JEE Course
JWT

Márcio Fuckner

HvA – Hogeschool van Amsterdam

Content

- JWT definition
- Usages
- Elements

What is JWT

- A JSON Web Token (JWT) is a URL-safe, formatted piece of JSON data
- Example:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.

TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

What is JWT

- A compact, printable representation of claims, with a signature to verify the authenticity

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9	{
	"alg": "HS256",
	"typ": "JWT"
	}

eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9	{
	"sub": "1234567890",
	"name": "John Doe",
	"admin": true
	}

TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ	Signature

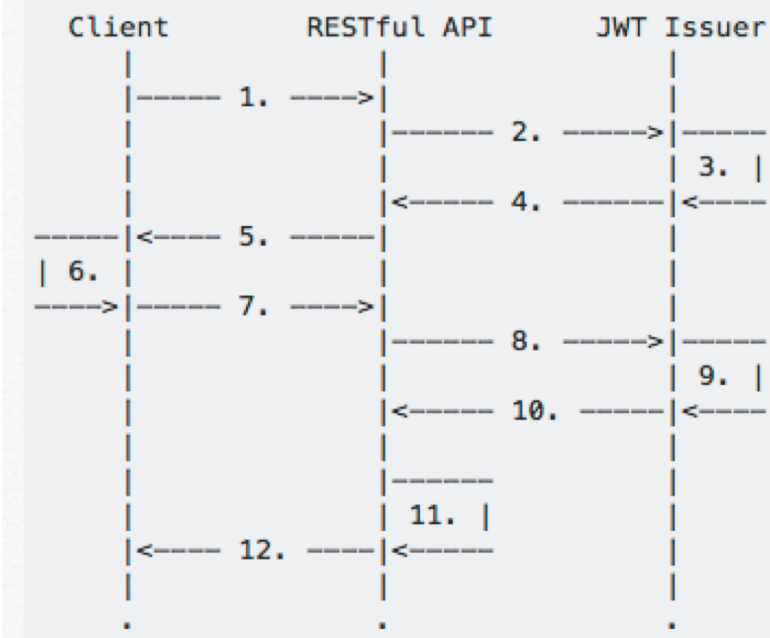
Usages

- Authentication
- Authorization
- Federated identity
- Client-side sessions (“stateless” sessions)

Why use it?

- JWTs are based off of the open standard RFC7519 and used across many different languages and platforms
- JWTs are self-contained pieces of data whose integrity can be validated without any additional information
- JWTs integrate nicely into HTTP transactions as headers or URL segments

Typical Flow



1. Ask RESTful API for a JWT using login endpoint.
2. Ask Issuer to create a new JWT.
3. Create JWT.
4. Return JWT to the RESTful API.
5. Return JWT to Client.
6. Store JWT to append it to all future API requests.
7. Ask for data from API providing JWT as authorization.
8. Send JWT to Issuer for verification.
9. Issuer verifies JWT.
10. Issuer returns 200 OK, verification successful.
11. Retrieve and format data for Client.
12. Return data to Client.

Elements: Header

- Contains information about the token
- Typically the token type and algorithm

```
{
        "alg": "HS256",
        "typ": "JWT"
}
```

Elements: Payload

- Contains a set of claims
- There are three different kinds of claims:
  - Public Claims: user-defined data
  - Private Claims: convention-based data
  - Registered Claims: metadata with reserved names

```
{
        "sub": "1234567890",
        "name": "John Doe",
        "admin": true
}
```

