HTCondor Annexes with AWS

As an initial product resulting from our collaboration with Amazon, we propose a tool to automate and simplify the process of making AWS resources available to a specific subset of HTCondor submission points. After an initial setup and authorization procedure, the on-demand addition of these AWS resources should be able to be initiated by an end-user without the need to involve the central HTCondor site administrator. These resources then need to be made available (exclusively) to that submission point, and configured to release themselves when there are no more jobs to run or after a specified time-period has expired. The process should not require the user to adjust or resubmit their jobs.

The initial goal is a command tool invoked at the submit point by the end-user along the form of:

condor\_annex --set-size 1000 –lease-time 24 --project “144PRJ22”

This would set the size of the project’s “annex” to 1000 EC2 instances, with a lease time of 24 hours, and tagging those instances with the given project ID for accounting purposes. No further action by the user should be required.

The tool will ensure clean-up at the end of the lease time without further user intervention and regardless of the state of the user’s submit point, i.e. even if the machine serving as the submission point is offline all EC2 instances should be halted when the lease expires. Furthermore, the tool should be idempotent such that setting the size of the annex to 1000 nodes should always result in 1000 nodes, neither more nor less.  This will simplify crash recovery and increase user confidence. Additionally, the tool should be highly scalable, enabling the addition of thousands of instances in a manner that does not require constant EC2 API calls (i.e. leverage batch AWS mechanisms such as auto-scaling groups).

# Roadmap and Milestones

The roadmap consists of development of a prototype which we currently anticipate will be ready by the end of November, followed by friendly-site beta testing starting in mid-December. After testing, product integration work to prepare for a general release will take place.

Prototype development milestones

* ETA start date: Already underway
* ETA for completion date: 11/30/2015

1. Make the secret information necessary to join the pool securely available to the new instances.  (Upload the pool password file to a private bucket on S3.) [Done, assuming we took the right approach]
2. Relay the lease duration and pool configuration to the instances.  (Make use of EC2 user data and/or *cfn-init*.)  Each instance will use this information to securely join the pool and to back up the lease by terminating after the lease expires.  Instances will also be configured, for parsimony, to terminate if they fail to join the pool in a reasonable amount of time or don’t get a job after a reasonable amount of time.  (With *cfn-init* or by using a preconfigured AMI.) [3 weeks]
3. Create a leased AWS AutoScaling Group tagged with the project ID, whose instances have permissions to access the secure store established in #1. The lease will consist of a custom heartbeat metric, and an AWS lambda function will tear down all aspects of the AutoScaling group should the lease expire. [1 week remaining]
4. Set the desired (and max) number of instances for that AutoScaling group. [Done]
5. Wait for that number of instances to report to the pool. [2 days]
6. Clean up the secret information.  (Delete the file from S3.) [2 days]
7. If invoked a second time, condor\_annex will check to determine if the leased AutoScaling Group already exists.  If so, it will skip that step.  It must still complete the other steps (because we deleted the credentials the first time around).  [2 days.]
8. Should HTCondor decide that a particular instance is no longer needed (i.e. there is not enough work queued to keep this instance busy), implement a transactional mechanism to allow this instance to terminate in a manner such that the AutoScaling Group will not attempt to replace it. [1 week]

## Testing milestones

* ETA start date: 12/16/2015
* ETA completion date: TBD

1. Beta release for friendly users.  Before showing this tool to our users, we need to make sure that, at least for us, it functions reliably and does the things we said it was going to do.  Much of this testing will happen incidental to development, but a specific effort should be undertaken.  The beta release should also include README-style documentation about what the users should do.
2. Perform scale tests with 100,000 cores.   Additionally, verify the functionality and performance of multiple annexes.

## Product integration milestones

* ETA start date: TBD
* ETA completion date: TBD

1. Code hardening, e.g. checks and error messages to validate user input and protect against things users shouldn’t do.
2. Address defects uncovered during testing phase.
3. Address feedback from beta users.
4. Add regression tests. HTCondor development uses a continuous integration model, and thus we run all regression tests upon every source code commit. It is not clear how we should test the implementation proper on every build against AWS; perhaps some sort of dry-run option or AWS simulator?
5. Documentation

# Future Work

* Make it easy to extend the lease.  (Add an option to the command-line tool.)
* Automatically extend the lease for some conditions, perhaps until a specified set of work is done.  (That is, by cluster or by DAG.)
* Use HTCondor’s scheduler to record the user’s intent for their annex.  (That is, submit a scheduler-universe job.)  This offers persistence, automatic restarts (after tool or system crash), and potentially a familiar interface (hold/remove).