Template based on the Centers for Medicare & Medicaid Services, Information Security & Privacy Management's Assessment

Security Assessment Report

Version N.1

May 1, 2023

Table of Contents

1. Summary

The overall goal of this security assessment is to fix holes that can be easily exploited for malicious intent and to prevent users from either accidentally or purposefully breaking the program.

1. Assessment Scope

The OS used was Windows 11, unfortunately no access to other OS was available to test the project on. The IDE used was CLion, this is where the code was written and tested.

2. Summary of Findings

Of the findings discovered during our assessment, 0 were considered High risks, 2 Moderate risks, 0 Low, and 0 Informational risks. The SWOT used for planning the assessment are broken down as shown in Figure 1.

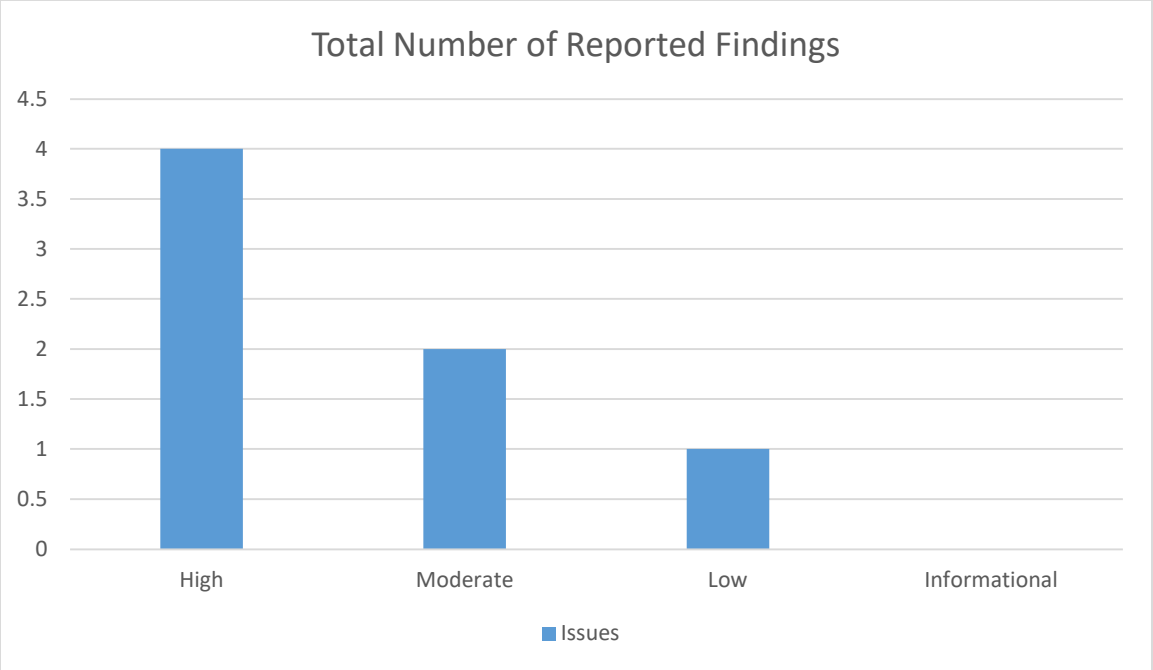


Figure 1. Findings by Risk Level

Explain above and link to full table of explanation of top risks like Figure 2.

