7/25/2019

Hi all,

In Google Cloud, the process for deleting data is completed in 3 high-level stages:

1. Deletion request is submitted.
2. Data is marked and isolated for deletion
3. Data is securely deleted.

All data stored is encrypted at rest by default, with no additional action required. The deletion process varies by the storage product being used. Attached is a white paper that covers deletion across Google Cloud products to the destruction of physical infrastructure media.

Below is a youtube video to provide an additional overview:
https://www.youtube.com/watch?v=WB6xrgl9eGk

7/26/2019

[External-Shared] 20190722 - UW Office Hours Follow up

- Documentation on setting up a VPN on Google Cloud?
  - Google Cloud VPN uses an IPsec VPN tunnel to encrypt data traveling between two networks over the public Internet.
  - GCP offers two types of Cloud VPN gateways, HA VPN and Classic VPN.
    - HA VPN uses multiple tunnels to provide a high-availability (HA) Cloud VPN solution.
      - It is in Beta so there is no current SLA. HA VPN will provide an SLA of 99.99% service availability at GA.
    - Classic VPN has a single interface, a single external IP address, and supports tunnels using dynamic (BGP) or static routing.
  - Below are documentation to create a GCP-Pear VPN gateway
    - HA VPN: https://cloud.google.com/vpn/docs/how-to/creating-ha-vpn
    - Classic VPN using dynamic routing: https://cloud.google.com/vpn/docs/how-to/creating-vpn-dynamic-routes
    - Classic VPN using static routing: https://cloud.google.com/vpn/docs/how-to/creating-static-vpns
    - Configuring the Peer VPN gateway: https://cloud.google.com/vpn/docs/how-to/configuring-peer-gateway
    - Checking VPN status: https://cloud.google.com/vpn/docs/how-to/checking-vpn-status

- Client libraries for Google Cloud Storage (GCS)
  - GCS has support for client libraries for languages:
    C++, C#, GO, JAVA, NODE.JS, PHP, PYTHON, RUBY
    - https://cloud.google.com/storage/docs/reference/libraries
  - GCS has REST APIs for JSON and XML:
    - JSON: https://cloud.google.com/storage/docs/json_api/
    - XML: https://cloud.google.com/storage/docs/xml-api/overview

- GCS has a CLI called gsutil:
  - https://cloud.google.com/storage/docs/gsutil

- What are alternatives to X11?
  - We've seen VNC used. Check out articles like these: https://medium.com/google-cloud/linux-gui-on-the-google-cloud-platform-800719ab27c5

- How to get detailed billing info?
  - Labels are used to segment billing costs to better understand the budget. Additional info here: https://cloud.google.com/resource-manager/docs/creating-managing-labels
  - Labels can be applied on each partition as key-value pairs in the slurm-cluster.yaml.

7/29/2019

20190729 - UW Office Hours Follow up

1. API documentation on modifying labels for GCP resources.

a. Labels are key-value pairs attached to GCP resources. Information about labels is forwarded to the billing system, so you can break down your billing charges by label.

b. Labels have a distinct key-value pair limit of 1000 per 1 hour window per service per project. For example, if you deploy a project with 2,000 VMs, each with a distinct label, the service reports metrics are preserved for only the first 1,000 labels that exist within the one-hour window.

c. For Google Compute Engine:

i. You can interact with the Google Compute API through REST, Google Cloud Python Client Library, or tools like libcloud.
   1. Below is an example of making REST calls using Python: https://2.python-requests.org//en/latest/user/quickstart/

ii. Below is documentation to authenticate to the API: https://cloud.google.com/compute/docs/api/how-tos/authorization

iii. Below is documentation to adding and removing labels: https://cloud.google.com/compute/docs/labeling-resources#adding_or_updating_labels_to_existing_resources

iv. Below is a general python client library example: https://cloud.google.com/compute/docs/tutorials/python-guide

v. Below are the API rate limits: https://cloud.google.com/compute/docs/api-rate-limits

d. For additional research, below is the API explorer for all Google APIs: https://developers.google.com/apis-explorer/

8/14/2019

Unfortunately, I see internally the request is still in line to get approved. Due to the large increase of cores, it requires additional approvals from our engineering team. I have added a comment to try to bump up the request. If this is affecting our project schedule, you can try to request an escalation from the support engineer.

For background, quota escalations are based on availability of resources and the reputation of the account. The NIH account has the highest marks for all reputation fields. For incremental increases, customers often get auto-approved. For example, a typical auto-approval for vCPU is +500 cores. This account has a auto-approval max of 5,000 cores which is still significantly lower than the requested amount. Therefore, it requires additional evaluation and approvals from our engineering team.

8/19/2019

[External-Shared] 20190819 - UW Office Hours Follow up

1. Examples to query BigQuery billing export can be found here: https://cloud.google.com/billing/docs/how-to/bq-examples

2. Additional details on the UNNEST BigQuery function can be found here: https://cloud.google.com/bigquery/docs/reference/standard-sql/arrays#flattening-arrays

   https://medium.com/firebase-developers/using-the-unnest-function-in-bigquery-to-analyze-event-parameters-in-analytics-fb828f890b42

3. BigQuery can be integrated with Data Studio for visualization of spending in a dashboard. Additional details here: https://cloud.google.com/billing/docs/how-to/visualize-data

4. The schema of the BigQuery billing export can be found below:

_PARTITIONTIME	TIMESTAMP	NULLABLE
billing_account_id	STRING	NULLABLE
service	RECORD	NULLABLE
service.id	STRING	NULLABLE
service.description	STRING	NULLABLE
sku	RECORD	NULLABLE
sku.id	STRING	NULLABLE
sku.description	STRING	NULLABLE
usage_start_time	TIMESTAMP	NULLABLE
usage_end_time	TIMESTAMP	NULLABLE
project	RECORD	NULLABLE
project.id	STRING	NULLABLE
project.name	STRING	NULLABLE
project.labels	RECORD	REPEATED
project.labels.key	STRING	NULLABLE

project.labels.value	STRING	NULLABLE
project.ancestry_numbers	STRING	NULLABLE
labels	RECORD	REPEATED
labels.key	STRING	NULLABLE
labels.value	STRING	NULLABLE
system_labels	RECORD	REPEATED
system_labels.key	STRING	NULLABLE
system_labels.value	STRING	NULLABLE
location	RECORD	NULLABLE
location.location	STRING	NULLABLE
location.country	STRING	NULLABLE
location.region	STRING	NULLABLE
location.zone	STRING	NULLABLE
export_time	TIMESTAMP	NULLABLE
cost	FLOAT	NULLABLE
currency	STRING	NULLABLE
currency_conversion_rate	FLOAT	NULLABLE
usage	RECORD	NULLABLE
usage.amount	FLOAT	NULLABLE
usage.unit	STRING	NULLABLE
usage.amount_in_pricing_units	FLOAT	NULLABLE
usage.pricing_unit	STRING	NULLABLE
credits	RECORD	REPEATED
credits.name	STRING	NULLABLE
credits.amount	FLOAT	NULLABLE
invoice	RECORD	NULLABLE
invoice.month	STRING	NULLABLE
cost_type	STRING	NULLABLE

8/26/2019

20190826 - UW Office Hours Follow up

1. How do you query based on timestamp rather than just date?

a. This one was quite obvious that we missed. We were originally converting the `usage_start_time` and `usage_end_time` using the `DATE()` function to be filtered in the following format: `2019-08-20`. However, since the dataset itself is in a `TIMESTAMP` format we don't need to pass any function to filter based on timestamp.

b. Below is an example:

```
SELECT
  labels.key as key,
  labels.value as value,
  SUM(cost) as cost
FROM roeric.demo.gcp_billing_export_v1_0175F5_3A4DFC_D724A7
LEFT JOIN UNNEST(labels) as labels
WHERE usage_start_time >= "2019-06-25 19:15:17 UTC" AND
usage_end_time < "2019-08-20 19:50:39 UTC"
GROUP BY 1, 2;
```

2. How do we identify the resources associated with no labels attached?

a. We can pull back all the column fields to better identify the resources that do not have labels associated with it.

b. Below is an example:

```
SELECT
  *
FROM roeric.demo.gcp_billing_export_v1_0175F5_3A4DFC_D724A7
LEFT JOIN UNNEST(labels) as labels
WHERE usage_start_time >= "2019-06-25 19:15:17 UTC" AND
usage_end_time < "2019-08-20 19:50:39 UTC" AND labels.key
IS NULL AND sku.description ="N1 Predefined Instance Core
running in Americas";
```

3. How do we install the Stackdriver agent?

a. Stackdriver has two optionally installed agents: a monitoring agent and a logging agent.

i. Stackdriver monitoring agent allows stackdriver to pull back additional system metrics (e.g. memory)

1. Further details can be found here: https://cloud.google.com/monitoring/agent/

ii. Stackdriver logging agent allows stackdriver to pull back application logs from a default list of applications (e.g. tomcat) or custom configuration.

