HTCondor Annex
(There are many clouds like it, but this one is mine.)
Annex means (an) Addition

› An annex is “a building joined to main building, providing additional space or accommodations”

› An HTCondor annex could provide:
  • more machines
  • specialized hardware
  • specialized policies

› Use `condor_annex` to acquire computational resources from the cloud
What is the cloud?

- Commercial services which rent computational resources by the hour
- They own the hardware
- You provide the software ("disk image")
  (OS, applications, configuration, maybe data)
- You can configure the networking and storage as well
Why not keep using the Grid?

- Cloud resources are typically available sooner and in greater quantity
- Cloud resources are more customizable (networking, software, configuration/policy, etc)
The `condor_annex` tool was first released two months ago, in HTCondor 8.7.0.

Improved in 8.7.1 and still under active development.

To add a GPU to the pool:

```
condor_annex -count 1 \  
   -annex-name ToddsGPU \  
   -aws-on-demand-instance-type p2.xlarge
```
Use Case 1: Deadlines

- How important is that user’s deadline?
  - Is she willing to spend money on it?
- Make it easy for the user to run jobs in the cloud, trading money for job completion
  - automation
  - sane defaults
  - admin configuration
Use Case 2: Capability

› Meet intermittent needs for hardware
  • with lots (TBs) of memory
  • with GPUs
  • with fast local storage of shared data
    • especially if one of the AWS public data sets

› Special job policies, like long runtimes
Use Case 3: Capacity

- Lower costs through higher utilization, with cloud rentals covering usage bursts
- Without `condor_annex`, expanding an HTCondor pool into the cloud isn’t easy
A brief overview of the

ANNEX LIFECYCLE
Annex Lifecycle

1. User requests resources
2. Then `condor_annex` starts resources
3. Resources join pool
4. Resources stop spending money
1. Request Resources

› User requests may specify:
  • hardware (CPUs, memory, disk, GPUs)
  • software (OS, applications, configuration, data)
  • number of resources and maximum lifetime

› Two types of resource
  • on-demand: pricier, yours until you stop them
  • spot: cheaper, can be lost to a higher bidder after a two-minute warning
    • only suitable for short or resumable jobs
Amazon offers, and condor_annex supports, a mechanism called “Spot Fleet”

A “Spot Fleet” automatically chooses the cheapest way to satisfy spot resource requests which aren’t picky about their hardware requirements
2. Start Resources

`condor_annex` machinery starts each resource, specifying two extra things:

- a “client token” (intended for fault tolerance); we use it to indelibly mark each resource as part of a particular annex
- a “role,” which helps connect the resource to your HTCondor pool
3. Resource Securely Joins Pool

- A “role” is a set of permissions.
- The annex role’s permissions are to:
  - read a file from otherwise-private cloud storage
  - look at the role
- When HTCondor starts up, it inspects the role and downloads the file named there.
- Admins: this leaves the “user data” available for you to use.

```bash
condor_status -annex ToddsGPU
```
4. Resources Stop Spending

› Fail-safe: the resources *always* stop
  • Even the user’s machine goes offline

› Implemented entirely in the cloud
  (Uses AWS Lambda and CloudWatch Events)

› Checks the duration every five minutes
  (Uses “client token” to identify annex instances)

```console
condor_off -master -annex ToddsGPU
```
Opportunities for Improvement

- Only works with Amazon
- Hard to learn about instances that haven't joined the pool yet
- Can’t change annex duration without adding nodes
- Requires admin help to run jobs from an existing pool
CUSTOMIZATION
A resource *must* have a disk image (OS, applications, configuration, maybe data)

HTCondor provides a default disk image that should work for most users

If you create disk images for your users, you can copy and customize the default image for them, or make your own from scratch, subject to a few restrictions
Disk Image Requirements

- The default disk image does all this
- Start-up to fetch config and security data
  - currently requires AWS CLI tool
- HTCondor configured to turn off when it’s idle for too long.
  - `STARTD_NOCLAIM_SHUTDOWN`
- HTCondor configured to turn instance off when the master exits.
  - `DEFAULT_MASTER_SHUTDOWN_SCRIPT`
Image Suggestions

- The default disk image does all this
- Advertise instance ID in master and startd
- Use public IP addresses and set `TCP_FORWARDING_HOST`
- Turn communications integrity and encryption on
- Encrypt the run directories
WHAT CAN YOU DO TODAY?
Initial Set-Up

› Follow the initial set-up instructions to connect condor_annex to an AWS account via HTCondor configuration

› Assumptions (mostly for simplicity):
  • new, private HTCondor pool
  • public IP address, open port
  • Linux

1. Deadlines
   • jobs in another queue require admin help
2. Capability – should be usable for admins
3. Capacity – should be usable for admins

Contact us if you have trouble adapting the instructions for your particular situation.