****

Artemis Financial Vulnerability Assessment Report

CS 305 Project One

Table of Contents

[Document Revision History 4](#_Toc108973310)

[Client 4](#_Toc108973311)

[Developer 5](#_Toc108973312)

[1. Interpreting Client Needs 5](#_Toc108973313)

[2. Areas of Security 5](#_Toc108973314)

[3. Manual Review 6](#_Toc108973315)

[Views 6](#_Toc108973316)

[Models 6](#_Toc108973317)

[CRUD 6](#_Toc108973318)

[Customer 7](#_Toc108973319)

[DocData 7](#_Toc108973320)

[Greeting 7](#_Toc108973321)

[MyDateTime 7](#_Toc108973322)

[Controllers 8](#_Toc108973323)

[CRUD Controller 8](#_Toc108973324)

[Greeting Controller 9](#_Toc108973325)

[Data Access 10](#_Toc108973326)

[Services 10](#_Toc108973327)

[Plugins 10](#_Toc108973328)

[APIs 10](#_Toc108973329)

[4. Static Testing 10](#_Toc108973330)

[5. Mitigation Plan 19](#_Toc108973331)

[APIs 19](#_Toc108973332)

[Client/Server 20](#_Toc108973333)

[Code Error 20](#_Toc108973334)

[Code Quality 20](#_Toc108973335)

[Data Access 20](#_Toc108973336)

[Encapsulation 21](#_Toc108973337)

[Input Validation 21](#_Toc108973338)

[References 21](#_Toc108973339)

# Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **7/16/2022** | **Cory Remick** | **Initial Version** |

# Client



# Developer

Cory Remick

# 1. Interpreting Client Needs

The client needs to use encrypted communications to prevent eavesdropping and stealing the client's account information and money. Governments have various regulations regarding the strength of encryption that can be used. The client faces threats from nation-states gathering intelligence, thieves, hacktivists, and pranksters. The use of open-source libraries presents a significant challenge. They require to be upgraded regularly as security vulnerabilities are found and fixed. But they must also be scrutinized, as faithless actors can inject code into the projects that compromise systems.

# 2. Areas of Security

The security assessment will address the following security concerns.

* Input Validation

Parameters will be passed to REST endpoints, and input into those methods should be validated before use to prevent injections.

* APIs

The project uses multiple frameworks and code libraries (application programming interfaces, APIs). These frameworks will be passed data derived from untrusted sources. These APIs must be reviewed to ensure no known vulnerabilities are in them. If there are, then steps must be taken to mitigate the issues.

* Cryptography

The transmission of user session data should be encrypted to prevent eavesdropping and interception of information. However, the project is not yet in a deployable state. Therefore, it is still acceptable not to have encryption in a development build. Thus, a cryptography review does not need to apply this early in the process but should be included in a future review. The review should focus on whether cryptography is appropriately used and whether the strength of the encryption is reasonable given current computing capabilities.

* Client/Server

A web client calls a REST service endpoint. Therefore, secure client/server composing is required. User identity and access control should ensure that only authorized users can retrieve and update information.

In addition, there is code in the project to call a MySQL database. This composition of services must also be evaluated.

* Code Error

Errors must be anticipated and handled. The code should be fail-safe, not disclose information, or allow unauthorized actions.

* Code Quality

The project is a Java application and should conform to Oracle's Secure Coding guidelines (Oracle, 2022).

* Encapsulation

Data structures should not allow internal or private data manipulation except by public methods. The project uses several data structures, so this area needs to be evaluated.

# 3. Manual Review

This assessment will now evaluate the project by applying the seven areas above to each part of the project below.

## Views

Views are outputs of information for display to a consumer. A typical example would be an HTML page. The concern is that if untrusted data is returned from the method, it could be used to compromise the client or set up the client for an attack on another part of the REST API. In the case of a REST endpoint, the views are the return data represented by JavaScript Object Notation (JSON).

Care must be taken not to return any information that can be used to compromise the privacy or integrity of the system.

In the REST controller, the view is a Java data model returned from a Java method as JSON over HTTP (Spring, n.d.). Since the project relies on the Spring Framework to encode the data as JSON, we will apply the API security area to the Spring Framework and the input validation security area to the REST controllers. Findings, if any, will be reported in subsequent sections of this document.

## Models

Models are classes that represent data in memory. There is a danger that an attacker can try to leverage untrusted data to manipulate the data in these models. To protect these models, the Object-Orientated concept of encapsulation is used to prevent changing the object's internal data except through public methods. This gives the programmer control over the input data, allows for checking input values, and changes the data in a consistent and controlled manner.

### CRUD

The "CRUD" class is found in the "CRUD.java" file. The class was evaluated using the encapsulation security area. The evaluation found the internal data is private and marked as unchangeable after creation. No mutators are trying to change the values after creation. There are two accessor methods to return the internal data without manipulation. There are no findings.

### Customer

The "customer" class is found in the "customer.java" file. The class was evaluated using the encapsulation security area. There are several problems with this class.

1. The "account\_balance" field is not private. It can be changed by external code.
2. The "account\_number" is not marked as "final, " meaning it can be changed after the object is created.
3. There is no constructor to set or initialize the internal fields.
4. The "showInfo" method returns the whole account number. This could be a violation of regulations against disclosing customer data.
5. The "account\_balance" appears to use an incorrect data type. Money should be represented as a decimal type. Instead, the class uses an integer type limited to just over 2.1 billion (Oracle, n.d.). This is a problem not just for representing cents; any customer with accounts larger than 2.1 billion will have problems. This is a likely scenario given the list of billionaires worldwide (Mille, 2022).
6. The "deposit" method does not check if the input parameter is a positive value. If an attacker can submit a negative value, they can decrease the account balance, which is a withdrawal. Standard accounting practices are to split account changes between separate "debit" and "credit" transactions that are logged to a transaction register (Investopedia, 2022). Debits and credits are always positive values. To calculate the balance, the transactions are applied one by one to a starting balance, either increasing or decreasing the balance until the last transaction is reached and an ending balance is calculated (Investopedia, 2020).
7. The "deposit" method does not check what happens if the integer data type max value is reached. Java will continue silently by taking the twos-complement of the addition which will make the number extremely negative (Oracle, n.d.).

### DocData

The "DocData" class is found in the "DocData.java" file. The class was evaluated using the encapsulation security area. There are several findings in this file.

1. The "id" field is not marked final, which means it is changeable after the class is created.
2. The constructor does not initialize the "id" field.
3. The "getId" method returns the uninitialized "id" field.
4. See the section on data access for further findings.

### Greeting

The class "Greeting" was found in the "Greeting.java" file. The class was evaluated using the encapsulation security area. The data types were consistent between parameters and internal storage. The internal fields are not accessible except through public methods. The fields are marked "final, " meaning they cannot change after being set by the constructor. There are no findings.

### MyDateTime

The "myDateTime" class was found in the "myDateTime.java" file. The class was evaluated using the encapsulation security area. There are several findings.

1. The fields are not marked "private". The values are accessible by external code.
2. There are no constructors for initializing the fields.
3. The "retrieveDataTime" method is not marked public. It cannot be called by external code. Internal fields must be wrapped by public methods for encapsulation.
4. The "retrieveDateTime" method returns an array without setting its values from the internal fields.
5. The "setMyDateTime" method is not marked public. It cannot be called by external code. Internal fields must be wrapped by public methods for encapsulation.
6. The "setMyDateTime" method is not implemented. The parameters never change the internal fields.

## Controllers

### CRUD Controller

A REST controller named "CRUDController" was found in the file "CRUDController.java". The controller was evaluated with input validation, client/server, code errors, and code quality security areas.

**Input Validation**

The "CRUD" method accepts a string parameter called "name." This parameter is not used currently but is also not checked. An attacker could try to cause a denial of service by sending an extremely large string value that will consume too much memory.

**Client/Server**

None of the REST methods have access control rules applied to them. Even in development, a developer should start with authentication and authorization.

**Code Error**

None of the methods implement exception handling. Structured exception handling will enable the REST endpoint to return meaningful HTTP error codes.

At best, the client will get a generic internal server error message from the framework when anything goes wrong.

The application entry point in the Application.java method "main" does not have structured error handling. There should be one added here to catch and log unhandled exceptions.

**Code Quality**

There are no findings here but refer to CRUD and DocData models for further findings.

### Greeting Controller

A controller named "GreetingController" was found in the file GreetingController.java. The controller was evaluated with input validation, client/server, code errors, and code quality security areas.

**Input Validation**

The "greeting" method accepts a string parameter called "name." This parameter is used without checking to see if it is valid. In addition, there is no check to see if its size is so large that it will create a denial-of-service by consuming too much memory.

**Client/Server**

None of the REST methods have access control rules applied to them. Even in development, a developer should start with authentication and authorization.

**Code Error**

None of the methods implement exception handling. Structured exception handling will enable the REST endpoint to return meaningful HTTP error codes.

At best, the client will get a generic internal server error message from the framework when anything goes wrong—also, logging to "Standard Out" will likely never occur if the error happens before the logging statement.

The application entry point in the Application.java method "main" does not have structured error handling. There should be one added here to catch and log unhandled exceptions.

**Code Quality**

Code is evaluated from [Secure Coding Guidelines for Java SE (oracle.com)](https://www.oracle.com/java/technologies/javase/seccodeguide.html). If another security area already covers a guideline, the guideline is not included below.

Guideline 1-3/DOS-3

There is no limit check on the counter AtomicLong and no defined way to handle what happens when the max value is reached. However, according to the documentation for integer types, overflows silently wrap around to the min value (Oracle, n.d.). There is no known problem with using a negative identifier, so there is no finding.

Guideline 3-1/INJECT-1

The "greeting" method uses "String.format" with an unchecked parameter using a template string. An attacker could use this to inject code into the application with a carefully constructed string.

## Data Access

Calls for MySQL were found in the "read\_document" method of the "DocData.java" file. The evaluation uses the client/server security area.

Hardcoded login information is being used. This presents a problem with storing code in source control as it will be stored in plaintext. This is even more problematic for cloud source control providers like GitHub, which can be public.

The database login information in "read\_document" method uses the admin user "root". This gives the code full access to all server settings, databases, data definitions, and write access to data. The connection should be made with a limited user with only enough permissions to perform the function required by the application. At a minimum, the user should not be able to access other databases, modify data structures, or read/write data outside the needs of the REST API.

Data access should be encapsulated in another class. This will decouple the connection details from the rest of the project and create a security boundary between services.

## Services

Only the MySQL service was discovered. See the Data Access section.

## Plugins

No extensibility system was found.

## APIs

See the static testing analysis in section 4.

# 4. Static Testing

|  |  |  |
| --- | --- | --- |
| **Dependency** | **CVE** | **Description** |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000338**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000342**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000343**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000344**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000352**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000341**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000345**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2017-13098**](http://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." |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2020-15522**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | **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 |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000339**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | **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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2015-7940**](http://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." |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2018-5382**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2013-1624**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | [**CVE-2016-1000346**](http://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. |
| [bcprov-jdk15on-1.46.jar](https://ossindex.sonatype.org/component/pkg:maven/org.bouncycastle/bcprov-jdk15on) | **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. |
| [hibernate-validator-6.0.18.Final.jar](https://ossindex.sonatype.org/component/pkg:maven/org.hibernate.validator/hibernate-validator) | [**CVE-2020-10693**](http://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. |
| [jackson-databind-2.10.2.jar](https://ossindex.sonatype.org/component/pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.10.2?utm_source=dependency-check&utm_medium=integration&utm_content=7.1.1) | [**CVE-2020-25649**](http://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. |
| [jackson-databind-2.10.2.jar](https://ossindex.sonatype.org/component/pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.10.2?utm_source=dependency-check&utm_medium=integration&utm_content=7.1.1) | [**CVE-2020-36518**](http://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. |
| [jakarta.annotation-api-1.3.5.jar](https://ossindex.sonatype.org/component/pkg:maven/jakarta.annotation/jakarta.annotation-api@1.3.5?utm_source=dependency-check&utm_medium=integration&utm_content=7.1.1) | [**CVE-2022-31569**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-31569) | The RipudamanKaushikDal/projects repository through 2022-04-03 on GitHub allows absolute path traversal because the Flask send\_file function is used unsafely. |
| [log4j-api-2.12.1.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.logging.log4j/log4j-api) | [**CVE-2020-9488**](http://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 |
| [logback-core-1.2.3.jar](https://ossindex.sonatype.org/component/pkg:maven/ch.qos.logback/logback-core) | [**CVE-2021-42550**](http://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. |
| [snakeyaml-1.25.jar](https://ossindex.sonatype.org/component/pkg:maven/org.yaml/snakeyaml) | [**CVE-2017-18640**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640) | The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564. |
| [spring-boot-2.2.4.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework.boot/spring-boot) | [**CVE-2022-27772**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2022-22965**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2021-22118**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2020-5421**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2022-22950**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950) | 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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2022-22971**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2022-22968**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2022-22970**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2021-22060**](http://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. |
| [spring-core-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core) | [**CVE-2021-22096**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2016-1000027**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2022-22965**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2021-22118**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2020-5421**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2022-22950**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-22950) | 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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2022-22971**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2022-22968**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2022-22970**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2021-22060**](http://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. |
| [spring-web-5.2.3.RELEASE.jar](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-web) | [**CVE-2021-22096**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-1938**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-11996**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-13934**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-13935**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-17527**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-25122**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-41079**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2022-29885**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-9484**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-25329**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-30640**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2022-34305**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-24122**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2021-33037**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2019-17569**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-1935**](http://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. |
| [tomcat-embed-core-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core) | [**CVE-2020-13943**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-1938**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-8022**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-11996**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-13934**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-13935**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-17527**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-25122**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-41079**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2022-29885**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-9484**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-25329**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-30640**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2022-34305**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-24122**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2021-33037**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2019-17569**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-1935**](http://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. |
| [tomcat-embed-websocket-9.0.30.jar](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket) | [**CVE-2020-13943**](http://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. |

