

**</> HTTP Requests and Responses**

Answer the following questions about the HTTP request and response process.

1. What type of architecture does the HTTP request and response process occur in?

**Answer: -** Client-Server Architecture and a stateless request/response protocol that operates by exchanging messages across a reliable TCP/IP connection.

2. What are the different parts of an HTTP request?

**Answer: -** Well in general an HTTP request is divided into three parts – 1. a request line, 2. a request header,

3. an optional request body.

3. Which part of an HTTP request is optional?

**Answer: -** The request body is optional, and body request may contain information such as login credentials (username & password) used for authentication, and I believe the body request is quite important because it represents different types of content.

4. What are the three parts of an HTTP response?

**Answer: -** An HTTP response is divided into three parts – 1. Status line, 2. response header, 3. response body.

5. Which number class of status codes represents errors?

**Answer: -** Well, the 400 codes indicate client errors, meaning the client sent an improperly formatted request, and 500 codes indicate server errors, meaning the server application failed somehow.

6. What are the two most common request methods that a security professional will encounter?

**Answer: -** GET & POST.

**GET requests: -** Ask for the data from a server to retrieve the data and display the data on webpage.

**POST request: -** Send data to a specified resource for example if someone want to login into website account, the client sends a POST request that contains credentials for logging in or if someone want to comment on webpage it’s sends as a POST request.

7. Which type of HTTP request method is used for sending data?

**Answer: -** The POST method.

8. Which part of an HTTP request contains the data being sent to the server?

**Answer: -** The request body and this body part contain data from login credentials or files data that need to be upload on server.

9. In which part of an HTTP response does the browser receive the web code to generate and style a web page?

**Answer: -** The response body, this contains the actual web codes and style of a web page.



**</> Using curl**

Answer the following questions about `curl`:

10. What are the advantages of using `curl` over the browser?

**Answer: -** Well, curl is way too quickly to test HTTP request, and curl have the ability to be flexible and complete complex tasks for example curl can do things like user authentication, HTTP post, SSL connections, proxy support, FTP uploads and can also do simple things with curl like download web pages and web images, Also this includes testing web page data security configurations and testing how secure the servers are that means not to leak important data and verifying that the servers only reply to specific requests and checking for mitigating vulnerabilities.

11. Which `curl` option is used to change the request method?

**Answer: -** To change the method into something else by using the curl -X.

12. Which `curl` option is used to set request headers?

**Answer: -** Curl- H option**,** this curl command has the ability to add an additional HTTP request header to requests, use the curl -H option and set the header name and value in enclosed quotes.

13. Which `curl` option is used to view the response header?

**Answer: -** curl -I example.com, this uses the `-I` flag to view the response headers only.

14. Which request method might an attacker use to figure out which HTTP requests an HTTP server will accept?

**Answer: -** Curl -v, it’s means "verbose mode please" and will make curl show its request, the response headers and additional informational details about the transfer performed.

**</> Sessions and Cookies**

Recall that HTTP servers need to be able to recognize clients from one another. They do this through sessions and cookies.

Answer the following questions about sessions and cookies:

15. Which response header sends a cookie to the client?

```HTTP

HTTP/1.1 200 OK

Content-type: text/html

Set-Cookie: cart=Bob

```

**Answer: -** Set-Cookie: cart=Bob, the Set-Cookie HTTP response header is used to send a cookie from the server to the user agent, so the user agent can send it back to the server later.



16. Which request header will continue the client's session?

```HTTP

GET /cart HTTP/1.1

Host: www.example.org

Cookie: cart=Bob

```

**Answer: -** <GET /cart HTTP/1.1>, the HTTP keep-alive header maintains a connection between a client and server, reducing the time needed to serve files.

**</> Example HTTP Requests and Responses**

Look through the following example HTTP request and response and answer the following questions:

\*\*HTTP Request\*\*

```HTTP

POST /login.php HTTP/1.1

Host: example.com

Accept-Encoding: gzip, deflate, br

Connection: keep-alive

Content-Type: application/x-www-form-urlencoded

Content-Length: 34

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.132 Mobile Safari/537.36

username=Barbara&password=password

```

17. What is the request method?

**Answer: -** The request method is `POST`. (POST /login.php HTTP/1.1)

18. Which header expresses the client's preference for an encrypted response?

**Answer: -** **Upgrade-Insecure-Requests request: 1** header expresses the client's preference for an encrypted response, it’s sending a signal to the server expressing the client's preference for an encrypted and authenticated response, and that it can successfully handle the upgrade-insecure-requests CSP directive.

