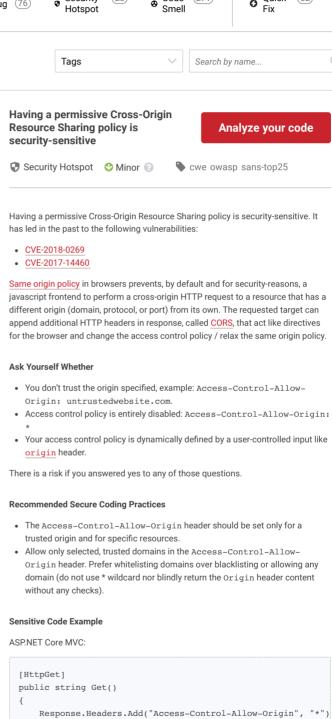


Code Smell



Response, Headers, Add (HeaderNames, AccessControlAllowOrigi

public void ConfigureServices(IServiceCollection services)

builder.WithOrigins("*"); // Sensitive

options.AddDefaultPolicy(builder =>

services.AddCors(options =>

Unused "using" should be removed

Code Smell

A close curly brace should be located at the beginning of a line

Code Smell

Tabulation characters should not be used

Code Smell

Methods and properties should be named in PascalCase

Code Smell

```
});

options.AddPolicy(name: "EnableAllPolicy", builder =
{
    builder.WithOrigins("*"); // Sensitive
});

options.AddPolicy(name: "OtherPolicy", builder =>
{
    builder.AllowAnyOrigin(); // Sensitive
});
});

services.AddControllers();
}
```

ASP.NET MVC:

```
public class HomeController : ApiController
{
   public HttpResponseMessage Get()
   {
      var response = HttpContext.Current.Response;

      response.Headers.Add("Access-Control-Allow-Origin",
      response.Headers.Add(HeaderNames.AccessControlAllowOresponse.AppendHeader(HeaderNames.AccessControlAllow)
   }
}
```

User-controlled origin:

```
String origin = Request.Headers["Origin"];
Response.Headers.Add("Access-Control-Allow-Origin", origin);
```

Compliant Solution

ASP.NET Core MVC:

```
[HttpGet]
public string Get()
{
    Response.Headers.Add("Access-Control-Allow-Origin", "htt
    Response.Headers.Add(HeaderNames.AccessControlAllowOrigi)
}
```

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddCors(options =>
    {
        options.AddDefaultPolicy(builder =>
        {
            builder.WithOrigins("https://trustedwebsite.com"
        });
        options.AddPolicy(name: "EnableAllPolicy", builder =
        {
            builder.WithOrigins("https://trustedwebsite.com"
        });
    });
    services.AddControllers();
}
```

ASP.Net MVC:

```
public class HomeController : ApiController
{
```

```
public HttpResponseMessage Get()
{
    var response = HttpContext.Current.Response;

    response.Headers.Add("Access-Control-Allow-Origin",
    response.Headers.Add(HeaderNames.AccessControlAllowOresponse.AppendHeader(HeaderNames.AccessControlAllow)
}
```

```
[EnableCors(origins: "https://trustedwebsite.com", headers:
public HttpResponseMessage Get() => new HttpResponseMessage(
{
    Content = new StringContent("content")
};
```

User-controlled origin validated with an allow-list:

```
String origin = Request.Headers["Origin"];

if (trustedOrigins.Contains(origin))
{
    Response.Headers.Add("Access-Control-Allow-Origin", orig
}
```

See

- OWASP Top 10 2021 Category A5 Security Misconfiguration
- OWASP Top 10 2021 Category A7 Identification and Authentication Failures
- developer.mozilla.org CORS
- developer.mozilla.org Same origin policy
- OWASP Top 10 2017 Category A6 Security Misconfiguration
- OWASP HTML5 Security Cheat Sheet Cross Origin Resource Sharing
- MITRE, CWE-346 Origin Validation Error
- MITRE, CWE-942 Overly Permissive Cross-domain Whitelist
- SANS Top 25 Porous Defenses

Available In:

sonarcloud 🙆 | sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Privacy Policy