# CWE Detail – CWE-1250

## Description

The product has or supports multiple distributed components or sub-systems that are each required to keep their own local copy of shared data - such as state or cache - but the product does not ensure that all local copies remain consistent with each other.

## Extended Description

In highly distributed environments, or on systems with distinct physical components that operate independently, there is often a need for each component to store and update its own local copy of key data such as state or cache, so that all components have the same "view" of the overall system and operate in a coordinated fashion. For example, users of a social media service or a massively multiplayer online game might be using their own personal computers while also interacting with different physical hosts in a globally distributed service, but all participants must be able to have the same "view" of the world. Alternately, a processor's Memory Management Unit (MMU) might have "shadow" MMUs to distribute its workload, and all shadow MMUs are expected to have the same accessible ranges of memory. In such environments, it becomes critical for
 the product to ensure that this "shared state" is
 consistently modified across all distributed systems.
 If state is not consistently maintained across all
 systems, then critical transactions might take place
 out of order, or some users might not get the same
 data as other users. When this inconsistency affects
 correctness of operations, it can introduce
 vulnerabilities in mechanisms that depend on
 consistent state.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Applicable Platforms

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

## Demonstrative Examples

**•** Suppose the interconnect fabric does not prioritize such "update" packets over other general traffic packets. This introduces a race condition. If an attacker can flood the target with enough messages so that some of those attack packets reach the target before the new access ranges gets updated, then the attacker can leverage this scenario.

## Notes

**•** Research Gap: Issues related to state and cache - creation,
 preservation, and update - are a significant gap in
 CWE that is expected to be addressed in future
 versions. It likely has relationships to concurrency
 and synchronization, incorrect behavior order, and
 other areas that already have some coverage in CWE,
 although the focus has typically been on independent
 processes on the same operating system - not on
 independent systems that are all a part of a larger
 system-of-systems.