HTCondor and HTCondor-CE News

WLCG Grid Deployment Board January 2021 Meeting

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

CENTER FOR HIGH THROUGHPUT COMPUTING

HTCondor

News!

In fall 2020 we were notified by the **NSF** that our proposal titled "**Partnership for Advanced Throughput Computing (PATh)**" was fully funded for five years with an annual budget of $4.5M

> *"The Partnership to Advance Throughput Computing (PATh) project will expand Distributed High Throughput Computing (dHTC) technologies and methodologies through innovation, translational effort, and large-scale adoption to advance the Science & Engineering goals of the broader community."*

www.nsf.gov/awardsearch/showAward?AWD_ID=2030508

PATh - a powerful partnership

- PATh is a partnership between the UW-Madison Center for High Throughput Computing (**CHTC**) and the Open Science Grid (**OSG**) Consortium
- PATh brings together the *technology* (of HTCondor) and the *services* (of OSG) under one unified project
- *We have funding to implement our roadmap for another 5 years…*
- PATh leadership: Brian Bockelman (Co-PI), Miron Livny (PI), Lauren Michael (Co-PI), Todd Tannenbaum (Co-PI), Frank Wuerthwein (Co-PI)

https://path-cc.io/about/

HTCSS

The transition from the "Condor System" to the "HTCondor Software Suite (HTCSS)"

- Software elements of the **HTCSS** can be used “stand alone” or “mix and match”
- On-the-fly and hot deployment and upgrades of **HTCSS** elements
- **HTCSS** interfaces with “other/external” technologies, execution environments and services
- **HTCSS** leverages functionality of widely adopted tools

HTCondor Release Series

- Stable Series (*bug fixes only*)
  - HTCondor v8.8.x – first introduced Jan 2019

    (Currently at v8.8.12)
- Development Series (*should be 'new features' series)*
  - HTCondor v8.9.x (Currently at v8.9.10)
- Detailed Version History in the Manual
  - https://htcondor.readthedocs.io/en/latest/version-history/

What's new in v8.8 and/or cooking for v8.9 and beyond?

UI and API Enhancements

- Working to simplify installation, e.g.

  `curl -fsSL https://get.htcondor.org | bash`

- Rethinking of the command-line UI
  - condor <noun> <verb>, e.g.

    `condor job status` instead of condor_q

    `condor pool status` instead of condor_status

- Removed Web Service API in v8.9
- Python API and REST API work …

Python

- Bring HTC into Python environments incl interactive environments e.g. Jupyter Notebooks
- HTCondor Bindings (HTCondor's Python API) has become very popular
- **Added new Python APIs**: DAGMan submission, DAG creation (htcondor.dags), credential management (i.e. Kerberos/Tokens)
- Initial integration with **Dask**
- Released our **HTMap package**
  - No HTCondor concepts to learn, just extensions of familiar Python functionality. Inspired by BNL!

htcondor package

```
import htcondor

# Describe jobs
sub = htcondor.Submit('''
    executable = my_program.exe
    output = 'run$(ProcId).out'
    ''')

# Submit jobs
schedd = htcondor.Schedd()
with schedd.transaction() as txn:
  clusterid = sub.queue(txn,count = 10)

# Wait for jobs
import time
while len(schedd.query(
   constraint='ClusterId=='+str(clusterid),
   attr_list=['ProcId'])):
      time.sleep(1)
```

htmap package

```
import htmap

# Describe work
def double(x):
    return 2 * x

# Do work
doubled = htmap.map(double,range(10))

# Use results!
print(list(doubled))
# [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

See https://github.com/htcondor/htmap

REST API

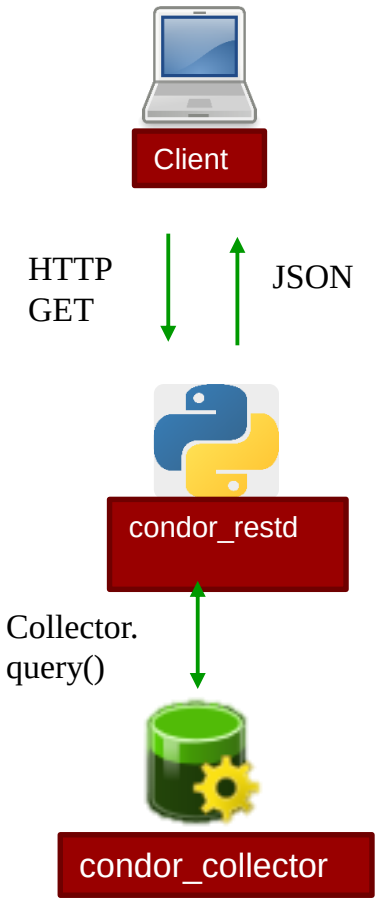
- Python (Flask) webapp for querying HTCondor jobs, machines, and config
- Runs alongside an HTCondor pool
- Listens to HTTP queries, responds with JSON
  - Built ontop of Python API
  - other cool tools coming

    courtesy Python API…

    ….like condor_watch_q !

https://htcondor.readthedocs.io/en/latest/man-pages/condor_watch_q.html

REST API, cont




