# CWE Detail – CWE-522

## Description

The product transmits or stores authentication credentials, but it uses an insecure method that is susceptible to unauthorized interception and/or retrieval.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 1.8

Priority: P4 - Informational (Low)

## Observed Examples (CVEs)

**•** CVE-2022-30018: A messaging platform serializes all elements of User/Group objects, making private information available to adversaries

**•** CVE-2022-29959: Initialization file contains credentials that can be decoded using a "simple string transformation"

**•** CVE-2022-35411: Python-based RPC framework enables pickle functionality by default, allowing clients to unpickle untrusted data.

**•** CVE-2022-29519: Programmable Logic Controller (PLC) sends sensitive information in plaintext, including passwords and session tokens.

**•** CVE-2022-30312: Building Controller uses a protocol that transmits authentication credentials in plaintext.

**•** CVE-2022-31204: Programmable Logic Controller (PLC) sends password in plaintext.

**•** CVE-2022-30275: Remote Terminal Unit (RTU) uses a driver that relies on a password stored in plaintext.

**•** CVE-2007-0681: Web app allows remote attackers to change the passwords of arbitrary users without providing the original password, and possibly perform other unauthorized actions.

**•** CVE-2000-0944: Web application password change utility doesn't check the original password.

**•** CVE-2005-3435: product authentication succeeds if user-provided MD5 hash matches the hash in its database; this can be subjected to replay attacks.

**•** CVE-2005-0408: chain: product generates predictable MD5 hashes using a constant value combined with username, allowing authentication bypass.

## Related Attack Patterns (CAPEC)

* CAPEC-102
* CAPEC-474
* CAPEC-50
* CAPEC-509
* CAPEC-551
* CAPEC-555
* CAPEC-560
* CAPEC-561
* CAPEC-600
* CAPEC-644
* CAPEC-645
* CAPEC-652
* CAPEC-653

## Attack TTPs

**•** T1543: Create or Modify System Process (Tactics: persistence, privilege-escalation)

**•** T1133: External Remote Services (Tactics: persistence, initial-access)

**•** T1558: Steal or Forge Kerberos Tickets (Tactics: credential-access)

**•** T1021.002: SMB/Windows Admin Shares (Tactics: lateral-movement)

**•** T1021: Remote Services (Tactics: lateral-movement)

**•** T1552.004: Private Keys (Tactics: credential-access)

**•** T1550.003: Pass the Ticket (Tactics: defense-evasion, lateral-movement)

**•** T1078: Valid Accounts (Tactics: defense-evasion, persistence, privilege-escalation, initial-access)

**•** T1110.004: Credential Stuffing (Tactics: credential-access)

**•** T1114.002: Remote Email Collection (Tactics: collection)

**•** T1550.002: Pass the Hash (Tactics: defense-evasion, lateral-movement)

**•** T1558.003: Kerberoasting (Tactics: credential-access)

## Modes of Introduction

**•** Architecture and Design: COMMISSION: This weakness refers to an incorrect design related to an architectural security tactic.

**•** Implementation: N/A

## Common Consequences

**•** Impact: Gain Privileges or Assume Identity — Notes: An attacker could gain access to user accounts and access sensitive data used by the user accounts.

## Potential Mitigations

**•** Architecture and Design: Use an appropriate security mechanism to protect the credentials. (Effectiveness: N/A)

**•** Architecture and Design: Make appropriate use of cryptography to protect the credentials. (Effectiveness: N/A)

**•** Implementation: Use industry standards to protect the credentials (e.g. LDAP, keystore, etc.). (Effectiveness: N/A)

## Applicable Platforms

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)

## Demonstrative Examples

**•** While the code confirms that the requesting user typed the same new password twice, it does not confirm that the user requesting the password change is the same user whose password will be changed. An attacker can request a change of another user's password and gain control of the victim's account.

**•** This code will run successfully, but anyone who has access to config.properties can read the value of password. If a devious employee has access to this information, they can use it to break into the system.

**•** This code will run successfully, but anyone who has access to the registry key used to store the password can read the value of password. If a devious employee has access to this information, they can use it to break into the system

**•** Because a compression algorithm is used instead of a one way hashing algorithm, an attacker can recover compressed passwords stored in the database.

**•** This Java example shows a properties file with a cleartext username / password pair.

**•** Multiple vendors used cleartext transmission or storage of passwords in their OT products.