snhu

CS 305 Project One Vulnerability Assessment Report

Document Revision History

Version	Date	Author	Comments
1.0	9/21/2024	Sarah Tomlinson	Filled out Vulnerability Assessment

Client

ARTEMIS
FINANCIAL

Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In this report, identify your security vulnerability findings and recommend the next steps to remedy the issues you have found.

- Respond to the five steps outlined below and include your findings.
- Respond using your own words. You may also include images or supporting materials. If you include them, make certain to insert them in the relevant locations in the document.
- Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

Developer
Sarah Tomlinson

1. Interpreting Client Needs

Artemis Financial is a finance consulting firm that develops individualized plans for their customers' finances. Secure communication is important for any organization, but especially within a financial corporation to protect company and client private data. Artemis Financial is certainly involved in international transactions due to their role in customer financials, insurance, investments, etc. There are several governmental recommendations surrounding secure communications and the protection of consumer data such as social security numbers, banking information, biometrics, etc. from being leaked or discovered via unsecure communication. This same information is the reason external threats are a concern, because this information can be utilized for financial gain by attackers. As far as modernization requirements are concerned, it is important to ensure that all libraries and web application technologies used stay up to date with the most current and secure versions to ensure that security threats are minimized.

2. Areas of Security

- Input Validation – Input validation will be important to Artemis Financial because all input needs to be validated to prevent malicious error triggering and/or SQL Injection.
- APIs – Artemis Financial will be using a RESTful web API that needs to have proper security protocols because this API will determine how users will interact with the program.
- Code Error – Error handling is necessary especially alongside input validation to ensure that any errors that do occur are handled properly to prevent unauthorized access to private accounts and information.
- Code Quality – Ensuring quality code means ensuring that the code follows all the best security practices including but not limited to user authentication and user access control.

3. Manual Review

- *Input Validation* is not properly handled in GreetingController.java. Because there is no input validation around line 16/17, the program is subject to injection attacks.
- *API* – There is code in place for the API (CRUD operations), but the API itself is not present in the code. This needs to be integrated into the program for full security and functionality.
- *Cryptography* – No encryption found in the program. Data encryption should be implemented to further protect sensitive consumer information.
- *Code Error* in line 27/30 of DocData.java. Line 27 is a link to a SQL database that uses credentials directly accessible in the same line of code ("root", "root"). The credentials should not be included directly (hard-coded) into the program. Line 30 is a catch block but does not properly handle and log exceptions properly. printStackTrace() is helpful for debugging code, but should be replaced with a logging framework before the code is released.
- *Old Version* of org.springframework.boot in pom.xml file. The program uses version 2.2.4, and the most up to date version is 3.3.4. Using the old version is a potential security risk, while the updated version has additional features and security measures.

4. Static Testing

Dependency	Description	Vulnerability Codes	Solution
bcprov-jdk15on-1.46.jar	The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7.	CVE-2013-1624, CVE-2015-6644, CVE-2015-7940, CVE-2016-1000338, CVE-2016-1000339, CVE-2016-1000341, CVE-2016-1000342, CVE-2016-1000343, CVE-2016-1000344, CVE-2016-1000346, CVE-2016-1000352, CVE-2017-13098, CVE-2018-5382, CVE-2020-0187, CVE-2020-26939, CVE-2023-33201, CVE-2024-29857, CVE-2024-29857, CVE-2024-30171, CVE-2024-34447	Update to Bouncy Castle Java version 1.78.1.
spring-boot-2.2.4.RELEASE.jar	In Spring Boot versions 3.0.0 - 3.0.5, 2.7.0 - 2.7.10, and older unsupported versions, an application that is deployed to Cloud Foundry could be susceptible to a security bypass.	CVE-2022-27772, CVE-2023-20873, CVE-2023-20883	Upgrade to version 3.0.6+ or 2.7.11+.
logback-core-1.2.3.jar	A serialization vulnerability in logback receiver component part of logback version 1.4.11 allows an attacker to mount a Denial-Of-Service attack by sending poisoned data.	CVE-2021-42550, CVE-2023-6378	Upgrade to version 1.3.14+ or 1.4.14+.
log4j-api-2.12.1.jar	Apache Log4j2 versions 2.0-beta7 through 2.17.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable to a remote code execution (RCE) attack when a configuration uses a JDBC Appender with a JNDI LDAP data source URI when an attacker has control of the target LDAP server.	CVE-2020-9488, CVE-2021-44228, CVE-2021-44832, CVE-2021-45046, CVE-2021-45105	Upgrade to version 2.17.1, 2.3.2, or 2.12.4.
snakeyaml-1.25.jar	SnakeYaml's Constructor () class does not restrict types which can be instantiated during deserialization. Deserializing yaml content provided by an attacker can lead to remote code execution.	CVE-2017-18640, CVE-2021-4235, CVE-2022-1471, CVE-2022-25857, CVE-2022-3064, CVE-2022-38749, CVE-2022-38750, CVE-2022-38751, CVE-2022-38752, CVE-2022-41854	Upgrade to version 2.0+
jackson-databind-2.10.2.jar	jackson-databind 2.10.x through 2.12.x before 2.12.6 and 2.13.x before 2.13.1 allows attackers to cause a denial of service (2 GB transient heap	CVE-2020-25649, CVE-2020-36518, CVE-2021-46877, CVE-2022-42003,	Upgrade to version 2.15.2+. This also fixes an issue that allows attackers

