Supporting Multi-factor Authentication

Kevin Dockx
Architect

@KevinDockx https://www.kevindockx.com

Coming Up

Introducing multi-factor authentication

Implementing multi-factor authentication

- Google authenticator application

Multi-factor authentication

Identification of users by means of the combination of two or more different factors

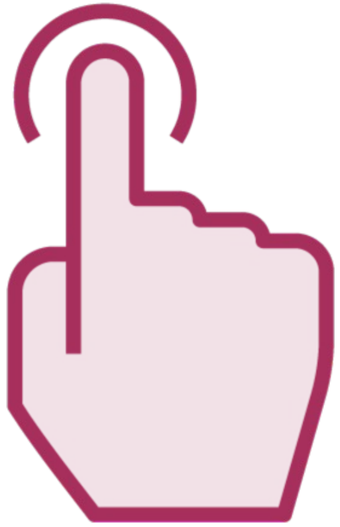
Introducing Multi-factor Authentication

Something you know (e.g.: password, pin)

Something you own (e.g.: digital pass, smartphone)

Something you are (e.g.: fingerprint, iris scan)

Introducing Multi-factor Authentication



Something you do
(e.g.: a gesture to unlock your phone)

Somewhere you are
(e.g.: IP address – typically this factor is not sufficient on its own)

Introducing Multi-factor Authentication

Withdrawing money from an ATM

- Something you know
- Something you have

Introducing Multi-factor Authentication

The authentication factors used in MFA/2FA must be of different types

- Username/pw + one-time password (as proof of something you possess)

One-time password

A generated unique password that's only valid for a single login session

Supporting MFA with a One-Time Password Through Email

One-time password

- As it cannot be reused it's not vulnerable for replay attacks

Can be delivered to the user via email to a verified email address

Supporting MFA with a One-Time Password Through Email

Sending an OTP via email isn’t true MFA

“The ability to receive email messages does not generally prove the possession of a specific device”

NIST

Supporting MFA with a One-Time Password Through Email

Sending an OTP via email isn't true MFA

… but it's still better than no "second" factor at all

- Use as backup when no better means are available

Supporting MFA with an Authenticator Application

An authenticator app (Google authenticator, Microsoft authenticator, ...) is an example of a soft OTP implementation

Soft one-time password (OTP)

A piece of software that generates OTPs on the device

Supporting MFA with an Authenticator Application

HMAC-based OTP (HOTP)

OTP based on Hashed Message Authentication Code

Event-based OTP algorithm, where the moving factor is an event counter

https://tools.ietf.org/html/rfc4226

Time-based OTP (TOTP)

OTP based on time

The moving factor is time, which results in short-lived OTPs

https://tools.ietf.org/html/rfc6238

Time on the device which generates the TOTP must be synced with the time on the server which validates the TOTP

Most authenticator apps (including Google authenticator and Microsoft authenticator) support HOTP and TOTP

A TOTP is generated from a secret

- Client and server must know that secret
- If, at the same time, the TOTP is generated from the same secret, they match

Safely transmitting the secret is essential

- Don’t send it over the wire
- Let the user manually input it in the authenticator app, or scan a QR code from the authenticator app

```
otpauth://totp/Marvin:kevin@someemailprovider.com
?secret=<16charsecret>&issuer=Marvin
```

TOTP Secret QR Code URL

Follows otpauth://TYPE/LABEL?PARAMETERS format

```
otpauth://totp/Marvin:kevin@someemailprovider.com
?secret=<16charsecret>&issuer=Marvin
```

TOTP Secret QR Code URL

Type: fixed value, totp

```
otpauth://totp/Marvin:kevin@someemailprovider.com
?secret=<16charsecret>&issuer=Marvin
```

TOTP Secret QR Code URL

Label: typically a combination of the issuer + user

```
otpauth://totp/Marvin:kevin@someemailprovider.com
?secret=<16charsecret>&issuer=Marvin
```

TOTP Secret QR Code URL

Secret: the alphanumerical secret (16 characters)

```
otpauth://totp/Marvin:kevin@someemailprovider.com
?secret=<16charsecret>&issuer=Marvin
```

TOTP Secret QR Code URL

Issuer: the issuer of the secret (our IDP)

Authentication

- A random number is generated for use at a fixed interval from the secret – this is the TOTP
- The user inputs the TOTP at IDP level
- The IDP generates a TOTP using the same secret
- When both match, authentication is successful (typically the last few TOTPs are accepted)

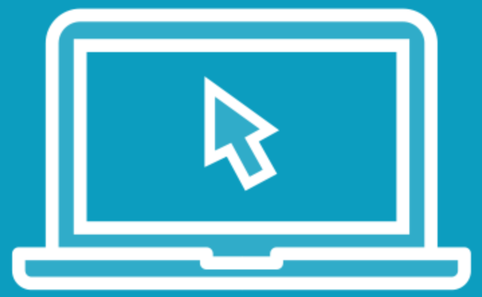
Often, an additional layer of security is provided by only allowing the user to open the authenticator app with a pin code, fingerprint or face scan

The authentication factor provided by the app is proof of something you own

Demo

Supporting MFA with an authenticator application (enhancing the database schema)

Demo



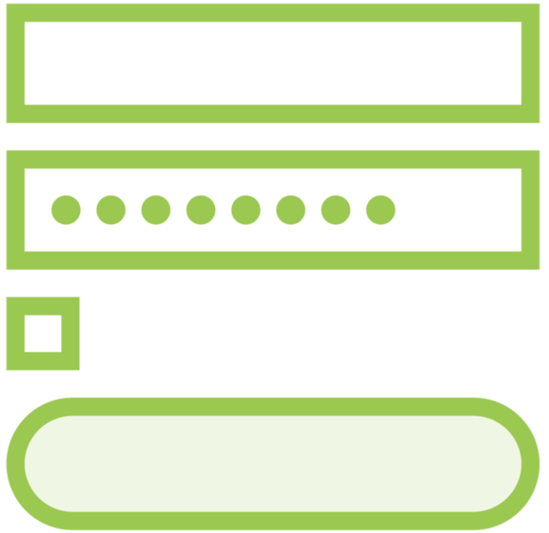
Supporting MFA with an authenticator application (registration)

Supporting MFA with an Authenticator Application - Authentication

On each authentication

Only when local credentials are used

Depending on information from the third-party provider



For a specific set of users and/or client applications

Supporting MFA with an Authenticator Application - Authentication

Upcoming demo

- Additional field on the local login view

Demo

Supporting MFA with an authenticator application (authentication)

Summary

Multi-factor authentication provides identification of users by means of the combination of two or more different factors

- A factor should be seen as a type of authentication

Summary

Commonly used types

- Something you know
- Something you own
- Something you are

Less common types

- Somewhere you are
- Something you do

Summary

It’s essential to use different factors – using the same factor twice isn’t MFA/2FA

Summary

OTP

- Generated unique password that's only valid for a single login session
- HOTP, TOTP

Up Next:
Integrating with ASP.NET Core Identity