azul

2026

# AI Vulnerability Gap Report

## AI now finds software flaws faster than teams can patch them

**3 days**

or less for attackers to exploit

**4+ days**

for testing a patch alone

Survey of 300 U.S. security leaders, with a spotlight on Java

# TABLE OF CONTENTS


INTRODUCTION ..... 3

SAME DEFENSES, FASTER ATTACKERS ..... 7

THE PATCH CYCLE WAS DESIGNED FOR A DIFFERENT ADVERSARY ..... 10

THE JAVA ESTATE NOBODY ACCOUNTS FOR ..... 14

INTENT TO PATCH IS NOT A DEADLINE ..... 18

THE NEXT TWELVE MONTHS ..... 21

CONCLUSION ..... 24

ABOUT THIS SURVEY ..... 27

ABOUT AZUL ..... 30

# INTRODUCTION:

## THE RULES CHANGED, THE CALENDAR DID NOT

# INTRODUCTION: THE RULES CHANGED, THE CALENDAR DID NOT

Java runs the systems enterprises can least afford to lose: core banking, trading, e-commerce, identity and claims. It's long-lived, widely deployed and rarely scrutinized, which makes it a revealing lens on a problem that isn't unique to it. Half of the questions in this report cover software vulnerability and patching broadly. Half look at how the gap plays out in Java estates, where the stakes are the highest.

For most of enterprise history, finding and weaponizing a software vulnerability required scarce, specialized expertise. It was the work of nation-states and elite offensive research teams, and it took time. That time was the defender's margin. It is the reason a quarterly or even monthly patch cycle, a deployment window measured in weeks and a "best effort" Common Vulnerabilities and Exposures (CVE) policy were defensible security positions rather than liabilities.

That margin has collapsed. In April, Anthropic said Claude Mythos Preview had found thousands of previously unknown vulnerabilities in every major operating system and web browser, including a 17-year-old FreeBSD flaw that it says the model found and exploited without human involvement after the first prompt. In May, Google reported the first zero-day exploit it believes a criminal group built with AI. In July, autonomous agents in an internal OpenAI evaluation chained previously unknown vulnerabilities to escape their sandbox and breached Hugging Face, with roughly 1,200 agents sharing exploits on an improvised message board.

The most recent research shows exploitation often begins before a patch exists, rather than after one ships:

- Mandiant's M-Trends 2026 report estimates average time-to-exploit at negative seven days for 2025.
- Roughly 29% of the known vulnerabilities VulnCheck first saw exploited in 2025 were exploited on or before the day their CVE was published.

Vendors have responded. Oracle broke from its quarterly rhythm in May 2026, adding a monthly Critical Security Patch Update because AI had accelerated the speed and scale of vulnerability discovery. Enterprises have been unable to keep up and have even slid backward: Verizon's 2026 Data Breach Investigations Report (DBIR) found that the median time to fully remediate known exploited vulnerabilities rose from 32 days to 43 days.

Three clocks now govern every disclosed vulnerability: the attacker's, the vendor's and the enterprise's. Only the first two have accelerated. This report measures that exploitation gap: the distance between how fast an attacker can weaponize a disclosed vulnerability and how fast an organization can resolve it.

Azul commissioned Censuswide to survey 300 U.S. security leaders — CISOs, VPs and directors of information security, heads of security, security architects, heads of SecOps / DevSecOps, AppSec leaders, VPs and directors of IT security, and AI czars — at organizations where Java and Java Virtual Machines (JVM)-based runtimes make up a significant share of the production application estate. They were asked how fast they believe AI-equipped attackers can now weaponize a disclosure, how fast they can realistically deploy a fix, what stops them from quickly deploying a fix, and how much of their Java production runtime estate they can actually see.

## THE CENTRAL FINDING

44% of security leaders say their organizations still need four days or more to deploy a critical fix — longer than the three days or less that 69% believe an attacker needs. And 85% have already had an incident or a near miss because of slow patching.

Three things hold the gap open: testing that adds four or more days to Java runtime updates at 58% of organizations, no mandated deadline for patching non-critical vulnerabilities (66%), and a Java estate that about half concede they cannot fully see.

The implication is straightforward: patch velocity — the speed at which a security fix safely reaches every affected system — has become a first-order business risk metric.

## Why this should matter to the C-suite and the Board

- **Exploiting a known, unpatched software vulnerability is now the number one way enterprises get breached.** It accounted for 31% of initial breach vectors in the 2026 Verizon DBIR, up 55% year over year and ahead of credential abuse at 13%.

- **The financial exposure is documented.** IBM's Cost of a Data Breach Report 2026 put the average breach at a record $4.99M globally and $11.5M in the U. S., with roughly $1M added when the attack was AI-driven.

- **The compliance clock is shorter than most patch cycles.** Under Binding Operational Directive (BOD) 26-04 from the U.S. Cybersecurity and Infrastructure Security Agency (CISA), federal civilian agencies must remediate the highest-risk known exploited vulnerabilities in as little as three days, and the EU's GDPR requires organizations to notify regulators of a personal-data breach within 72 hours of becoming aware of it. A patch cycle measured in weeks is functionally irrelevant against those windows.

# KEY FINDINGS

## SAME DEFENSES, FASTER ATTACKERS



**69%** BELIEVE AI-EQUIPPED ATTACKERS CAN BUILD A WORKING EXPLOIT WITHIN THREE DAYS OR LESS.

**22%** PUT IT AT LESS THAN 24 HOURS.

## THE PATCH CYCLE WAS DESIGNED FOR A DIFFERENT ADVERSARY

**58%** SAY PRE-DEPLOYMENT TESTING ADDS **FOUR +** DAYS TO EVERY JAVA RUNTIME PATCH.

**22%** SAY IT ADDS A WEEK OR MORE.

**85%** OF ORGANIZATIONS HAVE HAD AN INCIDENT, A SUSPECTED INCIDENT, OR A NEAR MISS ATTRIBUTABLE TO SLOW PATCHING.

**ONLY 13%** CAN DEPLOY A PATCH FOR A CRITICAL VULNERABILITY IN UNDER 24 HOURS.



**44%** TAKE FOUR DAYS OR LONGER TO DEPLOY A PATCH.

## THE JAVA ESTATE NOBODY ACCOUNTS FOR



**ONLY 19%**

**HAVE 90%** OR MORE OF THEIR PRODUCTION JAVA RUNTIME ESTATE ON THE LATEST AVAILABLE SECURITY PATCH LEVEL.

**51% CONCEDE** THEY HAVE ONLY PARTIAL OR LIMITED VISIBILITY INTO THEIR CRITICAL JAVA RUNTIMES AND SUSPECT THERE ARE INSTANCES THEY CANNOT SEE.

**38%** PATCH ON A "BEST EFFORT" BASIS.

**36%** SAY LIMITED VISIBILITY TO KNOWN VULNERABILITIES IS ONE OF THEIR TOP 3 BARRIERS TO PATCHING QUICKLY.

## INTENT TO PATCH IS NOT A DEADLINE

**66%** HAVE NO MANDATED, TIME-BOUND COMMITMENT TO PATCHING NON-CRITICAL CVES.



**44%** CITE EXTENSIVE PATCH TESTING BEFORE DEPLOYMENT AS A BARRIER TO PATCHING QUICKLY.

## THE NEXT 12 MONTHS

**32%** NAME FASTER PATCHING AS THEIR PRIMARY STRATEGY FOR THE NEXT 12 MONTHS

**& 26%** NAME BETTER VISIBILITY AND INVENTORY.



# SAME DEFENSES, FASTER ATTACKERS

**LEADERS SAY AGENTIC AI RAISED THEIR EXPOSURE, BUT MOST HAVEN'T CHANGED THEIR POSTURE. ATTACKERS AREN'T WAITING.**

# SAME DEFENSES, FASTER ATTACKERS

**Leaders say agentic AI raised their exposure, but most haven't changed their posture. Attackers aren't waiting.**

## Virtually every leader sees the risk. Yet 59% have not changed their posture.

Virtually every leader surveyed said the rise of AI-powered attack tools has increased their security exposure over the past 12 months [Figure 1]. They differ in how much and in what that has prompted so far. 41% say the increase was significant enough that they have already changed their security posture. 34% call it moderate – they are reassessing the risk, but no major changes yet. And 25% say it has increased only slightly and sits on the radar without being a priority. Net: 59% report an increase that has not yet changed their posture.