	usage per read) in uncommon situations involving JsonNode JDK serialization.	CVE-2022-42004, CVE-2023-35116	to cause a denial-of-service attack.
tomcat-embed-core-9.0.30.jar	Generation of Error Message Containing Sensitive Information vulnerability in Apache Tomcat.This issue affects Apache Tomcat: from 8.5.7 through 8.5.63, from 9.0.0-M11 through 9.0.43.	CVE-2019-17569, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-13943, CVE-2020-17527, CVE-2020-1935, CVE-2020-1938, CVE-2020-8022, CVE-2020-9484, CVE-2020-24122, CVE-2021-25122, CVE-2021-25329, CVE-2021-30640, CVE-2021-33037, CVE-2021-41079, CVE-2021-43980, CVE-2022-29885, CVE-2022-34305, CVE-2022-42252, CVE-2023-28708, CVE-2023-41080, CVE-2023-42795, CVE-2023-44487, CVE-2023-45648, CVE-2023-46589, CVE-2024-21733	Users are recommended to upgrade to version 8.5.64 onwards or 9.0.44 onwards, which contain a fix for the issue.
hibernate-validator-6.0.18.Final.jar	A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages.	CVE-2020-10693	Upgrade to version 6.0.20.Final.
spring-web-5.2.3.RELEASE.jar	Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data.	OSSINDEX [CVE-2016-1000027, CVE-2020-5421, CVE-2021-22096, CVE-2021-22118, CVE-2024-2243, CVE-2024-2262, CVE-2024-38809]	Upgrade to version 6.0.x+. (Current Version)
spring-beans-5.2.3.RELEASE.jar	A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.	OSSINDEX [CVE-2022-22965]	Upgrade to version 6.0.x+. (Current Version)
spring-webmvc-5.2.3.RELEASE.jar	In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries.	OSSINDEX [CVE-2021-22060, CVE-2024-38816]	Upgrade to version 6.0.x+. (Current Version)
spring-context-5.2.3.RELEASE.jar	In Spring Framework versions 5.3.0 - 5.3.18, 5.2.0 - 5.2.20, and older unsupported versions,	OSSINDEX [CVE-2022-22968]	Upgrade to version 6.0.x+. (Current Version)

	the patterns for disallowedFields on a DataBinder are case sensitive which means a field is not effectively protected unless it is listed with both upper and lower case for the first character of the field, including upper and lower case for the first character of all nested fields within the property path.		
spring-expression-5.2.3.RELEASE.jar	In Spring Framework versions 5.3.0 - 5.3.16 and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition.	OSSINDEX [CVE-2022-22950, CVE-2023-20861, CVE-2023-20863, CVE-2024-38808]	Upgrade to version 6.0.x +. (Current Version)

5. Mitigation Plan

For the vulnerabilities found in the static testing report, upgrading to the versions suggested by the vendor (listed in the solutions tab) will solve most of the issues. The upgraded versions contain fixes for the vulnerability codes listed.

For the issues found in the manual review, there are a variety of fixes needed. The springframework.boot currently used is version 2.2.4, this should be upgraded to version 3.3.4. The earlier versions contain a possible deserialization attack, along with other security flaws. Then the input validation issue needs to be resolved by properly handling the input and implementing stronger validation and sanitation to all endpoints that accept user input, reducing the program's vulnerability to SQL injection attacks. The credentials in the found code error need to be changed so that they are not hard coded into the program. This can be done any number of ways, including using variables or a config file, to protect the information. The API code needs to be added to the program for it to become more user friendly, and data encryption should be implemented to better protect sensitive consumer information. Finally, error handling should be implemented with more meaningful logging and clear and consistent error messages.