**A**

**Project**

**On**

Securing and Hardening Linux Server

**For the partial fulfilment of the Course**

***Linux Shell Programming (CSET-213/CBCA221)***

**Academic Year: 2024-24 (Odd Semester)**

**Submitted by**

**Anshula Lal**

**Under The Supervision of**

Dr Vimal Kumar/Dr. Debjani Ghosh/Dr Nitin Shukla

School of Computer Science Engineering & Technology

Bennett University (The Times Group), Greater Noida, UP-201310, India

****

**SCHOOL OF COMPUTER SCIENCE ENGINEERING & TECHNOLOGY**

**BENNETT UNIVERSITY (THE TIMES GROUP),**

**GREATER NOIDA, UP-201310, INDIA**

**November 2024**

**Contents**

[Introduction 4](#_Toc183719775)

[Background 4](#_Toc183719776)

[Project Scope 5](#_Toc183719777)

[In-Scope 5](#_Toc183719778)

[Out-of-Scope 5](#_Toc183719779)

[Deliverables 5](#_Toc183719780)

[Project Objectives 6](#_Toc183719781)

[Requirement Specification 7](#_Toc183719782)

[1. Hardware and Software Requirements 7](#_Toc183719783)

[2. Configuration and Management Requirements 7](#_Toc183719784)

[Functional and Non-Functional requirements 8](#_Toc183719785)

[Functional Requirements 8](#_Toc183719786)

[Non-Functional Requirements 8](#_Toc183719787)

[Implementation 10](#_Toc183719788)

[1. Creating A Linux Server: 10](#_Toc183719789)

[2. Installing System Updates and Patches 10](#_Toc183719790)

[3. Configure automatic Update 11](#_Toc183719791)

[4. Set Password Policy and Complexity 12](#_Toc183719792)

[Resources Used 15](#_Toc183719793)

[References 15](#_Toc183719794)

# Introduction

In an age where cyber threats are ever-evolving, securing and hardening Linux servers has become a critical aspect of IT infrastructure management. This project aims to reinforce the security of Linux servers by implementing a comprehensive set of best practices and advanced techniques. The project's scope includes identifying potential vulnerabilities, configuring firewall rules, enforcing strong authentication mechanisms, and employing system monitoring tools to detect and mitigate threats in real-time.

Furthermore, the project will cover the application of security patches, file integrity monitoring, and the use of intrusion detection systems to provide multiple layers of defences. By adopting a proactive approach to security, the project seeks to minimize the attack surface and ensure the Linux server environment is robust against unauthorized access and malicious activities. The outcome of this project will be a fortified Linux server setup, capable of withstanding a wide array of cyber threats, thereby enhancing the overall security posture of the organization.

# Background

Linux servers form the backbone of numerous organizations' IT infrastructures due to their reliability, scalability, and open-source nature. However, the very qualities that make Linux attractive also present unique security challenges. The growing sophistication of cyber threats necessitates robust measures to protect sensitive data, maintain service integrity, and ensure compliance with regulatory standards.

Historically, Linux servers have been targeted by various cyber-attacks, including unauthorized access attempts, malware infections, and denial-of-service attacks. These incidents underscore the critical need for a comprehensive security strategy. This project draws on well-established security principles and cutting-edge practices to address these threats. By focusing on proactive and reactive security measures, the project aims to build a resilient and secure Linux server environment.

The project builds upon established security frameworks and leverages the extensive range of security tools available in the Linux ecosystem. These tools are designed to minimize the attack surface, detect potential intrusions early, and respond effectively to security breaches. As cyber threats continue to evolve, the project emphasizes continuous monitoring, regular updates, and the adoption of the latest security practices to maintain a high level of security over time.

Ultimately, this project seeks to create a blueprint for securing and hardening Linux servers that can be adapted and implemented across various organizational contexts, ensuring that critical systems remain protected against emerging threats.

# Project Scope

## In-Scope

Vulnerability Assessment:

* Conduct a comprehensive audit to identify security vulnerabilities in the existing Linux server infrastructure.
* Use automated tools and manual methods to assess the server's security posture.

Configuration of Security Controls:

* Implement and configure firewalls, intrusion detection/prevention systems, and other security tools.
* Enforce strong authentication mechanisms and user access controls.
* Configure and enforce network security policies, including VPN setup for secure remote access.

