



Balancing Speed and Security:

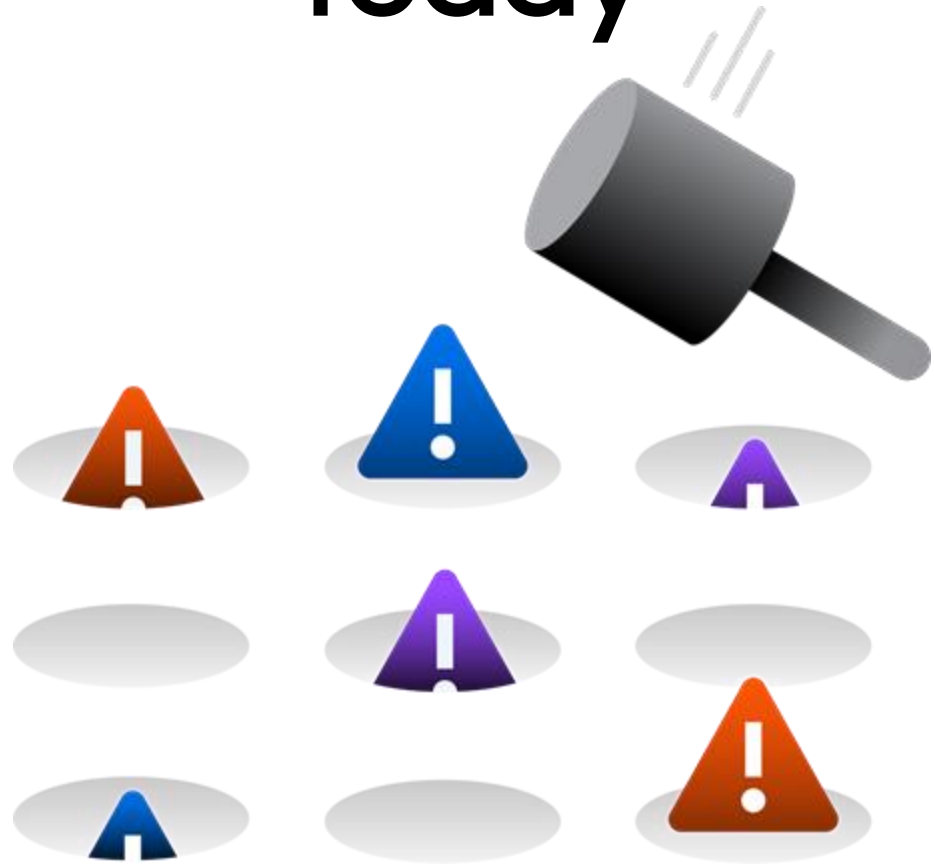
Software Development in a Zero Trust World

In today's rapidly evolving digital landscape, organizations face the dual challenge of accelerating software development while maintaining robust security measures. This presentation explores the intricacies of managing software development and security risks within a Zero Trust Software Access (ZTSA) framework.

