Introduction:

In today's digital era, cybersecurity has become an essential pillar in safeguarding organizational assets, sensitive information, and network infrastructures. The increasing frequency and sophistication of cyberattacks have highlighted the need for proactive security measures that identify and mitigate potential weaknesses before they can be exploited.

One of the most critical components of a strong cybersecurity framework is **Vulnerability Assessment and Remediation Planning**. This process focuses on detecting, analyzing, and addressing security vulnerabilities within systems, networks, or applications. By systematically identifying risks and implementing remediation actions, organizations can significantly enhance their resilience against cyber threats and reduce their overall attack surface.

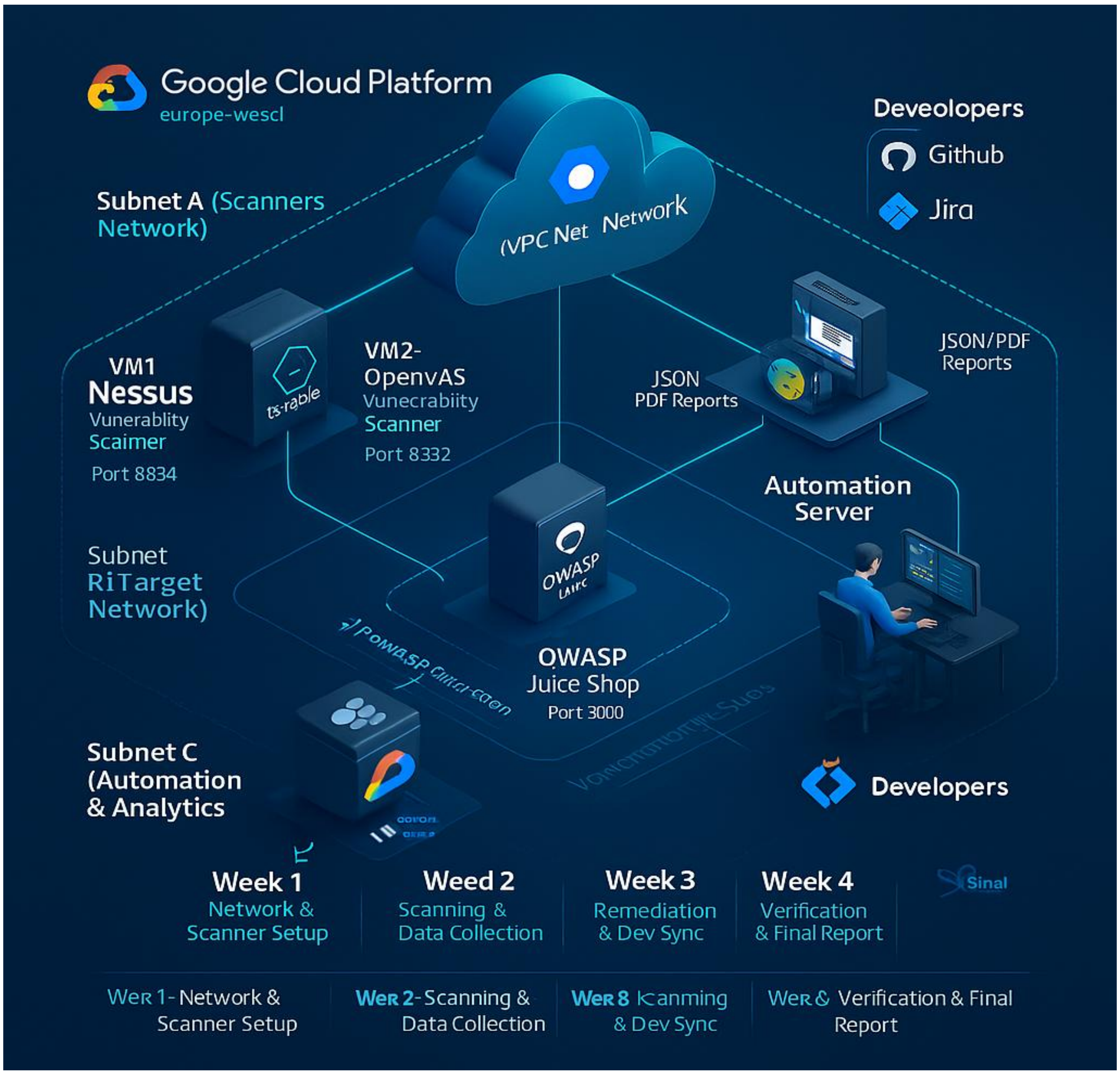
This project, titled **"Vulnerability Assessment and Remediation Plan"**, was developed under the **Digital Egypt Pioneers Initiative (DEPI)** in the **Incident Response Analyst (IR) track**, in collaboration with **AMIT Learning**. The project aims to simulate a real-world cybersecurity environment by applying industry-standard tools such as **Nessus, OpenVAS, and Qualys** to perform vulnerability scans, analyze identified threats, and develop a structured remediation plan.

The work is organized into four key phases:

1. **Preparation Phase:** Defining the assessment scope and configuring tools.
2. **Assessment Phase:** Performing vulnerability scans and analyzing results.
3. **Remediation Phase:** Developing a prioritized plan for mitigation and applying corrective actions.
4. **Verification Phase:** Reassessing the system to ensure that all vulnerabilities have been resolved.

Through this structured approach, the project demonstrates the practical application of cybersecurity principles and provides a clear framework for conducting vulnerability management in a professional context. The insights gained from this work contribute to improving system security posture and strengthening the overall defense strategy of digital infrastructures

Architecture Design:


Google Cloud Platform
europe-wescl
(VPC Net Network
Deveolopers
Github
Jira
Subnet A (Scanners Network)
VM1 Nessus Vunerablity Scaimer Port 8834
ts-rable
VM2- OpenVAS Vunecrabity Scanner Port 8332
JSON PDF Reports
JSON/PDF Reports
Automation Server
Subnet RiTarget Network)
OWASP Juice Shop Port 3000
Subnet C (Automation & Analytics
Developers
Week 1 Network & Scanner Setup
Weed 2 Scanning & Data Collection
Week 3 Remediation & Dev Sync
Week 4 Verification & Final Report
Sinal
WeR 1- Network & Scanner Setup
WeR 2-Scanning & Data Collection
WeR 8 Kanming & Dev Sync
WeR & Verification & Final Report

The diagram illustrates the cloud-based network architecture used to implement the Vulnerability Assessment and Remediation Plan. The project was deployed on Google Cloud Platform (GCP), leveraging virtual machines to host the vulnerability scanners (Nessus and OpenVAS) and the target environment (OWASP Juice Shop).The setup includes three main subnets:Subnet A (Scanners Network): Hosts the vulnerability scanners for system analysis.Subnet B (RiTarget Network): Contains the vulnerable application environment.Subnet C (Automation & Analytics): Handles automation tasks, data processing, and report generation.The Automation Server manages report generation (JSON/PDF) and integrates with development tools such as GitHub and Jira for continuous tracking and collaboration. The weekly workflow covers network setup, scanning, remediation, and final verification over four weeks, ensuring a structured and secure execution process.

Tools Selection:

The project infrastructure is designed and deployed entirely on the cloud to ensure scalability, flexibility, and easy collaboration among team members. For this purpose, the team selected Google Cloud Platform (GCP) as the main environment due to its reliability, integrated services, and student-friendly offerings.

Cloud Platform Setup:

The project was built and configured on **Google Cloud Platform (GCP)**, where each team member could access shared virtual machines and collaborate in real time. One of the key advantages of GCP is that **Google provides free credits worth $300 for 90 days**, allowing students and developers to explore and deploy various cloud-based solutions without initial costs. Before creating virtual machines (VMs), the team prepared the working environment to meet all necessary requirements and dependencies for the project's tools and scanners.

Collaborative Environment Configuration:

Since this project was developed collaboratively by the IR.Team, it was crucial to configure the environment so that each team member could access, modify, and monitor the same resources. Therefore, access control settings were adjusted to ensure that all users could securely connect to the shared VM instances through SSH (Secure Shell) using their Google accounts.

