Mariusz Wieczorek

mariusz.wieczorek@kabat.pl

Streszczenie

Uwierzytelnienie I autoryzacja  
Ochrona przed atakami typu: Cross Site Scripting XSS,  
 Cross Site Request Forgery CSRF lub XSRF

ASP.NET MVC

Bezpieczeństwo aplikacji webowej

**Uwierzytelnienie i autoryzacja** -czyli logowanie do aplikacji i sprawdzanie czy użytkownik ma uprawnienia do danej akcji jest pierwszym stopniem zabezpieczeń.  
Sprawdzamy w kontrolerze Id zalogowanego użytkownika, aby uniemożliwić manipulację w danych przesyłanych do serwera. Ustawiamy atrybuty **[Authorize]** przed akcjami kontrolera lub prze całą klasą kontrolera.  
  
**Cross Site Scripting** XSS, osadzenie w treści atakowanej strony kodu javascript  
ASP.NET ma automatyczną ochronę przed wprowadzaniem kodu HTML w polach tekstowych.  
Aby dopuścić kod HTML możemy ustawić ValidateImput na false.

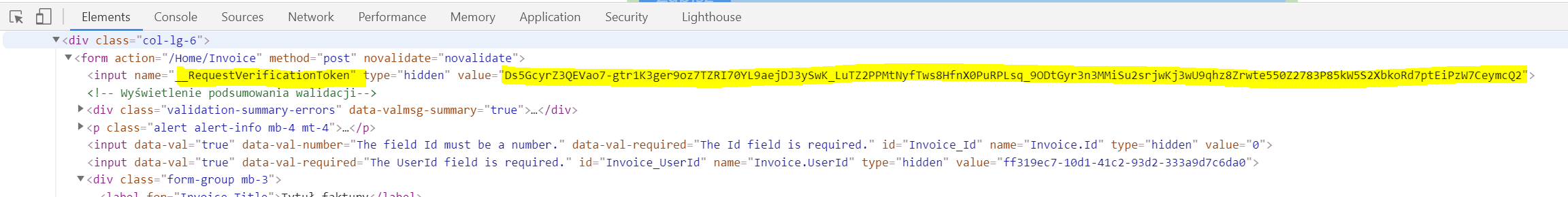
**Cross Site Request Forgery** CSRF lub XSRF – wysyłanie danych do serwera zmienionych przez osoby trzecie. (Forgery – Fałszerstwo) Jest to atak nie na serwer ale na przeglądarkę.  
Użytkownik Marek został zalogoweny do serwisu internetowego i zapamiętał swoje dane.  
Następnie mógł dostać link, który zawierał jakieś akcje do wykonania na portalu , na którym się zalogował. Np. taka, która może zmienić e-mail potrzebny do logowania.  
Potem haker klika na zapomniełem hasło i już dostaje nowe hasło na swojego maila.  
Zabezpieczenie:  
Pierwszy krok oznaczmy akcję atrybutem: [ValidateAntiForgeryToken]

|  |
| --- |
| [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult InvoicePosition(InvoicePossition invoicePosition)  {  var userId = User.Identity.GetUserId(); |

Drugi krok w widoku na formularzu umieszczamy helper @Html.AntiForgeryToken()

|  |
| --- |
| <div class="row">  <div class="col-lg-6">  @using (Html.BeginForm("Invoice", "Home"))  {  @Html.AntiForgeryToken()    <!-- Wyświetlenie podsumowania walidacji-->  @Html.ValidationSummary()  if (Model.Invoice.Id == 0)  {  <p class="alert alert-info mb-4 mt-4">Pola tytuł, klient, sposób płatności i termin   są <strong>wymagane</strong> </p>  } |

Po przejściu na formularz zostanie wygenerowany specjalny token dla użytkownika,  
po kliknięciu w submit ten token jest weryfikowany. Jeżeli nie ma tokena lub jest zły, to  
zostanie rzucony wyjątek.



Nie umieszczać w kontrolerze metod, które są publiczne, a nie są akcjami.  
Jak już muszą się tam takie znaleźć, to należy je oznaczyć atrybutem NonAction

|  |
| --- |
| [NonAction]  public void Testy()  {  } |