Computer Communication Network

Lab4A- Web Server Configuration

Student Name: Sem: Date:

Objective: Deploy and configure Apache web server on Ubuntu Linux 22.04 with both name-based and IP-based virtual hosting, ensuring successful installation, optimal performance, and effective troubleshooting capabilities.

Outcomes: After successful completion of lab students will be able to
1. Deploy Apache web server on Ubuntu Linux 22.04.
2. Configure name-based virtual hosting for multiple websites.
3. Setup IP-based virtual hosting for improved resource management.
4. Optimize through SSL/TLS implementation and fine-tuning settings.
5. Acquire effective troubleshooting,ensuring a robust and error-free operation of the Apache web server.

System Requirements:
The lab requires a system with a dual-core processor or higher, a minimum of 4 GB RAM, and at least 20 GB of free disk space. The machine should be running Ubuntu Linux 22.04 LTS, and participants should have a user account with sudo privileges for administrative tasks. A stable internet connection is essential for downloading and installing packages during the setup process. Properly configured DNS settings are recommended if working with domain names for name-based virtual hosting, and additional available IP addresses are needed if implementing IP-based virtual hosting. Access tools such as a terminal or SSH, a text editor for configuration file edits, and a web browser for testing are necessary for a comprehensive learning experience.

Procedure:
Here are the practical steps for installing, configuring, and operating the Apache web server on Ubuntu Linux 22.04 with both name-based and IP-based virtual hosting:

Step 1: Install Apache Web Server
sudo apt update
sudo apt install apache2

Verify Apache installation:
sudo systemctl status apache2

Step 2: Configure Name-Based Virtual Hosting

1. Create directories for your websites:
sudo mkdir /var/www/site1
sudo mkdir /var/www/site2

2. Set permissions:
sudo chown -R www-data:www-data /var/www/site1
sudo chown -R www-data:www-data /var/www/site2

3. Create virtual host configurations:
sudo nano /etc/apache2/sites-available/site1.conf
apache
<VirtualHost *:80>
    ServerAdmin webmaster@site1.com
    ServerName site1.com
    DocumentRoot /var/www/site1

    ErrorLog ${APACHE_LOG_DIR}/site1_error.log
    CustomLog ${APACHE_LOG_DIR}/site1_access.log combined
</VirtualHost>

Repeat the process for site2.

4. Enable virtual hosts:
sudo a2ensite site1.conf
sudo a2ensite site2.conf
sudo systemctl restart apache2

Step 3: Configure IP-Based Virtual Hosting

1. Assign additional IP addresses to your server:
sudo nano /etc/netplan/01-network-manager-all.yaml
yaml
network:
  version: 2
  renderer: networkd
  ethernets:
    ens33:
      dhcp4: true
      dhcp6: true
      addresses:
        - 192.168.1.2/24
        - 192.168.1.3/24
Apply the changes:
sudo netplan apply
2. Create IP-based virtual host configurations:
sudo nano /etc/apache2/sites-available/site1_ip.conf

apache
<VirtualHost 192.168.1.2:80>
ServerAdmin webmaster@site1.com
ServerName site1.com
DocumentRoot /var/www/site1

ErrorLog ${APACHE_LOG_DIR}/site1_error.log
CustomLog ${APACHE_LOG_DIR}/site1_access.log combined
</VirtualHost>

Repeat the process for site2.

3. Enable IP-based virtual hosts:
sudo a2ensite site1_ip.conf
sudo a2ensite site2_ip.conf
sudo systemctl restart apache2

Step 4: Test and Optimize Apache Server

1. Test name-based virtual hosts:

- Open your browser and visit `http://site1.com` and `http://site2.com`. Ensure they display the correct content.

2. Test IP-based virtual hosts:
- Open your browser and visit `http://192.168.1.2` and `http://192.168.1.3`. Ensure they display the correct content.

3. Optimize Apache server:
- Implement SSL/TLS for secure connections.
- Fine-tune Apache settings in the `apache2.conf` file.
- Set up a firewall to allow only necessary traffic.

Step 5: Troubleshooting and Common Issues
1. Check Apache logs:
sudo tail -f /var/log/apache2/error.log
sudo tail -f /var/log/apache2/access.log

2. Use Apache tools:

- Utilize `apachectl configtest` to check for syntax errors in configurations.
- Use `apache2ctl -S` to view the virtual host configurations.

By following these practical steps, you should have a fully functional Apache web server on Ubuntu Linux 22.04 with both name-based and IP-based virtual hosting. Troubleshoot any issues using logs and Apache tools for a seamless web hosting experience.

Conclusion: (Write in your own words)

References:

[1] Ubuntu 22.04 LTS : Apache2 : Install : Server World

[2] How To Install the Apache Web Server on Ubuntu 22.04 | DigitalOcean