The Container Ecosystem
Why Full Stack Container Security Makes Sense
Common Building Blocks Make up Containers

Containers make tooling and environments consistent by providing common building blocks reusable in any development stage. These building blocks are known as images, and contain the functionality of an OS but depend upon the host for all system calls. For tools, containers provide a disposable, reusable unit that modularizes the delivery pipeline. For environments, they extend the write once-deploy anywhere abstraction to infrastructure. - Codenvy

Enter security... Container security extends into all aspects of the building blocks that make up the container ecosystem, and not just to the well-known registries like Docker or those offered within the cloud service providers. Securing a container deployment may include best practices for companies supporting: the developer workspace, continuous integration, build automation, testing frameworks, release automation, and operations tools. And, container security is critical, now more than ever, as greater numbers are deployed in production environments.

Respondents use of containerization in a production capacity

56% already in production 40% planning to test in the next 12 months

ESG, 2018
How Containers are Evolving and Security

The idea of virtualized containers was introduced in 1979 with Unix V7, but it wasn’t until 2013, when Docker was debuted to the public at PyCon, an annual convention for the discussion and promotion of the Python programming language, that the world of containers was solidified. Since 2013 container adoption has continued to grow. Today it’s reported that approximately 35+ percent of mid- to large-sized organizations are using containers, and 40 percent of those are using orchestrators— the most popular being Kubernetes. It’s reported that there are 7 containers per Docker host, and 78% of containers last less than 1 hour (versus 23 days for a virtual machine). What all this means, in security terms, is that if you have errors in your configuring your containers, then your containers may quickly cause irremediable harm.

Over thirty-five percent of mid-to large-size organizations are using containers, and forty percent of those are using orchestrators.

Source: Datadog
Docker Security Concerns

Those new to the container ecosystem sometimes make simple mistakes. For example, running production containers as root opens the door for an attacker to exploit a vulnerability and become root on the host, which can have devastating consequences. There have been many articles written about Docker security concerns, with one of the best by O’Reilly, “5 Security Concerns When Using Docker”. In addition, there is also a good piece in The New Stack, “Assessing the Current State of Container Security”, that provides additional background.

The application container market size is estimated to reach USD 4.98 billion by 2023. However, security concerns and skill gaps in the market might hinder this growth.

Research and Market’s

Docker security solutions must keep track of what VMs or bare-metal servers have what workloads, when they start and stop, and then apply frameworks appropriate. And it is not just a one-time analysis. The system must continually scan all images running in production, and also address the access control issues described above.

<table>
<thead>
<tr>
<th>CONTAINER SECURITY CONCERNS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>43% Unauthorized Container Access</td>
<td></td>
</tr>
<tr>
<td>39% Malware Spread Between Containers</td>
<td></td>
</tr>
<tr>
<td>36% Privacy Violations - Shared Resources</td>
<td></td>
</tr>
<tr>
<td>35% Vulnerabilities in Container Images</td>
<td></td>
</tr>
<tr>
<td>28% Lack of Trust in 3rd Party Container Images</td>
<td></td>
</tr>
<tr>
<td>25% Patching and Updating Containers</td>
<td></td>
</tr>
<tr>
<td>8% Lack of Compliance Certifications for Container Images</td>
<td></td>
</tr>
</tbody>
</table>

*Thales Data Security*
Docker Images

Given that developers may download public images, they must ensure that these are secure. Docker Hub may be one thing, especially when combined with Docker Security Scanning, but some unknown open-source site may expose the developer to risk, no different than when downloading laptop software from an unknown site. The Morning Paper’s article “A study of security vulnerabilities on Docker Hub”. provides a detailed analysis of the types of vulnerabilities found in both community and official images.

A more recent analysis by Federacy states that 24% of Docker images have significant vulnerabilities.

A good approach is to use CI/CD tools to properly embed security best practices across the container lifecycle. Doing this creates a baseline that reduces the need for additional efforts and reducing the chance that security will become a barrier. And, via this baseline, IT is able to detect threats in real-time with a lower false-positive rate. This also has an effect of moving security upstream, integrated earlier in the software delivery pipeline. In DevOps-speak this is known as a shift-left.

