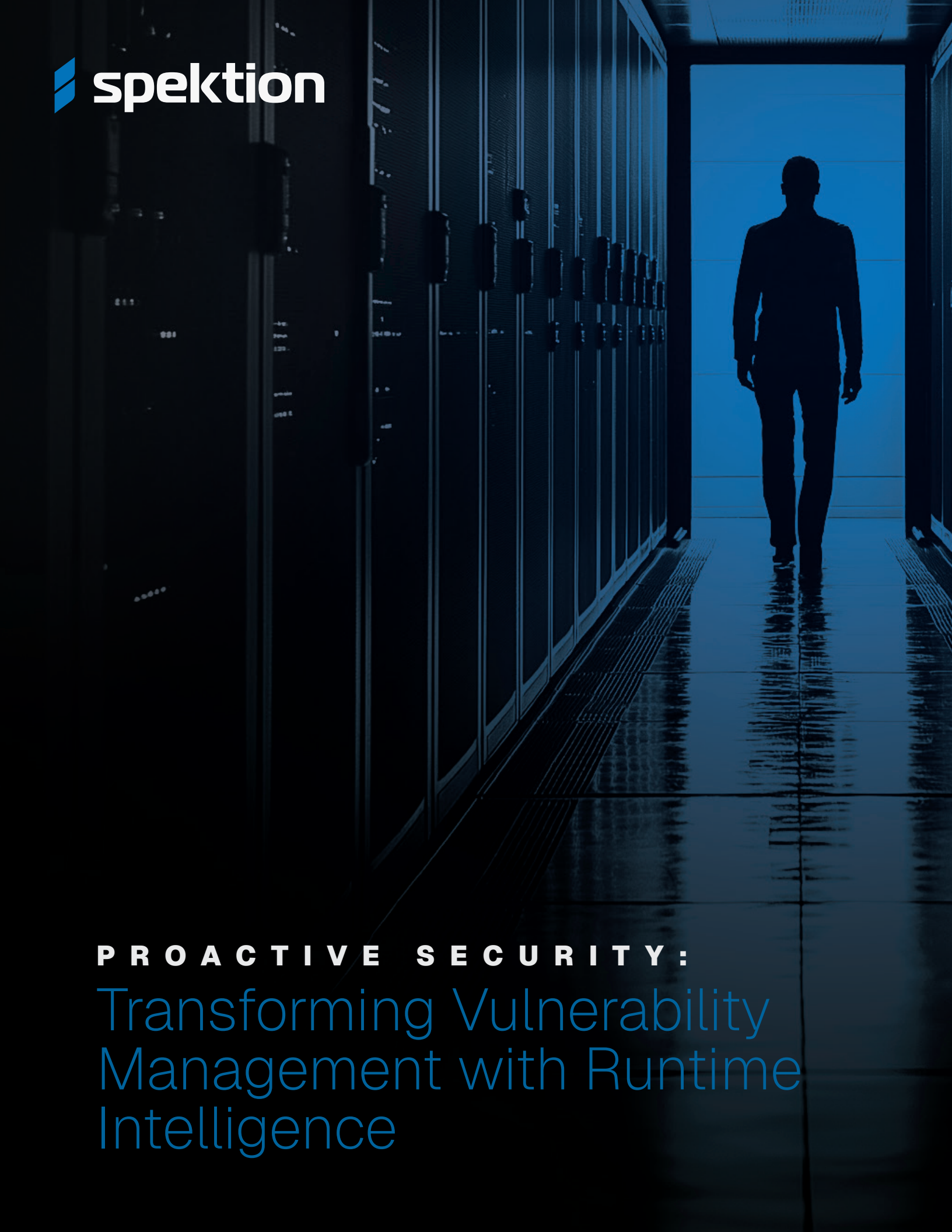
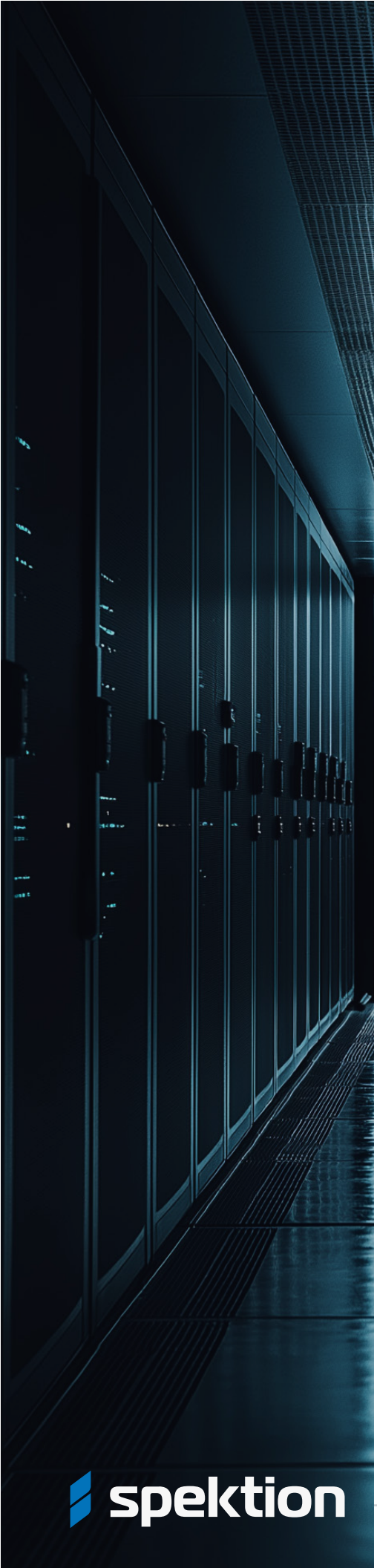
spektion



