azul

Azul Vulnerability Detection

Software Supply Chain: Vulnerable 3rd Party Code

MORE 3RD PARTY CODE

40% - 80%

Of code in new software projects[1]

✕

MORE VULNERABILITIES

200+

In 3rd party Java libraries and apps in Q1 alone[2]

✕

MORE EXPOSURE

56%

Of security pros lack confidence that running applications are tested to find and fix high risk vulnerabilities that an attacker may exploit[3]

Sources: 1. Gartner, A Software Bill of Materials Is Critical to Software Supply Chain Management, 2. Based on analysis of National Vulnerability Database, 3. IBM, The state of vulnerability management in the cloud and on-premises

How did you handle LOG4J

One of the Most Serious Software Vulnerabilities in History[1]

52% of Teams Spent a Month or More Remediating[2]

48% of Teams Gave Up Holiday or Weekends to Remediate[2]

Did You Have an Accurate List of Where Log4j Was Running?

Did Log4j Vulnerabilities Stay Away After You First Got Rid of Them?

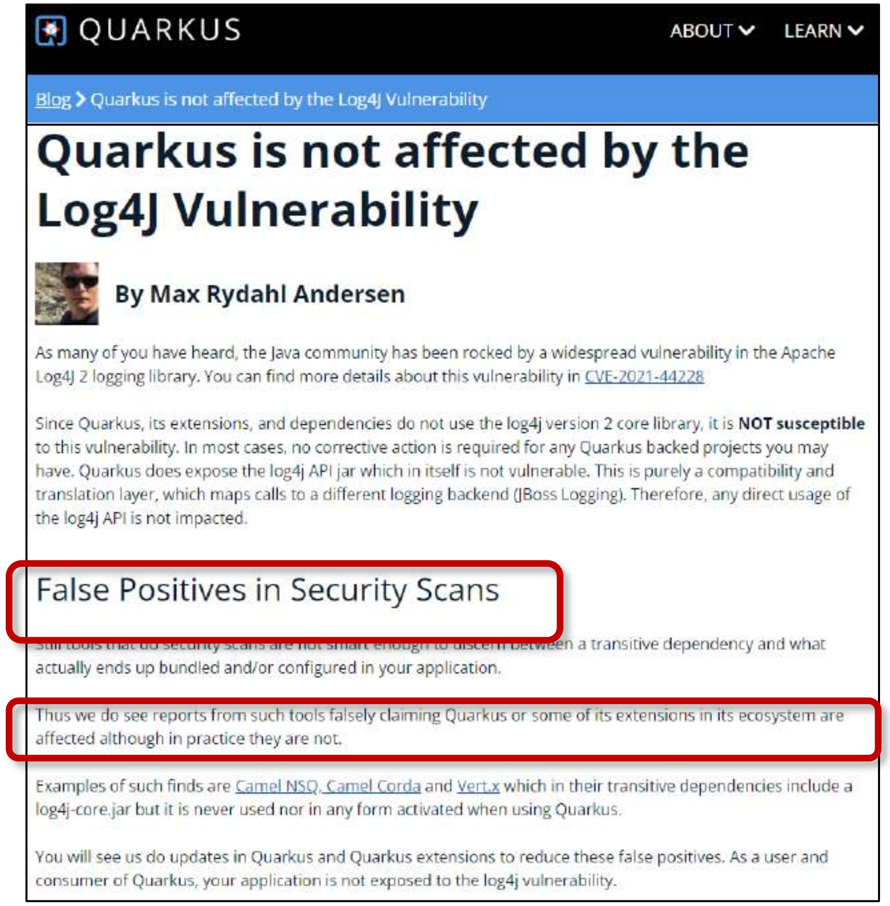
How Prepared Are You for the Next Serious Vulnerability ?

How Confident Are You in Knowing What's in Your Java Estate?

Sources: 1. US Dept of Homeland Security 2. (ISC)² Pulse Survey

Many Commercial Scanners Were/Are Wrong.

