https://docs.joomla.org/How_to_add_CSRF_anti-spoofing_to_forms

What is a CSRF Attack?

A Cross Site Request Forgery (CSRF) attack relies on the trust a website has for a user to execute unauthorized requests and or transactions. For example, say a user is logged into their Joomla! websites' administrator interface in one tab and is browsing a compromised site in another tab. A simple CSRF attack can be launched simply by tampering with IMG elements in some browsers so that they point to something like

`http://some/joomla/site/administrator/index2.php?option=com_users&task=delete`
…

When the user browses the compromised site, that image will be requested and because the user is currently logged in to the administrator interface of her Joomla! site, the forged request will be positively authenticated and executed. To prevent simple CSRF attacks like the one above, request tokens have been added to all forms in the front-end and back-end Joomla! interfaces. The tokens are randomized strings that are used to authenticate that the request being made is coming from a valid form and a valid session. This simple measure is very effective at preventing a large percentage of potential CSRF attacks, however, due to the nature of CSRF they are extremely difficult, if not impossible, to secure against completely. Protecting Against CSRF Attacks

Joomla! attempts to protect againt CSRF by inserting a random string called a token into each POST form and each GET query string that is able to modify something in the Joomla! system. This random string provides protection because not only does the compromised site need to know the URL of the target site and a valid request format for the target site, it also must know the random string which changes for each session and each user.

The Joomla! Framework makes it easy for you to include such protection in your components as well. This is simple to implement in both POST and GET requests.

POST Request

POST requests are submitted in HTML using forms. In order to add the token to your form, add the following line inside your form:

```
<?php echo JHtml::_( 'form.token' ); ?>
```

This will output something like the following:

```
<input type="hidden" name="1234567890abcdef1234567890abcdef" value="1" />
```

GET Request

GET requests are submitted in HTML using query strings. In order to add the token to your query string, use a URL like:

```
<?php echo JRoute::_(
'index.php?option=com_example&controller=object1&task=save&'.
JSession::getFormToken() .'=1' ); ?>
```

This will generate a URL with the token in the query string.

Checking the Token

Once you have included the token in your form or in your query string, you must check the token before your script carries out the request. This is done with the following line:

```
JSession::checkToken() or die( 'Invalid Token' );
```

If the request is coming from the query string, you must specify this. The code then becomes:

```
JSession::checkToken( 'get' ) or die( 'Invalid Token' );
```

Recommended Security Procedures

While these methods help to prevent against these types of attacks, it is important to realize that as a system administrator, there are good security practices to follow which will prevent a site from being compromised.

- Don't browse other sites in the same browser while you are logged into your site.
- Log out from your site after you are done.
- Don't stay logged into your site while you are not doing anything.
- Ensure that the address in the browser bar matches the address of your site.

By practicing these safe surfing habits you will eliminate most threats to your web site.

From:
https://wiki.merkatu.info/ - **Wiki de merkatu**

Permanent link:
https://wiki.merkatu.info/tokens:evitar_vulnerabilidades_csrf?rev=1447845651

Last update: **2017/03/27 17:43**