Interfacing HTCondor-CE with OpenStack: technical questions

Jose Caballero

HTCondor Week 2017

Disclaimer facts:

This work was done under the umbrella of OSG Technologies Investigations. So there were other people involved.

It was presented at CHEP, so this talk does not intent to repeat that one.

Slides can be found at
https://indico.cern.ch/event/505613/contributions/2227921/attachments/1340266/2043473/Oral-98.pdf

A copy of the future proceedings: http://www.usatlas.bnl.gov/~caballer/files/tmp/chep2016.pdf

A very similar work has also been done by Eric Sedore, who gave a talk yesterday.

I
Description of the project

Motivation:

In a nutshell: provide access to an OpenStack cluster via the HTCondor-CE.

Scenarios:

- The end user may or may not know about the OpenStack cluster. Jobs may be redirected transparently into it to satisfy special needs, or by demand.
- The particular VM image to be booted can be selected programmatically based on the job classads, or the end users may specify it directly by name convention, or by providing the URL to download it.

In any case, it should happen without requiring the users to have OpenStack credentials, OpenStack client installed or to know how to interact with the services.

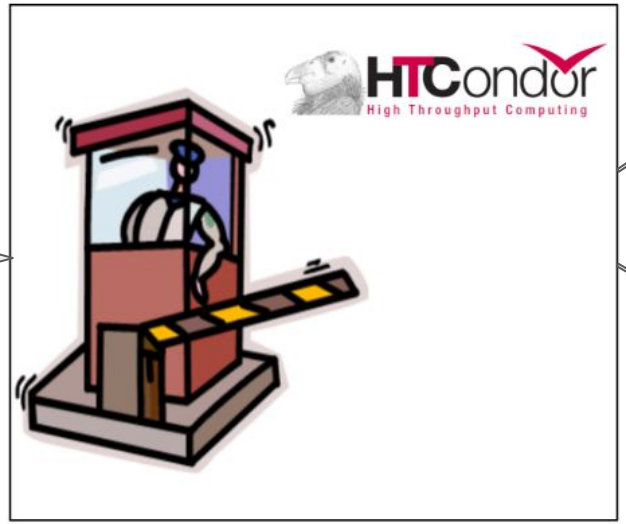
Job classads (proposal):

Requirements for the image Operative System	+opsys = "LINUX" +opsysname = "CentOS" +opsysmajorversion = 7
Requirements for the image flavor	+maxMemory = 28000 +disk = 10 +xcount = 8
To request a specific image or flavor	+virtualgridsite_image_name = "centos7-bare" +virtualgridsite_flavor_name = "m1.medium"
To provide a custom image	+virtualgridsite_url = <URL>
To request being able to ssh into the VM	+virtualgridsite_interactive_vm = true