1. Further details can be found here: https://cloud.google.com/logging/docs/agent/

b. Stackdriver monitoring agent can be installed with the following command:

i. `sudo echo "curl -sS https://dl.google.com/cloudagents/install-monitoring-agent.sh | sudo bash; sudo systemctl status stackdriver-agent" >> /home/$USR/$CMD_FILE`

ii. Metrics should be automatically loaded to the Stackdriver UI once installed. Additional installation details can be found here:

1. https://cloud.google.com/monitoring/agent/install-agent

c. Stackdriver logging agent can be installed with the following command:

.sudo echo "curl -sS https://dl.google.com/cloudagents/install-logging-agent.sh | sudo bash; sudo systemctl status google-fluentd" >> /home/$USR/$CMD_FILE

i. Logs should be automatically loaded to the Stackdriver UI once installed.Additional installation details can be found here:

1. https://cloud.google.com/logging/docs/agent/installation

d. Stackdriver monitoring agent can be stop and started through the following commands:

.sudo systemctl [start/stop/restart] stackdriver-agent

e. Stackdriver logging agent can be stop and started through the following commands:

i.sudo systemctl [start/stop/restart] google-fluentd

8/28/2019

Regular Persistent Disks (SSD and HDD) are stored in a specific zone (user can specify), e.g. us-west1-a *or* us-west1-b. This storage has built-in redundancy to protect your data against failure. Disks can be snapshotted to recover from a single point in time.

Regional Persistent Disks (SSD and HDD) are stored in two zones in a specific region (user can specify), e.g. us-west1-a *and* us-west1-b. Two use cases this can be helpful:

- Performance for workloads across two different zones that need to read from a single read-only attached PD (You cannot attach a PD to more than on VM in read/write mode).
- To meet disaster recovery and failover requirements. It will have lower Recovery Point Objective (RPO) and Recovery Time Objective (RTO) compared to using standard persistent disk snapshots.

In regards to performance, standard and regional PDs share the same IOPs performance but Regional PDs do you have less write throughput due to the need to write twice as much. https://cloud.google.com/compute/docs/disks/performance#type_comparison

Both options share the same compute engine SLA of 99.99%. https://cloud.google.com/compute/sla

8/29/2019

For Google compute engine as of writing:

For inside the Google Cloud Network:

- Egress in the same zone is free for internal IP, e.g. us-west1-a <-> us-west1-a
- Egress between zones in the same region, e.g. us-west1-a <-> us-west1-b is $0.01 per GB
- Egress between regions within the US and Canada, e.g. us-east1-a <-> us-west1-a is $0.01 per GB

For internet egress for us-west (based on monthly usage):

- $0.12 per GB for 1TB or less monthly usage
- $0.11 per GB for 1-10TB monthly usage
- $0.08 per GB for 10+ TB monthly usage

The price for egress outside of the Google Cloud network drops to $0.02 per GB if a Cloud Interconnect is used.
https://cloud.google.com/interconnect/pricing

The pricing calculator also has a "networking" tab (6th service from left) that can be used to calculate monthly egress rates.
https://cloud.google.com/products/calculator

9/2/2019

Snapshots takes a full disk backup for the first iteration and just the deltas for any iteration afterwards. If a new snapshot is requested and the data has not changed, the new snapshot will be just a pointer to the last previous snapshot. So the amount of storage snapshots use is a factor of disk utilization, frequency of snapshot runs, and frequency of data changes.

Snapshots are also automatically compressed. The compression ratio will depend on the data. In my very rough test, I saw a 3.4GB disk got compressed to 2.33GB.

Since snapshots are based on disks. You can organize your disk accordingly for your snapshot requirements. For example, you can have a separate disk for non-essential items like swap partitions, pagefiles, cache files, and non-critical logs that will not be part of a snapshot schedule.

Check out this page for some additional best practices for snapshots: https://cloud.google.com/compute/docs/disks/snapshot-best-practices

9/9/2019

20190909 - UW Office Hours Follow up

1. What is Kubernetes and Google Kubernetes Engine (GKE)?

a. Kubernetes (K8s) is an open-source container orchestration platform for automating deployment, scaling, and management of containerized applications. Google Kubernetes Engine (GKE) is a managed Kubernetes service that eliminates the need to install, manage, and operate your own Kubernetes clusters. Some features of GKE are:

i. Health checks to allow for auto-healing
ii. Auto-scale: Scale your application deployment up and down based on resource utilization (CPU, memory)
iii. Auto-update: keep your cluster up to date with the latest release version of Kubernetes
iv. Runs on Container-Optimized OS, a hardened OS built and managed by Google

b. Some good GCP Youtube videos on containers and GKE are:

. Getting Started with Containers and Google Kubernetes Engine (Cloud Next '18)
i. Deploy Your Next Application to Google Kubernetes Engine (Cloud Next '19)

c. Some more common alternatives to Kubernetes:

.Docker Swarm - https://docs.docker.com/engine/swarm/
i. Nomad - https://www.nomadproject.io
ii. Amazon ECS - https://aws.amazon.com/ecs/

d. Kubernetes is one of the largest and most popular open source projects today. In 2018, Github ranked it as the 8th project with the most contributors: https://octoverse.github.com/projects. It is used and available on many vendors like:

.Amazon - https://aws.amazon.com/kubernetes/
i. Microsoft - https://azure.microsoft.com/en-us/services/kubernetes-service
ii. Docker - https://www.docker.com/products/kubernetes
iii. Redhat - https://developers.redhat.com/topics/kubernetes/

2. What is Infrastructure as Code (IaC) and what are my available options?

a. Infrastructure as Code (IaC) is the process of provisioning and managing computing resources using definition files (yaml, json, etc). In a cloud environment, it allows you to consistently and repeatedly deploy infrastructure. It also allows for immutable infrastructure where instead of modifying components, the components are redeployed each time any change occurs.

b. There are two popular IaC platforms used in GCP: Cloud Deployment Manager and Terraform

i. Cloud Deployment Manager is a GCP specific IaC platform with strong integration with GCP component. Current versions of SchedMD and Fluidnumerics repositories use Cloud Deployment Manager to deploy Slurm.
    1. Documentation for Deployment Manager: https://cloud.google.com/deployment-manager/docs/
ii. Terraform is the most popular multi-platform IaC platform. It is created by HashiCorp and supports all major cloud platform: AWS, GCP, and Azure.
    1. Documentation for Terraform: https://www.terraform.io/intro/index.html

c. Some GCP Youtube videos on IaC:

.Infrastructure as a Code with Deployment Manager (Cloud Next '19)
i. Getting Started: Cloud Deployment Manager

d. Google engineers manage a program called Cloud Foundation Toolkit which contains commonly used templates for both Deployment Manager and Terraform: https://cloud.google.com/foundation-toolkit/

9/12/2019

To fill in some of the questions on billing, I just wanted to share this new Qwiklab Course on GCP billing.

NEW Qwiklab Quest on **Understanding Your GCP Costs** - Learn how to set up a billing account, organize resources, manage billing access permissions, view your invoice, track your GCP costs with Billing reports, analyze your billing data with BigQuery or Google Sheets, and create custom billing dashboards with Data Studio.

9/17/2019

20190917 - UW Office Hours Follow up

1. Google Container Registry

a. Google Container Registry (GCR) is a private docker repository service that allows you to manage and apply access controls to docker images. It also has advanced features such as the ability to scan Docker images for vulnerabilities.

b. GCR only charges for the underlying Cloud Storage and network egress used by your Docker images. For example, if the registry is created and us-west1 and the pull commands come from us-west1, there is no egress costs and a current multi-region bucket storage rate of $0.026 per GB per month. Additional details on pricing: https://cloud.google.com/container-registry/pricing

c. The free tier for Google Cloud Storage also applies to GCR: https://cloud.google.com/storage/pricing#cloud-storage-always-free

d. Documentation on GCR: https://cloud.google.com/container-registry/docs/

e. Helpful Youtube videos:

i. Container management and deployment: from development to production (Google Cloud Next '17)

ii. Uploading Docker Images to Google Container Registry

2. Cloud Build

. Cloud Build is a service that import source code from sources like GitHub, execute a build to your specifications, and produce artifacts such as Docker containers. Cloud build can be useful for certain workloads where a new Docker image needs to be built based on a schedule or trigger.

a. The pricing model is Cloud Build is free for the first 120 builds-minutes per day and then charges for build-minutes after that. I have seen quite a few customers keep their workloads in the free tier. Additional details on pricing: https://cloud.google.com/cloud-build/pricing

b. Blog article on Cloud Build: https://cloud.google.com/blog/products/gcp/google-cloud-container-builder-a-fast-and-flexible-way-to-package-your-software

c. Documentation on Cloud Build: https://cloud.google.com/cloud-build/docs/

d. Helpful Youtube videos:

. Google Container Builder, Part 1 (Cloud Rolling Update)

i. Google Container Builder, Part 2: Multi-Step Builds (Cloud Rolling Update)

3. Google Cloud Storage as a Container Registry

. You might encounter information about leveraging Google Cloud Storage and a docker registry driver to create a private registry to pull docker images. This is now considered legacy and Google recommends leveraging GCR as an improved option. Link to the legacy driver: https://github.com/GoogleCloudPlatform/docker-registry-driver-gcs

9/30/2019

- I re-ran the stress test and we received several failures. Using sacct we looked at one failure (job 5154_82) and identified the associated node to have been tm2-compute01081. Going over to stackdriver, we did not see any obvious errors for this node at the time of the failure.
  - That node appears to have started up successfully, ran for 10 minutes, and then turned off successfully.
  - Would need more context and review output logs?
- When running a trivial job (e.g., a bash script that loops and echo's a line; sleeps; etc) we can't get consistent behavior in running multiple jobs or tasks on a single node. For example, sbatch --output test_buflog_%A_%a.log -array 1-4 --mem 1000M --ntasks-per-node 4 -n 4 ./test_buflog.bash

    - Try set LLN (Least Loaded Nodes) setting to “no” - the slurm controller will try to use as much as possible the resources of a node before allocating jobs to other nodes
    - Possibly try:
        - --ntasks=4
        - --cpus-per-task=1
- There appears to be some sort of buffering associated with sbatch log files.
    - Slurm has a --unbuffered option. Check out: https://stackoverflow.com/questions/25170763/how-to-change-how-frequently-slurm-updates-the-output-file-stdout/25189364
    - If its a print in a python program: https://stackoverflow.com/questions/33178514/how-do-i-save-print-statements-when-running-a-program-in-slurm
    - Known issue srun --u/--unbuffered option adding a carriage character return to my output https://slurm.schedmd.com/faq.html#unbuffered_cr

10/2/2019

I believe I was able to replicate the issue we are encountering where Slurm is scheduling one tasks per node. I was able to schedule 4 tasks per node but only 2 array jobs per node. I have reached out to our internal Slurm SMEs to see if they have any suggestions.

