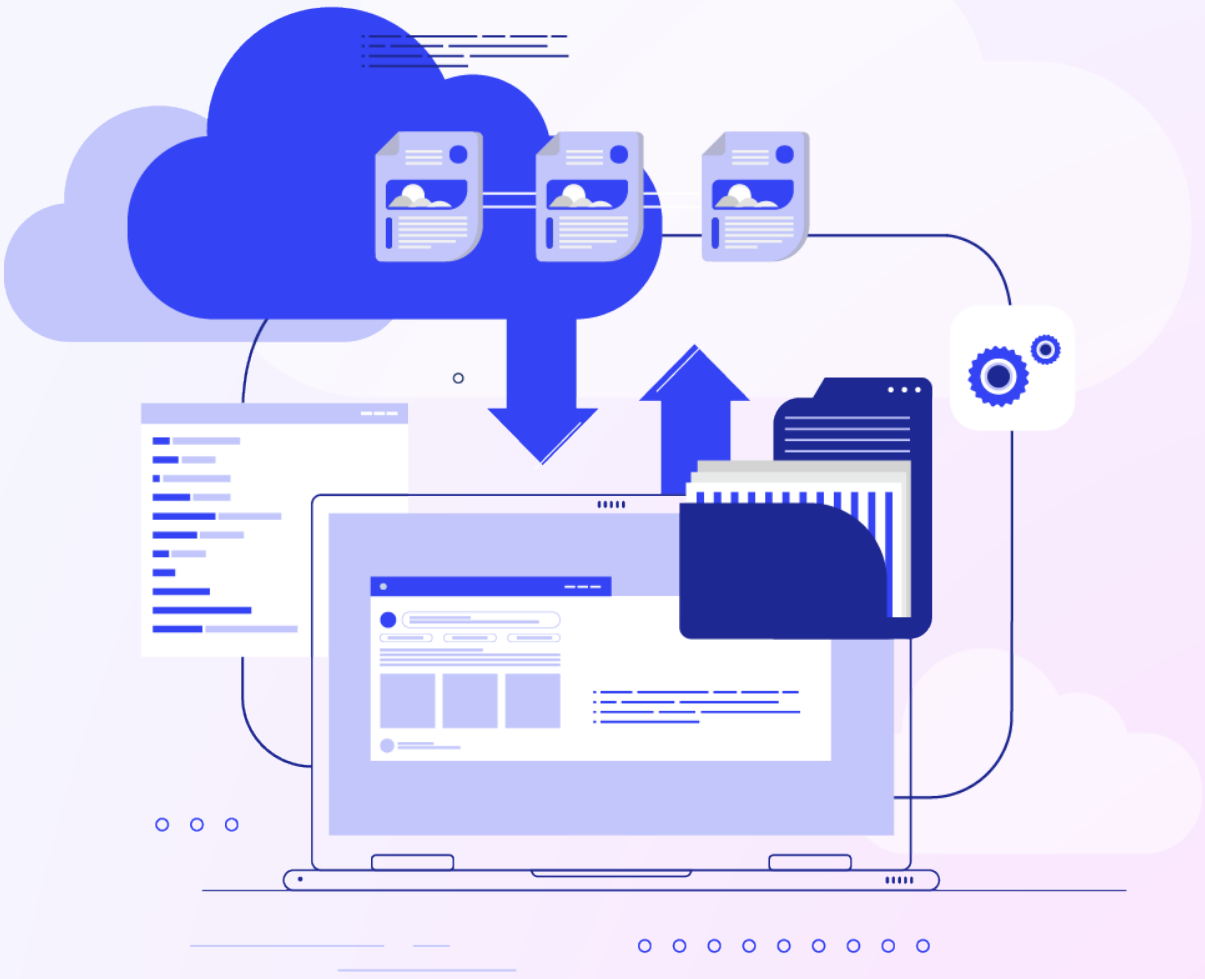
The State of Hardened Container Images Report

The Search for Hardened Container Images



Chainguard

John Speed Meyers and Paul Gilbert
Chainguard Labs
2024

Executive Summary

If a malevolent attacker could wish an insecure-by-default and pervasive software delivery model into existence, it's possible they would wish for the container image status quo. Most popular container images have hundreds of known software vulnerabilities (i.e. CVEs). This is partially the result of software bloat—too much unnecessary software being stuffed into a container image—and partially the result of slow update cycles. Most attempts to date to fix this problem have been incremental. Consequently, battling this problem, as many software teams will attest, often devolves into a nightmare.

In search of hardened containers, or containers with few or zero CVEs, this whitepaper therefore surveys the container image landscape. While there are occasional glimpses of security, this research indicates none offer a practical, secure-by-default approach.

Salient findings include:

- Popular Debian-based, community-supported images that have a Chainguard Images equivalent have, on average, nearly 300 CVEs. This number of CVEs is at least in part due to these container images including, on average, nearly 300 components, or open source packages.
- Simply updating the packages in a sub-sample of Docker official images to the latest version available in the underlying linux distribution provides only a modest five percent reduction in the total count of CVEs.
- Detailed analysis of one container "debloating" technology, which finds a CVE reduction rate of approximately 65 percent, suggests that this technique is only moderately effective.
- An analysis of Red Hat-provided container images suggests that these images contain, on average, nearly 200 CVEs. This count excludes the hundreds of CVEs that Red Hat's security team has labeled "will not fix."
- The 50 most downloaded images in Iron Bank, an Air Force repository of hardened container images, have, on average, 110 CVEs.
- Canonical's Chiselled images have few or no CVEs and are minimal, but the collection of available images is currently small and adoption requires power-users.

