Portfolio Project – Part 2

Java VS C++

Arianna Martinez

CSU – Global

CSC450

Professor Bari

November 5, 2023

Java VS C++

Programming language security is an important factor to take into account when developing software. The built-in security features of Java and C++, memory safety, exception handling, and the dangers of C++'s pointer usage will all be compared in this essay. We can learn more about the advantages and disadvantages of each language by looking at these factors, which will help us decide which language offers a safer environment for creating secure software applications.

Java comes with a number of built-in security mechanisms that enhance its overall security. The bytecode verification procedure is one such feature that makes sure the code complies with stringent guidelines prior to execution. This procedure offers an effective defense against security flaws by drastically lowering the possibility of malicious code execution. A sandbox environment with stringent access controls is made available by the Java Virtual Machine (JVM), prohibiting unwanted access to system resources. Because of this isolation, Java programs are more secure overall since malicious code cannot interfere with sensitive operations. Java's automatic garbage collection feature reduces the need for manual memory management, which lowers the possibility of memory-related vulnerabilities like buffer overflows and memory corruption.

Java performs very well in memory safety, which is a key part of program security. Memory leaks and weak pointers are largely avoided thanks to Java's garbage collection technology. Java lowers the possibility of memory-related vulnerabilities and enhances overall application stability by automatically recovering memory that is no longer in use. Compared to C++, Java offers a greater level of memory safety thanks to its usage of references and object encapsulation. Because of such references, which guarantee secure object access and prevent unwanted access to memory blocks, this encapsulation enhances data integrity and lowers the possibility of security flaws brought on by direct memory manipulation. Another important component of memory safety is Java's array bounds checking. Java prevents out-of-limits errors, which might access unwanted memory regions and result in security vulnerabilities, by enforcing array bounds.

Now to the errors, Java has a strong framework for handling many types of failures, and exception handling is a central component of software development. Java's exception-handling feature helps to prevent unexpected program termination and encourages strong error handling. Developers can avoid security risks and recover from errors gracefully by catching and handling exceptions. Unhandled exceptions can lead to security flaws. And developers are encouraged to explicitly handle any mistakes via Java's usage of verified exceptions. This method makes sure that possible problems are dealt with and not overlooked, which lowers the chance of security vulnerabilities. Exception handling is not enforced in C++, which might result in unhandled exceptions and security flaws. So error recovery may become more difficult and security vulnerabilities may become more likely in C++ due to the absence of a defined method for handling exceptions.

Due to its lack of integrated security protections, C++ is more prone to security flaws. In C++, memory-related vulnerabilities such as buffer overflows and memory corruption are more likely when manual memory management is used. Undefined behavior is another problem with C++ that can arise from out-of-bounds mistakes. “Out-of-Bounds Write vulnerabilities can be exploited by an attacker to crash a program or execute arbitrary code. In some cases, these vulnerabilities can beOut exploited to overwrite critical program data structures,” (Security Boulevard, 2022). Ultimately, attackers may use this undefinable behavior to intercept or inject data, jeopardizing the application's security. So this makes the use of pointers in C++ increase the danger of unauthorized access to memory blocks. Pointers give users immediate access to memory locations, so if other programs are able to access or modify them, this could lead to illegal memory access and security flaws.

Pointers are strong yet possibly hazardous because they give immediate access to memory locations. “…pointers allow us to access a memory location in the computer’s memory,” (GeeksForGeeks). Security flaws could result from other programs accessing memory blocks, so in an ideal world, they should not be able to if they are able to obtain or modify pointers. “Pointers are extremely powerful because they allows you to access addresses and manipulate their contents. But they are also extremely complex to handle. Using them correctly, they could greatly improve the efficiency and performance. On the other hand, using them incorrectly could lead to many problems…” (NTU, 2013). In C++, it counts to use secure coding techniques and proper pointer management to reduce the possibility of unwanted memory access. In the end, to avoid unwanted access, developers must make sure that pointers are utilized securely, checked, and cleaned up appropriately.

To sum this all up, software development must take application and implementation safety into account. Java offers a safer environment for creating secure software applications because of its memory safety measures, strong exception handling, and built-in security features. While C++ has fewer built-in security features, is prone to undefined behavior and memory-related vulnerabilities, and creates a danger of pointer-based unauthorized access to memory blocks. Both languages require constant upkeep and secure coding techniques, but Java is a more secure option for developing software in many areas of the process.

Works Cited

Ballman, A. (2016). SEI CERT C++ coding standard: Rules for developing safe, reliable, and secure systems in C++. Software Engineering Institute (Carnegie Mellon University). Hanscom, MA.

NTU. (2013, April). C++ programming language. C++ Pointers and References. (n.d.). https://www3.ntu.edu.sg/home/ehchua/programming/cpp/cp4\_PointerReference.html

GeeksforGeeks. (2022, October 25). C++ Pointers. GeeksforGeeks. https://www.geeksforgeeks.org/cpp-pointers/

What is an out-of-bounds read and out-of-bounds write error?. Security Boulevard. (2022, July 1). https://securityboulevard.com/2022/06/what-is-an-out-of-bounds-read-and-out-of-bounds-write-error/