

# GCP – VPCs and Compute Instances

---

## Introduction

In this article, we present two examples to show how VPC and Compute VM instances work in Google Cloud Platform (GCP)

Before we go into the examples, let's cover some basics

A few key concepts and terms used for GCP

## Network Terms and Concepts

- VPC – Virtual Private Cloud, a network entity that contains either one subnet in one region, multiple subnets in one region, or multiple subnets in multiple regions
- Network – a supernet that includes subnets. Subnets in the supernet may all reside in one region, or spread out in multiple regions.
- Subnet – Part of a network
- VPC and Network – The two terms are used interchangeably. In GCP web console, “VPC networks” is the place where networks are created and managed. In GCP command line interface (CLI) console, gcloud is used to create, update, or delete networks. gcloud requires the key word “networks”, not “VPCs”, to make such changes
- Default network – the network that GCP created for customer to use by default
- Custom network – a network created by customer when using GCP

A commonly adopted Internet standard

## RFC 1918

GCP networking supports IPv4. In GCP IPv4 networking, network and subnet use RFC 1918 private IP spaces for IP allocation. RFC 1918 defines three private IP spaces. Those spaces are listed in Table 1 below.

Table 1 – RFC 1918 Private IP Spaces

Starting IP	Netmask	Prefix
10.0.0.0	10.255.255.255	10/8
172.16.0.0	172.31.255.255	172.16/12
192.168.0.0	192.168.255.255	192.168/16

A network or a subnet specifies its IP range in a private IP space as its Classless Inter-Domain Routing (CIDR). CIDR is required when creating a subnet. CIDR is not required when creating a network. A network's CIDR is implied by its subnet CIDR(s).

---

## GCP – VPCs and Compute Instances

---

A few more details about GCP networks

### **Default Network**

The default network created by GCP has equal number of subnets to GCP regions. When GCP adds more regions, the number of subnets in the default network will increase correspondingly.

Currently GCP has 17 regions world-wide, 5 regions in US. Hence, the default network has 17 subnets

Table 2 – GCP Regions, Subnets in Default Network

Region	Type	CIDR	Gateway
asia-east1	Default	10.140.0.0/20	10.140.0.1
asia-northeast1	Default	10.146.0.0/20	10.146.0.1
asia-south1	Default	10.160.0.0/20	10.160.0.1
asia-southeast1	default	10.148.0.0/20	10.148.0.1
australia-southeast1	default	10.152.0.0/20	10.152.0.1
europa-north1	default	10.166.0.0/20	10.166.0.1
europa-west1	default	10.132.0.0/20	10.132.0.1
europa-west2	default	10.154.0.0/20	10.154.0.1
europa-west3	default	10.156.0.0/20	10.156.0.1
europa-west4	default	10.164.0.0/20	10.164.0.1
northamerica-northeast1	default	10.162.0.0/20	10.162.0.1
southamerica-east1	default	10.158.0.0/20	10.158.0.1
us-central1	default	10.128.0.0/20	10.128.0.1
us-east1	default	10.142.0.0/20	10.142.0.1
us-east4	default	10.150.0.0/20	10.150.0.1
us-west1	default	10.138.0.0/20	10.138.0.1
us-west2	default	10.168.0.0/20	10.168.0.1

As Table 2 shows above, each region has one subnet. The subnet, however, is not associated with any a zone within the region, though each region has multiple zones, typically 3, except us-central1 has 4 zones.

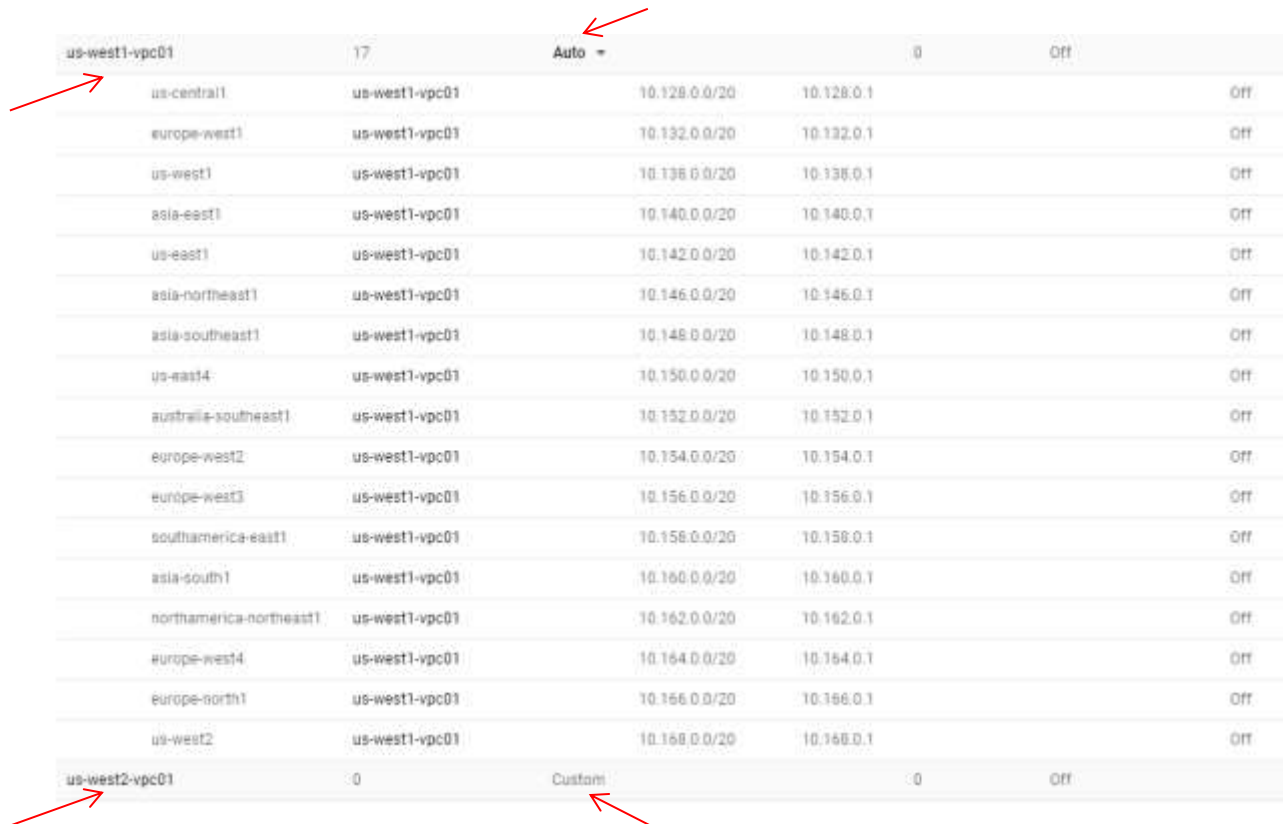
Attached to the default network are four default firewall rules. Those firewall rules permit ingress and internal traffic for multiple protocols such as SSH, RDP, and ICMP. The default network is intended to help customer get familiar with GCP environment once sign-up. Customer is expected to delete the default network and create their own custom networks before deploying applications.

## GCP – VPCs and Compute Instances

### Custom Network

A custom network is a non-default network. A custom network can be created differently to suite various needs. A custom network can be created in either “auto” mode or in “custom” mode. A custom network created in auto mode has 17 subnets in 17 regions, like the default network. A custom network created in custom mode can have one subnet or no subnet at all to begin with.

