VANIER COLLEGE - Computer Engineering Technology
247-609 Networked Embedded Systems

Lab 2 : BBB as Web Server

Leonardo Fusser (1946995)

Purpose:

a) To familiarize with network capability of BBB platform using Internet-over-USB.
b) To utilize BBB as a basic web server.
c) To familiarize with basic HTML and create simple web page.
d) To familiarize with basic CGI scripts.

To be submitted :

1. Deadline: by the end of the lab session.
2. **No formal report is required.** The followings are required as submission.
    a. Properly indented and commented code, screen shots (clearly labelled).
    b. Answer questions in the lab.
    c. Include a final discussion and conclusion session.

Theory:

There are low-overhead servers available such as lighttpd, Boa, Monkey and Nginx, but Apache is a good web server to use for getting started. In fact, it is currently installed in the BBB Debian image. Once you are familiar with Apache, you could then investigate web server solutions that are designed specifically for embedded Linux devices.

Running a web server on the BBB provides you with a number of application possibilities, including the following:

- Present general web content to the world.
- Integrate sensors and display their values to the world.
- Integrate sensors and use it to intercommunicate between devices.
- Provide web-based interfaces to tools that are running on the BBB.

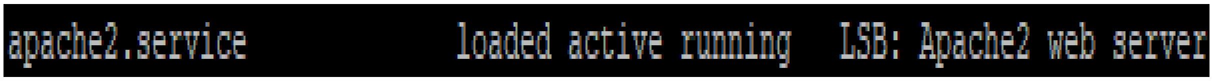
The basic connection between BBB and your desktop computer using the supplied USB lead includes a *Linux USB Ethernet/RNDIS* (for Internet-over-USB) driver.

Lab Work:

If your internet-over-usb does not work, see the following document:
BBB_connection.docx

Part 1 : Apache Server

1. Using "`systemctl`" command, verify that apache2 server is currently running. If apache2 is not installed, you will have to install it. Perform a screen shot to show that your apache2 is running.

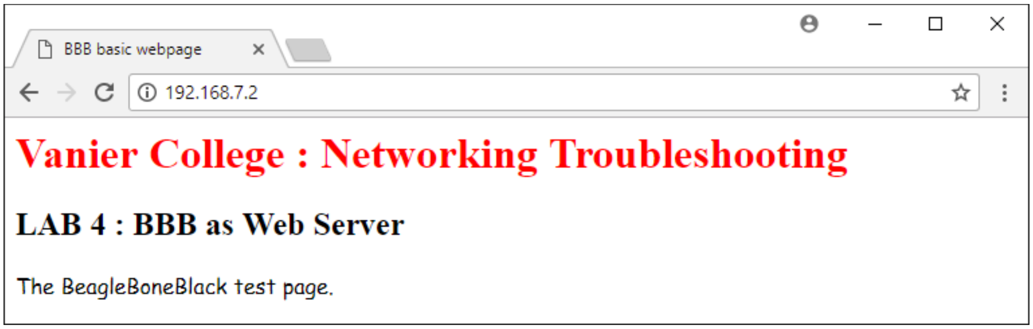


```
apache2.service               loaded active running   LSB: Apache2 web server
```

Figure 1. Screenshot above shows the apache2 web server service running on the BBB.

Part 2 : Creating Web Pages and Web Scripts

2. Create a simple web page as shown below for the BBB web server, you may use the nano editor to create an index.html file inside `/var/www/html` directory. Note that the font in paragraph is of type "comic sans MS".