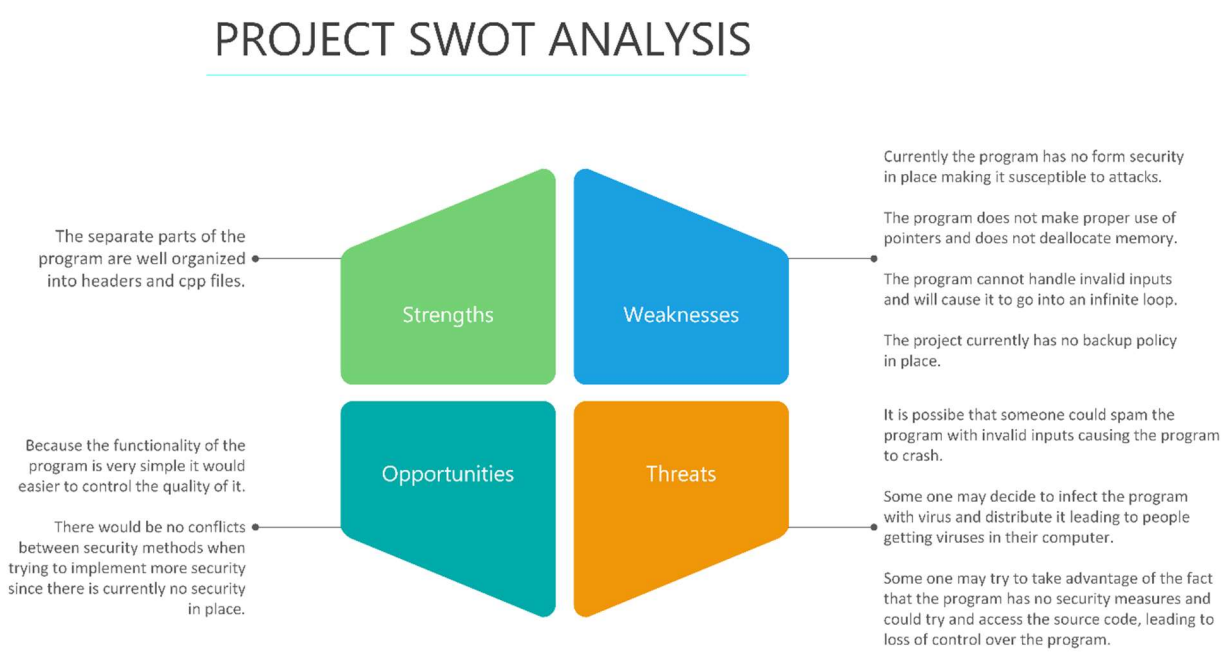


Figure 2. SWOT

Explain which issues were used from above SWOT (which are addressed in this assessment).

3. Summary of Recommendations

The changes that were made are that the pointers in the project were changed to smart pointers, the functions that set the values for the shape objects no longer return pointers and are now simply void functions. Another change that was made is the input provided by the user is now handled differently, the user must input numbers between -100 and 100, otherwise they will be given an error message and be asked to try again. The changes that are still recommended to be made are to have the program handle different types of invalid input such as strings, and to create a back up policy for the project.

2. Goals, Findings, and Recommendations

1. Assessment Goals

The purpose of this assessment was to do the following:

- To experience the process of securing and fixing a project.
- To maintain a previous project
- Have a project that can be added to a resume as a showcase of being aware of security
- Improve skill as a software engineer to better understand and recognize why code is unsafe

2. Detailed Findings

The project had memory leaks due to poorly implemented pointers such as making pointers using "new" but not deallocating the memory that the pointers used afterwards making the program less efficient, this would be considered a weakness. Similarly, the project also returned pointers in some of the functions this also led to memory that was created but not deallocated leading to the same issue of decreased performance, this would also be considered a weakness. Another issue the project had is that it does not properly handle improper input such as chars and strings, this leads to the project entering an infinite loop forcing the user to manually stop the program, this would be considered a weakness. Another issue with input is that the user would be able to input numbers larger than the int data type could hold, leading to buffer issues this would be considered a threat. Another vulnerability with the project is that the message that requests input is misleading and led to users accidentally providing input in a format that will cause the code to break, this would be considered a weakness.

A majority of these findings were pointed out by Professor Greenwell.

Advice and information for fixing these issues came from in class lectures, Microsoft Learn for information on pointers and smart pointers, and StackOverflow for code ideas on implementing the smart pointers with a vector.