In other words, most enterprises have largely the same defenses they started with 12 months ago, but are now facing an adversary that spent the year improving their offensive capabilities.

**Over the past 12 months, how, if at all, has the rise of AI-powered attack tools mainly changed your organization's view of its vulnerability exposure?**

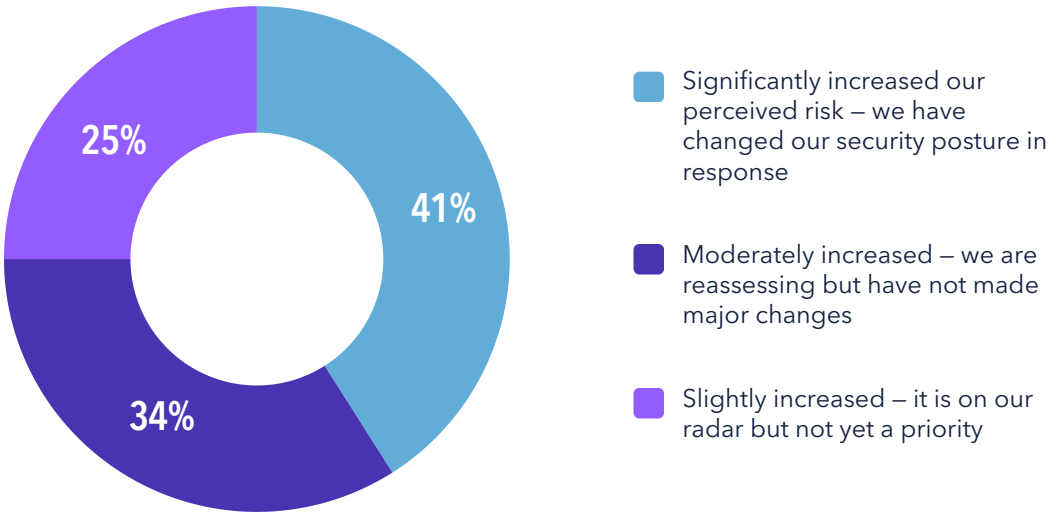



*Figure 1: 59% of security leaders acknowledge that they have not changed their security posture despite also acknowledging AI-powered attack tools have increased their security exposure over the past 12 months.*

# 69% of security leaders say AI-based vulnerability weaponization takes three days or less

Asked how quickly AI-equipped attackers can now develop a working exploit after a critical vulnerability is publicly disclosed, U.S. security leaders gave a strikingly compressed answer. Nearly half – 47% – say one to three days. More than one in five, or 22%, say less than 24 hours [Figure 2].

If anything, that is conservative. The documented figures run faster: Mandiant's M-Trends 2026 estimates average time-to-exploit at negative seven days for 2025, down from negative one day for 2024, meaning exploitation typically begins before a patch is available. Separately, roughly 29% of the known exploited vulnerabilities VulnCheck first saw exploited in 2025 were exploited on or before the day their CVE was published. Where leaders estimate three days or less, the evidence for a large share of vulnerabilities says zero or less.

## THE BENCHMARK

Three days is the generous estimate. 69% of leaders say AI-equipped attackers can build a working exploit in three days or less, and Mandiant estimates that exploitation now begins, on average, before a patch exists. Every patching deadline, testing regime and deployment window should be measured against that clock, not the calendar it was designed around.

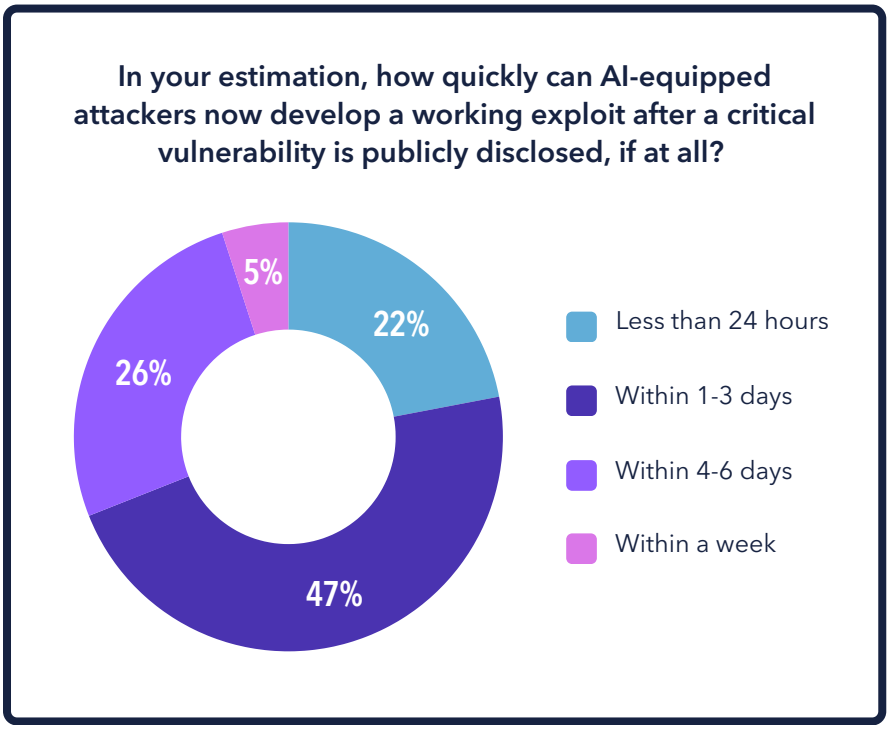



*Figure 2: 69% of security leaders say that AI-equipped attackers can develop a working exploit in three days or less.*

These estimates – and every deployment figure in this report – describe vulnerabilities that have been publicly disclosed and for which a patch exists. Zero-day and negative-day exploitation sits outside that frame entirely: no deployment speed addresses a flaw that has not been disclosed and for which no fix has been written. The gaps measured here are the ones an organization can close, but they are not the whole of the exposure.

# THE PATCH CYCLE WAS DESIGNED FOR A DIFFERENT ADVERSARY

**DEPLOYMENT SPEED, THE TESTING BURDEN THAT PRECEDES IT, AND THE INCIDENTS THAT HAVE ALREADY FOLLOWED.**

# THE PATCH CYCLE WAS DESIGNED FOR A DIFFERENT ADVERSARY

**Deployment speed, the testing burden that precedes it, and the incidents that have already followed.**

## The gap between patch and attack

Asked how long it realistically takes to deploy an available patch for a critical vulnerability across production, only 13% of security leaders said less than 24 hours [Figure 3]. 43% said one to three days and 32% said four to six days.

In other words, 69% of leaders believe an exploit exists within three days, yet only 56% of their organizations can get a critical fix out in that time. The other 44% need four days or more – which suggests that, on their most urgent vulnerabilities, many fixes will arrive after the attacker does if leaders' own estimates of attacker speed are right.

Note that those numbers are optimistic given Verizon's 2026 DBIR, which measured a median of 43 days to fully remediate known exploited vulnerabilities. The survey figures also assume three things are true at once:

1. **A patch already exists.** By definition, this assumption fails for every zero-day vulnerability.
2. **The vendor has already shipped their patch.** While Oracle ships their JDK patch on day zero, very few OpenJDK vendors are able to consistently ship their own patch on day zero.
3. **The affected systems sit in the part of the Java estate the organization can see.** This assumes the enterprise has a current, accurate map of which runtimes and versions are affected. About half of survey respondents concede they cannot see all of their Java estate.

Any of these assumptions can invalidate the responses.

### THE GAP, QUANTIFIED

Within 24 hours: 22% of leaders say an attacker can have a working exploit. 13% of organizations can have a critical patch deployed.

Within three days: 69% say an attacker can weaponize a vulnerability. 56% of organizations can deploy a fix.

For the other 44%, the fix takes four days or more.