In other words, the status quo is dismal. There are options, but not many good ones. Fortunately, a large collection of minimal, hardened container images with low-to-zero CVEs is not science fiction, as the patient reader will discover.

Introduction

Open source software security has become a hot topic. The White House and the U.S. Department of Homeland Security Cybersecurity and Infrastructure Security Agency now hold regular summits on open source software security. Open source software vulnerabilities sometimes make it to headlines of major newspapers. But the closely related topic of container security has, to date, played second fiddle.

Readers should know that container security is open source software security. Containers are simply a means of bundling together a software application and its dependencies. Containers enable easy deployment of software to the cloud. In practice, containers often contain lots of open source software. Analysis presented later in this report suggests that some of the most downloaded containers on Docker Hub, a popular means of storing and distributing containers, include approximately 270 open source software components. Each open source component is the result of one or more contributors collaborating and releasing their code for others to inspect, modify, and distribute.

Importantly, because open source software licenses mean the software is provided "as-is," the software vulnerabilities in these open source components, often called CVEs, become the problem of the organization operating that container. These CVEs mean time and toil for engineers and security and compliance risks for the organization. (More on how many CVEs are in popular containers shortly.) Again, container security is open source software security.

To shed light on the current security of containers, and by extension on the security of an important part of the open source software ecosystem, this study examined the number of CVEs in a range of container image ecosystems. The study searches for "hardened" containers, those with few or zero CVEs, an admittedly simple definition that we will dissect later. This analysis starts with a subset of Docker official images that also have a Chainguard Images equivalent. The report then evaluates a typical DIY approach to container security–explicitly updating out-of-date components within a container–and examines container "debloating" technology. Next, the report examines offerings from Red Hat, the U.S. Air Force's Iron Bank, and Canonical. The analyses also include comparative results from equivalent Chainguard Images, which are themselves explained more fully at the end of the report. While there are occasional glimpses of security, none offer a practical, secure-by-default approach.

The results are sobering. Using the popular open-source scanner Grype, most popular container images have hundreds of CVEs. Even when the analysis includes only high and critical severity vulnerabilities, the number of CVEs often totals in the dozens. This finding matches both industry analysis, which has focused mostly on images in Docker Hub, and also results from an often-overlooked academic literature on containers and vulnerabilities. This report, however, makes a novel contribution by extending the analysis beyond Docker official images to a range of ecosystems and CVE reduction techniques.

Of course, naysayers will shake their heads. But we love naysayers here–please contact us to consider conducting joint research ("adversarial collaboration") to get to the bottom of our differences. A later section will cover typical objections to this line of research, especially including the case for focusing on CVEs rather than, say, open source package malware or other popular topics.

Where does the search lead? To us, only minimal images derived from a Linux distribution devoted to both rapid updates and CVE patches can achieve secure and developer-friendly container images. Chainguard therefore built Chainguard Images, hardened, minimal images from Wolfi, a distribution designed for fast CVE patching and developer-first workflows. Chainguard Images generally have few or zero CVEs, reducing the attack surface and liberating defenders from the toil of constant CVE patching.

The Status Quo: Debian-Based, Community-Supported Images

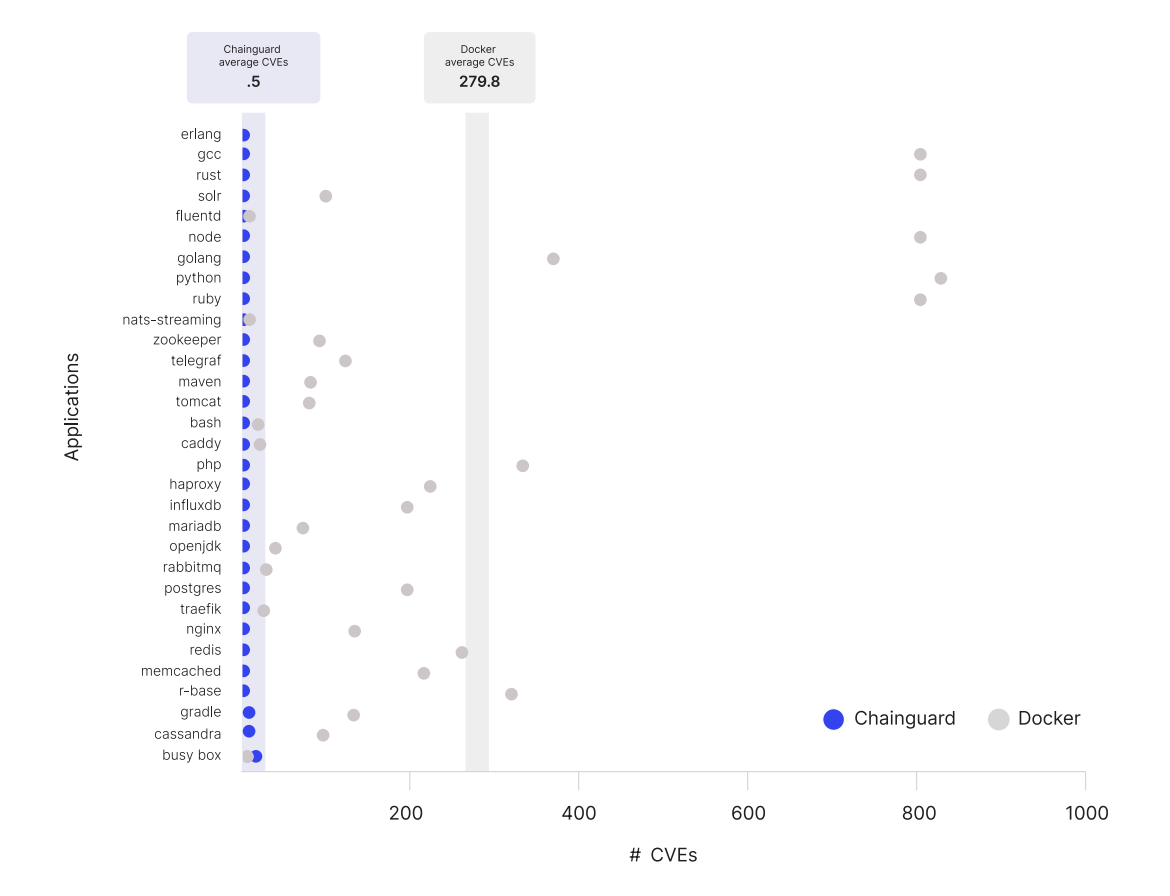
When many software developers begin learning about container images, the community-supported official images on Docker Hub is their first stop. Docker, the company that made containers popular, offers a set of curated images on Docker Hub, its container registry, and calls these images "Docker Official Images."

