Open Redirection Attacks

In this short LetsDefend Lab, I will find the Open Redirection Attack within an access log file.

First it is important to review what an Open Redirection Attack is. This is a vulnerability that occurs when a website or application redirects the user to a different URL without proper URL sanitation. This can be used by the attacker to trick users into visiting malicious pages or performing unwanted actions. There are different types of Open Redirection Attacks, commonly separated by where they are performed. For example:

**URL-based open redirection:** This is often considered the most common attack method which occurs when the website takes the URL parameter as an input and redirects it without sanitation. The attacker can craft a malicious domain to be used as the input parameter for the redirect leading to problems. For example, say we have a simple example.com. If their is no sanitization an attacker could append a URL redirect such as http://example.com/redirect?url=http://malicious.com. This could unknowingly send the user to malicious.com, created by the attacker.

**JavaScript-based open redirection:** As the name implies, this redirection occurs within the javascript of the webpage or application. For example:

window.location = new URLSearchParams(window.location.search).get('url');

Like URL-based redirection, the attacker could craft a URL like http://example.com/redirect?url=http://malicious.com to return http://malicious.com as window.location.

Note: main difference between URL and JavaScript based is where they occur. URL redirection occurs **Server side** via the URL passed to the server. In contrast JavaScript-based open redirection occurs **Client Side** in the browser.

**Meta refresh-based open redirection**: This type of open redirection vulnerability occurs when a website uses the HTML "meta refresh" tag to redirect users to another URL automatically. For example:

<meta http-equiv="refresh" content="0;url=http://example.com/redirect?url=http://malicious.com">

**Header-based open redirection:** This type of open redirection vulnerability occurs when a website uses HTTP headers, such as "Location" header, to perform a redirect, but the target URL is obtained from untrusted or user-controlled sources without proper validation or sanitization.

Example

<head>

Location: http://example.com/redirect?url=http://victim.com

</head>

**Parameter-based open redirection:** This type of open redirection vulnerability occurs when a website uses a parameter in the URL or in a form submission as part of the redirect process, but fails to properly validate or sanitize the parameter value.

For example: http://example.com/dashboard?redirect\_to=home

An attacker can manipulate the redirect\_to parameter for a malicious site.

http://example.com/dashboard?redirect\_to=http://malicious.com

Begining the example LetsDefend lab, I open the access.log text file and use the regular expression “"GET .\*postId=%2f.\*%2e.\*"” to help find suspicious activity. To quickly explain the important parts of this expression:

“Get: matches beginning of HTTP request.

postId=: matches the literal variable used in most lines of the file.

%2f : matches the URL encode for “ / ” often use for path traversal and redirect attacks

.\* : any characters

%2e : matches url “ . “ also used in directory travel path

Thus this regular expression will receive any GET request with postId parameter and suspicious characters including (/,.).

The attack itself is happening: 27/Apr/2023 15:45:22

It is being performed by the IP: 86.236.188.85

The parameter being attacked: postID