George Manuelian

 **RAPIDFORT**



Vulnerability management today



Small Enterprise
50-200k
Vulnerabilities

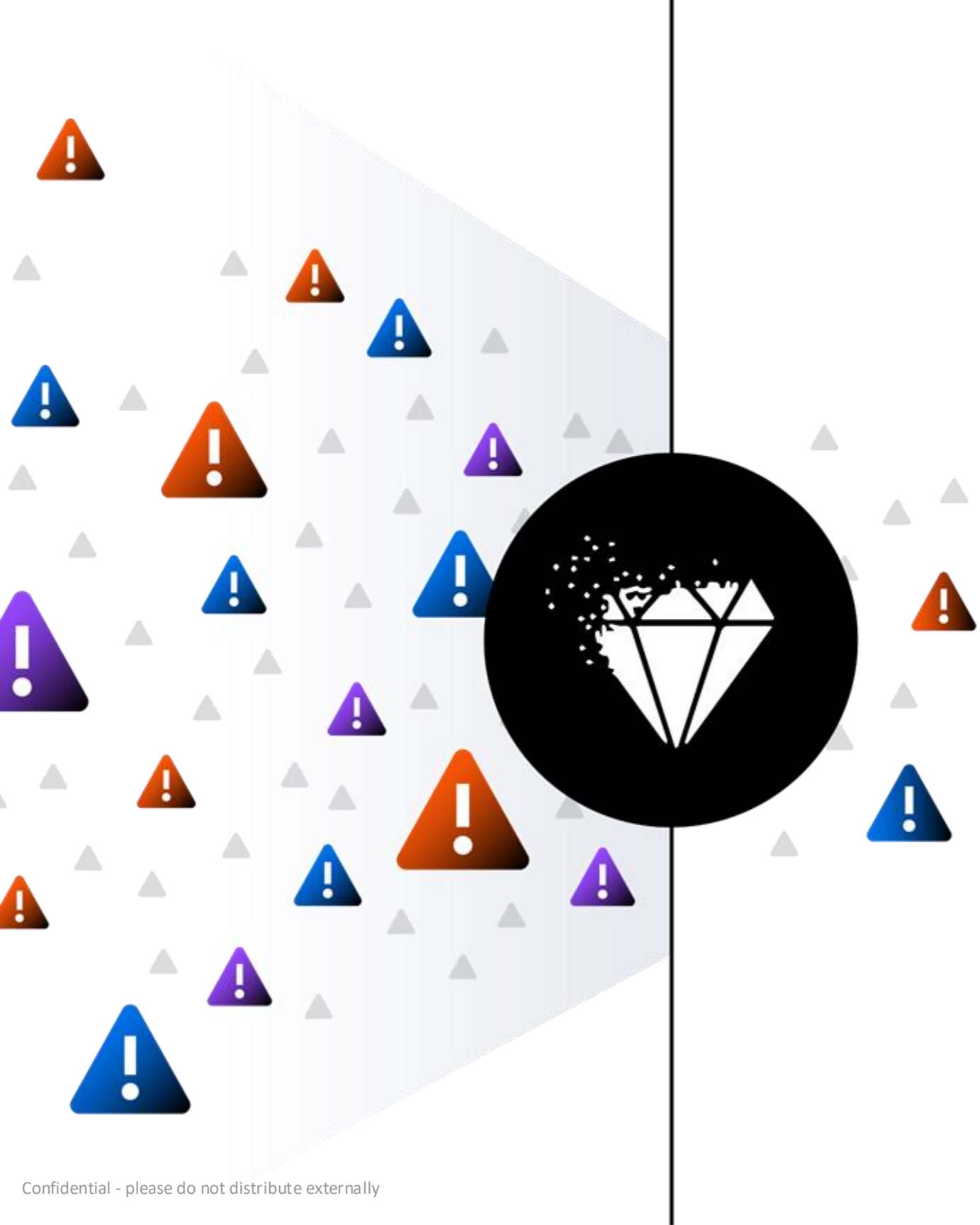


Medium Enterprise
1-3M
Vulnerabilities



Large Enterprise
10-20M
Vulnerabilities





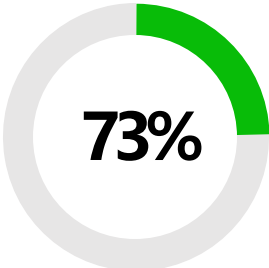
50-90% of software
in workloads is not used
in production

Impact Study on a Larger Set of Third-party Software Images

In a study of 1,578 unique community images, with over 10 Million downloads they automatically removed 73% of total vulnerabilities, 73% of critical and highs, and reduced the overall software attack surface by 64% **with no effect to the end application** resulting in automatic hardening of 155,400 vulnerabilities and 425GB of software!