19. Does the request have a user session associated with it?

**Answer: -** No, there is no session associated with it.

20. What kind of data is being sent from this request body?

**Answer: -** Login credential was sent (**username=Barbara&password=password).**



\*\*HTTP Response\*\*

```HTTP

HTTP/1.1 200 OK

Date: Mon, 16 Mar 2020 17:05:43 GMT

Last-Modified: Sat, 01 Feb 2020 00:00:00 GMT

Content-Encoding: gzip

Expires: Fri, 01 May 2020 00:00:00 GMT

Server: Apache

Set-Cookie: SessionID=5

Content-Type: text/html; charset=UTF-8

Strict-Transport-Security: max-age=31536000; includeSubDomains

X-Content-Type: NoSniff

X-Frame-Options: DENY

X-XSS-Protection: 1; mode=block

[page content]

```

21. What is the response status code?

**Answer: -** The status cod is 200.

22. What web server is handling this HTTP response?

**Answer: -** The web server is handling **Apache** server.

23. Does this response have a user session associated to it?

**Answer: -** Yes, this response have a user session id (**Set-Cookie: SessionID=5**).

24. What kind of content is likely to be in the [page content] response body?

**Answer: -** Content-Type: text/html; charset=UTF-8.

25. If your class covered security headers, what security request headers have been included?

**Answer: -** Strict-Transport-Security: max-age=31536000; includeSubDomains.



**</> Monoliths and Microservices**

Answer the following questions about monoliths and microservices:

26. What are the individual components of microservices called?

**Answer: -** Each independent machine is a component that executes one primary function or service. Microservices application components into their own machines, such as a jump-box or DVWA server and I used this service in my project-1. A service front-end server is really just an operating system, running the minimum software that was configured to serve one main purpose, such as handle HTTP requests and responses. The front-end server, back-end server, and database are the basic components of a typical web application and microservices application components into their own machines.

27. What is a service that writes to a database and communicates to other services?

**Answer: -** The back-end server, and an Application Programming Interface (API) will write and read corresponding data and from a data bases and communicate to the other services within a network or web application.

28. What type of underlying technology allows for microservices to become scalable and have redundancy?

**Answer: -** Well, to ensure scalability and performance microservices need to process tasks efficiently and in order to do this Linux the standard underlying operating system used for web applications however this need to be using it with docker containers, and both need to have concurrency and partitioning, concurrency requires that the service can’t have one single process that does all of the work and that process will pick up one task at a time, complete the steps in a specific order, and then move on to the next, which is a relatively inefficient way to process tasks and instead of architecting service to use a single process, introduce concurrency so that each task is broken up into smaller pieces.

**</> Deploying and Testing a Container Set**

Answer the following questions about multi-container deployment:

29. What tool can be used to deploy multiple containers at once?

**Answer: -** Docker Compose is a tool for defining and running multi-container Docker applications, in Compose we can use a YAML file to configure application's services.

30. What kind of file format is required for us to deploy a container set?

**Answer: -** A YAML file provides a concise format for specifying the instance settings and docker Compose YAML to define the containers their networking configurations and where to copy files from host machine into specified container.



**</> Databases**

31. Which type of SQL query would we use to see all of the information within a table called `customers`?

**Answer: -** Select \* from Customers.

32. Which type of SQL query would we use to enter new data into a table? (You don't need a full query, just the first part of the statement.)

