Pivotal

PRODUCT DOCUMENTATION

PCF Metrics®

Version 1.3

User's Guide

© Copyright Pivotal Software Inc, 2013-2018

Table of Contents

© Copyright Pivotal Software Inc, 2013-2019

PCF Metrics

Pivotal Cloud Foundry (PCF) Metrics stores logs, metrics data, and event data from apps running on PCF for the past two weeks. It graphically presents this data to help operators and developers better understand the health and performance of their apps.

Product Snapshot

The following table provides version and version-support information about PCF Metrics:

Element	Details
Version	v1.3
Release date	February 23, 2017
Compatible Ops Manager version(s)	v1.9.0 to v1.11.x
Compatible Elastic Runtime version(s)	v1.9.0 to v1.11.x
IaaS support	AWS, Azure, GCP, OpenStack, and vSphere

PCF Metrics User Guide

See the following topics for details about PCF Metrics:

- Installing PCF Metrics
- Sizing PCF Metrics For Your System
- Monitoring and Troubleshooting Apps with PCF Metrics
- Troubleshooting PCF Metrics
- PCF Metrics Product Architecture
- Release Notes and Known Issues

© Copyright Pivotal Software Inc, 2013-2019

Installing PCF Metrics

This document describes how to install and configure Pivotal Cloud Foundry (PCF) Metrics.

For information about the components deployed as part of this install procedure, see the PCF Metrics Product Architecture topic.

Prerequisites

- Ensure that you have installed the Elastic Runtime Tile.
- Ensure that you have installed v1.6 or later of the Redis tile.
- If you are running PCF on AWS, then ensure that, in Elastic Runtime, you have changed the **Loggregator Port** to `4443` from its value of `443`.
- If you are running PCF on Google Cloud Platform (GCP), then do the following to configure the DNS entries to accommodate web sockets:

    1. Log in to the GCP console.
    2. In the menu, navigate to the **Networking** tab and click **Load Balancing**.
    3. Find the load balancer that corresponds to `ENVIRONMENT-cf-ws`.
    4. Record the IP address.
    5. Click **Cloud DNS**, then click **ENVIRONMENT-zone**.
    6. Click **Add Record Set**.
    7. Enter a DNS name for mysql-logqueue. The DNS name should be `mysql-logqueue.SYSTEM_DOMAIN`. Refer to your ERT Tile's configuration of the System Domain under the Domains configuration section.
    8. In the IPv4 address field, enter the IP address of the load balancer that you recorded in Step 4.
    9. Leave the other fields as default.
    10. Repeat Steps 6–9 twice to create DNS records for elasticsearch-logqueue (`elasticsearch-logqueue.SYSTEM_DOMAIN`) and metrics (`metrics.SYSTEM_DOMAIN`).

Step 1: Add the PCF Metrics Tile to Ops Manager

Note: PCF Metrics should be installed on the same network as the Elastic Runtime Tile.

1. Download the PCF Metrics file from Pivotal Network.
2. Upload the PCF Metrics file to your Ops Manager installation.
3. Click **Add** next to the uploaded product description in the **Available Products** view to add PCF Metrics to your **Installation Dashboard**.

Step 2: Configure the PCF Metrics Tile

Note: The following procedures offer a standard configuration. To customize PCF Metrics for high capacity, see the Sizing PCF Metrics For Your System topic.

From the **Installation Dashboard**, click the **PCF Metrics** tile.

Assign Availability Zones (AZs) and Networks.

1. Click **Assign AZs and Networks**.
2. Select an Availability Zone under **Place singleton jobs**.
   Ops Manager runs Metrics jobs with a single instance in this Availability Zone.
3. Select one or more Availability Zones under **Balance other jobs**.
   Ops Manager balances instances of Metrics jobs with more than one instance across the Availability Zones that you specify.
4. Use the drop-down menu to select a network.

© Copyright Pivotal Software Inc, 2013-2019

5. Click **Save**.

Data Services Ports

For reference, the following table shows the port associated with each data service.

Service	Port
Elasticsearch	9200
MySQL	3306

MySQL Alerts

1. Click **MySQL Alerts**.
2. Set the **Email** value. Alerts for issues storing metrics into the MySQL cluster will be sent to this email address.

Data Store

1. Click **Data Store** (Only for PCF Metrics **1.3.4 and lower**)
2. Review the **Elastic Search Heap Size** value. Elastic Search memory allocation for Heap use. Set to 50% of the memory allocated to the smallest of the Elasticsearch instances in Resource Config or 31GB, whichever is smaller. Use a unit of M for megabytes or G for gigabytes.
3. Review the **MySQL InnoDB Buffer Size** value. Set to 80% of MySQL Server VM memory. Use a unit of M for megabytes or G for gigabytes.
4. Review the **MySQL Logqueue Count** value. You can increase this instance count at any time to accommodate higher levels of inbound metrics traffic.
5. Review the **Elasticsearch Logqueue Count** value. You can increase this instance count at any time to accommodate higher levels of inbound log traffic.
6. Review the **Ingestor Count** value. You can increase this instance count at any time to accommodate higher levels of Loggregator Firehose traffic.
7. Click **Save**.

Errands

To properly configure the **Errands** pane, follow the procedure that corresponds to your Ops Manager version.

Ops Manager v1.9

1. Click **Errands**.

   Note: The PCF Metrics tile selects all **Post-Deploy Errands** by default. Pivotal recommends that you do not deselect any errands as doing so can cause issues with the deployment of the tile. However, you can deselect the **Remove Legacy PCF Metrics CF Resources** errand after deploying v1.3 of the tile.

2. Review the **Post-Deploy Errands** and **Pre-Delete Errands**:

   a. If you are deploying the tile for the first time, select all **Post-Deploy Errands**.

   - The following list describes what the **Smoke tests** errand does. For information about resolving errors discovered by this errand, see the Smoke Test Errors section of the *Troubleshooting PCF Metrics* topic.
     - Confirms that MySQL ingests metrics
     - Confirms that Elasticsearch ingests logs
     - Confirms that the APIs return metrics and logs

Note: If you deselect **Remove PCF Metrics 1.3 CF Resources** under **Pre-Delete Errands**, artifacts may remain after the PCF Metrics tile uninstalls.

Ops Manager 1.10 and up

1. Click **Errands**.

warning: By default, The PCF Metrics tile sets all **Post-Deploy Errands** to only run when changed. You must make the configuration changes below for the first deployment of the PCF Metrics tile to succeed.

2. Review the **Post-Deploy Errands** and **Pre-Delete Errands**:

    a. If you are deploying the tile for the first time, set **Post-Deploy Errands** to **On**.
    b. For deployments after the initial install, configure the **Post-Deploy Errands** as follows:

        i. Set **Remove Legacy PCF Metrics CF Resources** to **When Changed**
        ii. Set **Remove PCF Metrics 1.2 CF Resources** to **When Changed**
        iii. Set **Push PCF Metrics components** to **On**
        iv. Set **Smoke tests for Metrics Data and Apm** to **On**

- The following list describes what the **Smoke tests** errand does. For information about resolving errors discovered by this errand, see the Smoke Test Errors section of the *Troubleshooting PCF Metrics* topic.
    - Confirms that MySQL ingests metrics
    - Confirms that Elasticsearch ingests logs
    - Confirms that the APIs return metrics and logs

Note: If you set **Remove PCF Metrics 1.3 CF Resources** under **Pre-Delete Errands** to **Off**, artifacts may remain after the PCF Metrics tile uninstalls.

Resource Config

1. Click **Resource Config**.

2. Review the resource configurations. By default, the settings match the instance types that are best suited for each job. For reference, the following table shows the default resource and IP requirements for installing the PCF Metrics tile:

Note: The actual default values you will see in your PCF Metrics tile will depend on the available resources for your PCF deployment. Refer to Sizing PCF Metrics For Your System to correctly configure Metrics resources for your deployment.

