# CWE Detail – CWE-378

## Description

Opening temporary files without appropriate measures or controls can leave the file, its contents and any function that it impacts vulnerable to attack.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2022-24823: A network application framework uses the Java function createTempFile(), which will create a file that is readable by other local users of the system

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Read Application Data — Notes: If the temporary file can be read by the attacker, sensitive information may be in that file which could be revealed.

**•** Impact: Other — Notes: If that file can be written to by the attacker, the file might be moved into a place to which the attacker does not have access. This will allow the attacker to gain selective resource access-control privileges.

**•** Impact: Other — Notes: Depending on the data stored in the temporary file, there is the potential for an attacker to gain an additional input vector which is trusted as non-malicious. It may be possible to make arbitrary changes to data structures, user information, or even process ownership.

## Potential Mitigations

**•** Requirements: Many contemporary languages have functions which properly handle this condition. Older C temp file functions are especially susceptible. (Effectiveness: N/A)

**•** Implementation: Ensure that you use proper file permissions. This can be achieved by using a safe temp file function. Temporary files should be writable and readable only by the process that owns the file. (Effectiveness: N/A)

**•** Implementation: Randomize temporary file names. This can also be achieved by using a safe temp-file function. This will ensure that temporary files will not be created in predictable places. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** However, within this C/C++ code the method tmpfile() is used to create and open the temp file. The tmpfile() method works the same way as the fopen() method would with read/write permission, allowing attackers to read potentially sensitive information contained in the temp file or modify the contents of the file.