Web Basic Admin & CGI

Created by Jason Madar
Modified by Ivan Wong

Basic Admin

Linux is an amazing OS where it runs from the smallest computers such as Raspberry Pi (*http://www.raspbian.org/*) to Smartphones (*https://haydenjames.io/85-of-all-smartphones-are-powered-by-linux/*) to Supercomputer clusters (*https://itsfoss.com/linux-runs-top-supercomputers/*).

But the most popular use of Linux has to be running web servers on the Internet. The goal of this assignment is to teach you to run your own web server so anyone on the Internet can access your content and run your scripts.

You should have received an email from AWS academy, which gives you $50 credit to access the Amazon Web Services (AWS) console.

Launch your own Linux instance with Apache HTTP server on AWS

1. Follow this video to setup your aws account: *https://youtu.be/puu_G8ANeTM*

   Note: if you are using PuTTY to connect to your ec2 instance, please follow these instructions: *https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html*

2. Follow this video to get a quick introduction on installing and using apache2 web server: *https://youtu.be/KyN0DTeSE8A*.

   Notes:

   a. We will be using Ubuntu Server 22.04, which is a very popular linux distribution.
   b. Make sure your instance type is t2.nano, this is minimize your cost
   c. By default, the apache http server stores files in these locations
      i. Static files: /var/www/html/
      ii. CGI scripts: /usr/lib/cgi-bin/
      iii. Configuration files: /etc/apache2/

3. Getting a static ip to your instance using AWS's "Elastic IP" feature
   a. You instance's IP may change between start and stop. For a web server, we want the IP to not change
   b. *https://youtu.be/mgfpduy5ZAo*

c. After following the instructions, try stopping and restarting your instance to make sure that the IP is preserved

4. Create DNS entry
    a. DNS is used to associate meaningful names with ip address
    b. I reserved the wmdd4950.com domain for this class
    c. I also setup a simple REST API so you can manage creating and deleting wmdd4950.com subdomains
    d. Pick a subdomain of your choice and associate your static ip with it
    e. DNS API
        i. List all subdomains: curl api.wmdd4950.com/arecord/list
        ii. Add a subdomain: curl api.wmdd4950.com/arecord/${sub_domain}?ip=${ip_address}
        iii. For example, the following command will associate the ip address 54.70.53.85 with jmadar.wmdd4950.com
        iv. curl api.wmdd4950.com/arecord/jmadar?ip=54.70.53.85
        v. To delete a subdomain: curl -XDELETE api.wmdd4950.com/arecord/${sub_domain}
        vi. For example, to delete the jmadar.wmdd4950.com subdomain: curl -XDELETE api.wmdd4950.com/arecord/jmadar

CGI Script

Bash scripts can be called by a special URL on a web server.

For this assignment, put all your scripts inside the /usr/lib/cgi-bin/web-admin-cgi-scripts/ directory.

NOTES
You will need to use sudo for some of the commands because you will be writing files / creating links in privileged areas of the filesystems. Make sure you review sudo and the file systems from the follow videos (optional but good to know):

- https://ww.linkedin.com/learning/learning-linux-command-line-2/user-roles-and-sudo?u=57075641
- https://www.linkedin.com/learning/learning-linux-command-line-2/the-linux-filesystem?u=57075641
- The most important directory is /usr/lib/cgi-bin/. If you put an executable file here, it can be called from the web via http://${your_server_ip}/cgi-bin/<script file name>
- Put your work inside the /usr/lib/cgi-bin/web-admin-cgi-scripts/ directory
- Your script can be accessed by anyone on the Internet!!!
- For scripts to be runnable by the web server, the output requires a couple of special lines. Below is the example script:

```
#!/bin/bash
# need to start with a line identifying the output mime type
# this is part of the HTTP response header
echo "content-type: text/plain"

# an empty line to indicate the end of header and the start
# of the content
echo

# you can now write the rest of your script below
echo "Hello World"
```

- For additional technical information on CGI scripts, you can read the official Apache documentation at https://httpd.apache.org/docs/2.4/howto/cgi.html#writing. Start at the section labeled "writing" as all the steps before have already been set up.
- It's important to learn how to read official software documentation as there will be times during your software engineering career when no tutorials are available!

