Lab

Web Security Basics

Review Questions

1. What is the CIA triad?
2. What is the difference between Http Cookies and Http Headers
3. What is CORS?
4. What is the difference between SSL and TLS?
5. Explain the purpose of the following HTTP Security Headers : **Content Security Policy(CSP)**, **X-XSS-Protection**, **HTTP Strict Transport Security (HSTS)**, **X-Frame-Options** ?
6. How to prevent ransomware attacks ?

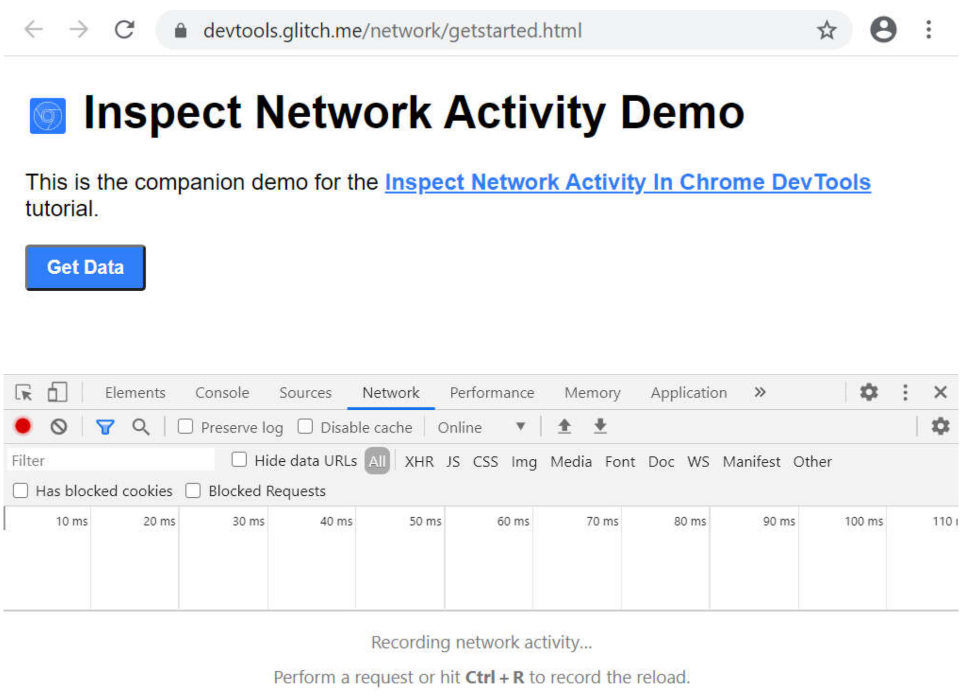
Hands-on Lab: Inspecting and Hardening HTTP

Activity 1 – Inspecting HTTP queries with the Google Chrome DevTools

All web browsers offer the capability to inspect the HTTP network activity and to discover the details related to the queries. We are practicing in this activity some of the most commonly-used DevTools features related to inspecting a page's network activity.

- **Launch the Chrome Web Browser and open the DevTools**

  - Under Chrome, open the Get Started Demo [https://devtools.glitch.me/network/getstarted.html]
  - Open DevTools by pressing **F12** or **Control+Shift+J**. You might prefer to dock DevTools to the bottom of your window. Click the **Network** tab. The Network panel opens.



devtools.glitch.me/network/getstarted.html

Inspect Network Activity Demo

This is the companion demo for the Inspect Network Activity In Chrome DevTools tutorial.

