Lab Document

Apache Web Server (CentOS7) Admin Training

1. Assumed that you have already setup the Virtual Machine Image and connected to it through Putty and WinSCP

2. Replace HOST\_IP with actual IP Address. Please update the IP address and port wherever required based on the lab configuration.

2. **The Aim:** We are finally going to deal with 2 web sites named ‘training.com’ and ‘labs.com’. For DNS configuration we need to modify our /etc/hosts file. Your VM already has 2 Virtual NIC (host-only) enabled and we have 2 IP addresses available. We map the IP address with hostnames as follows:

**a) Get the IP addresses:**

**$ifconfig** #will list all the ip addresses

Let us consider we have 192.168.56.106 and 192.168.56.107

**b) Append the following lines in the existing /etc/hosts file as follows:**

192.168.56.106 training.com

192.168.56.107 labs.com

|  |
| --- |
| **LAB-1: Install Apache Web Server Binary on Linux (CentOS 7)** |
| 1. Run the following command to install httpd (Apache Web Server)  # sudo yum install -y httpd  2. Check the version of installed Apache Web Server  # httpd –v  2. start Apache Web Server as Linux service  # sudo systemctl start httpd  3. Check the status of Apache Web Server  #sudo systemctl status httpd  4. Open a browser and use the url <http://localhost>. You should get the Apache Test Page  5. Stop Apache Service  #sudo systemctl stop httpd  6. Uninstall Apache Web Server  #sudo yum erase httpd  Follow the onscreen instructions  7. Delete the configuration Directory of httpd  #sudo rm –rf /etc/httpd  Now httpd is removed from your System!! |

|  |
| --- |
| **Lab-2: Install Apache From Source (As root user)** |
| Prerequisites:  1. c++ for compiling (installed)  2. PCRE v8.x  3. APR and APR-UTILS  4. Apache Source |
| 1. Download Apache Source from <http://httpd.apache.org/download.cgi>  #cd  # wget <http://mirrors.estointernet.in/apache//httpd/httpd-2.4.41.tar.gz>  2. Download PCRE  #wget <https://ftp.pcre.org/pub/pcre/pcre-8.42.tar.gz>  3. Download Apache Portable Runtime and APR UTils  #wget <http://mirrors.estointernet.in/apache//apr/apr-1.7.0.tar.gz>  #wget <http://mirrors.estointernet.in/apache//apr/apr-util-1.6.1.tar.gz>  4. All the downloaded sources are in your home directory /home/training  5. extract Apache httpd source  # tar zxvf httpd-2.4.41.tar.gz  This command will extract the source in httpd-2.4.41 directory  6. Extract APR and APR-Util inside httpd source folder  #tar zxvf apr-1.7.0.tar.gz –C httpd-2.4.41/srclib/  #tar zxvf apr-util-1.6.1.tar.gz –C httpd-2.4.41/srclib/  Rename apr and apr-util folders  #mv httpd-2.4.41/srclib/apr-1.7.0 httpd-2.4.41/srclib/apr  #mv httpd-2.4.41/srclib/apr-util-1.6.1/ httpd-2.4.41/srclib/apr-util  7. Extract PCRE and install it  #tar zxvf pcre-8.42.tar.gz  #cd pcre-8.42  #./configure  #make  #make install  **8. Compile Apache httpd**  # cd httpd-2.4.41  #./configure --prefix=/usr/local/apache2  #make  #make install  #/usr/local/apache2/bin/apachectl start  **9. Verify the server running status at http://localhost** |

|  |
| --- |
| **Lab-3: Run Apache With min Config (Use yum installed httpd binary installation)** |
| 1. Rename the actual httpd.conf file and preserve it.   $cd /etc/httpd/conf  $sudo mv httpd.conf httpd.conf.orig   1. Create a new file named httpd.conf and add Server configuration   $sudo nano httpd.conf   1. Add the following configuration entries and save the file   #min httpd config httpd 2.4.x  User root  Group root  ServerName www.training.com  ErrorLog /var/log/httpd/error.log  LoadModule authz\_host\_module modules/mod\_authz\_host.so  LoadModule dir\_module modules/mod\_dir.so  DocumentRoot /var/www/html  <Directory /var/www/html>  AllowOverride None  Require all granted  </Directory>   1. Start Apache Server and test   $sudo systemctl start httpd |

|  |
| --- |
| **LAB-4: Using Directives (Controlling access to Folders and Files)** |
| **Note: Create Directories under /var/www/html**   1. **public** 2. **private** 3. **local** 4. **dev** |
| **1. Use of <Directory> /<DirectoryMatch>**  **2. Use of <Files>/<FilesMatch>**  **3. Use of <Location>/<LocationMatch>**  **4. Use handlers : server-status, server-info,balancer-manager** |