2021 – Quarkus Accused a Lot



QUARKUS ABOUT LEARN

Blog > Quarkus is not affected by the Log4J Vulnerability

Quarkus is not affected by the Log4J Vulnerability

By Max Rydahl Andersen

As many of you have heard, the Java community has been rocked by a widespread vulnerability in the Apache Log4J 2 logging library. You can find more details about this vulnerability in CVE-2021-44228

Since Quarkus, its extensions, and dependencies do not use the log4j version 2 core library, it is **NOT susceptible** to this vulnerability. In most cases, no corrective action is required for any Quarkus backed projects you may have. Quarkus does expose the log4j API jar which in itself is not vulnerable. This is purely a compatibility and translation layer, which maps calls to a different logging backend (JBoss Logging). Therefore, any direct usage of the log4j API is not impacted.

False Positives in Security Scans

Still tools that do security scans are not smart enough to discern between a transitive dependency and what actually ends up bundled and/or configured in your application.

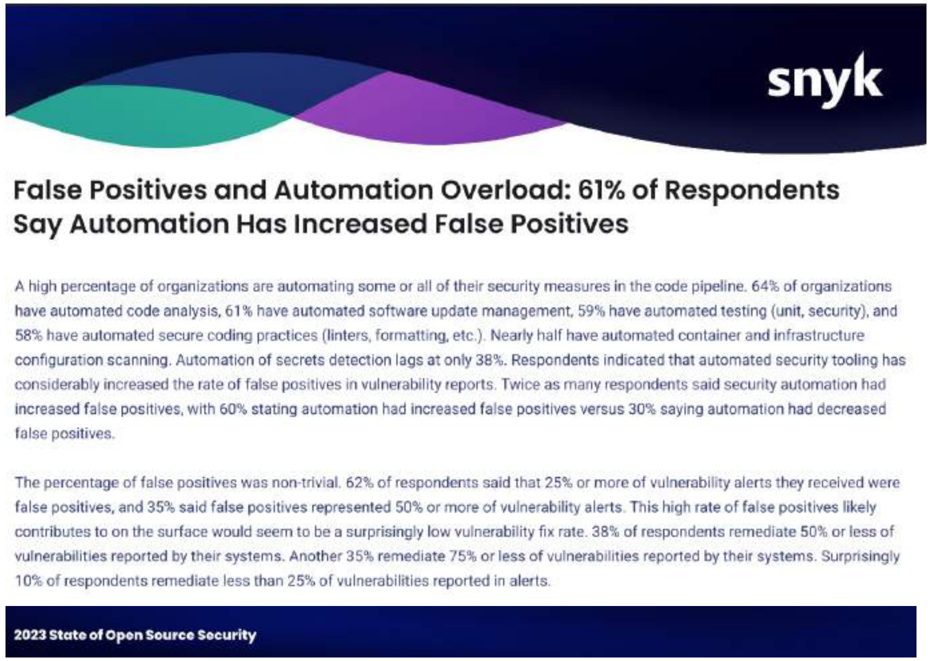
Thus we do see reports from such tools falsely claiming Quarkus or some of its extensions in its ecosystem are affected although in practice they are not.

Examples of such finds are Camel NSQ, Camel Corda and Vert.x which in their transitive dependencies include a log4j-core.jar but it is never used nor in any form activated when using Quarkus.

You will see us do updates in Quarkus and Quarkus extensions to reduce these false positives. As a user and consumer of Quarkus, your application is not exposed to the log4j vulnerability.

https://quarkus.io/blog/quarkus-and-CVE-2021-4428/

2023 – Snyk Still Cites False Positives



snyk

False Positives and Automation Overload: 61% of Respondents Say Automation Has Increased False Positives

A high percentage of organizations are automating some or all of their security measures in the code pipeline. 64% of organizations have automated code analysis, 61% have automated software update management, 59% have automated testing (unit, security), and 58% have automated secure coding practices (linters, formatting, etc.). Nearly half have automated container and infrastructure configuration scanning. Automation of secrets detection lags at only 38%. Respondents indicated that automated security tooling has considerably increased the rate of false positives in vulnerability reports. Twice as many respondents said security automation had increased false positives, with 60% stating automation had increased false positives versus 30% saying automation had decreased false positives.

