CSRF

CSRF (Cross-Site Request Forgery) in Spring allows an attacker to make unauthorized requests on behalf of a user.

By Eslam Khder

CSRF

CSRF is an attack where an attacker tricks a user into making an unwanted request to a web application where the user is authenticated. Since the user is logged in, the server assumes that the request is legitimate, and processes it accordingly. This can lead to unauthorized actions being taken on behalf of the user, such as making financial transactions or changing account settings.

CSRF

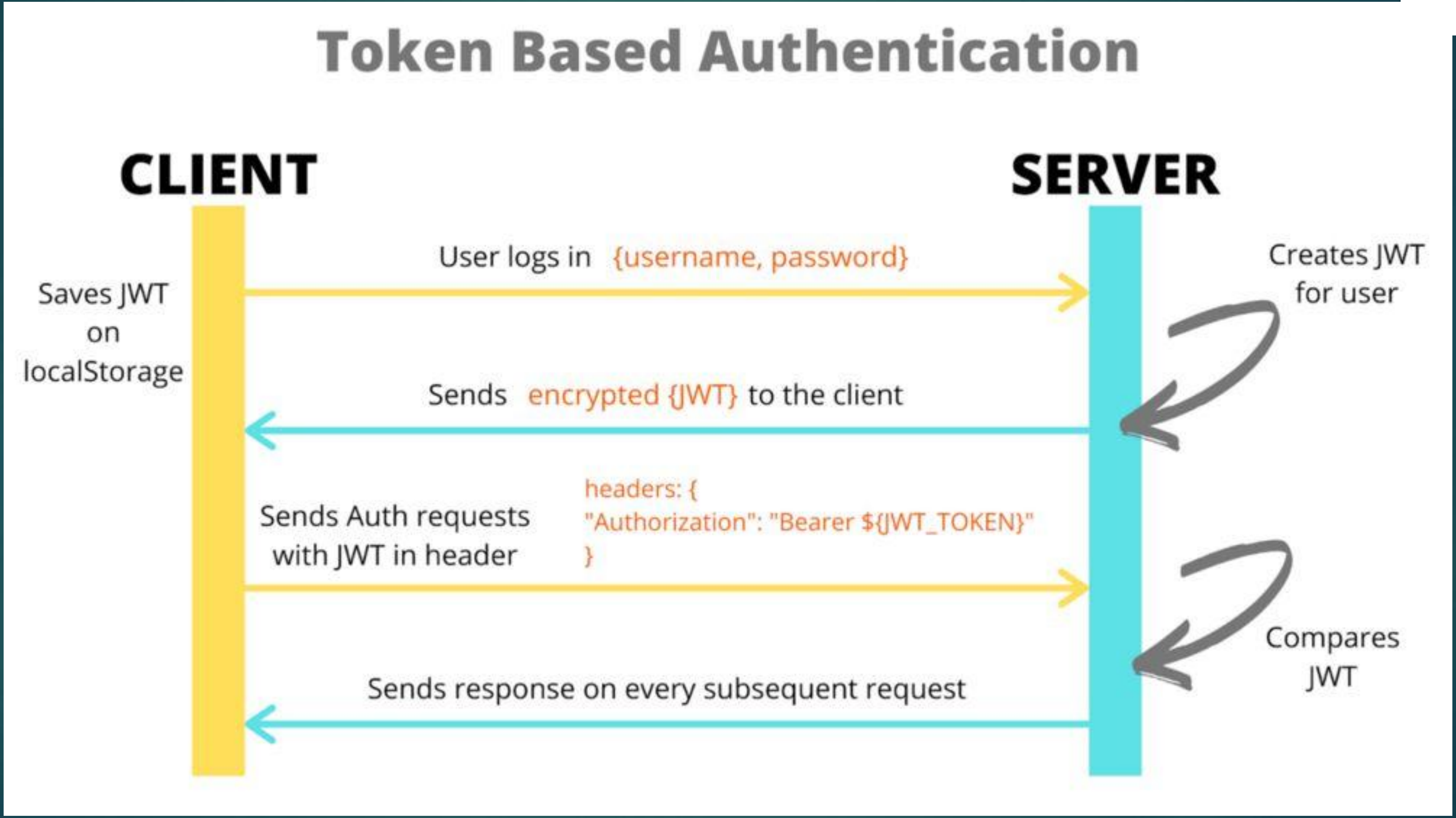
Spring provides built-in CSRF protection for web applications using Spring Security. CSRF protection in Spring works by ensuring that requests that modify data (such as POST, PUT, DELETE requests) include a special token. This token is tied to the user's session and must be included in the request. If the token is missing or invalid, the request will be rejected.

