



REPORT
CLOUD COMPUTING
ASSIGNMENT 2

Done by: Tolegen Ernazar
24MD0491
17.10.2024

Introduction	3
1. Virtual Machines in Google Cloud	4
1.1. VM Creation	4
1.2. Connection	5
1.3. Findings	6
2. Storage Solutions in Google Cloud	7
2.1 Bucket Creation:	7
2.2 Lifecycle Management	8
2.3. Findings	9
3. Networking in Google Cloud	10
3.1. VPC Setup	10
3.2. Connectivity	12
3.3. Findings	12
Conclusion	13
References	14

Introduction

The objective of this assignment is to set up and configure key networking and storage components in Google Cloud, including a Virtual Private Cloud (VPC), a Cloud Storage bucket, and a Virtual Machine (VM) instance. These tasks will help you understand how to create, configure, and manage cloud-based infrastructure, ensuring connectivity, security, and efficient storage management.

1. Virtual Machines in Google Cloud

1.1. VM Creation

- **Machine Type:** I selected the **e2-micro** instance type to minimize costs, as it provides enough resources for a simple web server test.

The screenshot shows the Google Cloud Platform console for project 'My Project 73448'. The 'CREATE VM FROM...' wizard is active. The 'Machine type' dropdown is open, showing options: N2, N2D, T2A, T2D, and N1. The 'e2-micro' option is selected, which is 0.25 vCPU (1 shared core), 1 GB memory. The 'Monthly estimate' on the right shows a total of \$3.44 for 2 vCPU + 1 GB memory and 10 GB balanced persistent disk.

Item	Monthly estimate
2 vCPU + 1 GB memory	\$2.44
10 GB balanced persistent disk	\$1.00
Total	\$3.44

- **Operating System:** I chose Ubuntu 20.04 LTS due to its stability, wide support, and easy compatibility with web server packages like Apache and Nginx.
- **Region:** I selected **us-central1-c**.

VM instances							
CREATE INSTANCE IMPORT VM REFRESH							
INSTANCES OBSERVABILITY INSTANCE SCHEDULES							
VM instances							
Filter Enter property name or value							
Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
✓	test-instance	us-central1-c			10.128.0.2 (nic0)	35.226.195.79 (nic0)	SSH

- **Firewall:** Enabled HTTP and HTTPS traffic to allow web traffic to the server.

Firewalls

- ☒ Allow HTTP traffic
- ☒ Allow HTTPS traffic
- ☐ Allow Load Balancer Health checks