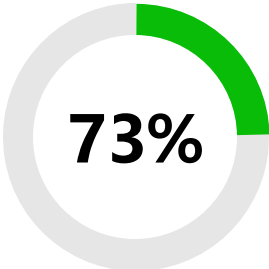
All Vulnerabilities

Original: 211699
Hardened: 56274



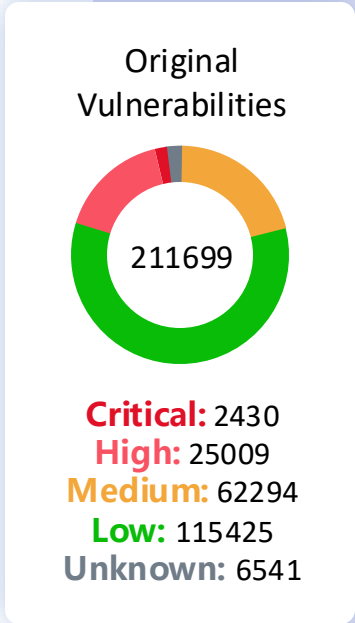
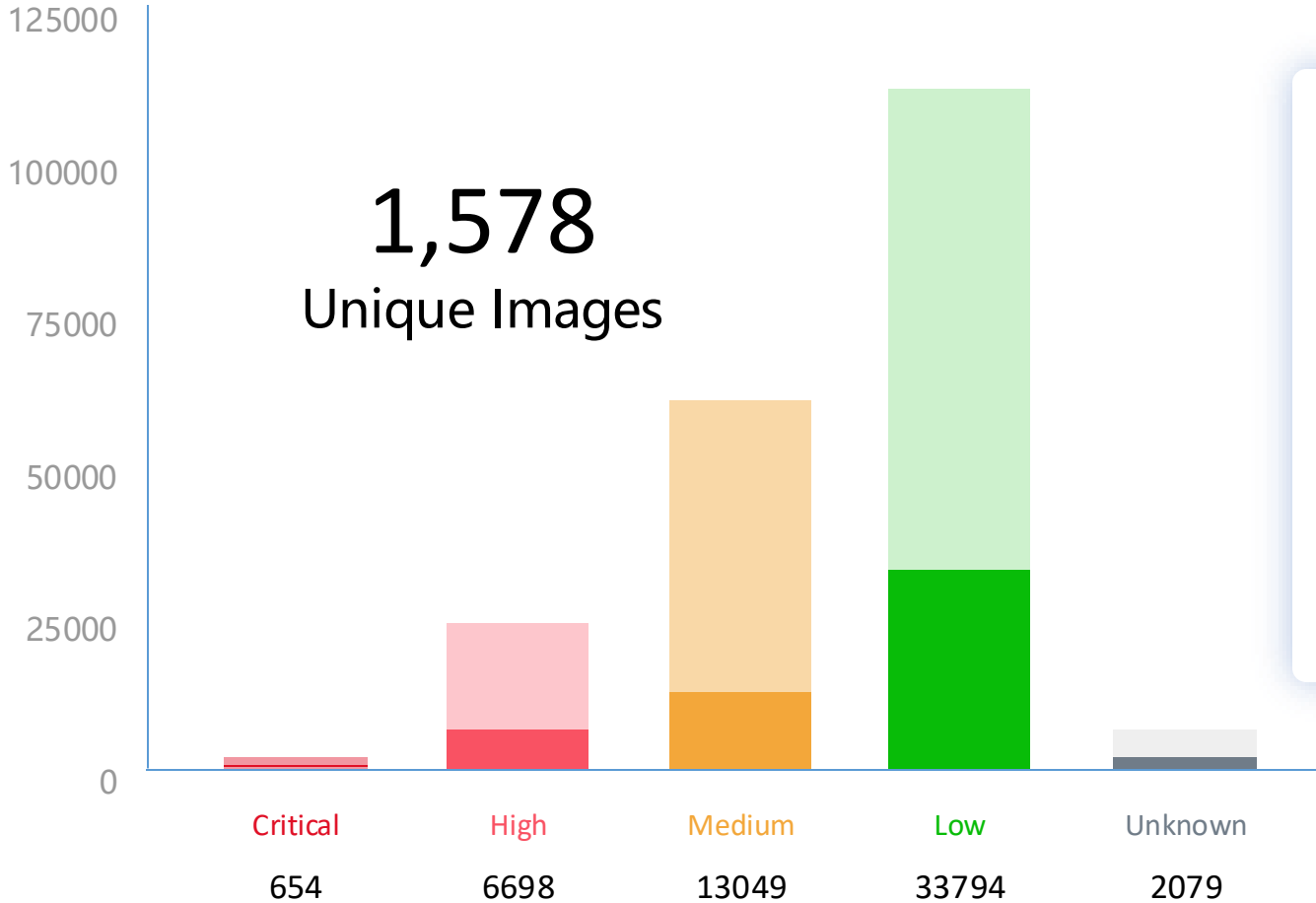
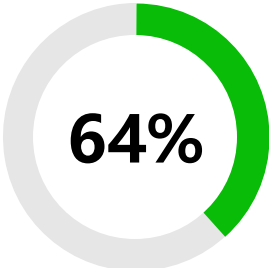
Critical & High Vulnerabilities