**Answer: -** The INSERT INTO query is used to add new rows of data to a table in the database.

33. Why would we never run `DELETE FROM <table-name>;` by itself?

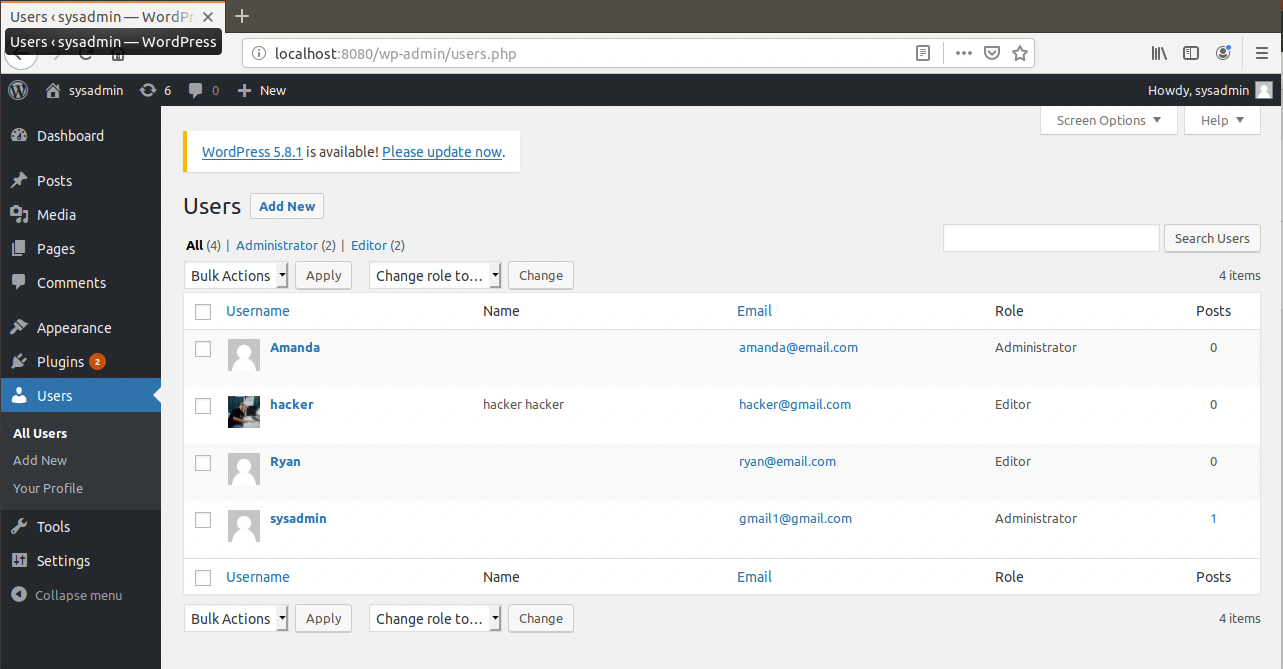
**Answer: -** As it would delete entire table and for example need to specify and run like this **`DELETE FROM employees WHERE email='example@gmail.com;`** and check the query again and then go ahead.

**</> Bonus Challenge Overview: The Cookie Jar**

**</> Step 1: Set Up**

Create two new users: Amanda and Ryan.

**Answer: -**

****



**</> Step 2: Baselining**

For these "baselining" steps, you'll want to log into two different types of accounts to see how the WordPress site looks at the `localhost:8080/wp-admin/users.php` page.

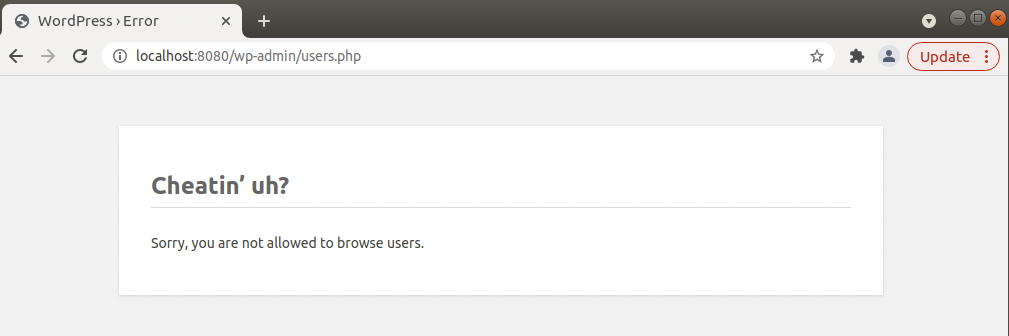
We want to see how the Users page looks from the perspective of an administrator, vs. a regular user.

1. Using your browser, log into your WordPress site as your sysadmin account and navigate to `localhost:8080/wp-admin/users.php`, where we previously created the user Ryan. Examine this page briefly. Log out.

2. Using your browser, log into your Ryan account and attempt to navigate to `localhost:8080/wp-admin/index.php`. Note the wording on your Dashboard.

3. Attempt to navigate to `localhost:8080/wp-admin/users.php`. Note what you see now.

**Answer: -**



**</> Step 3: Using Forms and a Cookie Jar**

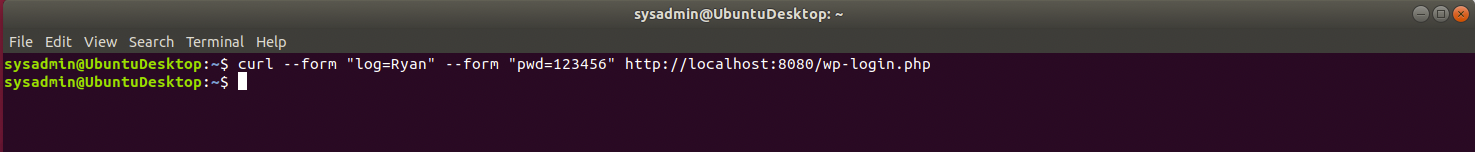
Navigate to `~/Documents` in a terminal to save your cookies.

