Overview

Docker Solution Briefs enable you to integrate the Docker Enterprise Edition (EE) container platform with popular 3rd party ecosystem solutions for networking, load balancing, storage, logging and monitoring, access management, and more. This document describes the setup needed to send Docker application logs to Log Insight.

This solution does not monitor Docker daemon logs.

Information on VMware vRealize Log Insight is provided by Docker as a known, working configuration for Docker engine 17.06.2-ee-12. Docker does not support VMware vRealize Log Insight. Please contact the vendor approved support methods if you have any questions or problems with them.

VMware vRealize Log Insight Overview

VMware vRealize Log Insight provides log management for infrastructure and applications in any environment. This scalable log management solution provides intuitive, actionable dashboards, sophisticated analytics, and third-party extensibility. It provides operational visibility and faster troubleshooting across physical, virtual, and cloud environments. vRealize Log Insight comes with built-in knowledge and native support for VMware vSphere with Operations Management.

Applications send data to Log Insight through a syslog interface or a native Log Insight API.

Log Insight comes in three versions: Standard, NSX, and Full. Third party integrations are available in the form of Content Packs, which contain pre-canned search expressions and dashboards. Only the Full version provides the ability to import content packs.
**Prerequisites**

This Solution Brief was tested with the following Docker Enterprise Edition and external components:

- ESXi 6.5.0 update 1
- VMware vCenter Appliance 6.5.0
- VMware Log Insight Appliance 4.5.1-6858700
- Logspout gliderlabs/logspout:3.2.5
- Docker EE 2.0
  - Docker EE Engine 17.06.2-ee-12
  - Docker UCP 3.0.1
  - Docker DTR 2.5.2

**Architecture**

Log Insight provides central storage for logs along with capabilities to parse, create patterns, and search through the logs quickly. It has a simple and intuitive interface to identify patterns that may be of interest to search through. Log Insight's dashboards can provide an overview of the system health.

Log Insight is a common tool in VMware environments since it generally comes packaged with vSphere licensing. There is native integration with VMware vCenter, ESXi, and vRealize Operations Management. The
integration between these tools allow the user to correlate metrics and logs together.

This Solution Brief shows how the same solution can be used to search the Docker application logs.

Log Insight can receive logs from sources including hardware, appliances, applications, and OSes. Logs are ingested through syslog (514/UDP, 514/TCP, 1514/TCP+SSL, 6514/TCP+SSL) or using the native VMware logging interface. Docker logs in this Solution Brief are ingested through the syslog interface.

Log Insight is installed as a virtual appliance. It can be configured in a clustered setup to handle large volumes of data ingestion. This solution has been tested against a single Log Insight instance.
This document describes how to send Docker application logs to Log Insight.

At a high level, a Logspout container is started on each Docker host. The Logspout container attaches to each container launched on that host and forwards the log streams to Log Insight. A custom Docker Content Pack can be loaded into Log Insight. This has pre-canned search expressions and dashboards that can be extended by the user.
Refer to VMware Log Insight documentation (https://docs.vmware.com/en/vRealize-Log-Insight/index.html) on how to deploy the appliance.

**Prerequisites**

The Docker EE worker nodes are assumed to be configured to handle Kubernetes or mixed workloads as the Logspout containers are deployed as a Kubernetes daemonset. If your environment has some nodes dedicated to Swarm, the Logspout services needs to be started on those nodes by running `docker stack deploy -c docker-compose.yml logspout-global` instead. Using this `docker-compose.yml` file (https://success.docker.com/api/asset/%2Fpublish%2Floginsight-logging%2F.%2Fsamples%2Fdocker-compose.yml) (click to download).

Confirm that the nodes are configured to handle Kubernetes only or mixed workloads:
Deployment

Run a Logspout container on all the nodes in the cluster including the manager nodes. Logspout is a log router for Docker containers and attaches to all containers on the host and routes their logs to a specified destination. Logspout captures stdout and stderr. This container is deployed on all the nodes using a Kubernetes YAML file.

Deploy this daemonset using the Logspout daemonset YAML file (https://success.docker.com/api/asset/%2Fpublish%2Floginsight-logging%2F.%2Fsamples%2Flogspout_ds.yml) (click to download) from the UCP console or a terminal console using a client bundle (https://docs.docker.com/ee/ucp/user-access/cli/).

The example logspout_ds.yml should be as follows:
The `nodeSelector` directive in the yaml file ensures that a Logspout pod is started on each host which is labeled with the key-value pair `app=logfwd`. This label needs to be applied manually to each node and the process described below.

1. Get the list of nodes, using `kubectl get node`. This example shows a setup with 3 nodes: dh1, dh2, dh3.

```
root@ubuntu:~# kubectl get nodes
NAME     STATUS   ROLES     AGE       VERSION
dh1      Ready    master   4d        v1.8.11-docker-8d637ae
dh2      Ready    <none>   4d        v1.8.11-docker-8d637ae
dh3      Ready    <none>   4d        v1.8.11-docker-8d637ae
```

2. Apply the label to each node with `kubectl label <node> app=logfwd`. This command should be run for on each node. This example shows the label applied to node dh2. Applied this command for nodes dh1 and dh3 as well.

```
root@ubuntu:~/bundle# kubectl label node dh2 app=logfwd
node "dh2" labeled
```
3. Confirm that each node has the label using `kubectl describe node dh2`. This example describes node dh2:

   ```
root@ubuntu:~# kubectl describe node dh1
Name:           dh1
Roles:          master
Labels:         app=logfwd
                beta.kubernetes.io/arch=amd64
                beta.kubernetes.io/os=linux
                com.docker.ucp.collection=system
                com.docker.ucp.collection.root=true
                com.docker.ucp.collection.swarm=true
                com.docker.ucp.collection.system=true
                kubernetes.io/hostname=dh1
                node-role.kubernetes.io/master=
Annotations:    alpha.kubernetes.io/provided-node-ip=44.44.0.11
                node.alpha.kubernetes.io/ttl=0
                volumes.kubernetes.io/controller-managed-attach-detach=true
CreationTimestamp: Mon, 04 Jun 2018 11:49:15 -0700
Taints:         com.docker.ucp.manager:NoSchedule
   ```

4. Once the label is applied on all the nodes, run `kubectl create -f logspout_ds.yml`. Confirm with daemonset is running - `kubectl get ds`:

   ```
root@ubuntu:~# kubectl get ds
NAME                     DESIRED  CURRENT  READY   UP-TO-DATE  AVAILABLE  NODE SELECTOR
logspout-forwarder-ds     3        3         3         3            3          app=logfwd
   ```

5. Confirm that the pods are running using `kubectl get po`:

   ```
root@ubuntu:~# kubectl get po
NAME                             READY  STATUS     RESTARTS  AGE
logspout-forwarder-ds-2bn9m      1/1    Running    0         4h
logspout-forwarder-ds-h8s9z      1/1    Running    0         3h
logspout-forwarder-ds-wzh4q      1/1    Running    0         4h
   ```

6. This can also be confirmed from the UCP UI:
The Logspout container exposes port 80 for an HTTP stream of the logs. This is remapped to port 9000 in the YAML file above.

Check the UCP scheduler settings to ensure that workloads can be deployed to manager nodes.

Views

Log Insight provides both log and dashboard views of data.
The log view or "Interactive Analytics" shows all data ingested by Log Insight. Log Insight automatically extracts fields from the data and allows the user to quickly filter on them.

In addition, a histogram showing the occurrence of events provides the user a graphical representation of events.

The dashboard view gives the user quick information about where events are coming from, the frequency, and event uniqueness. Log Insight is often used as a tool to quickly identify problems by correlating cause and effect scenarios.
Docker Content Pack

Docker has a custom content pack (https://success.docker.com/api/asset/%2Fpublish%2Floginsight-logging%2F.%2Fsamples%2FDocker-EE-ContentPack-v1.1.vlcp) that can be installed to filter logs from Docker containers including UCP and DTR output. The following shows the Dashboard view. Download and save this content pack to a file with the extension .vlcp.

From the dashboard view, the user can drill down by clicking into the particular widget to examine the logs. Click in the interactive analytics icon (red circle) to jump to a detailed filtered view of just those logs.
This shows the applications log details:

These content packs are pre-canned filters and can be modified.

**Installing Content Packs**

From the top right, click on the **Admin** dropdown menu:

Click on **IMPORT CONTENT PACK**:
Browse to the Content Pack, and load it into Log Insight. The content appears on the dashboards list:
Troubleshooting

The initial step in troubleshooting logs are as follows:

- Ensure that the Logspout container is running on all the Docker hosts.
- Ensure that the IP address of Log Insight in the Logspout compose file is correct.
- Next, try an application that generates console output; A suitable application is the well-known hello-world app:

  ```bash
  > docker run hello-world
  ```

- Dump the logs of the individual container under inspection to ensure it is generating logs:

  ```bash
  docker logs <container-name>
  ```

- Check that Logstream is getting the logs and forwarding them. For the first part, on the local machine, connect to the httpstream of the Logspout container. The following is an example of the output:
Check the Logspout logs to confirm a connection is being made to Log Insight. This sample shows a case where connection is refused:

```
root@ch1:-# docker ps | grep log
f6bd7f5fbd00  gliderlabs/logspout  "/bin/logspout sys..."  6 seconds ago  Up 5 seconds  vigorous_rosalind
```

Logspout is not available on Windows Server. Log Insight provides a native agent for Windows Server that can can forward Windows Event logs and monitor directories for changes. This section examines using the Log Insight Windows Agent to send container logs to the host.

**Requirements**

- Windows Server 2016 running Docker Engine version EE 17.06.2-ee-7 or higher
- Log Insight Windows Agent version 4.5.1

**Setup**

1. From the Log Insight UI, download the Windows Agent. Go to the **Admin** menu on the top right and select **Administration**. On the left hand side, click on **Agents** and then the **Download Log Insight Agent** link.
2. Copy the Agent to the Windows Server machine where you have Docker EE installed, and run the installer. Enter the Log Insight IP address in the configuration.

3. The Agent requires configuration. Since Docker keeps its container logs in \C:\ProgramData\Docker\containers, the Agent needs to be configured to monitor that directory. Open the file \C:\ProgramData\VMware\Log Insight Agent\liagent.ini. Add the following text into the file and save it:

```ini
[filelog|Docker]
directory=C:\ProgramData\docker\containers/*
tags={"Provider":"Docker EE"}
```

4. Restart the Log Insight Agent.

5. Run the `services` windows application and look for Log Insight Agent. Hit restart.
6. The agent now forwards logs to Log Insight. The Key Value Provider:Docker EE is added to the logs. Test this by running the `hello-world:nanoserver` container:
Windows Troubleshooting

- Ensure that the containers are generating logs. Go to `C:\ProgramData\Docker\containers\<containersid>`, and view the log file.
- Dump the Log Insight Agent, and review the logs. Run the Log Insight Agent support collector on the Windows machine. Examine the contents of the support bundle.
Further Reading

Refer to the following links for additional information:

- Logspout documentation (https://github.com/gliderlabs/logspout)