|  |  |
| --- | --- |
| **Lab 9a Basic Web Server** | |
| **Step 1** Install Apache.  "httpd" is the name for the Apache service in CentOS.  Apache is controlled by applying directives in configuration files:  **/etc/httpd/conf/httpd.conf** – Main Apache config file  /etc/httpd/ – Location for all config files  /etc/httpd/conf.d/ – All config files in this directory are included in the main config file  /etc/httpd/conf.modules.d/ – Location for Apache module config files | Verify httpd is installed  # rpm -q httpd  httpd-2.4.37-21.module\_el8.2.0+382+15b0afa8.x86\_64  #vi /etc/httpd/conf/httd.conf  Looking for the “ServerName” section, we can change it to our own servername  Looking for the “DocumentRoot” section, this is where the webpage located. Default is /var/www/html” |
| **Step 2** Start and configure Apache to run on startup.  **Note:** When making changes to configuration files, remember to always restart the Apache service to apply the new configuration. | # systemctl start httpd  # systemctl enable httpd  # systemctl status httpd |
| **Step 3** Check the default Apache Test Page.  The new Apache installation has a default test page, but we can also create a custom test page in **/var/www/html/index.html**  You can designate a directory to store the files for customized website. Use the configuration files to point to the directory you choose. Some typical locations include:  /home/username/my\_website  /var/www/my\_website  /var/www/html/my\_website  /opt/my\_website  Apache log files to monitor web server:  /var/log/httpd/ – Location of Apache log files  /var/log/httpd/access\_log – Shows a log of systems that accessed the server  /var/log/httpd/error\_log – Shows a list of any errors Apache encounters |  |
| **Step 4** Adjust Firewall for Apache | # firewall-cmd --permanent --zone=public --add-service=http  # firewall-cmd --reload |
| Step 5 Test Apache web page from another machine |  |
| **Lab 9b Configure Apache with SSL/TLS Certificates** | |
| The mod\_ssl module provides SSL v3 and TLS v1.x support for the Apache HTTP Server.  **Step 1** install mod\_ssl module.  Note: Start httpd Apache webserver will start mod\_ssl as well | #dnf install mod\_ssl (mod\_ssl is already installed in our VM)  #rpm -q mod\_ssl  mod\_ssl-2.4.37-21.module\_el8.2.0+382+15b0afa8.x86\_64 |
| **Step 2** Open TCP port 443 to allow external access to Apache over HTTPS | # firewall-cmd --zone=public --permanent --add-service=https  success  # firewall-cmd --reload  success |
| **Step 3** Generate self-signed SSL certificate.  There are multiple options to choose from when you want to secure Apache with SSL/TLS certificates.   * Use self-signed certificates for test purposes. * Order for a commercially trusted server certificate from preferred CA * Use the free, automated, and open CA.   We will use Openssl command to generate SSL/TLS certificate   * **req**: It is used to create CSR as well as the self signed certificates. CSR-certificate signing request. * **-newkey rsa:4096**: This option creates a new certificate request and a 4096 bits RSA key at the same time. * **-nodes**: When this option is specified then if a private key is created it will not be encrypted. * **-keyout** **/etc/pki/tls/private/httpd.key**: Writes the newly created private key to the specified filename. Replace the filename accordingly. * **-x509**: This option outputs a self signed certificate instead of a certificate request. * **-days 365**: Used to specify the validity period for the self signed certificate generated. This therefore is valid for 365 days. * **-out /etc/pki/tls/certs/httpd.crt**: Specifies the output filename to write the self signed certificate to.   After successful execution of the above command the private key has been written to /etc/pki/tls/private/httpd.key while the certificate has been written to /etc/pki/tls/certs/httpd.crt. | #openssl req -newkey rsa:4096 -nodes -keyout /etc/pki/tls/private/httpd.key -x509 -days 365 -out /etc/pki/tls/certs/httpd.crt  Generating a RSA private key  ..............................