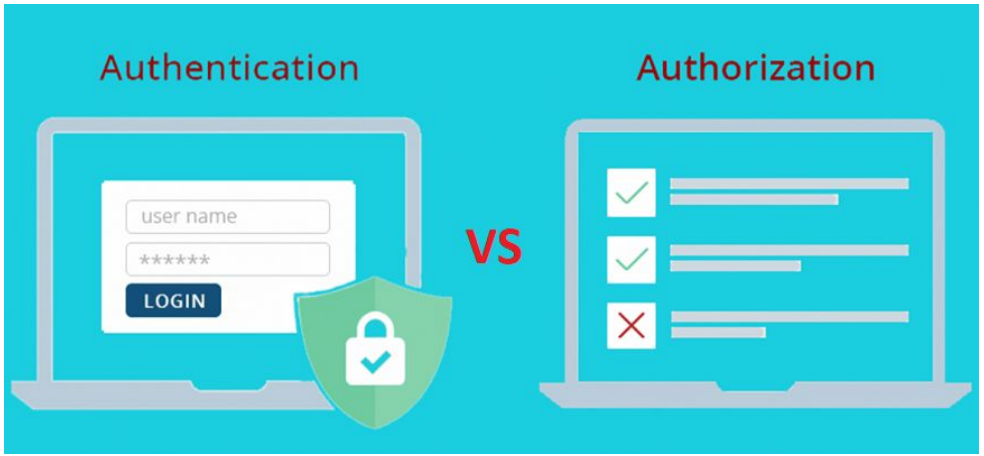
Funcionalidades avançadas e segurança em APIs RESTful

José Glauber UFCG 2024.1

Qual a diferença entre autenticação e autorização?



importância?

Autenticação

O objetivo geral é identificar se o usuário é quem diz ser.

- Administrador do sistema
- Usuário comum
- Professor em um sistema acadêmico

email? senha?

Autenticação

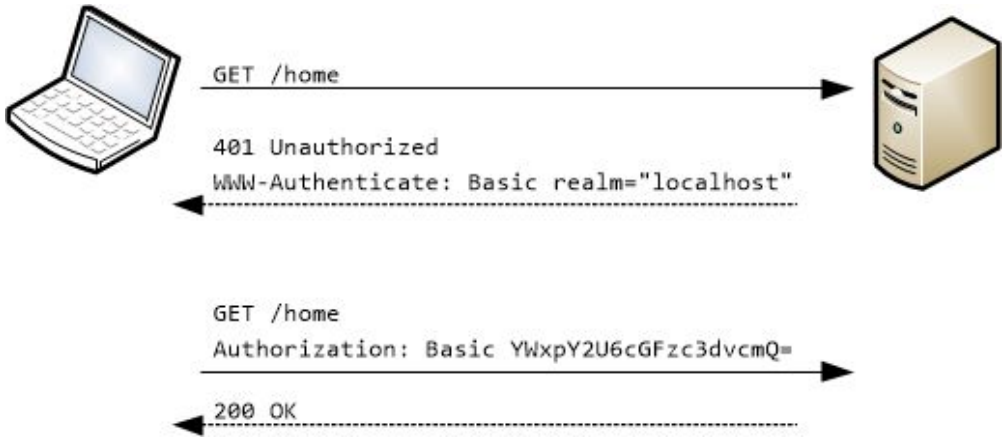
Alguns tipos de autenticação..

- Autenticação baseada em senhas
- Autenticação baseada em Multi-fator (MFA)
- Autenticação baseada em token
- OAuth 2.0

Autenticação baseada em senhas

Neste modelo a segurança depende do usuário ao definir nome de usuário e senha

ex: Basic Auth



cabeçalho de autorização com base64

Autenticação baseada em senhas

De forma geral é a autenticação mais básica e simples de ser implementada.

Quando utilizar?

Simplicidade…

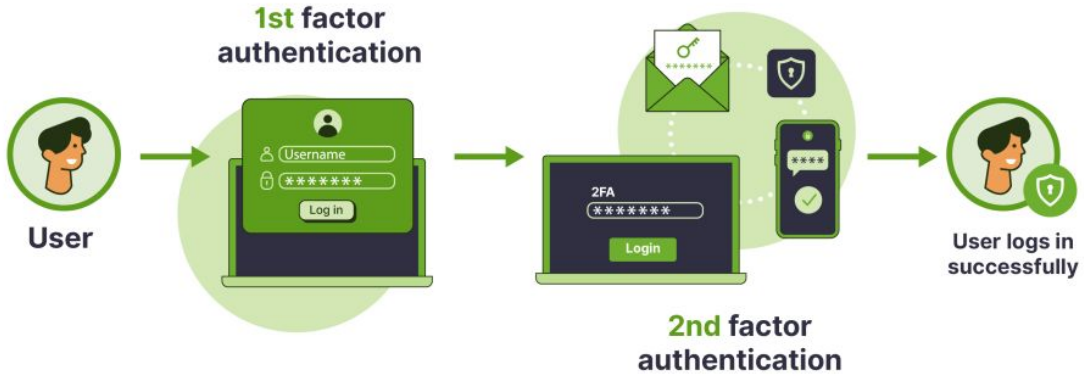
Está transportando dados sigilosos?