|  |
| --- |
| **LAB - 5: Virtual Hosting (IP Based)** |
| 1. The default DocumentRoot is /var/www/html. **Do not change it** 2. You will create a directory structure within **/var/www** for **the training.com** site, leaving **/var/www/html** in place as the default directory to be served if a client request doesn’t match any other sites. 3. Create the **html** directory for **training.com** as follows, using the **-p** flag to create any necessary parent directories:   $sudo mkdir -p /var/www/training.com/html   1. Create an additional directory to store log files for the site:   $sudo mkdir -p /var/www/training.com/log   1. Next, assign ownership of the html directory with the $USER environmental variable:   $sudo chown -R $USER:$USER /var/www/training.com/html   1. Make sure that your web root has the default permissions set:   $sudo chmod -R 755 /var/www   1. Next, create a sample index.html page using vi or your favorite editor:   $sudo nano /var/www/training.com/html/index.html  add the following sample HTML to the file:/var/www/training.com/html/index.html  **<html>**  **<head>**  **<title>Welcome to Training.com!</title>**  **</head>**  **<body>**  **<h1>Success! The training.com virtual host is working!</h1>**  **</body>**  **</html>**  Save and close the file.  Create the following directories:  $sudo mkdir /etc/httpd/sites-available /etc/httpd/sites-enabled  Edit /etc/httpd/conf/httpd.conf  $sudo nano /etc/httpd/conf/httpd.conf  Add this line to the end of the file:  **IncludeOptional sites-enabled/\*.conf**  Save and close the file.  Create your virtual host configuration in sites-available directory:  $sudo nano /etc/httpd/sites-available/training.com.conf  Add in the following configuration block, and change the training.com domain to your domain name:  /etc/httpd/sites-available/training.com.conf  <VirtualHost \*:80>  ServerName www.training.com  ServerAlias training.com  DocumentRoot /var/www/training.com/html  ErrorLog /var/www/training.com/log/error.log  CustomLog /var/www/training.com/log/requests.log combined  </VirtualHost>  This will tell Apache where to find the root directly that holds the publicly accessible web documents. It also tells Apache where to store error and request logs for this particular site.  Save and close the file when you are finished.  Now that you have created the virtual host files, you will enable them so that Apache knows to serve them to visitors. To do this, create a symbolic link for each virtual host in the sites-enabled directory:  $sudo ln -s /etc/httpd/sites-available/training.com.conf /etc/httpd/sites-enabled/training.com.conf  Your virtual host is now configured and ready to serve content.  **Repeat above steps for second virtual host ‘labs.com’** |

|  |
| --- |
| **LAB: Virtual Hosting (Name Based)** |
| **Prerequisites:**  **1. Map IP address to hostname in /etc/hosts file**  2. Create required folders for serving files  $sudo mkdir -p /var/www/example1.com/html /var/www/example2.com/html  $sudo nano /var/www/example1.com/html/index.html [add some content to index.html]  $sudo nano /var/www/example2.com/html/index.html [add some content to index.html] |
| 1. Locate ‘Listen’  Replace  Listen 80  With  Listen 192.168.56.106:80 (Your VM’s IP ADDRESS)  2. Add the following blocks in your /etc/httpd/conf/httpd.conf file or use sites-enabled approach  NameVirtualHost 192.168.56.106:80  <VirtualHost 192.168.56.106:80>  ServerAdmin webmaster@example.com  ServerName www.example.com  DocumentRoot /var/www/example.com/html  <Directory /var/www/example.com/html >  Options +FollowSymLinks +Indexes  AllowOverride None  Require all granted  </Directory>  </VirtualHost>  <VirtualHost 192.168.56.106:80>  ServerAdmin webmaster@example2.com  ServerName www.example2.com  DocumentRoot /var/www/example2.com/html  <Directory /var/www/example2.com/html >  Options +FollowSymLinks +Indexes  AllowOverride None  Require all granted  </Directory>  </VirtualHost> |

