Programação Orientada a Serviços

Prof. Diego Cirilo

Aula 13: Autenticação

Autenticação

- Autenticação
  - Verifica identidade do usuário para o serviço.
- Autorização
  - Verifica as permissões de acesso do usuário.
- Necessária para controle de acesso às APIs
- Basic auth (HTTP)
- API Keys
- OAuth

Basic Auth

- Usa o cabeçalho HTTP
- Necessário passar usuário e senha no request
- Mais simples
- Pouco segura

Exemplo API Github

```
import requests
from requests.auth import HTTPBasicAuth
from getpass import getpass

user = input("user: ")
password = getpass()

response = requests.get('https://api.github.com/user',
            auth = HTTPBasicAuth('dvcirilo', password))

print(response.text)
print(response)
```

Tarefa

- Faça um cliente que consiga listar os seguidores do usuário logado e seguir/parar de seguir um usuário no Github pelo terminal.

API Keys

- Serviço disponibiliza as chaves
- Chave é passada no cabeçalho HTTP

Exemplo SUAP

```
import requests
from getpass import getpass

api_url = "https://suap.ifrn.edu.br/api/"

user = input("user: ")
password = getpass()

data = {"username":user,"password":password}

response = requests.post(api_url+"v2/autenticacao/token/", json=data)
token = response.json()["access"]
print(response.json())

headers = {
    "Authorization": f'Bearer {token}'
}

print(headers)

response = requests.get(api_url+"v2/minhas-informacoes/meus-dados/", headers=headers)

print(response.text)
print(response)
```

