National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Bhanu Prakash Palukuri
Student ID: 23413999

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Bhanu Prakash Palukuri
Student ID:	23413999
Programme:	Cloud Computing **Year:** 2026
Module:	MSc Research Project
Supervisor:	Shaguna Gupta
Submission Due Date:	28/01/2026
Project Title:	Integrating Zero Trust, Threat Detection, and Encryption

Word Count: **Page Count:17**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Bhanu Prakash Palukuri

January 2026

1 Introduction

The practical design of a holistic cloud security framework is introduced in a setting of Amazon Web Services (AWS) in this section. It is concerned with the functionality of converting theoretical security into a real-world setting that helps to increase confidentiality, integrity, and availability of the cloud resources. The basic infrastructure including EC2, VPC, S3, etc are properly implemented and strengthened with a Zero Trust security model, multi-factor authentication, and enhanced SSH access control.

Automated threat detection processes are also incorporated with the help of tools such as Fail2ban and custom log analysis to detect and prevent harmful behavior in real-time. TLS and AWS Key Management Service are used to ensure strong encryption of data in transit and the rest. To prove the efficiency and stability of the secured AWS environment, the system is ultimately tested by the means of controlled security testing, comprising of the network scan, denial-to-service testing, and brute-force attacks.

1.1 System Configuration Setup

1.1.1 System Configuration Setup

Software Used

- AWS Lab
- Kali Linux
- WSL (Windows Subsystem for Linux)

Hardware Used

- Windows 10
- 8 GB RAM

2 Project Implementation

2.1 Environmental Setup

2.1.1 Step 1: Zero Trust Hardening

Figure 1: Installation of Google Authenticator

The success of the deployment of Google Authenticator on the server is confirmed by the installation of the packages and the available opportunities to achieve multi-factor authentication, thus increasing access controls due to the introduction of additional verification measures in accordance with the requirements of the Zero Trust model.

Figure 2: Configuring the Google Authenticator

Configuration introduces the generation of tokens, time-skewed exploitation and "brute-force" guarding. Allowing rate-limiting decrease the chances of abuse of authentication, and provide greater resistance to repeated logins and overall security of the authentication state.

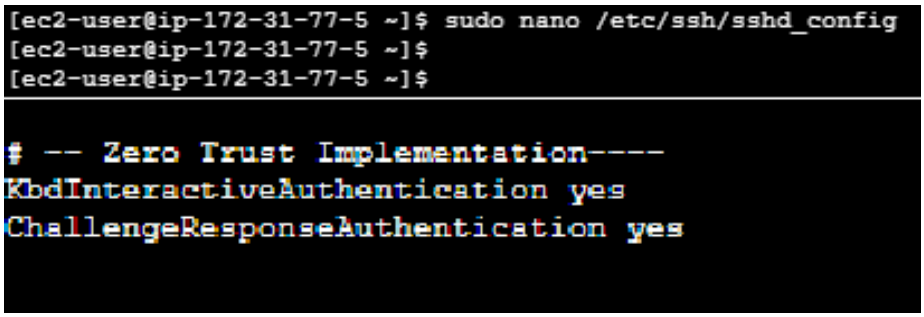

Figure 3: Enabling Keyboard-interactive Authetication

The SSH is configured so that there is the keyboard-interactive authentication has been activated and challenge response authientication is also activated that the security to the user logins on top of the password authentication can be there.

Figure 4: Checking the configuration file

Configuration check gets validation of syntax correctness whereby modified SSH parameters are verified to work correctly. This process helps avoid the risks of misconfiguration and ensures that secure authentication policies are implemented without appropriately interrupting the access to the servers.

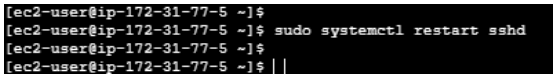

Figure 5: Restarting sshd configuration

The restart of the SSH daemon with new authentication options enables MFA and new security policies that are immediately enforced and the server immediately start to operate with enhanced Zero Trust.

