Segurança Web: Ataques

Cross-Site Request Forgery

A simple attack

Quem possui o cookie, consegue fazer pedidos.

Se **não houvesse CORS**, o atacante poderia simplesmente requisitar um cookie de um website comprometido, escutar a rede (e.g netcat) e conseguir o cookie.

Com **CORS e SOP** o cookie só deve ser enviado a websites que possuem a mesma origem ou que foram autorizados.

Ainda há outros riscos, pelo que alguem pode tentar visualizar o cookie por meio de javascript. Um meio de dificultar a obtenção do cookie é demarcar o cookie como `HTTPOnly`.

No entanto, é pedir um recurso que causa uma ação no servidor:

```
<img src="www.bank.ch/transfer"></img>
```

What Is Cross-Site Request Forgery (CSRF) and How Does It Work? | Synopsys

An attacker's aim for carrying out a CSRF attack is to force the user to submit a state-changing request. Examples include: Submitting or deleting a record. Submitting a transaction. Purchasing a product. Changing a password. Sending a message. Social engineering platforms are often used by attackers to launch a CSRF

https://www.synopsys.com/glossary/what-is-csrf.html

Outras medidas

- Usar CORS com preflight

- Usar SameSite=Strict
- **Impedir login: token secreto da sessão**
- Usar SOP
- Defesa em profundidade

SQL Injection

SQL injection segue o mesmo princípio fornecido por input injection: o input do utilizador pode ser malicioso.

Medidas

- **Nunca criar comandos de sql dinamicamente**
- Usar **comandos parametrizados** como: `INSERT INTO products (name, price) VALUES (?, ?);` .

What is parameterized query?

Thanks for contributing an answer to Stack Overflow! Please be sure to answer the question. Provide details and share your research! Asking for help, clarification, or responding to other

https://stackoverflow.com/questions/4712037/what-is-parameterized-query

A parameterized query (also known as a *prepared statement*) is a means of pre-compiling a SQL statement so that all you need to supply are the "parameters" (think "variables") that need to be inserted into the statement for it to be executed.

Ao pré compilarmos as queries evitamos as queries de serem modificadas, por isso não é sujeita SQL injection attacks.

- **Bibliotecas ORM**: estas bibliotecas oferecem uma mecanismos de abstração da base de dados de inputs, mas tambem não podemos confiar nas bibliotecas ORM. Algumas falhas podem acontecer.

Cross Site Scripting (XSS)

O ataque consiste em alojar um script malicioso.

Histórias

Paypal

Um exemplo foi o PayPal XSS em 2006. Basicamente os atacantes enviavam um email a vítima afirmando que a conta deles tinha sido comprometida. No fim pediam para o usuário entrar num link hosted pelo próprio website e que depois era redirecionado para outro. Depois eles pediam uma serie de dados, sendo que o link parecia ser legítimo.

SamyWorm

Era uma rede social em que quando a pessoa visualizava o perfil do samy ele basicamente virava amigo dele.

Prevenção (Content security policy)

- Scripts inline não são executados
- Executar scripts provindos apenas de sites autorizados (white-list)
- frame-ancestor ‘none’ : um website não pode ser framed por outro.

Subresource Integrity

Ainda é possível que a white-list seja comprometida! Para evitar isto podemos adicionar uma hash de `integrity` na nossa white-list. Dessa forma se a white-list for comprometida a hash é perdida e o script não é aceito.

```
<script src="https://code.jquery.com/jquery-2.1.4.min.js"
integrity="sha384-R4/ztc4ZlRqWjqIuvf6RX5yb/v90qNGx6fS48N0tRxi
GkqveZETq72KgDVJCp2TC"
crossorigin="anonymous"></script>
```

Perguntas

1) O que é CSRF? Cite um exemplo e medidas para impedí-lo.

Um CSRF é um cross site request forgery. Acontece quando o atacante faz com que o utilizador faça um request a um website a partir de uma fonte suspeita. Um caso comum é o user ter salvo no browser um cookie de sessão para um website específico e, ao clicar numa image, por exemplo, que possui um request a outro website (cujo o utilizador possui sessão iniciada), o request poderá ser aplicado.

Uma das formas de impedir o CSRF é adicionar secret tokens juntos a formulários, impedir requests a partir de outras origens, etc.

2) O que é um SQL injection e como podemos previní-lo?

O **SQLi** é quando o atacante consegue mudar operações na database a partir do seu input. Pode-se previní-lo realizando queries parametrizadas (não permitem alterações já que são pré-compiladas) e o uso de bibliotecas **ORM** que abstraem o uso da base de dados e o uso de queries. No entanto, a próprias bibliotecas **ORM** podem estar sujeitas a vulnerabilidades.

2.1) O que é uma parameterized query?

São queries pré-compiladas onde são apenas fornecidos os parâmetros dos users. Estas queries não podem ser alteradas já que são pré-compiladas.

2.2) O que são bibliotecas ORM?

São biliotecas que abstraem o uso de base de dados.

3) O que é XSS? Qual a diferença de um reflected XSS para um stored XSS?

É Cross Site Script. O utilizador consegue criar um script que é executado pelo website. O reflected XSS não é guardado pela base de dados. Alguem pode clicar num link que contém um script o qual envia a cookie do user para outro website.
O stored XSS acontece quando o atacante consegue guardar o script na base de dados do website e quando o administrador. Um exemplo disto é o Samy Worm, que guardou um código html no perfil.

3.1) Quais são as medidas atuais para a prevenção de um XSS?

Podemos evitar o XSS utilizando a content security policy, em que:

- Scripts inline não são executados
- Apenas scripts de white-list são aceitos

- Podemos adicionar ainda opção em que o site não pode ser framed.

O que ainda pode acontecer é uma source da whitelist ser comprometida. Então podemos adicionar a hash do que se esta esperando receber. Assim se a source for alterada, a flag será mudada.