****

# Artemis Financial Vulnerability Assessment Report

Table of Contents

[Document Revision History 3](#_Toc32574607)

[Client 3](#_Toc32574608)

[Instructions 3](#_Toc32574609)

[Developer 4](#_Toc32574610)

[1. Interpreting Client Needs 4](#_Toc32574611)

[2. Areas of Security 4](#_Toc32574612)

[3. Manual Review 4](#_Toc32574613)

[4. Static Testing 4](#_Toc32574614)

[5. Mitigation Plan 4](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **07/16/2023** | **Connie Knupp** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for 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 choose to include images or supporting materials. If you include them, make certain to insert them in all 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

Connie Knupp

## Interpreting Client Needs

As a financial institution Artemis Financial will “almost certainly be facing highly motivated and highly skilled attackers,” (Detlefsen & Manico, 2014). Recognizing the potential for security threats they have asked for assistance modernizing their RESTful web application with an emphasis on data security. Secure communications are imperative as their communications often contain sensitive data. Like most financial institutions Artemis Financial will be conducting business throughout the world. This means they need to abide by the *Gramm-Leach-Bliley Act* which requires Financial Institutions to notify customers of how their data is shared, as well as Execute Orders such as *Executive Order 13873 Securing the Information and Communications Technology and Service Supply Chain* and *Executive Order on Protecting Americans’ Sensitive Dat from Foreign Adversaries.* They will also have to abide by any data security laws in countries where they will be conducting business. These laws require companies to make all efforts to protect sensitive data and alert consumers when there has been a breach in security.

Artemis Financial will likely face phishing threats, threats of hackers attempting SQL injection as well as attempts to bypass security restrictions and permissions. They will need to guard against these as well as the possibility of disgruntled employees or unscrupulous competitors attempting to cause a denial of services errors or breaches to cause loss of consumer trust in the company. These threats can lead to catastrophic consequences when conducting business in the financial sector, including stolen money, regulatory fines, and loss of customers. As we work to modernize this API we need to carefully assess any open-sourced libraries included in the project. While Open-sourced libraries can speed the process of updating code they can also introduce vulnerabilities, this means assessing those vulnerabilities and working to mitigate any known weaknesses. It is imperative that all included libraries are kept as up-to-date as possible which helps mitigate errors as they are found.

## Areas of Security

|  |  |
| --- | --- |
| Area of Security | Importance in Project |
| Input Validation | Input needs to be checked for type as well as length to ensure only acceptable input is received. If inputs are not checked to ensure safety threats can manipulate the system. |
| Secure API interactions | The importance of secure API interactions was stated perfectly by Compiègne when he wrote “Vulnerabilities and weaknesses in REST APIs will offer attackers the opportunity to gain access to services and information, compromising the integrity of business systems. APIs, in general, provide more options to attackers than traditional network access, so robust REST API security is vital” (2021). |
| Cryptography | Sensitive data like passwords and account numbers as well as sensitive communications should be encrypted to ensure security if they are intercepted. |
| Coding Practices/Patterns | Creating a secure software application must begin with secure coding practices and patterns including permissions checks for function calls and query parameterization. This reduces the likelihood an attacker can inject unsafe commands into the SQL or other key areas in the program. “Because of the level of the risk posed by SQL injection, query parameterization is the most important technique a developer can use to build a secure website” (Detlefsen & Manico, 2014). |
| Error Handling | Handling errors appropriately and securely will ensure that errors do not cause the system to fail open and grant entry into the system for malicious threats. |
| Encapsulation | Encapsulating data into structures and limiting interaction with data to secure functions allows those functions to be accessed by only the program or individuals who have the appropriate permissions. This is an important extra layer of security in writing safe software. |

## Manual Review

Here are vulnerabilities found in reviewing the initial code sent to us here at Global Rain:

1. In the customer class at line 5 – account\_balance should be set to private and should be a float variable type.
2. In the customer class – there is no a public constructor. There is no set function for setting the customer account\_number or account\_balance. There is no get function for customer account\_balance. These functions need to be created and should include a check to ensure the user has an active session and has proper permission to access the information. They need to include a try-catch statement to notify users when they do not have permission to get or set these variables.
3. In the customer class, lines 7-9 – the function should pass through the access controls to ensure the user is in an active session and has authorization to do this function. Account numbers should also be encrypted when sent from the system and decoded on the receiving in to ensure it is not vulnerable to interception.
4. In the customer class, line 12 – variable a should be a float to allow for depositing a fraction of a dollar. This function should also pass through the access control to ensure the user is in an active session and has authorization to do this function.
5. In the GreetingController class, line 16 – RequestParam should be replaced with a parameterized query via the PreparedStatement class to protect from SQL injection. This should also be in a try-catch statement to ensure proper error handling. If it fails do not run the command.
6. In the CRUDController class, line 13 - RequestParam should be replaced with a parameterized query via the PreparedStatement class to protect from SQL injection. This should also be in a try-catch statement to ensure proper error handling. If it fails do not run the command.
7. In the CRUD class, lines 7-9 – both this. content and this.content2 point to the same input string. Test to ensure this does not cause unintended results.
8. In the DocData class, line 16 – this function should include a pass to access control to ensure the user has an active session and has proper permission. This should be encased in a try-catch statement to notify the user if unable to get ID due to permissions.
9. In the DocData class, line 29 – the catch block needs to be completed. This function should also include a call to the access controller to ensure the user has an active session and permission to connect to the database.
10. All libraries included should be updated to the latest version in order to protect against known threats.