```
$ curl "http://localhost:9680/v1/status\
?query=startd\
&projection=cpus,memory\
&constraint=memory>1024"
```

```
[
    {
        "name": "slot4@siren.cs.wisc.edu",
        "type": "Machine",
        "classad": {
            "cpus": 1,
            "memory": 1813
        }
    },
…
]
```

REST API, cont

- Swagger/OpenAPI spec to generate bindings for Java, Go, etc.
- Evolving, but see what we've got so far at
  - https://github.com/htcondor/htcondor-restd

- Potential Future improvements
  - Allow changes (job submission/removal, config editing)
  - Add auth
  - Improve scalability
  - Run under shared port

Federation of Compute resources: HTCondor Annexes

HTCondor "Annex"

- Instantiate an HTCondor Annex to dynamically add additional execute slots into your HTCondor environment
- Want to enable end-users to provision an Annex on
  - Clouds
  - HPC Centers / Supercomputers
    - Via edge services (i.e. HTCondor-CE)
  - Kubernetes clusters

A cost effective ExaFLOP hour in the Clouds for IceCube

Published on February 5, 2020

Igor Sfiligoi
Lead Scientific Software Developer and Researcher at San Diego Supercomputer Center

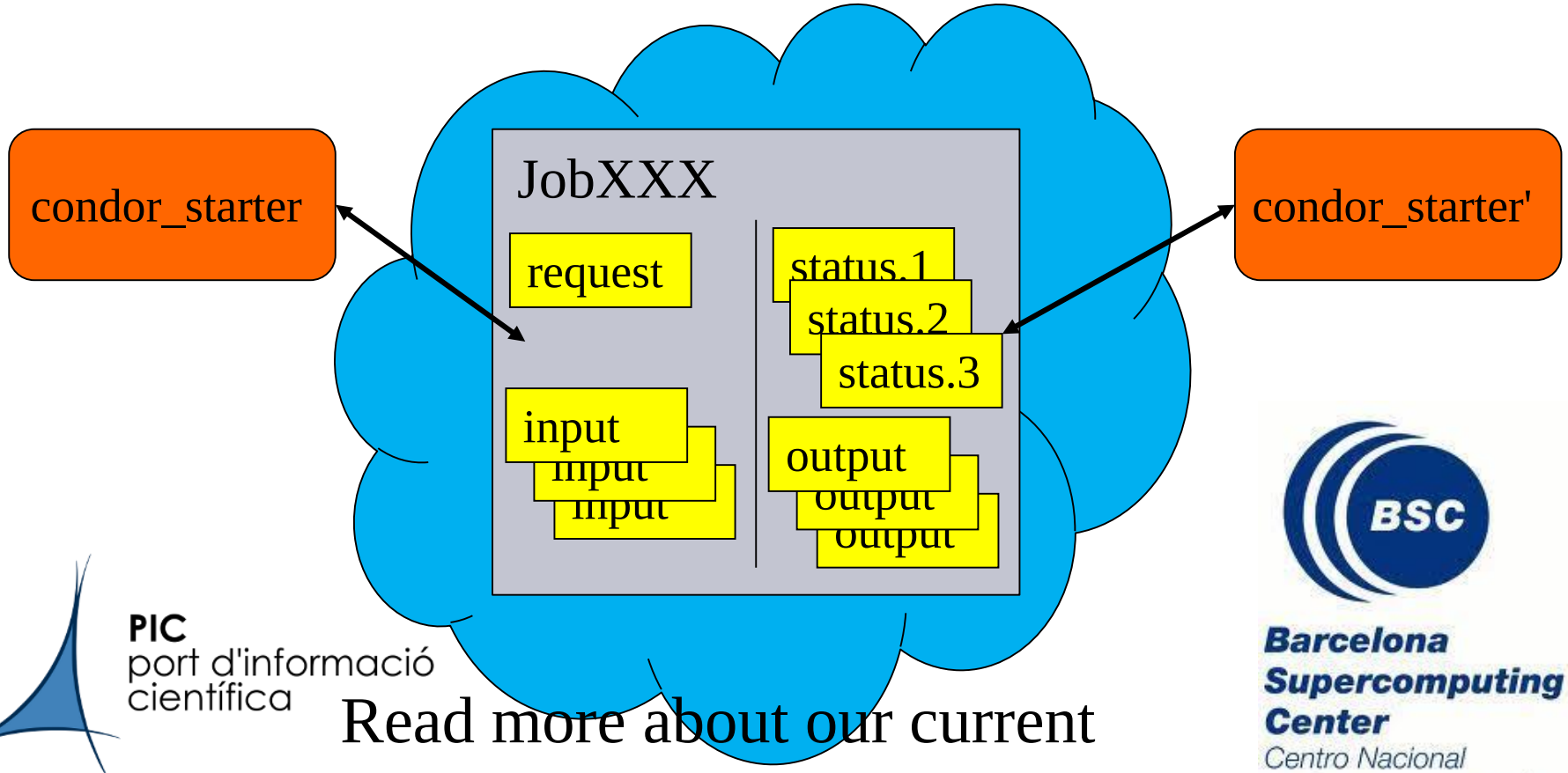
9 articles
✓ Following

We did it again! Another large scale Cloud GPU burst to run IceCube simulations, a dHTC application. If you missed our first run, see my other article.

The objective this time, on top of (obviously) the science output, was to demonstrate how much compute can someone integrate during a regular working day, using only the two most cost effective SKUs for each Cloud provider. As before, we used Cloud resources from Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP), but we also mixed in the on-prem resources available though the Open Science Grid (OSG), XSEDE, the Pacific Research Platform (PRP) and others. As before, HTCondor was used as the workload management system.

