****

# 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 5](#_Toc32574612)

[3. Manual Review 6](#_Toc32574613)

[4. Static Testing 6](#_Toc32574614)

[5. Mitigation Plan 22](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** |  |  | **Information provided for required areas.** |

## 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

Richard Howell

## Interpreting Client Needs

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

The value of secure communications to Artemis Financial is customer and consumer confidence as well as security and compliance with federal regulations. Artemis Financial provides financial services to its customers if the communications between the company and its customers are not secure there could be a loss of trust which may lead to customers using a competing financial institution. Secure communications are also extremely important due to federal regulations. If communications are not secured Artemis Financial may face fines and potential legal action which will also lead to a loss in consumer confidence. Artemis Financial also deals with large amounts of PII (personally identifiable information) such as names, bank accounts, social security numbers, date of birth, etc. The protection of this information is also federally regulated so secure communications are critical to consumer confidence in Artemis Financial.

* **Does the company make any international transactions?**

I believe that Artemis Financial does have some international transactions even if it is a U.S. based company that tailors itself to domestic clients. Artemis Financial is in the process of modernization and customers will be able to manage accounts, view accounts, and communicate with financial advisors from anywhere in the world. This makes the likelihood of international transactions very high.

* **Are there governmental restrictions about secure communications to consider?**

There are many government restrictions and regulations about secure communications that Artemis Financial will need to consider. A few examples are the Gramm-Leach-Bliley Act, The Sarbanes-Oxley Act, and The Payment Card Industry Data Security Standards to name a few. “The Gramm-Leach-Bliley Act requires financial institutions – companies that offer consumers financial products or services like loans, financial or investment advice, or insurance – to explain their information-sharing practices to their customers and to safeguard sensitive data” (Gramm-Leach-Bliley Act, 2022). “The Sarbanes-Oxley Act of 2002 is a federal law that established sweeping auditing and financial regulations for public companies. Lawmakers created the legislation to help protect shareholders, employees and the public from accounting errors and fraudulent financial practices” (Lutkevich, 2020). “The Payment Card Industry Data Security Standard (PCI DSS) is a set of security standards designed to ensure that ALL companies that accept, process, store or transmit credit card information maintain a secure environment” (PCI Compliance Guide, 2017).

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

The external threats that Artemis Financial is facing now and in the future are Phishing attacks, Ransomware attacks, SQL injections, Local File Inclusion, Cross-Site Scripting, OGNL Java Injections, DDoS attacks, Supply Chain attacks, and Bank Drops (*The 6 Biggest Cyber Threats for Financial Services in 2023 | UpGuard*, n.d.). All these attacks are current threats as well as possible threats that may develop in the future as the financial industry changes and modernizes. There are many more potential threats that have not materialized yet as well. Cyber-attacks are always changing, and we need to do our best to keep up with the potential threats.

* **What are the modernization requirements that you must consider? Such as open-source libraries and web application technologies.**

Modernization is extremely important for the protection of data and communications. Open-source libraries and platforms allow for the optimal and secure integration of new technologies and applications into the already existing infrastructure. The open-source libraries and web application technologies are constantly being developed and improved which enables better security and functionality practices. Open-source libraries and web application technologies allow for added layers of security to the application as well as providing better functionality to the end user.

## Areas of Security

* **API** -The web application used by Artemis Financial uses API calls and due to this we need to ensure that a secure configuration for the API is utilized. This will reduce the possibility of a potential security compromise between the system and the API connection. API security will also reduce the possibility of a security breach from outside threats and hackers. The API acts as a negotiator that allows two or more systems to communicate with each other as well as authenticating the user of the system with no third-party applications or programs.
* **Cryptography** -Cryptography is needed for the Artemis Financial system as well. This will protect all data being transmitted and accessed over the internet. Without proper cryptography Artemis Financial clients and users may have sensitive data, such as PII or financial information**,** become compromised due to middleman attacks, brute force attacks, cipher text attacks, etc. Overall cryptography will be used to ensure all data and information being passed through the system is safe and secure. Cryptography is also mandated by the FFIEC for all financial data stored by any financial institution.
* **Client/Server** – Client/Server refers to the communication between client (user) and server (system) through a web application. The client is communicating with the backend of the system through the front-end UI. Client/ Server also refers to the relationships the programs of the application have with the client or users and how they access the services provided by the application. The client/server needs to be secured from physical security of the servers and preventing unwanted access to the hardware of the system as well as the connection the client makes to the system. If this relationship is not properly secured the system may be left vulnerable to attacks.
* **Secure Coding** -Secure coding is necessary in Artemis Financials case to maintain the consistency of the business logic of the application. Secure coding is also extremely important to the development process because it is the one of the first steps in preventing attacks and vulnerabilities. This also prevents common defects in logic as well as developing code that prevents security vulnerabilities and defects. This is one of the most important aspects of the development process. Without secure code the entire system can be left open to attack.

