Balbix®

An Executive Guide to Log4j Vulnerabilities

For CEOs, CFO, and other CXOs

By: **Gaurav Banga**, *Founder and CEO Balbix*

If you are the CEO, CFO or CXO of a major corporation, you are probably aware of Log4j, and perhaps perplexed by the unprecedented impact it has had on your infosec and IT teams. You might be wondering how Log4j is different from vanilla cybersecurity issues. Perhaps you are waiting for your CISO to provide root cause analysis, lessons learnt and recommendations for change in how your organization manages cyber risk. A board member or two might be pestering you for details... Here is a short writeup that might be helpful.

For Those Who Came in Late, What Is Log4j?

In summary, attackers have found a new way to get inside. When bad guys walk up and see a locked digital door into your enterprise, all they have to do is something simple, like: "Hey Siri, abracadabra, open the door"—and it opens.

My good friend, Mahendra Ramsinghani, explains this best: *Software developers like to record everything so that they can review and debug their work. Recording in the developer world is called logging, and the "camera" they use for recording/logging events is called Log4j. Log4j is prepackaged open-source code that is incorporated in software applications, like a Lego block. But this camera has a glitch and can magically become a tunnel via which attackers can get in and control the house.*

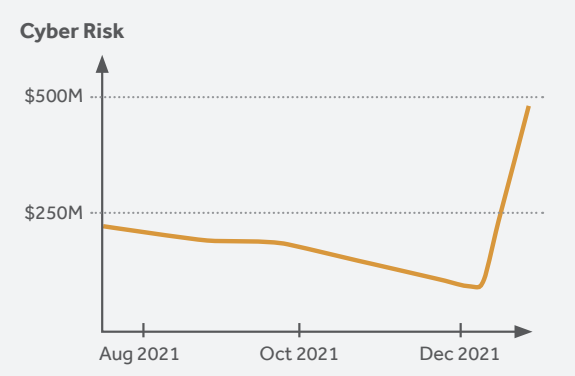


Figure 1: Cyber risk change due to Log4j

The Log4j vulnerability has received a lot of press and has been labelled by some as "the worst software vulnerability ever". The reason for this is because Log4j is everywhere. It is a part of the Apache/Java ecosystem used to program almost everything that is connected to the Internet. Your mission critical applications probably use Log4j. Your vendors and supply chain use Log4j. Most of your cybersecurity vendors also use Log4j! And, as you will see below, for several reasons it has been quite tough to mitigate Log4j.

Depending on the size of your business, your cyber risk from Log4j is likely to be 10s or 100s of millions of $s (or euros, pounds, etc.). If you were doing ok on Dec 8, 2021, in terms of cyber risk, you may not be doing ok anymore (**Figure 1**).

The Log4j vulnerability has been labeled by some as "the worst software vulnerability ever". This is because Log4j is everywhere. It is a part of the Apache/Java ecosystem used to program almost everything that is connected to the Internet. Your mission critical applications probably use Log4j. Your vendors and supply chain use Log4j. Most of your cybersecurity vendors also use Log4j...

Mitigating and Remediating Log4j

The typical methodology that infosec teams follow when dealing with a vulnerability like Log4j is outlined in **Figure 2**.