Figure 1 – Customer network created in auto mode and custom network created in custom mode



us-west1-vpc01	17	Auto	0	Off
us-central1	us-west1-vpc01	10.128.0.0/20	10.128.0.1	Off
europa-west1	us-west1-vpc01	10.132.0.0/20	10.132.0.1	Off
us-west1	us-west1-vpc01	10.136.0.0/20	10.136.0.1	Off
asia-east1	us-west1-vpc01	10.140.0.0/20	10.140.0.1	Off
us-east1	us-west1-vpc01	10.142.0.0/20	10.142.0.1	Off
asia-northeast1	us-west1-vpc01	10.146.0.0/20	10.146.0.1	Off
asia-southeast1	us-west1-vpc01	10.148.0.0/20	10.148.0.1	Off
us-east4	us-west1-vpc01	10.150.0.0/20	10.150.0.1	Off
australia-southeast1	us-west1-vpc01	10.152.0.0/20	10.152.0.1	Off
europa-west2	us-west1-vpc01	10.154.0.0/20	10.154.0.1	Off
europa-west3	us-west1-vpc01	10.156.0.0/20	10.156.0.1	Off
southamerica-east1	us-west1-vpc01	10.158.0.0/20	10.158.0.1	Off
asia-south1	us-west1-vpc01	10.160.0.0/20	10.160.0.1	Off
northamerica-northeast1	us-west1-vpc01	10.162.0.0/20	10.162.0.1	Off
europa-west4	us-west1-vpc01	10.164.0.0/20	10.164.0.1	Off
europa-north1	us-west1-vpc01	10.166.0.0/20	10.166.0.1	Off
us-west2	us-west1-vpc01	10.168.0.0/20	10.168.0.1	Off
us-west2-vpc01	0	Custom	0	Off

In Figure 1 above, us-west1-vpc01 was created in “auto” mode, resulting in 17 subnets in 17 regions, just as the default network does.

In Figure 1 above, us-west2-vpc01 was created in “custom” mode. It has no subnet.

Without subnets, a custom network

1. does not have its CIDR implied
2. does not have a region or regions associated with
3. VM instances cannot be launched into this empty custom network

In order for a custom network to be useful, subnets need to be added to it.

## GCP – VPCs and Compute Instances

Before adding a subnet, a RFC 1918 private IP space must be chosen, and the subnet's CIDR must be specified within that IP space. After the subnet is created and added to the network, the subnet's CIDR implies what the network's CIDR is, as the network's CIDR was not required when the network was created in the first place. More subnets can be added to the network as needed. All subnet CIDRs collectively define the network's CIDR inexplicitly. The custom network's CIDR must be larger or equal to the sum of all its subnet CIDRs.

What is the “implied” network CIDR in the case below?

**Table 3 - Custom Network us-central1-vpc01**

Region	Type	Subnet	CIDR	Gateway
us-central1	Custom	us-central1-vpc01-sub01	10.129.0.0/20	10.129.0.1
us-central1	Custom	us-central1-vpc01-sub02	10.129.16.0/20	10.129.16.1
us-central1	Custom	us-central1-vpc01-sub03	10.129.32.0/20	10.129.32.1
us-central1	Custom	us-central1-vpc01-sub04	10.129.48.0/20	10.129.48.1

**Figure 2 – Custom Network us-central1-vpc01 shown in GCP console**

us-central1-vpc01	4	Custom	5	Off
us-central1	us-central1-vpc01-sub01	10.129.0.0/20	10.129.0.1	Off
us-central1	us-central1-vpc01-sub02	10.129.16.0/20	10.129.16.1	Off
us-central1	us-central1-vpc01-sub03	10.129.32.0/20	10.129.32.1	Off
us-central1	us-central1-vpc01-sub04	10.129.48.0/20	10.129.48.1	Off

It is 10.129.0.0/18. The network in this case has four subnets. Each subnet had its CIDR specified when it was created. Together the four subnet CIDRs determine what the custom network's CIDR is. It is 10.129.0.0/18 because it is the minimum CIDR large enough to contain all four subnets. If no more subnets are added to the network, the network's CIDR stays as such. If more subnets are added to the network, the network's CIDR increases; conversely, decreases. In other words, the network CIDR changes in response to the number of subnets that it has, provided that

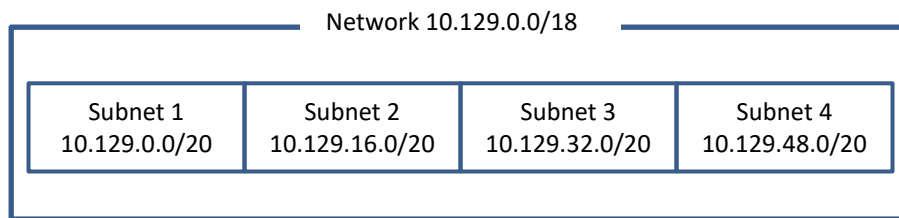
1. Subnet CIDRs are continuous
2. New subnet(s) are appended to the last one or
3. Last subnet(s), not middle ones, are removed

A diagram may help illustrate the relationship between subnet CIDRs and network CIDR

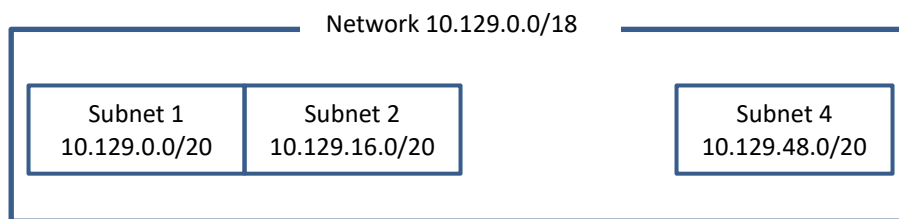
## GCP – VPCs and Compute Instances

---

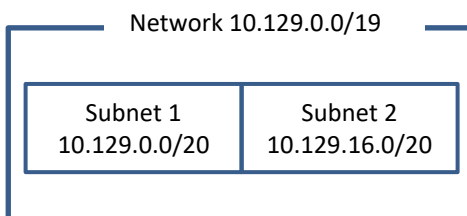
When the network has four subnets, they determine the network CIDR is 10.129.0.0/18



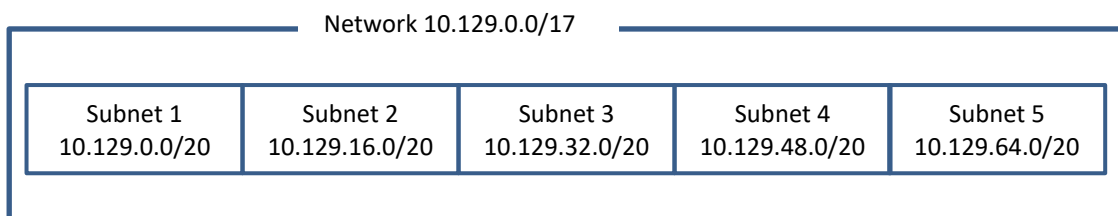
When the network has three subnets, with one removed from the middle, the network CIDR remains the same



When the network has two subnets, with last two removed, the network CIDR decreases from 10.129.0.0/18 to 10.129.0.0/19.



When the network has five subnets, with a new one appended to the last one, the network CIDR increases from 10.129.0.0/18 to 10.129.0.0/17.



IP resource in GCP can be used more efficiently this way than otherwise.

## GCP – VPCs and Compute Instances

---

A few words regarding GCP Console

### ***GCP Web Console and CLI Consoles***

All CSPs provide a web console for user to access services and perform admin tasks. GCP is of no exception. Yet, in GCP web console, user can open multiple command line interface (CLI) consoles to work on provisions, operations, monitoring and other tasks simultaneously, a feature not found in other CSP web consoles. A CLI console is essentially a Debian Linux OS shell with gcloud installed. gcloud is a powerful SDK for authorized user to interact with Compute Engine, App Engine and other GCP services. gcloud can be scripted. We used gcloud to install example networks and VM instances.

Now that we have visited enough basics, we are ready to move on to the examples.

---

## GCP – VPCs and Compute Instances

---

### Example 1 – Multi-Region VPC

The VPC spans across three regions, namely, us-central1, europe-west1, and asia-east1. The number of zones varies in the three regions.