This setup allowed every developer to work under the same environmental profile. In addition, a shared directory was created on the operating system (Ubuntu 24.04 LTS) to synchronize files, logs, and scripts. When any user connects to the VM through the shell, they are automatically directed to the shared project folder, ensuring that all team members can view, edit, and contribute to the same workspace.

Operating System and Initial Configuration:

When provisioning the virtual machines, the team selected Ubuntu 24.04 LTS as the base operating system due to its stability, strong community support, and compatibility with the security tools used in the project (such as Nessus, OpenVAS, Wazuh, and OWASP Juice Shop). After launching the VMs, the OS was updated, and user roles were configured. A shared folder was created with proper Access Control Lists (ACLs) to manage permissions, ensuring that every authorized member could access and modify project files without conflicts.

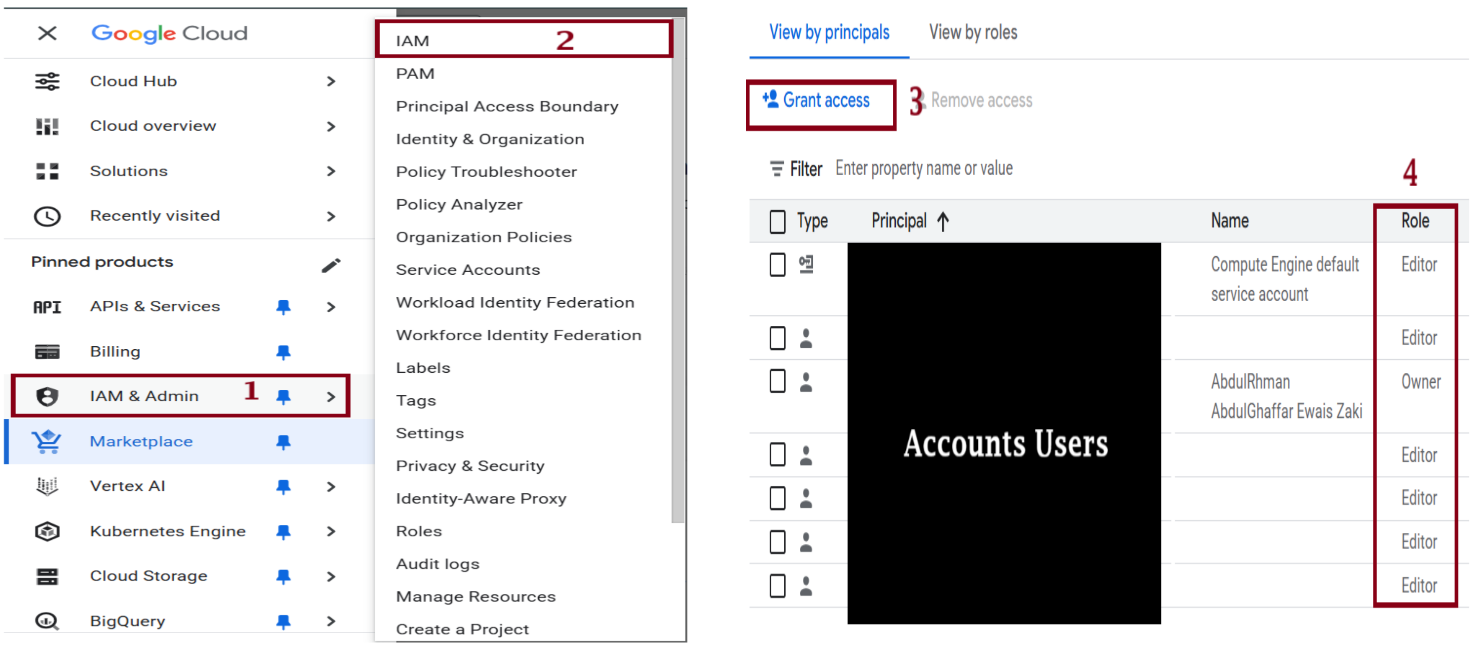
Firewall and Network Accessibility:

To maintain seamless communication between all virtual machines and team members' local networks, the firewall settings on GCP were adjusted carefully. Specific ports were opened to allow the operation of each security tool (e.g., Nessus on port 8834, OpenVAS on port 9392, Juice Shop on port 3000). This configuration prevents internal blocking between VMs and ensures that external access from developers' local machines remains smooth and secure.

Documentation and Visual Reference:

All commands and setup configurations used during this phase will be included with screenshots and CLI outputs to enhance clarity and reproducibility. A detailed architectural diagram illustrating this setup is also available on the project's GitHub repository, showing the relationships between cloud components, virtual machines, and network configuration.

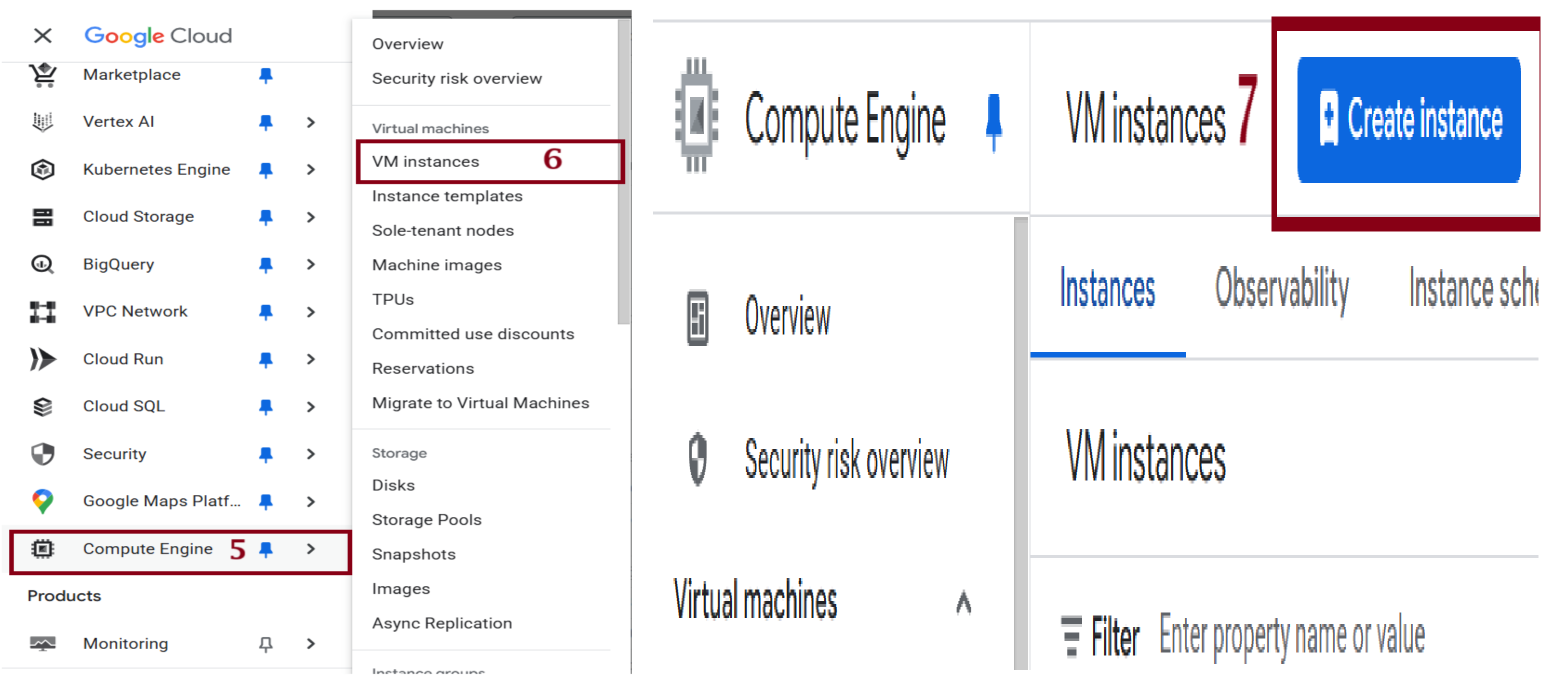
Figure 1: Cloud Infrastructure Overview (Available on GitHub — Project Documentation Section)

