# Implementation of 2-factor authentication for website login

Contents

[Implementation of 2-factor authentication for website login 1](#_Toc121871754)

[Introduction 3](#_Toc121871755)

[How does 2FA work? 3](#_Toc121871756)

[Pros and cons of each method: 5](#_Toc121871757)

[2FA Algorithm 6](#_Toc121871758)

[Examples of Coding 7](#_Toc121871759)

[Conclusion 8](#_Toc121871760)

[Reference 9](#_Toc121871761)

# Introduction

According to WebAuthn (2022), 81% of all data breaches leverage stolen or weak passwords, especially that the password is the only key to prove you are indeed you. If a hacker is successful to steal the password, they can easily impersonate you and access your account to get everything he desires. Therefore, it is vital for everyone to implement one more layer to verify their identity and 2-factor authentication (2FA) is the solution to address this issue.

# How does 2FA work?

2FA is widely used to provide an addition layer of protection during sign in. When accessing accounts or applications, users is requested to enter an additional identity verification, such as scanning a fingerprint or entering a code received by phone. Below are different methods to get the verification code:

* Hardware Tokens

Oldest form of 2FA. Small devices that produce new numeric code every minute. Could be either used to copy the code and enter in the login page or device plugged into computer’s USB port

* SMS or Email

After entering username and password on login, a verification code is sent to user’s mobile phone via text message or via email.

* Software Tokens (Authenticator apps)
* Host-based One Time Password (HOTP) & Time-based One Time Password (TOTP)
* Push Notifications

Apps like Google Authenticator, Lastpass Authenticator or Duo etc work by using TOTP When a user signs up, a secret phrase is generated for that user. Based on that secret, a QR code is generated. User scans that QR code on the authenticator app. Using the time and QR code/secret, a new TOTP is generated every minute.

Push notification has a similar function. When a user attempts to login, a push notification from the app will be sent to the user to confirm that you are the user trying to login.

* Biometrics (eg. Android Pay / Apple Pay)

Fingerprint scanner or retina scanners are the latest addition to multifactor authentication

Figure 1 shown below is the flow of 2FA on how it processes. Below is the step and the corresponding description (Ard 2022):

1. The user attempt to log in into the application or website using his credentials.
2. Firstly, it will check whether the user has already obtained a generated secret.
3. The answer will be either a ‘Yes’ or a ‘No’.
4. No: It means he did not yet use 2FA for this account. Continues to Step 4.
5. Yes: 2FA was used for this account before. Skip to Step 7.
6. Registration of the account to the device is required to generate the secret when using 2FA at the first time.
7. The secret is generated and stored in the application (e.g. in a database).
8. A QR code is then created, which includes several data such as the username, the name of the application as well as a generated secret.
9. The user can open the authentication device, e.g. the Google Authenticator app.
10. Scan the QR code through the authentication app on the mobile phone for registration of 2FA. It is only required at the first time.
11. A verification code is then generated by the device based on the secret and the current time. it is valid for a limited period of time (30 seconds by default). After this period, a new verification is generated and the pervious verification code becomes invalid.
12. The verification code is sent to the application.
13. The secret of the user is retrieved by the application (from a database).
14. The verification code is generated by the application based on the secret and the time.
15. The application compared both codes to see whether they match.
16. The answer will be either a ‘Yes’ or a ‘No’.
17. Yes: The user is identified and successfully logged in to the application.
18. No: The user is not authenticated and will be denied the access and shown why (e.g. invalid verification code).