Resource	Instances	Persistent	CPU	RAM	Ephemeral	Static IP	Dynamic IP
Elasticsearch Master	3	10 GB	4	16 GB	32 GB	3	0
Elasticsearch Coordinator	2	1 GB	2	16 GB	32 GB	1	0
Elasticsearch Data	4	100 GB	2	16 GB	32 GB	4	0
MySQL Server	2	100 GB	2	16 GB	32 GB	3	0
MySQL Proxy	2	*n/a*	2	16 GB	32 GB	2	0
MySQL Monitor	1 (not configurable)	*n/a*	2	16 GB	32 GB	0	1
Metron	1 (not configurable)	*n/a*	2	4 GB	32 GB	1	0

If you expect a high level of use, you may need to increase the disk resources available to your instances. For information about sizing, see Sizing PCF Metrics For Your System.

Note: There have been issues with the Ops Manager Bosh Director correctly partitioning persistent disks larger than 2 TB.

3. Click **Save**.

Stemcell

1. Navigate to Pivotal Network and click **Stemcells**.
2. Download the appropriate stemcell version for your IaaS.

   Note: On AWS make sure to use a HVM stemcell if you are using the default instance sizes.

3. Click **Import Stemcell** and select the stemcell file you downloaded.

Step 3: Configure the Temporary Datastore

PCF Metrics uses a temporary datastore during Elasticsearch downtime, including upgrades, to significantly reduce log loss by continuing to store app logs from the Loggregator Firehose.

To configure the temporary datastore, follow the instructions in the Configuring the Temporary Datastore section of *Sizing PCF Metrics for your System*.

Step 4: Deploy PCF Metrics

Click **Apply Changes** to install the service. If the smoke tests fail, see the Troubleshoot Smoke Test Errors section of the *Troubleshooting PCF Metrics* topic.

Review the Monitoring and Troubleshooting Apps with PCF Metrics topic for more information about how to log in, use, and interpret data from PCF Metrics.

© Copyright Pivotal Software Inc, 2013-2019

Sizing PCF Metrics for Your System

This topic describes how to configure Pivotal Cloud Foundry (PCF) Metrics for high availability. Operators can use these procedures to optimize PCF Metrics for high capacity.

For more information about PCF Metrics components, see the PCF Metrics Product Architecture topic.

Configuring the Metrics Datastore

PCF Metrics stores metrics in a MySQL cluster.

To customize PCF Metrics for high capacity, you can add memory and persistent disk to the MySQL server nodes.

Considerations for Scaling

Because apps emit logs at different volumes and frequencies, you should not scale the MySQL server nodes in accordance to the number of app instances in your deployment. Because of the ease in scaling these components, we recommend starting with a minimal configuration then evaluating performance over a period of time and scaling. As long as persistent disk is being scaled up, there should not be any fear of losing data.

To determine approximate starting memory and disk allocation for each MySQL server node, use the following example configurations:

Small: Test Use ~100 Apps

warning: Do NOT attempt to size your PCF Metrics deployment any smaller than this setting. These are the minimum resources required.

Job	Instances	Persistent Disk Type	VM Sizing
MySQL Server	1	50 GB	CPU: 1, RAM: 1 GB, disk: 8 GB
MySQL Proxy	1	None	CPU: 1, RAM: 1 GB, disk: 8 GB
MySQL Monitor	0	None	N/A

Medium: Production Use ~5000 Apps

Job	Instances	Persistent Disk Type	VM Sizing
MySQL Server	3	500 GB	CPU: 2, RAM: 12 GB, disk: 32 GB
MySQL Proxy	1	None	CPU: 2, RAM: 4 GB, disk: 8 GB
MySQL Monitor	1	None	CPU: 2, RAM: 4 GB, disk: 8 GB

Large: Production Use ~15000 Apps

Job	Instances	Persistent Disk Type	VM Sizing
MySQL Server	3	~1.5 TB	CPU: 8, RAM: 64 GB, disk: 160 GB
MySQL Proxy	1	None	CPU: 2, RAM: 8 GB, disk: 32 GB
MySQL Monitor	1	None	CPU: 2, RAM: 4 GB, disk: 8 GB

warning: The number of instances of MySQL Server MUST be an odd number. For more information about about scaling MySQL Server instances, see Avoid an Even Number of Nodes in the MySQL *Architecture* topic.

Use these results as guidelines. Consider configuring your MySQL server nodes with additional memory and disk. If your deployment adds additional app instances, consider configuring your MySQL server nodes with additional memory and disk.

© Copyright Pivotal Software Inc, 2013-2019

Procedures for Scaling

After determining the amount of memory and persistent disk required for each MySQL server node, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard and click the **Metrics** tile.
2. From the **Settings** tab of the **Metrics** tile, click **Resource Config**.
3. Modify the memory limit or persistent disk allocation as needed for your environment.

   > **warning:** There have been issues with Ops Manager BOSH Director using persistent disks larger than 2 TB.

4. If you modify the memory allocation for the MySQL server nodes on PCF Metrics v1.3.4 or older, you must also update the MySQL InnoDB Buffer Size setting. Pivotal recommends that you set the buffer size to 80% of the memory allocated to that VM. To change the MySQL InnoDB Buffer Size:
   a. Navigate to the Ops Manager Installation Dashboard and click the **Metrics** tile.
   b. From the **Settings** tab of the **Metrics** tile, click **Data Store**.
   c. Update the **MySQL InnoDB Buffer Size** input field.

Configuring the Log Datastore

PCF Metrics uses Elasticsearch to store logs. Each Elasticsearch node contains multiple shards of log data, divided by time slice. To customize PCF Metrics for high capacity, you can scale the number of Elasticsearch data nodes.

Considerations for Scaling

To determine the number of Elasticsearch data nodes required for PCF Metrics, consider how many logs the apps in your deployment emit and the average size of each log.

If your average log size is 1 kilobyte, and each node has 1 terabyte of available disk space, then each node has a maximum storage capacity of 1 billion log messages. If your apps emit 3 billion logs over a 24-hour period, you need at least 3 nodes to hold the data and 3 additional nodes for high-availability replication.

This example assumes that your apps emit logs at a continuous rate over 24 hours. However, apps typically do not emit logs continuously. If your apps emit 2 billion of the 3 billion logs between 8 AM and 4 PM, you must determine the minimum node-to-shard ratio to accommodate that rate over the 8-hour period. Because your apps emit 1 billion logs over a 4 hour span, you need at least 6 nodes (24 hours/6 nodes = 4 hours worth of shards per node) to hold the data and an additional 6 nodes for high-availability replication.

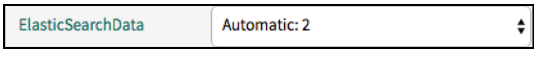
You can also use the throughput of logs per minute to help determine how many Elasticsearch data nodes to provision. As a general rule, provision one data node for every 5000 logs received in one minute.

Procedures for Scaling

> **warning**: If you modify the number of Elasticsearch instances, the Elasticsearch cluster temporarily enters an unhealthy period during which it does not ingest any new logs data, due to shard allocation.

After determining the number of Elasticsearch nodes needed for your deployment, perform the following steps to scale your nodes:

1. Navigate to the Ops Manager Installation Dashboard and click the **Metrics** tile.
2. From the **Settings** tab of the **Metrics** tile, click **Resource Config**.
3. Locate the **ElasticSearchData** job and select the dropdown menu under **Instances** to change the number of instances.



4. Click **Save**.

Configuring the Temporary Datastore

Note: PCF Metrics uses the temporary datastore only when upgrading from v1.3 or later.

During Elasticsearch downtime, including upgrades, PCF Metrics stores logs from the Loggregator Firehose in a temporary Redis datastore. When the upgrade finishes, the the Elasticsearch logqueue restores data from the temporary datastore by placing it in Elasticsearch. See PCF Metrics Product Architecture for more information.

