bright inventions.

OPTIMISE SIGN-IN EXPERIENCE

EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

2023

Passwords

Social Media

Security

Privacy

OpenID Connect

GDPR

Table of contents

Applications require signing user in

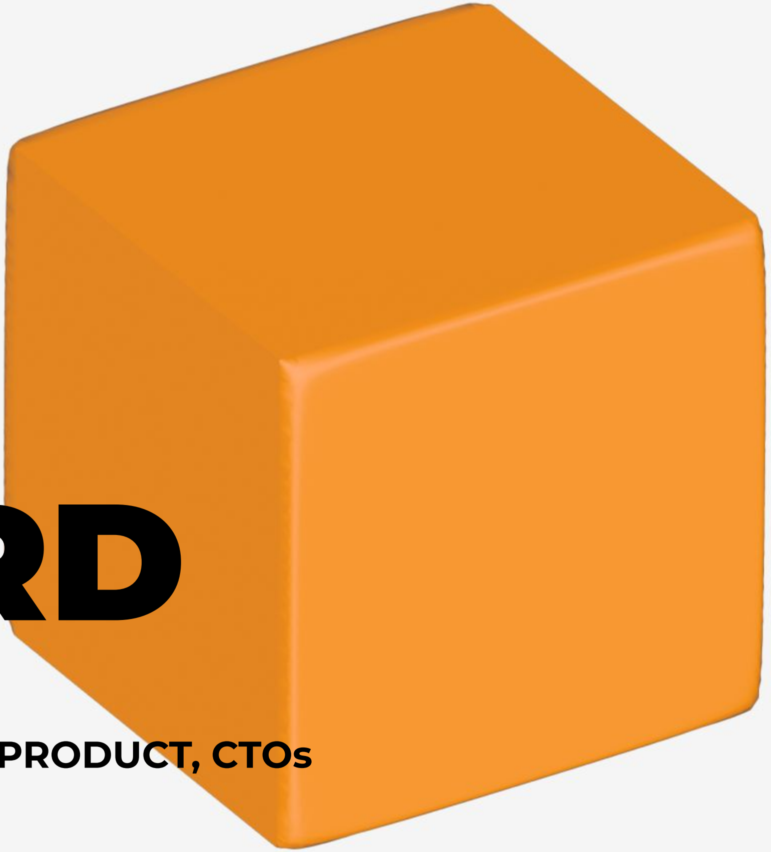
Passwords | **Social Media** | **Security**

Privacy | **OpenID Connect** | **GDPR**

Most applications require users to sign in to provide a personalised features e.g. synchronisation of private content. The sign-in can be realised in multiple ways which differ in terms of user experience.

In the following pages we will help you answer the following questions:

- Which sign-in option is the best for your application?
- What security aspects require consideration?
- How is GDPR related to sign-in?
- Should I build it myself or use off-the-shelf software?



ENTER PASSWORD

EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Password is the oldest and by far the most popular method of signing user in. Historically it required a user to memorise a sequence characters and keep it private.

The strength of the password security is directly linked to its length. However, the memoization requirement stands in direct opposition to security. Most users choose a short password, so they can remember it easily. However, short passwords are easier to guess, hence many applications will rightfully require longer and longer passwords.

The password managers alleviate the memory issue. The user is only required to remember a single, the longer the better, hard to guess master password. Other passwords used in applications are randomly generated and securely kept inside password manager. However, **majority of <u>users still don't use a password manager</u>**.

Password **pros**

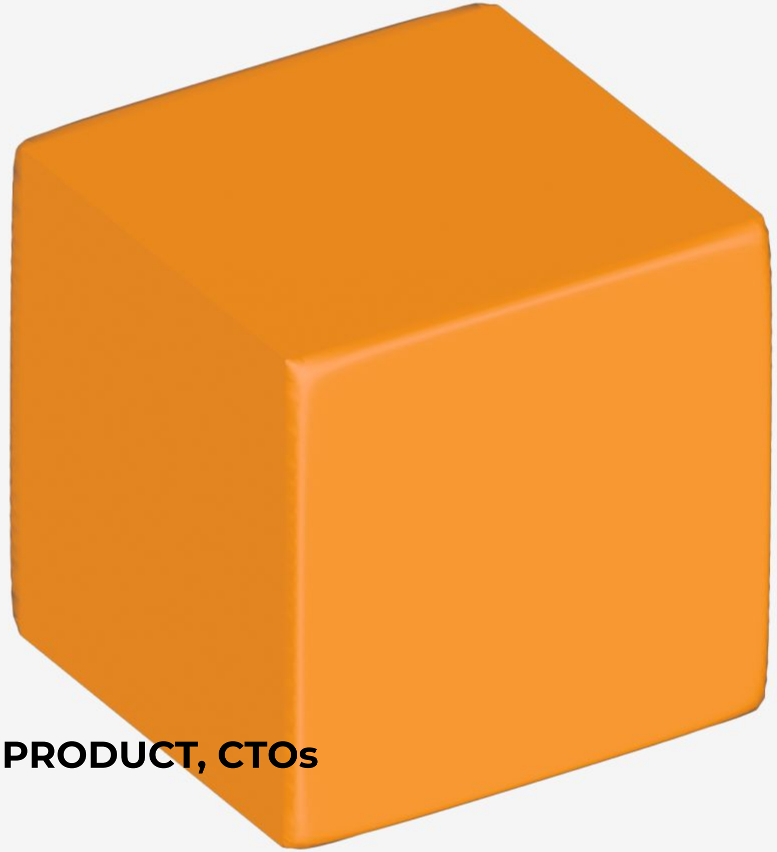
- Accessible to everyone regardless if they have social media or identity provider account.
- No user information shared with 3rd party.
- Plethora of established UX patterns available.
- Good implementations available freely.
- Supported by most of the shelf solutions.