> *Note: Chainguard currently partners with Docker to host our secure, hardened images on Docker Hub. This analysis is not intended to suggest that Docker Hub is an insecure platform.*

A simple starting point was a vulnerability scan of the :latest tag of nearly 30 Debian-based official images that also have an equivalent Chainguard Image. This analysis, and all later ones, uses Grype, a popular open-source vulnerability scanner capable of scanning containers for CVEs. Figure 1 provides a visual summary of the results.

FIGURE 1.

Total CVEs in Debian-Based Official Images versus Equivalent Chainguard Images



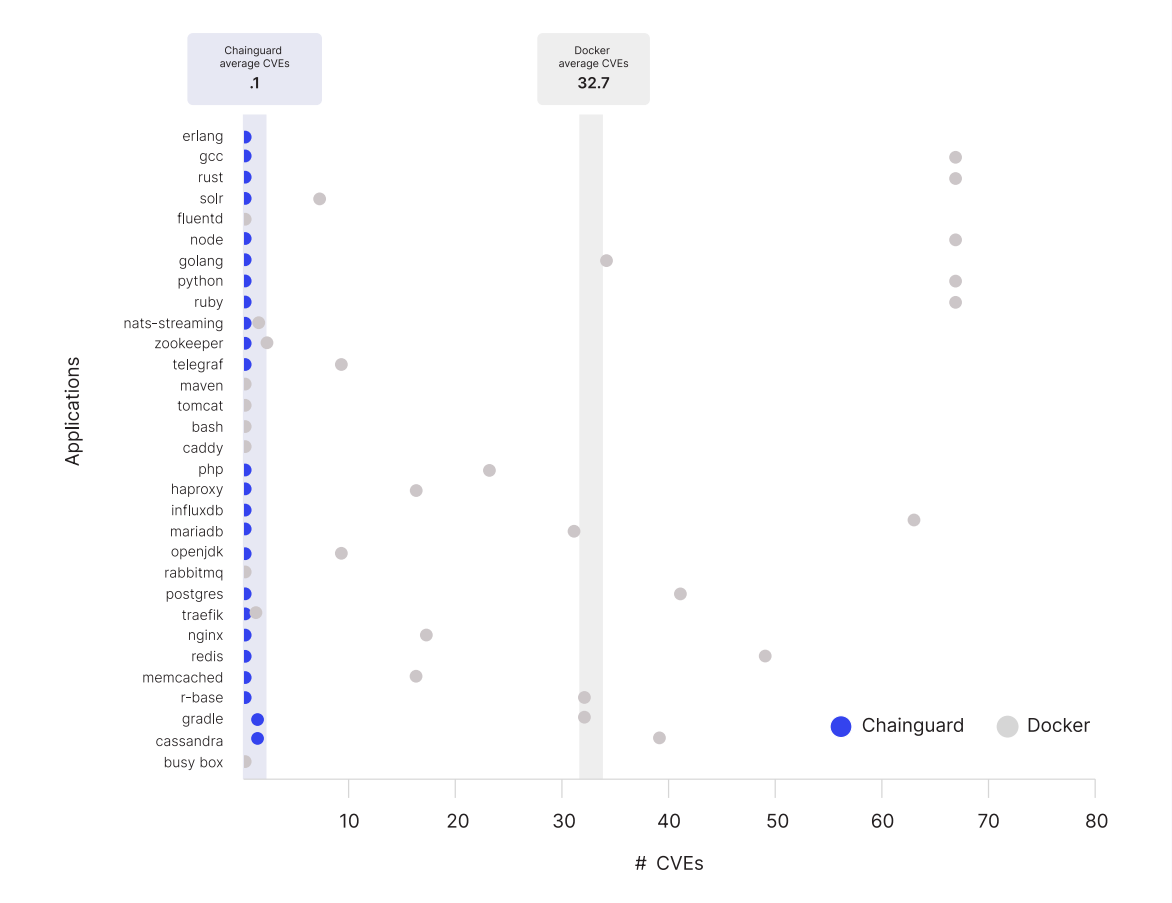
Notes: The erlang image is an extreme outlier and omitted in this plot. The image contained 1462 total CVEs.

This set of Debian-based official images contains, on average, nearly 280 CVEs. However, there is substantial variation. There are a number of images that appear to be outliers with approximately 800 CVEs, and many more with a total count of several dozen to 100 CVEs. Equivalent Chainguard Images, on average, contain less than one CVE.

Next, we performed a similar analysis that included only those CVEs that are high or critical severity CVEs. Figure 2 is the result of this more refined analysis.