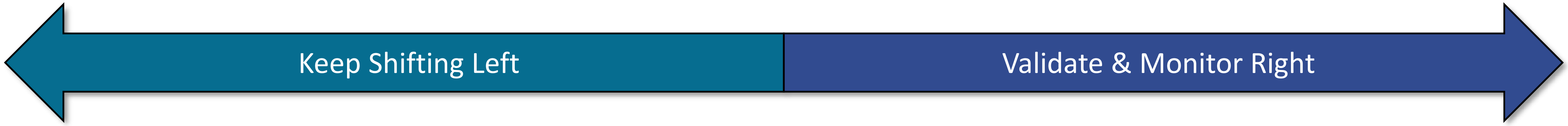
The percentage of false positives was non-trivial. 62% of respondents said that 25% or more of vulnerability alerts they received were false positives, and 35% said false positives represented 50% or more of vulnerability alerts. This high rate of false positives likely contributes to on the surface would seem to be a surprisingly low vulnerability fix rate. 38% of respondents remediate 50% or less of vulnerabilities reported by their systems. Another 35% remediate 75% or less of vulnerabilities reported by their systems. Surprisingly 10% of respondents remediate less than 25% of vulnerabilities reported in alerts.

2023 State of Open Source Security

"62% of respondents said that 25% or more of vulnerability alerts were false positives, and 35% said false positives represented 50% or more of vulnerability alerts."

https://resources.snyk.io/state-of-open-source-2023

The JVM Can Improve Your Accuracy



- Keep scanning custom code.
- Keep scanning containers & workloads.
- Keep working with teams.
- Do what you're doing today. It's great.

- Java applications are already using the JVM.

  Nothing extra to install or manage.
- Decrease False Positives by learning what code loads – focus on these vulnerabilities first.
- Performs well in dev/QA and Production.

How you benefit:

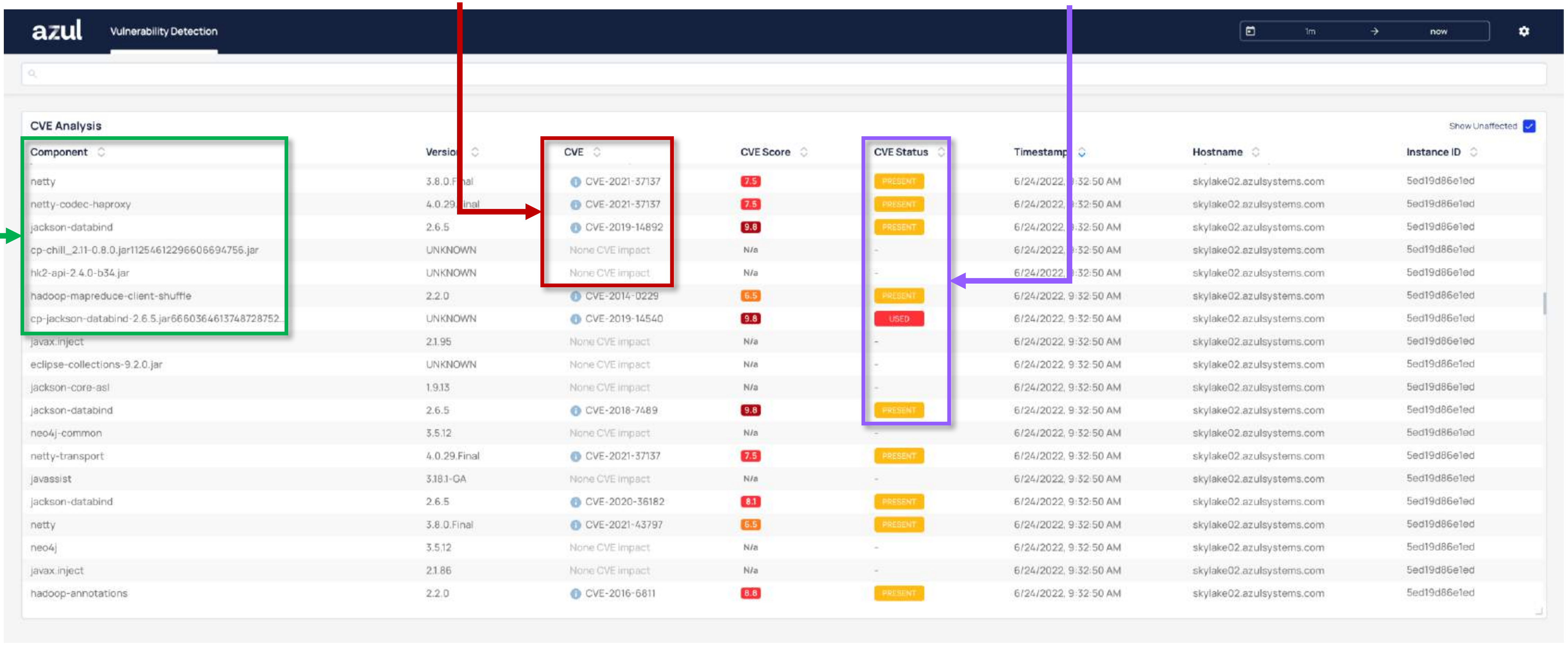
1. **Less time installing/managing security tools.**
2. **Focus on results that matter, not everything.**
3. **Decreased "alert fatigue" and "dashboard fatigue."**