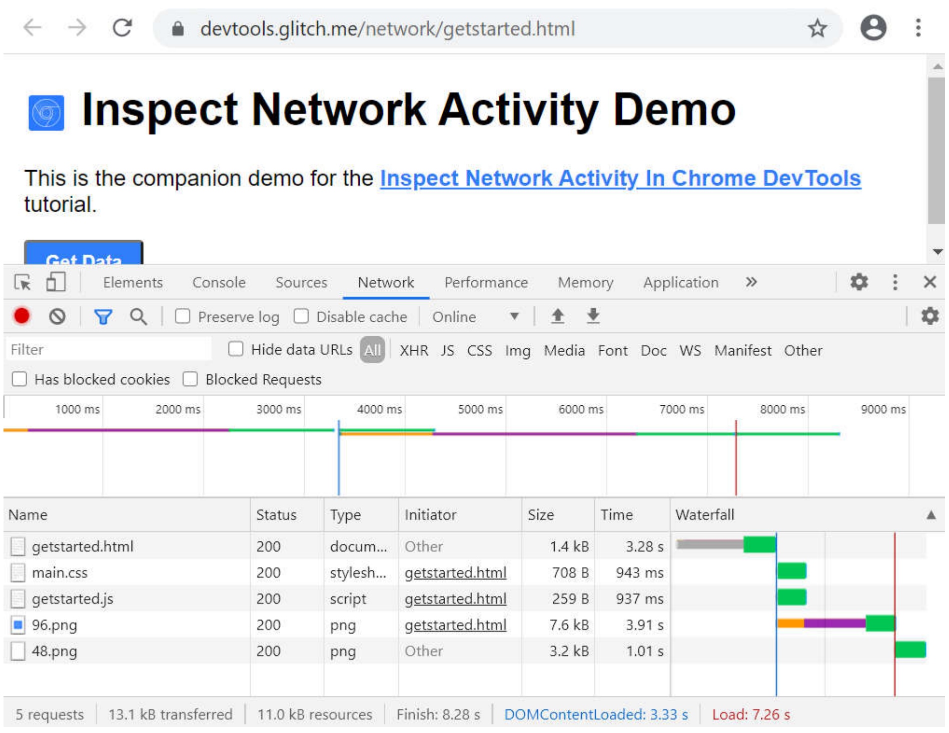
Recording network activity...

Perform a request or hit **Ctrl** + **R** to record the reload.

- **Inspecting the Network activity**

  - To view the network activity that a page causes:

    1. Reload the page. The Network panel logs all network activity in the **Network Log**.



Inspect Network Activity Demo

This is the companion demo for the Inspect Network Activity In Chrome DevTools tutorial.

Name	Status	Type	Initiator	Size	Time	Waterfall
getstarted.html	200	docum...	Other	1.4 kB	3.28 s	
main.css	200	stylesh...	getstarted.html	708 B	943 ms	
getstarted.js	200	script	getstarted.html	259 B	937 ms	
96.png	200	png	getstarted.html	7.6 kB	3.91 s	
48.png	200	png	Other	3.2 kB	1.01 s	

5 requests | 13.1 kB transferred | 11.0 kB resources | Finish: 8.28 s | DOMContentLoaded: 3.33 s | Load: 7.26 s

Each row of the **Network Log** represents a resource. By default the resources are listed chronologically. The top resource is usually the main HTML document. The bottom resource is whatever was requested last.

