Container Security 101

Understanding the Basics of Securing Containers

By now, it's apparent to cybersecurity teams everywhere that the proverbial container genie is out of the bottle. Developers have widely embraced containers because they make building and deploying so-called cloud native applications simpler than ever. Not only do containers eliminate much of the friction typically associated with moving application code from testing through to production, but application code packaged up as containers can also run anywhere. All the dependencies associated with any application are included within the containerized application. That makes a containerized application highly portable across virtual machines or bare metal servers running in a local data center or on a public cloud.

That level of flexibility enables developers to make huge gains in productivity that are too great to ignore. However, as is the case with the rise of any new IT architecture, cloud native applications still need to be secured. Container environments bring with them a range of cybersecurity issues involving images, containers, hosts, runtimes, registries, and orchestration platforms, which all need to be secured.

The challenge organizations will face is first understanding how the many layers of a cloud native computing environment interact with one another, and then finding the right tools to build a repeatable set of processes to secure each layer. Cybersecurity issues specific to containers include:

Images: Vulnerabilities can impact container images just like any other piece of code. Building a bill of materials, identifying any embedded secrets, classifying all the layers of an image, are all still fundamental cybersecurity tasks that still need to be addressed. Where things get complex is in the sheer number of containers running in an application environment and how frequently those containers are updated. Thanks to the rise of DevOps practices, it's not uncommon for organizations to now update containerized applications multiple times a week. Each update to what can quickly become thousands of containers running in an IT environment represents an opportunity for vulnerabilities to be introduced in that environment.

Container registries: A container registry provides a convenient, centralized source for storing and distributing application images. Today's organizations can easily have tens of thousands of images stored in their registries. Because the registry is central to the way a containerized environment operates, it's essential to secure it. A registry is critical for bringing order to potential container chaos, but it also can provide a path through which cybercriminals can easily compromise the entire environment. Continuously monitoring registries for any change in vulnerability status is a core security requirement that needs to include locking down the server that hosts the registry.

Container runtimes: The container runtime is one of the most difficult parts of a container stack to secure because traditional security tools were not designed to monitor running containers. Legacy tools typically can't see inside containers, much less establish a baseline for what a secure container environment looks like. Container runtime security issues require cybersecurity teams to focus on application security concerns not addressed by legacy firewalls.

Container orchestration: Access control to container orchestration platforms such as Kubernetes® to prevent risks from over-privileged accounts, attacks over the network, and unwanted lateral movement, need to be addressed using allow list techniques in much the same way access to legacy IT environments is handled. Where things become different within a container orchestration platform is in the need to also secure communications between pods on a Kubernetes cluster shared by multiple applications.

Host operating systems: The OS that hosts your container environment is perhaps the most important and often overlooked aspect of securing a container environment. Any compromise to the host environment provides cybercriminals with access to the entire application environment. Each host needs to have its own set of security access controls in place as well as be continuously monitored for any new vulnerabilities that might have been discovered since that host was deployed.