Akash Chakresh

Comprehensive Guide to Setting Up and Securing a Website on LAMP Stack

Step 1 : Setting up the LAMP STACK

Installing Apache :
sudo apt install apache2 -y

Installing MySQL :
sudo apt install mysql-server -y
Sudo mysql_secure_installation

Installing PHP :
Sudo apt install libapache2-mod-php php-mysql -y

Verifying installation :
Apache: Visit `http://your_server_ip` in a browser. You should see the Apache default page.

PHP: Create a PHP test file:

bash
```
sudo echo "<?php phpinfo(); ?>" > /var/www/html/info.php
```

Visit http://your_server_ip/info.php.

2. Create Your Website's Directory

Create a directory for your site:

```
sudo mkdir /var/www/yourwebsite
```

Set correct permissions:

```
sudo chown -R $USER:$USER /var/www/yourwebsite
sudo chmod -R 755 /var/www
```

Copy your website files (HTML, PHP, etc.) to `/var/www/yourwebsite`.

Step 3 : Apache2 Configuration according to your website

Find Your Server's IP Address: Run this command to get your public IP address:

```
curl ifconfig.me
```

Look for inet under the active network interface (like eth0 or wlan0).

Access Your Website via IP: Open your browser and enter the IP address in the address bar, for example:

```
http://your_server_ip
```

Point Apache to Your Website Directory

Modify the **default configuration** (if you're not hosting multiple sites yet):

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Update the `DocumentRoot` directive to:

```
DocumentRoot /var/www/yourwebsite
```

Save and exit the file, then restart Apache:

```
sudo systemctl restart apache2
```

Step 4 : Secure your Website

1.1 Set Up a Firewall

Use ufw (Uncomplicated Firewall) to allow necessary traffic and block everything else:

```
sudo ufw allow OpenSSH        # If using SSH
sudo ufw allow 80/tcp         # Allow HTTP traffic
sudo ufw allow 443/tcp        # Allow HTTPS traffic
sudo ufw enable               # Enable the firewall
```

1.2 Install and Configure Fail2Ban

Fail2Ban protects against brute-force attacks.

Install Fail2Ban:

```
sudo apt install fail2ban -y
```

Create a local configuration file:

```
sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Edit `/etc/fail2ban/jail.local` to enable protection for

Apache:

```
[apache-auth]
enabled = true
```

Restart Fail2Ban:

```
sudo systemctl restart fail2ban
```

1.3 Install SSL Certificates (HTTPS) :

Encrypts the connection between your website and the user's browser.

Enable SSL Module: Run the following command to ensure the SSL module is enabled in Apache:

```
sudo a2enmod ssl
```

Create an SSL Virtual Host: If you haven't already, create an SSL virtual host in Apache. Edit the SSL configuration file (e.g.,

`Sudo nano /etc/apache2/sites-available/default-ssl.conf`

or a custom configuration file).

Example configuration:

```
<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/selfsigned.key
    ServerName localhost
</VirtualHost>
```

Enable the SSL Site:
```
sudo a2ensite default-ssl.conf
```

Restart Apache:
```
sudo systemctl restart apache2
```

Use Let's Encrypt with Certbot to enable HTTPS:

Install Certbot:
```
sudo apt install certbot python3-certbot-apache -y
```

Run Certbot to secure your site:
```
sudo certbot --apache
```

Certbot will:

- Ask for your domain name
- Set up HTTPS
- Redirect HTTP to HTTPS if you choose
- Automatically renew the certificate

Note: If your laptop is behind a local network, you might need a domain name for this step. Alternatively, use a self-signed certificate for testing:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048-keyout
/etc/ssl/private/selfsigned.key -out
/etc/ssl/certs/selfsigned.crt
```

1.4 Additional Apache Hardening

Disable Directory Listing

Prevent users from seeing the contents of directories:

Edit **/etc/apache2/sites-available/000-default.conf :**

Where :

```
<Directory /var/www/yourwebsite>
   Options -Indexes        # Disable directory listing
   AllowOverride All       # Allow .htaccess to override
   Require all granted
</Directory>
```

Hide Apache Version Info :

Edit **/etc/apache2/sites-available/security.conf :**

Add :

```
ServerTokens Prod      # Hide detailed server version in headers
ServerSignature Off     # Remove Apache version from error pages

# Limit request body size (protect against DoS attacks)
LimitRequestBody 102400
```

1.5 PHP Hardening settings :

Disable Dangerous funstions :

```
disable_functions = exec,passthru,shell_exec,system
```

Turn off error display:

```
display_errors = Off
```

```
log_errors = On
```

Disable remote file inclusion:

```
allow_url_fopen = Off
```

```
allow_url_include = Off
```

Restart Apache :

```
sudo systemctl restart apache2
```

1.5 MySQL Hardening to create Database user to manage the web user:

Login to MySQL:

```
sudo mysql -u root -p
```

Create a New Database (if you haven't already):

```
CREATE DATABASE your_db_name;
```

Create a New Database User:

```
CREATE USER 'webuser'@'localhost' IDENTIFIED BY
'StrongPassword!';
```

Grant Limited Privileges to the New User:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON your_db_name.* TO
'webuser'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
EXIT;
```

Database User (webuser): This user is used by your application to interact with the database. It's created by the admin (you) and is granted limited privileges to read, insert, update, or delete data in the database.

Make the Website Accessible to Everyone

2.1 Check Your Public IP Address

Find your public IP address using:

```
curl ifconfig.me
```

2.2 Port Forwarding

To allow external devices to access your website:

1. Log into your router's admin panel (usually at `192.168.1.1` or similar).
2. Find the **Port Forwarding** section.
3. Forward ports **80 (HTTP)** and **443 (HTTPS)** to your laptop's local IP address.
    - Local IP: Find it using `ip addr show` (e.g., `192.168.1.100`).
4. Save the settings.After powering on your laptop:

Link Your Domain Name to Your Local Server

To make your purchased domain point to your local server, you need to update the DNS records of your domain.

1. **Get Your Public IP**: Find your public IP address (you can search "What is my IP" in a browser).

2. **Update DNS A Record**:
    - Log in to your domain registrar's website (where you bought your domain).
    - Go to the DNS settings section.
    - Find the **A Record** and change the value to your **public IP address**.
    - Save the changes. It may take some time for DNS propagation (anywhere from a few minutes to 24 hours).

Example:
Type: A

Host: @ (or leave it blank for the root domain)

Value: Your Public IP (e.g., 203.0.113.1)

TTL: 3600 (default)

3. **Test Your Domain**: After DNS propagation is complete, try accessing your website using your domain name (e.g., http://yourdomain.com).

Steps to Re-host Your Website

Start Apache (Web Server)

After rebooting, Apache might not start automatically. Start it manually:

```
sudo systemctl start apache2
```

Verify Your Website Files

Ensure your website files are still in the correct directory (`/var/www/yourwebsite`):

```
ls /var/www/yourwebsite
```

Re-enable Firewall Rules

If your firewall (e.g., UFW) was disabled after the reboot, re-enable it and ensure the correct rules are applied:

```
sudo ufw allow 80/tcp    # HTTP traffic

sudo ufw allow 443/tcp   # HTTPS traffic

sudo ufw enable
```

1. Start Apache and MySQL.
2. Check that your files are still in `/var/www/yourwebsite`.
3. Ensure firewall and port forwarding are configured correctly.
4. Access your site using your laptop's public IP or domain.