Highly available HTCondor pools in Google Cloud

Tom Downes

2023-07-13
Cloud HPC Toolkit Objective

“Make it easy for customers and partners to deploy repeatable turnkey HPC environments following Google Cloud’s HPC HTC best practices”
Open source and accepts contributions, open discussions and feature requests

Uses open source multi-cloud tools (Terraform/Packer) and Ansible for configuration

Supports scheduler and storage solutions from Google and partners
Cloud HPC Toolkit
https://cloud.google.com/hpc-toolkit
New Toolkit feature: simple deploy command

ghpc create blueprint.yaml

Creates a deployment (directory) from a blueprint. The directory contains:

- Terraform modules
- Packer "templates"
- Ansible plays and other scripts included with Toolkit or added by user

ghpc deploy deployment_directory

Automates execution of Terraform and Packer to deliver infrastructure with custom software and configuration.
New Toolkit support for HTCondor

- Automated image building via deploy
- CentOS 7 + Rocky Linux 8 for all nodes; Windows for execute points
- IDTOKEN authentication for all nodes
- Support for Spot instances
- Simplified blueprint (very soon)
- Central Manager and Access Points in auto-healing instance groups (already the case for Execute Points)
- Nearly as soon: Support for N>2 machine configurations
  - CPU/GPU/Region/etc.
Example: Auto-scaling HTCondor Pool Groups 1 and 2

vars:
  project_id: ## Set GCP Project ID Here ##
  deployment_name: throughput-computing-talk
  region: us-central1
  zone: us-central1-c
  new_image_family: htcondor-10x

deployment_groups:
  - group: primary
    modules:
      - id: network1
        source: modules/network/vpc
        outputs:
          - network_name

  - id: htcondor_install
    source: community/modules/scripts/htcondor-install
    settings:
      condor_version: 10.5.1

  - id: htcondor_install_script
    source: modules/scripts/startup-script
    use:
      - htcondor_install

- group: packer
  modules:
    - id: custom-image
      source: modules/packer/custom-image
      kind: packer
      use:
        - network1
        - htcondor_install_script
    settings:
      disk_size: $(vars.disk_size_gb)
      source_image_family: hpc-rocky-linux-8
      image_family: $(vars.new_image_family)
Example: Auto-scaling HTCondor Pool Groups 1 and 2

```python
vars:
  project_id:  ## Set GCP Project ID Here ##
  deployment_name: htcondor-pool
  region: us-central1
  zone: us-central1-c
  disk_size_gb: 100
  new_image_family: htcondor-10x

deployment_groups:
  - group: primary
    modules:
      - id: network1
        source: modules/network/vpc
        outputs:
          - network_name

      - id: htcondor_install
        source: community/modules/scripts/htcondor-install

      - id: htcondor_install_script
        source: modules/scripts/startup-script
        use:
          - htcondor_install

  - group: packer
    modules:
      - id: custom-image
        source: modules/packer/custom-image
        kind: packer
        use:
          - network1
          - htcondor_install_script
    settings:
      disk_size: $(vars.disk_size_gb)
      source_image_family: hpc-rocky-linux-8
      image_family: $(vars.new_image_family)
```

By itself, this blueprint would produce a generic HTCondor image for all pools in Cloud. The install script can be customized to include your applications.
Example: Auto-scaling HTCondor Pool Group 3

- group: pool
  modules:
  - id: htcondor_base
    source: community/modules/scheduler/htcondor-base
    use:
      - network1
  - id: htcondor_secrets
    source: community/modules/scheduler/htcondor-pool-secrets
    use:
      - htcondor_base
  - id: htcondor_cm
    source: community/modules/scheduler/htcondor-central-manager
    use:
      - network1
      - htcondor_secrets
      - htcondor_base
    settings: ...

- id: htcondor_execute_point
  source: community/modules/compute/htcondor-execute-point
  use:
    - network1
    - htcondor_secrets
    - htcondor_base
    - htcondor_cm
  settings:
    min_idle: 2
    ...

- id: htcondor_access
  source: community/modules/scheduler/htcondor-access-point
  use:
    - network1
    - htcondor_secrets
    - htcondor_base
    - htcondor_cm
    - htcondor_execute_point
  settings: ...
Example Runner snippet

Modular nature of solution enables you to refactor this to adopt, e.g., Vault

- name: Fetch IDTOKEN to advertise execute point
  ansible.builtin.copy:
    dest: /etc/systemd/system/condor.service.d/token-fetcher.conf
    mode: 0644
    content: |
      [Service]
      ExecStartPre=gcloud secrets versions access latest \\
      --secret {{ xp_idtoken_secret_id }} \\
      --out-file /etc/condor/tokens.d/condor@{{ trust_domain }}
    notify:
    - Reload SystemD
HTCondor on Windows

- Most challenges are my own inexperience automating Windows
- TJ has been a **significant** help!
- But you don’t have to! Toolkit performs
  - Python installation
  - GPU driver installation
  - HTCondor installation
  - IDTOKEN fetching
IDTOKENs

- "60% of the time, it works every time"
- Experience running commands as root on fresh install is poorer
- "condor_reconfig" becomes "systemctl reload condor"

```
UID_DOMAIN = c.toolkit-demo-zero-e913.internal
TRUST_DOMAIN = c.toolkit-demo-zero-e913.internal
use role:get_htcondor_central_manager

# due to https://tinyurl.com/htc-2023-trust
# this is a different configuration from

use role:get_htcondor_central_manager
UID_DOMAIN = c.toolkit-demo-zero-e913.internal
TRUST_DOMAIN = c.toolkit-demo-zero-e913.internal
```
+RequireSpot just works

- Each Cloud Machine advertises its attributes (region, zone, etc.)
- With N=2, "1 is Spot, 1 is Not"
- N>2 becomes less HTCondor-native in terms of scheduling
- Initial work with Todd Miller to "hijack" Rooster mechanism to develop a true autoscaling signal is fruitful but early days

```
JOB_TRANSFORM_NAMES = SPOT_DEFAULT, SPOT_REQS

JOB_TRANSFORM_SPOT_DEFAULT @=end
    DEFAULT RequireSpot False
@end

JOB_TRANSFORM_SPOT_REQS @=end
    REQUIREMENTS ! unresolved(Requirements, "^CloudInterruptible$")
    SET Requirements $(MY.Requirements) && (CloudInterruptible is My.RequireSpot)
@end

SUBMIT_REQUIREMENT_NAMES = REQSPOT
SUBMIT_REQUIREMENT_REQSPOT = isBoolean(RequireSpot)
SUBMIT_REQUIREMENT_REQSPOT_REASON = "Jobs must set +RequireSpot to either True or False"
```
Contributing back

- SchedD HA bug reported/\texttt{resolved}
- Filesystem mount order reported/\texttt{resolved}
- TRUST\_DOMAIN missing from manual reported/\texttt{resolved}
- I owe TJ a PR for Windows "Error 1722"

----- The following addresses had permanent fatal errors -----
<htcondor-admin@cs.wisc.edu>
(reason: 550 5.7.1 rejected by DMARC policy for google.com)

----- Transcript of session follows ----- 
... while talking to shale.cs.wisc.edu:
>>> DATA
<<< 550 5.7.1 rejected by DMARC policy for google.com
554 5.0.0 Service unavailable

So many reports, I've been blocked!