# CWE Detail – CWE-312

## Description

The product stores sensitive information in cleartext within a resource that might be accessible to another control sphere.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 3.25

Priority: P2 - Serious (High)

## Observed Examples (CVEs)

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

**•** CVE-2009-2272: password and username stored in cleartext in a cookie

**•** CVE-2009-1466: password stored in cleartext in a file with insecure permissions

**•** CVE-2009-0152: chat program disables SSL in some circumstances even when the user says to use SSL.

**•** CVE-2009-1603: Chain: product uses an incorrect public exponent when generating an RSA key, which effectively disables the encryption

**•** CVE-2009-0964: storage of unencrypted passwords in a database

**•** CVE-2008-6157: storage of unencrypted passwords in a database

**•** CVE-2008-6828: product stores a password in cleartext in memory

**•** CVE-2008-1567: storage of a secret key in cleartext in a temporary file

**•** CVE-2008-0174: SCADA product uses HTTP Basic Authentication, which is not encrypted

**•** CVE-2007-5778: login credentials stored unencrypted in a registry key

**•** CVE-2001-1481: Plaintext credentials in world-readable file.

**•** CVE-2005-1828: Password in cleartext in config file.

**•** CVE-2005-2209: Password in cleartext in config file.

**•** CVE-2002-1696: Decrypted copy of a message written to disk given a combination of options and when user replies to an encrypted message.

**•** CVE-2004-2397: Plaintext storage of private key and passphrase in log file when user imports the key.

**•** CVE-2002-1800: Admin password in plaintext in a cookie.

**•** CVE-2001-1537: Default configuration has cleartext usernames/passwords in cookie.

**•** CVE-2001-1536: Usernames/passwords in cleartext in cookies.

**•** CVE-2005-2160: Authentication information stored in cleartext in a cookie.

## Related Attack Patterns (CAPEC)

* CAPEC-37

## Attack TTPs

**•** T1005: Data from Local System (Tactics: collection)

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

## Modes of Introduction

**•** Architecture and Design: OMISSION: This weakness is caused by missing a security tactic during the architecture and design phase.

## Common Consequences

**•** Impact: Read Application Data — Notes: An attacker with access to the system could read sensitive information stored in cleartext (i.e., unencrypted). Even if the information is encoded in a way that is not human-readable, certain techniques could determine which encoding is being used, then decode the information.

## Potential Mitigations

**•** Implementation: When storing data in the cloud (e.g., S3 buckets, Azure blobs, Google Cloud Storage, etc.), use the provider's controls to encrypt the data at rest. [REF-1297] [REF-1299] [REF-1301] (Effectiveness: N/A)

**•** Implementation: In some systems/environments such as cloud, the use of "double encryption" (at both the software and hardware layer) might be required, and the developer might be solely responsible for both layers, instead of shared responsibility with the administrator of the broader system/environment. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** Because the account ID is in plaintext, the user's account information is exposed if their computer is compromised by an attacker.

**•** The code stores the user's username and password in plaintext in a cookie on the user's machine. This exposes the user's login information if their computer is compromised by an attacker. Even if the user's machine is not compromised, this weakness combined with cross-site scripting (CWE-79) could allow an attacker to remotely copy the cookie.

**•** While successful, the program does not encrypt the data before writing it to a buffer, possibly exposing it to unauthorized actors.

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

**•** At least one OT product stored a password in plaintext.

**•** While it was not publicly disclosed how the data was protected after discovery, multiple options could have been considered.

**•** The result (edited and formatted for readability) might be:

## Notes

**•** Terminology: Different people use "cleartext" and "plaintext" to mean the same thing: the lack of encryption. However, within cryptography, these have more precise meanings. Plaintext is the information just before it is fed into a cryptographic algorithm, including already-encrypted text. Cleartext is any information that is unencrypted, although it might be in an encoded form that is not easily human-readable (such as base64 encoding).

**•** Other: When organizations adopt cloud services, it can be easier for attackers to access the data from anywhere on the Internet.