Open Authentication

Authored by Chris Bohnet

Agenda

1. What is OAuth?
2. How does OAuth Authorization Code grant type work?
3. How does OAuth Implicit grant type work?
4. How does OAuth Password Credentials grant type work?
5. How does OAuth Client Credentials grant type work?
6. OAuth 1.0 vs OAuth 2.0
7. Why is OAuth popular?
8. Diagrams

What is OAuth?

(Open Authentication)

It's an open standard for access delegation.

It's a way for users or applications to get access to other applications or sites.

It works over HTTP and authorizes with access tokens.

Access tokens are issued by an authorization server by permission of the resource owner.

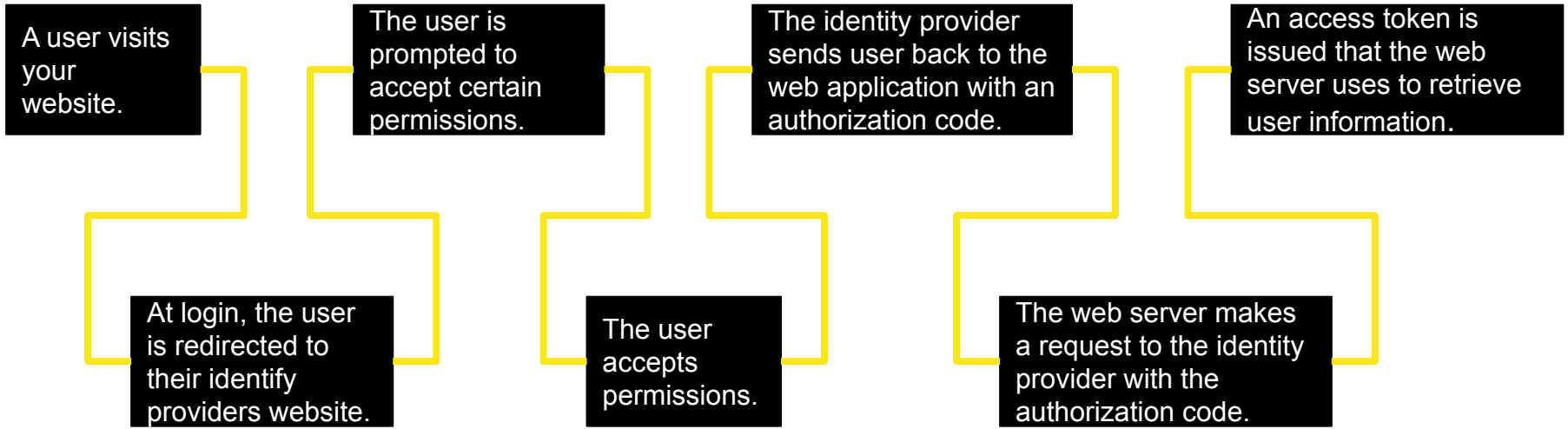
Access tokens are then used to get access to the protected sites on the resource server.

There are 4 OAuth grant types: Authorization code, implicit, password credentials and client credentials.

There are 2 versions of OAuth: OAuth 1.0 & OAuth 2.0.