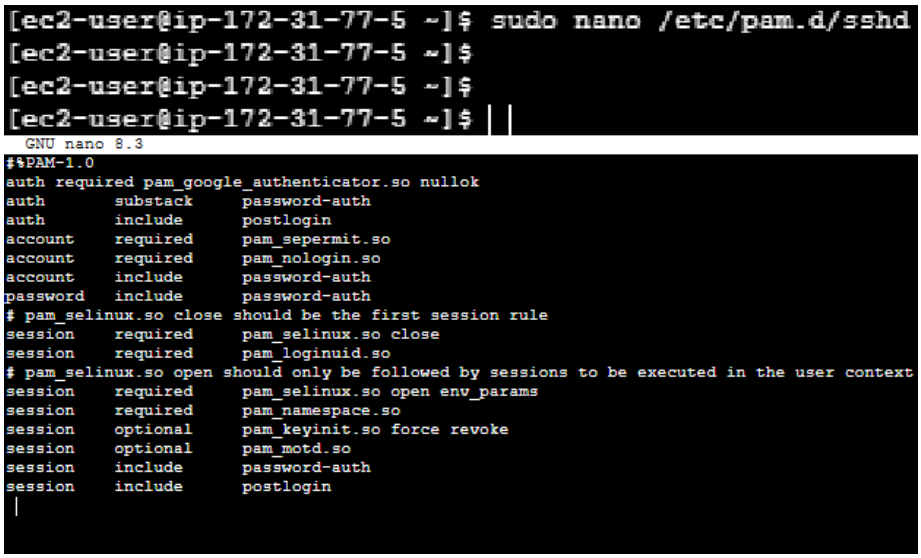

Figure 6: Configuring PAM File

PAM settings are modified to provide authentication control, Google refresher, session management, and secure login activities, which enforces orderly access by SSH access policy and controlled session policy.

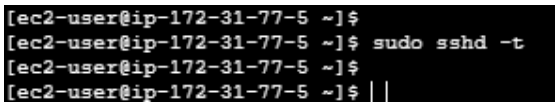

Figure 7: Testing the PAM File

The SSH daemon setup is tested with a sudo sshd -t command to ensure that the rules in authentication are free of errors before implementing changes on the server to ensure one operates the server securely.

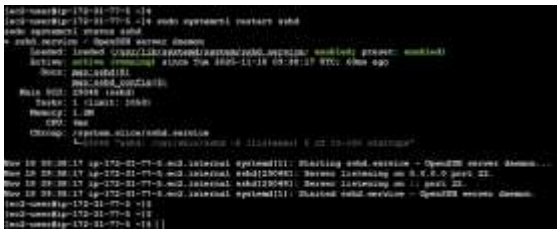

Figure 8: Restarting SSHD and Checking SSHD status

SSHD is restarted and its health is checked, which proves that it has been loaded successfully, is running and is bound to port 22, which is a working service of secure-shell.

```
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$ sudo nano /etc/ssh/sshd_config
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$
```

```
# -- Zero Trust Implementation----
KbdInteractiveAuthentication yes
ChallengeResponseAuthentication yes
PermitRootLogin no
PubkeyAuthentication yes
PermitEmptyPasswords no
PasswordAuthentication yes
UsePAM yes
AllowAgentForwarding no
AllowTcpForwarding no
GatewayPorts no
X11Forwarding no
MaxAuthTries 3
LoginGraceTime 30s
ClientAliveInterval 300
ClientAliveCountMax 2
UseDNS no
PermitUserEnvironment no
# AuthenticationMethods publickey keyboard-interactive
```

Figure 9: Applying Zero Trust Settings

SSH settings are adjusted to impose requirements on stronger authentication, to refuse unsafe features, to restrict the number of sessions, and to apply the principles of Zero Trust to provide better system security and keeping the attack surface at the minimal level.

```
[ec2-user@ip-172-31-77-5 bin]$ sudo passwd ec2-user
Changing password for user ec2-user.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-77-5 bin]$
```

Figure 10: Creating Password for EC2 User

The EC2 user is given a new password which changes authentication tokens and allows the user to log in with a password upon activation of more stringent access control and stricter security enforcement.