https://www.linkedin.com/pulse/cost-effective-exaflop-hour-clouds-icecube-igor-sfiligoi/

No internet access to/from HPC nodes? File-based communication between execute nodes



PIC
port d'informació
científica

Read more about our current approach at http://tiny.cc/f158cz

BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Containers and Kubernetes

HTCondor Singularity Integration

- What is Singularity?

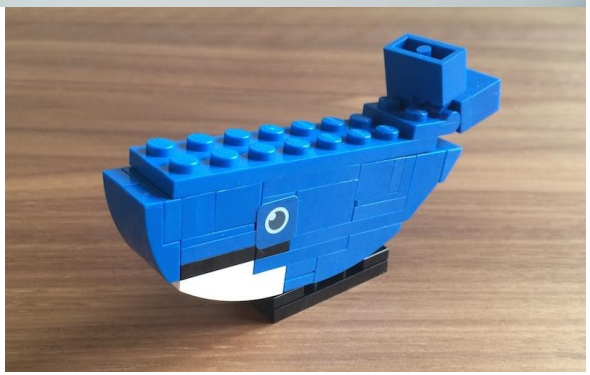
  Like Docker but…

  - No root owned daemon process, just a setuid
  - No setuid required (as of very latest RHEL7)
  - Easy access to host resources incl GPU, network, file systems

- HTCondor allows admin to define a policy (with access to job and machine attributes) to control
  - Singularity image to use
  - Volume (bind) mounts
  - Location where HTCondor transfers files

Docker Job Enhancements

- Docker jobs get usage updates (i.e. network usage) reported in job classad
- Admin can add additional volumes
- Conditionally drop capabilities
- Condor Chirp support
- Support for condor_ssh_to_job
  - For both Docker and Singularity
- Soft-kill (SIGTERM) of Docker jobs upon removal, preemption



More work coming

- From "Docker Universe" to just jobs with a container image specified
- Kubernetes
  - Package HTCondor as a set of container images
    - Check it out https://github.com/htcondor/htcondor/blob/master/build/docker/services/README.md
  - Launch a pool in a Kubernetes cluster
  - HTCondor-CE provision resources from a Kubernetes cluster (ie submit pilot pods)

GPUs

- HTCondor has long been able to detect GPU devices and schedule GPU jobs (CUDA/OpenCL)
- *New in v8.8:*
  - Monitor/report job GPU processor utilization
  - Monitor/report job GPU memory utilization
- *In the works*: simultaneously run multiple jobs on one GPU device
