DevOps Lab

CLOUD COMPUTE - GCP

NETWORKING

Home tasks

It's aiming to gain knowledge about Networking in Google Cloud.

TASK 1

Learn about two types of load balancers in Google Cloud Platform:

- a L3 Network Load Balancer and
- a L7 HTTP(s) Load Balancer.

Lab Link: codelabs: LoadBalancers

TASK 2

The Objectives are to learn:

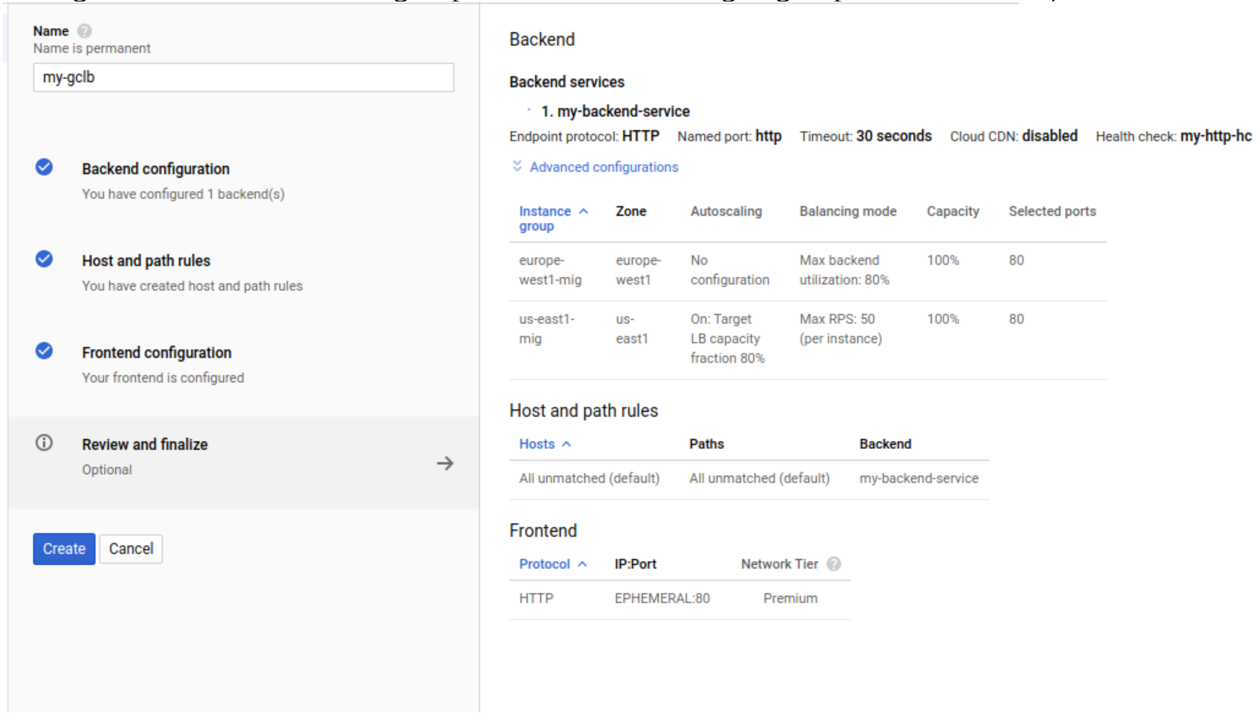
- How to measure latency between Google Compute Engine regions and zones
- How to test network connectivity and performance using open source tools
- How to set up up basic firewalling to secure your networks
- How to set up a global HTTP Load Balancer with Managed Instance Groups to automatically scale your resources up and down based on request load
- How to test and monitor your HTTP Load Balancer setup

These exercises are ordered to reflect a common cloud developer experience as follows:

1. Set up your lab environment and learn how to work with your GCP environment.
2. Use of common open source tools to explore your network around the world.
3. Deploy a common use case: use of HTTP Load Balancing and Managed Instance Groups to host a scalable, multi-region web server.
4. Testing and monitoring your network and instances.
5. Cleanup.

Lab Link: codelabs: Neworking 101

On screenshots below we can see HTTP(s) Load balancer with two configured backends (first of them - managed auto-scaled instance group, the second – managed group with 3 instance)



Name
Name is permanent
my-gclb

Backend configuration
You have configured 1 backend(s)

Host and path rules
You have created host and path rules

Frontend configuration
Your frontend is configured

Review and finalize
Optional

Backend

Backend services

1. my-backend-service

Endpoint protocol: **HTTP** Named port: **http** Timeout: **30 seconds** Cloud CDN: **disabled** Health check: **my-http-hc**

Advanced configurations

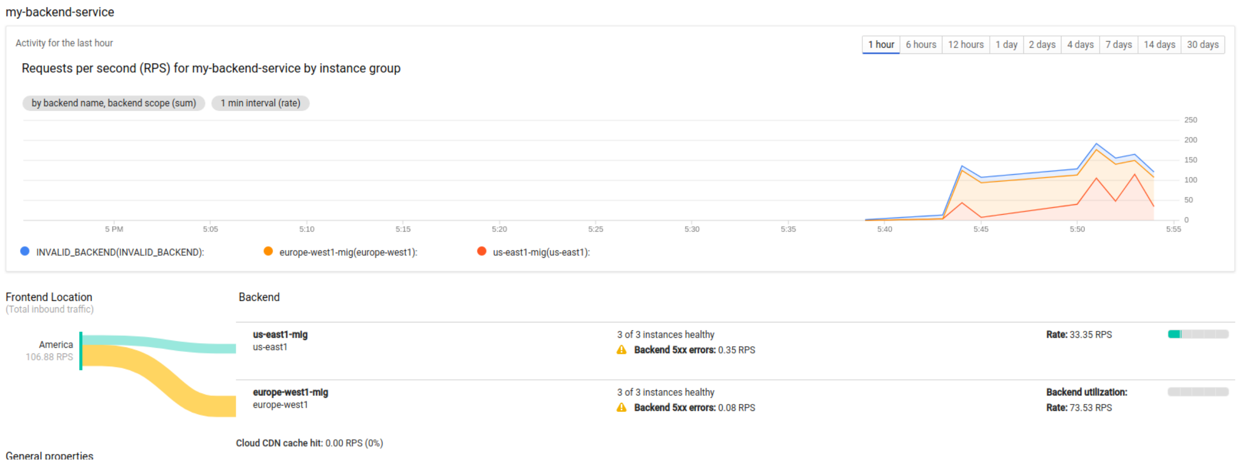
Instance group	Zone	Autoscaling	Balancing mode	Capacity	Selected ports
europe-west1-mig	europe-west1	No configuration	Max backend utilization: 80%	100%	80
us-east1-mig	us-east1	On: Target LB capacity fraction 80%	Max RPS: 50 (per instance)	100%	80

Host and path rules

Hosts	Paths	Backend
All unmatched (default)	All unmatched (default)	my-backend-service

Frontend

Protocol	IP:Port	Network Tier
HTTP	EPHEMERAL:80	Premium



my-backend-service

Activity for the last hour

Requests per second (RPS) for my-backend-service by instance group

by backend name, backend scope (sum) | 1 min interval (rate)

● INVALID_BACKEND(INVALID_BACKEND): ● europe-west1-mig(europe-west1): ● us-east1-mig(us-east1):

Frontend Location (Total inbound traffic)

