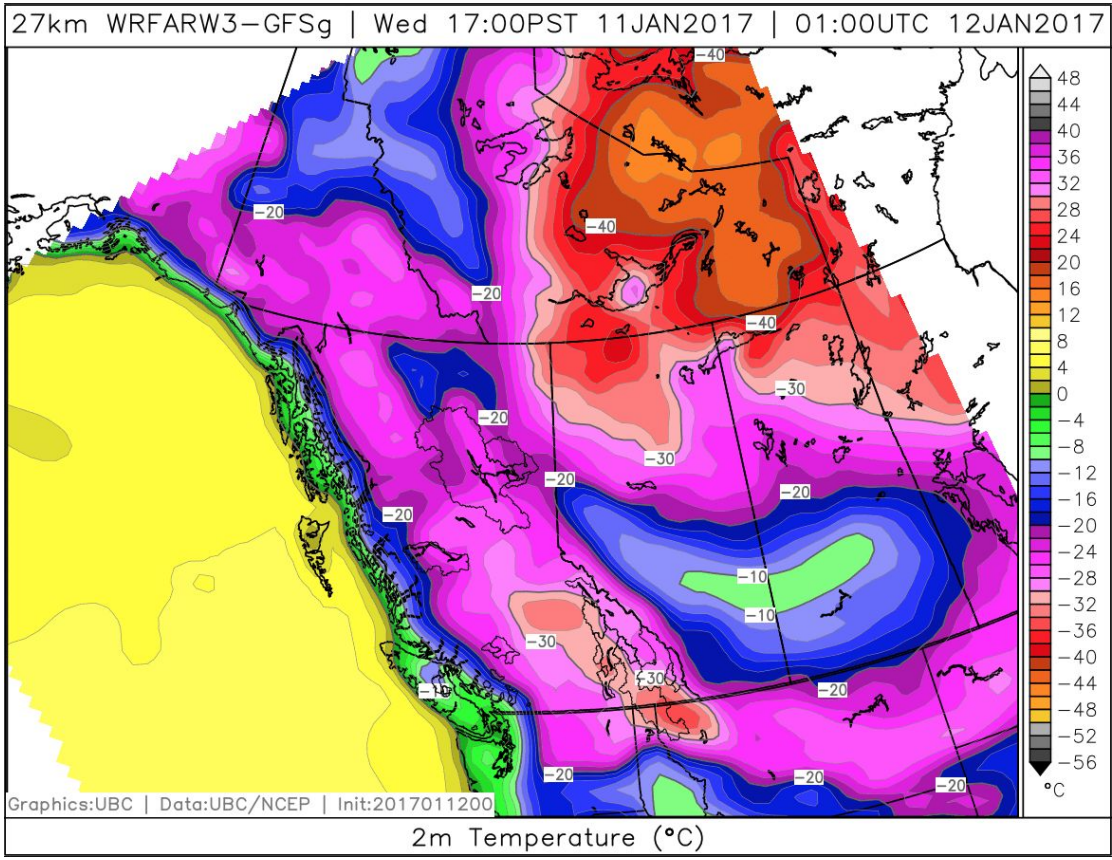
Viability of Cloud Computing for Real-time NWP at the Univ. of BC

David Siuta, Timothy Chui, Roland Stull, Henryk Modzelewski, Greg West, Roland Schigas

Weather Forecast Research Team
University of British Columbia
Vancouver, BC Canada

www.weather.eos.ubc.ca/wxfcst



The Weather Forecast Research Team (WFRT)

University research group producing daily real-time weather forecasts as a by-product of our research in:

- Energy (wind, hydro, solar, biomass)
- Transportation (highways, rail, shipping)
- Weather-related disasters (forest fires, avalanches, floods, and air quality)
- Special projects (2010 Vancouver Olympics)

Photo Credits: David Siuta (above), CBC (top right), and Global News (bottom right)

A paradigm shift for NWP operations

If you do research about operational NWP, then you can**not** have your jobs waiting in an input queue. In the old days, the solution was to buy your own cluster. Therefore, you needed to write a proposal for **BIG bucks up front** to buy the computer. For example, in:

- 1997 – we built our own Beowulf cluster from commodity PCs.
- 2000 – won a $1M grant to purchase an IBM cluster.
- 2010s – won $250k grants to purchase replacement clusters.

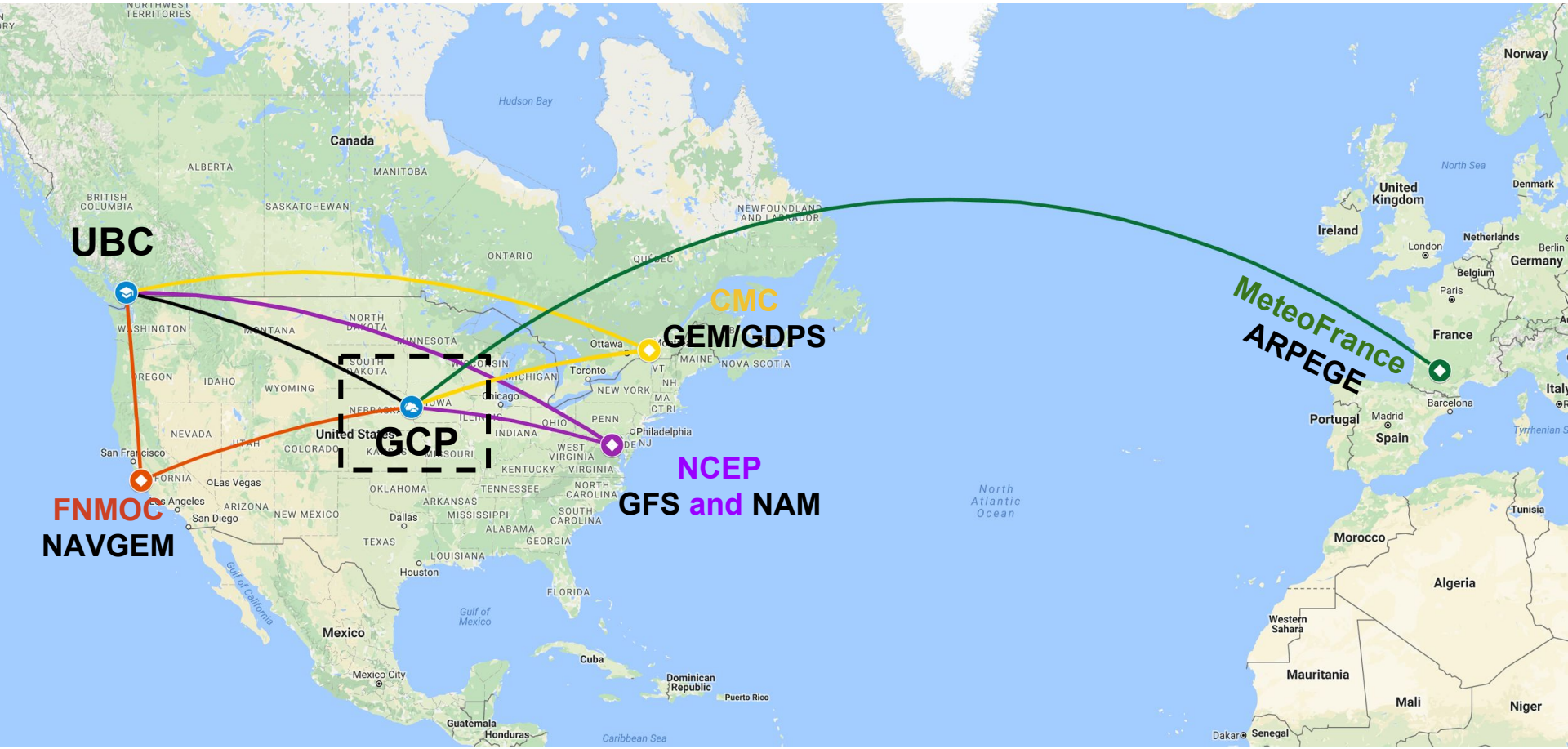
Large time lag between when we wanted and when we got the computer.

- Early 2015 – David Siuta experimented with WRF on Google Cloud.
- Mid 2015 – Siuta did optimization tests to find the best cloud configuration for WRF.
- Oct 2015 – We started daily operational runs of WRF on the cloud. Continues today.

Now, with the cloud:

- We don't need BIG bucks up front.
- Pay as you go - - smaller bucks - - easier to include into contract bids / grant proposals.
- You don't need referee approval to get access to large clusters.
- Don't need to pay for hardware service/maintenance contracts.
- Don't need to hire IT staff to run the computers.
- Reliability approaching 100%. Important for daily operational forecasts.
- Ability to instantly expand to larger/finer/more NWP forecast grids. (little time lag)

UBC modeling system



Multi-model ensemble produced using 5 initial-condition sources from 4 national meteorological centers in 3 dynamical cores: WRF-ARW, WRF-NMM, and MM5.

2 computing resources: 448-core cluster at UBC and Google Cloud Platform (GCP)

Maintain a weather database of 3178 stations for post-processing and verification.

Details of UBC modeling system

As of May 2017	WRF-ARW	MM5	WRF-NMM
Initial-Condition Sources	GFS NAM GEM (GDPS) NAVGEM ARPEGE	GFS NAM	GFS NAM
Max and Min Grid Spacing	108-km to 1.3-km	36-km to 1.3-km	36-km to 4-km
Number of Models per Initialization	00 UTC: 10 06 UTC: 1 12 UTC: 6 18 UTC: 1	00 UTC: 2	00 UTC: 2 12 UTC: 2
Total Ensemble Members (including multiple grids)	00 UTC: 41 over western Canada 12 UTC: 11 over western Canada 12 UTC: 7 over Arctic Canada		
Computing Resources	1. 38-node, 448-core on-premise HPC cluster 2. **Google Cloud Platform virtual-HPC cluster** (Siuta, D., and coauthors, 2016: Viability of cloud-computing for real-time numerical weather prediction, *Weather and Forecasting,* **31, (6)**, 1985-1996, doi:10.1175/WAF-D-16-0075.1).		

Motivation: Aging computing resources

Local High-performance Computing (HPC) cluster aging:

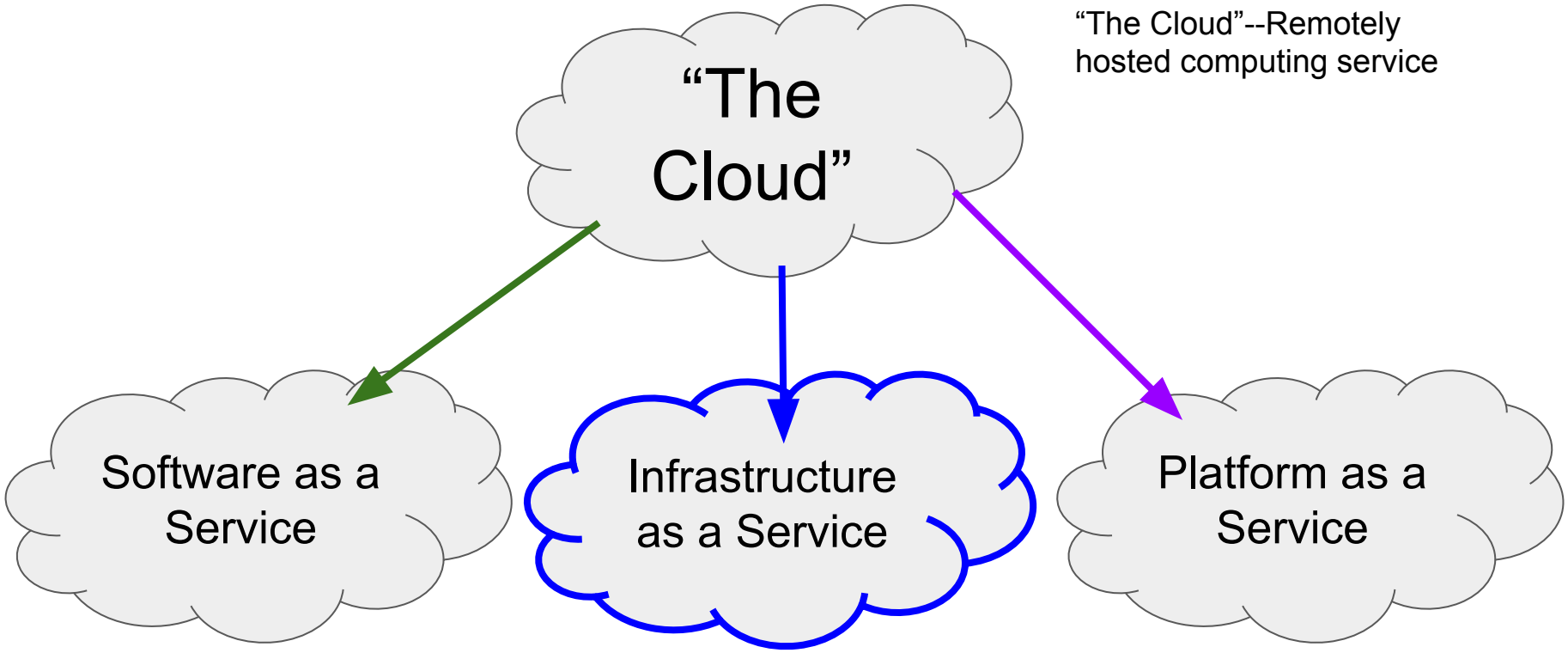
- 38 compute nodes
- 448 total processors

Replacement options:

- Another local cluster
- Cloud-based solution??

Photo Credit: Roland Stull

Cloud Computing Services



"The Cloud"--Remotely hosted computing service

-YouTube
-Netflix
-Google Drive and Docs
-Google Maps
-Microsoft Office Online

-**Google Cloud Platform Compute Engine**
-Amazon Web Services
-Microsoft Azure
-IBM Softlayer
-Smaller ventures

-Google App Engine
-Amazon AWS Beanstalk
-Others

Goal: **Test if the Google Cloud Platform (GCP) is viable for our (UBC) real-time Numerical Weather Prediction (NWP) needs.**

Real-time NWP **must** be:

???

Timely

Cost Effective

Reliable

99.95 % hardware uptime

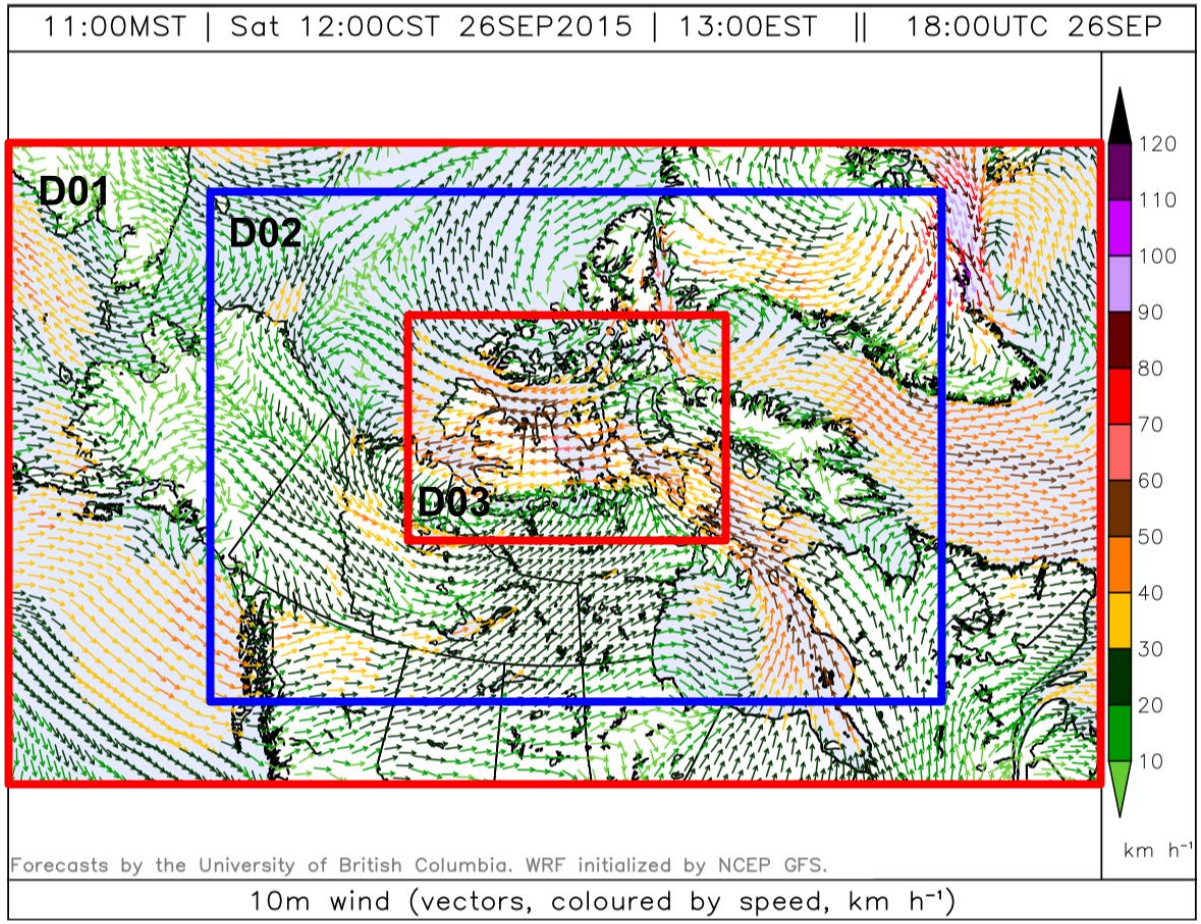
Expandable

Methods

1. Design a virtual HPC cluster on the GCP using several virtual machines (VMs).

2. Find optimization strategy to decrease WRF simulation runtime and cost based on results of a single case study.

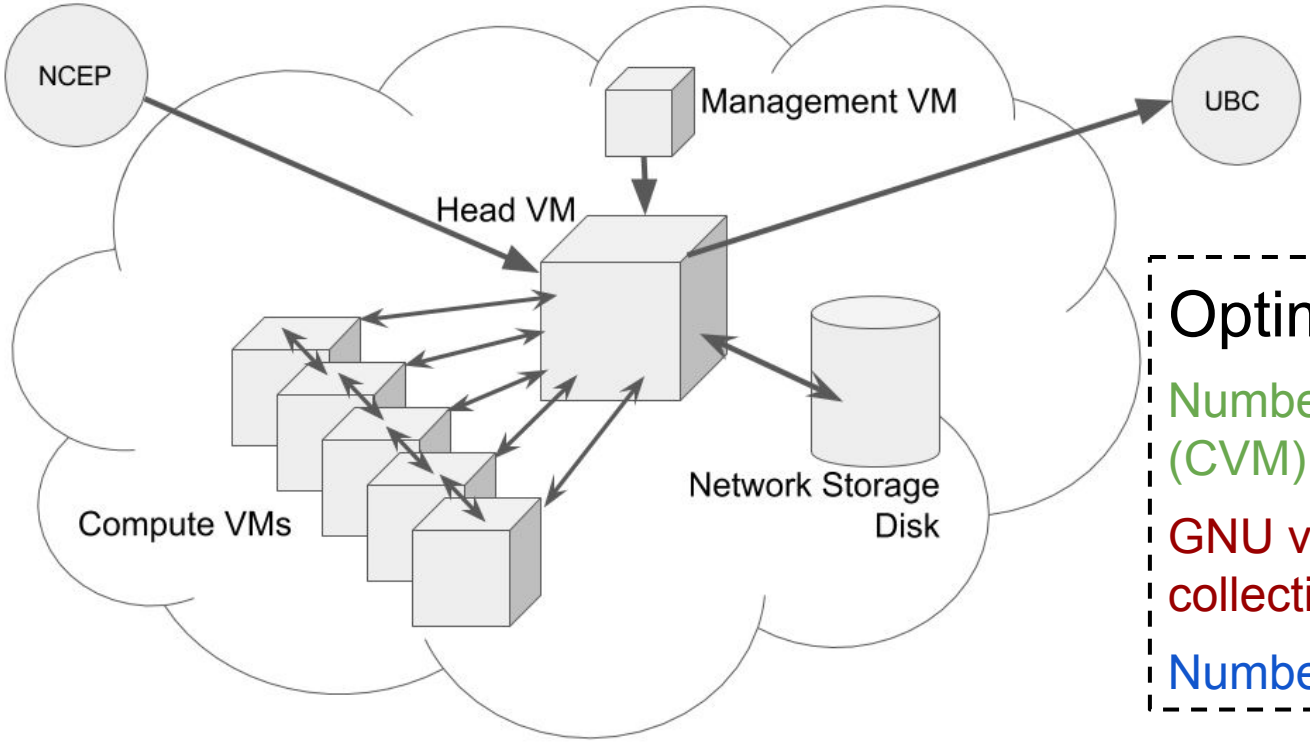
3. Case-study uses our Canadian Arctic/Northwest Passage domain (WRF-Arctic).
   D01 = 200x108,
   D02 = 346x250,
   D03 = 448x307,
   All with 41 levels