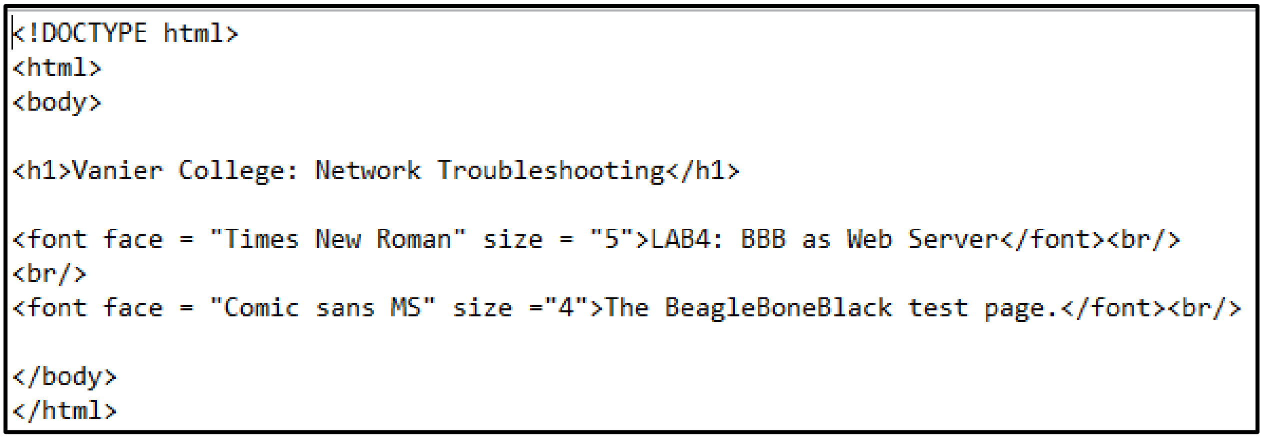
Vanier College : Networking Troubleshooting

LAB 4 : BBB as Web Server

The BeagleBoneBlack test page.

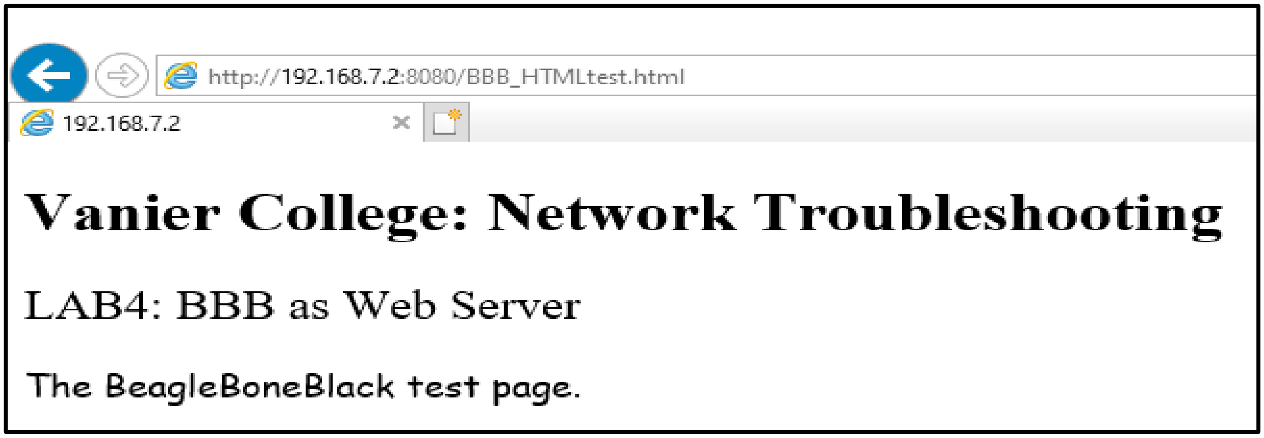
3. Connect to the web server on the BBB using a web browser, you will see the output of your HTML file. Attach a copy of your code and the corresponding screen shot.

   - ➢ Refer to screenshots on next page.



```
<!DOCTYPE html>
<html>
<body>

<h1>Vanier College: Network Troubleshooting</h1>

<font face = "Times New Roman" size = "5">LAB4: BBB as Web Server</font><br/>
<br/>
<font face = "Comic sans MS" size ="4">The BeagleBoneBlack test page.</font><br/>

</body>
</html>
```

Figure 2. Screenshot above shows simple HTML webpage code on BBB.



Vanier College: Network Troubleshooting

LAB4: BBB as Web Server

The BeagleBoneBlack test page.

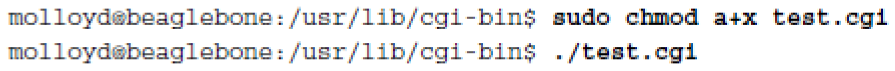
Figure 3. Screenshot above shows simple HTML webpage running from apache2 web server.