Procedure for Scaling

This procedure describes how to size the temporary datastore for your system by calculating your requirements and setting the VM size in Ops Manager.

Calculate Storage Requirements

Calculate the storage requirements of the temporary datastore using the table below as a guide:

Variable	Estimate a value for the variable
`AvgLogSize`	Collect a sample of logs from your apps and calculate the average size of a log file in KB. **Note**: PCF apps run on Diego cells that do not emit individual logs greater than approximately 60 KB. See Log Message Size Constraints.

`LogsPerSec`	1. Install the nozzle plugin for the cf CLI if you do not already have it.
	2. Run following command to estimate of the volume of `LogMessages` emitted from the Loggregator Firehose. Every 10 seconds, the command gives you the average number of log messages per second, averaged over the previous 10 seconds.
	`$ cf nozzle -filter LogMessage | pv -l -i10 -r >/dev/null`
	`[50.5 /s]`

`UpgradeTime`	Estimate how long your PCF Metrics upgrade will take. In tests in a large production environment, the upgrade takes 20–30 minutes.
`RedisRAM`	Find the value for `RedisRAM` using the following calculation: `RedisRAM = AvgLogSize * LogsPerSec * UpgradeTime * 60`
`AdjustedRedisRAM`	Because Redis only allows VMs to use 45% of their allocated RAM, adjust the storage requirement as follows: `AdjustedRedisRAM = RedisRAM / 0.45`

Set VM Size

Follow these steps to configure the size of the Redis VM for the temporary datastore based on your calculations.

Note: In the case that the temporary datastore becomes full, Redis uses the `volatile-ttl` eviction policy to continue storing incoming logs. For more information, see the *Eviction policies* section of Using Redis as an LRU cache.

1. Navigate to the Ops Manager Installation Dashboard and click the **Redis** tile.
2. From the **Settings** tab, click **Resource Config**.
3. In the **Dedicated Node** row, under **VM Type**, select an option with at least enough **RAM** to support the value for `AdjustedRedisRAM`.
4. Click **Save**.

Configuring the Ingestor

PCF Metrics deploys the Ingestor as an app within PCF. The Ingestor consumes logs and metrics from the Loggregator Firehose, sending metrics and logs to their respective Logqueue apps. To customize PCF Metrics for high capacity, you can scale the number of Ingestor app instances and increase the amount of memory per instance.

Considerations for Scaling

Because apps emit logs at different volumes and frequencies, you should not scale the Ingestor by matching the number of Ingestor instances to the number of app instances in your deployment.

Because Ingestor performance is affected by Loggregator performance, it can be difficult to determine in advance the proper configuration. Because of the ease in scaling these components, we recommend starting with a minimal configuration then evaluating performance over a period of time and scaling

The Ingestor app can handle relatively large loads. For high availability, you must have at least two instances of the Ingestor app running. If your deployment runs fewer than 2000 app instances, two instances of the Ingestor app are sufficient.

Procedures for Scaling

warning: If you decrease the number of Ingestor instances, you may lose data currently being processed on the instances you eliminate.

After determining the number of Ingestor app instances needed for your deployment, perform the following steps to scale the Ingestor:

1. Target your Cloud Controller with the Cloud Foundry Command Line Interface (cf CLI). If you have not installed the cf CLI, see the Installing the cf CLI topic.

```
$ cf api api.YOUR-SYSTEM-DOMAIN
Setting api endpoint to api.YOUR-SYSTEM-DOMAIN...
OK
API endpoint:   https://api.YOUR-SYSTEM-DOMAIN (API version: 2.54.0)
Not logged in. Use 'cf login' to log in.
```

2. Log in with your UAA administrator credentials. To retrieve these credentials, navigate to the **Pivotal Elastic Runtime** tile in the Ops Manager Installation Dashboard and click **Credentials**. Under **UAA**, click **Link to Credential** next to **Admin Credentials** and record the password.

```
$ cf login
API endpoint: https://api.YOUR-SYSTEM-DOMAIN


Email> admin
Password>
Authenticating...
OK
```

3. When prompted, target the `metrics` space.

```
Targeted org system


Select a space (or press enter to skip):
1. system
2. notifications-with-ui
3. autoscaling
4. metrics


Space> 4
Targeted space metrics


API endpoint:   https://api.YOUR-SYSTEM-DOMAIN (API version: 2.54.0)
User:           admin
Org:            system
Space:          metrics
```

4. Scale your Ingestor app to the desired number of instances:

```
$ cf scale metrics-ingestor -i INSTANCE-NUMBER
```

5. Evaluate the CPU and memory load on your Ingestor instances:

```
$ cf app metrics-ingestor
Showing health and status for app metrics-ingestor in org system / space metrics as admin...
OK


requested state: started
instances: 1/1
usage: 1G x 1 instances
urls:
last uploaded: Sat Apr 23 16:11:29 UTC 2016
stack: cflinuxfs2
buildpack: binary_buildpack



     state     since                  CPU    memory       disk         details
#0   running   2016-07-21 03:49:58 PM   2.9%   13.5M of 1G   12.9M of 1G
```

If your average memory usage exceeds 50% or your CPU consistently averages over 85%, add more instances with `cf scale metrics-ingestor -i INSTANCE-NUMBER`.

In general, you should scale the Ingestor app by adding additional instances. However, you can also scale the Ingestor app by increasing the amount of memory per instance:

```
$ cf scale metrics-ingestor -m NEW-MEMORY-LIMIT
```

For more information about scaling app instances, see Scaling an Application Using cf scale.

Configuring the Logqueues

PCF Metrics deploys a MySQL Logqueue and an Elasticsearch Logqueue as apps within PCF. The MySQL logqueue consumes metrics from the Ingestor and forwards them to MySQL. The Elasticsearch logqueue consumes logs from the Ingestor and forwards them to Elasticsearch. To customize PCF Metrics for high capacity, you can scale the number of Logqueue app instances and increase the amount of memory per instance.

Considerations for Scaling

The number of MySQL and Elasticsearch logqueues needed is dependent on the frequency of logs and metrics being forwarded by the Ingestor. As a general rule, for every 45,000 logs per minute, add 2 Elasticsearch logqueues. For every 17,000 metrics per minute, add 1 MySQL Logqueue. This is a general estimate and you may need fewer instances depending on your deployment. To optimize resource allocation, provision fewer instances initially and increase instances until you achieve desired performance.

Procedures for Scaling

To modify your Elasticsearch Logqueue app instances, you must first target your Cloud Controller, log in with your UAA administrator credentials, and target the `metrics` space by following steps 1-3 in the Procedures for Scaling.

To scale your Logqueue app instances, perform the following command:

```
$ cf scale elasticsearch-logqueue -i INSTANCE-NUMBER
```

To scale the memory limit per Logqueue app instance, perform the following command:

```
$ cf scale elasticsearch-logqueue -m NEW-MEMORY-LIMIT
```

To modify your MySQL Logqueue app instances, you must first target your Cloud Controller, log in with your UAA administrator credentials, and target the `metrics` space by following steps 1-3 in the Procedures for Scaling.

To scale your Logqueue app instances, perform the following command:

© Copyright Pivotal Software Inc, 2013-2019

```
$ cf scale mysql-logqueue -i INSTANCE-NUMBER
```

To scale the memory limit per Logqueue app instance, perform the following command:

```
$ cf scale mysql-logqueue -m NEW-MEMORY-LIMIT
```

warning: If you decrease the number of Logqueue instances, you may lose data currently being processed on the instances you eliminate.

© Copyright Pivotal Software Inc, 2013-2019

Monitoring and Troubleshooting Apps with PCF Metrics

This topic describes how developers can monitor and troubleshoot their apps using Pivotal Cloud Foundry (PCF) Metrics.

