RAPIDFORT

WHITEPAPER

Getting FedRAMP-Ready: A Six-Step Container Security Roadmap

FEDRAMP | BOD 26-04 | FIPS | STIG



© 2026 RapidFort, Inc. | www.rapidfort.com

Introduction

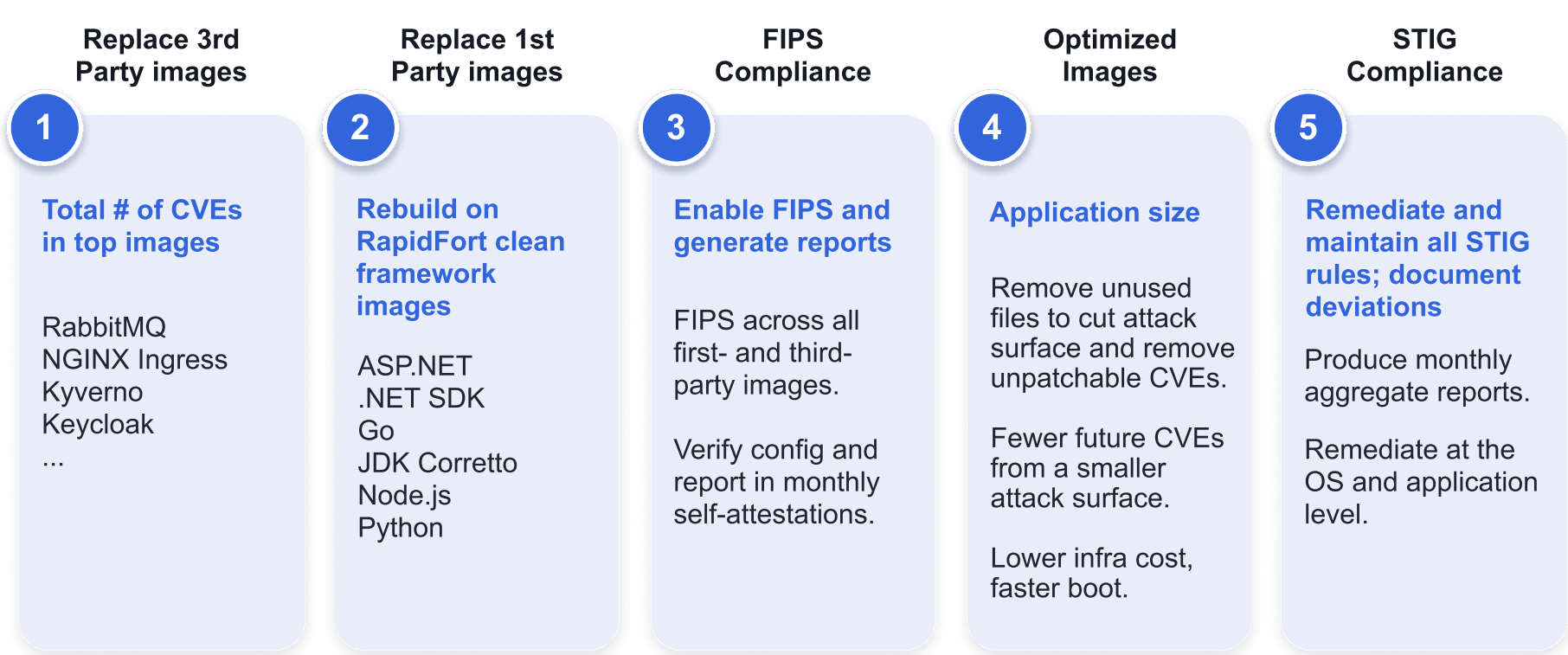
For a containerized cloud service, FedRAMP readiness is decided largely inside the container images. Most vulnerabilities are inherited from base and third-party images, and those same images must carry FIPS-validated cryptography, pass STIG configuration checks, and produce evidence every month.

The bar is also rising. FedRAMP will make its Vulnerability Detection and Response (VDR) and Vulnerability Evaluation and Reporting (VER) rules mandatory on December 7, 2026, to align with CISA Binding Operational Directive (BOD) 26-04. FedRAMP has stated that the legacy monthly scanning process most Rev5 services follow is no longer sufficient on its own.

This paper lays out six sequential steps that take a container estate from today's vulnerability backlog to a sustainable, audit-ready state.

> **Key Insight:** There is no silver bullet for CVE remediation in containers. Multiple steps, combined sequentially, are required: clean bases first, then cryptography, then minimization, then configuration, then continuous proof.

SIX SEQUENTIAL EFFORTS FOR FEDRAMP COMPLIANCE



Adapted from a RapidFort FedRAMP remediation roadmap for an enterprise SaaS platform. Figures are from that assessment and will vary by environment.

The FedRAMP Container Challenge

A typical enterprise container estate begins FedRAMP preparation with thousands of open CVEs, and the backlog grows every month. Manual patching does not scale against that curve. RapidFort estimates that about 88% of vulnerabilities are inherited from base images and dependencies rather than custom code, and that the average remediation cycle runs about 12 days.

ONE PRE-FEDRAMP ASSESSMENT OF AN ENTERPRISE SAAS PLATFORM

2,395
CVEs across the top 50 images

78 / 327
Critical / High findings in those images

~40
images needing FIPS-validated variants

~20 GB
footprint of ~90 containers, much of it unused

What FedRAMP asks of your containers

Requirement area	What it means for container images
Vulnerability detection and response	Detect, evaluate, mitigate, and remediate vulnerabilities continuously, on risk-based timelines aligned to BOD 26-04.
Reporting	Report vulnerability detection and response activity to all necessary parties at least monthly, in a consistent, human-readable format.
Cryptography	Use FIPS-validated cryptographic modules wherever federal data is protected.
Secure configuration	Apply a recognized configuration baseline (DISA STIG or CIS) and document every deviation.
Supply chain evidence	Maintain a component inventory (SBOM) and provenance for everything you deploy.