America 106.88 RPS

Backend		
us-east1-mig us-east1	3 of 3 instances healthy ⚠ **Backend 5xx errors:** 0.35 RPS	**Rate:** 33.35 RPS
europe-west1-mig europe-west1	3 of 3 instances healthy ⚠ **Backend 5xx errors:** 0.08 RPS	**Backend utilization:** **Rate:** 73.53 RPS

Cloud CDN cache hit: 0.00 RPS (0%)

General properties

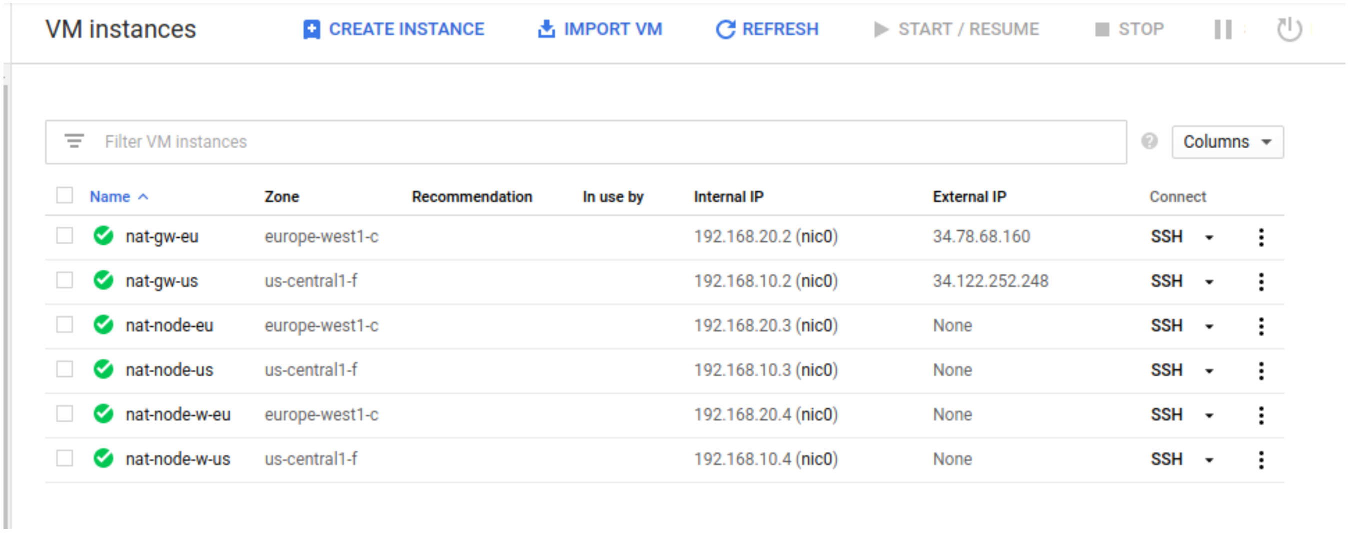
TASK 3

The Objectives are to learn:

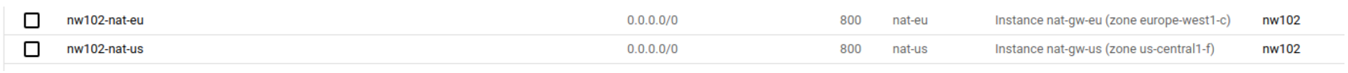
- Setting up NAT gateways
- How to restrict network traffic that certain tiers of an app cannot talk to each other
- Setting up alternate connectivity options to instances
- Map an external service to look like an internal service
- How to setup an Egress proxy limiting access to specific resources

Lab Link: [codelabs: Neworking 102]

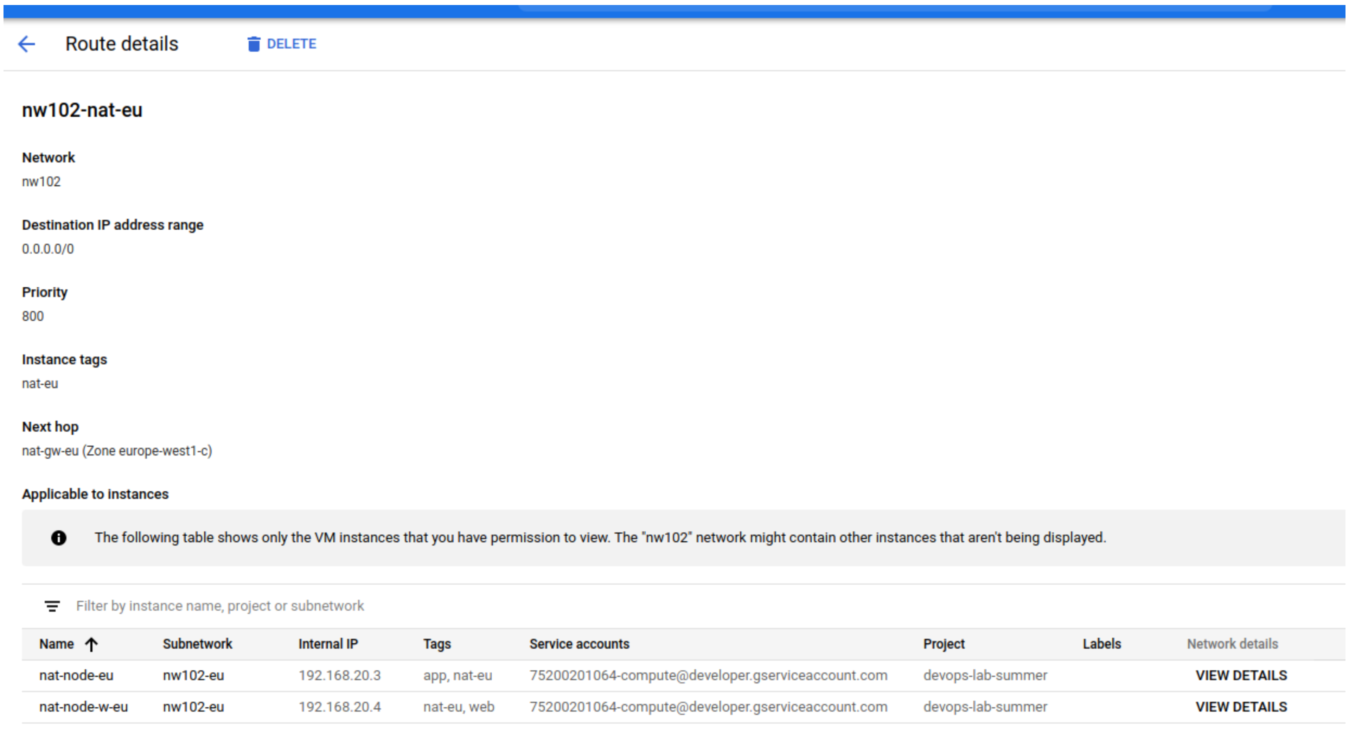
We can see our environment below. We have 2 NAT (nat-gw-eu, nat-gw-us), all traffic route with 2 route table. You will see that in screenshots below.