Overview

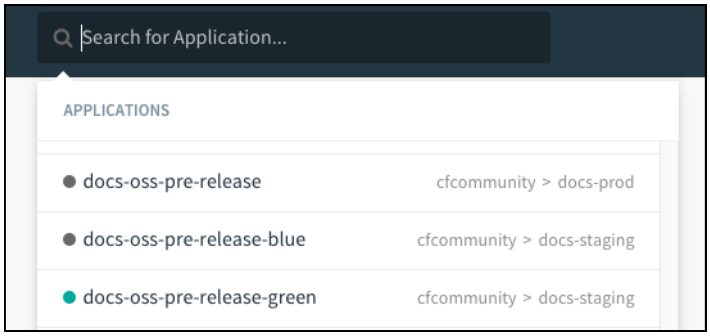
PCF Metrics helps you understand and troubleshoot the health and performance of your apps by displaying the following:

- Container Metrics: A graph of CPU, memory, and disk usage percentages
- Network Metrics: A graph of requests, HTTP errors, and response times
- App Events: A graph of update, start, stop, crash, SSH, and staging failure events
- Logs: A list of app logs that you can search, filter, and download
- Trace Explorer: A graph that traces a request as it flows through your apps and their endpoints, along with the corresponding logs

For example, if you see that an app crashed on the **Events** graph, you can zoom in and view the corresponding container metrics, network metrics, and logs.

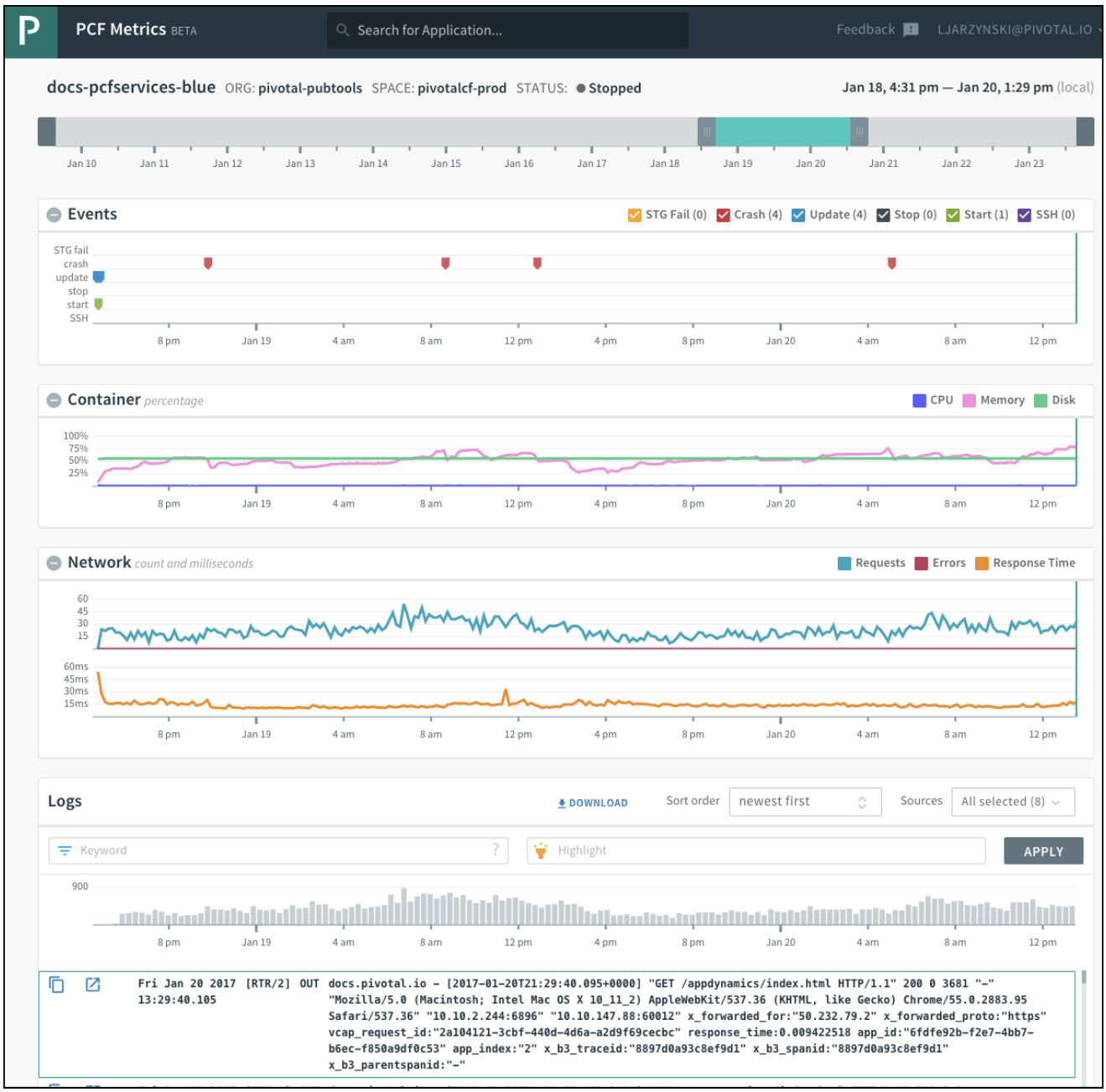
View an App

In a browser, navigate to `metrics.YOUR-SYSTEM-DOMAIN` and log in with your User Account and Authentication (UAA) credentials. Choose an app for which you want to view metrics or logs. You can view any app that runs in a space that you have access to.



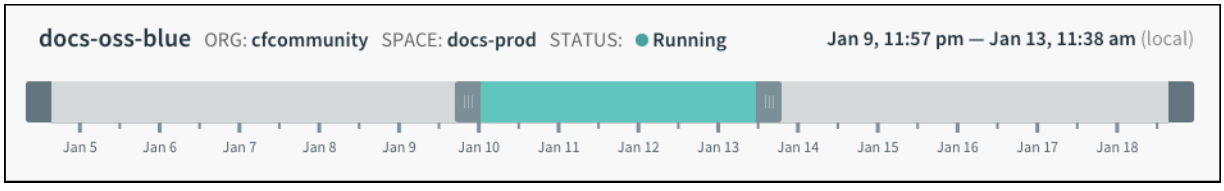
PCF Metrics displays app data for a given time frame. See the sections below to Change the Time Frame for the dashboard, Interpret Metrics information on each graph, and Trace App Requests with the Trace Explorer.

© Copyright Pivotal Software Inc, 2013-2019



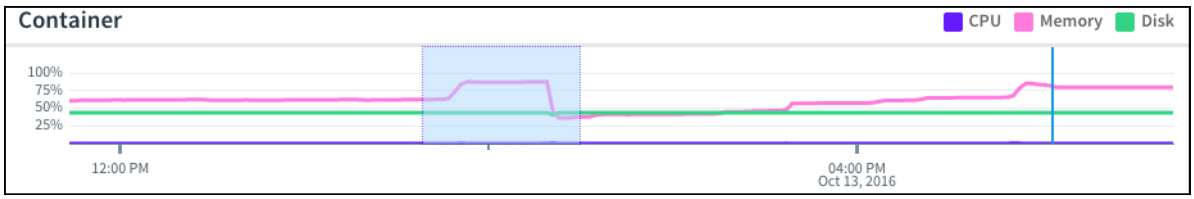
Change the Time Frame

The graphs show time along the horizontal axis. You can change the time frame for all graphs and the logs by using the time selector at the top of the window. Adjust either end of the selector, or click and drag.



Zoom: From within any graph, click and drag to zoom in on areas of interest. This adjusts all of the graphs, and the logs, to show data from that time frame.