Original: 27439
Hardened: 7352



Attack Surface

Original: 660.66 GB
Hardened: 235.7 GB





The Rise of Zero Trust Software Access (ZTSA)

Zero Trust Software Access (ZTSA) has emerged as a critical security paradigm, shifting the focus from the focus from perimeter security to verifying every access request, regardless of location or device. location or device.

Minimized Trust

ZTSA assumes no user or device is inherently trustworthy, requiring strict verification for every access attempt.

Strong Authentication

Multi-factor authentication (MFA), biometrics, and other robust authentication methods are crucial for for securing access.

Continuous Monitoring

ZTSA necessitates constant monitoring of monitoring of user activities and system system behavior to detect and respond to respond to potential threats.

Least Privilege

Users are granted only the minimum level of access required to perform their tasks, minimizing potential damage in case of compromise.

Foundation of Zero Trust Software Access (ZTSA)



1. Vulnerability Scanning

Automated tools identify known vulnerabilities in container images

Regular scans are essential to detect new vulnerabilities as they emerge.

2. Continuous Patching

Apply security updates and patches to address vulnerabilities and mitigate potential risks.

Comprehensive patching process, including testing and deploying patches in a timely manner.

3. Hardening

Remove unnecessary software bloat from containers – reduces future CVEs

Reduce Software Attack Surface - stops



Hardening Container Images: Reducing the Attack Surface

Container images are fundamental building blocks for modern software deployments, but they can also introduce vulnerabilities if not properly secured.

1 Minimize Base Image

Start with a minimal base image, reducing the attack surface by including only essential packages and dependencies.

2 Benchmarking – CIS / STIG

Follow security globally recognized and consensus-driven best practices guidelines to verify optimal configuration to maximize security.

3 Regular Vulnerability Scans

Conduct regular vulnerability scans of container images to identify and remediate any known security flaws before deployment.

4 Automate in CI/CD

Implement an automated system in the build pipeline to enable continuous scanning and container hardening

There Steps to Remediating Vulnerabilities

1



Step 1 : Curated Images (0-60% remediation)

- Near Zero-known CVE images
- Hardened NIST 800-70 images
- FIPS 140-2 validation
- RF Justification for Plan of Action and Milestones (POA&Ms) (POA&Ms)

2



Step 2 : Instrument & Profile (60-80% remediation)

- Understand your software attack surface
- Generate SBOM and RBOM
- Prioritize vulnerability remediation
- Obtain full list of unused packages for discussion with engineering
- Integrated CIS / STIG Benchmark

