Apache2 Webserver Ubuntu24

1. Installation and Setup

Install Apache2:

Auto

```
sudo apt update
sudo apt install apache2
```

Start Apache2:

Auto

```
sudo systemctl start apache2
```

Stop Apache2:

Auto

```
sudo systemctl stop apache2
```

Restart Apache2:

Auto

```
sudo systemctl restart apache2
```

Reload Apache2 (apply configuration changes without restarting the service):

Auto

```
sudo systemctl reload apache2
```

Enable Apache2 to start on boot:

Auto

```
sudo systemctl enable apache2
```

Check Apache2 status:

Auto

```
sudo systemctl status apache2
```

2. Apache2 Configuration Files

- **Main Apache Configuration File**:
  - `/etc/apache2/apache2.conf` : Global Apache settings (for most distributions including Ubuntu).
- **Virtual Hosts**:
  - Default configuration for websites: `/etc/apache2/sites-available/000-default.conf`
  - Custom sites configuration files are located under `/etc/apache2/sites-available/` .
  - To enable a site: `sudo a2ensite your-site.conf`
  - To disable a site: `sudo a2dissite your-site.conf`
- **Enable/Disable Modules**:
  - Enable a module (e.g., rewrite module):
  - Disable a module:
- **Global Configuration Directory**:
  - `/etc/apache2/conf-available/` for system-wide configuration snippets and `/etc/apache2/conf-enabled/` for active ones.

3. Managing Virtual Hosts

Default Virtual Host Configuration (`000-default.conf`):

Auto

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

Example configuration:

Auto (Bash)

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ServerName example.com
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # Optional: Allow access from all IP addresses
    <Directory /var/www/html>
        AllowOverride All  # Allow .htaccess files
        Require all granted
    </Directory>
</VirtualHost>
```

Enabling Virtual Hosts:

To enable a new site (virtual host), create a configuration file in `/etc/apache2/sites-available/your-site.conf` and enable it:

Auto

```
sudo a2ensite your-site.conf
sudo systemctl reload apache2
```

Disabling a Site:

Auto

```
sudo a2dissite your-site.conf
sudo systemctl reload apache2
```

4. Directory Permissions

4. Directory Permissions

Ensure proper directory permissions for Apache to access files:

Auto (Bash)

```
sudo chown -R www-data:www-data /var/www/html
sudo chmod -R 755 /var/www/html
```

5. Apache2 Modules

To enable or disable Apache2 modules:

- **Enable `mod_rewrite`** (for URL rewriting):
- **Enable `mod_ssl`** (for SSL support):
- **Enable `mod_headers`** (for custom headers):
- **Enable `mod_proxy`** (for reverse proxy):
- **Disable a module**:

6. SSL Configuration

Install SSL Module:

Auto

```
sudo apt install mod_ssl
```

Enable SSL module:

Auto

```
sudo a2enmod ssl
sudo systemctl restart apache2
```

Configure SSL in Virtual Host:

In your site configuration (e.g., `your-site.conf`), ensure you have SSL settings like:

Auto (Bash)

```
<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    ServerName example.com
    DocumentRoot /var/www/html

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/yourdomain.crt
    SSLCertificateKeyFile /etc/ssl/private/yourdomain.key

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Get a Free SSL Certificate (using Certbot and Let’s Encrypt):

Auto (CSS)

```
sudo apt install certbot python3-certbot-apache
sudo certbot --apache -d example.com
```

Renew SSL Certificates:

Auto

```
sudo certbot renew
```

7. Performance Tuning

Enable Gzip Compression:

Edit `/etc/apache2/mods-available/deflate.conf` or create a custom config file.

Auto

```
sudo nano /etc/apache2/mods-available/deflate.conf
```

Add or ensure the following is present:

Auto (Bash)

```
<IfModule mod_deflate.c>
    AddOutputFilterByType DEFLATE text/html text/plain text/xml application/xml application/xhtml+xml text/css application/javascript application/json
</IfModule>
```

Caching Static Content:

Add caching rules in your virtual host configuration or `.htaccess` file for static assets like images, CSS, and JS:

Auto (PowerShell)

```
<FilesMatch "\.(jpg|jpeg|png|gif|css|js|woff|woff2|ttf)$">
    ExpiresActive On
    ExpiresDefault "access plus 1 year"
</FilesMatch>
```

8. Security Best Practices

Disable Directory Listing:

Ensure `Options -Indexes` is set in your virtual host or `.htaccess` file to prevent directory listing.

Auto

```
<Directory /var/www/html>
    Options -Indexes
</Directory>
```

Restrict Access to Specific Directories:

To protect sensitive directories (e.g., `/admin`):

Auto (CSS)

```
<Directory /var/www/html/admin>
    Order Deny,Allow
    Deny from all
    Allow from 192.168.1.0/24  # Allow access from a specific IP range
</Directory>
```

Disable Server Signature:

To prevent Apache from exposing version details:

Auto

```
sudo nano /etc/apache2/conf-available/security.conf
```

Add or modify:

Auto

```
ServerSignature Off
ServerTokens Prod
```

9. Logging

Apache logs are stored by default in:

- **Access Log**: `/var/log/apache2/access.log`
- **Error Log**: `/var/log/apache2/error.log`

To customize log formats:

Auto (PowerShell)

```
LogFormat "%h %l %u %t \"%r\" %>s %b" combined
CustomLog ${APACHE_LOG_DIR}/access.log combined
```

10. Reverse Proxy Setup

To proxy traffic to a backend service (e.g., a Node.js app):

- Enable proxy modules:
- Add a reverse proxy configuration:

Auto (Bash)

```
<VirtualHost *:80>
```

```
    ServerName yourdomain.com
    DocumentRoot /var/www/html

    # Reverse proxy to backend app
    ProxyPass /api/ http://localhost:3000/
    ProxyPassReverse /api/ http://localhost:3000/

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

11. Firewalls and Port Forwarding

Make sure Apache is allowed through the firewall:

Auto

```
sudo ufw allow 'Apache Full'
```

12. Apache Status and Information

To get detailed information about Apache's performance and status:

Auto

```
sudo apache2ctl status
```

13. Restart Apache2 for Changes to Take Effect

After any configuration change:

Auto

```
sudo systemctl restart apache2
```

Additional Apache2 Optimization

- **Max Request Workers**: Adjust the `MaxRequestWorkers` directive in `apache2.conf` for handling high traffic.

- **KeepAlive**: Enable `KeepAlive` for persistent connections:
- **Worker MPM**: If you expect a high volume of traffic, consider switching to the `worker` or `event` MPM (multi-processing module), which is more efficient than the default `prefork` MPM: