# ***The information contained within this report is Sonatype, Inc.’s confidential and proprietary information and viewers may not (i) disclose this report (or any part thereof) to any third party (including but not limited to any competitor of Sonatype) and/or (ii) use this report for any purpose other than evaluating Sonatype’s software for your organization’s internal use.***

Sonatype / Black Duck Data Comparison

This report is to review the component and CVE findings and discrepancies between Sonatype and Black Duck. The goal of this report is to determine the level of accuracy between each of the tools. In the below sections we review the components, CVEs, and why the findings differ.

# Overview

Sonatype and Black Duck scanned the same application(s) and provided the list of components and CVEs for each (their respective SBOMs). There were immediately some significant differences between data found from both vendors regarding CVEs and components.

| **Vendor** | **CVEs Found** | **Components Found** |
| --- | --- | --- |
| Sonatype | 413 (238 Unique) | 1,179 |
| Black Duck | 2,849 (404 Unique) | 1,017 |
| Matching Findings | 139 Unique | 837 |

We did not examine *every single* CVE found by Black Duck because of the large number of findings. However, we will explain in as much detail as possible why the findings differ, and identify patterns which lead to the inaccuracies.

| *Note:*  *Component identification is proven in Sonatype Lifecycle by using the* ***Occurrences*** *view on the component details page in the application reports. The occurrences view shows the location in the scan directory to every single instance of each component and users can use this information to confirm the presence of the artifact/component file.* |
| --- |

Here is an overview of the False Positive (FPs) and False Negative (FNs) findings for the application:

| **Vendor** | **CVE FPs** | **CVE FNs** | **Component FPs** | **Component FNs** |
| --- | --- | --- | --- | --- |
| **Sonatype** | **0** | **0** | **0** | **0** |
| **Black Duck** | **940+** | **93+** | **168+** | **118+** |

*Below, we go into detail on how we came to these results.*

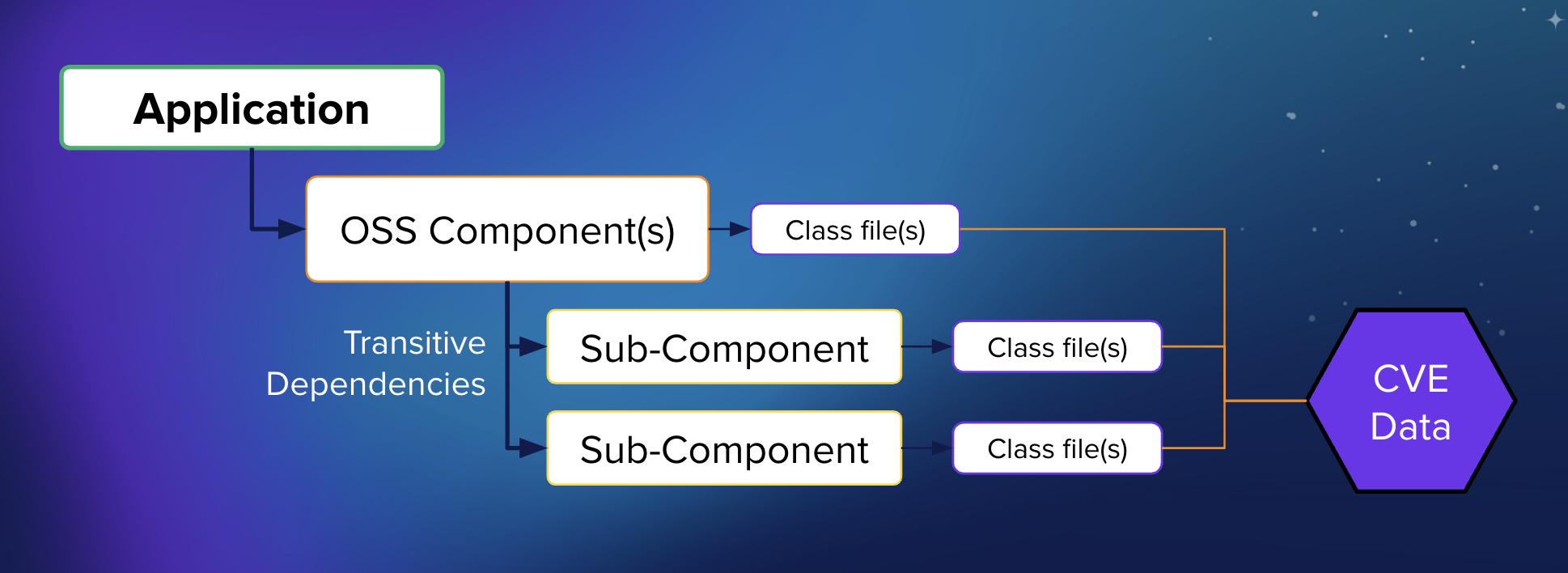
## Primary Causes of Data Inaccuracies

Manifest scanning, name-based matching, fuzzy matching, non-curated CVE results, CVEs generically attributed to component bundles, these are some of the most significant reasons we see inaccuracies in data from competitors' tools.

With Sonatype’s Advanced Binary Fingerprinting (ABF) scanning, Sonatype examines binary fingerprints (similar to a truncated sha1 hash) of all of the files and not just the file names and manifests. ABF is highly accurate because it examines everything included in the application after the build, including any embedded dependencies. Sonatype data is tied to the component fingerprints of any files where the vulnerability is discovered.

| **Sonatype**   * Precise Component Identification (ABF + Manifest) * Fast-Track Research (AI/ML, Pattern, Hash Based) * Deep Dive Research (60+ Human Security Researchers) * Policy Based Remediation Intelligence * Customer Success / Support Team | **Other Vendors**   * Manifest scanning, name-based matching, fuzzy/partial string identification * Reliance on public data (NVD) * No detailed review of data |
| --- | --- |

Sonatype component vulnerability attribution happens at the class level:



The Black Duck scan highlighted the advantage that ABF provides as Black Duck’s scan ended up repeating CVEs for components that were not affected but simply present in the bundle with the affected components. In addition it missed a number of components entirely.

# App 1 (Olv24)

## CVE False Positives

There were some massive discrepancies in the identification of CVEs in the reports:

* Sonatype: Found **158**CVEs not identified by Black Duck
* Black Duck: Found **1,744** CVEs not identified by Sonatype

***So… What happened?***

Well, the way that Black Duck apparently associates CVEs with components, is that they will see the bundle identifier, for instance (e.g. org.apache.camel, org.eclipse.jetty, org.apache.activemq, org.springframework, etc.) and read in the CVE description that a component in that bundle is affected. They will then associate every single component in that bundle with the CVE. This kind of linkage we might call fuzzy matching.

Because they use the substring of the CVE to match the components and CVEs, they reported hundreds of false positives. 3 components in particular generated over 210 false positives EACH. In total we reviewed over **975** (over 55%) of Black Duck’s findings and **determined over 940 False Positives**. Below are details on some notable findings:

| **CVE & Components** | **Explanation** | **False Positives** |
| --- | --- | --- |
| BDSA-2018-3185  (7.5)  -  **211** org.apache.camel components | Sonatype found CVE-2018-8041 (5.3) which seems to match the BDSA issue. The extractAttachmentsFromMultipart() function in MailBinding.class is what leads to the vulnerability and is only present in certain versions of camel-mail.jar. This CVE should only be applied to the applicable camel-mail versions, not every component in the bundle.  Root Cause:  Camel-mail-2.19.5.jar - org/apache/camel/component/mail/MailBinding.class [2.1.0, 2.20.4) | **210** |
| BDSA-2024-0477  (7.4)  -  **211** org.apache.camel components | Sonatype found CVE-2024-23114 (6.5) which seems to match the BDSA issue. The CassandraAggregationRepository.class and the CassandraCamelCodec.class both contain methods which leads to the vulnerability and is only present in certain versions of camel-cassandraql.jar. This CVE should only be applied to the applicable camel-cassandraql versions, not every component in the bundle.  Root Cause:   * Camel-cassandraql-2.19.5.jar - org/apache/camel/processor/aggregate/cassandra/CassandraAggregationRepository.class [2.15.0, 3.21.4) * Camel-cassandraql-2.19.5.jar - org/apache/camel/processor/aggregate/cassandra/CassandraCamelCodec.class [2.15.0, 3.21.4) | **210** |
| BDSA-2018-2585  (9.8)  -  **211** org.apache.camel components | Sonatype found CVE-2018-8027 (9.8) which seems to match the BDSA issue. The doProcess() function's conditional logic in ValidatingProcessor.class is what leads to the vulnerability and is only found in certain versions of camel-core.jar. This CVE should only be applied to the applicable camel-core versions, not every component in the bundle.  Root Cause: camel-core-2.16.2.jar - org/apache/camel/processor/validation/ValidatingProcessor.class [1.1.0, 2.20.4) | **209** |
| BDSA-2018-2904  (9.3)  -  **29** org.eclipse.jetty components | Sonatype found CVE-2019-10241 (6.1) affecting certain versions of jetty dependencies which seemed to match the BDSA issue:   * **org.eclipse.jetty.aggregate:jetty-all-9.2.25.v20180606.jar** - org/eclipse/jetty/servlet/DefaultServlet.class ( , 9.2.27.v20190403) * **org.eclipse.jetty.aggregate:jetty-all-9.2.25.v20180606.jar** - org/eclipse/jetty/util/resource/Resource.class ( , 9.2.27.v20190403) * **org.eclipse.jetty.aggregate:jetty-all-9.2.25.v20180606.jar** - org/eclipse/jetty/server/handler/ResourceHandler.class [7.0.0.RC2 , 9.2.27.v20190403) * **org.eclipse.jetty:jetty-servlet** - The sendDirectory() function in ResourceService.class and DefaultServlet.class files and the doDirectory() function in the ResourceHandler.class file use the getListHTML() function in the Resource.class file to fetch resource list as an HTML directory listing. Version Affected: [7.0.0.M0,9.2.26.v20180806] * **org.eclipse.jetty:jetty-util** - The sendDirectory() function in ResourceService.class and DefaultServlet.class files and the doDirectory() function in the ResourceHandler.class file use the getListHTML() function in the Resource.class file to fetch resource list as an HTML directory listing. Version Affected: [7.0.0.M0,9.2.26.v20180806] * **org.eclipse.jetty:jetty-server-9.3.14.v20161028.jar** - org/eclipse/jetty/server/handler/ResourceHandler.class [9.3.0.M0 , 9.3.26.v20190403) * **org.eclipse.jetty:jetty-servlet-9.3.14.v20161028.jar** - org/eclipse/jetty/servlet/DefaultServlet.class [9.3.0.M0 , 9.3.26.v20190403) * **org.eclipse.jetty:jetty-util-9.3.14.v20161028.jar** - org/eclipse/jetty/util/resource/Resource.class [9.3.0.M0 , 9.3.26.v20190403)   Only these 6 components should have been implicated, not all 29. | **23** |
