

**CS 305 Project One**

**Artemis Financial Vulnerability Assessment Report**

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| **1.0** | **07/23/2022** | **Jordan Barnes** |  |

Client



Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found.

Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

Developer

Jordan Barnes

1. Interpreting Client Needs

Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?

Securing communications is very important to the health of a company for several reasons. During the assessment, it is required that you protect the customer and client data so that intellectual property can not be compromised. When transactions occur between client and company there may be times when there is communication on unsecure mediums. This produces a risk to data integrity. There needs to be a secure protocol for data communication and transactions.

* Are there any international transactions that the company produces?

Global Rain is an engineering company that specializes in custom software design and development for entrepreneurs, businesses, and government agencies around the **world**. Apart from those company services, being a developer and promoted to examine client's system, your company can provide online assessment and training across the globe for international transaction. It is also likely, although not explicitly stated, that the company recieves assistance from foreign entitys for software solutions.

* Are there governmental restrictions about secure communications to consider?

For the mentioned transactions and processes, there is no government restrictions to consider. Secure communication is processed through an encrypted communication method. There is no third party during the conversation between client and company.

* What external threats might be present now and in the immediate future?

As the company increases in size, external threats become more likely. In most cases, phishing and external intrusion is inevitable. Humans tend to make common errors that lead to large problems. Mitigating these possibilties and placing barriers to prevent them from happening is the best you can do.

* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

Upgrade to to the latest versions of API packages and improve antiquated hardware.

2. Areas of Security

Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

The areas of security that are applicable to Artemi's Financial software application include:

Cryptography - It would be a significant feature to incorporate cryptography into the system. This is due to information moving through our system can be made more secure. Encryption ensures that information cannot be accessed by outside parties without proper authentication. This function aids in preventing unauthorized users from accessing and decoding data.

API - For the purposes of the APIs, the application's security area is required. Ensuring that the API guards against any security lapses that may result from prospective hackers and outside threats is crucial to the security of everything in the system. In essence, the API serves as a mediator between two or more apps so they can communicate with one another. API is also used to log people into the system without the use of external third-party applications.

Client server - This is the communication that takes place between the client and the server in order to complete the duties that have been given to it. The client server includes capabilities that can safely secure data so that unauthorized people cannot access it. The client server serves as a temporary storage location for the system's data before it is backed up and delivered to a remote server.

3. Manual Review

Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

* It doesn't seem to have any unit testing, integration testing, or end-to-end testing. This needs to be fixed.
* There is no explicit cryptographic protocol, which is highly recommended.
* Although not in the code, there should be firmware inspections completed aswell.

4. Static Testing

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from dependency check report. Include the following:

* The names or vulnerability codes of the known vulnerabilities. A brief description and recommended solutions provided by the dependency check report
* Vulnerable Dependencies: 11

1. In Bouncy Castle JCE Provider version 1.55 and earlier the DSA does not fully validate ASN.1 encoding of signature on verification. It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' data into a signed structure.
2. 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.
3. A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.
4. Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender. Fixed in Apache Log4j 2.12.3 and 2.13.1
5. In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers.
6. The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564.
7. spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.
8. A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.
9. Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data. Depending on how the library is implemented within a product, this issue may or not occur, and authentication may be required. NOTE: the vendor's position is that untrusted data is not an intended use case. The product's behavior will not be changed because some users rely on deserialization of trusted data.
10. When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.

5. Mitigation Plan

After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.

It is highly recommended that you first incorporporate an explicit cryptographic protocol for communication. You can switch to HTTPS for all direct contact and avoid most unwanted breaches. Redirect request paramaters to headers instead of what is currently in code. There should be a dirty word search for any sensitive data in the code and should be replaced. Establish two factor authentication systems for user login. Update all dependencies and address the security vunerabilities above. Finally, write unit tests, integration tests, and end-to-end tests to ensure code maintainability and security.

Citations:

Seetharaman , J. (n.d.). DevSecOps: A Systemic Approach for Secure Software Development. ISSA Journal.