PROACTIVE SECURITY:

Transforming Vulnerability Management with Runtime Intelligence



Executive Summary

The cybersecurity industry faces a critical inflection point: vulnerability management practices remain stubbornly static while the software landscape grows increasingly dynamic. This white paper examines the fundamental disconnect in current approaches and presents a transformative solution.

Today's vulnerability management strategies rely almost exclusively on CVE publications, CVSS scoring, and other static assessment techniques built mostly on CVE data. This approach creates a perpetual cycle of reaction that fails to effectively secure organizational environments for three key reasons:

- Static Solutions to Dynamic Problems: Traditional vulnerability management applies fixed, point-in-time assessments to software that is constantly evolving and behaving differently across environments.
- Over-reliance on CVE Data: Organizations continue to base security decisions primarily on CVE publications while missing critical new data sources that could drive more effective insights and proactive security measures.
- Scaling Crisis: With the accelerating pace of software development—further amplified by generative AI and modern development practices—CVE-based vulnerability management processes are becoming increasingly inadequate to secure organizations effectively.

As the volume of software deployed within organizations continues to grow exponentially, the gap between vulnerability discovery and mitigation widens. Software approval processes suffer from similar limitations, with organizations relying heavily on vendor questionnaires and point-in-time assessments that provide little insight into actual application behavior and associated risks.

Spektion offers a fundamentally different approach through dynamic runtime intelligence. By monitoring application behavior in real-time and establishing behavioral baselines, organizations gain:

- Visibility into software functionality before vulnerabilities are published

- The ability to detect suspicious behavior patterns across all installed applications
- Context-aware vulnerability prioritization based on actual observed behaviors
- Enhanced incident response capabilities through behavioral anomaly detection
- More effective software approval processes grounded in observed functionality

The white paper demonstrates how this approach transforms the entire security lifecycle, from initial software assessment during the approval process to ongoing monitoring and vulnerability management. Through real-world examples, we illustrate how dynamic behavioral analysis reveals critical risks in commonly used applications that would remain undetected through traditional methods.

By shifting from reactive vulnerability management to proactive behavioral monitoring, organizations can escape the endless cycle of “vulnerability whack-a-mole” and develop security programs that can scale with the rapidly expanding software ecosystem being accelerated by AI and modern development practices.

TRANSFORMING VULNERABILITY MANAGEMENT: Beyond CVE-centric Security

The current approach to vulnerability assessment relies heavily on static data gathering techniques that fail to consider software functionality. Security programs depend on Common Vulnerabilities and Exposures (CVE) announcements to implement mitigations or make risk decisions for applications within their organizations. This dependence on unpredictable, uncontrollable CVE publications creates an expanding attack surface and increases the risk of compromise.

Most existing solutions remain reactive, inefficient, and ineffective because they rely on static CVE data or previously defined vulnerabilities. As generative AI dramatically accelerates both commercial and internal software development, the CVE discovery and disclosure ecosystem will increasingly struggle to keep pace. This problem is particularly acute for the growing volume of AI-enabled homegrown applications being built and deployed within organizations, which typically fall outside traditional CVE monitoring frameworks altogether.

This CVE-centric approach cannot lead to a truly hardened environment, as it fails to provide security teams with the comprehensive data needed for success in today's rapidly evolving software landscape.