Password **cons**

- Users are required to remember the password.
- Custom implementations while seemingly easy are notoriously difficult to keep secure.
- Does not aid the sign-up process.
- Recovery mechanisms are necessary to implement and are often harder to implement securely.

SMS CODE

EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs



SMS code is a sign-in method popularised by mobile applications. The method requires sending a short code, usually in form of couple of digits, to user’s phone number. To improve UX the code can often be entered automatically. This is possible with mobile application cooperating with a device operating system to maintain privacy.

In addition, the method provides a confirmation of a user’s phone number. This may be a requirement depending on the nature of the application.

Sending sign-in SMS may turn out to be costly if abused. A care must be taken to rule out or reduce possibility of misuse or fraud. Moreover, in recent years phone number spoofing became more popular targeted attack vector.

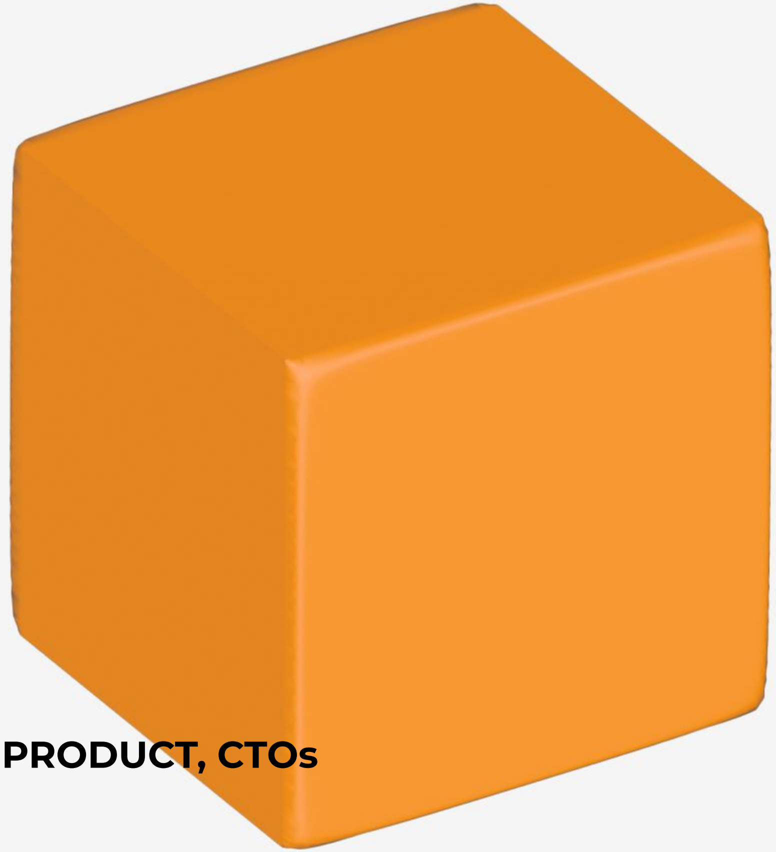
SMS pros

- Accessible to majority of mobile users regardless if they have social media or identity provider account.
- Very smooth user experience.
- Good implementations available.
- Supported by many of the shelf solutions.

SMS cons

- Works only if user has a phone number thus additional sign-in method may be needed.
- Desktop use may be awkward, so another implementation, e.g. a regular password, may provide better UX.
- Easy to abuse by hackers or even script kiddies.

EMAIL CODE



EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Email is a popular and widely-used authentication method in software solutions. This method involves sending a verification code or link to the user's email address during the registration or login process.

