CS-022 · Passwords & Authentication

# Passwords

The oldest and most attacked authentication method, and how to make it stronger.

## Executive Summary

A password is a secret string a person uses to prove their identity to a system. It remains the most common and most attacked authentication method in the world. Modern guidance favors long, unique, memorable secrets over short strings loaded with forced complexity rules, and it treats the password as one layer rather than the only layer of defense.

## What It Is

A password is a shared secret between a user and a system that authenticates the user by confirming they know something only they should know. It is the most familiar form of the something you know authentication factor. In practice a password rarely travels or is stored in plain text. A well built system converts it into a fixed length value using a slow, salted cryptographic hash, then compares hashes rather than the raw secret. The strength of a password comes from how hard it is to guess or crack, which depends mainly on its length and unpredictability, and on how the system stores and rate limits attempts against it.

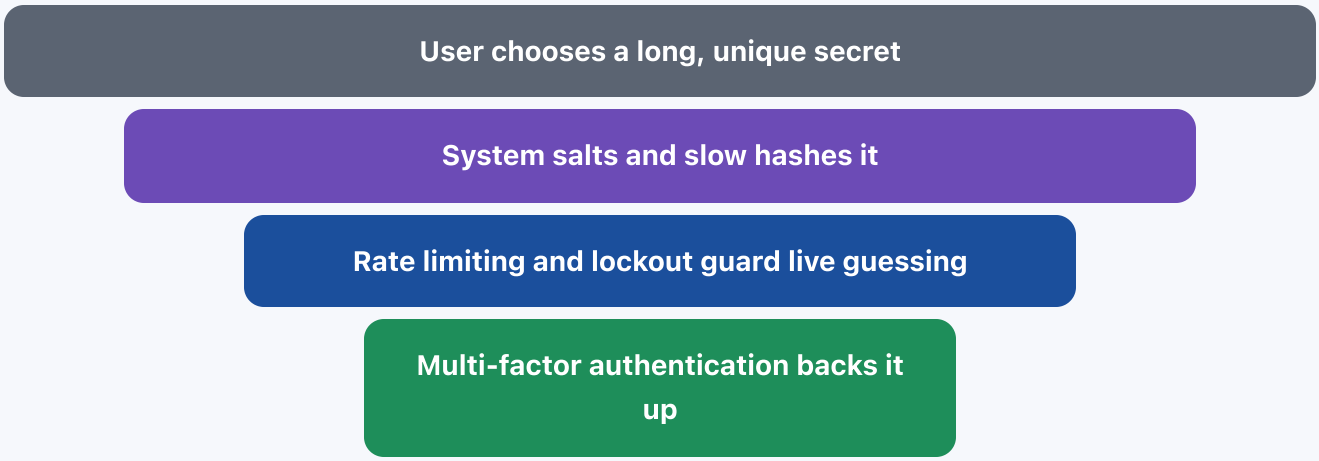
## Why It Matters

Stolen and reused passwords are behind a large share of real world breaches because a single working credential can hand an attacker the same access a legitimate employee has. Weak or shared passwords let opportunistic attackers walk through the front door without exploiting any software flaw. For an organization, one compromised admin password can expose customer data, trigger regulatory penalties, and stop operations. For a professional, understanding what actually makes passwords strong, and where they fall short, is foundational to identity and access work, security operations, and governance roles that must write and defend password policy.

## How It Works

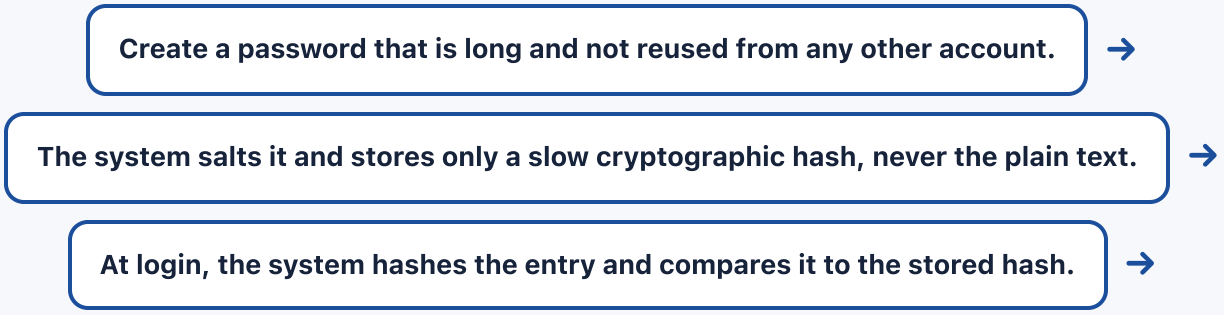
When you create a password, a secure system generates a random salt, combines it with your password, and runs the result through a purpose built password hashing function such as one designed to be slow and memory hard. It stores only the salt and the resulting hash, never the password itself. When you log in, the system repeats the process on what you typed and checks whether the new hash matches the stored one. Because the hash is one way, the stored value cannot be reversed back into the password, and the salt ensures two users with the same password do not share the same stored hash. Attackers who steal a hash database try to guess passwords offline, so slow hashing and long, unique passwords dramatically increase the time and cost of cracking.

