Project Report

on

Setting Up a Personal Web Server

Submitted by

Riya

UID: 24MCA20436

Under the guidance of

Mr. Navdeep Singh Sodhi

in partial fulfillment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

Chandigarh University

November 2024

Acknowledgment

I would like to express my heartfelt gratitude to **Mr. Navdeep Singh sir**, whose invaluable guidance, support, and encouragement were instrumental in making this project possible. Their insightful feedback and suggestions helped refine my approach and understanding of the complexities involved in setting up a web server.

I also extend my sincere appreciation to **Chandigarh University** for providing the necessary resources, tools, and a conducive learning environment. The access to laboratory facilities and library resources significantly enhanced my ability to conduct research and implement the project effectively.

Furthermore, I would like to acknowledge **my peers** who shared their knowledge and experiences with me throughout this journey. Their collaboration and willingness to help fostered an environment of learning and camaraderie, which made this project even more enjoyable.

Lastly, I owe a special thanks to **my family** for their unwavering support and encouragement during the project. Their understanding of the time and effort required allowed me to focus on my work without distraction. They inspired me to push through challenges and celebrate successes along the way.

This project is not just a reflection of my individual effort but also of the collective support I received from everyone mentioned above. I am truly grateful for all the assistance and encouragement that contributed to the successful completn of this project.

Riya

24MCA20436

Table of Contents

Abstract

This project involves setting up and configuring the Apache HTTP server on Red Hat Enterprise Linux (RHEL) 9.4, aiming to create a reliable personal web server capable of efficiently serving web pages and handling incoming requests. The project initiated with a thorough exploration of RHEL, highlighting its suitability for server environments and the advantages of using an open-source platform.

The installation of the Apache HTTP server was performed using standard package management commands, followed by detailed configurations to optimize performance and security. Key aspects included configuring firewall settings to control network traffic and managing SELinux policies to enhance the server's security posture. By ensuring that only necessary services were exposed, the project aimed to minimize vulnerabilities while maximizing accessibility.

Hands-on experience was pivotal in this project, as we encountered and resolved common issues that arose during the setup process. This troubleshooting experience not only reinforced theoretical knowledge but also provided practical insights into the intricacies of server management.

The successful establishment of the server facilitated the hosting of static content, allowing for a demonstration of web accessibility through standard browsers. Users could access the server using its IP address, validating the installation and configuration process.

This project underscores the importance of meticulous configuration and robust security practices in web server management. It provides a solid foundation for future endeavors in web hosting and system administration, illustrating how the integration of best practices can lead to a secure and efficient web service environment. Ultimately, this project serves as a stepping stone for further exploration into advanced topics, such as dynamic content hosting, database integration, and scalable web applications.

1. Introduction to Red Hat Enterprise Linux

Red Hat Enterprise Linux (RHEL) is one of the most widely-used enterprise-grade operating systems based on Linux. Developed and maintained by Red Hat, Inc., RHEL is designed for commercial use, offering a stable, secure, and supported platform for running enterprise applications, databases, and servers.

Key Features of Red Hat Enterprise Linux:
-Enterprise-Grade Security: RHEL provides advanced security features such as SELinux (Security-Enhanced Linux), which enforces access controls and security policies to protect the system.

Long-Term Support: RHEL offers long-term support, ensuring that enterprises can run their systems with stability and receive updates for an extended period, which is essential for mission-critical environments.

Scalability and Performance: RHEL is designed to support large-scale applications and systems, from small setups to cloud environments and data centers.

Subscription Mode: Red Hat provides subscription-based support, giving users access to certified software, updates, patches, and professional support.

Red Hat has become a go-to solution for hosting web servers, databases, and various other critical applications in corporate environments due to its reliability, security, and professional support.

1.1. Server Hosting Overview

Server hosting is the process of making services or resources (such as websites, databases, or applications) available via a computer server to users over the internet or a local network. A server hosting service typically provides the hardware, software, storage, and connectivity required for the operation of web services.

There are several types of server hosting:

Shared Hosting: Multiple websites are hosted on a single physical server, sharing its resources (CPU, RAM, disk space).

Virtual Private Server (VPS): A physical server is divided into multiple virtual servers, each with its own operating system and resources.

Dedicated Server: A single physical server is allocated exclusively to one user or organization, offering full control and resources.

Cloud Hosting: Servers are distributed across the cloud, allowing scalable resources and redundancy, ensuring high availability.

1.2. Why Set Up a Personal Web Server?

Setting up a personal web server provides full control over hosting resources, data, and configurations. It's ideal for developers, small businesses, or individuals who want to:

Host websites and applications without relying on third-party hosting services.
Learn server management and gain hands-on experience.
Run secure private services like personal file storage or test environments.

In this project, we use Red Hat Enterprise Linux (RHEL) as the operating system for the personal web server, with Apache as the web server software.

