HTCondor Recent Enhancement and Future Directions

HEPiX Fall 2015

Todd Tannenbaum
Center for High Throughput Computing
Department of Computer Sciences
University of Wisconsin-Madison

CENTER FOR HIGH THROUGHPUT COMPUTING

HTCondor

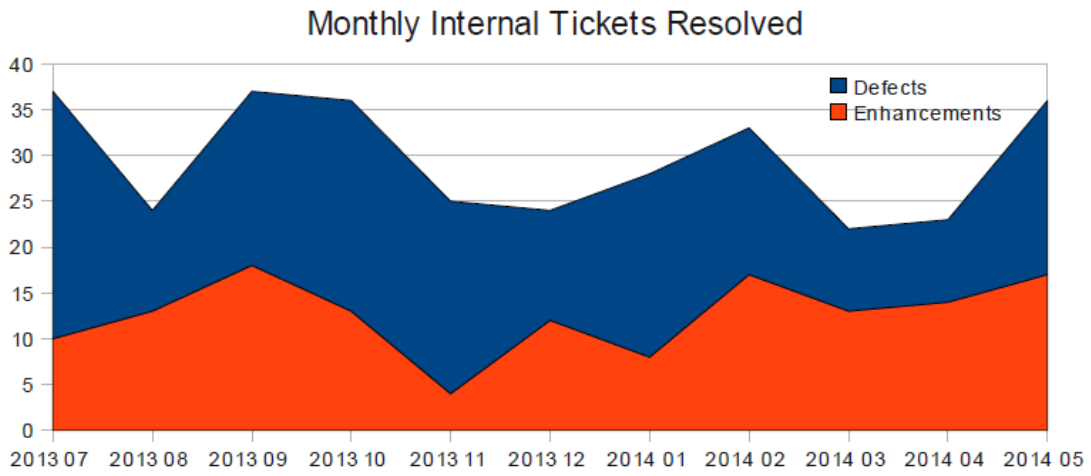
University of Wisconsin Center for High Throughput Computing

HTCondor

- Open source distributed high throughput computing
- Management of resources, jobs, and workflows
- Primary objective: assist the scientific community with their high throughput computing needs
- Mature technology…

Mature… but actively developed

- Last year : 17 releases, 2337 commits by 22 contributors
- Open source development model
- Evolve to meet the needs of the science community in a ever-changing computing landscape



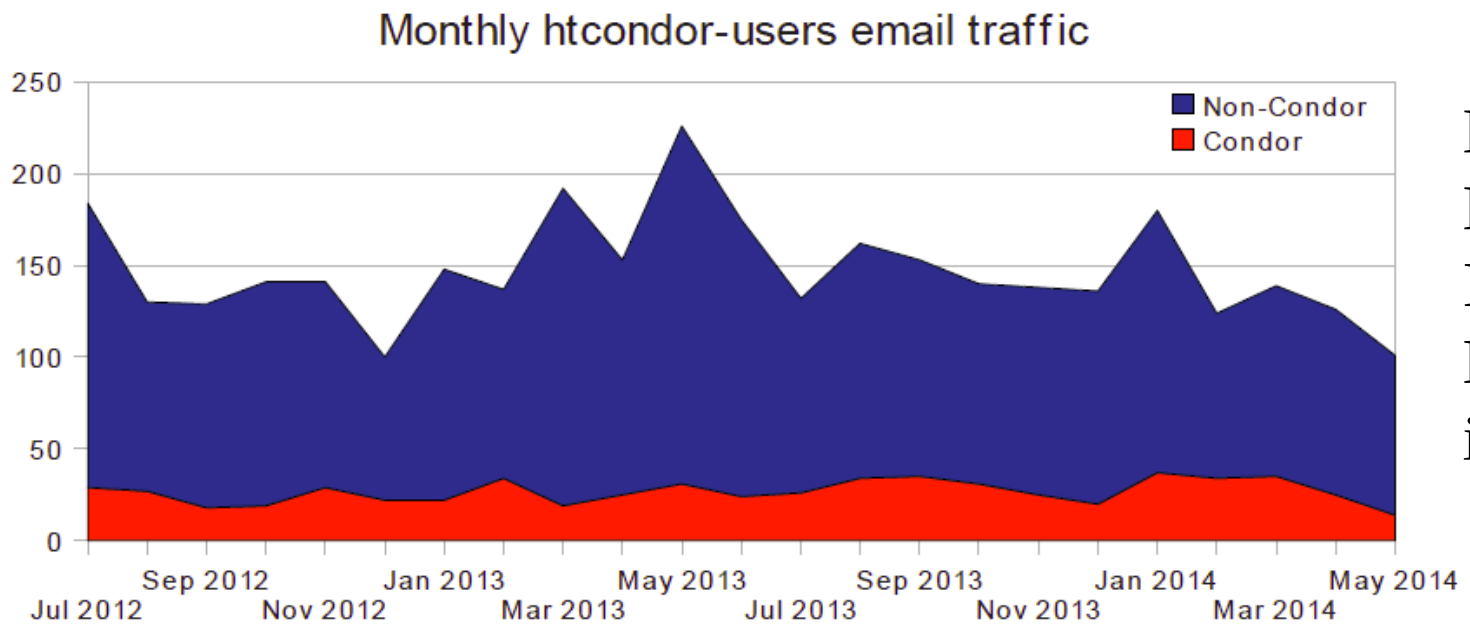
Why am I here?