Once the user receives the email, they must follow the instructions to verify their identity and gain access to the software solution. **Email authentication is a convenient and effective way to ensure that only authorized users can access the software.**

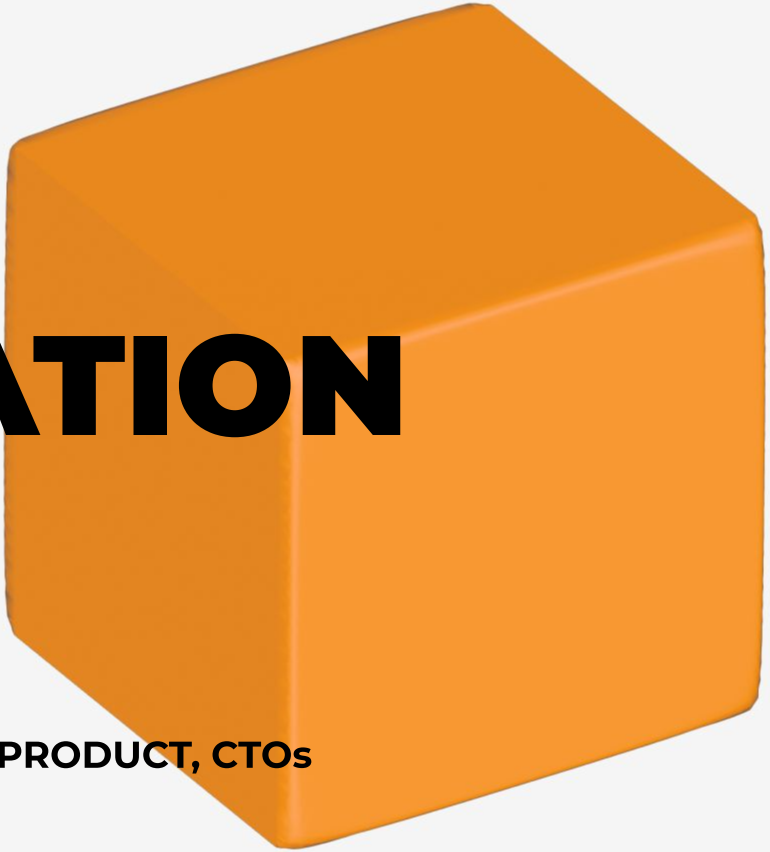
The method provides a good level of security. However, the UX is negatively affected. **The user is forced to change application context when logging in**. There’s no good way to make the experience as smooth on mobile as in the case of SMS codes .

Email pros

- Accessible to majority of users regardless if they have social media or identity provider account.
- User’s security burden is not increased.
- Good implementations available.
- We keep a reliable channel to communicate with a user.

Email cons

- The user experience is impaired.
- A robust email delivery is required.
- Rate limiting needs to be put in place.



OPEN AUTHORIZATION (OAuth)

EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

OAuth is an open standard authentication and authorization protocol. **It allows users to grant third-party applications access to their resources on a separate service without sharing their passwords**.

OAuth enables users to sign in to applications **using their existing credentials from another service, such as Google, Facebook, or X**, and provides a secure and standardized way for applications to access the user's data on those services.

OAuth is generally regarded as authorization protocol. In other words, while using it for authentication is possible it may not be always straightforward.

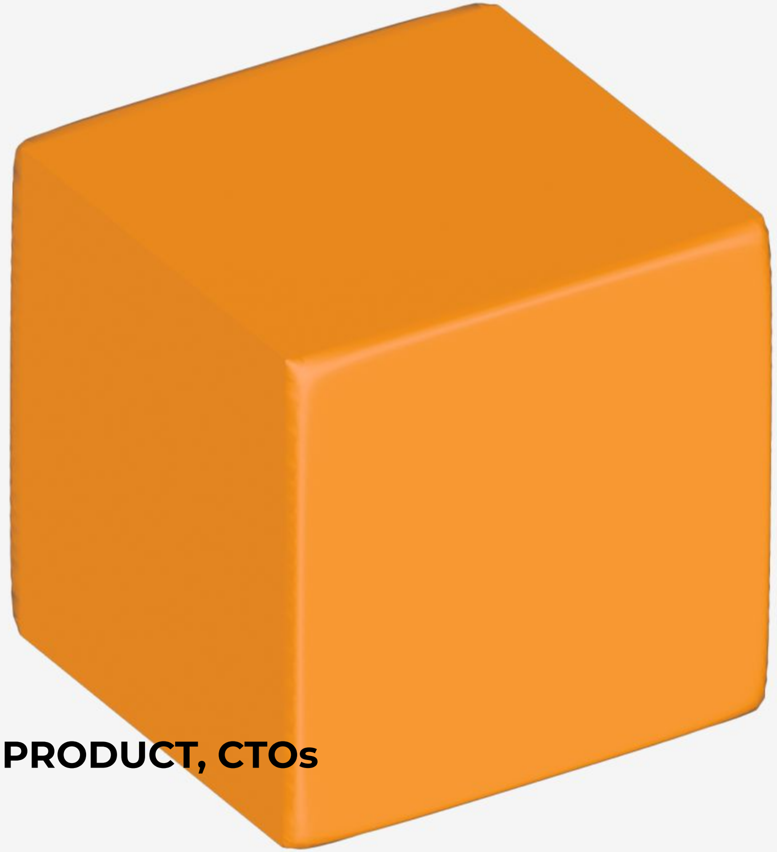
Oauth pros

- Accessible to majority of users that have social media or identity provider account.
- User’s security burden is not increased.
- Good implementations are available.
- We open the possibility to integrate with user’s social media accounts.