If you configure csrf().disable() in Spring Security, it completely disables CSRF protection for your application. This means that Spring Security will not check for or require CSRF tokens in requests, regardless of the HTTP method (GET, POST, PUT, DELETE, etc.).

JWT (JSON Web Token)

JWT (JSON Web Token) It is a standard used for securely transmitting information between a client and a server, often in the context of authentication and authorization in web applications.

By Eslam Khder


Token Based Authentication
CLIENT
SERVER
User logs in {username, password}
Creates JWT for user
Saves JWT on localStorage
Sends encrypted {JWT} to the client
Sends Auth requests with JWT in header
headers: {
"Authorization": "Bearer ${JWT_TOKEN}"
}
Compares JWT
Sends response on every subsequent request

By Eslam Khder

Key Concepts

Claims: These are statements about an entity (typically, the user) and additional data.

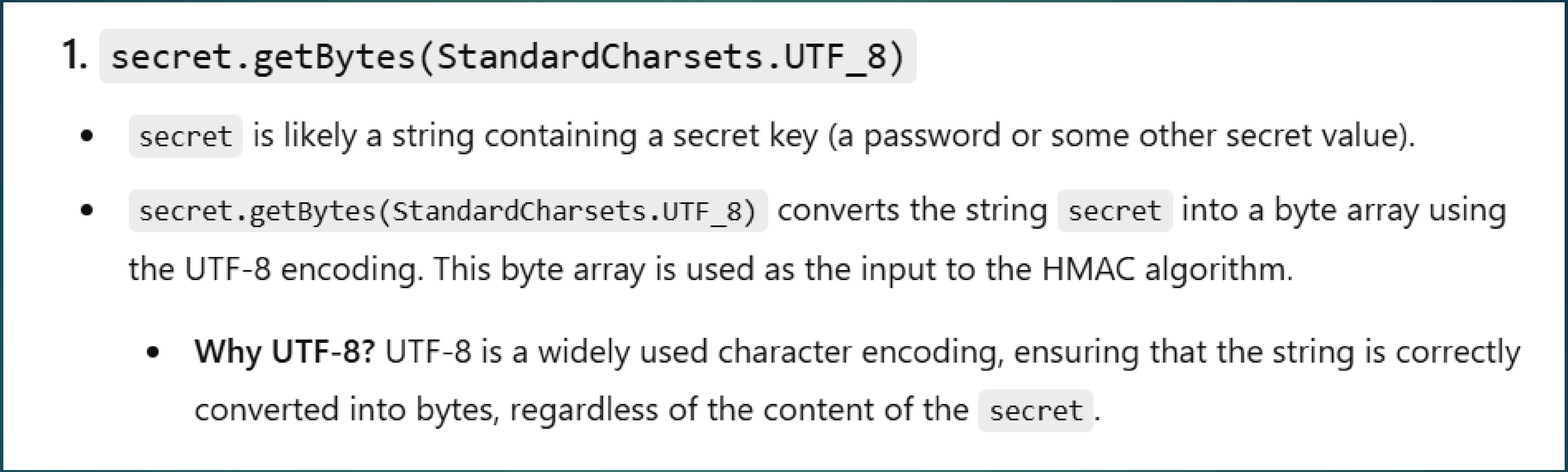
JWT Structure: A JWT consists of three parts:

Header: Typically contains the type of token (JWT) and the signing algorithm (e.g., HS256 or RS256).