- Desire to work together with the HEP community to leverage our collective experience / effort / know-how to offer an open source solution that meets the growing need of HEP high throughput computing in a challenging budget environment

Current Channels

- Documentation
- Community support email list (htcondor-users)
- Ticket-tracked developer support

- *Bi-weekly/monthly phone conferences*
  - Identify and track current problems
  - Communicate and plan future goals
  - Identify and collaborate on challenges, f2f
- Fully open development model
- Commercial options for 24/7



Meet w/ CMS, LIGO, IceCube, LSST, FNAL, iPlant, …

HTCondor Week

- Annually each May in Madison, WI
- May 17-20 2016

EU HTCondor+ARC Workshop

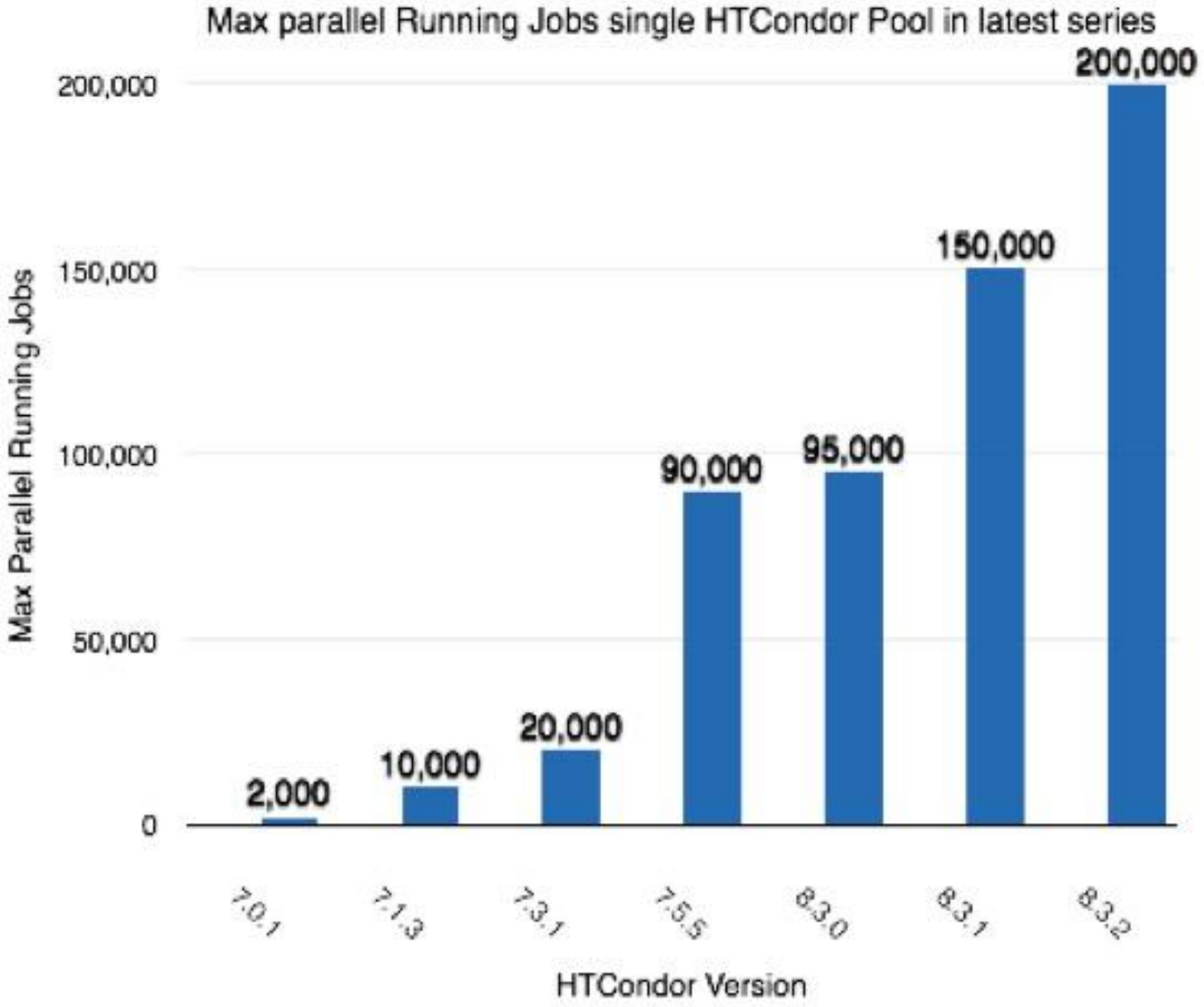
- When: Week of Feb 29, 2016
- Where: Barcelona!! (synchrotron radiation facility)
- HTCondor
  - Tutorials and community presentations
    - Monday PM – Wednesday
  - Office hours
    - Thursday - Friday AM
- ARC CE
  - Tutorials and community presentations
    - Thursday
  - Office hours
    - Weds and Friday AM

HTCondor v8.2 Enhancements

- EC2 Grid Job Improvements
- Better support for OpenStack
- Google Compute Engine Jobs
- HTCond[illegible] BOINC
- Scalabil[illegible]
- GPU Su[illegible]