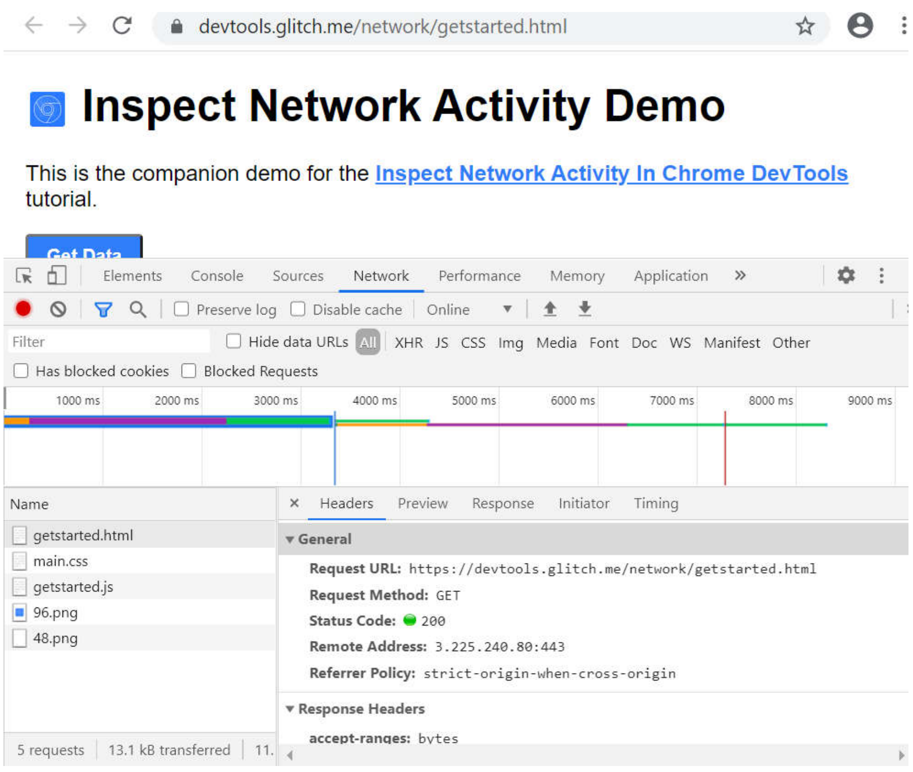
So long as you've got DevTools open, it will record network activity in the Network Log. To demonstrate this, first look at the bottom of the **Network Log** and make a mental note of the last activity.

2. Now, click the **Get Data button** in the demo. Look at the bottom of the Network Log again. There's a new resource called **getstarted.json**. Clicking the Get Data button caused the page to request this file.

- **Inspecting HTTP Requests details**

  - Click a resource to learn more information about it.

  1. Click **getstarted.html**. The Headers tab is shown. Use this tab to inspect HTTP headers.



Inspect Network Activity Demo

This is the companion demo for the Inspect Network Activity In Chrome DevTools tutorial.

Name
getstarted.html
main.css
getstarted.js
96.png
48.png

5 requests | 13.1 kB transferred | 11.

▾ General

Request URL: `https://devtools.glitch.me/network/getstarted.html`
Request Method: `GET`
Status Code: ● `200`
Remote Address: `3.225.240.80:443`
Referrer Policy: `strict-origin-when-cross-origin`

▾ Response Headers

accept-ranges: `bytes`

2. Click the **Preview tab**. A basic rendering of the HTML is shown. This tab is helpful when an API returns an error code in HTML and it's easier to read the rendered HTML than the HTML source code, or when inspecting images.