| CVE-2022-2048  (7.5)  -  29 org.eclipse.jetty components | This CVE affects the http2-server (<https://central.sonatype.com/artifact/org.eclipse.jetty.http2/http2-server/9.3.14.v20161028>) component which is normally a transitive dependency of certain jetty versions, but was not present in the project at all. Neither Black Duck nor Sonatype identified this dependency in the project. Not even the affected class file HttpChannelOverHTTP2 was included in the scanned directory.  Furthermore, the Sonatype security research team discovered that this vulnerability was introduced in version 9.3.9.M0 and therefore does not affect all versions prior to 9.4.47 as stated in the advisory meaning that even if the component was present, version 9.2.25.v20180606 would not be affected (version 9.3.14.v20161028 likely would). | **29** |
| CVE-2020-13947  (9.3)  -  **26** org.apache.activemq components | The CVE-2020-13947 (6.1) only affects the activemq-web-console package which is a transitive of activemq (<https://central.sonatype.com/artifact/org.apache.activemq/apache-activemq/5.4.2/dependencies>) but neither Sonatype nor Black Duck identified this component in the project at all.  Further, CVE-2020-13947 implicates versions after 5.15.12 because of an insufficient fix for CVE-2020-1941 and so should not be applicable in version 5.4.2. CVE-2020-1941 implicates that component for versions 4.0 - 5.15.11, but was also repeated for all 26 components in the activemq bundle. | **26** |
| CVE-2020-1941  (9.3)  -  **26** org.apache.activemq components | The CVE-2020-1941 (6.1) only affects the activemq-web-console package which is a transitive of activemq (<https://central.sonatype.com/artifact/org.apache.activemq/apache-activemq/5.4.2/dependencies> ) but neither Sonatype nor Black Duck identified this component in the project at all. | **26** |
| BDSA-2021-3236  (5.4)  -  **25** org.springframework components | Sonatype identified CVE-2021-22060 (4.3) which only affects certain versions spring-webmvc, spring-webflux, and spring-websocket. This issue should not have been applied to the other components in the springframework bundle.  Root Cause:   * spring-webmvc-4.3.18.RELEASE.jar - org/springframework/web/servlet/resource/ResourceHttpRequestHandler.class [3.0.4.RELEASE, 5.2.19.RELEASE) * spring-webmvc-4.3.18.RELEASE.jar - org/springframework/web/servlet/resource/PathResourceResolver.class [4.1.2.RELEASE, 5.2.19.RELEASE) | **24** |
| BDSA-2022-0011  (5.4)  **25** org.springframework components | Rather than implicating CVE-2021-22096 (4.3), Sonatype identified CVE-2021-22060 (4.3) which only affects certain versions spring-webmvc, spring-webflux, and spring-websocket.  This CVE is a follow-up to CVE-2021-22096 (4.3) that protects against additional types of input and in more places of the Spring Framework codebase. This means CVE-2021-22060 encapsulates issues from CVE-2021-22096 and adds additional risk vectors. The BDSA description is too vague to determine which component should actually be affected here.  Sonatype’s findings cleaned up these 25 “duplicate” CVEs. | **N/A** |
| CVE-2023-44487  (7.5)  -  **19** org.apache.tomcat components | Sonatype has identified that CVE-2023-44487 (7.5) affects the jetty-http2-common and http2-common packages in the Eclipse Jetty project and are vulnerable to Denial of Service (DoS). Several of the \*BodyParser.class files contain this exploit. Neither Sonatype nor Black Duck identified either of these components in the project at all. | **19** |
| CVE-2018-11039  (4.7)  **16** org.springframework components | Discussion with the Sonatype research team revealed that CVE-2018-11039 (5.3) only affects certain versions of org.springframework:spring-web [3.0.0.M3, 4.3.18) component, not the whole bundle. The relevant affected class file is org/springframework/web/filter/HiddenHttpMethodFilter.class  This project DOES contain the unaffected org.springframework:spring-web:4.3.18.RELEASE component version. Both tools *correctly* did NOT implicate this component with this CVE, however Black Duck still implicated all the older versions of all the other spring components in the bundle. | **16** |