Question 1

If you follow my instructional video (https://www.youtube.com/watch?v=KyN0DTeSE8A), you would have successfully created test.sh. Rename it to q1.sh and move it inside `/usr/lib/cgi-bin/web-admin-cgi-scripts/q1.sh`

For q2 to q7, you will be writing scripts that are executable on your web server. For example, we should be able to run your script by visiting:

`curl http://<your ip>/cgi-bin/web-admin-cgi-scripts/q[1-7].sh`

NOTE: For the rest of the questions, I have also provided a running version so you can test your against the answer key. However, you are not allowed to use my script in your answer (i.e. by doing a curl to my scripts)

Question 2

Write a script called q2.sh that will tell the visitor a random joke. The output of this script can be found at http://learn.operatoroverload.com/~jmadar/1280/q2.sh so you can check your output to see if it is correct.

You can get a new joke by doing a curl on https://icanhazdadjoke.com. i.e.

```
$ curl -L "https://icanhazdadjoke.com"
What did the pirate say on his 80th birthday? Aye Matey!
```

HINT: The answer to this question is very simple. When q2.sh is called, your script will simply make a call to icanhazdadjoke.com and return the result.

Question 3

Write a script q3.sh that retrieves all slugs from the current time.com home page but display they as titles (i.e. remove all dashes).

You can access the JSON data for time.com at https://time.com/wp-json/wp/v2/posts/?per_page=10&context=embed I only want the slugs, not the entire URL.

A couple of things to note:

- To retrieve the data that made up of a reddit page, use the following command:

  ```
  curl -s -L "https://time.com/wp-
  json/wp/v2/posts/?per_page=10&context=embed" | json_pp
  ```

- A running version of the script can be found at https://learn.operatoroverload.com/~jmadar/1280/q3.sh

Question 4

Just like the command line, you can pass arguments into your script running on a web server using the following scheme. Assuming you have a script called test.sh:

```
http:///cgi-bin/test.sh?arg1
```

Will pass the string arg1 into your script as $1 variable.

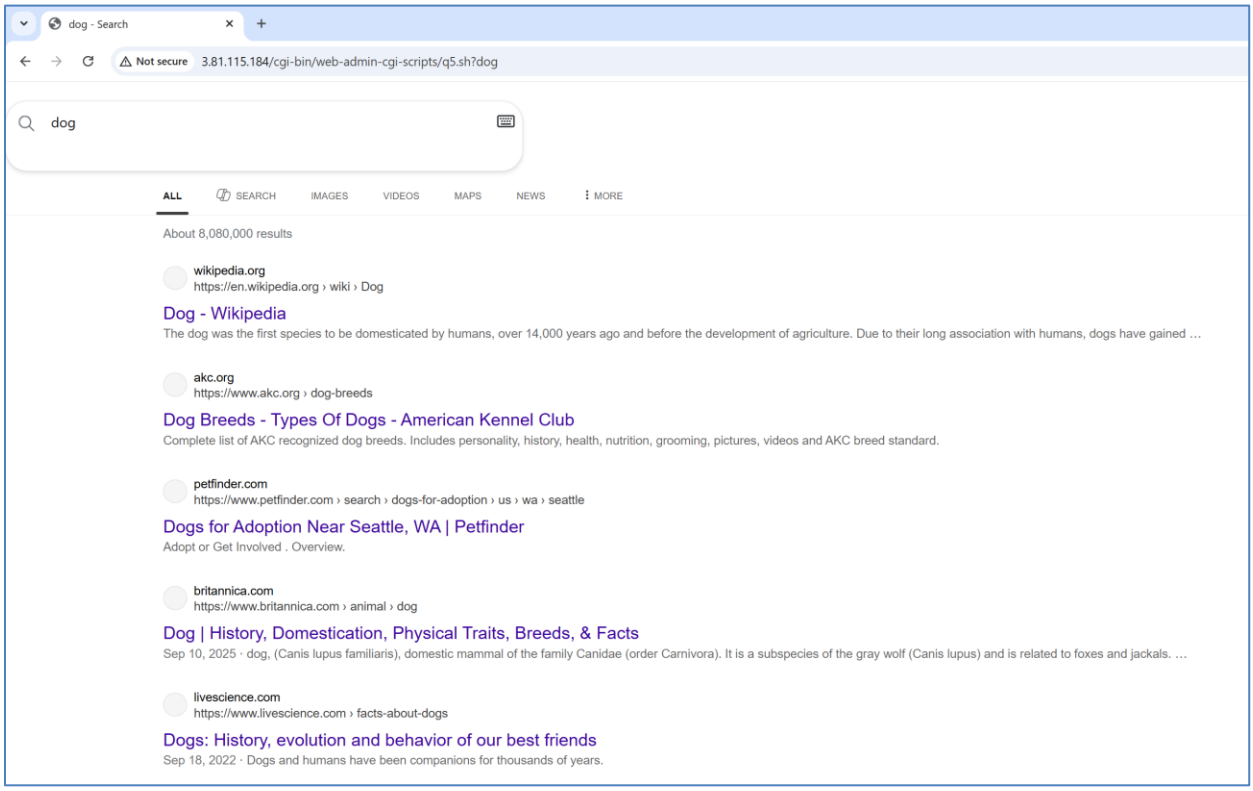
Create a script q4.sh by modifying q3.sh, where users can provide an argument to specify how many slugs to show. The script will return all the titles on that subreddit. For example:

