A07:2025 — Authentication Failures

January 6, 2026

Document Summary

This document consolidates the provided content for *A07:2025 — Authentication Failures* into a structured, print-ready reference, including background context, scoring metrics, common authentication weakness conditions (credential stuffing, brute force, weak defaults, weak recovery, session/token management flaws, missing MFA), prevention guidance (MFA, password policy alignment with NIST guidance, breached-credential checks, rate limiting, secure session management, and trusted identity platforms), attack scenarios, references, and the mapped CWE list.

Contents

1 Background 2

2 Score Table 2

3 Description 2

3.1 Common Authentication Weakness Conditions . . . 2

4 How to Prevent 3

4.1 Credential and MFA Controls . . . 3

4.2 Anti-enumeration and Abuse Resistance . . . 3

4.3 Session and Token Management . . . 4

5 Example Attack Scenarios 4

5.1 Scenario #1: Credential Stuffing and Password Spray as a "Password Oracle" . . . 4

5.2 Scenario #2: Password-only Authentication and Counterproductive Password Policies 4

5.3 Scenario #3: Incorrect Session Timeout and Logout Semantics (Public or Shared Machines) . . . 4

6 References 5

7 List of Mapped CWEs 6

1 Background

Authentication Failures maintains its position at #7 with a slight name change to more accurately reflect the 36 CWEs in this category. Despite benefits from standardized frameworks, this category has kept its #7 rank from 2021.

Notable CWEs include:

- CWE-259: Use of Hard-coded Password
- CWE-297: Improper Validation of Certificate with Host Mismatch
- CWE-287: Improper Authentication
- CWE-384: Session Fixation
- CWE-798: Use of Hard-coded Credentials

2 Score Table

Metric	Value
CWEs Mapped	36
Max Incidence Rate	15.80%
Avg Incidence Rate	2.92%
Max Coverage	100.00%
Avg Coverage	37.14%
Avg Weighted Exploit	7.69
Avg Weighted Impact	4.44
Total Occurrences	1,120,673
Total CVEs	7,147

Table 1: Provided scoring summary for Authentication Failures.

3 Description

This vulnerability is present when an attacker can trick a system into recognizing an invalid or incorrect user as legitimate.

3.1 Common Authentication Weakness Conditions

There may be authentication weaknesses if the application:

- Permits automated attacks such as **credential stuffing**, where the attacker has a breached list of valid usernames and passwords. More recently this has expanded to **hybrid credential stuffing** (also called **password spraying**), where the attacker tries variations or increments of spilled credentials (e.g., `Password1!`, `Password2!`, `Password3!`).
- Permits **brute force** or other automated, scripted attacks that are not quickly blocked.

- Permits **default, weak, or well-known passwords**, such as `Password1` or an `admin` username with an `admin` password.
- Allows users to create new accounts with already known-breached credentials.
- Allows weak or ineffective credential recovery and forgot-password processes, such as **knowledge-based answers**, which cannot be made safe.
- Uses plain text, encrypted, or weakly hashed password data stores (see A04:2025 — Cryptographic Failures).
- Has missing or ineffective multi-factor authentication (MFA).
- Allows weak or ineffective fallbacks if MFA is not available.
- Exposes session identifiers in the URL, a hidden field, or another insecure client-accessible location.
- Reuses the same session identifier after successful login.
- Does not correctly invalidate user sessions or authentication tokens (mainly SSO tokens) during logout or after inactivity.
- Does not correctly assert the scope and intended audience of the provided credentials.

4 How to Prevent

4.1 Credential and MFA Controls

1. Where possible, implement and enforce use of **multi-factor authentication** to prevent credential stuffing, brute force, and stolen credential reuse.
2. Where possible, encourage and enable the use of **password managers** to help users make stronger choices.
3. Do not ship or deploy with **default credentials**, particularly for admin users.
4. Implement **weak password checks**, such as testing new or changed passwords against the top 10,000 worst passwords list.
5. During new account creation and password changes, validate against **known breached credential** lists (e.g., using `https://haveibeenpwned.com`).
6. Align password length, complexity, and rotation policies with NIST 800-63b guidelines in Section 5.1.1 for Memorized Secrets (or other modern, evidence-based password policies).
7. Do not force human beings to rotate passwords unless you suspect a breach. If you suspect breach, force password resets immediately.