The six steps that follow address these areas in sequence. Each step reduces the work left for the next one.

Step 1: Replace Third-Party Images with Curated Images

Swapping open-source and vendor images for curated equivalents removes the largest block of inherited CVEs with no code changes. Third-party images are the software you run but do not build: identity (Keycloak), messaging and streaming (Kafka, Kafka Connect, Flink operators), observability (Fluentd, node-exporter), and cluster tooling (cert-manager, KEDA, kubectl). They often carry the most CVEs because you cannot patch them at the source.

RapidFort Images are curated drop-in replacements. They keep the original components, size, and major/minor version tags; a patched image simply no longer lists the fixed CVEs. The catalog covers more than forty thousands images across Ubuntu, Red Hat UBI, Debian, Alpine, Oracle Linux, and Amazon Linux, and images are rebuilt at least twice a week.

Zero or near-zero CVE images ■ Same tags, same components ■ Cosign-signed ■ SLSA Build Level 3 provenance

How to do it

1. **Inventory and baseline.** List every third-party image in your registries and Helm charts, and scan each to record its starting CVE count.

SHELL

```
rfscan keycloak/keycloak:25.0.6 --project-name "FedRAMP baseline" --reports-folder ./baseline
```

```
################################################################################################ 100.0%
2026-10-12 19:11:24,531: Scanning SBOM for vulnerabilities ...
2026-10-12 19:11:25,109: [0s]:     Downloading remote resources to the server... [100.0%]
2026-10-12 19:11:27,530: [2s]:     Creating vulnerabilities report for keycloak/keycloak:26.8.0...
2026-10-12 19:11:27,722: [0s]:     Uploading results to remote storage... [100.0%]
2026-10-12 19:11:27,932: Creating reports ...
2026-10-12 19:11:27,934: Downloading 2 files ...
2026-10-12 19:11:28,720: Downloaded 2 files, 1.17 MB.
2026-10-12 19:11:28,872: Saved reports in /Users/geoman/baseline: ['keycloak-keycloak-26.8.0_sbom.csv', '
k-keycloak-26.8.0_summary.csv']
2026-10-12 19:11:28,889: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: Image Name:                 keycloak/keycloak:26.8.0
2026-10-12 19:11:28,889: Image Size:                 466.36 MB
2026-10-12 19:11:28,889: Attack Surface:             466.36 MB
2026-10-12 19:11:28,889: Runnable Attack Surface:    250.44 MB
2026-10-12 19:11:28,889: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: OS:                         Red Hat Enterprise Linux 9.8 (Plow)
2026-10-12 19:11:28,889: Total Packages:             563
2026-10-12 19:11:28,889:            RPM:             43
2026-10-12 19:11:28,889:           Java:             520
2026-10-12 19:11:28,889: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: No supply chain risk found
2026-10-12 19:11:28,889: Packages analyzed for malware, tampering and reputation.
2026-10-12 19:11:28,889: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: Total Vulnerabilities:      39
2026-10-12 19:11:28,889:       Critical:             0
2026-10-12 19:11:28,889:           High:             5
2026-10-12 19:11:28,889:         Medium:             19
2026-10-12 19:11:28,889:            Low:             11
2026-10-12 19:11:28,889:        Unknown:             4
2026-10-12 19:11:28,889: ──────────────────     ──────────
2026-10-12 19:11:28,889:       Exploits:             11
2026-10-12 19:11:28,889:          Fixes:             17
2026-10-12 19:11:28,889:           KEVs:             0
2026-10-12 19:11:28,889:    RF Advisory:             3
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: ..................................................................
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,889: Improvement estimates -- if hardened:
2026-10-12 19:11:28,889: Attack Surface:              466.36 MB -> 163.97 MB  ~65%
2026-10-12 19:11:28,889:  Package Count:                    563 -> 423        ~25%
2026-10-12 19:11:28,889:    Vulns Count:                     39 -> 11         ~72%
2026-10-12 19:11:28,889: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,889:
2026-10-12 19:11:28,902: Your vulnerability database was last updated on: 2026-10-12
2026-10-12 19:11:28,938: Scan finished for keycloak/keycloak:26.8.0
2026-10-12 19:11:28,939:    Started          : 2026-10-12 19:11:19,214
2026-10-12 19:11:28,939:    Finished         : 2026-10-12 19:11:28,936
2026-10-12 19:11:28,939:    Elapsed          : 9s
2026-10-12 19:11:28,940: ──────────────────────────────────────────────────────────────────
2026-10-12 19:11:28,940:
2026-10-12 19:11:28,943:    Total size       : 466.36 MB
2026-10-12 19:11:28,944:    Speed            : 51.82 MB / sec
```

2. **Find the curated match.** Search the catalog by repository, then filter by version and OS.

