Security headers in ASP.NET Core

06/01/2020 Gérald Barré .NET, ASP.NET Core, Security, Web

Table Of Contents

Security Headers

Strict-Transport-Security

HTTP Strict Transport Security (HSTS) protect websites against man-in-the-middle attacks by indicating the browser to access the website using HTTPS instead of using HTTP.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security

X-Frame-Options

The `X-Frame-Options` HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a `<frame>`, `<iframe>`, `<embed>` or `<object>`. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into other sites.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

X-Permitted-Cross-Domain-Policies

The `X-Permitted-Cross-Domain-Policies` HTTP response header can be used to indicate whether or not an Adobe products such as Adobe Reader should be allowed to render a page. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into other applications.

https://www.adobe.com/devnet-docs/acrobatetk/tools/AppSec/xdomain.html

X-XSS-Protection

GÉRALD BARRÉ
aka. meziantou

RECENT POSTS

Reading Windows Application Manifest of an exe in .NET

Retrying a bash command

Detecting Dark and Light themes in a WPF application

Investigating a crash in Enumerable.LastOrDefault with a custom collection

Listing all available ETW events in a .NET application

LINKS

Dev Tool List

Dev News

Async/await resources

.NET Multithreading resources

Visual Studio tips

Online tools

The HTTP `X-XSS-Protection` response header is a feature that stops pages from loading when they detect reflected cross-site scripting (XSS) attacks. Although these protections are largely unnecessary in modern browsers when sites implement a strong `Content-Security-Policy` that disables the use of inline JavaScript (`'unsafe-inline'`), they can still provide protections for users of older web browsers that don't yet support CSP.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection

> Note
> This header is not supported in recent Chrome and Edge versions. Firefox has never implemented this feature.

X-Content-Type-Options

The `X-Content-Type-Options` response header is a marker used by the server to indicate that the MIME types advertised in the `Content-Type` headers should not be changed and be followed. This is a way to opt-out of MIME type sniffing, or, in other words, to say that the MIME types are deliberately configured.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

Referrer-Policy

The `Referrer-Policy` header controls how much referrer information (sent via the `Referer` header) should be included with requests. This may prevent information disclosure as URLs may contain sensitive data.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy

Feature-Policy

The `Feature-Policy` header provides a mechanism to allow and deny the use of browser features in its own frame, and in content within any `<iframe>` elements in the document. It can prevent the use of sensible APIs such as microphone, or it can help to fix performance issues such as using oversized images.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

Expect-CT

The `Expect-CT` header lets sites opt-in to reporting and/or enforcement of Certificate Transparency requirements, to prevent the use of mis-issued certificates for that site from going unnoticed. This helps detecting man-in-the-middle attacks by someone that could generate a certificate for your domain. Cloudflare has a service to monitor certificate generation: Introducing Certificate Transparency Monitoring

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Expect-CT

Content-Security-Policy

The `Content-Security-Policy` response header allows web site administrators to control resources the user agent is allowed to load for a given page. With a few exceptions, policies mostly involve specifying server origins and script endpoints. This helps guard against cross-site scripting attacks (XSS).

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy

ASP.NET Core middleware

In ASP.NET Core, you can set the headers for every request using a middleware. ASP.NET Core provides a middleware to set the HSTS headers when needed and redirecting to https. You'll have to set other security headers manually.

Note that you'll have to adapt the parameters depending on the features your application uses.

C# copy

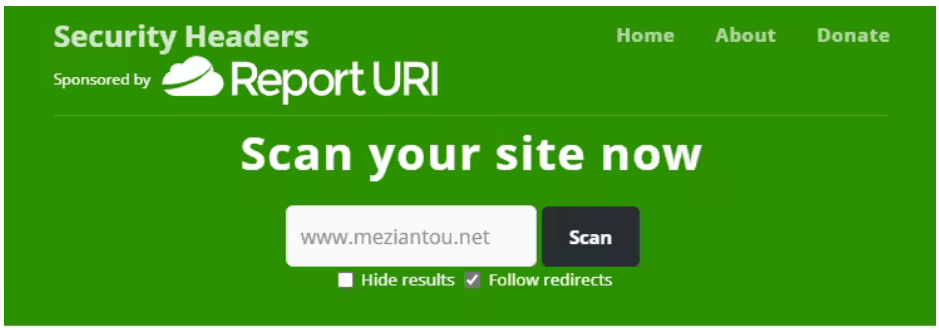
```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddRazorPages();

        // Configure HSTS
        // https://learn.microsoft.com/en-us/aspnet/core/security/enforcing-ssl?WT.mc_id=DT-MVP-
        // https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
        services.AddHsts(options =>
        {
            options.MaxAge = TimeSpan.FromDays(90);
            options.IncludeSubDomains = true;
            options.Preload = true;
        });

        // Configure HTTPS redirection
        services.AddHttpsRedirection(options =>
        {
            options.RedirectStatusCode = StatusCodes.Status301MovedPermanently;
            options.HttpsPort = 443;
        });
    }

    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (!env.IsDevelopment())
        {
            // HSTS should only be enabled on production, not on localhost
            app.UseHsts();
        }

        // Add other security headers
        app.UseMiddleware<SecurityHeadersMiddleware>();

        // Redirect http to https
        app.UseHttpsRedirection();

        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllers();
        });
    }
}

public sealed class SecurityHeadersMiddleware
{
    private readonly RequestDelegate _next;

    public SecurityHeadersMiddleware(RequestDelegate next)
    {
        _next = next;
    }

    public Task Invoke(HttpContext context)
    {
        // https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
        // TODO Change the value depending of your needs
        context.Response.Headers.Add("referrer-policy", new StringValues("strict-origin-when-cro

        // https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options
        context.Response.Headers.Add("x-content-type-options", new StringValues("nosniff"));

        // https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
```