Oauth cons

- Authentication usage may not always be obvious.
- Supporting multiple providers is often necessary.
- Experience is required to correctly assess security of the solution.

OpenID CONNECT



EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

OpenID Connect (OIDC) is an authentication protocol built on top of OAuth 2.0. **It allows for secure user authentication across different applications and services**. OIDC adds an identity layer to OAuth 2.0, providing a standardized way for applications to request and receive information about the user's identity. This information can include the user's name, email address, profile picture, and other relevant data.

OIDC is designed to be highly flexible and interoperable, making it easy for developers to integrate into their applications. It supports a variety of authentication mechanisms, including username and password, social login, and multi-factor authentication. OIDC also provides a range of security features, such as encrypted communication, signed tokens, and access control, to ensure that user data is protected at all times. Overall, OIDC provides a powerful and reliable way to authenticate users across different applications and services, while maintaining the highest levels of security and privacy.

OIDC pros

- Accessible to majority of users that have social media or identity provider account.
- User’s security burden is not increased.
- Good implementations available.
- We open the possibility to integrate with user’s social media accounts.

OIDC cons

- Supporting multiple providers is often necessary.
- Experience is required to correctly assess security of the solution.
- The implementation may be harder to debug.

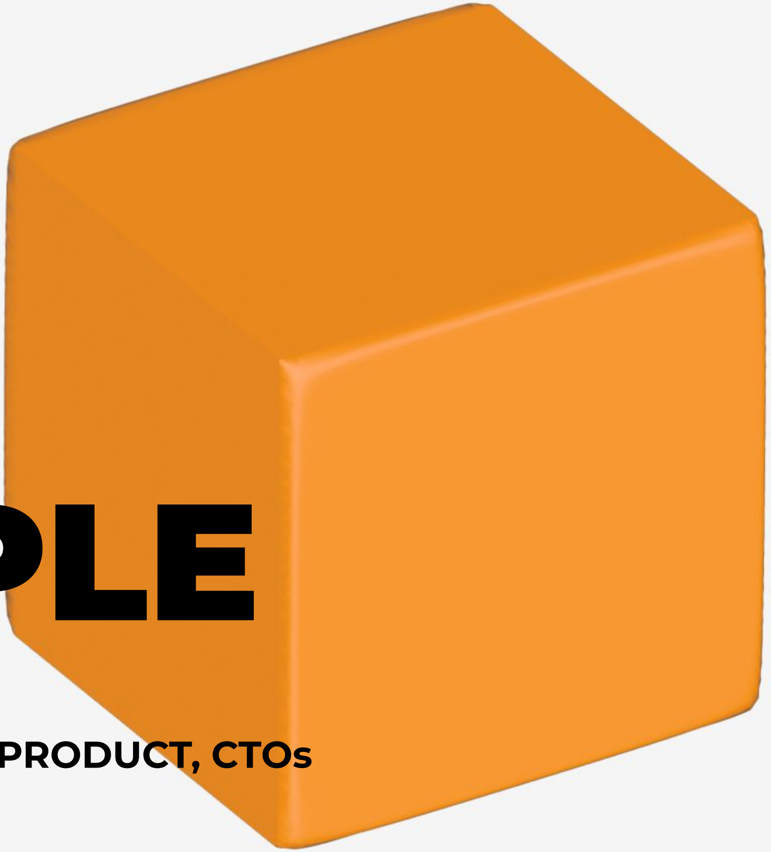
OpenID Connect vs OAuth

Feature	**OpenID Connect (OIDC)**	**OAuth**
Authentication	Adds an identity layer to OAuth, enabling user identity.	Enables access delegation for third-party applications.
Authorization	Provides a way to obtain user consent for sharing identity.	Provides a way to obtain user consent for accessing data.
Token types	Provides an ID Token and Access Token.	Provides an Access Token.
User Information	Provides user profile information.	Does not provide user profile information.

OpenID Connect vs OAuth

Feature	**OpenID Connect (OIDC)**	**OAuth**
Token Validation	Requires ID token validation.	Requires access token validation.
Scopes	Can include standard and custom scopes.	Only includes scopes for accessing resources.
Security Features	Provides a range of security features for identity.	Provides security features for access delegation.
Standardization	A standard identity protocol.	A standard access delegation protocol.
Use case	User authentication across applications and services.	Authorization for third-party applications.

SIGN IN WITH APPLE



EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Sign In with Apple is a secure authentication method introduced by Apple for iOS, macOS, and web applications. **It allows users to sign in to third-party services using their Apple ID**. The key advantage is its privacy-focused approach, offering users the option to share their email address or use a randomly generated one.