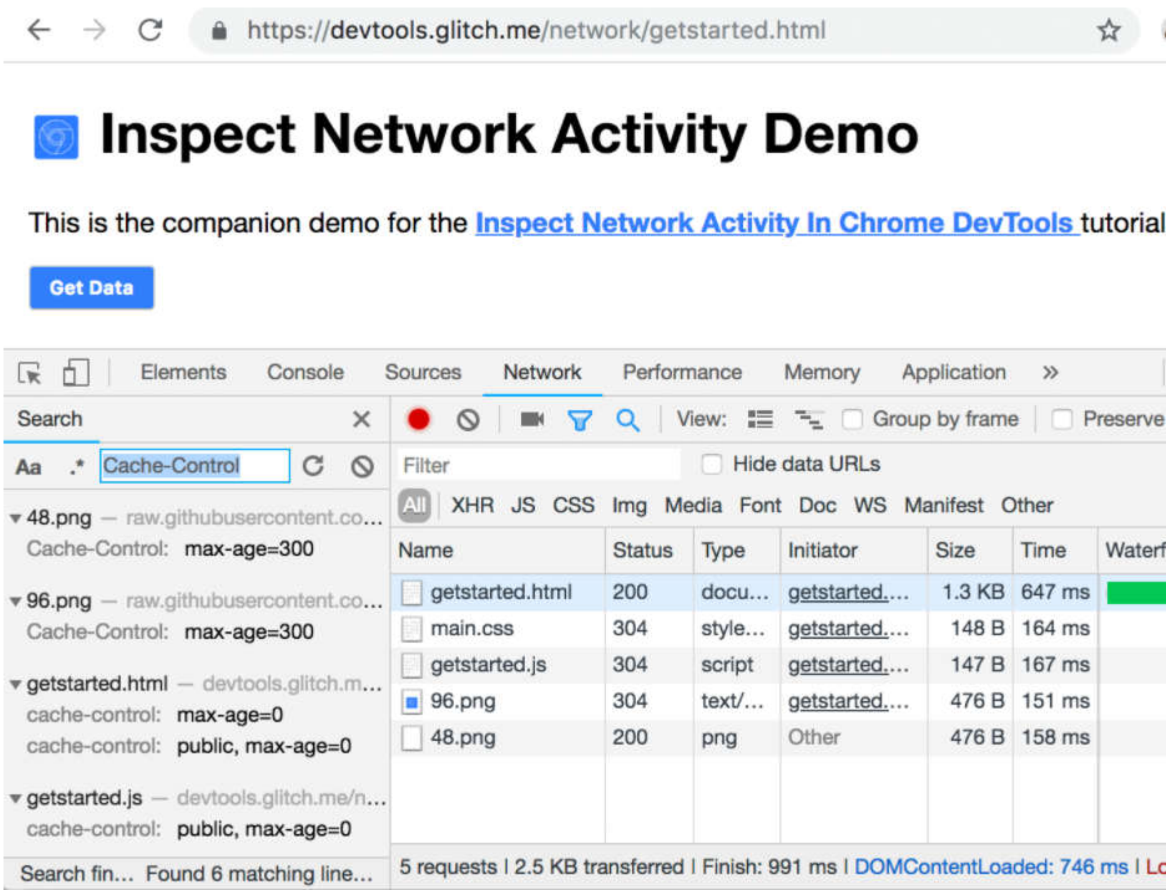
3. Click the **Timing tab**. A breakdown of the network activity for this resource is shown.

- **Search network headers and responses**

  Use the **Search pane** when you need to search the HTTP headers and responses of all resources for a certain string or regular expression. For example, suppose you want to check if your resources are using reasonable cache policies.

  1. Click Search . The Search pane opens to the left of the Network log.

  2. Type `Cache-Control` and press `Enter`. The Search pane lists all instances of `Cache-Control` that it finds in resource headers or content.



https://devtools.glitch.me/network/getstarted.html

Inspect Network Activity Demo

This is the companion demo for the Inspect Network Activity In Chrome DevTools tutorial

Cache-Control

- 48.png — raw.githubusercontent.co...
  - Cache-Control: max-age=300
- 96.png — raw.githubusercontent.co...
  - Cache-Control: max-age=300
- getstarted.html — devtools.glitch.m...
  - cache-control: max-age=0
  - cache-control: public, max-age=0
- getstarted.js — devtools.glitch.me/n...
  - cache-control: public, max-age=0

Search fin... Found 6 matching line...

Name	Status	Type	Initiator	Size	Time	Waterf
getstarted.html	200	docu...	getstarted....	1.3 KB	647 ms	🟩
main.css	304	style...	getstarted....	148 B	164 ms	
getstarted.js	304	script	getstarted....	147 B	167 ms	
96.png	304	text/...	getstarted....	476 B	151 ms	
48.png	200	png	Other	476 B	158 ms	

5 requests | 2.5 KB transferred | Finish: 991 ms | DOMContentLoaded: 746 ms | Lo

3. Click a result to view it. If the query was found in a header, the Headers tab opens. If the query was found in content, the Response tab opens.