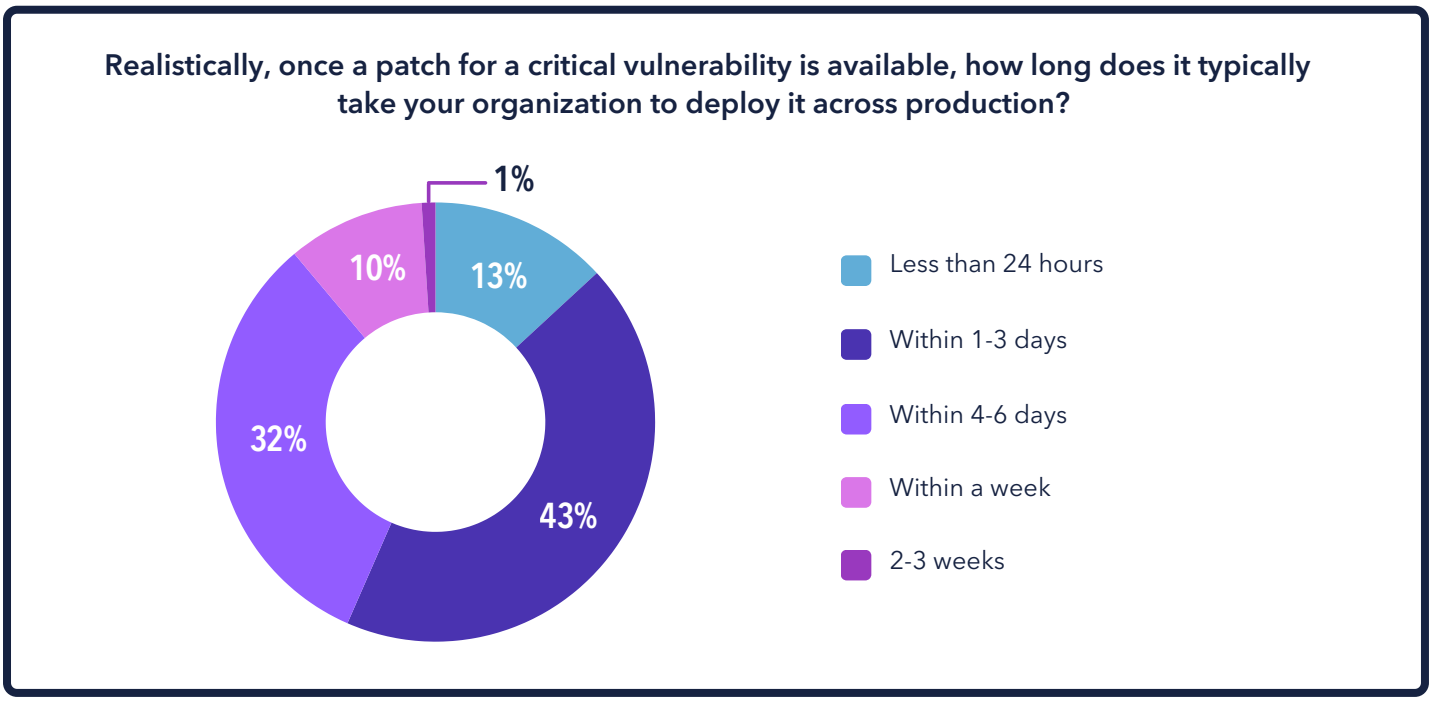



*Figure 3: Only 13% of security leaders said that their organization could deploy a patch for a critical vulnerability in under 24 hours once that patch is available.*

# The Java testing tax: four or more days before deployment begins

The deployment stats above answer how long it takes to get a patch from available to running in production. But there's one stage that precedes or overlaps with deployment: testing. Respondents may have read "deploy" as starting after testing ends, which means the exposure window is longer than the deployment figures alone suggest. Asked how long pre-deployment testing typically adds to their patch cycle for Java runtime updates, 36% say four to six days, 33% say one to three days and 19% say one to three weeks [Figure 4]. Only 9% say less than 24 hours, and no organization skips testing altogether. With Java security updates now shipping monthly as well as quarterly, fixes are reaching enterprises faster than ever — so the constraint is no longer when a patch arrives, but how much work each one creates before it can go live.

## THE TESTING TAX

58% add four or more days of testing to every Java runtime patch. 22% add a week or more. Not one respondent skips testing.

For the 58%, testing alone outlasts the three days or less that most leaders believe an attacker needs. Cutting it swaps potential breaches for potential performance or functional issues that may result in outages.

According to Azul's assessment of OpenJDK releases over the past 6+ years, 23% of quarterly updates (6 in 26) have caused a serious regression issue. That leaves enterprises with a significant challenge of trading off security vs. stability.

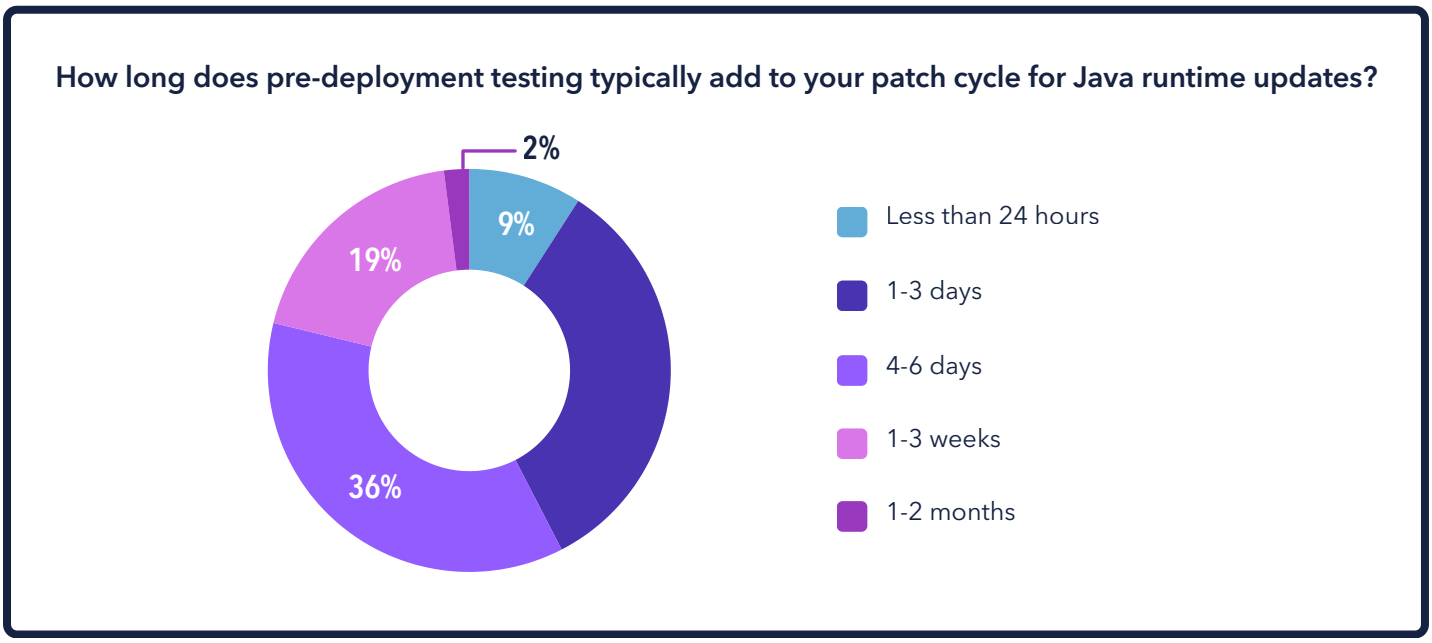



*Figure 4: 58% of organizations delay deployment by adding four days of testing or more to every JDK update.*

# 85% have already had an incident or a near miss

The previous two sections measured how long testing and deployment take. This one measures the cost when testing and deployment take too long. 85% of security leaders say their organization has experienced an incident or a near miss attributable to slow patching. 55% report a confirmed incident and 53% report a near miss [Figure 5]. Only 9% say they have never experienced any of the above.

Slow patching has already caused incidents at scale, even before attackers had fully industrialized agentic AI. That 85% is likely an undercount: attributing an incident to a missed patch takes someone to connect the two, which happens readily in systems with clear owners and full monitoring, and rarely in the parts of the Java estate nobody can fully account for. You cannot trace an incident back to an unpatched system you did not know you had.