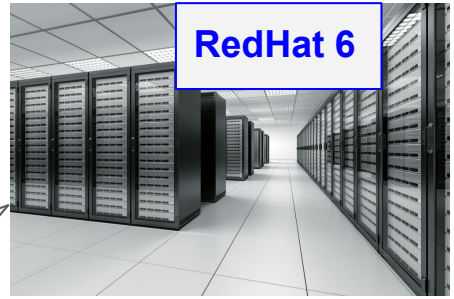
Scenario 1

Submit a job

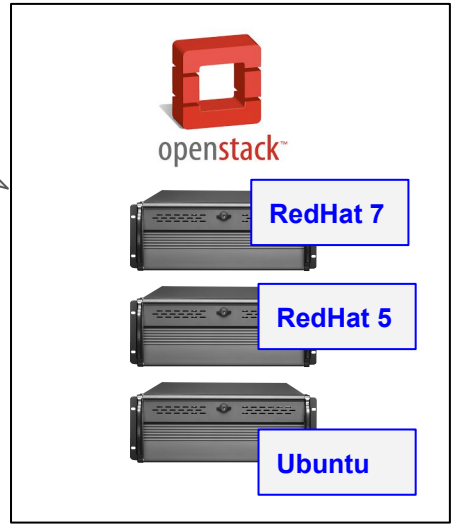
GRID Site



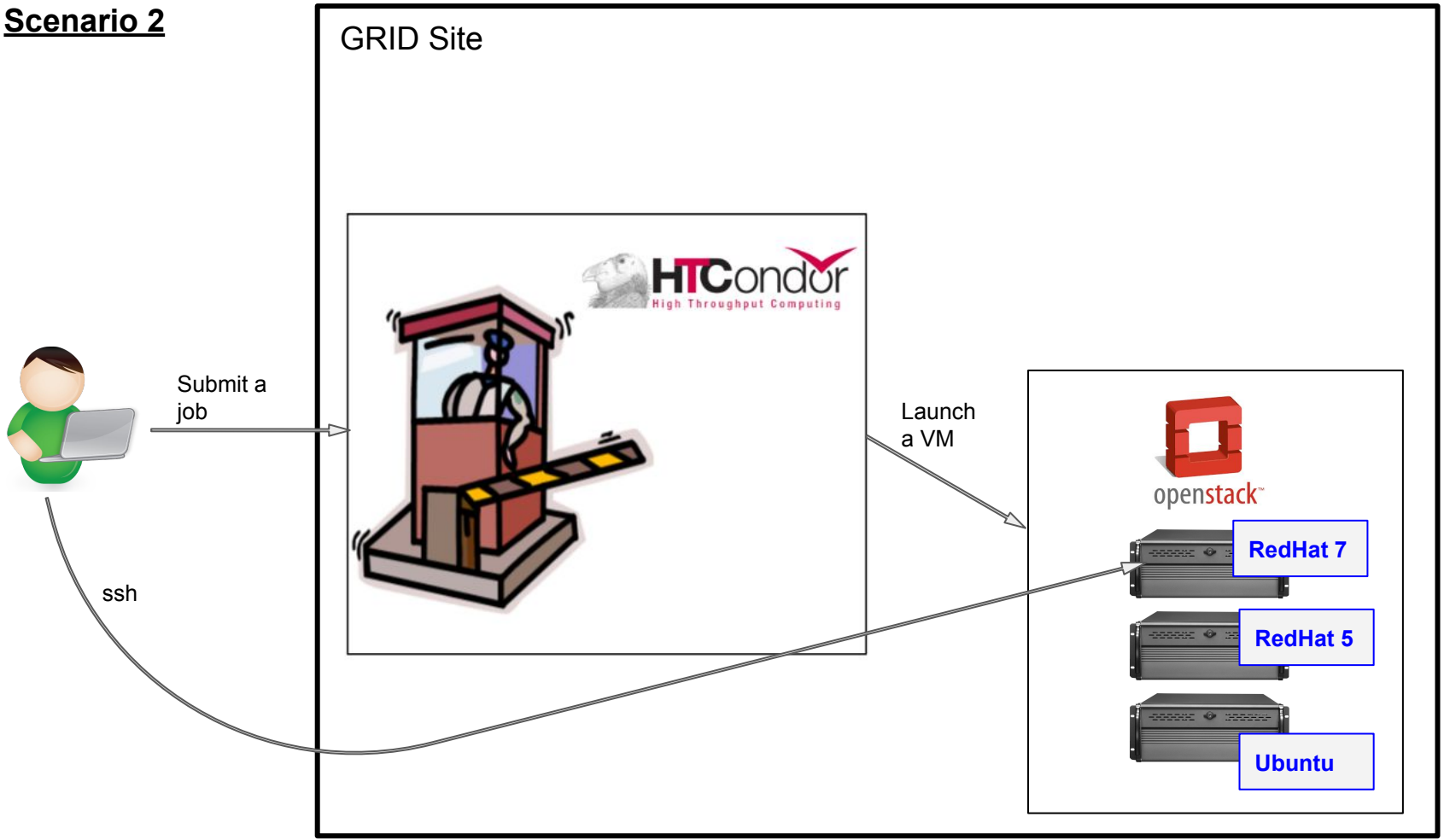
Run the job



Launch VM and run the job



Scenario 2


GRID Site
HTCondor
High Throughput Computing
Submit a job
Launch a VM
ssh
openstack
RedHat 7
RedHat 5
Ubuntu

Message for the community:

We also had an HTCondor instance in front of a CVMFS server.

If you want to provide access to a service but would prefer the users not to interact directly with it:

- to protect them from each other, and from themselves
- to protect the service
- to avoid forcing them to install client packages, configuration, credentials... and/or to learn new tricks
- ...

Just put HTCondor in front of it!