1. Construct a `curl` request that enters two forms: `"log={username}"` and `"pwd={password}"` and goes to `http://localhost:8080/wp-login.php`. Enter Ryan's credentials where there are placeholders.

- \*\*Question:\*\* Did you see any obvious confirmation of a login? (Y/N)

**Answer: -** I didn’t see or got any obvious notification of login.

**Command: - curl --form "log=Ryan" --form "pwd=123456"** [**http://localhost:8080/wp-login.php**](http://localhost:8080/wp-login.php%60)





2. Construct the same `curl` request, but this time add the option and path to save your cookie: `--cookie-jar ./ryancookies.txt`. This option tells `curl` to save the cookies to the `ryancookies.txt` text file.

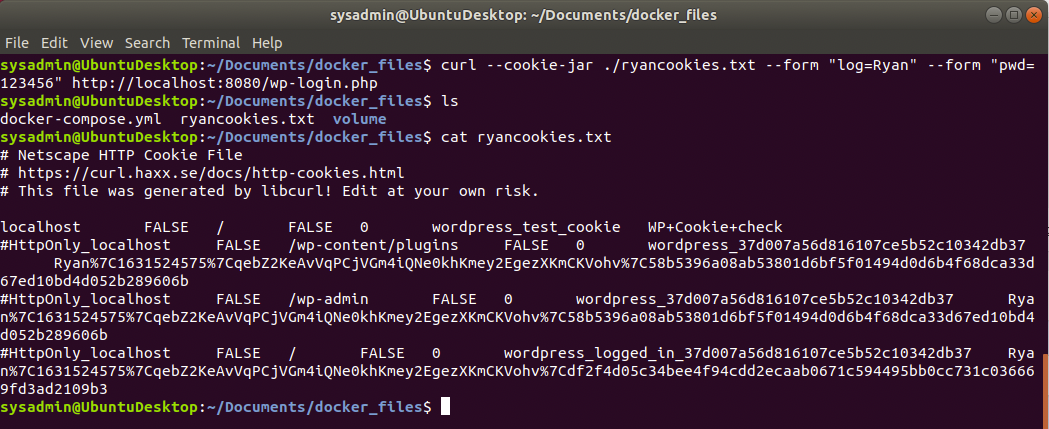
**Answer: -**

Command: - **curl --cookie-jar ./ryancookies.txt --form "log=Ryan" --form "pwd=123456" http://localhost:8080/wp-login.php**

3. Read the contents of the `ryancookies.txt` file.

- \*\*Question:\*\* How many items exist in this file?

**Answer: -** Three cookies exist in the `ryancookies.txt` file.



Note that each one of these is a cookie that was granted to Ryan after logging in.

**</> Step 4: Log in Using Cookies**

1. Craft a new `curl` command that now uses the `--cookie` option, followed by the path to your cookies file. For the URL, use `http://localhost:8080/wp-admin/index.php`.

- \*\*Question:\*\* Is it obvious that we can access the Dashboard? (Y/N)

**Answer: -** At the first look I can’t see access the Dashboard but once I do grep command, I can see the access the Dashboard.