3

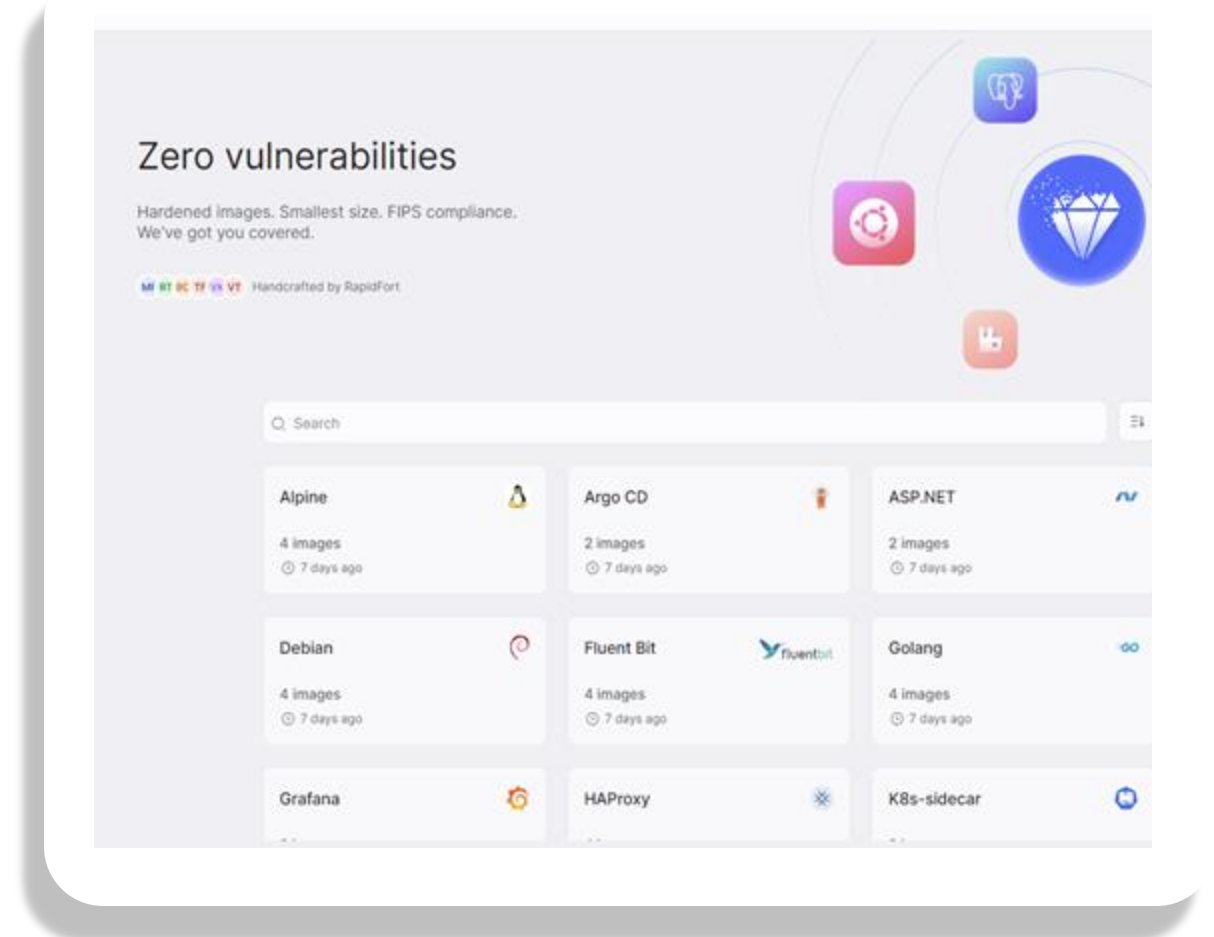


Step 3 : Harden & Monitor (80-95% remediation)

- Minimize your software attack surface
- Reduce your workload size
- Improve your security posture
- Manage 80% less software: Less risk, vulnerabilities, patches, alerts alerts

Step1 : Start with Curated Images

- Low vulnerability images with zero code changes
- Easy usage and drop in ... [the easy button.](#)
- Available across 4 largest Linux ecosystems
- FIPS compliant
- Integrated STIG/CIS benchmarking
- Deep, fast, accurate, image scans with researched justifications
- Version control diffs and alerts for new versions
- Extending **6M+** downloaded initial offering (was community images (80%) now curated images (95%))

