

**Finding Name:** Improper Authorisation Checks

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Team** | **Role** | **Project** | **Quality Assurance** | **Is this a re-tested Finding?** |
| Deakin Carr | SCR | Junior Team Member | Ontrack | Natalia Khobotova | Yes |
|  |  |  |  |  |  |

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

**Finding Description**

In the TutorialStreamsApi class of the tutorial\_streams\_api.rb file, there is a notable authorization flaw concerning how user permissions are checked. The methods for adding, updating, and deleting tutorial streams in the Ontrack application erroneously use the same permission check, :add\_tutorial, for all three operations. This flaw means that a user with permission only to add tutorial streams could potentially update or delete existing streams, bypassing the intended access controls.

**Risk Rating**  
Impact: **Major**  
 The vulnerability allows unauthorized users to update or delete tutorial streams, which could lead to significant disruptions in the educational process, affecting both the integrity of the course content and the trust in the application. This level of impact could severely impede regular activity and prevent normal operation of the Ontrack application.  
  
Likelihood: **Moderate**  
 Considering the application's use in a university setting, where multiple users with varying levels of permission access the system, the likelihood of someone exploiting this vulnerability is moderate. It is plausible that a user with add permissions might discover and exploit this flaw to perform unauthorized actions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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**

The identified vulnerability poses a significant risk to the integrity and confidentiality of the Ontrack application's data. Unauthorized modifications or deletions of tutorial streams could lead to inaccurate course material presentation, miscommunication between students and faculty, and loss of critical educational data. This could undermine the trust in the application's reliability and security, potentially affecting the university's reputation and causing administrative chaos.

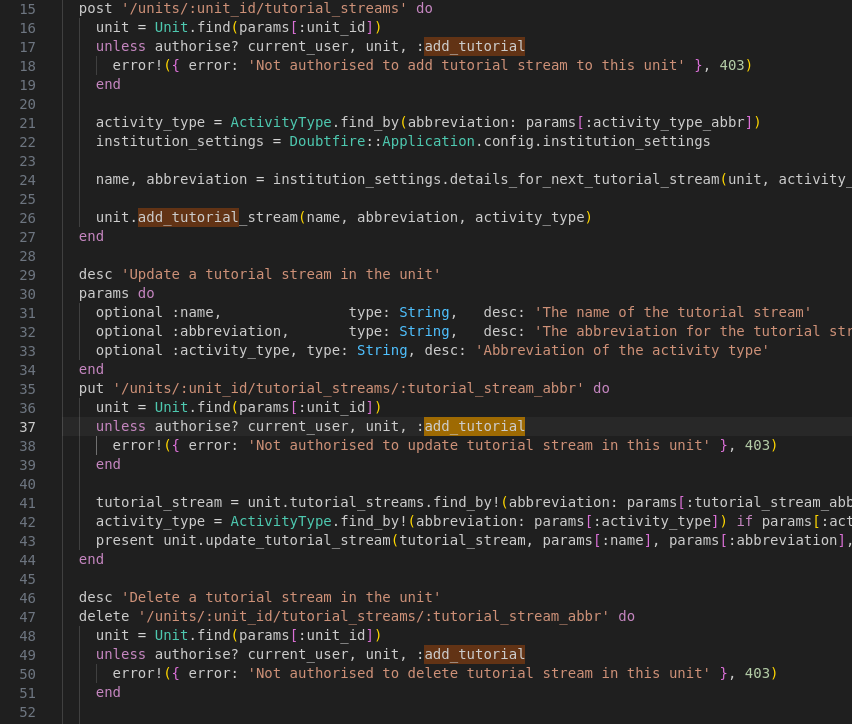
**Affected Assets**

The vulnerability directly affects the tutorial stream management functionality within the Ontrack application, specifically targeting the TutorialStreamsApi class in the ./doubtfire-api/app/api/tutorial\_streams\_api.rb script. This could indirectly impact all users of the application, including students, faculty, and administrative staff, who rely on accurate and authorized tutorial stream information.

**Evidence**

The vulnerability is evidenced in the source code of the tutorial\_streams\_api.rb file, where the before filter calls authorise? method with the :add\_tutorial symbol for all POST, PUT, and DELETE actions on tutorial streams, as shown in the following snippets:

* POST /units/:unit\_id/tutorial\_streams/:tutorial\_stream\_abbr:   
  unless authorise? current\_user, unit, :add\_tutorial
* PUT /units/:unit\_id/tutorial\_streams/:tutorial\_stream\_abbr:   
  unless authorise? current\_user, unit, :add\_tutorial
* DELETE /units/:unit\_id/tutorial\_streams/:tutorial\_stream\_abbr:   
  unless authorise? current\_user, unit, :add\_tutorial



**Remediation Advice**

To address this vulnerability, it is recommended to implement distinct permission checks for each action type. The application should define and enforce separate permissions for adding, updating, and deleting tutorial streams. The authorization checks should be updated as follows:

* For adding a tutorial stream, retain the :add\_tutorial permission check.
* For updating a tutorial stream, introduce and check for a :update\_tutorial permission.
* For deleting a tutorial stream, introduce and check for a :delete\_tutorial permission.

Additionally, thorough testing should be conducted to ensure that the new permission checks are enforced correctly and that users cannot perform actions outside their authorized scope. Regular audits of the permission model and implementation in the application are also advisable to prevent similar vulnerabilities in the future.

**References**

ChatGPT, “ChatGPT”, OpenAI [Large language model] Available: [https://chat.openai.com](https://chat.openai.com/) [Accessed: 2/4/2024].

**Contact Details**

* Deakin Carr
* Carrde@deakin.edu.au

**Pentest Leader Feedback.**

Hi Deakin, another great work! Could you please update the references if you used any other sources?

- Hi, no other references used. ChatGPT was used to format this report, however the actual vulnerability was just detected through me reading it. As this is simply a logic flaw, it has no reference to documentation.