Name ^	Zone	Recommendation	In use by	Internal IP	External IP	Connect
✓ nat-gw-eu	europe-west1-c			192.168.20.2 (nic0)	34.78.68.160	SSH
✓ nat-gw-us	us-central1-f			192.168.10.2 (nic0)	34.122.252.248	SSH
✓ nat-node-eu	europe-west1-c			192.168.20.3 (nic0)	None	SSH
✓ nat-node-us	us-central1-f			192.168.10.3 (nic0)	None	SSH
✓ nat-node-w-eu	europe-west1-c			192.168.20.4 (nic0)	None	SSH
✓ nat-node-w-us	us-central1-f			192.168.10.4 (nic0)	None	SSH



☐	nw102-nat-eu	0.0.0.0/0	800	nat-eu	Instance nat-gw-eu (zone europe-west1-c)	nw102
☐	nw102-nat-us	0.0.0.0/0	800	nat-us	Instance nat-gw-us (zone us-central1-f)	nw102



Route details DELETE

nw102-nat-eu

Network
nw102

Destination IP address range
0.0.0.0/0

Priority
800

Instance tags
nat-eu

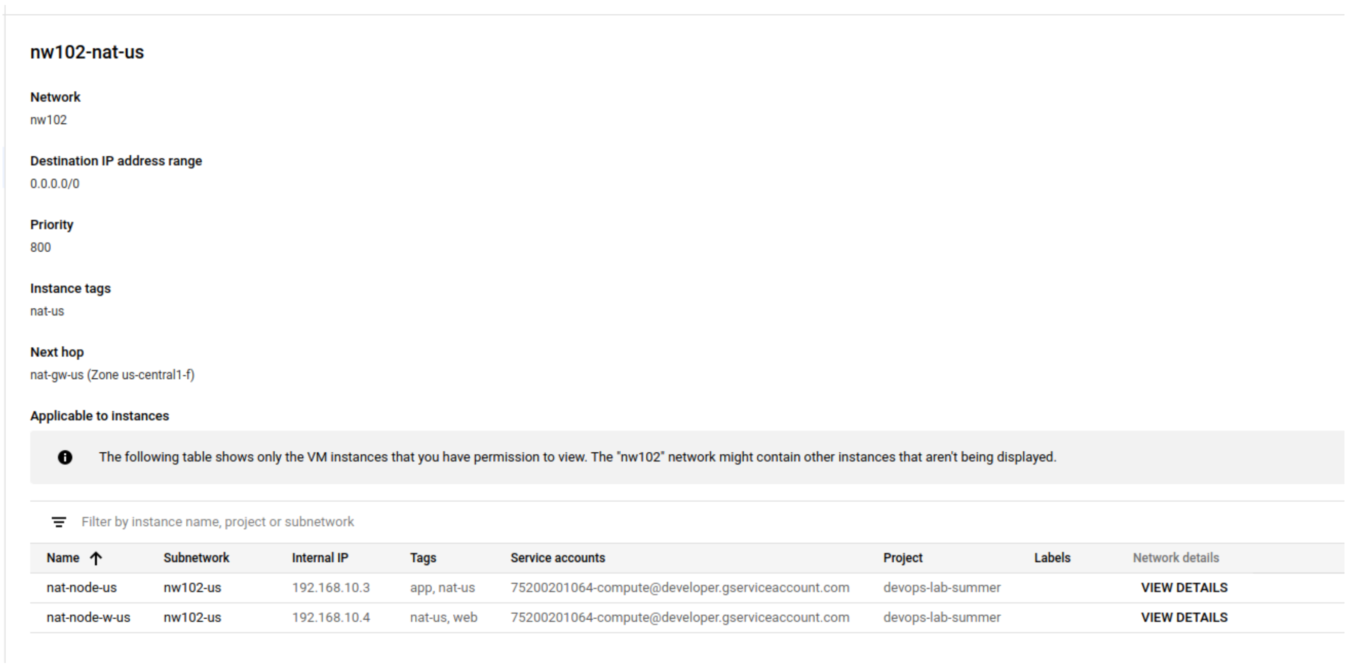
Next hop
nat-gw-eu (Zone europe-west1-c)

Applicable to instances

The following table shows only the VM instances that you have permission to view. The "nw102" network might contain other instances that aren't being displayed.

Filter by instance name, project or subnetwork

Name ↑	Subnetwork	Internal IP	Tags	Service accounts	Project	Labels	Network details
nat-node-eu	nw102-eu	192.168.20.3	app, nat-eu	75200201064-compute@developer.gserviceaccount.com	devops-lab-summer		VIEW DETAILS
nat-node-w-eu	nw102-eu	192.168.20.4	nat-eu, web	75200201064-compute@developer.gserviceaccount.com	devops-lab-summer		VIEW DETAILS



nw102-nat-us

Network
nw102

Destination IP address range
0.0.0.0/0

Priority
800

Instance tags
nat-us

Next hop
nat-gw-us (Zone us-central1-f)

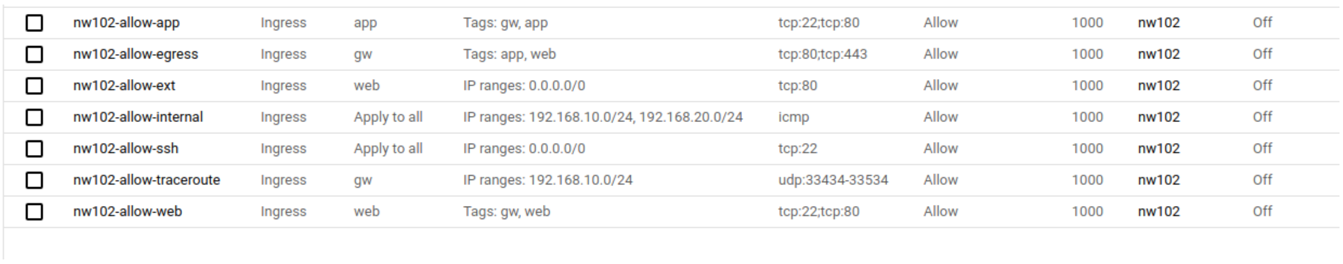
Applicable to instances

The following table shows only the VM instances that you have permission to view. The "nw102" network might contain other instances that aren't being displayed.