What Information Can My JVM Provide?

What do I have?

Is it vulnerable?

Do I actually load that code?

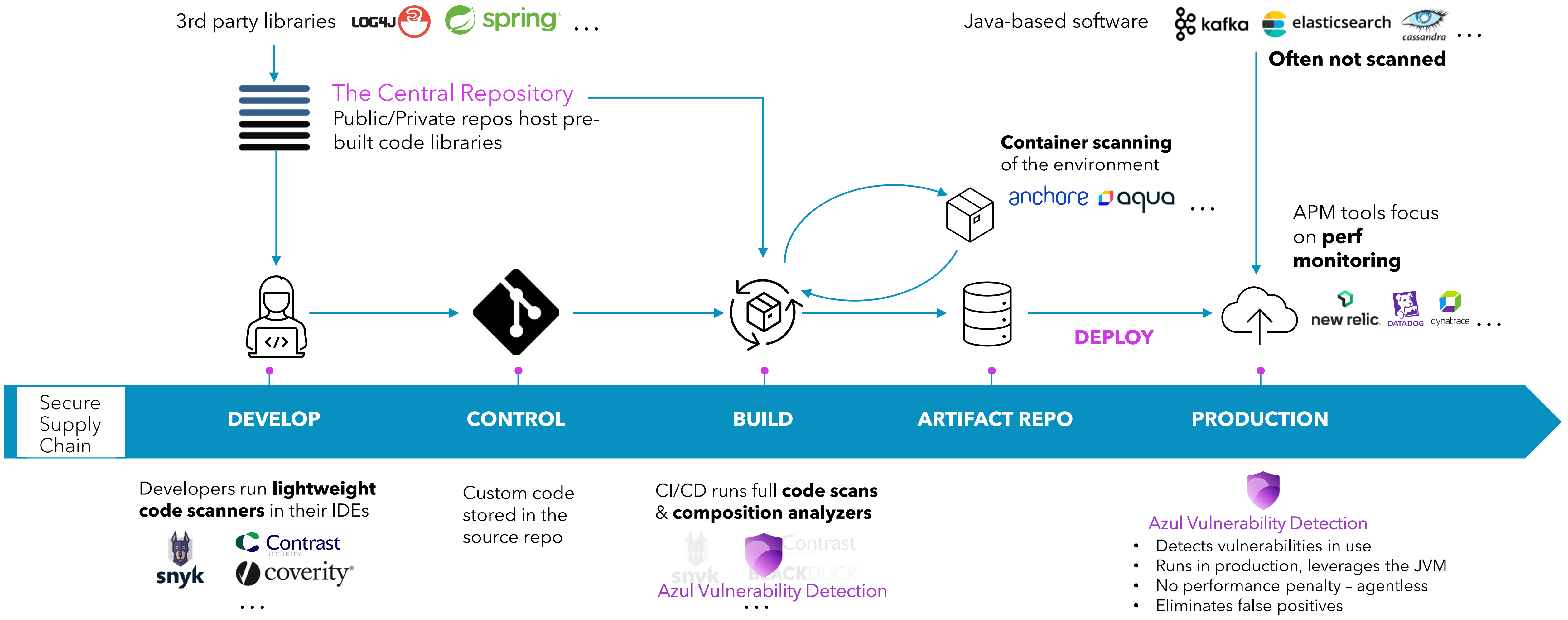


CVE Analysis

Show Unaffected ✓

Component	Versio	CVE	CVE Score	CVE Status	Timestamp	Hostname	Instance ID
netty	3.8.0.F nal	CVE-2021-37137	7.5	PRESENT	6/24/2022, :32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
netty-codec-haproxy	4.0.29 inal	CVE-2021-37137	7.5	PRESENT	6/24/2022, :32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
jackson-databind	2.6.5	CVE-2019-14892	9.8	PRESENT	6/24/2022, :32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
cp-chill_2.11-0.8.0.jar11254612296606694756.jar	UNKNOWN	None CVE impact	N/a	-	6/24/2022, :32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
hk2-api-2.4.0-b34.jar	UNKNOWN	None CVE impact	N/a	-	6/24/2022, :32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
hadoop-mapreduce-client-shuffle	2.2.0	CVE-2014-0229	6.5	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
cp-jackson-databind-2.6.5.jar6660364613748728752...	UNKNOWN	CVE-2019-14540	9.8	USED	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
javax.inject	2.1.95	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
eclipse-collections-9.2.0.jar	UNKNOWN	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
jackson-core-asl	1.9.13	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
jackson-databind	2.6.5	CVE-2018-7489	9.8	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
neo4j-common	3.5.12	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
netty-transport	4.0.29.Final	CVE-2021-37137	7.5	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
javassist	3.18.1-GA	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
jackson-databind	2.6.5	CVE-2020-36182	8.1	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
netty	3.8.0.Final	CVE-2021-43797	6.5	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
neo4j	3.5.12	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
javax.inject	2.1.86	None CVE impact	N/a	-	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed
hadoop-annotations	2.2.0	CVE-2016-6811	8.8	PRESENT	6/24/2022, 9:32:50 AM	skylake02.azulsystems.com	5ed19d86e1ed

Complete Visibility: Left & Right



Often not scanned

The Central Repository
Public/Private repos host pre-built code libraries

Container scanning of the environment

APM tools focus on **perf monitoring**

Developers run **lightweight code scanners** in their IDEs

Custom code stored in the source repo

CI/CD runs full **code scans** & **composition analyzers**

Azul Vulnerability Detection

Azul Vulnerability Detection
- Detects vulnerabilities in use
- Runs in production, leverages the JVM
- No performance penalty – agentless
- Eliminates false positives

Azul Vulnerability Detection

Renaissance in Java Security

Ongoing Detection at Point-of-Use In Production

- All environment coverage: dev, test, production
- All Java software: custom, 3rd party, contracted

