Felipe M Saboya
.NET Developer

Preventing CSRF attacks in ASP.NET Core

http://www.felipemsaboya.net/Blog/AspNetCore-CSRF-AntiForgeryToken

2018/04/01

Felipe M Saboya - .NET Developer

Microsoft MCSA, MCPS, MCP • EXIN ITIL Foundation

www.felipemsaboya.net

Microsoft Certified Professional
Microsoft
MCSA
Web Applications
Earned 2016

Microsoft
Exam passed
Developing ASP.NET MVC Web Applications

Microsoft
Exam passed
Programming in HTML5 with JavaScript and CSS3

ITIL
FOUNDATION
CPD

Microsoft
Exam passed
Querying Microsoft SQL Server 2012/2014

INTRODUCTION

When developing Web Applications we should be concerned with "information insecurity", yes "insecurity", since the approach to security should assume that nobody is 100% secure if they are connected to the internet, that is, we, professionals who propose to develop and deliver complex and connected products, we must keep in mind all the security features that are offered by the technologies we propose to work on, such as **Microsoft's ASP.NET**.

That said, I bring my first article on the importance of using the feature called **AntiForgeryToken** that helps us prevent attacks **CSRF - Cross-Site Request Forgery** (pronounced C-surf), also know as XSRF.

THE PROBLEM

CSRF is an attack that forces an authenticated user into a Web Application to perform unwanted actions on behalf of the user, ie the attack forces the victim's browser to send a request.

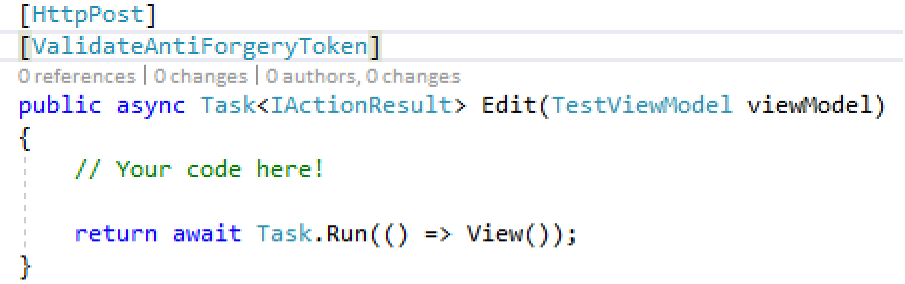
For example, you may be logged in to the bank's website and decide to browse another site, and by clicking on some advertisement it performs a "Post" function and transfers your funds, without your noticing, to another account.

THE SOLUTION

The solution is very simple because **ASP.NET MVC** have has since its initial versions the **ValidateAntiForgeryToken** class attribute, which must be applied in two steps, and this will cause the Web Application to only recognize the action of the currently authenticated user and will reject any another that tries to perform an action for it, in this case, the **CSFR attack will be prevented** and the Web Application will **return the Bad Request error**.

1ST STEP

In the decoration of the class "Post" also adding the ValidateAntiForgeryToken attribute. As shown in the side example.

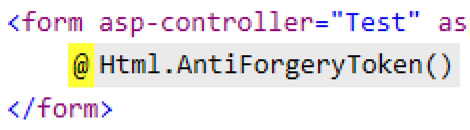


```
[HttpPost]
[ValidateAntiForgeryToken]
0 references | 0 changes | 0 authors, 0 changes
public async Task<IActionResult> Edit(TestViewModel viewModel)
{
    // Your code here!

    return await Task.Run(() => View());
}
```

2ND STEP

Adding the Helper to the View form. This will generate the "Token" to be recognized by the Web Application.



```
<form asp-controller="Test" asp-action="Edit" method="post">
    @Html.AntiForgeryToken()
</form>
```

* It will be generated at run time something similar to the input:

```
<input name="__RequestVerificationToken" type="hidden"
value="CfDJ8PIdqkEW0YRGiAj8BeRzSBbihqUysw0OL9LaEmtunQYO_YPhN1jS7YTAhPIsUY0HgD3yc1_ItbFMdcusdHHjQzuQGCrnew-
orLK5GjvV_Ar3N-gomVX7UoHFGOjSKqW84rh4xFcDCqVjbOzzd12qOnENuOLbUmpx_jb7mWyJa_5W5dEGL-b6Siqw" />
```

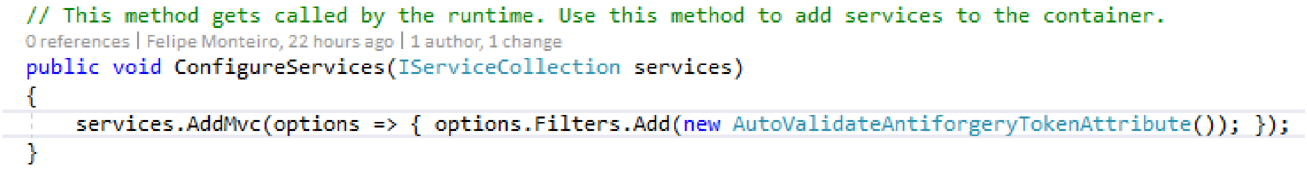
ADDITIONAL SOLUTION FOR ASP.NET CORE

In addition, the new ASP.NET Core technology has the ability to apply validation to the entire solution in the **Startup** class and apply only to state change classes (POST, PUT and DELETE).

OWASP recommends that the verbs listed above (POST, PUT and DELETE) be protected from this type of attack (see references on article page), so in summary, the implementation helps us to comply with this recommendation and, **if is necessary to ignore the validation** of the CSRF we can only decorate the class with the **IgnoreAntiforgeryToken** attribute.

STARTUP CLASS

Complementing the “AddMvc” with the line presented on side in your Startup class.



```
// This method gets called by the runtime. Use this method to add services to the container.
0 references | Felipe Monteiro, 22 hours ago | 1 author, 1 change
public void ConfigureServices(IServiceCollection services)
{
    services.AddMvc(options => { options.Filters.Add(new AutoValidateAntiforgeryTokenAttribute()); });
}
```

CONCLUSION

- I suggest that you always use the feature presented in this article;
- I suggest reading the document with a TOP 10 that I found at https://www.owasp.org/images/4/42/OWASP_TOP_10_2007_EN-BR.pdf, which also mentions other threats to be analyzed and dealt with;
- Analyse the OWASP website in general, because there is information that we simply need to know.

Thanks!

Felipe M Saboya - .NET Developer

Microsoft MCSA, MCPS, MCP • EXIN ITIL Foundation

www.felipemsaboya.net