1.3. Apache HTTP Server Introduction

Apache HTTP Server, commonly referred to as Apache, is open-source web server software developed and maintained by the Apache Software Foundation. It is one of the most popular web servers in the world due to its simplicity, robustness, and versatility.

Key Features of Apache HTTP Server:

Modular Design: Apache uses a modular architecture, meaning additional functionality can be added or removed as needed via modules. These modules support a variety of features such as URL rewriting, authentication, and SSL encryption.

Cross-Platform Compatibility: Apache works on a variety of operating systems, including Linux (like RHEL), macOS, and Windows.

Virtual Hosting: Apache can host multiple websites on a single server using "virtual hosts," which allows efficient resource use and centralized management.

Security: Apache supports a wide range of security features, including SSL/TLS encryption, password protection, access controls, and integration with SELinux for advanced security policies.

Open Source: Apache is free and open-source software, backed by a large community that continuously updates and improves it.

2. Project Objectives

- To install and configure Apache on RHEL 9.4.
- To enable secure and efficient web hosting.
- To understand and implement firewall configurations.
- To troubleshoot common server hosting issues.
-

3. System Requirements

3.1. Hardware Requirements

- A server or personal computer with at least 2GB of RAM.
- Storage: 10GB or more available disk space.
- Network interface for internet access.

3.2. Software Requirements

- Red Hat Enterprise Linux 9.4
- Apache HTTP Server 2.4.x

- SELinux (enabled by default in RHEL)
- Firewalld (firewall management tool)

4. Installation Commands:

4.1: Update Your System

Before installing Apache, update your RHEL system to ensure that all packages are current.

sudo dnf update -y

```
liveuser@localhost-live:~$ sudo dnf update -y
Fedora 40 - x86_64
               8.1 kB/s |  11 kB     00:01
Fedora 40 - x86_64 - Updates
               9.4 kB/s | 9.9 kB     00:01
Fedora 40 - x86_64 - Updates
               1.0 MB/s | 4.9 MB     00:04
Last metadata expiration check: 0:00:01 ago on Mon 28 Oct 2024 05:21:15 AM EDT.
Dependencies resolved.
================================================================================================================================
 Package                                                Architecture                    Version
                                      Size
================================================================================================================================
Installing:
 kernel                                                 x86_64                          6.11.4-201.fc4
                                    183 k
 kernel-modules                                         x86_64                          6.11.4-201.fc4
                                     64 M
 kernel-modules-extra                                   x86_64                          6.11.4-201.fc4
                                    2.9 M
Upgrading:
 Box2D                                                  x86_64                          2.4.2-1.fc40
                                    109 k
 ImageMagick                                            x86_64                          1:7.1.1.38-1.f
                                     73 k
 ImageMagick-libs                                       x86_64                          1:7.1.1.38-1.f
                                    2.6 M
 LibRaw                                                 x86_64                          0.21.3-1.fc40
                                    411 k
```

Explanation:

`dnf` is the package manager used in Red Hat-based distributions to install, update, and manage packages.

`sudo` grants superuser privileges, necessary for system updates.

`-y` automatically confirms the update.

4.2: Install Apache HTTP Server

Once the system is updated, you can install Apache using the following command:

sudo dnf install httpd -y

```
liveuser@localhost-live:~$ sudo dnf install httpd -y
Last metadata expiration check: 0:08:43 ago on Mon 28 Oct 2024 05:21:15 AM EDT.
Package httpd-2.4.58-7.fc40.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
liveuser@localhost-live:~$
```

Explanation:

`httpd` is the package name for Apache on RHEL and Fedora systems.

This command installs Apache and all its dependencies.

4.3: Start and Enable Apache

After installation, start the Apache service and configure it to start automatically on boot:

sudo systemctl start httpd

sudo systemctl enable httpd

```
liveuser@localhost-live:~$ sudo systemctl start httpd
Warning: The unit file, source configuration file or drop-ins of httpd.service changed on disk. Run 'systemctl daemon-reload' to reload units.
liveuser@localhost-live:~$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
liveuser@localhost-live:~$
```

Explanation:

`systemctl start httpd` starts the Apache service.

`systemctl enable httpd` ensures that Apache starts automatically when the system boots.

4.4: Configure the Firewall to Allow HTTP/HTTPS Traffic

To allow external traffic to access the web server, configure the firewall to allow HTTP (port 80) and HTTPS (port 443) connections:

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

sudo firewall-cmd --permanent --add-service=https

sudo firewall-cmd –reload

```
liveuser@localhost-live:~$ sudo firewall-cmd --permanent --add-service=http
success
liveuser@localhost-live:~$ sudo firewall-cmd --permanent --add-service=https
success
liveuser@localhost-live:~$ sudo firewall-cmd --reload
success
liveuser@localhost-live:~$
```

Explanation:

`firewall-cmd` is the command-line tool for managing firewall rules in RHEL.