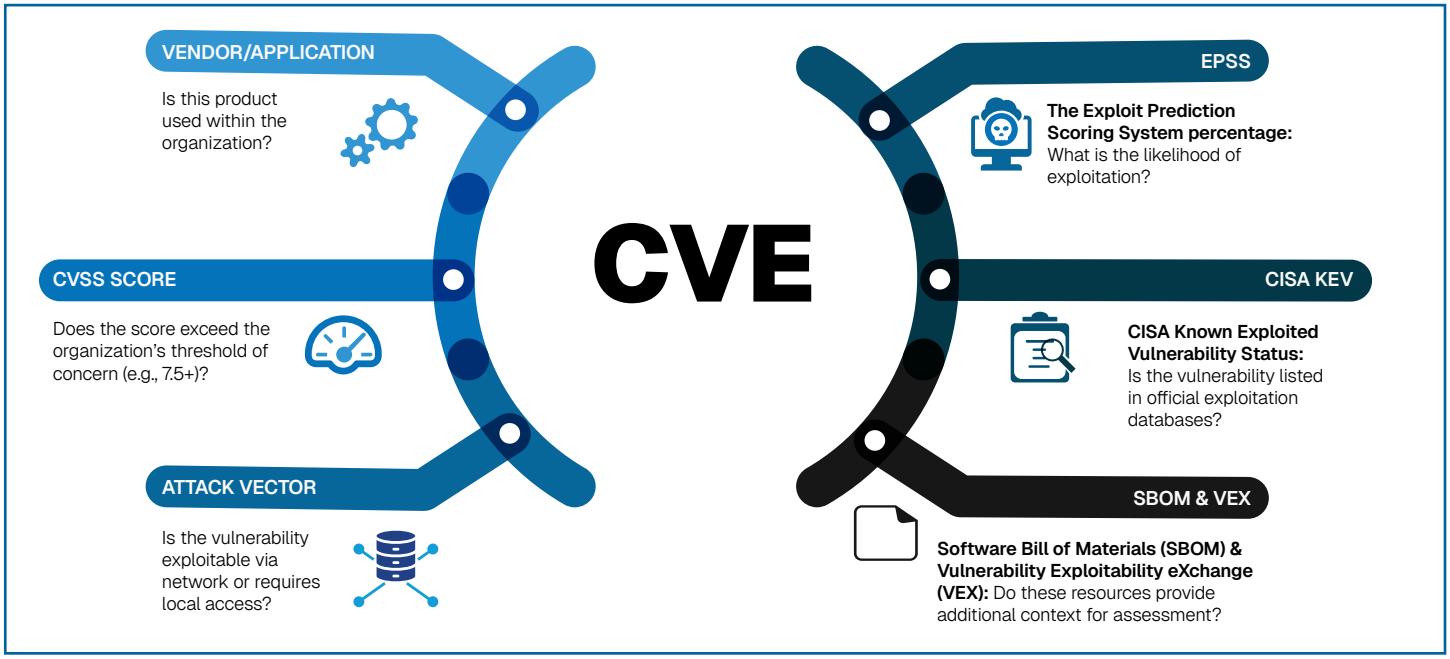
For teams involved in onboarding, deploying, and securing software to succeed, runtime data that monitors installed applications and details their functionality based on actual behavior—not just CVE vulnerability lists—is essential.

This report examines the limitations of current CVE-driven attack surface hardening methodologies and presents solutions for leveraging dynamic telemetry to improve the environment's overall risk posture.

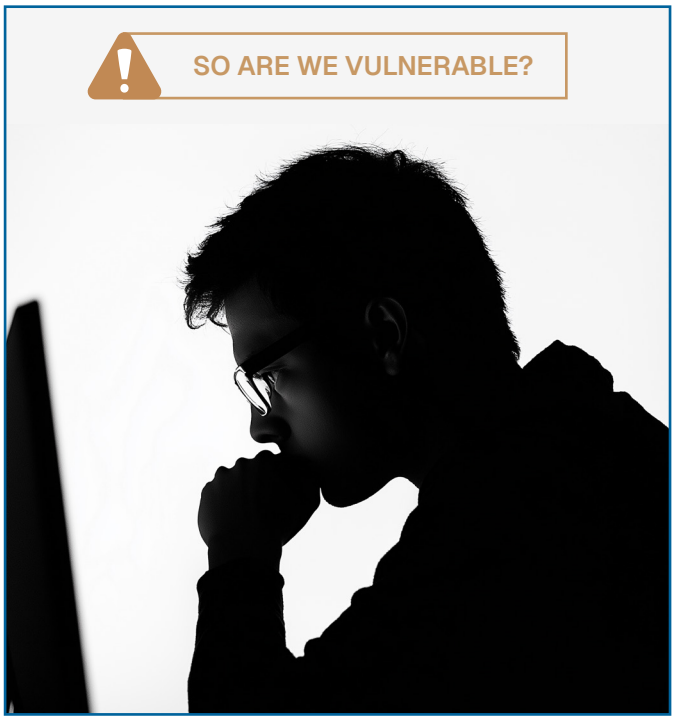
BEYOND STATIC METRICS:

The Flawed Foundations of Vulnerability Prioritization

The primary factors security programs consider when assessing vulnerability relevance include:



At its core, the publication of a CVE requires prior discovery and escalation. Once a vulnerability is scored and published, it often lacks sufficient information regarding reproduction steps, detailed mitigation strategies, or methods to determine if impacted endpoints are already compromised. Furthermore, CVSS scores for a given CVE vary widely depending on the source. Cyber defense teams receive a CVSS (Common Vulnerability Scoring System) score (0-10), CPE (Common Platform Enumeration) information, and a CVSS vector that incorporates multiple factors intended to aid in prioritization efforts.



RUN-TIME INTELLIGENCE:

Moving Beyond Static Vulnerability Management

To understand the benefits of dynamic intelligence versus statically acquired telemetry, it is important to understand what each approach delivers and how the data is gathered.

STATIC

- Deploy agent
- Collect inventory of installed applications + versions
- Cross reference installed versions (CPE) against published vulnerabilities
- Notify administrators when matches occur and include links to NVD, MITRE, and/or vendor websites for vulnerability information
- If available: vendor may provide additional scoring or research to assist in patch prioritization (often as a premium service)
- Repeat process cyclically

DYNAMIC

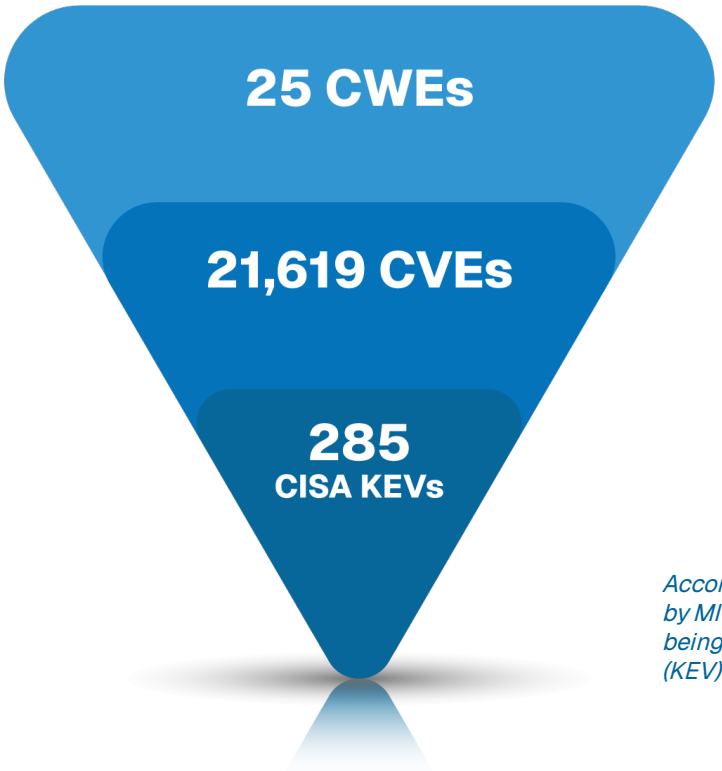
- Deploy agent
- Collect inventory of installed applications + versions
- Begin monitoring running applications to establish functionality baseline
- Take into account configurations and other factors specific to an organization's IT environment
- Perform initial risk assessment based on baseline data
- Group functionality for each application into defined risk categories and score based on criticality to aid in mitigation prioritization
- Provide compensating control recommendations and specific process/filename details for identified risks (e.g., which process is performing the risky behavior)
- Continuously monitor for changes to the established baseline for each application

When utilizing dynamic data, organizations can identify risky behavior before CVEs are published by evaluating actual functionality rather than relying on traditional risk categorization methods. Additionally, by focusing data collection on actively running applications rather than historical installations, organizations can develop more accurate asset inventories and gain better visibility into their true attack surface.

FROM CLASSIFICATION TO CONTEXT:

The CVE-CWE Disconnect

One of the more common methods for categorizing a weakness that may lead to a vulnerability is the Common Weakness Enumeration (CWE) list. According to the CWE site[1], a "weakness" is a condition in software, firmware, hardware, or a service component that, under certain circumstances, could contribute to the introduction of vulnerabilities.



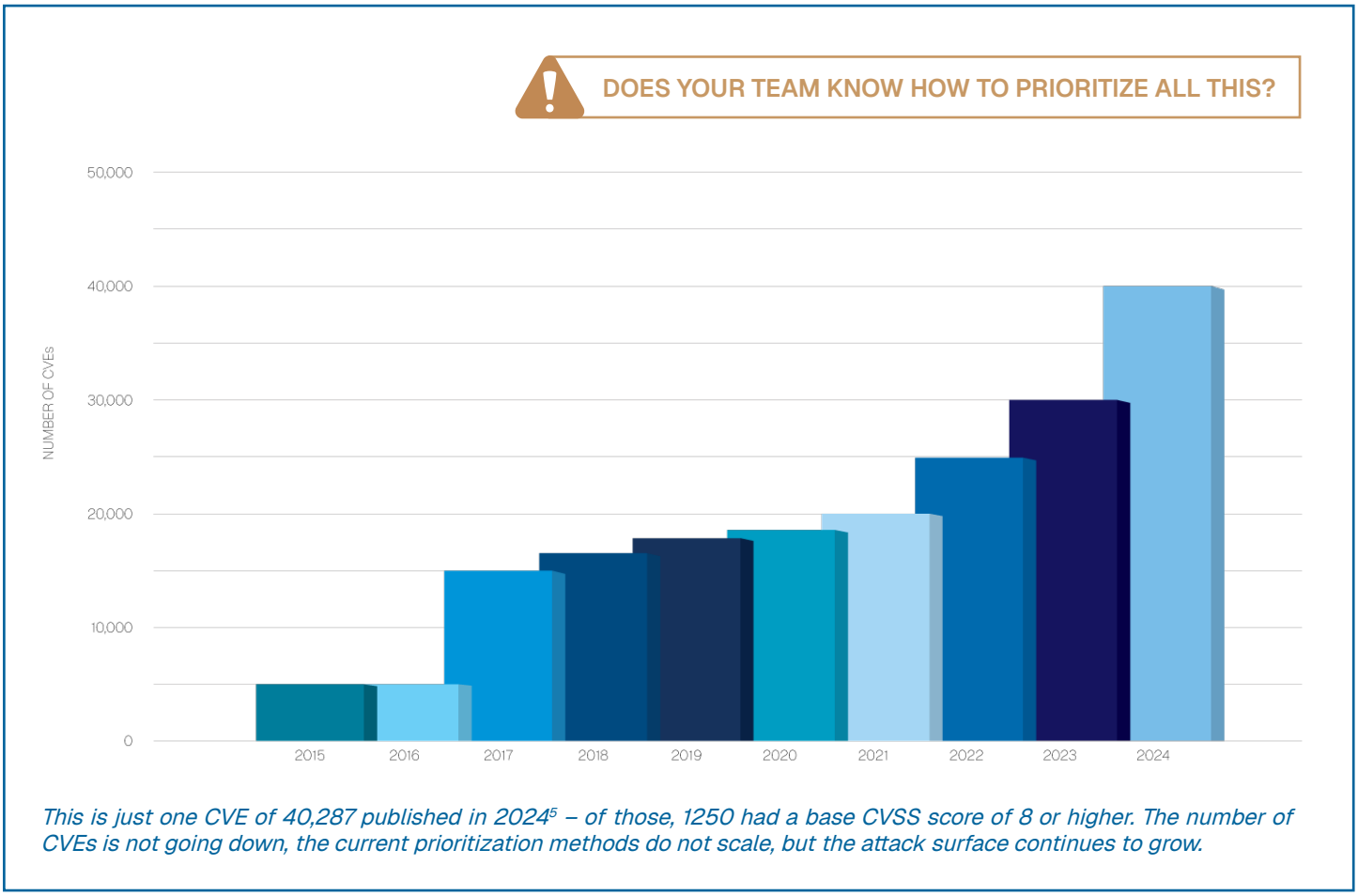
According to Vulncheck[2], the top 25 CWEs listed by MITRE[3] represented 21,619 total CVEs, with 285 being listed as Known Exploited Vulnerabilities (KEV) by CISA.