**Table 4 – Zones and Selected Zones**

Region	Number of Zones	Zone selected to use
us-central1	4	us-central1-a
europe-west1	3	europe-west1-b
asia-east1	3	asia-east1-a

The procedure to create the network, instances, and perform network performance test follows.

The steps in the execution procedure are:

1. Preparation
  - a. Login GCP web console
  - b. Open a CLI Console in GCP web console
  - c. Create a project if not already done so
  - d. Verify that your login account is set to being the active account for authentication, and the project your created is set to being the core project
  - e. Verify gcloud is in your shell execution path
2. Create VPC and Subnets
3. Create Firewalls
4. Create instances
5. Test network performance among instances

The procedure execution details follow, with preparation skipped

### Creating VPC and Subnets

```
gcloud compute networks subnets create subnet-us-central --network dcl-custom-network --region us-central1 --range 10.0.0.0/16
gcloud compute networks subnets create subnet-europe-west --network dcl-custom-network --region europe-west1 --range 10.1.0.0/16
gcloud compute networks subnets create subnet-asia-east --network dcl-custom-network --region asia-east1 --range 10.2.0.0/16
```

### Creating Firewalls

```
gcloud compute firewall-rules create nw101-allow-http --allow tcp:80 --network dcl-custom-network --source-ranges 0.0.0.0/0 --target-tags http
gcloud compute firewall-rules create nw101-allow-icmp --allow icmp --network dcl-custom-network --target-tags rules
```

## GCP – VPCs and Compute Instances

---

```
gcloud compute firewall-rules create "nw101-allow-internal" --allow tcp:0-65535,udp:0-65535,icmp --network "dcl-custom-network" --source-ranges "10.0.0.0/16","10.2.0.0/16","10.1.0.0/16"
gcloud compute firewall-rules create "nw101-allow-ssh" --allow tcp:22 --network "dcl-custom-network" --target-tags "ssh"
gcloud compute firewall-rules create "nw101-allow-rdp" --allow tcp:3389 --network "dcl-custom-network" --target-tags "rdp"
```

### Creating Instances

```
gcloud compute instances create us-deb-01 --subnet subnet-us-central --zone us-central1-a --tags ssh,http,rules
gcloud compute instances create europe-deb-01 --subnet subnet-europe-west --zone europe-west1-b --tags ssh,http,rules
gcloud compute instances create asia-deb-01 --subnet subnet-asia-east --zone asia-east1-a --tags ssh,http,rules
```

### Connecting to instances

```
gcloud compute ssh us-deb01
Did you mean zone [us-central1-a] for instance: [us-deb01]
(Y/n)? Y
```

Warning: Permanently added 'compute.7207030016277738648' (ECDSA) to the list of known hosts.

Linux us-deb01 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) x86\_64

The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

tony\_shen@us-deb01:~\$

### Testing Performance

Login each instance, and install network utilities to be used for performance testing

```
sudo apt-get update
sudo apt-get -y install traceroute mtr tcpdump iperf whois host dnsutils siege
```

### Figure 3 – Traceroute results





[illegible]

Instance	Iperf client to iperf server us-deb-01 in Mbps	Performance assessment
us-deb-02	1,940	Fast
europe-deb-01	218	Slow
asia-east1	144	Slowest

## Creating VPC and subnets

---

Tony Shen
Page 10

## GCP – VPCs and Compute Instances

---

```
gcloud compute networks subnets create us-central1-vpc01-sub03 --network us-central1-vpc01
--region us-central1 --range 10.129.32.0/20
gcloud compute networks subnets create us-central1-vpc01-sub04 --network us-central1-vpc01
--region us-central1 --range 10.129.48.0/20
```

### Creating Firewalls

```
gcloud compute firewall-rules create nw101-allow-http --allow tcp:80 --network us-central1-
vpc01 --source-ranges 0.0.0.0/0 --target-tags http
gcloud compute firewall-rules create nw101-allow-icmp --allow icmp --network us-central1-
vpc01 --target-tags rules
gcloud compute firewall-rules create "nw101-allow-ssh" --allow tcp:22 --network "us-central1-
vpc01" --target-tags "ssh"
gcloud compute firewall-rules create "nw101-allow-rdp" --allow tcp:3389 --network "us-
central1-vpc01" --target-tags "rdp"
gcloud compute firewall-rules create "nw101-allow-internal" --allow tcp:0-65535,udp:0-
65535,icmp --network "us-central1-vpc01" --source-ranges
```

### Updating firewalls

```
tony_shen@silicon-will-206614:~$ gcloud compute firewall-rules update "nw101-allow-
internal" --source-ranges "10.129.0.0/20","10.129.16.0/20","10.129.32.0/20","10.129.48.0/20"
Updated [https://www.googleapis.com/compute/v1/projects/silicon-will-
206614/global/firewalls/nw101-allow-internal].
```

### Verifying firewalls

```
tony_shen@silicon-will-206614:~$ gcloud compute firewall-rules list "nw101-allow-internal" --
format=json
```

WARNING: Argument `NAME` is deprecated. Use `--filter="name=( 'NAME' ... )"` instead.

```
[
  {
    "allowed": [
      {
        "IPProtocol": "tcp",
        "ports": [
          "0-65535"
        ]
      },
      {
        "IPProtocol": "udp",
        "ports": [
          "0-65535"
        ]
      },
      {
        "IPProtocol": "icmp"
```

```
}
],
"creationTimestamp": "2018-08-01T16:47:19.465-07:00",
"description": "",
"direction": "INGRESS",
"id": "5601193382695139432",
"kind": "compute#firewall",
"name": "nw101-allow-internal",
"network": "https://www.googleapis.com/compute/v1/projects/silicon-will-206614/global/networks/us-central1-vpc01",
"priority": 1000,
"selfLink": "https://www.googleapis.com/compute/v1/projects/silicon-will-206614/global/firewalls/nw101-allow-internal",
"sourceRanges": [
  "10.129.0.0/20",
  "10.129.16.0/20",
  "10.129.32.0/20",
  "10.129.48.0/20"
]
}
]
tony_shen@silicon-will-206614:~$
```

### Creating instances

```
tony_shen@silicon-will-206614:~$ gcloud compute instances create us-central1-a-deb01 --
subnet us-central1-vpc01-sub01 --zone us-central1-a --tags ssh,http,rules
Created [https://www.googleapis.com/compute/v1/projects/silicon-will-206614/zones/us-
central1-a/instances/us-central1-a-deb01].
NAME          ZONE          MACHINE_TYPE  PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP
STATUS
us-central1-a-deb01  us-central1-a  n1-standard-1      10.129.0.2   35.225.107.211
RUNNING
```

```
tony_shen@silicon-will-206614:~$ gcloud compute instances create us-central1-b-deb01 --
subnet us-central1-vpc01-sub02 --zone us-central1-b --tags ssh,http,rules
Created [https://www.googleapis.com/compute/v1/projects/silicon-will-206614/zones/us-
central1-b/instances/us-central1-b-deb01].
NAME          ZONE          MACHINE_TYPE  PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP
STATUS
us-central1-b-deb01  us-central1-b  n1-standard-1      10.129.16.2  35.188.189.231
RUNNING
```

```
tony_shen@silicon-will-206614:~$ gcloud compute instances create us-central1-c-deb01 --
subnet us-central1-vpc01-sub03 --zone us-central1-c --tags ssh,http,rules
```

## GCP – VPCs and Compute Instances

---

Created [<https://www.googleapis.com/compute/v1/projects/silicon-will-206614/zones/us-central1-c/instances/us-central1-c-deb01>].