Part 3 : Common Gateway Interface (CGI) scripts

4. The default location is the `/usr/lib/cgi-bin/` directory, where a simple script named test.cgi as shown below can be created. Replace `YourFirstName` with your first name.

```
#!/bin/bash
echo "Content-type: text/html"
echo ""
echo '<html>'
echo '<head>'
echo '<meta http-equiv="Content-Type" content="text/html";
charset=UTF-8">'
echo '<title>Lab 2 from YourFirstName</title></head>'
echo '<body><h1>Hello BeagleBone</h1><para>'
hostname
echo ' Today is '
date
echo '</para></html>'
```

5. The script must then be made executable, and it can be tested using the following command.



```
molloyd@beaglebone:/usr/lib/cgi-bin$ sudo chmod a+x test.cgi
molloyd@beaglebone:/usr/lib/cgi-bin$ ./test.cgi
```

The script is quite verbose, but it is very easy to call system commands (e.g., hostname and uptime) directly from within it. When the script is tested in the terminal window, its output displays HTML source code.

6. You may need to enable CGI scripts. Find out how that could be performed. Clearly list all the procedures and commands involved in this step.

- Two commands needed to be entered in order to enable CGI scripts on the BBB. The following two commands were:

  ```
  sudo ln -s /etc/apache2/mods-available/cgi.load /etc/apache2/mods-enabled/
  ```

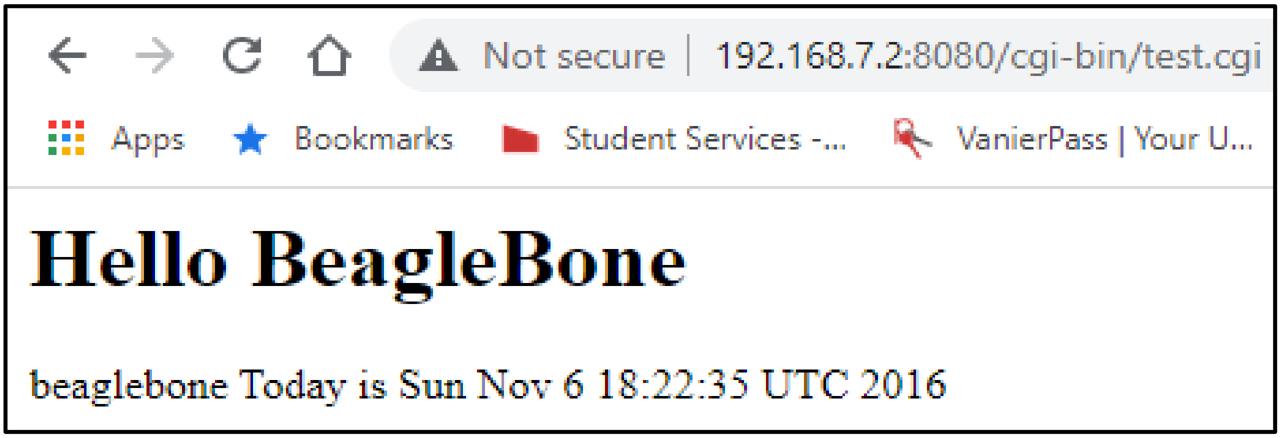
  Which enables CGI scripts.

  And...

  ```
  sudo systemctl restart apache2
  ```

  Which restarts the apache2 web server service on the BBB to apply the change.

7. Access the webpage. Make sure this basic CGI script is functioning as expected. Attached a screen shot of your corresponding webpage.