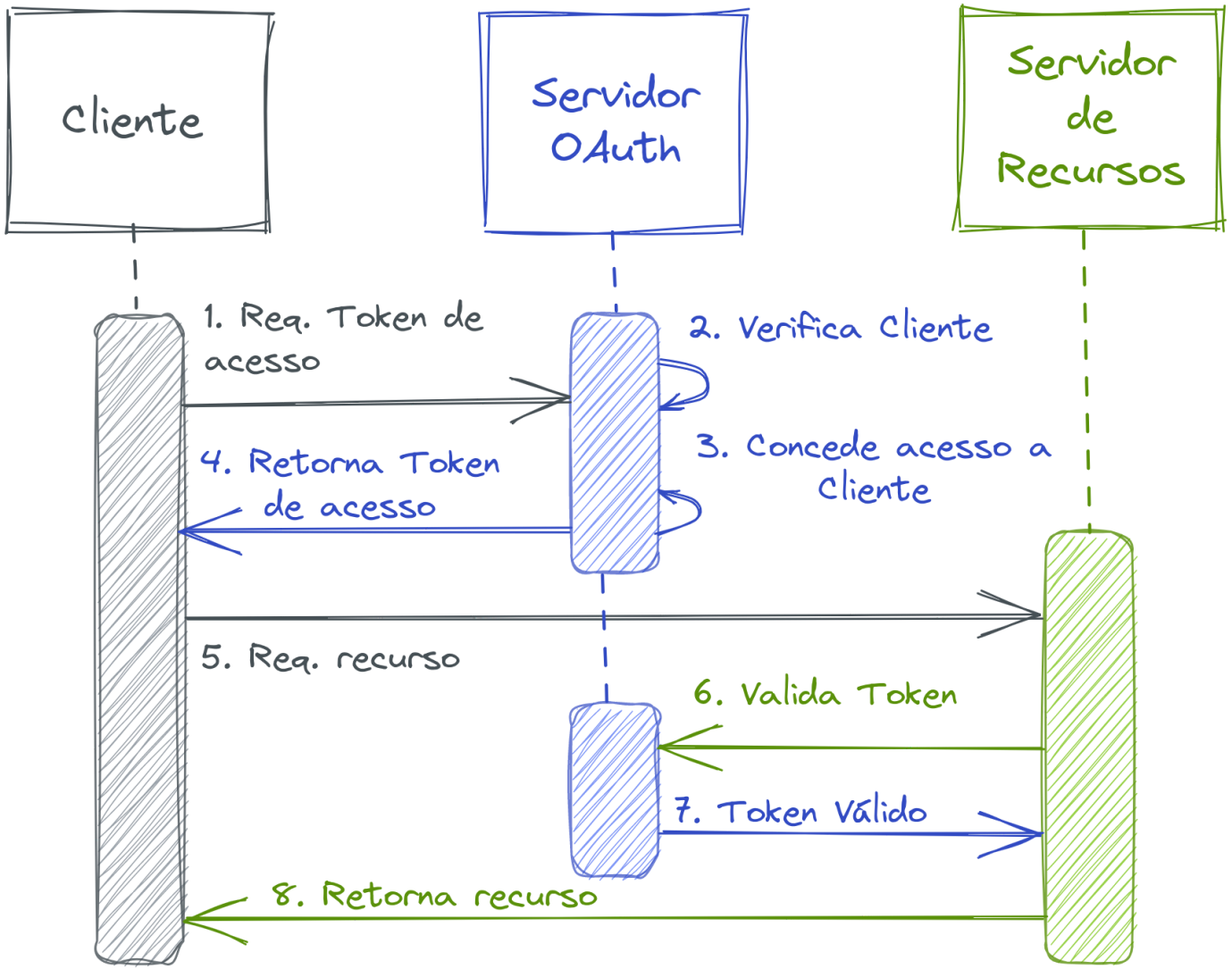
Tarefa

- Faça um cliente que se autentique com as API Keys do SUAP e retorne o boletim do aluno formatado no terminal.
- A formatação deve ser no estilo tabela, indicando a disciplina e notas de todas as unidades.

OAuth

- Open Authorization
- Padrão aberto de serviços de autorização
- Usado na maioria dos serviços grandes, ex. Google, Facebook, ~~Twitter~~, SUAP, etc.

OAuth


Cliente
Servidor OAuth
Servidor de Recursos
1. Req. Token de acesso
2. Verifica Cliente
3. Concede acesso a Cliente
4. Retorna Token de acesso
5. Req. recurso
6. Valida Token
7. Token Válido
8. Retorna recurso

OAuth

- Aplicações devem ser registradas no serviço
  - Client ID e Client Secret
  - Redirect URI
  - Client Type, Grant Type, etc.
- API Disponibiliza:
  - Access Token - URL
  - Authorization URL
  - Scopes, etc.
- Aplicação deve guardar token (session/cookies) para as próximas requisições.

Tipos de cliente

- *Public*: aplicações onde não é possível esconder o Client Secret (Front-end e Apps).
- *Confidential*: aplicações "*client-side*".

Grant Flows

- *Authorization Code*
- *Proof Key for Code Exchange* - PKCE
- *Implicit*
- *Device Code*
- *OIDC*

Riscos no OAuth

- Client Impersonation

Acesso OAuth do SUAP no Flask

- Registre sua aplicação em https://suap.ifrn.edu.br/api/
- Authorization grant type: `authorization-code`
- Redicert URIs:

  `http://localhost:5000/login/authorized`
- Guarde o Client ID e Client Secret
- Exemplo

Acesso OAuth do SUAP com JavaScript

- Registre sua aplicação em https://suap.ifrn.edu.br/api/
- Authorization grant type: `authorization-code`
- Redicert URIs:

  `http://localhost:8888/login/authorized`
- Guarde o Client ID e Client Secret
- Exemplo

Tarefa Final - Segunda Unidade

- Faça um cliente do SUAP com autenticação OAuth que apresente o perfil do usuário com foto e permita a visualização dos boletins, com seleção de ano/semestre.
- Caprichem no front-end
  😄

Referências

- https://developer.okta.com/blog/2022/06/01/oauth-public-client-identity
- https://auth0.com/docs/get-started/applications/confidential-and-public-applications
- https://www.oauth.com/playground/
- https://www.youtube.com/watch?v=CHzERullHe8

Dúvidas?