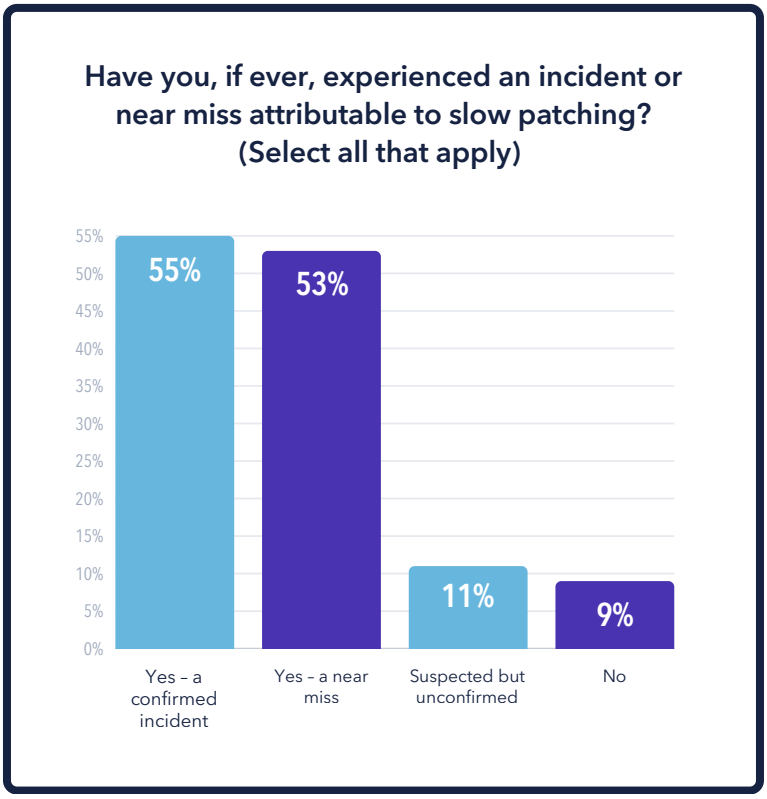



*Figure 5: 85% of organizations have experienced an incident or a near miss due to slow patching.*

# THE JAVA ESTATE NOBODY ACCOUNTS FOR

## PATCH CURRENCY AND VISIBILITY ACROSS THE JAVA RUNTIME ESTATE.

# THE JAVA ESTATE NOBODY ACCOUNTS FOR

**Patch currency and visibility across the Java runtime estate.**

## "Mostly patched" is the normal state

Java runs underneath many of an enterprise's most critical business applications, and as teams, acquisitions and cloud migrations have multiplied the number of runtimes, few organizations have a complete picture of where they all are. It runs reliably, and because it rarely causes problems, it often gets less scrutiny than it deserves.

Asked what portion of their production Java runtime estate is on the latest available security patch level, only 19% said nearly all of it – 90% or more [Figure 6]. 59% said between 60% and 89%, 19% said about half, and 2% said less than 40%. Net: 81% have less than 90% of their estate on the current patch level, and 21% are at half-current or worse. Averaged across response band midpoints, U.S. organizations report that roughly 72% of their production Java runtime estate is on the latest security patch. That sounds pretty good until you read it the way an attacker reads it: in a 3,000-JVM estate, a 72% average means roughly 840 runtimes are sitting below the current security baseline in production.

### WHAT THE AVERAGE HIDES

Only 19% have 90% or more of their production Java estate on the latest security patch. The average organization has about 72% of their estate up to date, which means in a 3,000-JVM estate, roughly 840 runtimes may carry known, unpatched vulnerabilities.

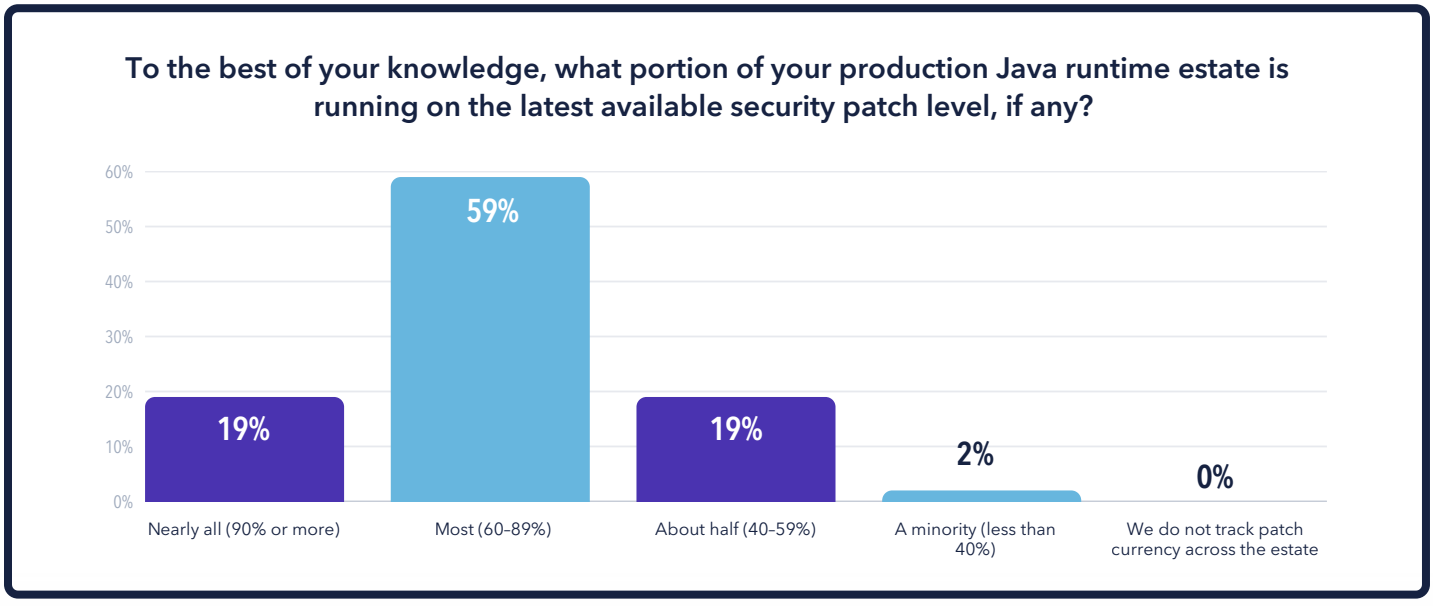



*Figure 6: 81% of organizations have less than 90% of their production Java estate on the latest security patch level.*

In Azul's experience, four patterns recur across large Java estates and explain how the drift accumulates:

- End-of-life versions such as Java 6 and 7 persist in production long after free public updates end, but most enterprises are unaware that Azul continues to provide security fixes for them.
- Many JVMs sit well behind the current baseline because organizations take a "Not broken? Don't fix it" attitude.
- Other JVMs are believed to be secure enough since they run behind a firewall.
- Existing IT asset tools miss what they were never designed to find: runtimes installed by teams outside IT, runtimes bundled inside containers, and runtimes baked into golden-master images at provisioning time and never touched since.

## The Java runtimes nobody has inventoried

Asked how much visibility they have into every instance of their critical Java runtimes and JVMs in production — explicitly including embedded, legacy and unmanaged instances — 49% claim full visibility, while 51% concede partial or limited visibility, admitting that they suspect there are instances they cannot see [Figure 7].

While 49% of enterprises claim they have complete Java runtime visibility, it's likely overly optimistic for three reasons:

**1. Self-reported completeness is self-confirming.**

After all, missing instances are invisible to the tool that should report them.

**2. Systems of record are passive.**

"Every instance — including embedded, legacy, and unmanaged" describes exactly the runtimes outside the systems of record. A configuration management database (CMDB) or IT asset management (ITAM) tool can be complete with respect to everything it was designed to track and still miss a JVM bundled inside a vendor appliance, a dynamically built container, or an application server nobody has logged into in years.

**3. It sits awkwardly against the previous finding.**

81% of the same respondents report an estate that is not fully patch-current. Patch drift can coexist with full visibility, but given the rising awareness of the risk, it's harder to explain how an organization sees every runtime and still has so much of its estate below the current patch level.