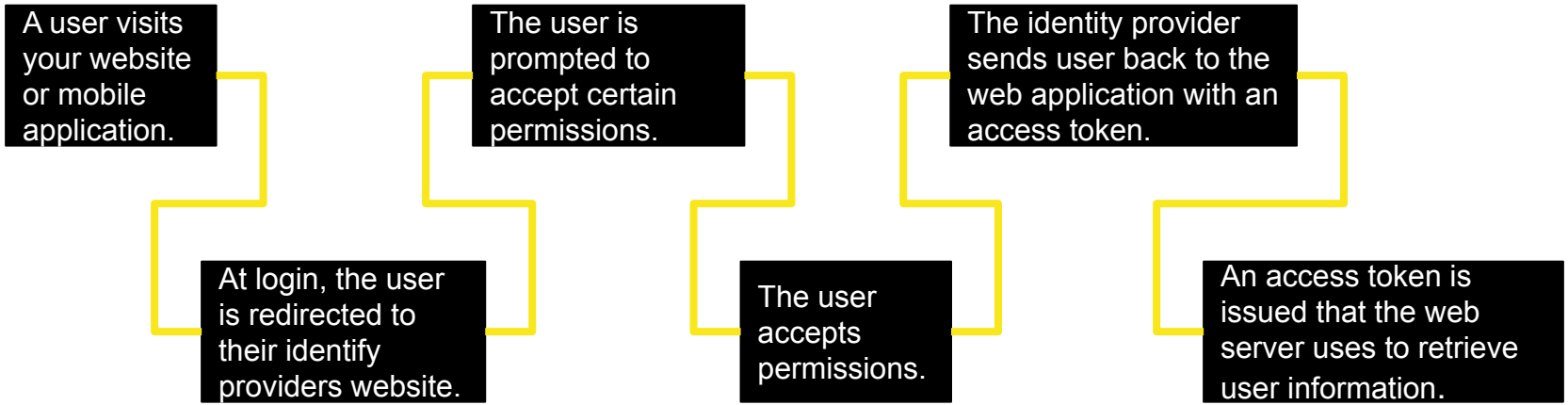
HOW DOES THE OAuth Authorization Code Grant Type WORK?

- Used on web servers when building web applications with server side code that is **not** public.



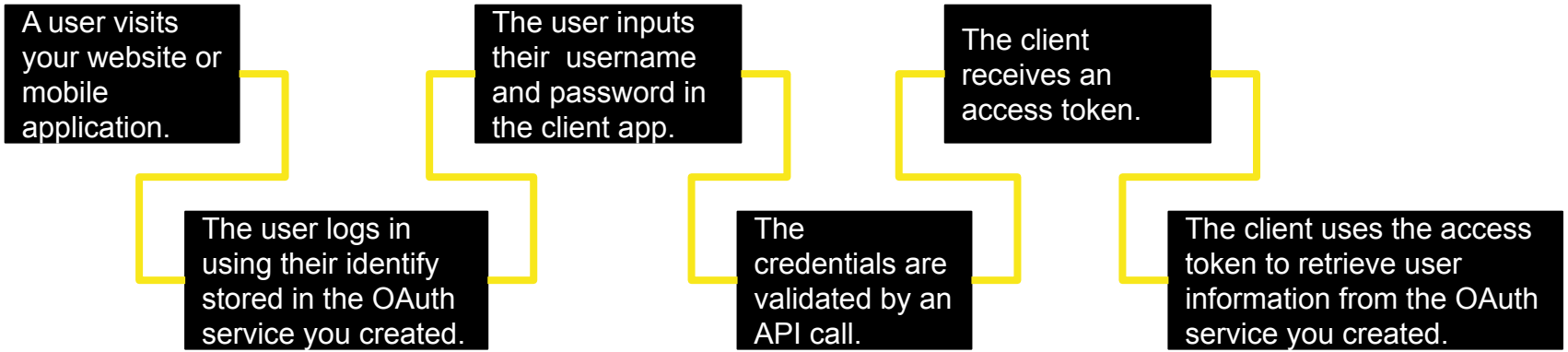
HOW DOES THE OAuth Implicit Grant Type WORK?

- Used for client-side web applications that do not have a server-side counterpart or mobile applications that use mobile web browsers.
- Doesn't require any secret key information to be stored so users can log in without knowing the secret



