Hash functions and Digital Signatures.

Foros Valentin

December 16, 2022

1 Theory

While often used interchangeably, authentication and authorization represent fundamentally different functions. In simple terms, authentication is the process of verifying who a user is, while authorization is the process of verifying what they have access to. Both authentication and authorization are necessary for an application to be secure.

2 Authentication

Authentication is the act of validating that users are whom they claim to be. This is the first step in any security process. Complete an authentication process with:

- Passwords. Usernames and passwords are the most common authentication factors. If a user enters the correct data, the system assumes the identity is valid and grants access.
- One-time pins. Grant access for only one session or transaction.
- Authentication apps. Generate security codes via an outside party that grants access.
- Biometrics. A user presents a fingerprint or eye scan to gain access to the system.

3 Authorisation

Authorization in system security is the process of giving the user permission to access a specific resource or function. This term is often used interchangeably with access control or client privilege. Giving someone permission to download a particular file on a server or providing individual users with administrative access to an application are good examples of authorization. In secure environments, authorization must always follow authentication. Users should first prove that their identities are genuine before an organization's administrators grant them access to the requested resources.

4 Implementation

In order to implement the server for this laboratory work, I used Google Authenticator. It's a second step in login sequence that asks to enter 6-digits code sent to you by email, text message or Google Authenticator app and this code expires in 30 or 60 seconds.

Here is created and set repository for user credentials :

```
    public class Server {
    public static void main(String[] args) {
        SpringApplication.run(AuthServer.class, args);
    }
}
```

"CredentialRepository" stores credential data that will be used during authorization. I added a simple username in the form of an email into a hashmap, which represents the userKeys :

```
    public class CredentialRepository implements ICredentialRepository {

    private final Map<String, UserTOTP> usersKeys = new HashMap<String, UserTOTP>() {{
        put("nastisha29@gmail.com", null);
    }};

    @Override
    public String getSecretKey(String userName) {
        return usersKeys.get(userName).getSecretKey();
    }

    @Override
    public void saveUserCredentials(String userName,
                                    String secretKey,
                                    int validationCode,
                                    List<Integer> scratchCodes) {
        usersKeys.put(userName, new UserTOTP(userName, secretKey, validationCode, scratchCodes));
    }

    public UserTOTP getUser(String username) {
        return usersKeys.get(username);
    }

    @Data
    @NoArgsConstructor
    @AllArgsConstructor
    public static class UserTOTP {
        private String username;
        private String secretKey;
        private int validationCode;
        private List<Integer> scratchCodes;
    }
}
```

A password is going to be generated by the device when somebody wants to log in. The following code checks the validity of the specified password against the provided Base32-encoded secretKey:

```
    public GoogleAuthenticator gAuth() {
        GoogleAuthenticator googleAuthenticator = new GoogleAuthenticator();
        googleAuthenticator.setCredentialRepository(credentialRepository);
        return googleAuthenticator;
    }
```

Process the message in successive 512-bit chunks:

```
    private static void msgProcessing() {
        for (int i=0; i<16; i++) {
            w[i] = 0;
            for (int j = 0; j<4; j++) {
                w[i] |= ((0x000000FF&chunk[j+4*i]) << (24-j*8));
            }
        }
        for (int i=16; i<64; i++) {
         w[i]=0;
            int s0 = Integer.rotateRight(w[i-15],7) ^ Integer.rotateRight(w[i-15],18) ^
                    (w[i - 15] >>> 3);
            int s1 = Integer.rotateRight(w[i-2],17) ^ Integer.rotateRight(w[i-2],19) ^
                    (w[i-2] >>> 10);
```

```
            w[i] = w[i-16] + s0 + w[i-7] + s1;
        }
    }
```

After this, user generates a QR Code for his username on the generate/username endpoint :

```
 @GetMapping("/generate/{username}")
public void generate(@PathVariable String username, HttpServletResponse response) {
    final GoogleAuthenticatorKey key = gAuth.createCredentials(username);

    QRCodeWriter qrCodeWriter = new QRCodeWriter();

    String otpAuthURL = GoogleAuthenticatorQRGenerator.getOtpAuthTotpURL("CS-LAB", username, key);

    BitMatrix bitMatrix = qrCodeWriter.encode(otpAuthURL, BarcodeFormat.QR_CODE, 200, 200);

    ServletOutputStream outputStream = response.getOutputStream();
    MatrixToImageWriter.writeToStream(bitMatrix, "PNG", outputStream);
    outputStream.close();
}
```

Google Authenticator is configured for generating TOTP passwords and in the end user can use /classic/caesar/encrypt endpoint for the Caesar Cipher:

```
 @PostMapping("/classical/caesar/encrypt")
public String caesarEncrypt(@RequestBody Encryption body) {
    if (gAuth.authorizeUser(body.getUsername(), body.getCode())) {
        return new CaesarCipher(body.getKey()).encrypt(body.getMessage());
    }
    else return "Not valid 2FA Code";
}
```

5 Consclusion

In conclusion, despite the similar-sounding terms, authentication and authorization are separate steps in the login process. While performing this laboratory work I learned how Google Authenticator works. It is is a software-based authenticator by Google that implements two-step verification services using the Time-based One-time Password Algorithm and HMAC-based One-time Password algorithm, for authenticating users of software applications. Also, I learned more about the Authentication Authorisation ans the differences between these two.