  - Specify GPU memory for scheduling ?
  - NVIDIA Multi-Instance GPU (MIG) for partitioning ?
  - Working with LIGO on requirements

Scheduling Enhancements

2 kinds of scheduling today:

Submitters (users)

- "Fair Share"
- Keeps historical usage
- Proportional priority
- No ceiling
- Ordered by priority, always!

Groups

- Quotas with max
- Hierarchy of groups share quota
- No historical usage
- Starvation order
  - But configurable!

Nikhef had some questions…

- How do we meet SLAs with communities?
- We've got monthly guarantees?
- How about a classical physics model?
  - "Power" vs "Work"

    (i.e. percentage of cluster vs annual allocation)

Work in Progress:

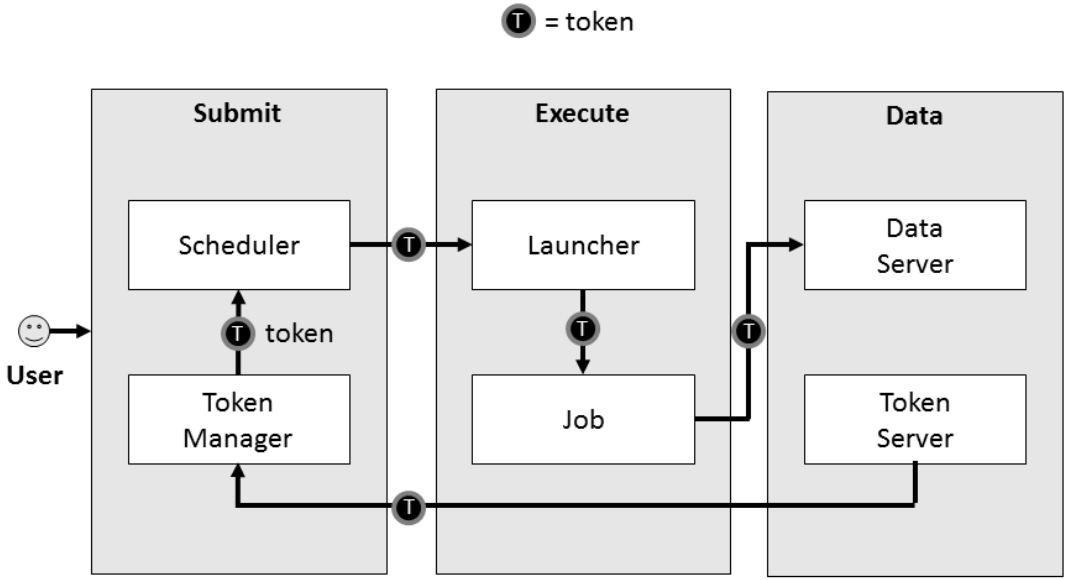
Merge Groups & submitters

- Hierarchy of groups ending in individual submitters
- Each group can have
  - History (optional)
  - Maximum
  - Minimum
  - Configurable ordering
- And knobs to allow existing system, if you like it

Security Enhancements

- Modernize ciphers support
  - No more MD5
  - AES encryption (hardware assisted)
- Tokens, tokens, tokens!

SciTokens: From identity certs to authorization tokens



SCITOKENS

- HTCondor has long supported GSI certs
- Then added Kerberos/AFS tokens w/ CERN, DESY
- Now adding standardized token support
  - SciTokens (http://scitokens.org) for HTCondor-CE, data
  - OAuth 2.0 Workflow → Box, Google Drive, AWS S3, …

IDTOKENS Authentication Method

- Several Authentication Methods
  - File system (FS), SSL, pool password….
- Adding a new "IDTOKENS" method
  - Administrator can run a command-line tool to create a token to authenticate a new submit node or execute node
  - Users can run a command-line tool to create a token to authenticate as themselves for job submission

… and this is a good time to transition to news about the HTCondor-CE!

 @HTCondor

https://twitter.com/HTCondor