MCAST

Securing Application

Assignment Guidelines

Read the following instructions carefully before you start the assignment. If you do not understand any of them, ask your lecturer.

- The assignment coversheet should be the first sheet in your assignment. Moreover, the coversheet should be fully completed with all the necessary details.
- All text\code must be properly referenced. In the absence of proper referencing, the assignment will be regarded as plagiarised.
- Copying is strictly prohibited and will be penalized in line with the College's disciplinary procedures.
- When the deadline specified by your lecturer is due, you shall hand all the required deliverables as explained in class.
- You are also required to submit your assignment to the relevant plagiarism detection service by the same deadline. If necessary, your lecturer will forward you details in order to submit your assignment to this service.
- The lecturer may hold a post-submission interview. Attendance to such interview is mandatory. Moreover, marks assigned to the criteria may be affected by the interview performance.
- **All work that has been carried out, must be written down and included within the assignment as evidence. No marks will be awarded for work that is not presented.**
- The deadline for this assignment is Monday 9th May 2022.

Scenario

Scenario:

You are building a file sharing website that will allow users to register, upload files and share the files with authorised users. The website will be built with security in mind.

The website will have the following features and functionality:

- User registration, authentication and authorisation controls (see Sections 3 and 4);
- File upload, file sharing and downloads in a secure manner (see Sections 4, 5 and 6); and
- Logging (see Section 7).

You need to implement code and functionality that meets all the requirements set out in the Sections below.

Further, you need to write documentation, carry out tests and write reports as described in the Sections below.

MCAST

Section 1

Identify Threats and Possible Exploits (KU1.3, 5 marks)

Identify at least 2 assets and at least 2 threats to the website to be developed.

Use the templates in Appendix A and write appropriate documentation to:

identify at least 3 trust levels **(1 mark)**;

identify at least 2 assets **(2 marks)**; and

identify at least 2 threats related to the identified assets **(2 marks)**.

Note: the marks for the mitigation strategy are awarded in Section 2.

Note: you need to identify your own assets; and threats. Using the assets; and threats in the sample documentation do not count towards the limit.

Note: you may use relevant trust levels in the appendix if the trust levels are correct for the identified assets and threats.

Section 2

Mitigation of Threats (KU1.2, 5 marks)

Use DREAD to identify the priority of the different threats identified in Section 1 **(1 mark)**;

What is threat mitigation? **(1 mark)**; and

Describe ways to mitigate each of the identified threats **(3 marks)**.

Note: The DREAD rating is subjective and different results may be obtained for the same threat by different people. It is important that working showing how the result is obtained is shown. Use the DREAD template in the Appendix. If the template is not used, the marks will not be awarded.

MCAST

Section 3

Strong authentication and authorisation (SE2.4, 10 marks)

Configure identity options to force the password to contain: digits; uppercase characters; lowercase characters; special characters; and have a length of at least 8 **(1 mark)**.

Create website roles and default users during setup **(2 marks)**.

Use Authorize filters, where relevant to implement authentication and authorisation and prevent directory browsing **(1 mark)**.

Implement OAuth for your website to allow new users to register using a 3rd party OAuth service, such as Google Authentication, or otherwise **(3 marks)**.

Implement 2FA (two-factor authentication) **(3 marks)**.

Section 4

Mitigate XXE and XSS (AA2.1, 7 marks)

User input should be validated properly. Use data annotations in the model to validated user input and the ModelState.IsValid check. Show, using screenshots your use of:

- A DataType annotation to verify that the file expiry is a date; and
- Data Annotation that makes sure that the file expiry date is in the future (you may require to create a custom annotation for this) **(2 marks)**.

When a user submits a file, he should be able to select the users that are authorized to view the file. File access should be restricted only to the owner of the file and authorized users. Restrict all direct file access – file access should only be provided through relevant Actions. Access control should be checked using ActionFilters **(3 marks)**.

When a user submits a file, the user can optionally set an expiry date to the file. Only future expiry dates can be set. If an expiry date is set, the file should be marked as expired after the expiry date and it should no longer be accessible, and a custom error page will be displayed if a user attempts to access this file. You should have at least one expired file for the interview **(2 marks)**.