The fundamental disconnect lies in how organizations use these classifications: CWEs describe general weakness categories, while CVEs represent specific vulnerability instances. This disconnect means security programs often lack the context to understand which CWE patterns pose the greatest actual risk in their specific environments.

One of these top twenty-five is CWE-502 (Deserialization of Untrusted Data), also tracked as CAPEC-586 (Common Attack Pattern Enumeration and Classification) with a typical severity of HIGH[4]. An example CVE that lists CWE-502 is CVE-2022-31199, a vulnerability impacting Netwrix Auditor that made its way onto the CISA KEV list on July 11, 2023—seven months after its initial publication on November 7, 2022. This vulnerability allowed for remote code execution via an unsecured remoting port, leading to SYSTEM level privileges. In many organizations, this application would not be patched immediately due to downtime impact, potential change freezes (often December to January), and because many teams assume that non-public-facing applications are not immediate concerns.

Due to the nature of this application, its installation location, and the level of access required for its operation, a target organization relying on CISA KEV as its primary basis for patch prioritization would have remained completely exposed for 11 months.

This represents just one CVE of 40,287 published in 2024[5]—of which 1,250 had a base CVSS score of 8 or higher. The number of CVEs continues to grow, current prioritization methods fail to scale, and attack surfaces persistently expand.



This is just one CVE of 40,287 published in 2024[5] – of those, 1250 had a base CVSS score of 8 or higher. The number of CVEs is not going down, the current prioritization methods do not scale, but the attack surface continues to grow.

1 https://cwe.mitre.org/about/index.html

2 https://vulncheck.com/blog/cwe-top-25-2024

3 https://cwe.mitre.org/top25/archive/2024/2024_cwe_top25.html

4 https://capec.mitre.org/data/definitions/586.html

5 https://www.cvedetails.com/browse-by-date.php

SOFTWARE APPROVAL THEATER:

The Dangerous Illusion of Security

When applications enter the Third-Party Risk Management (TPRM) process for approval within an organization's environment, there is often insufficient information to make truly informed decisions. The TPRM process, sometimes supported by vulnerability management teams for software assessments, heavily relies on vendor questionnaires, Gartner reports, or occasionally a Software Bill of Materials (SBOM) provided by the vendor. Vendor questionnaires create a false sense of security by incorrectly assuming that vendors with mature cybersecurity programs also excel at application security. Organizations have minimal visibility into whether vendors prioritize application security or are simply proficient at completing forms. This flawed premise allows software to be deployed throughout the organization without proper due diligence, often earning the coveted "Approved Software" designation.

Similarly, open-source solutions under consideration for adoption frequently undergo security architecture reviews based primarily on documentation and CVEs in referenced libraries. However, many of these identified CVEs may not actually be exposed when the code runs in production, leading to both false positives that waste resources and false negatives that miss genuine risks based on actual runtime behavior.

In a hypothetical world of unlimited resources, organizations would have their Red Team and Cyber Defense Team conduct comprehensive "purple team" exercises for all software to understand exploitable risks and develop targeted mitigations. However, the reality of resource constraints and the sheer volume of software being deployed makes this approach impractical for most organizations, leaving them to rely on far less effective methods.

Once the TPRM process or open-source solution review concludes with application approval, the endpoint management team focuses primarily on functionality and deployment prerequisites, assuming thorough due diligence preceded approval. Upon this uncertain foundation, deployment proceeds, further expanding the organization's attack surface.