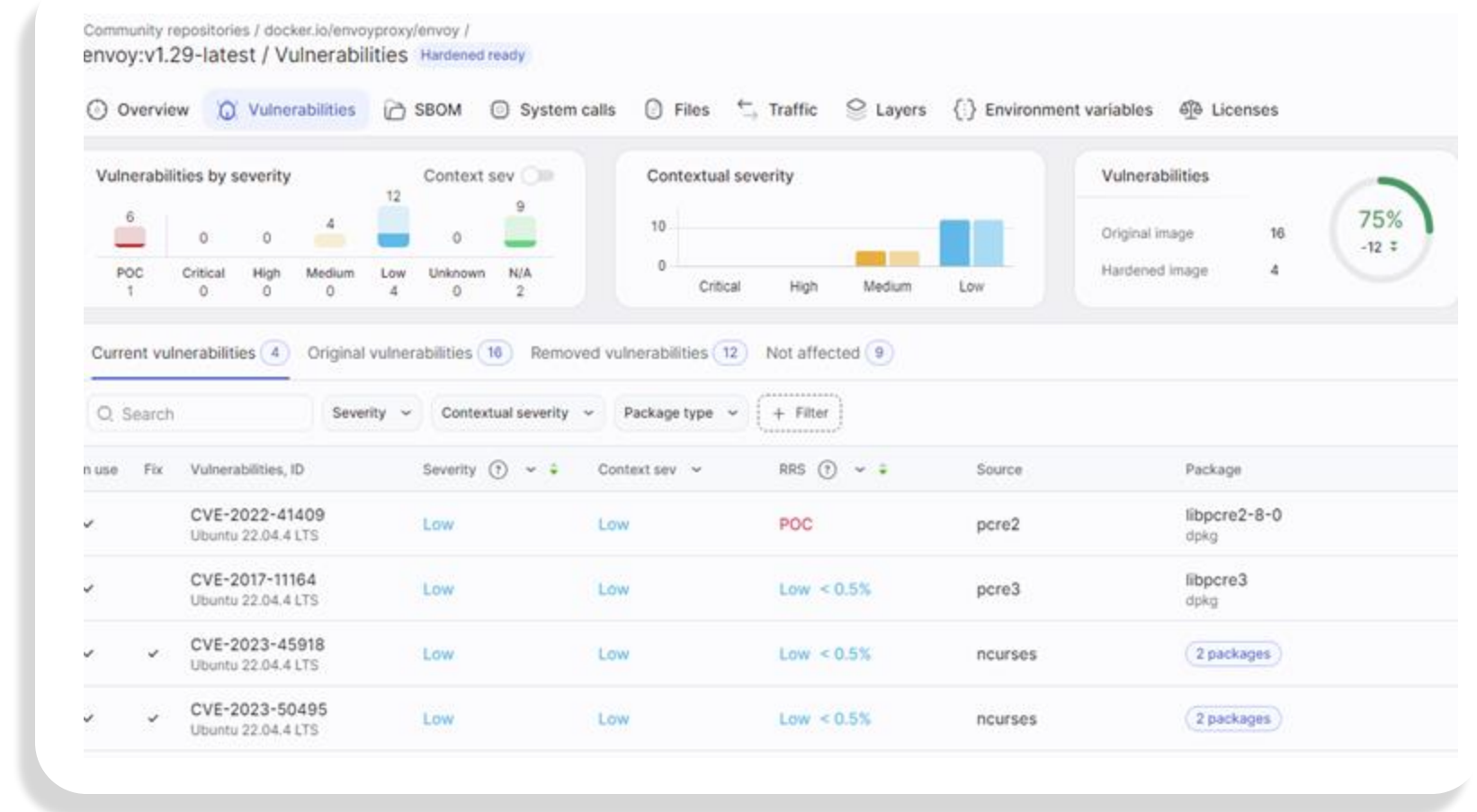


Step 2: Instrument and Profile – Build SBOM and RBOM

1. **SCA** (Software Composition Analysis) scanners detect what components are in software and provide risk metrics about these components mainly around vulnerabilities and patches
 - Detect the packages to get a list of components
 - Cross reference against databases to identify risks
1. Profiling requires :
 - **Comprehensive:** they cover a wide range of languages and operating systems
 - **Accurate:** in detecting packages having low false positive, false negative rates
 - **Current:** in that the databases they cross reference are timely and broad
 - **Broad:** support registry scanning, build time scanning, run-time scanning in all 3 major CSPs
 - **Feature Rich:** have a wide toolset to compare scans, evaluate scan quality, warehouse the results
 - **Actionable Reporting:** actionable reports, ingesting & publishing differing formats (SPDX and CycloneDX)
 - **Responsive:** Provide excellent customer support and rapidly add features that DoD users demand
 - **Deep:** include binary scans and not just inspecting meta-data

Step 3 : Harden and Monitor

- Profile (instrument and learn) container behavior
- Remove unused Components
- An unsecure container goes into the CI/CD pipeline and a hardened container comes out
- Tune the hardening as needed:
 - Remove all critical
 - Remove all vs some packages
 - Keep/exclude files/directories etc