### THE EXECUTIVE TEST

49% of security leaders claim full visibility. A simple test can check the claim. Ask for the current inventory of production runtimes — version, patch level and named owner — and ask how it is generated. If the answer is a spreadsheet, a CMDB with manual entry, or an ITAM tool that was never designed to detect embedded runtimes, the inventory is almost certainly incomplete, whatever the organization believes it has.

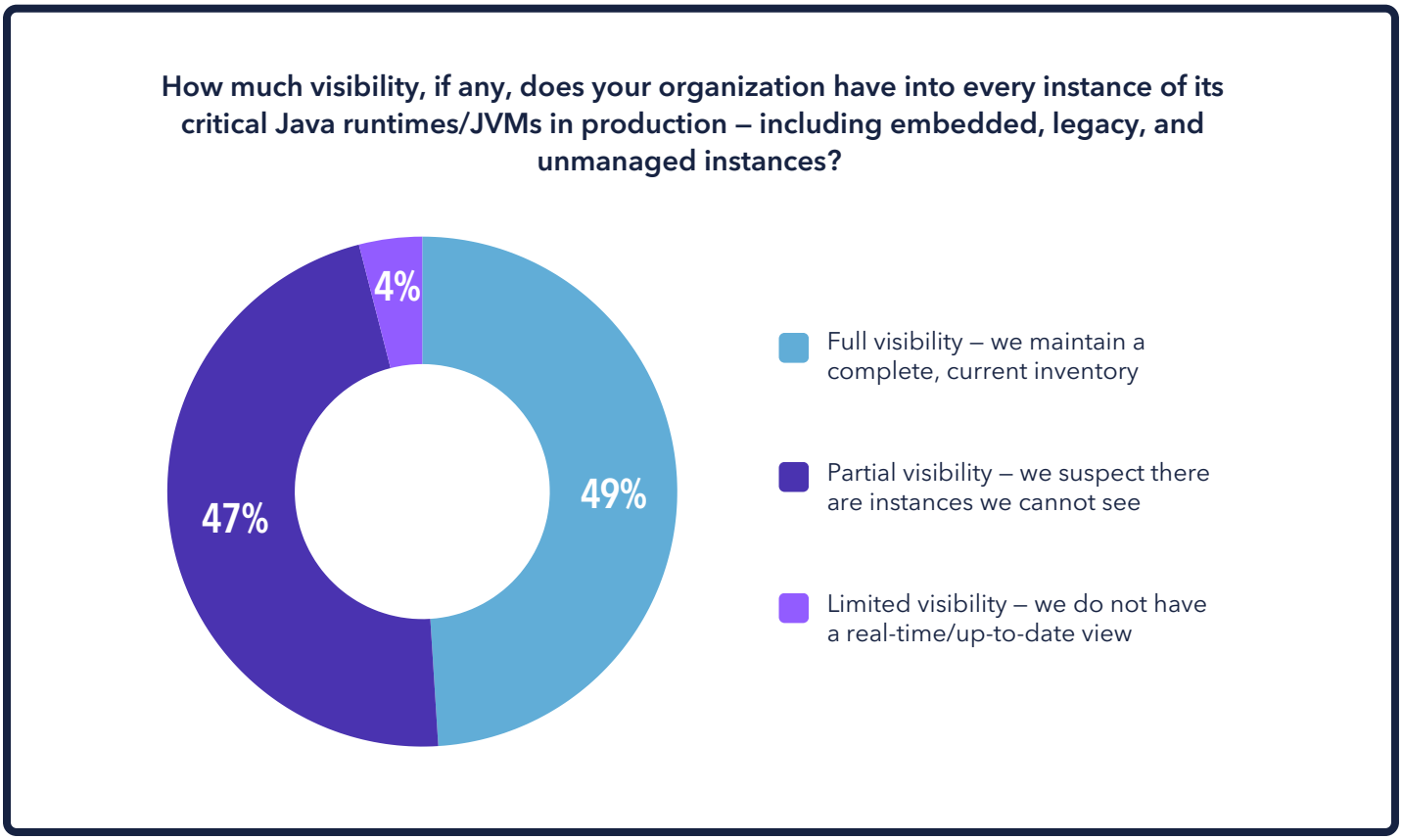



*Figure 7: 51% of security leaders note that their organization has partial or limited visibility into its critical Java runtimes.*

An incomplete inventory matters more than it used to. When exploitation followed disclosure by weeks, it was a hygiene problem you could resolve during the response. When exploitation precedes disclosure, the inventory has to already exist. You cannot patch what you have not found, and you no longer have time to go looking.

# INTENT TO PATCH IS NOT A DEADLINE

**LACK OF A POLICY PERMITS THE GAP, AND BARRIERS SUSTAIN IT.**

# INTENT TO PATCH IS NOT A DEADLINE

**Lack of a policy permits the gap, and barriers sustain it.**

## Two-thirds of U.S. enterprises have no mandated deadline for patching non-critical vulnerabilities

Critical vulnerabilities are, by definition, the ones that get priority. However, it's the low-risk CVEs that AI agents can chain together to create a critical outcome. For example, in the OpenAI Hugging Face incident, a chain of at least eight low-risk CVEs let agents escape a sandbox and reach the internet.

In many organizations, low-risk vulnerabilities do not have a remediation deadline. 38% of organizations patch non-critical CVEs on a "best effort" basis, when resources and testing windows allow [Figure 8]. 33% mandate a defined SLA, 25% take a risk-based approach and patch only what is judged high priority, and 3% have no formal policy at all. Net: 66% have no time-bound commitment.

### THE LIABILITY

66% have no time-bound commitment for patching non-critical vulnerabilities. "Best effort" (38%) is the most common approach, and the hardest to defend after an incident.

Triage is still required, and at current volumes unavoidable, especially given that CISA added roughly a third more Known Exploited Vulnerabilities (KEV) catalog entries in 2025 than in 2024. But ranking CVEs assumes severity only needs to be associated with each individual vulnerability. Chaining of CVEs breaks that assumption, and most triage solutions remain blind to it. KEV entries and internet-facing systems should still go first. The problem is what happens to all the lower-risk CVEs, where two-thirds of these organizations have committed to no deadline [Figure 8].

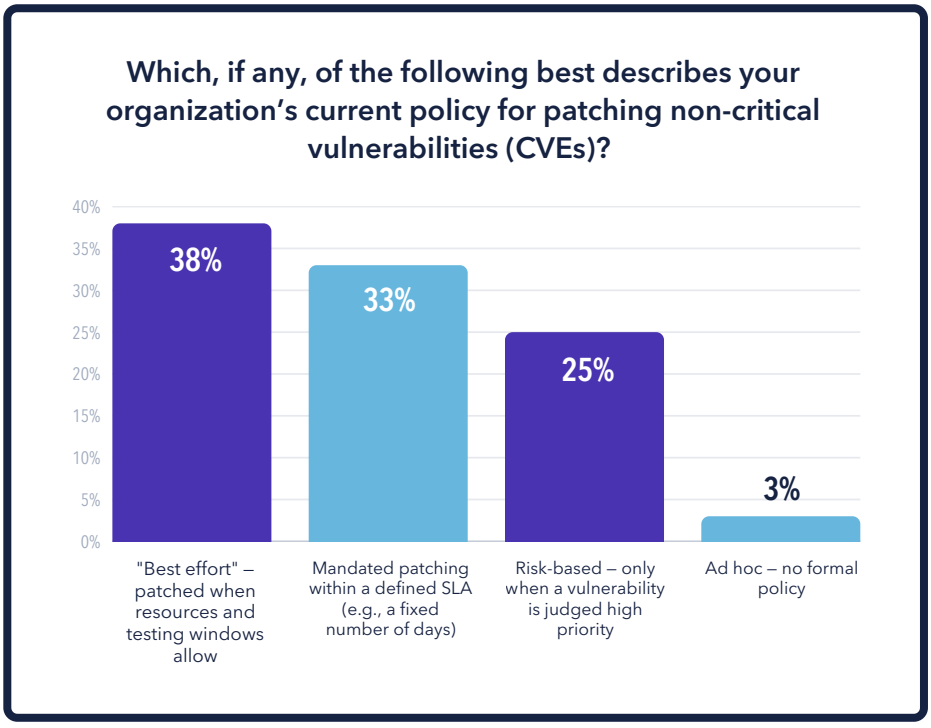



*Figure 8: 38% of organizations patch non-critical CVEs on a "best effort" basis.*

# Only one barrier to patching is outside the organization's control

Asked to name their three biggest barriers to patching quickly, security leaders described the limitations imposed by their own organizations. The three most-cited: extensive patch testing required before deployment to mitigate regressions or breaking changes (44%), the complexity or age of the affected systems (40%), and limited visibility into known vulnerabilities across affected systems (36%) [Figure 9]. Delays in receiving patches from software or runtime vendors tied for last at 25%, which is the only barrier on the list that sits outside the enterprise.

