LAB-1 : Cloud Infrastructure Management Lab

Introduction to Apache2

Apache2, commonly known as the Apache HTTP Server, is an open-source and highly configurable web server software developed and maintained by the Apache Software Foundation. It is designed to serve web content such as HTML pages, images, and other files to users' browsers over the HTTP or HTTPS protocols.

Key Features:
Cross-Platform Support: Available for various operating systems including Linux, Windows, macOS, and Unix-like systems.
Modular Architecture: Supports a wide range of modules (e.g., mod_ssl for SSL/TLS support, mod_rewrite for URL rewriting, mod_proxy for reverse proxy functionality) to extend its functionality.
Virtual Hosting: Allows hosting multiple websites on the same server using name-based or IP-based virtual hosts.
Support for Dynamic Content: Integrates with programming languages and technologies like PHP, Python, Perl, and Ruby through modules.
HTTPS/SSL Support: Ensures secure communication via SSL/TLS encryption.
Security: Built-in mechanisms for access control, authentication, and support for third-party security tools.

Static vs. Dynamic Web Servers
Apache2 can function as both a static and a dynamic web server, depending on how it is configured and what kind of content it is serving.
Static Web Servers: ideal for sites that do not need user interaction or frequent updates, such as documentation or portfolios sites.
Dynamic Web Servers: suited for interactive, personalized, or database-driven applications where content changes based on user actions. When configured with modules like mod_php, mod_python, or proxy settings for application servers (e.g., Node.js or Django), Apache2 serves dynamic content by executing server-side code and generating content on the fly.

Virtual Hosting:

The default Ubuntu document root is /var/www/html. You can make your own virtual hosts under /var/www.
Virtual Hosting allows Apache2 to host multiple websites on a single server. This can be done in two primary ways:
Name-Based Virtual Hosting: Multiple websites are served using the same IP address but differentiated by domain names.
IP-Based Virtual Hosting: Each website is served using a unique IP address.

Steps to install on Lab desktop

1. Update the local package index to reflect the latest upstream changes

sudo apt update

2. Install the apache2 package
   sudo apt install apache2
   After confirming the installation, apt will install Apache and all required dependencies
3. Commands to check status, start, stop , restart the webserver
   sudo systemctl start apache2
   sudo systemctl stop apache2
   sudo systemctl status apache2
   sudo systemctl restart apache2
   sudo systemctl reload apache2 → If you are making configuration changes, Apache can reload without dropping connections
   sudo systemctl disable apache2 → By default, Apache is configured to start automatically when the server boots. You can disable this behavior
   sudo systemctl enable apache2 → To re-enable the service to start up at boot
4. Default welcome page used to test the correct operation of the Apache2 server after installation
   http://127.0.0.1/index.html
5. Update File permissions: -rw-r--r–
   cd /var/www/html
   sudo chmod 777 index.html
   You can update the file contents and reload the page (note: use *sudo nano index.html* command)
   Changes should be reflected *http://127.0.0.1/index.html*
6. Steps to Set Up Name-Based Virtual Hosting
   A. Set Up Domain Names
      Point your domain names to the server's IP address by configuring the DNS records or using your /etc/hosts file (note: use *sudo nano hosts* command)
      127.0.0.1 example1.com
      127.0.0.1 example2.com
   B. Create Directories for Websites
      Create separate directories for each website's content.
      sudo mkdir -p /var/www/example1.com/public_html
      sudo mkdir -p /var/www/example2.com/public_html
   C. Set permissions for these directories:
      sudo chown -R $USER:$USER /var/www/example1.com/public_html
      sudo chown -R $USER:$USER /var/www/example2.com/public_html
      sudo chmod -R 755 /var/www
   D. Add Content to Each Website
      Add a basic HTML file to each directory:
      echo "<h1>Welcome to Example1.com</h1>" | sudo tee /var/www/example1.com/public_html/index.html
      echo "<h1>Welcome to Example2.com</h1>" | sudo tee /var/www/example2.com/public_html/index.html
   E. Create Virtual Host Configuration Files
      Create a separate configuration file for each website.
      sudo nano /etc/apache2/sites-available/example1.com.conf
      Add following contents to the file
      *<VirtualHost *:80>*

```
    ServerAdmin admin@example1.com
    ServerName example1.com
    ServerAlias www.example1.com
    DocumentRoot /var/www/example1.com/public_html
    ErrorLog ${APACHE_LOG_DIR}/example1.com_error.log
    CustomLog ${APACHE_LOG_DIR}/example1.com_access.log combined
</VirtualHost>
```