Subsequently, the vulnerability management program becomes responsible for ensuring patching efficacy and recommending or implementing mitigating controls when vulnerabilities lack immediate patches. Their success depends on knowing which assets have vulnerable versions and having access to published CVEs or other vulnerability disclosures. More mature organizations may establish internal scoring criteria to assess actual criticality relative to their environment, rather than relying solely on CVE Numbering Authorities (CNA) or vendor-defined metrics.

This process fundamentally assumes that all software on company assets has undergone TPRM evaluation, end users lack local admin privileges, no ad hoc software installations occur, and the cyber defense team maintains an accurate inventory of all applications with risk categorization

capabilities. Few, if any, companies meet these baseline assumptions. Even those that do rely on reactive, static, point-in-time data rather than dynamic information.

So how do we address this problem with commercial software assessment when current methods are failing?

Enter dynamic data collection...

SOFTWARE BEHAVIOR ANALYSIS:

A Window into True Application Risk

The primary distinction between current solutions and dynamic collection lies in the ability to analyze an application's behavior, highlight specific risks based on observed activities, and monitor deviations from an established functionality baseline. Dynamic collection also evaluates how applications are deployed, configured, and interact with secondary and external services. To truly assess the risks associated with third-party compromise, organizations must also consider undocumented third-party interactions that applications depend on for functionality.

Even when running applications within dynamic detonation environments (sandboxes), observations are limited to brief time windows. These limited snapshots cannot compare to the insights gained from monitoring an application's behavior over extended periods in actual deployment environments. Furthermore, sandboxes are primarily designed to detect malicious behavior rather than comprehensively assess operational risk.

Armed with dynamically gathered information and recommendations, organizations become equipped to mitigate risky behavior when deployed software exceeds internally defined risk thresholds. Additionally, when high-risk vulnerabilities emerge, security programs can quickly identify which assets are running impacted versions and create targeted detection mechanisms to hunt for potential exploitation activity.

With this high-level overview of dynamic collection in mind, let us revisit our previously mentioned workflow involving the TPRM, Vulnerability, and Cyber Defense teams.

BEHAVIOR-DRIVEN SOFTWARE EVALUATION:

Reimagining the Software Assessment Process

When a new onboarding request enters the TPRM process or when evaluating open-source solutions, the user or team requesting the software first evaluates it for functionality with a dynamic data collection sensor installed on their test system. During this pre-determined learning period, the application runs in an environment that mirrors the organization's standard image while being monitored and its risk profile established. This approach produces the type of deep insights at scale that would otherwise only be possible through resource-intensive purple team exercises, effectively democratizing advanced security analysis for all software evaluations.

With comprehensive behavioral data collected, vulnerability management and security architecture teams review identified risks and make informed decisions about deploying the application to wider end-user systems. If the application is approved but requires mitigations before broader deployment, these recommendations can be delivered to the endpoint team. Additionally, if the security architecture determines that the application would require a security exception, they can now effectively articulate specific risks to potential risk owners.

Once deployed, the "approved" application's established baseline from the learning period can be shared with cyber defense teams, enabling them to monitor for potentially risky functionality changes and create detection mechanisms for any deviations from the baseline. When vulnerabilities requiring immediate patching are published, the vulnerability management team can quickly determine which assets are running the impacted versions.

Furthermore, the TPRM process and open-source solution reviews can now include meaningful annual evaluations to determine if any software has experienced an increase in its risk profile. If risk profiles have risen, security teams can identify the earliest versions where changes occurred and initiate formal inquiries with vendors requesting clarification.

While these steps assume standardized application onboarding, many organizations struggle with non-standard deployments. Even in these scenarios, dynamic collection and analysis enables security programs to monitor any installed application with risky functionality and take appropriate action.

FROM INVENTORY TO INSIGHTS:

Dynamic Data's Expanded Capabilities

Taking a step back from the standard use case of patching prioritization, dynamic data collection enables several additional valuable capabilities:

ACCURATE ASSET INVENTORY

Resolve the persistent challenge of identifying which applications and versions are actually installed and running on computers within your environment.

HOST RISK-BASED PRIORITIZATION

Develop host-specific risk scoring that extends beyond installed applications, associated published CVEs, and OS-specific configurations. Administrators can implement segmentation controls based on which computers run higher-risk software.

CVE PRIORITIZATION

Transform vulnerability management by prioritizing patching efforts based on the observed runtime behavior of software, including behavior that would amplify the blast radius of exploitation. This context-aware approach ensures critical vulnerabilities in applications with high-risk behaviors receive attention before less impactful vulnerabilities, even when the latter have higher CVSS scores.

INCIDENT RESPONSE ENHANCEMENT

Empower incident response (IR) teams with detailed baseline knowledge of normal application behavior, enabling them to quickly identify and respond to exploitation attempts. When software behaves outside its established runtime profile, IR teams can pinpoint exactly what functionality is being abused, accelerate root cause analysis, and implement targeted containment strategies that minimize business disruption.

FUNCTIONALITY-BASED THREAT HUNTING

Move beyond searching for application names or process identifiers by hunting for specific functionalities deemed risky or suspicious within your environment, regardless of the application performing them.

ANNUAL BASELINE REVIEWS FOR APPROVED SOFTWARE

Systematically review installed applications yearly to maintain your approved software list, remove unauthorized software, or proactively implement safeguards based on evidence of new functionality in previously approved applications.

