Accelerated Security Course - Episode 4 - CSRF

Accelerated Security Course - Episode 4: CSRF

This article was written by Damien Metzger, and first published on the PrestaShop blog, on November 30th, 2011.

A CSRF breach consists of exploiting a trusted user's identity by forcing the browser to send commands unbeknownst to the user. Basically, if a page is protected by a login/password system (e.g. stored in a cookie) then you cannot access it without signing in, unlike the retailer who is already signed in.

Therefore a hacker simply needs to send the retailer to the page of his choice by sending an instant message or email such as "Hi! Take a look at my new photos… Do you think I'm hot?" with a link to redirect the victim to http://www.maboutique.com/admin/index.php?tab=AdminCustomers&deletecustomer&id_customer=1.

In theory, this link deletes customer number 1 from the maboutique.com shop. This won't work with PrestaShop, as the software has been made secure to avoid this type of usage, but you get the idea: the hacker just needs to get the retailer to click on a link which carries out the required action. You can't do it yourself but the retailer could do it in your place without knowing it.

This is a very malicious attack, as the shop does not have to have been directly breached. A more active protection system to that used for XSS or injections is required.

The solution lies in using security tokens, as you can see in PrestaShop or phpMyAdmin for example. The developer must generate a unique hash code based on data specific to the retailer for each page and even each activity: combine the username, shop URL, a salt generated upon installation, page URLand activity as a parameter of the `sha1()` function for a truly complete hash code. Next, every time a page loads and before processing, recalculate the token and compare it to what you passed as a parameter of each link. This way the hacker will have the impossible task of calculating the correct sha1 to exploit the breach.

But we can't end here. For example, a hacker can try to combine XSS and CSRF. By exploiting an XSS breach the hacker can make you use a JavaScript code. The hacker can then use this JavaScript code to get the token in the URL or a token on other links on the page. **This is why security is essential. A single breach is enough to get in and once they're over the first hurdle, it's much easier for a hacker to weave his web.**