Securing APIs

Reign

Time to give feedback on your Pairs partner. Compliments, complaints, whatever. It gives us a chance to help those who need help overcoming their antisocial tendencies.
https://te-reign.azureedge.net/
user: email
password: TechElevatorStudent

Whiteboarding

1. In comments, write down the task given
2. In comments, write out what you think you need to do
3. Start with a valid method signature (practice these)
4. Write code to do the thing under the comment that says to do the thing

The importance here is that even if you never got to step 4, you would have shown that you could listen, follow directions, break a problem down into parts, and think logically. Also, that you are familiar enough with Java to write the method signature. After all that, it matters less how well you actually write the code, which is a hard thing to do well under pressure.

Notes from https://jwt.io/introduction/

JSON Web Tokens consist of three parts separated by dots (`.`), which are:

- Header
- Payload
- Signature

Therefore, a JWT typically looks like the following.

`xxxxx.yyyyy.zzzzz`

Let's break down the different parts.

Header

The header *typically* consists of two parts: the type of the token, which is JWT, and the signing algorithm being used, such as HMAC SHA256 or RSA.

For example:

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

Then, this JSON is **Base64Url** encoded to form the first part of the JWT.

Payload

The second part of the token is the payload, which contains the claims. Claims are statements about an entity (typically, the user) and additional data. There are three types of claims: *registered*, *public*, and *private* claims.

- **Registered claims**: These are a set of predefined claims which are not mandatory but recommended, to provide a set of useful, interoperable claims. Some of them are: **iss** (issuer), **exp** (expiration time), **sub** (subject), **aud**

(audience), and others.
Notice that the claim names are only three characters long as JWT is meant to be compact.

- **Public claims**: These can be defined at will by those using JWTs. But to avoid collisions they should be defined in the IANA JSON Web Token Registry or be defined as a URI that contains a collision resistant namespace.
- **Private claims**: These are the custom claims created to share information between parties that agree on using them and are neither *registered* or *public* claims.

An example payload could be:

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

The payload is then **Base64Url** encoded to form the second part of the JSON Web Token.

Do note that for signed tokens this information, though protected against tampering, is readable by anyone. Do not put secret information in the payload or header elements of a JWT unless it is encrypted.

Signature

To create the signature part you have to take the encoded header, the encoded payload, a secret, the algorithm specified in the header, and sign that.

For example if you want to use the HMAC SHA256 algorithm, the signature will be created in the following way:

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

The signature is used to verify the message wasn't changed along the way, and, in the case of tokens signed with a private key, it can also verify that the sender of the JWT is who it says it is.