- New Co[illegible] File Constru[illegible]ncluding includes, conditionals, meta-knobs
- Asynchronous Stage-out of Job Output
- Ga[illegible]ia M[illegible]g via co[illegible]
- Co[illegible]
- [illegible] transfer [illegible]duling via disk I/O Load
- Daily pool job run statistics via condor_job_report
- Monitoring via BigPanDAmon

Some HTCondor v8.4 Enhancements

- Encrypted Job Execute Directory
- ENABLE_KERNEL_TUNING = True
- SUBMIT_REQUIREMENT rules
- New packaging
- Scalability and stability
  - Goal: 200k slots in one pool, 10 schedds managing 400k jobs
- Tool improvements, esp condor_submit
- IPv6 mixed mode
- Docker Job Universe


Max parallel Running Jobs single HTCondor Pool in latest series
Max Parallel Running Jobs
200,000
150,000
100,000
50,000
0
2,000
10,000
20,000
90,000
95,000
150,000
200,000
7.0.1
7.1.3
7.3.1
7.5.5
8.3.0
8.3.1
8.3.2
HTCondor Version

Tool improvements
Example: condor_submit

- Could always do numeric parameter sweeps. Now can submit a job for each
  - File or subdirectory
  - Line in a file

  More…

Simple Submit file:

```
Executable = foo.exe
Universe = vanilla
Input = data.in
Output = data.out
Queue
```

Submit a job per file:

```
Executable = foo.exe
Universe = vanilla
Input = $(Item).in
Output = $(Item).out
Queue Item matching (*.in, *.input)
```