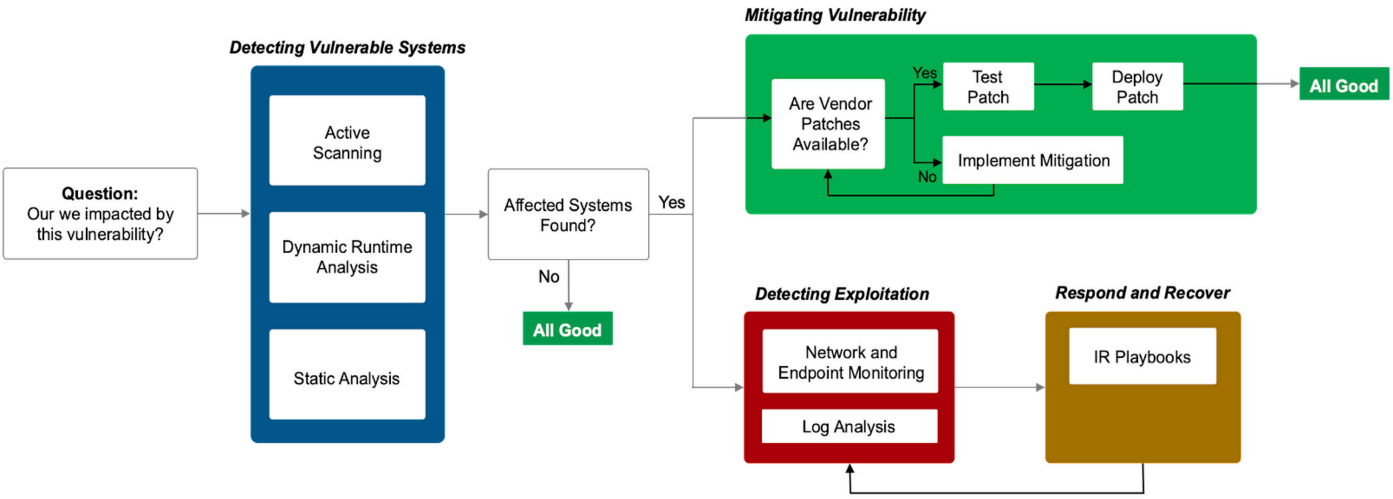


Figure 2: Response playbook for vulnerabilities like Log4j

First, your infosec team must identify enterprise assets that are affected. This involves running authenticated and unauthenticated scans. In some cases, static and dynamic analysis is also needed. Depending on the maturity of your infosec program, this identification step could take a few hours to several weeks.

After identification, the vulnerability must be remediated (or mitigated) while simultaneously looking for systems that might have already been exploited.

If you get lucky, you will not have any affected systems because you do not use the specific vulnerable module. If you are vulnerable, this mitigation process can take weeks. Obviously, your objective is to remediate or mitigate the vulnerability quickly—before any attacker can weaponize it.

Challenges With Log4j

Log4j has been the most difficult security issue that the cybersecurity industry has worked on in a long time. Some of the challenges that infosec professionals have encountered are summarized below.

1. Finding Nemo

Your infosec team simply did not have (and likely still does not have) an easy way of finding out what applications were vulnerable to Log4j. The situation is like someone warning your CISO: *50% of your apps are on fire, but I cannot say which 50%. You go figure.*

Active scanning, which is the foundation of vulnerability assessment, was not very useful for Log4j. Traditional vulnerability assessment tools do not have a comprehensive (or even close to complete) list of application versions that are vulnerable. This is primarily due to the embedded library nature of Log4j – it is part of thousands of applications without being explicitly listed in the registry of installed applications on a system. See **Figure 3**.

Many custom applications that utilize Log4j are also vulnerable. Vulnerable custom apps are typically not identified by standard vulnerability assessment tools. Usually this does not matter as most publicly disclosed software vulnerabilities only affect standard applications. In this specific case, being blind to custom apps is not acceptable since Log4j is embedded in so many of them. Infosec professionals are being forced to use deep endpoint scans, dynamic run-time analysis and static analysis combined with complex data correlation to find all vulnerable systems. ***This will take weeks!***

One point to note: many organizations are even worse off than described above. They do not even have an accurate, up-to-date inventory of their digital assets, so obviously don't know where to look for Log4j.

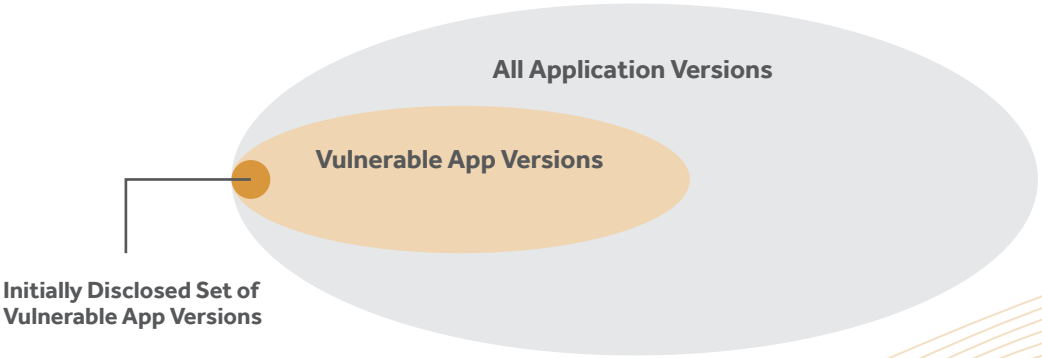


Figure 3: Application versions vulnerable to Log4j

2. Too many vulnerabilities

Due to the sheer number of systems and applications impacted by Log4j, mitigation and remediation activities need to be sharply prioritized based on risk. Unfortunately, many organizations do not have a good way to quantify the cyber risk of vulnerable assets or workloads. They cannot differentiate between highly exposed assets vs well-protected assets or critically important assets vs less important ones, and therefore cannot prioritize their efforts.

In the case of Log4j, many of your 3rd party SaaS vendors (including cybersecurity tool vendors) have also reported vulnerabilities in their systems and tools, raising high priority work for your infosec team to an unprecedented level, reviewing, quantifying, mitigating and tracking cyber risk.

3. Triple whammy

If this was not bad enough, multiple Log4j vulnerabilities were discovered in the days following the initial disclosure, with 3 new patches released in back-to-back fashion. Log4j 1.x and 2.x have different issues and upgrading to the latest version 2.17.0 is not always possible, necessitating the use of other types of mitigations.

These dynamics have tripled the work that your internal teams and your vendors had to do, and your mitigation processes are likely severely backlogged.