## Manual Review

* **CRUDController.java** – There is a direct object reference vulnerability in the CRUDController.Java file where inside objects may be exposed. These inside objects are then retrieved and passed through a code injection sequence. The “business\_name” is passed through the CRUD method potentially exposing the “DocData” unwanted access and vulnerabilities.
* **DocData.Java** – There is a data access vulnerability in the DocData.Java file. The method that accesses the data uses the definition of the location of that database. Both the username and password are “root” which is not suggested and may lead to unauthorized access to the system by brute force attacks.
* **Greeting.Java** – In the Greeting.Java file the method get is used instead of post. Get is considered a less secure method than post because requests are sent in the URL section. Since Artemis Financial is dealing with sensitive data the post method should be used instead for extra security. Also, in the Greeting.Java file the input is not validated before being put into the string pattern. This allows for potentially dangerous data to be included in the string leaving the system open to attack.

## Graphical user interface, text, application Description automatically generatedStatic Testing

* **bcprov-jdk15on-1.46.jar**
  + **CVE-2016-1000338** - 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.
    - **Solutions** – upgrade to current Bouncy Castle version 1.70
  + **CVE-2016-1000342** - In the Bouncy Castle JCE Provider version 1.55 and earlier ECDSA 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.
    - **Solutions** – upgrade to current Bouncy Castle version 1.70
  + **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 initialized with DSA parameters, 1.55 and earlier generates a private value assuming a 1024-bit key size.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70 also can be dealt with by explicitly passing parameters to the key pair generator.
  + **CVE-2016-1000344** - 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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **CVE-2016-1000352** - In the Bouncy Castle JCE Provider version 1.55 and earlier the ECIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider.
    - **Solutions** – upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** – upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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."
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70. 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.
    - **Solutions** – Fix was introduced in version 1.61, but we should upgrade to the current Bouncy Castle version 1.70.
  + **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."
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions** - upgrade to current Bouncy Castle version 1.70
  + **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.
    - **Solutions -** upgrade to current Bouncy Castle version 1.70
* **hibernate-validator-6.0.18.Final.jar**
  + **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.
    - **Solutions** – upgrade hibernate.validator to version 8.0.0.Final
* **jackson-databind-2.10.2.jar**
  + **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.
    - **Solutions** – Update to Jackson-databind version 2.14.1
  + **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.
    - **Solutions** - This vulnerability was fixed in versions 2.13.2.1 and 2.12.6.1 and not 2.13.0 as stated in the advisory. We should Update to Jackson-databind version 2.14.1
  + **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
    - **Solutions** - Update to Jackson-databind version 2.14.1
  + **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.
    - **Solutions** - Update to Jackson-databind version 2.14.1
* **log4j-api-2.12.1.jar** 
  + **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.
    - **Solutions** –Update to either Apache Log4j 2.12.3 or 2.13.1
* **logback-core-1.2.3.jar** 
  + **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.
    - **Solutions** – update to logback version 1.4.5.
* **snakeyaml-1.25.jar**
  + **CVE-2022-1471 (OSSINDEX)** - 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.
    - **Solutions** – update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization.
  + **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.
    - **Solutions** - update to version 1.33 of SnakeYaml
  + **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.
    - **Solutions** - update to version 1.33 of SnakeYaml
  + **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.
    - **Solutions** - update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content.
  + **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.
    - **Solutions -** update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content.
  + **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.
    - **Solutions -** update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content.
  + **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.
    - **Solutions -** update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content.
  + **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.
    - **Solutions -** update to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content.
* **spring-boot-2.2.4.RELEASE.jar**
  + **CVE-2022-27772** - 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.
    - **Solutions** – update to version 3.0.1 of spring-boot
* **spring-boot-starter-web-2.2.4.RELEASE.jar** 
  + **CVE-2022-27772** - 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.
    - **Solutions** - update to version 3.0.1 of spring-boot-starter-web.
* **spring-core-5.2.3.RELEASE.jar**
  + **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.
    - **Solutions** – update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-core.
* **spring-web-5.2.3.RELEASE.jar**
  + **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.
    - **Solutions** – Do not use untrusted data as a use case. Also we should update to the new version of 6.0.4 spring-web.
  + **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.
    - **Solutions** – update to version 6.0.4 of spring-web and deploy application as Spring Boot executable jar and the exploit is not viable.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-web
* **spring-webmvc-5.2.3.RELEASE.jar**
  + **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.
    - **Solutions** – update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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 disallowed Fields 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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
  + **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.
    - **Solutions** - update to version 6.0.4 of spring-webmvc.
* **tomcat-embed-core-9.0.30.jar**
  + **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.
    - **Solutions** – upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions** - upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **CVE-2021-30640** - A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid username 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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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 example’s web application displayed user provided data without filtering, exposing a XSS vulnerability.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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 behavior of the JRE API File.getCanonicalPath() which in turn was caused by the inconsistent behavior of the Windows API (FindFirstFileW) in some circumstances.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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 honored the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
  + **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.
    - **Solutions -** upgrade to version 11.0.0-M1 of tomcat-embed-core.
* **tomcat-embed-websocket-9.0.30.jar**
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **CVE-2021-30640** - A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid username 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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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 example’s web application displayed user provided data without filtering, exposing a XSS vulnerability.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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 honored the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - **Solutions** – Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.
  + **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.
    - Upgrade to version 11.0.0-M1 of tomcat-embed-websocket.

