

**Finding Name:** Incomplete Role Determination

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Team** | **Role** | **Project** | **Quality Assurance** | **Is this a re-tested Finding?** |
| Sangeeth Subburam | SCR | Senior Team Member | Ontrack |  | No |

|  |
| --- |
| **Was this Finding Successful?** |
| Yes |

**Finding Description**

This vulnerability happens when an application's logic for defining a user's position is insufficient and does not consider all conceivable circumstances. In this example, the Task model's **role\_for** function does not address circumstances in which the user is not assigned to the project or a member of the related group. This can cause the method to return nil, resulting in unexpected behaviour.

**Risk Rating**  
Impact: Significant   
Likelihood: Unlikely

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Impact values** | | | | |
| **Very Minor** | **Minor** | **Significant** | **Major** | **Severe** |
| Risk that holds little to no impact. Will not cause damage and regular activity can continue. | Risk that holds minor form of impact, but not significant enough to be of threat. Can cause some damage but not enough to impede regular activity. | Risk that holds enough impact to be somewhat of a threat. Will cause damage that can impede regular activity but will be able to run normally. | Risk that holds major impact to be of threat. Will cause damage that will impede regular activity and will not be able to run normally. | Risk that holds severe impact and is a threat. Will cause critical damage that can cease activity to be run. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Likelihood** | | | | |
| **Rare** | **Unlikely** | **Moderate** | **High** | **Certain** |
| Event may occur and/or if it did, it happens in specific circumstances. | Event could occur occasionally and/or could happen (at some point) | Event may occur and/or happens. | Event occurs at times and/or probably happens a lot. | Event is occurring now and/or happens frequently. |

**Business Impact**

This vulnerability may provide unauthorised access to sensitive functionality or data within the program. Depending on how access control logic uses the role\_for function, attackers might exploit this issue to obtain unwanted privileges or conduct operations designated for certain roles.

Unauthorised Access: If access control depends exclusively on the **role\_for** method and ignores the **nil** situation, users with undefined roles may get unintentional access to sensitive data or functionality.

Denial of Service: When encountering a **nil** role, the application may face difficulties or unexpected behaviour, perhaps resulting in a denial of service to legitimate users.

Inconsistent Behaviour: The application's behaviour may become unpredictable depending on how various portions of the code handle the **nil** role.

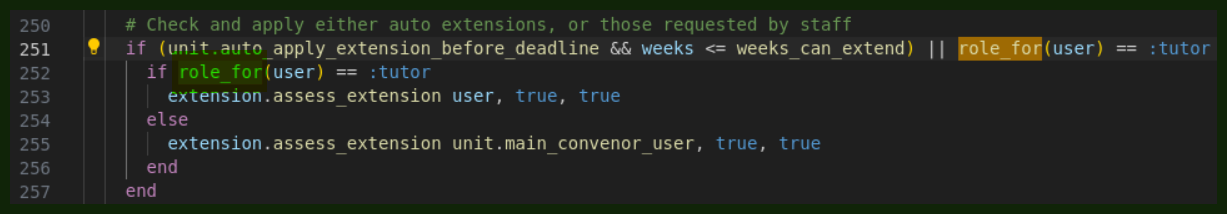
**Affected Assets**

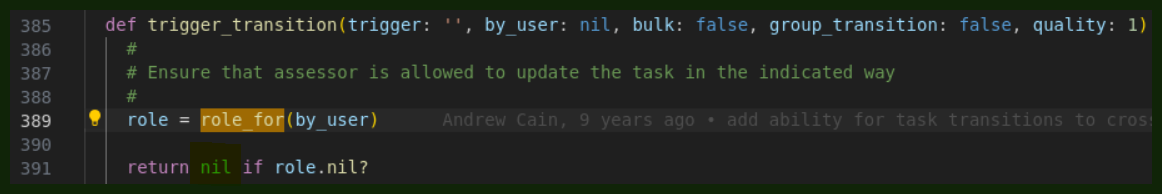
doubtfire-deploy < doubtfire-api < app < models < task.rb

**Evidence**

**A screen shot of a computer

Description automatically generated**

****

****

**Remediation Advice**

Complete Role Determination: Improve the **role\_for** technique to provide a more complete role determination approach. Consider adding user permissions, project settings, or default roles to assign when the user's role cannot be identified directly.

Fallback method: Create a fallback method to handle situations when the user's role cannot be established. Instead of returning **nil**, set the default role to a safe one, such as guest, to guarantee that access control logic always works with a stated role.

Error Handling: Implement error handling capabilities to deal with unforeseen circumstances appropriately. Log errors or raise exceptions to notify developers and administrators when role determination fails or produces unexpected outcomes.

Explicit handling of nil: Change the access control mechanism to explicitly handle the nil scenario returned by **role\_for.** which involves:

* Redirecting to the login page.
* Displaying the proper error message.
* Preventing access to critical resources.

**References**

M. Korneeva, “Reactive error-handling in Angular,” *Medium*, Nov. 27, 2020. https://medium.com/ngconf/reactive-error-handling-in-angular-2bde9dd223a0 (accessed Apr. 08, 2024).

ngconf

N. Mehlhorn, “How to navigate to previous page in Angular,” *DEV Community*, Oct. 21, 2020. https://dev.to/angular/how-to-navigate-to-previous-page-in-angular-16jm

Angular Role Determination ErrorHandling (OpenAI’s ChatGPT, private communication, 08 April 2024).

**Contact Details**

Sangeeth Subburam (s222434398@deakin.edu.au)

**Pentest Leader Feedback.**

The lead will provide feedback to enact on.