IDC ANALYZE THE FUTURE

IDC WORKBOOK

*Sponsored by: Sysdig*

**As organizations move to containers for next-generation infrastructure and applications, they must balance the need for security without negatively impacting the frequency of software deployments.**

# Definitive Guide for Evaluating Container and Kubernetes Security Tools

*December 2019*

**Written by:** Gary Chen, Research Director, Software Defined Compute, and Frank Dickson, Program Vice President, Cybersecurity Products

## Introduction

Containers are rapidly becoming the foundation for next-generation infrastructure and applications. Enterprises use Kubernetes (K8s), the de facto industry standard for container orchestration, to more rapidly deploy container-based software in order to support their digital transformation initiatives. However, containers and Kubernetes do not cause disruption just because they are new technologies. Kubernetes and containers are disruptive because they are often tied to culture, process, and organizational change. Operations and development teams are learning to work more closely with DevOps, and applications are architected with microservices for greater efficiency, scalability, and economies of parallel development.

### AT A GLANCE

#### WHAT'S IMPORTANT

Organizations are increasingly turning to containers and Kubernetes to improve the efficiency and scalability of software development efforts. Containers introduce new security issues, highlighting the need for container-specific security tools.

Given these changes, IDC data regarding container adopters shows that security is the number 1 challenge. As organizations move to automated software build pipelines with continuous integration and continuous delivery (CI/CD) and agile development methodologies, building security into this new approach without slowing down the speed and frequency of software deployments is one of the security challenges. Another challenge is that containers introduce a new environment with a new packaging format and a different management paradigm with Kubernetes. Without container- and Kubernetes-specific security tools, visibility into the container layer will be nonexistent, creating a security blind spot.

## Considerations for Container and Kubernetes Security

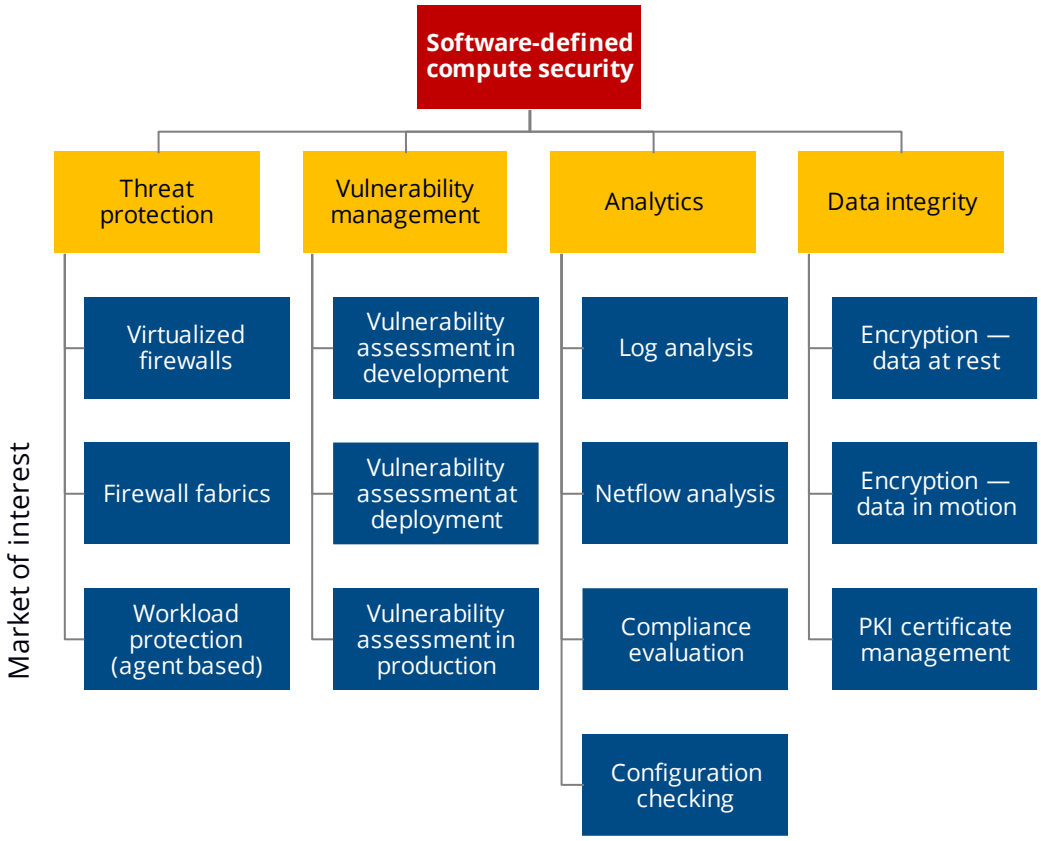
Just like with server virtualization, containers introduce a new layer, and all infrastructure disciplines must build integration with and insight into this layer in order to remain relevant. The network flows, input/output (I/O), and program execution happening at the container layer are invisible to existing virtual machine (VM) or bare metal tools. While containers don't invalidate the need for tools at that level, the use of containers does require the need for container-specific insights. Containers are also intimately tied to many other transformations, such as CI/CD, DevOps, and microservices. Some of the impacts these transformations have on security are as follows:

- Software is increasingly delivered through complex, automated CI/CD pipelines. While these pipelines simplify the process of software delivery, they can also be leveraged for security by inserting vulnerability scanning and tests into the process earlier rather than doing it all at once at the end. This is often referred to as "shifting left."

- While many organizations are already transitioning to DevOps, which brings developers and operations closer together, advanced organizations concerned about security are doing more. Secure DevOps or DevSecOps also introduce security into the process and organization to integrate security closely while still allowing fast and frequent software deployments.
- With microservices application architectures, applications are more distributed and componentized than ever. Security tools need to understand how to map containers to services, applications, and users in order to provide context. Because Kubernetes is the industry-standard platform for containers, security tools must be deeply integrated with it to provide proper context. Kubernetes has also built out many auditing and enforcing capabilities, such as admission controllers and security policies, that security tools should leverage. These Kubernetes-native controls do not impact application performance.

Although container security solutions have no need to protect the integrity of the underlying infrastructure (the VM layer or the bare metal layer), container security solutions clearly need to protect what runs on top of the core compute infrastructure. Container security is executed not by a single technology or offering but by an integrated set of offerings that span threat protection, vulnerability management, analytics, and data integrity (see Figure 1).

FIGURE 1: ***IDC's Cloud Security Framework, 2019***




*Source: IDC, 2019*

Vulnerability scanning and management are core features in container security solutions. Addressing vulnerabilities is key in any mature DevSecOps implementation looking to deploy hardened code. Scanning should also be implemented throughout the entire container delivery pipeline. Thus, vulnerability scanning does not happen once; rather, it is managed throughout the application life cycle during the following phases:

1. **Development.** As developers select various software components and libraries for their applications, all components must be scanned and vetted for known common vulnerabilities and exposures (CVEs). Hardened applications begin with a foundation of trusted components.
2. **Deployment.** As software is deployed, admission controllers leverage the latest image scanning results to determine whether the images are safe to deploy according to the configured policy. Vulnerabilities are discovered with the passage of time. What was vetted and trusted in the build phase may subsequently have a new known CVE at deployment. Trust and verify is the motto here.
3. **Production.** After deployment, a database of all container components should be maintained. This approach allows for the management of new vulnerabilities as they are discovered. Once a vulnerability has been identified, a workflow or an integration with a service management platform to remediate the issue should be facilitated automatically. The solution needs to be able to map a container image to an application/service and identify the owner. Thus, once a vulnerability in production is discovered, workflows can be immediately initiated to orchestrate prioritized remediation.

Scanning should encompass known vulnerabilities and be able to detect secrets. It should also extend beyond the core operating system (OS) base image and into any packages and frameworks that may be added.

Beyond scanning images, the container orchestration and management platform such as Kubernetes must be secured, along with the containers themselves, by three primary approaches:

- **Virtualized firewalls.** Virtualized firewall "appliances" are built with a specialized OS and provide network traffic filtering and monitoring for virtualized environments, including public cloud and private cloud (e.g., AWS, Azure, Google, IBM, KVM, and VMware).
- **Firewall fabrics.** Firewall fabrics implement a mesh of firewalls around VMs or containers, controlling access to the VM or container based on Internet Protocol (IP) and/or instruction. Firewall fabric solutions typically implement security from outside of the VM or container (not agent-centric protection) and often employ analytics to discover connections between the protected resources outside of the VM or container.
- **Workload protection.** Workload protection products provision security using or leveraging an endpoint agent or client as a core or fundamental component. The emphasis on threat protection and vulnerability scanning is provided only as part of a container solution. Even if both are perfectly executed, configuration issues can become an Achilles' heel. Additionally, compliance is a fact of life as violating relevant regulations can be punishing. Thus, guidelines such as CIS benchmarks provide best practices that security solutions can check against. At runtime, an additional layer of checks also needs to be implemented to protect against the things image scanning doesn't cover. These capabilities include:
    - Apply compliance checks at runtime with policies that map into compliance frameworks, including NIST, PCI, and MITRE.
    - Check configurations, such as enablement of defenses (SELinux, AppArmor, seccomp, etc.), or insecure configurations, such as exposure of daemons and ports. Configuration checks should happen early in the image life cycle, at the build or deployment stage, and be enforced through Kubernetes' Pod Security Policy (PSP).

Solutions should track compliance over time, not just at certain points. Many customers will also need auditing capabilities. In these situations, a separate compliance dashboard is ideal.

Audits and forensics are other challenges associated with containers. Containers are ephemeral, with short life spans due to autoscaling; in effect, containers are always being replaced by new versions. Solutions need to capture relevant state data for each container instance. Microservices architectures also increase the number of containers and elements that an application is composed of, leading to an exponential increase in the amount of data that has to be collected and analyzed. Systems capable of handling large amounts of data are required, and often artificial intelligence and machine learning technology needs to be used to help sift through it all.

To address the previously mentioned challenges, organizations require new instrumentation and security integrations to gain visibility into the container layer to help monitor and enforce. Obviously, any instrumentation should be nonintrusive and not impact performance while providing deep insights. Because Kubernetes is the de facto choice for container orchestration and management, solutions must deeply integrate with it. Kubernetes integration points to consider are as follows:

- How closely and fast does the solution track against the latest Kubernetes versions? Kubernetes is evolving very fast, with quarterly upstream releases.
- Kubernetes provides many security features that solutions should leverage, such as admission controllers, Pod Security Policies, and network policies. Sustainable solutions should leverage as much Kubernetes-native functionality as possible. Leveraging functionality will keep deployments closer to standard, which in turn enables more portability and reproducibility because solutions will not diverge from Kubernetes project efforts.
- Integration with Kubernetes is also crucial to mapping issues at the container level to something actionable. Kubernetes manages how individual containers map to pods, clusters, applications/services, and users. This mapping is crucial to identifying impacts as well as users to contact about issues. These mappings and other Kubernetes constructs, including labels, are also needed for context for monitoring, queries, and forensics.

Users also need to closely examine the tooling and user interface (UI) for building policies because they can be very complex. Prebuilt templates and automatic policy creation based on history or profiling can assist greatly with getting started, and users can customize from there. The other underappreciated element is policy testing, where users can effectively test new policies or changes in a sandboxed environment before rolling out to production.

Many changes are needed for effective security at the container layer. Containers introduce an entirely new layer of operations that requires insight. The rapidly changing methodologies of architecting, building, and operating modern containerized applications also alter the way infrastructure and applications behave, challenging many of the current assumptions that exist for traditional VM- and bare metal–based applications. To effectively secure this new environment, users will need to seek out new solutions that have specialized container functionality and Kubernetes integrations.

To select a solution that works best for your organization, consider the questions listed in Tables 1–6.

## *RFP Template Section*

### TABLE 1: *Critical Capabilities for Container and Kubernetes Security*

| | |
|---|---|
| **Does the solution have the ability to support the top on-premises Kubernetes software distributions and Kubernetes cloud services for hybrid cloud support?** | |
| **Is the security solution available as self-hosted software on-premises and in SaaS delivery models?** | |
| **Can the solution support any Docker-compatible registry?** | |
| **How quickly does the solution add compatibility with upstream Kubernetes features?** | |

*Source: IDC, 2019*

### TABLE 2: *Critical Capabilities for Container Scanning*

| | |
|---|---|
| **Can the solution integrate with CI/CD pipelines (both software and cloud based)?** | |
| **Does the solution detect insecure image configuration settings such as Dockerfile instructions, file contents, credentials, packages, and licenses?** | |
| **Can the solution scan vulnerabilities in third-party libraries/frameworks?** | |
| **Can the solution assign a specific policy based on registry/repo:tag information?** | |
| **Can the solution scan locally created images outside a registry?** | |
| **Does the solution provide reporting of vulnerabilities and packages affecting running containers based on cloud, Kubernetes, or container metadata?** | |
| **Does the solution map vulnerabilities back to application owners and notify them?** | |