`--permanent` makes the changes persist across reboots.

`--reload` reloads the firewall with the new settings, without restarting the system.

5. Testing and Validation:

5.1: Network Configuration:

Once Apache is up and running, verify that the web server is working by navigating to your server's IP address in a web browser.

To check the server's IP address, run:

hostname -I

```
liveuser@localhost-live:~$ hostname -I
10.0.2.15
```

Then open a browser and type `http://<your-server-IP>`. You should see the Apache test page confirming the server is running.

5.2: Create a Simple HTML Web Page

To serve content, you need to create an HTML file in the Apache document root (`/var/www/html`).

echo "<h1>hello world!</h1>" | sudo tee /var/www/html/index.html

```
liveuser@localhost-live:~$ echo "<h1> Hello Riya</h1>" | sudo tee /var/www/html/index.html
<h1> Hello Riya</h1>
liveuser@localhost-live:~$
```

Explanation:

`echo` prints the HTML content.

The pipe (`|`) sends the output to the `tee` command, which writes it to the specified file.

`/var/www/html/index.html` is the default location where Apache looks for web content.

5.3: Set Permissions

Make sure Apache has the appropriate permissions to access the website files:

sudo chown -R apache:apache /var/www/html

```
liveuser@localhost-live:~$ sudo chown -R apache:apache /var/www/html
liveuser@localhost-live:~$ sudo systemctl restart httpd
liveuser@localhost-live:~$
```

Explanation:

`chown` changes the ownership of files and directories.

This command sets the `apache` user and group as the owner of `/var/www/html`.

5.4: Configure SELinux (if applicable)

SELinux, a security feature in RHEL, might restrict Apache from accessing user content. To allow it, use the following command:

sudo setsebool -P httpd_read_user_content 1

sudo systemctl restart httpd

```
liveuser@localhost-live:~$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
liveuser@localhost-live:~$ sudo setsebool -P httpd_read_user_content 1
liveuser@localhost-live:~$
```

Explanation:

`setsebool` sets the SELinux boolean to allow Apache to read user content.

`-P` ensures that this change persists across reboots.

` restart httpd` command restarts Apache, allowing changes to take effect.

5.5: Ensure Apache is Running:

Check if Apache is running with the following command.

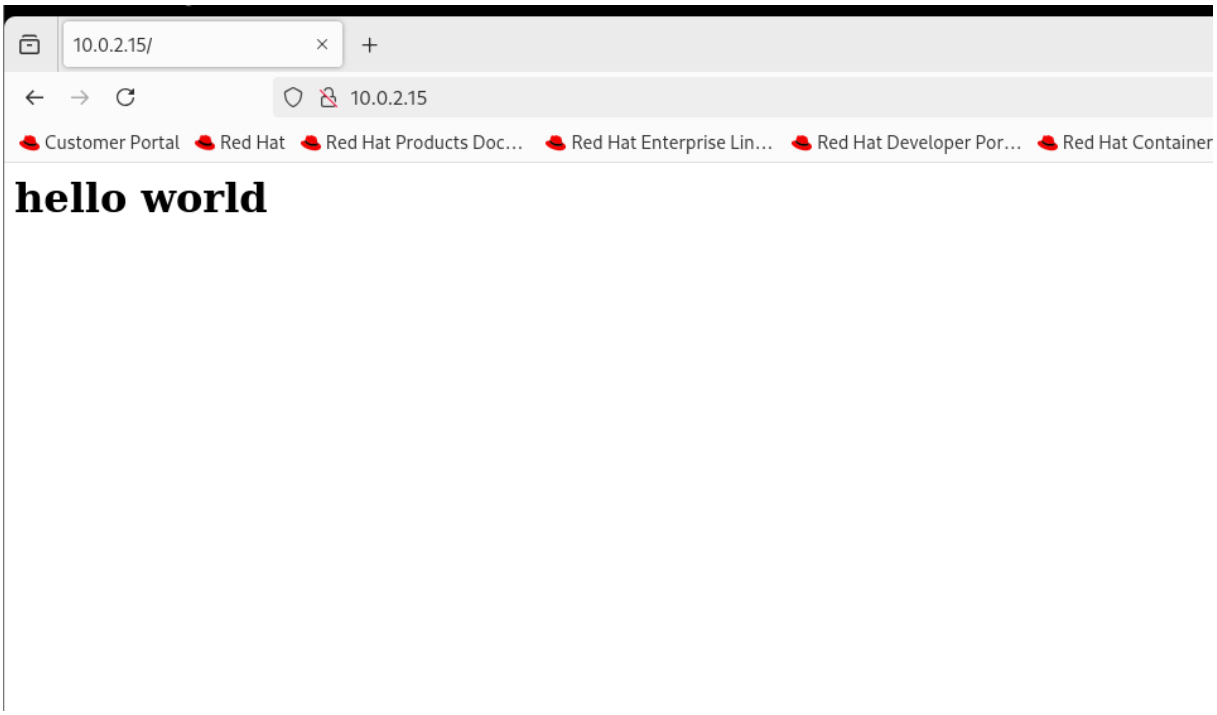
`sudo systemctl status httpd`

```
liveuser@localhost-live:~$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/service.d
             └─10-timeout-abort.conf
     Active: active (running) since Mon 2024-10-28 05:45:54 EDT; 5min ago
       Docs: man:httpd.service(8)
   Main PID: 4694 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
      Tasks: 177 (limit: 4528)
     Memory: 14.5M (peak: 14.8M)
        CPU: 950ms
     CGroup: /system.slice/httpd.service
             ├─4694 /usr/sbin/httpd -DFOREGROUND
             ├─4695 /usr/sbin/httpd -DFOREGROUND
             ├─4697 /usr/sbin/httpd -DFOREGROUND
             ├─4698 /usr/sbin/httpd -DFOREGROUND
             └─4746 /usr/sbin/httpd -DFOREGROUND

Oct 28 05:45:52 localhost-live systemd[1]: Starting httpd.service - The Apache HTTP Server...
Oct 28 05:45:52 localhost-live (httpd)[4694]: httpd.service: Referenced but unset environment variable
Oct 28 05:45:53 localhost-live httpd[4694]: AH00558: httpd: Could not reliably determine the server's f
Oct 28 05:45:54 localhost-live httpd[4694]: Server configured, listening on: port 80
Oct 28 05:45:54 localhost-live systemd[1]: Started httpd.service - The Apache HTTP Server.
lines 1-23/23 (END)
```