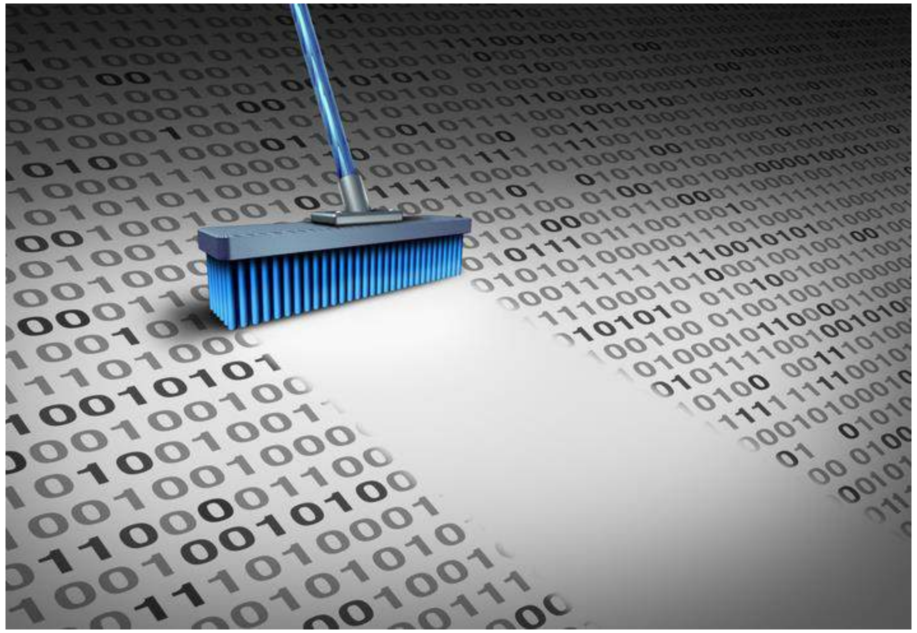
Focus Effort by Eliminating False Positives

- Accurate by watching code load
- Prioritize code that runs over code that rests
- Detection that understands Java conventions

No-Ops with Transparent Performance

- No additional agent
- No additional work