**Command: -** **curl --cookie ./ryancookies.txt http://localhost:8080/wp-admin/index.php**

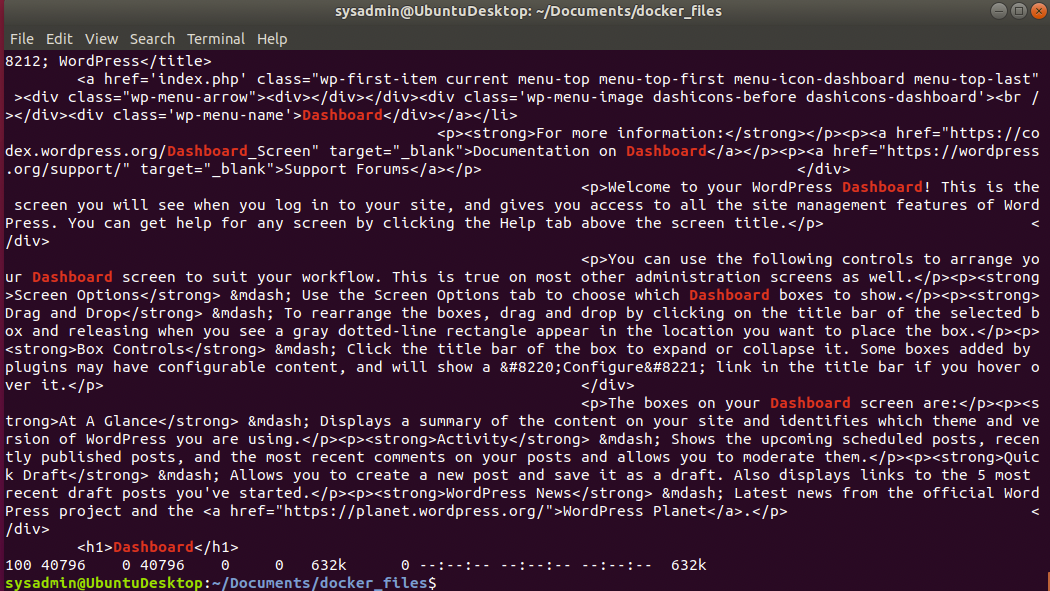


2. Press the up arrow on your keyboard to run the same command, but this time, pipe `| grep Dashboard` to the end of your command to return all instances of the word `Dashboard` on the page.

- \*\*Question:\*\* Look through the output where `Dashboard` is highlighted. Does any of the wording on this page seem familiar? (Y/N) If so, you should be successfully logged in to your Editor's dashboard.

**Answer: -** Yes, after adding the grep pipe in to below command, I can see all occurrences of the word Dashboard in the returned response body, and also it’s showing us a successfully returned index.php session.

**Command: -** **curl --cookie ./ryancookies.txt http://localhost:8080/wp-admin/index.php | grep Dashboard**



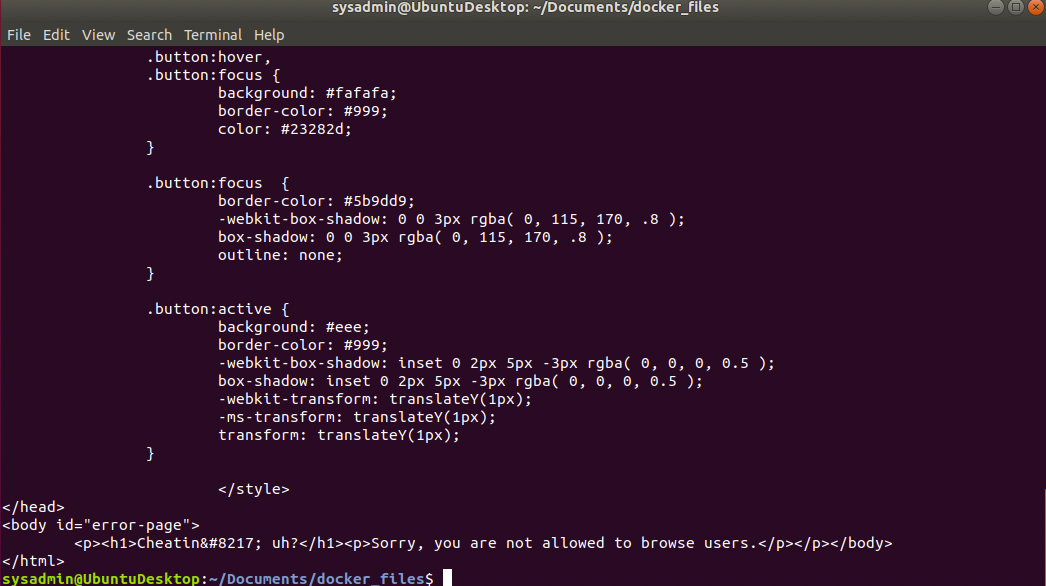
**</>** **Step 5: Test the Users.php Page**

1. Finally, write a `curl` command using the same `--cookie ryancookies.txt` option, but attempt to access `http://localhost:8080/wp-admin/users.php`.

- \*\*Question:\*\* What happens this time?

**Answer: -** I can see this message in terminal “Sorry, you are not allowed to browse users”.

**Command: -** **curl --cookie ./ryancookies.txt http://localhost:8080/wp-admin/users.php**



Thank you,

Aziz Somani