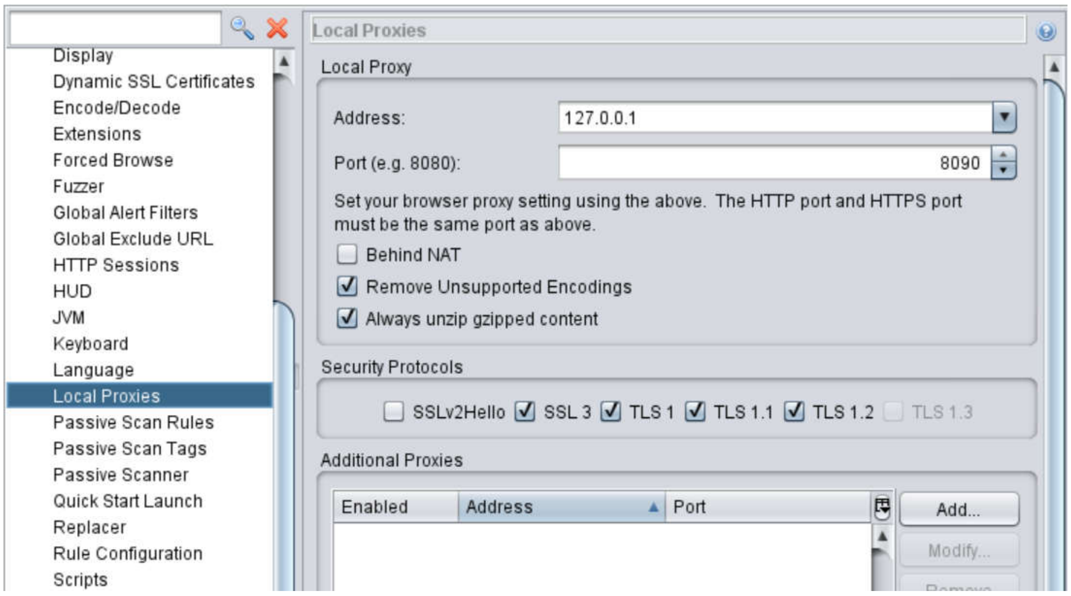
Activity 2 – Inspecting HTTP queries with the OWASP ZAP Proxy Browser

A proxy is some forwarder application that connects your http client to backend resources. HTTP clients can be browsers, or applications like curl, SOAP UI, Postman, etc. HTTP Proxies receive requests from a client and relay them. They also typically record them. They act as a man-in-the-middle.

With OWASP ZAP you can record traffic, inspect traffic, modify requests and response from and to your browser, and get reports on a range of known vulnerabilities that are detected by ZAP through the inspection of the traffic.

- **Setting up ZAP 2.9.0**

- Navigate to the training tools folder **Tools_Web_Security_Training**, and then to the sub-folder **Owasp_ZAP**. Unzip the package included in the **ZAP_2.9.0_Crossplatform.zip**. ZAP requires Java
- Start ZAP by lauching the **zap.bat** batch file included in the folder where you have uncompressed the ZAP package. You should be greeted with a page that looks like this, asking if you want to persist the ZAP Session. Select no for the time being (as we're just trying to set the app and poke around) and hit "Start".
- Configure the proxy to use a free port, e.g. **8090**. This can be changed via the **Tools -> Options -> Local Proxies screen**. By default ZAP uses an Address of 'localhost' and a Port of '8080'.



Address: 127.0.0.1

Port (e.g. 8080): 8090

Set your browser proxy setting using the above. The HTTP port and HTTPS port must be the same port as above.

☐ Behind NAT
☑ Remove Unsupported Encodings
☑ Always unzip gzipped content

☐ SSLv2Hello ☑ SSL 3 ☑ TLS 1 ☑ TLS 1.1 ☑ TLS 1.2 ☐ TLS 1.3

- In this activity, we are going to use the WebGoat web application. It is a deliberately insecure web application maintained by the OWASP organization. The easiest way to start WebGoat is tu use a Docker is to use the all-in-one docker container which is publicly available eon dokerhub.com. This is a Docker image that has WebGoat and WebWolf running inside.

```
docker run -p 8080:8080 -p 9090:9090 -e TZ=Europe/Amsterdam webgoat/goatandwolf:v8.1.0
```