## Static Testing

The information on the following table was imported from the Maven dependency report.

|  |  |  |
| --- | --- | --- |
| Dependency | Description of Vulnerabilities Found | Vulnerabilities Count and Highest Severity |
| [bcprov-jdk15on-1.46.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l1_991c96a4e31e6c19e2b9136c8955bd423f2dc4c7) | [**CVE-2016-1000338**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000338)**,** [**CVE-2016-1000342**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000342)**-** 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.  [**CVE-2016-1000343**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000343) **-** In the Bouncy Castle JCE Provider version 1.55 and earlier the DSA key pair generator generates a weak private key if used with default values. If the JCA key pair generator is not explicitly initialised with DSA parameters, 1.55 and earlier generates a private value assuming a 1024 bit key size. In earlier releases this can be dealt with by explicitly passing parameters to the key pair generator.  [**CVE-2016-1000344**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000344)**,** [**CVE-2016-1000352**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000352) **-** In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider.  [**CVE-2016-1000341**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000341) - In the Bouncy Castle JCE Provider version 1.55 and earlier DSA signature generation is vulnerable to timing attack. Where timings can be closely observed for the generation of signatures, the lack of blinding in 1.55, or earlier, may allow an attacker to gain information about the signature's k value and ultimately the private value as well.  [**CVE-2016-1000345**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000345) **-** In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES/ECIES CBC mode vulnerable to padding oracle attack. For BC 1.55 and older, in an environment where timings can be easily observed, it is possible with enough observations to identify when the decryption is failing due to padding.  [**CVE-2017-13098**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-13098) **-** BouncyCastle TLS prior to version 1.0.3, when configured to use the JCE (Java Cryptography Extension) for cryptographic functions, provides a weak Bleichenbacher oracle when any TLS cipher suite using RSA key exchange is negotiated. An attacker can recover the private key from a vulnerable application. This vulnerability is referred to as "ROBOT."  [**CVE-2020-15522**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-15522)  - Bouncy Castle BC Java before 1.66, BC C# .NET before 1.8.7, BC-FJA before 1.0.1.2, 1.0.2.1, and BC-FNA before 1.0.1.1 have a timing issue within the EC math library that can expose information about the private key when an attacker is able to observe timing information for the generation of multiple deterministic ECDSA signatures.  **CVE-2020-0187** (OSSINDEX) **-** In engineSetMode of BaseBlockCipher.java, there is a possible incorrect cryptographic algorithm chosen due to an incomplete comparison. This could lead to local information disclosure with no additional execution privileges needed. User interaction is not needed for exploitation.Product: AndroidVersions: Android-10Android ID: A-148517383  [**CVE-2016-1000339**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000339) - In the Bouncy Castle JCE Provider version 1.55 and earlier the primary engine class used for AES was AESFastEngine. Due to the highly table driven approach used in the algorithm it turns out that if the data channel on the CPU can be monitored the lookup table accesses are sufficient to leak information on the AES key being used. There was also a leak in AESEngine although it was substantially less. AESEngine has been modified to remove any signs of leakage (testing carried out on Intel X86-64) and is now the primary AES class for the BC JCE provider from 1.56. Use of AESFastEngine is now only recommended where otherwise deemed appropriate.  **CVE-2020-26939** (OSSINDEX - In Legion of the Bouncy Castle BC before 1.61 and BC-FJA before 1.0.1.2, attackers can obtain sensitive information about a private exponent because of Observable Differences in Behavior to Error Inputs. This occurs in org.bouncycastle.crypto.encodings.OAEPEncoding. Sending invalid ciphertext that decrypts to a short payload in the OAEP Decoder could result in the throwing of an early exception, potentially leaking some information about the private exponent of the RSA private key performing the encryption.  **CVE-2023-33201** (OSSINDEX) - Bouncy Castle For Java before 1.74 is affected by an LDAP injection vulnerability. The vulnerability only affects applications that use an LDAP CertStore from Bouncy Castle to validate X.509 certificates. During the certificate validation process, Bouncy Castle inserts the certificate's Subject Name into an LDAP search filter without any escaping, which leads to an LDAP injection vulnerability.  [**CVE-2015-7940**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-7940) **-** The Bouncy Castle Java library before 1.51 does not validate a point is withing the elliptic curve, which makes it easier for remote attackers to obtain private keys via a series of crafted elliptic curve Diffie Hellman (ECDH) key exchanges, aka an "invalid curve attack."  [**CVE-2018-5382**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2018-5382) **-** The default BKS keystore use an HMAC that is only 16 bits long, which can allow an attacker to compromise the integrity of a BKS keystore. Bouncy Castle release 1.47 changes the BKS format to a format which uses a 160 bit HMAC instead. This applies to any BKS keystore generated prior to BC 1.47. For situations where people need to create the files for legacy reasons a specific keystore type "BKS-V1" was introduced in 1.49. It should be noted that the use of "BKS-V1" is discouraged by the library authors and should only be used where it is otherwise safe to do so, as in where the use of a 16 bit checksum for the file integrity check is not going to cause a security issue in itself.  [**CVE-2013-1624**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-1624) **-**  The TLS implementation in the Bouncy Castle Java library before 1.48 and C# library before 1.8 does not properly consider timing side-channel attacks on a noncompliant MAC check operation during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets, a related issue to CVE-2013-0169.  [**CVE-2016-1000346**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000346) **-** In the Bouncy Castle JCE Provider version 1.55 and earlier the other party DH public key is not fully validated. This can cause issues as invalid keys can be used to reveal details about the other party's private key where static Diffie-Hellman is in use. As of release 1.56 the key parameters are checked on agreement calculation. **CVE-2015-6644** (OSSINDEX) - Bouncy Castle in Android before 5.1.1 LMY49F and 6.0 before 2016-01-01 allows attackers to obtain sensitive information via a crafted application, aka internal bug 24106146. | 18 - High |
| [classmate-1.5.1.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l2_3fe0bed568c62df5e89f4f174c101eab25345b6c) | Dependency without known vulnerability. Library for introspecting types with full generic information including resolving of field and method types. | 0 |
| [hibernate-validator-6.0.18.Final.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l3_7fd00bcd87e14b6ba66279282ef15efa30dd2492) | Hibernate's Bean Validation (JSR-380) reference implementation.  [**CVE-2020-10693**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-10693) - 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. | 1 - Medium |
| [jackson-core-2.10.2.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l4_73d4322a6bda684f676a2b5fe918361c4e5c7cca) | Core Jackson processing abstractions (aka Streaming API), implementation for JSON. Dependency without reported vulnerability. | 0 |
| [jackson-databind-2.10.2.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l5_0528de95f198afafbcfb0c09d2e43b6e0ea663ec) | General data-binding functionality for Jackson: works on core streaming API.  [**CVE-2020-25649**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-25649) **-** 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.  [**CVE-2020-36518**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-36518) - jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects.  [**CVE-2021-46877**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-46877) **-** 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 usage per read) in uncommon situations involving JsonNode JDK serialization.  [**CVE-2022-42003**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42003) **-** In FasterXML jackson-databind before 2.14.0-rc1, resource exhaustion can occur because of a lack of a check in primitive value deserializers to avoid deep wrapper array nesting, when the UNWRAP\_SINGLE\_VALUE\_ARRAYS feature is enabled. Additional fix version in 2.13.4.1 and 2.12.17.1  [**CVE-2022-42004**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42004) **-** In FasterXML jackson-databind before 2.13.4, resource exhaustion can occur because of a lack of a check in BeanDeserializer.\_deserializeFromArray to prevent use of deeply nested arrays. An application is vulnerable only with certain customized choices for deserialization.  [**CVE-2023-35116**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-35116) **-** \*\* DISPUTED \*\* An issue was discovered jackson-databind thru 2.15.2 allows attackers to cause a denial of service or other unspecified impacts via crafted object that uses cyclic dependencies. NOTE: the vendor's perspective is that the product is not intended for use with untrusted input. | 6 - High |
| [jakarta.annotation-api-1.3.5.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l6_59eb84ee0d616332ff44aba065f3888cf002cd2d) | Jakarta Annotations API  Dependency without reported vulnerability. | 0 |
| [jakarta.validation-api-2.0.2.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l7_5eacc6522521f7eacb081f95cee1e231648461e7) | Jakarta Bean Validation API  Dependency without reported vulnerability. | 0 |
| [jboss-logging-3.4.1.Final.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l8_40fd4d696c55793e996d1ff3c475833f836c2498) | The JBoss Logging Framework  Dependency without reported vulnerability. | 0 |
| [jul-to-slf4j-1.7.30.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l9_d58bebff8cbf70ff52b59208586095f467656c30) | JUL to SLF4J bridge  Dependency without reported vulnerability. | 0 |
| [log4j-api-2.12.1.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l10_a55e6d987f50a515c9260b0451b4fa217dc539cb) | The Apache Log4j API  [**CVE-2020-9488**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9488) **-**  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 | 1 - Low |
| [log4j-to-slf4j-2.12.1.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l11_dfb42ea8ce1a399bcf7218efe8115a0b7ab3788a) | The Apache Log4j binding between Log4j 2 API and SLF4J.  Dependency without reported vulnerability. | 0 |
| [logback-core-1.2.3.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l12_864344400c3d4d92dfeb0a305dc87d953677c03c) | logback-core module  [**CVE-2021-42550**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550) **-** 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. | 1 – Medium |