5.6: Access Your Website

Open your web browser and go to your server's IP address. You should see the "hello world!" message displayed.



Explanation: This command restarts Apache, allowing changes to take effect.

Conclusion

In this project, we successfully set up a personal web server using Red Hat Enterprise Linux (RHEL) 9.4 and the Apache HTTP Server. The primary goal was to gain a comprehensive understanding of server hosting and web server management, which we achieved through the step-by-step configuration and deployment process.

We began by installing and configuring Apache, one of the most popular open-source web servers in the world, known for its reliability, flexibility, and security. The installation process included updating the system, starting the Apache service, and ensuring it would start automatically after every system reboot. By properly configuring the firewall, we enabled HTTP and HTTPS traffic, allowing external users to access the server.

During the course of the project, we explored key aspects of server administration, such as managing firewall rules, setting up SELinux permissions, and ensuring the proper ownership and permissions of the document root to serve web content. These elements are crucial for maintaining a secure, stable, and functional web server environment.

We also learned to verify the functionality of the server by accessing the default Apache test page via a web browser and troubleshooting common issues like incorrect firewall settings or service configurations. This hands-on experience provided valuable insight into the challenges and complexities involved in real-world server hosting and management.

One of the highlights of the project was understanding the interaction between the different components of the web server environment—operating system (RHEL), web server software (Apache), and networking tools (firewall, SELinux). We gained practical knowledge of how these elements work together to host a website and ensure it is accessible to users.

In conclusion, this project has provided a solid foundation in web server management, network configuration, and the deployment of web content. The skills gained from setting up and managing a personal web server can be extended to more complex server environments and large-scale web applications, making this a valuable learning experience in the field of system administration and web hosting.

References

1. Red Hat Documentation

Red Hat, Inc. (2024). Red Hat Enterprise Linux 9.4 Documentation. Retrieved from https://access.redhat.com/documentation

2. Apache HTTP Server Documentation

The Apache Software Foundation. (2024). Apache HTTP Server Version 2.4 Documentation. Retrieved from https://httpd.apache.org/docs/

3. Setting Up and Managing Firewalld on Red Hat Enterprise Linux: Red Hat, Inc. (2024). Firewalld: Configuring and Managing Firewall Rules in RHEL. Retrieved from https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/

4. Systemd Service Management in RHEL Red Hat, Inc. (2024). Managing Services with systemd on RHEL. Retrieved from https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/

5. Linux Security Modules: SELinux and Apache

Tresys Technology. (2024). SELinux Policy Management and Apache Integration. Retrieved from https://selinuxproject.org/page/Main_Page

6. Practical Guide to Web Hosting with Apache

Griffith, J. (2021). Mastering Apache: Advanced Techniques and Best Practices for Web Hosting. Apress Publishing.

These references provide the basis for understanding the installation and configuration of Apache HTTP Server on Red Hat Enterprise Linux, as well as server management and security practices.