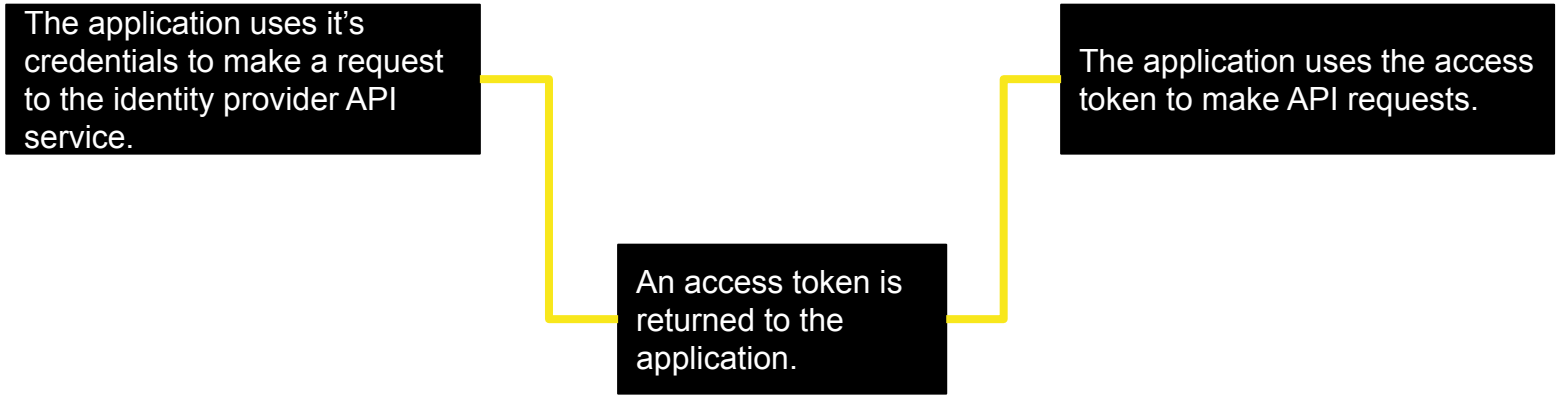
HOW DOES THE OAuth Password Credentials Grant Type WORK?

- Used when integrating with an OAuth service you created.
- Used with first class or mobile web applications that want to simplify the authorization workflow and only use a username and password.
- Used to authenticate users for **only** your native apps since you don't want to risk a third-party vendor capturing the credentials.



HOW DOES THE OAuth Client Credentials Grant Type WORK?

- Meant to be used for application code that performs non-user related tasks and doesn't interact with user data.
- Used for background processes or tasks such as updating application metadata.



OAuth 1.0 vs OAuth 2.0

For server-to-server communication.

Uses cryptography making it more difficult to implement.

Transport independent: Does not need to use HTTPS.

Messages are each cryptographically signed & if one is done improperly, the whole transaction fails.

Web workflow uses only.

Defines 3 roles: consumer, user and service provider. Does not separate service provider & authorization server roles.

For user/browser/device-to-server communication.

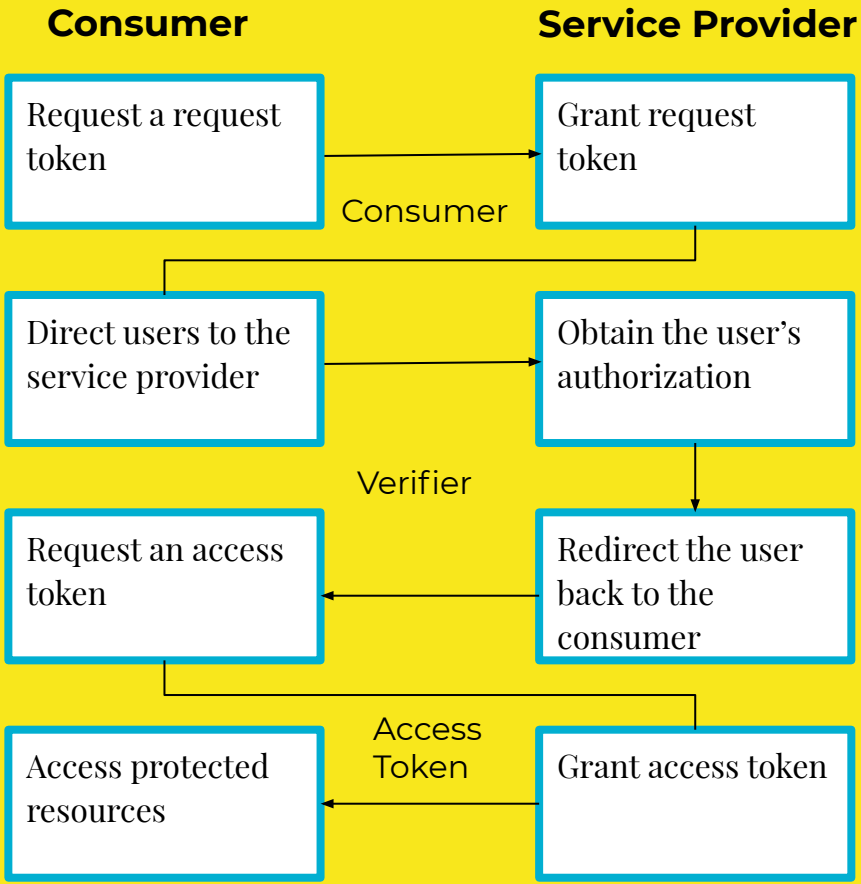
Less complex with a well-defined framework centered around bearer tokens making it easier to implement.