The top barrier is the testing delay measured in Figure 4. None of these barriers respond to pressure. Testing exists for good reason – a bad patch in a core system can potentially take the business down, and as previously stated, regression risk is real and well evidenced. The way to go faster is to give teams far fewer changes to test in the patch.

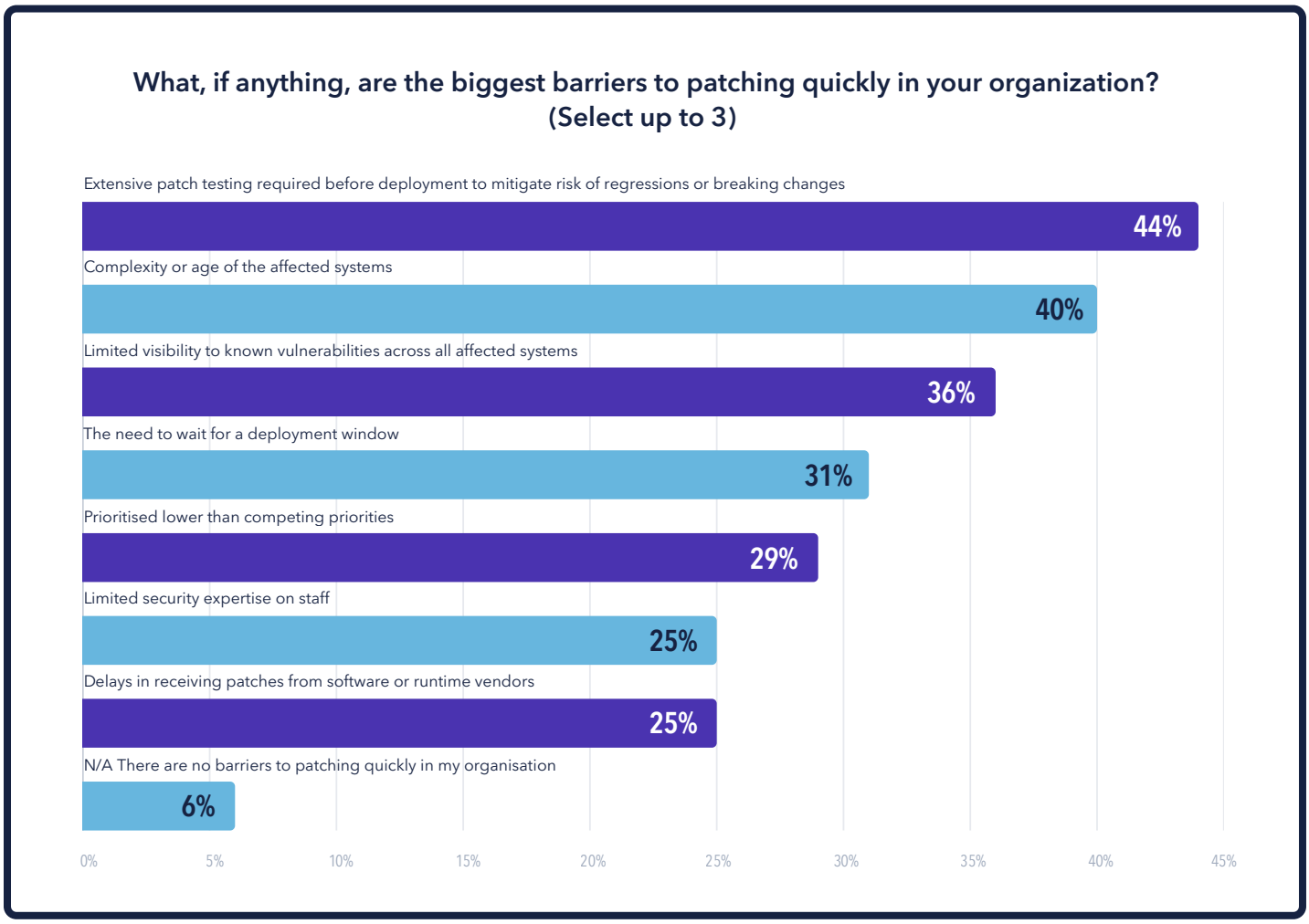



*Figure 9: The top three barriers to patching quickly include extensive testing required before deployment (44%), complexity or age of the affected system (40%) and limited visibility to known vulnerabilities (36%).*

# THE NEXT TWELVE MONTHS

## HOW SECURITY LEADERS INTEND TO CLOSE THE GAP BETWEEN DISCLOSURE AND PATCH.

# THE NEXT TWELVE MONTHS

**How security leaders intend to close the gap between disclosure and patch.**

## Where U.S. security leaders are placing their bets

Asked for their primary strategy for reducing vulnerability exposure over the next 12 months, security leaders gave three answers above all others:

- 32% – increasing the speed and frequency of patching across the estate to minimize the exploitable attack surface.
- 26% – improving visibility and inventory of all software and runtimes in production.
- 17% – investing in detection of, and response to, exploits rather than faster patching [Figure 10].

Patching speed is the top priority, and adding visibility/inventory totals 58% of the plan. Those are the two problems this report traces end to end: you cannot patch what you have not found, and finding it does not help if deployment takes longer than exploitation.

Detection and response is an essential investment, but not an alternative to patch velocity. Against an adversary that can weaponize a disclosure in three days or less, the 17% who intend to detect and respond to an exploit rather than block it with faster patching are betting they will notice the intrusion in time. This is a materially harder problem than just closing the hole.