SHELL

```
rfcatalog -r keycloak
rfcatalog -r keycloak -iv ^26.1 -t jammy
```

```
geoman@Mac ~ % rfcatalog -r keycloak -iv ^26.1 -t jammy
MANIFEST                                            VERSION  OS                  ARCHITECTURES + DIGESTS                                                             T/C/H/M/L/U/N           HELM  EOL
--------------------------------------------------  -------  ------------------  ---------------------------------------------------------------------------------  ----------------------  ----  ---
quay.io/rfcurated/keycloak:26.1.4-jammy-rfcurated   26.1.4   Ubuntu 22.04.5 LTS  arm64 → sha256:155ced372cd3461ba816c1bfe62e45d22c4bc522b4df71847c10bb5af1796132  434/11/132/207/68/16/40  -     -
                                                                                 amd64 → sha256:38450a4cf9474d0b1503b4bfe5b387d4b4c6df9a217e7a0c97f3dc68e3acdb9e
--------------------------------------------------  -------  ------------------  ---------------------------------------------------------------------------------  ----------------------  ----  ---
```

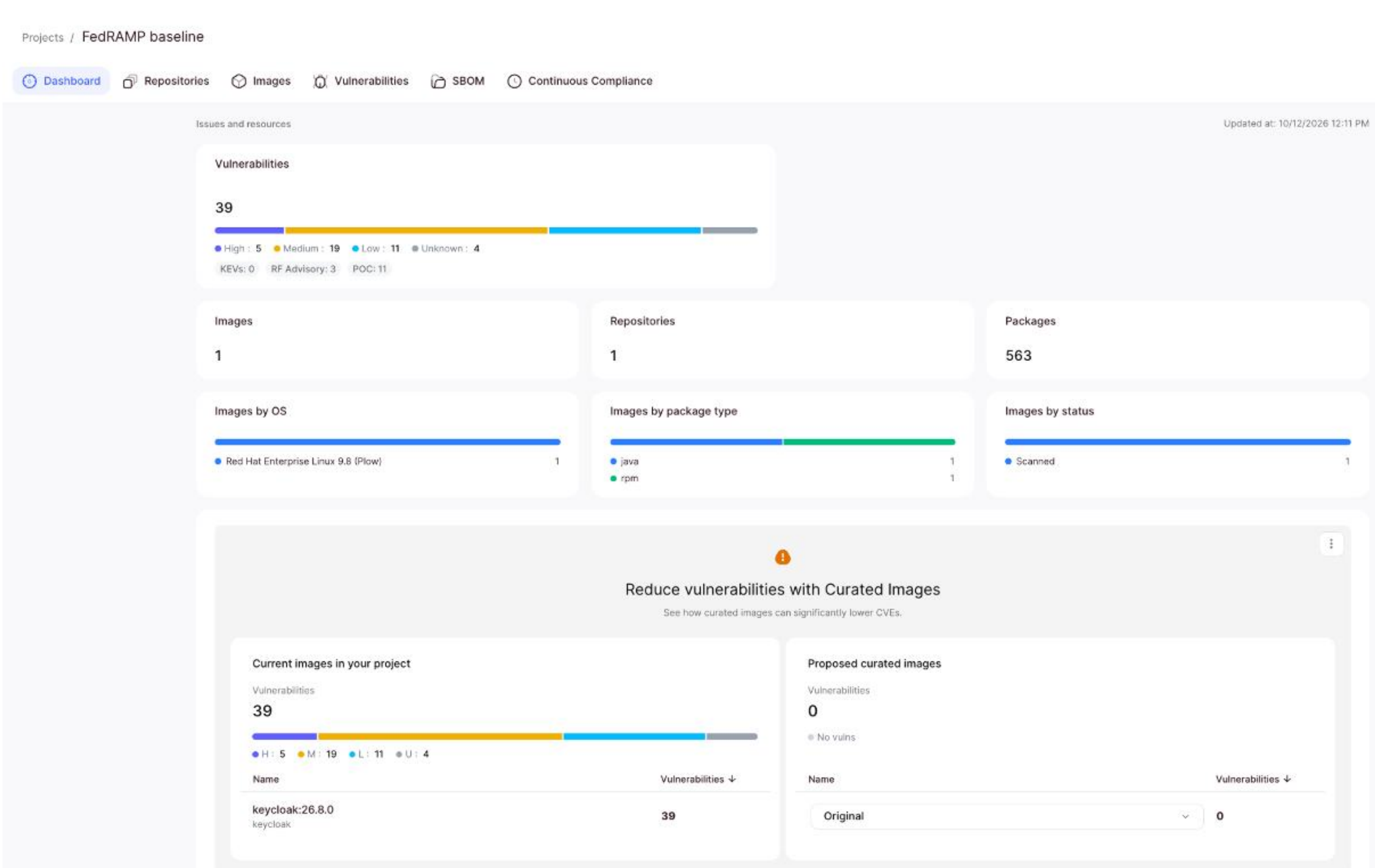
3. **Swap the image reference.** Update the tag in your Helm values or manifests. Use Bitnami-compatible (bnt) variants where your charts expect them.
4. **Verify signature and provenance** at ingestion, before the image enters your internal registry.

SHELL

```
cosign verify --key cosign.pub --insecure-ignore-tlog=true <image>
```

5. **Rescan, compare, and pin by digest,** so admission control (Kyverno or Gatekeeper) only admits signed, approved images.

Here is the dashboard output of the project called “FedRAMP baseline” comparing Keycloak:26.8.0 with 39 CVEs on the left hand side compared to the RapidFort Keycload:26.8.0 down to 0 CVEs on the right. This kind of platform dashboard is critical to show you results of daily scans and available alternatives from curated catalogs of near zero CVE alternatives.



New curated images are built daily. Check your compliance dashboard regularly and upgrade to the latest version to stay ahead of newly disclosed CVEs. Typical SLAs are shown below.

Patch service levels to plan around

Severity	Response	Target resolution
KEV (CISA-listed)	30 minutes	1 calendar day
Critical	30 minutes	7 calendar days
High	2 hours	7 calendar days
Moderate	Next business day	21 calendar days
Low	2 business days	28 calendar days

RapidFort standard targets, measured from upstream patch availability; your commercial agreement governs. FedRAMP clocks start at CVE discovery, which is tighter, so build that gap into your plan.