|  |
| --- |
| **LAB: Use Mod\_PROXY** |
| **Prerequisites: Prepare the backend servers**   1. We are going to use node.js and express server 2. Download node.js 3. CD to your home directory   $cd   1. Download and install node.js and setup the backend express server   $su  #wget <https://nodejs.org/dist/v10.16.3/node-v10.16.3-linux-x64.tar.gz>  # tar xvf node-v10.16.3-linux-x64.tar.gz -C /usr/local  # ln -s /usr/local/node-v10.16.3/bin/node /usr/local/bin/node  #ln -s /usr/local/node-v10.16.3/bin/npm /usr/local/bin/npm  #exit   1. Use WinSCP to copy the given backend folder to /home/training 2. Execute the following commands   $cd backend  $ npm init --y  $npm install --save express cors body-parser jsonfile  $ node server.js   1. Open a browser and type in the url (use correct IP) <http://ip-address-of-host:3000/test>   The Server responds with “Hello From Server..” |
| **Setup mod\_proxy and use**  **1**. Load the required modules by adding the below lines in /etc/httpd/conf.modules.d/00-proxy.conf if not already added or enabled  LoadModule proxy\_module modules/mod\_proxy.so  LoadModule lbmethod\_byrequests\_module modules/mod\_lbmethod\_byrequests.so  LoadModule proxy\_balancer\_module modules/mod\_proxy\_balancer.so  LoadModule proxy\_http\_module modules/mod\_proxy\_http.so  **2. Reverse Proxying a Single Backend Server**  3. Modify your virtualHost configuration as given  <VirtualHost \*:80>  ProxyPreserveHost On  ProxyPass / http://HOST\_IP:3000/  ProxyPassReverse / http://HOST\_IP:300/  </VirtualHost>  3. Restart Apache httpd  $sudo systemctl restart httpd  4. Test the proxy by using the url <http://localhost/list> . You should see a list of employees json data from backend server |

|  |
| --- |
| **LAB: Load Balancing backend Web Servers (Reverse Proxy)** |
| 1. After you have done proxying one server, you need to come to this lab.  2. Modify the virtual host section of your conf file like the following one  <VirtualHost \*:80>  <Proxy balancer://mycluster>  BalancerMember http://HOST\_IP:3000  BalancerMember http://HOST\_IP:3000  </Proxy>  ProxyPreserveHost On  ProxyPass / balancer://mycluster/  ProxyPassReverse / balancer://mycluster/  </VirtualHost> |

|  |
| --- |
| **LAB: Custom Logging** |
|  |

|  |
| --- |
| **LAB: Error Page** |
|  |

|  |
| --- |
| **LAB: Alias** |
|  |

|  |
| --- |
| **LAB: Install Self Signed SSL Certificate** |
| **SSL-On-Apache2.4**  Step 1 – Install mod\_ssl and openssl Package  $sudo yum install mod\_ssl openssl # Redhat / CentOS systems  Step 2 – Create Self Signed Certificate  $sudo openssl req -x509 -nodes -newkey rsa:2048 -keyout training.com.key -out training.com.crt  Step 3 – Install Self Signed Certificate in Apache  Listen 443  <VirtualHost \_default\_:443>  ServerAdmin admin@training.com  ServerName www.training.com  ServerAlias training.com  DocumentRoot /var/www/training.com/html  SSLEngine on  SSLCertificateFile /etc/pki/tls/certs/training.com.crt  SSLCertificateKeyFile /etc/pki/tls/certs/training.com.key  </VirtualHost>  Step 4 – Restart Apache  $sudo systemctl restart httpd  Access the Web Site  https://www.training.com  **Approach2 (Alternative Approach, Optional here):**  Step 1: Generate Private Key  Openssl genrsa –out ca.key 1024  Step 2: Generate Certificate Request File  Openssl req –new –key ca.key –out ca.csr  Step 3: Generate the Self Signed Certificate  Openssl x509 –req –days 365 –in ca.csr –signkey ca.key –out ca.crt  Configure Apache Web Server to Run on SSL (HTTPS)  **See Approach 1 Virtual Host Section** |

|  |
| --- |
| **LAB: Authentication and Authorization** |
| **Authentication**  Step1: Create a file named .htpasswd in /etc/httpd (you need to use -c for the first time you create the .htpasswd file)  $sudo htpasswd -c /etc/httpd/.htpasswd admin  $sudo htpasswd /etc/httpd/.htpasswd user1  View the content of .htpasswd file  cat /etc/httpd/.htpasswd  Modify your virtual host configuration file as  <VirtualHost \*:80>  ServerAdmin webmaster@localhost  DocumentRoot /var/www/html  ErrorLog ${APACHE\_LOG\_DIR}/error.log  CustomLog ${APACHE\_LOG\_DIR}/access.log combined    #for authe config  <Directory "/var/www/html">  AuthType Basic  AuthName "Restricted Content"  **AuthUserFile /etc/httpd/.htpasswd**  Require valid-user  </Directory>  #auth config ends  </VirtualHost>  Restart httpd and test  **Set up Authorization:**  A file named .htaccess is required in the folder to be restricted. This file contains auth config details.  Step1: We will restrict /var/www/training.com folder (Document Root)  Create a file named .htaccess with the following entries and place it in /var/www/training.com/  sudo vi /var/www/training.com/.htaccess  AuthType Basic  AuthName "Restricted Content"  AuthUserFile /etc/httpd/.htpasswd  Require valid-user  Next open /etc/httpd/conf/httpd.conf file and locate  <Directory /var/www/>  Options Indexes FollowSymLinks  **AllowOverride All**  Require all granted  </Directory>  change 'AllowOverride None' to 'AllowOverride All' |