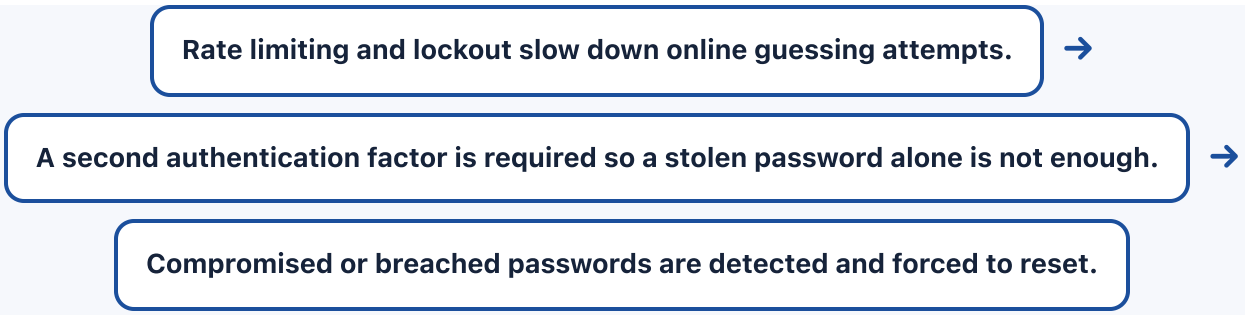
## Architecture Diagram




A password's real protection is layered: length and uniqueness in the secret, safe storage on the server, and a second factor around it.

## Visual Workflow






## Common Attacks

- Credential stuffing that replays passwords leaked from other breaches
- Brute force and dictionary guessing against weak or short passwords
- Phishing pages that trick users into typing their password directly to attackers
- Offline cracking of stolen password hash databases
- Shoulder surfing, keylogging, and reuse across personal and work accounts

## Common Mistakes

- Reusing the same password across many sites and services
- Choosing short passwords padded with predictable substitutions like p@ssw0rd
- Forcing frequent scheduled expiration, which pushes users toward weak patterns
- Storing passwords in plain text or with fast, unsalted hashing
- Blocking password managers by disabling paste in the login field

## Best Practices

- Favor length over forced complexity, and allow long passphrases
- Require every password to be unique to one account
- Screen new passwords against known breached and common password lists
- Store passwords with a salted, slow, modern password hashing function
- Layer multi-factor authentication on top of the password
- Only force a reset when there is evidence of compromise, not on a fixed calendar

## Quick Checklist

- [x] Minimum length set to at least the modern recommended floor
- [x] New passwords screened against breached and common lists
- [x] Salted, slow password hashing in use, never plain text
- [x] Rate limiting and account lockout configured against online guessing
- [x] MFA enabled on all accounts, especially admin and email
- [x] Copy and paste allowed so password managers work

## Recommended Tools

**Password manager**
Generates and stores long, unique passwords so users do not reuse them

**Breached password screening service**
Checks new passwords against known compromised credential lists

**Slow password hashing library**
Stores passwords using memory hard, salted, one way functions

**Authenticator app**
Adds a second factor so a stolen password is not enough

## Industry Standards

**NIST SP 800-63B**
Modern digital identity guidance favoring length, screening, and against forced periodic resets

**OWASP Authentication Cheat Sheet**
Practical guidance on password storage, policy, and login controls

**CIS Critical Security Controls**
Account and access management safeguards that include credential hygiene

## Career Relevance

Password design and policy touch nearly every security role. Identity and access management engineers configure hashing, screening, and lockout; SOC analysts investigate credential based intrusions; security engineers harden login flows; and GRC analysts write, audit, and defend password policy against frameworks and regulators. Even developers and IT staff need to know why length beats complexity and how safe storage works, which makes this a core skill for the audience AI-Governance-Jobs.com serves.

## Interview Questions

- Why does modern guidance favor password length over forced character complexity?
- What is a salt, and why does salting matter when storing passwords?
- Why is a slow hashing function preferred over a fast one for passwords?
- How does breached password screening improve on traditional complexity rules?
- Why did NIST move away from mandatory periodic password expiration?

## Related Certifications

**CompTIA Security+** | **ISC2 Certified in Cybersecurity (CC)** | **ISC2 SSCP**

## Further Reading

- NIST SP 800-63B Digital Identity Guidelines
- OWASP Authentication Cheat Sheet
- CISA: Use Strong Passwords

## Key Takeaways

- A password is a shared secret and still the most attacked authentication method.
- Length and uniqueness matter more than forced complexity symbols.
- Safe systems store salted, slow hashes, never the plain text password.
- Screen new passwords against breached lists instead of forcing calendar resets.
- Always back a password with multi-factor authentication.

## FAQ



## Related Careers

### RELATED ROLES

Iam Engineer | Soc Analyst | Grc Analyst

### RELATED CERTIFICATIONS

CompTIA Security+ | ISC2 Certified in Cybersecurity (CC) | ISC2 SSCP

### CURRENT OPENINGS

Live roles are on the job board.

Browse all jobs

### GUIDES & SALARY

Career guides | Role roadmaps | Salary data | Career tools

### SUGGESTED LEARNING PATH

1. Ground the basics with CS-001 Cybersecurity
2. Study this sheet: **Passwords**
3. Go deeper: Passphrases
4. Go deeper: Password Managers
5. Validate it: work toward **CompTIA Security+**
6. Find the role: browse current openings

## RELATED SHEETS

- CS-023 Passphrases
- CS-024 Password Managers
- CS-025 Multi-Factor Authentication (MFA)
- CS-027 Credential Stuffing
- CS-062 Authentication