Transport dependent: Uses HTTPS/TLS.

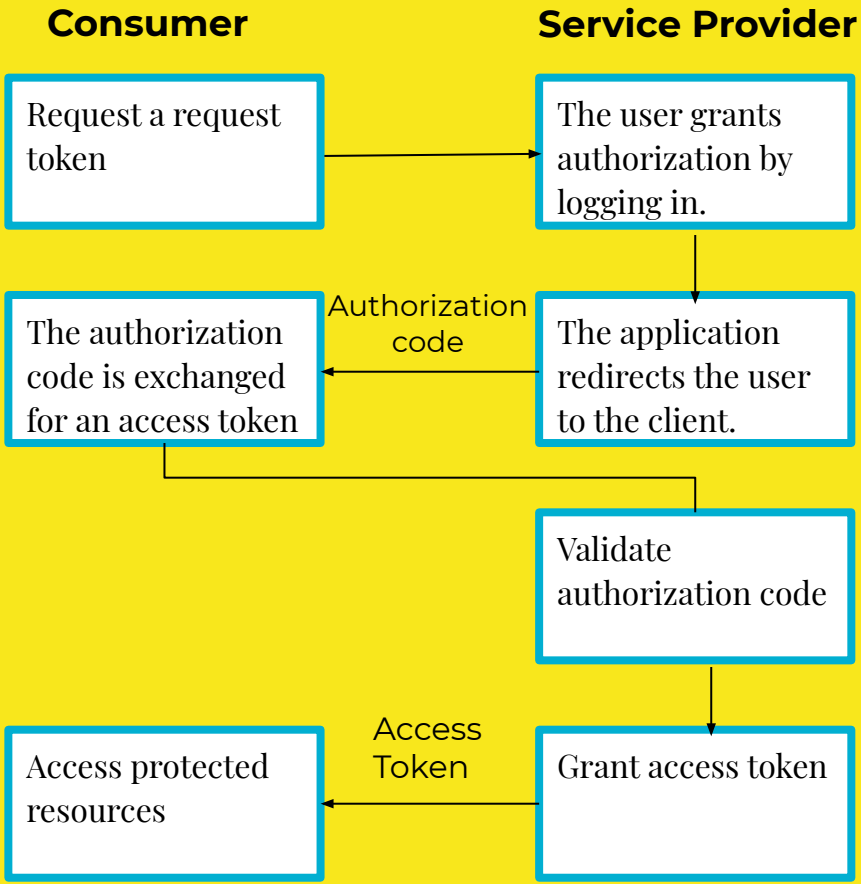
Decoupling of resource & authorization request handling.

Both web workflows and non-web client uses.

Defines 4 roles: client, resource owner, resource server & authorization server.



OAuth 1.0 Flow



OAuth 2.0 Flow

WHY IS OAUTH POPULAR?

OAuth is an industry standard authentication protocol.

Authentication is managed by a third-party provider so your application does not need to collect and store data.

OAuth saves time on developing authentication, it's easier to maintain and configure and less data is stored on servers.

No need for support for password renewal, forgotten passwords or authentication of users.