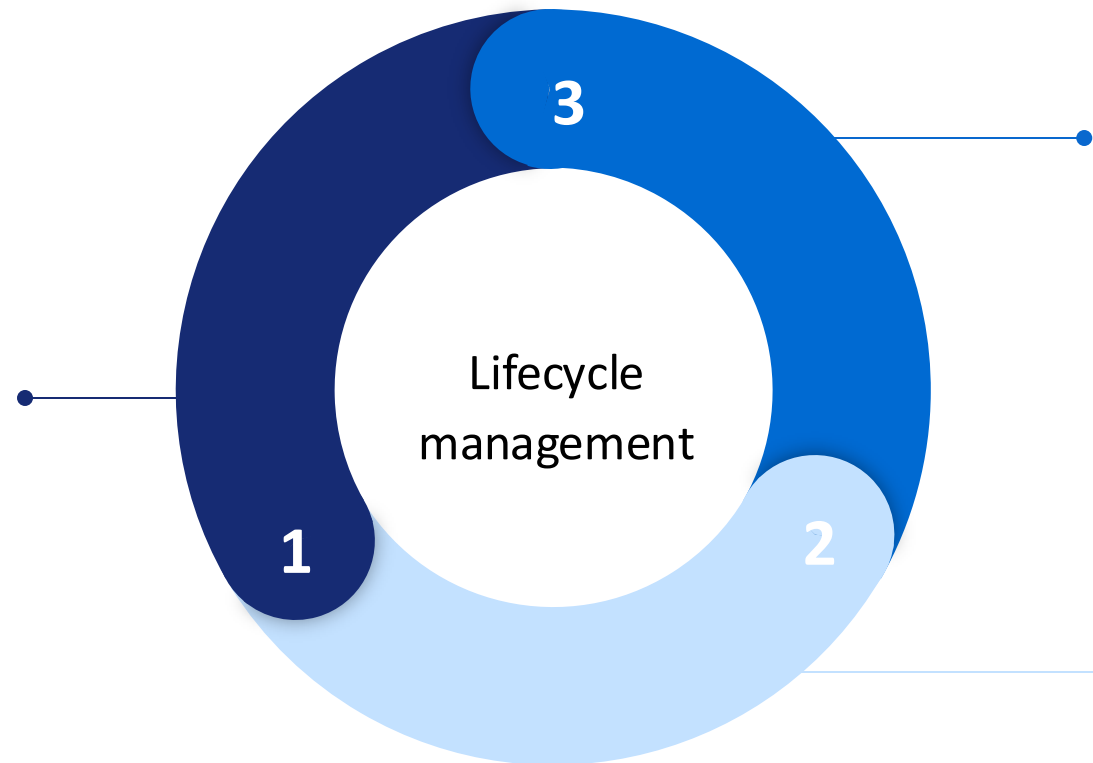


Secure Software Development Flywheel for Zero Trust Software (ZTSA)

Reduce Software Attack Surface Management (SASM) and automate vulnerability remediation without code changes:

Curated Base Images

- Minimal to zero vulns images
- Continuously patched
- OS and app bundles
- 3rd party apps
- Minimal break risk (minor version updates)
- Researched justifications



Optimization and hardening

- Reduce software footprint
- Remove vulnerable packages
- Only update if there are new vulnerabilities or fixes
- Manage and track exceptions

Scanning and Profiling

- Scan images
- Profile app behavior
- Identify unnecessary components
- Remove vs. patch decision

Conclusion and Key Takeaways

Balancing speed and security is paramount in software development. By embracing Zero Trust principles and implementing robust security measures, organizations can build and deploy secure software at speed.



Proactive Security

Implement a proactive security strategy that includes vulnerability scanning, patching, and secure coding practices.



Automation

Automate security checks in the CI/CD pipeline to streamline security processes and improve efficiency.



Secure Software Supply Chains

Establish secure software supply chains by verifying the integrity of all components and code.



Continuous Monitoring

Continuously monitor container environments for threats and have a robust incident response plan in place.



Thank you

George Manuelian

 **RAPIDFORT**