II
Technical Part

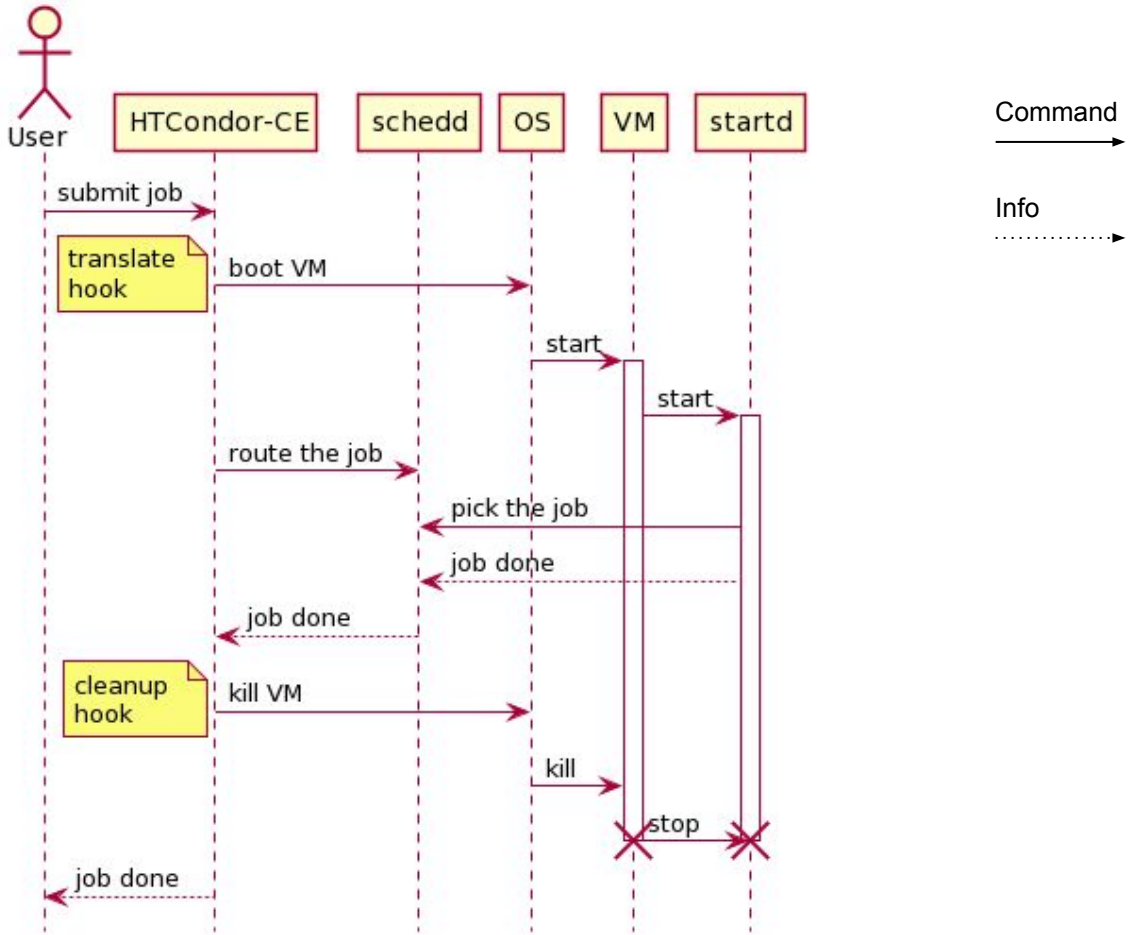
Solution: the HTCondor Job Router Hooks

Def: The HTCondor Job Router is an add-on to the condor_schedd that transforms jobs from one type into another according to a configurable policy.
http://research.cs.wisc.edu/htcondor/manual/v8.4/4_4Hooks.html#SECTION00542000000000000000

In other words: arbitrary code that can be executed at some points of the job life cycle:

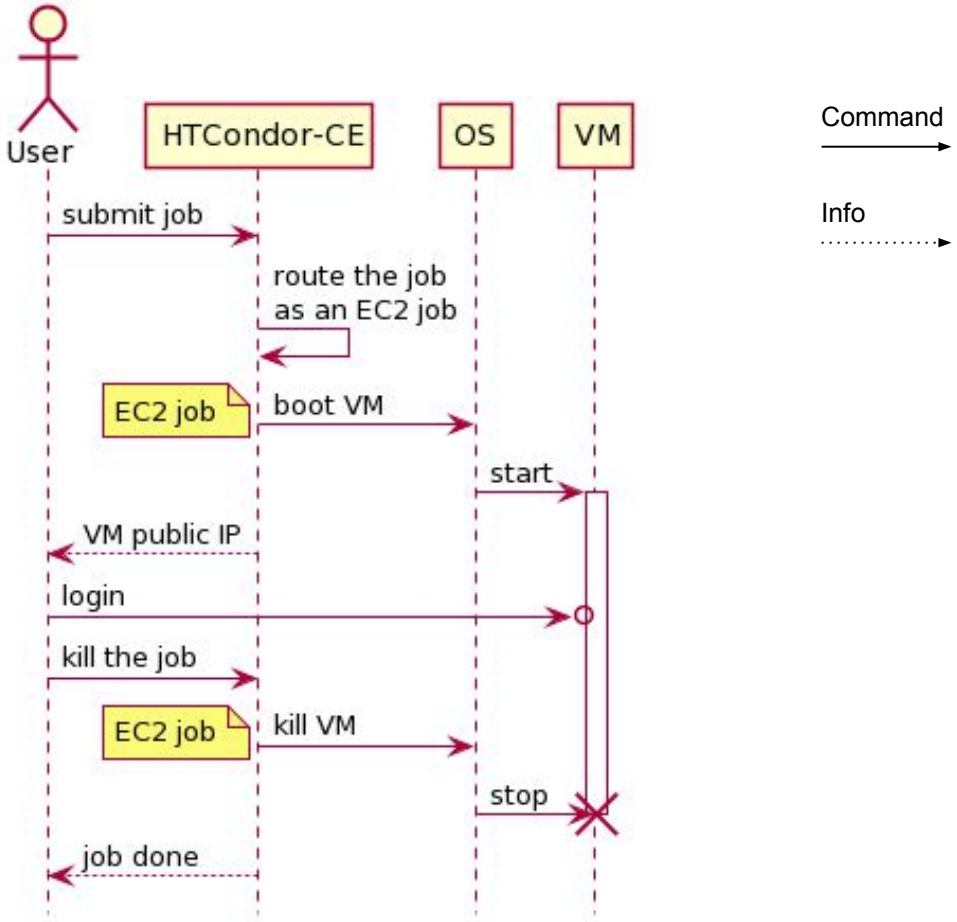
Translate	responsible for doing the transformation of the job and configuring any resources that are external to HTCondor if applicable.
Update	invoked to provide status on the specified routed job when the Job Router polls the status of routed jobs
Exit	invoked when the job has completed
Cleanup	invoked when the Job Router finishes managing the job

Scenario 1:



I know, I know, this is not real UML ...

Scenario 2:


User
HTCondor-CE
OS
VM
Command
Info
submit job
route the job
as an EC2 job
EC2 job
boot VM
start
VM public IP
login
kill the job
EC2 job
kill VM
stop
job done

This work was a new [at least to me] way to use HTCondor -beyond just a blind submission or simply passing jobs from CE queue to local batch queue-. Therefore:

- Not sure I chose the right, or optimal, way of doing things.
- We may have encountered unexpected scenarios where the current HTCondor features are used in ways that differ from the original purpose.
- And therefore there may be chances to make HTCondor even more robust and flexible.

So I have a few questions...

Question 1:

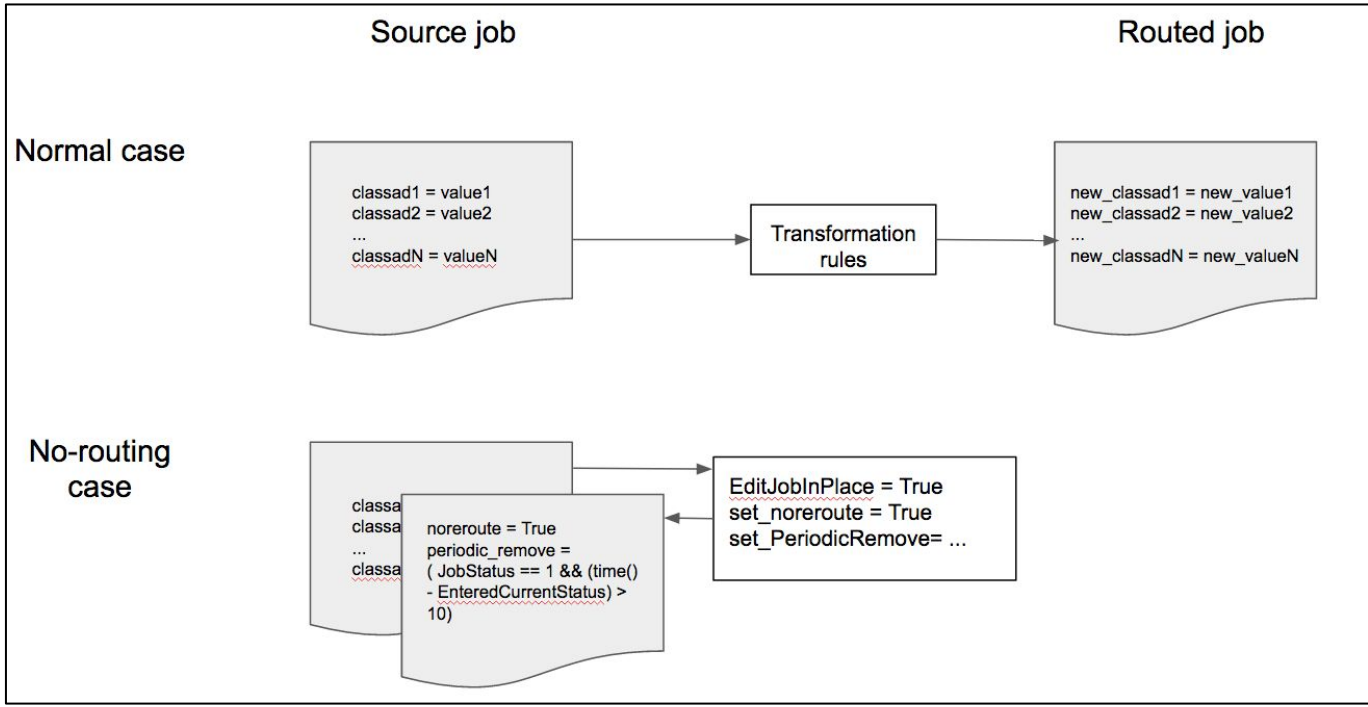
If the VM instantiation fails, I didn't find a clean way to "terminate" the job. A failed JobRouter hook makes the job to be re-routed again.

- Is there currently any mechanism to prevent failed jobs to be re-routed?
- If not, what about a new configuration variable (set to "False" by default) like **NoRouteOnFailure** or similar?
- Any other idea, new or existing?

Question 2:

If there is no host or VM that can run the job, I didn't find a clear way to not to route it. I inspect a set of config files with description of available resources (both static and images), and use a different route when there is no host/image to run the job.
So I had the policies defines in 3 different places:
the routing tables, the hooks, and the code to decide if routing or not.

- Is there a clean way to merge the 3 in 1?
- How does CMS do it?
- And the previous question still stands: a clean way to "abort" jobs that cannot be routed?



Question 3:

To get the value of the EC2 public IP, in order to allow the user to see it and ssh into, I query directly the collector in the CE looking for a new classad EC2ElasticIp to be added to the job.

IIRC, we try to mirror it into the source job classad (so a plain condor_q would show it), but it requires the cleanup hook to print out the job classads. We found that that triggers a couple of hidden bugs. I know that Jaime was working on fixing those.

- What is the status of that? Already fixed?
- Is that the [only] mechanism to inject new classads into the job in such a way the user can see them via condor_q?

I can see that the ability of injecting new items into the source job classad as the routed one is being managed could have many uses. So any mechanism to makes it possible -and easy- can be very useful.

Question 4:

Knowing that HTCondor is job-based...

Let's image 100 jobs are submitted, all identical, but they cannot be routed. Right now each one of them needs to be processed, to "fail" in the same way.

It would be interesting if, once the job with ProcId=0 has been processed, the outcome is stored, so the rest of jobs with same ClusterId can skip all the checks/matchmaking/calculations/... and jump to that same result.

- Is HTCondor actually doing something like that already?

Question 5:

This work was done using the HTCondor-CE (focus on the -CE part) because it comes with the whole JobRouter enabled by default. But it imposes, by default, the usage of GSI authentication. If we wanted to avoid the needs for X509 credentials, what would be the best/easiest way?

- Removing the GSI-related configuration from the HTCondor-CE installation?
- Installing a regular HTCondor (w/o -CE) and run the JobRouter daemon?
- Something else?

Question 6:

The HTCondor-CE installation comes with some restrictive setup in its configuration that needs to be overriden:

- By default, only allow jobs to be routed to JobUniverse 1 or 5. JobUniverse 9 needed for routing to EC2.
- it wants to report to some OSG monitoring. HTCondor-CE is being used outside OSG, and more other purposes than a CE (like this talk). Can we make this more optional?

Many thanks to all experts that helped me answering tons of questions during the development of the prototype, and, hopefully, today.