Figure 4 shows a mind map of Log4j related thinking that your infosec teams are working with. (Thanks to Loïc Castel for sharing this.)

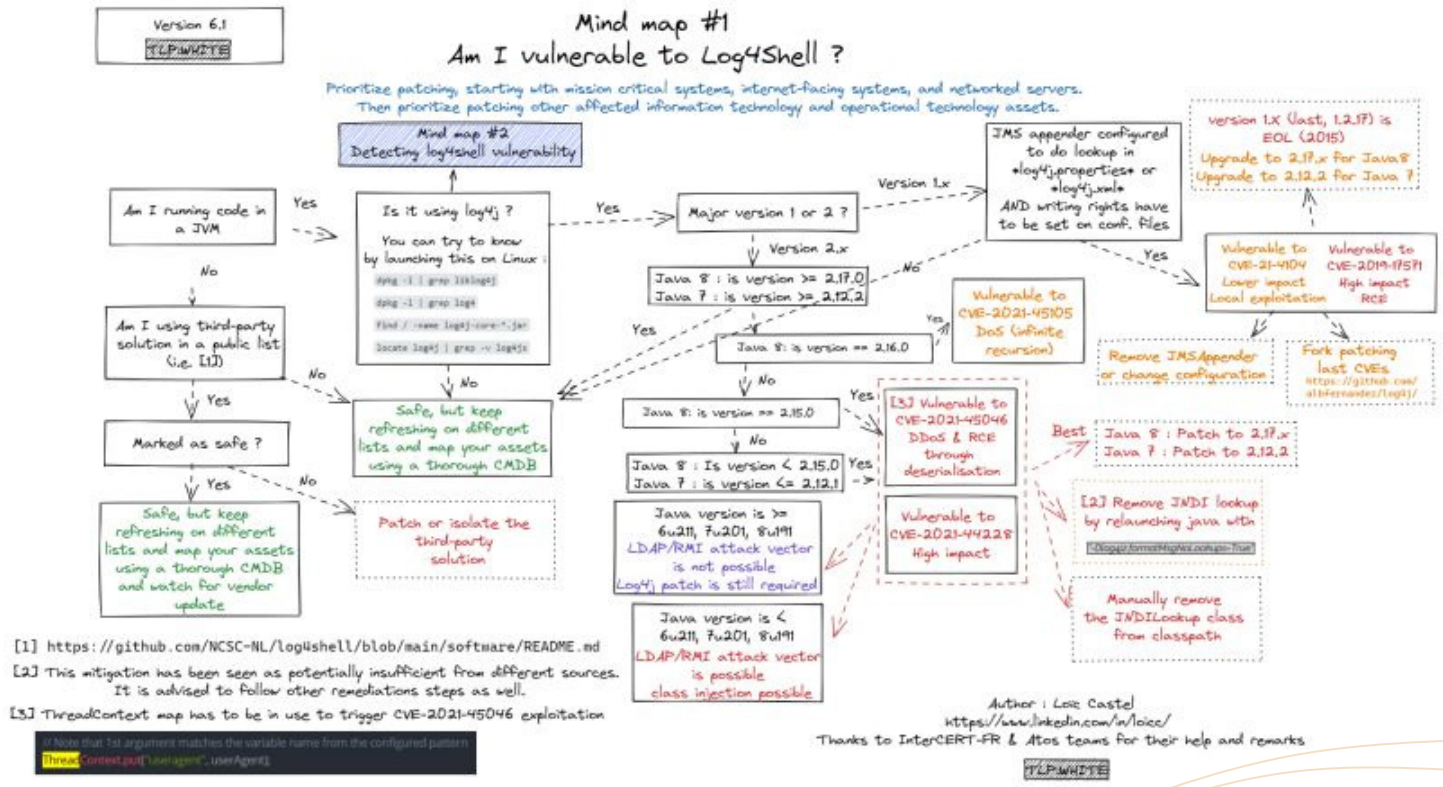
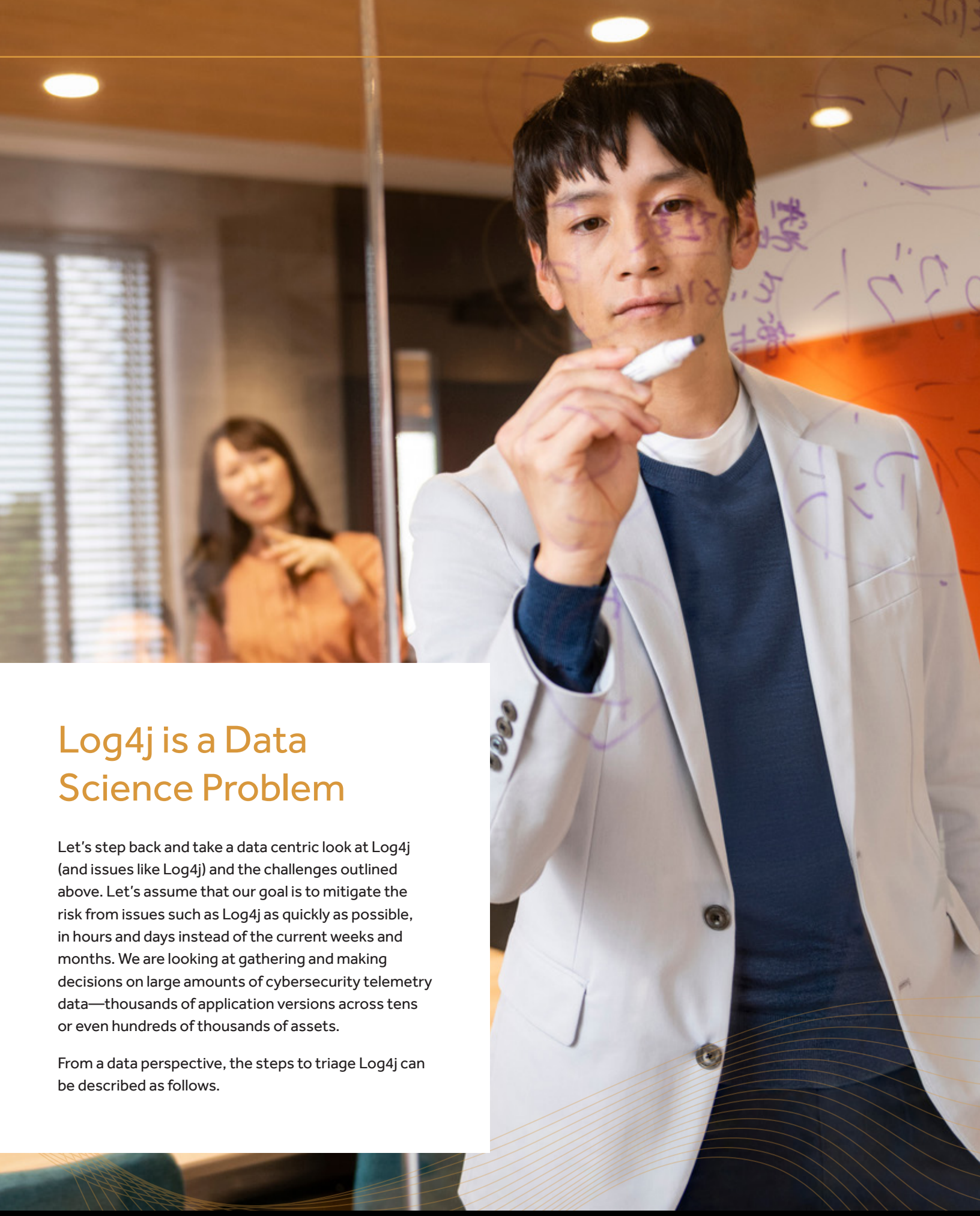


