Raspberry Pi USB Security Camera System:

3. Creating a Central Access Point for Your Security System Using "lighttpd" and PHP

Once you have both "motion" and "pagekite" up and streaming globally from the RPi, the next step will be to create a central access point for your system. This will allow access to both the stream and the database from a single location (web page). Currently, the stream and database each has its own separate link, which looks like the following:

https://subdomain-maindomain.pagekite.me

However, we would like to use a single link to access both webpages, and this should ideally have the primary domain name, like so:

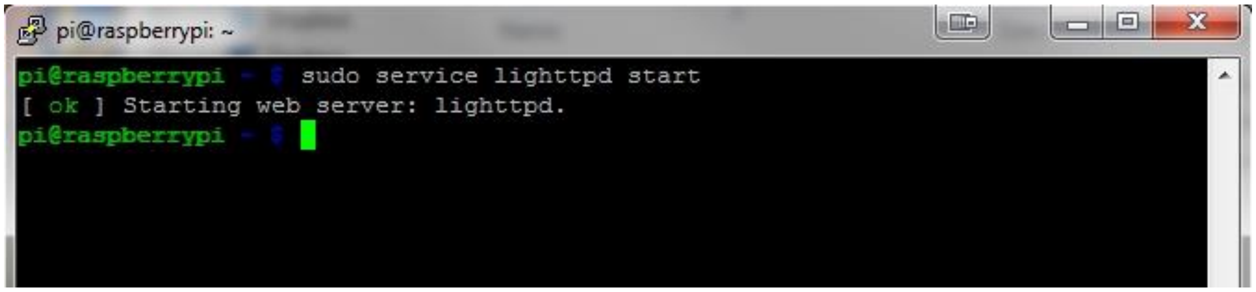
https://maindomain.pagekite.me

Therefore, we require a method to link from a single page to multiple other pages. This will not only allow access to a single stream and database, but multiple pi/camera combinations all streaming from different places. This way, you can access the entire system by only inputting a single address, which is, of course, ideal. The complete guide to this next bit is available at the following link, though we will not be installing/configuring everything that they cover: http://www.penguintutor.com/linux/light-webserver

To start, we will need to install a web server to host the web page that we will link each of the streams and corresponding databases from. Any web server should work, but for this project we only need a lightweight one such as "lighttpd". We can install this via apt-get by inputting the following:

$ sudo apt-get install lighttpd

This will install both "lighttpd" and any dependencies related to the package. If prompted to do so, install these other components. Once the installation completes, you should receive the following message:



```
pi@raspberrypi ~ $ sudo service lighttpd start
[ ok ] Starting web server: lighttpd.
pi@raspberrypi ~ $
```

If the package fails to install, try updating using the **$ sudo apt-get update** command, and then trying again. Note that since I already had the package installed, I simply stopped and started the process to display the same message. "Lighttpd" can be started using the command shown in the image above, and stopped by replacing "start" with "stop" in the same command.

Once “lighttpd” and all related components have been installed, we need to install PHP to enable us to write a script for embed the links to the various “pagekite” pages on the web page. This can also be accomplished using simple “apt-get” commands, but this must be done in the exact order listed:

1. Install PHP5 (Note: version 5, specifically) by inputting the following:

   ```
   $ sudo apt-get install php5-common php5-cgi php5
   ```

   If this is not done in this order, you may also install “Apache” web server, which is obviously not needed since we have already installed “lighttpd”. Again, note the version.

2. Install “fastcgi” by inputting the following:

   ```
   $ sudo lighty-enable-mod fastcgi-php
   ```

   This will install the “fastcgi” module for PHP. The purpose of this is to allow PHP scripts to be handled by the “lighttpd” web server, which is crucial for our purposes.

3. Reload the “lighttpd” server to apply the changes made from PHP by inputting the following:

   ```
   $ sudo service lighttpd force-reload
   ```

In the guide I followed, there were additional instructions for installing/configuring MySQL and other packages, but these can be ignored as they do not help us in this particular project.