Network tags

Network tags

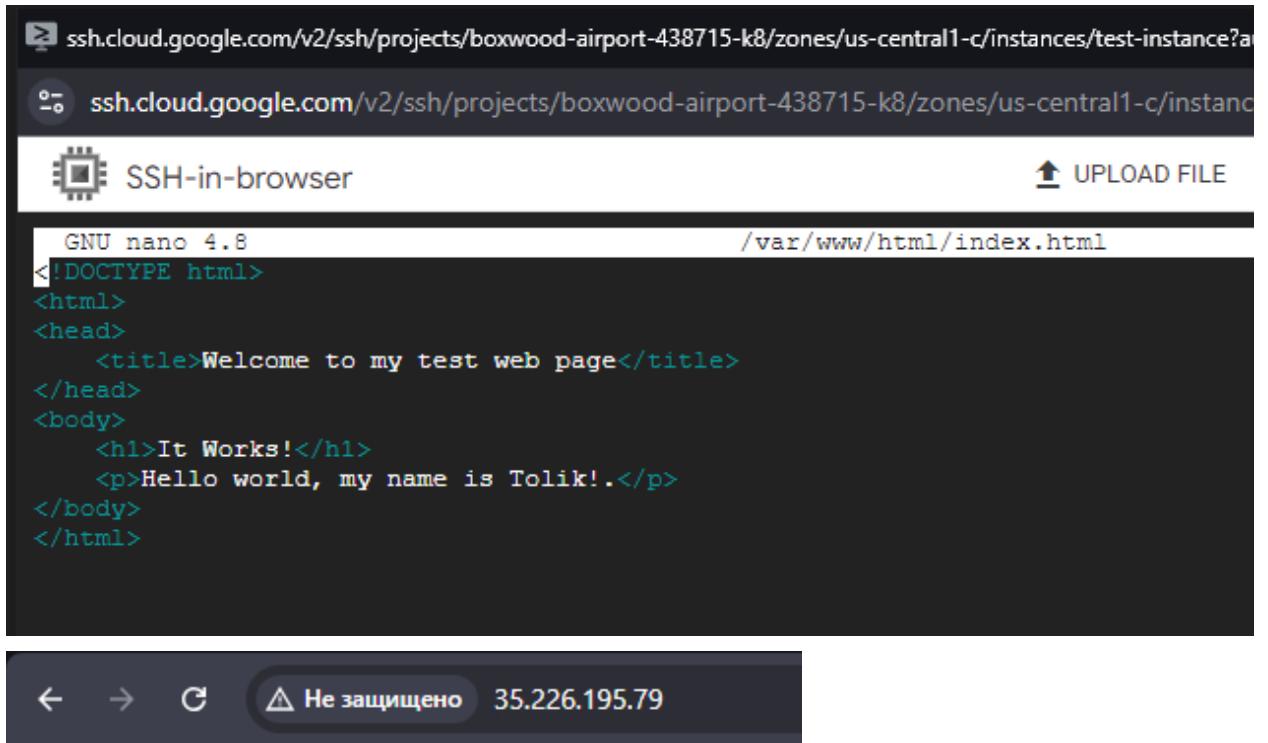
1.2. Connection

- Connected via SSH using the Cloud Console's built-in SSH option, which allows easy access without needing external SSH clients.
- I chose Nginx because it's a widely-used web server with a simple setup process. Installed it using **apt** and verified the service was running.

```
ssh.cloud.google.com/v2/ssh/projects/boxwood-airport-438715-k8/zones/us-central1-c/instances/test-instance?authuser=0&hl=en_US&...
ssh.cloud.google.com/v2/ssh/projects/boxwood-airport-438715-k8/zones/us-central1-c/instances/test-instance?authuser=0&hl=en_US&...
SSH-in-browser
[+] UPLOAD FILE [v] DOWNLOAD FILE [v] [v] [v] [v]
Setting up nginx-common (1.18.0-0ubuntu1.6) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
Setting up libjbig0:amd64 (2.1-3.1ubuntu0.20.04.1) ...
Setting up libnginx-mod-http-xslt-filter (1.18.0-0ubuntu1.6) ...
Setting up libwebp6:amd64 (0.6.1-2ubuntu0.20.04.3) ...
Setting up fonts-dejavu-core (2.37-1) ...
Setting up libjpeg-turbo8:amd64 (2.0.3-0ubuntu1.20.04.3) ...
Setting up libjpeg8:amd64 (8c-2ubuntu8) ...
Setting up libnginx-mod-mail (1.18.0-0ubuntu1.6) ...
Setting up fontconfig-config (2.13.1-2ubuntu3) ...
Setting up libnginx-mod-stream (1.18.0-0ubuntu1.6) ...
Setting up libtiff5:amd64 (4.1.0+git191117-2ubuntu0.20.04.14) ...
Setting up libfontconfig1:amd64 (2.13.1-2ubuntu3) ...
Setting up libgd3:amd64 (2.2.5-5.2ubuntu2.1) ...
Setting up libnginx-mod-http-image-filter (1.18.0-0ubuntu1.6) ...
Setting up nginx-core (1.18.0-0ubuntu1.6) ...
Setting up nginx (1.18.0-0ubuntu1.6) ...
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.23) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
ernazartolegen@test-instance:~$ sudo systemctl start nginx
ernazartolegen@test-instance:~$ sudo systemctl start nginx
ernazartolegen@test-instance:~$ sudo systemctl status nginx
• nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-10-17 14:52:02 UTC; 1min 39s ago
     Docs: man:nginx(8)
   Main PID: 45065 (nginx)
    Tasks: 3 (limit: 1134)
   Memory: 5.0M
   CGroup: /system.slice/nginx.service
           └─45065 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
             └─45066 nginx: worker process
               └─45067 nginx: worker process

Oct 17 14:52:02 test-instance systemd[1]: Starting A high performance web server and a reverse proxy server...
Oct 17 14:52:02 test-instance systemd[1]: Started A high performance web server and a reverse proxy server.
ernazartolegen@test-instance:~$
```

- Created a basic HTML file to confirm the web server was serving content correctly.



```
ssh.cloud.google.com/v2/ssh/projects/boxwood-airport-438715-k8/zones/us-central1-c/instances/test-instance?u=...
ssh.cloud.google.com/v2/ssh/projects/boxwood-airport-438715-k8/zones/us-central1-c/instanc

SSH-in-browser UPLOAD FILE

GNU nano 4.8 /var/www/html/index.html
<!DOCTYPE html>
<html>
<head>
  <title>Welcome to my test web page</title>
</head>
<body>
  <h1>It Works!</h1>
  <p>Hello world, my name is Tolik!.</p>
</body>
</html>

← → ↻ ⚠ Не защищено 35.226.195.79
```

It Works!