Step 2: Replace First-Party Images by Rebasing on Curated Framework Images

Your own applications inherit most of their CVEs from the base image in their `FROM` line, so rebasing them on curated framework images is usually a one-line change per Dockerfile. First-party images are the services your teams build and maintain: public APIs, internal services, data-access layers, and webhooks. Rebuild them on curated .NET, Node.js, Java/JDK, Python, Alpine, Oracle Linux, or Ubuntu bases; you keep your existing OS family and language stack.

How to do it

- **Baseline:** Build the current Dockerfile as-is and record the CVE count with `rfscan`.
- **Find the matching base:** Use `rfcatalog` to locate the same runtime and OS, for example `rfcatalog -r python -iv 3.13 -t jammy`. Confirm the exact, fully patch-versioned tag before committing it.
- **Swap every relevant `FROM` line,** including build stages in multi-stage Dockerfiles.

DOCKERFILE

```
# Before
FROM python:3.13-bookworm

# After (confirm the current tag with rfcatalog)
FROM quay.io/rfcurated/python:3.13.14-jammy-rfcurated
```

- **Rebuild, test, and rescan** against the baseline.
- **Gate the pipeline** so regressions cannot ship. `rfscan` exits with code 2 when thresholds are exceeded, and a Dockerfile linter such as Hadolint can reject any `FROM` outside your trusted registry.

SHELL

```
rfscan myapp:latest --max-critical 0 --max-high 5
```

Expected results example for a Java application

Stage (Java 21 Spring Boot on Debian 12)	Image size	CVEs	Packages
Original image	839 MB	101	244
After curated base swap	528 MB	4	170
After swap + hardening (Step 4)	241 MB	0	63

Illustrative; results vary by image and application. A 60%+ reduction from the base swap alone is common. Tags roll forward frequently, so treat any tag in this paper as an example.

Step 3: Enable FIPS-Validated Cryptography

Every image that encrypts, signs, hashes, or terminates TLS for federal data must use FIPS-validated cryptographic modules. The fastest route is to move those workloads to FIPS-tagged curated images rather than building and maintaining FIPS toolchains yourself. In the assessment above, about 40 images needed FIPS variants, spanning identity, APIs, messaging, and cluster webhooks.

What RapidFort FIPS images provide

Validated providers

FIPS 140-3 validated OpenSSL providers, covering downstream consumers such as cURL.

Java coverage

BouncyCastle FIPS injected at startup; the app will not start if it fails to load.

Kernel-tested

Every FIPS image is tested with the FIPS kernel enabled.

Strict mode

`strict-fips` variants hard-fail if a non-FIPS algorithm is called.

How to do it

- **Map crypto boundaries:** List each service that uses cryptography and the library it calls (OpenSSL, Java JCE, Go crypto, language-native libraries).
- **Select FIPS variants:** The `fips` tag means the image's cryptographic boundary maps to a module certificate in NIST's CMVP database.

```shell
rfcatalog -r awscli -t fips,focal
```

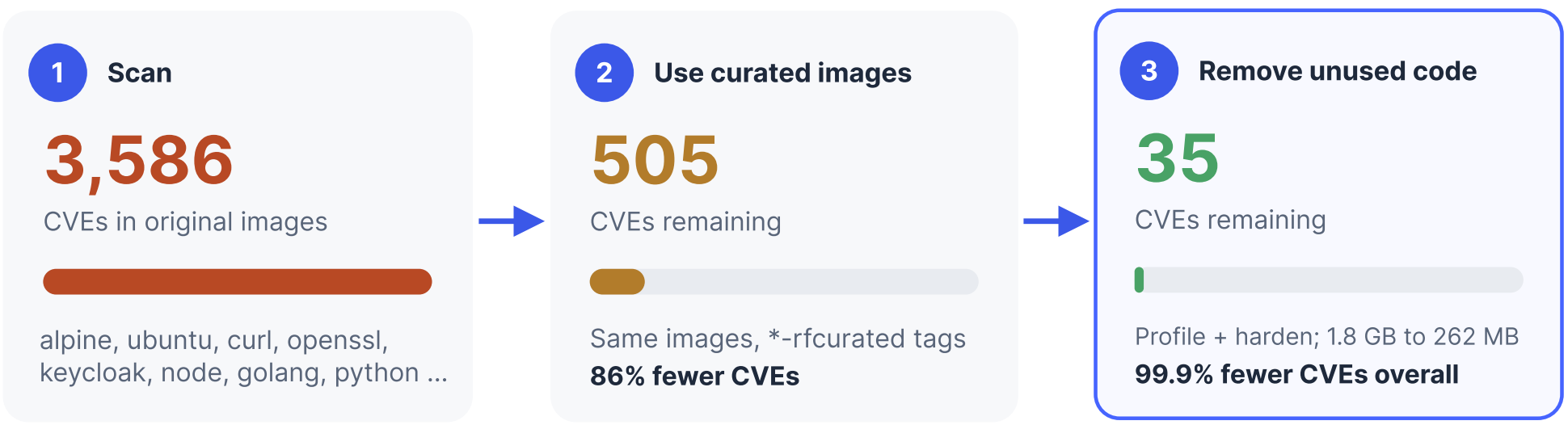
- **Configure the application:** FIPS images do not automatically force strict FIPS mode in your app. Set approved algorithms, cipher suites, and TLS settings explicitly, or use `strict-fips` images to enforce it.
- **Run on FIPS-enabled hosts:** Container cryptography relies on the host kernel.
- **Test for non-approved algorithm calls,** ideally with the strict variant in staging, before production cutover.
- **Record module certificates** in your System Security Plan, mapping each image to its CMVP certificate.