NAME	ZONE	MACHINE_TYPE	PREEMPTIBLE	INTERNAL_IP	EXTERNAL_IP
us-central1-c-deb01	us-central1-c	n1-standard-1		10.129.32.2	35.202.121.161
RUNNING					

```
tony_shen@silicon-will-206614:~$ gcloud compute instances create us-central1-f-deb01 --  
subnet us-central1-vpc01-sub04 --zone us-central1-f --tags ssh,http,rules
```

Created [<https://www.googleapis.com/compute/v1/projects/silicon-will-206614/zones/us-central1-f/instances/us-central1-f-deb01>].

NAME	ZONE	MACHINE_TYPE	PREEMPTIBLE	INTERNAL_IP	EXTERNAL_IP
us-central1-f-deb01	us-central1-f	n1-standard-1		10.129.48.2	35.232.226.209
RUNNING					

```
tony_shen@silicon-will-206614:~$
```

### Connecting to instances

```
tony_shen@silicon-will-206614:~$ gcloud compute ssh us-central1-a-deb01
```

Did you mean zone [us-central1-a] for instance: [us-central1-a-deb01]

(Y/n)? Y

Warning: Permanently added 'compute.7207030016277738648' (ECDSA) to the list of known hosts.

Linux us-central1-a-deb01 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) x86\_64

The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.

```
tony_shen@us-central1-a-deb01:~$
```

```
tony_shen@us-central1-a-deb01:~$ ssh us-central1-b-deb01
```

The authenticity of host 'us-central1-b-deb01 (10.129.16.2)' can't be established.

ECDSA key fingerprint is SHA256:M6lRRczHlCLXili8pLvp8jk44+noS4ri24jB+P8SU.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'us-central1-b-deb01,10.129.16.2' (ECDSA) to the list of known hosts.

Permission denied (publickey).

```
tony_shen@us-central1-a-deb01:~$
```

### Performance testing with ping

## GCP – VPCs and Compute Instances

---

Connected, host fingerprint: ssh-rsa 2048

7B:1B:9F:8C:79:74:BE:23:57:A0:D3:EA:5A:0A:4C:D4:CF:15:27:36:42:B2:36:77  
:42:5C:0F:AC:31:F3:3F:D7

Linux us-central1-c-deb01 4.9.0-7-amd64 #1 SMP Debian 4.9.110-1 (2018-07-05) x86\_64

The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.

tony\_shen@us-central1-c-deb01:~\$ ping -c3 us-central1-a-deb01

PING us-central1-a-deb01.c.silicon-will-206614.internal (10.129.0.2) 56(84) bytes of data.  
64 bytes from us-central1-a-deb01.c.silicon-will-206614.internal (10.129.0.2): icmp\_seq=1  
ttl=64 time=1.22 ms  
64 bytes from us-central1-a-deb01.c.silicon-will-206614.internal (10.129.0.2): icmp\_seq=2  
ttl=64 time=0.327 ms  
64 bytes from us-central1-a-deb01.c.silicon-will-206614.internal (10.129.0.2): icmp\_seq=3  
ttl=64 time=0.327 ms  
--- us-central1-a-deb01.c.silicon-will-206614.internal ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2005ms  
rtt min/avg/max/mdev = 0.327/0.624/1.220/0.421 ms

tony\_shen@us-central1-c-deb01:~\$ ping -c3 us-central1-b-deb01

PING us-central1-b-deb01.c.silicon-will-206614.internal (10.129.16.2) 56(84) bytes of data.  
64 bytes from us-central1-b-deb01.c.silicon-will-206614.internal (10.129.16.2): icmp\_seq=1  
ttl=64 time=1.62 ms  
64 bytes from us-central1-b-deb01.c.silicon-will-206614.internal (10.129.16.2): icmp\_seq=2  
ttl=64 time=0.327 ms  
64 bytes from us-central1-b-deb01.c.silicon-will-206614.internal (10.129.16.2): icmp\_seq=3  
ttl=64 time=0.347 ms  
--- us-central1-b-deb01.c.silicon-will-206614.internal ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2006ms  
rtt min/avg/max/mdev = 0.327/0.766/1.624/0.606 ms

tony\_shen@us-central1-c-deb01:~\$ ping -c3 us-central1-c-deb01

PING us-central1-c-deb01.c.silicon-will-206614.internal (10.129.32.2) 56(84) bytes of data.  
64 bytes from us-central1-c-deb01.c.silicon-will-206614.internal (10.129.32.2): icmp\_seq=1  
ttl=64 time=0.015 ms  
64 bytes from us-central1-c-deb01.c.silicon-will-206614.internal (10.129.32.2): icmp\_seq=2  
ttl=64 time=0.028 ms  
64 bytes from us-central1-c-deb01.c.silicon-will-206614.internal (10.129.32.2): icmp\_seq=3  
ttl=64 time=0.029 ms  
--- us-central1-c-deb01.c.silicon-will-206614.internal ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2044ms



## GCP – VPCs and Compute Instances

---

rtt min/avg/max/mdev = 0.015/0.024/0.029/0.006 ms

```
tony_shen@us-central1-c-deb01:~$ ping -c3 us-central1-f-deb01
PING us-central1-f-deb01.c.silicon-will-206614.internal (10.129.48.2) 56(84) bytes of data.
64 bytes from us-central1-f-deb01.c.silicon-will-206614.internal (10.129.48.2): icmp_seq=1
ttl=64 time=1.09 ms
64 bytes from us-central1-f-deb01.c.silicon-will-206614.internal (10.129.48.2): icmp_seq=2
ttl=64 time=0.464 ms
64 bytes from us-central1-f-deb01.c.silicon-will-206614.internal (10.129.48.2): icmp_seq=3
ttl=64 time=0.322 ms
--- us-central1-f-deb01.c.silicon-will-206614.internal ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 0.322/0.626/1.092/0.334 ms
tony_shen@us-central1-c-deb01:~$
```

### Performance testing with traceroute

```
tony_shen@us-central1-a-deb01:~$ traceroute www.icann.org
traceroute to www.icann.org (192.0.32.7), 30 hops max, 60 byte packets
 1 108.170.243.168 (108.170.243.168) 10.693 ms 108.170.243.247 (108.170.243.247) 10.806
ms 108.170.244.7 (108.170.244.7)
10.797 ms
 2 ae1.cr7-chi1.ip4.gtt.net (199.229.231.233) 10.785 ms * 11.046 ms
 3 xe-0-3-2.cr6-lax2.ip4.gtt.net (89.149.182.217) 46.776 ms 46.467 ms 46.470 ms
 4 ip4.gtt.net (69.174.9.218) 74.359 ms 74.345 ms 74.327 ms
 5 www.icann.org (192.0.32.7) 74.286 ms 74.289 ms 74.278 ms
tony_shen@us-central1-a-deb01:~$
```

```
tony_shen@us-central1-b-deb01:~$ traceroute www.icann.org
traceroute to www.icann.org (192.0.32.7), 30 hops max, 60 byte packets
 1 108.170.243.168 (108.170.243.168) 11.164 ms 108.170.244.7 (108.170.244.7) 11.155 ms
108.170.243.247 (108.170.243.247) 10.852 ms
 2 ae1.cr7-chi1.ip4.gtt.net (199.229.231.233) 10.852 ms * 10.637 ms
 3 xe-1-1-3.cr6-lax2.ip4.gtt.net (89.149.182.165) 46.159 ms 46.243 ms 46.127 ms
 4 ip4.gtt.net (69.174.9.218) 73.081 ms 73.093 ms 73.051 ms
 5 www.icann.org (192.0.32.7) 73.128 ms 73.112 ms 73.108 ms
tony_shen@us-central1-b-deb01:~$
```

### Performance testing with iperf

```
tony_shen@us-central1-b-deb01:~$ iperf -c us-central1-a-deb01
```

```
-----
Client connecting to us-central1-a-deb01, TCP port 5001
TCP window size: 45.0 KByte (default)
-----
```

```
[ 3] local 10.129.16.2 port 58708 connected with 10.129.0.2 port 5001
[ ID] Interval      Transfer    Bandwidth
```