Hello world, my name is Tolik!.

1.3. Findings

- Creating a Virtual Machine (VM) in Google Cloud was pretty straightforward using the Cloud Console. I was able to connect to the VM via SSH and install a web server without any major issues. One challenge I encountered was that once a VM is created, I couldn't change its network settings. To connect the VM to a new VPC, I had to create a new instance, which taught me the importance of planning the network configuration carefully from the beginning.

2. Storage Solutions in Google Cloud

2.1 Bucket Creation:

- **Bucket Name:** I named the bucket **ernazar-test-bucket** to ensure it was globally unique.

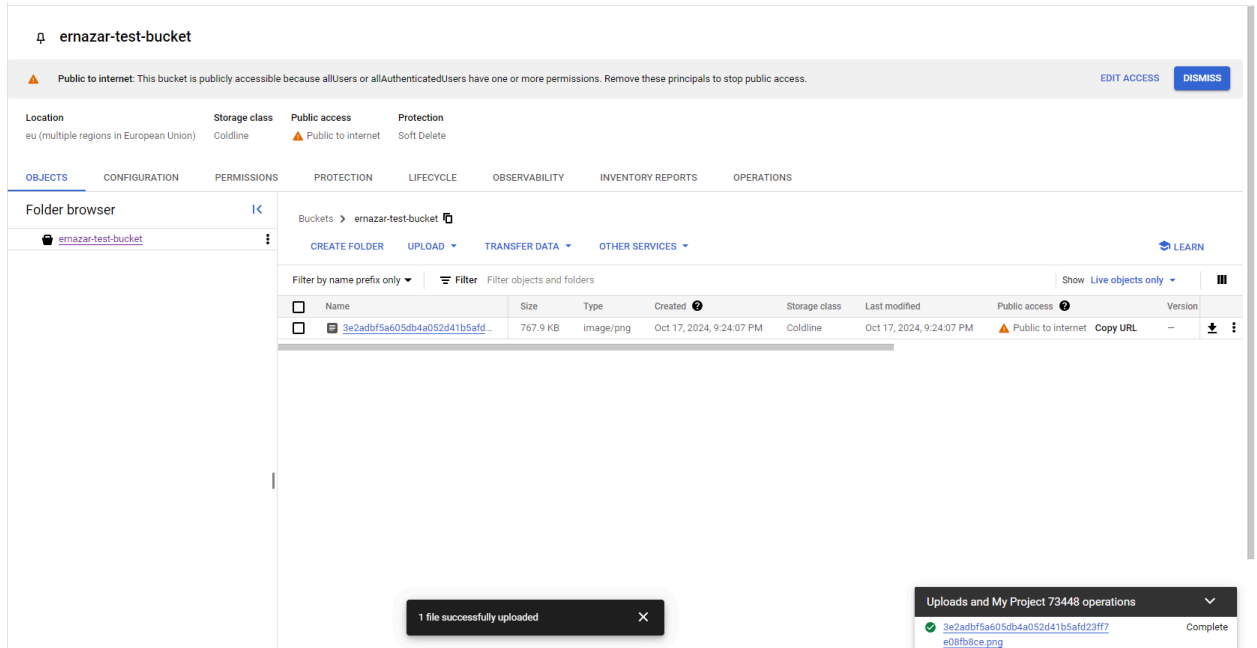
The screenshot shows the 'Create a bucket' wizard in Google Cloud. The left sidebar has 'Cloud Storage' selected. The main area shows a checklist of steps: 'Get Started' (Name: ernazar-test-bucket), 'Choose where to store your data' (Location: eu, Location type: Multi-region), 'Choose a storage class for your data' (Default storage class: Coldline), 'Choose how to control access to objects' (Public access prevention: On, Access control: Uniform), and 'Choose how to protect object data' (Soft delete policy: Default, Object versioning: Disabled, Bucket retention policy: Disabled, Object retention: Disabled, Encryption type: Google-managed). At the bottom are 'CREATE' and 'CANCEL' buttons. On the right, the 'Good to know' section includes 'Location pricing' with a table showing costs for 'eu' and 'With default replication', and an 'ESTIMATE YOUR MONTHLY COST' section.

Item	Cost
eu (multiple regions in European Union)	\$0.007 per GB-month
With default replication	\$0.020 per GB written

- **Region:** I chose **eu** for better latency in my region.
- **Storage Class:** I selected **Coldline** as I don't need quick access to files and more cost-efficient. However, for quick access, **Standard** class might be more appropriate.
- **Access Control:** I chose **Uniform** control for the private access, but then added principle to **allUsers**.