http://learn.operatoroverload.com/~jmadar/1280/q4.sh?20 will return 20 slugs from time.com

If no argument is provided, output an error message You can use http://learn.operatoroverload.com/~jmadar/1280/q4.sh to see the intended output of your script.

Question 5

Check out http://learn.operatoroverload.com/~jmadar/1280/q5.sh?dog. It returns the bing search result of ‘dog’. The actual call to bing is performed via curl within my script! (*Note: The above script may not work properly now, but you can refer to the following screenshot.)



To achieve this yourself, your script will need to do the bing search on behalf of the user. The key observation is that you can provide a search term as part of the URL as follows https://wwww.bing.com/search?q=${SEARCH_TERM}

You can prove this by issuing the following curl command on the terminal:

```
$ curl -s -L "https://wwww.bing.com/search?q=dog"
```

Create a script q5.sh that behaves like http://learn.operatoroverload.com/~jmadar/1280/q5.sh, where when provided a parameter, it will perform a google search and return the result.

NOTE: since HTML is returned from google, you will need to set the content-type to be ‘text/html’ in your script. Otherwise the browser won’t display the search result properly.

So why do this? When you do a search on google, google will keep track of your search history, preferences, etc. by keeping track of your IP, browser login, etc. What we are doing here is

have your web server do the search on behalf of you. This way google doesn't know who is actually doing the search as all searches will appear to come from the web server. This is the primary idea behind various VPN technologies.

Question 6

Note: getting more challenging
The Internet Movie Database (IMDB) has all their data available for the public to download, the dataset explanation is at https://www.imdb.com/interfaces/ and the download location can be found at https://datasets.imdbws.com/

For this exercise, you'll first need to download and decompress name.basics.tsv.gz and title.basics.tsv.gz files onto your server.

These are "tab separated files" (tab being the delimiter between fields). The name.basics.tsv contains the names actors, directors, etc. while titles.basics.tsv contains information about a specific title.

Your job is to write a script, q6.sh, that takes in an actor or actress's name as input, and outputs all the movie titles that the actor is known for. If more than one actor matches, a message will display telling the user that the script expects only one actor or actress to match. Take a look at this script in action on the command line:

```
[jmadar@ip-172-31-18-4 1280]$ ./q6.sh
content-type: text/plain

You need to provide an actor
[jmadar@ip-172-31-18-4 1280]$ ./q6.sh 'John Smith'
content-type: text/plain

There are 98 with the name John Smith, we expect only one.
[jmadar@ip-172-31-18-4 1280]$ ./q6.sh 'Henry Cavil'
content-type: text/plain

Actor Henry Cavil is known for the following titles:
   Man of Steel
   Justice League
   Batman v Superman: Dawn of Justice
   Mission: Impossible - Fallout
[jmadar@ip-172-31-18-4 1280]$
```

Make sure this script can be called via your web server. You can test it at http://learn.operatoroverload.com/~jmadar/1280/q6.sh