It is sort of hard to summarize what I did to troubleshoot so I documented my thought process in the attached Google doc. Two items to clarify on that might make it easier to understand:

- Sbatch vs. Srun: There might be some confusion that srun is only for interactive use. Sbatch is used to submit a job and srun can be used in the job submission script to create job steps as Slurm calls them. Check out this tutorial where the sbatch script uses srun to parallelize the command hostname.
- Ntasks vs. Array: Ntasks specifies how many times to run a process under a single job with a single output. Array job specifies how many times to run a process using indexable array ids as separate array jobs with separate array outputs.

Troubleshooting steps I took to try to get multiple process to run on a single node.

1. I copied the sleep job provided from the UW team and placed it into the home directory. I set LLN to no and confirmed it with: `scontrol show partitions`

   I ran the following job:

   ```
   sbatch --array 1-4 --ntasks-per-node 4 --ntasks 4 ~/sleep-test.sh
   ```

   I saw 4 nodes created and an output of each node with the first 2 outputs immediately available and the next two appearing later. Each output ran the bash script once. I believe this is what is similar to what the UW team is running into.

2. I thought that specifying the job array and the number of tasks (ntasks) is redundant. I tried running with just the ntasks option:

```
sbatch --ntasks-per-node 4 --ntasks 4 ~/sleep-test.sh
```

I saw only one node created and one output. The output showed that the bash script ran once, not as specified.

3. On reading guides on Slurm on the differences between `sbatch` and `srun`, it appears you typically use sbatch to submit a job and `srun` in the job submission script to create job steps as Slurm calls them. `Srun` is used to launch the processes.

I created a wrapper script to `srun-wrapper.batch` to use `srun` to run the sleep bash script:

```
#!/bin/bash
      srun bash ~/sleep-test.sh
```

This should identify to Slurm as a single process. I then ran the wrapper script using sbatch.

```
sbatch --ntasks-per-node 4 --ntasks 4 ~/srun-wrapper.batch
```

I saw only one node created and one output. The output showed that the bash script ran 4 times as specified. I have also noticed adding `--unbuffered` option onto the srun command provided a faster log output.

4. To confirm that tasks per node is being properly applied, I submitted another job with a greater number of tasks than what is configured per node.

```
sbatch --ntasks-per-node 4 --ntasks 5 ~/srun-wrapper.batch
```

I saw two nodes created and one output. The output showed that the bash script ran 5 times as specified.

5. The --array options allows to single sbatch script multiple times under an index. I ran the following:

```
sbatch --array 1-4 --ntasks-per-node 4 --ntasks 1 ~/srun-wrapper.batch
```

I saw 4 array jobs across 4 separate nodes with 4 outputs. Each output showed that the bash script ran once. The outputs for the array jobs in my testing took longer to appear than a single job with multiple tasks.

6. I then removed all options except the array to see how it allocates the array jobs across nodes. I ran the following:

```
sbatch --array 1-4 ~/srun-wrapper.batch
```

Interestingly, I saw 4 array jobs across 2 nodes with 4 outputs. Each output showed that the bash script ran once.

Since only two array jobs were able to run on a single node, I thought that there may be an issue with requiring physical cores rather than multi-threaded.

I first confirmed that Slurm was able to capture the correct hardware specs of the cluster:

```
sinfo -o '%9P %4c %8z %8X %8Y %8Z'
PARTITION CPUS S:C:T    SOCKETS  CORES    THREADS
debug*    4    1:2:2    1        2        2
```

Then I confirmed there was no core/threads requirements set when the job was running.

```
squeue -o '%.5i %.2t %.4M %.5D %9c %7H %6I %8J'
JOBID ST TIME NODES MIN_CPUS  SOCKETS CORES_ THREADS_
164_1  R 0:09     1 1         *       *      *
164_2  R 0:09     1 1         *       *      *
164_3  R 0:09     1 1         *       *      *
164_4  R 0:09     1 1         *       *      *
```

Then I confirmed post job completion that there was no core/thread requirements set.

```
scontrol show job 158
```

This is where I got stuck. I tried multiple methods to try to force more than 2 array jobs into a single node but none of them work. Some documented methods were:

- LLN set to no
- -N 1 option
- --overcommit option
- --sockets-per-node, --cores-per-socket, --threads-per-core options
- --mem option, to limit the amount of memory on the sbatch and srun job
- --hint=multithread option
- --distribution=block:block option
- In slurm.conf, modified SelectTypeParameters from CR_Core_Memory to CR_CPU_Memory

10/3/2019

Yup, I also ran into the Stackoverflow thread and towards the end of the comment it says, "you typically use sbatch to submit a job and srun in the submission script to create job steps as Slurm calls them." My confusion is why is this typical... why not run the bash script directly with sbatch and not use srun at all. There is a sub-comment answer on this which states "[Sbatch with

or without srun will be the same except] (1) the allocation is for one CPU and (2) the program is purely sequential. To see differences, request more than one task. Another difference is that if you do not use srun in sbatch, the sstat command will not return any useful information."

Those comments still confuse me and I still don't have a clear understanding of this. But in my testing Slurm will only parallelize using the ntask option with the commands wrapped in srun (see my troubleshooting notes). But as to why ¯_(ツ)_/¯

So troubleshooting with our internal Slurm SMEs and the greater discussion group, we are able to identify the issue in the slurm.conf. The original slurm.conf had:

```
NodeName=DEFAULT Sockets=1 CoresPerSocket=2 ThreadsPerCore=2
RealMemory=25444 State=UNKNOWN
```

Can be changed it to:

```
NodeName=DEFAULT CPUS=4 RealMemory=25444 State=UNKNOWN
```

I tested this and now it is able to submit arrays and multiple jobs onto a single node as expected.

Let me know if this does or doesn't work on your side.

Yes, running the bash script directly (no srun) will work on one node.

However, I did notice sbatch options will affect how its scheduled, ntasks-per-node caused it to run on separate nodes. This is a command I got to work under a single node:

sbatch --array 1-4 --ntasks 1 ~/sleep-test.sh

Pushing the array up to 5, caused it to launch an extra node as expected (no overcommit).

10/25/2019

This is quite a big difference and unexpected behaviour. To help identify if there is a performance difference between the two NFS instances or an issue how SLURM handles mounting Filestore, I would recommend the following troubleshooting tests (if not already completed):

1. Is there an analysis job that successfully completes on both clusters? If so, is there a job runtime difference between the two runs?
2. Are we able to performance benchmark the two NFS instances outside of SLURM? We can set up a standalone GCP VM (8+ vCPU), mount the two instances, and use a benchmark utility like fio. This link provides examples on using fio to benchmark a NFS

mount. Note, that these tests can take a bit of time and the parameters can be tuned accordingly.

Below are some additional recommendations for troubleshooting:

- The Slurm controller is the key component that coordinates the resource scheduling of the jobs. I would recommend reviewing the Stackdriver monitoring metrics (CPU, Memory, Network, I/O) during the job run.
- Working around Slurm output may give us more information, but I believe you mentioned that you were unable to pipe directly to a text file. In my testing, I was able to send the echo output of the sleep job to another file without any issue. Below is an example:
    - echo "sleeping for $st seconds ..." >> /home/roeric_google_com/test.out
- There are quite a few options for the sbatch command. Below are some ones that might be worth exploring (but I am definitely not an expert on).
    - --wait, -W : Do not exit until the submitted job terminates.
    - --wait-all-nodes=1 : Do not being execution until all nodes are ready for use. This might be useful if there is any cross dependency on parallel execution.
    - --no-kill : Prevent SLURM from terminating the entire batch job upon one step failure.

Below are some additional questions that may help troubleshooting:

- For analysis of 1 chromosome with Filestore, how many jobs succeeded?
- What are the specs of the FileStore instance (Premium?, Capacity?)
- What are the specs of the SLURM controller node? Are they the same on both clusters?

Lastly, I think the "no output" issue is a major pain point because it is preventing us from understanding the job behaviour. I think we should reach out the google-cloud-slurm-discuss to get any recommendations on resolving this issue.