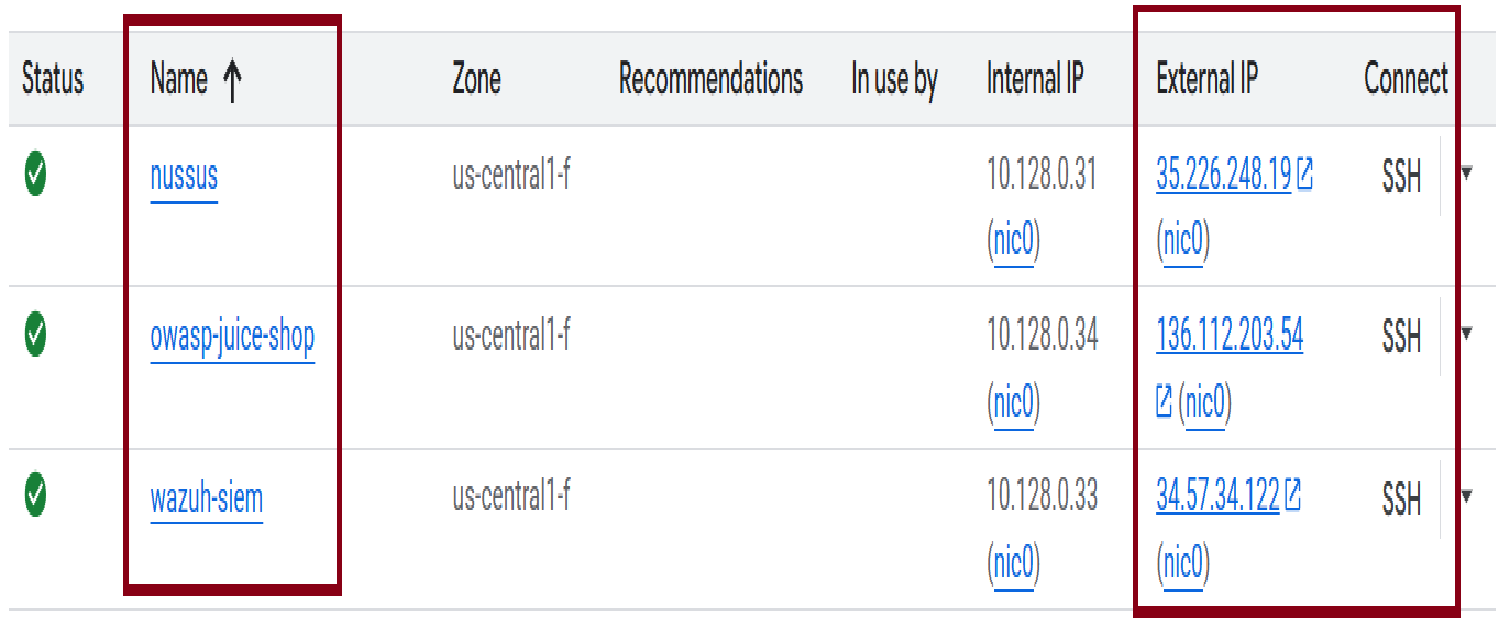


Type	Principal	Name	Role
		Compute Engine default service account	Editor
			Editor
		AbdulRhman AbdulGhaffar Ewais Zaki	Owner
			Editor
			Editor
			Editor
			Editor

Grant proper IAM roles and enable OS Login for centralized access:

```
gcloud projects add-iam-policy-binding <project-id> \
--member="user:<email>" --role="roles/compute.instanceAdmin.v1"
gcloud projects add-iam-policy-binding <project-id> \
--member="user:<email>" --role="roles/compute.osLogin"
gcloud compute project-info add-metadata --metadata enable-oslogin=TRUE
```





Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
✓	nussus	us-central1-f			10.128.0.31 (nic0)	35.226.248.19 (nic0)	SSH
✓	owasp-juice-shop	us-central1-f			10.128.0.34 (nic0)	136.112.203.54 (nic0)	SSH
✓	wazuh-siem	us-central1-f			10.128.0.33 (nic0)	34.57.34.122 (nic0)	SSH

Virtual Machine Creation and Configuration on Google Cloud Platform:

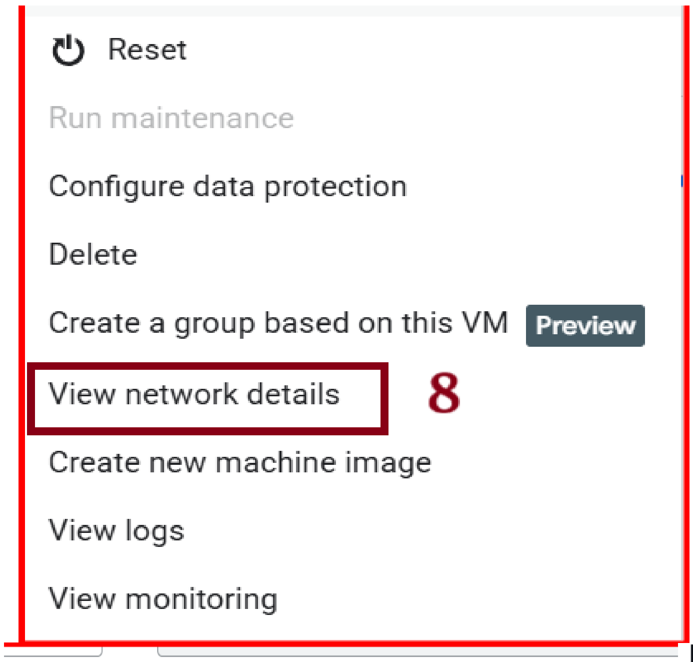
The image above illustrates the process of creating Virtual Machine (VM) instances on Google Cloud Platform (GCP), which represents the first practical step in setting up the project environment.

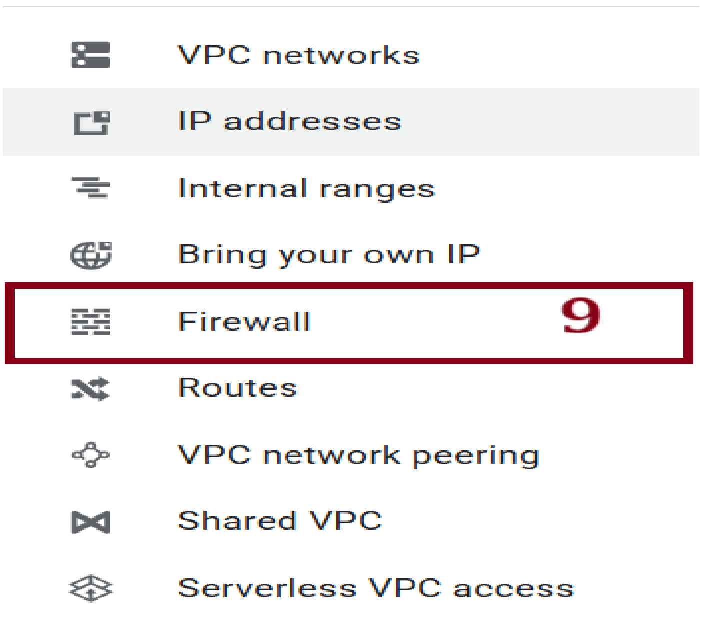
As shown in the left panel, the process starts by navigating to Compute Engine (Step 5) in the main GCP console menu, then selecting VM Instances (Step 6) to view or create new virtual machines.