Virtual HPC cluster: Aggregate of virtual machines

Virtual Machine (VM): A user-designed, remotely hosted, computing environment.

Operating system, number of virtual CPUs, memory amount, disk space, user-installed libraries, etc. are specified by the user.



Part 1

Optimization tests:

Number of Compute VMs (CVM)

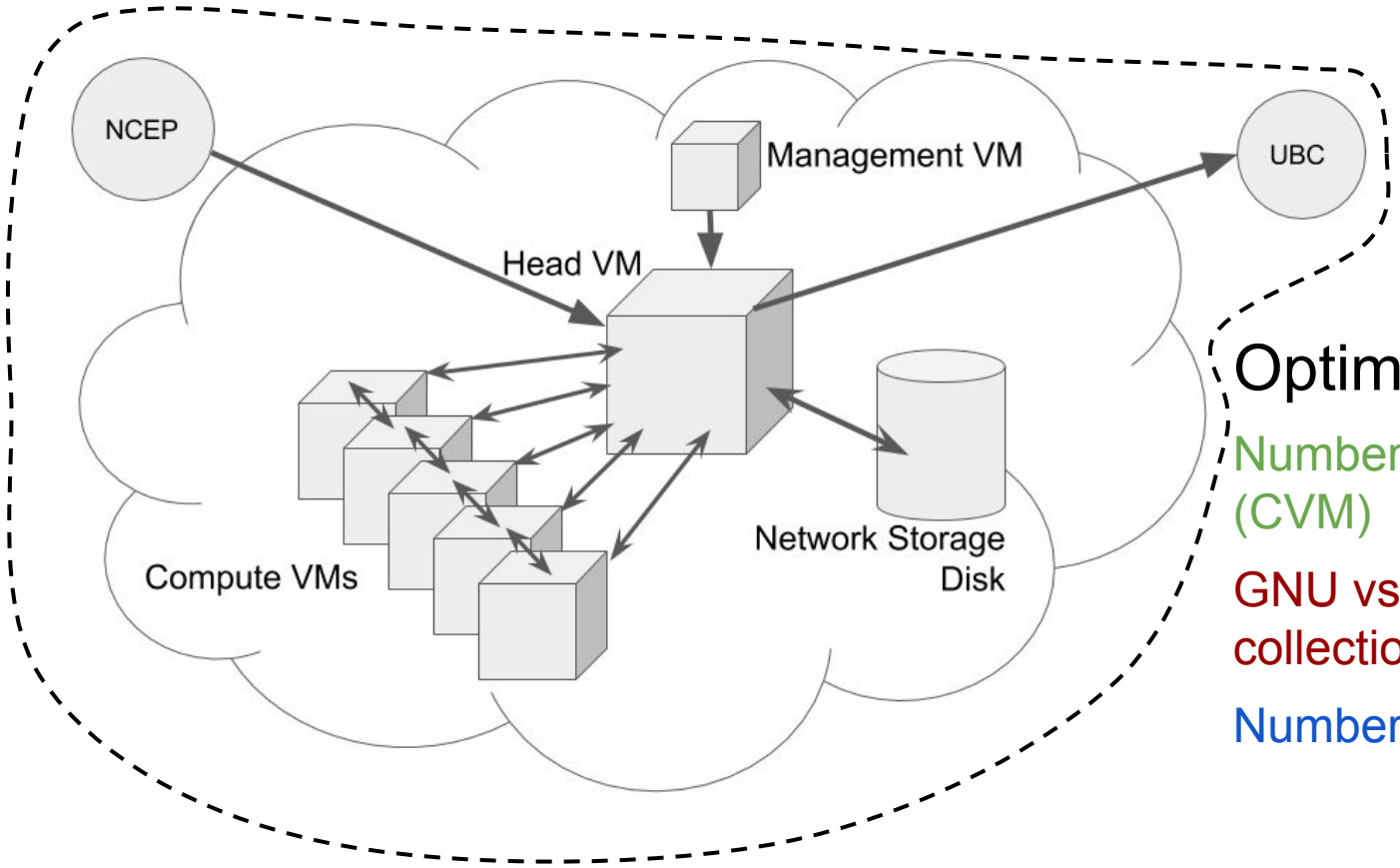
GNU vs. Intel compiler collections

Number of vCPUs per CVM

Virtual HPC cluster: Aggregate of virtual machines

Virtual Machine (VM): A user-designed, remotely hosted, computing environment.

Operating system, number of virtual CPUs, memory amount, disk space, user-installed libraries, etc. are specified by the user.



Part 2

Optimization tests:

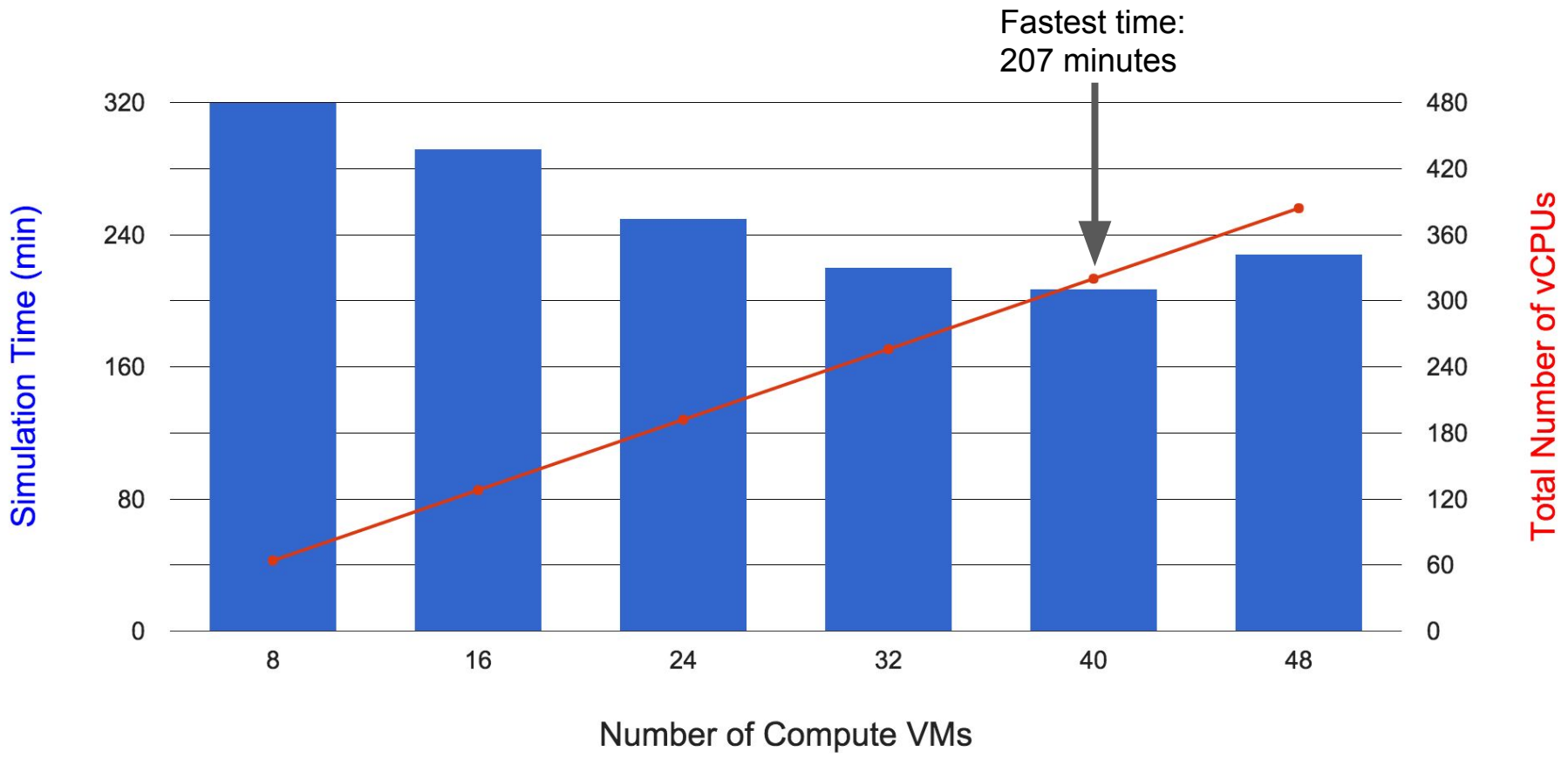
Number of Compute VMs (CVM)

GNU vs. Intel compiler collections

Number of vCPUs per CVM

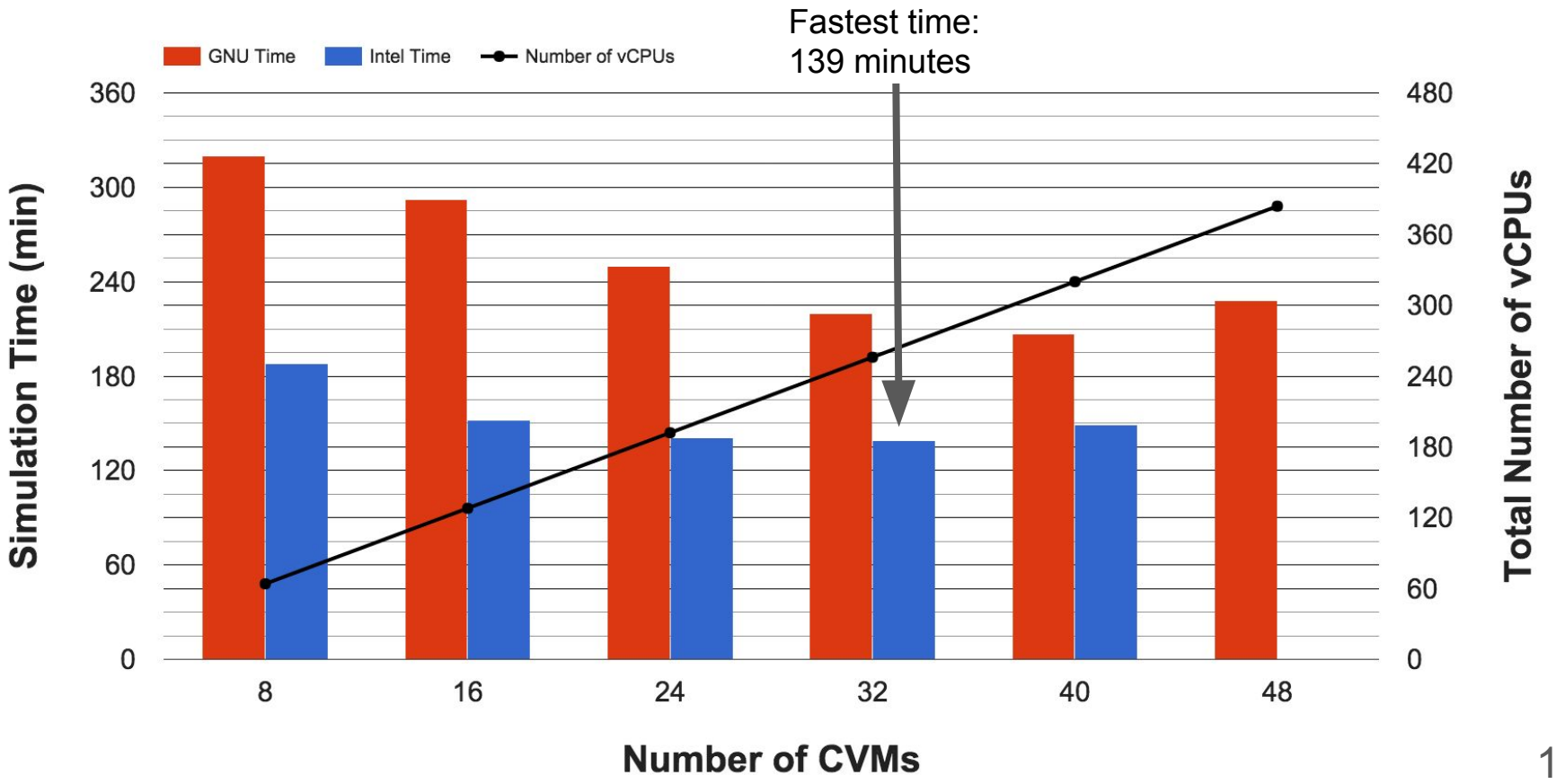
Effect of Number of Compute VMs (CVMs)

Test for scaling using multiples of 8-vCPU CVMs.



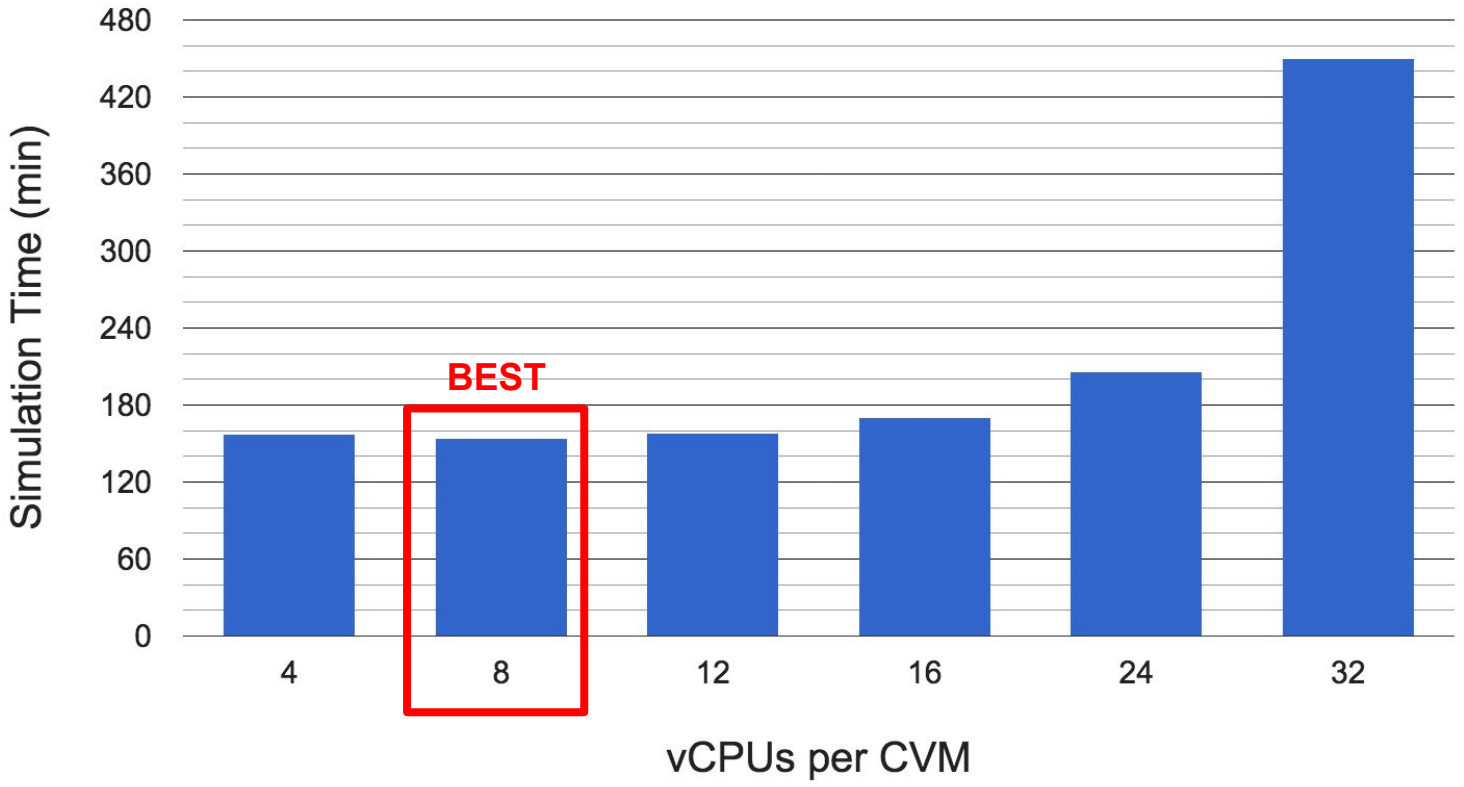
Effect of compiler choice

- Re-run same scaling tests with Intel-compiled WRF
- Intel compiled WRF ~ 45% faster completion: 139 min
- UBC's local HPC cluster time: 121 min



