ENPM 809W – Introduction Secure Software Engineering Project Phase 3

(DRAFT)

Project Phase 3 Description:

Now that you have implemented your projects based on requirements and design, you will be performing a secure code review for another student's project. You will receive an email from the instructor/grader/TA on which project you will be required to perform the secure code review on.

Important Project Deliverables and Due Dates:

Deliverable	Due Date and Time
Project Phase 3	*November 30, 2021 by 11:59 PM*

Grading Criteria

Project Phase	Grading Criteria
Project Phase 3	30% of the project grade. Depends on: • The quality of findings and description from the code review. • The quality of proposed fixes based on the findings. • Timely submission.

Project Phase 3: Code Review Process

In Project Phase 3, you will be assigned another student's project (here onwards referred to as the student project) for which you will get read only access and you will perform a Secure Code Review. If a student requests access to your repository, you will be **required** to give them read-only access (your grade will also be impacted if you do not).

This Phase consists of two steps. First, is to perform an oral interview with the student that developed the project. Second, is to perform the code review itself.

1. Oral Interview

You will first prepare for an oral interview by reading the requirements, design documents for the student project you were assigned. You will prepare a *list of questions* that **must be included in your final report**. The questions should be along the lines as specified below.

You will first conduct an oral interview with the student whose project you were assigned in regards with the following:

- What the application does at a high level and what functionality it provides.
- What tests are provided and how to setup and run the application and tests locally.

- Ask questions with regards to whether (if it implements anything in regards to the category) and what the application performs as related to the following security categories:
    - Inputs
    - Cryptography
    - Authentication
    - Authorization
    - Session Management
    - Error Handling
    - Logging
    - Debugging
- Ask about any assumptions that the reviewer should make on behalf of the developer when performing the review. For example, should we assume the environment where the code will run is already secured (here the security team may have already gone and secured the webserver where the code will be hosted in production).

The oral interview does not have to take more than 60 – 90 minutes.

2. Secure Code Review

Now you will perform a secure code review based on the answers to the questions, your own manual code review and analysis as well as any code analysis tools you use to find issues related to the security categories listed above. A recommendation is to first assemble your findings in an Excel spreadsheet with the following columns:

- CWE-ID
- Security Category
- Filename
- Line number: Position
- Description: This field should be very detailed on what the security issue found is and why it is a real issue (for example, if an attack cannot take advantage of the weakness, it should not be included as a finding).
- Technical Impact: Each weakness CWE has a potential technical impact associated with it. What is the technical impact for the weakness you have chosen specifically with regards the application you are reviewing?
- Criticality: Critical, High, Medium, Low, Info. You should choose this based on the potential technical impact an attack (that you identified in the two bullets above) can have on the application.
- Potential Mitigation: This field should include how a potential fix could be implemented to avoid the weakness identified.
- Comments: Your comments on the finding to the developer of the project.

Now assemble a report addressed to the development team (and manager if any) containing the following sections:

1. **Summary** section – This should summarize all the findings you found based on the impacts it can have on the application. You should also include whether your findings included any weaknesses in the latest list for Top 25 most dangerous weaknesses list. For

example: This review found four (4) critical issues related to Input Handling, Authentication and Authorization that could result in Execution of Arbitrary Code, Denial of Service: CPU and Memory. These issues should be addressed immediately or the application could suffer from severe security vulnerabilities that adversaries could take advantage of. The review also found two (2) High, three (3) Medium and one (1) Info level issues related to … The review also included two findings that were in the Top 25 most dangerous weaknesses list. They are …

2. **Introduction** section – This should explain the process you went through to do the review including the questions that were asked and what methodology you followed to do the actual review.
    - You should include the name(s) of the developer who wrote the code and how they provided you with the code base (Gitlab URL for example) and which commit-id they provided that you reviewed.
    - You should include any assumptions the developer told you to make during your oral interview.
    - You should also include how you performed the secure code review. How did you manually go through the application to review the code base? What strategy did you use? What, if any, automated tools static analysis tools or dynamic analysis tools did you use and how did you remove all the false positives?
    - You should also explain your process for selecting the criticality of the issues. For example, the code review team chose to label issues as *Critical* when a finding weakness had a technical impact of *Denial of Service: Application* since the availability of the application was listed as important as per the application requirements OR the application use case seems to have a need for high availability.
    - It should also include whether you found all the possible bugs or not.
3. **Interview** section – This section should go over what questions you asked and some of your notes on answers. This is typically used to guide your review and findings that you make. This section should include, for example, what assumptions the developers wanted you to make for the review.
4. **Code Review Findings** section – This should go into detail on each of the findings and include the fields from the Excel spreadsheet you created above in a table format. You should *group* findings by **weakness** and *sort by* **criticality**. This should also include the potential mitigations for each finding. For example:

CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	
Criticality	Critical
Security Category	Input Handling
Filename	WebGoat/Content/SqlInjection.aspx.cs
Line number/position	570
Description	The code on this line constructs a SQL query using user provided input without any sanitization leading to potential SQL Injection. On this page, an adversary is able to leak all the login email ids of customers by crafting a SQL injection

	input. This could lead to a loss of confidentiality for sensitive user's data in the application.
Technical Impact	Read Application Data – Note: Here the SQL injection could result in loss of confidentiality. This page does not allow for bypassing access control or other technical impacts.
Potential Mitigations	Consider using parameterized queries, sanitizing user inputs using regular expressions or encoding inputs …
Comments	This seems to be a test page, and if so, it should be removed from the application before releasing it to production.

5. **Impact** statement – This statement should state what would happen with the application if the findings are **not** addressed. Will this result in horrendous implications for the organization hosting the running code if these weaknesses are discovered and not addressed?

Submission Criteria

Please upload a **Word** or **PDF** document of your report as part of the ELMS Project Phase 3 assignment and make sure it includes the following:

- Your Name
- Your UMD email id
- The student project you were assigned (for example: ENPM809WProject-gkini)
- The content for the report as outlined above.