- Part of Azul JVMs (including free builds of OpenJDK)

Code Inventory: Do I use that? Can I remove code?



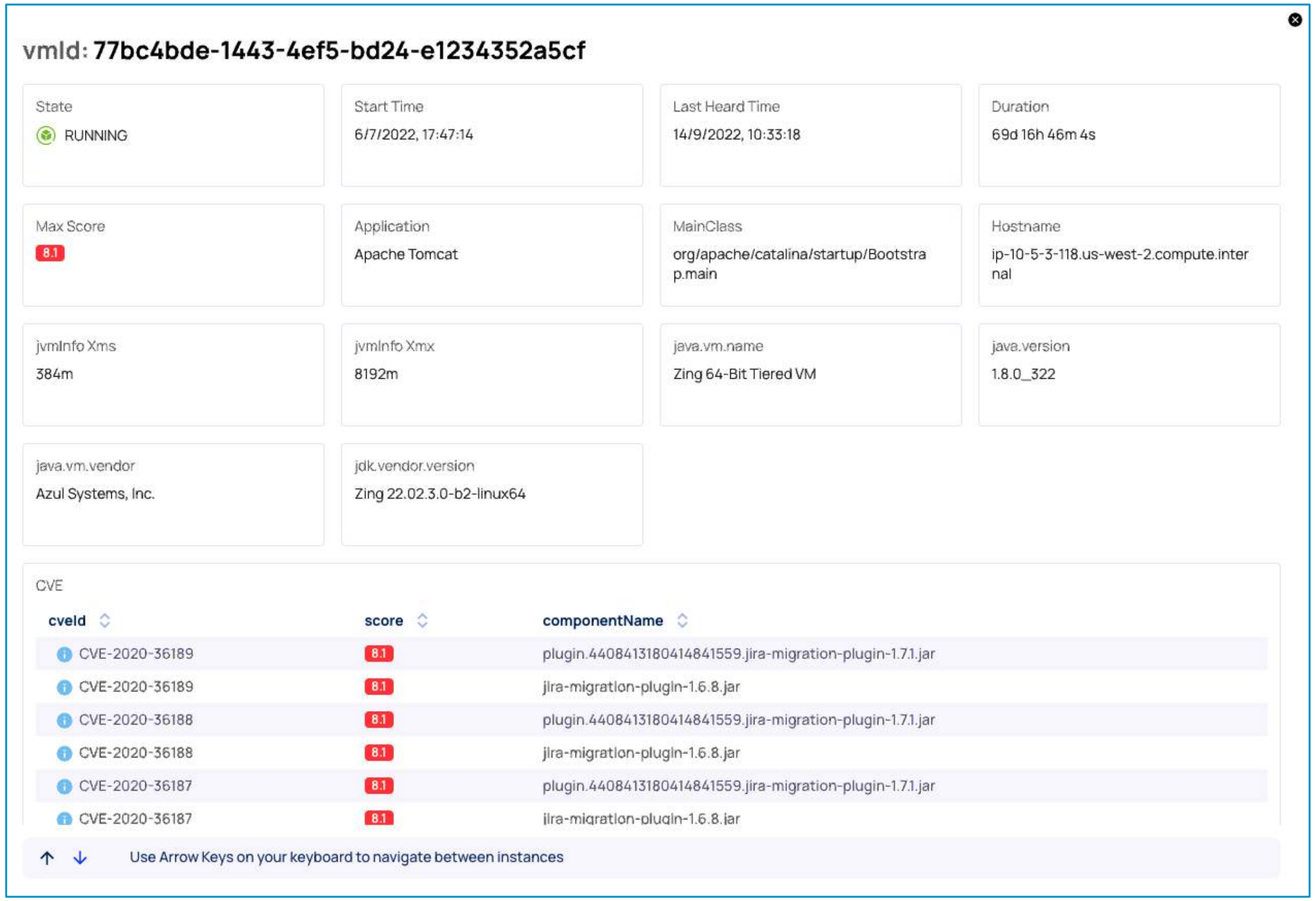
Many applications have had code written over the years that teams are afraid to touch. How can you know what you can remove?

