CPSC 525 Project (Fall 2025)

Due on last day of classes, December 5, 2025.
Late policy if submitted by Sunday, December 14 (9 days late): no late penalty.
Late policy if submitted after December 14 (10 or more days late): 100% late penalty.
Note that this is different from, but more lenient than, the late policy for assignments.

You will work on the project in groups of 4-5 students. Each group will be assigned a different topic, corresponding to one CWE (Common Weakness Enumeration). Your group will need to submit 3 deliverables:

- Blog post related to your CWE;
- Your own vulnerable application, corresponding attack code and a description of fixed code;
- A video presentation summarizing your project.

Separate links to each of the above items must be submitted on D2L by one of the group members.

Each member of the group will optionally also submit a self-evaluation and a peer-evaluation.

Selecting your group

Approximately 40 groups have been created on D2L, and the title of each group represents the assigned CWE topic for that group. Each student must self-register for one of these groups, ideally based on their interest in the assigned topic. The signup will start on Wednesday, October 1 at 4pm, and the deadline to sign up is Monday, October 5 at 11pm. Students that do not pick a group by the deadline will be assigned a random group.

The group self-registration will be on a first-come, first-served basis. Once a group fills out, it will be unavailable for selection. If you want to be in a group with a friend, I suggest that you pick a mostly empty group, and you both sign up at the same time.

Please note that once you enroll in a group, you cannot unenroll yourself from it. If you make a mistake, and need to change your group, you will need to make a private post on Ed Discussion, requesting that you be removed from your current group. Once we remove you, you will be able to select a new group. Requests for changing groups will be honored until October 12.

Deliverable: Blog Post (33%)

Write a ~3000-word (not including code snippets) blog post related to your group's CWE. You can find information about specific CWEs from https://cwe.mitre.org. The target audience of your post will be other CPSC 525 students. Your blog post will consist of the following:

1. CWE ID and name;
2. Overview;
3. Real World Example;
4. Citations.

Your blog post should be read like an article. Write full paragraphs as much as possible and minimize use of bullet points.

Overview section

The purpose of the Overview section is for you to provide a high-level explanation of your CWE. You must clearly articulate each of the following:

- which programming languages and contexts your CWE most commonly applies to;

- key ideas underlying the CWE and its exploitation;
- implications of successful exploitation of the CWE;
- standard techniques for avoiding or securing against the CWE;
- at least one concise pair of annotated code snippets contrasting vulnerable versus equivalent non-vulnerable code;
- pointers to additional resources relevant to the CWE that you found on the internet.

Real World Example section

In this section, you will explore your assigned CWE at a deeper technical level. For this, you need to find a real-world example of the CWE leading to actual exploitation and describe it in detail. You can search for exploits using sites such as:

- https://www.cisa.gov/known-exploited-vulnerabilities-catalog and
- https://nvd.nist.gov/vuln/search

Ideally, the exploit you pick should be (a) technically interesting, (b) high-profile and (c) somewhat recent. If you cannot find an exploit matching all three criteria, prioritize (a). Your blog post should include:

1. CVE number; KEV number; as appropriate;
2. technical description of the exploit, including:
    a. short snippet(s) of source code, if appropriate;
    b. short snippet of input/code triggering the exploit, if appropriate;
    c. diagrams, if appropriate;
3. business, legal and social implications of the exploit.

If you are unable to find an actual exploit, you can substitute a documented vulnerability (CVE) and then describe a hypothetical exploit using it. You can search for CVEs arising from CWE using https://www.cve.org.

Submission

- Post your Blog Post on Ed Discussion in the "Blog_Post" category and make it public.
- Submit a link to the blog post in the comment box of the D2L dropbox.

Deliverable: Vulnerable Application (33%)

Write a simple (but vulnerable) application that runs on departmental Linux servers (csx1-3) and demonstrates the CWE assigned to your group and its exploitation in practice. You can pick any type of application to implement, but it must be fully functional and non-trivial, and it must contain an exploitable vulnerability directly related to your group's CWE.

To give you an idea of the complexity we will be expecting your application to have, here are some examples:

- Text-based to-do application;
- Web-based cooking recipe application;
- GUI application for note taking.

Your application should have around 1000 lines of code and can be written in any language(s) supported by our Linux servers.

The application and should work correctly under "normal" circumstances but should be also exploitable. You will need to submit attack code that can successfully exploit the vulnerability embedded in your application.

Your code should be well formatted, use reasonable naming conventions, and include comments as appropriate – especially to explain important CWE-relevant steps. It should also be easy to compile and run your code, e.g., using a Makefile or similar.

Unusual technical requirements

Depending on the language and nature of the vulnerability, you may need additional software and/or hardware to get things running. Examples would be: mongodb server, particular PHP version, sudo access. Make a private post on Ed Discussion to inquire whether we can accommodate such requests using our resources. Some of you may even be able to provision your own servers to satisfy unusual requirements. Before you do, please make sure to talk to us on Ed Discussion to verify we will be able to access your server for grading purposes.

Submission

You must use some type of version control for your code (e.g. GitHub, GitLab, or similar) in a public repository. Your repository must contain:

- all code necessary to compile/run your vulnerable application;
- all code necessary to compile/run your exploit;
- README.md file describing:
  - high level description of your code and exploit (~1 paragraph, can be bulleted);
  - how and where to compile/run your code;
  - how and where to compile/run your exploit;

In other words, you must make available to us all materials and instructions to be able to run your code and exploit. If you miss anything, we will not be able to run your code, and you will receive no marks for this component.

Submit a link to your repository on D2L. Verify that the link works in an incognito browser.

Deliverable: Video presentation (34%)

Record a ~10-minute video presentation containing a summary of the blog post and your vulnerable application. Viewers should gain a high-level understanding of the CWE, the real-world example of it, and they should learn how to use the vulnerable code and attack samples your group produced. The video should not attempt to include all of the technical details covered in the post and your application; rather, it should focus on the highlights to entice viewers to dig deeper by reading your group's post and playing with your code.

The video should start with a 5 second title screen, containing the CWE number and title. If you want, you can include your names on the title screen as well, but this is not required. You do not have to make yourself visible during the video recording. Not all group members need to speak or otherwise participate during the video recording. If you wish, a single group member can narrate the entire presentation.

We suggest the following format for the presentation:

- title slide;
- ~5 minute slide presentation with voice-over related to your blog posts;
  - Make some PowerPoint or Google Slides.
  - Record yourself while you narrate the slides.
- ~5 minute demo of your vulnerable application and the exploit;
  - Record yourself compiling & running your application, showing that it functions correctly;
  - Show and explain the part of your code where the vulnerability exists.
  - Record yourself exploiting the application, showing that it is vulnerable;
  - Add voiceover to the video and/or add subtitles & background music.

You are free to use whichever software you prefer to record, edit and publish the video. If you have never created videos before, a good starting point might be Yuja, as it can be used to record, edit and publish your videos, and our university has a license for it. Feel free to use more sophisticated tools if you are comfortable with them, including OBS Studio, Davinci Resolve, YouTube.

Submission

Submit a link to your video (e.g. YouTube, Yuja) in the comment box in D2L dropbox.

Self-evaluation and Peer-evaluation

Each group member will have an option to submit an evaluation of themselves and their peers. These evaluations will in most cases not directly factor into your grade, though we reserve the right to penalize or reward you if all group members feel strongly that you contributed far more or far less than your fair share. Your groupmates will not see the assessments you write, and you will not see the assessments they write about you. If you decide to submit an evaluation, fill out the following survey by December 15:

https://forms.office.com/r/HTsi6TZZUj

If you do not fill out this survey, we will assume that all members of your group, including yourself, contributed **roughly** the same amount of effort, and that all group members deserve the same grade. In other words, you only need to fill out this survey if you believe some of your group members (including you) deserve higher or lower grade than the rest of the group.

Miscellaneous comments

- all written work you submit must be your own;
- use of generative AI is forbidden;
- in your blog posts and your recording, you may use code snippets, graphics, diagrams that are not your own, but you must cite them;
- post questions about the project on Ed Discussion in the "Project" folder;

D2L Submission

Submit 3 links on D2L:

- link to your Blog Post on Ed Discussion
- link to your code (e.g. GitLab)
- link to your video (e.g. YouTube, YuJa)

List of topics (CWEs)

For convenience, here is the list of topics associated with the different D2L groups:

- CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
- CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
- CWE-94: Improper Control of Generation of Code ('Code Injection')
- CWE-111: Direct Use of Unsafe JNI
- CWE-122: Heap-based Buffer Overflow
- CWE-184: Incomplete List of Disallowed Inputs
- CWE-190: Integer Overflow or Wraparound
- CWE-193: Off-by-one Error
- CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
- CWE-209: Generation of Error Message Containing Sensitive Information
- CWE-215: Insertion of Sensitive Information Into Debugging Code

- CWE-259: Use of Hard-coded Password
- CWE-269: Improper Privilege Management
- CWE-276: Incorrect Default Permissions
- CWE-285: Improper Authorization
- CWE-306: Missing Authentication for Critical Function
- CWE-312: Cleartext Storage of Sensitive Information
- CWE-330: Use of Insufficiently Random Values
- CWE-352: Cross-Site Request Forgery (CSRF)
- CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
- CWE-367: Time-of-check Time-of-use (TOCTOU) Race Condition
- CWE-415: Double Free
- CWE-416: Use After Free
- CWE-441: Unintended Proxy or Intermediary ('Confused Deputy')
- CWE-446: UI Discrepancy for Security Feature
- CWE-476: NULL Pointer Dereference
- CWE-489: Active Debug Code
- CWE-502: Deserialization of Untrusted Data
- CWE-532: Insertion of Sensitive Information into Log File
- CWE-611: Improper Restriction of XML External Entity Reference
- CWE-625: Permissive Regular Expression
- CWE-656: Reliance on Security Through Obscurity
- CWE-681: Incorrect Conversion between Numeric Types
- CWE-682: Incorrect Calculation
- CWE-754: Improper Check for Unusual or Exceptional Conditions
- CWE-918: Server-Side Request Forgery (SSRF)
- CWE-1284: Improper Validation of Specified Quantity in Input
- CWE-1339: Insufficient Precision or Accuracy of a Real Number
- CWE-1395: Dependency on Vulnerable Third-Party Component