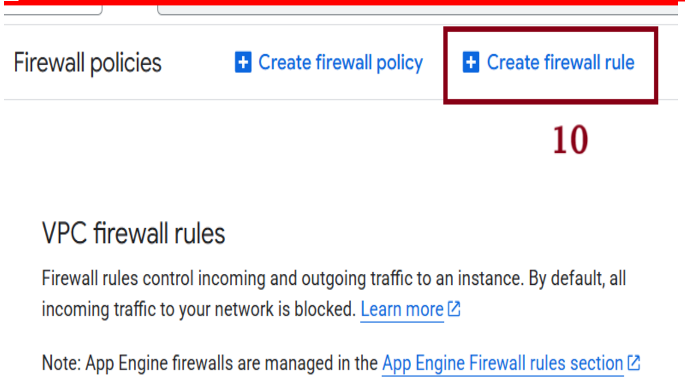
In the top-right section, the "Create Instance" button (Step 7) is used to deploy a new virtual machine. This allows users to specify configurations such as the operating system, machine type, and network access settings.

The bottom section of the figure displays the list of all created VMs for this project — including Nessus, OWASP Juice Shop, and Wazuh SIEM — each running on the us-central1-f zone.

For each VM, both Internal IP and External IP addresses are provided. External IP allows secure web.

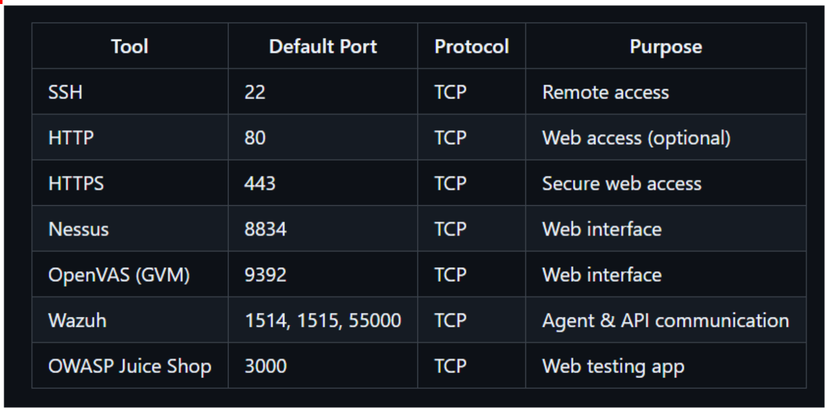






Firewall rules control incoming and outgoing traffic to an instance. By default, all incoming traffic to your network is blocked. Learn more

Note: App Engine firewalls are managed in the App Engine Firewall rules section



Tool	Default Port	Protocol	Purpose
SSH	22	TCP	Remote access
HTTP	80	TCP	Web access (optional)
HTTPS	443	TCP	Secure web access
Nessus	8834	TCP	Web interface
OpenVAS (GVM)	9392	TCP	Web interface
Wazuh	1514, 1515, 55000	TCP	Agent & API communication
OWASP Juice Shop	3000	TCP	Web testing app

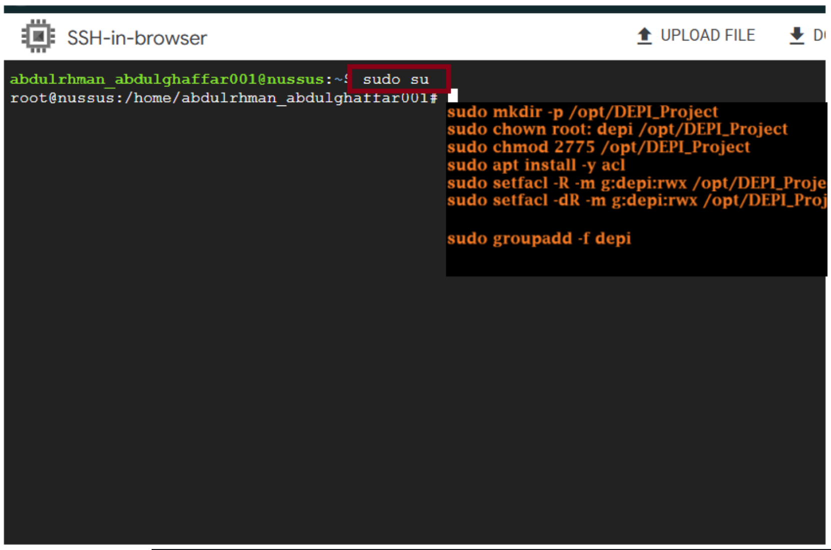
Configuring Network Access and Firewall Rules on Google Cloud Platform:

The figure above shows the steps for setting up network details and firewall rules on Google Cloud Platform (GCP).

First, from the VM options menu, select “View Network Details” (Step 8) to access the instance’s network configuration.

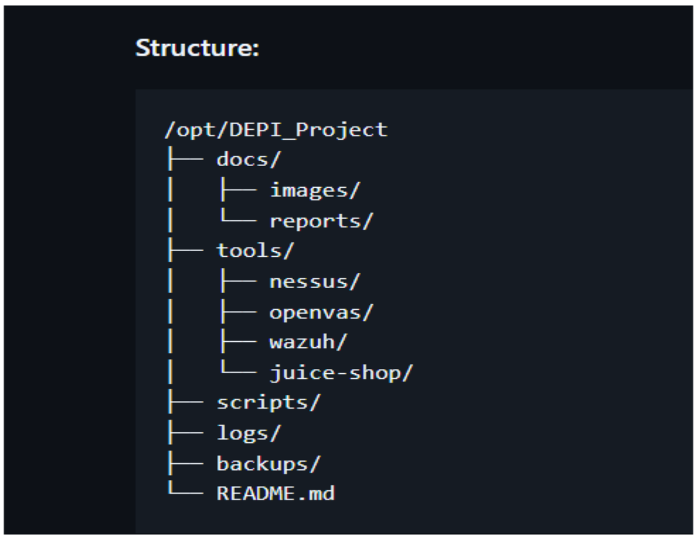
Then, under VPC Network → Firewall (Step 9), create new rules using “Create Firewall Rule” (Step 10) to allow secure communication between all virtual machines.

The rules enable access for essential services and tools, including SSH (22), HTTPS (443), Nessus (8834), OpenVAS (9392), Wazuh (1514–55000), and OWASP Juice Shop (3000). These configurations ensure smooth connectivity between VMs and allow remote access for all team members while maintaining network security.



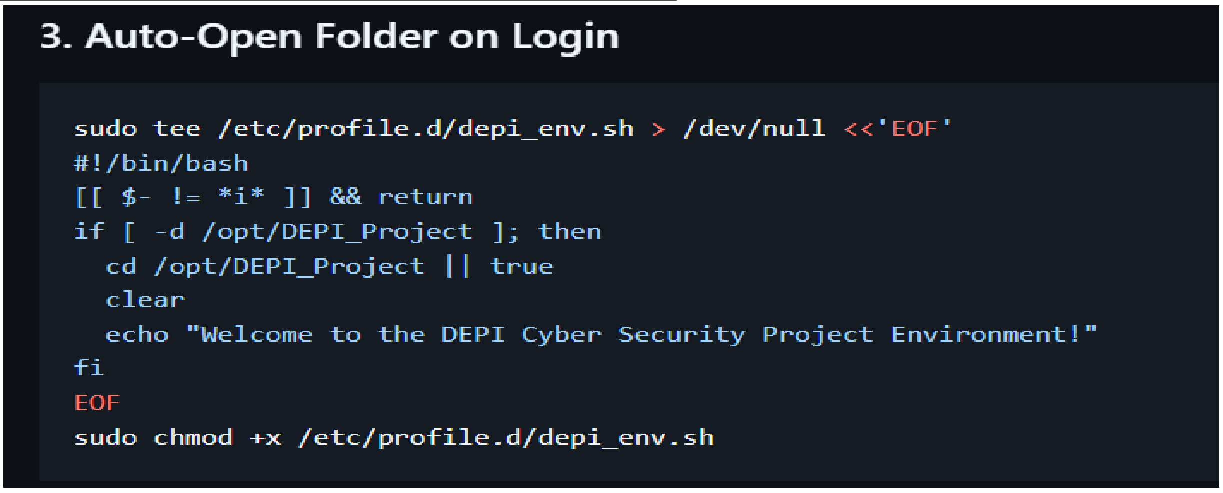
```
abdulrhman_abdulghaffar001@nussus:~$ sudo su
root@nussus:/home/abdulrhman_abdulghaffar001#
```

```
sudo mkdir -p /opt/DEPI_Project
sudo chown root: depi /opt/DEPI_Project
sudo chmod 2775 /opt/DEPI_Project
sudo apt install -y acl
sudo setfacl -R -m g:depi:rwx /opt/DEPI_Proje
sudo setfacl -dR -m g:depi:rwx /opt/DEPI_Proj

sudo groupadd -f depi
```