Evaluate random-number-generator requirements separately; FIPS images do not cover them automatically. NIST's CMVP is transitioning from FIPS 140-2 to FIPS 140-3, so confirm the current status of each certificate you cite.

Step 4: Optimize Images to Reduce the Attack Surface

Profiling and optimization removes the code your application never executes. That eliminates CVEs with no upstream patch and lowers the odds of future ones. Curated images fix what can be patched; optimization removes unused shells, package managers, debug utilities, and libraries that are present but never loaded. Fewer components also mean fewer new CVEs each month, which keeps the Step 6 reporting burden flat.

VULNERABILITY REDUCTION EXAMPLE: 13 PUBLIC IMAGES



Bars drawn to scale against the original 3,586 CVEs

Adapted from the RapidFort vulnerability reduction example in the FedRAMP remediation roadmap.

Typical outcomes

- Attack surface reduction up to 90% on well-profiled workloads.
- Image size reductions of 40–80%; in the assessment above, the estimated footprint of about 90 containers fell from roughly 20 GB to 6 GB.
- Lower infrastructure cost and faster container start-up.

How to do it

- **Instrument the image.** `rfstub` injects profiling instrumentation and appends `-rfstub` to the name. Use `-s bpfwrap` on ARM or macOS.

```shell
rfstub myapp:latest
```

- **Exercise the application** in test, QA, or staging, ideally with your automated test suite. Anything not observed is removed later.
- **Watch and snapshot (optional).** `rflens` streams live file activity; `rfrbom` captures the Runtime Bill of Materials (RBOM®) of what actually runs.
- **Optimize with a preset** that matches your risk tolerance.

```shell
rfharden myapp-rfstub --preset aggressive        # remove everything unused (default)
rfharden myapp-rfstub --preset standard          # remove unused components with any vulnerability
rfharden myapp-rfstub --preset light             # remove unused Critical/High components only
```

- **Pin hard-to-exercise paths** with a profile file (`rfharden myapp-rfstub -p profile`) instead of re-testing. A hardening profile (profile file) is a list of entries naming the components that should **always be retained** in a hardened image, even if runtime profiling didn't detect them as used.
- **Reuse profiles across releases** with `-t <workload-tag>` when functionality in your application has not changed and you don't need to rerun a full test cycle.

> *Hardened images contain no instrumentation or tracing code. Start with light or standard on critical services, then move to aggressive once test coverage is proven.*

Step 5: Run STIG Benchmarks

Once images are clean, FIPS-enabled, and minimized, benchmark them against DISA STIG profiles and either remediate or formally document every failing rule. Running STIG last on hardened images means fewer rules to fix and fewer deviations to justify. STIGs check configuration, not CVEs. Assessors expect a repeatable scan, a remediation record, and a documented rationale for each rule you do not apply.

Choose your base OS with STIG coverage in mind. DISA publishes official STIGs for Red Hat Enterprise Linux and Ubuntu. Vendor-specific container distributions such as Wolfi and Chainguard OS have no DISA-published STIG. Chainguard offers its own profile based on DISA's General Purpose OS SRG, but it is self-published, not DISA-issued. Henry J. Sienkiewicz, former DISA CIO and Authorizing Official, argues that "a large part of the value of a STIG lies in its independence," and that self-published STIGs sever the chain of trust behind federal accreditation (Self-Published STIGs Are Not a Breakthrough — They're a Breakdown, November 2025).

Images without a DISA-published baseline typically require a deviation request, which can take an estimated 30–90 days to resolve and can delay FedRAMP authorization. Building on curated UBI or Ubuntu images in Steps 1 and 2 avoids this gap. Use the following DoD Cyber Exchange link to verify if your base OS is DISA approved and has a corresponding STIG

How to do it

- **Start from STIG-hardened bases where available:** RapidFort publishes STIG-hardened images for Ubuntu, Red Hat UBI, and Amazon Linux.
- **Find the right benchmark profile and scan each image against it,** attaching results to a compliance project.

```shell
rfcat -l    # list available benchmark profiles
rfcat <image> -b <stig-profile-id> --project-name "FedRAMP STIG" -s ./stig_reports
```

- **Remediate:** Generate a remediated Dockerfile with `-a`, or apply your own script with `-r remediation.sh`, then rescan.
- **Tailor and document deviations:** Remove rules that do not apply to containers with a tailoring file `(-t tailor.json)`, and record each justification in your System Security Plan or deviation log.
- **Scan hosts too** with rfcat `--scan-host` for the nodes that run your workloads.
- **Re-run on every rebuild** so drift is caught before release. CIS benchmarks are also available through `rfcat` if your agency sponsor accepts them.

Step 6: Operate on a FedRAMP Platform for Continuous Reporting

Steps 1–5 get you clean; Step 6 keeps you clean and proves it every month. FedRAMP's VDR rules require providers to report vulnerability detection and response activity to all necessary parties at least monthly, in a consistent, human-readable format.