2.2 Step 2: Threat Detection

Figure 11: Installation of Fail2ban (Anomaly Detection for SSH)

The information in the package installation and system logs proves that Fail2ban is deployed successfully, and the dependencies, repositories, and service components are installed correctly and strengthening the anomaly detection ability of SSH.

Figure 12: Enable monitoring of suspicious SSH activity

Configuration entries can turn SSH protection on with set ports, filters, log paths, and ban rules to be reflected, and block repeated and failed authentication attempts to mitigate them with automatic blocking.

Figure 13: Start the anomaly monitor

The start and verification of the operating status and its running parameters are allowed and activated by service commands. Effective activation allow to monitor the process of login and prevent brute-force attacks in advance.

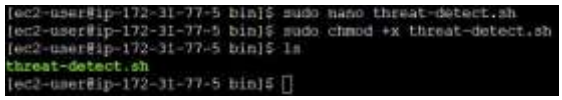

Figure 14: Adding Local Log Analyzer

The commands create a new script, grant execution permission, and verify its presence in the directory. This establishes a functional local log-analysis tool ready for automated threat-detection tasks.

2.3 Step 3: Encryption

Figure 15: AWS CLI Configuration and STS Identity Verification on EC2

AWS CLI commands are configured credentials and identify with STS to ensure access to the going-on programs is secure on the EC2 instance and the authenticity of the IAM utilization in cloud operations.

Figure 16: Listing the existing KMS keys

The aws kms list-keys retrieve a list of available KMS keys and its KeyId and

ARNs, ensuring the presence of several customer managed keys. Applicable in auditing encryption resources and ensuring that there is availability of keys to be operated on later.

Figure 17: Verifying Prerequisites

The command aws sts get-caller-identity shows identity of callers, IAM roles, configuration and EC2 instances metadata. Authenticates using AWS credentials, region configuration and environment preparedness with key management or encryption operation.

```
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$ cat > key_policy.json << 'EOF'
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow root",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::004220161871:root" },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow role to use key",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::004220161871:role/voclabs"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Resource": "*"
    }
  ]
}
EOF
[ec2-user@ip-172-31-77-5 ~]$
```

Figure 18: Creating a KMS Key

The cat > key policy.json << 'EOF' command displays development of a critical policy that allows authorization to root and that of a particular IAM role. Provides protection to the access control operations of encryption, decryption and key generation of data.

Figure 19: Creating a CMK

Provision on-demand KMS key, which is managed by the customer, and the metadata is shown, with KeyId, ARN, type and status. Certifies that it successfully provides a symmetric encryption key which uses in secure data operations.

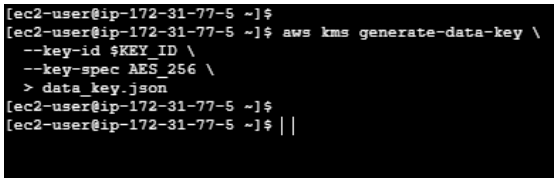

Figure 20: Generate a Data Key

Creates a data key based on KMS key and AES-256 specification. Generates encrypted key material and plaintext material needed to remain consistent with application-level encryption processes that are secure.

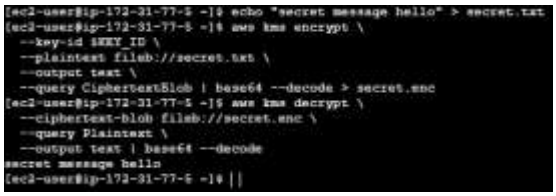

Figure 21: Testing the KMS Encryption and Decryption

KMS encryption is effective and ensures the confidential message is secure thereby illustrating a smooth encryption and decryption process with AWS Key Management Service.