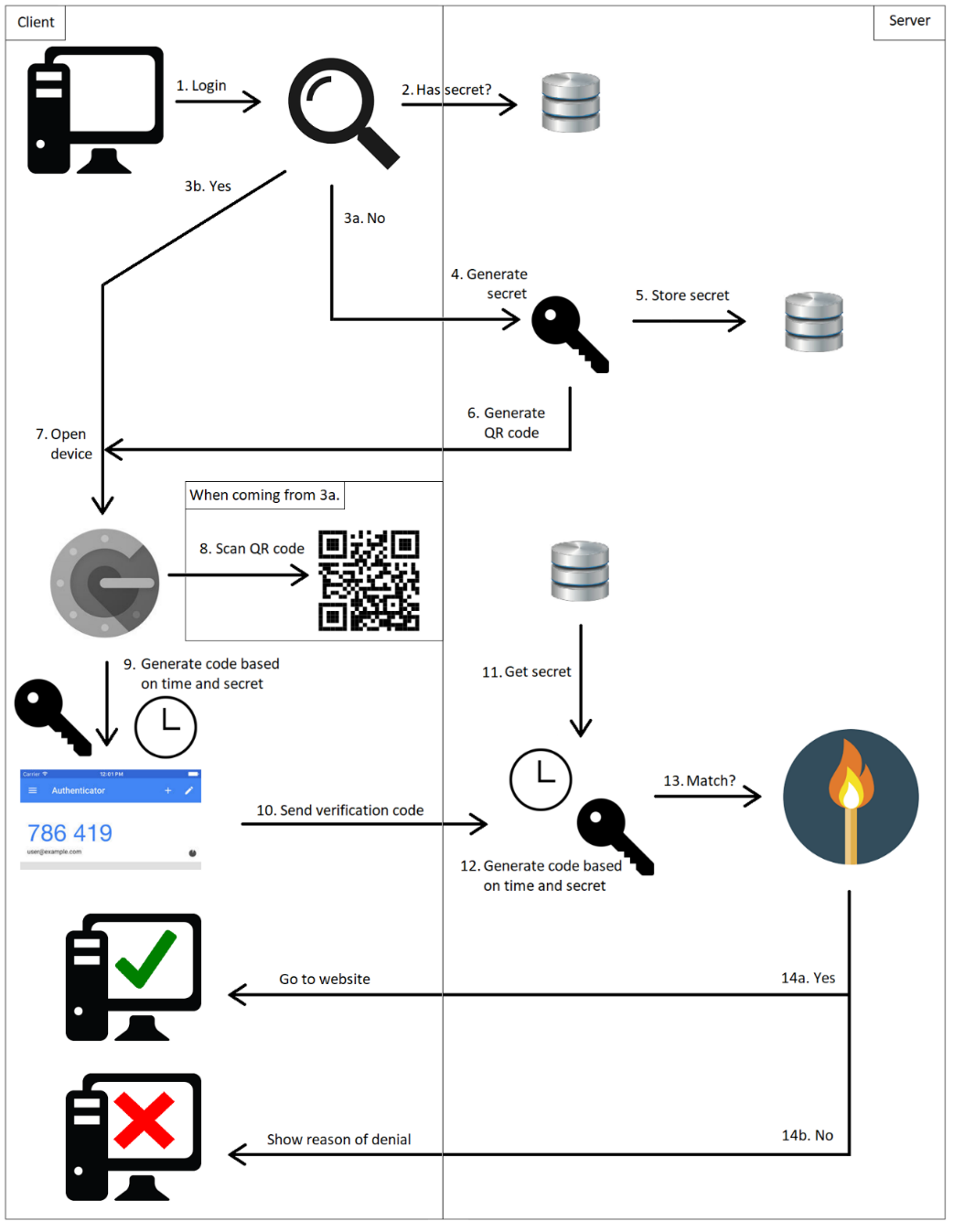


Figure 1: The flow of 2FA (Ard 2022)

In Figure 2 below, it simply describes how to apply 2FA in Redback website.

Diagram

Description automatically generated

Figure 2: The flow of 2FA applied in Redback website

# Pros and cons of each method:

Hardware Tokens

Pro: No internet is required

Con: Need extra hardware

SMS

Pro: Ubiquitous

Con: Not (totally) secure but much better than no 2FA

HOTP & TOTP

Pro: Secure

Con: Annoying?

Push Notification

Pro: Secure, Easy

Con: Many apps needed

# 2FA Algorithm

In this project, 2FA will be implemented with the use of OTP algorithms. It will generate a numeric code, usually 6 or 8 digits, based on a dynamic piece (a moving factor) as well as a static piece (secret key).

In HOTP, the moving factor is a counter. It will increase every time so as to generate different verification code. In TOTP, time is a moving factor and hence the code can be different each time.

