W08 Readings

**JWT – JSON Web Token**

JWT securely transmits information as a JSON object. JWT is trusted because it’s signed using a secret public/private key using RSA. It’s compact and can be sent through a URL, POST parameter, or inside an HTTP header.

When JWT is useful:

It’s useful for authentication, allowing users who are logged in to access permitted routes, services and resources with that token.

It’s also great for secure information exchange using public/private key pair.

JWT has three parts separated by dots:

* Header: has two parts, the type of the token (JWT) and the hashing algorithm.

Example:

{

'alg': 'HS256',

'typ': 'JWT'

}

* Payload: contains the claims. They are typically claims about the user. There are three types of claims:
  + Reserved claims: recommended predefined claims. Some of them are iss (issuer), exp (expiration time), sub (subject), aud (audience), among others.
  + Public claims: they should be defined in the IANA JSON Web Token Registry to avoid collisions.
  + Private claims: These are created claims to share information between parties who agree to use them.

Example of a payload:

{

'sub': '1234567890',

'name': 'John Doe',

'admin': true

}

* Signature: It is meant to verify that the JWT sender is who it says it is and ensures the message wasn’t changed. To create a signature, you have to take the encoded header, encoded payload, a secret, the algorithm specified in the header, and sign that.

Signature example of a HMAC SHA256 algorithm:

HMACSHA256(

base64UrlEncode(header) + '.' +

base64UrlEncode(payload),

secret)

Example: xxxxx.yyyyy.zzzzz

How JWTs work:

Browser Server

POST/users/login with username and password > Creates a JWT with a secret

< Returns the JWT to the browser

Sends the JWT on the Authorization Header > Checks JWT signature. Get user information

from the JWT

< Sends response to client.