Figure 22: Installing nginx and certbot

Installing of nginx and certbot provides the base web server and automation tools to create a secure site via the installation of automation tools in the form of nginx and certbot.

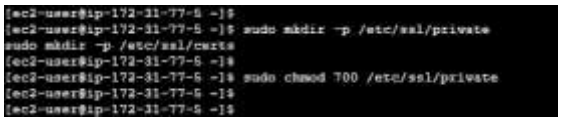

Figure 23: Creating necessary directories and setting the permissions

The most important security measure is to create directories with the right permissions in order to work with sensitive files of TLS certificates correctly.

Figure 24: Generating TLS Certificate

Using Certbot to generate a TLS certificate happens to be an automated method of achieving a trusted, valid certificate to use in HTTPS.

```
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$ sudo nano /etc/nginx/conf.d/ssl.conf
[ec2-user@ip-172-31-77-5 ~]$
[ec2-user@ip-172-31-77-5 ~]$
```

```
  GNU nano 8.3
server {
    listen 443 ssl;
    server_name ec2-server;

    ssl_certificate /etc/ssl/certs/server.crt;
    ssl_certificate_key /etc/ssl/private/server.key;

    location / {
        root /usr/share/nginx/html;
        index index.html;
    }
}
```

Figure 25: Configuration of the NGINX to use TLS

The password of the configuration of the NGINX TLS makes the server bound to port 443 and give the right path of the key and certificate of the server.

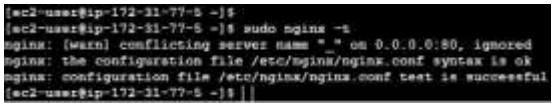

Figure 26: Testing the nginx configuration

The configuration test is successful, and for testing the command sudo nginx -t command has been used in the AWS EC2 terminal accordingly.

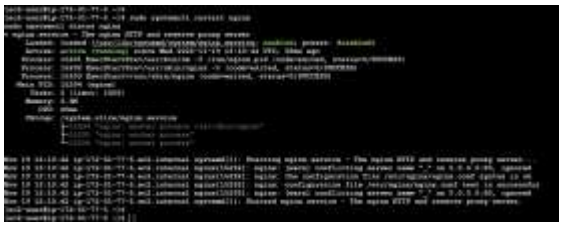

Figure 27: Restarting the NGINX

NGINX service is successfully started and with a successful restart, but the logs of the start-ups also indicate the same server name alert, as well.

2.4 Step 4: Testing

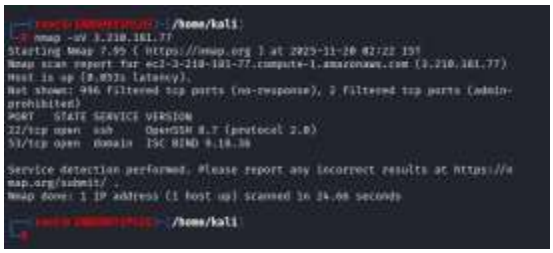

Figure 28: Nmap scan using Public IP of AWS

A Nmap scan has been done using the public IP of the AWS EC2 Instance. The scan shows that of one of AWS EC2 Instance ports 22 and 53 are open, which means that the AWS server has an SSH port and a DNS port open, indicating that its attack surface is potentially well-lockdown.