Monthly evidence package

Report	What it shows	How to produce it
CVE report	Open findings by severity, KEV status, exploit availability, contextual severity, and RF Advisory determinations	`rfscan <image> --reports-folder ./monthly --formats csv`
SBOM	Full component inventory per image, in SPDX or CycloneDX	`rfscan <image> --reports-folder ./monthly --formats spdx-json,cdx-json`
STIG / CIS report	Pass, fail, and tailored rules per image and host	`rfcat <image> -b <profile> -s ./monthly`
Project roll-up	Aggregate posture across every image in the boundary	`rfprojects -r <project-id> --reports-dir ./monthly`
Scanner enrichment	Existing Wiz, Trivy, Prisma, or Grype results annotated with curated-image patch data	`rfmod <image> -f report.json -t wiz`
Update and fix log	Newer curated digests and the CVEs they fix	`rfcatalog --check-update <image> --fixed-cves`

Automate these in CI/CD with a service account so the package builds itself on a schedule. RF Advisory determinations provide evidence-backed false-positive records, which reduces the findings you must carry and explain.

Why you also need a BOD 26-04 compliance dashboard

Monthly reports show where you were; FedRAMP's new rules require you to know, per finding, how much time you have left. CISA issued BOD 26-04 on June 10, 2026. It replaces CVSS-driven patch schedules with deadlines based on four risk factors, some as short as three days.

FedRAMP responded in Public Notice NTC-0014 by making its VDR and VER rules mandatory for every cloud service obtaining or maintaining certification on **December 7, 2026**. Services not following them can stay certified under a corrective action plan only through **March 7, 2027**; after that, certification is revoked.

THE FOUR BOD 26-04 FACTORS AND THE FEDRAMP RULES THAT COVER THEM

Internet exposure

Is the asset publicly reachable?

VER-EVA-EIR

KEV status

Is the CVE in CISA's KEV catalog?

VER-EVA-ELX, VDR-TFR-KEV

Automatable

Can exploitation be automated? Assume yes without evidence.

VER-EVA-AIA

Technical impact

Total or partial control after exploitation?

VER-EVA-EFA

MAXIMUM REMEDIATION TIMELINES UNDER BOD 26-04

Internet-exposed	KEV	Automatable	Technical impact	Maximum time
Yes	Yes	Yes	Total	**3 days + forensic triage**
Yes	Yes	Yes	Partial	**3 days**
Yes	Yes	No	Total	**3 days + forensic triage**
Yes	Yes	No	Partial	**14 days**
Yes	No	Yes	Total	**3 days**
Yes	No	Yes	Partial	**14 days**
Yes	No	No	Total	**14 days**
Yes	No	No	Partial	**60 days**
No	Yes	Yes	Total	**3 days + forensic triage**
No	Yes	Yes	Partial	**14 days**
No	Yes	No	Total	**14 days**
No	Yes	No	Partial	**14 days**
No	No	Yes	Total	**60 days**
No	No	Yes	Partial	**60 days**
No	No	No	Total	**Fix on system upgrade**
No	No	No	Partial	**Fix on system upgrade**

Source: CISA BOD 26-04 These are maximums; FedRAMP expects providers to move faster.

What the dashboard must show

Because VER-EVA-AIA treats every finding as automatable by default, an exposed KEV finding should be handled as a 3-day item unless you can prove otherwise. A BOD 26-04 dashboard turns the matrix into a live, per-finding view. At minimum it should:

- Record all four factors for every finding on every asset, with the evidence behind each answer.
- Calculate the deadline automatically and show days remaining, starting the clock at discovery.
- Flag 3-day items, especially those requiring forensic triage, the moment a CVE is added to KEV.
- Track mitigations that change the clock, such as removing internet exposure, separately from full remediation.
- Separate accepted vulnerabilities and evidence-backed false positives from open work.
- Export the monthly human-readable report and the machine-readable data FedRAMP increasingly expects.

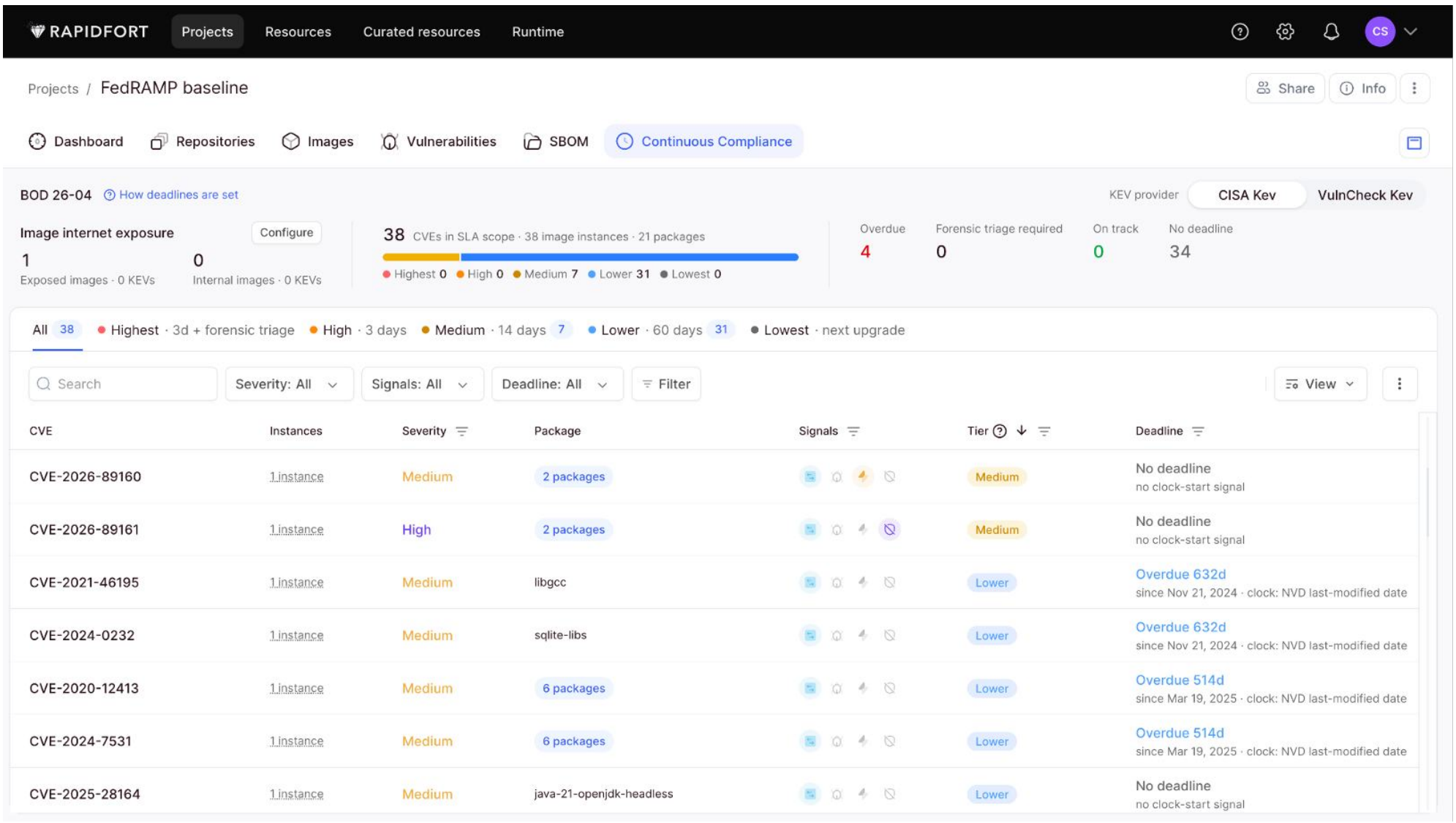
The earlier steps make this dashboard manageable. Curated, hardened images leave far fewer findings to track, and RapidFort scan data already supplies KEV status, exploit availability, and false-positive determinations as inputs.

