Engineer

Multi-factor authentication

sign-in that needs two or more different kinds of proof

What it is

Multi-factor authentication, usually shortened to MFA, is a sign-in that accepts a claimed identity only when it is backed by more than one kind of evidence. NIST SP 800-63B-4, the US federal guideline on authentication that many organisations outside government also follow, names the three kinds: something you know, such as a password; something you have, such as a phone or a security key; and something you are, such as a fingerprint. Its definition turns on one word. The system must require more than one distinct type of factor, so a password and the answer to a security question add up to a single factor, because both are things you know. Two-factor authentication, or 2FA, is the common case of exactly two.

The guideline accepts two ways of getting there. The first combines separate single-factor authenticators: a password, plus a phone app that shows one-time codes or a list of printed backup codes. The second uses one multi-factor authenticator, such as a security key or a phone that holds a private key and will only use it after a PIN or a fingerprint. In that case the PIN or fingerprint is called an activation factor. It is checked on the device itself, and the server receives only the result, a signature it can verify, so a single touch presents two factors.

Biometrics have a narrow place in this. NIST allows a fingerprint or a face only when a physical authenticator is also involved in the authentication, never as a factor on its own, and requires that a non-biometric alternative is always offered. The guideline is plain that biometric characteristics are not secret: a face can be photographed, and a fingerprint lifted from something its owner has touched.

Use the concept in a decision

Put “What is multi-factor authentication?” to work in a real decision. We can help relate the concept to the system you are building or buying. Accounts, passwords and permissions arranged so people hold exactly the access their job needs, and leavers lose it that day.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Wrote 1,000 RBAC rules to secure Linux servers — 1,000 access rules written to close a security gap

Wrote 1,000 RBAC rules to secure Linux servers Get in touch →

Why it matters

A password can be copied without its owner noticing: typed into a convincing copy of a sign-in page, guessed, or reused on a site whose database is later stolen. MFA makes a copied password insufficient on its own, because the attacker still needs something of a different kind.

NIST makes this the baseline for its second assurance level. At AAL2, the user must authenticate through two factors, either combined in one authenticator or supplied separately, and the combination must include a physical authenticator together with a password or a biometric comparison. Verifiers at that level must offer at least one phishing-resistant option, and federal agencies must require phishing-resistant authentication of their staff, contractors and partners.

The methods sold under the same three letters differ from one another more than MFA differs from a password alone, and the guideline sorts them clearly:

- **Codes sent by text message or voice call** are the only authenticator type NIST labels restricted. They may still be used, but the verifier must offer an alternative without that label, tell the user about the risk and have a plan for moving away from it. Changing the registered phone number counts as binding a new authenticator, with all the checks that implies.
- **Codes from an app and push prompts on a phone** are acceptable at AAL2 but are not phishing-resistant: what the person types or approves is not tied to the site that asked for it.
- **Cryptographic authenticators that bind their answer to the site**, such as passkeys and smart cards, are phishing-resistant. AAL3 requires one whose private key cannot be exported from its hardware.

Email is missing from the list on purpose: the guideline rules it out as a channel for out-of-band authentication.

What it looks like in practice

The code in an authenticator app is the most common second factor, and RFC 6238 (May 2011) defines it in a few pages. At enrolment the server generates a secret key and hands it to the app, usually as a QR code, so that both sides now hold the same key. To produce a code, each side divides the current Unix time by a time step, 30 seconds by default, and treats the whole number as a counter. It computes an HMAC of that counter with the key and cuts the result down to a short decimal number, using the method of HOTP, which RFC 4226 defined in 2005 and which requires at least six digits. In Python the whole calculation fits in a few lines:

```
import hashlib, hmac, struct

def totp(key: bytes, unix_time: int, digits: int = 6, step: int = 30) -> str:
    counter = struct.pack(">Q", unix_time // step)
    mac = hmac.new(key, counter, hashlib.sha1).digest()
    offset = mac[-1] & 0x0F
    code = struct.unpack(">I", mac[offset:offset + 4])[0] & 0x7FFFFFFF
    return str(code % 10**digits).zfill(digits)

print(totp(b"12345678901234567890", 59, digits=8))  # 94287082
```

The test vectors in the RFC use that 20-byte ASCII key and eight digits. At 59 seconds after the epoch the counter is 1, and the SHA-1 variant gives 94287082, which is what the function prints; the six-digit code an app would show is the last six of those digits, 287082. Nothing in the calculation touches the network, which is why the app works on a phone in flight mode, and also why the server has to keep its own copy of every user’s key.

