SECURING CONTAINERS AND KUBERNETES AT SCALE WITH POLYMORPHING

Stop attacks before they start

TABLE OF CONTENTS

POLYVERSE

Why Read this Whitepaper?

Containers have revolutionized application development by making it far easier for engineers and product teams to quickly develop software that's more reliable and easier to scale, but these benefits have always come with a few worrying realities. It can be hard to know what's inside a publicly available image and, though they're handy, containers just aren't patched and maintained at the same pace as VMs and bare-metal servers.

This whitepaper takes a closer look at the challenges of securing containers, particularly in large-scale Kubernetes use cases, where pod replication is easy and presents an inviting opportunity for hackers looking to use common exploits to compromise hundreds or thousands of instances. It explains the need for proactive security solutions that address the unique nature of containers, the build process, and an innovative approach to securing them that can enable you to take full advantage of the power of Kubernetes.

Introduction

Building and deploying containerized applications, either in small clusters or at Kubernetes scale, has always brought with it the challenges of vetting Docker images and upstream packages and of setting security policies that lock down cross-client access. Unfortunately, managing security this way means a never-ending battle to stay ahead of hackers and may prevent you from taking full advantage of the inherent power and flexibility of Kubernetes to make your environment as secure as possible.

In a typical Kubernetes deployment, with many identical pods running similar work for different clients on a single instance, a vulnerability opened on just one pod can allow malware to spread to hundreds or thousands of container replicas. If the pod is based on an unpatched version of Linux or contains an outdated version of SSL, every pod created with that image is vulnerable.

In this scenario, regardless of whether your cluster is on-prem or in the cloud, hackers are looking to take advantage of known exploits common to publicly available stock Linux distributions and packages. All they need is to find one vulnerability to wreak havoc across your cluster.

Once inside a single database pod, for example, hackers can gather data from multiple applications and client accounts, even if those accounts have no business connection. Unfortunately, you've linked their fates by building them with the same vulnerable database image.

In a VM paradigm, you might create physical and network barriers to halt that kind of spread. But in Kubernetes, with virtual networks and shared resources, isolation comes in the form of security policies and layers that restrict client applications from escaping their containers. Still, Kubernetes pods remain fundamentally identical. Hackers love to exploit this sort of environment because it gives them maximum benefit with minimum effort.

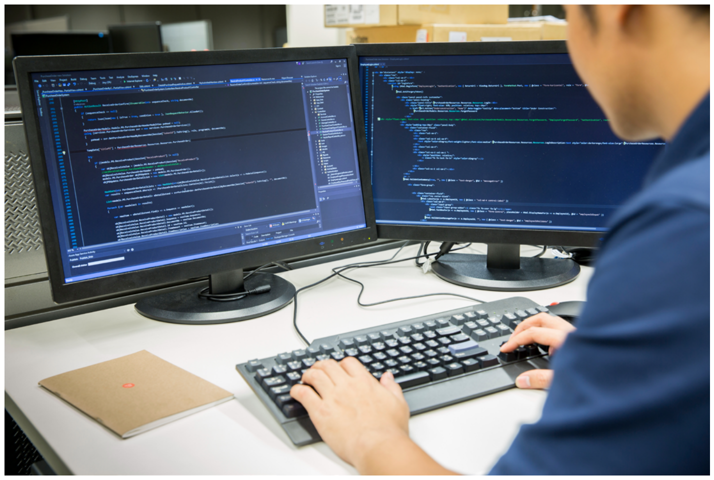
To thwart hackers, cluster maintainers need a way to deploy pods based on truly unique images, but the very idea seems to undermine the one-to-many Kubernetes paradigm, and the work involved in doing this manually is enough to make any cluster maintainer shudder. Manually creating multiple images would surely require retooling your entire CI/CD pipeline and trigger a massive re-think—and still wouldn't block hackers for long.

However, there is a way to improve your cybersecurity and secure containers without adding work or breaking your existing workflows—all while allowing you to leverage the full power of Kubernetes.

Polymorphism is the key.