Structure:

```
/opt/DEPI_Project
├── docs/
│   ├── images/
│   └── reports/
├── tools/
│   ├── nessus/
│   ├── openvas/
│   ├── wazuh/
│   └── juice-shop/
├── scripts/
├── logs/
├── backups/
└── README.md
```



3. Auto-Open Folder on Login

```
sudo tee /etc/profile.d/depi_env.sh > /dev/null <<'EOF'
#!/bin/bash
[[ $- != *i* ]] && return
if [ -d /opt/DEPI_Project ]; then
  cd /opt/DEPI_Project || true
  clear
  echo "Welcome to the DEPI Cyber Security Project Environment!"
fi
EOF
sudo chmod +x /etc/profile.d/depi_env.sh
```

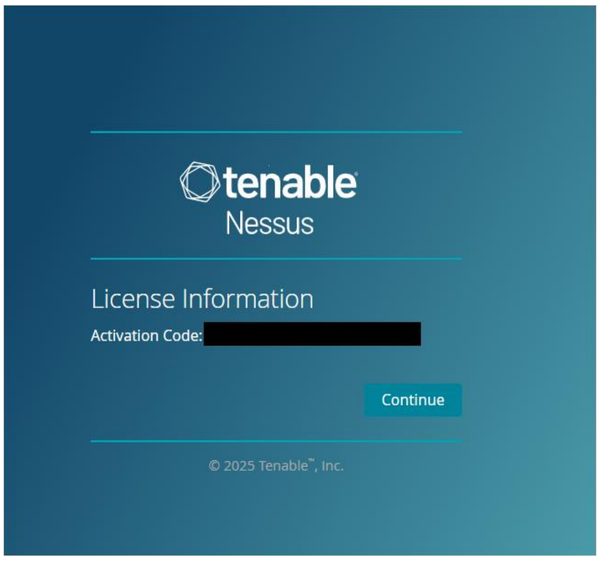
Shared Project Folder Configuration and Structure:

The figures above illustrate the process of creating and configuring the shared working directory for the DEPI Cyber Security Project.

After connecting to the virtual machine via SSH-in-browser, the administrator switched to root privileges using sudo su and created the directory /opt/DEPI_Project.

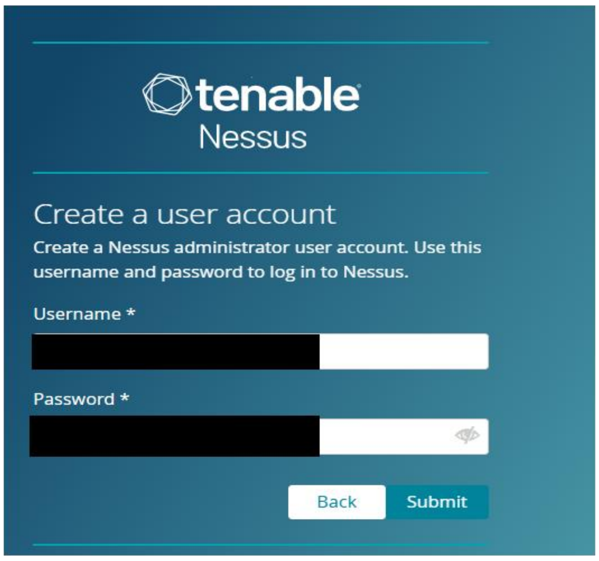
Ownership and permissions were configured for the “depi” group using ACLs to ensure secure and shared access for all team members.

An automation script (depi_env.sh) was placed in /etc/profile.d/ to automatically redirect users to the shared directory upon login and display a welcome message, ensuring a consistent workspace for everyone.
The final structure includes organized subfolders for docs, tools, scripts, logs, and backups, providing a clear and efficient workflow for collaboration and project maintenance.



License Information

Activation Code: [redacted]

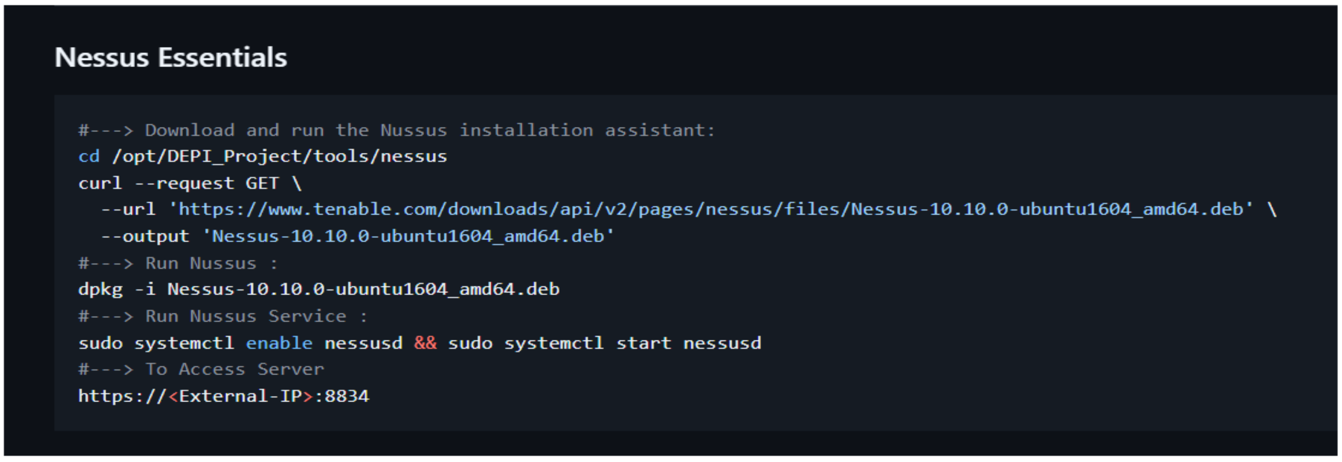


Create a user account

Create a Nessus administrator user account. Use this username and password to log in to Nessus.

Username * [redacted]

Password * [redacted]



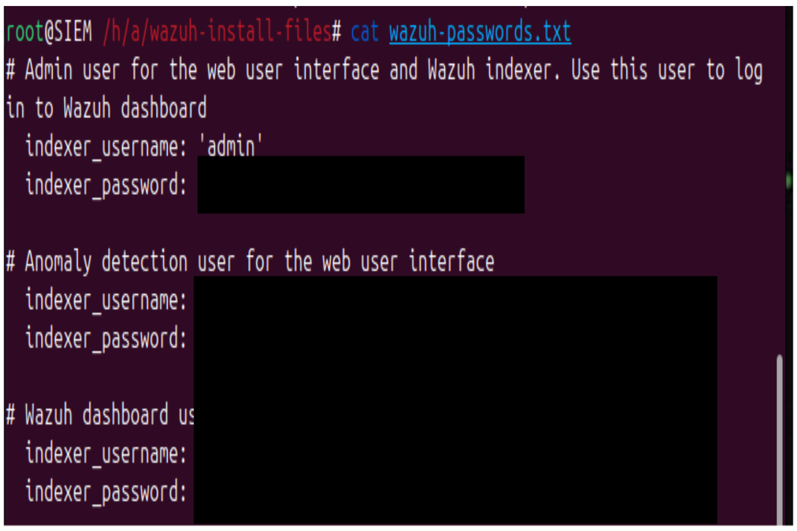
Nessus Essentials

```
#---> Download and run the Nussus installation assistant:
cd /opt/DEPI_Project/tools/nessus
curl --request GET \
  --url 'https://www.tenable.com/downloads/api/v2/pages/nessus/files/Nessus-10.10.0-ubuntu1604_amd64.deb' \
  --output 'Nessus-10.10.0-ubuntu1604_amd64.deb'
#---> Run Nussus :
dpkg -i Nessus-10.10.0-ubuntu1604_amd64.deb
#---> Run Nussus Service :
sudo systemctl enable nessusd && sudo systemctl start nessusd
#---> To Access Server
https://<External-IP>:8834
```