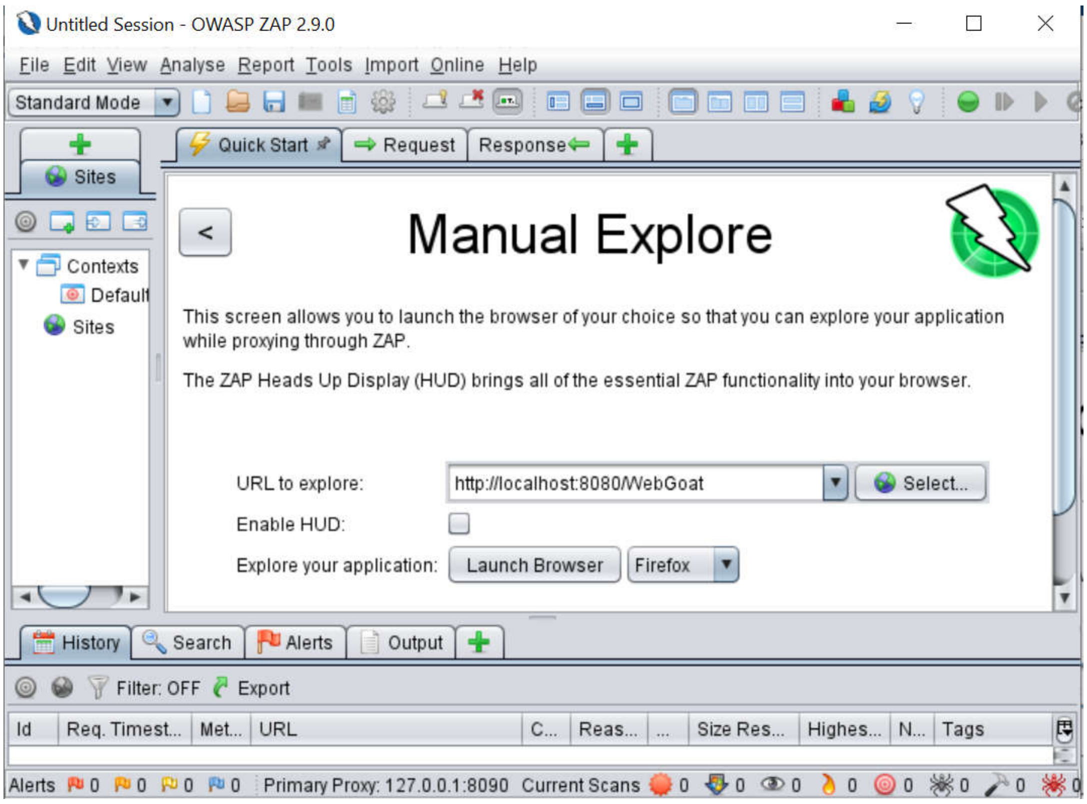
  WebGoat will be located at: `http://127.0.0.1:8080/WebGoat`

  In order to stop webgoat, proceed as follows

  (i)identify the container id using the command: `docker ps`,

  (ii)then remove it it the command `docker rm id`. You can type in just the first letters of the id.

- Back to Zap, choose **Manual explore** from the welcome page and initialize the URL to explore to the WebGoat URL and then launch the browser of your choice. When WebGoat loads, register a new user.



Manual Explore

This screen allows you to launch the browser of your choice so that you can explore your application while proxying through ZAP.

The ZAP Heads Up Display (HUD) brings all of the essential ZAP functionality into your browser.

URL to explore: http://localhost:8080/WebGoat Select...

Enable HUD: ☐

Explore your application: Launch Browser Firefox

Alerts 0 0 0 0 Primary Proxy: 127.0.0.1:8090 Current Scans 0 0 0 0 0 0 0 0

- From the left sidebar menu, select **General**, then the **HTTP Proxies** lesson. Implement the instructions of the exercises of this lesson and focus particularly on these exercises:
  - Exercise 5: **Filter requests in history panel**
  - Exercise 6: **Configure a breakpoint filter / Intercept and modify a request**
  - Exercise 7: **Use the "Edit and resend" functionality in ZAP**

Activity 3 – Inspecting and Fixing HTTP Security Headers

There are several tools that scan web sites for security headers. Online tools cannot be used to scan local sites that are being developped.