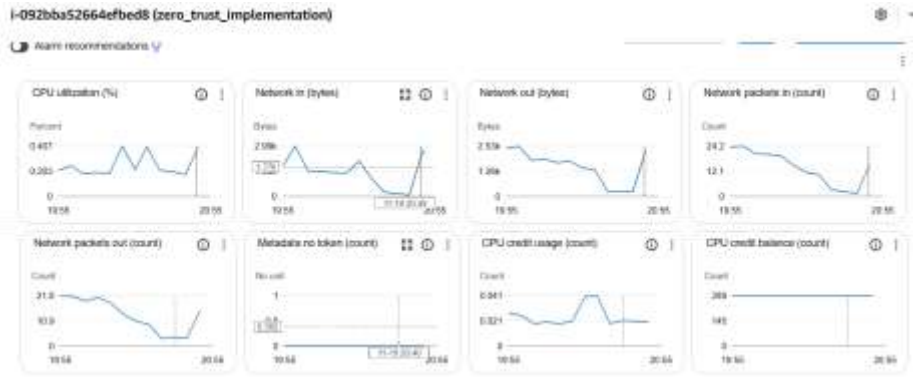

Figure 29: Monitoring EC2 Instance during Nmap scan

EC2 indicators reveal that in the measured period, CPU usage and credit balance are in a good condition and did not show any effects of the nmap scan on the performance.

Figure 30: Performing DoS attack on AWS Server

SYN flood DoS attack on port 22 result in 100 percent loss of packets by the attacker to prove the resilience of server or the presence of network filters.

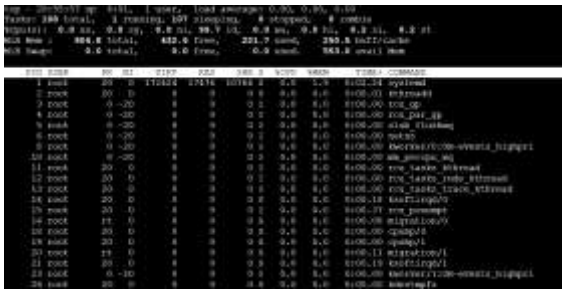

Figure 31: Monitoring the system during DoS Attack

System monitoring in case of “DoS attack” indicate normal process load and minimal resources utilization, as evidence of the stability of the server in the state of stress.