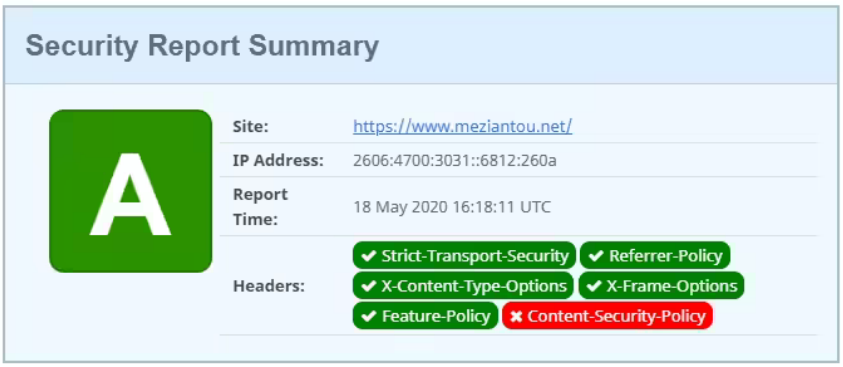
```
context.Response.Headers.Add("x-frame-options", new StringValues("DENY"));

// https://security.stackexchange.com/questions/166024/does-the-x-permitted-cross-domain
context.Response.Headers.Add("X-Permitted-Cross-Domain-Policies", new StringValues("none

// https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
context.Response.Headers.Add("x-xss-protection", new StringValues("1; mode=block"));

// https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Expect-CT
// You can use https://report-uri.com/ to get notified when a misissued certificate is d
context.Response.Headers.Add("Expect-CT", new StringValues("max-age=0, enforce, report-u

// https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy
// https://github.com/w3c/webappsec-feature-policy/blob/master/features.md
// https://developers.google.com/web/updates/2018/06/feature-policy
// TODO change the value of each rule and check the documentation to see if new features
context.Response.Headers.Add("Feature-Policy", new StringValues(
    "accelerometer 'none';" +
    "ambient-light-sensor 'none';" +
    "autoplay 'none';" +
    "battery 'none';" +
    "camera 'none';" +
    "display-capture 'none';" +
    "document-domain 'none';" +
    "encrypted-media 'none';" +
    "execution-while-not-rendered 'none';" +
    "execution-while-out-of-viewport 'none';" +
    "gyroscope 'none';" +
    "magnetometer 'none';" +
    "microphone 'none';" +
    "midi 'none';" +
    "navigation-override 'none';" +
    "payment 'none';" +
    "picture-in-picture 'none';" +
    "publickey-credentials-get 'none';" +
    "sync-xhr 'none';" +
    "usb 'none';" +
    "wake-lock 'none';" +
    "xr-spatial-tracking 'none';"
    ));

// https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
// https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy
// TODO change the value of each rule and check the documentation to see if new rules ar
context.Response.Headers.Add("Content-Security-Policy", new StringValues(
    "base-uri 'none';" +
    "block-all-mixed-content;" +
    "child-src 'none';" +
    "connect-src 'none';" +
    "default-src 'none';" +
    "font-src 'none';" +
    "form-action 'none';" +
    "frame-ancestors 'none';" +
    "frame-src 'none';" +
    "img-src 'none';" +
    "manifest-src 'none';" +
    "media-src 'none';" +
    "object-src 'none';" +
    "sandbox;" +
    "script-src 'none';" +
    "script-src-attr 'none';" +
    "script-src-elem 'none';" +
    "style-src 'none';" +
    "style-src-attr 'none';" +
    "style-src-elem 'none';" +
    "upgrade-insecure-requests;" +
    "worker-src 'none';"
    ));
```

```
        return _next(context);
    }
}
```

Validation

You can check you have correctly set the security headers by using the following service: https://securityheaders.com





Do you have a question or a suggestion about this post? Contact me!

Follow me:

Enjoy this blog? Buy me a coffee Sponsor on GitHub