Problema principal:

Bastante suscetível a ataque de hackers

Autenticação de dois fatores (2FA)

Este modelo traz mais uma camada de segurança ao acesso das nossas aplicações. A 2FA dá aos negócios a capacidade de monitorar e ajudar a proteger suas informações e redes mais vulneráveis.



Autenticação de dois fatores (2FA) - Benefícios

- Não é necessário usar tokens físicos para autenticação de dois fatores (2FA), pois esses dispositivos podem ser perdidos. Hoje, há métodos 2FA mais práticos e convenientes.

- Geradores de senhas são mais seguros e eficientes que senhas tradicionais, já que cada senha gerada é única.

- Limitar o número de tentativas de senha ajuda a impedir que hackers acessem dados confidenciais.

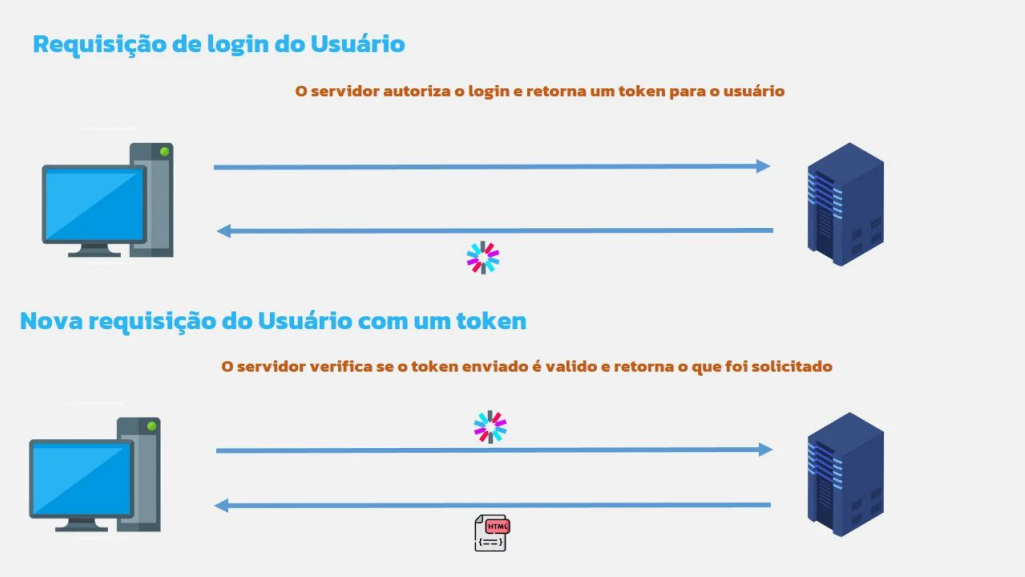
- O processo de segurança é fácil de gerenciar e usar.

Autenticação por sessão

Um dos primeiros métodos de autenticação. Após a requisição feita pelo cliente o servidor, por sua vez, cria uma sessão em sua memória ou banco e devolve a informação de usuário através de um cookie com o identificador da sessão criada.

Autenticação por token

Após ter login e senha validados pelo servidor, é criado um token e que permitirá o acesso à algum recurso. O padrão mais adotado é o JWT (JSON Web Token).