Section 5

Prevention of file injection (AA2.2, 7 marks)

Users can submit two types of files to the system:

- PDF files; and
- JPG files.

When a file is submitted to the system:

- The system must check that the extension is either .pdf or .jpg **(1 mark)**;
- Check that the file is of the type specified by the extension by inspecting the file prefix "magic bytes" and verifying that the file is of the correct type **(3 marks)**;
- Every user must have his own cryptographic key pair. This cryptographic key pair must be used to digitally sign the file, so that each time a file is downloaded, it is possible to verify that **(3 marks)**:
    - the file was submitted by the user claiming to have submitted the file; and
    - the file did not change since the submission.

Section 6

Hybrid encryption (SE3.3, 10 marks)

Files should be uploaded and downloaded through https **(1 mark)**.

Whenever a file is uploaded it needs to be stored encrypted at rest using hybrid encryption **(5 marks)**.

Whenever a file is requested for download, it needs to be decrypted **(4 marks)**.

MCAST

Section 7

Keep sufficient logging and prevent information leaks (AA2.3, 7 marks)

Keep File Access Logs and Error Logs should be kept in a database. Implement custom exceptions that do not disclose system information to the end user.

To do so, you must:

- Implement custom error pages in a proper way **(2 marks)**;
- Implement error logs and store the error logs in the database **(2 marks)**; and
- Keep File Access logs. File Access logs should be stored in the database and include information such as the: ip address; timestamp; user; and other information **(3 marks)**.

Section 8

Apply tools and scanners (AA4.2, 7 marks)

Make use of any tools which you are familiar with, such as the ZAP tool, to test and explore two of these types of threats for vulnerabilities in your web application **(3.5 marks for each threat type)**:

a. Broken Authentication
b. Sensitive Data Exposure
c. Broken Access Control
d. Injection

Add step-by-step screenshots describing how you tested for vulnerabilities, adding descriptions for each screenshot.

MCAST

Section 9

Create a report based on findings using a security tool (SE4.3, 10 marks)

Compile a detailed report (**worthy of 10 marks**) and explain in detail the testing you have carried out in Section 8. Report must clearly show:

- The conclusions and implication of your security testing results, explaining the reasoning behind your conclusion **(2 marks per threat type)**;
- Whether there could be additional vulnerabilities for each threat type that were not discovered in your testing. Describe further how you could test for these vulnerabilities and mitigations that you could apply **(3 marks per threat type)**.

Use screenshots, images, and examples as necessary to explain and justify the results. Your conclusions must be correct and based on sound reasoning.

Appendix

Trust Levels:

ID	Name	Description
T1	Remote anonymous user	A user who has not yet authenticated to the website
T2	Authenticated user	A registered user who has valid credentials
T3	Database server administrator	User who can do any operation on the underlying database
T4	Website administrator	A user who can configure the website by for example upload new website versions, deleting files from the web server etc
T5	Clerk	A user having this role is able to maintain products, their pri categories and sub categories.

Assets:

ID	Name	Description	Trust level
A1	User	Assets that relate to a website user	
A1.1	User's login data	User's credentials username and password. This asset needs protection because if it is stolen another user would be able to do anything which the user can do	T2 Authenticated user T3 Database server admin
A1.2	User's personal data	User's personal data including contact information. This needs protection because some personal data might be important such as telephone number	T2 Authenticated user T3 Database server admin

Threats:

Id	TR1
Name	Adversary tries to supply malicious data when logging in
Description	Adversary tries to input special characters to be able pose as another user, or logs in without having an appropriate username and password. Handling of data is critical in this regards.
Stride	Tampering, Elevation of privilege
Entry Points	(E1.1) Login page
Assets	(A1.2) User's personal data, (A2) Backend database
Mitigation Strategy	Using stored procedures or parameterized queries

DREAD:

Threat Id	TR1
Threat Name	Adversary tries to supply malicious data when logging in
Damage Potential	A successful attack can give the attacker administrative privileges. 5
Reproducibility	An injection attack is easy to reproduce 5
Exploitability	An injection attack is easy to exploit 5
Affected Users	All the users can be affected by the attack 5
Discoverability	Injection attacks are a common and well-known attack vector 5
Dread Rating	15