| [slf4j-api-1.7.30.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l13_b5a4b6d16ab13e34a88fae84c35cd5d68cac922c) | The slf4j API  Dependency without reported vulnerability. | 0 |
| [snakeyaml-1.25.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l14_8b6e01ef661d8378ae6dd7b511a7f2a33fae1421) | YAML 1.1 parser and emitter for Java  [**CVE-2022-1471**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-1471) **-** 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. We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. We recommend upgrading to version 2.0 and beyond.  [**CVE-2017-18640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640) **-** The Alias feature in SnakeYAML before 1.26 allows entity expansion during a load operation, a related issue to CVE-2003-1564.  [**CVE-2022-25857**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-25857) **-** The package org.yaml:snakeyaml from 0 and before 1.31 are vulnerable to Denial of Service (DoS) due missing to nested depth limitation for collections.  [**CVE-2022-38749**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38749) **-** Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow.  [**CVE-2022-38751**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38751) - Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow.  [**CVE-2022-38752**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38752) **-** Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stack-overflow.  [**CVE-2022-41854**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-41854) **-** Those using Snakeyaml to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stack overflow. This effect may support a denial of service attack.  [**CVE-2022-38750**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38750) **-** Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow. | 8 - Critical |
| [spring-boot-2.2.4.RELEASE.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l15_225a4fd31156c254e3bb92adb42ee8c6de812714) | Spring Boot  [**CVE-2022-27772**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772) - \*\* UNSUPPORTED WHEN ASSIGNED \*\* 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.  [**CVE-2023-20883**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883) **-** In Spring Boot versions 3.0.0 - 3.0.6, 2.7.0 - 2.7.11, 2.6.0 - 2.6.14, 2.5.0 - 2.5.14 and older unsupported versions, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache. | 2 - High |
| [spring-boot-starter-web-2.2.4.RELEASE.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l16_ec75d01d212b5229c16d872fb127744c0ed46ed8) | Starter for building web, including RESTful, applications using Spring  MVC. Uses Tomcat as the default embedded container  [**CVE-2022-27772**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772) **-** \*\* UNSUPPORTED WHEN ASSIGNED \*\* 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.  [**CVE-2023-20883**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883) **-** In Spring Boot versions 3.0.0 - 3.0.6, 2.7.0 - 2.7.11, 2.6.0 - 2.6.14, 2.5.0 - 2.5.14 and older unsupported versions, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache. | 2 - High |
| [spring-core-5.2.3.RELEASE.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l17_3734223040040e8c3fecd5faa3ae8a1ed6da146b) | Spring Core  [**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965) - 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.  [**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118) **-** In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data.  [**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421) **-** In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.  [**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950) **- I**n 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.  [**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971) **-** In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user.  [**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861) **-** In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863) **-** In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968) **-** In Spring Framework versions 5.3.0 - 5.3.18, 5.2.0 - 5.2.20, and older unsupported versions, 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.  [**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970) - In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, applications that handle file uploads are vulnerable to DoS attack if they rely on data binding to set a MultipartFile or javax.servlet.Part to a field in a model object.  [**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060) **-** 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. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase.  [**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096) **-** In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. | 11 – Critical |
| [spring-web-5.2.3.RELEASE.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l18_dd386a02e40b915ab400a3bf9f586d2dc4c0852c) | Spring Web  [**CVE-2016-1000027**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000027) **-** 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.  [**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965) **-** 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.  [**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118) **-** In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data.  [**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421) - In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.  [**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950) - 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.  [**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971) - In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user.  [**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861) - In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863) **-** In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968) **-** In Spring Framework versions 5.3.0 - 5.3.18, 5.2.0 - 5.2.20, and older unsupported versions, 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.  [**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970) **-** In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, applications that handle file uploads are vulnerable to DoS attack if they rely on data binding to set a MultipartFile or javax.servlet.Part to a field in a model object.  [**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060) **-** 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. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase.  [**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096) **-** In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. | 12 – Critical |
| [spring-webmvc-5.2.3.RELEASE.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l19_745a62502023d2496b565b7fe102bb1ee229d6b7) | Spring Web MVC  [**CVE-2022-22965**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22965) - 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.  [**CVE-2021-22118**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22118) **-** In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data.  [**CVE-2020-5421**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421) **-** In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.  [**CVE-2022-22950**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950) **- I**n 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.  [**CVE-2022-22971**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22971) **-** In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user.  [**CVE-2023-20861**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20861) **-** In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2023-20863**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20863) **-** In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.  [**CVE-2022-22968**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22968) **-** In Spring Framework versions 5.3.0 - 5.3.18, 5.2.0 - 5.2.20, and older unsupported versions, 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.  [**CVE-2022-22970**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22970) - In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, applications that handle file uploads are vulnerable to DoS attack if they rely on data binding to set a MultipartFile or javax.servlet.Part to a field in a model object.  [**CVE-2021-22060**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22060) **-** 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. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase.  [**CVE-2021-22096**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-22096) **-** In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. | 11 – Critical |
| [tomcat-embed-core-9.0.30.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l20_ad32909314fe2ba02cec036434c0addd19bcc580) | [**CVE-2020-1938**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938) **-** 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.  [**CVE-2020-11996**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996) - A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive.  [**CVE-2020-13934**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934) - An h2c direct connection to Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M5 to 9.0.36 and 8.5.1 to 8.5.56 did not release the HTTP/1.1 processor after the upgrade to HTTP/2. If a sufficient number of such requests were made, an OutOfMemoryException could occur leading to a denial of service.  [**CVE-2020-13935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935) - The payload length in a WebSocket frame was not correctly validated in Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M1 to 9.0.36, 8.5.0 to 8.5.56 and 7.0.27 to 7.0.104. Invalid payload lengths could trigger an infinite loop. Multiple requests with invalid payload lengths could lead to a denial of service.  [**CVE-2020-17527**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527) **-** While investigating bug 64830 it was discovered that Apache Tomcat 10.0.0-M1 to 10.0.0-M9, 9.0.0-M1 to 9.0.39 and 8.5.0 to 8.5.59 could re-use an HTTP request header value from the previous stream received on an HTTP/2 connection for the request associated with the subsequent stream. While this would most likely lead to an error and the closure of the HTTP/2 connection, it is possible that information could leak between requests.  [**CVE-2021-25122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25122) **-** When responding to new h2c connection requests, Apache Tomcat versions 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41 and 8.5.0 to 8.5.61 could duplicate request headers and a limited amount of request body from one request to another meaning user A and user B could both see the results of user A's request.  [**CVE-2021-41079**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-41079) **-** Apache Tomcat 8.5.0 to 8.5.63, 9.0.0-M1 to 9.0.43 and 10.0.0-M1 to 10.0.2 did not properly validate incoming TLS packets. When Tomcat was configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS, a specially crafted packet could be used to trigger an infinite loop resulting in a denial of service.  [**CVE-2022-29885**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-29885) **-** The documentation of Apache Tomcat 10.1.0-M1 to 10.1.0-M14, 10.0.0-M1 to 10.0.20, 9.0.13 to 9.0.62 and 8.5.38 to 8.5.78 for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.  [**CVE-2022-42252**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42252) **-** If Apache Tomcat 8.5.0 to 8.5.82, 9.0.0-M1 to 9.0.67, 10.0.0-M1 to 10.0.26 or 10.1.0-M1 to 10.1.0 was configured to ignore invalid HTTP headers via setting rejectIllegalHeader to false (the default for 8.5.x only), Tomcat did not reject a request containing an invalid Content-Length header making a request smuggling attack possible if Tomcat was located behind a reverse proxy that also failed to reject the request with the invalid header.  [**CVE-2020-9484**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484) **-** When using Apache Tomcat versions 10.0.0-M1 to 10.0.0-M4, 9.0.0.M1 to 9.0.34, 8.5.0 to 8.5.54 and 7.0.0 to 7.0.103 if a) an attacker is able to control the contents and name of a file on the server; and b) the server is configured to use the PersistenceManager with a FileStore; and c) the PersistenceManager is configured with sessionAttributeValueClassNameFilter="null" (the default unless a SecurityManager is used) or a sufficiently lax filter to allow the attacker provided object to be deserialized; and d) the attacker knows the relative file path from the storage location used by FileStore to the file the attacker has control over; then, using a specifically crafted request, the attacker will be able to trigger remote code execution via deserialization of the file under their control. Note that all of conditions a) to d) must be true for the attack to succeed.  [**CVE-2021-25329**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25329) **-** The fix for CVE-2020-9484 was incomplete. When using Apache Tomcat 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41, 8.5.0 to 8.5.61 or 7.0.0. to 7.0.107 with a configuration edge case that was highly unlikely to be used, the Tomcat instance was still vulnerable to CVE-2020-9494. Note that both the previously published prerequisites for CVE-2020-9484 and the previously published mitigations for CVE-2020-9484 also apply to this issue.  [**CVE-2021-30640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-30640) **-** A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid user name and/or to bypass some of the protection provided by the LockOut Realm. This issue affects Apache Tomcat 10.0.0-M1 to 10.0.5; 9.0.0.M1 to 9.0.45; 8.5.0 to 8.5.65.  [**CVE-2022-34305**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-34305) **-** In Apache Tomcat 10.1.0-M1 to 10.1.0-M16, 10.0.0-M1 to 10.0.22, 9.0.30 to 9.0.64 and 8.5.50 to 8.5.81 the Form authentication example in the examples web application displayed user provided data without filtering, exposing a XSS vulnerability.  [**CVE-2021-24122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122) **-** When serving resources from a network location using the NTFS file system, Apache Tomcat versions 10.0.0-M1 to 10.0.0-M9, 9.0.0.M1 to 9.0.39, 8.5.0 to 8.5.59 and 7.0.0 to 7.0.106 were susceptible to JSP source code disclosure in some configurations. The root cause was the unexpected behaviour of the JRE API File.getCanonicalPath() which in turn was caused by the inconsistent behaviour of the Windows API (FindFirstFileW) in some circumstances.  [**CVE-2021-33037**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-33037) **-** Apache Tomcat 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46 and 8.5.0 to 8.5.66 did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. Specifically: - Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response; - Tomcat honoured the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.  [**CVE-2019-17569**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569) **-** The refactoring present in Apache Tomcat 9.0.28 to 9.0.30, 8.5.48 to 8.5.50 and 7.0.98 to 7.0.99 introduced a regression. The result of the regression was that invalid Transfer-Encoding headers were incorrectly processed leading to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely.  [**CVE-2020-1935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935) **-** In Apache Tomcat 9.0.0.M1 to 9.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99 the HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. This led to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely.  [**CVE-2020-13943**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13943) - If an HTTP/2 client connecting to Apache Tomcat 10.0.0-M1 to 10.0.0-M7, 9.0.0.M1 to 9.0.37 or 8.5.0 to 8.5.57 exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. This could lead to users seeing responses for unexpected resources.  [**CVE-2023-28708**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-28708) **-** When using the RemoteIpFilter with requests received from a reverse proxy via HTTP that include the X-Forwarded-Proto header set to https, session cookies created by Apache Tomcat 11.0.0-M1 to 11.0.0.-M2, 10.1.0-M1 to 10.1.5, 9.0.0-M1 to 9.0.71 and 8.5.0 to 8.5.85 did not include the secure attribute. This could result in the user agent transmitting the session cookie over an insecure channel.  [**CVE-2021-43980**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-43980) **-** The simplified implementation of blocking reads and writes introduced in Tomcat 10 and back-ported to Tomcat 9.0.47 onwards exposed a long standing (but extremely hard to trigger) concurrency bug in Apache Tomcat 10.1.0 to 10.1.0-M12, 10.0.0-M1 to 10.0.18, 9.0.0-M1 to 9.0.60 and 8.5.0 to 8.5.77 that could cause client connections to share an Http11Processor instance resulting in responses, or part responses, to be received by the wrong client. | 20 – Critical |
| [tomcat-embed-el-9.0.30.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l21_406e0c7cc45204c0f81853d73e8cfedfa4e00945) | Core Tomcat implementation  Dependency without reported vulnerability. | 0 |
