SPLUNK FOR CONTAINERS

A single solution to unify insights across container environments and the entire technology stack

The market is shifting towards a container approach to consuming data, which has led to the adoption of Kubernetes, Docker and other container technologies. Gartner predicts that by 2020, more than 50% of global organizations will be running containerized applications in production, up from less than 20% in 2017. It's evident that container orchestration is the foundation for current and future applications. This tech brief will help you understand how to use Splunk Connect for Kubernetes, Splunk Connect for Docker & Splunk official docker images, to forward your data in a scalable and flexible manner.

What Is a Container?

A container is a lightweight software package that bundles application code along with the runtime, tools, system libraries and environment settings required to execute it. This enables an application to be abstracted from the environment in which it's executed and isolated from other applications. Containers are extremely portable and can run on a variety of operating systems, hypervisors and bare metal, at massive scale. Containers can also be spun up and down quickly, allowing for more flexibility.

A container ecosystem has many benefits, including:

- Low total cost of ownership (TCO): A well-designed containerized deployment leads to a smaller operating system, hypervisor and hardware footprint. Containerized applications provide better system utilization as multiple applications can be deployed on a single host.
- Faster time to value: Containerized deployments are much faster to deploy than hypervisor-based deployments. Applications can be deployed and upgraded quickly, enabling end users to get value out of their applications faster.
- Increased agility and flexibility: With containerized applications, enterprises can both expand their Splunk footprint and bring new teams onboard quickly. The concept behind containerization

aligns naturally to microservices architecture, where complex applications are decomposed into smaller, loosely coupled microservices that can be developed and deployed independently.

Splunk unifies insights across container environments and the entire technology stack with a single solution, enabling developers to focus on what's most important — the application itself. Splunk for container monitoring provides a solution to help you ensure your containers are available and issues are fixed quickly with minimal effort.

Splunk Connect for Kubernetes

Getting Data In. Open Source and Splunk Supported.

Splunk Connect for Kubernetes is a simple, secure and scalable collector that can be broken down into three components — logging, objects and metrics. Leveraging popular CNCF-sponsored projects, you can manage your deployment with Helm and collect data via the HTTP Event Collector (HEC). Splunk Connect for Kubernetes is Splunk-built and supported in open source, and can be found at our GitHub Repository.



Who is this for?

Splunk Admins, container platform Engineering/ Admins, DevOps/Cloud/Digital teams running any Kubernetes-certified distribution - EKS, GKE, OpenShift, AKS, Docker EE, Private Cloud, Rancher.

Users like it because:

- Runs natively in Kubernetes
- Logs, metrics and metadata in one place
- Agent-based. Decoupled from the docker engine
- Memory & File buffering, resiliency against HEC interruption
- · Filtering allows routing & RBAC and re-shaping of data pre-Splunk to optimize ingest, reducing TCO and optimizing data retention
- Extensible as it builds on popular CNCF projects

Splunk Connect for Docker

Getting Data In. Open Source and Splunk Supported.

The Docker Logging Driver was originally introduced in 2015, and since then it has expanded to fit the needs of engineers. In 2018, Splunk introduced Splunk Connect for Docker, a certified Docker logging plug-in replacement for Docker Logging Driver that is also supported to Open Source. Like Splunk Connect for Kubernetes, it is optimized for consumption into Splunk through a HEC.

splunk

HTTP



Getting Data In & Running Splunk.

The Splunk Enterprise and Splunk Universal Forwarder are available as a Docker Images and can be deployed in supported configurations.

Teams using Splunk can get started with a Splunk Standalone Instances to analyze data from any source, or move large amounts of data with the Universal Forwarder, using a Node Agent, or Sidecar patterns.

Running a Splunk Universal Forwarder as a daemonset or global service is best suited for when you do not rely on Docker JSON Driver & Journald as log sources. If you plan to collect from those sources, check out Splunk Connect for Kubernetes and the configurations employed there as they are better suited to handle those sources.

Splunk can forward logs that aren't available to docker stderr/stdout, collecting them from within a container's volume mounts as a sidecar, or can be run to monitor common logging mounts you might employ.



Who is this for?

Docker Node

Container

Container

Splunk Admins, container platform Engineering/ Admins, and DevOps/Cloud/Digital teams running "Pure Docker" or Amazon ECS

docker

Users like it because:

- Quick setup, easy to install
- Flexible, can be used per container or at cluster level
- Removes the need for storage of logs on the Docker nodes
- Prefer not running an Node level Agent or Sidecars

Who is this for?

Splunk admins, container platform engineering/ admins, DevOps/cloud/digital teams running containers in any container platform that don't log to stderr/stdout, or that do not rely on docker json driver and/or journaled.

Users like it because:

- Well-known Splunk UF experience
- Collect from Apps that don't log to stderr/stdout
- Can be used in concert with Splunk Connect for Kubernetes/Docker

Learn more: Supporting documents for Splunk Deployments in Docker on **GitHub** and system requirements for **Containerized Computing Platforms**.

Running Splunk in Containers

There are a range of options you can explore with the Official Splunk Docker image.

Splunk Validated Architectures are recommended for any Splunk deployment and with docker, that is no different. Currently Splunk supports the "S" architecture, signifying single instance deployments. If you would like to explore more ambitious validated architectures, please join with us in our **community** supported test scenarios.

Conclusion

There are a number of advantages to running Splunk as a container in your environment — containers are easy, can be started and stopped quickly, have the ability to scale, can start up more instances when needed, and can shut down when a load subsides. You can create pre-configured tools that make it easy to load an instance for testing, proof-of-concept, or various other needs. Pre-configured images provided by Splunk via public repositories, forwarding options, and connectors, means running the Splunk platform in a container is easy, fast and efficient.

Splunk on Docker – Customer Outcomes

The following table depicts the types of deployments we see customers explore and have success with, based on the Splunk and container platform expertise available to them.

of Customer	Cluster Admin	Prototyping	Dev/Test Environment with S1/D1/C1/C3 Splunk Validated Architectures	Production deployments with S1/D1/C1/C3 Splunk Validated Architectures
Expertise	Basic Admin	Prototyping	Dev/Test Environment following S1/D1 Splunk Validated Architectures	Production deployments following S1/D1 Splunk Validated Architectures
Splunk	User	Prototyping	Prototyping	Prototyping
		BASIC	ADVANCED	PRODUCTION EXPERIENCE

Docker and Orchestration Expertise of Customer

Try Splunk Industrial Asset Intelligence for free or learn more about it **here**. Want to do more than infrastructure monitoring with Splunk? **Try Splunk Enterprise or Splunk Cloud** for free to see first hand the benefits it can bring to your organization.

splunk>

Learn more: www.splunk.com/asksales

www.splunk.com

© 2019 Splunk Inc. All rights reserved. Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners.