

**Finding Name:** Insecure File Permission Assignment

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

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

**Finding Description**

In the OverseerAssessment class of the Ontrack application, there is a method named grant\_access\_to\_submission which executes a system command to modify file permissions. Specifically, the method uses chmod o+w on the submission output directory, granting write permissions to all users. This practice can lead to unauthorized file modifications or access, as it allows any user on the system to write to this directory.

**Risk Rating**  
Impact: **Major**  
 This vulnerability exposes student submissions and assessment feedback to unauthorized access or modification. Since it affects the integrity and confidentiality of sensitive data, it can significantly impede regular activity and damage the trust in the university's digital systems.  
  
Likelihood: **Moderate**  
 Given that this vulnerability requires specific circumstances to be exploited (e.g., an attacker gaining access to the system), it's not certain but definitely possible, especially if attackers are aware of this flaw.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 business impact of this vulnerability is significant. By allowing write access to all users, sensitive data related to student assessments and feedback could be exposed, altered, or deleted. This not only compromises the confidentiality and integrity of the data but also risks the reputation of the educational institution using the Ontrack application. Furthermore, if an attacker exploits this vulnerability, it could lead to a breach of trust between the university and its students or potentially legal consequences if personal data is mishandled.

**Affected Assets**

The affected asset is the Ontrack application, specifically the OverseerAssessment class within the Ruby on Rails backend at ./doubtfire-api/app/modals/overseer\_assessment.rb. The output directory where student submissions are stored is the primary concern, as this is where the insecure permissions are applied.

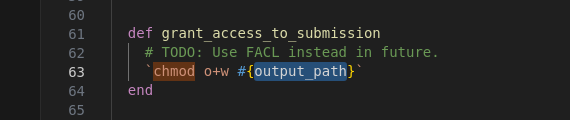
**Evidence**

The vulnerability is evidenced in the grant\_access\_to\_submission method of the OverseerAssessment class. The method's implementation uses a system call to change the directory permissions, making it writable by all users:

ruby

def grant\_access\_to\_submission  
 `chmod o+w #{output\_path}`  
end

This code snippet clearly demonstrates the insecure permission setting on the submission output directory.



**Remediation Advice**

To remediate this vulnerability, the application should implement more restrictive file permissions. Instead of granting write access to all users, permissions should be limited to only the application and specific users who need access. A more secure approach would be to use file access control lists (FACLs, as noted in a comment within the code) or similar mechanisms to finely control access permissions. Additionally, conducting a thorough audit of file permissions throughout the application to ensure that they follow the principle of least privilege can prevent similar vulnerabilities.

**References**

1. Brakeman, "A static analysis security vulnerability scanner for Ruby on Rails applications," GitHub repository, Accessed: Apr. 3, 2024. [Online]. Available: <https://github.com/presidentbeef/brakeman>
2. Brakeman, "Introduction to Brakeman," Brakeman Scanner, Accessed: Apr. 3, 2024. [Online]. Available: <https://brakemanscanner.org/docs/introduction/>
3. 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.**

The lead will provide feedback to enact on.