Figure 4: Mind map—Am I vulnerable to Log4shell?

Net, net, you are looking at weeks (if not months) of work in mitigating Log4j. During this time, your organization will stay at a very high state of cyber risk. Ransomware, operational integrity, theft of customer data or intellectual property are all at high likelihood of happening.



Log4j is a Data Science Problem

Let's step back and take a data centric look at Log4j (and issues like Log4j) and the challenges outlined above. Let's assume that our goal is to mitigate the risk from issues such as Log4j as quickly as possible, in hours and days instead of the current weeks and months. We are looking at gathering and making decisions on large amounts of cybersecurity telemetry data—thousands of application versions across tens or even hundreds of thousands of assets.

From a data perspective, the steps to triage Log4j can be described as follows.

STEP 1

Map from vulnerable Log4j version numbers to application versions that have vulnerable Log4j modules.

A typical initial disclosure of a software vulnerability contains information about vulnerable version numbers, and hashes for vulnerable files. In the case of Log4j, the initial disclosure indicated that Log4j v 2.x up to 2.14.0 were vulnerable, with a recommendation to update to 2.15.0. The severity of the vulnerability was flagged as 10 (which is the maximum level).

In the case of Log4j, this disclosure did not cover many impacted apps—e.g., standard apps and custom apps that embedded Log4j. Therefore, every organization needed to build up its own list of vulnerable applications. This is shown in **Figure 5.**