Repeat for the second website:

sudo nano /etc/apache2/sites-available/example2.com.conf

Add following contents to the file

```
<VirtualHost *:80>
    ServerAdmin admin@example2.com
    ServerName example2.com
    ServerAlias www.example2.com
    DocumentRoot /var/www/example2.com/public_html
    ErrorLog ${APACHE_LOG_DIR}/example2.com_error.log
    CustomLog ${APACHE_LOG_DIR}/example2.com_access.log combined
</VirtualHost>
```

F. Enable the Virtual Host Files
   Enable the new virtual hosts using the a2ensite command:
   sudo a2ensite example1.com.conf
   sudo a2ensite example2.com.conf
G. Disable the default site if not needed: *sudo a2dissite 000-default.conf*
H. Test the Configuration: *sudo apache2ctl configtest*
I. Reload Apache2
   sudo systemctl reload apache2
J. Access the Websites
   Visit the domains in your browser:
   http://example1.com
   http://example2.com

(Exercise): Steps to install on AWS EC2 Instance

Prerequisite: Sign or Signup

Steps to signup to AWS:
https://aws.amazon.com/
Root user email address: mspcimlab@gmail.com
AWS account name: AWS for CIM lab
Root user password :
How do you plan to use AWS?: Personal - for your own projects
Enter all your Personal and Contact Details
Enter your Billing Information, **Credit or Debit card number**
Click submit
Note: AWS will send OTP and deduct Rs. 2/-. However, It will credit it back in few minutes/hours
Primary purpose of account registration : choose Academic

Ownership type : Individual
India document type : Choose one of the options (PAN Card , Driving License etc)
Date of birth: YYYY/MM/DD
Upload the Document
Before you can use your AWS account, **you must verify your phone number.**
Enter your Mobile phone number
Click Submit
Select a support plan : Basic support - Free
System shows confirmation page : Congratulations!
Personalize Your Experience
- My role is: Student
- I am interested in: Websites & WebApps

Click Signin into AWS Console

EC2(Elastic Compute Cloud):
EC2 is a managed service provided by Amazon that allows users to rent virtual servers, known as "instances," to run their applications in the cloud. The term "Elastic" refers to the ability to scale the compute capacity up or down based on demand.

Create EC2 instance:
Note: Choose region
1) Click Launch instance:
   Name and tags :
   Application and OS Images (Amazon Machine Image) : Choose ubuntu
   Instance type : t2.micro
   Key pair (login) : Create new Key Pair
   - Key pair name:
   - Key pair type: RSA
   - Private key file format: .pem

   Click Launch Instance
2) Wait for few seconds
   Click Instances on left menu options
   Click on instanceID
   Note down following
   - Public IPv4 address (3.145.86.243)
   - Public IPv4 DNS: ec2-3-145-86-243.us-east-2.compute.amazonaws.com
   - Instance state: Running
3) Click Connect
   Connection Type: Connect using EC2 Instance Connect
   Username: ubuntu
4) Following steps 1~5 completed in Lab setup (except, access the page using public IP Address)

5) Does it work ? If not, perform following steps from AWS Console
   Click InstanceId

Choose Security Tab
Click Security groups, launch-wizard-1
Click Security group ID
Edit inbound rules

- Type: HTTP
- Source: Anywhere-IPv4

Now access the page using public IP Address) (http://3.145.86.243/index.html)

Note: Skip step #6: Set Up Name-Based Virtual Hosting , as we are in public domain and we will do these changes when we setup DNS in AWS

Important Note : Before your end your Lab, remember to do following

- Choose instance , Instance State , Stop Instance
- Logout from AWS

(Additional Exercise): Setup IP-Based Virtual Hosting on Lab desktop