Effect of number of vCPUs per CVM

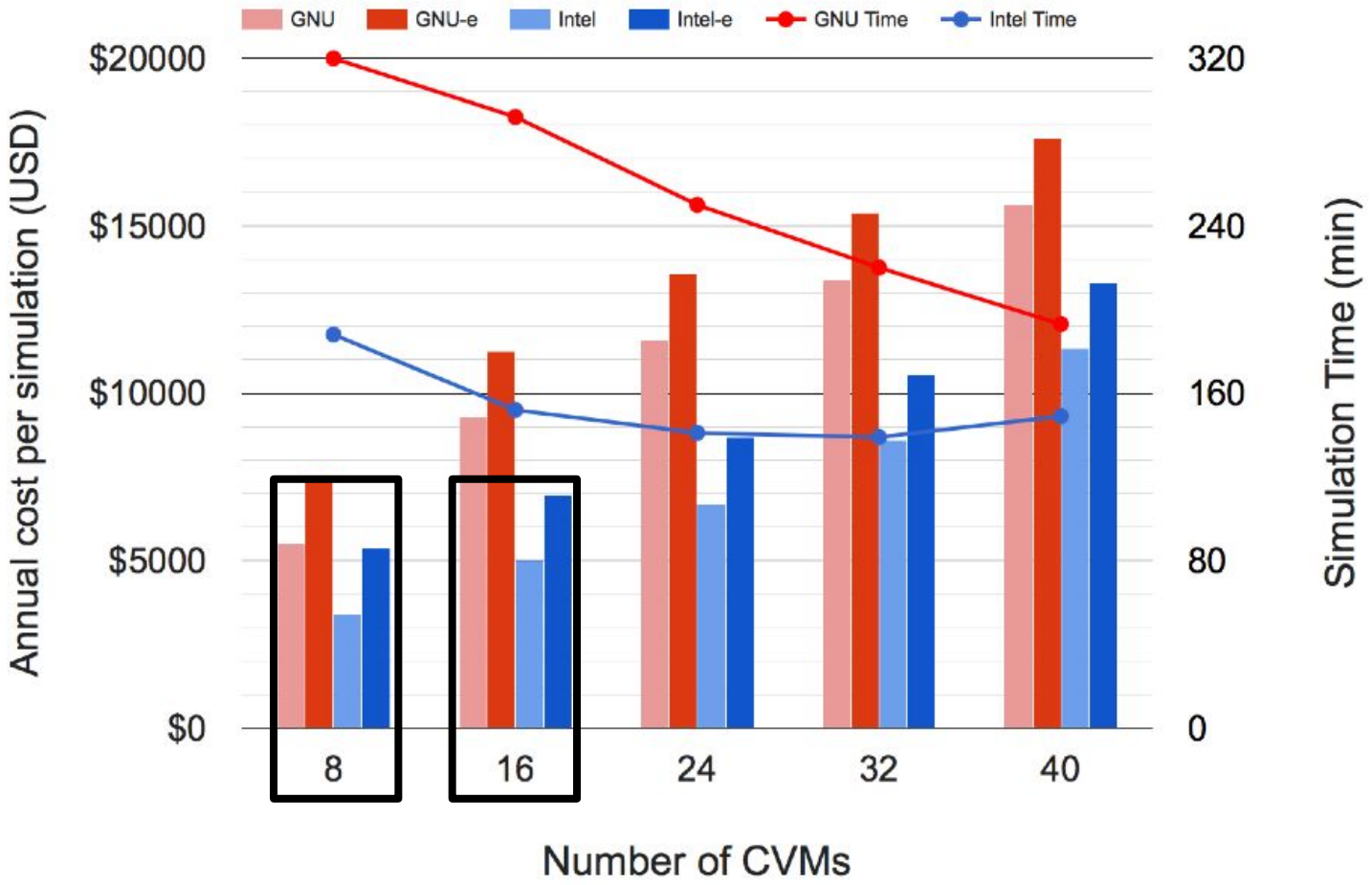
- VMs can be customized to have between 1 to 64 vCPUs
- Using more vCPUs per CVM is not necessarily better



Interim Conclusions

- The GCP can be designed to operate as a virtual HPC cluster, is **easily expandable**, and **reliable**.
- The following configuration decreased simulation time:
  - Using the Intel Compiler Collection
  - Using additional CVMs, up to the scaling limit
  - Using no more than 8-vCPUs per CVM
- Simulation times are similar to local HPC hardware.

Is the cost competitive?



Costs are determined by:

- Amount of resources used (vCPUs, memory, disk space)
- Per minute the resource is used
- Any data egress (ingress is free)

Our optimum setup (8-vCPUs per CVM with the Intel compilers) yields an annual cost of $3k-$5k (or $5k-$7k including data egress).

Is NWP in the cloud feasible for our needs?

Property	Local-HPC based NWP	Cloud-based NWP
Hardware Reliability		Better
Easily Expandable		Better
Speed	Slight edge	
Cost		Edge*

- Cost of small local HPC cluster is ~ $143k to $226k (approx. $29k to $75k / yr, amortized over 3-5 years)
- Using 8-CVM setup: can afford 10 to 25 (6 to 15) WRF-Arctic-sized runs on GCP without (with) data egress for same cost as local HPC cluster.
- Using 16-CVM setup: can afford 6 to 15 (4 to 10) runs on the GCP without (with) data egress.

Yes, real-time NWP in the cloud is economical.

Conclusion: Real-time NWP on the Google Cloud Platform is **economically viable for our needs** when optimized.

Limitations:

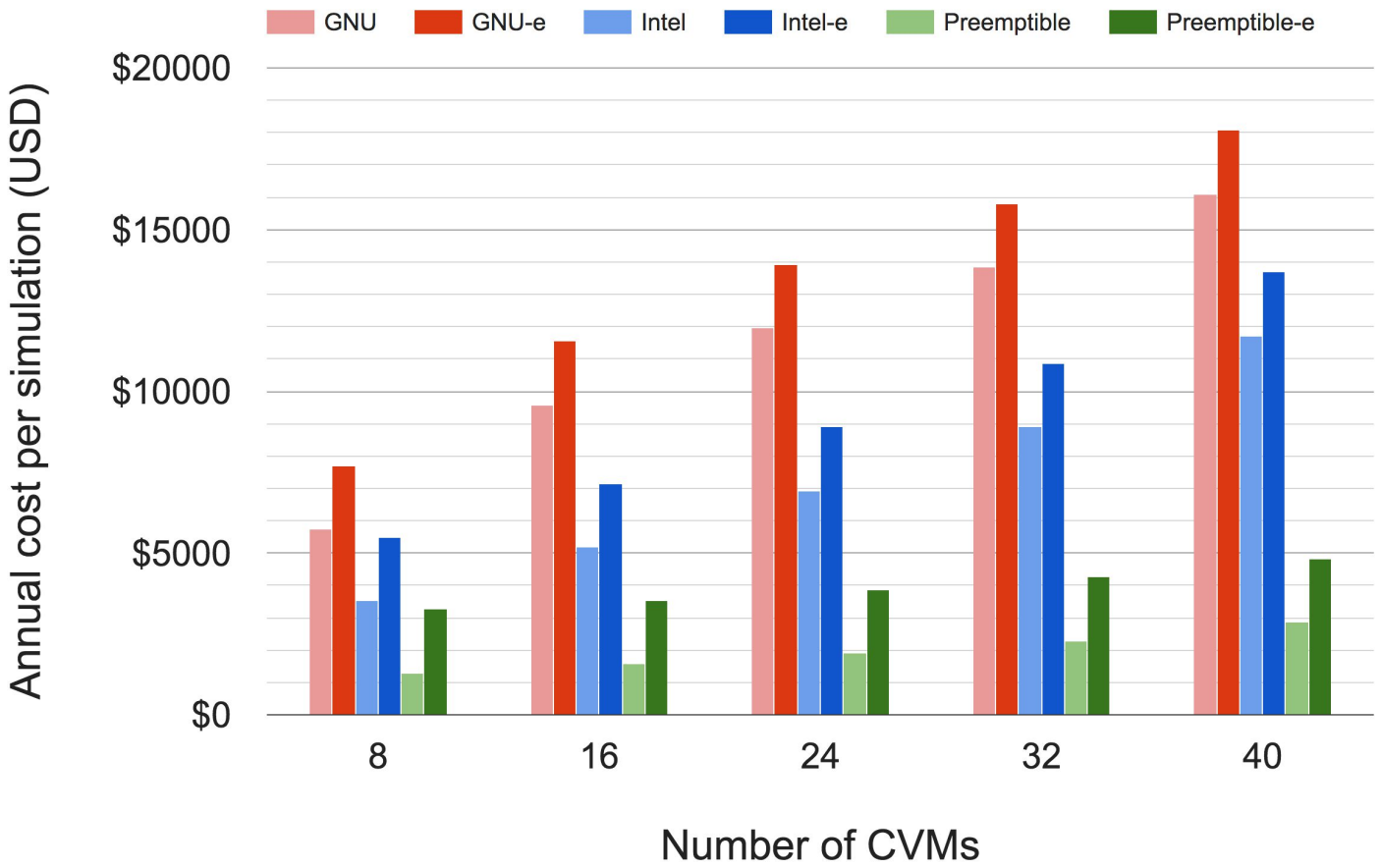
- Data egress is costly, but it is possible costs may be minimized with an entirely cloud-based system (for our use case).
- Long-term data archival may not be cheaper than storage on tape for infrequently used datasets.

How can we further reduce costs?

Further tests by UBC (Tim Chui poster)

Preemptible (interrupting) CVMs:

- Preemptible machines are 80% discounted, but jobs can be interrupted at any time without advanced warning.
- How often do preemptions occur, and can impacts be reduced?



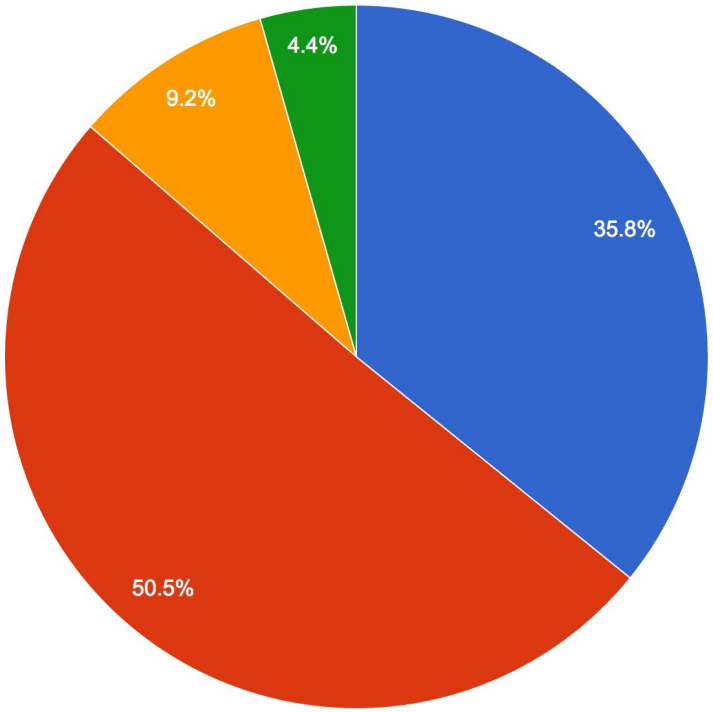
"-e" includes data egress

Further tests by UBC (Tim Chui poster)

Preemptible (interrupting) CVMs:

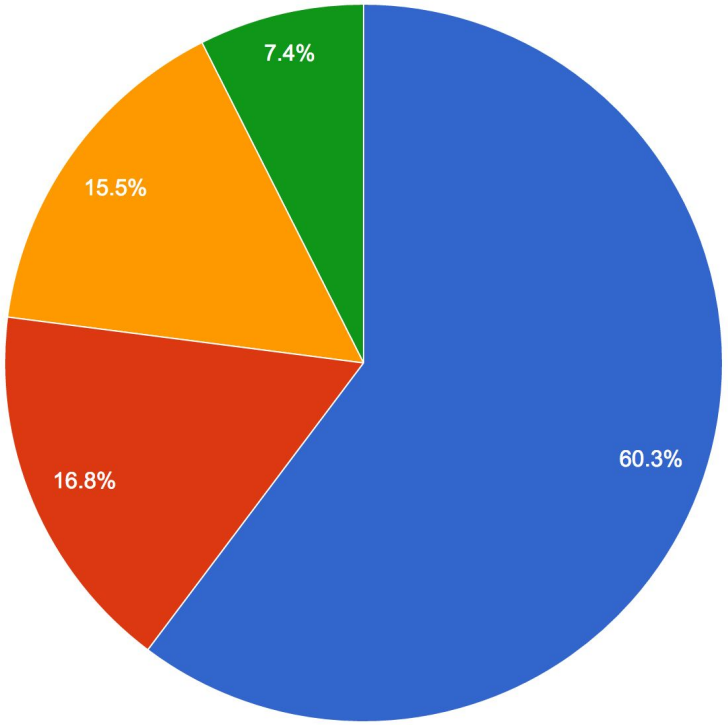
- Data egress becomes the largest portion of cost as computing cost is reduced

Standard VMs (8 CVMs)



Computing and Storage: $3532
Egress: $1971
Total: $5503

Preemptible VMs (8 CVMs)



Computing and Storage: $1300
Egress: $1971
Total: $3271

Further tests by UBC (Tim Chui poster)

Larger 64-vCPU VMs:

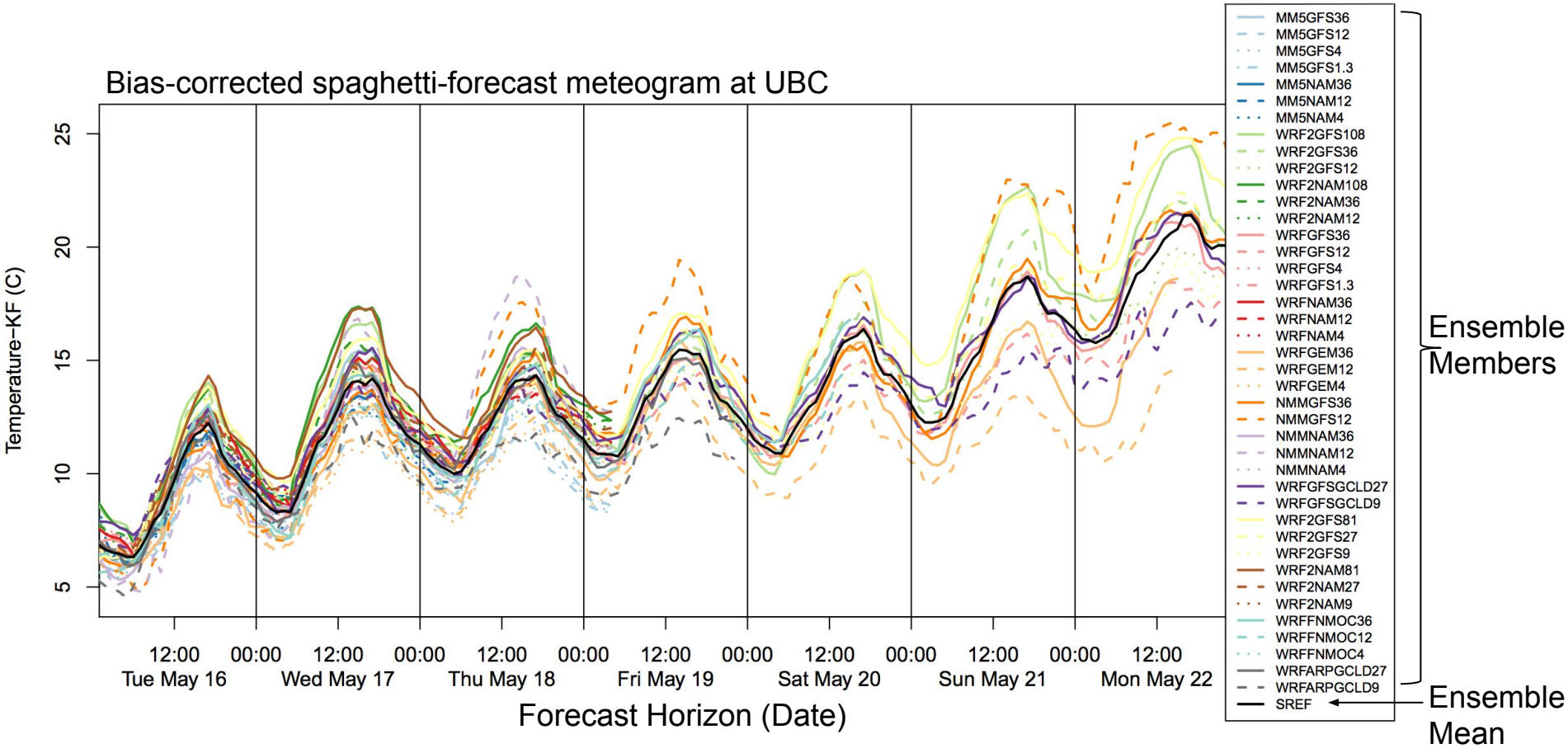
- GCP now has machines with up to 64 vCPUs.
- Can these larger machines avoid the bottlenecks we observed by eliminating the inter-node communications for some of our smaller runs?