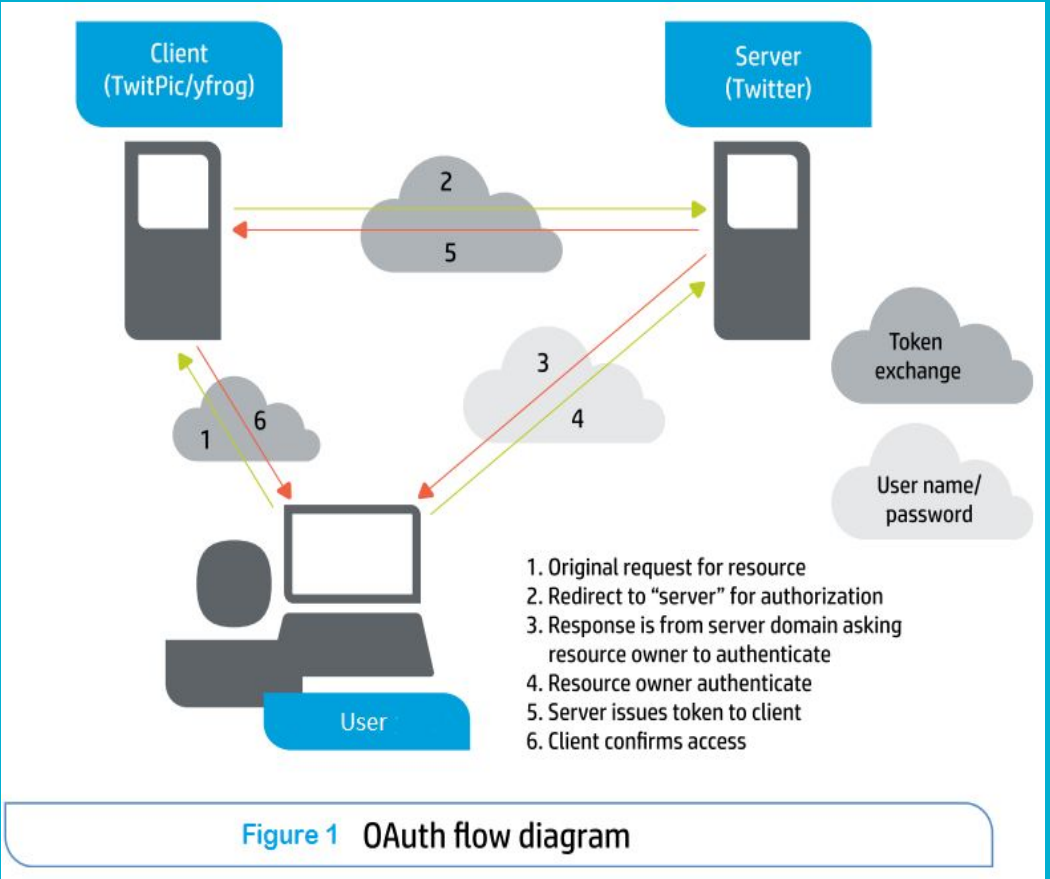
Lower risk of security breach since authentication takes place at the provider and not in application

More secure since the application only receives an encrypted OAuth token and not user credentials.

Fewer passwords to remember for the user.

