College Name	**University of Hertfordshire SEGi College Subang Jaya**		
Programme Name	**BACHELOR OF SCIENCE (HONS) COMPUTER SCIENCE (CYBER SECURITY AND NETWORKS)**		
Module Name	**CYBER SECURITY AND NETWORKS PROJECT**	Module Code	**6COM1040**
		Semester	**September 2025**
Module Leader	**Dr. Aneshkumar Thangaveloo**	Assessment Type	**Implementation Evidence**
Lecturer Name	**Ms. Nur Diana Madinah Binti Ab Hadi**		

Student's declaration

I hereby certify that this assignment is my own work and where materials have been used from other resources, they have been properly acknowledged. I also understand I will face the possibility of failing the module if the content of this assignment is plagiarized.

No.	Name	Student ID	Signature / Initial
1	PATRICK ROGERS	SCSJ2100424	PAT

Date:

Release Date		Submission Due Date		Marks obtained:
Date Received		Student's work assessed by / date		

Module Leader's Feedback.

Module Leader's comments / feedback	
Student's comments	

Table of Contents

1.0 Implementation of Security Measures 4
1.1 UFW Firewall Configuration 4
1.2 Docker Containerization of Web Application 5
1.3 Suricata Intrusion Detection System 6

Table of Figures

Figure 1: UFW Firewall Status 4
Figure 2 Docker Containers Running 5
Figure 3 Application Accessible from Browser 5
Figure 4 Suricata Running Successfully 6

1.0 Implementation of Security Measures

Three primary security measures were implemented: UFW firewall, Docker containerization, and Suricata IDS. These measures were carefully chosen to address A2Z Corporation's critical security risks and provide multiple layers of defence against potential attacks.

1.1 UFW Firewall Configuration

The Ubuntu server's UFW firewall was configured with a default policy to deny all incoming traffic and allow all outgoing traffic. Only essential ports 80 (HTTP), 443 (HTTPS), and 3000 (Juice Shop) were explicitly allowed, and SSH access (port 22) was restricted to the trusted IP of the administrative Kali machine.

```
vboxuser@A2z:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
80/tcp                     ALLOW IN    Anywhere
443/tcp                    ALLOW IN    Anywhere
3000/tcp                   ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
80/tcp (v6)                ALLOW IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)
3000/tcp (v6)              ALLOW IN    Anywhere (v6)
```

Figure 1: UFW Firewall Status

This configuration protects the system from unauthorized access and brute-force attacks by ensuring that only necessary services are exposed to the internet. Any attempts to connect to non-essential ports are automatically blocked, effectively preventing attackers from exploiting unused services. By restricting SSH access to a single trusted source, the firewall also mitigates the risk of remote login attacks. Overall, UFW acts as a first line of defence, controlling network traffic and reducing the attack surface of the Ubuntu server.

1.2 Docker Containerization of Web Application

The OWASP Juice Shop application was deployed using Docker containers, exposing the application on port 3000. Containerization isolates the application from the host operating system and other system services.

```
vboxuser@A2Z:~$ sudo docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATE
D       STATUS       PORTS                                         NAM
ES
a85f1cc7958c   bkimminich/juice-shop     "/nodejs/bin/node /j…"   2 hour
s ago   Up 2 hours   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp   har
dcore_panini
606193d7c9b2   nginx:latest              "/docker-entrypoint.…"   2 hour
s ago   Up 2 hours   0.0.0.0:80->80/tcp, [::]:80->80/tcp           web
server
```