The “*Root Cause*” data here is the class-file Sonatype has linked to the CVE as well as the versions that are implicated by this issue. The CVE details in Sonatype Lifecycle further identify the affected methods within these class files.

*These drastically false findings further highlight the benefits of Sonatype’s scanning method. The Advanced Binary Fingerprinting (ABF) paired with Sonatype’s deep dive data research ensures CVEs are associated, not only to the appropriate component, but even to the class file within. This means every issue will be uniquely assigned and this massively reduces our false positive identification rate.*

For the Sonatype False Positives we reviewed **80+ CVE** (over 50% of Sonatype’s finding) associations from the Sonatype results in detail and were unable to identify any CVE findings which were missed by Sonatype. Further, the Root Cause and Occurrences data on the Sonatype component details pages should reveal that there are no true false positives present in the application.

## CVE False Negatives

Both Sonatype and Black Duck utilize some proprietary CVE identification; SONATYPE-xxx and BDSA-xxx (although we were able to consolidate some CVEs based on the BDSA description). Because of this we were unable to fully analyze exactly which CVEs were missed by each of the tools. Rather, we looked at the NVD CVE IDs and ensured both tools were successfully finding all relevant issues for components.

When reviewing the public CVEs, Sonatype identified over **93 False Negatives** (CVEs missed by Black Duck). Here are some of the most critical examples of CVEs we believe Black Duck missed based on the reports:

| **CVE** | **Sonatype True Positives** |
| --- | --- |
| CVE-2015-5254 (9.8) | The **activemq-client** package is vulnerable to Unsafe Deserialization.  Root Cause   * activemq-core-5.4.2.jar - org/apache/activemq/command/ActiveMQObjectMessage.class [4.0.0 , 5.7.0) * activemq-core-5.4.2.jar - org/apache/activemq/util/ClassLoadingAwareObjectInputStream.class [4.0.0 , 5.7.0) |
| CVE-2020-11973 (9.8) | The **camel-netty** and **camel-netty4** packages are vulnerable to Deserialization of Untrusted Data.  Root Cause: camel-netty-2.19.5.jar - org/apache/camel/component/netty/NettyConfiguration.class ( , 3.1.0) |
| CVE-2022-45046 (9.8) | The **org.apache.camel:camel-ldap** package is vulnerable to LDAP Injection.  Root Cause: camel-ldap-2.19.5.jar - org/apache/camel/component/ldap/LdapProducer.class ( , 3.18.4) |
| CVE-2017-9096 (8.8) | The **itextpdf** and **itext7-core** packages are vulnerable to XML External Entity (XXE) attacks.  Root Cause: itext-2.1.3.jar - com/lowagie/text/pdf/XfaForm.class [1.4.8,) |
| CVE-2018-10899 (8.8) | The **jolokia-core** package is vulnerable to Cross-Site Request Forgery (CSRF).  Root Cause:   * jolokia-core-1.6.0.jar - org/jolokia/backend/BackendManager.class [1.0.3, 1.6.1) * Jolokia-core-1.6.0.jar - org/jolokia/http/AgentServlet.class [1.0.3, 1.6.1) * Jolokia-core-1.6.0.jar - org/jolokia/http/HttpRequestHandler.class [1.0.3, 1.6.1) * Jolokia-core-1.6.0.jar - org/jolokia/restrictor/AbstractConstantRestrictor.class [1.0.3, 1.6.1) * Jolokia-core-1.6.0.jar - org/jolokia/restrictor/PolicyRestrictor.class [1.0.3, 1.6.1) * Jolokia-core-1.6.0.jar - org/jolokia/restrictor/policy/CorsChecker.class [1.0.3, 1.6.1) |
| CVE-2016-2510 (8.1) | **BeanShell** is vulnerable to Remote Code Execution (RCE) due to unsafe deserialization of data.  Root Cause: bsh-1.3.0.jar - bsh/XThis.class (,2.0b6) |
| CVE-2024-22262 (8.1) | The **spring-web** package is vulnerable due to Improper Input Validation.  Root Cause: spring-web-4.3.18.RELEASE.jar - org/springframework/web/util/UriComponentsBuilder.class [3.1.0.RC1, 5.3.34) |

For the Sonatype False Negatives we reviewed **over 975** CVE associations from the Black Duck results in detail and were unable to identify any CVE findings which were missed by Sonatype.

## Component False Positives and Negatives

There were some notable differences in the component findings as well.

* Sonatype: Found **309** components not identified by Black Duck
* Black Duck: Found **168** components not identified by Sonatype

We will explore these results in more depth below.

### Partial Matches

Before we dive into the comparison of components missed by each tool, there were some partial matches. These were components which either had differing version numbers or had differences in the component names with nuances like the “.dll” extension included on the Sonatype result while not included in Black Duck. There were **59 partial matches** where Black Duck found a different version than Sonatype. Due to Sonatype’s ABF and occurrences listing where we found the affected binary, we believe these to be False positives by Black Duck, although we cannot be 100% certain without reviewing the directory path information where Black Duck believes to have found the components.

### Black Duck Component Results

Furthermore, Black Duck identified **109** components not identified by Sonatype. We believe these to be False positives as well due to their method of scanning. However, we cannot be 100% certain without reviewing the directory path information where Black Duck believes to have found the components.

### Sonatype Component Results

With Sonatype’s **309** identified components which were not in the Black Duck data, we prove each finding to be true using the occurrences. However, there were some findings that likely should not have been included in the scan target. These included a number of IDE extension libraries and developer tools (which would not usually make it to a CI/CD build). Here are some examples:

