Introduction to JSON Web Tokens

What is JSON Web Token?

- JSON Web Token (JWT) adalah standar terbuka (RFC 7519) yang mendefinisikan cara padu dan mandiri untuk mentransmisikan informasi dengan aman antar pihak sebagai objek JSON. Informasi ini dapat diverifikasi dan dipercaya karena ditandatangani secara digital. JWT dapat ditandatangani menggunakan pasangan kunci rahasia (dengan algoritme HMAC) atau pasangan kunci publik / pribadi menggunakan RSA atau ECDSA .

When should you use JSON Web Tokens?

- **Otorisasi** : Ini adalah skenario paling umum untuk menggunakan JWT.
- **Pertukaran Informasi** : Token Web JSON adalah cara yang baik untuk mengirimkan informasi antar pihak dengan aman

What is the JSON Web Token structure?

- Dalam bentuknya yang ringkas, JSON Web Token terdiri dari tiga bagian yang dipisahkan oleh titik (.), yaitu:

  a) Header → x

  b) Payload → y

  c) Signature → z

JWT biasanya terlihat seperti berikut.

```
xxxxx.yyyyy.zzzzz
```

a) Header

- Header biasanya terdiri dari dua bagian: jenis token, yaitu JWT, dan algoritma penandatanganan yang digunakan, seperti HMAC SHA256 atau RSA.
- Sebagai contoh:

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

- Header kemudian dikodekan ke Base64Url untuk membentuk bagian pertama JWT.

b) Payload

- Bagian kedua dari token adalah payload, yang berisi klaim. Klaim adalah pernyataan tentang suatu entitas (biasanya, pengguna) dan data tambahan. Ada tiga jenis klaim:

  1)Klaim terdaftar ,

  2)Klaim publik , dan

  3)Klaim pribadi .

b.1) Klaim Terdaftar (*Registered claims*)

- Ini adalah sekumpulan klaim yang ditentukan sebelumnya yang tidak wajib tetapi direkomendasikan, untuk memberikan sekumpulan klaim yang berguna dan dapat dioperasikan. Beberapa di antaranya adalah: **iss** (Issuer), **Exp** (Expiration Time), **Sub** (Subject), **Aud** (Audience), dan lain - lain .

b.1) Klaim Publik (*Public claims*)

- Ini dapat ditentukan secara bebas oleh mereka yang menggunakan JWT. Tetapi untuk menghindari tabrakan, mereka harus didefinisikan di **IANA JSON Web Token Registry** (https://www.iana.org/assignments/jwt/jwt.xhtml) atau didefinisikan sebagai **URI** yang berisi namespace tahan tabrakan.

b.1) Klaim Pribadi (Private *claims*)

- Ini adalah klaim khusus yang dibuat untuk berbagi informasi antara pihak-pihak yang setuju menggunakannya dan bukan merupakan klaim terdaftar atau publik.
- Contoh payload:

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

- Payload kemudian dikodekan ke Base64Url untuk membentuk bagian kedua dari Token Web JSON.

c) Signature (tanda tangan)

- Untuk membuat bagian *Signature* kita harus mengambil *header* yang dikodekan, *payload* yang disandikan, kunci rahasia, algoritma yang ditentukan di header, dan menandatanganinya.
- Misalnya jika Anda ingin menggunakan algoritme HMAC SHA256, tanda tangan akan dibuat dengan cara berikut:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

c) Signature (tanda tangan)

- Tanda tangan digunakan untuk memverifikasi bahwa pesan tidak berubah di sepanjang jalan, dan, dalam kasus token yang ditandatangani dengan kunci pribadi, itu juga dapat memverifikasi bahwa pengirim JWT adalah sama seperti yang dikatakannya.

Putting all together (header.payload.signature)

- Hasilnya adalah tiga string URL Base64 yang dipisahkan oleh titik-titik yang dapat dengan mudah diteruskan dalam lingkungan HTML dan HTTP, sekaligus lebih ringkas jika dibandingkan dengan standar berbasis XML seperti SAML.
- Berikut ini adalah JWT yang header dan payloadnya dienkodekan sebelumnya, dan ditandatangani dengan kunci rahasia.