© Copyright Pivotal Software Inc, 2013-2019



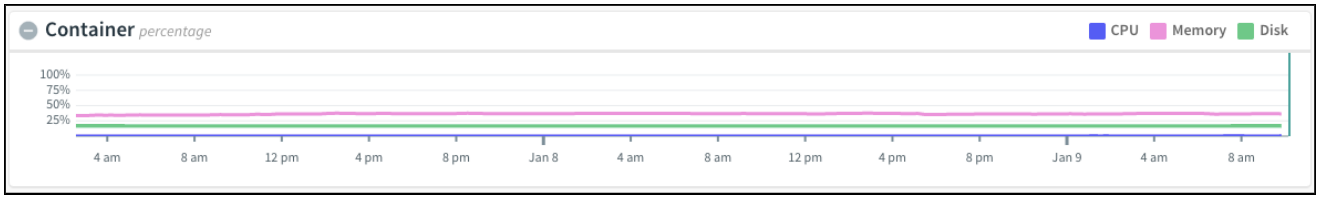
Drag: From underneath the x-axis of any graph, drag left or right to view data for an earlier or later time.

Interpret Metrics

See the following sections to understand how to use each of the views in the dashboard to monitor and troubleshoot your app.

Container Metrics

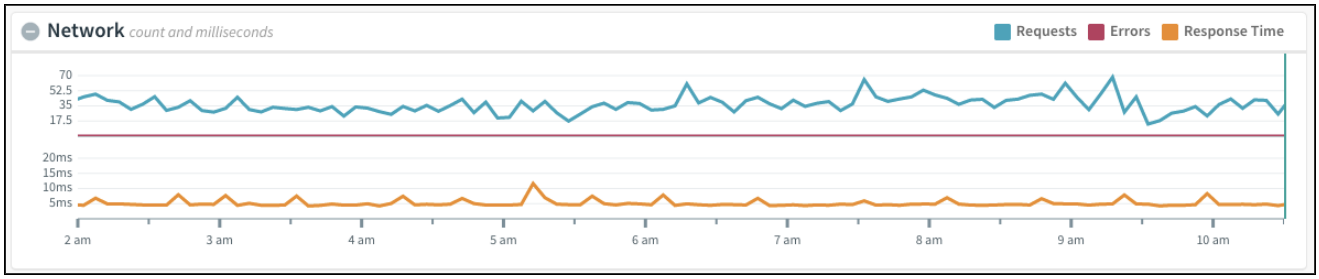
The **Container Metrics** graph displays **CPU**, **Memory**, and **Disk** usage:



- A spike in CPU might point to a process that is computationally heavy. Scaling app instances can relieve the immediate pressure, but investigate the app to better understand and fix the root cause.
- A spike in memory might mean a resource leak in the code. Scaling app memory can also relieve the immediate pressure, but look for and resolve the underlying issue so that it does not occur again.
- A spike in disk might mean the app is writing logs to files instead of STDOUT, caching data to local disk, or serializing huge sessions to disk.

Network Metrics

The **Network Metric** graph displays HTTP **Requests and Errors** and **Response Time**:

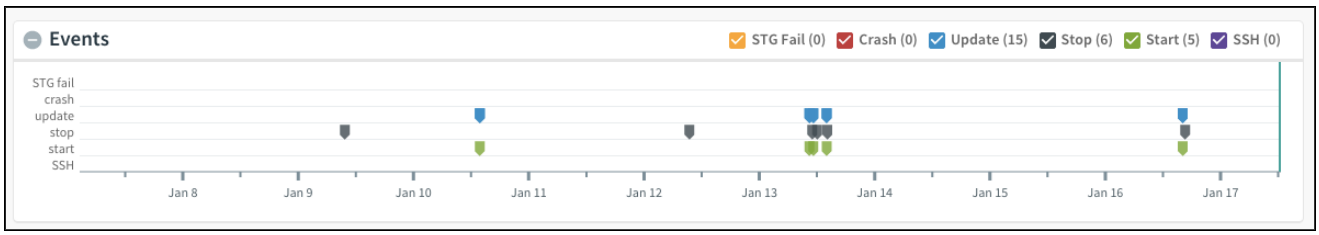


- A spike in response time means your users are waiting longer to use your app. Scaling app instances can spread that workload over more resources and result in faster response times.
- A spike in HTTP errors means one or more 5xx errors have occurred. Check your app logs for more information.
- A spike in HTTP requests means more users are using your app. Scaling app instances can reduce the higher response time that may result.

Events

The **Events** graph shows the following app events: staging failures (**STG Fail**), **Crash**, **Update**, **Stop**, **Start**, and **SSH**. You can change which events you see using the checkboxes in the upper right.

© Copyright Pivotal Software Inc, 2013-2019



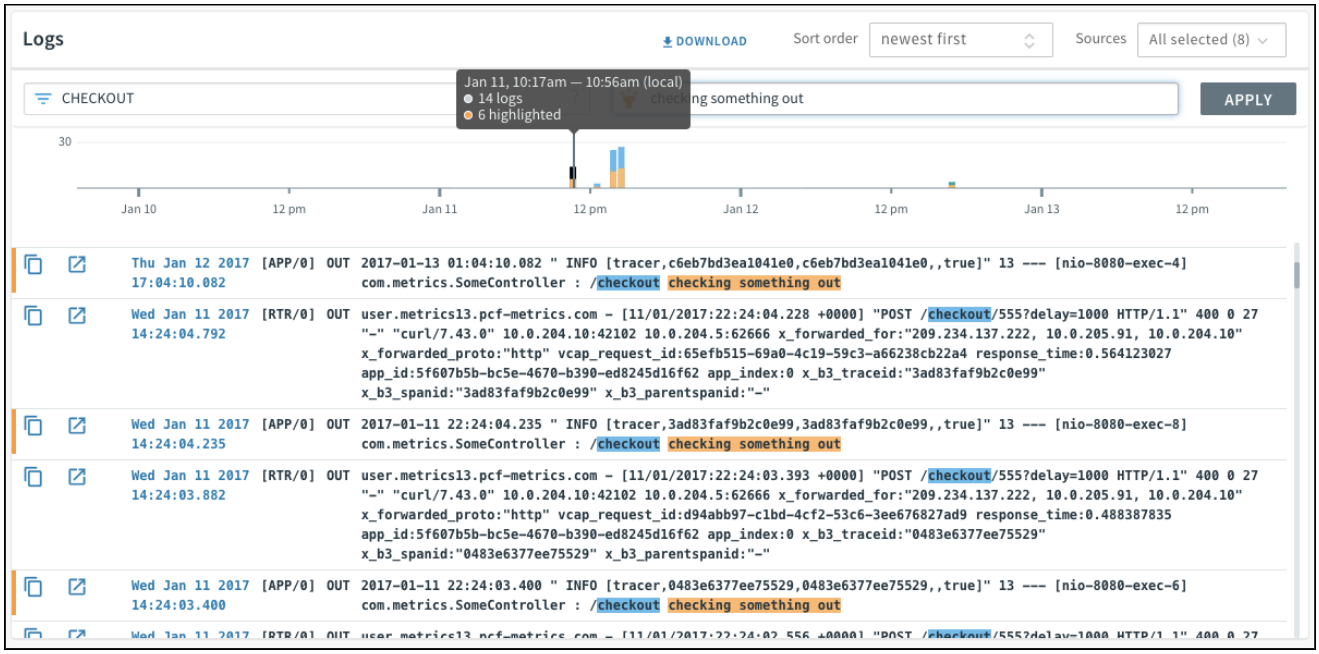
Note: The **SSH** event corresponds to someone successfully using SSH to access a container that runs an instance of the app.