11/4/2019

The customer (cc'ed on this email) has an initial dataset of 23 files (# of chromosomes) which is about 380TB total stored on the NFS server. The dataset is submitted to an analysis pipeline that uses the Slurm HPC scheduler to execute a series of jobs across hundreds of GCE compute nodes. Each job stage only writes less than 200MB of result output so the overall process is read heavy.

The customer is not able to achieve the same results when using Filestore compared to their own built GCE NFS server. Their analysis pipeline had some failed jobs on a single chromosome when the working directory and logging directory was the same Filestore instance. When the logging directory was modified to a mount outside of Filestore, they were able to obtain an error: `couldn't chdir to `/filestore1_data/projects/topmed/benchmarks/fr6/assoc_smmat/sparse_km': No such file or directory.` The same pipeline using their own GCE

NFS server was able to process this pipeline without any issues. The compute nodes are in the same zone as NFS server/Filestore but are on different VPC networks.

The specs of their Filestore instance are:
Name: filestore1
Tier: Premium
Capacity: 3TB
Network: default
Location: us-west1-a
Project: topmed-dcc-xpn-host
NFS Version: 3

The specs of their GCE NFS server are:
Instance Type: n1-standard-32
Disk: 100GB PD SSD (boot), 3TB PD SSD (data) (Customer understands max throughput is achieved at 4TB)
Network: default
Location: us-west1-a
NFS Daemons: 514
NFS Version: 4

During job execution, we did not see any alarming metrics or graph bottleneck plateaus in the Stackeriver for Filestore. Below are some peak metrics during execution:
Read Bytes: 44MiB/s
Write Bytes: 390 KiB/s
Read Operation Count: 215.5 /s
Write Operation Count: 10 /s
Procedure Calls: 732.8 /s
Read Operations Time: 2.2 ms

One difference that I can think that might be important is that their NFS server is mounted with NFSv4 while Filestore is mounted and current supports NFSv3.

11/4/2019

Good point, correction on that, I believe they are using a shared VPC. The Filestore instance is created in the host shared VPC project and the compute nodes are configured on the service projects. are

However, I have on my notes that they have Filestore on the default network. I will confirm with the customer if they have configured this as part of the Shared VPC (and configured correctly) or its a typo on my end.

On Mon, Nov 4, 2019 at 7:06 PM Katie Ewing <katiee@google.com> wrote:

Can you clarify what you mean by "The compute nodes are in the same zone as NFS server/Filestore but are on different VPC networks."? The Filestore instance must be on the same VPC network as the compute instance mounting it.

- Yup, definitely meant 380GB rather than 380TB.
- Yeah, in my head, I think I wanted to say the Filestore instance and the compute nodes are on separate projects rather than VPCs. However, I wrote down the Filestore is using the default VPC network. This is possible as part of the shared VPC network but a not typical configuration because most users would want to customize their subnets. This is the configuration I want to review later today.

11/5/2019

This is in regards to running multiple jobs on a single node. Below are my findings:

- Modifying the slurm.conf will require a restart of the Slurm daemon to make changes take effect immediately: `sudo systemctl restart slurmctld`
- In the slurm.conf, the SelectTypeParameters parameter will control how resources are scheduled. Below are the configurations I tested:
    - CR_CPU_Memory - will schedule resources based on CPU and Memory. In my testing, the memory configuration will need to be passed in the sbatch command when this configuration value is supplied, e.g. `sbatch --array 1-8 --ntasks 1 --mem 3000 /home/roeric_google_com/sleep.sh`
    - CR_CPU - will schedule resources based on only CPU, e.g. `sbatch --array 1-8 --ntasks 1 /home/roeric_google_com/sleep.sh`
- In the slurm.conf, the LLN parameter will affect how the job allocates resources. For example, for a size 8 job array, LLN set to no was able to assign all jobs to a single n1-standard-8 node. While LLN set to yes, spun up 4 x n1-standard-8 nodes with 2 jobs on each.

11/22/2019

- How does Stackdriver set timezones for metrics and logs?
    - In Stackdriver logs, the time zone is a selectable field when you select a custom time range. In Stackdriver Monitoring, the time zone is a configured field of the logged in user. The default setting for this is local time. This can be changed by selecting the user in the top right corner, selecting profile and editing the timezone field.
- Why does Slurm default to /tmp when it is unable to find the directory?
    - I have confirmed from Wyatt that this is default Slurm behavior. It defaults to this to try workaround unassessable directory and try to operate at some level. Wyatt mentioned in future releases they plan on making this configurable.
- Why didn't Slurm account for the project.mount configuration in the modified startup script?

- Still in discussions with Wyatt on this. My hypothesis is due to how systemd handles the mount. In my testing, I never see projects.mount, apps.mount or home.mount generated under /usr/lib/systemd/system/ even though they are in the /etc/fstab file and see it successfully mounted. I do see tmp.mount, munge.service, & network.target.

12/10/2019

Below are some follow up resources to yesterday's call.

- Below is the documentation for the persistent disk (PD) and persistent disk SSD (PD SSD) pricing. As of today, PD is $0.040 per GB per month and PD SSD is $0.170 per GB per month. https://cloud.google.com/compute/disks-image-pricing#persistentdisk
- Below is the documentation outlining the performance of available block storage (PD, PD SDD, Local SSD, etc) available. https://cloud.google.com/compute/docs/disks/performance#top_of_page
- In general, larger volumes means better performance. PD and PD SSD can reach the same max read throughput at 1.2GB (at different volume sizes) but PD SSDs have significantly higher max IOPs (~100k vs 7.5K Read <=16KB).

12/2/2019

I have seen updating disk notifications when you modify the disk such as resizing the disk. To verify the activities being completed on the disk, we can select the bell notification icon on the top right > select See All Activities and review the activities that occurred during that time period. If the update was due to resizing, you will see an activity called Resize Disk.

1/6/2020

1. We are building a procedure to provision accounts to GCP pipeline users. What are the minimum IAM permissions are necessary to for a user account to login, move data into/out of the project, and submit jobs? "Compute Network User" seems to be insufficient. We have often given ourselves owner permission as a way to avoid the granular permissions offered on GCP. Is there a GCP best practices document somewhere we can build from?

Compute Network User

Compute Viewer

Compute OS Login

Logs Viewer

Monitoring Viewer

GCS (can be applied at bucket level)

2. We are about to test Preemptible VMs. Can we go over your understanding of how these work in the context of Slurm? Some questions we have are whether a whole job (and how that is defined) will restart or not? Is there a "retry option" and a max retry if a compute node gets pre-empted? Note we do not currently have checkpointing in our code.

If there were any batch jobs on the preempted node, they will be requeued -- interactive (e.g. srun, salloc) jobs can't be requeued

scontrol show job

3. We use OSLogin and would like to be able to pass a user's UID and GID to the Docker image for creating files. We had to use sudo when Docker was running as root and have created a "temporary" user "topmed" for Docker to run as so we don't sudo, but would still like to provide the user's actual in house UID and GID.

Do you have access to admin.google.com

Google Cloud Directory Sync (GCDS)

Need to follow up to see if there is a way to manually edit

4. Re disk notifications, we are not resizing the disk and so suspect it's an internal GCP maintenance operations. If so, is there a way can we see a history of these events with date and time?

5. Google container repository (gcr) is just a bucket and its contents (e.g., docker images) in Google storage are associated with a project. When the first docker image is pushed to a gcr, the bucket and subsequent objects (like folders and files) are created within Google storage. How can the gcr be created from outside the project (e.g., home computer; vm in another project, etc)? Ca other projects be granted permission to create buckets in another project's storage?

One repo per project. Need to first push a image to create bucket. https://cloud.google.com/container-registry/docs/access-control

https://cloud.google.com/container-registry/docs/advanced-authentication#gcloud_as_a_docker_credential_helper

6. We have tested a cluster using HDD for our NFS server and see performance similar to what we get using SSDs. Roy's performance graphs show us getting 2GB/sec max reads?. How does this square with what I thought was a 1.2GB/sec max read performance?

Is this reading from NFS or local persistent disk?

Nodes with 8 vCPU+ have 16 Gbit/s is equal to 2 GB/s. Nodes are most likely reading from memory for max throughput.

