An explanation of authentication, authorization, and PKI.

Owner: Amo Samuel
Reviewer: Thomas Darko

Authentication

Authentication is the process of verifying the identity of a user or system. In my Spring Boot application, authentication ensures that only users with valid credentials (username and password) can access the system.

How It Works in my Code:

1. JWT Authentication: The JwtAuthenticationFilter class checks the Authorization header in HTTP requests for a valid JWT (JSON Web Token).
2. The token is parsed to extract the username, which is then used to load the user's details from the database.
3. If the token is valid and the user is authenticated, the user's identity is stored in the SecurityContextHolder, allowing them access to secure resources.

Authorization

Authorization determines what an authenticated user is allowed to do. It defines the permissions and access levels for different users in the system.

How It Works in my Code:

1. The SecurityConfig class configures authorization rules: Certain endpoints, like **/api/v1/login/** and /api/v1/register/**,** are open to everyone
2. All other requests require the user to be authenticated.
3. This ensures that only authenticated users can access sensitive parts of the application, while public endpoints remain accessible to all.

Public Key Infrastructure (PKI)

PKI is a framework that uses public and private cryptographic keys to secure communications. It's often used for creating and validating digital certificates that authenticate users or devices.

How It Works in my Code:

1. RSA Keys: The application uses RSA keys for securing JWT tokens.
2. The RsaKeyProperties class loads the RSA public and private keys from files (private.pem and public.pem) specified in the application's configuration.
3. The private key is used to sign the JWT, ensuring its integrity and authenticity.
4. The public key is used to verify the JWT when it’s received by the server.
5. I’ve set my Spring Boot app to use HTTPS on port 8443 with SSL/TLS enabled and a self-signed certificate.
6. I configured SSL in application properties, specifying the keystore file, password, and type.
7. In SecurityConfig, I enforced HTTPS for all requests
8. I used keytool to generate a self-signed certificate