The next part will be simply allowing the user (you) to make changes to the “lighttpd” and PHP configurations by changing the permissions for the web directory of the pi, named */var/www/*, to allow the default pi user to have the same privileges as root, thus eliminating the need to login as root every time you want to make a change to one of these packages. To do this, enter the following commands in this order:

1. Change directory owner/group to *www-data*:

   ```
   $ sudo chown www-data:www-data /var/www
   ```

2. Give current user write privileges to the */var/www/* directory:

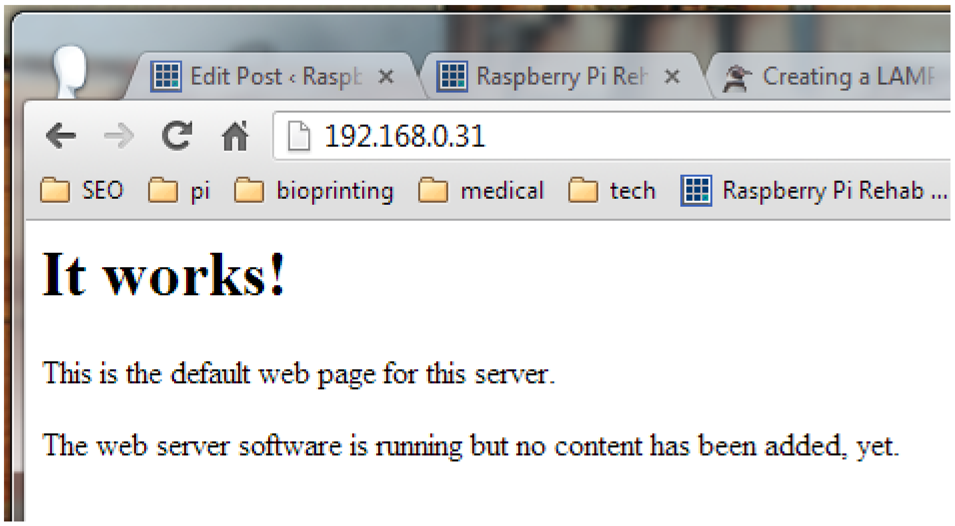
   ```
   $ sudo chmod 775 /var/www
   ```

3. Add the default “pi” user to this group:

   ```
   $ sudo usermod -a -G www-data pi
   ```

Reboot the system to ensure that all changes have taken place before testing the web server.

Next, test to make sure everything has installed and been configured properly by opening a web browser and entering the IP address of the RPi in the address bar. This will go to the default web server on port 80, which should be "lighttpd". Below is an example, though your IP will be different, most likely starting with 192.168.X.X, depending on your current connection.



It works!

This is the default web page for this server.

The web server software is running but no content has been added, yet.

Next, we will want to test that PHP is working properly, rather than the default HTML script. To do this, first navigate to the web directory */var/www/* using the following command:

```
$ cd /var/www/
```

In this directory, there will be a file called *index.lighttpd.html*, which is the default HTML script that displayed the "It works!" message. We will want to remove this HTML file, create a new PHP script file, and edit it to test PHP using the following series of commands:

1. Remove the default index file:

   ```
   $ rm index.lighttpd.html
   ```

2. Create the new PHP script file:

   ```
   $ touch index.php
   ```

3. Open the new file in any text editor, here I've used "nano":

   ```
   $ sudo nano index.php
   ```

4. There should be a blank file. Copy the following code to test PHP:

```
<?php
  print <<< EOT
<!doctype html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Test successful</title>
</head>
<body>
<h1>Test successful</h1>
<p>Congratulations.</p>
<p>Your webserver and PHP are working.</p>
</body>
</html>
EOT;

?>
```

5. Test that PHP is working by again entering the IP of your RPi into the address bar in a web browser. If all goes well, you'll see another confirmation message. If not, make sure the code is the same as above and the "index.php" file is located in the */var/www/* directory.

Now that PHP is functional, we need to simply edit this default script for our needs. Note that this is nothing fancy looking, but purely functional. This includes changing titles and headers, which are clearly labeled in the script, but also embedding the links to the streams/databases. To embed a link in text on the web page, place the following line of code in the script:

<a href="*LINK*">*TEXT*</a>

An example "index.php" script is shown below. Note that additional links could be added for more stream/database combinations from additional pi/camera combinations being set up:

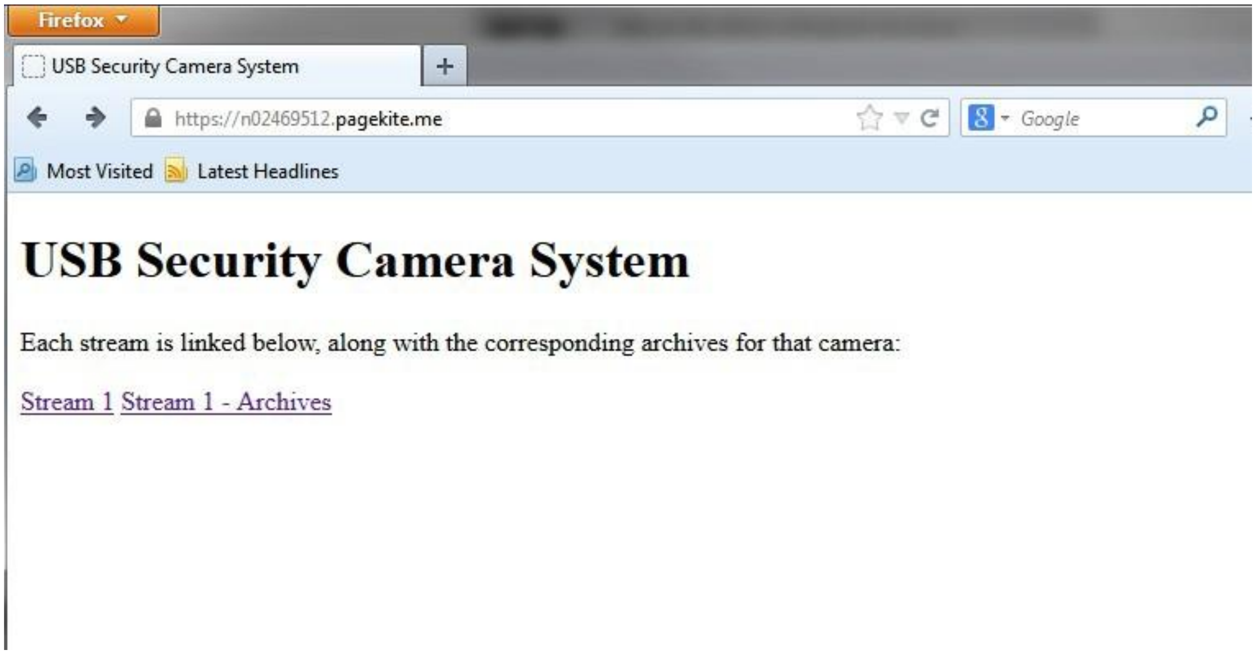
```
<?php
  print <<< EOT
<!doctype html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>USB Security Camera System</title>
</head>
<body>
<h1>N02469512 Security Camera System</h1>
<p>Each stream is linked below, along with the corresponding archives for
that camera:</p>
<a href="https://stream1-n02469512.pagekite.me">Stream 1</a>
<a href="https://archives1-n02469512.pagekite.me">Stream 1 - Archives</a>
<br>
<a href="https://stream2-n02469512.pagekite.me">Stream 2</a>
<a href="https://archives2-n02469512.pagekite.me">Stream 2 - Archives</a>
<br>
</body>
</html>
EOT;

?>
```

Save this script in the */var/www/* directory.

Now it is time to test the entire thing. We will need an additional "pagekite" page for the web server to run on. We will include this when we launch the other two pages (for the stream and database). Get all three "pagekite" pages up and running by entering the following command, inserting your specific kite names and username/password combinations. Note that we were able to add the "lighttpd" server by specifying port 80 as a source and using the main domain name as the address:

```
$ pagekite /tmp/motion archives-yourkitename.pagekite.me +indexes
+password/USER=PASS AND 8081 stream-yourkitename.pagekite.me
+password/USER=PASS AND 80 yourkitename.pagekite.me
```

Again access the web server in a web browser by entering the address of your kite with the main domain name into the address bar (this will be the address you specified above for the lighttpd server). You should see something that looks like this, but yours should have a second stream and database if you embedded a second link:



USB Security Camera System

Each stream is linked below, along with the corresponding archives for that camera:

Stream 1 Stream 1 - Archives

Test it out by clicking each one of the links and seeing if it correctly links to the appropriate stream/database. Note that you will be prompted to login to each of these sub-domains but not the main page. This is done to ensure maximum security, due to the fact that anyone with the direct link to a sub-domain could bypass this main page by directly entering the link to one of the databases/streams. Also note that this is easily expandable by adding additional "pagekite" pages with additional streams/databases for other pi/camera combinations and then linking them here on this page. The project is now complete! While this may not be a "high-tech" security system, it definitely provides the basic functions of a camera system with a reasonable amount of security from both the SSL and username/password protected links.