The screenshot shows the 'Edit access to "ernazar-test-bucket"' dialog. It has a table with 'Principal' and 'Resource' columns, showing 'allUsers' and 'ernazar-test-bucket'. Below is the 'Assign roles' section with a description: 'Roles are composed of sets of permissions and determine what the principal can do with this resource. [Learn more](#)'. There is a dropdown for 'Role' set to 'Storage Legacy Object Reader' and an 'IAM condition (optional)' section with a '+ ADD IAM CONDITION' button. Below the role dropdown is a description: 'Grants permission to view objects and their metadata, excluding ACLs.' and a '+ ADD ANOTHER ROLE' button. At the bottom are 'SAVE', 'TEST CHANGES', and 'CANCEL' buttons.

- Uploaded a sample image (sample.jpg) to test file storage. This could be useful for storing assets for a website, backups, or logs.



If you want to see what's inside, you can check it by following link below, but it will be accessible for only **30 days** from now, because of **lifecycle** rule:
<https://storage.googleapis.com/ernazar-test-bucket/3e2adbf5a605db4a052d41b5afd23ff7e08fb8ce.png>

2.2 Lifecycle Management

- Configured a lifecycle rule to automatically delete files older than 30 days. This can help manage storage costs by ensuring that old files that are no longer needed are automatically removed, particularly in use cases such as log files or backups.

←

Edit object lifecycle rule

Delete multi-part uploads

Sets a time limit and removes unfinished or idle multi-part uploads

CONTINUE

•

Select object conditions

This rule will apply the action to current and future objects or multi-part uploads that meet all the selected conditions below. [Learn more](#)

Set Rule Scopes

Use prefix and suffix rule scopes to filter objects by their paths. You can specify up to 50 prefix and 50 suffix matches per bucket, across all rules.

☐ Object name matches prefix

☐ Object name matches suffix

Set Conditions

☒ Age ?

30

days

Age is counted from when an object was uploaded to the current bucket, even if it moved from another

☐ Created before ?

☐ Storage class matches

☐ Number of newer versions ?

☐ Days since becoming noncurrent ?

☐ Became noncurrent before ?

☐ Live state

☐ Days since custom time ?

☐ Custom time before ?

CONTINUE

2.3. Findings

- Creating a Cloud Storage bucket and uploading files was easy to follow. I was able to set access controls (either public or private) and successfully uploaded a test file. Configuring lifecycle management was also simple, and I learned how lifecycle rules can help automate the deletion of

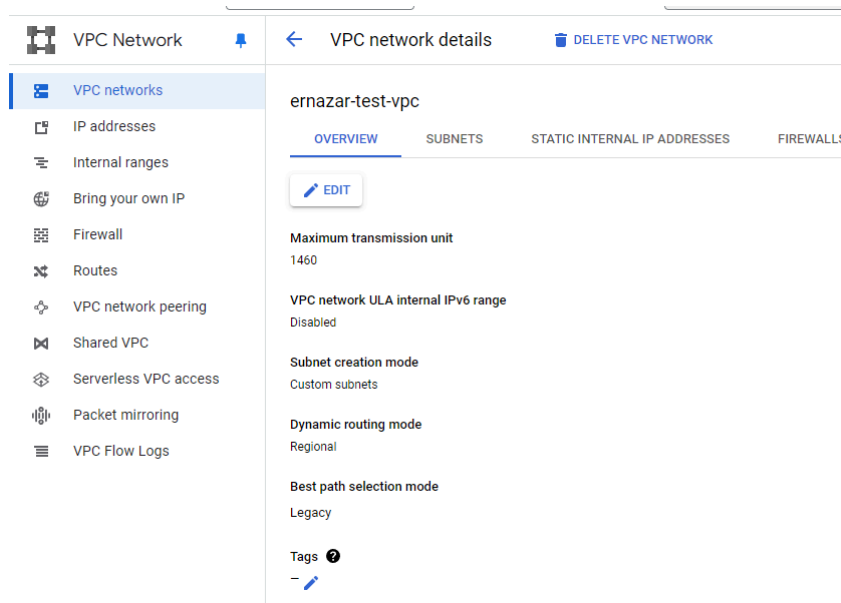
9

files after a certain time, which is really useful for managing costs and storage over time. Overall, it was a good lesson in how cloud storage can be managed efficiently.

3. Networking in Google Cloud

3.1. VPC Setup

- **VPC Name:** I named the VPC **ernazar-test-vpc** to easily identify it.