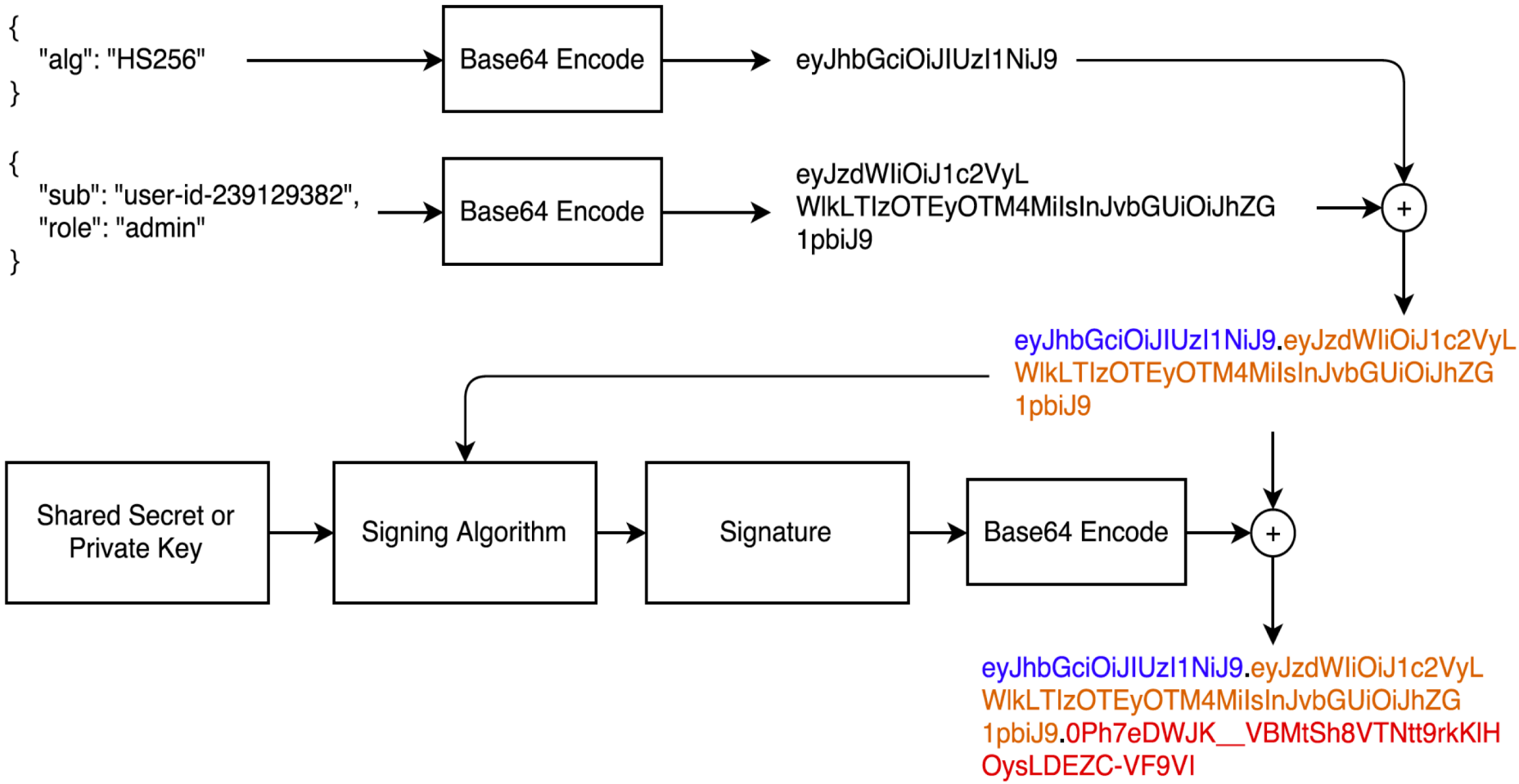
Elements: Payload: Registered claims

Iss	Issuer. from the word issuer. A case-sensitive string or URI that uniquely identifies the party that issued the JWT. Handling of this claim is application specific
Sub	Subject. A case-sensitive string or URI that uniquely identifies the party that this JWT carries information about. Handling of this claim is application specific
Aud	Audience: Either a single case-sensitive string or URI or an array of such values that uniquely identify the intended recipients of this JWT. Handling of this claim is application specific
Exp	Expiration: A number representing a specific date and time in the format “seconds since epoch” as defined by POSIX6. This claims sets the exact moment from which this JWT
Nbf	From not before: The opposite of the exp claim. This claim sets the exact moment from which this JWT is considered valid
Iat	Issued at.
Jti	Unique identifier. This claim may be used to differentiate JWTs with other similar content (preventing replays, for instance). It is up to the implementation to guarantee uniqueness.

Elements

- Signature
  - An encrypted summary of the header and payload sections
  - Can be used to validate integrity of a received JWT

Putting it all together



Source: JWT Handbook, Sebastian Peyrott

Further reading

https://jwt.io/introduction/

https://auth0.com/docs/jwt

https://scotch.io/tutorials/the-anatomy-of-a-json-web-token

http://www.seedbox.com/en/blog/2015/06/05/oauth-2-vs-json-web-tokens-comment-securiser-un-api/