System Hardening:

* Apply best practices for system hardening, including disabling unnecessary services and ports.
* Configure secure settings for server software and applications.
* Regularly update and patch the operating system and installed software to mitigate vulnerabilities.

Data Protection:

* Implement data encryption for sensitive information at rest and in transit.
* Set up secure backup and recovery processes.

Monitoring and Logging:

* Establish centralized logging and real-time monitoring systems to detect and respond to security incidents.
* Implement log analysis tools to identify patterns and potential threats.

## Out-of-Scope

Physical Security:

The project will not cover physical security measures for the data center or server location.

Non-Linux Systems:

The project will focus solely on Linux servers and will not include securing other operating systems.

Development of Custom Security Tools:

The project will use existing, widely accepted security tools and best practices rather than developing new security solutions from scratch.

## Deliverables

Security Assessment Report:

A detailed report identifying current vulnerabilities and risks.

Security Configuration Guidelines:

Documented best practices and configurations implemented during the project.

# Project Objectives

Identify Vulnerabilities: Conduct a comprehensive security assessment to identify potential vulnerabilities in the existing Linux server environment.

Implement Security Best Practices: Apply industry-standard security practices and configurations to enhance the baseline security of the Linux servers.

Strengthen Authentication Mechanisms: Enforce strong authentication and authorization protocols to ensure that only authorized users can access critical systems and data.

Configure Firewall and Network Security: Establish robust firewall rules and network configurations to protect against external threats and unauthorized access.

Apply Regular Security Updates: Develop a schedule for regular updates and patch management to ensure the system remains protected against newly discovered vulnerabilities.

Deploy Intrusion Detection and Prevention Systems: Implement intrusion detection and prevention systems to monitor and respond to suspicious activities in real-time.

Conduct Continuous Monitoring and Auditing: Establish continuous monitoring and auditing processes to detect and address security incidents promptly.

Enhance Data Protection: Ensure the protection of sensitive data through encryption, secure backups, and access controls.

These objectives aim to create a fortified, resilient Linux server environment that can withstand a wide array of cyber threats, ensuring the confidentiality, integrity, and availability of the organization's critical systems and data.

# Requirement Specification

Requirement Specifications: Securing and hardening a Linux Server

### Hardware and Software Requirements

#### Server Hardware:

Adequate CPU, RAM, and disk space to support the server and security tools.

Redundant power supplies and network interfaces for high availability.

#### Operating System:

Latest stable version of a Linux distribution (e.g., Ubuntu, CentOS, Debian).

#### Security Software:

Firewall software (e.g., iptables, firewalld).

Intrusion detection systems (e.g., Snort, OSSEC).

File integrity monitoring tools (e.g., AIDE, Tripwire).

Anti-virus and anti-malware software (e.g., ClamAV).

VPN software (e.g., OpenVPN) for secure remote access.

### Configuration and Management Requirements

#### User Authentication:

* Implement multi-factor authentication (MFA).
* Configure SSH with key-based authentication.
* Disable root login and enforce the use of sudo for administrative tasks.

#### Network Security:

* Define and enforce firewall rules to limit inbound and outbound traffic.
* Configure a VPN for remote access.
* Implement network segmentation and DMZ for separating critical services.

#### System Updates:

* Schedule regular updates and patch management for the OS and installed software.
* Automate security updates wherever possible.

#### Access Control:

* Define and enforce strict access control policies.
* Use role-based access control (RBAC) to limit user permissions.

#### Monitoring and Logging:

* Set up centralized logging (e.g., using syslog or a log management tool).
* Implement continuous monitoring tools (e.g., Nagios, Prometheus).
* Configure alerting for suspicious activities or potential breaches.

#### Data Protection Requirements

Encryption:

* Encrypt sensitive data at rest and in transit using strong encryption algorithms.
* Use tools like LUKS for disk encryption and SSL/TLS for data in transit.

#### Backup and Recovery:

* Establish regular backup schedules.
* Store backups securely and test recovery procedures periodically.

# Functional and Non-Functional requirements

## Functional Requirements

User Authentication and Authorization:

Implement multi-factor authentication (MFA).

Configure SSH with key-based authentication.

Enforce role-based access control (RBAC).

Firewall Configuration:

Establish robust firewall rules to limit inbound and outbound traffic.

Implement network segmentation and DMZ for separating critical services.

Intrusion Detection and Prevention:

Deploy intrusion detection systems (IDS) and intrusion prevention systems (IPS) to monitor and respond to suspicious activities.

Set up alerting mechanisms for potential breaches.