Figure 2 Docker Containers Running

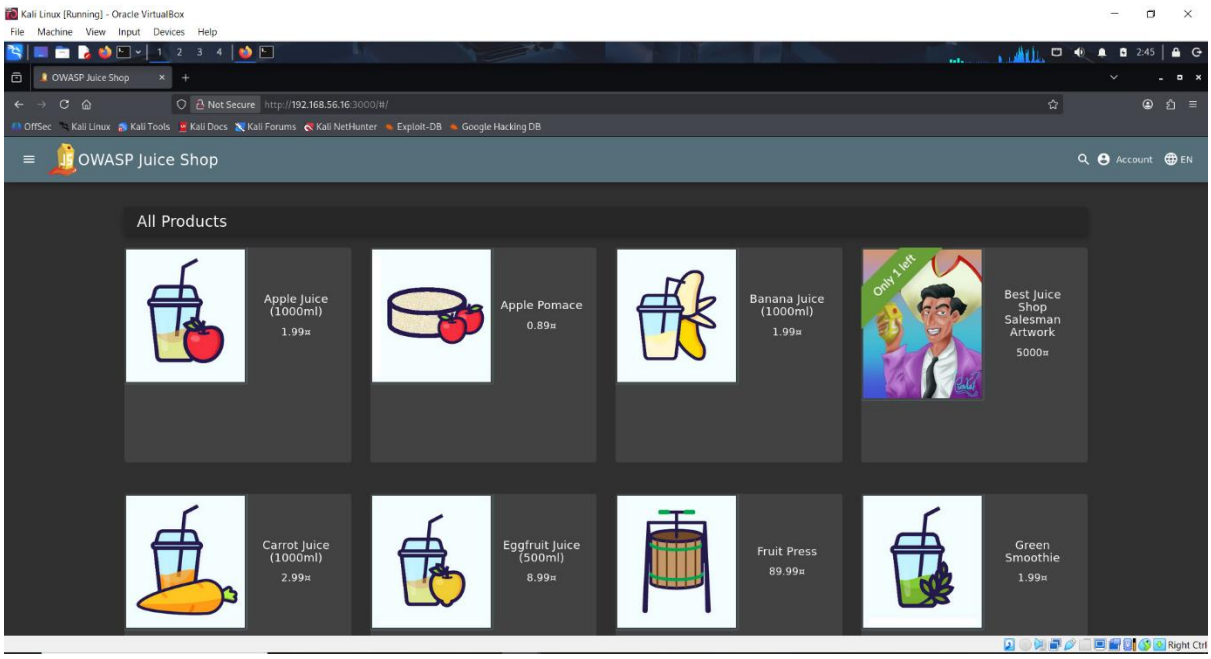

Figure 3 Application Accessible from Browser

Docker enhances security by containing potential breaches within the container environment, preventing attackers from gaining root access to the host server or moving laterally to other systems in the network. If a vulnerability in the web application is exploited, the attacker remains confined to the container, and any malicious changes are erased when the container is

restarted. Containerization also simplifies patch management and rollback, allowing administrators to quickly update or replace compromised applications without impacting other services.

1.3 Suricata Intrusion Detection System

Suricata was installed and configured to monitor network traffic on the DMZ interface (enp0s8). Rule sets were loaded successfully, and the IDS engine started in detection mode, analyzing both IP-only and payload-based signatures.

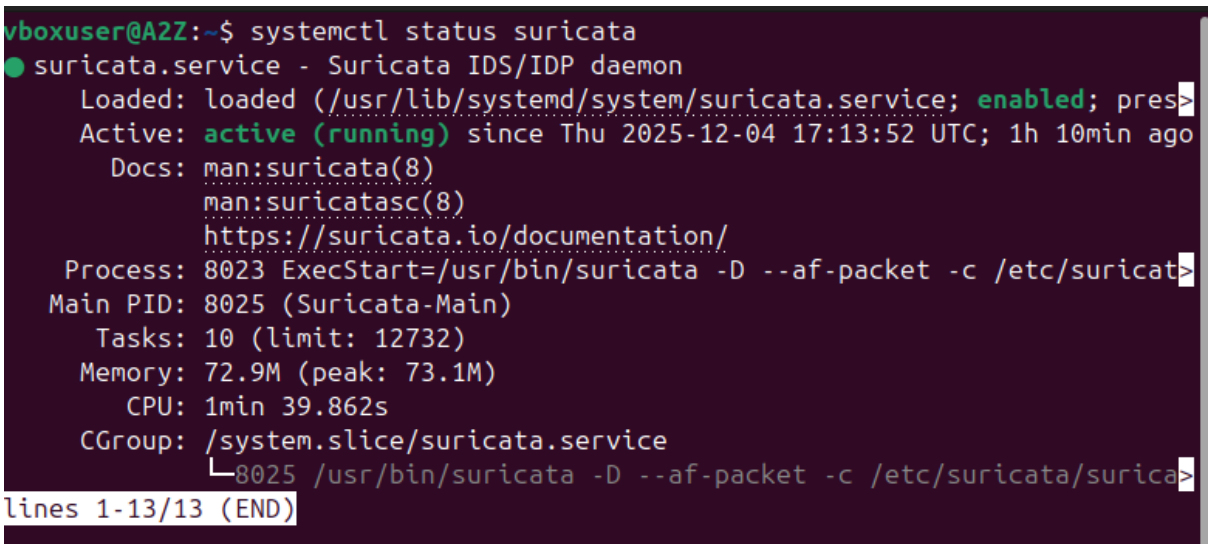

Figure 4 Suricata Running Successfully

Suricata detects real-time tracking, port scanning, operating system fingerprinting, and brute-force attacks. For example, when an attacker runs an aggressive Nmap scan, Suricata quickly detects and logs strange patterns such as SYN flood attempts or OS detection probes. This allows administrators to respond immediately to potential threats, whether by blocking offending IP addresses or reviewing suspicious activity. Suricata thus serves as a proactive security solution, supporting the firewall and container isolation by detecting threats that get past perimeter protections.

By combining these three measures firewall, containerization, and IDS the system implements a defence-in-depth strategy. Each layer addresses specific threat vectors: UFW limits access to

essential services, Docker isolates potentially vulnerable applications, and Suricata detects and logs malicious activity for timely intervention.