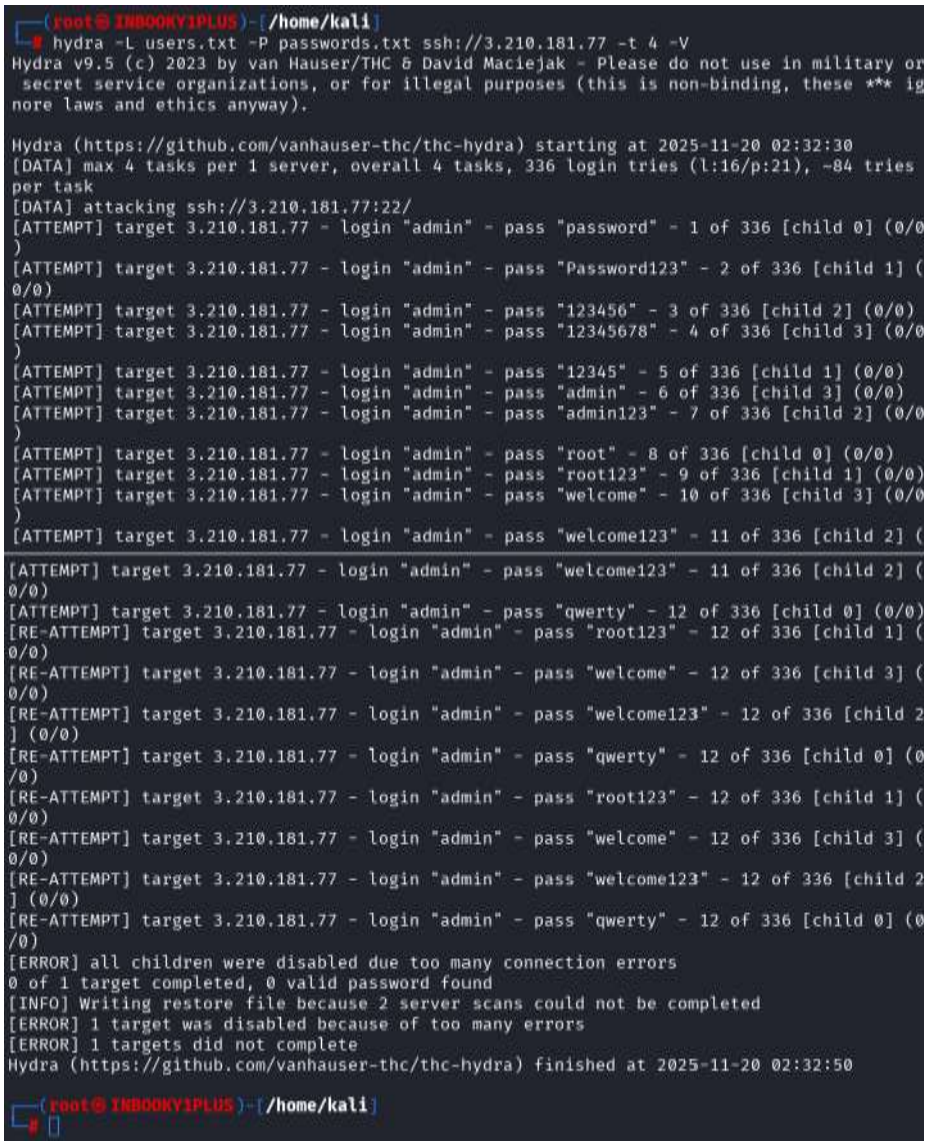

Figure 32: Performing Brute-Force Attack on AWS Server

A brute force attack with Hydra tries the SSH service with a list of usernames and passwords that have been found to be common and tries to verify the strength of the credential. The attack has been failed and hence it is proved that the AWS sever is blocking unauthorized password attempts.

Figure 33: Monitoring and Detecting Failed Logging Attempts

The monitoring tools are proactively alerting and recording when a login fails, and this fact is important to determine possible brute force attacks.