System Updates and Patch Management:

Schedule regular updates and patches for the operating system and installed software.

Automate the application of security updates.

Data Encryption:

Encrypt sensitive data at rest and in transit using strong encryption algorithms.

Use SSL/TLS for secure data transmission.

Backup and Recovery:

Implement a regular backup schedule.

Store backups securely and test recovery procedures periodically.

Logging and Monitoring:

Configure centralized logging and monitoring tools.

Set up real-time monitoring for system health and security events.

## Non-Functional Requirements

Performance:

Ensure that security measures do not significantly impact server performance.

Optimize security tools and configurations to maintain high system efficiency.

Reliability:

Maintain high system availability and minimize downtime.

Implement redundancy for critical components to ensure continuous operation.

Scalability:

Design the security infrastructure to scale with growing demands.

Ensure that security measures can handle increased load and traffic.

Usability:

Provide clear documentation and guidelines for managing and maintaining security.

Ensure that security configurations are easy to implement and manage.

Compliance:

Adhere to relevant regulatory standards and industry best practices (e.g., GDPR, PCI-DSS).

Ensure that security measures meet compliance requirements.

Maintainability:

Implement modular security configurations to facilitate updates and maintenance.

Ensure that security tools and configurations can be easily modified and extended.

# Implementation

### Creating A Linux Server:

For the purpose of This Project A Linux Server is created on a HP Laptop (8GB RAM/ 250 GB HDD) Installing RedHat Linux 9 on it by

1. Downloading RHEL 9 ISO image.
2. Downloading rufus utility
3. Using Rufus Utility, Creating bootable installation media on USB Flash drive.
4. Changed boot order in BIOS setting of Laptop to boot from USB drive
5. Inserted USB drive in laptop and powered in on.
6. Installed the Linux server on Laptop by following installation instructions.

### Installing System Updates and Patches

Package management Tools: DNF and YUM are package management tools used in RHEL distributions like Fedora, CentOS, and Rocky Linux. They simplify package management, installation, updates, and removal by handling all the package dependencies. YUM is the older tool, and DNF is its modern successor, offering improved performance and features. Both package managers are essential for maintaining a stable and updated Linux system.

The Apt (Advanced Package Tool) package management system is a set of tools to download, install, remove, upgrade, configure and manage Debian packages, and therefore all software installed on a Debian system. Ubuntu, Kali Linux, Linux Mint and MX Linux are some of the Debian based Distributions

To run the package Updates on Red Hat Enterprise Linux (RHEL) 9 open a terminal and login as root user or use sudo to gain administrative privileges as run following code:

Check for available updates:

sudo dnf check-update

This command will list all packages that have updates available.

To Update all packages run :

sudo dnf upgrade

Update security-related packages:

sudo dnf upgrade --security

This command will upgrade packages that have security errata.

Update minimal security-related packages:

sudo dnf upgrade-minimal --security

This command will upgrade only the minimal set of packages needed to address security issues.

After running these commands, your system should be up-to-date with the latest packages and security updates.

### Configure automatic Update

To set up automatic updates on RHEL 9, the dnf-automatic tool is used.

Install dnf-automatic:

sudo dnf install dnf-automatic -y

Configure dnf-automatic: Open the configuration file:

sudo vi /etc/dnf/automatic.conf

Edit the file to set desired options. Here's an example configuration:

ini

[commands]

upgrade\_type=security

apply\_updates=yes

emit\_via=stdio

Enable and start the dnf-automatic.timer:

sudo systemctl enable --now dnf-automatic.timer

This will configure the system to automatically check for and apply security updates. The configuration file can be adjusted to suit specific needs, such as setting the upgrade\_type to all to download all updates, or changing the emit\_via option to send notifications via email.

By default, dnf-automatic is set to check for updates daily. However, its interval can customize by editing the timer configuration.

Open the timer configuration file:

sudo nano /etc/systemd/system/dnf-automatic-install.timer

Modify the [Timer] section to set your desired interval. For example, to check for updates every week, use:

ini

[Timer]

OnCalendar=weekly

Persistent=true

Save the file and exit the editor.

Reload the systemd manager configuration:

sudo systemctl daemon-reload

Restart the timer:

sudo systemctl restart dnf-automatic-install.timer

This will configure dnf-automatic to check for updates once a week. You can adjust the OnCalendar value to set different intervals, such as daily, monthly, or even specific times of the day.