# 5. Mitigation Plan

The following steps should be taken to mitigate the findings.

## APIs

The application depends on the Java Development Kit 1.8, Apache Tomcat 9.0.30, the Spring Framework 5.2.3, the Spring Boot API 2.2.4, and several more libraries. These dependencies should be upgraded to versions with no known vulnerabilities and updated regularly.

The application relies on the Java Development Kit (JDK) 1.8 platform. There are too many CVEs reported against JDK 1.8 to list all here. Many of them have been fixed by subsequent builds. During this evaluation, it is impossible to know which build will be used to deploy the application. The recommendation is to ensure that developers and installers use the latest build of the JDK to eliminate as many of the CVEs as possible. Once a deployment build is selected, another report can be done against that specific build to find any CVEs that still apply, and mitigation strategies can be recommended.

Apache Tomcat contains both the core and WebSocket modules. Recommend upgrading to the latest version, 9.0.64, as it is compatible with the current version.

The Spring Framework includes sub-components for "core," "expression," and "web" project dependencies. The latest version is 5.3.21. The recommendation is to upgrade to the newest version. However, [CVE-2016-1000027](https://nvd.nist.gov/vuln/detail/CVE-2016-1000027) is still an issue, and the vendor will not fix it. A code review is recommended to determine if the vulnerability exists and, if so, add guards against using untrusted data.

Spring Framework Boot is an additional module to the Spring Framework. The l[atest version is 2.7.1. No vulnerabilities have been reported since 2.3.5](https://ossindex.sonatype.org/component/pkg:maven/org.springframework.boot/spring-boot). Open-source ([OSS) support for 2.6 ends 11/24/2022. OSS Support for 2.7 ends 11/18/23](https://spring.io/projects/spring-boot#support). The recommendation is to upgrade to 2.7.1 to eliminate known vulnerabilities and maintain support.

Bouncy Castle should be upgraded to 1.70.

Unless the REST platform is served from an Android operating system, CVEs from the "OSSINDEX" should be suppressed. There are currently three in the list above.

Hibernate Validator should be upgraded to 6.1.5.Final+. The latest version is "7.0.4.Final."

Jackson Databind should be upgraded to the latest version 2.13.3.

Jakarta Annotation should be re-evaluated every month. The current CVE is not yet confirmed as a problem and could be a false positive.

Log4j should be upgraded to 2.12.3+. The latest version is 2.18.0.

Logback should be upgraded to 1.2.11.

Snake YAML should be upgraded to 1.26+. The latest version is 1.30.