## GCP – VPCs and Compute Instances

```
[ 3] 0.0-10.0 sec 2.26 GBytes 1.94 Gbits/sec
tony_shen@us-central1-b-deb01:~$
```

```
tony_shen@us-central1-c-deb01:~$ iperf -c us-central1-a-deb01
```

```
-----
Client connecting to us-central1-a-deb01, TCP port 5001
TCP window size: 45.0 KByte (default)
-----
```

```
[ 3] local 10.129.32.2 port 59176 connected with 10.129.0.2 port 5001
[ ID] Interval    Transfer    Bandwidth
[ 3] 0.0-10.0 sec 2.12 GBytes 1.82 Gbits/sec
tony_shen@us-central1-c-deb01:~$
```

```
tony_shen@us-central1-f-deb01:~$ iperf -c us-central1-a-deb01
```

```
-----
Client connecting to us-central1-a-deb01, TCP port 5001
TCP window size: 45.0 KByte (default)
-----
```

```
[ 3] local 10.129.48.2 port 51430 connected with 10.129.0.2 port 5001
[ ID] Interval    Transfer    Bandwidth
[ 3] 0.0-10.0 sec 2.22 GBytes 1.91 Gbits/sec
tony_shen@us-central1-f-deb01:~$
```

### Performance Testing Results Assessment

**Table 7 – traceroute results and performance assessment**

Instance	Traceroute to <a href="http://www.icann.org">www.icann.org</a> in ms	Performance assessment
us-central1-a-deb01	74	Fast
us-central1-b-deb01	72	Fast
us-central1-c-deb01	73	Fast
Us-central1-f-deb01	73	Fast

**Table 7 – iperf results and performance assessment**

Instance	Iperf client to iperf server us-centrral1-a-deb01 in Mbps	Performance assessment
us-central1-b-deb01	1,940	Fast
us-central1-c-deb01	1,820	Fast
Us-central1-f-deb01	1,910	Fast



### Conclusions

From these two examples we learn:

1. A VPC can include multiple regions world-wide. Pros and cons ought to be carefully evaluated when planning for a VPC to cover regions geographically dispersed, as network performance within such a VPC is to be uneven and may have undesired effects on workloads.
2. A VPC can include only one region with subnets distributed over all available zones. High and consistent network performance can be expected in such a VPC.
3. A VPC can be created empty without any subnets. An empty VPC hangs in the air without touching any region(s) and zone(s) on the ground
4. For a VPC to be useful, subnets have to be added. Yet, a VPC with subnets added is still not associated with regions and zones until instances are launched
5. When launching instance(s) in a VPC, three parameters are required
  - a. Subnet
  - b. A zone in a region
  - c. VPC

It is these three parameters that instances must have tie down VPC to zones and regions

6. Instances are the underpinnings of VPC to regions and zones. VPC planning, therefore, should start with instances. How many instances are needed to run workloads, where those instances should be running, how they should be networked, what traffic they will generate, and what network performance they can tolerate? Answers to these questions dictate what your VPC should look like
7. VPC's CIDR is implied by its subnet CIDRs. VPC's CIDR is elastic, its scope increases when subnets are appended to VPC, or decreases when last subnets are removed from VPC, assuming that subnet CIDRs are continuous. Keep VPC CIDRs as small as possible. Use IP ranges for subnet CIDRs in the VPC as much as possible without gaps. Unoccupied private IP spaces can then be used for expanding existing VPCs and creating new VPCs to accommodate growth

### Appendix - Browsing GCP Console

Figure 1 – Dashboard (1)

# GCP – VPCs and Compute Instances

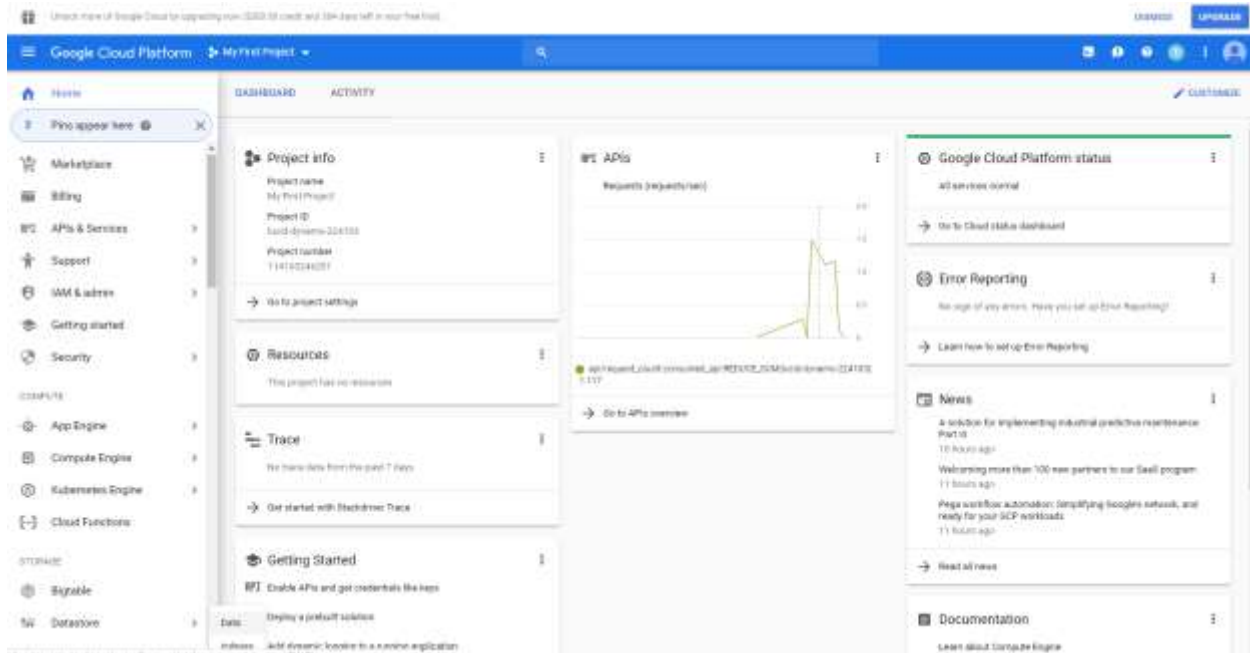


Figure 2 – Dashboard (2)

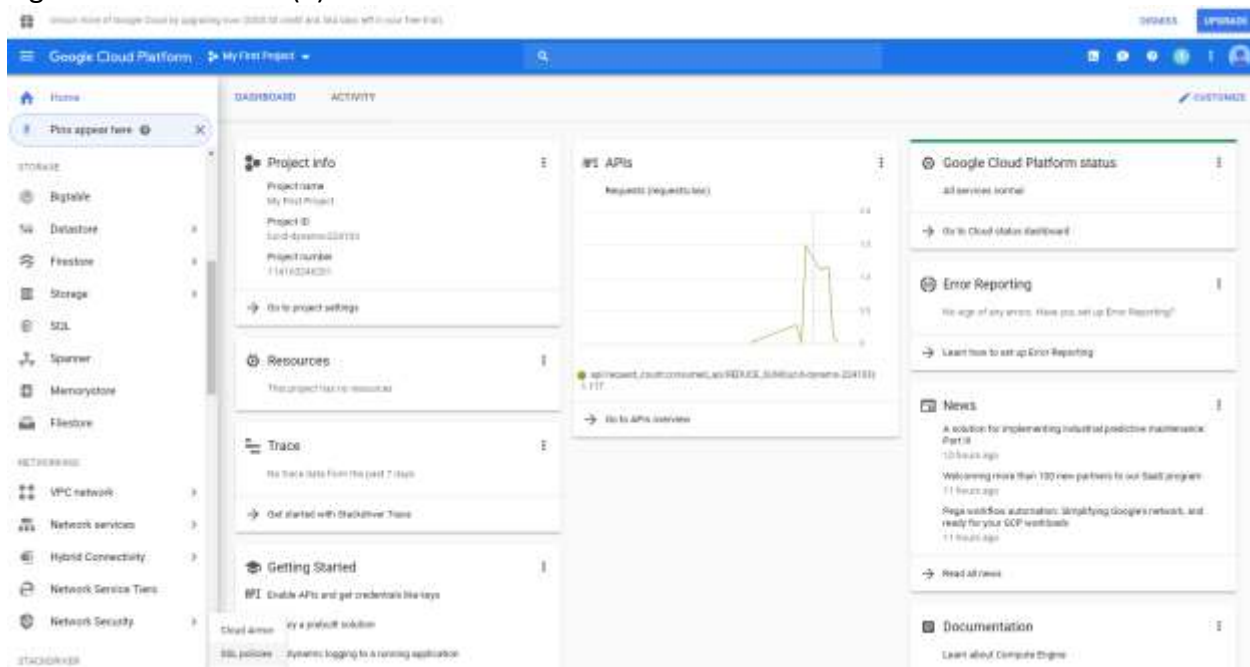


Figure 3 – Dashboard (3)