Filter by instance name, project or subnetwork

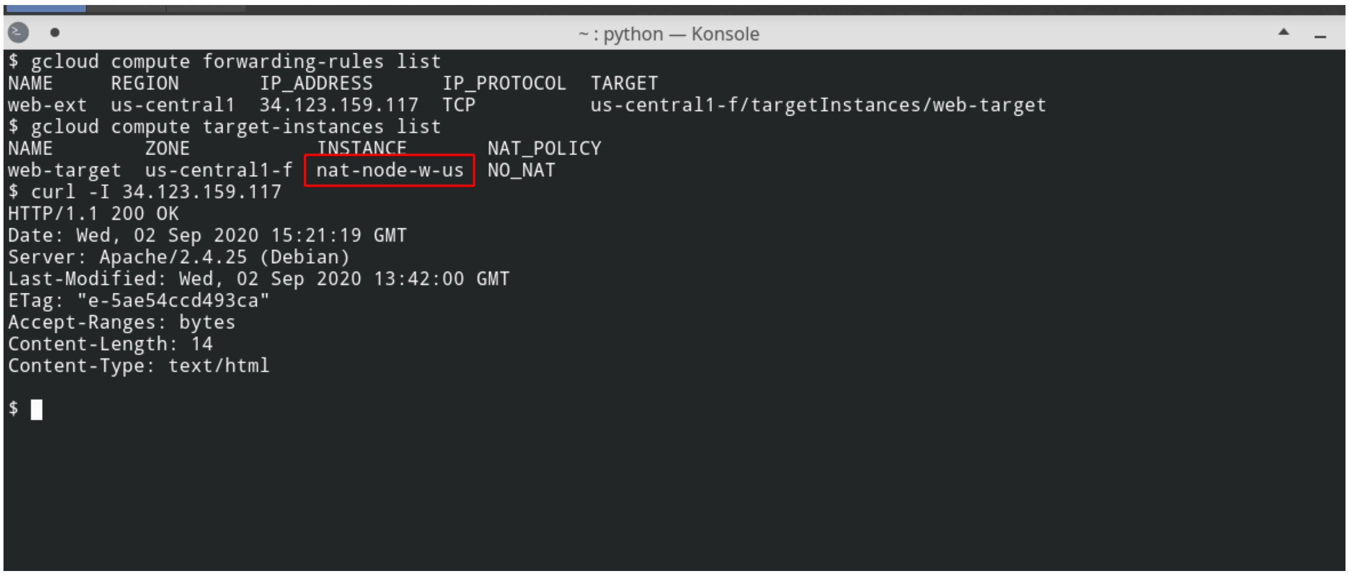
Name ↑	Subnetwork	Internal IP	Tags	Service accounts	Project	Labels	Network details
nat-node-us	nw102-us	192.168.10.3	app, nat-us	75200201064-compute@developer.gserviceaccount.com	devops-lab-summer		VIEW DETAILS
nat-node-w-us	nw102-us	192.168.10.4	nat-us, web	75200201064-compute@developer.gserviceaccount.com	devops-lab-summer		VIEW DETAILS

ALL network traffic restrict with custom created firewall rule.

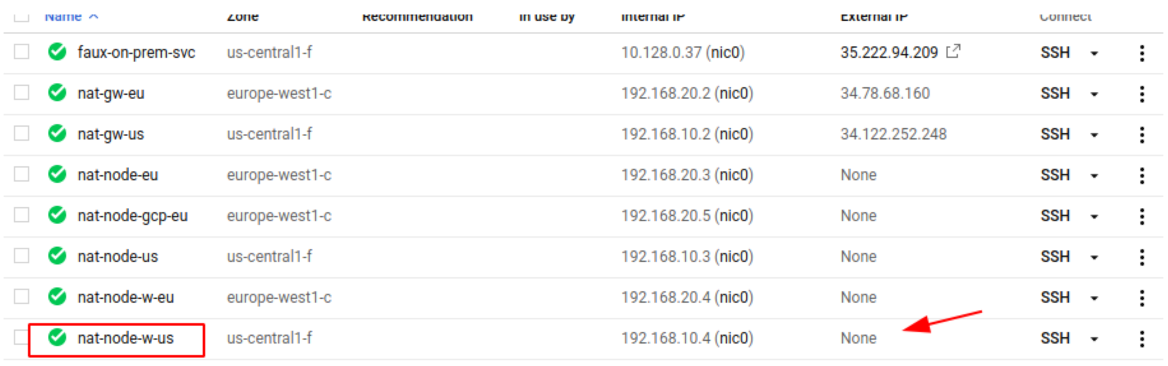


☐	nw102-allow-app	Ingress	app	Tags: gw, app	tcp:22;tcp:80	Allow	1000	nw102	Off
☐	nw102-allow-egress	Ingress	gw	Tags: app, web	tcp:80;tcp:443	Allow	1000	nw102	Off
☐	nw102-allow-ext	Ingress	web	IP ranges: 0.0.0.0/0	tcp:80	Allow	1000	nw102	Off
☐	nw102-allow-internal	Ingress	Apply to all	IP ranges: 192.168.10.0/24, 192.168.20.0/24	icmp	Allow	1000	nw102	Off
☐	nw102-allow-ssh	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:22	Allow	1000	nw102	Off
☐	nw102-allow-traceroute	Ingress	gw	IP ranges: 192.168.10.0/24	udp:33434-33534	Allow	1000	nw102	Off
☐	nw102-allow-web	Ingress	web	Tags: gw, web	tcp:22;tcp:80	Allow	1000	nw102	Off

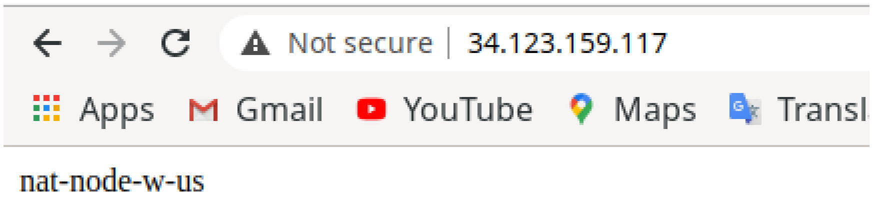
One of the option to alternative connect to VM instance – that expose internal services via forwarding rule instead of the standard external IP.



```
$ gcloud compute forwarding-rules list
NAME     REGION       IP_ADDRESS      IP_PROTOCOL  TARGET
web-ext  us-central1  34.123.159.117  TCP          us-central1-f/targetInstances/web-target
$ gcloud compute target-instances list
NAME        ZONE           INSTANCE       NAT_POLICY
web-target  us-central1-f  nat-node-w-us  NO_NAT
$ curl -I 34.123.159.117
HTTP/1.1 200 OK
Date: Wed, 02 Sep 2020 15:21:19 GMT
Server: Apache/2.4.25 (Debian)
Last-Modified: Wed, 02 Sep 2020 13:42:00 GMT
ETag: "e-5ae54ccd493ca"
Accept-Ranges: bytes
Content-Length: 14
Content-Type: text/html

$ 
```



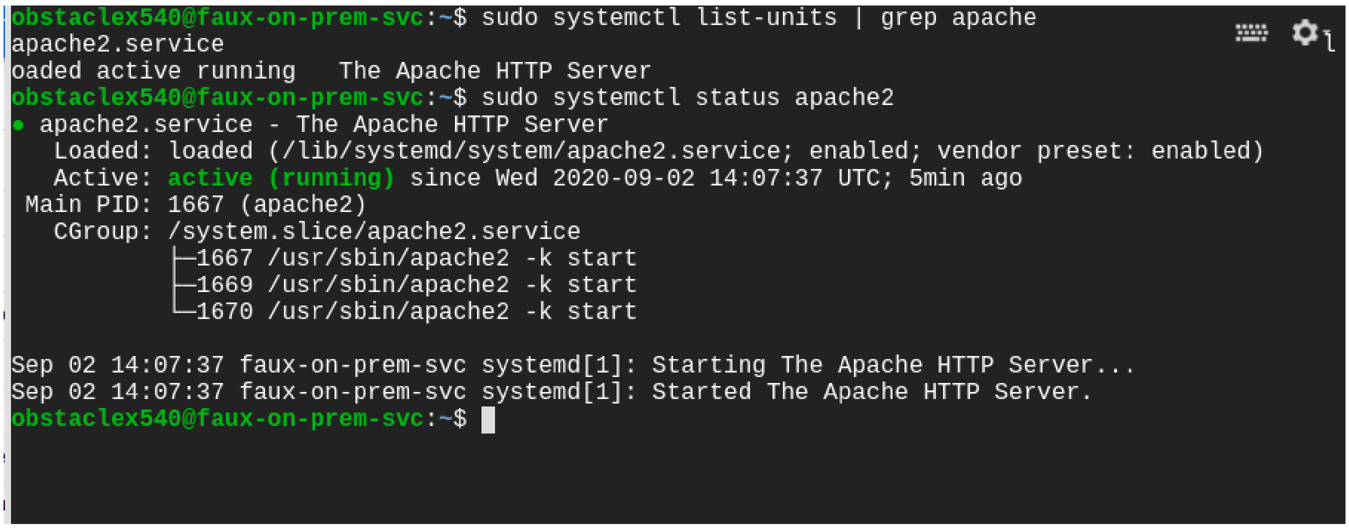
Name ^	Zone	Recommendation	In use by	Internal IP	External IP	Connect
faux-on-prem-svc	us-central1-f			10.128.0.37 (nic0)	35.222.94.209	SSH
nat-gw-eu	europe-west1-c			192.168.20.2 (nic0)	34.78.68.160	SSH
nat-gw-us	us-central1-f			192.168.10.2 (nic0)	34.122.252.248	SSH
nat-node-eu	europe-west1-c			192.168.20.3 (nic0)	None	SSH
nat-node-gcp-eu	europe-west1-c			192.168.20.5 (nic0)	None	SSH
nat-node-us	us-central1-f			192.168.10.3 (nic0)	None	SSH
nat-node-w-eu	europe-west1-c			192.168.20.4 (nic0)	None	SSH
nat-node-w-us	us-central1-f			192.168.10.4 (nic0)	None	SSH