FIGURE 2.

Total High and Critical Severity CVEs in Debian-Based Official Images versus Equivalent Chainguard Images



This set of Debian-based official images contains, on average, over 30 high or critical severity CVEs. Many organizations often prioritize these CVEs for prompt remediation. Equivalent Chainguard Images contain, on average, less than one high or critical severity CVE.

Skeptics might wonder if an analysis of only those Debian-based official images that have an equivalent Chainguard Image somehow inflates the CVE counts of official images. Similar industry analysis, which focused on a larger Docker Hub dataset, finds over 300 CVEs per container.

Part of the explanation for these CVE findings can be found in the composition of these images. These analyzed Debian-based official images contain, on average, 273 components (i.e. open source projects bundled as packages). More components means, all things equal, more opportunities for CVEs and, hence, more attack surface. Equivalent Chainguard Images, however, contain only 125 components.

In sum, latest versions of Debian-based official images often have hundreds of CVEs. But a question we've been asked before is: can simple updates or other incremental improvements remove these CVEs from popular images?

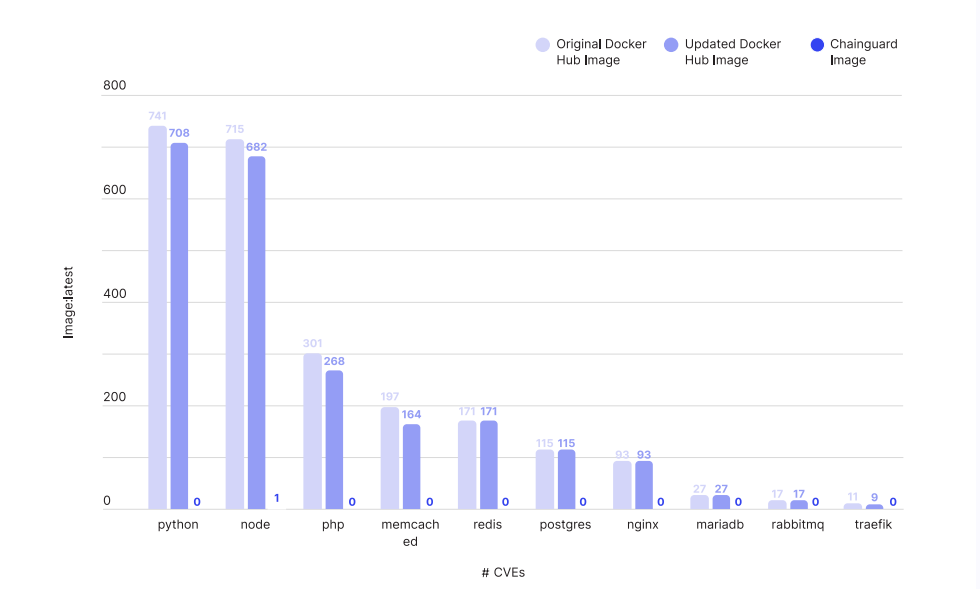
An Incremental Improvement: Frequent Updates of Operating System Packages

Platform teams and security engineering organizations have asked us whether updating the operating system packages in the container is a short-cut to dramatically reducing the CVE count in popular container images. This idea rests on the assumption that the container is composed of Linux distribution packages that have non-vulnerable versions available. Chainguard therefore tried to answer this question. (You can read the full technical details elsewhere.)

We performed a simple experiment. Again, using Grype, we scanned ten popular images (a small subset of Docker official images). Next, we performed updates on all the Debian packages inside these containers, because 98 percent of the existing CVEs were inside Debian operating system packages. We then scanned these updated images with Grype to measure the CVE reduction. Finally, we also scanned 10 equivalent Chainguard Images. Figure 3 resulted.

FIGURE 3.

Total CVE Count for Ten Popular Images, Pre- and Post-Update, and Equivalent Chainguard Images



The analysis reveals that updating packages reduces CVEs by less than six percent. Why the meager benefits? Because the packages associated with even the newest Debian release still contain vulnerabilities.

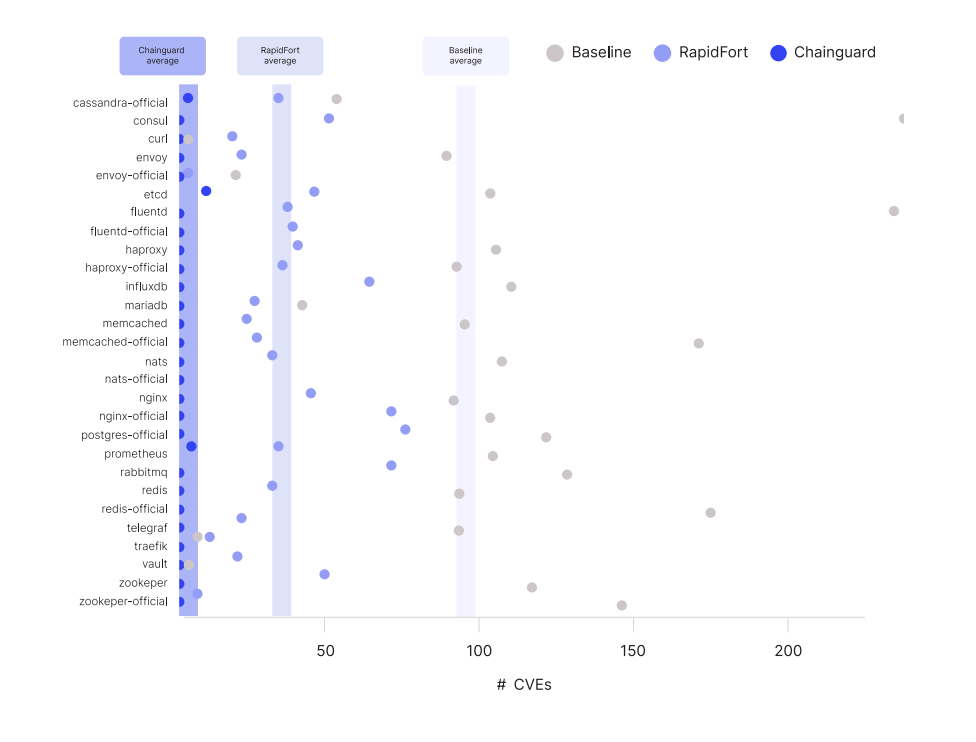
Another Incremental Improvement: “Debloating” Containers