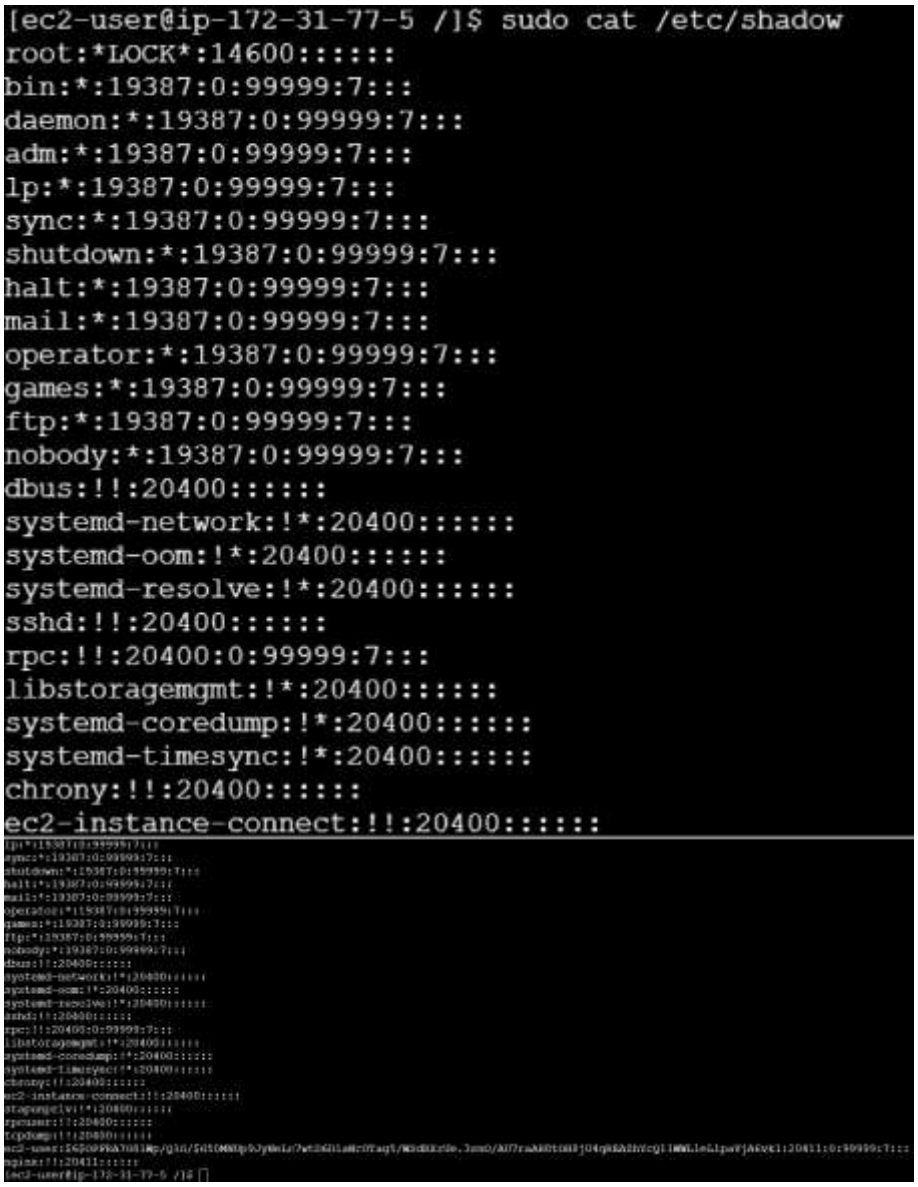

Figure 34: Password Encryption Test on AWS Server

The system processes and encryption tests have been recorded in the logs of the system, which gives evidence of continued security configuration and monitoring processes on the server.

References

1. Nutalapati, P. (2023). Zero Trust Architecture in Cloud-Based Fintech Applications. Journal of Artificial Intelligence & Cloud Computing, 2(1), 1-8. https://srcpublishers.com/index.php/ai-cloud-computing/article/view/150

2. Reddy, A. R. P. (2021). The role of artificial intelligence in proactive cyber threat detection in cloud environments. NeuroQuantology, 19(12), 764-773. https://www.researchgate.net/profile/Abhilash-Reddy-Pabbath-Reddy/publication/378693448_THE_ROLE_OF_ARTIFICIAL_INTELLIGENCE_IN_PROACTIVE_CYBER_THREAT_DETECTION_IN_CLOUD_ENVIRONMENTS/links/65e5351bc3b52a11700a2759/THE-ROLE-OF-ARTIFICIAL-INTELLIGENCE-IN-PROACTIVE-CYBER-THREAT-DETECTION-IN-CLOUD-ENVIRONMENTS.pdf

3. Gudimetla, S.R., 2024. Data encryption in cloud storage. International Research Journal of Modernization in Engineering Technology and Science, 6, pp.2582-5208. https://www.researchgate.net/profile/Sandeep-Gudimetla/publication/379567678_DATA_ENCRYPTION_IN_CLOUD_STORAGE/links/660ee8e0f5a5de0a9ffb8904/DATA-ENCRYPTION-IN-CLOUD-STORAGE.pdf

4. Ortega-Fernandez, I., & Liberati, F. (2023). A review of denial of service attack and mitigation in the smart grid using reinforcement learning. Energies, 16(2), 635. https://www.mdpi.com/1996-1073/16/2/635

5. Sharma, P., Tanwar, S., & Kukreja, V. (2024). Implementation of Brute Force Attack. In Emerging Trends in IoT and Computing Technologies (pp. 53-57). CRC Press. https://www.taylorfrancis.com/chapters/edit/10.1201/9781003535423-10/implementation-brute-force-attack-priyanshu-sharma-sarvesh-tanwar-vinay-kukreja

6. Everson, D., & Cheng, L. (2024). A survey on network attack surface mapping. Digital Threats: Research and Practice, 5(2), 1-25. https://dl.acm.org/doi/abs/10.1145/3640019