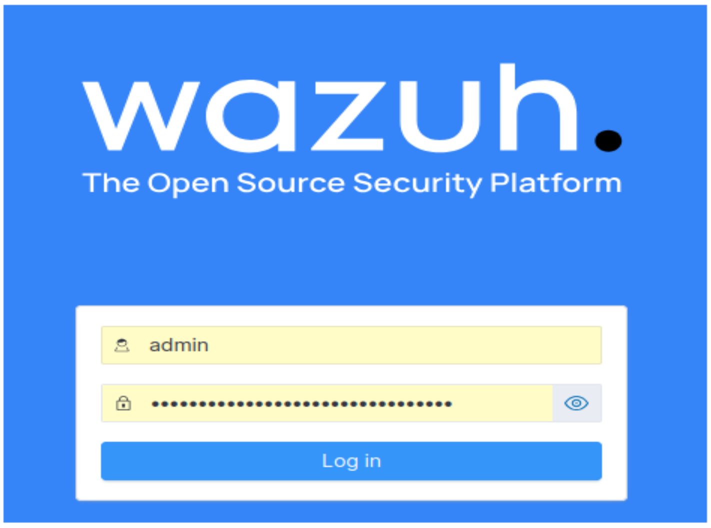
Nessus Essentials Installation and Setup:

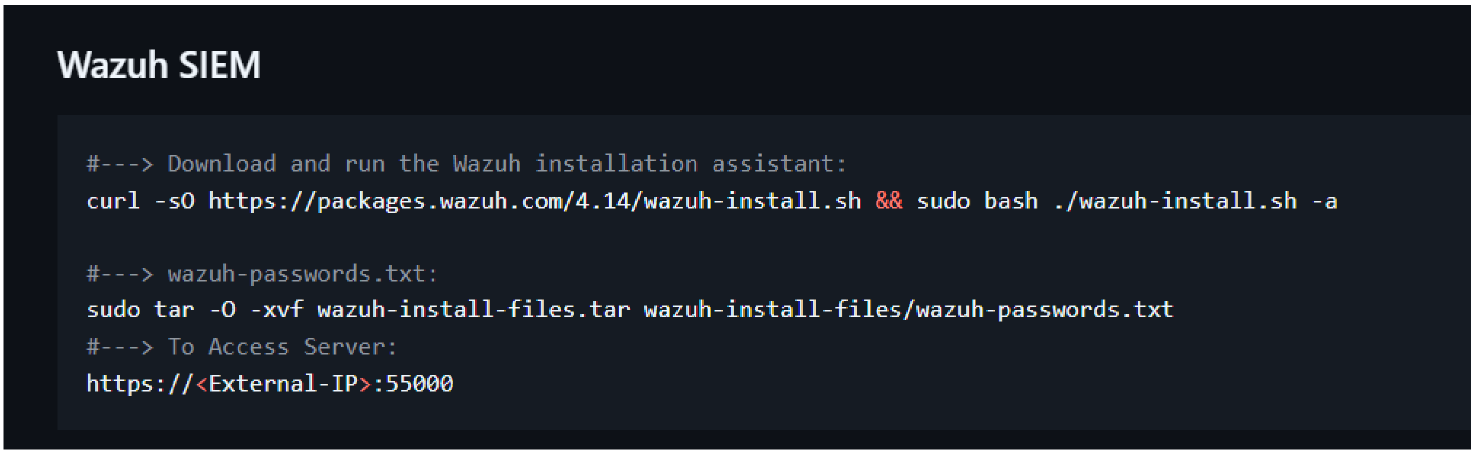
The figures above illustrate the installation and configuration process of Nessus Essentials, the main vulnerability scanning tool used in this project.
The tool was first downloaded and installed inside the directory /opt/DEPI_Project/tools/nessus, as shown in the command sequence. Once the installation was complete, the Nessus service was enabled and started, making the web interface accessible at https://<External-IP>:8834.

After launching the interface, the license activation code was entered to register the software, followed by creating an administrator account for the project team (Username: ir.team). These steps ensured that Nessus was fully operational and ready for performing comprehensive vulnerability assessments in later project stages.



```
root@SIEM /h/a/wazuh-install-files# cat wazuh-passwords.txt
# Admin user for the web user interface and Wazuh indexer. Use this user to log
in to Wazuh dashboard
  indexer_username: 'admin'
  indexer_password: [redacted]

# Anomaly detection user for the web user interface
  indexer_username: [redacted]
  indexer_password:

# Wazuh dashboard us
  indexer_username:
  indexer_password:
```





Wazuh SIEM

```
#---> Download and run the Wazuh installation assistant:
curl -sO https://packages.wazuh.com/4.14/wazuh-install.sh && sudo bash ./wazuh-install.sh -a

#---> wazuh-passwords.txt:
sudo tar -O -xvf wazuh-install-files.tar wazuh-install-files/wazuh-passwords.txt
#---> To Access Server:
https://<External-IP>:55000
```

Installing and Configuring Wazuh SIEM:

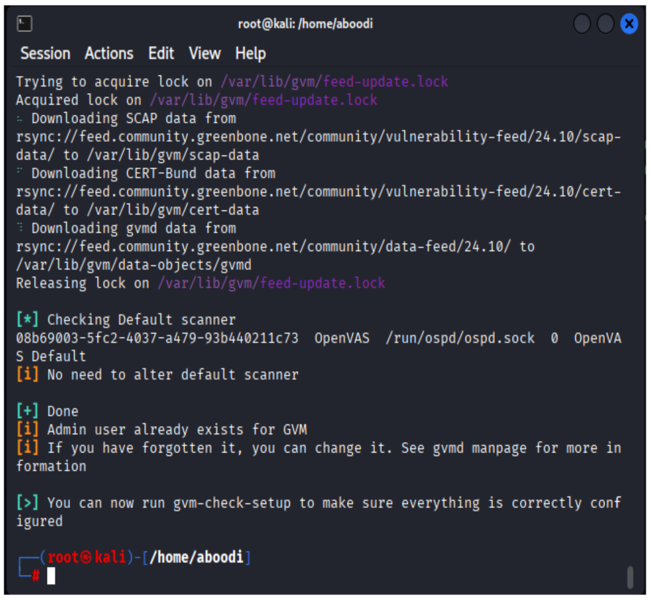
The figures above illustrate the installation and configuration process of Wazuh SIEM, which serves as the project's central security monitoring and event management system.

The installation began by downloading and running the official Wazuh installation script using the command:

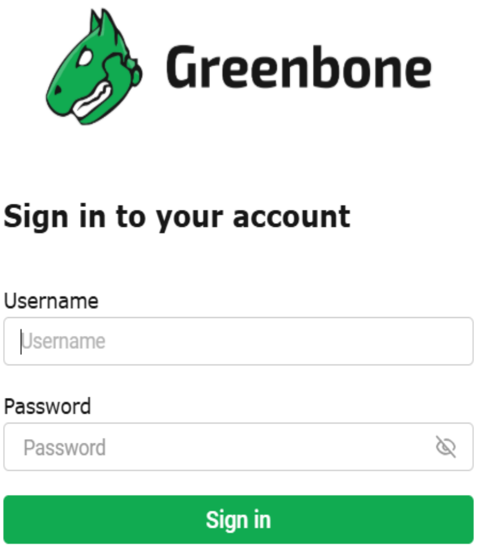
curl -sO https://packages.wazuh.com/4.14/wazuh-install.sh && sudo bash ./wazuh-install.sh -a.