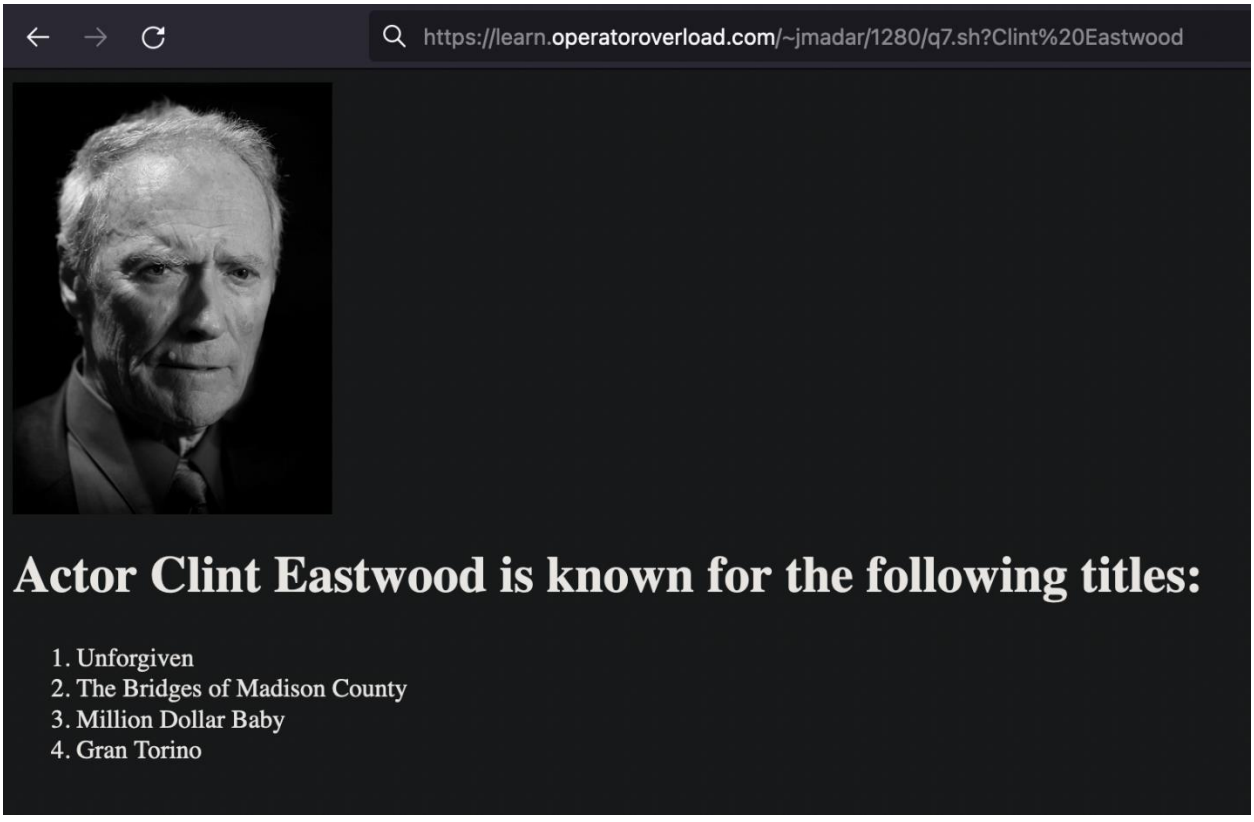
Question 7

WARNING: This question is a lot of work but worth very little compared to the rest of the assignment. It may not be worth your time and is only here to provide additional challenges for those who are ready for it. Don't feel bad if you want to skip this!!

q6.sh is a lot of fun, and there are many enhancements that you can make. For this question, first make a copy of your q6.sh and call it q7.sh, then perform the following:

Change the content-type to text/html, and format the output in HTML, along with including the actor's image in the output. HINT: noticed that the
url https://www.imdb.com/name/nm0147147/ retrieves information about the actor by it's id.

Here's a running version of q7.sh.



Running from the command line:

```
jmadar@ip-172-31-2-248:~/public_html/1280$ ./q7.sh "Clint Eastwood"
content-type: text/html

<img src='https://m.media-amazon.com/images/M/MV5BMTg3MDc0MjY0OV5BMl5BanBnXkFtZTcwNzU1MDAxOA@@._V1_.jpg
gallery' width=200>
<h1>Actor Clint Eastwood is known for the following titles:</h1>
<ol>
<li>Unforgiven</li>
<li>The Bridges of Madison County</li>
<li>Million Dollar Baby</li>
<li>Gran Torino</li>
</ol>
jmadar@ip-172-31-2-248:~/public_html/1280$
```

You can test it using this url: http://learn.operatoroverload.com/~jmadar/1280/q7.sh

Hand-in

Test each script in your browser, and capture the screen. The screen shot needs to show the URL and the results. Name each screen shot q[1-7].jpg (or jpeg)

At the end of the lab, you will seven .sh files and .jpg:

- **q[1-7].sh**
- **q[1-7].jpg**

zip all files into one single file called <lastname>_lab03.zip. Upload it to D2L.

Note: If you do the work in AWS, please refer to this thread to learn how to download the files from the instance. https://stackoverflow.com/questions/9441008/how-can-i-download-a-file-from-ec2