nat-node-w-us

Map an external service through an internal IP.

We have installed apache on faux-on-prem-svc instance

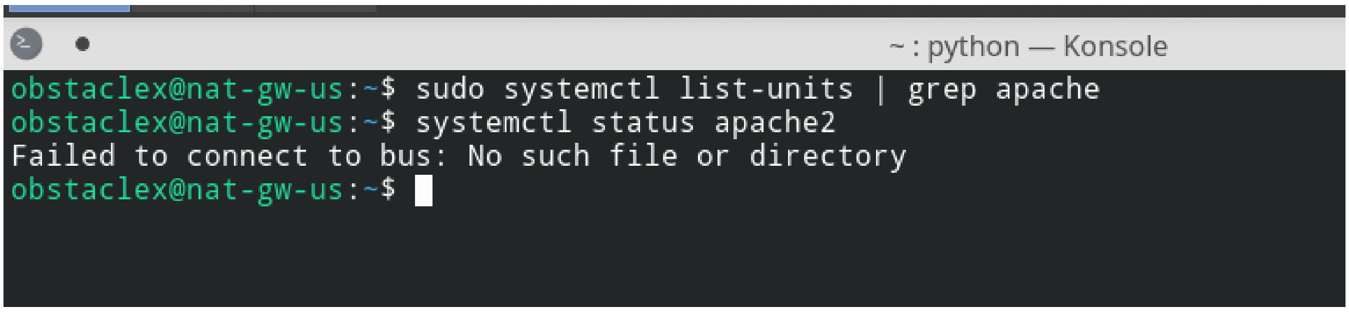


```
obstaclex540@faux-on-prem-svc:~$ sudo systemctl list-units | grep apache
apache2.service
oaded active running   The Apache HTTP Server
obstaclex540@faux-on-prem-svc:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2020-09-02 14:07:37 UTC; 5min ago
 Main PID: 1667 (apache2)
   CGroup: /system.slice/apache2.service
           ├─1667 /usr/sbin/apache2 -k start
           ├─1669 /usr/sbin/apache2 -k start
           └─1670 /usr/sbin/apache2 -k start

Sep 02 14:07:37 faux-on-prem-svc systemd[1]: Starting The Apache HTTP Server...
Sep 02 14:07:37 faux-on-prem-svc systemd[1]: Started The Apache HTTP Server.
obstaclex540@faux-on-prem-svc:~$ 
```

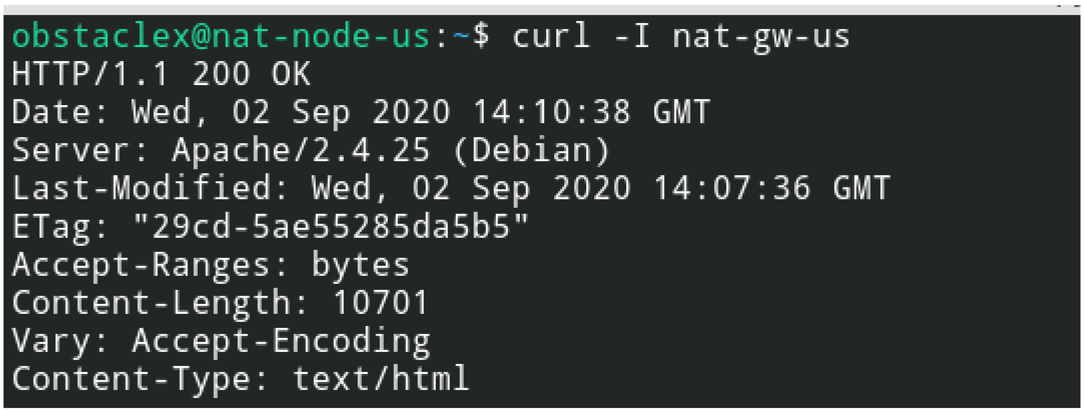
After use follow commands we mapped external service through an internal NAT-gw IP.

sudo iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j DNAT --to <faux-on-prem-svc-external-ip>:80

sudo iptables -A POSTROUTING -t nat -o eth0 -j SNAT --to-source <nat-gw-us-internal-ip>



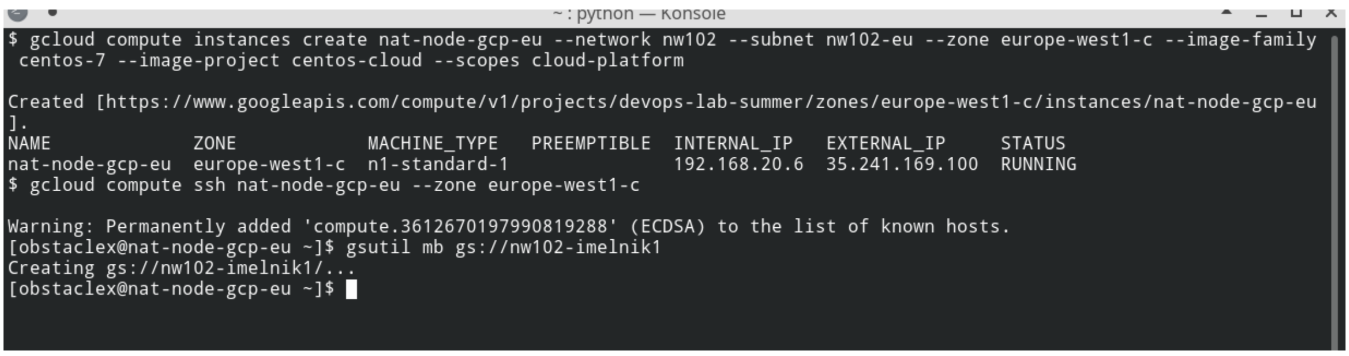
```
obstaclex@nat-gw-us:~$ sudo systemctl list-units | grep apache
obstaclex@nat-gw-us:~$ systemctl status apache2
Failed to connect to bus: No such file or directory
obstaclex@nat-gw-us:~$
```



```
obstaclex@nat-node-us:~$ curl -I nat-gw-us
HTTP/1.1 200 OK
Date: Wed, 02 Sep 2020 14:10:38 GMT
Server: Apache/2.4.25 (Debian)
Last-Modified: Wed, 02 Sep 2020 14:07:36 GMT
ETag: "29cd-5ae55285da5b5"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Content-Type: text/html
```