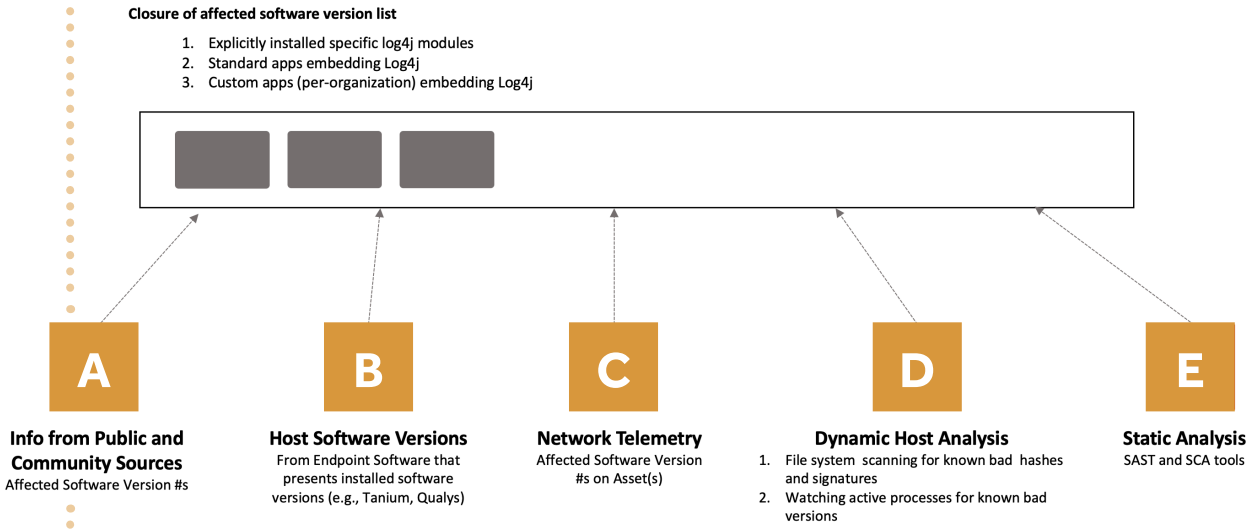


Figure 5: Quickly building a list of application versions vulnerable to Log4j

Block D in Figure 5 is very important for vulnerabilities like Log4j where the vulnerable module is embedded in an application and has no direct footprint in the installed software list. Without Block D, you will also not be able to discover your vulnerable custom apps—this information is never present in Block A. As mentioned earlier, traditional vulnerability assessment tools do not provide this information.

Block B represents your software inventory. Individual tools such as ServiceNow, Tanium, Armis, Qualys (or other vuln assessment), cloud security posture tools, etc. each contain a slice of this information and you need to unify these data sets. If you don't have both Block B and D implemented for your security program, you don't have your software bill-of-materials (S-BOM). Without an accurate S-BOM, you will struggle, and dealing with issues like Log4j will be time consuming and painful.

Block E is also a useful source for your S-BOM, but not always practical in a war time situation where you need to quickly find and mitigate risk to production software deployed on thousands of endpoints.

STEP 2

Next, we need to map to a list of assets and workloads that use these vulnerable application versions. This requires you to have an up-to-date asset inventory unifying data from all your different IT and cybersecurity tools. If you don't have this capability, you probably don't have an accurate S-BOM (Block #2 in Figure 5) either. When Log4j type issues come along, you will need to deal with multiple, sometimes conflicting inventory sources at a time when you can least afford to.

STEP 3

Once we have a list of vulnerable assets, we need to quantify the risk for each impacted asset and workload by considering vulnerability severity, asset exposure, threat level of the vulnerability, any compensating security controls and the business criticality of the asset/workload. This is a crucial calculation since you need to know which of your assets need immediate attention vs those that can wait a day or so.

STEP 4

Following risk quantification, a prioritized list of assets needing vulnerability mitigation must be dispatched to various risk owners in your organization with a directive to mitigate risk from the new vulnerabilities to acceptable levels. To be ready for this, you need to maintain an up-to-date mapping of assets to risk owners, and workloads to risk owners. You will also need to provide your risk owners with best practices around mitigation options—including the context that will let them decide when to remediate and when to utilize some other mitigation.

STEP 5

You will need capabilities around tracking, analytics and reporting that will enable you to measure risk burn down as it happens (or not) and communicate with all stakeholders.

STEP 6

Finally, you would want your 3rd party SaaS vendors to follow a systematic process like the one outlined above and share quantitative reports with you during times like this.