## Client/Server

The application presents its API via the REST interface in two controllers, "GreetingController" and "CRUDController." Boundary entry points should be secured against unauthorized access ([Guideline 0-4/Fundamentals-4](https://www.oracle.com/java/technologies/javase/seccodeguide.html)). User authentication and user authorization should be added to the REST API.

Data Access should be isolated from the rest of the project to protect the login information to MySQL.

## Code Error

The REST controllers need structured exception handling around the REST methods. Any exception in the methods will bubble up to the application entry point. Because there is no exception handling at the entry point, this will force the application to exit prematurely, causing a denial of service. Structured error handling in the REST methods will ensure that proper return codes and messages are passed back to the client.

The application entry point should have structured error handling to catch unhandled exceptions and log them before exiting. Any unhandled exception in the application or API could lead to a premature exit and a denial of service. Logging unhandled exceptions are essential to understanding what happened and what to do to ensure it does not happen again.

Exceptions should be logged to pinpoint and fix problems.

## Code Quality

The "greeting" method of the "GreetingController" class uses unchecked input when using the "String.format" method. The input should be checked before use. Also, the input can be escaped or sanitized before use depending on the output format, HTML sanitization, XML sanitization, Database quotation escapes, etc.

## Data Access

We recommend removing the hardcoded login information and securing it in an encrypted configuration file. We also recommend changing the database user to a non-admin account with minimal permissions to perform the task.

## Encapsulation

The "customer" class needs some changes. The "account\_balance" field should be private, so it cannot be directly accessed by external code. The "account\_number" field should be marked final to prevent changes after object creation. A constructor is needed to initialize values at object creation. The "showInfo" method should only return part of the account number to avoid information disclosure. Change the data type of the "account\_balance" field from int to BigDecimal, which can handle currency. Check the parameter for the "deposit" method to ensure that it is positive and the add operation will not overflow the data type.

The "DocData" class needs to mark its "id" field as "final" to prevent it from being changed after creation. The constructor should initialize the "id" field.

The "myDateTime" class needs to have its fields marked "private" to prevent access by external code. A constructor should be added to initialize the internal fields. The methods should be marked public. The "retrieveDateTime" method needs to copy the internal values to the array returned from the method. Lastly, the "setMyDateTime" method is not implemented and should check that the input values represent valid values.

## Input Validation

Input validation is needed for the REST controller "GreetingController." In the method "greeting," the "name" parameter's length should be limited to prevent denial-of-service attacks against memory. In addition, consider checking the parameter for a set number of words. It might not be considered a name if more than three words exist. Also, consider checking the input for value for any data that might fall outside the expected character set specifications (Unicode, ASCII, expected language code page).

# References

Investopedia. (2020, Nov. 6th). *Ledger Balance*. Retrieved July 17th, 2022, from Investopedia: https://www.investopedia.com/terms/l/ledger-balance.asp

Investopedia. (2022, April 22nd). *How to Calculate Credit and Debit Balances in a General Ledger*. Retrieved July 17th, 2022, from Investopedia.com: https://www.investopedia.com/ask/answers/030915/how-do-you-calculate-credits-and-debits-general-ledger.asp#:~:text=For%20a%20general%20ledger%20to%20be%20balanced%2C%20credits,revenue%2C%20and%20equity%20accounts%2C%20while%20debits%20decrease%20them.

Mille, R. (2022, March 11th). *World's Billionaires List*. Retrieved July 17th, 2022, from forbes.com: https://www.forbes.com/billionaires/

Oracle. (2022, Jan. 22nd). *Secure Coding Guidelines for Java SE*. Retrieved July 17th, 2022, from oracle.com: https://www.oracle.com/java/technologies/javase/seccodeguide.html

Oracle. (n.d.). *Java Language Specification Chapter 15. Expressions*. Retrieved July 17th, 2022, from oracle.com: https://docs.oracle.com/javase/specs/jls/se7/html/jls-15.html#jls-15.18.2

Oracle. (n.d.). *Java Language Specification Chapter 4. Types, Values, and Variables*. Retrieved July 17th, 2022, from oracle.com: https://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jls-4.2.2

Spring. (n.d.). *Building REST services with Spring*. Retrieved July 17th, 2022, from spring.io: https://spring.io/guides/tutorials/rest/