Spring Security (RSA)

By Srinivas Basha

JWT(JSON Web Token)

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed.

JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA

A JWT (JSON Web Token) consists of three parts:

- **Header:** Specifies the signing algorithm
- **Payload:** Contains the claims
- **Signature:** Verify the sender of the JWT

JWT Implementation (HMAC)

- Both signing and verification use the same secret key.
- If an attacker gains access to the secret key, they can create and validate tokens.
- An attacker with access to the key can forge the token.

JWT Asymmetric Encryption With RSA

RSA (Rivest-Shamir-Adleman) is a widely used public-key cryptographic algorithm. RSA is used for secure data transmission and relies on the mathematical properties of large prime numbers

Signing and Verification:

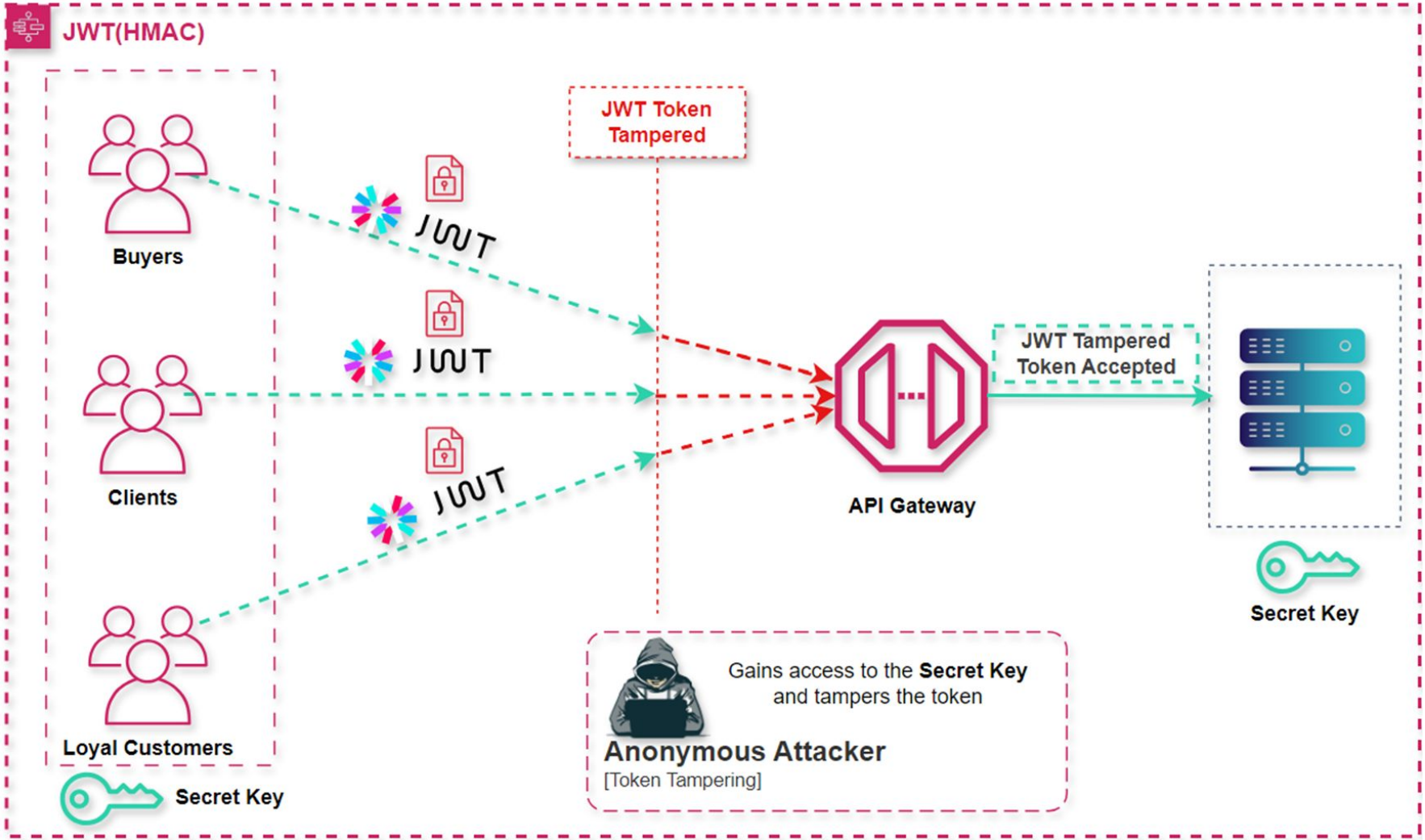
Private Key: Used for signing the JWT. Only the issuer of the token has access to the private key.

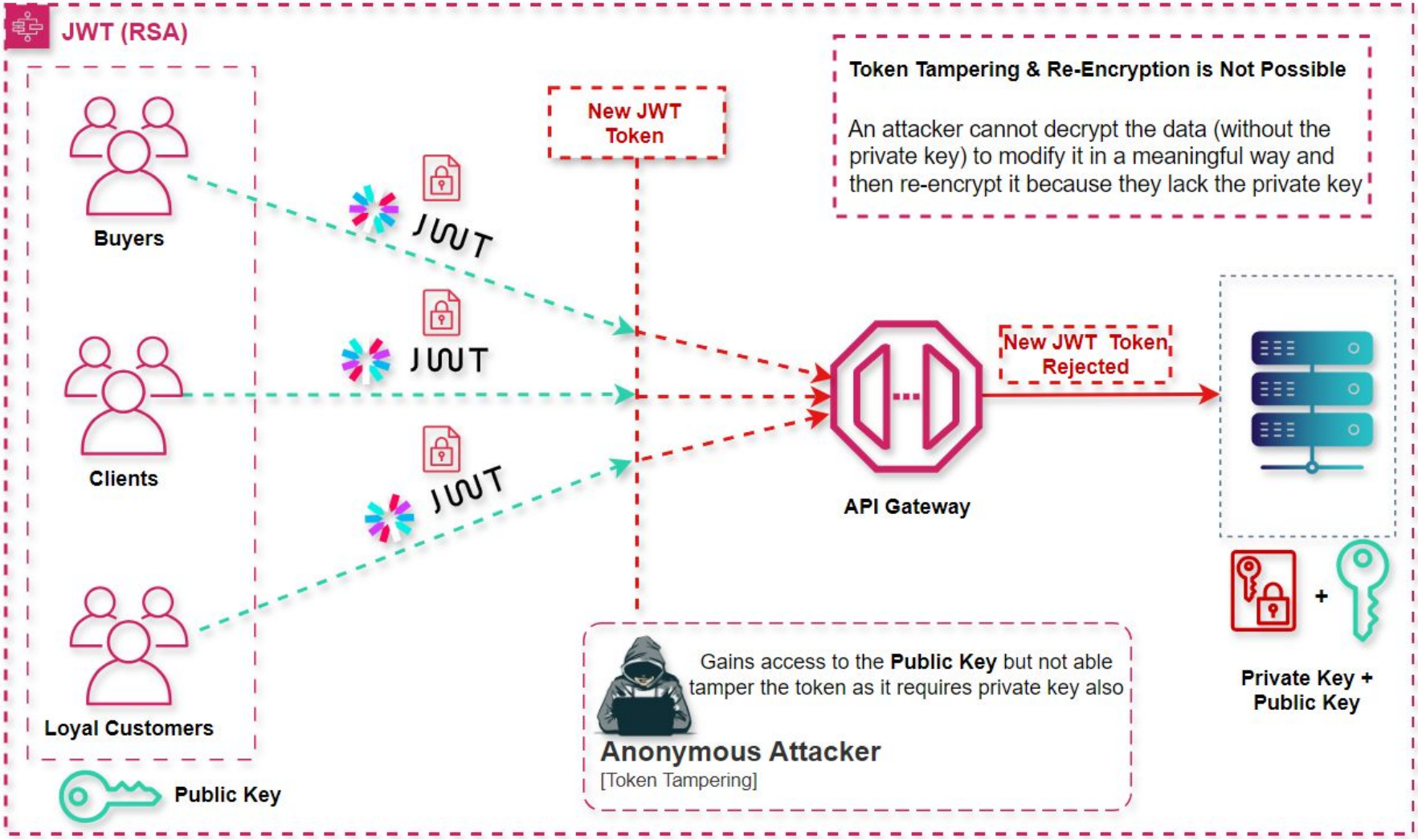
Public Key: Used for verifying the signature of the JWT. The public key is shared and can be used by any recipient to verify that the JWT was indeed signed by the issuer's private key.

Encryption and Decryption:

Public Key: Used for encrypting data. Anyone can use the public key to encrypt data, but only the holder of the corresponding private key can decrypt it.

Private Key: Used for decrypting data that was encrypted with the public key.


JWT(HMAC)
JWT Token Tampered
Buyers
Clients
Loyal Customers
Secret Key
JWT
JWT
JWT
API Gateway
JWT Tampered Token Accepted
Secret Key
Gains access to the Secret Key and tampers the token
Anonymous Attacker
[Token Tampering]


JWT (RSA)
Token Tampering & Re-Encryption is Not Possible
An attacker cannot decrypt the data (without the private key) to modify it in a meaningful way and then re-encrypt it because they lack the private key
New JWT Token
Buyers
Clients
Loyal Customers
Public Key
JWT
JWT
JWT
API Gateway
New JWT Token Rejected
Private Key + Public Key
Gains access to the Public Key but not able tamper the token as it requires private key also
Anonymous Attacker
[Token Tampering]

RSA Asymmetric Encryption with JWT (Hands ON)

1. Create a **UserController** with simple GET endpoint then add spring–security dependency and re-run the application.When the user calls GET endpoint, It will show the Login form by default. So, Create a Bean of **SecurityFilterChain** and make the GET API as a public endpoint.
2. Create RSA public and private keys with openssl.
3. Create **User** and **Role** entities and implement **register** and **login** endpoints.
   a. During registration, provide the valid details in the form of JSON and store user details (password must be encrypted and stored in DB in a encrypted way).
   b. Create **AuthenticationManager** Bean to perform authentication and also make **User Entity** to implement UserDetails(provided by spring security) and create a **CustomUserDetailService** implementing the **UserDetailsService** interface to return **UserDetails**.
4. Create the **JwtEncoder** and **JwtDecoder** Beans for encrypting and validating the token.
5. Implement **TokenService** to generate token if credentials are valid.
6. During login, send the required details to like email and password to login and it must return a token which will be used for subsequent requests.