Overview

This module gives students hands-on experience installing and interacting with a web server. Students will install and start a web server, generate a simple HTML file, and use a client node to retrieve the file.

Objectives

Upon completing this module you will:

- Be able to install and start a web server
- Be exposed to creating and editing files using a console editor
- Be able to retrieve a file from the web server

Tutorial

A. Slice Creation and Instrumentation

This module assumes you have an active slice with two connected nodes and SSH terminals to both nodes. If you don't, use RSpecs to reserve a server and client nodes, then continue here.

C. Installing An HTTP Server

1. SSH to the server node.

2. Install a web server by typing

```
 sudo apt-get update
```

```
 sudo apt-get -y install apache2
```

"sudo" allows you to perform this action as root and apt-get is a package-management utility for this distributions of Linux. Once completed, you should see the "Starting web server apache2" message, which means a web server in default configuration is now running and bound to port 80 (the default HTTP port).

D. Create a message to retrieve

1. Still on the server, navigate to the http root directory by typing in the following command:

```
cd /var/www/html
```

This is the default directory from which the web server will serve content.

2. Remove the existing default webpage by typing:

```
sudo rm index.html
```

3. Now, use nano or vim to make a new index.html and type a message that will be retrieved later:

```
sudo nano index.html
```

or

```
sudo vim index.html
```

4. Type in any message that you wish to transmit from the server to the client; it need not be HTML.
nano: type the message and then press Ctrl-X to exit. When it asks "Save modified buffer?", press "y" and "enter" to use the same filename.
vim: press the ‘i’ key, then type the message. Once you’ve finished, hit escape (Esc) and then type ":wq" to save changes and quit.

E. Retrieve the message

1. Login to the client node. Also get the server IP by running the "ifconfig".

2. In the **client's SSH terminal**, run the following command:

```
curl -v http://SERVER-IP/test.html
```

After you hit enter, you will see lines being printed on the shell with ">" or "<" signs at the begning, "<" show the messages client is sending to the server and ">" indicates the messages being recived from the server. Sounds familiar?
Question 1: What was the output of the above command and why? Provide screenshots.

Now, in the **client's SSH terminal**, run the following command:

```
curl -v http://SERVER-IP
```

Question 2: What was the output of the above command and why? Provide screenshots.

Going Further

Question 3: Create your own correctly formatted HTML document in place of the message in index.html. Follow the same procedure as above to retrieve this document. Take a screenshot!

Question 4: You’ve learned that HTTP has many different status codes. By modifying your requests made from the client machine you already observered two of them, explain briefly what do they mean and explain 2 more status codes by reading up on http.

Final Step

Upon completion of the experiment free the resources. And submit answers to above 4 questions on gradescope by **Monday, October 14, 2019**.