Implementing **Sign In with Apple is beneficial for developers targeting Apple users and apps within the Apple ecosystem**. It streamlines the authentication process, complies with App Store guidelines, and is built on industry standards like OAuth 2.0 and OpenID Connect for secure authentication and token exchange.

In summary, Sign In with Apple providing a seamless and secure login experience, compliance with guidelines, and integration with existing authentication protocols. With its focus on user privacy and the convenience it offers, **Sign In with Apple is a valuable authentication option for applications and services targeting Apple users**.

When an app must include Sign in with Apple?

Apps that use a third-party or social login service (such as Facebook Login, Google Sign-In, Login with Amazon, or WeChat Login) to set up or authenticate the user's primary account with the app must also offer Sign in with Apple as an equivalent option. A user's primary account is the account they establish with your app for the purposes of identifying themselves, signing in, and accessing your features and associated services.

Sign in with Apple is not required if:

- Your app exclusively uses your company's own account setup and sign-in systems.
- Your app is an education, enterprise, or business app that requires the user to sign in with an existing education
  or enterprise account.
- Your app uses a government or industry-backed citizen identification system or electronic ID to authenticate users.
- Your app is a client for a specific third-party service and users are required to sign in to their mail, social media,
  or other third-party account directly to access their content.

OIDC vs Sign in with Apple

	OIDC	**Sign In with Apple**
Definition	Open standard for authentication and authorization, built on top of OAuth 2.0.	Authentication method introduced by Apple for iOS, macOS, and web applications.
User Identification	Relies on unique user identifier (subject).	Uses Apple ID as the unique user identifier.
Privacy Focus	Depends on the implementation and configuration, it can vary.	Emphasizes user privacy by offering anonymous email options and not sharing personal information with third-party apps.

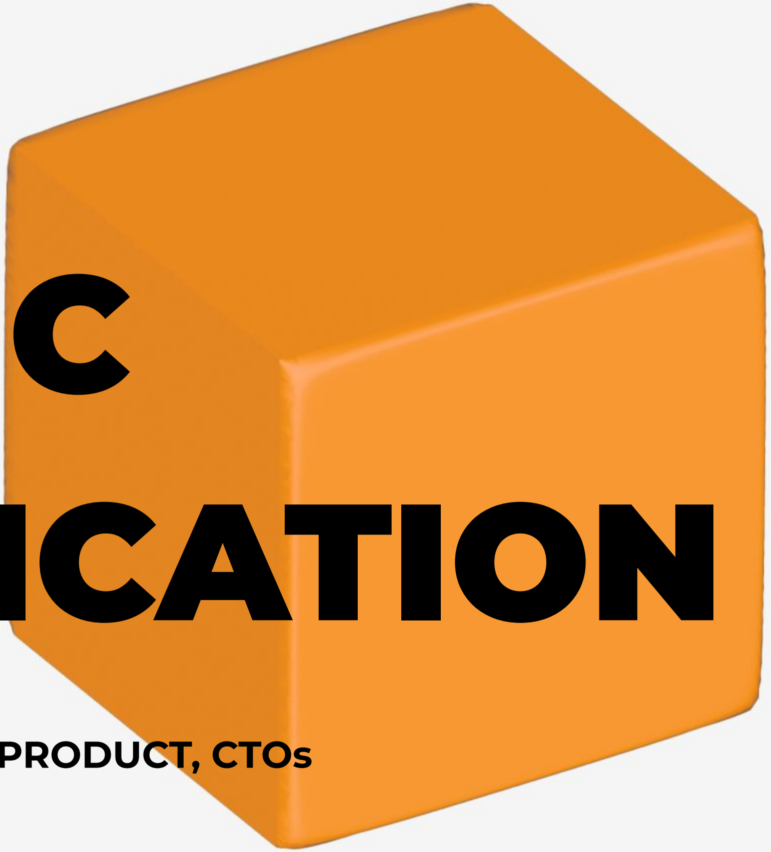
OIDC vs Sign in with Apple (continuation)

	OIDC	Sign In with Apple
Single Sign-On (SSO)	Can be used for SSO through the exchange of tokens and user information.	Provides convenient SSO functionality for Apple ecosystem applications and services.
Third-Party Integration	Can be integrated with various identity providers and platforms.	Designed primarily for use within the Apple ecosystem, may be limited when integrating with non-Apple systems.

OIDC vs Sign in with Apple (continuation)

	OIDC	Sign In with Apple
Developer Support	Widely supported, with SDKs and libraries available for different languages.	Apple provides developer documentation and resources for integrating Sign In with Apple.
Platform Compatibility	Can be used across different platforms and web applications.	Primarily designed for iOS, macOS, and web applications.

