# SA Threat Documentation - Matthias Borg

## Trust Levels:

|  |  |  |
| --- | --- | --- |
| ID | Name | Description |
| **T1** | Authenticated Teacher | A user who is logged in as a teacher – can create assignment, view all assignments, view all submitted assignments, add comment, view comments |
| **T2** | Authenticated Student | A user who is logged in as a student – can submit an assignment, view all his assignments, view assignment details, add comment, view comments |
| **T3** | Anonymous User | A user who is not logged in at all – register as teacher, login as student or teacher |
| **T4** | Database Admin | A non-authenticated user who is on the backend of the website – responsible for taking care of the database |

## Assets:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Description | Trust Level |
| **A1** | Teacher | Assets that relate to a teacher |  |
| **A1.1** | Teacher login data | This includes only the email as teachers use google authentication to login | T1, T4 |
| **A2** | Student | Assets that relate to a student |  |
| **A2.1** | Student login data | This includes the email and password of the student. The password needs to be protected, because if it is stolen any user will be able to do anything that student can do | T2, T4 |
| **A3** | Assignment | Assets that relate to an assignment |  |
| **A3.1** | Assignment data | This includes name, description, and other information about the assignment | T1, T2, T4 |
| **A4** | StudentAssignment | Assets that relate to a student assignment |  |
| **A4.1** | StudentAssignment Data | This includes which assignment and student does it belong to, if it was uploaded, file path, signature, and other keys. The file here needs to be encrypted so it cannot be opened and copied by anybody else. Same applies to the keys | T1, T2, T4 |
| **A5** | Comment | Assets that relate to a comment |  |
| **A5.1** | Comment data | This includes which StudentAssignment does it belong to, who uploaded it and comment information | T1, T2, T4 |
| **A6** | Backend database | Assets that relate to the database |  |
| **A6.1** | Backend data | This includes all the data related to the website, all accounts, assignments, StudentAssignments and comments | T4 |

## Threats:

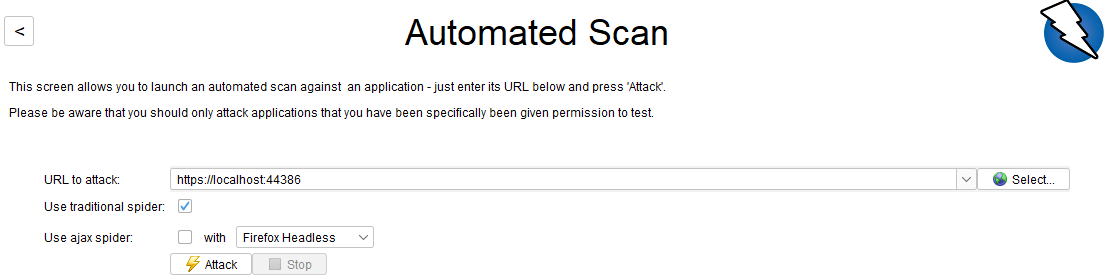
|  |
| --- |
| Threat 1 |
| **ID** | TR1 |
| **Name** | Tries to supply malicious data when logging in |
| **Description** | Adversary tries to input malicious data to try and login |
| **Stride** | Tampering, Elevation of privilege |
| **Entry Points** | External Login |
| **Assets** | A1.1 - Teacher login data, A4.1 – Backend data |
| **Mitigation Strategy** | Strong framework used to handle such threats, using a clean architecture. |

|  |
| --- |
| Threat 2 |
| **ID** | TR2 |
| **Name** | Tries to supply malicious data when adding a comment |
| **Description** | Adversary tries to input malicious data to try and add a comment |
| **Stride** | Tampering |
| **Entry Points** | Add Comment Page |
| **Assets** | A4.1 - Comment data, A4.1 – Backend data |
| **Mitigation Strategy** | Has a strong framework that can filter out such attacks and validation filters on inputs |

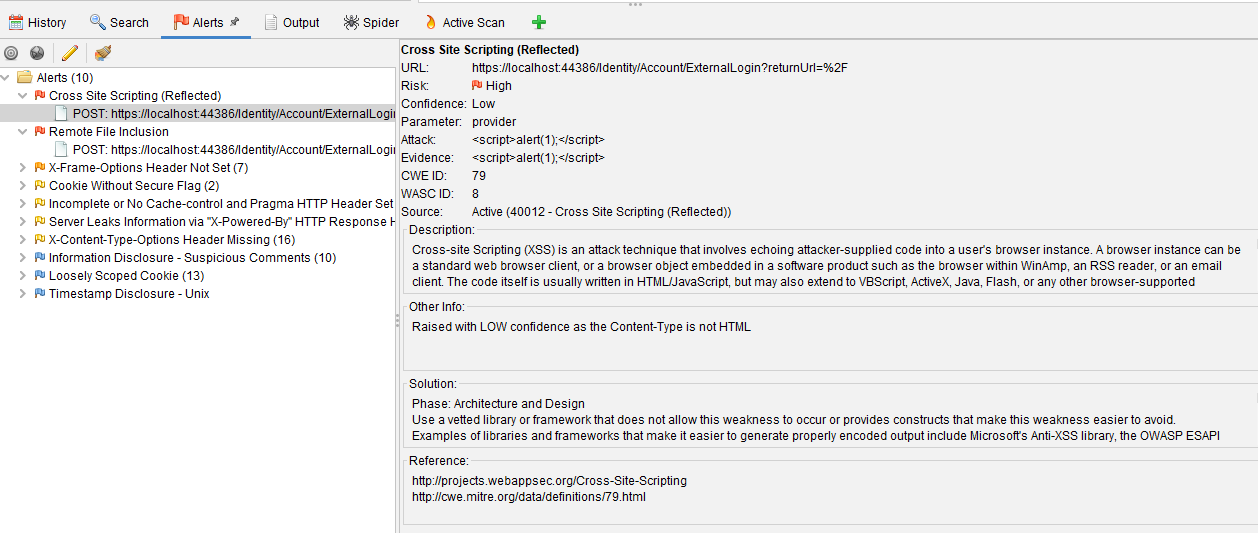
|  |
| --- |
| Threat 3 |
| **ID** | TR3 |
| **Name** | Tries to supply malicious data when registering student |
| **Description** | Adversary tries to input malicious file to try and register student |
| **Stride** | Tampering |
| **Entry Points** | Student registration |
| **Assets** | A5.1 - Student data, A4.1 – Backend data |
| **Mitigation Strategy** | Has a framework that can filter out attacks, and also have validation on inputs |