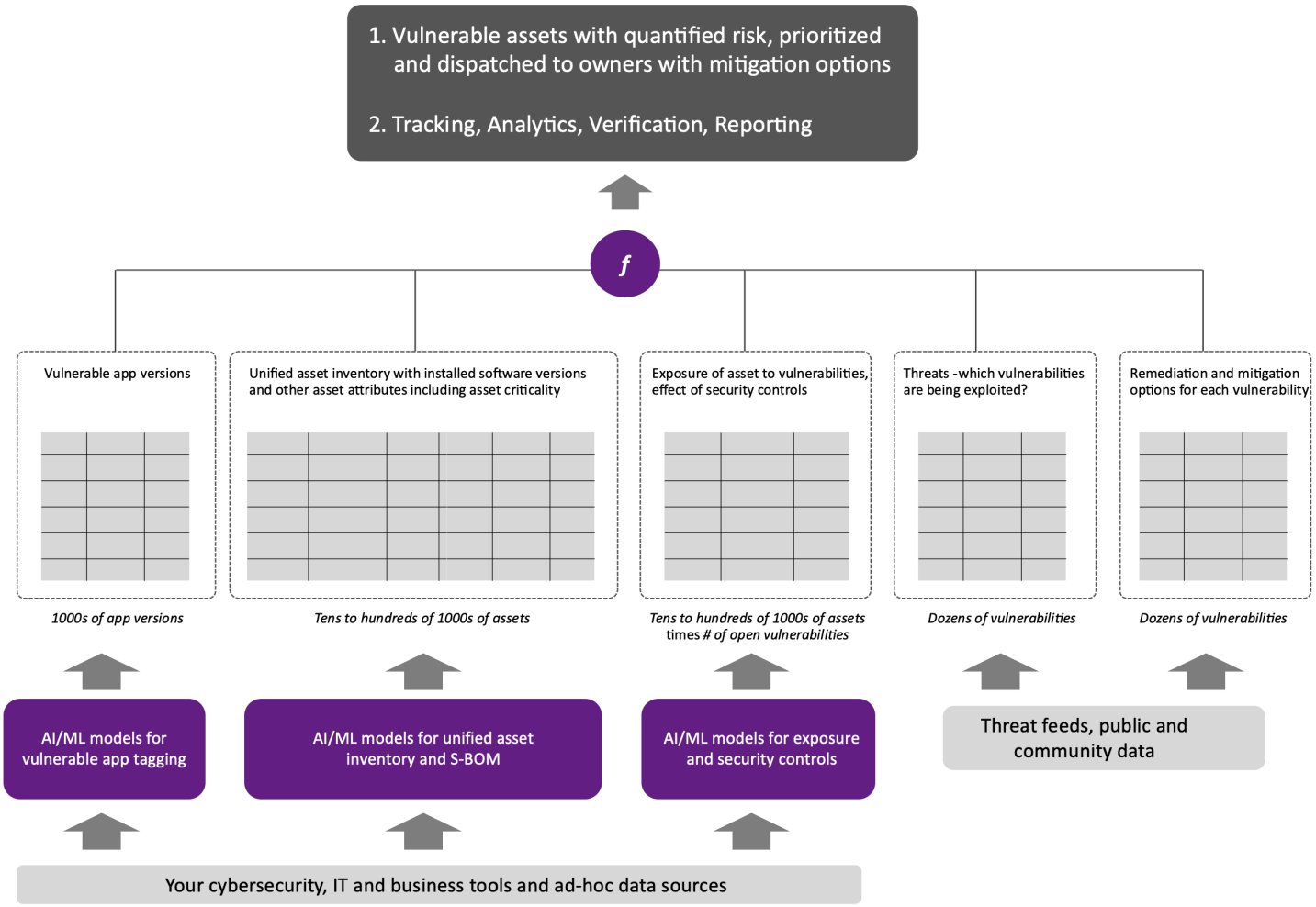


Figure 6: Data view of vulnerability mitigation

Step 1-6 outlined above are the elements of cybersecurity posture automation. **Figure 6** shows this as a data computation.

As you can see from the sizes of the data sets involved, this is not an easy calculation. In the Fortune 500, the sum of these data sets exceeds 100s of terabytes. If you want this computation to happen on a real-time and continuous basis, you will need an appropriate data platform. You need AI/ML models for dealing with incomplete, inconsistent, sometimes contradictory data across 100s of dimensions. Also, if you miss rows (or columns) in the various tables involved, you will have gaps in your visibility and have weak areas of your cybersecurity posture.

Invest in Cybersecurity Posture Automation

Hopefully, with the Log4j saga, it is evident to you that your organization needs to invest in a cybersecurity posture automation platform, such as Balbix. Some of your organizations might be trying to build this capability in-house. Log4j is a good test of how well that system performs.

The Balbix platform provides unified asset Inventory, continuous vulnerability analysis and prioritization, and cyber risk quantification while supporting the dispatch, mitigation and verification of vulnerabilities. In essence, Balbix enables you to do the calculations of **Figure 5** and **Figure 6**, with the mind map of **Figure 4**, implemented and automated in software.

At the time of writing, Balbix has helped our customers identify over 25,000 application versions vulnerable to Log4j including 1000s of custom apps. For each identified application version, we have quantified cyber risk and prioritized the app while providing auxiliary information that is key to mitigating or remediating the vulnerability. We have saved our customers tens of 1000s of hours trying to do the calculations of **Figure 6** by hand. For our customers, the Mind map of **Figure 4** is maximally automated as shown in **Figure 7**.