+++++  ...............+++++  writing new private key to '/etc/pki/tls/private/httpd.key'  -----  You are about to be asked to enter information that will be incorporated  into your certificate request.  What you are about to enter is what is called a Distinguished Name or a DN.  There are quite a few fields but you can leave some blank  For some fields there will be a default value,  If you enter '.', the field will be left blank.  -----  Country Name (2 letter code) [XX]:AU  State or Province Name (full name) []:NSW  Locality Name (eg, city) [Default City]:Sydney  Organization Name (eg, company) [Default Company Ltd]:UTS  Organizational Unit Name (eg, section) []:IT  Common Name (eg, your name or your server's hostname) []:www.it.netserv.edu.au  Email Address []:  -----BEGIN CERTIFICATE-----  MDkyNTAwMTQyMloXDTIxMDkyNTAwMTQyMlowVjELMAkGA1UEBhMCQVUxDDAKBgNV  AklUMQ0wCwYDVQQDDARZaW5nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKC  Q4DoXWGEUCz6QoVSH9OifZZf4aGgr5KA/O+0p2iQYzPDq3gUqeAPsshE1WXFrBOx  gOtd9li/BSq09Ybg4SDM++uAkLEDqr5Tigmb0jf+8gfvjokdjrJIz5kd3IDhxhlr  cQ==  -----END CERTIFICATE----- |
| **Step 4** Configure Apache to use SSL/TLS Certificate  Check the Apache configuration for syntax errors:  # apachectl configtest | #vi /etc/httpd/conf.d/ssl.conf  Change the certificate file and certificate key file to the one we just created.  FROM:  SSLCertificateFile /etc/pki/tls/certs/localhost.crt  SSLCertificateKeyFile /etc/pki/tls/private/localhost.key  TO:  SSLCertificateFile /etc/pki/tls/certs/httpd.crt  SSLCertificateKeyFile /etc/pki/tls/private/httpd.key  To check the Apache configuration for syntax errors:  # apachectl configtest  Or  # httpd -t  Syntax OK |
| **Step 5** Reload Apache and test Apache  You may see an error. This is normal for a self-signed certificate! The browser is warning you that it can’t verify the identity of the server, because our certificate is not signed by any of the browser’s known certificate authorities. For testing purposes and personal use this can be fine. | # systemctl reload httpd  Open web browser, use https:// to test  For our lab, we are going to use different webpage for testing, we create a new index.html file in /var/www/secure directory, so modify the DocumentRoot “/var/www/secure” to point to the new webpage, then test it with https. |
| **Lab 9c Virtual Hosting with Apache** | |
| By default, Apache web server is configured to serve or host only one website. If you plan to host multiple domains on your server, then you need to configure Apache virtual hosts.  A virtual host is a separate file that contains configurations that allow you to set up a separate domain from the default one. The default virtual host is located at the /var/www/html directory. This works only for a single site. To create a separate virtual host for our domain, we will create another directory structure within the /var/www directory then edit the hhtpd.conf file to include information about the virtual host.  We also need a second domain name which can be resolved to an IP address. We can create a second domain name in the DNS server (you will need a separate entry in the named.conf file, and you will need to create a separate DNS zone file). An easier alternative, which is sufficient to prove your implementation of virtual hosts, is to define the new web server fully qualified domain names in the /etc/hosts file of the VM that you will use to send HTTP requests to the VM running Apache. | # mkdir /var/www/a  #echo “This is aaaa webpage” > /var/www/a/index.html  # vi /etc/httpd/conf/httpd.conf  <VirtualHost \*:80>  DocumentRoot "/var/www/a"  ServerName www.it.netserv.edu.au  # Other directives here  </VirtualHost>  <VirtualHost \*:80>  DocumentRoot "/var/www/a"  ServerName www2.it.netserv.edu.au  # Other directives here  </VirtualHost>  #vi /etc/hosts  10.0.2.3 www2.it.netserv.edu.au  Restart httpd  #systemctl restart httpd |