CONTENT SECURITY POLICY

- ***1.1: CSP policy that you injected***
- ***1.1: screenshot of the burp match and replace rule***
- ***1.2: explanation***

1.1 Fix the nodegoat XSS attack via CSP

Ideally, XSS should be solved by the application developer through context-sensitive output encoding. However, in this exercise we are going to fix an XSS vulnerability using CSP headers only.

We will try fix the attack that you executed in one of the previous exercises. Re-execute the XSS attack against the profile page (Exercise 13: 1.2). Login to http://nodegoat-websec.herokuapp.com/benefits. The XSS payload will now run in your browser.

Since you do not have control over the nodegoat server in this demo exercise, you'll have to inject the CSP header using Burp. **The fact that you have to inject it this way is of course not the way to do it in a real production environment**. However, it's just a simple way to quickly test the impact of a CSP header locally and it helps you get used to the more advanced functionalities of Burp.

Use the match and replace functionality of Burp to inject a CSP header before the response reaches your browser. The browser will think the response came from the server and will enforce the CSP.

Can you fix the XSS vulnerability without breaking the rest of the functionality?

1.2 SRI

Explain in your own words what SRI is and use your favorite search engine to find an example of an attack that occurred in the past which could have been prevented by using SRI.