

A7 - Cross-Site Scripting



globo
.com



GTS 33 | PA 05/2019



Agenda - A7 - Cross-Site Scripting (XSS)



1. Do que se trata?
2. XSS Refletido
3. XSS Persistente
4. XSS DOM based
5. Como se proteger?
6. Gossip World

Do que se trata?



Cross-Site Scripting (XSS) é uma técnica de injeção de código javascript através de algum parâmetro da aplicação. Em geral, o usuário envia um código malicioso para o servidor, este código não é tratado e, ao retornar para o usuário, é executado em seu navegador.



https://www.site.com/

SITE

search

[Redacted content]

[Redacted content]



https://www.site.com/?s=<script>alert(1);</script>

SITE

<script>alert(1);</script>

██
████████████████████████████████████
████████████████████████████████████
██

██
████████████████████████████████████
████████████████████████████████████
██



https://www.site.com/?s=<script>alert(1);</script>

SITE

Resultados para: `<script>alert(1);</script>`

É uma tag script!
Devo executar!



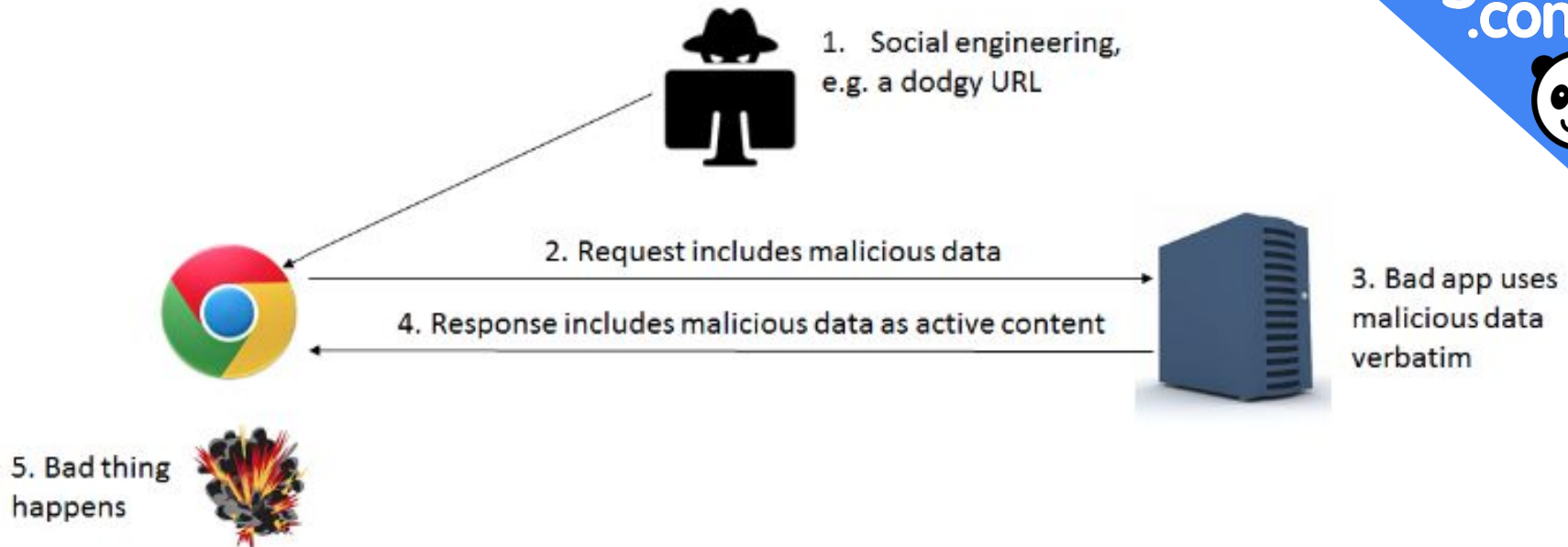
https://www.site.com/?s=<script>alert(1);</script>

SITE

Resulta

1

The diagram illustrates a reflected XSS attack. At the top, a URL is shown: `https://www.site.com/?s=<script>alert(1);</script>`. Below this, a box labeled "SITE" represents the web application. Inside the site, a search result is displayed. The word "Resulta" is on the left, and a rounded rectangular box contains the number "1". This represents the server reflecting the malicious payload back to the user's browser.



```
http://www.example.com/search.asp?q=<script>a  
    lert(1);</script>
```

