# CWE Detail – CWE-639

## Description

The system's authorization functionality does not prevent one user from gaining access to another user's data or record by modifying the key value identifying the data.

## Extended Description

Retrieval of a user record occurs in the system based on some key value that is under user control. The key would typically identify a user-related record stored in the system and would be used to lookup that record for presentation to the user. It is likely that an attacker would have to be an authenticated user in the system. However, the authorization process would not properly check the data access operation to ensure that the authenticated user performing the operation has sufficient entitlements to perform the requested data access, hence bypassing any other authorization checks present in the system. For example, attackers can look at places where user specific data is retrieved (e.g. search screens) and determine whether the key for the item being looked up is controllable externally. The key may be a hidden field in the HTML form field, might be passed as a URL parameter or as an unencrypted cookie variable, then in each of these cases it will be possible to tamper with the key value. One manifestation of this weakness is when a system uses sequential or otherwise easily-guessable session IDs that would allow one user to easily switch to another user's session and read/modify their data.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2021-36539: An educational application does not appropriately restrict file IDs to a particular user. The attacker can brute-force guess IDs, indicating IDOR.

## Modes of Introduction

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

## Common Consequences

**•** Impact: Bypass Protection Mechanism — Notes: Access control checks for specific user data or functionality can be bypassed.

**•** Impact: Gain Privileges or Assume Identity — Notes: Horizontal escalation of privilege is possible (one user can view/modify information of another user).

**•** Impact: Gain Privileges or Assume Identity — Notes: Vertical escalation of privilege is possible if the user-controlled key is actually a flag that indicates administrator status, allowing the attacker to gain administrative access.

## Potential Mitigations

**•** Architecture and Design: For each and every data access, ensure that the user has sufficient privilege to access the record that is being requested. (Effectiveness: N/A)

**•** Architecture and Design: Make sure that the key that is used in the lookup of a specific user's record is not controllable externally by the user or that any tampering can be detected. (Effectiveness: N/A)

**•** Architecture and Design: Use encryption in order to make it more difficult to guess other legitimate values of the key or associate a digital signature with the key so that the server can verify that there has been no tampering. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** The problem is that the developer has not considered all of the possible values of id. Although the interface generates a list of invoice identifiers that belong to the current user, an attacker can bypass this interface to request any desired invoice. Because the code in this example does not check to ensure that the user has permission to access the requested invoice, it will display any invoice, even if it does not belong to the current user.