Google Cloud

Security in containers

So far, you've learned that both virtual machines (VMs) and containers are used for virtualization. Containers are standalone packages of software that include everything you need to run the application contained in them. VMs are complete virtual computers that include the core program of the operating system, the kernel. Containers are dependent on shared hardware and kernels from a host, while VMs include the hardware layer. In this reading, you'll learn more about keeping containers secure.

Differences between virtual machines (VMs) and containers

Containers and VMs have some differences that are important to consider when preparing security measures. Since VMs contain every part of the infrastructure, including the kernel and the complete operating system, they can isolate whole machines. Containers share the operating system from their hosts. You can isolate the application processes in them from the rest of the system they share.

VMs and containers serve specific purposes and are useful in specific circumstances. For example, VMs are suitable for hosting traditional enterprise workloads and applications that have dependencies on the operating system. Containers are lightweight and easier to move across environments. So, containers are good for building and deploying cloud-native applications across clouds, including public, private, and hybrid clouds.

Containers and VMs are not mutually exclusive, so you'll often use them together. For example, you can use a container to run an application while a virtual machine gives you the underlying infrastructure.

The need for container security

Imagine there's a vulnerability inside a container, along with some exposed credentials, and other misconfigurations. This can compromise your entire infrastructure in the cloud. An attacker can use this set of wrong configurations and exploits to run malware like crypto mining applications right in your cloud.

Remember, containers are standalone packages you can use for self-contained applications. They can run processes in an isolated environment. So, they use kernel namespaces, but remove layers like the CPU, hardware virtualization, and a kernel.

Since containers share these layers with their hosts, exploits can jump in and out of the containers.

Pro tip: Containers can be hosted by VMs, and if the host is not securely configured, it can compromise the security of the containers on the host. Ensure the host is completely secure before keeping any containers. To help you do this, use a container-optimized OS.

Note: Containers themselves are not security boundaries. They provide restrictions on access to resources on their shared host, but they may not prevent an attacker from going around those restrictions. A VM is the boundary for the application, including resource allocation and resources.

Container security is critical for your cloud's overall security. A few best practices you can use to secure your containers include:

- Shift-left strategy
- Vulnerability scanning
- Runtime configuration and mitigating vulnerabilities, including RBAC for containers

Shift-left strategy

Container security practices include practices for applications and container images along with the host, the container's runtime, the technology of the clusters, and the cloud provider configuration.

A shift-left strategy is a way to embed security into your development process. It can also help you think about security from the first steps of system design or application. So in container security, it means thinking about security in all parts of container development. Remember, DevSecOps is development, security, and operations. So when development is happening, security is a part of it.

Vulnerability scanning

Before you build and deploy your application, you can scan the code and look for possible vulnerabilities in your code. To do this, you can use host scanning, Infrastructure as code (IaC), and application scanning.

Host scanning

Scanning the host will help you find vulnerabilities in the kernel, libraries, and any container runtime that's in the host. If any components in the host are misconfigured or vulnerable, they could be entry points for your running containers.

Infrastructure as code (IaC) scanning

Remember, IaC is software defined infrastructure. Using IaC tools like Terraform can help leverage cloud resource management. Use IaC scanning tools to validate your configuration before you start building your infrastructure. You need to have your automation ready to scan

for vulnerabilities, but to successfully scan for other vulnerabilities, it needs to be free from vulnerabilities itself.

Application scanning

Be sure to scan all third party dependencies like libraries and frameworks. And, be sure to analyze your container images with image scanning. Also, examine the container for outdated assets, and always scan everything before deploying the container.

Runtime configuration and mitigating vulnerabilities

You can use runtime configuration to mitigate vulnerabilities in these ways:

- Effective user: Never run a container as root. If your container tool has the user namespace feature, use it.
    - User namespaces is a Linux-based feature that allows you to map container users to different users in the host.
- Restrict privileges: Some container platforms like Kubernetes have ways to set up Role-based access control (RBAC).

RBAC for containers - Kubernetes

You may already be familiar with the idea of RBAC and how to use them to secure your containers. As with any other use of RBAC, be very careful who you grant access to, and restrict it to those who need it.

In the example of Kubernetes, the application programming interface (API) has four kinds of objects defined:

1. Role: Sets permissions within a namespace.
    a. Note: You need to decide and specify which namespace it belongs in.
2. ClusterRole: A non-namespaced resource
3. RoleBinding: Gives permissions defined in a role to a set of users or a user, and binds the role to within a set namespace
4. ClusterRoleBinding: Gives permissions defined in a role to the whole cluster
    a. Note: You can use a ClusterRoleBinding to bind a ClusterRole to all the namespaces in a cluster.

Key takeaways

Containers are different from VMs, but security is vital to both. Containers are tied to hosts, which can be VMs, so you need to verify security of the host before using it for containers. Security should be a consideration for containers starting at the very beginning of the coding for the containerized application. Also, scanning is necessary to keep containers secure before

they're deployed. Runtime configuration, including container RBAC, can help mitigate vulnerabilities.

Resources for more information

Check out this resource to learn more information:

- Click the link to learn more about container-optimized OS.