	Machine type	Virtual CPUs	Memory	Price (USD)	Preemptible price (USD)
	n1-highcpu-2	2	1.80GB	$0.0709	$0.0150
	n1-highcpu-4	4	3.60GB	$0.1418	$0.0300
8x	n1-highcpu-8	8	7.20GB	$0.2836	$0.0600
	n1-highcpu-16	16	14.40GB	$0.5672	$0.1200
OR	n1-highcpu-32	32	28.80GB	$1.1344	$0.2400
1x	n1-highcpu-64	64	57.6GB	$2.2688	$0.4800

Source: https://cloud.google.com/compute/pricing (as of 16 May 2017)

Further tests by UBC (Tim Chui poster)

Redundancy on GCP and ensemble expansion



Part 2: How to setup a virtual HPC cluster on GCP

1. Create a virtual machine (VM) and install the required WRF dependencies. This is the head node (HVM).
2. Mount a persistent disk to the HVM.
3. Download and compile the WRF code on the mounted disk following the standard WRF installation instructions.
4. Save a ‘snapshot’ of the HVM.
5. Replicate the snapshot to make a compute node (CVM).
6. NFS mount the CVM to the HVM storage disk.
7. Save snapshot of the CVM and replicate for multiple CVMs.
8. Transfer ssh keys to the CVMs.
9. Run WRF!

Google Cloud Platform wrf-arctic

DASHBOARD ACTIVITY

Google Cloud Console--Dashboard

CUSTOMIZE

Project info

wrf-arctic

Manage project settings

Resources

Compute Engine
100+ instances

Trace

No trace data from the past 7 days

Get started with Stackdriver Trace

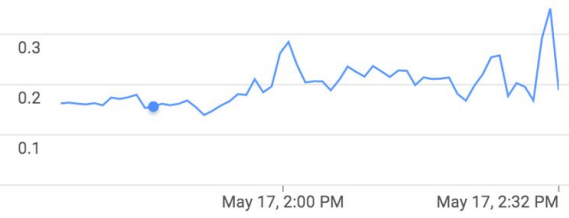
Getting Started

- Enable APIs and get credentials like keys
- Deploy a prebuilt solution
- View source and logs on the Debug page
- Monitor errors with Error Reporting
- Deploy a Hello World app
- Create a Cloud Storage bucket
- Install the Cloud SDK

Explore all tutorials

Compute Engine

CPU (%)

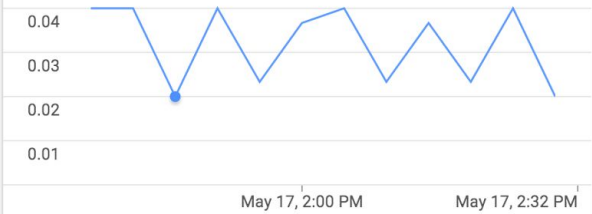


CPU: 0.156

Go to the Compute Engine dashboard

APIs

Requests (requests/sec)



Requests: 0.02

Go to APIs overview

Google Cloud Platform status

All services normal

Go to Cloud status dashboard

Billing

$222.13

Approximate charges so far this month

View detailed charges

Error Reporting

No sign of any errors. Have you set up Error Reporting?

Set up Error Reporting

News

Introducing Google Cloud IoT Core: for securely connecting and managing IoT devices at scale
1 day ago

Cloud Spanner is now production-ready; let the migrations begin!
1 day ago

Mapping your organization with the Google Cloud Platform resource hierarchy
6 days ago

Read all news

Project info

wrf-arctic

Manage project settings

Resources

Compute Engine
100+ instances

Add or manage Compute Engine resources

Trace

No trace data from the past 7 days

Get started with Stackdriver Trace

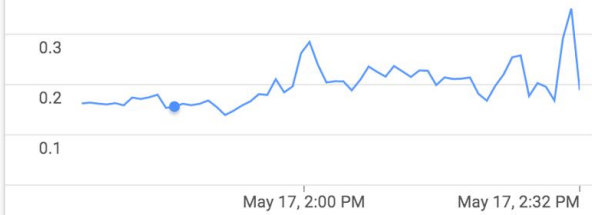
Getting Started

- Enable APIs and get credentials like keys
- Deploy a prebuilt solution
- View source and logs on the Debug page
- Monitor errors with Error Reporting
- Deploy a Hello World app
- Create a Cloud Storage bucket
- Install the Cloud SDK

Explore all tutorials

Compute Engine

CPU (%)

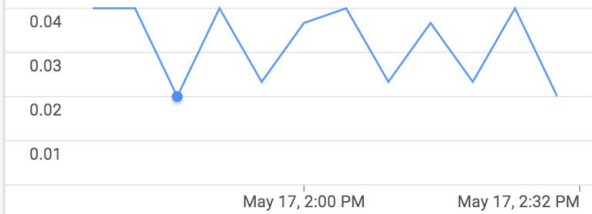


CPU: 0.156

Go to the Compute Engine dashboard

APIs

Requests (requests/sec)



Requests: 0.02

Go to APIs overview

Google Cloud Platform status

All services normal

Go to Cloud status dashboard

Billing

$222.13

Approximate charges so far this month

View detailed charges

Error Reporting

No sign of any errors. Have you set up Error Reporting?

Set up Error Reporting

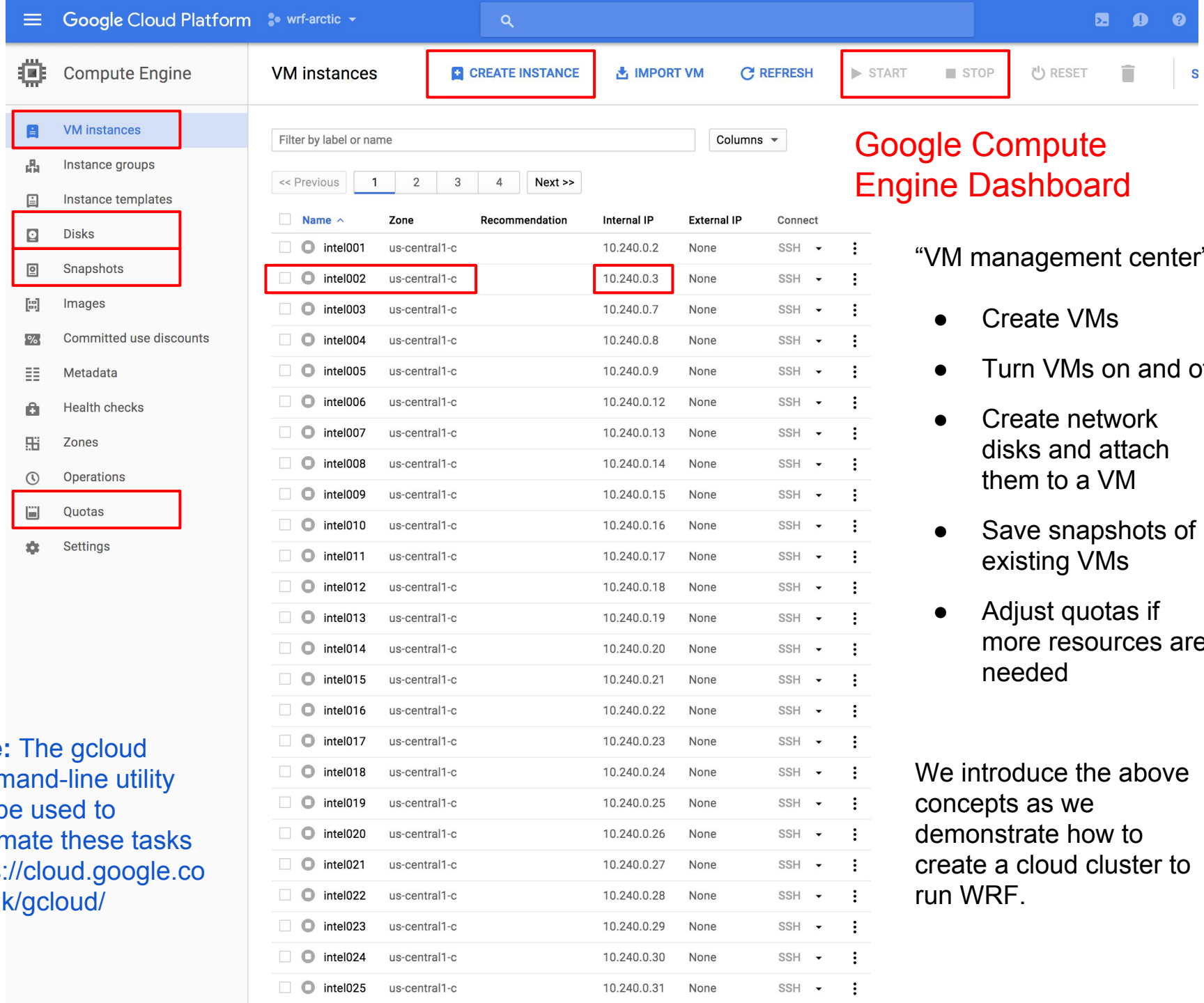
News

Introducing Google Cloud IoT Core: for securely connecting and managing IoT devices at scale
1 day ago

Cloud Spanner is now production-ready; let the migrations begin!
1 day ago

Mapping your organization with the Google Cloud Platform resource hierarchy
6 days ago

Read all news



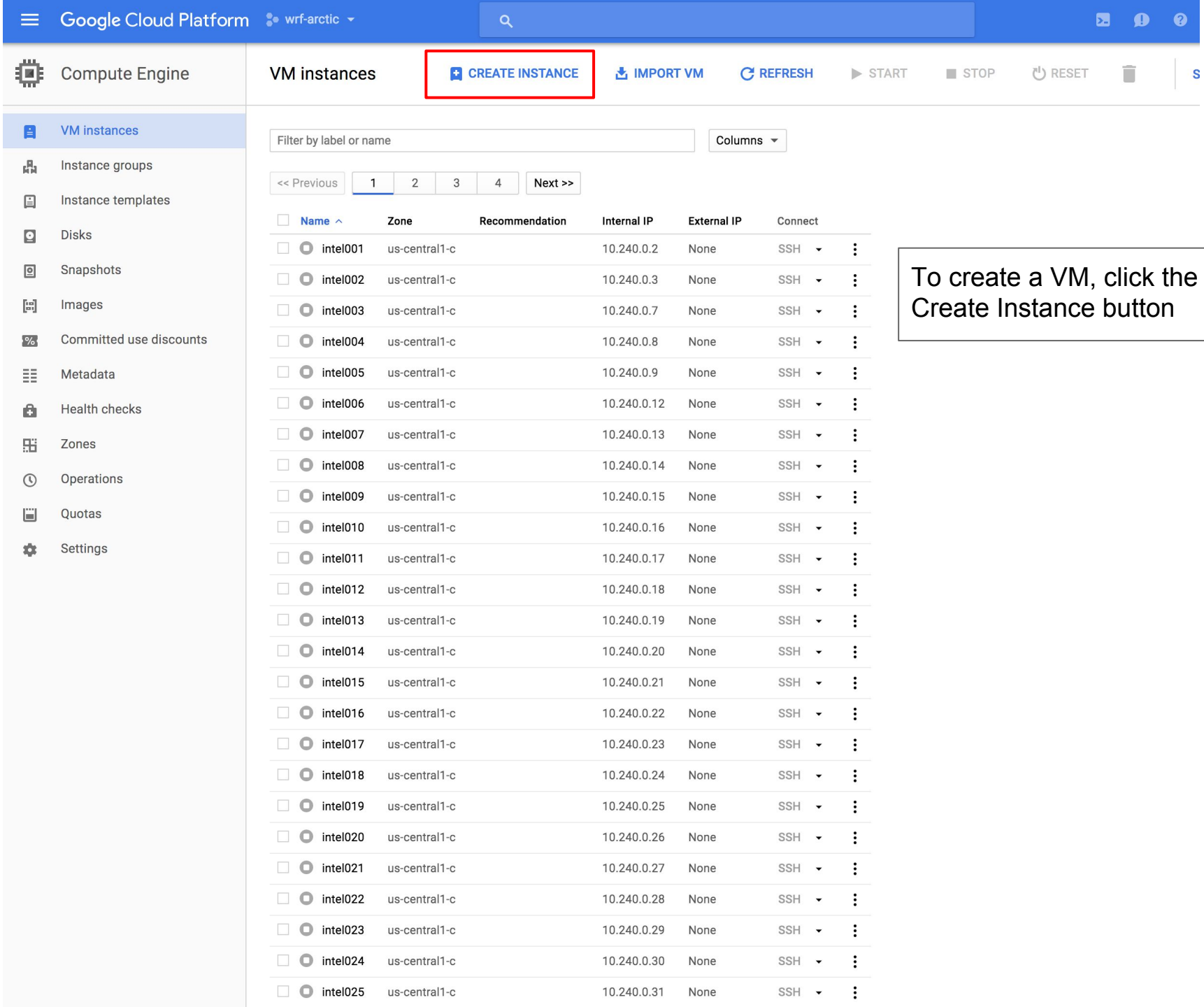
Google Compute Engine Dashboard

"VM management center"

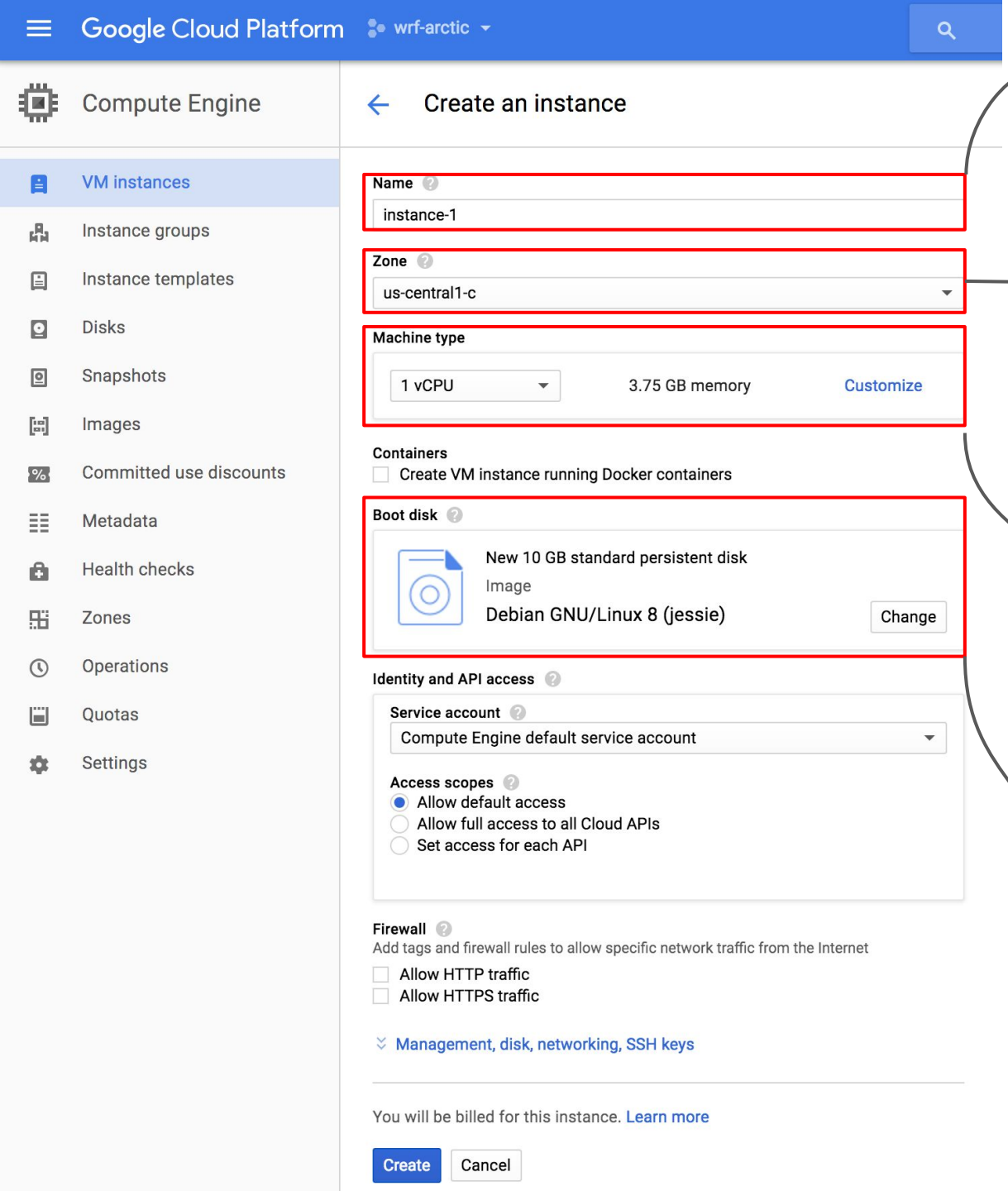
- Create VMs
- Turn VMs on and off
- Create network disks and attach them to a VM
- Save snapshots of existing VMs
- Adjust quotas if more resources are needed

We introduce the above concepts as we demonstrate how to create a cloud cluster to run WRF.

Note: The gcloud command-line utility can be used to automate these tasks https://cloud.google.com/sdk/gcloud/



To create a VM, click the Create Instance button

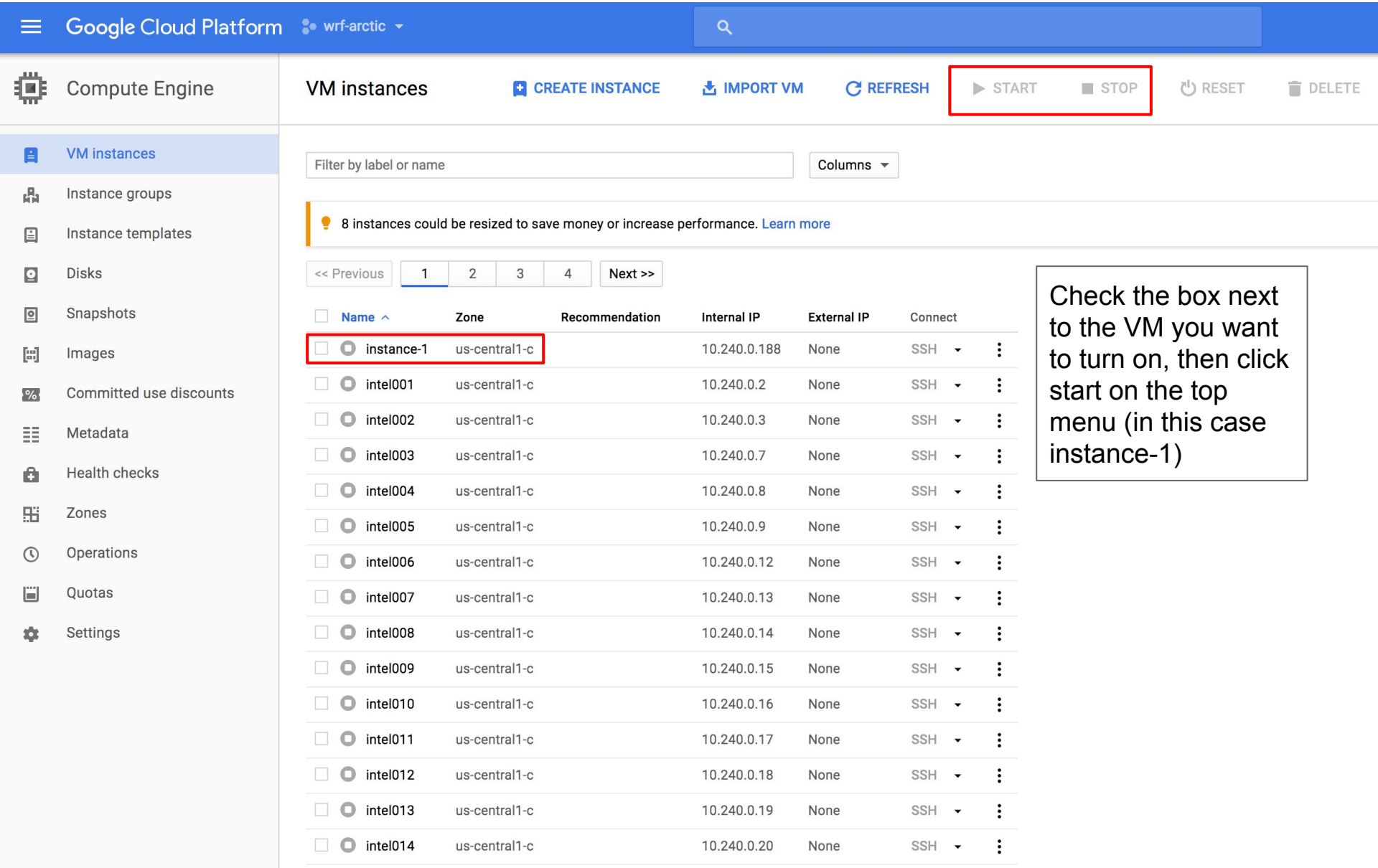


VM name (e.g., instance-1)

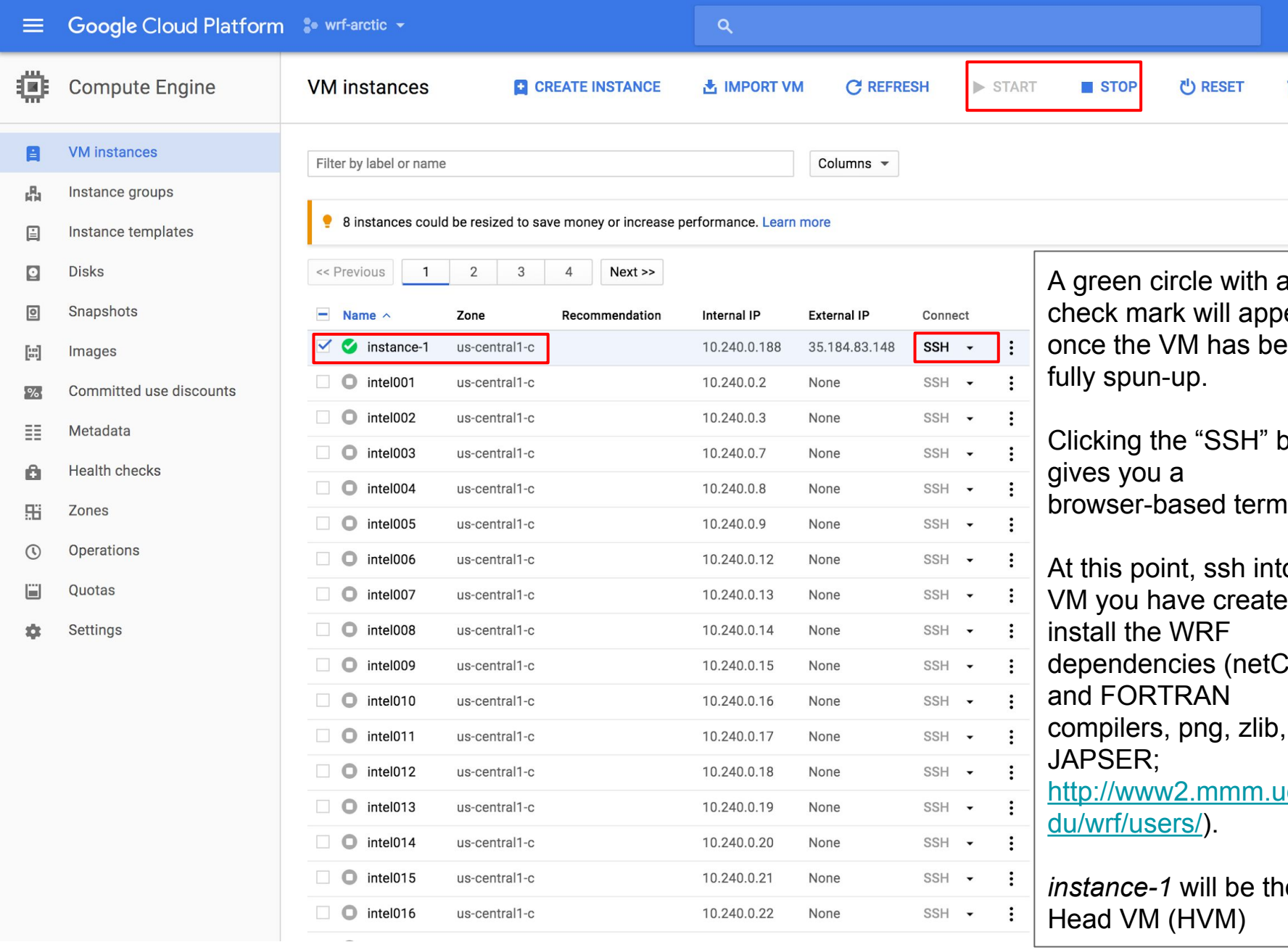
Location of computing resources. Different zones have different types of resources (processor types, machine sizes, etc: https://cloud.google.com/compute/docs/regions-zones/regions-zones)

VM configuration (# of vCPUs and memory). We suggest 8-vCPU VMs.

OS and boot disk size—can be a saved configuration (snapshot). We used CentOS in our tests.



Check the box next to the VM you want to turn on, then click start on the top menu (in this case instance-1)

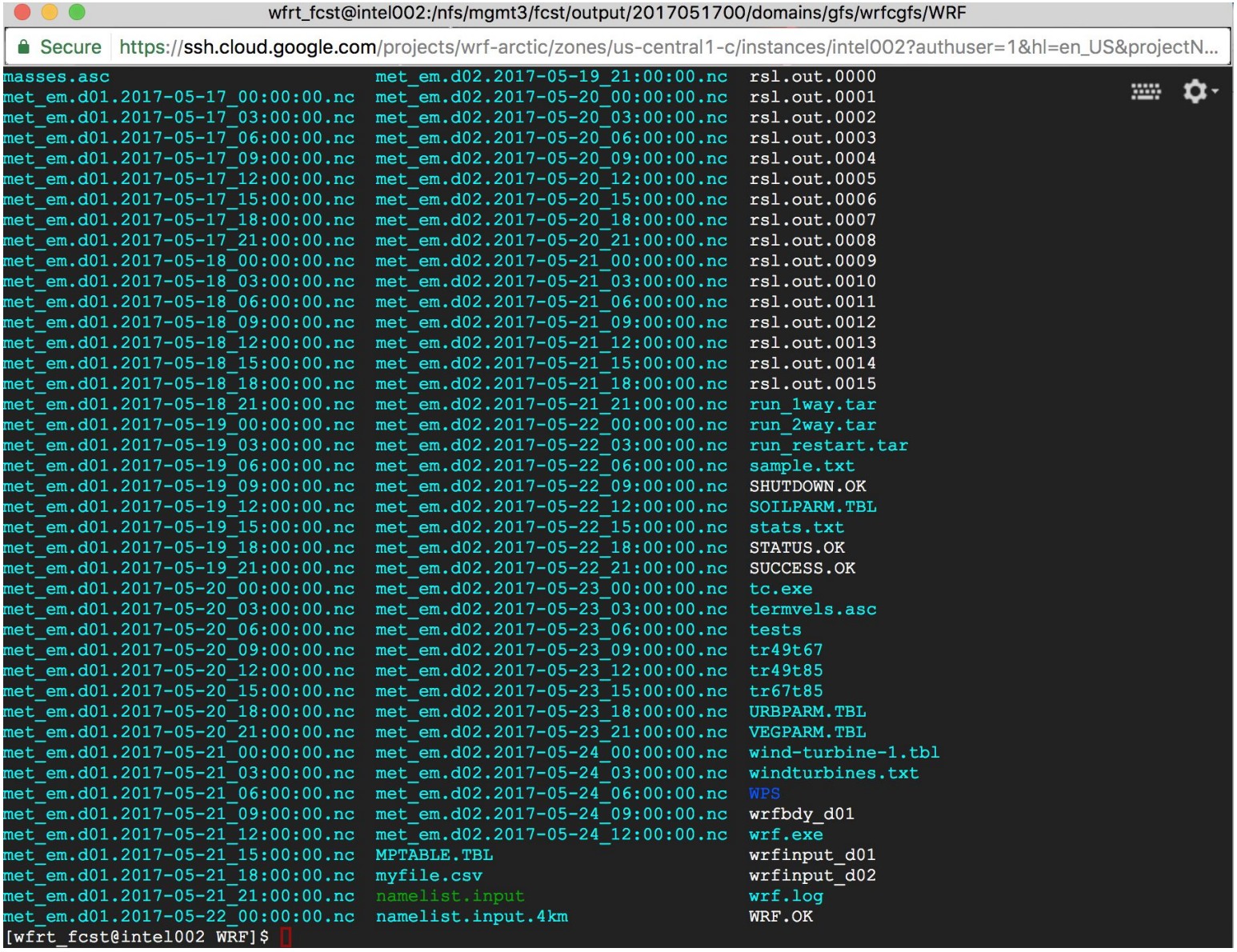


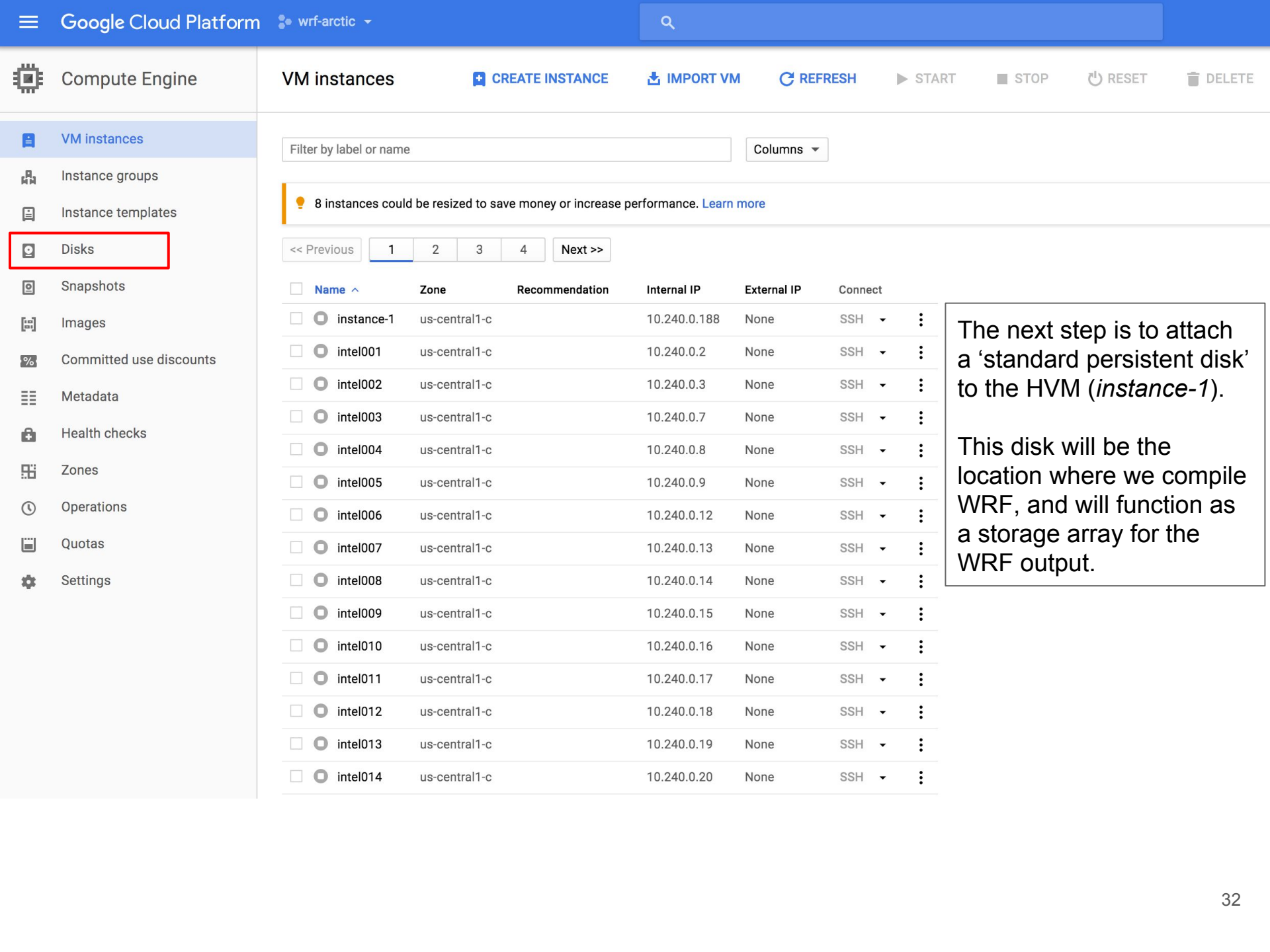
A green circle with a check mark will appear once the VM has been fully spun-up.

Clicking the “SSH” button gives you a browser-based terminal.

At this point, ssh into the VM you have created and install the WRF dependencies (netCDF, c and FORTRAN compilers, png, zlib, and JAPSER; http://www2.mmm.ucar.edu/wrf/users/).

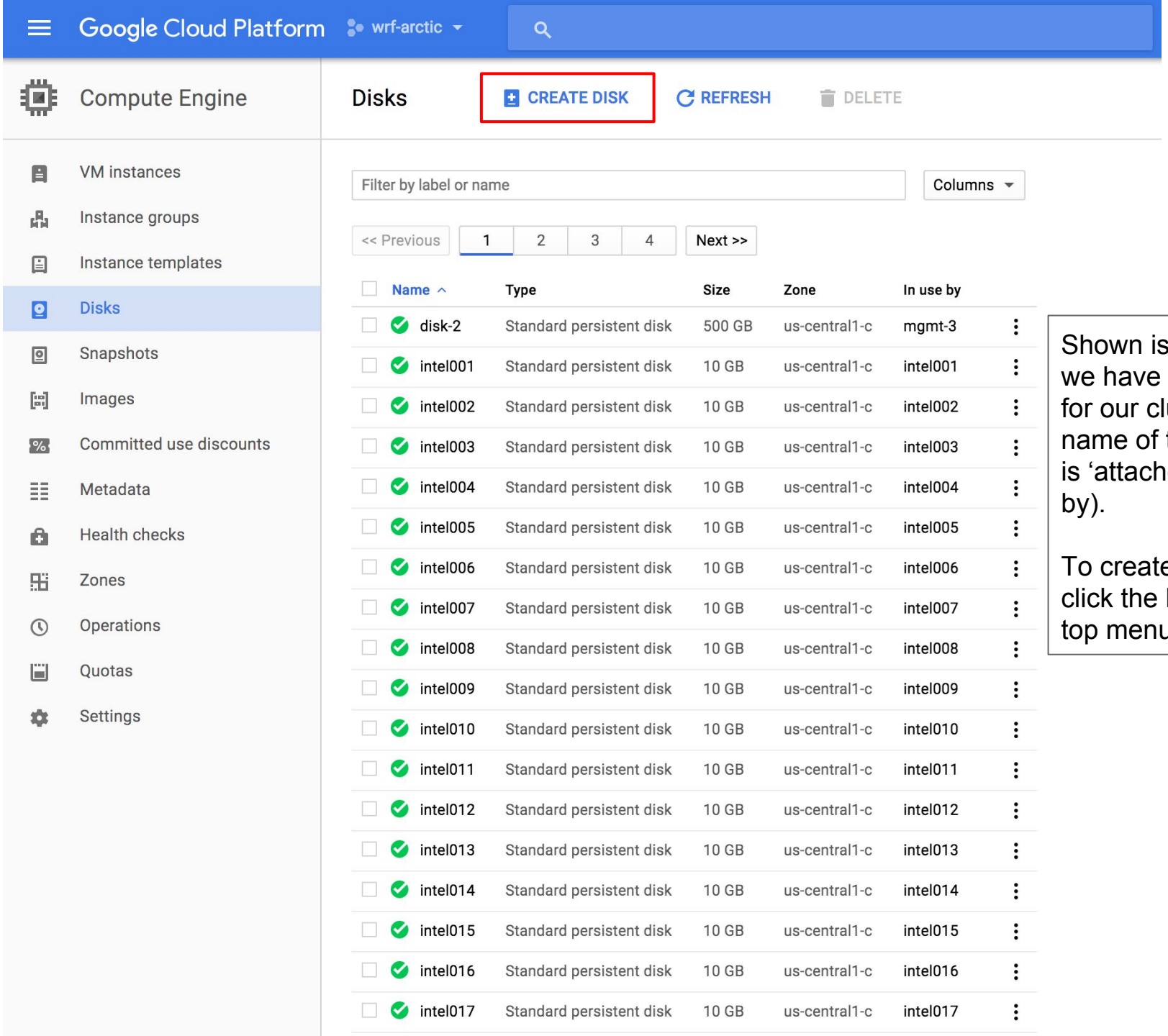
instance-1 will be the Head VM (HVM)