# GCP – VPCs and Compute Instances

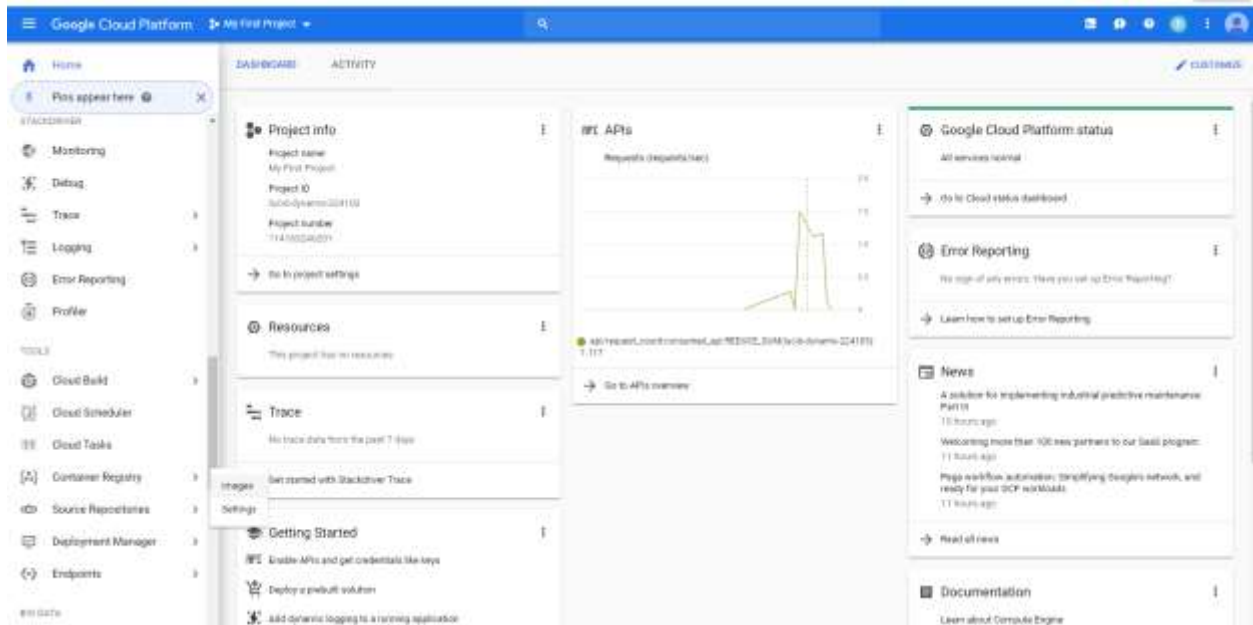


Figure 4 – Dashboard (4)