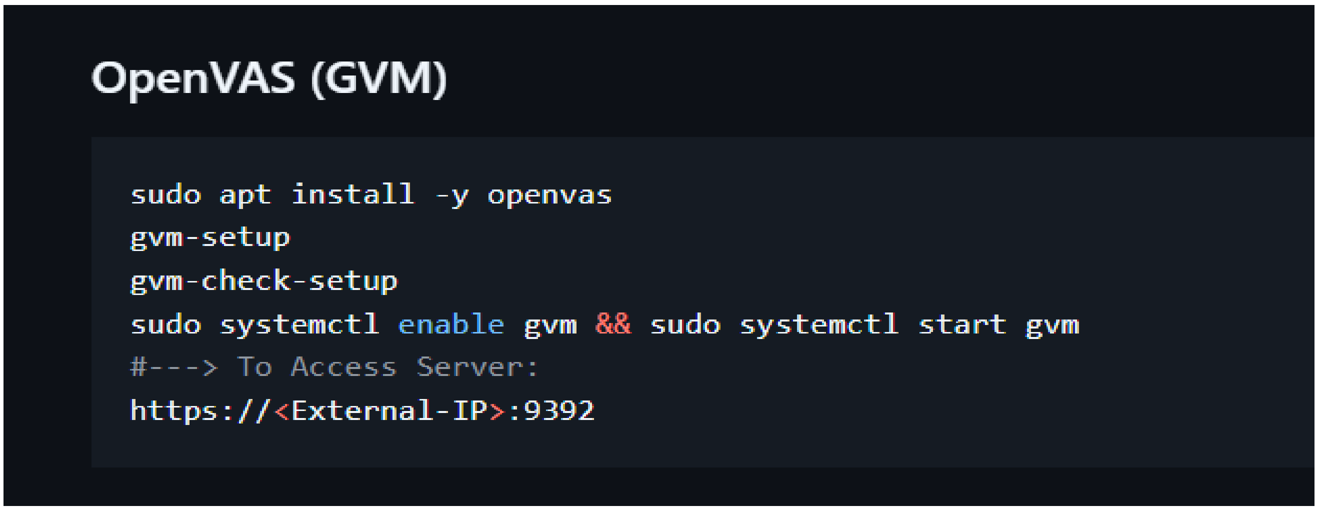
Once installation completed, the file wazuh-passwords.txt was extracted from the setup package, containing credentials for different system users such as admin, anomalyadmin, and kibanaserver. These credentials are used to access and configure the Wazuh dashboard and its indexing services.

Finally, access to the Wazuh web interface was verified through port 55000 or 443 at https://<External-IP>:55000, where the admin user successfully logged in.
This setup provided a real-time monitoring and alerting platform, enabling continuous analysis of system logs and security events across all components of the project environment.



```
Trying to acquire lock on /var/lib/gvm/feed-update.lock
Acquired lock on /var/lib/gvm/feed-update.lock
 Downloading SCAP data from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/scap-
data/ to /var/lib/gvm/scap-data
 Downloading CERT-Bund data from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/cert-
data/ to /var/lib/gvm/cert-data
 Downloading gvmd data from
rsync://feed.community.greenbone.net/community/data-feed/24.10/ to
/var/lib/gvm/data-objects/gvmd
Releasing lock on /var/lib/gvm/feed-update.lock

[*] Checking Default scanner
08b69003-5fc2-4037-a479-93b440211c73  OpenVAS  /run/ospd/ospd.sock  0  OpenVA
S Default
[i] No need to alter default scanner

[+] Done
[i] Admin user already exists for GVM
[i] If you have forgotten it, you can change it. See gvmd manpage for more in
formation

[>] You can now run gvm-check-setup to make sure everything is correctly conf
igured

┌──(root㉿kali)-[/home/aboodi]
└─# 
```





OpenVAS (GVM)

```
sudo apt install -y openvas
gvm-setup
gvm-check-setup
sudo systemctl enable gvm && sudo systemctl start gvm
#---> To Access Server:
https://<External-IP>:9392
```

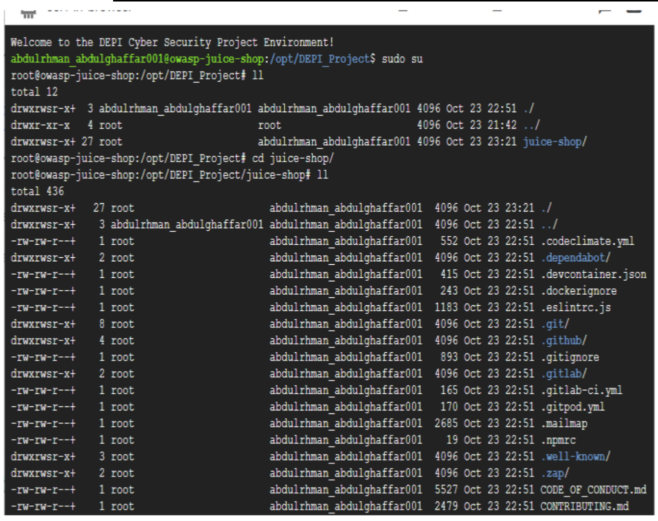
Installing and Configuring OpenVAS (GVM):

The figures above demonstrate the installation and setup process of OpenVAS (Greenbone Vulnerability Manager – GVM), an open-source vulnerability scanning tool used alongside Nessus for comparative analysis.
The installation began by running the commands:
sudo apt install -y openvas → gvm-setup → gvm-check-setup

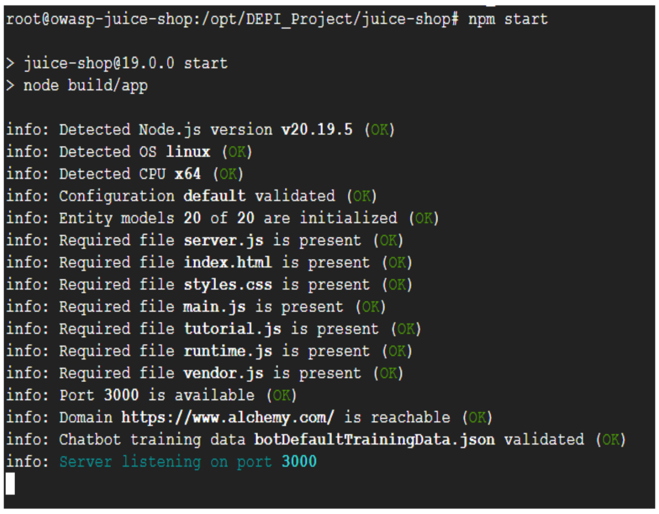
to install, initialize, and verify the scanner configuration. Once completed, the GVM service was enabled and started using systemctl, making the web interface accessible through port 9392 at https://<External-IP>:9392.

During setup, the system automatically created a default admin user, and the scanner feeds were synchronized with Greenbone's official vulnerability database.