Another conceptual approach to reducing CVEs is to “debloat” containers. This technical method removes unessential components from a pre-existing image. The removal of non-essential components also removes the CVEs associated with those components.

To determine the efficacy of “debloating” containers, Chainguard selected a sample of 28 popular containers. We then scanned three versions of those containers: the original or “baseline” image, a debloated version, and a Chainguard Images equivalent. The debloating technology evaluated in this example was created by the company Rapidfort. The full technical details are elsewhere. Figure 4 visualizes the results.

FIGURE 4.

Total CVE Count for Baseline Images, Debloated Images, and Chainguard Images Equivalents



This analysis finds that debloating reduces the number of CVEs by, on average, 64 percent. It’s a substantial improvement over simply updating operating system packages. The Chainguard Images equivalents in this sample, however, reduce CVEs by, on average, 99 percent.

The Search for Secure Images: Red Hat's Universal Base Images and Beyond

In contrast to modifying or improving existing container images, another approach to finding hardened container images is to look for alternate container images. So we did that. And we started with one of the most common alternatives: Red Hat's Universal Base Image (often abbreviated UBI) and other Red Hat-provided container images.

Red Hat provides four UBI variants, each optimized for different purposes, and advertised as light-weight and enterprise-grade. Table 1 summarizes the four types.

TABLE 1.

Red Hat UBI Variants Names and Descriptions

Variant Name	Description
Platform (ubi)	Supports large applications not optimized for container deployment. Includes a crypto stack, OS utilities, and package manager. This image is most similar to Chainguard's "wolfi-base."
Minimal (ubi-minimal)	Size conscious with basic utilities and a minimal package manager. The closest Chainguard equivalent is glibc-dynamic:latest-dev.
Init (ubi-init)	Full system implementation configured to run at container start.
Micro (ubi-micro)	No package manager; the smallest base image offered. Contains "only the most basic software needed to get a shell inside the container". The closest Chainguard equivalent is glibc-dynamic:latest.

We scanned each variant and a set of equivalent Chainguard Images. The analysis also considered the number of high and critical severity CVEs, the number of components, and the size in MB. Additionally, the analysis considered CVE count with and without CVEs that Red Hat has determined that it will not fix. Red Hat's product lifecycle intentionally does not patch less severe CVEs in the name of software stability.

Table 2 summarizes the comparative analysis of these Red Hat UBI variants with equivalent Chainguard images. All scans were done on the “latest” tag.

TABLE 2.

CVE count, Component Count, and Size in MB for Red Hat UBI images vs Chainguard Equivalents

	Platform	Chainguard
Version	9	wolfi-base
CVEs	72	0
CVEs (+WNF)	188	0
High/Crit CVEs	0	0
High/Crit CVEs (+WNF)	0	0
Components	201	28
Size (MB)	211	15.6

	Minimal	Chainguard
Version	9	glibc-dynamic (latest-dev)
CVEs	30	0
CVEs (+WNF)	42	0
High/Crit CVEs	0	0
High/Crit CVEs (+WNF)	0	0
Components	103	27
Size (MB)	96	48.5

	Micro	Chainguard
Version	9	glibc-dynamic (latest)
CVEs	1	0
CVEs (+WNF)	6	0
High/Crit CVEs	0	0
High/Crit CVEs (+WNF)	0	0
Components	20	7
Size (MB)	23	9.8