BIOMETRIC AUTHENTICATION



EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Biometric authentication refers to the use of unique physical or behavioral characteristics of individuals to verify their identity. It leverages biometric data, such as **fingerprints, facial features, voice patterns, iris scans**, or even behavioral traits like **typing patterns or gait**, to authenticate users. Biometric authentication offers a convenient and secure alternative to traditional username/password-based authentication methods.

The process of biometric authentication typically **involves two steps: enrollment and verification**. During enrollment, the user's biometric data is captured and stored securely. This data is then used as a reference for subsequent verification attempts. When a user attempts to authenticate, their biometric data is compared to the stored reference to determine a match.

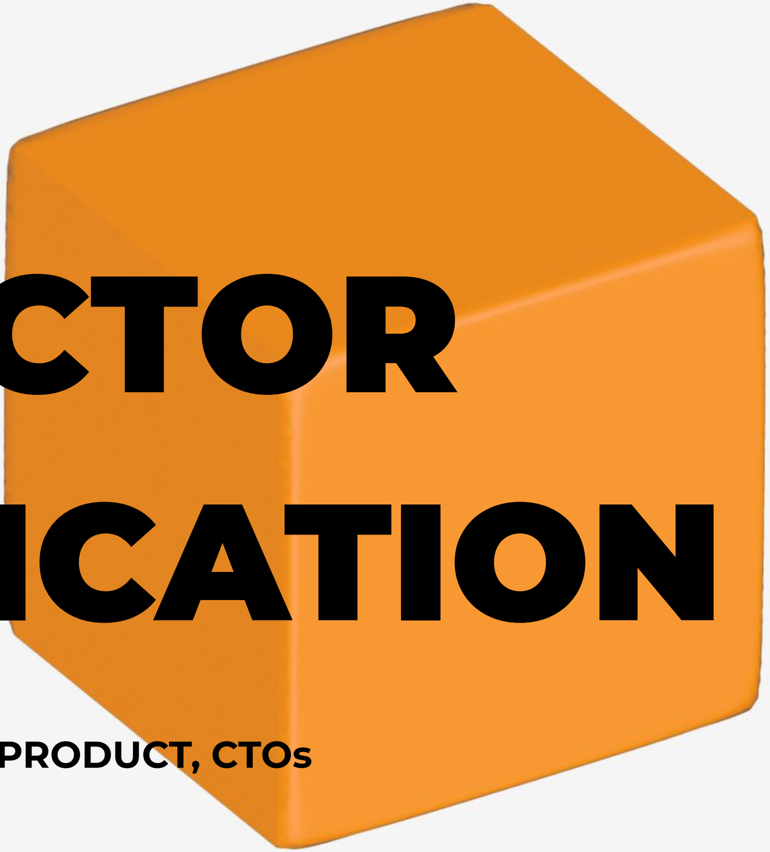
Biometric authentication is used in various domains, including smartphones, laptops, physical access control systems, financial services, and border control, due to its effectiveness and convenience in providing secure user identification.

Biometric pros

- Enhanced security by use of unique characteristics, reducing the risk of unauthorized access.
- Convenience through a seamless user experience.
- Reduced reliance on passwords limiting password-related risks and vulnerabilities.
- Accessibility by inclusive approach for individuals with disabilities.
- Non-repudiation with stronger accountability and identity proof.

Biometric cons

- Privacy concerns about biometric data storage security.
- Consequences of biometric data being compromised are harder to tackle as it cannot be easily changed.
- Cost and infrastructure requirements as it needs specialized hardware and infrastructure.
- User acceptance and consent: concerns and hesitations about providing biometric data.



MULTI-FACTOR AUTHENTICATION

EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Multi-factor authentication (MFA) is a security measure that requires users to provide multiple forms of identification to verify their identity and gain access to a system or application. It adds an extra layer of protection by combining two or more independent factors, typically categorized as something the user knows, something the user possesses, and something the user is.

This multi-layered approach significantly enhances security compared to relying on a single factor, such as a password. By requiring multiple factors for authentication, MFA reduces the risk of unauthorized access even if one factor is compromised.

Most popular 2nd factors

1. **One-Time Passwords (OTP)**

   One-time passwords are **temporary codes that are generated for a single authentication session**. They are typically sent to the user via SMS, email, or generated by an authenticator app. The user enters the OTP alongside their password or other primary factor to complete the authentication process.

2. **Authenticator Apps**

   Authenticator apps, also known as OTP apps, **generate time-based or event-based OTPs directly on the user's mobile device**. These apps, such as Google Authenticator or Microsoft Authenticator, provide a more secure and convenient alternative to receiving OTPs via SMS. Users can access the app to retrieve the current OTP when prompted during the authentication process.

2nd factor authentication standards

1. **Time-based One-Time Password (TOTP)**

   TOTP is an algorithm that generates **one-time passwords based on a shared secret key and the current time**. It is specified in RFC 6238 and is widely used for second factor authentication, particularly with authenticator apps.