1/14/2020

VM preemption is logged in the system audit logs. We can navigate to these logs through the hamburger menu (on top right) > Stackdriver: Logging > Log Viewer. We can then filter based on: GCE VM Instances and the log: cloudaudit.googleapis.com/system_audit or we can put in an advanced query such as the one below:

```
resource.type="gce_instance"
logName="projects/[PROJECT_ID]/logs/cloudaudit.googleapis.com%2F
system_event"
protoPayload.methodName="compute.instances.preempted"
```

The best way to narrow it down by workload is to filter to the correct time period and labels attached to the VMs. System audit logs are enabled by default and have a default retention period of 400 days. Stackdriver logging has a command line interface: https://cloud.google.com/logging/docs/reference/tools/gcloud-logging and REST API: https://cloud.google.com/logging/docs/reference/v2/rest/

For billing, preemptible VMs will be charged with a different SKU than regular VMs. Like regular VMs, Preemptible VMs are bill by the total amount of CPU cores and RAM rather than individual instances. For preemptible VMs in West region, billing will name it as **Preemptible N1 Predefined Instance Core running in Americas** with SKU: **D498-1ECA-87C1** and **Preemptible N1 Predefined Instance Ram running in Americas** with SKU: **5451-0A15-0123.** Similar to logging, my opinion is one way to narrow this down by workload/job is to filter based on time period and labels on the VM (applied during startup or onto existing resources: https://cloud.google.com/compute/docs/labeling-resources#adding_or_updating_labels_to_existing_resources)

Below is a list of all the preemptible VM SKUs for reference: https://cloud.google.com/skus/?currency=USD&filter=Preemptible+N1+Predefined+Instance

1/20/2020

Sorry for the delay in following up on this. Below is an example gcloud logging command to return preemptible VM events from the system events log:

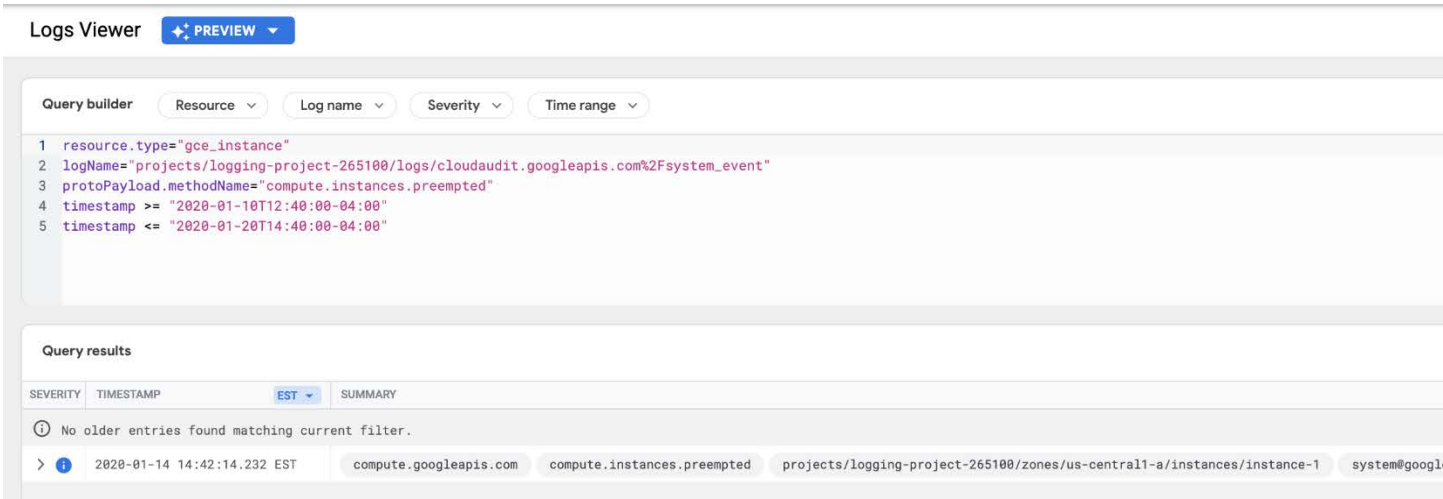
```
gcloud logging read 'resource.type="gce_instance" AND
logName="projects/logging-project-
265100/logs/cloudaudit.googleapis.com%2Fsystem_event" AND
protoPayload.methodName="compute.instances.preempted" AND
timestamp >= "2020-01-10T12:40:00-04:00" AND timestamp <= "2020-
01-20T14:40:00-04:00"' --format json
```

This example looks for preempted instance events in the system event logs between 2020-01-10T12:40:00-04:00 and 2020-01-20T14:40:00-04:00 and returns the results in JSON format. This is an example result:


```
[
  {
    "insertId": "-ksqp9kd2zey",
    "logName": "projects/logging-project-
265100/logs/cloudaudit.googleapis.com%2Fsystem_event",
    "operation": {
      "first": true,
      "id": "systemevent-1579030930194-59c1ec9aeff9d-7bf5fff7-
d005d9a8",
      "last": true,
      "producer": "compute.instances.preempted"
    },
    "protoPayload": {
      "@type":
"type.googleapis.com/google.cloud.audit.AuditLog",
      "authenticationInfo": {
        "principalEmail": "system@google.com"
      },
      "methodName": "compute.instances.preempted",
      "request": {
        "@type":
"type.googleapis.com/compute.instances.preempted"
      },
      "requestMetadata": {
        "destinationAttributes": {},
        "requestAttributes": {}
      },
      "resourceName": "projects/logging-project-265100/zones/us-
central1-a/instances/instance-1",
      "serviceName": "compute.googleapis.com",
      "status": {}
    },
    "receiveTimestamp": "2020-01-14T19:42:14.941892637Z",
    "resource": {
      "labels": {
        "instance_id": "2350320567408895912",
        "project_id": "logging-project-265100",
        "zone": "us-central1-a"
      },
      "type": "gce_instance"
    },
    "severity": "INFO",
    "timestamp": "2020-01-14T19:42:14.232Z"
```

```
  }
]
```

This example results show that there was one preemptible event that occured on 2020-01-14T19:42:14.232Z. Additional documentation on advanced query log filters can be found here. In my opinion, the CLI returns too much information for interactive analysis and the UI is easier to view all events. Below is an example of the same query and results in the UI:



Labels can be added and removed using the gcloud CLI. Below is an example to add a pair of labels. Additional documentation can be found here.

```
gcloud compute instances add-labels sample-instance --
labels=env=prod,job=etl --zone=us-central1-a
```

Below is an example to remove all labels from a VM. Additional documentation can be found here.

```
gcloud compute instances remove-labels sample-instance --all --
zone=us-central1-a
```

As you have mentioned, labels are pretty static. They can only be applied at the VM resource level and not the job/analysis level. Labels can be added to VM instances as part of the job/analysis execution. However, for workloads that can have multiple jobs run at the same time on the same VM and multiple labels can be applied, it becomes much tricker to extrapolate the billing for each job and some logic will need to be performed by the user end (labels on VMs, size of VMs, job vCPUs and Mem).

1/20/2020

Thanks Robert for submitting this support case. On my end, the engineering documents that I have visibility to does not explain the logic for the uid assignment range.. just what we already

know it has to be between 1001 and 65000 or 65535 and 2147483647. I see that the support case has escalated to a SME and hopefully they can provide some additional insight to this.

One additional thought, I don't believe the directory API will be managed by NIH. Since we are using the uw.edu accounts, the directory api should be referencing the admin console controlling the uw domain. This may be the university GSuite administrators. I think it might be worthwhile to see if UW admins are explicitly setting the uids for the user accounts.

1/14/2020

To add on to the labeling recommendation, an added benefit of doing so is that you can compare costs via labels through the billing dashboard (assuming you have access to view the billing account used for your project).

This should work pretty well assuming the VM instances have a 1:1 mapping to a particular job. If you are running multiple concurrent jobs on the same VM's, you can apply multiple labels to a VM, but breaking the costs down between the two jobs could potentially be more difficult.

2/4/2020

Dev branch for GCP slurm: https://github.com/SchedMD/slurm-gcp/tree/dev

Docker documentation states:
"Note: if you pass a numeric uid, it must be in the range of 0-2147483647"
However, from within docker (e.g., via the docker run command) I can create a user whose uid is larger than this range (e.g., 2147483800). I can then run a command in the container and then specify this large uid (2147483800). But I still cannot run a command with a uid outside of the documented range (unless there is an existing account within the docker container)

Can you verify that if I build the guest-login within the docker image (and does not have specific osglogin accounts), that I should be able to specify a uid larger than the documented range?

"You can install the OS Login Linux Guest Environment on a container, but that will not make the docker --user command work with UIDs greater than 2^31 (even if the user exists in the container). That said, installing OS Login Guest Environment would make the users exist inside of the container so you can log in as root and run `sudo su - username` to switch to that user."

2/5/2020

I agree I believe it is designed for a regular VM and not a container. What base ubuntu container image are you using?

Based on those logs it looks like it got to the end of the installation though, can you try committing the changes and then running the container to see what happens?

RE: The question if this has to run on a gcp instance, do you mean running this installation/container building process? If so, I honestly don't know, I will ask.

I tried the official install instructions but I think it's really designed for a google instance. On my docker image based on ubuntu 18.04, I had to first install add-apt-repository (since it's not a regular package with ubuntu lts 18.04; but it's probably part of google's ubuntu) which was no big deal. Any, at the end of installing the guest environment package I received the following msg/error:
Processing triggers for systemd (237-3ubuntu10.33) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for dbus (1.12.2-1ubuntu1.1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for rsyslog (8.32.0-1ubuntu4) ...
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of try-restart.
I don't think you can restart ubuntu within the container; I can restart the container but I think I might have to commit the changes to container before restarting; but I'm not certain.

Also, if this is to work in the docker image, the docker image have to be running on a gcp instance? And this doesn't seem to integrate well with programmatically building our docker images (via docker build files).

I have some more official installation instructions for you that should hopefully be a bit simpler: https://cloud.google.com/compute/docs/images/install-guest-environment#in_place

I cloned the guest-oslogin github repository and built the debian package via packaging/build_deb.sh within my docker image. There were a couple of minor problems in build_deb.sh which were easily fixed:

1. apt-get update needs to be executed before installing dependencies
2. the version of debian in my docker image is "buster/sid"; the later sed statement of build_deb.sh (line 53) failed (probably due to the "/" in debian version). I hard-coded the debian version to be "buster" in build_deb.sh which fixed that problem.