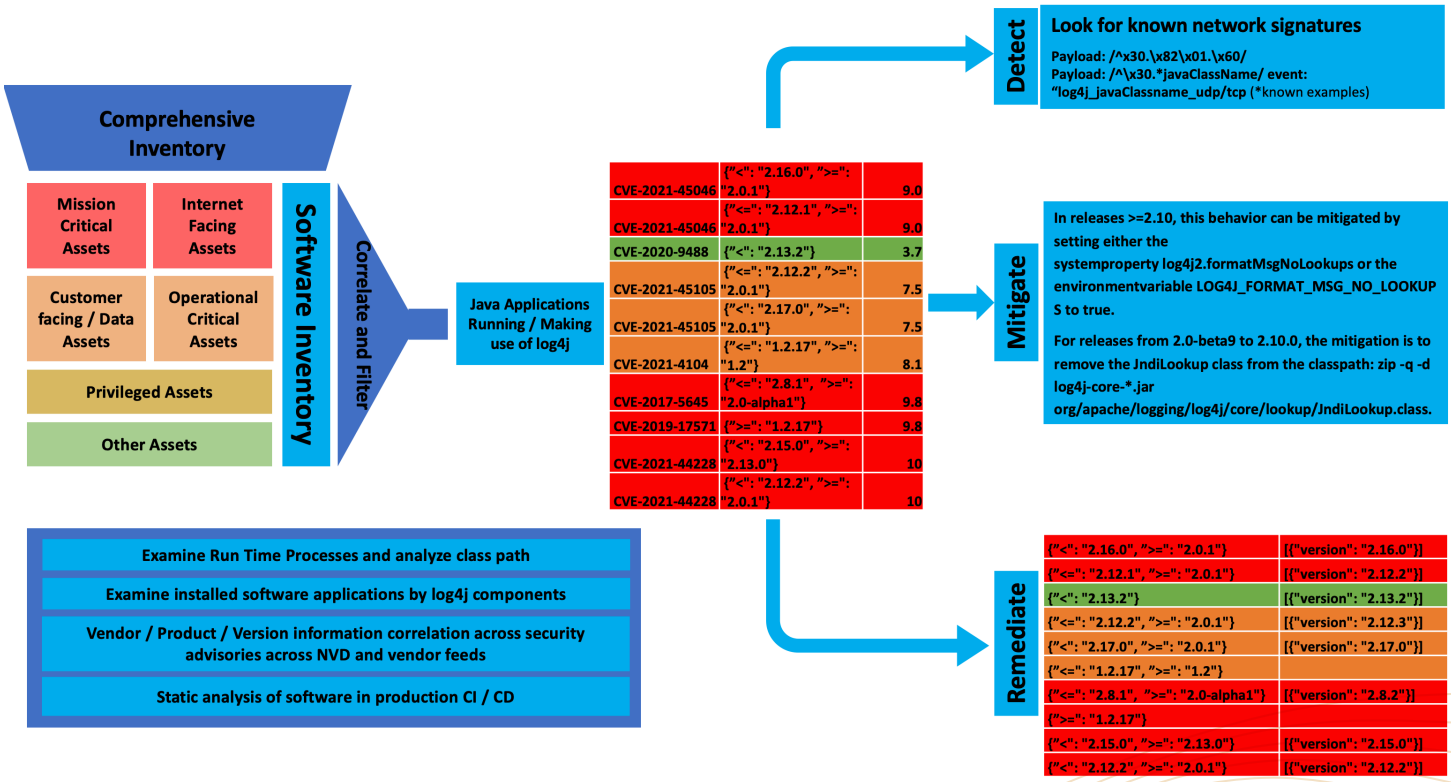


Figure 7: Systematically mitigating Log4j CVEs

Figure 8 is a view of the Balbix dashboard that a typical customer sees.