See the following topics for more information about app events:

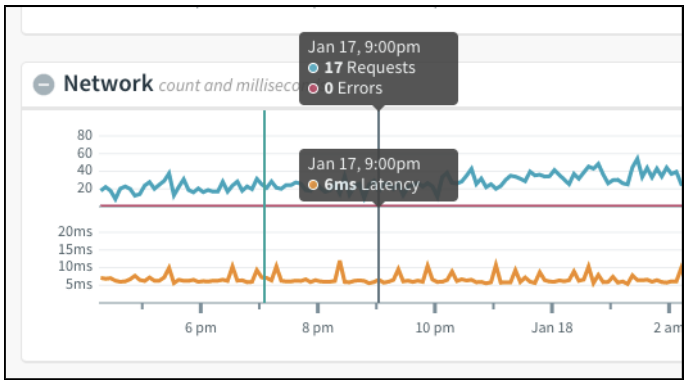
- About Starting Applications
- Troubleshooting Application Deployment and Health

Logs

The **Logs** view displays app log data ingested from the Loggregator Firehose, including a histogram that displays log frequency for the current time frame:



The list of logs begins at the time indicated by the placement of the log line in the **Events**, **Container**, and **Network** graphs. To adjust the placement of the log line, hover over the graph and click a new location:



You can interact with the **Logs** view in the following ways:

- **Keyword**: Perform a keyword search. The histogram updates with blue bars based on what you enter. Hover over a histogram bar to view the amount

of logs for a specific time based on your filter.

- **Highlight**: Enter a term to highlight within your search. The histogram updates with yellow bars based on the results. Hover over a histogram bar to view the amount of logs for a specific time that contain the highlighted term.
- **Sources**: Choose which sources to display logs from. For more information, see Log Types and Their Messages.
- **Order**: Modify the order in which logs appear.
- **Download**: Download a file containing logs for the current search.
- **Copy**: Click the copy icon to copy the text of the log.
- **View in Trace Explorer**: Open a window to see the trace of the request associated with the log. See Trace App Requests.

Trace App Requests

A request to one of your apps initiates a workflow within the app or system of apps. The record of this workflow is a *trace*, which you can use to troubleshoot app failures and latency issues. In the Trace Explorer view, PCF Metrics displays an interactive graph of a trace and its corresponding logs. See the sections below to understand how to use the Trace Explorer.

For more information about traces, see the What is a Trace? section of the *Open Tracing* documentation.

Prerequisite

PCF Metrics constructs the Trace Explorer view using trace IDs shared across app logs. Before you use the Trace Explorer, examine the following list to ensure PCF metrics can extract the necessary data from your app logs for your specific app type.

- **Spring**: Follow the steps below.

    1. Ensure you are using Spring Boot v1.4.3 or later.
    2. Ensure you are using Spring Cloud Sleuth v1.0.12 or later.
    3. Add the following to your app dependency file:

    ```
    dependencies { (2)
    compile "org.springframework.cloud:spring-cloud-starter-sleuth"
    }
    ```

- **Node.js**, **Go**, **Python**: Ensure that the servers associated with your app do not modify HTTP requests in a way that removes the `X-B3-TraceId`, `X-B3-SpanId`, and `X-B3-ParentSpan` headers from a request. You do not have to add any dependency to your app.
- **Ruby**: Ruby servers that use a library depending on Rack modify HTTP request headers in way that is incompatible with PCF Metrics. If you want to trace app requests for your Ruby apps, ensure that your framework does not rely on Rack. You may need to write a raw Ruby server that preserves the `X-B3-TraceId`, `X-B3-SpanId`, and `X-B3-ParentSpan` headers in the request.

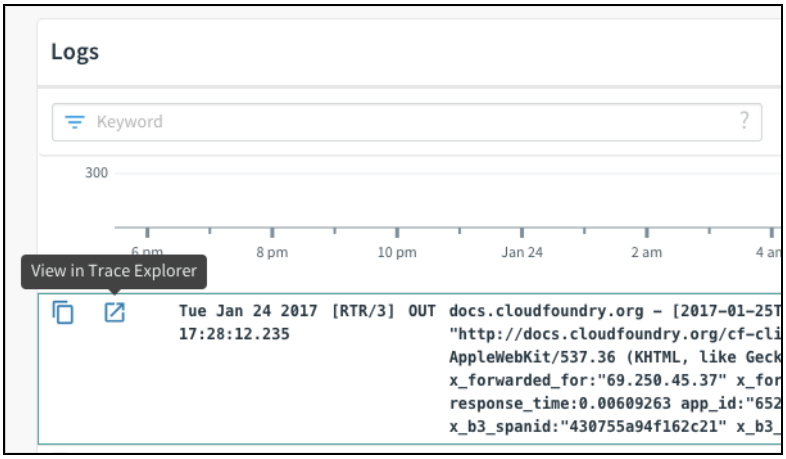
Use the Trace Explorer

This section explains how to view the trace for a request received by your app and interact with the Trace Explorer.

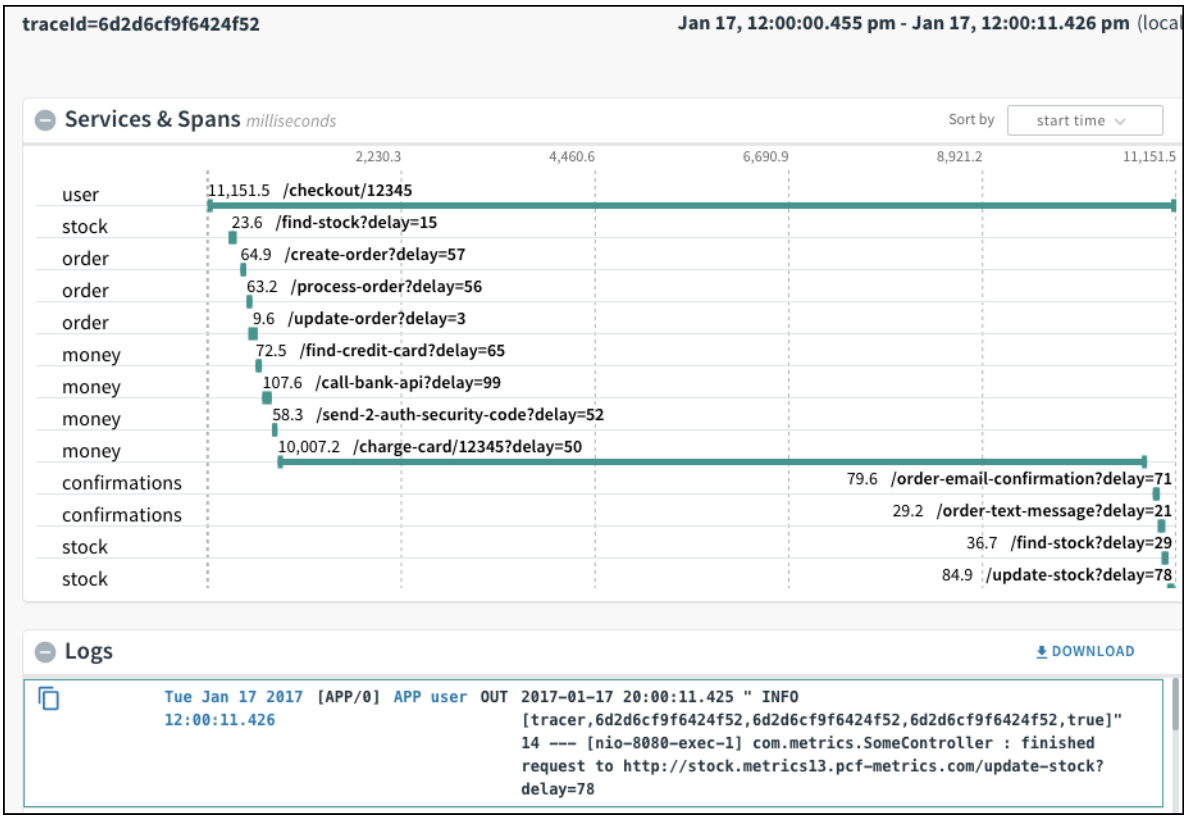
1. Select an app in the PCF Metrics dashboard.