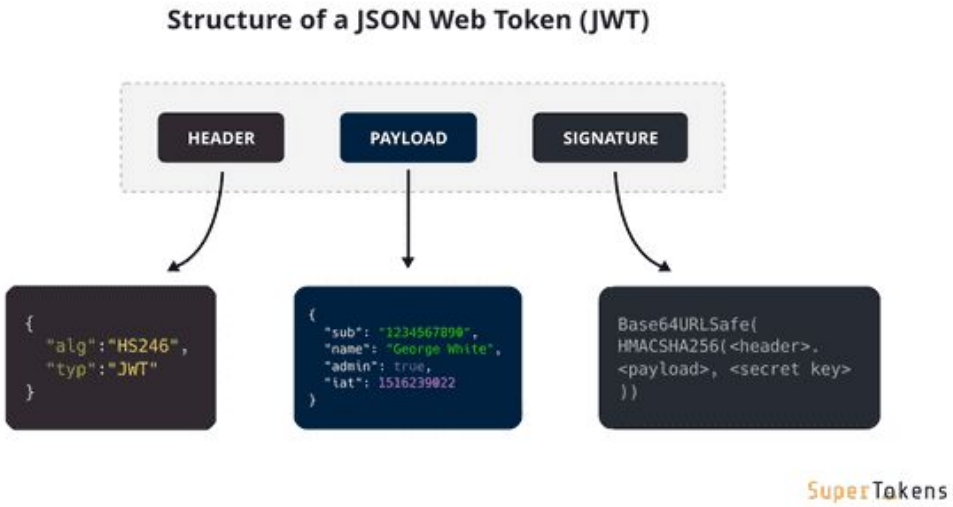
Autenticação por token

O JWT contém informações relevantes sobre o usuário, identificação, permissões e etc. Esse token é assinado digitalmente a fim de garantir a sua autenticidade e integridade.

O servidor não precisa armazenar nenhum estado relacionado a sessão do usuário. O token é incluído automaticamente em cada requisição subsequente ao servidor, o que faz com que não exista a necessidade de consultar um armazenamento de sessões.

Autenticação por token - JWT

É um padrão da indústria definido pela RFC7519 que tem como objetivo transmitir ou armazenar de forma compacta e segura objetos JSON entre diferentes aplicações.



Autenticação por token - JWT

Vantagens:

1. Independência do estado do servidor;
2. Tokens carrega consigo todas as informações referentes aos usuários;
3. Revogação eficiente de tokens;
4. Interoperabilidade, portabilidade e flexibilidade;

Desvantagens:

1. Necessidade de estratégias eficazes para revogar tokens;
2. Estratégias de sincronização e gerenciamento de tokens em diferentes dispositivos precisam ser cuidadosamente implementadas;
3. Prazo de validade para tokens;

Autenticação por sessão X token

Sessão: estado mantido pelo servidor, podendo ser armazenado em um bd ou na memória (Stateful).

- Limites de hardware;
- Muitas chamadas ao garbage collector;

Tokens: não são mantidas no servidor (Stateless). Token possui tudo que é importante.

- Id do usuário;
- Assinatura;
- Data de expiração;
- Método de autenticação utilizado.

Autorização

O objetivo geral é determinar permissões para acesso a determinados recursos, de usuários já autenticados.

Acessar configurações da conta

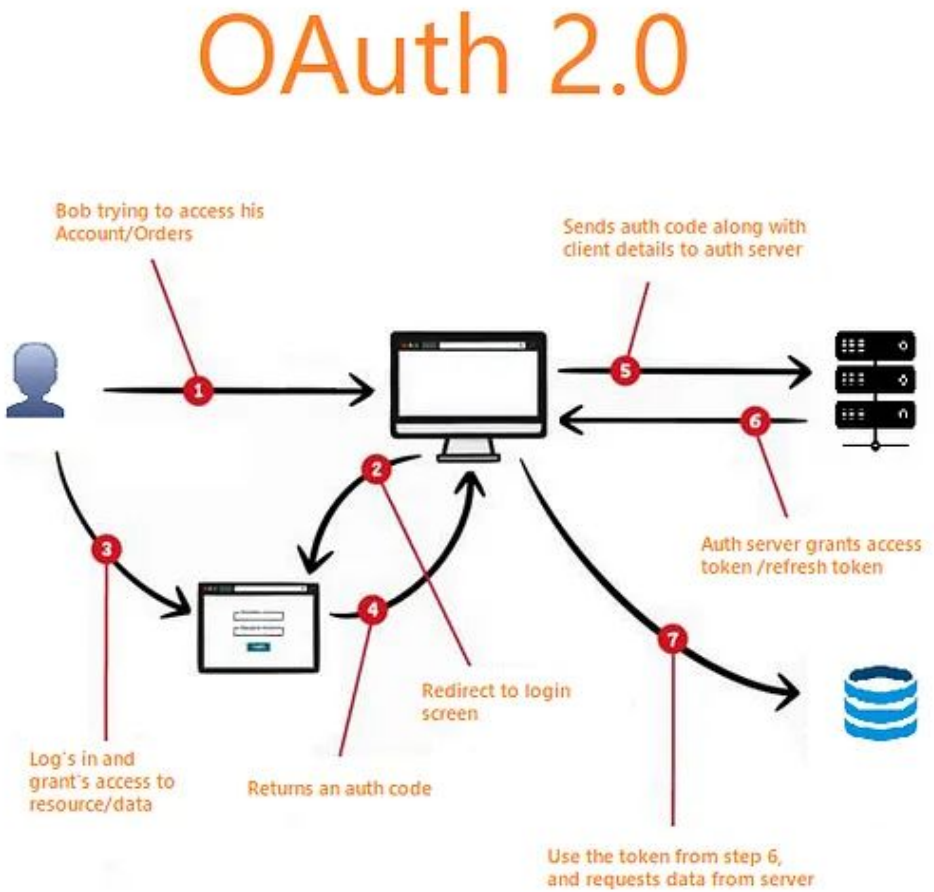
excluir e-mails

acessar lista de documentos dos usuários..

