HTCondor-CE: Updates and Futures

Brian Bockelman
ADC TIM 2017
HTCondor-CE is a CE

- HTCondor-CE performs the typical duties of a CE:
  - Exposes a remote service on the internet.
  - Integrates with the site AAI (does the X509/GSI “stuff”).
  - Provides resource allocation capabilities.
- This slide is boring! Basically, “it does what you expect”. More importantly…
HTCondor-CE is HTCondor

- With a few exceptions*, HTCondor-CE is only a special configuration of HTCondor.

  - HTCondor-CE can submit to SLURM because HTCondor natively submits to SLURM.

  - HTCondor-CE can authenticate with X509 because HTCondor can authenticate with X509.

- The HTCondor-CE is unique because sites/VOs that already use HTCondor depend on no new external projects.

*Exceptions include debug utilities, wrapper scripts, and monitoring add-ons
What Can HTCondor-CE Do?

- Because HTCondor-CE is HTCondor, most features pass through unmodified.
  - A new HTCondor feature automatically becomes a new HTCondor-CE feature.
- So, what can HTCondor do?
  - I want to spend my time talking about areas of potential collaboration I can see.
• At the core, the HTCondor-CE has a condor_schedd for managing jobs, which sends a ClassAd describing its status to a local collector.

• One status ad is forwarded from each CE to a central collector.

• The central collector serves as an information service about all installs. This can be correlated with MyOSG to get dynamic info about all registered CEs.

• For ATLAS, the central collector also exports a JSON file translating the ClassAd format into AGIS key-value pairs.
GLUE 1? GLUE 2?
Amazon EC2!

- For the HTCondor-CE, we wanted to go in a strongly different direction than GLUE:
  - DO NOT attempt to describe site batch system configuration or fairshare.
  - DO describe the **types of site resources and their properties** and what needs to be set to access them.
    - Put up a strong abstraction layer between site configs and the grid.
  - The CE’s ad has a *resource catalog* describing resources and what pilots need to define to access these.
    - A simplified version of this is exposed in the OSG configuration files; the underlying language is more powerful though!
    - Unfortunately, this currently is reduced out in the AGIS JSON compatibility layer.
  - Inspiration: Amazon EC2’s page of resource types.
Information Services

• A ClassAd is a set of key-expression pairs. Unlike a JSON dictionary of key-values, expressions are more powerful as they can reference other attributes or ads. Particularly, ClassAds provides a standardized matching language.

  • Given a resource catalog, one can search for all entries with \((\text{Cpus} > 4) \&\& (\text{Memory} > 7000) \&\& (\text{Color} =\neq \text{“green”})\).

  • Important: valid attributes do not need to be pre-defined.

• We are PURPOSELY schema-free: we have a rough consensus on commonly-defined attributes (CPUs, Memory, Whole Node requests) but WANT VOs to experiment with their own extensions.

  • The support for whole-node jobs started because one site introduced a new attribute; CMS picked it up; as interest increased, it became a standardized attribute in HTCondor-CE 2.2.

  • Custom attributes can be passed all the way through to the underlying batch system (this can get tricky for non-HTCondor systems, however...).

• We want your ideas and participation!

“We reject kings, presidents and voting. We believe in rough consensus and running code”

- David Clark (chief protocol architect in the development of the Internet)
Interested in more?

https://indico.fnal.gov/getFile.py/access?contribId=19&sessionId=8&resId=0&materialId=slides&confId=8580
Local Monitoring and Traceability

- HTCondor-CE’s collector is setup so each pilot job can upload a ClassAd of its current status.
  
  - gWMS-based pilots do this automatically (everything is HTCondor, after all!).
  
  - The “condor_advertise” binary can be invoked directly.
  
  - Starting in HTCondor-CE 3.0, the collector logs all updates. If you tell the collector which payload users are running, the CE logs this — fulfilling site traceability requirements.
  
  - Currently a fairly limited mechanism, but suffices to meet minimal requirements - room to grow!
Batch System Integration

• Except for HTCondor sites*, HTCondor speaks to the batch system via the `blahp` (same as CREAM).

• We extensively patch the `blahp` to pass along additional attributes.
  
  • Steady stream of tickets and improvements. **Please**: report bugs so we can fix bugs!

• Since the JobRouter can do any arbitrary transform of a pilot (including calling out to a script), there’s a huge variety of things one can do besides simply launch a batch system job.

• See Jose’s work on submitting to OpenStack: https://research.cs.wisc.edu/htcondor/HTCondorWeek2017/presentations/ThuCaballero_OpenStack.pdf

  *HTCondor-CE has a native pass-through.
Bonus: HTCondor for pulling jobs

• Starting in 2008, a HTCondor worker node has had the ability to fetch arbitrary work from non-HTCondor queues.

  • This is now called “the Vac model”.

• Has not (yet?) lit the HTCondor world on fire, but perhaps something that is feasible at large sites that can manage the credential responsibly and do some worker node specialization.

  • Would be quite interested in a demo that pulls directly from Panda into the worker node, with no grid anywhere!

  • Can start simple: work for a summer or GSoC student?

• See: http://research.cs.wisc.edu/htcondor/manual/current/4_4Hooks.html
CE as a Pilot Factory

• Before the HTCondor-CE, there was a prototype gLite CE based on HTCondor.
  • The interesting research topic was the CE aimed to be very lightweight - but spawn a “per-VO instance” on the CE host.
  • This is still feasible with the HTCondor-CE: it’s possible to spawn a lightweight pilot factory.
  • Or use the new “factory job” features where the schedd automatically instantiates new jobs.
• It should be possible for the CE to have an embedded ATLAS pilot factory that speaks to the CE locally.
  • Why stop there? It would also be possible to pull directly from PanDA and push to the schedd.
• N.B.: If you are interacting with HTCondor programmatically and using the command line tools, you are doing it wrong.
  • Always use the HTCondor python bindings.
  • htcondor-python.readthedocs.io
  • Now available in PyPI! Now with Python3 support!
Credential Handling

• It’s time to make plans for an orderly transition off X509/GSI.

  • This is most pressing for data transfers, but will eventually hit pilot job submission.

• HTCondor has always had multi-authentication-protocol support and flexible (too flexible?) authorization settings.

  • I.e., Kerberos, “normal” X509, GSI, password based.

• Looking for replacements to GSI: might more explicitly split up authentication step from passing a credential to the pilot.

  • HTCondor has a new “CREDMON” subsystem that allows users to manage / move tokens without HTCondor needing to understand the token itself.

  • The SciTokens project ([www.scitokens.org](http://www.scitokens.org)) is looking at OAuth2-inspired schemes and adding OAuth2 workflow support to HTCondor directly.

• There is a rich ecosystem of token-based authorization schemes out there: we would miss out if we muddled through with GSI.
Summary

• Areas of greatest potential overlap with ADC:
  
  • **Improving expressivity of the information services.** I’d like to see ATLAS sites experiment with new attributes and eventually get them in the base HTCondor-CE. I’d like to see more interesting projects done with the service discovery in the resource catalog.

  • **Different models for delivering jobs.** The CE is a rich platform for playing with pull-based models.

  • **New credential formats.** May benefit from a unified approach for data and jobs.