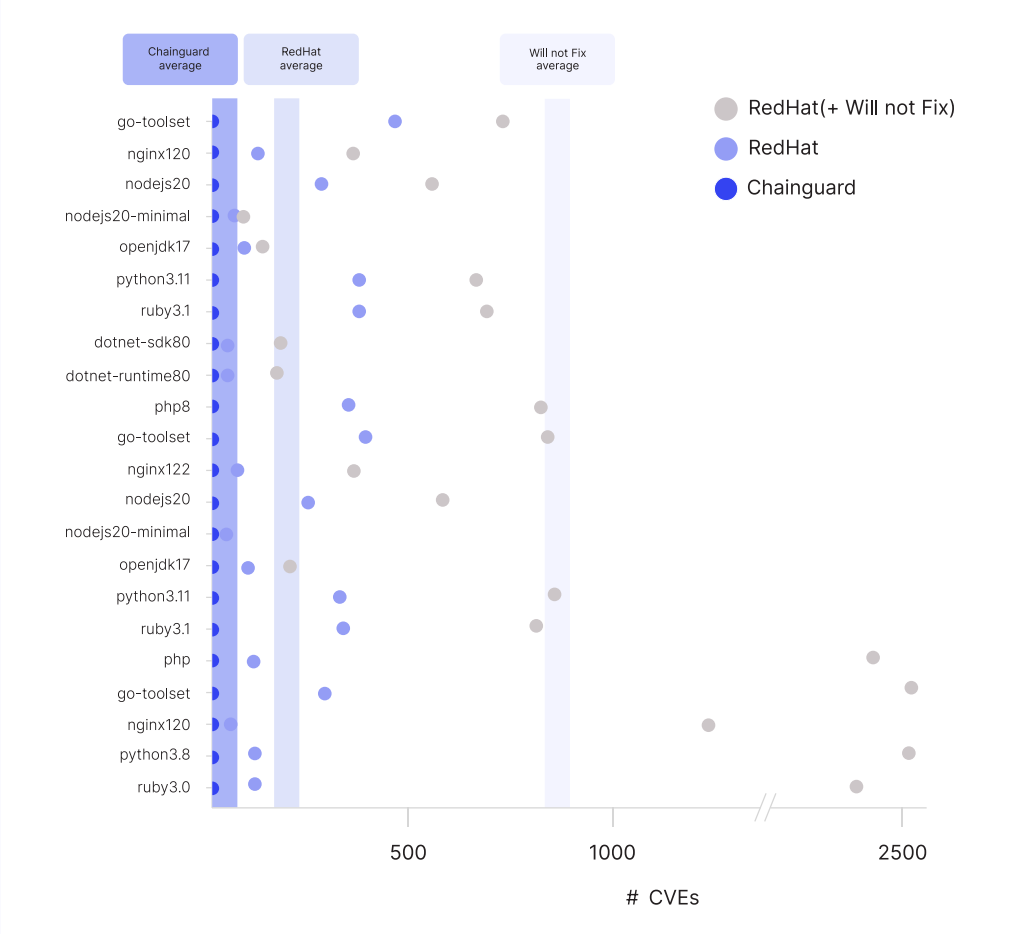
Notes: Measured on Jan 18, 2024. “+WNF” indicates the count included CVEs labeled “will not fix”.

Red Hat's UBI images have more total CVEs than Chainguard equivalents, no matter whether the analysis includes or excludes "will not fix" CVEs. Chainguard Image equivalents of these UBI images also have, on average, fewer components and are smaller.

Figure 5 presents a related analysis: the total number of CVEs for Red Hat application images versus equivalent Chainguard Images. The figure includes the total count for Red Hat application images excluding "will not fix" CVEs (in dark purple) and including "will not fix" CVEs (the light purple).

FIGURE 5.

Count of Total CVEs for Red Hat Application Images versus Equivalent Chainguard Images



Excluding "will not fix" CVEs, Red Hat application images contain, on average, 190 CVEs. Including "will not fix" CVEs, Red Hat application images contain, on average, 872 CVEs. Equivalent Chainguard Images contained, on average, zero CVEs.

The Search for Secure Images: The Air Force's Iron Bank Repository

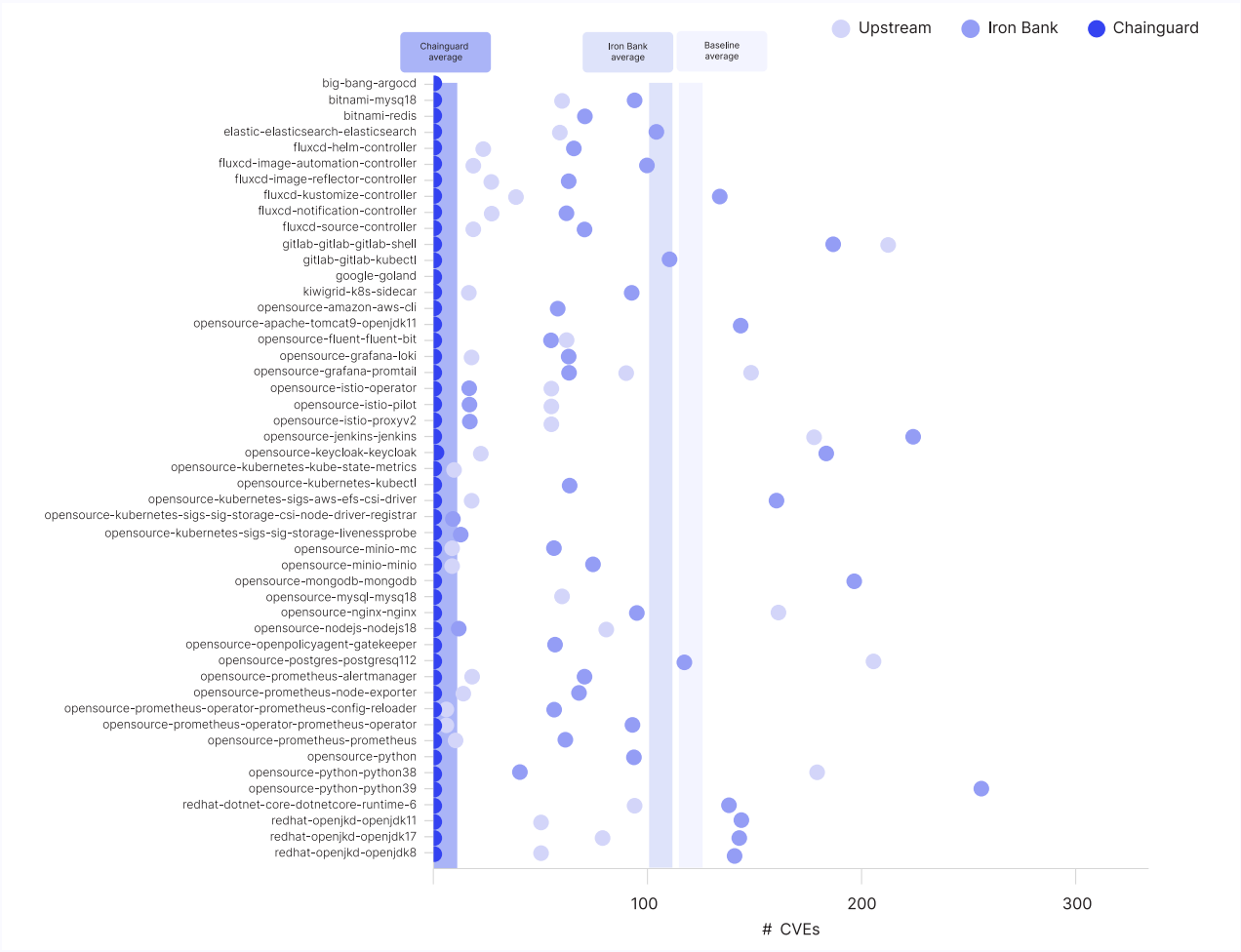
Platform teams that are seeking "hardened" or "secure" images sometimes turn to Iron Bank images because Iron Bank advertises itself as a "hardened" container image repository. It is maintained by the U.S. Air Force and often used by military units and military contractors seeking secure images.

