Comprehensive Report on Secure Linux Server Deployment Project

Sriker Paturi

November 28th, 2024

Contents

1 Introduction

This report outlines the Secure Linux Server Deployment project, detailing its objective, theoretical background, implementation techniques, and the tools utilized. The information provided aims to enhance understanding of server security best practices and demonstrate a practical implementation that hiring managers can easily relate to.

—

2 Objective

The primary objective of this project is to deploy a Linux-based server capable of:

- Achieving high uptime (99.8% or higher) for consistent availability.
- Providing robust security through SSH key-based authentication and firewalls.
- Simplifying server management with automated maintenance tools and scripts.

—

3 Theoretical Background

3.1 Linux as a Server Platform

Linux is an open-source operating system widely used in server environments due to its:

- **Stability:** Ensures high uptime and reliability.
- **Security:** Provides built-in tools and frameworks for secure configurations.
- **Customization:** Offers flexibility for a wide range of server applications.

3.2 Key Security Concepts

SSH (Secure Shell): SSH is a cryptographic network protocol for secure remote login and command execution. It prevents unauthorized access through encryption and authentication mechanisms.

Firewalls: A firewall monitors and controls incoming and outgoing network traffic based on predetermined security rules, acting as a barrier between trusted and untrusted networks.

Monitoring and Maintenance: Tools like **Fail2Ban** and **Logwatch** help in identifying and mitigating potential threats by analyzing logs and automating response mechanisms.

—

4 Tools and Platforms Used

4.1 Ubuntu

Why Ubuntu?

- User-friendly and well-documented.
- Broad community support for troubleshooting.
- Long-term support (LTS) releases ensure stability.

4.2 OpenSSH

Benefits:

- Ensures encrypted communication between the client and server.
- Provides advanced authentication mechanisms such as public key authentication.

4.3 UFW (Uncomplicated Firewall)

Advantages:

- Simplifies firewall management with an easy-to-use command-line interface.
- Offers pre-configured rules for common services like HTTP and SSH.

4.4 Fail2Ban

Fail2Ban is a log-parsing utility that protects servers from brute-force attacks by banning IPs that show malicious intent.

- Dynamically updates firewall rules to block attackers.
- Sends alerts for suspicious activity.

4.5 Logwatch

Logwatch provides a daily summary of server activity, including errors, login attempts, and potential security issues. It simplifies log analysis for system administrators.

4.6 Nginx and Let's Encrypt

Nginx: A high-performance web server and reverse proxy server, ideal for handling high loads efficiently.

Let's Encrypt: A free, automated, and open Certificate Authority (CA) that simplifies the process of obtaining and renewing SSL/TLS certificates for secure HTTPS connections.

—

5 Step-by-Step Implementation

5.1 Setting Up the Server

Installation and Updates: Use official ISO images for installation. Regularly update the server to patch vulnerabilities:

```
sudo apt update && sudo apt upgrade -y
```

5.2 Securing SSH Access

Key-Based Authentication: SSH keys are more secure than passwords because they use asymmetric cryptography.

```
ssh-keygen -t rsa -b 4096
ssh-copy-id user@server_ip
```

Configuration Hardening: Disable root login and password authentication:

```
sudo nano /etc/ssh/sshd_config
# Set:
PermitRootLogin no
PasswordAuthentication no
```

5.3 Configuring the Firewall

UFW simplifies the process of allowing or blocking specific services:

```
sudo ufw allow 2222/tcp
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable
```

5.4 Monitoring and Maintenance

Fail2Ban Setup: Install and configure Fail2Ban to prevent brute-force attacks:

```
sudo apt install fail2ban
sudo systemctl enable fail2ban
```

Logwatch Setup: Schedule daily reports for server activity:

```
sudo logwatch --output mail --mailto you@example.com --detail high
```

5.5 Optimizing Server Performance

Nginx Installation:

```
sudo apt install nginx
sudo systemctl start nginx
sudo systemctl enable nginx
```

Let's Encrypt for HTTPS: Secure your site with SSL/TLS certificates:

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx
```

—

6 Automation Scripts

6.1 Backup Automation

Backup script to archive important files:

```
#!/bin/bash
tar -czvf /backup/backup_$(date +%F).tar.gz /important_data
```

Schedule using cron:

```
crontab -e
0 0 * * * /path/to/backup_script.sh
```

—

7 Troubleshooting Techniques

7.1 SSH Issues

Verify permissions on SSH key files:

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
```

7.2 Firewall Blockage

Allow a specific IP:

```
sudo ufw allow from <IP_Address>
```

—

8 Conclusion

This comprehensive approach to deploying and securing a Linux server demonstrates practical skills in server management, security, and automation. The knowledge and tools used align with industry standards, making this project a strong addition to your professional portfolio.