Sense of Security®

MAKING SENSE OF

Testing Container Security

As containers continue their march into the IT mainstream, like most new technologies, they hold great prospects for improvements in efficiency, scalability and security. However, poor implementation practice generally lets the organisation down, and container technology is no different.

As the design and implementation runs at break neck speed, not much has been done to ensure the fundamentals of security are in place including end-to-end integrity, hygiene, auditability, visibility and overall confidence in the compute platform.

Containers, by name, have the primary objective of keeping things (processes) contained. And while the innate nature of the container (smaller footprint, lightweight, discrete services etc.) should by default deliver a more isolation centric compute environment, the lack of attention to detail in the service delivery lifecycle (SDL) can result in an insecure implementation where the containers do not deliver isolation. This creates an opportunity for an attacker to exploit the weakness, traverse the environment and ultimately perform a high impact attack (think data breach and hacker persistence in the DevOps chain).

Vulnerability management, testing and validating container security is therefore critical to maintain confidence in the ongoing use of the technology.

The Divide Between Developers and Security Testers (Hackers)

The divide between developers and security testers (let's just say that unsolicited "testing" is executed by hackers) centres around one word – abstraction. By exploring where abstraction occurs we understand what it takes to implement and validate secure container-centric systems.

The modern web application

Containers are designed to virtualise a single application, hence the speed. You can run multiple containerised apps on a single common OS kernel. For containers, the entire "boot" process that a normal virtual machine goes through is essentially skipped and only the last steps where the root filesystem is loaded, and a shell is launched happens. More precisely, the container environment is "started rather than booted".

The modern web application, designed, deployed and operationally managed through an agile process is most likely going to be using cloud-native technology and intrinsically developed for the modularity, scalability and speed of the DevOps SDL. As such, we expect it to be designed for a microservices architecture, heavily leveraging containers.

While containerisation focuses on abstraction of the app from the OS (generally a lower layer of abstraction), modern app development itself has tended towards higher levels of abstraction. Given the speed at which apps need to be delivered, developers leverage frameworks and development environments that offer a higher level of abstraction, making it easier to develop without coding to cater for the lower levels of the stack (the framework takes care of that).

We have developers racing to the top (of the stack) but to execute a high-impact hack, the attacker needs to traverse the stack to the lower layers where the implication is greater. So, along the lines of the mantra "hooking lowest wins" we have the attacker racing to the bottom of the stack.

Effectively Testing Containerised Deployments

To test and validate the security of an environment utilising containers, requires a discrete understanding of the language the app is written in, the framework is it deployed through and all the subsequent layers in the stack, including the container orchestration and the cloud platform it is operating in.

As a result, security testing for the modern microservices app is a discipline to itself, and essentially very different to testing your traditional monolithic app (even if it was hosted in the same cloud).

North-South, East-West

The goal with any form of cyber-attack is persistence.

Persistence in a dynamic environment requires a particularly savvy attacker. Remember that containers can be especially ephemeral (lifetimes often measured in seconds, minutes and hours rather than days, months or years) so the very nature of their low time-to-live could kill the persistence that an attacker is looking to achieve.

The initial attack vector is most likely going to be a pinhole entry. We call this the North-South Attack. That is, the attacker finds a vulnerability (say in a web app) and manages to exploit it. Your traditional SQL injection is an example. But to further the attack, the attacker needs to remain inside the environment and needs to be persistent. This requires the attacker to step outside the application layer attack and move horizontally in the environment – we call this the East-West attack pivot.

The East-West attack is made all the more complicated by the isolation that the container delivers. Bravo containers! So containers can deliver an inherently more secure environment. But not all containers are created and maintained equally and as such there remains (plenty) of opportunity for the more illustrious attacker to operate.

Kernel Level Exploit vs Living off the Land.

Escaping from the container is likely to be achieved one of two ways: Either the container deployment is vulnerable to some form of kernel level issue or the attacker is going to have to make use of what is in the container to exploit it. Kernel level vulnerabilities don't occur that often. When they do they are quite devastating because all deployments are susceptible. Insecure container implementations are far more prevalent with issues ranging from vulnerable third-party code, inflated container footprints, unhardened environments, poor configuration and wider than necessary communications between system components at the software defined networking layer.

Penetration Testing for the Vulnerability Management Program.

To gain confidence about the security of your deployment you need the finesse of a surgeon to peel away each layer of the system under review.

Such finesse can only come through expert knowledge of all the layers in the stack and the ability to transform from an application centric test to an infrastructure centric test. Deep knowledge of the cloud platform that you operate on is also integral to maintain persistence and move horizontally through the environment.

Most organisations have environments hosting monolithic apps, traditional data centres and public clouds hosting dynamic microservices architectures. This means that an effective penetration testing regime needs to accommodate for the variation and as a result you need to expect more from your testers!

Sense of Security®

To discuss how our security solutions can help protect your most vital assets,
please call us on 1300 922 923 or +61 (2) 9290 4444.
info@senseofsecurity.com.au www.senseofsecurity.com.au