* **142 Visual Studio extension libraries** (ex: V24\_0\_03112024.zip/V24\_0\_03112024/VSTEST\_BRIDGE\_1\_1\_0\_1048\_30JAN2024\_V24\_0\_MSWinX64.zip/bin/test/vs\_test\_platform.zip/vs\_test\_platform/16.4.0/tools/net451/Common7/IDE/Extensions/
* **26 ASP.NET extensions** (ex: V24\_0\_03112024.zip/V24\_0\_03112024/CLOUD\_CONTROL\_2\_1\_1\_1132\_09JUN2023\_V24\_0\_MSWinX64.zip/bin/cloudcontrol/client/)
* **12 Microsoft.TestPlatform dependencies** (ex: V24\_0\_03112024.zip/V24\_0\_03112024/VSTEST\_BRIDGE\_1\_1\_0\_1048\_30JAN2024\_V24\_0\_MSWinX64.zip/bin/test/vs\_test\_platform.zip/vs\_test\_platform/16.4.0/tools/net451/Common7/IDE/Extensions/TestPlatform/ko)
* **23 Developer Express Inc libraries** (ex: V24\_0\_03112024.zip/V24\_0\_03112024/CUT\_V24\_0\_03112024\_BUILD\_1021\_MSWinX64.zip/bin/elf/viewer/DevExpress

This cleaned up about **191** of the components, and in a true implementation of Sonatype Lifecycle developer dependencies like this would be omitted.

Otherwise, because of the occurrences data proving the location of each of the components, this results in **118 Component False Negatives** from Black Duck (and True Positives for Sonatype). These Sonatype True Positives included about 25 org.apache.tomcat components which Black Duck did not identify.

These are the top 8 components with critical issues missed by Black Duck:

| **Component** | **CVE Count** | **Severity Sum** | **Severity Range** |
| --- | --- | --- | --- |
| 1. org.bouncycastle:bcprov-jdk16:1.45  2. org.bouncycastle:bcprov-ext-jdk16:1.45 | 17 | 100.2 | 3.7 - 7.7 |
| org.noear:luffy.cap.extend.excel:1.6.1 | 10 | 66.6 | 5.3 - 7.5 |
| org.apache.tomcat:tomcat-coyote:10.0.5 | 9 | 59.3 | 3.7 - 7.5 |
| com.sun.xml.ws:webservices-rt:1.3 | 7 | 43.1 | 4.3 - 9.8 |
| org.apache.tomcat:tomcat-catalina:10.0.5 | 6 | 36.7 | 4.3 - 7.5 |
| org.apache.zookeeper:zookeeper:3.4.6 | 4 | 26.8 | 5.9 - 7.5 |
| Newtonsoft/Json.NET:Newtonsoft.Json.dll:6.0.5.17707 | 2 | 15 | 7.5 |
| bsh:bsh:1.3.0 | 1 | 8.1 | 8.1 |

## Sonatype Enhanced Data

#### Sonatype Deviation Advisories

Sonatype identified a total **238 unique CVEs** in the application report, and **49 (over 20%)** of them contained *Sonatype Advisory Deviation Notices* in the vulnerability details. These notices provide details on inaccuracies Sonatype identified in the public CVE database details and corrected in our Sonatype CVE database.

These deviations include the incorrect implication of versions by the NVD, incomplete fixes, newly discovered vulnerable class-files affected by the CVE, and more. Below are some critical (8+) CVE advisories from the project that were worth noting:

| **CVE** | **Category** | **Sonatype Explanation** |
| --- | --- | --- |
| CVE-2018-8027  (9.8) | Wrong version(s) implicated | The **Apache Camel** package is vulnerable to XML eXternal Entity (XXE) Injection. The Sonatype security research team discovered that the root cause of the vulnerability is in 1.1.0, not in 2.20.0 as the advisory states. |
| CVE-2020-11973  (9.8) | Wrong version(s) implicated | The `**camel-netty**` and `**camel-netty4**` packages are vulnerable to Deserialization of Untrusted Data. The Sonatype security research team discovered that this vulnerability was fixed in version 3.1.0 of `camel-netty` and version 2.25.1 of `camel-netty4`, not 3.2.0 and 2.25.1 as stated in the advisory. |
| CVE-2024-22369  (8.8) | Wrong version(s) implicated | The `**camel-sql**` component of Apache Camel is vulnerable due to the Deserialization of Untrusted Data. The Sonatype Security Research team discovered that this vulnerability was actually introduced in version 2.7.0, when the Camel JDBC aggregation repository was added, instead of 3.0.0 as stated in the advisory. |
| CVE-2022-45046 (9.8) | Wrong version(s) implicated | The `**org.apache.camel:camel-ldap**` package is vulnerable to LDAP Injection. The Sonatype Security Research team discovered that the fix for this vulnerability was not backported to version 3.14.6 as stated in the advisory. |
| CVE-2017-9096  (8.8) | Adjusted CVE Severity | The `**itextpdf**` and `**itext7-core**` packages are vulnerable to XML External Entity (XXE) attacks. The Sonatype Security Research team has created a CVSS score that differs from the publicly available score that, based on our research, more accurately reflects the severity of the vulnerability. |
| CVE-2020-10683  (9.8) | Adjusted CVE Severity | The `**dom4j**` package is vulnerable to XML eXternal Entity (XXE) attacks. The Sonatype Security Research team has created a CVSS score that differs from the publicly available score that, based on our research, more accurately reflects the severity of the vulnerability. |
| CVE-2017-7525  (9.8) | Adjusted CVE Severity | `**jackson-databind**` is vulnerable to Remote Code Execution (RCE). The Sonatype Security Research team has created a CVSS score that differs from the publicly available score that, based on our research, more accurately reflects the severity of the vulnerability. |
| CVE-2019-17571  (9.8) | Additional affected class file | The `**log4j:log4j**` package is vulnerable to Remote Code Execution (RCE) due to Deserialization of Untrusted Data. The Sonatype security research team discovered that the vulnerability actually impacts the `SocketNode` class being referenced by `SocketServer` class and not the `SocketServer` class itself as indicated in the NVD advisory. |
| CVE-2023-46604  (9.8) | Wrong components implicated | The Apache `**activemq-client**` and `**activemq-openwire-legacy**` packages are vulnerable due to the Deserialization of Untrusted Data. The Sonatype Security Research team understands that despite the advisory (<https://activemq.apache.org/security-advisories.data/CVE-2023-46604-announcement.txt>) recommending that users upgrade "both brokers and clients" to a higher version, the vulnerability primarily concerns and is exploitable for the `activemq-client`, `activemq-core`, and `activemq-openwire-legacy` packages, as well as the components containing these packages. There is no evidence of `activemq-broker` being affected. |
| CVE-2017-7658  (9.8) | Typo correction and wrong version implicated | **Eclipse Jetty Server** is vulnerable to HTTP request smuggling. The Sonatype security research teams believes that the parenthetical about non HTTP/1.x configurations being vulnerable is a typo, and that \*only\* the HTTP/1.x configurations are vulnerable. In addition, a fix that remediates this vulnerability regardless of configuration has been released in version 9.2.25.v20180606 for the 9.2.x branch, version 9.3.24.v20180605 for the 9.3.x branch, and version 9.4.11.v20180605 for the 9.4.x branch. |

These corrections to the public data can have some significant impacts on your true level of risk as well as how to actually fix these issues. By relying on tools that utilize public data sources without significant effort in researching the accuracy of issues, organizations open themselves up to significant levels of true risk due to false negatives.

#### Workarounds

In addition to the enhanced CVE details and corrective advisory notices, our research team provides added context and remediation advice ranging from version change remediation to workarounds and alternatives so that remediation can take place even if a safe version of the components is not available. This is a huge added value for developers who are responsible for taking the necessary steps to actually fix the security issues.

In this report Sonatype identified a total **238 unique CVEs** and coincidentally (given the above stats) it also contained **49 (over 20%)** alternative fixes for CVEs. Below we have listed a few of the noteworthy workarounds from this project:

| **CVE** | **Workaround** |
| --- | --- |
| CVE-2022-23305  (9.8) | There is no non-vulnerable upgrade path for `log4j:log4j` 1.x. We recommend upgrading to `log4j` 2.x, which introduced a `JDBCAppender` that makes use of prepared statements and parameterization to protect against SQL Injection. Alternatively, this vulnerability can be mitigated in `log4j` 1.x by removing all usages of the `JDBCAppender` from the application's configuration. This vulnerability has been fixed in version `1.3.0.Final` of `org.jboss.logmanager:log4j-jboss-logmanager` by removing the vulnerable class from the project." |
| CVE-2017-7525  (9.8) | As of version 2.10.0, Jackson now provides a safe default typing solution that fully mitigates this vulnerability.  In order to mitigate this vulnerability, we recommend upgrading to at least version 2.10.0 and changing any usages of `enableDefaultTyping()` to `activateDefaultTyping()`. Alternatively, if upgrading is not a viable option, this vulnerability can be mitigated by disabling default typing. Instead, you will need to implement your own:>It is also possible to customize global defaulting, using ObjectMapper.setDefaultTyping(...) -- you just have to implement your own TypeResolverBuilder (which is not very difficult); and by doing so, can actually configure all aspects of type information. Builder itself is just a short-cut for building actual handlers. |
| CVE-2017-7657  (9.8) | Alternatively, this vulnerability can be mitigated by disabling HTTP/1.1 support. |
| CVE-2020-25638  (7.4) | This vulnerability can be mitigated by disabling SQL comments: An upgrade is recommended, but if you are using a very old version which makes it difficult to upgrade to the latest supported versions (series 5.4 and 5.3 at time of writing this), you can disable SQL comments by setting: `hibernate.use\_sql\_comments=false`. This also is the default, so if you didn't set the use\_sql\_comments at all you are not affected. |
| CVE-2018-17187  (7.4) | If upgrading is not currently possible then potential workarounds include providing a custom SSLContext which enables hostname verification or omitting use of the 'transport.ssl(...)' methods and performing TLS through other means such as utilizing existing IO framework support or supplying a custom transport wrapper layer. |
| CVE-2019-0201  (5.9) | This vulnerability can be mitigated by using an authentication method other than `DigestAuthenticationProvider`. |
| CVE-2023-26048  (5.3) | Alternatively, if upgrading is not a viable option, this vulnerability may be mitigated by setting the `maxRequestSize` parameter of affected `:MultipartConfig` configurations which, while still read into memory, will limit the entirety of multipart request content. |

#### Additional Licenses

Although we did not do a full license analysis, Black Duck missed 4 components which contained severe license issues:

| **License** | **Component** | **License Obligation** |
| --- | --- | --- |
| Copyleft | Org.apache.tomcat:  Tomcat-servlet-api:  10.0.5 | **GPL-2.0-with-classpath-exception — Obligation Text** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.) |
| Copyleft | Org.tmatesoft.svnkit:  Svnkit:  1.2.0.4949 | **TMate-OSL — Obligation Text** Redistributions in any form must be accompanied by information on how to obtain complete source code for the software that uses SVNKit and any accompanying software that uses the software that uses SVNKit. The source code must either be included in the distribution or be available for no more than the cost of distribution plus a nominal fee, and must be freely redistributable under reasonable conditions. For an executable file, complete source code means the source code for all modules it contains. It does not include source code for modules or files that typically accompany the major components of the operating system on which the executable file runs |
| Copyleft | Org.eclipse.jetty.orbit:  Javax.mail.glassfish:  1.4.1.v201005082020 | **GPL-2.0-with-classpath-exception — Obligation Text** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.) |
| Commercial | Com.couchbase.cblite:  Servlet:  2-3 | **Couchbase-Community-Edition-LA — Obligation Text** Licensee will not: (a) reverse engineer, disassemble, or decompile the Software (except to the extent such restrictions are prohibited by law) |

### App 1 Conclusion

In summary, Sonatype’s deep dive research revealed how necessary it is to review and correct results from public data sources. Relying solely on public resources such as the NVD, will inevitably lead to false positives, false negatives, and in turn, unnecessary remediation and potential blind spots for security. Sonatype’s Advanced Binary Fingerprinting was able to pinpoint exactly where components contained security vulnerabilities, which allows teams to remediate faster, and trust the results without significant manual review.

Blackduck did not seem to pay close attention to detail for these CVE associations, and due to their limited data review, they loosely grouped components to CVEs based on their substring descriptions. They reported hundreds of issues which were not present at all or not correctly associated, seeming to get confused by artifact versions and groups. These inaccuracies are detrimental to developer teams, resulting in significant manual review, distrust in the tool, and massive gaps in security insight.

# ***The information contained within this report is Sonatype, Inc.’s confidential and proprietary information and viewers may not (i) disclose this report (or any part thereof) to any third party (including but not limited to any competitor of Sonatype) and/or (ii) use this report for any purpose other than evaluating Sonatype’s software for your organization’s internal use.***