Turning CISA's directive into live clocks

The dashboard described in the previous section is already available as part of RapidFort Continuous Compliance. RapidFort Continuous Compliance is a BOD 26-04–aligned capability in the RapidFort Platform that gives every container CVE its own live remediation clock. It combines CISA KEV and Vulnrichment data, refreshed every 15 minutes, with operator-defined exposure to score each finding against the directive's four risk signals and sort it into one of five tiers: from three days plus forensic triage to next system upgrade. Tiers, deadlines, and SLA status recalculate whenever risk changes, across scanned images and running workloads. Every determination links to its source evidence, and remediation is one step away through curated near-zero CVE images and automated hardening.

RapidFort Continuous Compliance delivers the following benefits:

- **Precise Risk Prioritization:** Identifies the critical subset of CVEs that actually trigger a 3-day BOD 26-04 remediation clock, ensuring triage efforts focus on real risk rather than inherited noise.
- **Defensible Compliance:** Replaces subjective SLA debates with transparent, CISA-sourced evidence (combining KEV, Vulnrichment, and runtime exposure) that auditors, AOs, and reviewers accept.
- **Streamlined Audits & Faster ATOs:** Automatically generates continuous audit evidence and reporting-ready exports aligned with CISA's standardized data schema, accelerating FedRAMP, ATO, and cATO cycles.
- **Backlog Reduction via Elimination:** Pairs live compliance clocks with direct remediation paths such as curated near-zero CVE images and automated hardening allowing teams to retire clocks by removing vulnerable components entirely.
- **Unified Build-to-Runtime Model:** Applies consistent risk scoring across CI/CD, registries, curated images, and live container instances so your security posture remains accurate as workloads drift.



RapidFort Continuous Compliance dashboard example

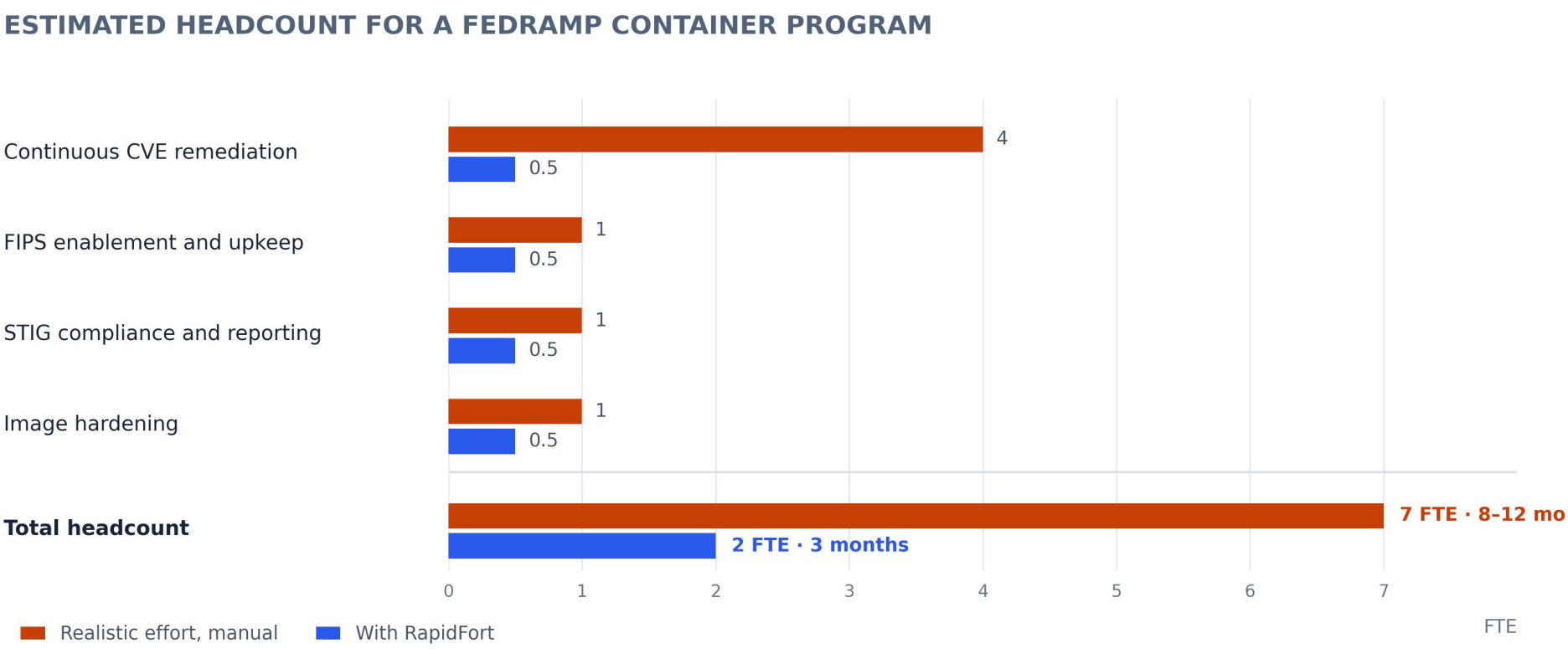
Continuous End-to-End Threat Elimination