Autorização

OAuth 2.0 -> é um protocolo de autorização que permite que uma aplicação acesse recursos em nome de um usuário sem que o usuário precise fornecer suas credenciais diretamente à aplicação

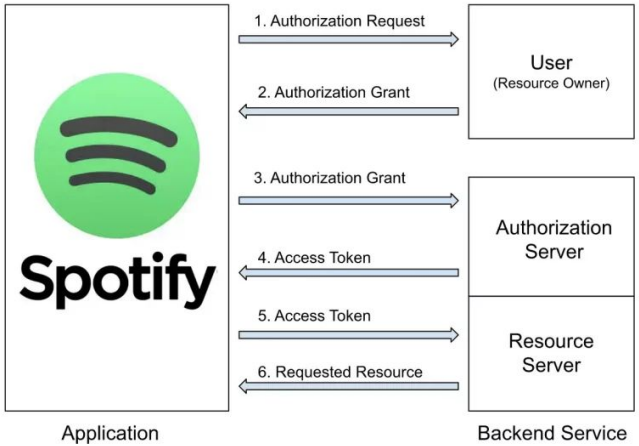
é um protocolo de autorização e NÃO um protocolo de autenticação



Autorização

Problema antes do OAuth -> Usuários tinham que compartilhar suas senhas com diferentes aplicativos para permitir o acesso aos seus dados.

- Risco de roubo de senha e violações de segurança;
- Comprometimento de todas as contas vinculadas àquela senha;



Quando ele é útil?

OAuth 2.0 é útil em situações em que os usuários desejam conceder acesso a seus recursos a aplicativos de terceiros, mas não desejam fornecer suas credenciais ou senhas.

é bastante útil para garantir que os usuários mantenham o controle sobre seus recursos e possam revogar o acesso a qualquer momento

Integração OAuth 2.0 com JWT

Quando OAuth 2.0 é usado com JWT, o servidor de autorização pode emitir um JWT como um token de acesso. Esse token pode então ser usado pelo cliente para acessar recursos no servidor de recursos.

como funciona?

1. **Autenticação do Usuário:** O usuário faz login no servidor de autorização (ex.: Google) e concede permissão ao cliente.
2. **Autorização:** O servidor de autorização emite um Access Token (um JWT) que autoriza o cliente a acessar os recursos do usuário.
3. **Acesso ao Recurso:** O cliente usa o JWT (Access Token) para acessar os recursos protegidos no servidor de recursos.
4. **Validação:** O servidor de recursos valida o JWT para verificar a autorização antes de conceder acesso.

Benefícios OAuth 2.0

- Segurança aprimorada
- Controle granular de permissões
- Experiência de usuário melhorada
- Suporte a diversos tipos de clientes
- Interoperabilidade

Benefícios JWT

- Autenticação e Autorização seguras
- Eficiência e desempenho
- Flexibilidade e escalabilidade
- Independência da plataforma e linguagem
- Suporte para expiração e renovação

OAuth 2.0 e JWT

	JWT	OAuth
Purpose	Token-based authentication mechanism for transmitting claims	Protocol for authorization and authentication in web and mobile apps
Centralized Server	None	Authorization server
Used For	Authentication and authorization	Authorization, not authentication
Applications	Single-page apps, mobile apps	Apps that rely on external APIs or services
Relationship	Directly between parties	Between third-party apps and resource owners
Authentication	Yes	No
Authorization	Yes	Yes

Referências

1. https://oauth.net/2/
2. https://jwt.io/introduction
3. https://openid.net/developers/how-connect-works/#:~:text=OAuth%202.0%2C%20the%20substrate%20for,structures%20when%20signatures%20are%20required.
4. Designing Web APIs: Building APIs That Developers Love - Designing Web APIs: Building APIs That Developers Love
5. RESTful Web APIs - Leonard Richardson, Mike Amundsen, Sam Ruby
6. O que é a 2FA (Autenticação de Dois Fatores)? | Segurança da Microsoft