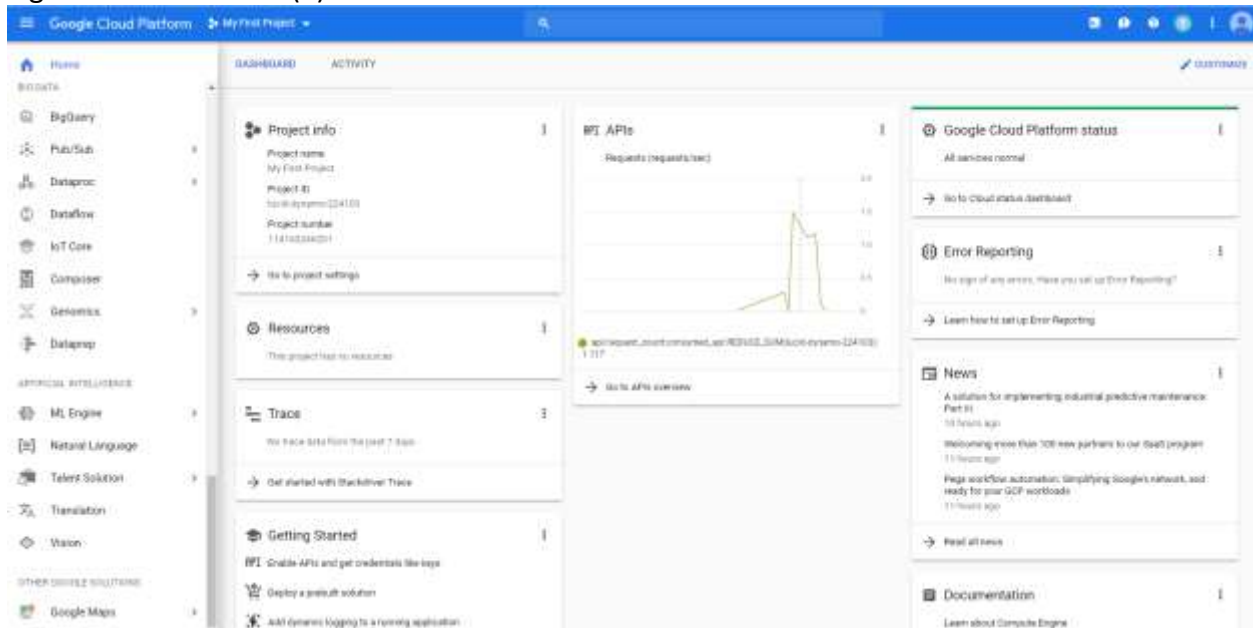


Figure 5 – Market Place

# GCP – VPCs and Compute Instances

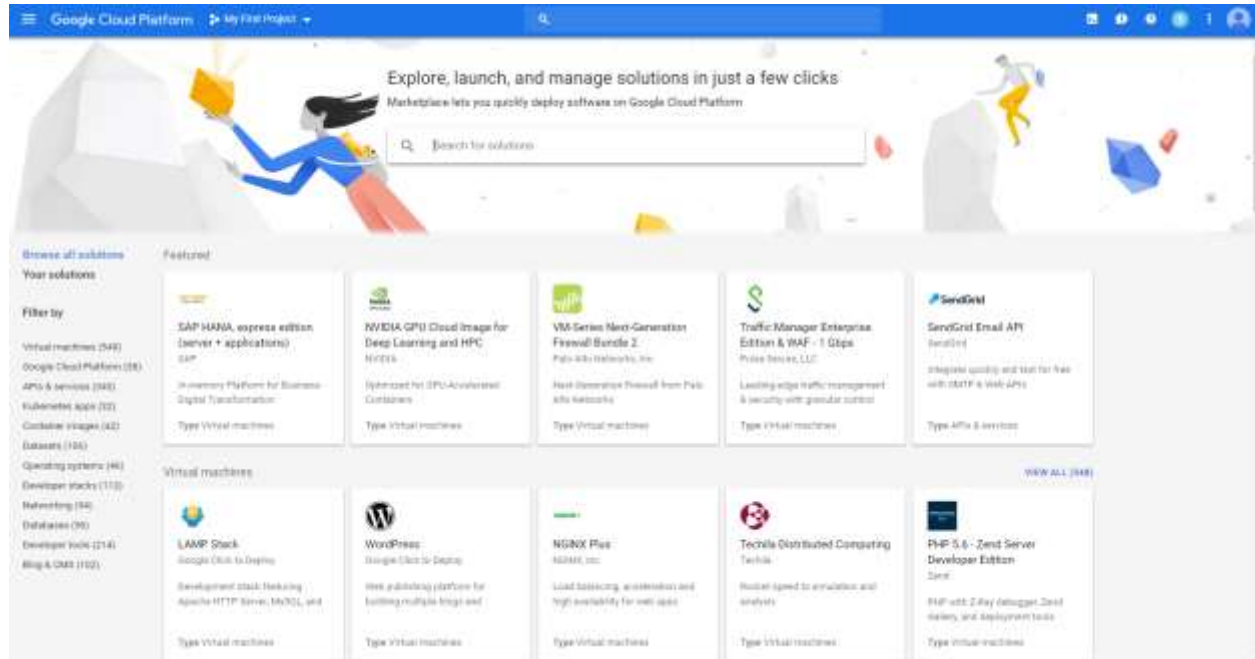


Figure 6 - Compute

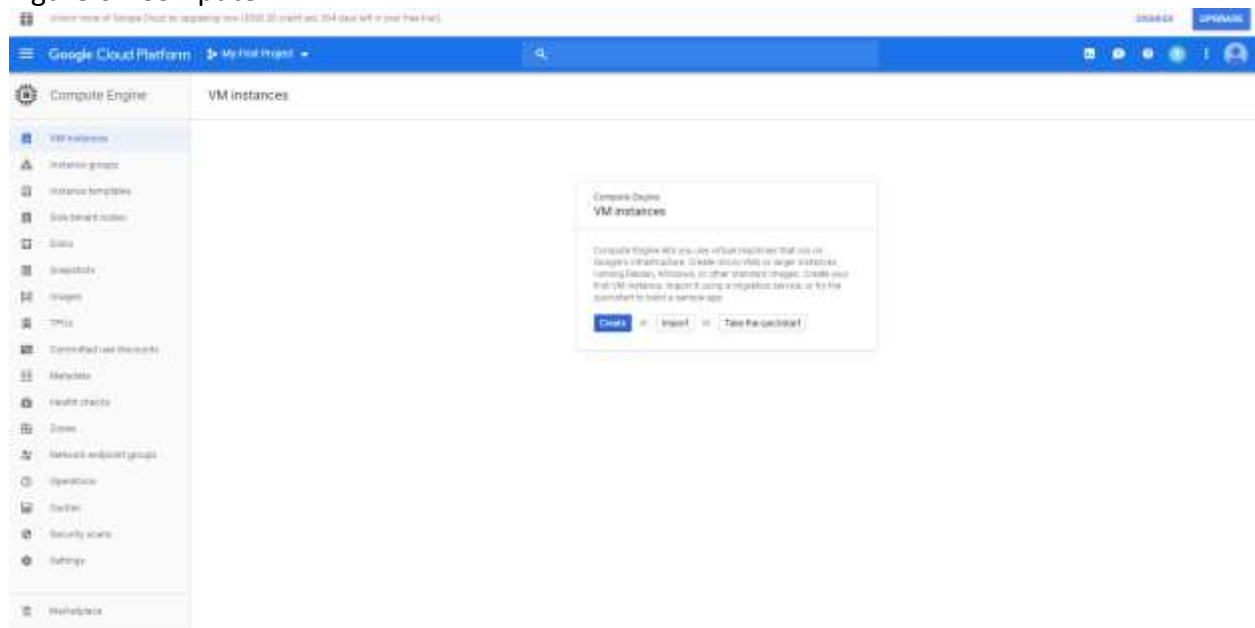


Figure 7 – Networks

---

\_\_\_\_\_

\_\_\_\_\_



Downloaded from <http://ajphaphysocpharm.sagepub.com/> at 11:01 11 November 2014



\_\_\_\_\_

## GCP – VPCs and Compute Instances

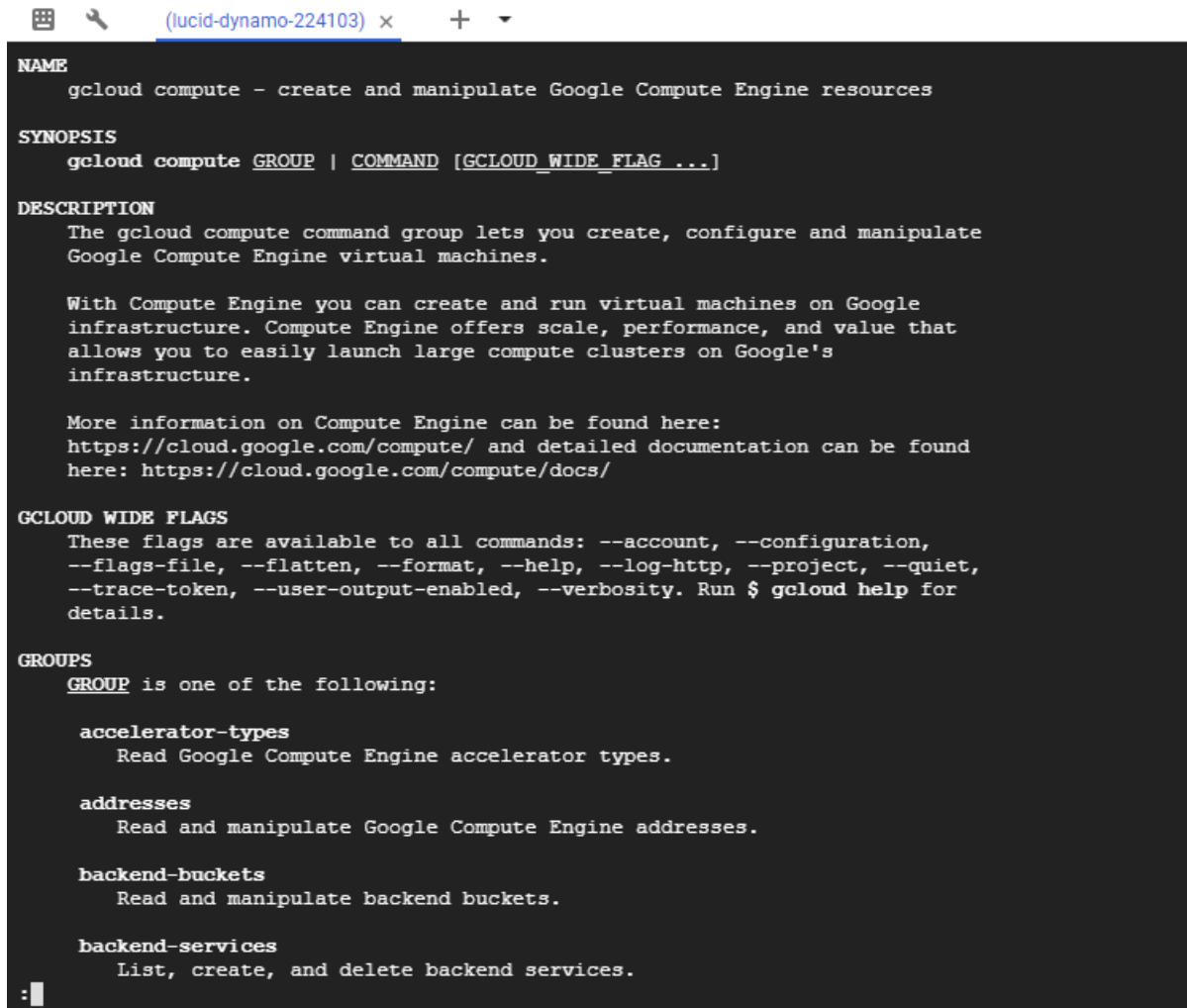
```
(lucid-dynamo-224103) x + ▾  
  
NAME  
gcloud - manage Google Cloud Platform resources and developer workflow  
  
SYNOPSIS  
gcloud GROUP | COMMAND [--account=ACCOUNT] [--configuration=CONFIGURATION]  
  [--flags-file=YAML_FILE] [--flatten=KEY,...] [--format=FORMAT]  
  [--help] [--project=PROJECT_ID] [--quiet, -q]  
  [--verbosity=VERBOSITY; default="warning"] [--version, -v] [-h]  
  [--log-http] [--trace-token=TRACE_TOKEN] [--no-user-output-enabled]  
  
DESCRIPTION  
The gcloud CLI manages authentication, local configuration, developer  
workflow, and interactions with the Google Cloud Platform APIs.  
  
GLOBAL FLAGS  
--account=ACCOUNT  
  Google Cloud Platform user account to use for invocation. Overrides the  
  default core/account property value for this command invocation.  
  
--configuration=CONFIGURATION  
  The configuration to use for this command invocation. For more  
  information on how to use configurations, run: gcloud topic  
  configurations. You can also use the [CLOUDSDK_ACTIVE_CONFIG_NAME]  
  environment variable to set the equivalent of this flag for a terminal  
  session.  
  
--flags-file=YAML_FILE  
  A YAML or JSON file that specifies a --flag:value dictionary. Useful  
  for specifying complex flag values with special characters that work  
  with any command interpreter. Additionally, each --flags-file arg is  
  replaced by its constituent flags. See $ gcloud topic flags-file for  
  more information.  
  
--flatten=KEY,...]  
  Flatten name[] output resource slices in KEY into separate records for  
  each item in each slice. Multiple keys and slices may be specified.  
  This also flattens keys for --format and --filter. For example,  
  --flatten=abc.def flattens abc.def[].ghi references to abc.def.ghi. A  
  resource record containing abc.def[] with N elements will expand to N  
:  
:
```