3. Recommendations

The issue of fixing the memory leaks would be a very difficult aspect to fix since it would require either using something other than pointers which would mean that you would need to change every single function in the project that handles pointers essentially resulting a different project altogether, or it would require you to switch to using smart pointers and implementing them would also lead to other bugs popping up. The input handling would be moderately difficult because it may require that you make relatively big changes to a certain portion of the code but it will not require an entire redesign of it, it is also not as time consuming as some more difficult fixes. Creating a back up policy is something that is easy to do since it is quick to do and does not require you to make changes to the code.

3. Methodology for the Security Control Assessment

3.1.1 Risk Level Assessment

Each Business Risk has been assigned a Risk Level value of High, Moderate, or Low. The rating is, in actuality, an assessment of the priority with which each Business Risk will be viewed. The definitions in **Error! Reference source not found.** apply to risk level assessment values (based on probability and severity of risk). While Table 2 describes the estimation values used for a risk's "ease-of-fix".

Table 1 - Risk Values

Rating	Definition of Risk Rating
High Risk	Exploitation of the technical or procedural vulnerability will cause substantial harm to the business processes. Significant political, financial, and legal damage is likely to result
Moderate Risk	Exploitation of the technical or procedural vulnerability will significantly impact the confidentiality, integrity and/or availability of the system, or data. Exploitation of the vulnerability may cause moderate financial loss or public embarrassment to organization.
Low Risk	Exploitation of the technical or procedural vulnerability will cause minimal impact to operations. The confidentiality, integrity and availability of sensitive information are not at risk of compromise. Exploitation of the vulnerability may cause slight financial loss or public embarrassment
Informational	An "Informational" finding, is a risk that has been identified during this assessment which is reassigned to another Major Application (MA) or General Support System (GSS). As these already exist or are handled by a different department, the informational finding will simply be noted as it is not the responsibility of this group to create a Corrective Action Plan.
Observations	An observation risk will need to be "watched" as it may arise as a result of various changes raising it to a higher risk category. However, until and unless the change happens it remains a low risk.

Table 2 - Ease of Fix Definitions

Rating	Definition of Risk Rating
Easy	The corrective action(s) can be completed quickly with minimal resources, and without causing disruption to the system or data
Moderately Difficult	Remediation efforts will likely cause a noticeable service disruption • A vendor patch or major configuration change may be required to close the vulnerability • An upgrade to a different version of the software may be required to address the impact severity • The system may require a reconfiguration to mitigate the threat exposure • Corrective action may require construction or significant alterations to the manner in which business is undertaken
Very Difficult	The high risk of substantial service disruption makes it impractical to complete the corrective action for mission critical systems without careful scheduling • An obscure, hard-to-find vendor patch may be required to close the vulnerability • Significant, time-consuming configuration changes may be required to address the threat exposure or impact severity • Corrective action requires major construction or redesign of an entire business process
No Known Fix	No known solution to the problem currently exists. The Risk may require the Business Owner to: • Discontinue use of the software or protocol • Isolate the information system within the enterprise, thereby eliminating reliance on the system In some cases, the vulnerability is due to a design-level flaw that cannot be resolved through the application of vendor patches or the reconfiguration of the system. If the system is critical and must be used to support on-going business functions, no less than quarterly monitoring shall be conducted by the Business Owner, and reviewed by IS Management, to validate that security incidents have not occurred

Access Control and Network Security Checklist

ID	Security Control issue	Applicable (Y/N)	Complete (Y/N/NA)	To be Done Priority (High, Mid, Low)	Ease of fix (Easy, Moderate, Difficult, Not Fixable)	Full Description of processes to address issue
1	A cloud based platform is being used for access control with only public available items being readable by general public.	N	N/A			N/A
2	The cloud based platform provides for hiding information and this is used to protect sensitive information and code.	N	N/A			N/A
3	OS access controls are used to only allow authorized changes to be made to code.	N	N/A			N/A
4	used to only allow changes to be made to code by authorized individuals.	N	N/A			N/A
5	Backup Policy is in place and being used.	Y	N	High	Easy	The source code should be saved on a backup drive in case the computer that it is stored in is compromised or destroyed.
6	Third-Party libraries used in code are up-to-date and have been checked to ensure no security issues exist.	N	N/A			N/A
7	Physical Security of actual computer code is stored on is adequete	Y	Y	High	Easy	The computer that stores the code should have the basic security measurements in place such as an antivirus, a firewall, and a VPN
8	Accounting: Logging is integrated into the code itself (for exceptions, errors, and user input failures at minimum)	Y	N	High	Difficult	The project should handle invalid input in a way that does not cause the code to abruptly break.
9	Accounting: Process includes logging (tracking of changes, user making changes, access attempts, etc)	N	N/A			N/A
10	PKI and other encryption and authentication methods are used to connect to cloud platform	Y	N	Mid	Moderate	Which Certifying Authority & what authentication methods?
11	Internal Actor threats are accounted for and policies/planning is in place for these.	Y	N	Low	Not Fixable	What is the planning/policies?
12	Standard Unit Testing used	N	N/A			N/A
13	Security Testing used (the type varies)	Y	N	High	Moderate	The project should be tested for any possible weaknesses before being released.
14	The project uses an adequate amount of	Y	N	Mid	Easy	The project will have enough comments that properly detail how the less obvious parts of the code work in order be able to maintain and

3.1.2 Tests and Analyses

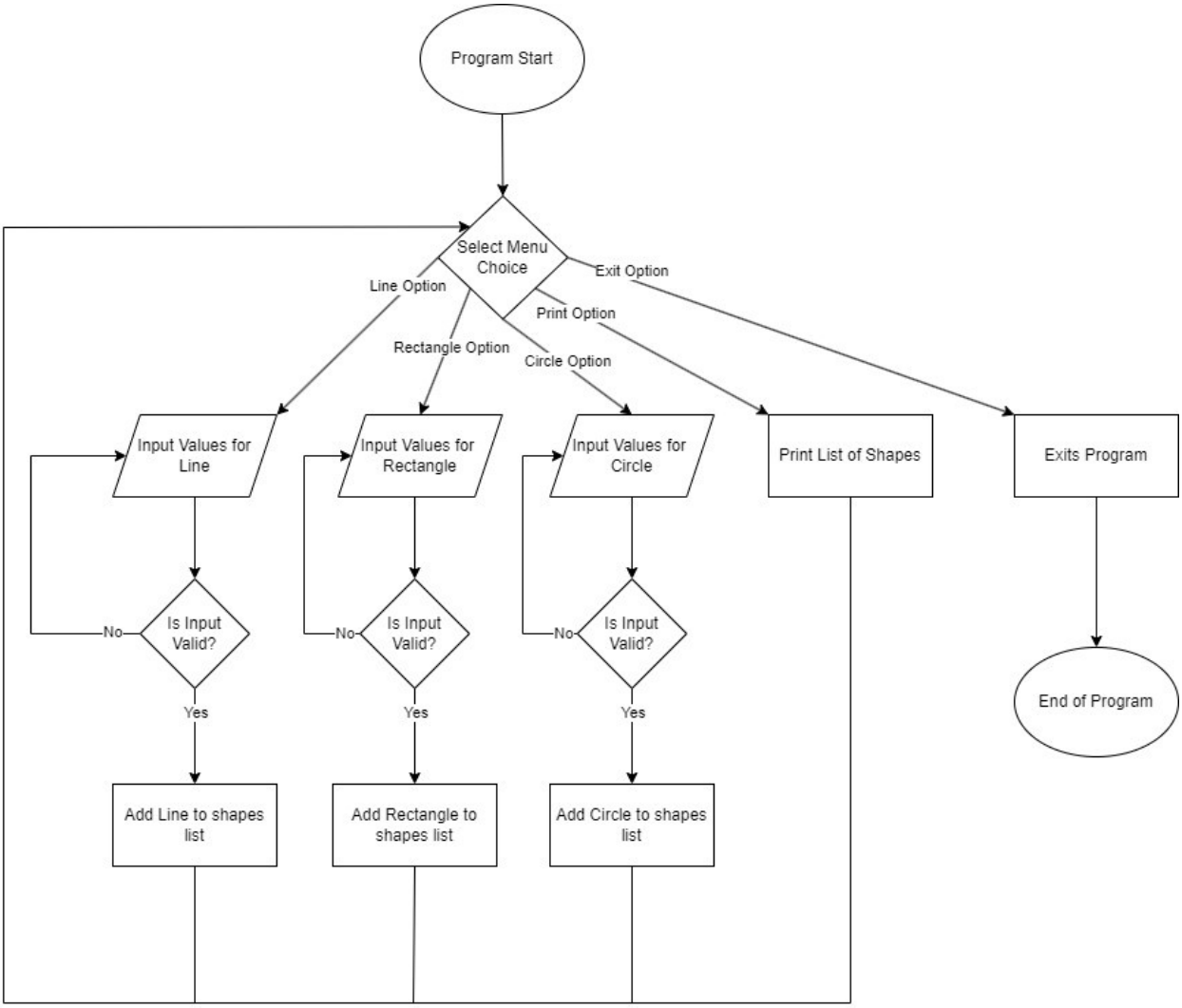
Both white box and black box testing was conducted. For white box testing I ran through every possible scenario I could think of, testing whether different inputs could be handled, checking if the loops after receiving invalid numbers worked, and I also checked each shape calculation multiple times. Then for black box testing I had 2 users that have no knowledge of how the program functions use the program and then recorded how often it would break, and I found that one managed to break the program 3 out of the 10 times that it was ran, and the other broke the program 2 out of the 10 times.

3.1.3 Tools

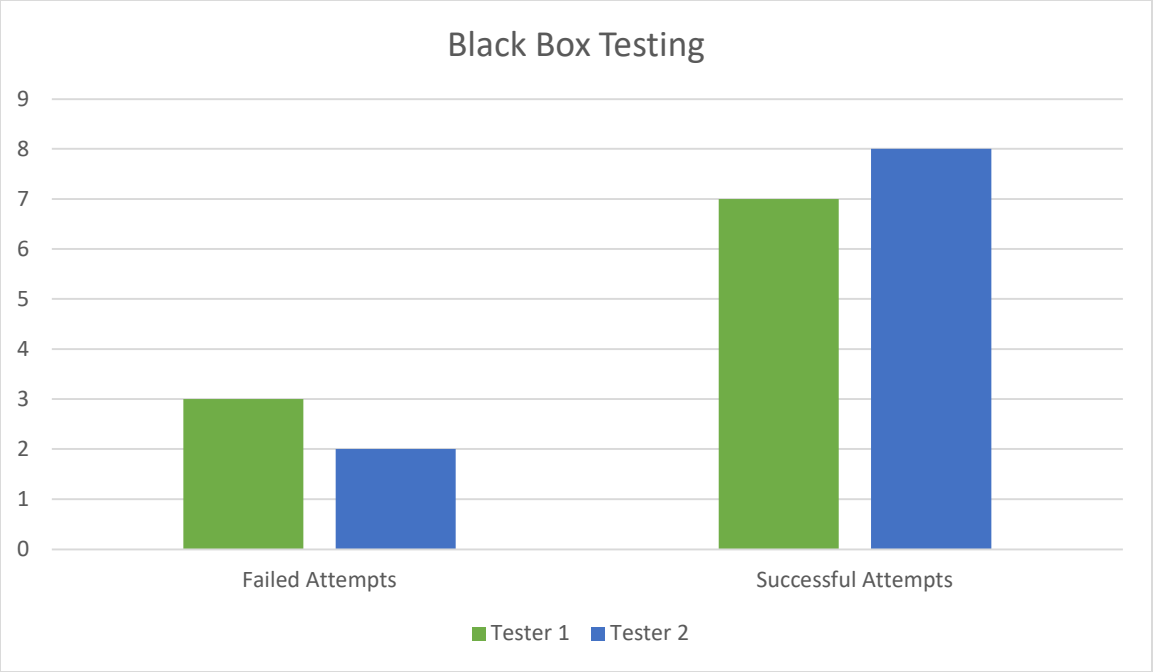
No special tools were used for testing the software as all testing occurred in the IDE, however in the future I would be interested in using the different tools that are offered in git hub, but for this assessment I was unsure of how the different tools were meant to be used.