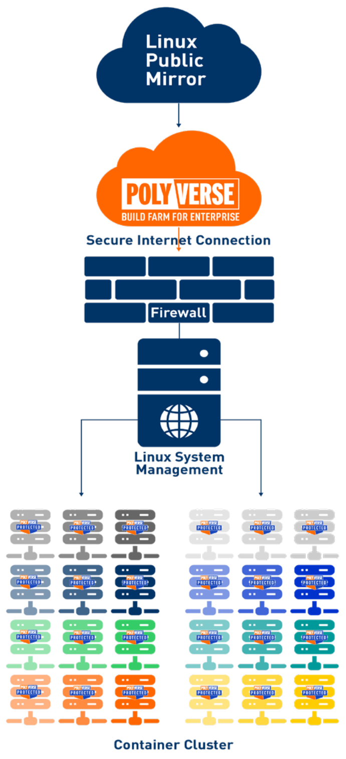
Polymorphing™ Makes Each Pod Image Unique

Polymorphism works by randomly scrambling and recompiling the base Linux, packages and binaries you already know and use, changing file locations, memory addresses, network parameters and every aspect of an image or .iso without changing how it does its work. Everything performs as it should, but nothing about the running system looks familiar to a hacker seeking to exploit a stock Linux OS or package vulnerability.



Polymorphism also provides a ready way to defend against memory-based attacks, CWEs and CVEs that take aim at specific function locations, CPU registers and other targets because most of these attacks become unviable. Your systems don't look like anything a hacker is expecting.

Polyverse® Polymorphing is a technology that uses polymorphism to make your images unique and integrate into your existing CI/CD pipeline so you can build and distribute ever-changing images for your Kubernetes clusters. Patched or unpatched, these scrambled



images can prevent 70 percent of the most common exploits and nearly all zero-day vulnerabilities. This is particularly good for containers, which don't get patched nearly as often as VMs and bare-metal machines.

Polymorphing works by providing you with custom images that you can redefine as needed and store in your container registries. These images provide everything you need without adding extra container layers that can slow down or bloat your services. You can use publicly available packages or your own with the same result, and Polymorphic Build Farm™ for Open source from Polyverse adds the ability to distribute your images at a scale that has no practical limit. Since the scrambling process is built into your CI/CD pipeline, it automatically creates new, differently scrambled images each time you deploy or as often as your use-case requires.

Image Diversity Forces Hackers to Chase You—Not the Other Way Around

Polyverse's suite of tools runs on top of Kubernetes and provides a higher-level abstraction layer for dynamic and intelligent goal-based management of pods. It fundamentally changes the traditional deployment strategy, which first launches containers and then exposes them through a routing infrastructure. Polymorphing operates the other way around.

For every deployment request, Polymorphing queries an application definition file that you define, grabs your polymorphed images, and rapidly launches the necessary pods. It creates volumes, tight network policies and other resources as needed before your pods start and then allows you to monitor their health, age and consistency.

All this works with standard private Docker registries, and can be configured to simultaneously produce container, VM or PXE boot images using your own builders or the Polyverse build service.

This Moving Target Defense (MTD) introduces constant, proactive change that can be applied to containers, physical machines, binary instructions, ports, networks, keys, passwords and nearly all aspects of your Kubernetes images. The more things you move and change, the more costly it is for attackers to exploit the target. The more costly the effort, the more likely they are to give up and move on.

Despite all this pod diversity, images compiled with Polymorphing and delivered with Polymorphic Build Farm remain easy to monitor and maintain. Together, they allow you to build in trust by verifying your code end-to-end, test your configurations in real time, and deploy hundreds or thousands of pods quickly and reliably.

Planned Entropy Can Make Kubernetes Even More Secure

Using a polymorphic approach not only makes your cluster less enticing to hackers, it can enable you to further protect your applications and clients by allowing you to take full advantage of the power of Kubernetes to manage thousands of pods in different ways.

In the scenario above of a vulnerable database image, Polymorphing lowered risk by scrambling the code. Still, all replicas remain alike. What if you want to limit a particularly crafty hacker's ability to move laterally across your cluster?

With Polymorphing and Kubernetes together, you can readily build and distribute images to your cluster based on rules. In this way, you can create pods that are unique to each user, not just one homogeneous pool. The persistent database hacker, finding that the pods running the same services are actually different, will immediately be stymied and then forced to rework the exploit all over again, and again, every time you redeploy.

Polymorphing is Built On and For Containers and Much More

Polymorphing was designed on and built to run on containers. This experience—along with knowledge about layers, block-level architectures, system requirements, and how to keep them high-performing—is at the heart of Polyverse cybersecurity solutions.

Polyverse also understands threats to the entire DevSecOps stack and built Polymorphing to address vulnerabilities that can compromise everything from the infrastructure layer to the application layer. Now you can leverage the power of polymorphism to better secure all your infrastructure, including hardware, HPC clusters, edge devices, containers and Kubernetes clusters.

For More Information

Contact us at:

sales@polyverse.com

sales-emea@polyverse.com

sales-apac@polyverse.com

Or visit our website:

https://polyverse.com