

Ataques de CSRF o Cross-Site Request Forgery

El Cross-Site Request Forgery (CSRF), también conocido como ataque de falsificación de solicitudes entre sitios, es un tipo de vulnerabilidad de seguridad en aplicaciones web que explota la confianza de un sistema en las solicitudes realizadas desde el navegador del usuario autenticado. En un ataque CSRF, un atacante engaña a un usuario autenticado para que realice una acción no deseada en un sitio web al aprovechar la sesión activa del usuario como por ejemplo, cambiar su dirección de correo electrónico, su contraseña o realizar una transferencia de dinero.

Mediante una CSRF, un atacante puede eludir parcialmente la política que evita que diferentes sitios web se interfieran entre sí (Same-Origin Policy).

Aquí hay una explicación más detallada de cómo funciona un ataque CSRF:

1. **Autenticación del Usuario:** El usuario autenticado accede a un sitio web legítimo y se autentica con éxito. Durante este proceso, el sitio web establece una sesión válida para el usuario.
2. **Solicitud Maliciosa:** Mientras la sesión del usuario sigue activa, el usuario visita otro sitio web controlado por el atacante, que contiene un código malicioso, como un enlace o un formulario oculto, que realiza una solicitud a la aplicación vulnerable.
3. **Ejecución de la Solicitud:** La solicitud maliciosa se ejecuta en el contexto de la sesión válida del usuario en el sitio web vulnerable. Como resultado, la aplicación web procesa la solicitud como si fuera legítima, ya que proviene de un usuario autenticado.
4. **Acción No Deseada:** Dependiendo de la naturaleza de la solicitud maliciosa, puede llevar a cabo acciones no autorizadas en nombre del usuario, como cambiar la contraseña, realizar compras, eliminar datos, etc.
5. **Impacto:** El atacante puede aprovechar el ataque CSRF para realizar acciones maliciosas sin que el usuario sea consciente de ello. Esto puede comprometer la integridad y la seguridad de los datos del usuario, así como llevar a cabo actividades fraudulentas.

Para prevenir los ataques CSRF, se pueden implementar varias medidas de seguridad, incluyendo:

- **Tokens Anti-CSRF:** Las aplicaciones web pueden generar tokens únicos y aleatorios y asociarlos con las acciones sensibles realizadas por los usuarios. Estos tokens que genera la aplicación del lado del servidor y se transmite al cliente de tal manera que se incluye en la siguiente solicitud realizada por el cliente y se incluyen en los formularios o las solicitudes y se verifican en el servidor para garantizar que la solicitud provenga de una fuente legítima y no de un atacante. Los Tokens previenen estos ataques dado que el atacante no puede predecir el valor del Token CSRF del usuario y no puede construir una

solicitud con todos los parámetros necesarios para que la aplicación cumpla con la solicitud.

- **Cabeceras HTTP:** El uso de cabeceras HTTP como `SameSite` y `Referer` puede ayudar a mitigar los ataques CSRF al restringir el envío de cookies en solicitudes cruzadas y verificar el origen de las solicitudes, respectivamente.
- **Autenticación de Doble Factor (2FA):** La implementación de la autenticación de doble factor puede agregar una capa adicional de seguridad al requerir que los usuarios proporcionen un segundo factor de autenticación, como un código único enviado a su teléfono móvil, antes de realizar acciones sensibles.

Al aplicar estas medidas de seguridad, las aplicaciones web pueden protegerse de los ataques CSRF y garantizar la integridad y la seguridad de los datos del usuario.

Ejemplo

Vamos a crear un sitio malicioso llamado **bancolsoeduardos.es**

Nuestra víctima visita nuestro sitio malicioso "**bancolsoeduardos.es**", se activa una solicitud POST y se envía a la aplicación legítima de **bancoloseduardos.es**. El JavaScript que se encuentra en las etiquetas "script" garantiza que el formulario se envíe tan pronto como el usuario cargue la página, sin que se requiera ninguna interacción del usuario, ni que el usuario se dé cuenta de lo que está sucediendo.

```
<html>
  <body>
    <form action="https://bancoloseduardos.es/transfer" method="POST">
      <input type="hidden" bsb="421314" accountNo="123456789" amount="100" />
    </form>
    <script>
      document.forms[0].submit();
    </script>
  </body>
</html>
```

El formulario anterior crea la siguiente solicitud para la aplicación legítima de **bancoloseduardos.es**. La solicitud contiene la cookie de sesión del usuario legítimo, pero contiene nuestro número de cuenta bancaria.

```
POST /transfer HTTP/1.1
Host: bancoloseduardos.es
Content-Length: 42
Content-Type: application/x-www-form-urlencoded
Cookie: session=3PhtXFzhEWzdhFpQfTqrcjW2ItpDAkDm
```

bsb=421314&accountNo=1736123125&amount=100

Este ataque fue posible debido a algunas condiciones:

El usuario inició sesión en **bancoloseduardos.es**

El usuario que visitó nuestro sitio también inició sesión en la aplicación **bancoloseduardos.es**. Su cookie de sesión se estaba almacenando en su navegador y, como no tenía el atributo SameSite, pudimos robarla para nuestra solicitud.

Revision #4

Created 4 February 2024 09:11:58 by etaboda

Updated 4 February 2024 09:44:12 by etaboda