Challenge URL: /dvwa/vulnerabilities/csp/
Objective: Bypass Content Security Policy (CSP) and execute JavaScript in the page.
Tools needed: Burp Suite (optional)

Did you remember to read this section's README?

The Guide

Discovering the CSP

Okay, here's what we see:

We know from the first link under "More Information" that we can discover the current CSP by viewing the HTTP response header the server sends back to us. We can do this via Burp Suite, and I'd actually recommend that due to the multiple useful features Burp offers. Firefox also allows us to view headers via the Firefox developer console. Since it takes less time to configure, let's pull the developer console up by pressing the keyboard keys `Control+Shift+K` simultaneously. Navigate to the "Network" tab.

It immediately tells us to perform a request, so why don't we input a test URL in the form and see what happens?

We see a bunch of 200 response codes, which helps. Let's double click on the POST request to see what we learn.

In the bottom right corner, we see a response header called "Content-Security-Policy". That's what we need! Examining the contents tells us the full CSP:

```
script-src 'self' https://pastebin.com example.com code.jquery.com https://ssl.google-analytics.com ;
```

This tells us we can load Javascript from the following locations:

- Scripts from within DVWA (Challenge 5, anyone?);
- Anything hosted on the domain pastebin.com;
- Anything hosted on the domain example.com;
- Anything hosted on the domain code.jquery.com; or
- https://ssl.google-analytics.com/.

We could pretty easily upload a Javascript file to DVWA, but we've already done that. Let's try something different. Of the other options, Pastebin allows us to upload any files we want, painlessly and anonymously. Sounds good to me!

Bypassing the CSP

Let's upload some example code to Pastebin.

By way of explanation, I uploaded the code snippet `console.log("CSP Bypass");` to Pastebin. This snippet will make the string "CSP Bypass" visible in our browser's console log, if our attack works. I told Pastebin that I:

1. Uploaded a Javascript file ("Syntax Highlighting");
2. Made the upload an "unlisted" post (only people who know the direct link of the upload will know it exists; done via "Post Exposure"); and
3. Told Pastebin to delete the upload after an hour ("Paste Expiration").

After clicking the "Create New Paste" button, you should get a success screen similar to this:

Click the "Raw" button above the top right corner of the "RAW Paste Data" field.

This removes all the Pastebin-added stuff and give us only what we uploaded. That's what we can use to check if our exploit works!

Let's navigate back to the challenge and open up the developer console again. Enter our raw Pastebin link into the form field. Click the "Include" button and check the console log.

We did it! Our message was successfully logged to the console, and we can verify we see it. Challenge complete!