CS 305 Project One Template

Document Revision History

Version	Date	Author	Comments
1.0	5/22/24	Summer Bernotas	Updated client needs and areas of security
1.1	5/24/24	Summer Bernotas	Performed static and manual testing on code base, documented findings here

Client

Developer
Summer Bernotas

1. Interpreting Client Needs

Artemis Financial is a consulting company that develops financial plans for savings, retirement, investment, and insurance accounts for its customers. Having effective security on their software is absolutely crucial to their success as they handle people's most sensitive information on a daily basis. Due to this, they expect to implement the newest and most effective software security possible as they move forward with modernizing their operations. There has also been no specification as to whether Artemis Financial handles international transactions, with that in mind, we will proceed with the assumption that they handle both foreign and non-foreign transactions to ensure security in all areas.

Artemis Financial is required to follow government regulations regarding any and all financial transactions. This, as well as basic legal compliance, means that data retention will have restrictions and certain policies set that affect how security measures are implemented. With the company handling financial transactions, there are endless amounts of threats that may become present. Secure authentication, attention to data interception, third parties, and hackers are just some of the risks to be aware of that could cause serious threat to the system.

2. Areas of Security

Input validation – This will be required to validate user accounts and owner information
APIs – This controls information both internally and externally and will account for what information is accessible to either source
Cryptography – This will ensure proper encryption of information that is being passed through the API is protected
Code Error – This will allow us to understand which areas of the API need fixed in order to ensure security and safety of the information being passes
Encapsulation – This will ensure that users do not have access to specific information that is not their own or is solely held for the purpose of internal security measures

3. Manual Review
After reviewing the code and applying the vulnerability assessment, I have found the following…

GreetingController.java – This class is lacking input validation
POM.XML – This file does not have any Apache validation

Aside from specific classes, the system is also missing authentication and cryptography all together. There is also no HTTP, which is vital for security.

4. Static Testing

Dependency	Vulnerability	Description
bcprov-jdk15on-1.46.jar	cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:*:*:*:*:*:*:*	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.
spring-boot-2.2.4.RELEASE.jar	cpe:2.3:a:vmware:spring_boot:2.2.4:release:*:*:*:*:*:*	Spring Boot
logback-core-1.2.3.jar	cpe:2.3:a:qos:logback:1.2.3:*:*:*:*:*:*:*	logback-core module
log4j-api-2.12.1.jar	cpe:2.3:a:apache:log4j:2.12.1:*:*:*:*:*:*:*	The Apache Log4j API
snakeyaml-1.25.jar	cpe:2.3:a:snakeyaml_project:snakeyaml:1.25:*:*:*:*:*:*:* cpe:2.3:a:yaml_project:yaml:1.25:*:*:*:*:*:*:*	YAML 1.1 parser and emitter for Java
jackson-databind-2.10.2.jar	cpe:2.3:a:fasterxml:jackson-databind:2.10.2:*:*:*:*:*:*:*	General data-binding functionality for Jackson: works on core streaming API
tomcat-embed-core-9.0.30.jar	cpe:2.3:a:apache:tomcat:9.0.30:*:*:*:*:*:*:* cpe:2.3:a:apache_tomcat:apache_tomcat:9.0.30:*:*:*:*:*:*:*	Core Tomcat implementation
hibernate-validator-6.0.18.Final.jar	cpe:2.3:a:redhat:hibernate_validator:6.0.18:*:*:*:*:*:*:*	Hibernate's Bean Validation (JSR-380) reference

		implementation.
spring-web-5.2.3.RELEASE.jar	cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:* cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*	Spring Web
spring-beans-5.2.3.RELEASE.jar	cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:* cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*	Spring Beans
spring-webmvc-5.2.3.RELEASE.jar	cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:* cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*	Spring Web MVC
spring-context-5.2.3.RELEASE.jar	cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:* cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*	Spring Context
spring-expression-5.2.3.RELEASE.jar	cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:*:* cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:*:*	Spring Expression Language (SpEL)

Almost all dependencies can be fixed if each individual software is upgraded to its newer version. This excludes *snakeyaml-1.25.jar* which needs migrated to the SnakeYAML Enginge that will allow for configuration options

5. Mitigation Plan

After evaluation of both the static and manual review, it is safe to say that the best and easiest starting point to remedy the identified security vulnerabilities would be to upgrade all of the dependencies. From there we can work on missing aspects such as input validation and the use of HTTP