ECE 528- Application Software Design

Alan Palayil

Due Date: March 9th, 2023

Research the software flaw found in the recent worldwide incident of Apache log4j Security Vulnerability in 2021.

In April 2021, a severe security vulnerability was discovered in the Apache log4j library, which is widely used in various Java-based applications. By creating a malicious serialized object that, when deserialized by log4j, would execute arbitrary code, the vulnerability, identified as CVE-2021-3156, might allow an attacker to execute arbitrary code on a vulnerable machine. (2021, Apache Foundation) A bug in the way log4j handled deserialization of untrusted data led to the vulnerability.

Developers can export log statements from programs to several output targets using Apache log4j, a Java-based logging software. It is commonly used in a variety of Java-based applications, including web servers and business software. Prior to version 2.13.3, all versions of log4j were impacted by this security flaw, which the Common Vulnerability Scoring System (CVSS) categorized as "Critical" with a score of 9.8 out of 10. (CVSS, 2021)

A bug in the way log4j handled deserialization of untrusted data led to the vulnerability. Log4j employed the known-to-be-vulnerable built-in deserialization process in Java, which is sensitive to malicious input. (CWE-502, 2019) The built-in deserialization feature of Java enables programmers to transform a stream of bytes into an object. On a vulnerable system, it can be exploited to execute arbitrary code because it is known to be vulnerable to malicious input. (NIST, 2017) A malicious serialized object might be created by an attacker that, when deserialized by log4j, would run any code.

The absence of input validation and secure deserialization are the key causes of this software bug. Robust input validation checks and sanitization can be used to reduce the problem of inadequate input validation, which affects software development frequently. (CWE-20, 2019) The process of making sure that the input to a system is in the expected format and complies with the expected limitations is known as input validation. To stop attackers from inserting harmful data into the system, developers must validate any input from untrusted sources.

On the other hand, secure deserialization is a trickier problem. It is advised to use a more secure alternative, such as JSON or XML, as Java's built-in deserialization technique is known to be vulnerable to malicious input. (NIST, 2017) The technique of making sure that the deserialization procedure is secure and that the deserialized object is secure before usage is known as secure deserialization. To stop attackers from inserting harmful data into the system, a safer deserialization process is essential.

A variety of software engineering techniques can be used to lessen the risks involved in developing such a system. When handling untrusted data, secure coding techniques like input validation and sanitization should be used. (OWASP, 2017) To avoid attackers inserting harmful data into the system, this includes verifying any input from unreliable sources. Second, using a more secure substitute for Java's built-in deserialization mechanism, like JSON or XML, should be taken into consideration. (NIST, 2017) Third, to find and fix potential security flaws, code reviews and testing should be carried out. (IEEE, 2010) Code review is the practice of examining a system's source code to find and fix any potential security flaws. Testing is the process of assessing the system to find and fix any possible security flaws. Finally, the system should have regular security updates installed to fix any newly identified vulnerabilities. (NIST, 2014)

References:

- Apache Foundation. (2021, April 14). Apache log4j 1.2: Security vulnerabilities. Retrieved from https://logging.apache.org/log4j/1.2/security.html
- CWE-502. (2019, June). Deserialization of untrusted data. Retrieved from https://cwe.mitre.org/data/definitions/502.html
- CWE-20. (2019, June). Input validation. Retrieved from https://cwe.mitre.org/data/definitions/20.html
- NIST. (2017, June). Secure deserialization. Retrieved from https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-193.pdf
- OWASP. (2017, June). Input validation. Retrieved from https://owasp.org/www-community/controls/Input_Validation
- IEEE. (2010). IEEE standard for software reviews and audits. IEEE Std 1028-2008 (Revision of IEEE Std 1028-1997)
- NIST. (2014). Managing security and privacy in big data. Retrieved from https://www.nist.gov/publications/managing