NGINX Core - Module M11 Labs

**** The Labs Begin on Page 2 ****

PLEASE READ THESE SYSTEM USE INSTRUCTIONS:

The lab systems default to text mode, if you want to take advantage of copy-and-paste functionality, or need to use multiple **Terminal** applications, you'll need to start the GUI Desktop environment using the following instructions.

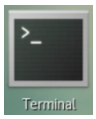
Steps to Prepare for a GUI Lab Environment:

1. Sign on to the system with the user credential **root** and password **training**.

2. Start the GUI desktop with the command:

   ```
   $ startx
   ```

3. When the Graphical Desktop appears, click on the **Terminal** app icon on the left-side of the desktop to open the Terminal.



   Note: You can resize the **Terminal** app to suit your preferences or to show more text output on the screen.

To Copy and Paste to a Lab System in GUI Mode:

1. Select the desired text from this lab guide and use **Cmd+c** (macOS) or **Ctrl-c** (Windows and Linux) to copy the text to the system clipboard.

2. Then click into the **Terminal** app in the lab system, and press Ctrl-Shift-V or select the Terminal application's **Edit** -> **Paste** menu item.

3. You may need to press it twice to insert the copied text.

   Note: When pasting into **Vim**, you MUST be in **Insert** mode first, or the keystrokes will get mis-interpreted. If you get a strange `^M` or other character first, just delete it and re-paste again.

NOTE: The **commands** you should be typing, (or copying & pasting for convenience) are displayed like the below text; (don't type or copy the **$** prompt, just the `green` text):

```
$ command
```

The **output** from commands, or the file contents shown in commands like **vim** are displayed like this:

```
Line of output
Another line of output
```

Labs for Module 11 - Restricting Access

Labs Included

- Lab 11.1 - Create the **`ssl_test.conf`** File
- Lab 11.2 - Generate a self-signed certificate and key
- Lab 11.3 - Set protocols and ciphers
- Lab 11.4 - Configure the **`add_header`** directive
- Lab 11.5 - Configure the Diffie-Hellman key exchange
- Lab 11.6 - Force HTTP traffic to HTTPS

PLEASE READ!

You have been allocated **1 hour** for the labs in this section. Please leave the Virtual Lab Machine browser window/tab open after completing the first lab, then continue the section content. You will then be asked to complete the rest of the labs throughout the rest of the section.

If you let too much time lapse and the Virtual Lab Machine times out, simply restart it by clicking on the Launch button for the Lab and start again with the first lab.

Note: We have included configuration file samples in their entirety to make catching up with a particular lab easier. For example, if your lab times out and you were in a later lab, simply copy the configuration file contents into the appropriate file and continue with the next lab.

Lab 11.1: Create the limit_test.conf Configuration File

Learning Objectives

By the end of the lab you will be able to:

- Create a new configuration file and add a service block.
- Test for results

Overview

In this exercise, you will create a new configuration file in preparation for testing restricting access and the view the results of your requests.

Steps

1. Rename the **`default.conf`** configuration file:

   ```
   $ mv /etc/nginx/conf.d/default.{conf,bak}
   ```

2. Create a new configuration file named **`limit_test.conf`**:

```
$ vim /etc/nginx/conf.d/limit_test.conf
```

3. Create a server block in the **limit_test.conf** file:

   Note: When in Vim, press **i** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;

  location / {
  }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

4. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

```
:wq
```

5. Reload your NGINX configuration with

```
$ nginx -s reload
```

6. Test your NGINX configuration with:

```
$ curl localhost | grep -i "welcome"
```

   Inspect the output for the phrase **Welcome to nginx!**, which should at the bottom of the command's output.

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   612  100   612    0     0   111k      0 --:--:-- --:--:-- --:--:--
119k
<title>Welcome to nginx!</title>
<h1>Welcome to nginx!</h1>
```

End of Lab

Lab 11.2: Deny Requests

Learning Objectives

By the end of the lab you will be able to:

- Add a directive to the server context that denies all connections
- Test for results

Overview

In this exercise, you will configure your server to deny all connections using the deny directive and a parameter of all. You'll then test to see the results are as expected.

Steps

1. Edit the **`limit_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/limit_test.conf
   ```

2. Add a **deny** directive in the server context:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;
  deny all;

  location / {
  }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   ```
   :wq
   ```

4. Reload your NGINX configuration with

   ```
   $ nginx -s reload
   ```

5. Test the your configuration results with:

   ```
   $ curl http://localhost
   ```

   ```
   <html>
   <head><title>403 Forbidden</title></head>
   <body>
   <center><h1>403 Forbidden</h1></center>
   <hr><center>nginx/1.17.6</center>
   </body>
   </html>
   ```

   The system should display a **403 Forbidden** message.

End of Lab

Lab 11.3: Allow Requests from a Specific Server

Learning Objectives

By the end of the lab you will be able to:

- Configure the server to allow requests from a specific system
- Test for results

Overview

In this exercise, you configure your recently secured server to allow requests from a specific system.

Steps

1. Edit the **`limit_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/limit_test.conf
   ```