Based on what the system detects, active remediation may include additional logging, implementing additional isolation, or even deleting the container. This must all be automated and under control of the security management platform.
Container and Hybrid Cloud Challenges

Containers, if properly secured, will have a major beneficial impact on how organizations deploy in a hybrid cloud environment, both on-premise and across multiple CSPs. They provide improved portability and application integration, effectively abstracting the lower-layer cloud architecture. They allow users to separate and isolate a single application, creating a boundary at the app level rather than at the server level. Especially popular with developers, they allow for testing without affecting other applications, servers, or data. But with all the benefits come some challenges:

- Scaling containers while adhering to corporate policies and governance across a hybrid cloud
- The provisioning of large deployments is unfeasible without a container orchestration mechanism (e.g. Kubernetes).
- Scanning of container images for security vulnerabilities prior to deployment into production.
- Security keeping pace with the rapid number and locations of container deployments.
- Monitoring throughout the container lifecycle, regardless of location
- The potential for container sprawl creates loose access controls between containers that could leave production environments vulnerable (ESG, 2018)
- The host-level equivalent of a hypervisor is a standard operating system which appears to be more susceptible to compromise (ESG, 2018)

“IT cannot simply keep using the same machines [that] the earlier version of a container got deployed into. This is because containers for cloud-native applications, in particular, are meant to be moved around depending on the load in those containers.”

Srikanth Natarajan
Micro Focus
Full Stack Container Security

At Cavirin, borrowing a term from software development, we call our approach ‘full-stack’ container security. This implies that you need to look at all layers of the container environment (cloud, virtualization, operating system, container runtime, and orchestration), taking both a vertical and horizontal security approach. However, containers can’t be secured in isolation. The overall security posture of the hybrid cloud is critical, and this includes both the workloads as well as the cloud account and services. Specific capabilities required when protecting a hybrid cloud include:

1. Ensuring the integrity and security of images hosted on public and private registries. This includes image ‘hardening’ against best practices as well as patches and vulnerabilities.

2. Securing the container hosts via the CIS Docker benchmark (i.e., ‘Allow Docker to make changes to iptables’), NIST, CIS, HIPAA, PCI, ISO, SOC2, and GDPR. Container hosts supported include Linux operating systems including specialized distributions such as CoreOS Container Linux.

3. Monitoring container runtime activity for actions out of the ordinary (i.e., starting new services or adding a new privileged user).

4. Securing container orchestration via the Kubernetes CIS benchmark.

5. Securing the security posture of the cloud account via monitoring and CIS benchmarks.

6. DevOps automation to integrate container security with the CI/CD pipeline.

Point 6 is an important stepping stone in enabling DevSecOps by automating security.
Full Stack Container Security (Jenkins Script) Example

The diagram below shows how DevSecOps is enabled by automating security. Promotion of the image is under control of a Jenkins Script:

1. The developer initiates a build
2. The script is now called as part of the build process
3. The assessment platform pulls the Docker image from the proper registry
4. Calls the proper benchmark/policy framework
5. Automatically triggers the assessment
6. The assessment platform generates a score for the image (note: in advance, the developer set a threshold for promotion, in this case, it was 75)
7. Within the script the score is compared against the threshold (acceptance criteria)
8. Script promotes or sends back for rework

This overall framework can be leveraged for other parts of the CI/CD pipeline such as determining whether an AWS AMI has met the necessary security baseline for activation.
About Cavirin Systems

Cavirin delivers cyberposture intelligence for the hybrid cloud by providing real-time risk & cybersecurity posture management, continuous compliance, and by integrating security into DevOps. The Cavirin platform combines automated discovery, infrastructure risk scoring, predictive analytics, full stack container security, and intelligent remediation to help organizations of all sizes leverage the cost savings and agility of the cloud without increasing operational risk or reducing their security posture. For more information, visit [www.cavirin.com](http://www.cavirin.com) or follow us at [www.twitter.com/cavirin](http://www.twitter.com/cavirin).