Understanding JSON Web Tokens

In this lesson we will learn about JSON Web Tokens, their structure and some of the elements you will usually find in them.

What is a JSON Web Token (JWT)?

A JSON Web Token, also known as JWT (jot), is a compact, URL-safe means of representing claims to be transferred between two parties. These set of claims can be digitally signed to protect their integrity, at which point the JWT payload is part of a structure known as a JSON Web Signature, or JWS.

JWTs can also be encrypted, but in this course we will focus on signed tokens only.

JSON Web Token (JWT) structure

Let's now look at the structure of a JWT. A JWT typically looks like this, which at first glance is just a lot of gibberish. However, there are two periods among all these characters that divide the JWT in three well known parts:

- **The Header**, which contains information about the type of token and how it was signed
- **The Payload**, which contains all the transmitted claims
- **And, the Signature,** which is calculated from the encoded header and payload, an added secret and the hashing algorithm specified in the header. It is used to verify that the message wasn't changed along the way

JWT Header

In the header of a JWT we will typically find at least these two fields:

- **alg**. The algorithm used to sign the token. RS256 for instance would mean the JWT was signed using RSA signature with SHA-256.
- **typ:** The type of token. The type here is usually just JWT but in the case of OAuth access tokens the value would be at+jwt

The header can also include an optional **kid** field, which is an identifier of the key used to sign the token, in the case the identity provider has more than one key available.

JWT Payload

Like mentioned before, the JWT payload contains the many claims that are being transmitted via the token. Some are well known and some are more specific to specific identity providers. Here for a few of the ones you will find during this course:

Field	Description	Sample Value
iss	Issuer. Who created and signed the token	https://localhost:5003
aud	Audience. Who the token is intended for.	Catalog, Inventory, postman
client_id	The identifier of the OAuth client	postman

sub	Subject. The principal that is the subject of the JWT. This is the Id of the signed in user in the Identity DB.	ccf2b778-2cc9-4478-82ca-2780a82f63ce
scope	The type of access granted to the client.	openid, profile, catalog.readaccess, inventory.fullaccess
at_hash	Access Token hash value. Can be used to correlate the id token with the access_token	dzJsGOPnXdR-haI8OtRSjA
amr	Authentication Methods References. Array of strings that identify the authentication methods used in the authentication.	“pwd”
nbf	Not valid before. The time before which the JWT MUST NOT be accepted for processing. This and all date/time values are defined in seconds since Unix epoch, which is the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT), not counting leap seconds	1615097311
exp	Expiration time after which the JWT MUST NOT be accepted for processing	1615100911
iat	identifies the time at which the JWT was issued	1617140901
auth_time	The time at which the end user last authenticated.	1615093118
idp	Identity provider name	local
sid	Unique identifier of the session of the end user on a particular device/user agent. Useful when the user signed into multiple devices and sever might want to terminate a particular session.	9927C19C7742C9323BEA6C1D01575995
jti	The unique identifier of the JWT. Prevents the JWT from being replayed.	93CC732B92ECEA50E7DAEA8E7E85A7C3

In the next lesson we will secure one of our microservices to make sure only clients with a valid access token can get access to it.