2. Add an **`allow`** directive **ABOVE** the **`deny`** directive to allow a host to access the system.

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;
  allow 127.0.0.1;
  deny all;

  location / {
  }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   `:wq`

4. Reload your NGINX configuration with

   `$ nginx -s reload`

5. Test the your configuration results with:

   `$ curl http//localhost`

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   612  100   612    0     0   111k      0 --:--:-- --:--:-- --:--:--
119k
<title>Welcome to nginx!</title>
<h1>Welcome to nginx!</h1>
```

   The test shows the configuration now allows the localhost to make requests.

   Note: If you do NOT have the correct output, check the configuration file to confirm that the **allow** directive comes **FIRST**, then the **deny** directive.

End of Lab

Lab 11.4: Setup Basic Authentication with auth_basic

Learning Objectives

By the end of the lab you will be able to:

- Configure your system to use **`auth_basic`** for authentication
- Configure a password in the **`htpasswd`** file
- Generate a password with **`openssl`**
- Test for results

Overview

In this exercise, you will be setting up your system to use basic authentication by adding `auth_basic` directives to the configuration, including adding passwords to the `htpasswd` file and then testing to see what requests work and how.

Steps

1. Edit the **`limit_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/limit_test.conf
   ```

2. Add the **`auth_basic`** and **`auth_basic_user_file`** directives to your configuration:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;
  allow 127.0.0.1;
  deny all;
  auth_basic "restricted";
  auth_basic_user_file /etc/nginx/htpasswd;

  location / {
  }
```

```
}
```

Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   ```
   :wq
   ```

4. Generate a password using the **openssl** command:

   ```
   $ openssl passwd
   ```

   You'll be prompted to enter a password and then enter it again to re-confirm it's the same. The encrypted password will then be displayed as output from the command.

   Use the indicated password below in **red**.

   ```
   Password: somepwd
   Verifying - Password: somepwd
   0iCu812.Br549
   ```

   Note: COPY THE PASSWORD to your clipboard by highlighting the text with the mouse and either pressing **Ctrl-Shift-C** or right-clicking and choosing the **Copy** menu item.

5. Create an **htpasswd** file:

   ```
   $ vim /etc/nginx/htpasswd
   ```

6. Create an **admin** account and associate the newly-generated password with it:

   Note: When in Vim, press **i** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text. Please be certain to put in YOUR openssl-generated password, not our example one.

   ```
   admin:0iCu812.Br549
   ```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

7. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

```
:wq
```

8. Reload your NGINX configuration with

```
$ nginx -s reload
```

9. Test the your configuration results with:

```
$ curl https://localhost
```

```
<html>
<head><title>401 Authorization Required</title></head>
<body>
<center><h1>401 Authorization Required</h1></center>
<hr><center>nginx/1.17.6</center>
</body>
</html>
```

The request should generate a return that states that a **401 Authorization Required** message.

End of Lab

Lab 11.5: Limit Connection Rate

Learning Objectives

By the end of the lab you will be able to:

- Add a connection limitation directive to the **http** context
- Turn off basic authentication
- Add a connection limitation directive to the **server** context
- Test the results

Overview

In this exercise, you will add a **`limit_conn_zone`** directive to the http context, then turn off basic authentication and add a **`limit_conn`** directive the server context.

Steps

1. Edit the **`limit_test.conf`** file:

```
$ vim /etc/nginx/conf.d/limit_test.conf
```

2. Add a **`limit_conn_zone`** directive in the http context (above the server context), then add a **`limit_conn`** directive to the server context, then finally modify the location by turning off basic authentication:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New/changed configuration is shown in **bold** text.

```
limit_conn_zone $remote_addr zone=ip:10m;

server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;
  allow 127.0.0.1;
  deny all;
  auth_basic "restricted";
  auth_basic_user_file /etc/nginx/htpasswd;
  limit_conn ip 1;

  location / {
    auth_basic off;
  }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

10. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

    ```
    :wq
    ```

11. Reload your NGINX configuration with

    ```
    $ nginx -s reload
    ```

End of Lab

Lab 11.6: Limit Request Rate

Learning Objectives

By the end of the lab you will be able to:

- Set up a request limit zone
- Implement the request limit in the server context
- Test the results

Overview

In this exercise, you will setup a request limit zone directive.

Steps

1. Edit the **`limit_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/ssl_test.conf
   ```

2. Setup a **`limit_req_zone`** directive in the http context (above the server context), and then add a **`limit_req`** directive to the server context:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
limit_conn_zone $remote_addr zone=ip:10m;
limit_req_zone $server_name zone=one:10m rate=1r/s;

server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/limit.access.log combined;
  error_log /var/log/nginx/limit.error.log notice;
  allow 127.0.0.1;
  deny all;
  auth_basic "restricted";
  auth_basic_user_file /etc/nginx/htpasswd;
  limit_conn ip 1;
  limit_req zone=one burst=5;

  location / {
    auth_basic off;
  }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   ```
   :wq
   ```

4. Reload your NGINX configuration with

   ```
   $ nginx -s reload
   ```

End of Lab