wfrt_fcst@intel002:/nfs/mgmt3/fcst/output/2017051700/domains/gfs/wrfcgfs/WRF
Secure
https://ssh.cloud.google.com/projects/wrf-arctic/zones/us-central1-c/instances/intel002?authuser=1&hl=en_US&projectN...
masses.asc
met_em.d01.2017-05-17_00:00:00.nc
met_em.d01.2017-05-17_03:00:00.nc
met_em.d01.2017-05-17_06:00:00.nc
met_em.d01.2017-05-17_09:00:00.nc
met_em.d01.2017-05-17_12:00:00.nc
met_em.d01.2017-05-17_15:00:00.nc
met_em.d01.2017-05-17_18:00:00.nc
met_em.d01.2017-05-17_21:00:00.nc
met_em.d01.2017-05-18_00:00:00.nc
met_em.d01.2017-05-18_03:00:00.nc
met_em.d01.2017-05-18_06:00:00.nc
met_em.d01.2017-05-18_09:00:00.nc
met_em.d01.2017-05-18_12:00:00.nc
met_em.d01.2017-05-18_15:00:00.nc
met_em.d01.2017-05-18_18:00:00.nc
met_em.d01.2017-05-18_21:00:00.nc
met_em.d01.2017-05-19_00:00:00.nc
met_em.d01.2017-05-19_03:00:00.nc
met_em.d01.2017-05-19_06:00:00.nc
met_em.d01.2017-05-19_09:00:00.nc
met_em.d01.2017-05-19_12:00:00.nc
met_em.d01.2017-05-19_15:00:00.nc
met_em.d01.2017-05-19_18:00:00.nc
met_em.d01.2017-05-19_21:00:00.nc
met_em.d01.2017-05-20_00:00:00.nc
met_em.d01.2017-05-20_03:00:00.nc
met_em.d01.2017-05-20_06:00:00.nc
met_em.d01.2017-05-20_09:00:00.nc
met_em.d01.2017-05-20_12:00:00.nc
met_em.d01.2017-05-20_15:00:00.nc
met_em.d01.2017-05-20_18:00:00.nc
met_em.d01.2017-05-20_21:00:00.nc
met_em.d01.2017-05-21_00:00:00.nc
met_em.d01.2017-05-21_03:00:00.nc
met_em.d01.2017-05-21_06:00:00.nc
met_em.d01.2017-05-21_09:00:00.nc
met_em.d01.2017-05-21_12:00:00.nc
met_em.d01.2017-05-21_15:00:00.nc
met_em.d01.2017-05-21_18:00:00.nc
met_em.d01.2017-05-21_21:00:00.nc
met_em.d01.2017-05-22_00:00:00.nc
met_em.d02.2017-05-19_21:00:00.nc
met_em.d02.2017-05-20_00:00:00.nc
met_em.d02.2017-05-20_03:00:00.nc
met_em.d02.2017-05-20_06:00:00.nc
met_em.d02.2017-05-20_09:00:00.nc
met_em.d02.2017-05-20_12:00:00.nc
met_em.d02.2017-05-20_15:00:00.nc
met_em.d02.2017-05-20_18:00:00.nc
met_em.d02.2017-05-20_21:00:00.nc
met_em.d02.2017-05-21_00:00:00.nc
met_em.d02.2017-05-21_03:00:00.nc
met_em.d02.2017-05-21_06:00:00.nc
met_em.d02.2017-05-21_09:00:00.nc
met_em.d02.2017-05-21_12:00:00.nc
met_em.d02.2017-05-21_15:00:00.nc
met_em.d02.2017-05-21_18:00:00.nc
met_em.d02.2017-05-21_21:00:00.nc
met_em.d02.2017-05-22_00:00:00.nc
met_em.d02.2017-05-22_03:00:00.nc
met_em.d02.2017-05-22_06:00:00.nc
met_em.d02.2017-05-22_09:00:00.nc
met_em.d02.2017-05-22_12:00:00.nc
met_em.d02.2017-05-22_15:00:00.nc
met_em.d02.2017-05-22_18:00:00.nc
met_em.d02.2017-05-22_21:00:00.nc
met_em.d02.2017-05-23_00:00:00.nc
met_em.d02.2017-05-23_03:00:00.nc
met_em.d02.2017-05-23_06:00:00.nc
met_em.d02.2017-05-23_09:00:00.nc
met_em.d02.2017-05-23_12:00:00.nc
met_em.d02.2017-05-23_15:00:00.nc
met_em.d02.2017-05-23_18:00:00.nc
met_em.d02.2017-05-23_21:00:00.nc
met_em.d02.2017-05-24_00:00:00.nc
met_em.d02.2017-05-24_03:00:00.nc
met_em.d02.2017-05-24_06:00:00.nc
met_em.d02.2017-05-24_09:00:00.nc
met_em.d02.2017-05-24_12:00:00.nc
MPTABLE.TBL
myfile.csv
namelist.input
namelist.input.4km
rsl.out.0000
rsl.out.0001
rsl.out.0002
rsl.out.0003
rsl.out.0004
rsl.out.0005
rsl.out.0006
rsl.out.0007
rsl.out.0008
rsl.out.0009
rsl.out.0010
rsl.out.0011
rsl.out.0012
rsl.out.0013
rsl.out.0014
rsl.out.0015
run_1way.tar
run_2way.tar
run_restart.tar
sample.txt
SHUTDOWN.OK
SOILPARM.TBL
stats.txt
STATUS.OK
SUCCESS.OK
tc.exe
termvels.asc
tests
tr49t67
tr49t85
tr67t85
URBPARM.TBL
VEGPARM.TBL
wind-turbine-1.tbl
windturbines.txt
WPS
wrfbdy_d01
wrf.exe
wrfinput_d01
wrfinput_d02
wrf.log
WRF.OK
[wfrt_fcst@intel002 WRF]$



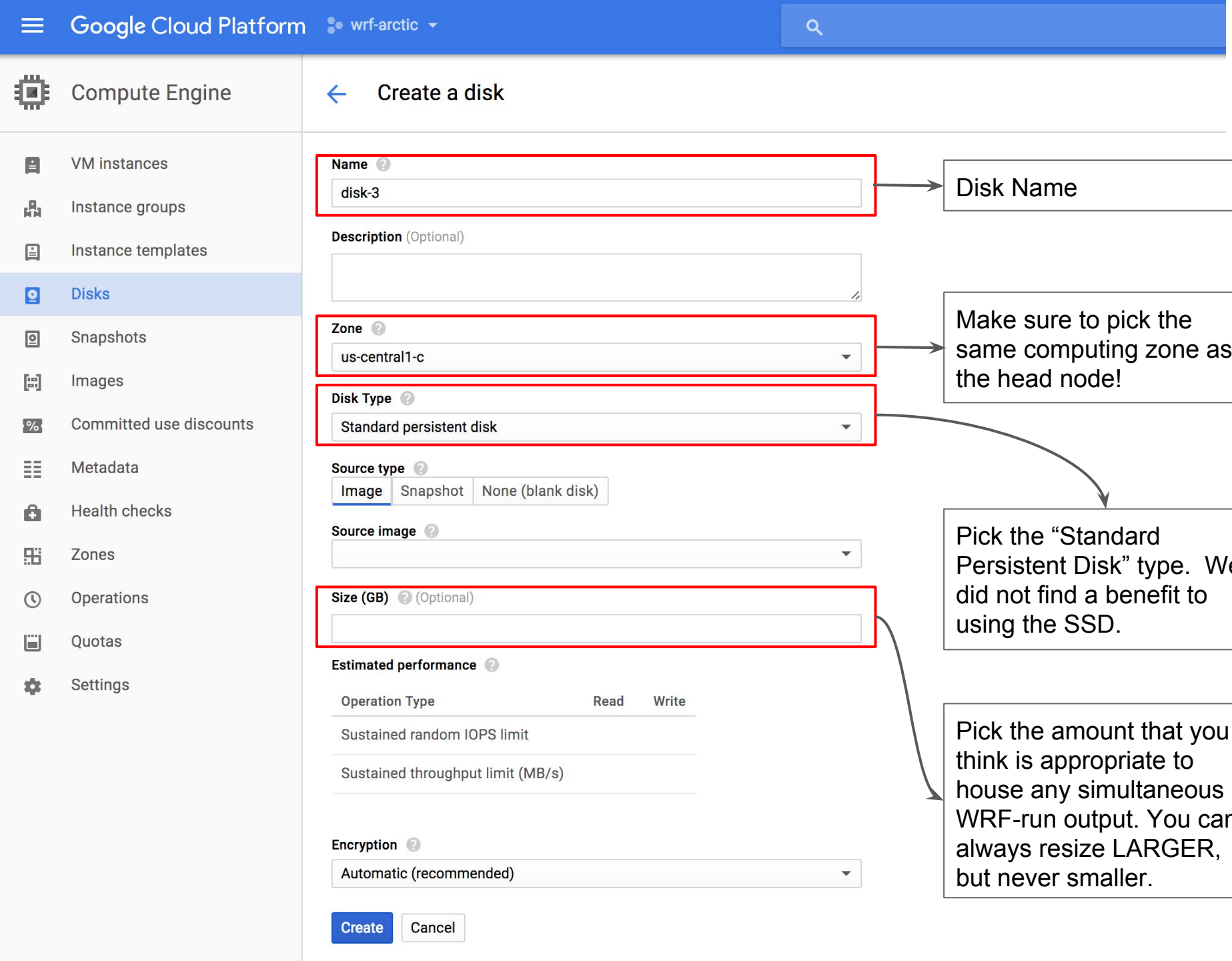
The next step is to attach a ‘standard persistent disk’ to the HVM (*instance-1*).

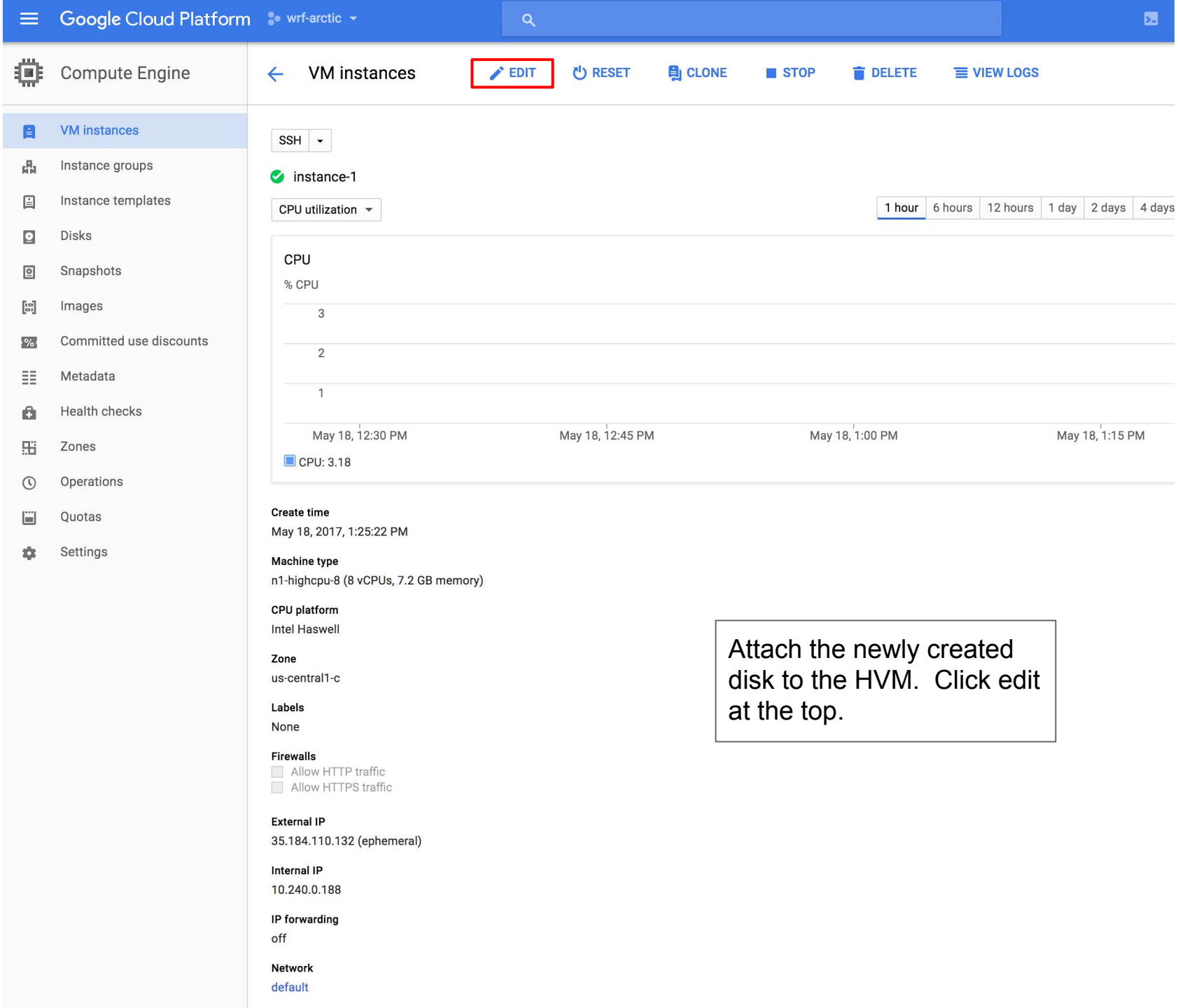
This disk will be the location where we compile WRF, and will function as a storage array for the WRF output.



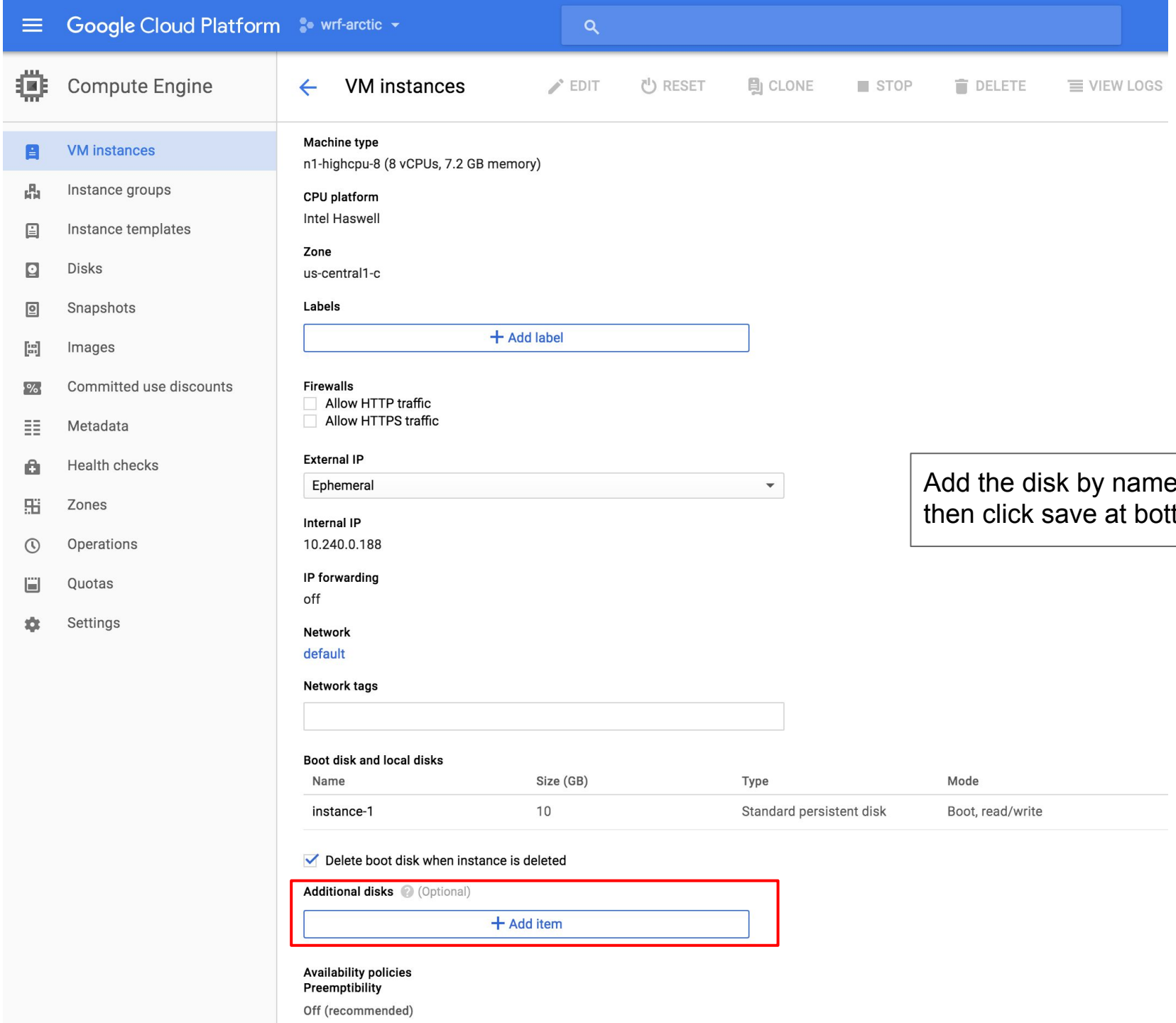
Shown is a list of ‘disks’ we have already created for our cluster, and the name of the VM the disk is ‘attached’ to (in use by).

To create a new disk, click the button on the top menu.