Will process all files matching pattern *.in and *.input

Submit a job per line in a file:

```
Executable = foo.exe
Universe = vanilla
Arguments = -gene $(Genome)
Output = $(Genome).out
Queue Genome from GeneList.txt
```

IPv6 Support

- New in 8.4 is support for “mixed mode,” using IPv4 and IPv6 simultaneously.
- A mixed-mode pool’s central manager and submit (schedd) nodes must each be reachable on both IPv4 and IPv6.
- Execute nodes and (other) tool-hosting machines may be IPv4, IPv6, or both.
- ENABLE_IPV4 = TRUE
  ENABLE_IPV6 = TRUE

Containers in HTCondor

- HTCondor can currently leverage Linux containers / cgroups to run jobs
  - Limiting/monitoring CPU core usage
  - Limiting/monitoring physical RAM usage
  - Tracking all subprocesses
  - Private file namespace (each job can have its own /tmp!)
  - Private PID namespace
  - Chroot jail
  - Private network namespace (coming soon! each job can have its own network address)

More containers...
HTCondor Docker Jobs

(Docker Universe)

Installation of docker universe

Need HTcondor 8.4+

Need docker (maybe from EPEL)

```
$ yum install docker-io
```

Docker is moving fast: docker 1.6+, ideally

odd bugs with older dockers!

Condor needs to be in the docker group!

```
$ useradd -G docker condor
$ service docker start
```

HTCondor detects docker

```
$ condor_status -l | grep -i docker
HasDocker = true
DockerVersion = "Docker version
1.5.0, build a8a31ef/1.5.0“
```

Docker jobs will only be scheduled where Docker is installed and operational.

Check StarterLog for error messages if needed

Submit a docker job

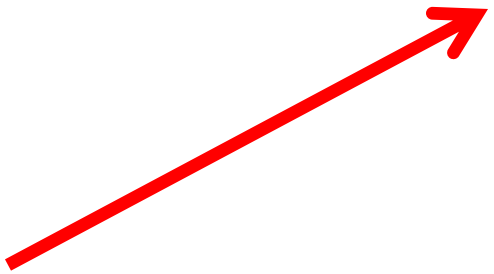
```
universe = docker
executable = /bin/my_executable
arguments = arg1
docker_image = deb7_and_HEP_stack
transfer_input_files = some_input
output = out
error = err
log = log
queue
```

Docker Universe Job Is still a job

- Docker containers have the job-nature
  - condor_submit
  - condor_rm
  - condor_hold
  - Write entries to the job event log
  - condor_dagman works with them
  - Policy expressions work.
  - Matchmaking works
  - User prio / job prio / group quotas all work
  - Stdin, stdout, stderr work
  - Etc. etc. etc.*

Docker Universe

```
universe = docker
executable = /bin/my_executable
```



Executable comes either from submit machine or image. (or a volume mount)

Docker Universe

```
universe = docker
# executable = /bin/my_executable
```

Executable can even be omitted!

trivia: true for what other universe?

(Images can name a default command)

Docker Universe

```
universe = docker
executable = ./my_executable
input_files = my_executable
```

If executable is transferred,
Executable copied from submit machine
(useful for scripts)

Docker Universe

```
universe = docker
executable = /bin/my_executable
docker_image =deb7_and_HEP_stack
```

Image is the name of the docker image stored on execute machine. HTCondor will fetch it if needed, and will remove images off the execute machine with a LRU replacement strategy.

Docker Universe

```
universe = docker
transfer_input_files= some_input
```

HTCondor can transfer input files from submit machine into container

(same with output in reverse)

HTCondor's use of Docker

Condor volume mounts the scratch dir

- File transfer works same
- Any changes to the container are not xfered
- Container is removed on job exit

Condor sets the cwd of job to the scratch dir

Condor runs the job with the usual uid rules

Sets container name to

```
HTCJob_$(CLUSTER)_$(PROC)_slotName
```

Docker Resource limiting

```
RequestCpus = 4
RequestMemory = 1024M
RequestDisk = Somewhat ignored…
```

RequestCpus translated into cgroup shares

RequestMemory enforced

- If exceeded, job gets OOM killed
- job goes on hold

RequestDisk applies to the scratch dir only

10 Gb limit rest of container

Why is my job on hold?