Below are the steps and example introducing how to generate the code (Ard 2020).

**Step 1: Generate an HMAC-SHA-1 value**

A random 20-byte string will be generated.

**Step 2: Generate a 4-byte string (Dynamic Truncation)**

OffsetBits = the low-order 4 bits of HmacString [19]

Offset = FromBitsToNumber(OffsetBits)  
OffsetValue = HmacString [OffSet]… HmacString [OffSet+3]  
Bits = the last 31 bits of OffsetValue

**Step 3: Compute an HOTP value**

Number = FromBitsToNumber(Bits)  
Result = Number % (10 ^ Digit)

**Example:**

20-byte string: 1f 86 98 69 0e 02 ca 16 61 85 50 ef 7f 19 da 8e 94 5b 55 5a

It is then to calculate the 4-byte string:

1. The byte on the 19th index is 5a (HmacString[19]).
2. The binary representation of 5a is 0101 1010.
3. The low-order 4 bits are 1010 (OffsetBits).
4. The numeric representation of 1010 is 10 (Offset).
5. The byte on the 10th index is 50 (HmacString[Offset]).
6. The bytes of the next 3 indices are:

* ef (index 11)
* 7f (index 12)
* 19 (index 13)

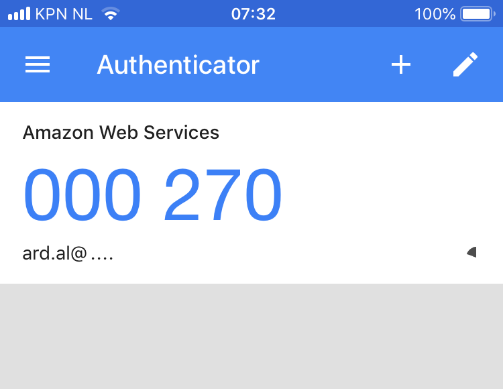
Thus, 4-byte string is generated as 50ef7f19 in this case

The last 31 bits of the binary representation of 50ef7f19 is 101 0000 1110 1111 0111 1111 0001 1001 which can be represented in number as 1357872921.

In this case the digit is 6 and therefore the result is 1357872921 % 10 ^ 6 = 872921.

As a result, the verification code is: 872921

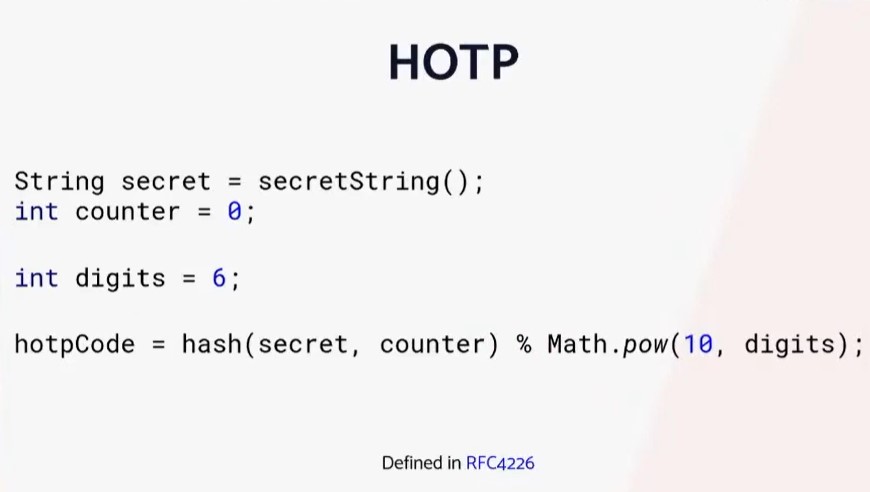
Below is another example of the verification code generated from the Google Authenticator app. The result was 000 270 which is a 6-digit code.