To conduct an analysis, we created a list of the 100 most downloaded ("pulled" in container-speak) Iron Bank images, filtering out those that were end-of-life, and then determined equivalent upstream images and Chainguard Images. We then scanned these different images, using the "latest" tag or other similar tag as appropriate.

Figure 6 lists the total CVE counts for these images, including the Iron Bank version, the upstream equivalent, and the Chainguard Images equivalent.

FIGURE 6.

Number of CVEs for Popular Iron Bank Images, Upstream Equivalents, and Equivalent Chainguard Versions



Iron Bank Images have, on average, nearly 110 CVEs. Chainguard Images typically have zero CVEs. A further sub-analysis revealed that Iron Bank images had, on average, 8 high or critical severity CVEs.

We also performed an analysis on the images that comprise "Big Bang," the military's "software factory in a box." These images are a popular offering with Iron Bank. Big Bang images are meant to be deployed together and enable military teams and military contractors to, theoretically, build secure-by-default software. Table 3 presents an analysis of the total CVE count and the total number of critical and high severity CVEs for the Iron Bank version of Big Bang and equivalent Chainguard Images.

TABLE 3.

Number of CVEs for the Iron Bank Version of Big Bang and the Chainguard Version of Big Bang

Big Bang Version	Total CVE Count	Count of Critical and High Severity CVEs
Iron Bank Version	1,372	31
Chainguard Images Version	16	1

**To ensure a fair comparison, this analysis includes only 24 of 32 images, since Chainguard's image catalog does not yet include all Big Bang images. All Grype scans were of the latest version, The scans were performed on March 27, 2024.*

Big Bang Images, as currently provided by Iron Bank, have over 1,000 CVEs in total. In other words, the military's "software factory in a box" has over 1,000 known security vulnerabilities and 31 of these are critical or high severity CVEs. The equivalent Chainguard Images have only 16 CVEs and only 1 CVE with a severity of critical or high.

The Search for Secure Images: Canonical's Chiselled Images

In November 2023, Canonical announced the release of chiselled images, a new take on distroless images. These images are intended to be "secure by design," featuring an "ultra-small image size [that] greatly reduces the attack surface." Chiselled images are built with Canonical's tool, chisel. Chisel allows users to define their image contents at the sub-subpackage level.

Presently, Microsoft has made the switch to chiselled images for .NET containers found on the Microsoft Container Registry (MCR). Canonical also maintains chiselled .NET images on Docker Hub. It appears that Canonical's current goals do not include maintaining a large repository of base images for various frameworks and languages.

This analysis therefore analyzed two available chiselled images: .NET and ASP.NET for all available versions. The analysis examined the total number of CVEs, the number of high and critical severity CVEs, the number of components, and the size in MB. Equivalent Chainguard Images were also analyzed.

Findings for the .NET runtime images are in Table 4.

TABLE 4.

Comparison of .NET Runtime 6, 7, and 8 Images

	Total CVEs			Crit/Hi CVEs			Components			Size (MB)		
Version	6	7	8	6	7	8	6	7	8	6	7	8
Chainguard	0	0	0	0	0	0	200	204	200	133	136	133
MCR (Chiselled)	7	7	7	0	0	0	171	175	175	84	86	85
Docker (Chiselled)	0	0	0	0	0	0	164	168	168	118	85	122

Notes: Chainguard measurements were taken on 2/13/24. All other measurements were taken on 2/8/24.

The CVE count for these images are zero or near zero. Taking an average across the .NET versions, equivalent Chainguard Images have, on average, 15 percent more components and are 28 percent larger.

Findings for the ASP.NET images are in Table 5.

TABLE 5.

Comparison of ASP.NET 6, 7, and 8 Images

	Total CVEs			Crit/Hi CVEs			Components			Size (MB)		
Version	6	7	8	6	7	8	6	7	8	6	7	8
Chainguard	0	0	0	0	0	0	332	340	341	154	158	157
MCR (Chiselled)	7	7	7	0	0	0	301	309	314	104	108	109
Docker (Chiselled)	0	0	0	0	0	0	294	302	307	138	107	147

Notes: Chainguard measurements were taken on 2/13/24. All other measurements were taken on 2/8/24.

The CVE count for these ASP.NET images are, again, zero or near zero. Taking an average across the .NET versions, the Chainguard ASP.NET images have 33 percent more components and are 43 percent larger.