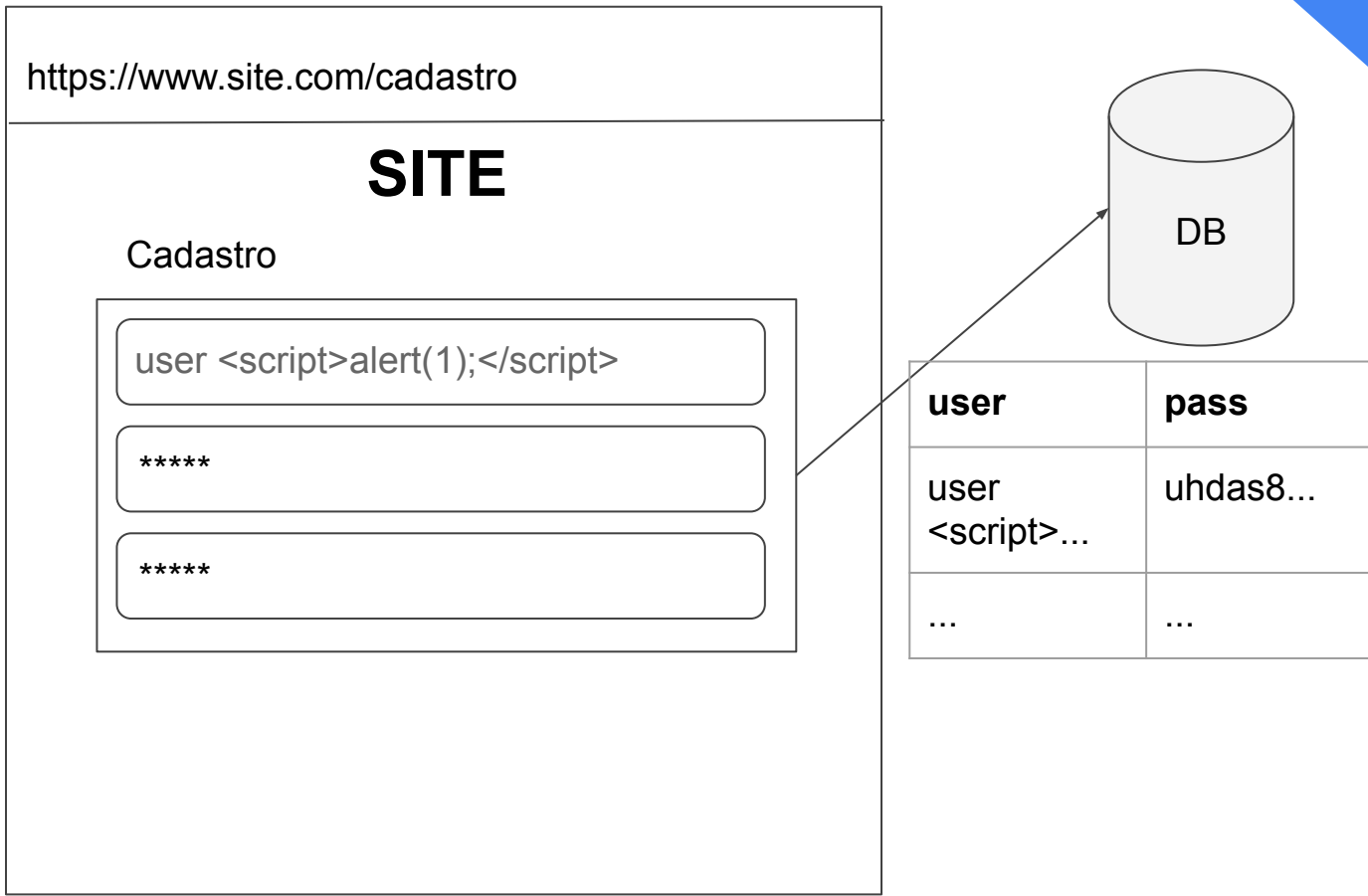



https://www.site.com/cadastro

SITE

Cadastro

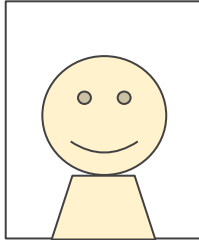
user <script>alert(1);</script>





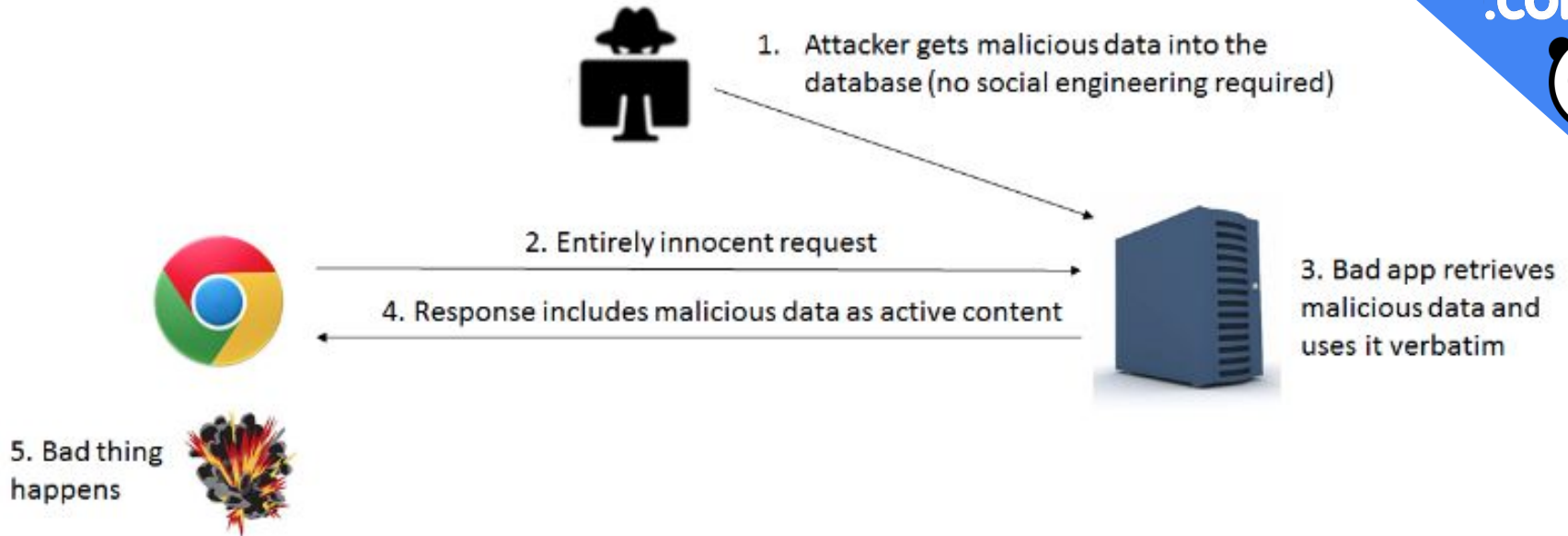
https://www.site.com/perfil?id=23241332

SITE



user `<script>alert(1);</script>`

É uma tag script!
Devo executar!



```
http://www.example.com/profile.asp
```



```
<script>
document.write("<b>Current URL</b> : " + document.baseURI);
</script>
```

```
http://www.example.com/test.html#<b>script>alert(1);</script>
```

Como prevenir?



1. Evite sempre **inserir dados** do cliente diretamente em **scripts**, em comentários HTML, em nomes de atributos e tags ou diretamente no CSS.
2. Use **frameworks** que automaticamente **escapem XSS**.
Exemplos: Flask, Django, React.
3. **Recomendações:**
 - a. bleach - <https://github.com/mozilla/bleach/> (Python)
 - b. DOMPurify - <https://github.com/cure53/DOMPurify> (Javascript)
 - c. html/template - HTMLEscapeString (Go)



Gossip World

This is a simple Flask app that contains an example of multiple Cross-site Scripting vulnerabilities.

Gossip World!

Login

[Create a new free account!](#)



Narrativa do ataque



1. Entrar na pasta da app

```
$ cd owasp-top10-2017-apps/a7/gossip-world/
```

2. Inicializar o container

```
$ make install
```

3. Acessar a página

```
localhost:3001
```



A7 - Cross-Site Scripting