2. Click the Trace Explorer icon in a log for which you want to trace the request.

© Copyright Pivotal Software Inc, 2013-2019



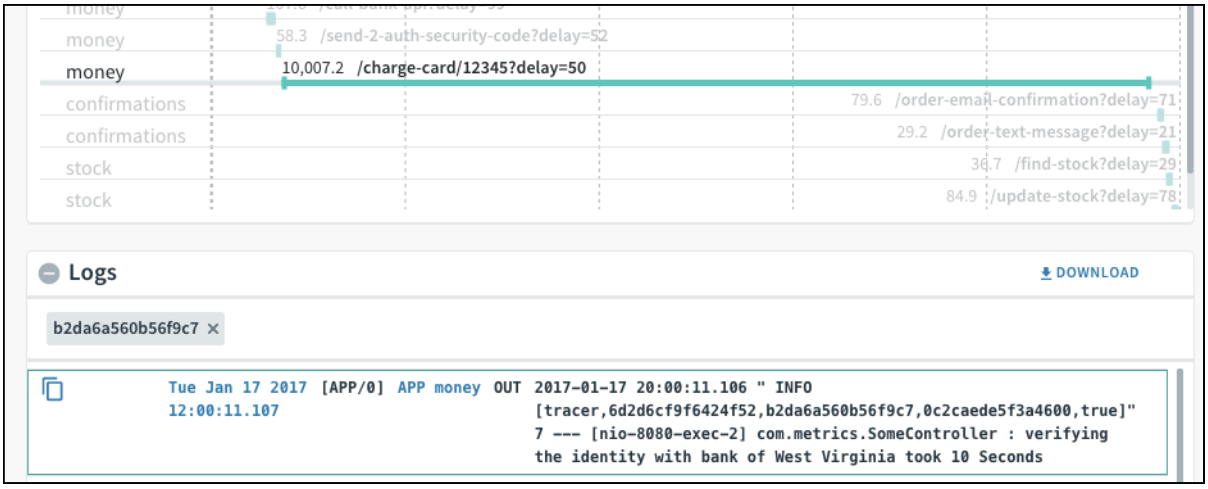
- The Trace Explorer displays the apps and endpoints involved in a completing a request, along with the corresponding logs:



A request corresponds to a single trace ID, displayed in the top left. Each row includes an app in the left column and a *span* in the right column. A span is a particular endpoint within the app and the time it took to execute, in milliseconds. By default, the graph lists each app and endpoint in the order they were called.

> **Note**: If you do not have access to the space for an app involved in the request, you cannot see the spans or logs from that app.

- You can click a span to show only logs from that span, or any number of spans to toggle which logs appear. Clicking a span also creates a box with that particular span ID in the **Logs** view:



- If you click `APP APP-NAME` within a log, PCF Metrics returns you to the dashboard view for that app, with the time frame focused on the time of the log that you clicked from.

© Copyright Pivotal Software Inc, 2013-2019

Troubleshooting PCF Metrics

This topic describes how to resolve common issues experienced while operating or using Pivotal Cloud Foundry (PCF) Metrics.

Errors during Deployment

The following sections describe errors that cause failure during a PCF Metrics tile and how to troubleshoot them.

Smoke Test Errors

PCF Metrics runs a set of smoke tests during installation to confirm system health. If the smoke tests discover any errors, you can find a summary of those errors at the end of the errand log output, including detailed logs about where the failure occurred.

The following tables describe common failures and how to resolve them.

Insufficient Resources

Error	`Insufficient Resources`
Cause	Your PCF deployment has insufficient Diego resources to handle the apps pushed as part of a PCF Metrics installation. The PCF Metrics tile deploys the following apps: **App** / **Memory** / **Disk** `metrics-ingestor`* / 512MB / 1GB `mysql-logqueue`* / 1GB / 1GB `elasticsearch-logqueue`* / 512MB / 1GB `metrics-aggregator` / 256MB / 1GB `metrics` / 1GB / 1GB `worker-app-dev` / 1GB / 1GB `worker-app-logs` / 1GB / 1GB `worker-health-check` / 1GB / 1GB `worker-reaper` / 1GB / 1GB *You may have more than one instance of each of the Ingestor and Loqueue apps depending your sizing needs. You configure these instance counts as part of the Data Store pane of the tile.
Solution	Increase the number of Diego cells so that your PCF deployment can support the apps pushed as part of the PCF Metrics installation: 1. Navigate to the **Resource Config** section of the Elastic Runtime tile. 2. In the **Diego Cell** row, add another **Instance**.

Nginx Load Balancer

Error	The Smoke tests for Metrics UI errand failed. Or, the **Smoke tests for Metrics UI** checkbox is not selected and installation was successful, but the UI keeps loading and the graphs do not populate with data.
Cause	The Nginx `proxy_buffering` property is on and causes Nginx to block SSE traffic.
	1. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment: `$ cf target -o system -s metrics-v1-3`

Solution	2. Confirm that **Smoke tests for Metrics UI**errand was not run during installation by listing recent logs from the `worker-app-logs` and `worker-app-dev` apps:
	`$ cf logs --recent worker-app-logs`
	`$ cf logs --recent worker-app-dev`
	If neither log contains the text `jobStarted`, then the jobs are not queued because Nginx is blocking SSEs.
	3. Turn off the Nginx `proxy_buffering` property.

Failed querying mysql

Error	`Failed querying mysql`
Cause	The tile deployed without the necessary errands selected to keep the internal database schema in sync with apps.
Solution	Re-deploy the tile with the following errands selected: • **Database migrations for PCF Metrics** • **Push PCF Metrics Data components** • **Push PCF Metrics UI component**

Received no results back from mysql - failing

Error	`Received no results back from mysql - failing`
Cause	The Ingestor is not functioning properly.

Solution	1. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	2. Run `cf apps` to see if these apps are running:
	○ `metrics-ingestor`
	○ `mysql-logqueue`
	3. If the apps are not running, run the following commands to start them:
	`$ cf start metrics-ingestor`
	`$ cf start mysql-logqueue`
	4. Run the following commands and search the app logs for `ERROR` messages containing additional information:
	`$ cf logs metrics-ingestor --recent`
	`$ cf logs mysql-logqueue --recent`
	Note: In some cases, the apps cannot communicate due to TLS certificate verification failure. If your deployment uses self-signed certs, ensure the **Disable SSL certificate verification for this environment** box is selected in the Elastic Runtime **Networking** pane.

Failed to connect to mysql

Error	`Failed to connect to mysql`
Cause	MySQL is not running properly.
	1. Check the logs of the MySQL Server and MySQL Proxy jobs for errors. ○ You can download the logs from the PCF Metrics tile under the**Status** tab.

Solution	2. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	3. Run the following command and ensure the security group can access the MySQL jobs:
	Note: PCF Metrics creates a default security group to allow all traffic to its apps.
	`$ cf security-group metrics-api`

Failed to start elasticsearch client

Error	`Failed to start elasticsearch client`
Cause	Elasticsearch is not running correctly.

Solution	1. Check the logs of the Elasticsearch Master, Elasticsearch Coordinator (for 1.3.4 or lower), and Elasticsearch Data jobs for errors.
	◦ You can download the logs from the PCF Metrics tile under the**Status** tab.
	2. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	3. Run the following command and ensure the security group can access the Elasticsearch jobs:
	Note: PCF Metrics creates a default security group to allow all traffic to its apps.
	`$ cf security-group metrics-api`

Never received app logs

Error	`Never received app logs - something in the firehose -> elasticsearch flow is broken`
Cause	Ingestor is not inserting logs correctly.

Solution	1. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	2. Run `cf apps` to see if these apps are running:
	◦ `metrics-ingestor`
	◦ `elasticsearch-logqueue`
	3. If the apps are not running, run the following commands to start them:
	`$ cf start metrics-ingestor`
	`$ cf start elasticsearch-logqueue`
	4. Run the following commands and search the app logs for `ERROR` messages containing additional information:
	`$ cf logs metrics-ingestor --recent`
	`$ cf logs elasticsearch-logqueue --recent`
	Note: In some cases, you might discover a failure to communicate with Loggregator in the form of a bad handshake error.