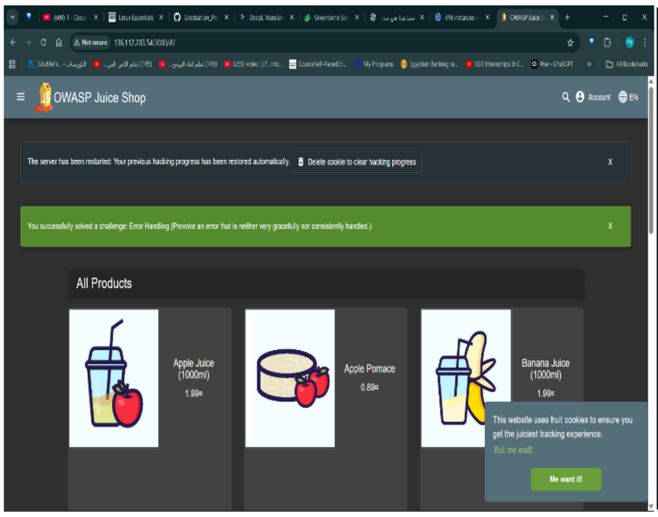
Finally, as shown in the figures, the administrator logged into the Greenbone web interface using the assigned credentials, confirming that OpenVAS was successfully installed and fully operational for vulnerability assessment tasks.



```
Welcome to the DEPI Cyber Security Project Environment!
abdulrhman_abdulghaffar001@owasp-juice-shop:/opt/DEPI_Project$ sudo su
root@owasp-juice-shop:/opt/DEPI_Project# ll
total 12
drwxrwsr-x+  3 abdulrhman_abdulghaffar001 abdulrhman_abdulghaffar001 4096 Oct 23 22:51 ./
drwxr-xr-x   4 root                       root                       4096 Oct 23 21:42 ../
drwxrwsr-x+ 27 root                       abdulrhman_abdulghaffar001 4096 Oct 23 23:21 juice-shop/
root@owasp-juice-shop:/opt/DEPI_Project# cd juice-shop/
root@owasp-juice-shop:/opt/DEPI_Project/juice-shop# ll
total 436
drwxrwsr-x+   27 root                       abdulrhman_abdulghaffar001  4096 Oct 23 23:21 ./
drwxrwsr-x+    3 abdulrhman_abdulghaffar001 abdulrhman_abdulghaffar001  4096 Oct 23 22:51 ../
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   552 Oct 23 22:51 .codeclimate.yml
drwxrwsr-x+    2 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .dependabot/
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   415 Oct 23 22:51 .devcontainer.json
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   243 Oct 23 22:51 .dockerignore
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001  1183 Oct 23 22:51 .eslintrc.js
drwxrwsr-x+    8 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .git/
drwxrwsr-x+    4 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .github/
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   893 Oct 23 22:51 .gitignore
drwxrwsr-x+    2 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .gitlab/
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   165 Oct 23 22:51 .gitlab-ci.yml
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001   170 Oct 23 22:51 .gitpod.yml
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001  2685 Oct 23 22:51 .mailmap
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001    19 Oct 23 22:51 .npmrc
drwxrwsr-x+    3 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .well-known/
drwxrwsr-x+    2 root                       abdulrhman_abdulghaffar001  4096 Oct 23 22:51 .zap/
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001  5527 Oct 23 22:51 CODE_OF_CONDUCT.md
-rw-rw-r--+    1 root                       abdulrhman_abdulghaffar001  2479 Oct 23 22:51 CONTRIBUTING.md
```



```
root@owasp-juice-shop:/opt/DEPI_Project/juice-shop# npm start

> juice-shop@19.0.0 start
> node build/app

info: Detected Node.js version v20.19.5 (OK)
info: Detected OS linux (OK)
info: Detected CPU x64 (OK)
info: Configuration default validated (OK)
info: Entity models 20 of 20 are initialized (OK)
info: Required file server.js is present (OK)
info: Required file index.html is present (OK)
info: Required file styles.css is present (OK)
info: Required file main.js is present (OK)
info: Required file tutorial.js is present (OK)
info: Required file runtime.js is present (OK)
info: Required file vendor.js is present (OK)
info: Port 3000 is available (OK)
info: Domain https://www.alchemy.com/ is reachable (OK)
info: Chatbot training data botDefaultTrainingData.json validated (OK)
info: Server listening on port 3000
```



136.112.203.54:3000/#/

OWASP Juice Shop

The server has been restarted: Your previous hacking progress has been restored automatically. Delete cookie to clear hacking progress

You successfully solved a challenge: Error Handing (Provoke an error that is neither very gracefully nor consistently handled.)

All Products

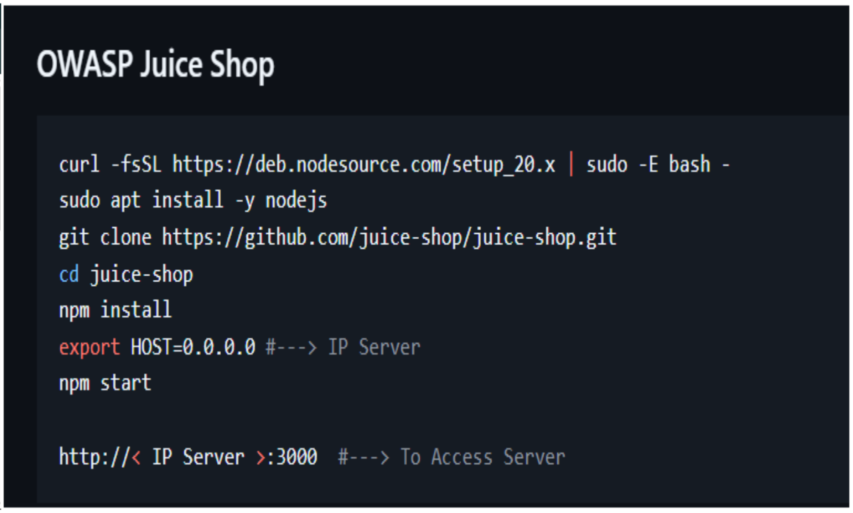
Apple Juice (1000ml) 1.99¤

Apple Pomace 0.89¤

Banana Juice (1000ml) 1.99¤

This website uses fruit cookies to ensure you get the juiciest tracking experience.

But me wait!



OWASP Juice Shop

```
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt install -y nodejs
git clone https://github.com/juice-shop/juice-shop.git
cd juice-shop
npm install
export HOST=0.0.0.0 #---> IP Server
npm start

http://< IP Server >:3000  #---> To Access Server
```

Installing and Running OWASP Juice Shop:

The figures above illustrate the setup and execution of OWASP Juice Shop, the intentionally vulnerable web application used as the target for penetration testing and vulnerability assessment.

The installation began by configuring Node.js and cloning the official Juice Shop repository from GitHub into the shared project directory using the commands shown. After installing the dependencies with npm install, the application was launched with npm start.
As shown in the terminal output, all required configuration files (such as server.js, index.html, and main.js) were validated successfully, and the server was started on port 3000. Once running, the application was accessible at http://<External-IP>:3000, where the OWASP Juice Shop interface loaded correctly, confirming a successful deployment.
This setup provided the core testing environment for the project's vulnerability scanning and remediation phases, enabling the team to simulate real-world attacks in a controlled and ethical manner.

Targets and Authorization:

During this project we performed vulnerability assessments on controlled and authorized environments only. The primary live target was a well-known university; however, to avoid any potential harm or disclosure we do not reveal the institution's identity in this report. All tests on this target were carried out under explicit authorization and with strict limitations to prevent disruption.In addition to the live engagement, we used purpose-built test targets for training and verification — notably Metasploitable (an intentionally vulnerable VM) and a local home-lab instance on a team member's laptop. The Metasploitable package used is available from SourceForge: https://sourceforge.net/projects/metasploitable/. These targets allowed safe validation of scan configurations and remediation workflows without risking production systems.Evidence of authorization for the university engagement is retained in the project records and can be provided to authorized reviewers on request.

Week 1 – Closing Remarks:

The first week successfully established a secure, shared, and reproducible cloud environment for the DEPI Cyber Security Project. Core infrastructure components — virtual machines, network segmentation, firewall rules, and the shared project workspace — were deployed and verified, and all primary tools (Nessus, OpenVAS, Wazuh, OWASP Juice Shop) were installed and reachable. Access controls and collaboration settings were configured to ensure every team member can work within a consistent environment.
With the foundation now in place, the project is ready to move from preparation to action. In Week 2 we will perform systematic vulnerability scans, collect and compare findings from multiple scanners, and produce the initial analysis that will drive our remediation plan. The following chapter details the scanning strategy, policies, and expected deliverables.