# Homework 5 – Content Security Policy

## Lab Information

### Due Date:

Homework 5 Dropbox Deadline

### Objectives/Goal:

In this homework, we will be implementing one of the newest client-side security controls –content security policy. You will find why many major organizations tend not to employ the control on existing pages. All work will be done using the provided source code.

### Deliverables:

- The Dockerfile from activity 1
- The Docker file with updated source code from activity 2
- Headers and Database dump demonstrating CSP reporting
- Table of supported Referrer-Policy options
- Screenshot of HSTS working

### Table of Contents:

# Activity 1: Setup Armbook!

You've done it before now you'll do it again, but with a twist. One of the most important aspects we've talked about is the use of HTTPS (TLS) when setting up a webpage. Setup your webserver to serve Armbook over HTTP/2

## Step 1: Get Armbook working!!!!

Alright, this step should be easy. Setup a webserver with PHP and database. Use the source provided to get the application up and running.

## Step 2: Add a pinch of TLS

Now that you have Armbook working properly it's time to add TLS. Because you're hosting this locally you won't be able to use a public Certificate Authority (CA) like let's encrypt. Instead, create your own Certificate Authority and roll your own certificate for Armbook. Make sure to also add your CA to the list of trusted root CAs, to avoid getting a warning.

## Step 3: oooohhhh shiny!

We've learned in class that HTTP/2 will not work on modern browsers without HTTPS support -fortunately, we've just added HTTPS support. Configure your webserver to serve Armbook of HTTP/2!

**Submit a Dockerfile that builds Armbook and serves it using HTTPS and HTTP/2 (You may use a self-signed certificate)**

# Activity 2: Feel for your developer

Often penetration testers will make blanket suggestions or offer broad findings such as 'Missing CSP Headers'. Your goal for this activity is to implement these controls on a small application – Armbook. The source code for this application has been provided as part of the dropbox.

## Step 1: Help a script out

CSP has many controls, some of them are specific to scripts. You must modify the Armbook application to properly/safely support the `script-src`. You may not use `unsafe-inline` or `unsafe-eval` while achieving this goal. JQuery can be added as an exception as desired (because I'm not THAT much of a monster).

## Step 2: Moving onwards

Now that you've taken care of the bulk of the problems add support for `font-src`, `child-src`, `connect-src`, `img-src`, `media-src`, `font-src`, `worker-src`, `object-src`, and `style-src`. If desired you may `default-src` for directives that aren't required.

### Step 3: Other Directives

Now that your application is really cooking with fire, implement form-action, frame-ancestors, and block-all-mixed-content.

**Submit a Dockerfile that builds Armbook and serves it with the required CSP directives.**

## Activity 3: Setup Reporting

CSPs biggest issue is that it tends to be very annoying to tell if any of your changes falsely triggered, particularly on dynamic sites. In order to solve this CSP added a reporting feature.

### Step 1: Setup Reporting

Now that you've setup CSP it's time to configure it to detect issues in the future. CSP supports reporting to a third party site in case of a violation. This can be specified using report-to (or report-uri prior to version 3). Use both of these to configure the where your violation reports will go to.

### Step 2: Create a listener

Now that you've got the configuration down create a page at the location you specified that listens for the policy violation reports and adds it's to a MySQL database.

**Provide your headers CSP header demonstrating that reporting is functional. Also, provide a dump of your database proving that the CSP reporting is working.**

## Activity 4: Referrer-Policy HSTS

Now that we've got CSP working it's time to add a few more headers to get the most out of our modern browser.

### Step 1: Referrer-Policy

CSP used to have a policy called `referrer` but that has been obsoleted in favor of the `Referrer-Policy` header. Test the new header and see if it functions correctly on both IE (IE or Edge) and Chrome. Note that the Referrer-Policy has many different options, you should be able to discuss what each of them does and have a table explaining which are supported on the respective browsers.

```
Referrer-Policy Headers: "no-referrer" / "no-referrer-when-downgrade" /
"strict-origin" / "strict-origin-when-cross-origin" / "same-origin" /
"origin" / "origin-when-cross-origin" / "unsafe-url"
```

**Submit a table breaking down the different options, what they do and if they're supported.**

## Step 2: One step closer to the edge

The last step of this lab is to implement HSTS. This should be pretty simple, configure your webserver to use HSTS

**Provide a screenshot demonstrating that it is working**

## Signoffs

Activity 1.3 – Submit a docker-compose file that builds Armbook and serves it using HTTPS and HTTP/2 – 30%

Activity 2.3 – Submit a docker-compose file that builds Armbook and serves it with the required CSP directives. – 20%

Activity 3.2 – Provide your CSP demonstrating reporting is working and a database dump showing you can write it to MySQL = 20%

Activity 4.1 – Submit a table overviewing the different options, what they do, and if they're supported. – 15%

Activity 4.2 – Provide a screenshot demonstrating HSTS is working – 15%