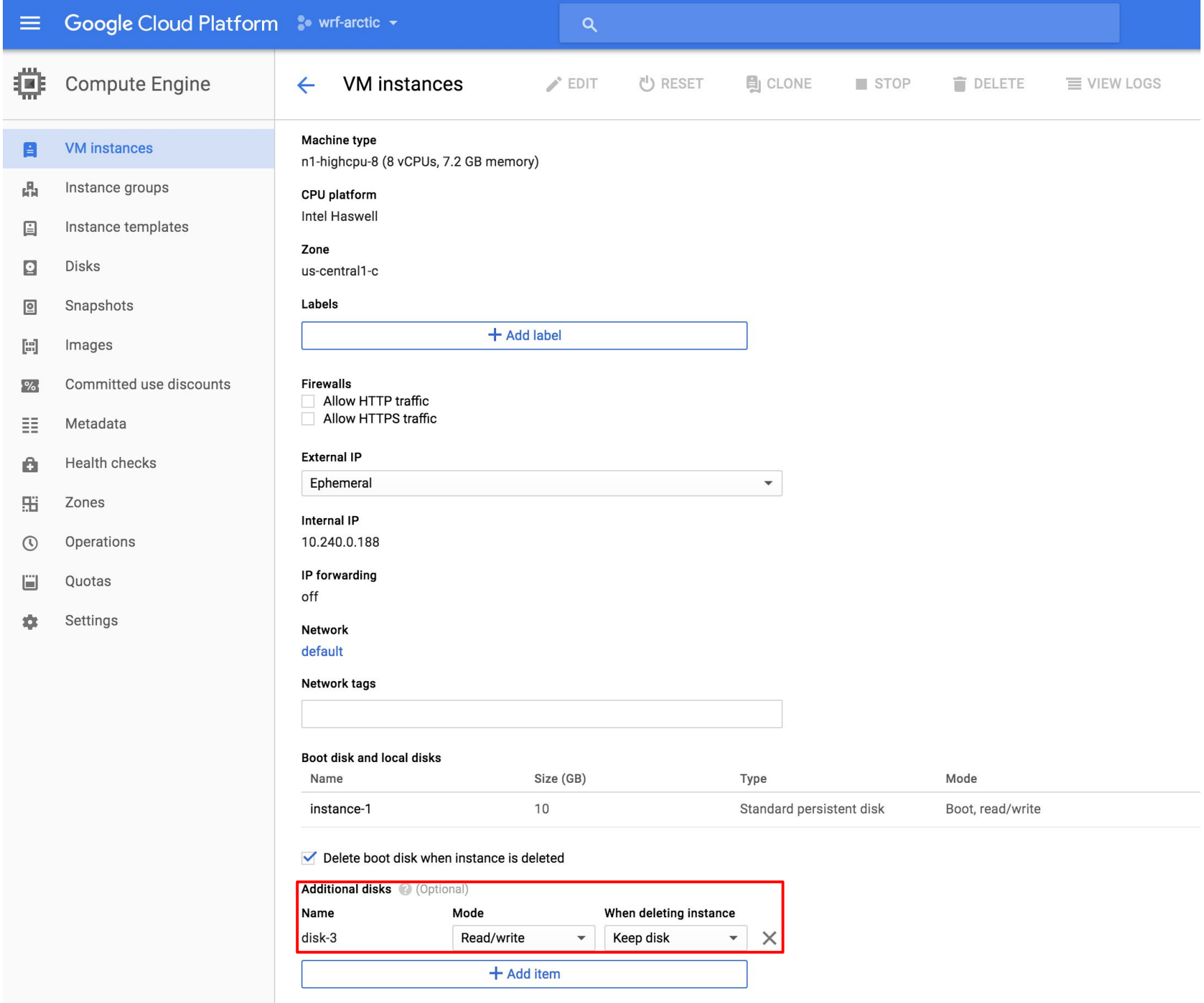

Google Cloud Platform
wrf-arctic
Compute Engine
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Committed use discounts
Metadata
Health checks
Zones
Operations
Quotas
Settings
Create a disk
Name
disk-3
Description (Optional)
Zone
us-central1-c
Disk Type
Standard persistent disk
Source type
Image
Snapshot
None (blank disk)
Source image
Size (GB) (Optional)
Estimated performance
Operation Type
Read
Write
Sustained random IOPS limit
Sustained throughput limit (MB/s)
Encryption
Automatic (recommended)
Create
Cancel
Disk Name
Make sure to pick the same computing zone as the head node!
Pick the “Standard Persistent Disk” type. We did not find a benefit to using the SSD.
Pick the amount that you think is appropriate to house any simultaneous WRF-run output. You can always resize LARGER, but never smaller.



Attach the newly created disk to the HVM. Click edit at the top.



Add the disk by name, then click save at bottom.


Google Cloud Platform
wrf-arctic
Compute Engine
VM instances
EDIT
RESET
CLONE
STOP
DELETE
VIEW LOGS
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Committed use discounts
Metadata
Health checks
Zones
Operations
Quotas
Settings
Machine type
n1-highcpu-8 (8 vCPUs, 7.2 GB memory)
CPU platform
Intel Haswell
Zone
us-central1-c
Labels
+ Add label
Firewalls
Allow HTTP traffic
Allow HTTPS traffic
External IP
Ephemeral
Internal IP
10.240.0.188
IP forwarding
off
Network
default
Network tags
Boot disk and local disks
Name
Size (GB)
Type
Mode
instance-1
10
Standard persistent disk
Boot, read/write
Delete boot disk when instance is deleted
Additional disks
(Optional)
Name
Mode
When deleting instance
disk-3
Read/write
Keep disk
+ Add item

ssh into the HVM and setup the HVM as NFS server

1. Partition the attached disk by running the following commands

```
sudo fdisk /dev/sdb
n  (for new partition)
p  (for primary partition)
1  (for the first partition)
Accept the default for next 2 prompts (hit return)
w  (write partition table to disk and exit)
```

2. Mount the attached disk

```
sudo mkfs.ext4 /dev/sdb1
sudo mkdir /mnt/disk
sudo mount -t ext4 /dev/sdb1 /mnt/disk
```

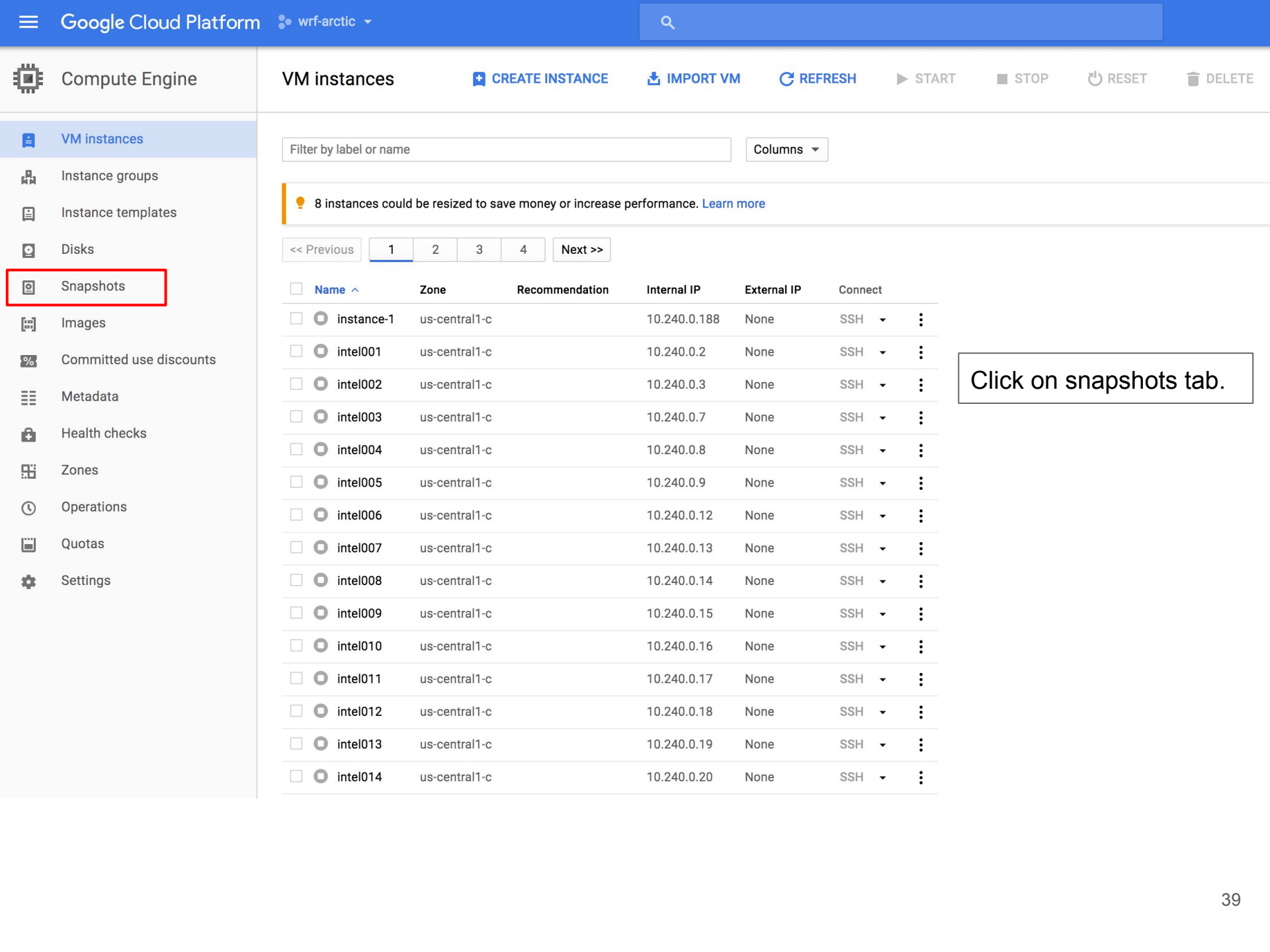
3. Edit fstab to allow for auto-mounting on VM startup

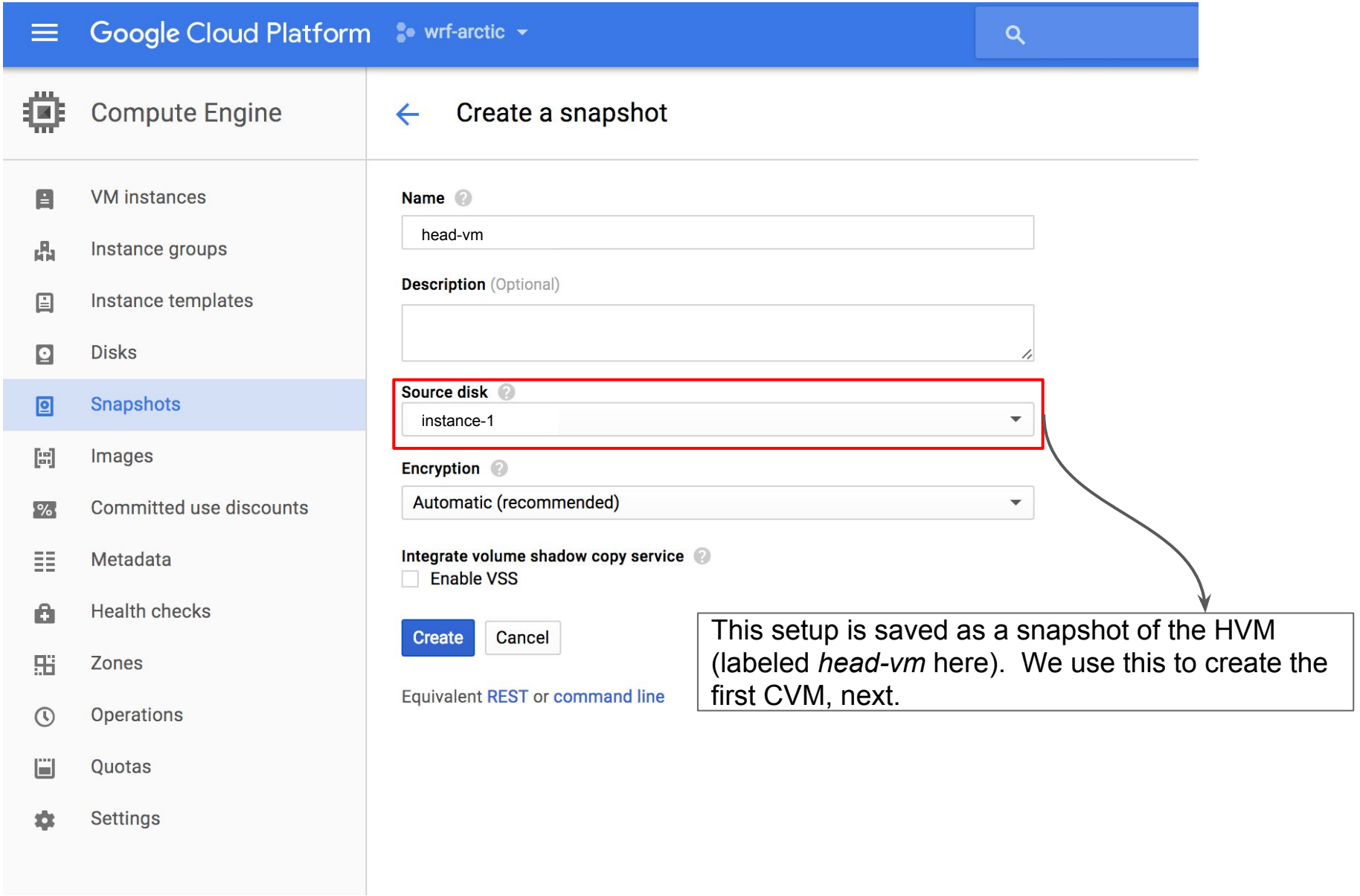
```
sudo tune2fs -l /dev/sdb1
sudo vi /etc/fstab
    Add: UUID=[Output from tunefs] /mnt/disk ext4 defaults 0 0
sudo vi /etc/exports
    Add: /mnt/disk 10.240.0.0/255.255.0.0(rw,no_root_squash,sync)
```

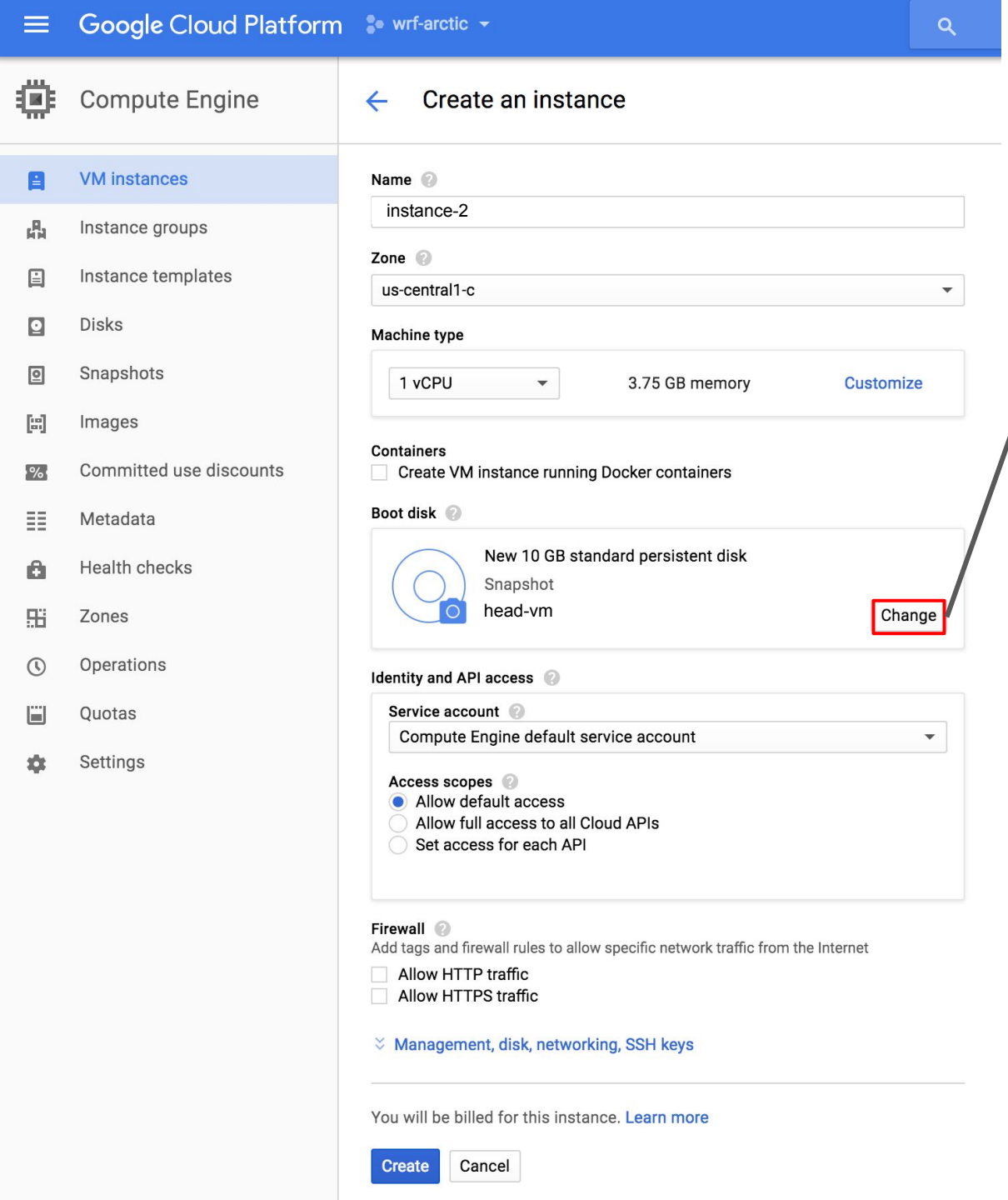
4. Start the NFS server (install exportfs and rpcbind, if necessary)

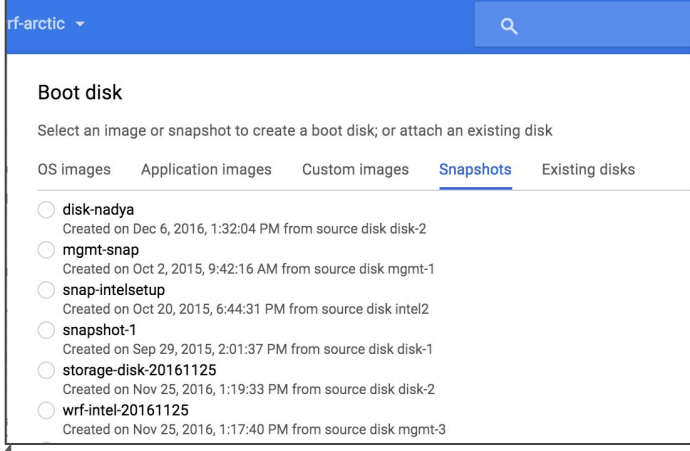
```
sudo /sbin/service rpcbind start
sudo /sbin/service nfs start
sudo exportfs -a
```

cd into `/mnt/disk` and install WRF here. Save this setup as a snapshot (next), and replicate it on another VM. The replicated VM is becomes the first CVM.