Figure 12 – gcloud top level groups

```
jane_shen2018@cloudshell:~ (lucid-dynamo-224103)$ gcloud somegroup  
ERROR: (gcloud) Invalid choice: 'somegroup'.  
Usage: gcloud [optional flags] <group | command>  
  group may be  
    alpha | app | auth | beta | bigtable | builds |  
    components | composer | compute | config | container |  
    dataflow | dataproc | datastore | debug |  
    deployment-manager | dns | domains | endpoints |  
    firebase | functions | iam | iot | kms | logging | ml |  
    ml-engine | organizations | projects | pubsub | redis |  
    services | source | spanner | sql | topic  
  command may be  
    docker | feedback | help | info | init | version  
  
For detailed information on this command and its flags, run:  
gcloud --help  
jane_shen2018@cloudshell:~ (lucid-dynamo-224103)$
```

Figure 13 – gcloud compute help

## GCP – VPCs and Compute Instances



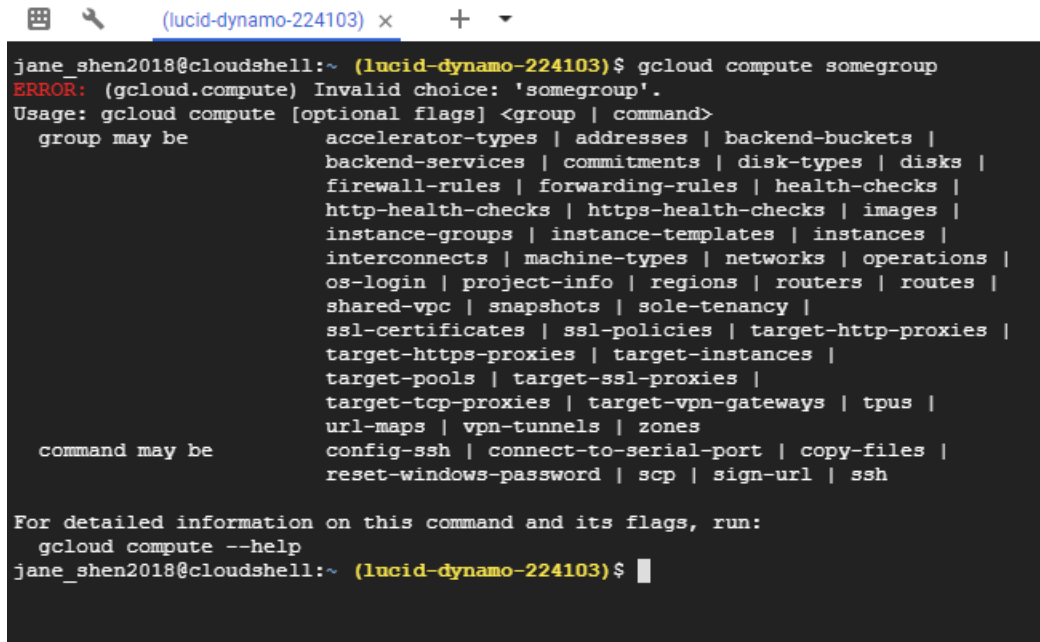
```
(lucid-dynamo-224103) x + ▾  
NAME  
gcloud compute - create and manipulate Google Compute Engine resources  
  
SYNOPSIS  
gcloud compute GROUP | COMMAND [G_CLOUD_WIDE_FLAG ...]  
  
DESCRIPTION  
The gcloud compute command group lets you create, configure and manipulate Google Compute Engine virtual machines.  
  
With Compute Engine you can create and run virtual machines on Google infrastructure. Compute Engine offers scale, performance, and value that allows you to easily launch large compute clusters on Google's infrastructure.  
  
More information on Compute Engine can be found here:  
https://cloud.google.com/compute/ and detailed documentation can be found here: https://cloud.google.com/compute/docs/  
  
G_CLOUD_WIDE_FLAGS  
These flags are available to all commands: --account, --configuration, --flags-file, --flatten, --format, --help, --log-http, --project, --quiet, --trace-token, --user-output-enabled, --verbosity. Run $ gcloud help for details.  
  
GROUPS  
GROUP is one of the following:  
  
  accelerator-types  
    Read Google Compute Engine accelerator types.  
  
  addresses  
    Read and manipulate Google Compute Engine addresses.  
  
  backend-buckets  
    Read and manipulate backend buckets.  
  
  backend-services  
    List, create, and delete backend services.  
:
```

Figure 14 – gcloud compute level groups



## GCP – VPCs and Compute Instances

---



```
jane_shen2018@cloudshell:~ (lucid-dynamo-224103) x + -
jane_shen2018@cloudshell:~ (lucid-dynamo-224103)$ gcloud compute somegroup
ERROR: (gcloud.compute) Invalid choice: 'somegroup'.
Usage: gcloud compute [optional flags] <group | command>
  group may be
    accelerator-types | addresses | backend-buckets |
    backend-services | commitments | disk-types | disks |
    firewall-rules | forwarding-rules | health-checks |
    http-health-checks | https-health-checks | images |
    instance-groups | instance-templates | instances |
    interconnects | machine-types | networks | operations |
    os-login | project-info | regions | routers | routes |
    shared-vpc | snapshots | sole-tenancy |
    ssl-certificates | ssl-policies | target-http-proxies |
    target-https-proxies | target-instances |
    target-pools | target-ssl-proxies |
    target-tcp-proxies | target-vpn-gateways | tpus |
    url-maps | vpn-tunnels | zones
  command may be
    config-ssh | connect-to-serial-port | copy-files |
    reset-windows-password | scp | sign-url | ssh

For detailed information on this command and its flags, run:
  gcloud compute --help
jane_shen2018@cloudshell:~ (lucid-dynamo-224103)$
```