- Primarily for long-developed applications.
- Decreases difficulty barriers to deprecating/removing unused and dead code:
  - Removing code reduces maintenance.
  - Removing code reduces attack surface.
- Data must come from production.
  - Little to no performance impact.
- An API-First way to learn what you do/don’t call.

Azul Vulnerability Detection – Metadata Collected

NO application data
NO secrets

VM Instance
VM Event
VM Jar Loaded
VM Class Loaded
VM Method First Called
VM Performance Metrics
System Info
Customer defined tags



vmId: **77bc4bde-1443-4ef5-bd24-e1234352a5cf**

State RUNNING	Start Time 6/7/2022, 17:47:14	Last Heard Time 14/9/2022, 10:33:18	Duration 69d 16h 46m 4s
Max Score 8.1	Application Apache Tomcat	MainClass org/apache/catalina/startup/Bootstrap.main	Hostname ip-10-5-3-118.us-west-2.compute.internal
jvmInfo Xms 384m	jvmInfo Xmx 8192m	java.vm.name Zing 64-Bit Tiered VM	java.version 1.8.0_322
java.vm.vendor Azul Systems, Inc.	jdk.vendor.version Zing 22.02.3.0-b2-linux64		

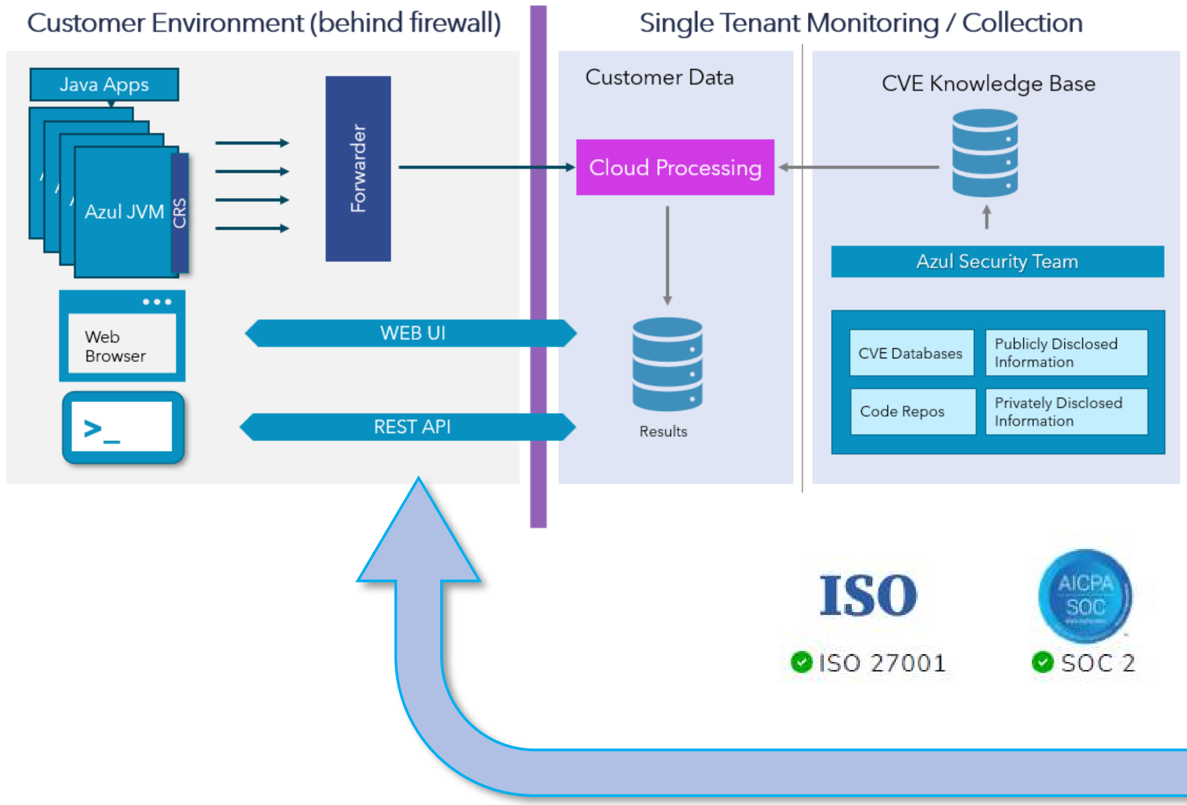
CVE

cveId	score	componentName
CVE-2020-36189	8.1	plugin.4408413180414841559.jira-migration-plugin-1.7.1.jar