4.2 Anti-enumeration and Abuse Resistance

8. Harden registration, credential recovery, and API pathways against **account enumeration** by using the same messages for all outcomes (e.g., “Invalid username or password.”).
9. Limit or increasingly delay failed login attempts, but avoid creating a denial-of-service scenario. Log all failures and alert administrators when credential stuffing, brute force, or other automated attacks are detected or suspected.

4.3 Session and Token Management

10. Use a server-side, secure, built-in session manager that generates a new random session ID with high entropy after login.
11. Ensure session identifiers are not in the URL, are securely stored in a secure cookie, and are invalidated after logout, idle timeouts, and absolute timeouts.
12. Prefer a premade, well-trusted system for authentication, identity, and session management. Transfer this risk where possible by buying and using a hardened and well-tested system.
13. Verify the intended use of provided credentials (e.g., for JWTs validate `aud`, `iss` claims and scopes).

5 Example Attack Scenarios

5.1 Scenario #1: Credential Stuffing and Password Spray as a "Password Oracle"

Credential stuffing (the use of lists of known username and password combinations) is now a very common attack. More recently, attackers "increment" or otherwise adjust passwords based on common human behavior (e.g., changing `Winter2025` to `Winter2026`, or `ILoveMyDog6` to `ILoveMyDog7`).

This adjusting of password attempts is called a **hybrid credential stuffing attack** or **password spray attack**, and it can be even more effective than the traditional version. If an application does not implement defenses against automated threats (brute force, scripts, bots) or credential stuffing, the application can be used as a password oracle to determine whether credentials are valid and to gain unauthorized access.

5.2 Scenario #2: Password-only Authentication and Counterproductive Password Policies

Most successful authentication attacks occur due to continued use of passwords as the sole authentication factor. Password rotation and complexity rules (once considered best practices) can encourage password reuse and weak password choices. Organizations are recommended to stop these practices per NIST 800-63 and to enforce multi-factor authentication on important systems.

5.3 Scenario #3: Incorrect Session Timeout and Logout Semantics (Public or Shared Machines)

Session timeouts are not implemented correctly. A user accesses an application on a public computer and closes the browser tab instead of selecting *Logout*. Another example is when a Single Sign-On (SSO) session cannot be closed via Single Logout (SLO): a single login authenticates the user to multiple systems (mail, documents, chat), but logging out only signs out of the current system.

If an attacker uses the same browser after the victim believes they logged out, but the user remains authenticated to one or more applications, the attacker can access the victim's accounts. Similar issues arise in offices and enterprises when a sensitive application has not been properly exited and a colleague has temporary access to an unlocked computer.

6 References

- OWASP Authentication Cheat Sheet
- OWASP Secure Coding Practices

7 List of Mapped CWEs

CWE	Title
CWE-258	Empty Password in Configuration File
CWE-259	Use of Hard-coded Password
CWE-287	Improper Authentication
CWE-288	Authentication Bypass Using an Alternate Path or Channel
CWE-289	Authentication Bypass by Alternate Name
CWE-290	Authentication Bypass by Spoofing
CWE-291	Reliance on IP Address for Authentication
CWE-293	Using Referer Field for Authentication
CWE-294	Authentication Bypass by Capture-replay
CWE-295	Improper Certificate Validation
CWE-297	Improper Validation of Certificate with Host Mismatch
CWE-298	Improper Validation of Certificate with Host Mismatch
CWE-299	Improper Validation of Certificate with Host Mismatch
CWE-300	Channel Accessible by Non-Endpoint
CWE-302	Authentication Bypass by Assumed-Immutable Data
CWE-303	Incorrect Implementation of Authentication Algorithm
CWE-304	Missing Critical Step in Authentication
CWE-305	Authentication Bypass by Primary Weakness
CWE-306	Missing Authentication for Critical Function
CWE-307	Improper Restriction of Excessive Authentication Attempts
CWE-308	Use of Single-factor Authentication
CWE-309	Use of Password System for Primary Authentication
CWE-346	Origin Validation Error
CWE-350	Reliance on Reverse DNS Resolution for a Security-Critical Action
CWE-384	Session Fixation
CWE-521	Weak Password Requirements
CWE-613	Insufficient Session Expiration
CWE-620	Unverified Password Change
CWE-640	Weak Password Recovery Mechanism for Forgotten Password
CWE-798	Use of Hard-coded Credentials
CWE-940	Improper Verification of Source of a Communication Channel
CWE-941	Incorrectly Specified Destination in a Communication Channel
CWE-1390	Weak Authentication
CWE-1391	Use of Weak Credentials
CWE-1392	Use of Default Credentials
CWE-1393	Use of Default Password