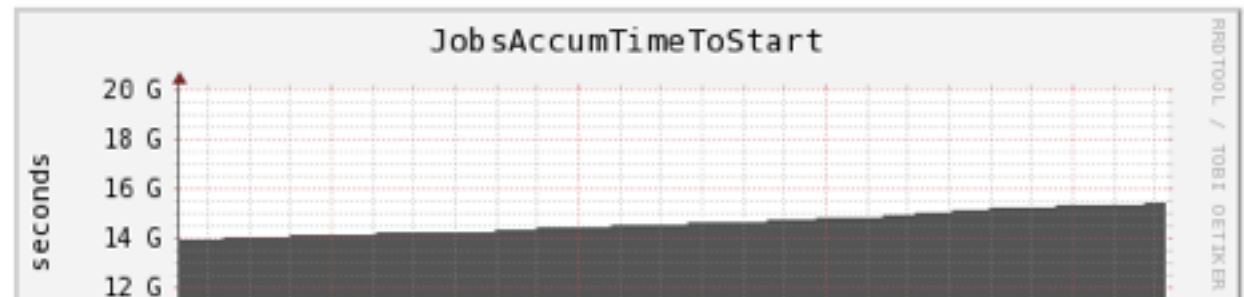


JobsAccumTimeToStart

CSV JSON Inspect Trend

Hide/Show Events

Timeshift



# 2 Python Bindings

- Basically all client functionality is accessible through a python module.
- Module invokes the appropriate C++ code directly; no fork/exec of client tools.
- Goal is to be “pythonic”: failures are turned into python exceptions, ClassAd types are converted to their python equivalent types where possible.

```
Last login: Mon Apr 29 14:25:48 on ttys004
Brians-MacBook-Air:~ bbockelm$ grid-proxy-
Brians-MacBook-Air:~ bbockelm$ python
Python 2.7.2 (default, Jun 20 2012, 16:23:33)
[GCC 4.2.1 Compatible Apple Clang 4.0 (tags/Applet/clang-418.0.60)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import htcondor
>>> coll = htcondor.Collector("red-condor.unl.edu")
>>> schedd_ad = coll.locate(htcondor.DaemonTypes.Schedd, "red.unl.edu")
>>> schedd = htcondor.Schedd(schedd_ad)
>>> jobs = schedd.query()
>>> print jobs[0],
```

```
[
  CurrentTime = time();
  BufferSize = 524288;
  JobNotification = 0;
  BufferBlockSize = 32768;
  Err = "/var/lib/globus/job_home/uscmsPool2295/.globus/job/red/16290030
2317236126.1905433861141216178/stderr";
  CumulativeSlotTime = 0;
  CoreSize = -1;
  NiceUser = false;
  x509UserProxyExpiration = 1367424183;
```

```
[demo@ip-10-62-61-234 ~]$ python
Python 2.6.6 (r266:84292, Dec 7 2011, 20:48:22)
[GCC 4.4.6 20110731 (Red Hat 4.4.6-3)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import htcondor
>>> import classad
>>> schedd = htcondor.Schedd()
>>> ad_results = []
>>> cluster = schedd.submit(classad.ClassAd({"Cmd": "/bin/sh", "Arguments": "-c
'echo Hello world && sleep 1m'"}), 1, True, ad_results)
>>> cluster
5
>>> ad_results[0]
[ BufferSize = 524288; NiceUser = false; CoreSize = -1; CumulativeSlotTime = 0;
OnExitHold = false; RequestCpus = 1; Err = "/dev/null"; BufferBlockSize = 32768;
  ImageSize = 100; CurrentTime = time(); WantCheckpoint = false; CommittedTime =
0; TargetType = "Machine"; WhenToTransferOutput = "ON_EXIT"; Cmd = "/bin/sh"; Jo
bUniverse = 5; ExitBySignal = false; HoldReasonCode = 16; Iwd = "/home/demo"; Nu
mRestarts = 0; CommittedSuspensionTime = 0; Owner = undefined; NumSystemHolds =
0; CumulativeSuspensionTime = 0; RequestDisk = DiskUsage; Requirements = true &&
  TARGET.OPSYS == "LINUX" && TARGET.ARCH == "X86_64" && TARGET.HasFileTransfer &&
  TARGET.Disk >= RequestDisk && TARGET.Memory >= RequestMemory; MinHosts = 1; Job
Notification = 0; NumCkpts = 0; LastSuspensionTime = 0; NumJobStarts = 0; WantRe
moteSyscalls = false; JobPrio = 0; RootDir = "/"; CurrentHosts = 0; StreamOut =
```

# 1 Containers

- Container virtualization without us
- Over the past three years, we have been adding various container-based features:
  - **cgroups**: HTCondor creates a unique cgroup for each job. We use:
    - *freezer*: Assists in killing all the processes in the job.
    - *cpuacct*: Calculate total CPU used by job.
    - *cpu*: Fairshare CPU usage between jobs.
    - *memory*: Job memory accounting and limiting.
    - *blkio*: Accounting of block IO caused by jobs (not very good in SL6).

# 1 Containers

- **namespaces:** Jobs are spawned with:
  - *PID namespaces:* Only processes from the running job are visible; nothing from the system or other jobs.
  - *FS namespaces:* Certain system directories (/tmp, /var/tmp) can be overwritten with the HTCondor scratch directory; only visible to the job.
  - *Network namespaces:* (Not yet part of base HTCondor) Each job is allocated its own IP address and network card, which is bridged to the host network for the duration of the job.
- **chroot:** Sysadmin can setup multiple chroots (SL5, SL6) and allow the job to choose one.
  - With combination of chroot and FS namespaces, there are no persistent directories the job can write into and no writable directories visible to other jobs.
- In the end, jobs get the system resources they requested *and* are heavily isolated from others.
- I'm glad the rest of the Linux community has found containers!

# The Artist Commune

- If there's an art to running HTCondor, then surely there must be an artist commune somewhere!
- You can find like-minded people at:
  - #distcomp on IRC
  - [htcondor-users@cs.wisc.edu](mailto:htcondor-users@cs.wisc.edu) mail list
  - Annually at HTCondor Week in Madison, WI.