Docker couldn't find image name:

```
$ condor_q -hold

-- Submitter: localhost : <127.0.0.1:49411?addrs=127.0.0.1:49411>
: localhost
 ID      OWNER           HELD_SINCE  HOLD_REASON

 286.0   gthain           5/10 10:13 Error from slot1@localhost:
Cannot start container: invalid image name: debain
```

Exceeded memory limit?

Just like vanilla job with cgroups

```
 297.0   gthain           5/19 11:15 Error from slot1@localhost:
Docker job exhausted 128 Mb memory
```

Surprises with Docker Universe

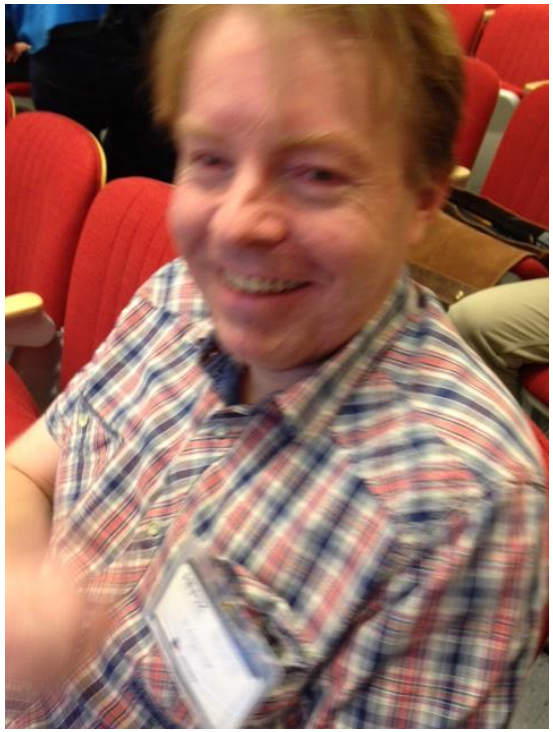
condor_ssh_to_job doesn't work (yet)

condor_chirp doesn't work

Suspend doesn't work

Networking is only NAT

Can't access NFS/shared filesystems in HTCondor v8.4.0 ….



…But admin can specify volume mounts in v8.5.1!

- Admin can add additional volumes
  - That all docker universe jobs get
- Why?
  - CVMFS
  - Large shared data
- Details

https://htcondor-wiki.cs.wisc.edu/index.cgi/tktview?tn=5308

Likely Coming soon…

- Advertise images we already have
- Report resource usage back to job ad
  - E.g. network in and out
- Support for condor_ssh_to_job
- Package and release HTCondor into Docker Hub

Potential Future Features?

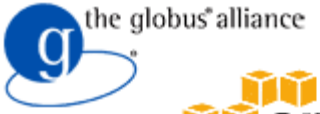
Network support beyond NAT?

Run containers as root?

Automatic checkpoint and restart of containers! (via CRIU)

Grid Universe

- Reliable, durable submission of a job to a remote scheduler
- Popular way to send pilot jobs
- Supports many “back end” types:
  - HTCondor
  - PBS
  - LSF
  - Grid Engine
  - Google Compute Engine
  - Amazon EC2
  - OpenStack
  - Deltacloud
  - Cream
  - NorduGrid ARC
  - BOINC
  - Globus: GT2, GT5
  - UNICORE



Scalable mechanism to grow pool into the Cloud

- Leverage efficient AWS APIs such as Auto Scaling Groups
  - Implement a “lease” so charges cease if lease expires
- Secure mechanism for cloud instances to join the HTCondor pool at home institution

```
condor_annex --set-size 2000
   --lease 24 --project “144PRJ22”
```

Also in the works…

- Kerberos/AFS support (joint effort w/ CERN)
- more scalability, power to the schedd
- shared_port and cgroups on by default
- condor_q and condor_status revamp
- late materialization of jobs in the schedd
- direct interface to slurm in grid universe
- direct interface to openstack in grid universe (via NOVA api)
- data caching
- built-in utilization graphs w/ JSON export

Thank you!