The major disadvantages of using chiselled images cannot be found in the tables above. Most importantly, there are, at the time of writing, only a few chiselled images available as pre-built containers, limiting the ability of an organization to quickly adopt chiselled images, and, relatedly, there are relatively few “chiselled” Ubuntu packages. Second, directly using the chisel technology requires power users and a potentially significant investment of time and expertise. Most importantly, “chiselling” a package requires expert knowledge: a user must be able to logically divide a package into “slices,” which requires intimate knowledge of that package and its functionality.

But Do CVEs Matter? Taking the Skeptics Seriously

Skeptics will rightly point out that reducing security to a count of CVEs is an over-simplification. Software and container images can still be insecure even if there are zero CVEs. This section therefore tries to address common concerns about the use of CVEs as a measurement of security.

First, some critics point out that only a small fraction of CVEs are known to be exploited. Analysis published by Red Hat and others make this point. Red Hat's 2022 analysis of 1,656 CVEs associated with its products finds that only 7 (or .4 percent) were known to be exploited "in the wild." This determination that a vulnerability was exploited in the wild is derived from the U.S. Department of Homeland Security's Cybersecurity and Infrastructure Security Administration's known exploited vulnerabilities (KEV) catalog.

Admittedly, this is a small fraction and does suggest that many vulnerabilities are likely never exploited. But there are a couple of unacknowledged wrinkles. First, the KEV represents a lower bound, those CVEs that are known to be exploited. And if you think that most cybersecurity compromises go undetected and that many detected compromises go unreported, then the KEV list is a potentially dramatic under-estimate of the number of CVEs that are actually exploited in the wild. In other words, the actual rate of exploitation in the wild of CVEs could be much higher. Second, even if only a small fraction of CVEs are known to be exploited in the wild, it can still be worthwhile to prioritize remediating CVEs. In addition to the need for compliance, some fraction of CVEs are exploitable and so merit remediation.

> **And if it's difficult to predict which vulnerabilities will be exploited, why not remediate CVEs if there is a straightforward option?**

Second, some critics will rightly point out that there are a range of security threats that CVEs do not capture. It's true! This includes, for instance, whether an open source package or a container has been compromised by malicious actors and whether an open source project has had no recent activity or signs of continuous quality improvement. The recent XZ Utils backdoor is, for instance, only the latest example of the threat of malicious open source software.

While open source package malware is a threat (it's even one we write about at Chainguard and have published about in academic conferences), the harm to users of open source malware is easy to exaggerate. Parties focused on open source software package malware often emphasize the number of open source packages flagged as malware. Sonatype's research often emphasizes this aspect of open source malware. The headlines are scary: "55,000 newly published packages [are] malicious."

Unmentioned is that many of these packages, though admittedly not all, have few downloads and that many of these downloads are potentially mirrors. Nonetheless, future research should examine the prevalence of malicious open source software across different container ecosystems.

Additionally, repository activity and other forms of project "health" are indeed useful signals, though whether these signals are useful for security is not entirely clear. Research ought to analyze this claim closely, pinning down "health" and providing a statistical measure of its usefulness for predicting "security."

So does open source package malware matter? Yes. Is it possible to exaggerate the usefulness of CVEs? Yes. Are some high percentage of CVEs not known to be exploited in the wild? Yes, with caveats. Should organizations ignore CVEs in their cloud infrastructure? Probably not. But skeptics should contact us and we'd be glad to work on an "adversarial collaboration."

The Search for Secure Images: Chainguard Images

Where does the search lead? For us, and we're biased, the search leads to Chainguard Images. The evidence suggests to us that if Chainguard Images didn't exist, then we would have to invent it.

Existing containers are far from "hardened." In fact, the analysis presented earlier suggests that most popular containers have hundreds of CVEs. To put it clearly, the container, the crucial building block of the "cloud" and the unit upon which many other elements of digital infrastructure are built, is a security dumpster fire. And though our analysis did not explicitly focus on this aspect, this problem is rooted in the distribution model of open source software included in containers. The packages in containers are often out-of-date and unpatched. **To reduce CVEs in containers requires reducing CVEs in the underlying linux distribution, in the "packages" themselves.**

That's why Chainguard Images are derived from a Linux distribution, Wolfi, devoted to both rapid updates and CVE patches. This is why Chainguard Images, no matter the exact comparison, generally reduce CVEs by 97.6 percent, reducing the attack surface and liberating defenders from the toil of constant CVE patching. Check out our Chainguard Images Directory and take Chainguard Images for a test drive today.

Thank you to Kaylin Trychon, Will Dolinsky, Sarah O'Rourke, Crystal Poenisch, Jordi Mon Companys, Jed Salazar, Dan Luhring, Mark McCormick, Josh Wolf, and numerous other Chainguardians that made this report possible.

Glossary

Container Image: A container image is a static, immutable filesystem bundle that serves as a blueprint for building containers. *From Chainguard Academy*

CVE: A CVE entry represents a known weakness in a software product and contains information to help address any potential risks to the integrity of a system caused by the vulnerability. Each CVE is assigned a unique CVE ID and is recorded with a description of the vulnerability, a list of affected software releases, any relevant references, and other pertinent information. *From Chainguard Academy*

"Hardened" Container Image: There is active debate on what exactly constitutes a "hardened" container image. This whitepaper employs the definition that emphasizes few or zero CVEs. A broader definition could include the inclusion of secure-by-design features in containers: the use of digital signatures and the inclusion of SBOMs, for instance.

Software vulnerability: A software vulnerability is a weakness in a program which, if left unaddressed, may be used by attackers to access, manipulate, or compromise a computer system.
From Chainguard Academy