Workshop tasks

Welcome to Booster and our workshop - thanks for showing up! Just grab one of us if you have a question! We'll assist you as much as we can. Don't sweat it if you don't complete everything, there is way more tasks here than we expect you to complete - do the rest on your way home.. 🙂

0 Basic info

Your goal is to set up an web application on Azure with decent public grades on SSL Labs and securityheaders.io.

We have provided two sample applications:
Reference BAD app (**F rating**): https://**bad**boosterconf.azurewebsites.net/
Reference GOOD app (**A+ rating**): https://boosterconf.azurewebsites.net/

1 Get things ready

You will need a text-editor and a git client. We recommend installing Visual Studio Code and .NET Core in order to build the sample apps locally.

- .NET core - https://www.microsoft.com/net/core
- Git - https://git-scm.com/downloads
- Visual studio code - https://code.visualstudio.com/

- Download our sample apps - open a terminal and clone our git repo:

```
git clone https://github.com/jorgis/boosterconf2017.git
```

Then you will need to provision a test Azure webapp:

- Visit https://azure.microsoft.com/en-us/try/app-service/web/
- Choose .Net Core 1.0, select "*Create*" - log in with one of the available options.
- After provosioning - click "*Extend your trial to 24 hours*"
- Click "*Clone or push with Git*"
- Copy the Azure git remote URL and save it somewhere (also make sure to keep the link to your app)

2 Publish the sample project

Add Azure remote and push (you may have to use --force if you get a warning):

```
git remote add Azure [your-azure-remote-url]
git push Azure master
```

Drink some coffee, this may take a few minutes. And voila! Your app is now running in the cloud!

3 Scan with SSL Labs

Scan your newly published webapp with Qualys SSL Labs - https://www.ssllabs.com/ssltest/analyze.html

- How did you do?
- What's good?
- What's bad - what's missing?
- Is it vulnerable to the exploit techniques shown in the previous session?
  - Clickjacking?
  - SSL stripping?

Bonus: Scan a site you've previously worked on - remember to check the "hide the results" box if you fear the result will be embarrassing..

4 Scan with securityheaders.io

Scan your site with https://securityheaders.io

- What score did you get?
- What's missing?

Bonus: Scan a site you've previously worked on - again, remember to hide the results if you fear the result..

5 Fix it!

You should now add an HSTS header to mitigate against SSL stripping. Open the project `Boosterproject`, and write an ASP.NET middleware to add the HSTS header. In `Startup.cs` - add middleware in your `Configure()` method:

```
app.Use(async (context, next) =>
{
 context.Response.Headers.Add("name", "value");

 await next.Invoke();
});
```

- Re-scan your site on https://securityheaders.io
- Extra credit: only add HSTS-header for https requests

More on ASP.Net Core and middleware can be found here: https://docs.microsoft.com/en-us/aspnet/core/fundamentals/middleware

Docs about the HSTS header can be found here: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security

6 XSS

There is a XSS-vulnerability in the `Boosterproject` project (click "XSS" in the top menu to show the page)

Exploit it

- Try to exploit it - could you exploit it? (Chrome stops you by default - try another browser)
- If you need help OWASP has a great writeup on XSS and injection testing https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Fix it

- Try to fix it without touching the actual webapp logic - only using headers
- Hint: X-XSS-Protection and CSP

> MDN has excellent security header documentation at https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

7 Clickjacking

There is a page that could be impacted by clickjacking in the `Boosterproject` project (click "Clickjackable" in the top menu)

Exploit it

- Try crafting an exploit - or go to evil.skandlabanken.no/ck/iframeyou.html for a "reference" exploit

Fix it

- Try to fix it without touching the webapp - using `CSP frame-ancestors` and/or `X-Frame-Options`

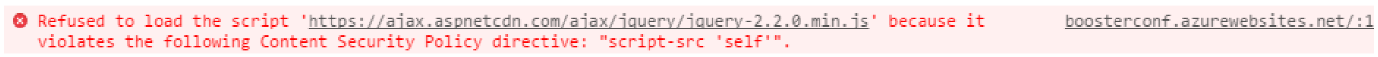
8 Violation reports

CSP includes a directive for making the browser send a report to a specified URI if violations to the policy are detected.

- Register for a free account at the excellent report-uri.io (by Scott Helme)
- Try to set up a report-uri pointing to report-uri.io
- Trigger a violation by fiddling with the DOM
- Observe what data gets sent, and what doesn't

9 Allowing certain content

Even the "good" project `Boosterproject_solved` has some CSP warnings:

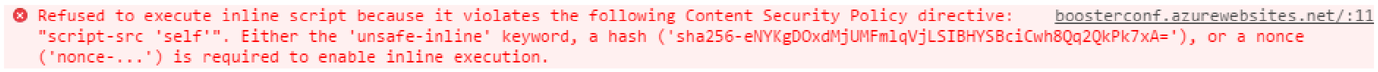


```
Refused to load the script 'https://ajax.aspnetcdn.com/ajax/jquery/jquery-2.2.0.min.js' because it      boosterconf.azurewebsites.net/:1
violates the following Content Security Policy directive: "script-src 'self'".
```

- Create a `default-src` directive for your CSP (if you haven't already) to only allow scripts from `'self'`
- Whitelist the domains needed to serve the javascripts the website "needs"

        - Hint: CSP `script-src`

Bonus: Someone also insist that your project **must** use inline scripts somewhere:



```
Refused to execute inline script because it violates the following Content Security Policy directive:    boosterconf.azurewebsites.net/:11
"script-src 'self'". Either the 'unsafe-inline' keyword, a hash ('sha256-eNYKgDOxdMjUMFmlqVjLSIBHYSBciCwh8Qq2QkPk7xA='), or a nonce
('nonce-...') is required to enable inline execution.
```

- Try to allow these scripts in your CSP **without** opening `unsafe-inline`.
    - Hint: Whitelist the hash of the script

10 Advanced: HTTP Public Key Pinning (HPKP)

Try adding HTTP Public Key Pinning (HPKP) to your project.

> Read about HPKP at MDN: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Public-Key-Pins

A partial implementation can be found in the reference project `Boosterproject_solved.`

11 NWebSec

NWebSec is an open source project by Andre Klingsheim. NWebSec allows for setting a security policy through configuration rather than code. Installation is thorugh a NuGet package.

> NWebSec can be found at https://github.com/NWebsec/NWebsec/ - docs are at https://docs.nwebsec.com/en/latest/

- Try to add NWebSec to your project and replicate your existing configuration using a single nwebsec configuration

12 securityheaders.io to the max

- Try to achieve an A+ on securityheaders.io!
- You should by now only be missing `X-Content-Type-Options` and `Referrer-Policy`, add these and run another scan

13 Let's encrypt on Azure (optional)

The handouts also include a semi-detailed explanation on how to install Lets Encrypt on an Azure web-app. Look to the handout `AzureLetsEncrypt_Guide.pdf` and walk through the steps. This requires you to either have an Azure-account or sign up for a 30-days trial.

If you need a custom domain during the setup - talk to Vidar and he will hook you up.