



Building Secure Multi-Factor Authentication

Three best practices
for engineering and
product leaders

Okta Inc.
301 Brannan Street
San Francisco, CA 94107

info@okta.com
1-888-722-7871

Introduction

As threats to password security have increased in recent years, multi-factor authentication (MFA) has rapidly gained adoption as a method for increasing the assurance of authentication for consumer and enterprise web and mobile applications.

Authentication is generally accomplished by validating one of three types of factors: something you know (e.g. a password), something you have (e.g. an ID card), and something you are (e.g. a fingerprint). Multi-factor authentication employs two or more types of factors. Web and mobile products most commonly employ the use of multi-factor authentication with a password used in conjunction with a time-based token that the user possesses, though approaches to MFA vary widely and present different tradeoffs.

Volumes have been written about how to design secure authentication for electronic systems. In this note we provide some selected practical advice for people building multi-factor authentication for their applications, based on our observations working with engineering and product teams. We explore three ways to increase the security of your MFA feature:

1. Understand and manage the vulnerability of your account recovery flow
2. Protect your login flow from brute force attacks
3. Design to manage tradeoffs between risk, usability, and cost

Throughout this note we assume that the password has been compromised, and examine the second factor through this lens.

Understand and manage the vulnerability of your account recovery flow

Multi-factor authentication is only as secure as its account recovery flows. In many highly publicized recent cases, attackers have been able to exploit vulnerabilities in the account recovery process to gain control of an account.

For example, Acme's web application provides for MFA based on a soft token app installed on a user's phone and allows the user to enroll a phone number for the purposes of receiving a backup second factor for account recovery in the event that the user is unable to access their soft token. The strength of Acme's second factor now depends on the strength of the telecom provider's processes for authenticating the customer and forwarding calls or sms. Will the attacker be able to impersonate the user and convince or pressure a customer service rep to route calls or sms to a number she controls?

Every second factor will need a method for replacement, and so this begs the question of how to develop secure recovery flows. Here are some tips for designing a secure recovery flow for your second factor, noting that different approaches will suit different circumstances:

- Independence of primary and secondary factors. Separate the recovery of the second factor from the recovery of the primary factor. Should an attacker gain access to primary authentication factor, the second factor becomes immaterial if it can be reset with possession of just the password. Further, the recovery flow for the second factor should be completely separate from the recovery flow for the password. For example, if an email message is the method for recovering the password, make sure to recover the second factor through an altogether separate channel.
- Involve an administrator. An administrator can in many scenarios implement a sophisticated high assurance authentication method.

In enterprise scenarios, companies will be in the best position to authenticate members of their own organization through shared secrets derived from the content of the employee's work or profile, the company, and human relationships. One notable approach is to ask an employee's manager to authenticate the user and then authorize IT to execute the MFA reset.

Protect login flows from brute force attacks

In consumer scenarios an administrator will be able to interrogate a user across a large set of shared secrets. For example, upon onboarding, consumer banking applications will collect a large set of obscure personal details that become shared secrets for the purposes of account recovery. Recent events in the person's history with the application or company can also constitute viable shared secrets. The evaluation of a set of shared secrets can be automated via web or voice and can in many cases provide better assurance than a human through lower vulnerability to social engineering.

- Provide a backup second factor. Many scenarios require an automated method for recovering the second factor (for example, products serving large numbers of users where 1:1 support is prohibitively expensive, or there is a need to reduce operational costs). Enrolling the user in more than one second factor at the time of onboarding allows the user to recover a second factor by completing authentication through a backup second factor. One notable, simple and low cost example is to provide users with a card (either physical or printable) with a set of codes that can be used only once, and that can be used as a backup second factor.

As the availability of inexpensive computing resources increases so does the vulnerability of authentication systems to brute force guessing attacks. However several simple techniques can be used to significantly improve the security of your multi-factor authentication in the circumstance where the password has been compromised:

- Login flow sequence, rate limits, and account locking. Placing the challenge for the second factor on a page beneath the login page has two benefits. First, it protects your user from an attack aimed at locking them out of their account once a failed login attempts limit is reached (with rate limits applied to the primary factor). Second, obscuring the second factor provides an attacker with less visibility into another layer of security. Implement a rate limit and lock policy on the second factor. The probability that a user enters their token incorrectly multiple times is low. As such, your suspicion of attack should grow with each failed attempt. Response times should grow with each subsequent attempt to decrease the aggregate number of attempts possible per unit time, with a complete account lockout (where feasible) upon several consecutive failed attempts. For time-based second factors, manage rate limits according to the life of the token.
- Logs and alerts. Collect and analyze unsuccessful second factor attempts. In the event of several failed second-factor challenges, alert the user or an administrator of this suspicious behavior, and prompt the user to enroll a new token.
- Use an out-of-band token. A second factor that is verified through a channel separate from the primary factor adds extra protection against brute force attacks (and phishing). For example, a popular new factor sends the user a push notification on a mobile phone with details about the authentication request and a prompt to accept or deny the request. This channel is inaccessible to a traditional brute force guessing approach.

Design to manage risk, usability and cost

The design of a multi-factor authentication feature will have significant implications on security, usability, and cost in any context. A higher assurance second factor can in some cases present the burden of increased hassle for end-users and administrators which can impact the adoption of MFA for your product and thereby decrease security. Here are some best practices for balancing risk, usability and cost:

- Offer a spectrum of options to serve diverse user populations. Different user populations present different levels of risk and hence, warrant different levels of assurance. For example, an administrator can have a larger scope of access than an individual user. As such, you may wish to provide relatively stronger second factors for administrators, while offering more convenient options for users. In consumer scenarios different users will have different preferences and a lower assurance more convenient option that is actually used may provide more security than a high assurance option that lacks adoption.
- Support federated authentication. In enterprise scenarios many companies are implementing authentication and MFA locally for identities they manage, and federating to resources. This approach allows product development teams to outsource administration of policy and security processes to customers. Enabling customers to implement MFA independently allows them to optimize across the aforementioned considerations according to their specific circumstances and constraints. For example, a customer can design administration of account recovery to suit their specific IT function. This outsourced approach has the added advantage of allowing users to use one token for access to all resources.

Conclusion

Multi-factor authentication is a compelling method for application developers to increase the security of access to their applications. Many steps must be taken to ensure the security of an MFA feature, including analyzing the 2nd factor recovery flow, designing against brute force attacks, and balancing security, usability and cost.

About Okta

Okta is the foundation for secure connections between people and technology. By harnessing the power of the cloud, Okta allows people to access applications on any device at any time, while still enforcing strong security policies. It integrates directly with an organization's existing directories and identity systems, as well as 4,000+ applications. Because Okta runs on an integrated platform, organizations can implement the service quickly at large scale and low total cost. More than 2,500 customers, including Adobe, Allergan, Chiquita, LinkedIn, MGM Resorts International and Western Union, trust Okta to help their organizations work faster, boost revenue and stay secure.

For more information, visit us at www.okta.com or follow us on www.okta.com/blog.