Everything seemed to build fine but at the end I received these warnings and errors when lintian ran at the end. If you (or engineering) could help addressing the not so obvious ones (e.g., the errors) that would be great.
dpkg-buildpackage: info: full upload (original source is included)
Now running lintian google-compute-engine-oslogin_20191018.00-g1+debbuster_amd64.changes ...
warning: the authors of lintian do not recommend running it with root privileges!
E: google-compute-engine-oslogin changes: bad-distribution-in-changes-file stable
E: google-compute-engine-oslogin source: missing-build-dependency-for-dh-addon systemd => debhelper (>= 9.20160709~) || dh-systemd
W: google-compute-engine-oslogin: package-name-doesnt-match-sonames libnss-cache-oslogin2 libnss-oslogin2
W: google-compute-engine-oslogin: binary-without-manpage usr/bin/google_authorized_keys
W: google-compute-engine-oslogin: binary-without-manpage usr/bin/google_oslogin_control

W: google-compute-engine-oslogin: binary-without-manpage usr/bin/google_oslogin_nss_cache
Finished running lintian.

I ran docker execute in the above container (i.e., with the building of guest_login) but I couldn't switch to my user --
topmed@95202e527e8a:/$ sudo su - xxxuser
No passwd entry for user 'xxxuser'
But that might have been due to the error. And I'm still not clear how docker will have knowledge of my oslogin user name; but hopefully you can get this cleared up.

2/10/2020

Labels can be added either at instance creation time, or while they are running. This is done manually with the --labels flag during the `gcloud compute instances create` command. The challenge here is that slurm is creating the instances for you.

It looks like labels can be added in the https://github.com/SchedMD/slurm-gcp/blob/master/slurm-cluster.yaml config file. Not sure about the fluid numerics version yet. To be more granular and set labels when a job is run, you may have to add the appropriate gcloud command within your custom scripts and set the proper permissions for the service account. Again, happy to chat more on this today.

As we talked about in a previous meeting, we would like programmatically add a label to our compute instances in slurm as we run an analysis (e.g., to track costs). Is there a way to have the launched instance have a label that was previously created (e.g., on the boot disk/image of the instance)? I couldn't get this to work by creating a label on the boot image) Is the only way for a launched instance to have a label, is to create the label after it's launched?

2/10/2020

Some code to potentially make os-login work on an Ubuntu Container:

```
mkdir /etc/sudoers.d/ # OS Login relies on this directory for sudo
permission management.
apt update
apt install -y google-compute-engine-oslogin
/usr/bin/google_oslogin_control activate # Enable OS Login since the
accounts daemon is not installed
/usr/bin/google_oslogin_nss_cache # Create a local cache of users so `getent
passwd` works.
```

I just took a first look at the SchedMD/slurm-gcp dev branch. Unfortunately I found it effectively unusable in its current state. Not only does any attempt at a multi-partition deployment fail, but even the simple deployment example using all of the provided default values fails, reporting a slurm network/subnet error:

$ gcloud deployment-manager deployments create topmed-dcc-g1-cluster --config slurm-cluster.yaml
The fingerprint of the deployment is -GhAR_MOmumENZCfnFXPjQ==
Waiting for create [operation-1581111438783-59e0331a8b301-f1017ba2-924a2c7f]...failed.
ERROR: (gcloud.deployment-manager.deployments.create) Error in Operation [operation-1581111438783-59e0331a8b301-f1017ba2-924a2c7f]: errors:
- code: CONDITION_NOT_MET
  message: Referenced resource g1-slurm-network could not be found. At resource b'g1-slurm-subnet'.

One thing I noticed is that scripts included with deployment differ significantly between branches:

$ diff -q slurm-gcp-master/scripts slurm-gcp-dev/scripts
Files slurm-gcp-master/scripts/resume.py and slurm-gcp-dev/scripts/resume.py differ
Only in slurm-gcp-dev/scripts: setup.py
Only in slurm-gcp-master/scripts: slurm-gcp-sync.py
Only in slurm-gcp-dev/scripts: slurmsync.py
Only in slurm-gcp-master/scripts: startup-script.py
Only in slurm-gcp-dev/scripts: startup.sh
Files slurm-gcp-master/scripts/suspend.py and slurm-gcp-dev/scripts/suspend.py differ
Only in slurm-gcp-dev/scripts: util.py

2/10/2020

If you are using default service accounts or default project editor roles you should have the IAM permissions necessary. You also need to pass in the custom header though: Metadata-Flavor:Google, it looks like that is the error that you ran into.

It should look something like this:

```
curl -s "http://metadata.google.internal/computeMetadata/v1/${ATTRIBUTE}" -H "Metadata-Flavor: Google"

Where ${ATTRIBUTE} is a variable for the rest of the metadata path.
```

I'll send a followup later tonight or tomorrow on how to check to make sure you have the right permissions.

I tried to get some metadata. The doc states that I need the following permission to get metadata:

- compute.projects.get
- compute.instances.get

I don't know where to check if I have these permissions but I received an error (looks to be permission related) when attempting to get info:

_4172$ curl http://metadata.google.internal/computeMetadata/v1/project/project-id
<!DOCTYPE html>

```
<html lang=en>
  <meta charset=utf-8>
  <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
  <title>Error 403 (Forbidden)!!1</title>
  <style>
    *{margin:0;padding:0}html,code{font:15px/22px arial,sans-
serif}html{background:#fff;color:#222;padding:15px}body{margin:7% auto 0;max-width:390px;min-
height:180px;padding:30px 0 15px}* > body{background:url(//www.google.com/images/errors/robot.png)
100% 5px no-repeat;padding-right:205px}p{margin:11px 0 22px;overflow:hidden}ins{color:#777;text-
decoration:none}a img{border:0}@media screen and (max-width:772px){body{background:none;margin-
top:0;max-width:none;padding-
right:0}}#logo{background:url(//www.google.com/images/branding/googlelogo/1x/googlelogo_color_150x54
dp.png) no-repeat;margin-left:-5px}@media only screen and (min-
resolution:192dpi){#logo{background:url(//www.google.com/images/branding/googlelogo/2x/googlelogo_col
or_150x54dp.png) no-repeat 0% 0%/100% 100%;-moz-border-
image:url(//www.google.com/images/branding/googlelogo/2x/googlelogo_color_150x54dp.png) 0}}@media
only screen and (-webkit-min-device-pixel-
ratio:2){#logo{background:url(//www.google.com/images/branding/googlelogo/2x/googlelogo_color_150x54
dp.png) no-repeat;-webkit-background-size:100% 100%}}#logo{display:inline-
block;height:54px;width:150px}
  </style>
  <a href=//www.google.com/><span id=logo aria-label=Google></span></a>
  <p><b>403.</b> <ins>That’s an error.</ins>
  <p>Your client does not have permission to get URL <code>/computeMetadata/v1/project/project-id</code>
from this server. Missing Metadata-Flavor:Google header. <ins>That’s all we know.</ins>
```

I received a similar msg when executing:
curl http://metadata.google.internal/computeMetadata/v1/project/attributes

Could you describe how to check for these permissions and how to set them? We might need NIH help in case we (UW) do not have sufficient permissions to enable the desired permissions.

Querying Metadata: https://cloud.google.com/compute/docs/storing-retrieving-metadata

Here is an example of how I have queried instance metadata in the past: https://github.com/GoogleCloudPlatform/solutions-compute-cloudsql-pega7/blob/master/pega/env.sh (Sourced from full tutorial here)

2/13/2020

amended group-and-user instructions:

```
# create docker and topmed groups
sudo groupadd -g 994 docker
sudo groupadd -g 2049 topmed

# create topmed user
sudo useradd -u 2049 -g 2049 topmed
```

```
# add topmed user to docker group
sudo usermod -aG docker topmed

# add oslogin user(s) to docker group
sudo usermod -aG docker ext_rmoulton_uw_edu
```

roy kuraisa (uw) wrote on 2/13/20 1:25 PM:
And don't forget to create the Linux group topmed for users and the dummy user topmed. I forgot to mention this.

On Thu, Feb 13, 2020, 1:02 PM Robert Moulton <rmoulton@uw.edu> wrote:

we use the deployment-provided 'custom-*-install' scripts for creation of local groups and docker installation, among other things:

```
$ ls -1 slurm-gcp-master/scripts/custom-co*
slurm-gcp-master/scripts/custom-compute-install
slurm-gcp-master/scripts/custom-controller-install
```

For compute nodes (and login node), for example:

custom-compute-install:

```
-------------------------------------------------
#!/usr/bin/env bash

# update system
sudo yum -y update

# create docker group
sudo groupadd -g 994 docker

# add oslogin user(s) to docker group
sudo usermod -aG docker ext_rmoulton_uw_edu

# install and enable docker
curl -fsSL https://get.docker.com/ | sh
sudo systemctl start docker
sudo systemctl enable docker

# stop and disable yum-cron
sudo systemctl stop yum-cron
sudo systemctl disable yum-cron
-------------------------------------------------
```

custom-controller-install script could be identical in this case. Note that the docker group *must*

be created on the controller as well as on the compute node images. Installation of docker itself isn't strictly necessary on the controller, however.