4. Figures and Code

4.1.1 Process or Data flow of System (this one just describes the process for requesting), use-cases, security checklist, graphs, etc.



The program starts by presenting a menu with 5 choices, a line option, a rectangle option, a circle option, a print shapes option, and an exit option. When either the line, rectangle, or circle options are selected the program will then prompt the use to input the values for the shape, then if the input is valid the shape created by the user will be added to list of shapes and it will loop back to present the user with the menu again, but if the input is not valid then the program will prompt the user for input again, this will repeat until the input received from the user is valid. I the user selects the print shapes option, then the program will print the list of all the shapes that the user input into the program, then it will loop back to the menu and prompt the user to select an option again. If the user selects the exit option, then the program will simply exit.



4.1.2 Other figure of code

Pointer Fixes:

```
std::vector<std::shared_ptr<Shape>> shapes{};
auto line = std::make_shared<Line>();
auto rectangle = std::make_shared<Rectangle>();
auto circle = std::make_shared<Circle>();
```

Example of Input Handling:

```
while(loop){
    std::cout << "Enter the first point (values between -100 and 100) \n";
    std::cout << "Enter x: ";
    std::cin >> x1;
    std::cout << "Enter y: ";
    std::cin >> y1;
    //values are limited in size to avoid buffer issue
    if((x1 >= -100 && x1 <= 100) && (y1 >= -100 && y1 <= 100)){
        loop = false;
    }
    else{
        std::cout << "\nInvalid Input.\n Please enter points that are between
-100 and 100.";
        loop = true;
    }
}
```

5. Works Cited

TylerMSFT. (2022, November 6). *Raw pointers (C++)*. Microsoft Learn. Retrieved April 30, 2023, from https://learn.microsoft.com/en-us/cpp/cpp/raw-pointers?view=msvc-170

EmerEmer 3, Georg FritzscheGeorg Fritzsche 97.1k2626 gold badges193193 silver badges235235 bronze badges, sigfpesigfpe 7, StephenStephen 47.6k77 gold badges6161 silver badges6969 bronze badges, KBurchfielKBurchfiel 62566 silver badges1515 bronze badges, Clark GaebelClark Gaebel 17.1k1919 gold badges6565 silver badges9393 bronze badges, mattmatt 3, Aakash NaikAakash Naik 6111 silver badge11 bronze badge, & Subh_bSubh_b 1344 bronze badges. (2010, June 30). *How to delete a pointer after returning its value inside a function*. Stack Overflow. Retrieved April 30, 2023, from https://stackoverflow.com/questions/3145799/how-to-delete-a-pointer-after-returning-its-value-inside-a-function

user2434918user2434918 31711 gold badge22 silver badges99 bronze badges, & vsoftcovsoftco 55k1010 gold badges135135 silver badges247247 bronze badges. (2015, February 26). *C++11 vector of smart pointer*. Stack Overflow. Retrieved April 30, 2023, from https://stackoverflow.com/questions/28733385/c11-vector-of-smart-pointer

TylerMSFT. (2021, August 2). *Smart pointers (modern C++)*. Microsoft Learn. Retrieved April 30, 2023, from https://learn.microsoft.com/en-us/cpp/cpp/smart-pointers-modern-cpp?view=msvc-170

Greenwell, J. (2023, March). *From Web to Software*. *In Class Lecture*. Fort Myers; Florida Gulf Coast University.

Greenwell, J. (2023, April). *Whitebox and Blackbox Testing*. *In Class Lecture*. Fort Myers; Florida Gulf Coast University.