Hello, my name is John Zuccato, and I'm a member of the IBM Security system Ethical Hacking team. In this video I will be talking about Missing Function Level Access Control.

What is Missing Function Level Access Control?

Can a user directly browse to a resource? Does the UI expose an unauthorized resource? The server should not solely rely on user-supplied input.

Anyone can send a request to a web application. The web application needs to verify that the request has verified access to the resource. The web application needs to verify the request at the UI level, as well as the backend function level. An attacker will ignore the UI and a forge requests that access unauthorized functionality.

Administrative functions are key targets. Here we have two pages. Page 1, which is accessible to an administrator, and Page 2, which is accessible by any user. If an unauthenticated user can access either page or if a regular user can access the admin page, then we have what is known as missing function level access control.

Using AppScan Standard, we can test for missing level access. How can we do this? There are a set of tests known as privilege escalation tests. By comparison with a higher privileged user, we can point AppScan to point to scan results that were produced using a higher level of access or permissions than used by the current scan.

During the scan, AppScan attempts to access the additional links that were available to the higher level user. Using the current lower level access permissions, the scan results indicate where these attempts were successful.

So in this case, I'm going to show you how a regular user can access administration pages. What I need to do first is create a scan using the administrator account. I go to demo.testfire.net and I'm going to log in here as an administrator.

The first thing I have to do is to record my login. Now, I have two options. I can run just an explore scan or a full scan. For our example, I'm going to run an "explore only" scan, so that I can get all the pages I have access to as an administrator.

I'm going to create a scan using the user account jsmith. I previously recorded the login. You can see the details here. In the Test area, in the Privilege Escalation page, I want to add the administration scan I did earlier.

I point to my scan file with the administrator privileges and I'm going to add it here as Admin. I'm going to make the scan run a lot faster so I'm going to change the test policy to just run the Privilege Escalation policy.

In the test policy, I select No Grouping, and type in 'Privilege', and just select these tests, and click Apply. So now I'm going to kick off my scan and I'll do a comparison to see if JSmith the user can access any pages that are only accessible by the administrator.

So here I have a completed scan with Privilege Escalation results. As you can see, there are 52. Let's go look up one.

Hmn, this admin.aspx result looks interesting. Let's see if we can get to this actual page using the application.

So here I have the demo.testfire application, and I'm going to see if I can get to the admin page as an unauthenticated user.

As you can see, I'm getting challenged for a username and password. So let's sign in as JSmith.

So I'm logged in as John Smith. Let's see if I can directly browse to the admin page that was discovered by the scan. As you can see here, I've loaded up the Edit User Information page. As a user John Smith, I shouldn't be able to get here. I can go and look at all the users and change their passwords. Or I can go and add a new user.

Preventing Missing Function Level Access Control.

The default approach should be to deny access. If the user has access, then allow it. Don't rely on your UI to protect the system. Make sure that there is protection on the backend as well; that is, at the function level. And thoroughly check every URL in your systems for access control.

This concludes our coverage of issue #7 of the OWASP Top 10 list. I hope you found this information useful. I wish you the

best of luck writing and maintaining secure software.