**JWT – JSON WEB TOKEN**

A JSON web token(JWT) is *JSON Object* which is used to securely transfer information over the web (between two parties). It can be used for an authentication system and can also be used for information exchange. The token is mainly composed of *header, payload, signature*. These three parts are separated by dots(.). JWT defines the structure of information we are sending from one party to the another, and it comes in two forms – **Serialized, Deserialized**. The Serialized approach is mainly used to transfer the data through the network with each request and response. While the deserialized approach is used to read and write data to the web token.

***Deserialized***

JWT in the deserialized form contains only the header and the payload. Both of them are plain JSON objects.

***Header***

A header in a JWT is mostly used to describe the cryptographic operations applied to the JWT like signing/decryption technique used on it. It can also contain the data about the media/content type of the information we are sending. This information is present as a JSON object then this JSON object is encoded to BASE64URL. The cryptographic operations in the header define whether the JWT is signed/unsigned or encrypted and are so then what algorithm techniques to use. A simple header of a JWT looks like the code below:

 {

    "typ":"JWT",

    "alg":"HS256"

 }

The ‘alg’ and ‘typ’ are object key’s having different values and different functions like the ‘typ’ gives us the type of the header this information packet is, whereas the ‘alg’ tells us about the encryption algorithm used.

**Note:** HS256 and RS256 are the two main algorithms we make use of in the header section of a JWT.

Some JWT’s can also be created without a signature or encryption. Such a token is referred to as unsecured and its header should have the value of the alg object key assigned to as ‘none’.

{

    "alg":"none"

}

***Payload***

The payload is the part of the JWT where all the user data is actually added. This data is also referred to as the ‘claims’ of the JWT. This information is readable by anyone so it is always advised to not put any confidential information in here. This part generally contains user information. This information is present as a JSON object then this JSON object is encoded to BASE64URL. We can put as many claims as we want inside a payload, though unlike header, no claims are mandatory in a payload. The JWT with the payload will look something like this:

 {

     "userId":"b07f85be-45da",

     "iss": "https://provider.domain.com/",

     "sub": "auth/some-hash-here",

     "exp": 153452683

 }

The above JWT contains *userId, iss, sub, and exp*. All these play a different role as userId is the ID of the user we are storing, ‘iss’ tells us about the issuer, ‘sub’ stands for subject, and ‘exp’ stands for expiration date.

***Serialized***

JWT in the serialized form represents a string of the following format:

[header].[payload].[signature]

all these three components make up the serialized JWT. We already know what header and payload are and what they are used for. Let’s talk about signature.

***Signature***

This is the third part of JWT and used to verify the authenticity of token. BASE64URL encoded header and payload are joined together with dot(.) and it is then hashed using the hashing algorithm defined in a header with a secret key. This signature is then appended to header and payload using dot(.) which forms our actual token *header.payload.signature*

***Syntax :***

HASHINGALGO( base64UrlEncode(header) + “.” + base64UrlEncode(payload),secret)

So all these above components together are what makes up a JWT. Now let’s see how our actual token will look like:

***JWT Example***

*header:*

{

  "alg": "HS256",

  "typ": "JWT"

}

*Payload:*

{

  "id": 123456789,

  "name": "Joseph"

}

Secret: SecretToken

***JSON Web Token***

*eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MTIzNDU2Nzg5LCJuYW1lIjoiSm9zZXBoIn0.OpOSSw7e485LOP5PrzScxHb7SR6sAOMRckfFwi4rp7o*