- **Using an online Scanner:** https://securityheaders.com/

  Security Headers was created by me, Scott Helme, Security Consultant and blogger based in the UK. This tool analyzes the HTTP response headers and adds a rating system to the results. The HTTP response headers that this site analyses provide huge levels of protection and it's important that sites deploy them.

  - Try to audit some known sites like https://www.facebook.com, https://www.google.com, and https://webgate.ec.europa.eu/.

- **Installing a local scanner**

  shchesk is a basic tool to check security headers. It is written in Python and available at GitHub https://github.com/meliot/shcheck.
  First clone it from Github using the following command:

  ```
  git clone https://github.com/meliot/shcheck.git
  ```

  Next, cd into shcheck folder and build your image using something like this:

  ```
  docker build -t shcheck .
  ```

  Then simply run the local website you want to check headers on:

  ```
  docker run -d -p 4200:80  mromdhani/demowebapp
  ```

  Run the shcheck check. **host.docker.internal** is address used by the container to access to the host.

  ```
  docker run -it --rm shcheck http://host.docker.internal:4200
  ```

You will get the following result. Many Security headers are missing:

```
=======================================================
 > shcheck.py - meliot.................................
-------------------------------------------------------
 Simple tool to check security headers on a webserver
=======================================================

[*] Analyzing headers of http://host.docker.internal:4300
[*] Effective URL: http://host.docker.internal:4300
[!] Missing security header: X-XSS-Protection
[!] Missing security header: X-Frame-Options
[!] Missing security header: X-Content-Type-Options
[!] Missing security header: Content-Security-Policy
[!] Missing security header: X-Permitted-Cross-Domain-Policies
[!] Missing security header: Referrer-Policy
-------------------------------------------------------
[!] Headers analyzed for http://host.docker.internal:4300
[+] There are 0 security headers
[-] There are not 6 security headers
```

- **Fixing the header issues**

The application we have audited is an nginx application. Let's add a new Docker layer containing additional headers activation.

- Create a new file using VSCode and name it Dockerfile (D in capital). Initialize it with this content:

```
FROM mromdhani/demowebapp
COPY ./nginx.conf /etc/nginx/conf.d/default.conf
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

- Create a new file named nginx.cong having the following content:

```
server {
    listen  80;
    root    /usr/share/nginx/html;
    include /etc/nginx/mime.types;

    location / {
        try_files $uri $uri/ /index.html;
    }

    gzip            on;
    gzip_vary       on;
    gzip_comp_level 5;

    add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-
inline' 'unsafe-eval'; connect-src 'self'; img-src 'self' data:; style-src 'self' 'unsafe-
inline' fonts.googleapis.com; font-src 'self' fonts.gstatic.com;" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-Frame-Options SAMEORIGIN always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header X-Permitted-Cross-Domain-Policies none always;
    add_header Referrer-Policy 'origin';
}
```

- Create the new image using the following docker command :

```
Docker build -t mromdhani/demowebapp-fixed .
```

Then simply run the local website you want to check headers on:

```
docker run -d -p 4300:80  mromdhani/demowebapp-fixed
```

Run the shcheck check. **host.docker.internal** is address used by the container to access to the host.

```
docker run -it --rm shcheck http://host.docker.internal:4300
```

The security headers are now there.

```
=======================================================
 > shcheck.py - meliot.................................
```

```
------------------------------------------------------
 Simple tool to check security headers on a webserver
======================================================