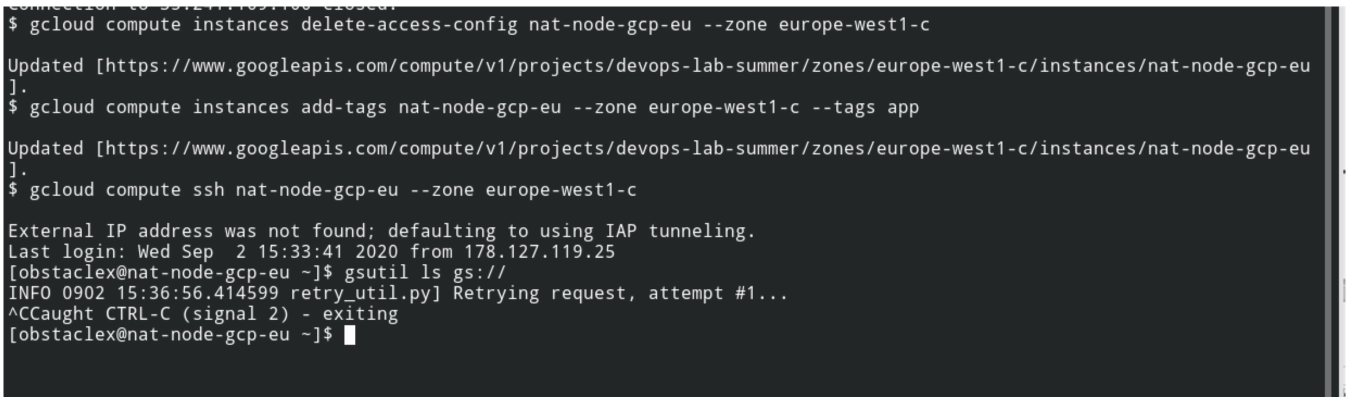
Setup an Egress proxy limiting access to specific resources

We are create a new VM with the full access scope to Compute Engine.



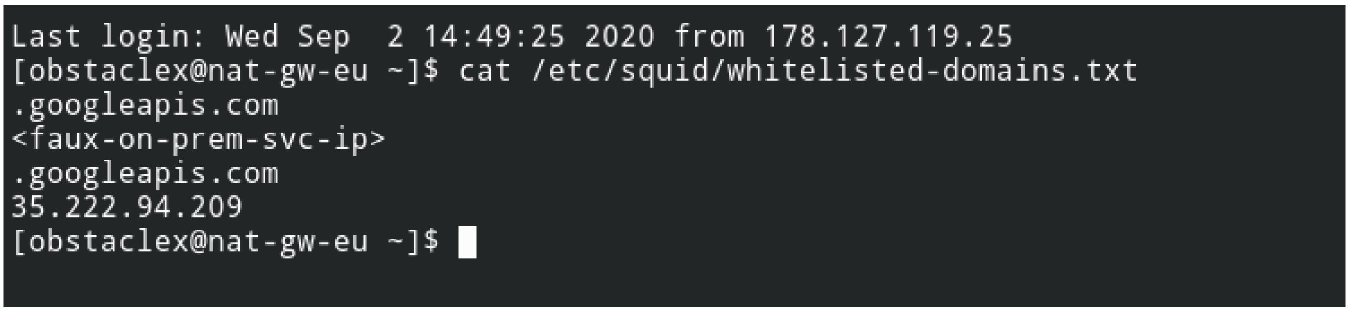
```
$ gcloud compute instances create nat-node-gcp-eu --network nw102 --subnet nw102-eu --zone europe-west1-c --image-family
 centos-7 --image-project centos-cloud --scopes cloud-platform

Created [https://www.googleapis.com/compute/v1/projects/devops-lab-summer/zones/europe-west1-c/instances/nat-node-gcp-eu
].
NAME             ZONE            MACHINE_TYPE   PREEMPTIBLE  INTERNAL_IP   EXTERNAL_IP     STATUS
nat-node-gcp-eu  europe-west1-c  n1-standard-1               192.168.20.6  35.241.169.100  RUNNING
$ gcloud compute ssh nat-node-gcp-eu --zone europe-west1-c

Warning: Permanently added 'compute.3612670197990819288' (ECDSA) to the list of known hosts.
[obstaclex@nat-node-gcp-eu ~]$ gsutil mb gs://nw102-imelnik1
Creating gs://nw102-imelnik1/...
[obstaclex@nat-node-gcp-eu ~]$
```

After this we are realesed external ip and now we can’t get access to google apis

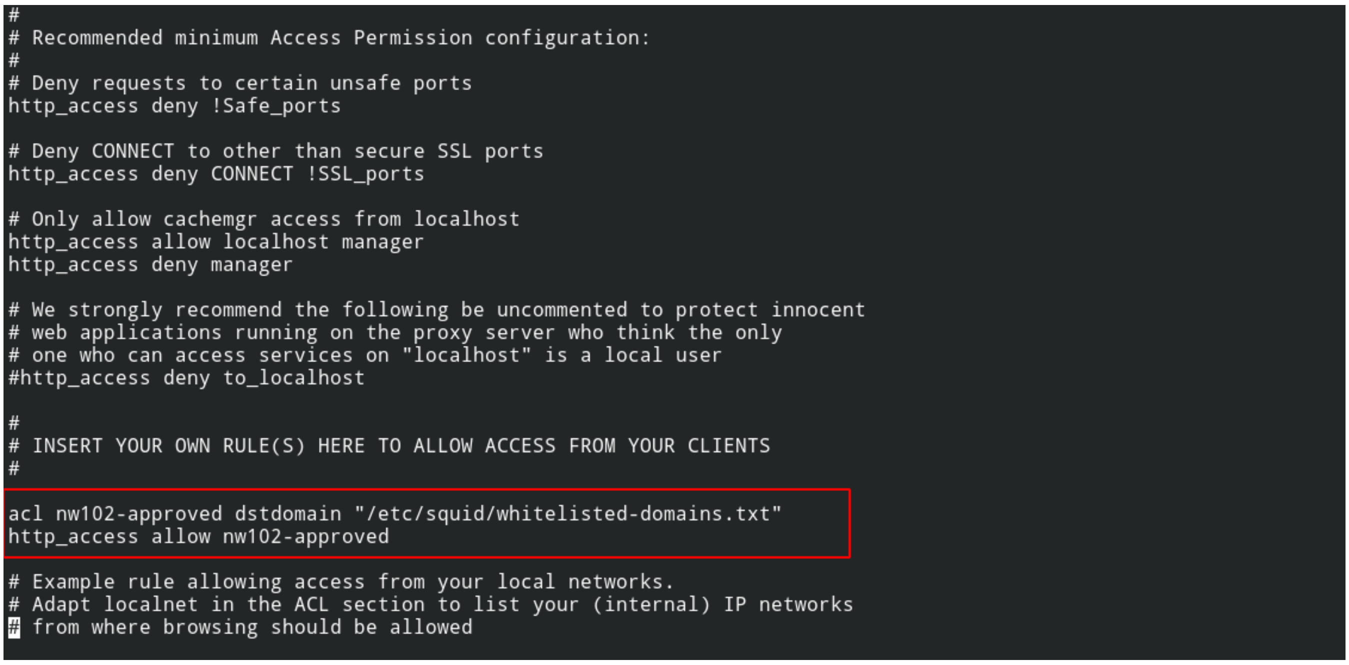


```
$ gcloud compute instances delete-access-config nat-node-gcp-eu --zone europe-west1-c

Updated [https://www.googleapis.com/compute/v1/projects/devops-lab-summer/zones/europe-west1-c/instances/nat-node-gcp-eu
].
$ gcloud compute instances add-tags nat-node-gcp-eu --zone europe-west1-c --tags app

Updated [https://www.googleapis.com/compute/v1/projects/devops-lab-summer/zones/europe-west1-c/instances/nat-node-gcp-eu
].
$ gcloud compute ssh nat-node-gcp-eu --zone europe-west1-c

External IP address was not found; defaulting to using IAP tunneling.
Last login: Wed Sep  2 15:33:41 2020 from 178.127.119.25
[obstaclex@nat-node-gcp-eu ~]$ gsutil ls gs://
INFO 0902 15:36:56.414599 retry_util.py] Retrying request, attempt #1...
^CCaught CTRL-C (signal 2) - exiting
[obstaclex@nat-node-gcp-eu ~]$
```