CVE-2020-36189	8.1	jira-migration-plugin-1.6.8.jar
CVE-2020-36188	8.1	plugin.4408413180414841559.jira-migration-plugin-1.7.1.jar
CVE-2020-36188	8.1	jira-migration-plugin-1.6.8.jar
CVE-2020-36187	8.1	plugin.4408413180414841559.jira-migration-plugin-1.7.1.jar
CVE-2020-36187	8.1	jira-migration-plugin-1.6.8.jar

↑ ↓ Use Arrow Keys on your keyboard to navigate between instances

https://docs.azul.com/vulnerability-detection/detailed-information/what-info-is-transmitted

AVD Architecture



- No performance hit. NO AGENTS.
  - JVM already collects this data.
- Forwarder keeps JVMs isolated on customer network and consolidates data to send.
- JVMs are isolated one-way communicators. No communication back to JVM.
- CVE and Component knowledgebase in cloud.

No New Dashboards!
Integrate into what you use.

Azul Vulnerability Detection

WHAT IS IT

- A New Approach to Security
- Leverages Azul JVMs (including free builds of OpenJDK)
- Runs in Production

WHAT CAN IT DO FOR YOU

✓ Fills Critical Security Gap

✓ Detection at Point of Use

✓ Eliminates False Positives

WHAT IT IS NOT

🚫 Yet Another Code Scanner

🚫 Yet Another Dashboard

🚫 Agent-based Solution