Google Cloud Platform
wrf-arctic
Compute Engine
VM instances
CREATE INSTANCE
IMPORT VM
REFRESH
START
STOP
RESET
DELETE
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Committed use discounts
Metadata
Health checks
Zones
Operations
Quotas
Settings
Filter by label or name
Columns
8 instances could be resized to save money or increase performance. Learn more
<< Previous
1
2
3
4
Next >>
Name
Zone
Recommendation
Internal IP
External IP
Connect
instance-1 us-central1-c 10.240.0.188 None SSH
intel001 us-central1-c 10.240.0.2 None SSH
intel002 us-central1-c 10.240.0.3 None SSH
intel003 us-central1-c 10.240.0.7 None SSH
intel004 us-central1-c 10.240.0.8 None SSH
intel005 us-central1-c 10.240.0.9 None SSH
intel006 us-central1-c 10.240.0.12 None SSH
intel007 us-central1-c 10.240.0.13 None SSH
intel008 us-central1-c 10.240.0.14 None SSH
intel009 us-central1-c 10.240.0.15 None SSH
intel010 us-central1-c 10.240.0.16 None SSH
intel011 us-central1-c 10.240.0.17 None SSH
intel012 us-central1-c 10.240.0.18 None SSH
intel013 us-central1-c 10.240.0.19 None SSH
intel014 us-central1-c 10.240.0.20 None SSH
Click on snapshots tab.


Google Cloud Platform
wrf-arctic
Compute Engine
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Committed use discounts
Metadata
Health checks
Zones
Operations
Quotas
Settings
Create a snapshot
Name
head-vm
Description (Optional)
Source disk
instance-1
Encryption
Automatic (recommended)
Integrate volume shadow copy service
Enable VSS
Create
Cancel
Equivalent REST or command line
This setup is saved as a snapshot of the HVM (labeled *head-vm* here). We use this to create the first CVM, next.





Loading the snapshot *head-vm* will replicate the saved contents of *instance-1* (the HVM) onto another VM (called *instance-2* here).

instance-2 is the first CVM.

NFS mount the storage disk from CVM (*instance-2*)

1. Edit auto.master (you may need to install autofs first)

   ```
   sudo vi /etc/auto.master
   ```

   Add:

   ```
   /nfs/hvm /etc/auto.hvm rsize=32678,wsize=32678,timeo=14,intr
   ```

2. Create auto.hvm

   ```
   sudo vi /etc/auto.hvm
   ```

   Add:

   ```
   disk -rw,nosuid 10.240.0.188:/mnt/disk
   ```

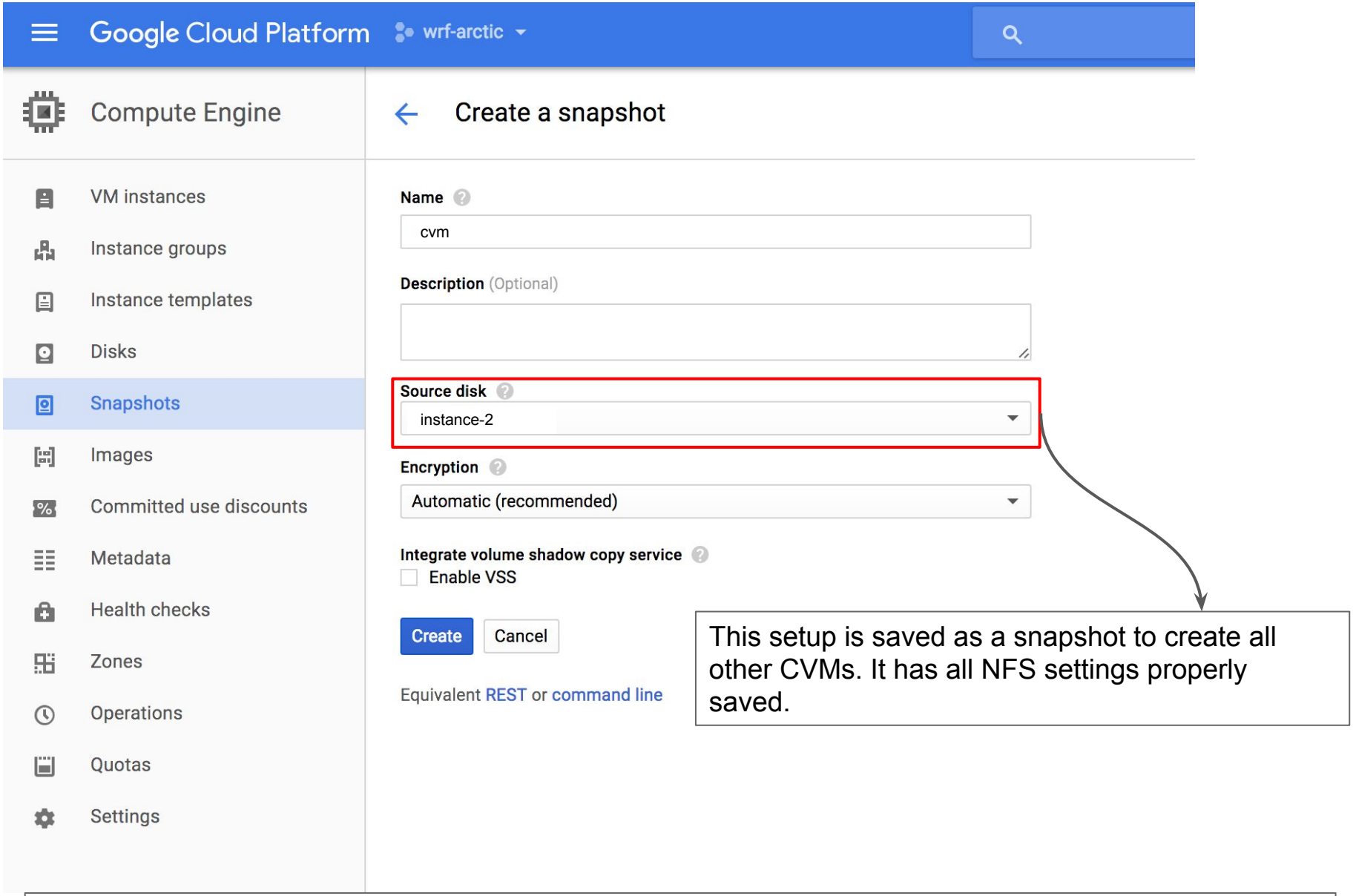
   10.240.0.188 is the local network address of the HVM, and /mnt/disk is the mount point on the HVM.

3. Restart autofs services

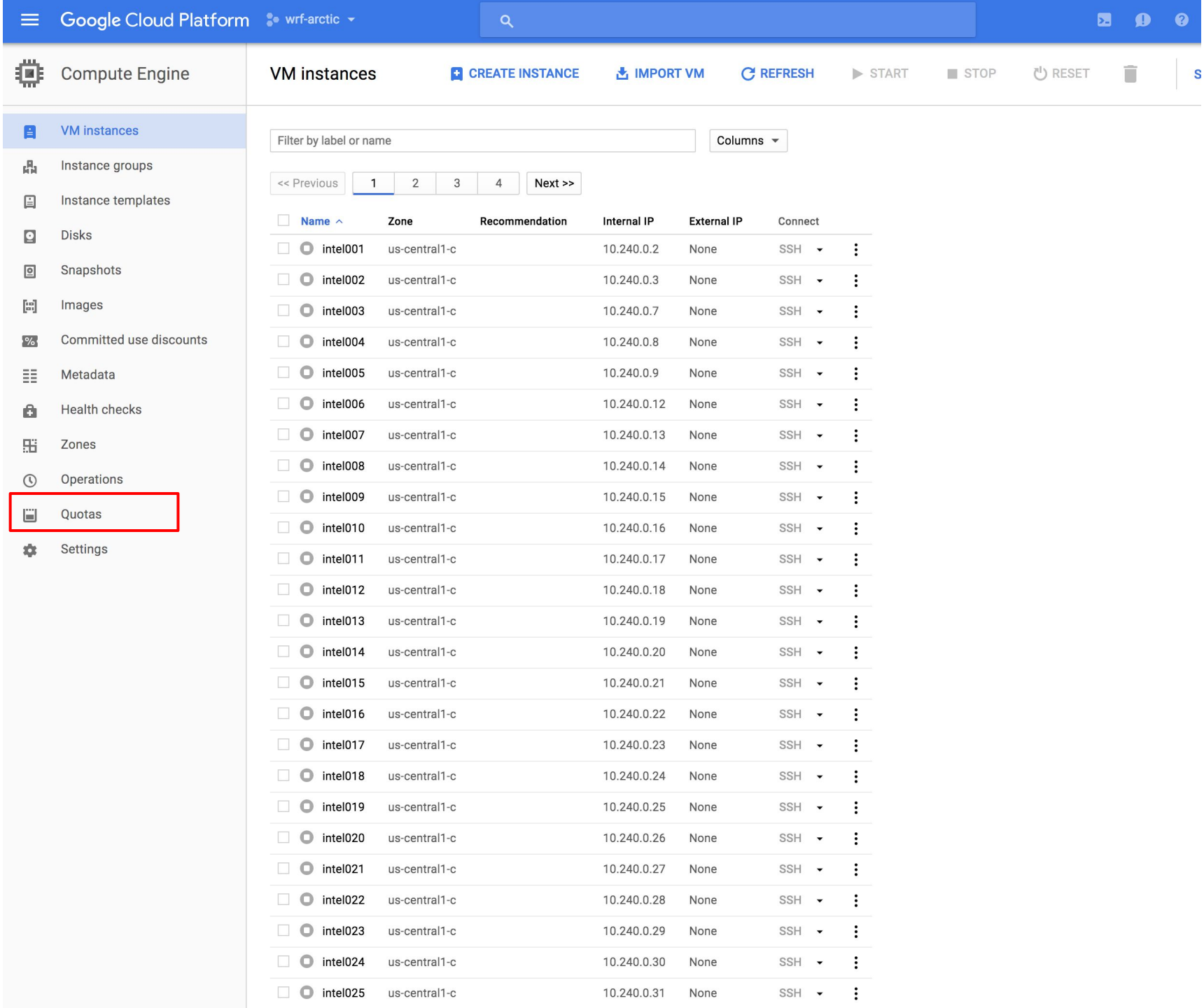
   ```
   sudo service autofs restart
   ```

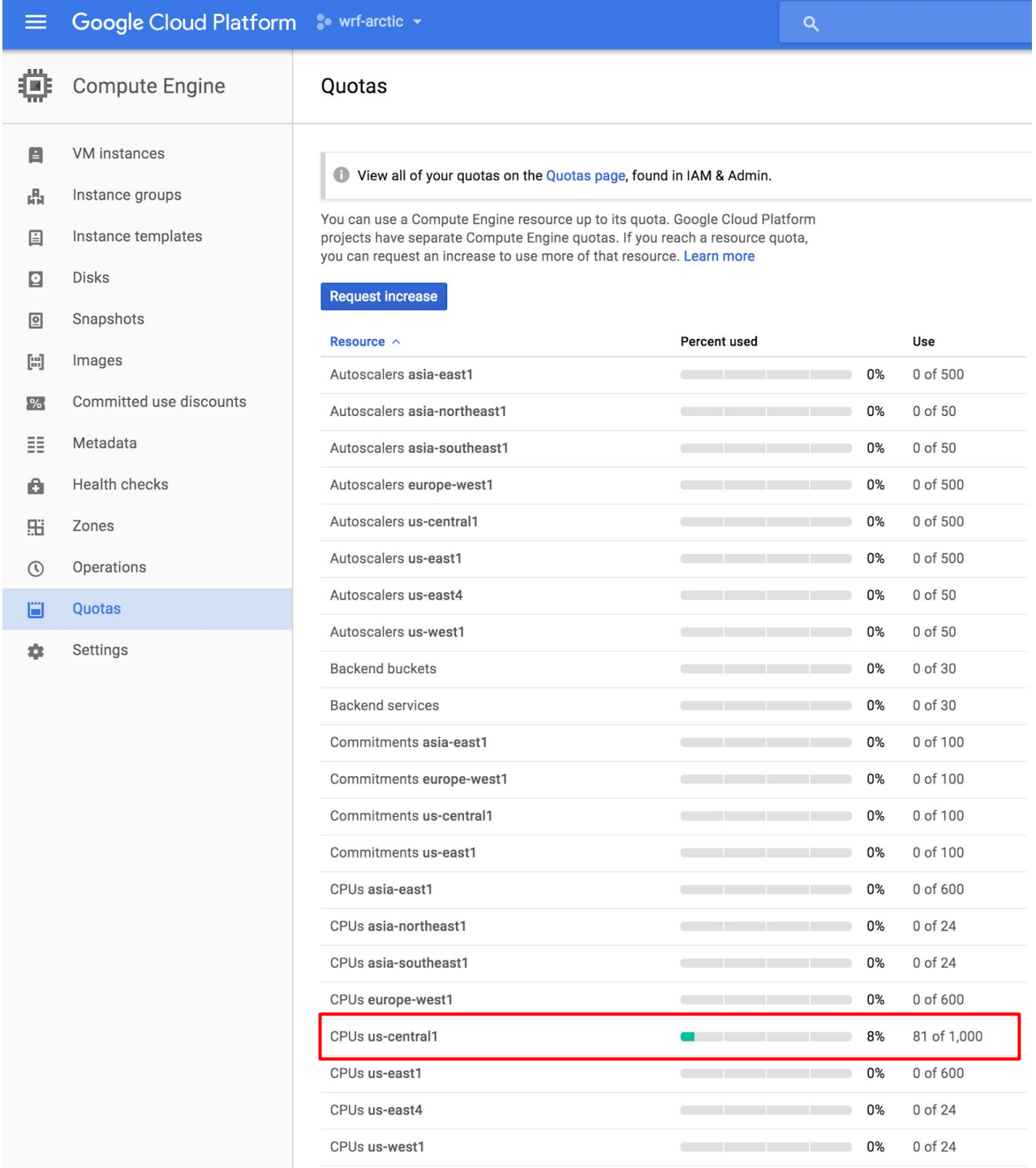
You should now be able to cd into `/nfs/hvm/disk`.

Save a snapshot of the CVM, replicate onto as many additional CVMs as needed. You *will not* need to repeat the NFS mounting when new CVMs are loaded.



After you create the desired amount of CVMs, ssh into each CVM from the HVM, then copy the *known_hosts* file from within ~/.ssh into each .ssh directory of every CVM. This allows mpi to work!


Google Cloud Platform
wrf-arctic
Compute Engine
VM instances
CREATE INSTANCE
IMPORT VM
REFRESH
START
STOP
RESET
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Committed use discounts
Metadata
Health checks
Zones
Operations
Quotas
Settings
Filter by label or name
Columns
<< Previous
1
2
3
4
Next >>
Name
Zone
Recommendation
Internal IP
External IP
Connect
intel001 us-central1-c 10.240.0.2 None SSH
intel002 us-central1-c 10.240.0.3 None SSH
intel003 us-central1-c 10.240.0.7 None SSH
intel004 us-central1-c 10.240.0.8 None SSH
intel005 us-central1-c 10.240.0.9 None SSH
intel006 us-central1-c 10.240.0.12 None SSH
intel007 us-central1-c 10.240.0.13 None SSH
intel008 us-central1-c 10.240.0.14 None SSH
intel009 us-central1-c 10.240.0.15 None SSH
intel010 us-central1-c 10.240.0.16 None SSH
intel011 us-central1-c 10.240.0.17 None SSH
intel012 us-central1-c 10.240.0.18 None SSH
intel013 us-central1-c 10.240.0.19 None SSH
intel014 us-central1-c 10.240.0.20 None SSH
intel015 us-central1-c 10.240.0.21 None SSH
intel016 us-central1-c 10.240.0.22 None SSH
intel017 us-central1-c 10.240.0.23 None SSH
intel018 us-central1-c 10.240.0.24 None SSH
intel019 us-central1-c 10.240.0.25 None SSH
intel020 us-central1-c 10.240.0.26 None SSH
intel021 us-central1-c 10.240.0.27 None SSH
intel022 us-central1-c 10.240.0.28 None SSH
intel023 us-central1-c 10.240.0.29 None SSH
intel024 us-central1-c 10.240.0.30 None SSH
intel025 us-central1-c 10.240.0.31 None SSH



You may find that you need to request more CPUs than the default allotted to a user account.

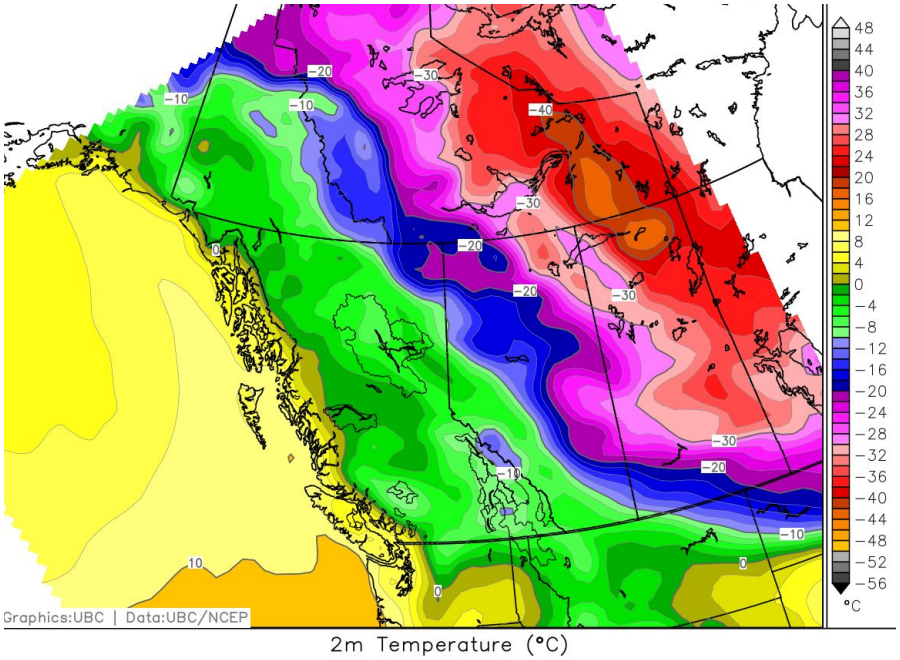
Given these results, UBC has extended our ensemble NWP suite onto the GCP since Oct. 2015.

Thank you! Questions?

David Siuta, Timothy Chui, Henryk Modzelewski, Greg West, Roland Schigas, and Roland Stull

email: dsiuta@eoas.ubc.ca
Postdoctoral Research Fellow
Weather Forecast Research Team
University of British Columbia
Vancouver, BC Canada
www.weather.eos.ubc.ca/wxfcst

Google Cloud Platform:
https://cloud.google.com/



Acknowledgements:

BC Hydro Power smart
MITACS Accelerate
NSERC CRSNG

Jasen Baker, Marcos Novaes, David Engelbrecht

Google Cloud Platform