- **Subnets:** Created a subnet called **test-subnet** in the **europa-west1** region with the CIDR range 10.0.0.0/24. This allows for a flexible network architecture, and additional subnets can be added

for future scaling or isolation of resources.

The screenshot shows the 'Create a VPC network' page in Google Cloud. The 'Edit subnet' section is active, displaying the following configuration:

- Name ***: test-subnet (with a help icon and note: 'Lowercase letters, numbers, hyphens allowed')
- Description**: (empty text field)
- Region ***: europe-west1 (with a help icon)
- IP stack type**:
 - ☒ IPv4 (single-stack)
 - ☐ IPv4 and IPv6 (dual-stack) (with a help icon)
- IPv4 range ***: 10.0.0.0/24 (with a help icon and example: 'E.g. 10.0.0.0/24')
- CREATE SECONDARY IPV4 RANGE**: (button)
- Private Google Access**:
 - ☐ On
 - ☒ Off (with a help icon)
- Flow logs**:
 - ☐ On
 - ☒ Off
- Hybrid Subnets**:
 - ☐ On
 - ☒ Off (with a help icon)

A 'DONE' button is visible at the bottom right of the configuration panel.

- **Use Case:** The VPC allows me to isolate my VM from other resources and networks in the cloud, ensuring security and performance.
- Created a firewall rule named **test-firewall** with SSH (TCP port 22) and HTTP (TCP port 80) traffic to the instances in the VPC.

The screenshot shows the Google Cloud Network Security console. The left sidebar lists various security services, with 'Firewall policies' selected under 'Cloud NGFW'. The main panel shows the configuration for a firewall rule named 'test-firewall'.

Source filter: IPv4 ranges

Source IPv4 ranges *: 0.0.0.0/0 (with a help icon)

Second source filter: None

Destination filter: None

Protocols and ports (with a help icon):

- ☐ Allow all
- ☒ Specified protocols and ports

TCP (checked):

Ports: 80,22 (with a help icon and example: 'E.g. 20, 50-60')

UDP (unchecked):

Ports: (empty text field with example: 'E.g. all')

SCTP (unchecked):

Ports: (empty text field with example: 'E.g. 20, 50-60')

Other (unchecked):

Protocols: (empty text field with example: 'Separate multiple protocols by commas, e.g. ah, icmp')

DISABLE RULE (button)

SAVE (button) **CANCEL** (button)

- **Importance:** Firewall rules are critical for securing cloud resources, allowing only necessary traffic while blocking unauthorized access.

3.2. Connectivity

- I tried to connect VM to VPC, but could not do it due to some restrictions in adding **new Network Interface** in my VM instance. I try to describe the actions without screenshots.



- Reconfigured the VM network interface to ensure it is connected to the newly created VPC (**ernazar-test-vpc**).
- Ping Google's DNS server (8.8.8.8) from the VM to verify internet connectivity through the VPC.
- **Verification:** The successful ping confirms that the VM is connected to the internet and can communicate with external servers.

3.3. Findings

- Setting up a Virtual Private Cloud (VPC) and configuring subnets and firewall rules was a clear process. I learned how to connect my VM to the correct VPC, though I had to create a new VM to ensure it was attached to the proper network. I also tested connectivity by pinging an external server, which confirmed that the setup worked. This process showed me how critical VPCs and firewall rules are for securing resources and ensuring the VM can communicate with the internet properly.

Conclusion

Through this assignment, I gained valuable hands-on experience with essential Google Cloud services, including Virtual Machines, Cloud Storage, and networking via Virtual Private Clouds (VPCs). Key learnings include the importance of carefully planning network configurations from the start when setting up VMs, as network settings cannot be changed later without recreating the instance. I also learned how lifecycle management in Cloud Storage can optimize storage use and costs by automating file deletion. Setting up VPCs and firewall rules underscored the importance of networking in securing and managing cloud resources.

These Google Cloud services have a wide range of applications. VMs can be used for hosting websites, running applications, or even managing databases. Cloud Storage offers a scalable solution for storing and managing data, while lifecycle management helps with long-term data retention and cost efficiency. The ability to set up secure, isolated VPC networks is essential for any cloud-based infrastructure, especially for businesses looking to ensure secure and efficient communication between their resources and the internet. Overall, these tools provide the foundation for building and managing scalable, secure cloud environments.

References

Google Cloud. (n.d.). *VPC documentation*. Google Cloud. Retrieved from <https://cloud.google.com/vpc/docs>

Google Cloud. (n.d.). *VPC networking and security*. Google Cloud. Retrieved from <https://cloud.google.com/networking/docs/vpc>