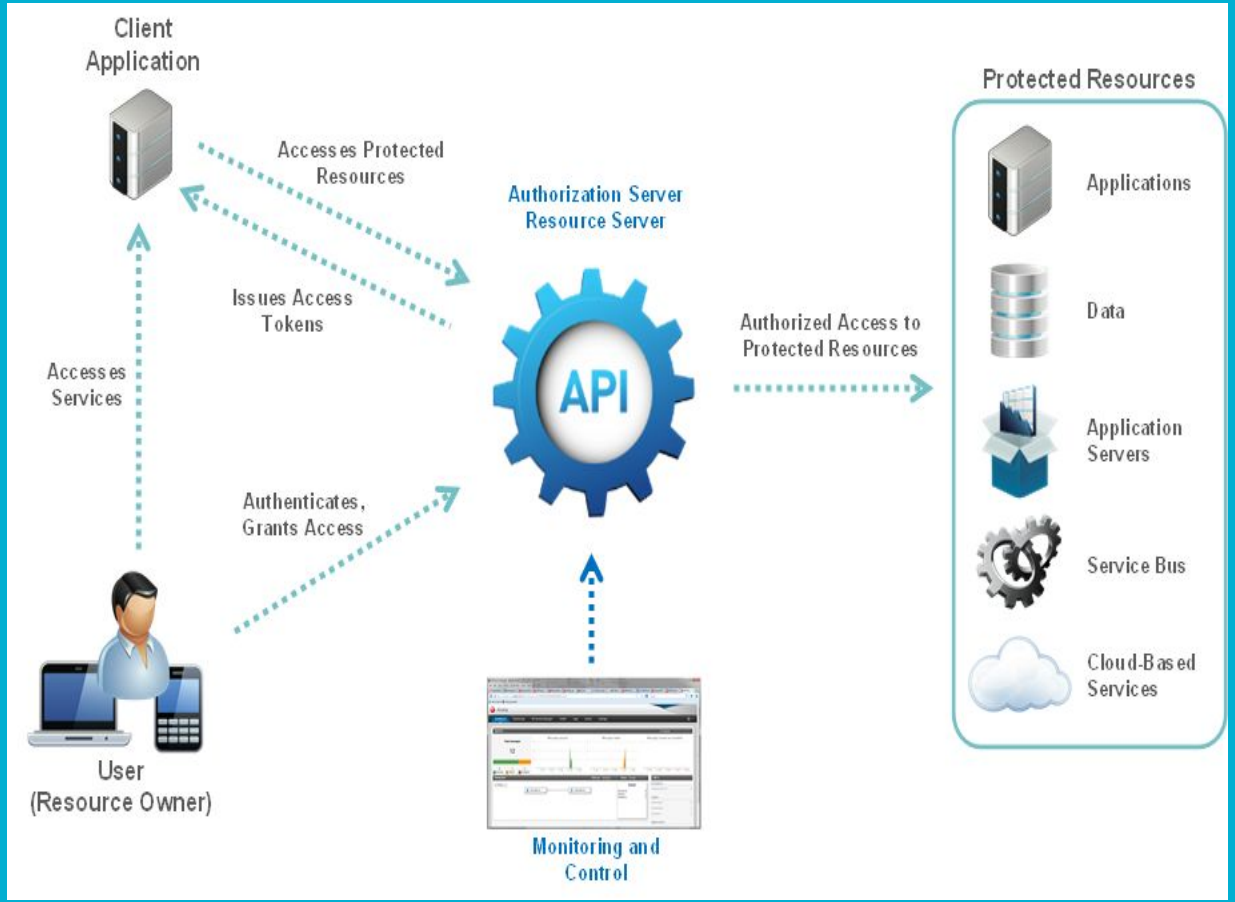
Users can prevent access to the application from the OAuth provider.

OAUTH FLOW DIAGRAM



Hewlett Packard Enterprise, Figure 1 OAuth flow diagram, as referenced at https://community.hpe.com/t5/image/serverpage/image-id/27081i8E0E5881BA360900?v=v2

OAuth API End To End Data Flow Diagram



Oracle, Diagram of roles of an API Gateway with OAuth, as referened at https://docs.oracle.com/cd/E50612_01/doc.11122/oauth_guide/content/oauth_intro.html

RESOURCES

Stormpath, “What the Heck is OAuth?”, as referenced at https://stormpath.com/blog/what-the-heck-is-oauth

Synopsys, “What’s the difference? OAuth 1.0 vs OAuth 2.0”, as referenced at https://www.synopsys.com/blogs/software-security/oauth-2-0-vs-oauth-1-0/

WikiTechy, “OAuth 1.0 vs OAuth 2.0”, as referenced at https://www.wikitechy.com/tutorials/oauth/oauth-1-dot-0-vs-oauth-2-dot-0

Fuzzy Blog, “Benefits of using OAuth as your login provider”, as referenced at https://fuzzzyblog.blogspot.com/2010/09/benefits-of-using-oauth-as-your-login.html

O’Reilly, “Securing the Login with OAuth2 and OpenID Connect”, as reference at https://www.oreilly.com/library/view/identity-and-data/9781491937006/ch04.html

Community HPE, “Low hanging threads to OAuth Security- Hewlett Packard”, as referenced at https://community.hpe.com/t5/image/serverpage/image-id/27081i8E0E5881BA360900?v=v2

Oracle, “Introduction to API Gateway OAuth 2.0”, as referenced at https://docs.oracle.com/cd/E50612_01/doc.11122/oauth_guide/content/oauth_intro.html

Microsoft, “Authentication and EWS in Exchange ”, as refereneced at https://docs.microsoft.com/en-us/exchange/client-developer/exchange-web-services/authentication-and-ews-in-exchange