Here's a quick description of what you need to do in order to run our pipeline on your cluster:

1. check that docker is installed on the compute nodes via srun to a partition and execute a docker command like: *docker images* (Note: docker should be installed and the users should be members in the linux group *docker*)
2. create a folder in the an nfs volume accessible by the compute nodes (e.g., */apps/topmed*)
3. cd to the directory and download the analysis pipeline from git hub: *git clone -b roybranch https://github.com/UW-GAC/analysis_pipeline.git*
4. cd to *analysis_pipeline* and edit *slurm_partitions.json* to reflect the name of your cluster, the names of your partitions; and for each partition change the cores and memory accordingly (there are currently 3 clusters tm1, tm2, and tm3 configured - you can just modify one of them)
5. edit *slurm_cfg.json* as follows:
    - change the *configuration->cluster* from *tm1* to the name of your cluster
    - change *configuration->memory_limits->null_model* from *62000* to *8000*
6. create a working directory in the nfs volume (e.g., */apps/topmed/work*)
7. cd to your work directory and copy the test data from the analysis pipeline, e.g., *cp -R /apps/topmed/testdata ./*
8. from your work directory, copy the null model config file from the testdata: *cp ./testdata/null_model.config ./*

From your working directory, you can now execute the analysis pipeline for the null_model. First execute a print only command to see if things are configured correctly: */apps/topmed/analysis_pipeline/null_model.py null_model.config --cluster_type Slurm_Cluster --print*
See below for an example. You should see the different 3 jobs that will be submitted to slurm. Make sure the correct cluster name is being specified. If this looks good just execute the same command without the *--print* option.

Let me know if you have any questions or encounter any problems. After we get this analysis to run, we can submit a more complicated analysis.

Here's the output of the print option from my cluster (where the analysis pipeline is installed at a different location). You'll see there are only three jobs that will be submitted. If the jobs complete without errors, all of the logs, data, plots are copied to subdirectories in your working directory.

```
+++++++++  Print Only ++++++++++
Analysis: null_model
Analysis log file: analysis_null_model_ext_kuraisa_uw_edu_158162168568.log
null_model start time: Thu, 13 Feb 2020 07:21:25 PM
Slurm cluster: tm1
Slurm submit script: submitAnalysis.sh /pre-analysis: none /resume: False
```

Job: null_model /array: no /cores: 1 /memlim: 8000M /cluster: tm1 /parition: tm1-2-13 /tasks_per_partition: 1/machine: n1-highmem-2 (0.1184/hr for 1 task(s))
sbatch --job-name null_model --mem 8000M --output null_model_%j.log --partition tm1-2-13 /projects/topmed/working_code/analysis_pipeline/submitAnalysis.sh create_label ap_auto_label feb-13-2020-07_21_25pm /projects/topmed/working_code/analysis_pipeline/runDocker.py --dockerimage uwgac/topmed-roybranch:latest --runcmd /usr/local/analysis_pipeline/runRscript.sh --runargs "/usr/local/analysis_pipeline/R/null_model.R config/test_null_model.config --version 2.7.3" --mem_limit 8 --working_dir /projects/topmed/benchmarks/fr8_test/all --machine n1-highmem-2 --cost 0.1184 --stats

Job: null_model_report /array: no /cores: 1 /memlim: 8000M /cluster: tm1 /parition: tm1-2-13 /tasks_per_partition: 1/machine: n1-highmem-2 (0.1184/hr for 1 task(s))
sbatch --job-name null_model_report --dependency afterok:null_model --mem 8000M --output null_model_report_%j.log --partition tm1-2-13 /projects/topmed/working_code/analysis_pipeline/submitAnalysis.sh create_label ap_auto_label feb-13-2020-07_21_25pm /projects/topmed/working_code/analysis_pipeline/runDocker.py --dockerimage uwgac/topmed-roybranch:latest --runcmd /usr/local/analysis_pipeline/runRscript.sh --runargs "/usr/local/analysis_pipeline/R/null_model_report.R config/test_null_model_report.config --version 2.7.3" --mem_limit 8 --working_dir /projects/topmed/benchmarks/fr8_test/all --machine n1-highmem-2 --cost 0.1184 --stats

Job: null_model_post_analysis /array: no /cores: 1 /memlim: 8000M /cluster: tm1 /parition: tm1-2-13 /tasks_per_partition: 1/machine: n1-highmem-2 (0.1184/hr for 1 task(s))
sbatch --job-name null_model_post_analysis --dependency afterok:null_model_report --mem 8000M --output null_model_post_analysis_%j.log --partition tm1-2-13 /projects/topmed/working_code/analysis_pipeline/submitAnalysis.sh create_label ap_auto_label feb-13-2020-07_21_25pm /projects/topmed/working_code/analysis_pipeline/runDocker.py --dockerimage uwgac/topmed-roybranch:latest --runcmd /usr/local/analysis_pipeline/post_analysis.py --runargs "-a null_model -l analysis_null_model_ext_kuraisa_uw_edu_158162168568.log -s Thu,_13_Feb_2020_07:21:25_PM" --mem_limit 8 --working_dir /projects/topmed/benchmarks/fr8_test/all --machine n1-highmem-2 --cost 0.1184 --stats

2/28/2020

Overview

Below are various options for interacting with Google Cloud Storage. For scientific computing, it could be best leveraged by downloading large files at the beginning of a job (or possibly, a subset of files), then uploading any result sets at the end for future analysis or integration into a downstream pipeline that is loosely coupled (ie, a delay of 30 seconds or more between uploading the result sets and a subsequent pipeline running would not be detrimental.)

Throughput for GCS is generally pretty high, however latency is higher than local disk due to a variety of factors, one being the replication process as mentioned in the consistency overview.

GCS Fuse

Here are some links to help get you started with GCS Fuse. Keep in mind that this is an open source project and only supported by the community. More details are available in the overview link below:

- GCS Fuse Overview

- GCS Fuse Installation Instructions
- GCS Fuse Semantics

Basically, GCS will show as a file mount, just as an NFS share would show up. To an OS user or application, it won't look any different, but there are some nuances that you need to make sure you take into consideration that are explained in the various links above. Take a look and let me know if you have any questions.

Usage Example:

```
#Create a directory.

mkdir /path/to/mount

#Create the bucket you wish to mount, if it doesn't already exist, using the Google Cloud Console, gsutil or client libraries.

#Use Cloud Storage FUSE to mount the bucket (e.g. example-bucket).

gcsfuse example-bucket /path/to/mount

#Start working with the mounted bucket.

ls /path/to/mount
```

gsutil

This is the standard command line interface for working with GCS. You can read more in the official documentation. In general, this follows a similar cadence as you would when working with files on a local directory, for example:

```
#Copy to a bucket
gsutil cp file.txt gs://bucket-name/folder/file.txt
#Copy from a bucket to local directory
gsutil cp gs://bucket-name/folder/file.txt .
#List Bucket Contents
gsutil ls gs://bucket-name
```

Check the GCS How To Guides for many more examples and scenarios.

Keep in mind for all of these examples that there is no actual construct of a "folder" within a GCS bucket. The "folder path" is really just part of the filename.

Python Client Library

Here is a launchpoint for the Cloud Storage Client Libraries. Please note, when running on a GCE instance you don't need to worry about the authentication component as it automatically inherits the service account tied to that particular instance. You only need to configure the authentication piece if you want to leverage a different service account, or if you are running outside of GCP.

For in-depth examples, check out the Python sections of the various How To Guides.

Here is one example from those guides on uploading and downloading an object:

```
from google.cloud import storage


def upload_blob(bucket_name, source_file_name, destination_blob_name):
    """Uploads a file to the bucket."""
    # bucket_name = "your-bucket-name"
    # source_file_name = "local/path/to/file"
    # destination_blob_name = "storage-object-name"

    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(destination_blob_name)

    blob.upload_from_filename(source_file_name)

    print(
        "File {} uploaded to {}.".format(
            source_file_name, destination_blob_name
        )
    )

def download_blob(bucket_name, source_blob_name, destination_file_name):
    """Downloads a blob from the bucket."""
    # bucket_name = "your-bucket-name"
    # source_blob_name = "storage-object-name"
    # destination_file_name = "local/path/to/file"

    storage_client = storage.Client()

    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(source_blob_name)
    blob.download_to_filename(destination_file_name)

    print(
        "Blob {} downloaded to {}.".format(
            source_blob_name, destination_file_name
        )
    )
```

3/8/2020

Are you getting any specific error messages? I think even if you use sudo, you can pass the uid and gid flags and it will apply the desired user/group settings. I haven't had a chance to test it on my own yet though.

I can mount as a non-root user provided my UID is < 2**31. However, using projects where we have deployed OS Login, I have to use sudo because (I think) my UID > 2**31. Very similar to docker.

Thanks for the update! It looks like there might be some flags you can apply that would address the root user ownership problem . Here is an fstab example from the mounting.md doc

You can also mount the file system automatically as a non-root user by specifying the options `uid` and/or `gid`:

```
my-bucket /mount/point gcsfuse rw,allow_other,uid=1001,gid=1001
```

Did you happen to try those flags already? If not give it a shot and let me know how it goes.

Here's an update on gcsfuse that we can talk more about in our next meeting.

I've installed and did some very preliminary testing with gcsfuse on two different projects - one project where we have slurm installed, run Centos; and one project without slurm, running Ubuntu.

Also on the slurm project, we're using OS Login; and the Ubuntu project we're not using OS Login.