Hello BeagleBone

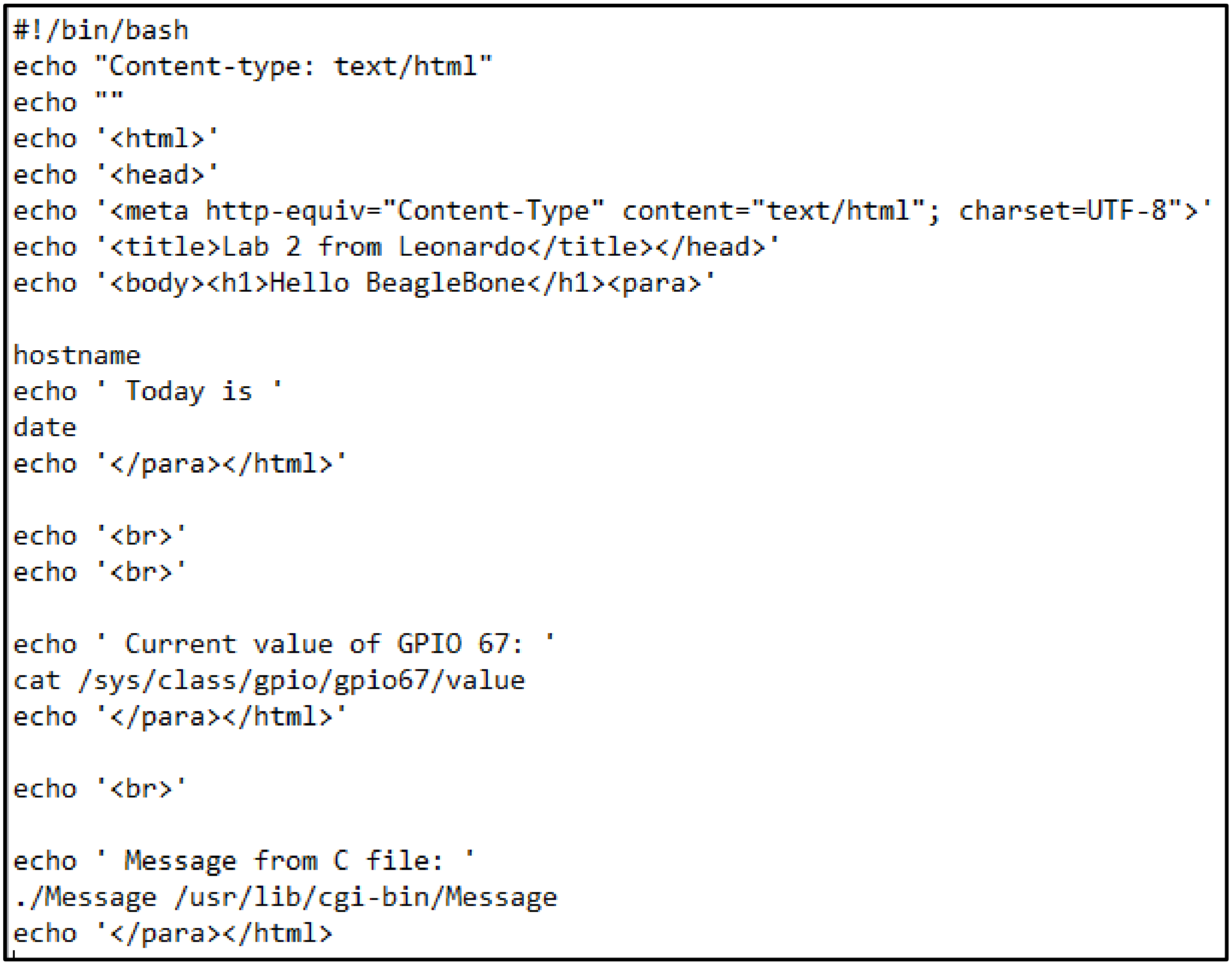
beaglebone Today is Sun Nov 6 18:22:35 UTC 2016

Figure 4. Screenshot above shows simple CGI script running from apache2 web server.

8. Modify the script to include the following features :
    a. Modify the script to read the PB value and display it in the web page.
    `(Hint: remember /sys/class/gpio…)`
    b. ~~Add an html line to refresh the page every second~~.
    c. Write a C file to print hello and run this binary from your CGI.

