JSON WEB TOKEN

JWT

ABOUT ME

Arely Viana

Software Engineer

/areviana

MEETING
MIND MAP

01 CONCEPTS

02 SECURITY

03 REVOCATION STRATEGIES

04 Q&A

About me

References

01 CONCEPTS

TOKEN TYPES

JWT
JSON Web Tokens

encode JSON

Smaller

Good for HTML and HTTP environments

Easy handling on the client-side and REST API

SWT
Simple Web Tokens

String key

Smaller

Good for HTML and HTTP environments

SAML
Security Assertion Markup Language Tokens

encode XML

Largest

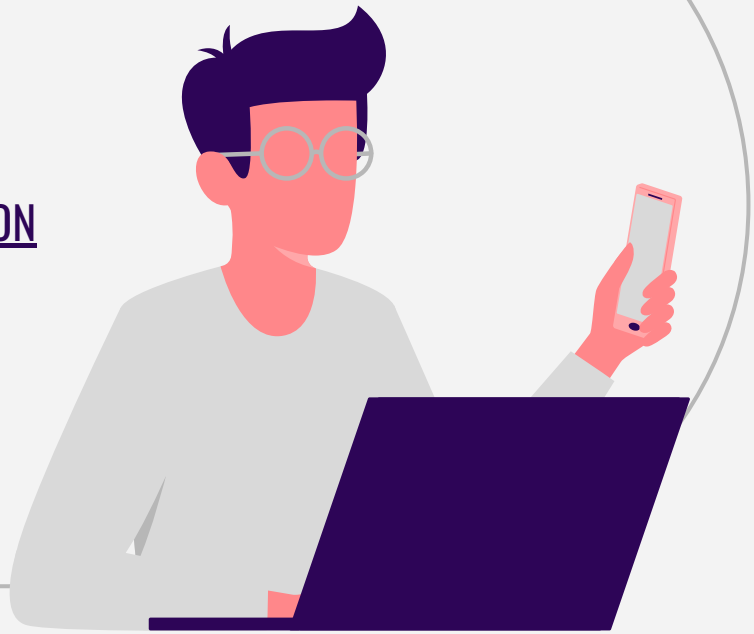
Good for SOAP implementations

Reference

“JWT tokens are self-describing”

It’s an standard, the tokens are a compact and url-safe way for securely transmitting information(claims) between two parties as a JSON object.

Authorization: Bearer <token>



HMACSHA256

Encoding

Encryption

JWS

JWE

JWS

JSON WEB SIGNATURE

Represents digitally signed or MAC content using JSON data structures and base64url encoding.

Base64URL(Header) . Base64URL(Payload) . Base64URL(Signature)

HEADER

JWA

More here!

alg	algorithm	identifies the cryptographic algorithm used to secure the JWS
typ	type	used to declare the type of the signed content

BASE64URL

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

= "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9"

Base64URL(Header)

PAYLOAD

sub	subject	Who are we talking about?
exp	Expiration time	

BASE64URL

```
{
   "sub": "123456",
   "exp": "1654837135"
}
```

= **"eyJzdWIiOiIxMjM0NTYiLCJleHAiOjE2NTQ4MzcxMzV9"**

Base64URL(Payload)

CLAIMS

Reference

Types

- **Registered**
- Public
- Private

REGISTERED

There are not mandatory but recommended, to provide a set of useful, interoperable claims.

iss	issuer	string_or_uri	**exp**	expiration time	tomorrow_int
sub	subject	user_uuid	**iat**	issued at	created_at_int
aud	audience	device_id	**jti**	id	token_uuid
scp	scope	entity_type	**nbf**	not before	>=now_int

SIGNATURE

alg	HS256
key	96e0ec80-e60b-437c-b714-29f7d6e1813e

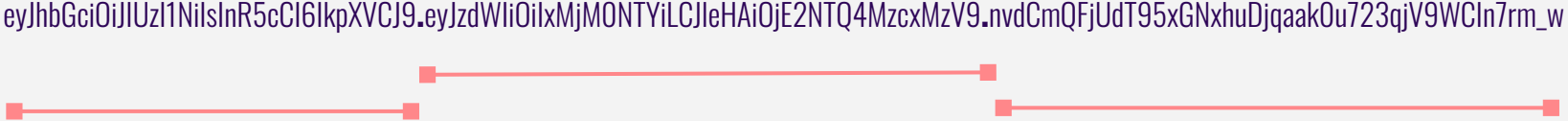
BASE64URL

HMACSHA256 (
Base64URL(header) + '.' +
Base64URL(payload),
secret)

= “nvdCmQFjUdT95xGNxhuDjqaak0u723qjV9WCIn7rm_w”

Base64URL(Signature)

JWT - JWS



Base64URL(Header) . Base64URL(Payload) . Base64URL(Signature)

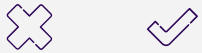
Ruby Demo

```ruby
require 'base64'
require 'json'
require 'time'
require 'securerandom'
require 'openssl'

########## HEADER ##########
header = { alg: "HS256", typ: "JWT" }.to_json
header_encode = Base64.urlsafe_encode64(header)
header == Base64.urlsafe_decode64(header_encode)

########## PAYLOAD ##########
# Expires in 10 minutes from now
exp = Time.new + (60 * 10)
sub = SecureRandom.uuid
payload = { sub: sub, exp: exp.to_i }.to_json
payload_encode = Base64.urlsafe_encode64(payload)
payload == Base64.urlsafe_decode64(payload_encode)

########## SIGNATURE ##########
key = SecureRandom.uuid
algorithm = 'SHA256'
data = header_encode + '.' + payload_encode

hmac_sha256 = OpenSSL::HMAC.hexdigest(algorithm, key, data)
signature = Base64.urlsafe_encode64(hmac_sha256)

token = data + '.' + signature
```

02

SECURITY

CONSIDERATIONS



1. The server needs check if the token was signed by itself
2. Set short-lived tokens
3. Don’t store the whole token that has sensitive information.
4. Don’t add private information unless you encrypt your tokens.
5. Don’t add changing information that could change user behavior.
6. Be careful on what you use to identify a user.

03

REVOCATION STRATEGIES

WHY do we need a revocation layer?

- A JWT already has an expiration time and a signature from the server
- Are we abandoning the benefits of a JWT token?
- Can we trust the client-side for the revocation process?

Revocation ways

Change the Secret

Invalidate absolutely all issued tokens

Denylist

Invalidate specific tokens

Attached token

A entity has one token

Allowlist

Validate specific tokens

04

Q & A

RESOURCES

- JWT Interactive
- RFC 7519 - JWT
- RFC7515 - JWS
- RFC7516 - JWE
- RFC7518 - JWA
- JWT Revocation
- Encryption vs. Encoding

THANKS!!

My contacs

/areviana