As an fyi, in order to really use gcsfuse by multiple users and to appear more file system like, the following options were used when mounting a bucket via gcsfuse:

- --implicit-dirs (to be able to ls just "directories")
- -o allow_other (to enable others to access the "files")

The last option also required 'user_allow_other' to be set in /etc/fuse.conf

In the Ubuntu project (without OS Login), all worked as expected.

However in our slurm project (and probably due to a uid > 2**31), I had to sudo mount the bucket. The result was that files were owned by root and all files permission bits were 0644 (and 0755 for directories).

4/2/2020

I did not try the _netdev option yesterday, but just tried it now and that worked as well.

That's a great option -- we could have used that for various issues related to mounting devices (typically network based) that requires the network. Did you by chance also test with the

"_netdev" option (which doesn't depend on systemd)? I found that option after checking out the systemd approach.

fstab was trying to mount the bucket before the network started. Add the following flag to your fstab file: x-systemd.requires=network-online.target

My fstab line looks like this: gcp-testing-197919-test-bucket /fuse gcsfuse rw,x-systemd.requires=network-online.target,allow_other,implicit_dirs

Give that a shot, worked perfectly for me once I added that flag.

I was able to reproduce the issue using the instructions you sent. I also tested waiting a period of time after rebooting before issuing the ls command, and when I did so it worked successfully the first time.

I didn't track the time I waited closely, so more investigation is needed to see exactly how long it needs to fully establish a connection, but wanted to pass this on.

It turns out that the gcsfuse mount issue we described
today -- the i/o error(s) for initial access attempt(s) -- has nothing
to do with slurm. Rather, it seems to be happening for fstab-defined
mounts invariably (on CentOS 7 systems, at least). I reproduced the
problem today on a relatively generic CentOS 7 instance, as described
in the attached text file. Note that this was working nicely for us
until recently. Would you please take a look, to help us figure out
what might be wrong?

4/3/2020

I like using Cloud Shell to run things via command line, as that handles most authentication out of the box for you. You are most likely running into issues with permissions on the service account that is assigned to the VM you are using.

Try either using cloud shell, or when logged into the vm, run "gcloud init" and follow the prompts to enable it under your own user ID.

If that still gives you errors, let me know and we can dig into it more on Monday.

I'm following the doc on "Managing Batch on GKE clusters" (https://cloud.google.com/kubernetes-engine/docs/how-to/batch/managing-clusters). When I attempt to create a cluster (compatible with Batch on GKE) from my project's vm instance, I get an error. Here is the command and error message:

```
_112$ gcloud beta container clusters create tm-roy-cluster --region us-west --node-locations us-west2a --num-nodes 1   --machine-type n1-standard-8   --release-channel regular   --enable-stackdriver-kubernetes   --identity-namespace=topmed-dcc-roy.svc.id.goog   --enable-ip-alias
```

WARNING: *Newly created clusters and node-pools will have node auto-upgrade enabled by default. This can be disabled using the `--no-enable-autoupgrade` flag.*
WARNING: *Starting with version 1.18, clusters will have shielded GKE nodes by default.*
WARNING: *The Pod address range limits the maximum size of the cluster. Please refer to https://cloud.google.com/kubernetes-engine/docs/how-to/flexible-pod-cidr to learn how to optimize IP address allocation.*
This will enable the autorepair feature for nodes. Please see https://cloud.google.com/kubernetes-engine/docs/node-auto-repair for more information on node autorepairs.
ERROR: (gcloud.beta.container.clusters.create) ResponseError: code=403, message=Request had insufficient authentication scopes.

I couldn't find anything that fixed this; although . So, I just tried:

1. Creating a generic kubernetes cluster from the web console
2. Creating a generic kubernetes cluster via the command line on the vm instance

Creating the cluster from the console was successful. On the vm instance, I tried the following with the same results as before:

```
_111$ gcloud container clusters get-credentials cluster-1 --region us-west1 --project topmed-dcc-roy \
> && kubectl edit secret default-token-pv998 --namespace kube-node-lease
Fetching cluster endpoint and auth data.
ERROR: (gcloud.container.clusters.get-credentials) ResponseError: code=403, message=Request had insufficient authentication scopes.
```

I've attempted to configure the service accounts; gcloud config; etc -- but I'm sure I just missed something. It's ok if you want to wait until we meet on Monday and then we can checkout the project, etc.

4/13/2020

Ah, makes sense. I was running as root. Thanks! I think it's working now, still running after a little while.

It's a permission problem in your cwd where you've ran the pipeline. When you run the pipeline (via */apps/topmed/analysis_pipeline/null_model.py null_model.config --cluster_type Slurm_Cluster*), the pipeline code (i.e., null_model.py) creates some config files in the folder <cwd>/config before submitting the various jobs to slurm. I'm not certain where your cwd is but check to see if the folder <cwd/config folder has been created and if it exists you should see several .config files including "test_null_model.config". You cwd should be someplace under /apps (e.g., /apps/topmed/work).

Now, note the permissions of your cwd. After the pipeline creates the config folder and the various config files, it python code submits various jobs to slurm. When the slurm job executes a "docker run " command to execute some R code. However, the user is not root; instead its a user named "topmed" and this docker user is in the linux group 2049.

If you haven't done so you need to create a linux group whose gid is 2049 and your account (where you're running the pipeline) should be in the 2049 group. Your cwd (and all subfolders) should be owned by your uid, the 2049 group and with SGID set. This will insure all

folders being created will have the correct permissions so docker can read and write to these folders/subfolders within your cwd.

I fixed my disk size issues, and jobs are running the container code. Its now throwing this error, and I can't find where the file is being referenced to troubleshoot further. The command to launch was:

/apps/topmed/analysis_pipeline/null_model.py null_model.config --cluster_type Slurm_Cluster

Error in file(file, "rt") : cannot open the connection
Calls: readConfig -> read.table -> file
In addition: Warning message:
In file(file, "rt") :
cannot open file 'config/test_null_model.config': No such file or directory
Execution halted
>>> Error: R status code 1
touch: cannot touch 'fail.17': Permission denied
>>> End job: 17 at Mon Apr 13 18:56:20 PDT 2020
Command exited with non-zero status 1

Thanks! Was working on resizing my disk, adding this in now.

Reelevant post-deployment partition-image update steps
are attached. We also update the slurm config (SelectTypeParameters,
for one thing) at that point, but I don't think that's necessary for
your testing.

4/16/2020

Hope you are doing well. I'm running into some odd issues and figured I'd check with you to see if you were familiar with the root cause.

I realized the users and groups have to exist on the controller, otherise slurm thinks they don't exist. That was a fun lesson to learn.

Unfortunately I'm still hitting permission errors. I got a bit further in that I was able to fix the errors at the slurm level, for example:

srun cat /apps/topmed/work/config/test_null_model.config would have failed earlier, but now returns the file contents properly after I fixed my permissions.

But when the null_model script, it still errors out with the same permission error as before, unable to read that config file.

The permissions on my directory: (/apps/topmed/work)

[jon_jmgtest_com@jmg1-login0 work]$ ll
total 28
-rw-rw-r--. 1 jon_jmgtest_com topmed 2377 Apr 17 03:00
analysis_null_model_jon_jmgtest_com_158709245404.log
drwxrwsr-x. 2 jon_jmgtest_com topmed 73 Apr 17 03:00 config
drwxrwsr-x. 2 jon_jmgtest_com topmed   6 Apr 17 03:00 data
drwxrwsr-x. 2 jon_jmgtest_com topmed   6 Apr 17 03:00 log
-rw-rw-r--. 1 jon_jmgtest_com topmed 6595 Apr 17 03:02 null_model_82.log
-rw-r--r--. 1 jon_jmgtest_com topmed 263 Apr 17 03:00 null_model.config
drwxrwsr-x. 2 jon_jmgtest_com topmed   6 Apr 17 03:00 report
drwxrwsr-x. 2 jon_jmgtest_com topmed 8192 Apr 17 03:00 testdata

5/14/2020

Just following up from our discussion today about creating disk snapshots. Below are a few scripts I used for a previous customer who was doing a similar task. They ultimately wanted to download their data on-prem, but you could use similar steps for archiving:

The steps are basically: Create snapshots > Create images from snapshots > export images

Snapshot overview documentation is available
here: https://cloud.google.com/compute/docs/disks/create-snapshots

You can then delete the images and snapshots once you are happy that the exported images work properly. Please note, there is a public doc that has a few more nuances
here: https://cloud.google.com/compute/docs/images/export-image I did not incorperate all of the tasks in that document, so it is possible the exports created from my sample code might not work properly. Please validate before deleting any critical content.

```
#Create disk snapshots of all disks in a zone

for name in `gcloud compute disks list --format="value(name)"`; do
    `gcloud compute disks snapshot $name --zone=us-east1-b --snapshot-
names=$name-gcp-backup`;
done


#create images from all snapshots

for snapshot in `gcloud compute snapshots list --filter=gcp-backup --
format="value(name)"`; do `nohup gcloud compute images create $snapshot --
source-snapshot=$snapshot --family=gcp-backup >image-create.log 2>&1 &`; done

#create image backups for all images. Export format is not required, default
format is a disk.raw format.
for image in `gcloud compute images list --filter=gcp-backup --
format="value(name)"`; do `gcloud compute images export --image=$image --
async --destination-uri="gs://destination-bucket/path/$image.vmdk" --export-
format=vmdk`; done
```