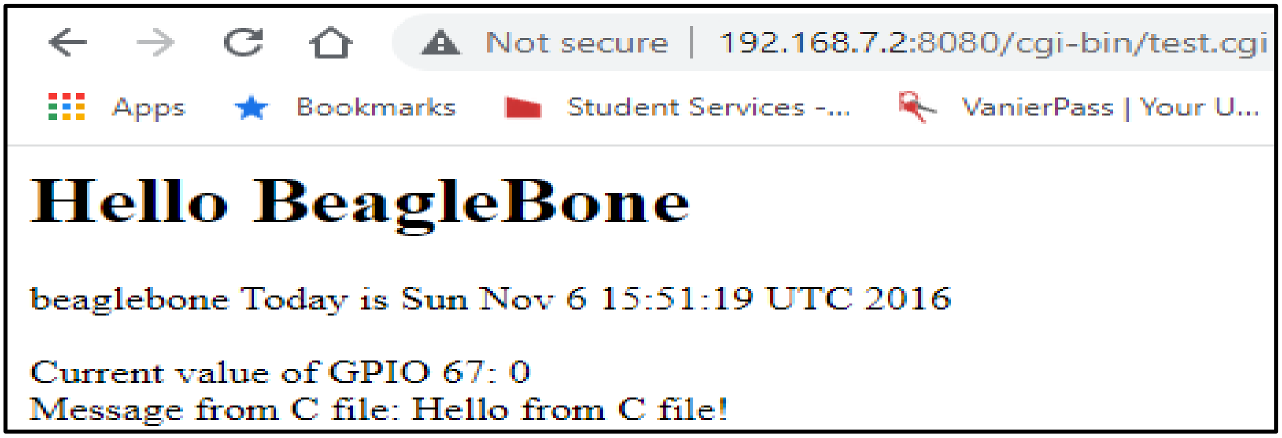
9. Attached all source code, and a screen shot on the final look of the webpage.

- ➢ Refer to screenshot on next page.



```
#!/bin/bash
echo "Content-type: text/html"
echo ""
echo '<html>'
echo '<head>'
echo '<meta http-equiv="Content-Type" content="text/html"; charset=UTF-8">'
echo '<title>Lab 2 from Leonardo</title></head>'
echo '<body><h1>Hello BeagleBone</h1><para>'

hostname
echo ' Today is '
date
echo '</para></html>'

echo '<br>'
echo '<br>'

echo ' Current value of GPIO 67: '
cat /sys/class/gpio/gpio67/value
echo '</para></html>'

echo '<br>'

echo ' Message from C file: '
./Message /usr/lib/cgi-bin/Message
echo '</para></html>
```

Figure 5. Screenshot above shows CGI script code on BBB.



Hello BeagleBone

beaglebone Today is Sun Nov 6 15:51:19 UTC 2016

Current value of GPIO 67: 0
Message from C file: Hello from C file!

Figure 6. Screenshot above shows CGI scrip code running from apache2 web server.

Discussion:

For the first part of the lab, it was verified that the apache2 web server was installed and running on the BBB. For our version of Debian for BBB, apache2 web server came already pre-installed. When checked for operation, the apache2 web server service was already running. No real configuration needed to be done for this part.

For the second part of the lab, a simple HTML webpage was created. Various font types were used for experimentation purposes. The webpage was verified for functionality by running it from the apache2 web server on the BBB. This was accomplished by connecting to the apache2 web server from Internet Explorer on Windows (by typing BBB ip address followed by specifying port 8080 in URL). From there, the newly created HTML webpage was opened and was verified for correct functionality.

For the final part of the lab, a basic CGI script that incorporated HTML functionality was created. But before doing so, CGI scripts needed to be enabled on the BBB in order for this to work properly. Only two commands needed to be entered to accomplish this. This CGI script was tested in the same fashion as the HTML webpage from the previous part of the lab. The CGI script was then modified to read the status of one of the BBB GPIO pins and to read a message from a separate C file. To read the status of the GPIO pin, cat was used and displayed on the output of the CGI script. To read the message, a separate C file needed to be created, compiled (using GCC) and made executable. Inside the C file, a simple message prints. When this file is called from the CGI script, it reads the message by calling the executable for the C file to be shown on the output of the CGI script.

The overall lab was a success.

Conclusion:

- Successfully familiarized Linux Internet-over-USB capability on BBB platform.
- Successfully utilized BBB as a basic web server using apache2.
- Successfully familiarized with HTML and created simple webpages in HTML.
- Successfully familiarized with CGI scripts and implemented HTML within it.