2. **Universal Second Factor (U2F)**

   U2F is an open authentication standard developed by the FIDO Alliance. It allows for **strong second factor authentication using hardware tokens or devices**. U2F is specified in the FIDO U2F specifications and provides a standardized approach to enhance security for online services.

3. **WebAuthn**

   Web Authentication (WebAuthn) is a web standard developed by the World Wide Web Consortium (W3C) and FIDO Alliance. It provides a standard **API for web applications to support strong authentication, including second factor authentication**, using various methods such as biometrics, hardware tokens, or platform authenticators.

TOTP pros

- **Strong Security:** TOTP provides a high level of security by generating unique passwords that are valid only for a short period of time.
- **Wide Adoption:** TOTP is widely adopted and supported by various authentication systems, platforms, and authenticator apps.
- **User Convenience:** TOTP can be generated using mobile apps, which makes it convenient for users to generate and enter the one-time passwords during authentication.
- **No Network Dependency:** TOTP does not rely on an active network connection during authentication
- **Non-repudiation** with stronger accountability and identity proof.

TOTP cons

- **Dependency on Clocks:** TOTP relies on the accurate timekeeping of both the server and the user's device. Any discrepancy or time drift can result in failed authentication.
- **User Education and Setup:** Users need to be educated on set up and use, including scanning QR codes or entering secret keys during the initial setup.
- **Recovery Challenges:** In case of device loss or failure, users may face challenges in recovering access to their accounts since TOTP does not have built-in account recovery mechanisms.

U2F pros

- **Strong Security**: U2F provides strong security by utilizing hardware tokens or devices for authentication. This hardware-based approach offers protection against phishing, replay attacks, and other common security threats.
- **User-Friendly Experience**: U2F offers a seamless and user-friendly authentication experience. Users simply need to plug in or tap their U2F hardware token.
- **Phishing Resistance:** U2F provides strong protection against phishing attacks since it relies on cryptographic keys that are unique to each service.

U2F cons

- **Hardware Dependency**: U2F requires users to have dedicated hardware tokens or devices for authentication, which can be an additional cost
- **Limited Device Compatibility**: U2F requires support from devices and applications. While the standard is gaining broader adoption, not all platforms and services may fully support U2F yet.
- **Backup and Recovery**: U2F does not inherently provide account recovery mechanisms, and losing or damaging the hardware token may result in a loss of access to the associated account.
- **Deployment and User Adoption**: Deploying U2F may require user education and awareness to ensure proper usage and adoption of hardware tokens.

WebAuthn pros

- **Strong Security**: WebAuthn provides strong security by leveraging public key cryptography and eliminating the need for passwords.
- **Passwordless Authentication**: WebAuthn enables passwordless authentication, eliminating the reliance on passwords as the primary authentication method.
- **User Convenience**: WebAuthn offers a convenient user experience by allowing users to authenticate using biometrics, security keys, or other authentication factors supported by their devices.
- **Privacy Enhancements**: WebAuthn allows for anonymous authentication, reducing the exposure of user identities and personal information.

WebAuthn cons

- **Implementation Complexity**: Integrating WebAuthn into existing systems may require additional development effort and expertise.
- **Device Support**: WebAuthn's support may vary across different devices and browsers.
- **Legacy System Compatibility:** Implementing WebAuthn in systems that rely on legacy authentication methods or infrastructure may require additional considerations and potential **modifications.**
- **Initial Setup Complexity**: The initial setup process for WebAuthn, such as registering a device or generating cryptographic keys, may introduce some complexity for users, especially for first-time setup.

2nd factor may be required by law

The requirement for providing two-factor authentication (2FA) can vary depending on the jurisdiction and specific laws and regulations in place. Here are a few **examples where 2FA may be required by law in certain cases**:

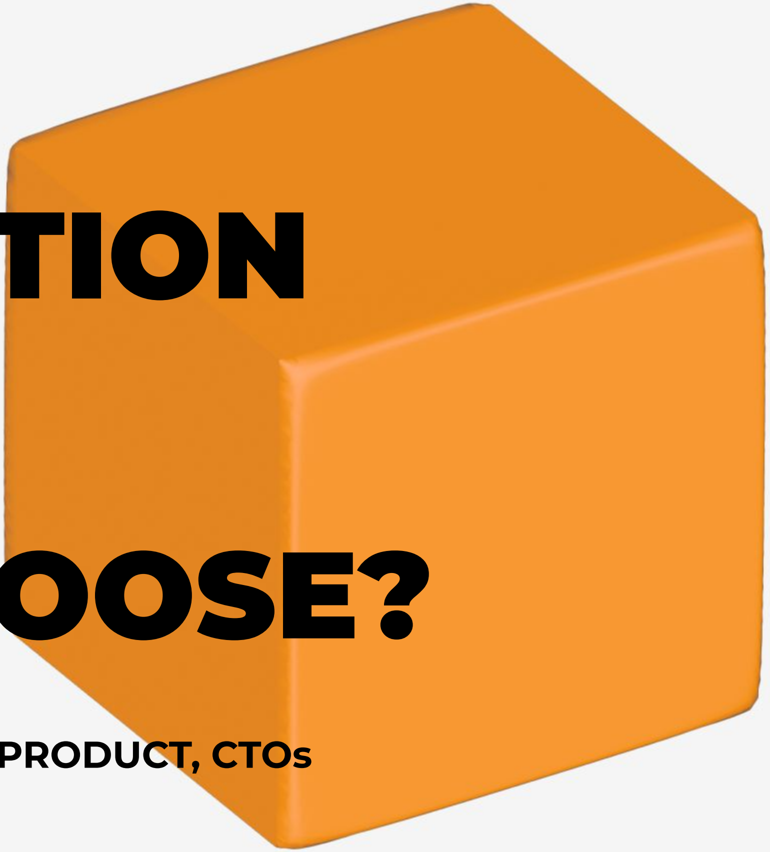
- **Payment Card Industry Data Security Standard (PCI DSS)**

The PCI DSS is a set of security standards that organizations handling payment card data must comply with. While it does not explicitly mandate 2FA, it strongly recommends the use of multi-factor authentication as a security control to protect cardholder data. Some organizations may interpret this recommendation as a requirement and implement 2FA as part of their PCI DSS compliance efforts.

- **Health Insurance Portability and Accountability Act (HIPAA)**

HIPAA is a U.S. law that sets standards for the protection and privacy of individuals' health information. While HIPAA does not explicitly require 2FA, it mandates the implementation of security safeguards to protect electronic protected health information (ePHI). 2FA can be seen as a recommended security measure to meet HIPAA's requirements and ensure the confidentiality and integrity of ePHI.

WHICH AUTHENTICATION METHOD SHOULD I CHOOSE?



EBOOK FOR PRODUCT OWNERS, HEADS OF PRODUCT, CTOs

Authentication method decision tree

Here's a decision tree that takes into account user experience, regulatory requirements, privacy concerns, and security when selecting an authentication method for an application:

1. Are there specific regulatory or compliance requirements?

- If yes, review the regulations or compliance standards to identify any mandated or recommended authentication methods. Implement the required or recommended methods to ensure compliance.
- If no specific regulatory requirements exist, proceed to the next question.

2. What is the sensitivity of the application or data?

- If the application involves highly sensitive data, prioritize authentication methods that provide strong security measures, such as multi-factor authentication (MFA) or biometrics, to mitigate risks.
- If the sensitivity is moderate, consider authentication methods like password-based authentication or email verification, while ensuring adequate security controls are in place.

Authentication method decision tree

3. Are there privacy concerns associated with the application?

- If privacy is a significant concern, prioritize authentication methods that minimize the collection and storage of personal data, such as federated authentication or token-based authentication.
- If privacy is less of a concern, options like username/password or social login (OAuth) can be considered.

4. What are the user experience requirements?

- If providing a seamless and convenient user experience is paramount, consider authentication methods like social login, single sign-on (SSO), or biometrics that minimize user effort and friction.
- If user experience is a consideration but balanced with security, options like passwordless authentication or remember me functionality can be explored.

Authentication method decision tree

5. Does the application support multiple platforms and devices?

- If the application needs to support various platforms and devices, consider authentication methods that are widely supported and compatible, such as standards like OpenID Connect (OIDC) or Security Assertion Markup Language (SAML).
- If the application is specific to certain platforms or devices, explore authentication methods that are optimized for those platforms, such as biometrics on mobile devices.

6. Are there budget or resource constraints?

- If budget or resource constraints exist, focus on authentication methods that strike a balance between security, privacy, and usability within the available resources, such as username/password with additional security measures.
- If budget allows, consider investing in more robust authentication methods like MFA or hardware tokens for enhanced security and privacy.

author

Piotr Mionskowski,
Head of Technology & Partner
@ bright inventions

Piotr is a Head of Technology with **over 12 years of professional experience** in backend APIs, distributed systems, backend frameworks, and databases. He possesses an **exceptional ability to learn almost every new technology that appears**. Always up-to-date. Always focused.

Bright Inventions is a software consulting studio based in Gdansk, Poland **operating since 2012**. Our expertise in **mobile, web, blockchain and IOT** systems has been highly appreciated by our clients from **UK, Germany, Netherlands, Norway, Israel** and more.

bright inventions.

If you have any questions, contact me

Piotr Mionskowski,
Head of Technology & Partner
@ bright inventions

email me to discuss your solution

Book a free consultation with our team.
We will:

- evaluate your sign-in experience,
- provide you with recommendations,
- estimate the features or new solutions you want to build.

Trusted by JUST EAT Takeaway.com Relevo Circles finebite