RapidFort Runtime extends the RapidFort Platform beyond build-time optimization by deploying lightweight eBPF and ptrace instrumentation into live production containers to observe actual software behavior without requiring code changes. As a core component of the end-to-end security suite - bridging intake, build, and runtime - Runtime addresses the critical security gap where production environments constantly drift due to hourly CVE disclosures, package updates, and unmonitored process changes. By continuously capturing execution evidence, it establishes a reliable runtime baseline and generates a dynamic Runtime Bill of Materials (RBOM®), ensuring organizations have real-time visibility into what is actively executing across their live environments.

For organizations pursuing or maintaining FedRAMP authorization, RapidFort Runtime delivers essential audit-ready evidence and automated continuous monitoring required by rigorous federal standards like BOD 26-04 and NIST guidelines. By proactively detecting runtime tampering, correlating newly disclosed vulnerabilities with deployed workloads, and eliminating false positives through direct execution profiling, Runtime drastically reduces manual compliance overhead. This continuous validation empowers security teams to maintain a hardened posture, substantiate their continuous monitoring deliverables, and accelerate authorization cycles without slowing down software delivery.

Putting It Together

Run the steps in order and plan backward from December 7, 2026. Steps 1 and 2 deliver the largest CVE reduction fastest; Steps 3–5 close the compliance gaps; Step 6 keeps the result current. Together they shrink the team a FedRAMP program needs.



Adapted from the headcount estimate in a RapidFort FedRAMP remediation roadmap for a ~90-container estate. Illustrative only; your figures will vary.

Readiness checklist

Inventory every first- and third-party image in the authorization boundary and record baseline CVE counts.

Replace third-party images with curated equivalents and verify signatures and provenance.

Rebase first-party Dockerfiles on curated framework images and add CI threshold gates.

Identify crypto-handling services, move them to FIPS variants, and record CMVP certificates.

Profile and harden images, starting with the services carrying the most CVEs.

Run STIG benchmarks on images and hosts; remediate or document every deviation.

Automate the monthly CVE, SBOM, and STIG evidence package.

Stand up a BOD 26-04 dashboard tracking all four factors and deadlines per finding.

Confirm the plan with your 3PAO and agency sponsor before December 7, 2026.

Confirm Continuous compliance dashboard and export of artifacts for monthly self attestation.

Closing Remarks

FedRAMP readiness for containers is not a one-time cleanup. It is a disciplined, sequential program: start from curated images, rebuild your own applications on clean bases, add validated cryptography, strip what you do not use, prove secure configuration, and report continuously. With BOD 26-04 and FedRAMP's new vulnerability rules compressing deadlines to as little as three days, the organizations that succeed will be those that reduce their finding volume first and automate their evidence second.

RapidFort enables this model end to end, so security and development teams can meet federal requirements without slowing delivery.

Sources

- FedRAMP NTC-0014: FedRAMP Response to CISA BOD 26-04 (June 16, 2026)
- FedRAMP Vulnerability Detection and Response
- CISA BOD 26-04: Prioritizing Security Updates Based on Risk (June 10, 2026)
- BOD 26-04: Prioritizing Security Updates Based on Risk
- Henry J. Sienkiewicz: Self-Published STIGs Are Not a Breakthrough — They're a Breakdown (LinkedIn, November 2, 2025)
- Chainguard: STIGs for Chainguard Containers
- RapidFort Customer Guide, v2.1 (September 2026): capabilities, commands, results, and service levels

RapidFort results vary with your baseline and profiling coverage. Service levels are governed by your executed agreement. Federal requirements change often; confirm against current FedRAMP and CISA publications.

RAPIDFORT

ABOUT RAPIDFORT

RapidFort is the leader in Software Supply Chain Security, enabling organizations to eliminate risk across their software stack at scale. Its platform combines curated near-zero CVE container images, runtime profiling, and attack surface management to remove up to 99.9% of vulnerabilities within hours and reduce the attack surface by up to 90%, without code changes. RapidFort appears in the inaugural Gartner® Magic Quadrant™ for Software Supply Chain Security and is a Gartner® Cool Vendor™ and Nutanix Global Cloud Native Partner of the Year.

© 2026 RapidFort, Inc.

Eliminate attack vectors at the source

Schedule a Call ↗

www.rapidfort.com | sales@rapidfort.com