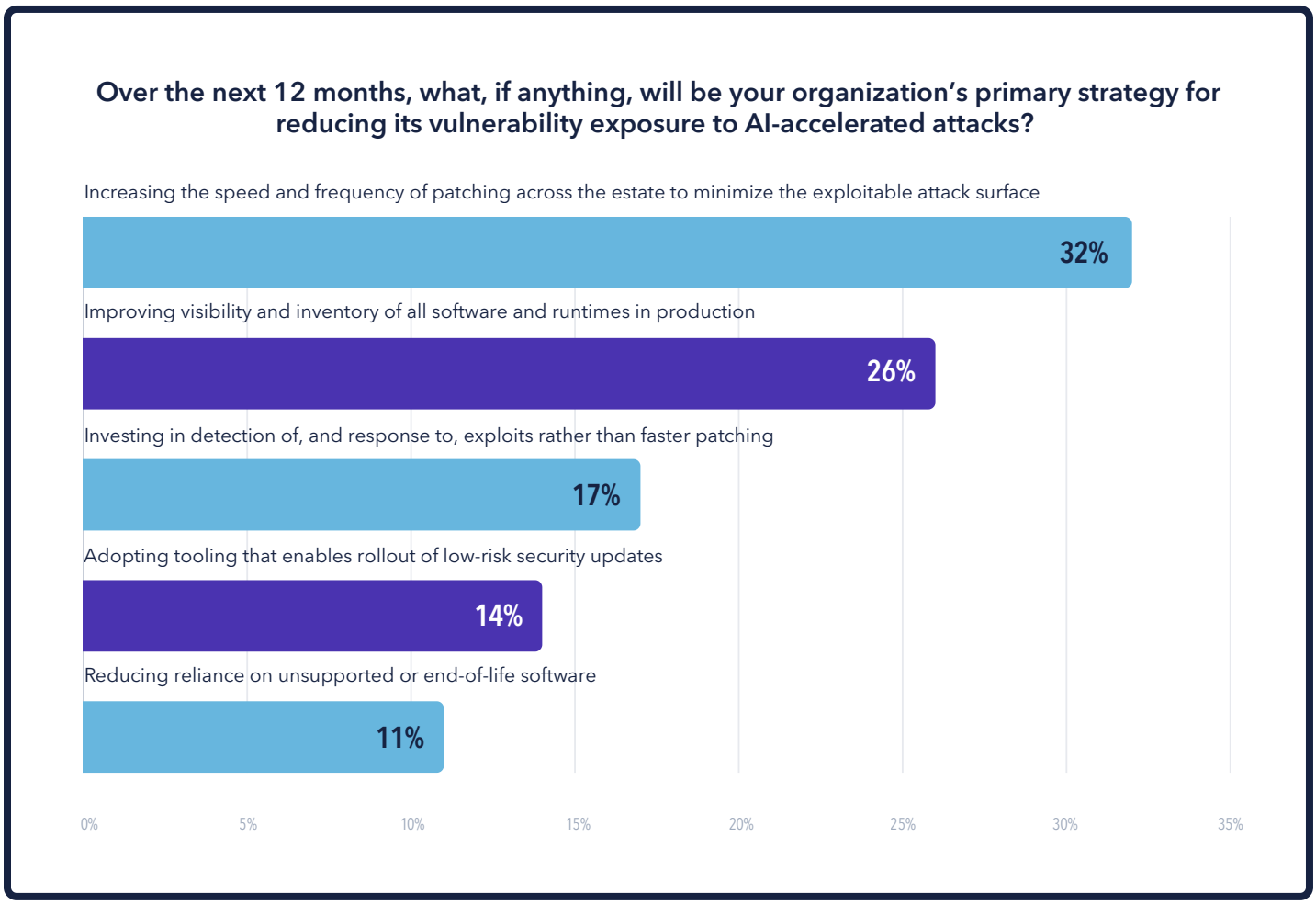



*Figure 10: Increasing the speed and frequency of patching across the estate to minimize the exploitable attack surface is the primary strategy for reducing an organization's vulnerability exposure.*

Three out of every four organizations name one of these strategies, and 85% have already had an incident or a near miss. In other words, leaders are planning for success based on past precedent, but the measure that matters will not be whether a plan existed. It will be how many days the last critical patch took to reach production.

# CONCLUSION:

**PREPARATION DOES NOT BEAT A ZERO-DAY. BUT IT DOES DECIDE HOW LONG YOU STAY EXPOSED**

# CONCLUSION

**Preparation Does Not Beat a Zero-Day. But It Does Decide How Long You Stay Exposed**

Security leaders are not unaware. 69% have correctly identified that adversaries can now weaponize a disclosed vulnerability in three days or less. But their patching strategy is still late. 44% take four days or more to deploy a critical patch. 58% spend four days or more on Java runtime testing alone, before deployment begins. 66% have no mandated deadline for patching non-critical vulnerabilities. 51% concede partial or limited visibility into their production Java estate. And 85% have already had an incident or a near miss because of it.

The bottom line is that awareness is no longer the gap, execution is. Attackers will only keep getting faster. But every day between a fix becoming available and a fix reaching production is a day an organization's own testing, deadlines and inventory leave open. Those are the only days it controls, and the only ones it can win back.

## What this means for the C-suite

### 1. Patch velocity is now a board-level risk metric

Vulnerability exploitation is the leading way enterprises now get breached (Verizon 2026 DBIR), and slow patching has already produced a confirmed incident at 55% of U.S. enterprises [Figure 5]. Yet patch velocity rarely reaches the board the way uptime and audit findings do. Even when it does, it is usually measured against an inventory everyone knows is incomplete. Three questions a board can ask: How many days does an available patch take to reach every system it affects? How much of the estate is on the latest security patch? And how much of that answer rests on an inventory nobody has to take on trust?

### 2. "Best effort" is a liability position, not just a risk posture

38% of U.S. organizations patch non-critical CVEs when resources and testing windows allow [Figure 8]. When an incident does occur, one of the first questions is likely to be whether the organization had a stated remediation window, and whether it met it. Under CISA's BOD 26-04, federal civilian agencies can have as little as three days to remediate the highest-risk known exploited vulnerabilities, and the GDPR gives 72 hours from becoming aware of a personal-data breach to notify the regulator. A policy that provides no interval is hard to defend as a control against clocks like those, which makes this a general counsel conversation as much as a CISO one.

## 3. The uninventoried Java estate is an audit failure waiting to happen

51% of U.S. security leaders cannot confirm a complete inventory of their production Java runtimes [Figure 7]. That gap does more than slow patching: it undermines risk reporting, breach scoping, regulatory response and license position. When something is exploited in the part of the estate nobody inventoried, the finding is not "we were slow." It is "we did not know it was there" — much harder to explain to a board, a regulator or a customer.

Vendors have done their part — Java security updates now ship monthly as well as quarterly. But average time-to-exploit is now measured in negative time (M-Trends 2026), which means vulnerabilities are often being exploited before a patch exists. And with zero-days, there is simply nothing to deploy. No governance framework, inventory or testing regime changes that. No product or process makes an organization safe from zero-days, and any claim that one does deserves scrutiny.

That doesn't mean organizations should just give up. Proactive organizations that keep current with patching have a Java estate characterized by fewer unpatched known flaws, which means fewer footholds for agentic AI to assemble. Stringing together low-risk, individually unremarkable vulnerabilities is what these systems do best, and at least eight were enough to escape a sandbox in the OpenAI–Hugging Face incident. When a disclosure lands, knowing where you are affected is either a query that can be answered in minutes or a three-week project. And the organizations that close the window faster are the ones that have already adopted best practices when it comes to Java estate visibility and minimizing its attack surface.

So the question is not whether your Java estate is bulletproof. Against a flaw with no patch, no estate is ready. The question is how much of it is exposed when the next disclosure lands, and how long it stays that way. That is decided long before the disclosure by how much of the estate you can see, and how fast a fix can move through it. That distance is the exploitation gap. Attackers set one side of it. You set the other.