Payload: Contains the claims (the data), which is not encrypted, only base64 encoded. The payload is where you put the user data or claims.

Signature: The signature is created using the header, payload, and a secret key (or a private key in asymmetric s

By Eslam Khder

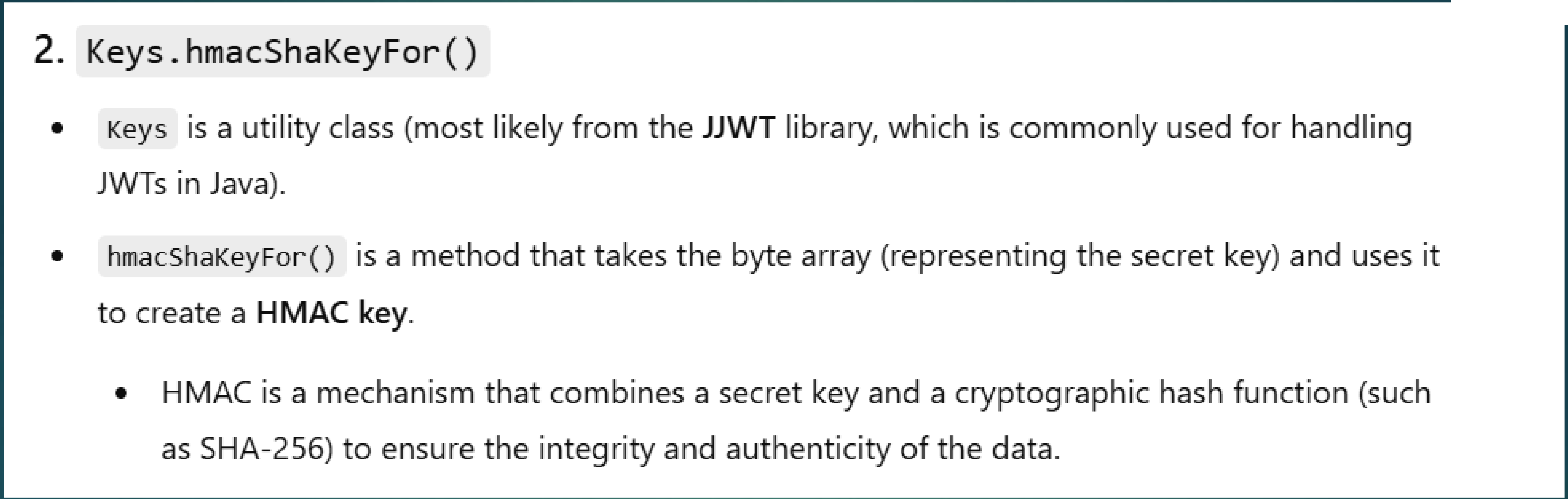


1. `secret.getBytes(StandardCharsets.UTF_8)`

- `secret` is likely a string containing a secret key (a password or some other secret value).
- `secret.getBytes(StandardCharsets.UTF_8)` converts the string `secret` into a byte array using the UTF-8 encoding. This byte array is used as the input to the HMAC algorithm.
  - **Why UTF-8?** UTF-8 is a widely used character encoding, ensuring that the string is correctly converted into bytes, regardless of the content of the `secret`.

Using StandardCharsets.UTF_8 ensures that the String is always converted to bytes the same way, regardless of the system running the code — which is crucial for consistency and correctness in cryptographic operations like generating HMAC keys.

By Eslam Khder



2. `Keys.hmacShaKeyFor()`

- `Keys` is a utility class (most likely from the **JJWT** library, which is commonly used for handling JWTs in Java).
- `hmacShaKeyFor()` is a method that takes the byte array (representing the secret key) and uses it to create a **HMAC key**.
  - HMAC is a mechanism that combines a secret key and a cryptographic hash function (such as SHA-256) to ensure the integrity and authenticity of the data.

HMAC : Hash-based Message Authentication Code

SHA (Secure Hash Algorithm)

SHA-256?
SHA-256 (**Secure Hash Algorithm 256-bit**) is a **cryptographic hash function**

By Eslam Khder