TREND ANALYSIS

While static data provides a snapshot of your current environment and installed applications, dynamic intelligence allows you to compare your present state against historical baselines, revealing important trends and changes over time.

SPEKTION RUNTIME INTELLIGENCE:

Transforming Software Behavior into Security Insights

EXAMPLE: Popular Remote Monitoring & Management (RMM) Software

To better understand how dynamic data collection can help identify and mitigate insecure applications in your environment, let's examine a widely used remote monitoring and management (RMM) application.

Dynamic collection from an endpoint running this software identified the following critical and high severity risks:

UNQUOTED SERVICE PATH DETECTION

An attacker could place a malicious executable in the path that gets executed before the intended service.

HOST NAME VERIFICATION DISABLED

Exposes the application to potential Man-in-the-Middle (MitM) attacks and other security risks.

DEBUG MESSAGES ENABLED

Can reveal sensitive application logic and data.

READ-WRITE-EXECUTE MEMORY PAGE

By allowing a memory page to be both writable and executable, an attacker could write malicious code into the page and then execute it.

KEYSTROKE CAPTURE USING THE WINDOWS API

Capturing keystrokes can lead to the unauthorized collection and potential misuse of sensitive information.

PROCESS MEMORY DUMPING

Can lead to the exposure of sensitive data, cryptographic keys, or personal information.

WINDOWS COMMAND SHELL

Executing processes through command interpreters, such as CMD.exe, with elevated privileges can lead to the execution of malicious commands with system-level access, significantly amplifying the potential impact.

When viewed through the lens of functionality versus the industry's current method of waiting for a CVE to be published, it becomes clear that even applications not traditionally viewed as risky could lead to potential compromise. The assessment of risk is further elevated when noting that some functionality, like Windows Command Shell utilization, occurs with elevated privileges, creating an even greater attack vector.

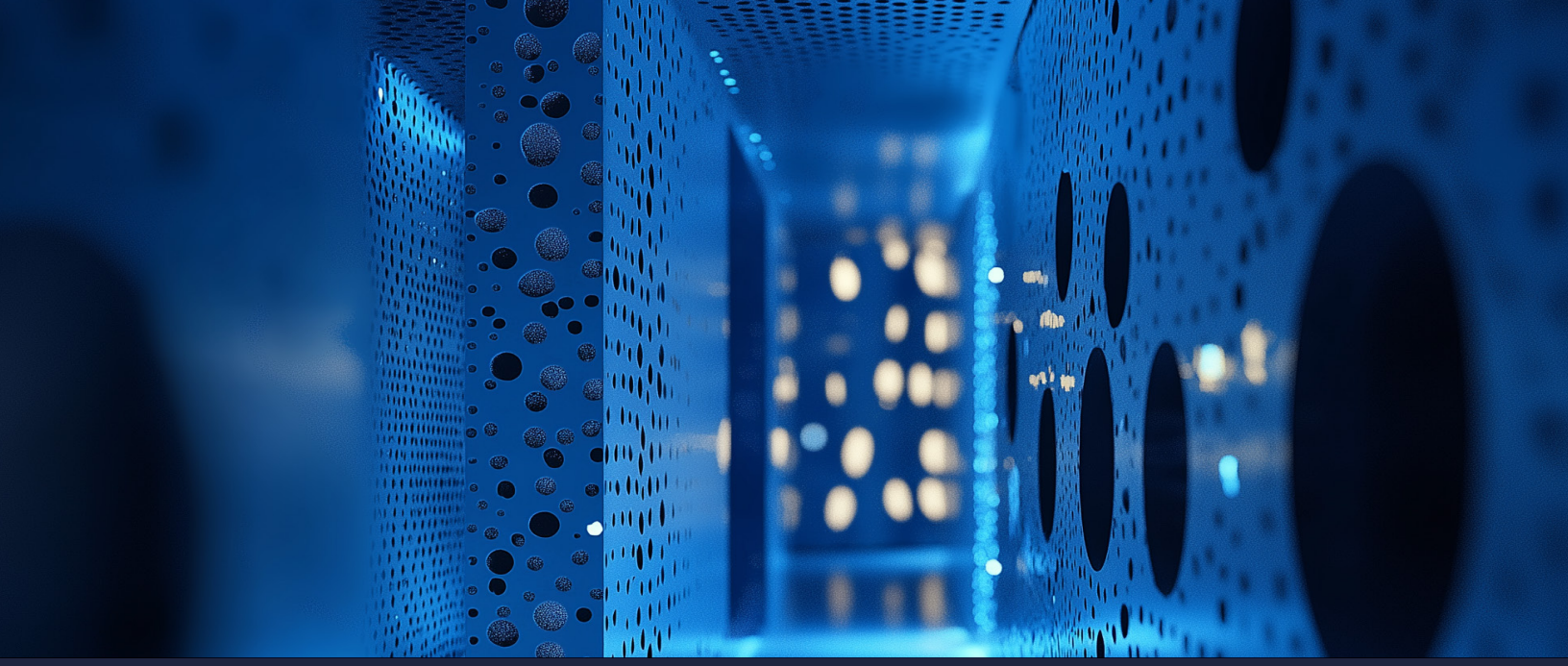
LSASS Process Memory Read Access Risk

One of the riskier access levels utilized by many applications and identifiable with runtime data analysis is LSASS process memory read access. While legitimate use cases exist that necessitate this level of access, organizations must rely on developer-implemented protections to prevent misuse by malicious actors seeking sensitive authentication data, saved browser passwords, encryption keys, and other cryptographic secrets.