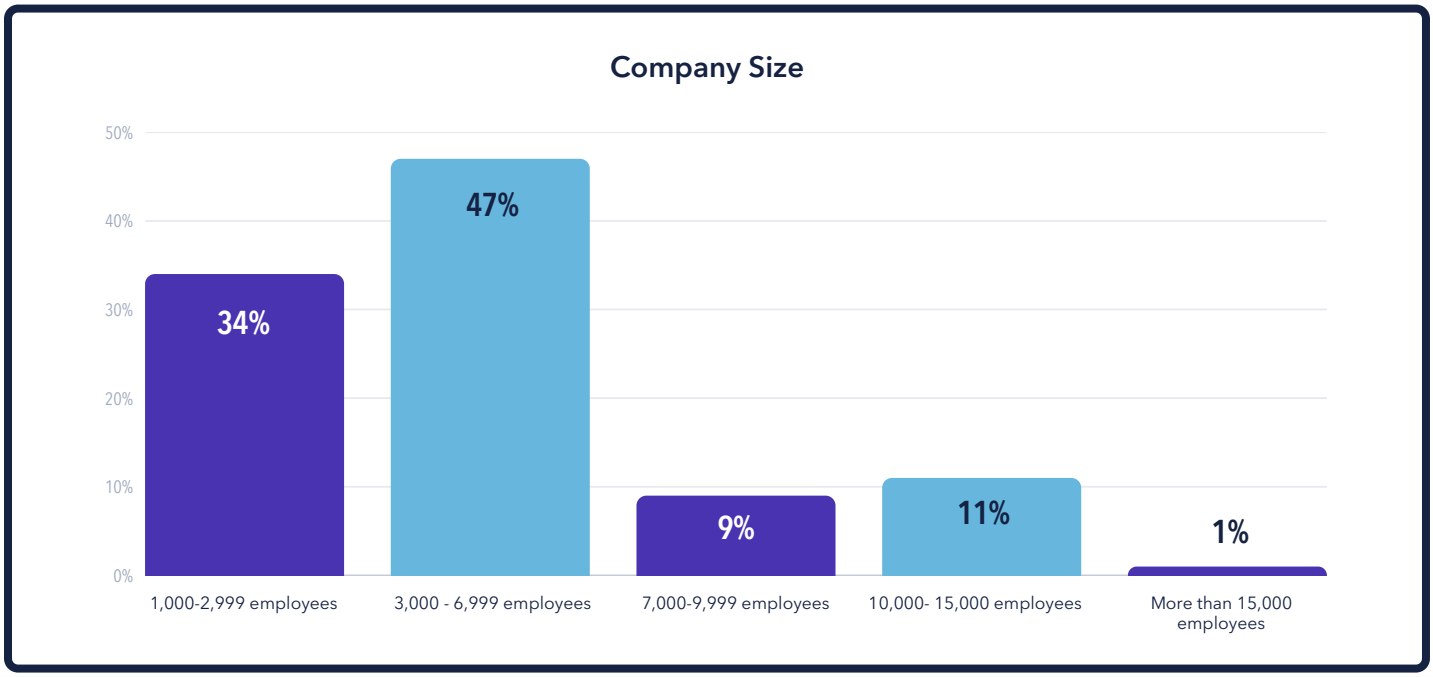
# ABOUT THIS SURVEY

# ABOUT THIS SURVEY

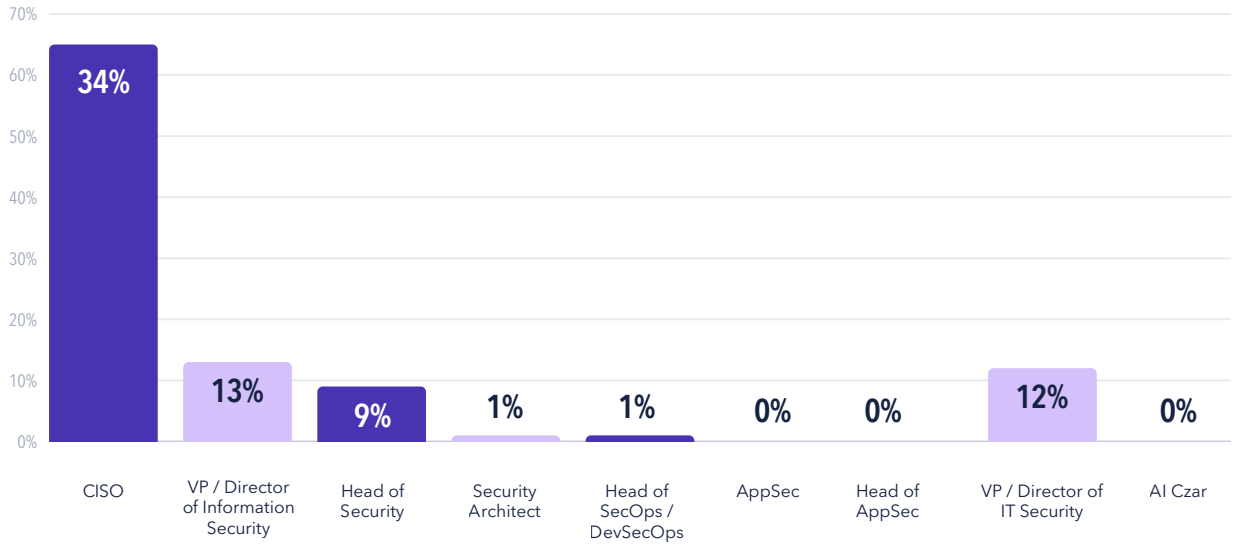
This report presents United States data only. The research was conducted by Censuswide among a sample of 300 U.S. security leaders – CISOs, VPs and directors of information security, heads of security, security architects, heads of SecOps / DevSecOps, AppSec leaders, VPs and directors of IT security, and AI Czars – at organizations with 1,000 or more employees. All respondents have direct responsibility for, or significant influence over, cybersecurity strategy, vulnerability management or patching policy, and all qualified on a screener confirming that Java/JVM runtimes make up a significant share of their organization's production application estate.

Fieldwork was conducted between August 3 and August 7, 2026. The wider study covered 919 security professionals across four markets; this report presents the U.S. sample only, and every survey figure cited is a U.S. figure. International data is not reported here.

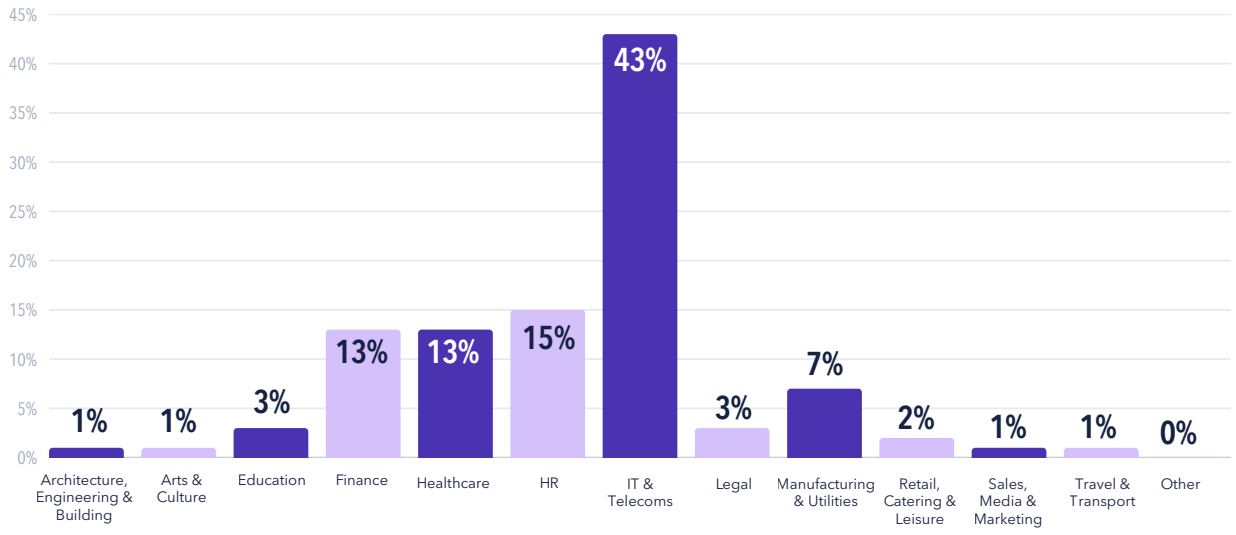
Censuswide is a member of the Market Research Society (MRS) and the British Polling Council (BPC), and a signatory of the Global Data Quality Pledge. The research adheres to the MRS Code of Conduct and ESOMAR principles. Percentages are rounded to the nearest whole number and may not total 100; where the response options cited for a question do not sum to 100, the balance sits in options not reported here. Where a figure is reported as 100%, the unrounded value is 99.7%. "Net" figures combine all applicable answer options. Questions permitting multiple selections are noted in the relevant figure caption.



## Job Title




## Sector



# ABOUT AZUL

Azul is the trusted leader in enterprise Java for today's AI-first world. Its open source-based Java platform empowers organizations to optimize the entire Java lifecycle to accelerate performance, strengthen security, reduce licensing and cloud costs, and boost developer productivity. Azul powers mission-critical systems for 37% of the Fortune 100, 50% of the Forbes Top Ten World's Most Valuable Brands, and the world's top 10 financial trading companies. Learn more at azul.com and follow @azulsystems.

# SECURITY-ONLY, STABILITY-FIRST™

Agentic AI has shrunk the window between vulnerability disclosure and exploitation from months to days, but the typical enterprise patch cycle still runs weeks. Azul closes that gap with its Security-only, Stability-first™ quarterly, monthly and out-of-cycle updates that contain security fixes and nothing else: no new features, enhancements or unrelated changes that can introduce regressions. Each update is built on the prior quarter's full patch update, already tested by Azul and the broader Java community, so it can go straight to production without a testing delay.

azul

385 Moffett Park Drive, Suite 115, Sunnyvale, CA 94089 USA
+1.650.230.6500 | www.azul.com
Copyright © 2026 Azul Systems, Inc.