# CWE Detail – CWE-201

## Description

The code transmits data to another actor, but a portion of the data includes sensitive information that should not be accessible to that actor.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 3.0

Priority: P2 - Serious (High)

## Observed Examples (CVEs)

**•** CVE-2022-0708: Collaboration platform does not clear team emails in a response, allowing leak of email addresses

## Related Attack Patterns (CAPEC)

* CAPEC-12
* CAPEC-217
* CAPEC-612
* CAPEC-613
* CAPEC-618
* CAPEC-619
* CAPEC-621
* CAPEC-622
* CAPEC-623

## Modes of Introduction

**•** Architecture and Design: N/A

**•** Implementation: REALIZATION: This weakness is caused during implementation of an architectural security tactic.

## Common Consequences

**•** Impact: Read Files or Directories, Read Memory, Read Application Data — Notes: Sensitive data may be exposed to attackers.

## Potential Mitigations

**•** Requirements: Specify which data in the software should be regarded as sensitive. Consider which types of users should have access to which types of data. (Effectiveness: N/A)

**•** Implementation: Ensure that any possibly sensitive data specified in the requirements is verified with designers to ensure that it is either a calculated risk or mitigated elsewhere. Any information that is not necessary to the functionality should be removed in order to lower both the overhead and the possibility of security sensitive data being sent. (Effectiveness: N/A)

**•** System Configuration: Setup default error messages so that unexpected errors do not disclose sensitive information. (Effectiveness: N/A)

**•** Architecture and Design: Compartmentalize the system to have "safe" areas where trust boundaries can be unambiguously drawn. Do not allow sensitive data to go outside of the trust boundary and always be careful when interfacing with a compartment outside of the safe area. Ensure that appropriate compartmentalization is built into the system design, and the compartmentalization allows for and reinforces privilege separation functionality. Architects and designers should rely on the principle of least privilege to decide the appropriate time to use privileges and the time to drop privileges. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** The error clearly exposes the database credentials.

## Notes

**•** Other: Sensitive information could include data that is sensitive in and of itself (such as credentials or private messages), or otherwise useful in the further exploitation of the system (such as internal file system structure).