The next step – we are install Squid on NAT gw (we can use another instance with external ip) Configuration file for squid we could see below.

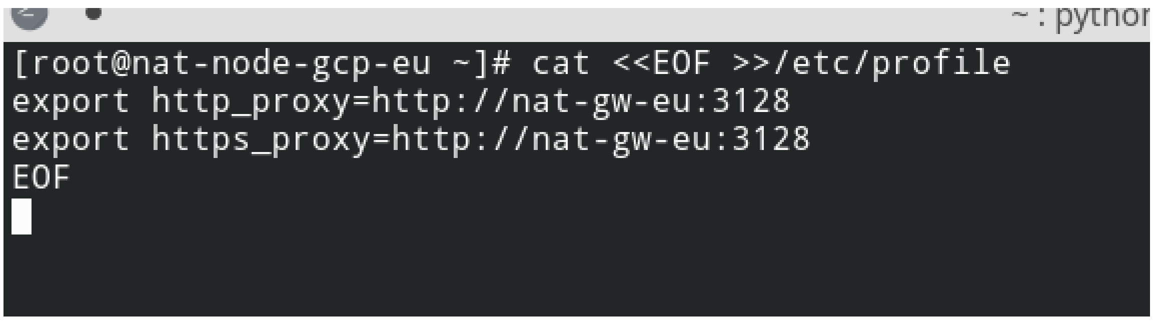


```
Last login: Wed Sep  2 14:49:25 2020 from 178.127.119.25
[obstaclex@nat-gw-eu ~]$ cat /etc/squid/whitelisted-domains.txt
.googleapis.com
<faux-on-prem-svc-ip>
.googleapis.com
35.222.94.209
[obstaclex@nat-gw-eu ~]$
```



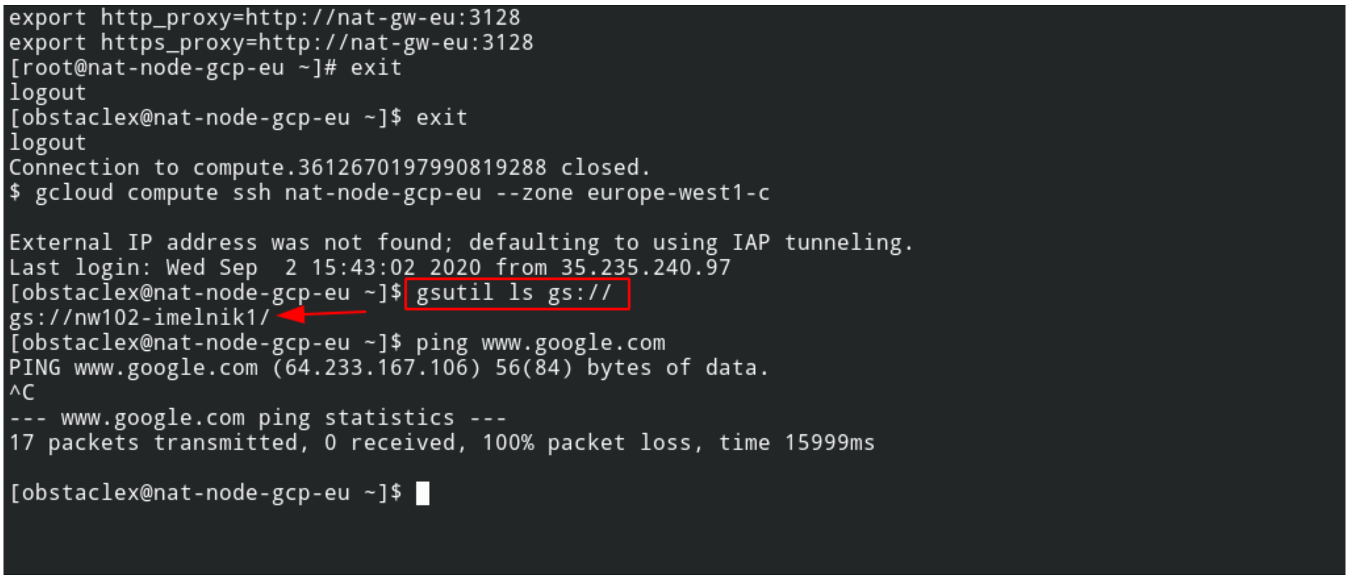
```
#
# Recommended minimum Access Permission configuration:
#
# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# Only allow cachemgr access from localhost
http_access allow localhost manager
http_access deny manager

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#

acl nw102-approved dstdomain "/etc/squid/whitelisted-domains.txt"
http_access allow nw102-approved

# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
```

After that all we setup our instance, make additional changes so that the proxy server is configured for all connections.



```
[root@nat-node-gcp-eu ~]# cat <<EOF >>/etc/profile
export http_proxy=http://nat-gw-eu:3128
export https_proxy=http://nat-gw-eu:3128
EOF
```