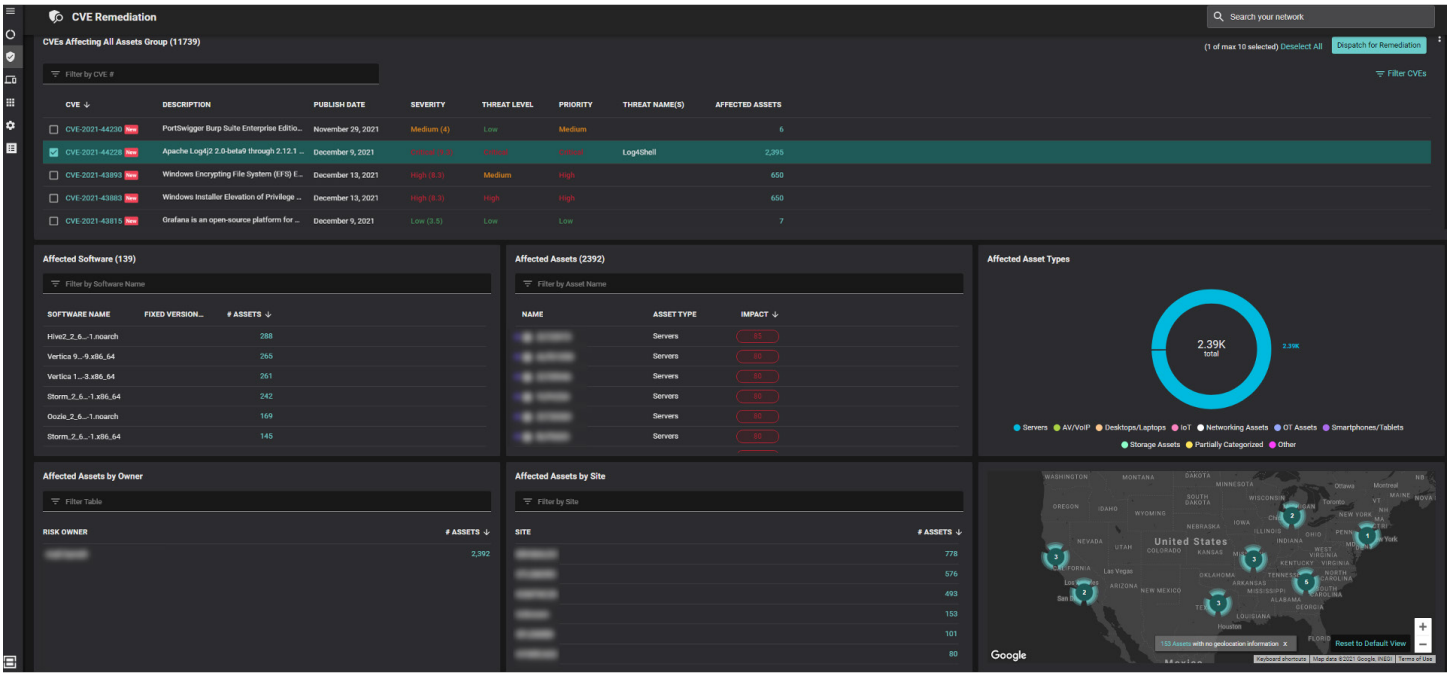


Figure 8: Sample Balbix dashboard for Log4j

A Log4j-type vulnerability will happen again. If you can consider cybersecurity posture and risk management as a data science problem, and you adopt an approach as outlined above, you'll mitigate cyber risk quickly and be ok. Otherwise, mitigating such issues will continue to ruin the day, or month, for you and your team (**Figure 9**).

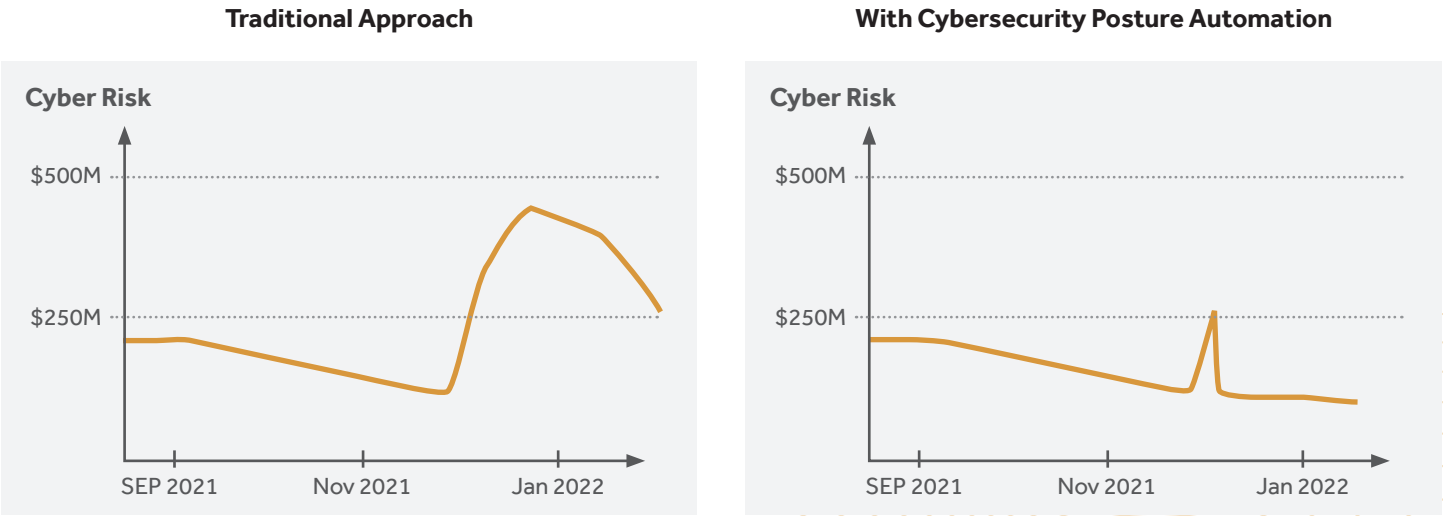


Figure 9: Cyber risk levels with speed of vulnerability mitigation

LEARN MORE

Balbix®