To emphasize the benefits of dynamic telemetry, consider this brief mental exercise: Which applications deployed to your endpoints today have LSASS Process Memory Read Access? Of those applications, which ones have compensating controls to ensure they only perform their intended core functions? As a bonus question, what are the legitimate core functions of these applications that require reading LSASS process memory?

With current industry tooling, you're likely struggling with the first question. Even if you could compile that initial list, how would you accurately determine which applications are legitimate and establish their baseline functionality? Simply put, you cannot—you need dynamic data that at-scale and accessible.

COMPETITOR	SPEKTION
Point-in-time data collection	Dynamic, runtime collection of software and potential for risks based on functionality
List of all installed applications	List of all installed application grouped by those that are active
Ability to determine if a given software version is installed	Ability to determine if a given software is installed and monitor for changes in how a given software functions version to version and over time
Evidence path, or how they detected a given version is installed	Evidence path, recommended mitigation controls, dynamic functionality association to a specific process name/path



Escaping the Vulnerability Management Treadmill

Throughout this report, we have outlined how static solutions to dynamic problems will never result in a hardened attack surface. At best, current approaches provide scoring and parsing methods that are not specific to individual organizations, and at worst, they give a false sense of security that leads to compromise. These approaches also fundamentally assume that all software is in scope for CVE research and publication, which is far from reality. Custom-developed applications, open-source projects with limited community support, and rapidly evolving software often fall outside the scope of traditional CVE monitoring, creating significant blind spots in security posture.

Dynamic data collection reveals how applications function in your specific environment—not someone else's—leading to informed risk assessments and targeted mitigations, regardless of whether the software is covered by formal vulnerability research.

This leaves us with the question: Are we truly doing our best, or are we settling for current solutions because "that's how we've always done it"?

Glossary of Terms

CVSS

CVSS scoring, vectors, and exploitability scores are inherently subjective. It is not uncommon to see one source classify an Attack Vector (AV) as local, while another lists the same CVE as network, with both assigning different scores to the identical vulnerability. This scoring inconsistency becomes further complicated when organizations rely on third-party tools that attempt to score CVEs independently.

EPSS

The Exploit Prediction Scoring System (EPSS) attempts to help teams prioritize patching by using relevant vulnerability data to determine the probability of a given CVE being exploited. While the logic and methodology are sound, it remains a probability model that is often overprioritized when determining which vulnerabilities pose the greatest risk. Furthermore, EPSS does not consider the likelihood of exploitation within specific environments, as the model is trained on publicly available data and may reference scoring metrics that do not align with internal methodologies.

SBOM

The Software Bill of Materials (SBOM) is an inventory of elements comprising software components. With the introduction of NIST SP 800-218 and CISA's push for SBOM adoption, it is rapidly becoming a new standard for development teams. However, from a defense perspective, its use cases are limited primarily to determining if applications contain vulnerable components. Since SBOM only lists components used by an application—not how each component is implemented—it often leads to wasted mitigation efforts on components that are not actually vulnerable. The Vulnerability Exploitability eXchange (VEX) concept, which provides developer attestation regarding specific known vulnerabilities, attempts to address this SBOM reliability issue.

Unfortunately, SBOM and VEX implementations are not scalable for most cyber defense organizations to request/parse or for development teams to produce. While automated SBOM creation can be integrated into development pipelines, the more valuable hardening resource (VEX) would be exceedingly difficult to scale given the high volume of CVEs generated annually.

All methodologies outlined above represent lagging, reactive risk indicators that rely either on public data or static, point-in-time assessments that fail to evaluate an application's true risk. They are founded on three core fallacies: complete knowledge of all installed applications across all endpoints, accuracy and organizational relevance of scoring data, and the sufficiency of static data for risk mitigation. Continued reliance on these methods prevents vulnerability management programs from achieving proactive hardening, instead trapping them in a Sisyphean cycle of reactivity.

CLOSING THE GAP:

Spektion's Vision for Software Security

At Spektion, we're reshaping how organizations approach software security. By combining cutting-edge technology with deep expertise, we're creating a future where proactive risk management is standard practice. Our mission is to continually evolve our solutions to meet the challenges of tomorrow's threat landscape, ensuring our customers remain a step ahead in securing their digital environments.

Our team unites security experts with backgrounds as leaders of global enterprise security programs, offensive security practitioners, military cyber operations specialists, and intelligence operations professionals. Having led and supported enterprise security programs at the highest levels, we understand the real-world obstacles organizations face from both offensive and defensive perspectives.

We chose to address a critically underserved yet fundamental challenge in cybersecurity: managing risks within the software supply chain. We recognized that this gap is at the root of numerous security incidents and drives many of the complex vulnerability management and third-party risk issues that organizations face. Despite the critical importance of this challenge, solutions have been lacking. Leveraging our experience managing this problem and our expertise in how it is exploited, we committed to closing this gap for our customers.

spektion

Learn More at spektion.com