## Mitigation Plan

* **CRUDController.java Direct object reference venerability** – “Preventing insecure direct object references requires selecting an approach for protecting each user accessible object (e.g., object number, filename). Use per user or session indirect object references. This prevents attackers from directly targeting unauthorized resources.” (Insecure Direct Object References - Enterprise Security | Montana State University, n.d.). Each use of the direct object reference from an untrusted source needs to have an access control check. This will ensure that the user is authorized to view the requested object.
* **DocData.Java data access vulnerability** – The password and username should have a strict whitelisted set of usable characters with a set character length. The password should also not be allowed to contain the username. The database should also not allow the password or username to be “root”. This will help prevent brute force attacks from gaining access to the Artemis Financial system.
* **Greeting.Java Post vulnerability** – The “get” method is used instead of “post”. “Get” is considered a less secure method than “post” because requests are sent in the URL section. The “Post” method is considered more secure and should be used instead of “get”. This will help prevent unwanted injections into the URL. All input in the Greeting.Java file needs to have input validation before being placed into the string pattern. This will prevent unwanted potentially malicious data from being included in the string.
* **bcprov-jdk15on-1.46.jar vulnerabilities** – All the provided CVE codes found during static testing for bcprov-jdk15on-1.46.jar can be rectified by upgrading to the current Bouncy Castle version 1.70. This version of bouncy castle has fixed all the previous CVE codes and has no current known vulnerabilities.
* **hibernate-validator-6.0.18.Final.jar vulnerabilities** – All the provided CVE codes found during static testing for hibernate-validator-6.0.18.Final.jar can be fixed by updating to version 8.0.0. of Final hibernate.validator.
* **jackson-databind-2.10.2.jar vulnerabilities** – All the provided CVE codes found during static testing for jackson-databind-2.10.2.jar can be fixed by updating to version 2.14.1 of Jackson-databind.
* **log4j-api-2.12.1.jar vulnerabilities** - All the provided CVE codes found during static testing for log4j-api-2.12.1.jar can be fixed by updating to version 2.12.3 or 2.13.1 of Apache Log4j.
* **logback-core-1.2.3.jar vulnerabilities** - All the provided CVE codes found during static testing for logback-core-1.2.3.jar can be fixed by updating to version 1.4.5. of logback.
* **snakeyaml-1.25.jar vulnerabilities** – All the provided CVE codes found during static testing for snakeyaml-1.25.jar can be fixed by updating to version 1.33 of SnakeYaml and using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization.
* **spring-boot-2.2.4.RELEASE.jar vulnerabilities** - All the provided CVE codes found during static testing for spring-boot-2.2.4.RELEASE.jar can be fixed by updating to version 3.0.1 of spring-boot.
* **spring-boot-starter-web-2.2.4.RELEASE.jar vulnerabilities** - All the provided CVE codes found during static testing for spring-boot-starter-web-2.2.4.RELEASE.jar can be fixed by updating to version 3.0.1 of spring-boot-starter-web.
* **spring-core-5.2.3.RELEASE.jar vulnerabilities** - All the provided CVE codes found during static testing for spring-core-5.2.3.RELEASE.jar can be fixed by updating to version 6.0.4 of spring-core.
* **spring-web-5.2.3.RELEASE.jar vulnerabilities** - All the provided CVE codes found during static testing for spring-web-5.2.3.RELEASE.jarcan be fixed by updating to version 6.0.4 of spring-web as well as not using untrusted data as a use case and deploying the application as Spring Boot executable jar.
* **spring-webmvc-5.2.3.RELEASE.jar vulnerabilities** - All the provided CVE codes found during static testing for spring-webmvc-5.2.3.RELEASE.jar can be fixed by updating to version 6.0.4 of spring-webmvc.
* **tomcat-embed-core-9.0.30.jar vulnerabilities** - All the provided CVE codes found during static testing for tomcat-embed-core-9.0.30.jar can be fixed by updating to version 11.0.0-M1 of tomcat-embed-core.
* **tomcat-embed-websocket-9.0.30.jar vulnerabilities** - All the provided CVE codes found during static testing for tomcat-embed-websocket-9.0.30.jar can be fixed by updating to version 11.0.0-M1 of tomcat-embed-websocket.

**Citations**

*Gramm-Leach-Bliley Act*. (2022, February 11). Federal Trade Commission.

https://www.ftc.gov/business-guidance/privacy-security/gramm-leach-bliley-act

*Insecure Direct Object References - Enterprise Security | Montana State University*. (n.d.).

https://www.montana.edu/uit/security/web/insecure-direct-object-references.html

Lutkevich, B. (2020, December 11). *Sarbanes-Oxley Act*. CIO.

https://www.techtarget.com/searchcio/definition/Sarbanes-Oxley-Act

PCI Compliance Guide. (2017, September 5). *PCI FAQs*. https://www.pcicomplianceguide.org/faq/

*The 6 Biggest Cyber Threats for Financial Services in 2023 | UpGuard*. (n.d.).

https://www.upguard.com/blog/biggest-cyber-threats-for-financial-services