Engineer

Password

∗ a secret string that proves who is signing in

What it is

A password is a string of characters that a person keeps in memory, or in a password manager, and presents to show that an account is theirs. The definitions in NIST's glossary agree on the core: letters, numbers and other symbols "used to authenticate an identity or to verify access authorization", and in some documents also to derive cryptographic keys. The current digital identity guideline, SP 800-63-4, calls it the "something you know" factor and notes that its first release used the name memorized secret. A passphrase, made of a sequence of words, and a PIN, made of decimal digits only, are both passwords in the wider sense.

The same string can do two jobs that look alike. A password used to sign in is sent to a verifier, which keeps enough to recognise it and should never keep the password itself. A password used to unlock something local stays on the device. SP 800-63B calls that kind an activation secret, with the unlock PIN of a multi-factor authenticator as the usual example: it is used on the device to reach the secret the authenticator holds.

The difficulty is that a password is a shared secret. The user knows it, the service has to recognise it, and anyone who learns it can use it exactly as the user would. Nothing in the string ties it to the site that asked for it, which is why SP 800-63B states plainly that passwords are not phishing-resistant. Most of the engineering around passwords is about limiting what that shared knowledge costs when it leaks.

Use the concept in a decision

Put "What is a password?" 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 is often the only thing guarding an account. Two kinds of guessing matter, and different things hold each of them back.

Online guessing means trying passwords against the live sign-in form, and there the service sets the pace. SP 800-63B caps consecutive failed attempts with one authenticator on one account at 100, and requires new passwords to be checked against a blocklist of common, expected and breached values, so the choices an attacker would try first are never accepted.

Offline guessing begins once a copy of the stored password data has left the building. There is no form and no limit. The guideline notes that attackers can compute many billions of hashes per second offline, and that passwords meant to survive this have to be “orders of magnitude more complex” than ones that only face a sign-in form. So storage decides the outcome. SP 800-63B requires passwords to be salted and hashed with a password hashing scheme, which takes the password, a salt and a cost factor and exists to make every guess expensive. The salt must be at least 32 bits and chosen so that salt values rarely collide, which keeps two accounts with the same password from sharing a stored hash. The cost factor should be as high as the verifier can afford and raised over time. The guideline also recommends recording which scheme and cost produced each hash, so the system can move to a stronger one later.

RFC 9106 describes Argon2, a memory-hard function built for this job, and recommends its Argon2id variant: one pass over 2 GiB of memory as the default for all environments, or three passes over 64 MiB where memory is scarce. The RFC says the first setting maximises an attacker’s costs on dedicated brute-force hardware, which is the property that counts once a hash file has leaked. The OWASP Password Storage Cheat Sheet gives lower working minimums and fallbacks for systems that cannot use Argon2id, and states the rule under all of them. Fast hashes such as SHA-256 are not suitable, because they let an attacker make large numbers of guesses quickly, and passwords are hashed rather than encrypted, because encryption runs both ways and the original can be recovered from it.

What it looks like in practice

Before a company-wide password management system went in, credentials were stored and shared in whatever way each person preferred. Day to day, that rarely looks like a dramatic breach and more often like a password in a chat message, a shared login nobody rotates and a slow pile of small exposures. The system gave the organisation one consistent way to store and share credentials, and it was not optional per person, because a password manager only half the team uses barely helps when the risk lives in the half that did not. Once the credentials lived in one managed place, access control became simpler and more consistent. SP 800-63B points the same way from the service side: verifiers must allow password managers and autofill, should allow paste, and the guideline names a password manager as the way to keep a distinct password for every service.

When a second server was provisioned from code, the first machine's history supplied the argument. It had been created by hand and its firewall added afterwards, which leaves a window in which a fresh host with a default password policy is reachable from the internet. The fix is ordering: the new machine is born behind its firewall, because the firewall play runs before the server play.

On a sign-in against a major mail provider, the application told the user their password was incorrect. It was not. That provider requires an application-specific password for third-party clients, so one account had two kinds of password, and a message that knew about only one sent the user to retype something that could never work. The rewritten message names the provider, says it requires an app-specific password and says where to create one, and a test asserts the provider's name and the credential type.

Where it goes wrong

One password, many accounts. When the same password protects several services, a breach at the weakest opens the rest. SP 800-63B calls this password stuffing: a password taken from one site is used against the same person's accounts on other sites. Users defend with a distinct password per service, which in practice means a password manager, and services by checking new passwords against values from earlier breaches.

Rules that make passwords weaker. Composition demands feel strict and produce predictable results. The guideline's own example is a user who would have picked "password" choosing "Password1" when asked for a capital letter

and a digit, and “Password1!” once a symbol is required. It rules out composition rules, scheduled changes, hints readable before sign-in and security questions such as the name of a first pet, and it still requires a change as soon as there is evidence of compromise.

A fast hash, or one salt for every row. A password table hashed with a general-purpose function, or salted with the same value throughout, hands an offline attacker cheap guesses that can be tested against every account at once. Systems that inherit old hashes can rehash each password with the new scheme at the user’s next sign-in, as the OWASP cheat sheet describes, and the stored scheme reference shows which rows are still on the old one.

The password is cut short, or cannot be pasted. SP 800-63B requires the whole submitted password to be verified, never a truncated part, and wants paste allowed so that password managers work where autofill does not. Some hashing functions bring their own limit: the OWASP cheat sheet notes that bcrypt takes at most 72 bytes of input in most implementations and tells services that use it to enforce that maximum.

The password lives where it was never meant to be. A password pasted into a chat, written into a ticket or printed in a log is a copy outside every control the real store has. Changing it is the only way to take that copy back, which is why one managed place matters more than any rule about what the password looks like.

Further reading

- Password, in the NIST CSRC glossary — definitions from FIPS 140-2, SP 800-63-4 and others, with passphrase as a special case.
- NIST SP 800-63B-4 — the current rules for choosing, checking and storing passwords, with the reasons in Appendix A.
- RFC 9106 — Argon2, with the two recommended Argon2id settings for password hashing.
- OWASP Password Storage Cheat Sheet — working parameters for Argon2id, scrypt, bcrypt and PBKDF2, and how to upgrade old hashes.



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 a password?” 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/password/