Preventing Cross Site Request Forgery

John F. Gnazzo

Introduction

Cross Site Request Forgery (CSRF) is an attack against a web application which forces a user of a web application to execute unwanted actions within the web application in which they are currently authenticated. By sending a malicious link via email, social network post, or chat, an attacker may trick the users of a web application into executing actions of the attacker's choosing. A successful CSRF exploit can compromise end user data, allow unwanted transactions to occur, redirect purchase goods to fraudulent shipping addresses, and other unwanted and unexpected actions.

Microsoft ASP.Net provides a mechanism to prevent CSRF attacks in MVC architected web applications.

CSRF Prevention

To prevent CSRF attacks in Microsoft ASP.net MVC applications, code must be added to the View and Controller classes. The following two (2) sections detail the specifics of adding code to prevent CSRF attacks at the View and Controller level.

View

The View class is instantiated on the server and generates HTML to render the form to the client in the client's browser.

To prevent a CSRF attack, a helper method, specifically ***@Html.AntiForgeryToken***, is added to the view class.

This will add a hidden field to the form sent to the browser.

The following code is a View page written in C# and uses the Razor View Engine. This code demonstrates the insertion of the ***@Html.AntiForgeryToken*** method within the code that will generate a hidden field to the form being posted.

Upon clicking the submit button within this form from the browser, a HTTP Post request is sent to the Login Method within the Account Controller on the Server.

```
@using (Html.BeginForm("Login", "Account", new { ReturnUrl = ViewBag.ReturnUrl },
FormMethod.Post, new { @class = "form-horizontal", role = "form" }))
            {
                @Html.AntiForgeryToken()
                … Input fields follow

              <input type="submit" value="Log in" class="btn btn-default" />

            }
```

The hacker will not be able to generate a token that will be accepted by the controller on the server.

Controller

The Controller is run on the server and responds to events including the posting of forms from a browser.

To eliminate CSRF threats, a ***ValidateAntiForgeryToken*** attribute is added to the controller Method. This will cause the method to interpret the antiForgery token generated from the view and to generate an error if the token is missing or does not match the specific token generated by the view.

The following code shows the insertion of the ***ValidateAntiForgeryToken*** attribute in front of the Login controller method.

```
        [HttpPost]
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
        {
              // Process Form Input and redirect appropriately

        }
```

Impact of implementing CSFT prevention code

If the ***@Html.AntiForgeryToken*** method is added to the view, and if the method on the controller called by the view has the ***ValidateAntiForgeryToken*** attribute, the controller will process the data within the posted form, appropriately.

If the form posted to the controller does not include a token, or if the token is invalid, the controller will throw an exception and the form will not be processed, and no harm will come to the application or data.