## Documentation:

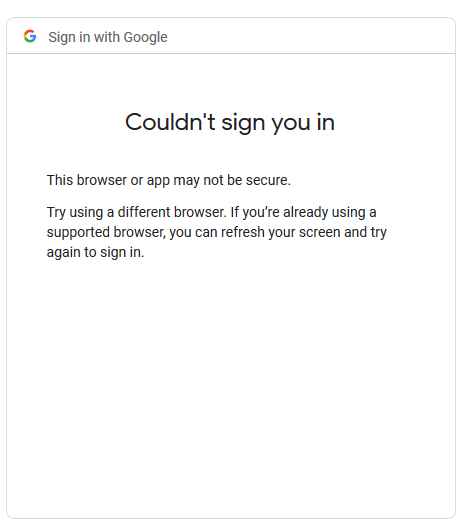
To attack my website the ZAP tool was used. Together with this tool I also attacked the website manually. When using ZAP, the automated scan was used to attack the site, as shown in the picture below.



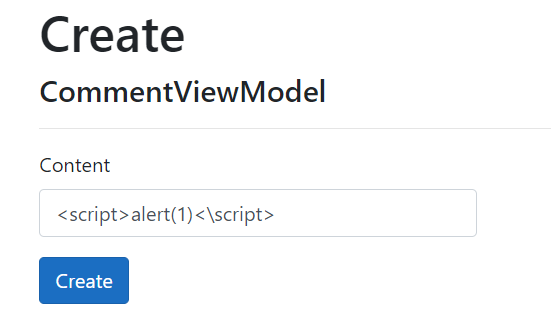
From this flag such as the external login error are flagged. As shown in the picture below.



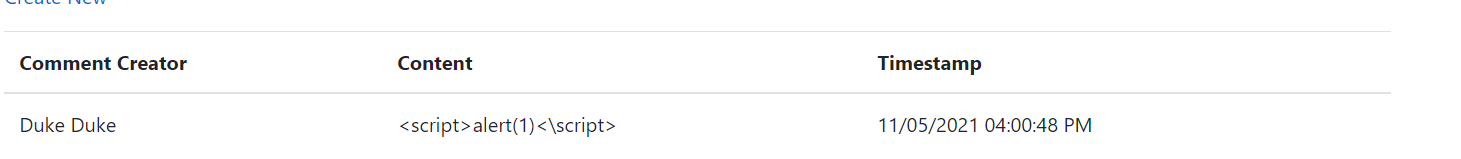
To test the teacher functionalities, I also tried using the ZAP Manual Explore feature, however when trying to log in with the google account I was encountering the problem, shown below.



Due to this problem I decided to attack the website manually, without using the ZAP tool. In the images below the process of this is shown.

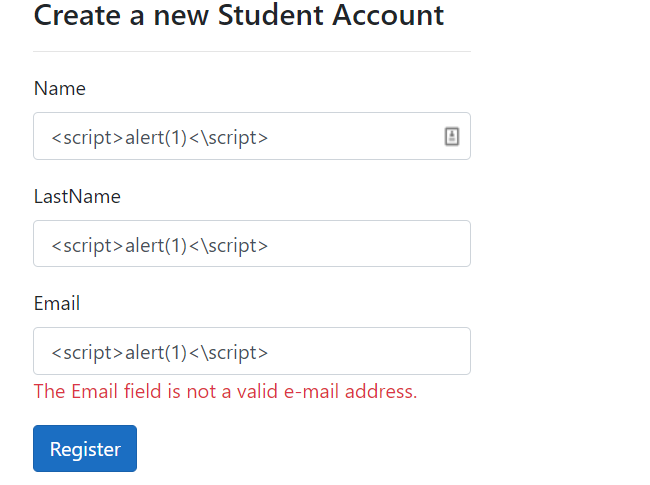


In the image above shows the form.



This image shows that it has been created however no alerts were shown when displaying it.

To add a student, I also did it manually as shown in the image below.



The image above shows the form, the email also has validation for it to be a proper email. In this case everything will still be created however none will be affecting the website. By using Linq it helps to prevent scripts to be entered in the database.