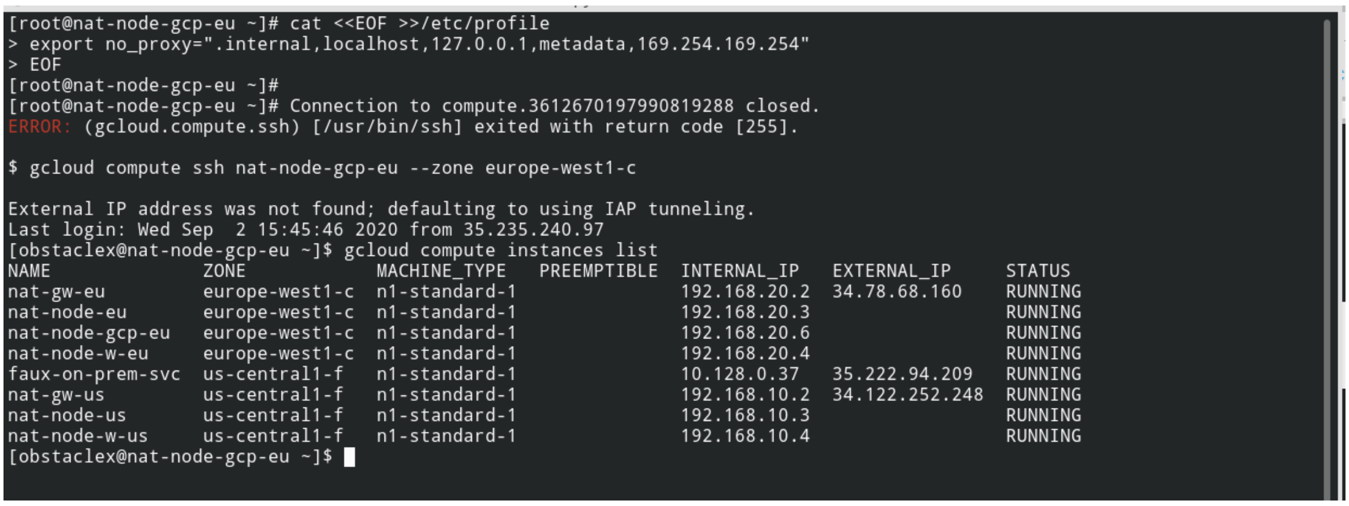
Now we can use some google services



```
export http_proxy=http://nat-gw-eu:3128
export https_proxy=http://nat-gw-eu:3128
[root@nat-node-gcp-eu ~]# exit
logout
[obstaclex@nat-node-gcp-eu ~]$ exit
logout
Connection to compute.3612670197990819288 closed.
$ gcloud compute ssh nat-node-gcp-eu --zone europe-west1-c

External IP address was not found; defaulting to using IAP tunneling.
Last login: Wed Sep  2 15:43:02 2020 from 35.235.240.97
[obstaclex@nat-node-gcp-eu ~]$ gsutil ls gs://
gs://nw102-imelnik1/
[obstaclex@nat-node-gcp-eu ~]$ ping www.google.com
PING www.google.com (64.233.167.106) 56(84) bytes of data.
^C
--- www.google.com ping statistics ---
17 packets transmitted, 0 received, 100% packet loss, time 15999ms

[obstaclex@nat-node-gcp-eu ~]$
```

For Compute Engine to function in such a restrictive environment, it needs to access the Compute Engine metadata service (on metadata.google.internal). These should not use the proxy.

To do this, add a proxy exception for those.



```
[root@nat-node-gcp-eu ~]# cat <<EOF >>/etc/profile
> export no_proxy=".internal,localhost,127.0.0.1,metadata,169.254.169.254"
> EOF
[root@nat-node-gcp-eu ~]#
[root@nat-node-gcp-eu ~]# Connection to compute.3612670197990819288 closed.
ERROR: (gcloud.compute.ssh) [/usr/bin/ssh] exited with return code [255].

$ gcloud compute ssh nat-node-gcp-eu --zone europe-west1-c

External IP address was not found; defaulting to using IAP tunneling.
Last login: Wed Sep  2 15:45:46 2020 from 35.235.240.97
[obstaclex@nat-node-gcp-eu ~]$ gcloud compute instances list
NAME              ZONE            MACHINE_TYPE   PREEMPTIBLE  INTERNAL_IP   EXTERNAL_IP     STATUS
nat-gw-eu         europe-west1-c  n1-standard-1               192.168.20.2  34.78.68.160    RUNNING
nat-node-eu       europe-west1-c  n1-standard-1               192.168.20.3                  RUNNING
nat-node-gcp-eu   europe-west1-c  n1-standard-1               192.168.20.6                  RUNNING
nat-node-w-eu     europe-west1-c  n1-standard-1               192.168.20.4                  RUNNING
faux-on-prem-svc  us-central1-f   n1-standard-1               10.128.0.37   35.222.94.209   RUNNING
nat-gw-us         us-central1-f   n1-standard-1               192.168.10.2  34.122.252.248  RUNNING
nat-node-us       us-central1-f   n1-standard-1               192.168.10.3                  RUNNING
nat-node-w-us     us-central1-f   n1-standard-1               192.168.10.4                  RUNNING
[obstaclex@nat-node-gcp-eu ~]$
```

TASK 5

Create network configuration via terraform.
Resources should be used:

1) **google_compute_network** (to create network)
   https://www.terraform.io/docs/providers/google/r/compute_network.html

   Network name: ${student_name}-vpc

2) **google_compute_firewall**
   (to create rules for external (allow 80,22) /internal access (allow 0-65535))
   https://www.terraform.io/docs/providers/google/r/compute_firewall.html

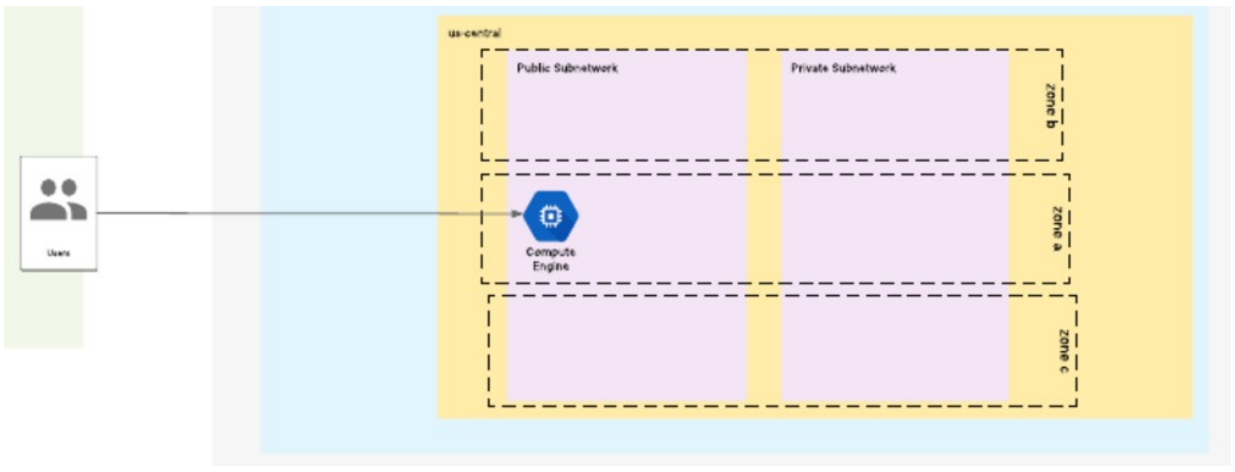
3) **google_compute_subnetwork**
   https://www.terraform.io/docs/providers/google/r/compute_subnetwork.html
   ranges:
   - Public range: 10."${student_IDnum}".1.0/24
   - Private range: 10."${student_IDnum}".2.0/24

4) **google_compute_instance**
   https://www.terraform.io/docs/providers/google/r/compute_instance.html

   1. nginx with default page "Hello from ${student_name}"

All resources should contain description (where it's possible)

Network topology.



All **reports**/code please place into repository:
https://github.com/MNT-Lab/google-cloud-module into appropriate branches: *first char of name + surname*.

For example:
Student: Siarhei Ivanou
Branch Name: **sivanou**

Format depends on case: README.md/scripts/terraform files

Email pattern: [MNT-CD-8.3]-FirstName-LastName

Email should contain the link to personalized branch.