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4
gRG9lIiwiaXNTb2NpYWwiOnRydWV9.
4pcPyMD09olPSyXnrXCjTwXyr4BsezdI1AVTmud2fU4
```

How do JSON Web Tokens work?

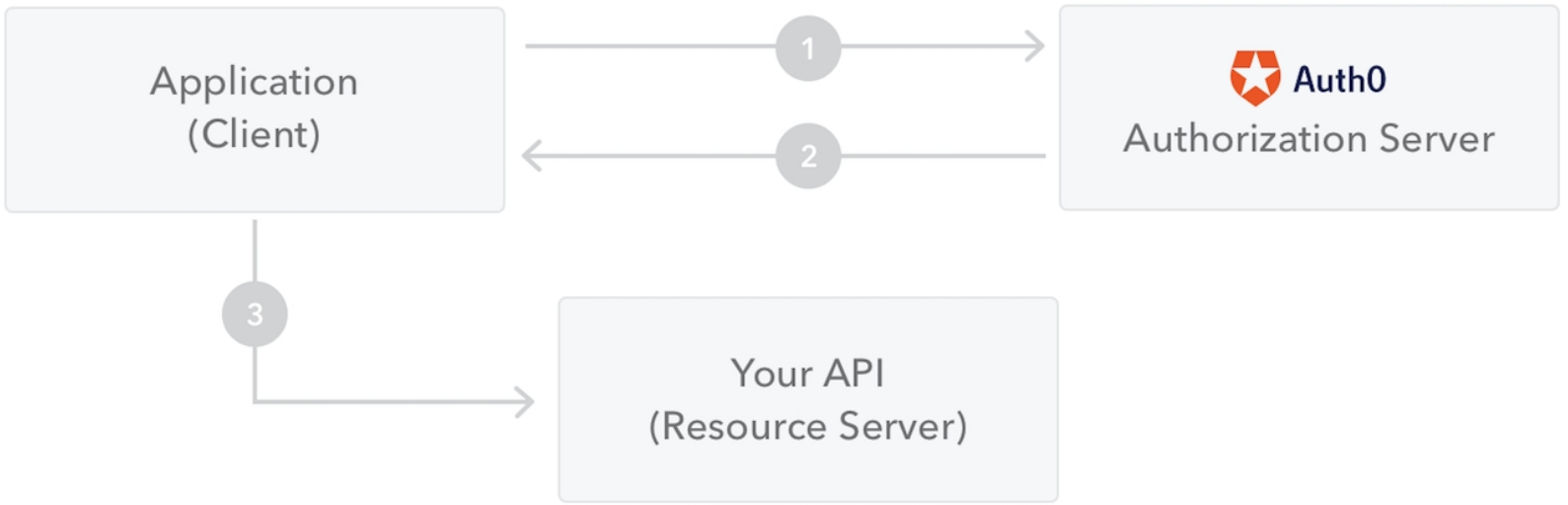
- Dalam otentikasi, ketika pengguna berhasil masuk menggunakan kredensial mereka, Token Web JSON akan dikembalikan. Karena token adalah kredensial, kehati-hatian harus dilakukan untuk mencegah masalah keamanan.
- Kapan pun pengguna ingin mengakses rute atau sumber daya yang dilindungi, pengguna harus mengirimkan JWT, biasanya di header **Otorisasi** menggunakan skema **Bearer**

```
Authorization: Bearer <token>
```

How do JSON Web Tokens work?

- Rute server yang terlindungi akan memeriksa JWT yang valid di Authorization header, dan jika ada, pengguna akan diizinkan untuk mengakses sumber daya yang dilindungi. Jika JWT berisi data yang diperlukan, kebutuhan untuk melakukan query database untuk operasi tertentu dapat dikurangi.
- Jika token dikirim di Authorization header, Cross-Origin Resource Sharing (CORS) tidak akan menjadi masalah karena tidak menggunakan cookie.

Diagram berikut menunjukkan bagaimana JWT diperoleh dan digunakan untuk mengakses API atau sumber daya:



1. Aplikasi atau klien meminta otorisasi ke server otorisasi. Ini dilakukan melalui salah satu aliran otorisasi yang berbeda. Misalnya, aplikasi web tipikal yang mendukung OpenID Connect akan melewati /oauth/authorize *endpoint* menggunakan alur kode otorisasi .
2. Saat otorisasi diberikan, server otorisasi mengembalikan token akses ke aplikasi.
3. Aplikasi menggunakan token akses untuk mengakses sumber daya yang dilindungi (seperti API).