### Set Password Policy and Complexity

To set password complexity and quality in RHEL 9, the pam\_pwquality module is used. Here's how to configure it to enforce following rules:

Number of attempts allowed before locking the account - 3.

Minimum password length 10 character.

Minimum number of character classes 3 (lowercase, uppercase, digits, special characters).

Maximum number of repeated characters 3.

Install authselect

sudo dnf install authselect -y

Switch to the profile that enforces password quality

sudo authselect select sssd

Edit PAM configuration file

sudo vi /etc/pam.d/password-auth

Add following line to enforce password quality

password required pam\_pwquality.so retry=3 minlen=12 minclass=3 maxrepeat=3

Save and exit editor

Restart PAM service

sudo systemctl restart sssd

This configuration will enforce password complexity rules for user accounts on RHEL 9 system. the parameters can be adjusted to meet specific security requirements.

### Firewall Configuration

Inatall firewalld

sudo dnf install firewalld -y

Enable and start firewalld service

sudo systemctl enable firewalld

sudo systemctl start firewalld

Check status of firewalld service

sudo systemctl status firewalld

List available zones

sudo firewall-cmd --get-zones

Set default zone (public)

sudo firewall-cmd --set-default-zone=public

Warning: ZONE\_ALREADY\_SET: public

success

To allow a specific service (eg HTTP) run

sudo firewall-cmd --zone=public --add-service=http --permanent

sudo firewall-cmd --reload

To allow a specific port(eg 8080) run

sudo firewall-cmd --zone=public --add-port=8080/tcp --permanent

sudo firewall-cmd --reload

### Disable ssh access for root user

Disabling SSH access for the root user is a crucial security measure to help protect your server from unauthorized access. Here's how it can be done on a RHEL 9 system

Open SSH configuration file

sudo vi /etc/ssh/sshd\_config

Find the line that says PermitRootLogin (it might be commented out with a #):

#PermitRootLogin yes

Change this line to:

PermitRootLogin no

If the line is commented out, make sure to uncomment it by removing the # at the beginning.

Save the file and exit the editor.

Restart the SSH service to apply the changes:

sudo systemctl restart sshd

After making these changes, the root user will no longer be able to log in via SSH. You should ensure that you have another user with sudo privileges to maintain administrative access to the server.

Configure secure boot settings.

Restrict SSH access by disabling root login and enforcing key-based authentication.

Configure Firewall:

Set up a firewall using tools like iptables or firewalld to control inbound and outbound traffic.

Define rules to only allow necessary services.

Apply System Hardening Scripts:

Utilize hardening scripts (e.g., Lynis, CIS Benchmark scripts) to automatically apply security configurations.

Step 3: Implement Security Controls

User and Group Management:

Create and enforce strong password policies.

Implement role-based access control (RBAC) to limit user permissions.

Intrusion Detection and Prevention:

Install and configure intrusion detection systems (IDS) such as Snort or OSSEC.

Implement intrusion prevention systems (IPS) to automatically respond to threats.

Configure Logging and Monitoring:

Set up centralized logging using tools like syslog-ng or rsyslog.

Implement monitoring tools like Nagios or Prometheus for continuous monitoring and alerting.

Data Encryption:

Encrypt sensitive data at rest using tools like LUKS or eCryptfs.

Ensure data in transit is encrypted using SSL/TLS.

Step 4: Backup and Recovery

Regular Backups:

Schedule regular backups of critical data and system configurations.

Store backups securely and test recovery procedures periodically.

Disaster Recovery Plan:

Develop and document a disaster recovery plan.

Conduct regular drills to ensure readiness in case of an incident.

Step 5: Continuous Monitoring and Maintenance

Regular Updates and Patching:

Establish a schedule for regular updates and patches.

Automate the update process wherever possible.

Continuous Monitoring:

Monitor server activity continuously to detect and respond to anomalies.

Utilize tools like OSSEC or Fail2ban to prevent brute force attacks and other threats.

Regular Audits and Reviews:

Conduct periodic security audits and reviews to ensure compliance with the security policy.

Update the security policy and configurations based on audit findings and emerging threats.

# Resources Used

1. A Laptop to Install Red Hat Linux
2. Redhat Enterprise Linux Software ISO image
3. 64 GB USB 2.0 Flash Drive to Create bootable Installation media
4. Rufus utility to create bootable USB flash drive from RHEL 9 ISO image

# References

Red Hat Documentation:

<https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9>