| [tomcat-embed-websocket-9.0.30.jar](file:///C:\Users\c20wr\Documents\ProjectOneCS305\rest-service\target\dependency-check-report.html#l22_33157f6bc5bfd03380ebb5ac476db0600a04168d) | [**CVE-2020-1938**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938) **-** 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.  [**CVE-2020-8022**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-8022)  - A Incorrect Default Permissions vulnerability in the packaging of tomcat on SUSE Enterprise Storage 5, SUSE Linux Enterprise Server 12-SP2-BCL, SUSE Linux Enterprise Server 12-SP2-LTSS, SUSE Linux Enterprise Server 12-SP3-BCL, SUSE Linux Enterprise Server 12-SP3-LTSS, SUSE Linux Enterprise Server 12-SP4, SUSE Linux Enterprise Server 12-SP5, SUSE Linux Enterprise Server 15-LTSS, SUSE Linux Enterprise Server for SAP 12-SP2, SUSE Linux Enterprise Server for SAP 12-SP3, SUSE Linux Enterprise Server for SAP 15, SUSE OpenStack Cloud 7, SUSE OpenStack Cloud 8, SUSE OpenStack Cloud Crowbar 8 allows local attackers to escalate from group tomcat to root. This issue affects: SUSE Enterprise Storage 5 tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server 12-SP2-BCL tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server 12-SP2-LTSS tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server 12-SP3-BCL tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server 12-SP3-LTSS tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server 12-SP4 tomcat versions prior to 9.0.35-3.39.1. SUSE Linux Enterprise Server 12-SP5 tomcat versions prior to 9.0.35-3.39.1. SUSE Linux Enterprise Server 15-LTSS tomcat versions prior to 9.0.35-3.57.3. SUSE Linux Enterprise Server for SAP 12-SP2 tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server for SAP 12-SP3 tomcat versions prior to 8.0.53-29.32.1. SUSE Linux Enterprise Server for SAP 15 tomcat versions prior to 9.0.35-3.57.3. SUSE OpenStack Cloud 7 tomcat versions prior to 8.0.53-29.32.1. SUSE OpenStack Cloud 8 tomcat versions prior to 8.0.53-29.32.1. SUSE OpenStack Cloud Crowbar 8 tomcat versions prior to 8.0.53-29.32.1.  [**CVE-2020-11996**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996) - A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive.  [**CVE-2020-13934**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934) - An h2c direct connection to Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M5 to 9.0.36 and 8.5.1 to 8.5.56 did not release the HTTP/1.1 processor after the upgrade to HTTP/2. If a sufficient number of such requests were made, an OutOfMemoryException could occur leading to a denial of service.  [**CVE-2020-13935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935) - The payload length in a WebSocket frame was not correctly validated in Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M1 to 9.0.36, 8.5.0 to 8.5.56 and 7.0.27 to 7.0.104. Invalid payload lengths could trigger an infinite loop. Multiple requests with invalid payload lengths could lead to a denial of service.  [**CVE-2020-17527**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527) **-** While investigating bug 64830 it was discovered that Apache Tomcat 10.0.0-M1 to 10.0.0-M9, 9.0.0-M1 to 9.0.39 and 8.5.0 to 8.5.59 could re-use an HTTP request header value from the previous stream received on an HTTP/2 connection for the request associated with the subsequent stream. While this would most likely lead to an error and the closure of the HTTP/2 connection, it is possible that information could leak between requests.  [**CVE-2021-25122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25122) **-** When responding to new h2c connection requests, Apache Tomcat versions 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41 and 8.5.0 to 8.5.61 could duplicate request headers and a limited amount of request body from one request to another meaning user A and user B could both see the results of user A's request.  [**CVE-2021-41079**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-41079) **-** Apache Tomcat 8.5.0 to 8.5.63, 9.0.0-M1 to 9.0.43 and 10.0.0-M1 to 10.0.2 did not properly validate incoming TLS packets. When Tomcat was configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS, a specially crafted packet could be used to trigger an infinite loop resulting in a denial of service.  [**CVE-2022-29885**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-29885) **-** The documentation of Apache Tomcat 10.1.0-M1 to 10.1.0-M14, 10.0.0-M1 to 10.0.20, 9.0.13 to 9.0.62 and 8.5.38 to 8.5.78 for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.  [**CVE-2022-42252**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42252) **-** If Apache Tomcat 8.5.0 to 8.5.82, 9.0.0-M1 to 9.0.67, 10.0.0-M1 to 10.0.26 or 10.1.0-M1 to 10.1.0 was configured to ignore invalid HTTP headers via setting rejectIllegalHeader to false (the default for 8.5.x only), Tomcat did not reject a request containing an invalid Content-Length header making a request smuggling attack possible if Tomcat was located behind a reverse proxy that also failed to reject the request with the invalid header.  [**CVE-2020-9484**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484) **-** When using Apache Tomcat versions 10.0.0-M1 to 10.0.0-M4, 9.0.0.M1 to 9.0.34, 8.5.0 to 8.5.54 and 7.0.0 to 7.0.103 if a) an attacker is able to control the contents and name of a file on the server; and b) the server is configured to use the PersistenceManager with a FileStore; and c) the PersistenceManager is configured with sessionAttributeValueClassNameFilter="null" (the default unless a SecurityManager is used) or a sufficiently lax filter to allow the attacker provided object to be deserialized; and d) the attacker knows the relative file path from the storage location used by FileStore to the file the attacker has control over; then, using a specifically crafted request, the attacker will be able to trigger remote code execution via deserialization of the file under their control. Note that all of conditions a) to d) must be true for the attack to succeed.  [**CVE-2021-25329**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25329) **-** The fix for CVE-2020-9484 was incomplete. When using Apache Tomcat 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41, 8.5.0 to 8.5.61 or 7.0.0. to 7.0.107 with a configuration edge case that was highly unlikely to be used, the Tomcat instance was still vulnerable to CVE-2020-9494. Note that both the previously published prerequisites for CVE-2020-9484 and the previously published mitigations for CVE-2020-9484 also apply to this issue.  [**CVE-2021-30640**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-30640) **-** A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid user name and/or to bypass some of the protection provided by the LockOut Realm. This issue affects Apache Tomcat 10.0.0-M1 to 10.0.5; 9.0.0.M1 to 9.0.45; 8.5.0 to 8.5.65.  [**CVE-2022-34305**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-34305) **-** In Apache Tomcat 10.1.0-M1 to 10.1.0-M16, 10.0.0-M1 to 10.0.22, 9.0.30 to 9.0.64 and 8.5.50 to 8.5.81 the Form authentication example in the examples web application displayed user provided data without filtering, exposing a XSS vulnerability.  [**CVE-2021-24122**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122) **-** When serving resources from a network location using the NTFS file system, Apache Tomcat versions 10.0.0-M1 to 10.0.0-M9, 9.0.0.M1 to 9.0.39, 8.5.0 to 8.5.59 and 7.0.0 to 7.0.106 were susceptible to JSP source code disclosure in some configurations. The root cause was the unexpected behaviour of the JRE API File.getCanonicalPath() which in turn was caused by the inconsistent behaviour of the Windows API (FindFirstFileW) in some circumstances.  [**CVE-2021-33037**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-33037) **-** Apache Tomcat 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46 and 8.5.0 to 8.5.66 did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. Specifically: - Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response; - Tomcat honoured the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.  [**CVE-2019-17569**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569) **-** The refactoring present in Apache Tomcat 9.0.28 to 9.0.30, 8.5.48 to 8.5.50 and 7.0.98 to 7.0.99 introduced a regression. The result of the regression was that invalid Transfer-Encoding headers were incorrectly processed leading to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely.  [**CVE-2020-1935**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935) **-** In Apache Tomcat 9.0.0.M1 to 9.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99 the HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. This led to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely.  [**CVE-2020-13943**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13943) - If an HTTP/2 client connecting to Apache Tomcat 10.0.0-M1 to 10.0.0-M7, 9.0.0.M1 to 9.0.37 or 8.5.0 to 8.5.57 exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. This could lead to users seeing responses for unexpected resources.  [**CVE-2023-28708**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-28708) **-** When using the RemoteIpFilter with requests received from a reverse proxy via HTTP that include the X-Forwarded-Proto header set to https, session cookies created by Apache Tomcat 11.0.0-M1 to 11.0.0.-M2, 10.1.0-M1 to 10.1.5, 9.0.0-M1 to 9.0.71 and 8.5.0 to 8.5.85 did not include the secure attribute. This could result in the user agent transmitting the session cookie over an insecure channel.  [**CVE-2021-43980**](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-43980) **-** The simplified implementation of blocking reads and writes introduced in Tomcat 10 and back-ported to Tomcat 9.0.47 onwards exposed a long standing (but extremely hard to trigger) concurrency bug in Apache Tomcat 10.1.0 to 10.1.0-M12, 10.0.0-M1 to 10.0.18, 9.0.0-M1 to 9.0.60 and 8.5.0 to 8.5.77 that could cause client connections to share an Http11Processor instance resulting in responses, or part responses, to be received by the wrong client. | 21 - Critical |

## Mitigation Plan

As part of the process for modernizing the web application for Artemis Financial we will update all libraries listed in dependencies to ensure the latest, many of the vulnerabilities found in older versions were corrected in later versions. Once this is completed, we will re-run the dependency and vulnerability check. If there remains a library with an unacceptably long list of vulnerabilities we will research to determine if a better option exists. Correct all errors found in the code manual code review, once this step is complete code will undergo a second review to see if the new code is secure. Update all functionality as desired by Artemis Financial with regular testing, code reviews, and dependency checks. This way the security checks can find issues as the software is upgraded and work to mitigate the threats. All staff at Global Rain has received training concerning secure software engineering and will work together to ensure the updated application meets current industry standards.

References

Compiègne, T. (2021, November 25). *Rest API Security - Why it Matters to be Secure*. Gravitee.io - API Management Platform. https://www.gravitee.io/blog/rest-api-security-why-it-matters-to-be-secure

Detlefsen, J. Manico. A. (2014). *Iron-clad Java: Building secure web applications*. *Oreilly*. Oracle Press. Retrieved July 8, 2023, from https://learning.oreilly.com/library/view/iron-clad-java/9780071835886/.