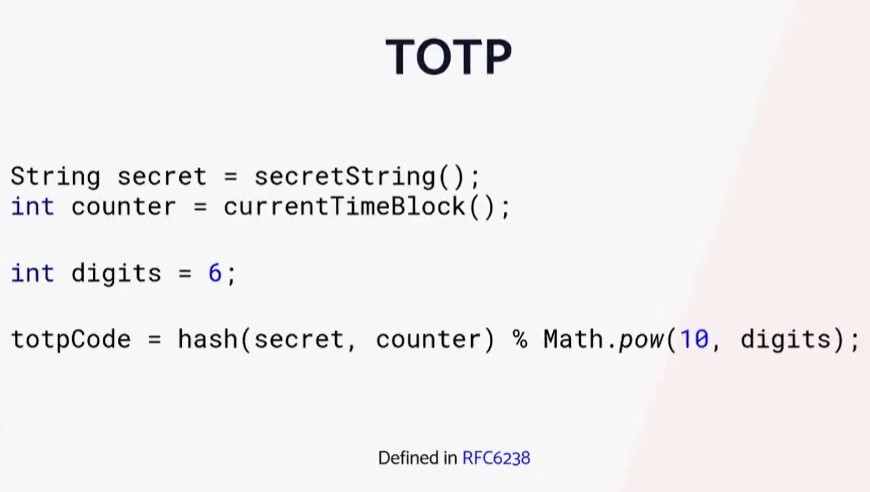
# Examples of Coding

2FA algorithms can be applied with the use of Javascript and below are the examples of the coding for HOTP and TOTP.

HOTP Coding Example:



TOTP Coding Example:



# Conclusion

To conclude, 2FA is an effective way to prevent the application from data compromise. There are various methods to implement 2FA which are hardware tokens, SMS or email, software tokens and biometrics. In this project, Google Authenticator app will be adopted. In addition, 2FA consists of 4 main parts to accomplish the authentication. First is to generate the secret randomly. The second one is to generate the QR code. After that, a verification code can be generated. The last step is to verify the code. When the code is verified, the user can be successfully authenticated and then granted the access to the application. In contrast, the user fails to login. As a result, it can significantly avoid any data breaches due to losing the password.

# Reference

Ard (2022), ‘Secure your own application with Two Factor Authentication’, Craftsmen, accessed on 26 Nov. 22.

<https://craftsmen.nl/secure-your-own-application-with-two-factor-authentication/>

Authy (2022), ‘What is 2FA’, Authy, accessed 25 Nov. 22

<https://authy.com/what-is-2fa/>

Gilliard M (2021), ‘2FA 2Furious: A Workshop on Two-Factor-Authentication with Java’, accessed 26 Nov. 22

<https://github.com/mjg123/Spring-Boot-2FA/blob/main/README.md>

Kenton W (2022), ‘What Is Two-Factor Authentication (2FA)? How It Works and Example’, Investopedia, accessed 18 Nov. 22

<https://www.investopedia.com/terms/t/twofactor-authentication-2fa.asp>

Microsoft (2022), ‘Secure access to resources with multifactor authentication’, Microsoft, , accessed 22 Nov. 22

<https://www.microsoft.com/en-au/security/business/identity-access/azure-active-directory-mfa-multi-factor-authentication?&ef_id=Cj0KCQiAg_KbBhDLARIsANx7wAz-UHDRNL9kYeEGhagH42dVmmWPpMUT4sz0oQTt6dAaXQtbeiart-AaAnhJEALw_wcB:G:s&OCID=AIDcmmtlibe7fa_SEM_Cj0KCQiAg_KbBhDLARIsANx7wAz-UHDRNL9kYeEGhagH42dVmmWPpMUT4sz0oQTt6dAaXQtbeiart-AaAnhJEALw_wcB:G:s&gclid=Cj0KCQiAg_KbBhDLARIsANx7wAz-UHDRNL9kYeEGhagH42dVmmWPpMUT4sz0oQTt6dAaXQtbeiart-AaAnhJEALw_wcB>

[Vonage Dev](https://vonagedev.medium.com/?source=post_page-----6389409159b8--------------------------------) (2020), ‘How to Add Two-Factor Authentication (2FA) Using Java and Spark’, [Level Up Coding, accessed 19 Nov. 22](https://levelup.gitconnected.com/?source=post_page-----6389409159b8--------------------------------)

<https://levelup.gitconnected.com/how-to-add-two-factor-authentication-2fa-using-java-and-spark-6389409159b8>

<https://webauthn.guide/>