Checking a code is the same calculation on the server, followed by a comparison. RFC 6238 recommends accepting at most one earlier time step to cover network delay, and forbids accepting a code a second time once it has been used successfully.

A passkey replaces the shared secret with a key pair. Under Web Authentication Level 3, a W3C Recommendation since 25 August 2026, the authenticator creates a key pair for one site, identified by its domain, and gives the server only the public key. At sign-in the browser passes along the origin it is actually connected to, and the authenticator signs the server’s challenge together with that origin, so a signature produced on a lookalike domain is useless at the real

one. With a PIN or a fingerprint required to use the key, a passkey is a multi-factor authenticator in a single step. WebAuthn calls passkeys that are copied across a person's devices multi-device credentials, and NIST accepts these synced keys at AAL2 but not at AAL3, because a key that can be synced is by definition a key that can be exported.

Turning MFA on is the easy part. Deciding who must use it, which methods count where and what happens when someone loses a phone is account administration, part of the wider arrangement of access. This site's security and access management service describes that arrangement: accounts, passwords and permissions set up so that each person holds exactly the access their job needs, and leavers lose it on the day they leave.

Where it goes wrong

A phishing page relays the code. A one-time code is a number the user reads and types, and nothing in it records which site the user was looking at. A lookalike page that forwards the password and the code to the real site within the time step signs in as the user. NIST therefore states that no authenticator whose output is entered by hand counts as phishing-resistant, and adds that how the user reached the fake page is irrelevant. The fix is an authenticator that binds its answer to the verifier: by its domain name, as WebAuthn does, or to the TLS channel itself, as client certificates and smart cards do. Require one of these for administrators and offer it to everyone else.

The phone number is the factor. A code sent by text message goes to whoever the mobile operator currently routes that number to. NIST's own list of threats includes an attacker who has persuaded the operator to redirect the victim's number and then receives the code. Before sending a code by text message or voice call, verifiers should look for signs such as a recent SIM change, number porting or a device swap. Keep text messages as a fallback rather than the main method, and require a full sign-in before the number on file can change.

Approve, approve, approve. A push prompt with a single Approve button asks for a decision, not for evidence. An attacker who already holds the password can trigger prompts until the user, tired of the noise, accepts one. NIST calls this authentication fatigue and no longer considers approve-only flows acceptable: a secret must travel from one channel to the other, for example a number shown

on the sign-in screen that the user types into the phone. Picking it from a short list is not enough, since the user may simply guess. Limit how many prompts can be sent since the last successful sign-in, as the guideline also recommends.

Recovery skips the second factor. A “lost my phone” link that asks only for the password, or sends a reset link by email, turns MFA into an option the attacker can decline. For accounts at AAL2, NIST requires recovery to use two recovery codes obtained by different methods, one recovery code plus a single-factor authenticator still bound to the account, or repeating the identity proofing. Every recovery must send the user a notification. More generally, one factor must never be usable to obtain an authenticator of another kind: a password alone must not be enough to print a fresh set of backup codes.

The code is accepted twice, or for too long. A six-digit code has a million possible values, so its strength comes from the rules around it. A verifier that accepts several time steps either side of now, or accepts a code that has already been used, gives an attacker who has seen one code a window in which to use it again. Record the last time step accepted for each user and reject anything at or before it, keep the window to the single step that RFC 6238 allows for delay, and cap consecutive failures: NIST allows no more than 100 before the authenticator is disabled.

The seeds sit beside the passwords. Codes from an app are symmetric: the server holds every user’s key, and anyone who copies those keys can generate valid codes for every account. NIST requires the keys to be strongly protected, with access limited to the software components that need them. Do not keep them as plain columns next to the password hashes. Encrypt them with a key kept in a separate secret store, and let only the service that verifies codes read them.

Further reading

- NIST SP 800-63B-4, Definitions and Acronyms — the definitions of multi-factor authentication, multi-factor authenticator and phishing resistance in the current guideline.
- NIST SP 800-63B-4 at the Computer Security Resource Center — the publication record: published in July 2025, superseding SP 800-63B.
- RFC 6238: TOTP — the time-based one-time password algorithm, its test vectors and the rules for clock drift.

- RFC 4226: HOTP — the counter-based algorithm underneath it, including the truncation to six or more digits.
- Web Authentication, Level 3 — the W3C specification behind passkeys, with origin-scoped credentials and synced passkeys.

Glossary

Choose for your actual constraints

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is multi-factor authentication?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/multi-factor-authentication/