*Source: IDC, 2019*

TABLE 3: ***Critical Capabilities for Runtime Security***

| | |
|---|---|
| **Is it possible for the instrumentation to avoid modifying the image or the running container or its libraries?** | |
| **Does the instrumentation have minimal impact on performance?** | |
| **Can policies be automatically generated based on history and profiling?** | |
| **Does the solution provide tools and capabilities to create custom policies as needed?** | |
| **Does the solution provide out-of-the-box libraries that can be leveraged to build policies?** | |
| **Can the solution leverage Kubernetes labels to apply policies to specific clusters, namespaces, or services?** | |
| **Can the solution audit and detect suspicious activity at the Kubernetes API level?** | |
| **Can the solution take response and remediation actions?** | |
| **Does the solution leverage Kubernetes-native controls such as Pod Security Policy (PSP) configurations or admission controllers as a Kubernetes enforcement mechanism?** | |
| **Can the solution observe and visualize container networking traffic flows?** | |

*Source: IDC, 2019*

## TABLE 4: *Critical Capabilities for Incident Response and Forensics*

| | |
|---|---|
| **Can the solution create a complete snapshot of container activity after the container is gone, such as executed commands, network connections, file system activity, performance indicators, and container events?** | |
| **Can the solution audit and correlate Kubernetes API activity with container and system activity to provide incident response capabilities?** | |
| **Does the solution integrate with notification systems and security information and event management (SIEM) systems?** | |
| **Does the solution provide tools or capabilities to help reduce noise and false positives?** | |

*Source: IDC, 2019*

## TABLE 5: *Critical Capabilities for Compliance and Audit*

| | |
|---|---|
| **Can the solution verify the security of the orchestration platform itself (Kubernetes) against the CIS Kubernetes benchmark, CIS Docker benchmark, etc.?** | |
| **Does the solution provide remediation content to fix violations?** | |
| **Does the solution provide scanning policies that map to compliance checks (e.g., NIST, PCI)?** | |
| **Can the solution detect compliance violations at runtime (e.g., NIST, PC)?** | |
| **Are metrics about your compliance posture stored to provide easy reporting and alerting on how your compliance posture changes over time?** | |
| **Does the solution allow custom compliance tests that can be run across different clusters, nodes, or containers?** | |

*Source: IDC, 2019*

TABLE 6: ***Critical Capabilities for Security Monitoring, Operational Maintenance and Health***

| | |
|---|---|
| **Does the solution monitor the operational status of the cluster to understand the service impact of a security incident?** | |
| **Does the solution identify pods consuming excessive resources and monitor capacity limits caused by cryptomining abuse, etc.?** | |
| **Can the solution monitor auto-scaling behavior to control unexpected billing caused by a distributed denial-of-service (DDoS) attack, etc.?** | |
| **Can the solution monitor all network connections and understand the process that initiated them to uncover suspicious activity within the system?** | |

*Source: IDC, 2019*

# About the Analysts

***Gary Chen,*** *Research Director, Software Defined Compute*

Gary Chen is IDC's Research Director, Software Defined Compute. His research focuses on server virtualization, container infrastructure and management, and cloud system software (system software used to build IaaS clouds such as OpenStack).

***Frank Dickson,*** *Program Vice President, Cybersecurity Products*

Frank Dickson is a Program Vice President within IDC's Cybersecurity Products research practice. In this role, he leads the team that delivers compelling research in the areas of Network Security; Endpoint Security; Cybersecurity Analytics, Intelligence, Response, and Orchestration (AIRO); Identity and Digital Trust; Legal, Risk and Compliance; Data Security; IoT Security; and Cloud Security.

## MESSAGE FROM THE SPONSOR

As Kubernetes becomes the de facto operating system in the cloud, cloud teams are taking ownership for security in addition to application performance and availability. This new secure DevOps workflow requires purpose-built tools to run Kubernetes and containers in production.

Confidently run cloud-native workloads in production using the Sysdig Secure DevOps Platform. With Sysdig, you can embed security, maximize availability and validate compliance. The Sysdig platform is open by design, with the scale, performance and usability enterprises demand.

Sysdig developed a method to show the business value of implementing a secure DevOps workflow with the Sysdig platform. Discover your personalized ROI today at: ***https://sysdigroi.com/***

IDC Custom Solutions

The content in this paper was adapted from existing IDC research published on www.idc.com.

**IDC Corporate USA**

5 Speen Street
Framingham, MA 01701, USA

T 508.872.8200

F 508.935.4015

Twitter @IDC

idc-insights-community.com

www.idc.com