[*] Analyzing headers of http://host.docker.internal:4400
[*] Effective URL: http://host.docker.internal:4400
[*] Header X-XSS-Protection is present! (Value: 1; mode=block)
[*] Header X-Frame-Options is present! (Value: SAMEORIGIN)
[*] Header X-Content-Type-Options is present! (Value: nosniff)
[*] Header Content-Security-Policy is present! (Value: default-src 'self'; script-src
'self' 'unsafe-inline' 'unsafe-eval'; connect-src 'self'; img-src 'self' data:; style-
src 'self' 'unsafe-inline' fonts.googleapis.com; font-src 'self' fonts.gstatic.com;)
[*] Header X-Permitted-Cross-Domain-Policies is present! (Value: none)
[*] Header Referrer-Policy is present! (Value: origin)
------------------------------------------------------
[!] Headers analyzed for http://host.docker.internal:4400
[+] There are 6 security headers
[-] There are not 0 security headers
```

Answers to Review Questions

7. What is the CIA triad?

   CIA stands for Confidentiality, Integrity, and Availability. These are the three basic information security concepts. If we relate these concepts with the people who use that information, then it will be authentication, authorization, and non-repudiation.

   Confidentiality. When information is read or copied by someone not authorized to do so, then it will be "loss of confidentiality". For sensitive information, confidentiality is a very important criterion.

   Integrity. Information can be corrupted or manipulated if it's available on an insecure network, and is referred to as "loss of integrity." This means that unauthorized changes are made to information, whether by human error or intentional tampering. Integrity is particularly important for critical safety and financial data used for activities such as electronic funds transfers, air traffic control, and financial accounting.

   Availability. Information can be erased or become inaccessible, resulting in "loss of availability." This means that people who are authorized to get information are restricted from accessing. Availability is often the most important attribute in service-oriented businesses that depend on information.

8. What is the difference between Http Cookies and Http Headers

   An *HTTP cookie* (web cookie, browser cookie) is a small piece of data that a server sends to the user's web browser. The browser may store it and send it back with the next request to the same server. Typically, it's used to tell if two requests came from the same browser — keeping a user logged-in, for example. It remembers stateful information for the stateless HTTP protocol.

   HTTP headers let the client and the server pass additional information with an HTTP request or response. An HTTP header consists of its case-insensitive name followed by a colon (:), then by its value.

9. What is CORS?

   Cross-Origin Resource Sharing (CORS) is a mechanism that uses additional HTTP headers to tell browsers to give a web application running at one origin, access to selected resources from a different origin.

   For security reasons, browsers restrict cross-origin HTTP requests initiated from scripts. For example, XMLHttpRequest and the Fetch API follow the Same-Origin Policy (SOP). This means that a web application using those APIs can only request resources from the same origin the application was loaded from, unless the response from other origins includes the right CORS headers.

10. What is the difference between SSL and TLS?

    SSL refers to Secure Sockets Layer whereas TLS refers to Transport Layer Security.

    How similar both are? SSL and TLS are cryptographic protocols that authenticate data transfer between servers, systems, applications and users. For example, a cryptographic protocol encrypts the data that is exchanged between a web server and a user.

    SSL was a first of its kind of cryptographic protocol. TLS on the other hand, was a recent upgraded version of SSL.

11. Explain the purpose of the following HTTP Security Headers : **Content Security Policy(CSP)**, **X-XSS-Protection**, **HTTP Strict Transport Security (HSTS)**,
    X-Frame-Options ?

    Content Security Policy(CSP): This header helps prevent attacks such as Cross Site Scripting (XSS) and other code injection attacks by defining content sources which are approved and thus allowing the browser to load them.

    X-XSS-Protection: This header is designed to enable the cross-site scripting (XSS) filter built into modern web browsers. This is usually enabled by default, but using it will enforce it. It is supported by Internet Explorer 8+, Chrome, and Safari.

    HTTP Strict Transport Security (HSTS): This header is a security enhancement that restricts web browsers to access web servers solely over HTTPS. This ensures the connection cannot be establish through an insecure HTTP connection which could be susceptible to attacks.

X-Frame-Options: header provides clickjacking protection by not allowing iframes to load on your website. It is supported by IE 8+, Chrome 4.1+, Firefox 3.6.9+, Opera 10.5+, Safari 4+.

12. How to prevent ransomware attacks?

These are tips on how to prevent ransomware attacks

- Never click on unverified links
- Do not open untrusted email attachments
- Only download from sites you trust
- Avoid giving out personal data
- Use mail server content scanning and filtering
- Never use unfamiliar USBs
- Keep your software and operating system updated
- Backup your data