© Copyright Pivotal Software Inc, 2013-2019

	Ensure the **Loggregator Port** setting in the Elastic Runtime tile **Networking** pane is set to the correct value. For AWS, it is `4443`. For all other IaaSes, it is `443`.

Metrics and Events not available

Error	`Network metrics are not available.`
	`Container metrics are not available.`
	`App events are not available.`

Cause	PCF Metrics is misconfigured and the front end API does not receive logs from MySQL.

Solution	1. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	2. Run the following command to check the app logs and investigate the error:
	`$ cf logs metrics --recent`

Logs and Histograms not available

Error	`Logs are not available.` `Histograms are not available.`
Cause	PCF Metrics is misconfigured and the front end API does not receive logs from Elasticsearch.

Solution	1. From the cf CLI, target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	2. Run the following command to check the app logs and investigate the error:
	`$ cf logs metrics --recent`

Elasticsearch Instance does not Start

Error	The Deployment fails because an Elasticsearch instance does not start (for 1.3.4 or lower)
Cause	The instance might not start because its configured heap size is greater than that of the VM that hosts it.
Solution	1. From the PCF Metrics tile in Ops Manager, select the **Data Store** settings pane. 2. Record the value in the **Elastic Search Heap Size** field. 3. Select the **Resource Config** pane and ensure the following jobs have RAM greater than or equal to the **Elastic Search Heap Size** ◦ **Elasticsearch Master** ◦ **Elasticsearch Coordinator** ◦ **Elasticsearch Data** 4. If any of the jobs do not have enough memory, do one of the following: ◦ Give the job more RAM ◦ Lower the **Elastic Search Heap Size**

© Copyright Pivotal Software Inc, 2013-2019

No Logs or Metrics in the UI

In some cases, the PCF Metrics UI might not display metrics and logs after successfully deploying.

Follow the steps in this section to help locate the app or component causing the problem.

Step 1: Check your Load Balancer Configuration

If you use a load balancer, the event-stream mechanism used by the Metrics UI might be blocked. See the table below to resolve this error.

If you do not use a load balancer, or this issue does not apply to your deployment, proceed to Step 2: Check the PCF Metrics Apps.

Error	In the case of a customer using an F5 load balancer, metrics and logs were not visible in the UI despite successful ingestion and no UI errors reported.
Cause	The root of the issue was the traffic of type text/event-stream was blocked by the F5 load balancer.
Solution	When F5 was configured to allow event-stream traffic, the issue was resolved.

Step 2: Check the PCF Metrics Apps

1. From Ops Manager, click the Elastic Runtime Tile.
    a. Click the **Credentials** tab.
    b. Under the **UAA** job, next to **Admin Credentials**, click **Link to Credential**.
    c. Record the username and password for use in the next step.
2. Log in to the Cloud Foundry Command Line Interface (cf CLI) using the credentials from the previous step.

    ```
    $ cf login -a https://api.YOUR-SYSTEM-DOMAIN -u admin -p PASSWORD
    ```

3. When prompted, select the `system` org and the `metrics-v1-3` space.
4. Ensure that the output displays the following apps, each in a `started` state:
    - `metrics-ingestor`
    - `mysql-logqueue`
    - `elasticsearch-logqueue`
    - `metrics-aggregator`
    - `metrics`
    - `worker-app-dev`
    - `worker-app-logs`
    - `worker-health-check`
    - `worker-reaper`
5. Check the logs of each app for errors using the following command:

    ```
    $ cf logs APP-NAME --recent
    ```

    If you do not see any output, or if you did not find any errors, proceed to Step 3: Check the Elasticsearch Cluster.

Step 3: Check the Elasticsearch Cluster

1. From Ops Manager, select the PCF Metrics tile.
2. Under the **Status** tab, record the IP of an **Elasticsearch Master** node.
3. Use `bosh ssh` to access the VM from the previous step. For instructions, see Advanced Troubleshooting with the BOSH CLI.
4. Run the following command to list all the Elasticsearch indices:

© Copyright Pivotal Software Inc, 2013-2019

```
$ curl localhost:9200/_cat/indices?v | sort


green  open   app_logs_1477512000   8   1  125459066          0    59.6gb       29.8gb
green  open   app_logs_1477526400   8   1  129356671          0    59.1gb       29.5gb
green  open   app_logs_1478174400   8   1  129747170          0    61.9gb       30.9gb
. . .
green  open   app_logs_1478707200   8   1  128392686          0    63.2gb       31.6gb
green  open   app_logs_1478721600   8   1  102005754          0    53.5gb       26.5gb
health status index              pri rep docs.count docs.deleted store.size pri.store.size
```

a. If the curl does not return a `success` response, Elasticsearch might not even be running correctly. Inspect the following logs for any failures or errors:

- `/var/vcap/sys/log/elasticsearch/elasticsearch.stdout.log`
- `/var/vcap/sys/log/elasticsearch/elasticsearch.stderr.log`

5. Examine the `status` column of the output.

a. If the status of any of the indices is not `green`, restart the Logqueue app:

```
$ cf restart elasticsearch-logqueue
```

b. Run the curl command periodically to see if the indices recover to a `green` status.

6. Run the curl command several more times and examine the most recent index to see if the number of stored documents periodically increases.

> **Note**: The last row of the output corresponds to the most recent index. The sixth column displays the number of documents for the index.

a. If all indices show a `green` status, but the number of documents does not increase, there is likely a problem further up in ingestion. Proceed to to Step 4: Check the Elasticsearch Logqueue.

Step 4: Check the Elasticsearch Logqueue

1. Run `cf apps` to see if the `elasticsearch-logqueue` app instances are `started`.

2. If any instance of the app is `stopped`, run the following command to increase logging:

```
$ cf set-env elasticsearch-logqueue LOG_LEVEL DEBUG
```

a. Run the following command to stream logs:

```
$ cf logs elasticsearch-logqueue
```

b. In a different terminal window, run the following command:

```
$ cf restage elasticsearch-logqueue
```

c. Watch the logs emitted by the `elasticsearch-logqueue` app for errors.

- A common error is that the app cannot connect to Elasticsearch because a user deleted the application security group (ASG) that PCF Metrics creates to allow the Logqueue app to connect to the Elasticsearch VMs. You can run `cf security-group metrics-api` to see if the ASG exists. If not, see Creating Application Security Groups.

3. If the app is started and you do not find any errors, proceed to Step 5: Check the Metrics Ingestor.

Step 5: Check the Metrics Ingestor

1. Run `cf apps` to see if the `metrics-ingestor` app instances are `started`.

2. If any of the app instances are `stopped`, run the following command to increase logging:

```
$ cf set-env metrics-ingestor LOG_LEVEL DEBUG
```

© Copyright Pivotal Software Inc, 2013-2019

a. Run the following command to stream logs:

```
$ cf logs metrics-ingestor
```

b. In a different terminal window, run the following command:

```
$ cf restage metrics-ingestor
```

c. Watch the logs emitted by the `metrics-ingestor` app for errors. See the list below for common errors:

- **Cannot connect to the firehose**: PCF Metrics creates a UAA user to authenticate the connection to the Firehose. This user must have the `doppler.firehose` authority.
- **Cannot connect to the logqueues**: There might be a problem with the UAA, or it could be throttling traffic.
- **WebSocket Disconnects**: If you see Websocket disconnects logs in the Ingestor app, consider adding additional Ingestor instances. The Firehose might be dropping the Ingestor connection to avoid back pressure.

3. If the app is started and you do not find any errors, proceed to Step 6: Check MySQL.

Step 6: Check MySQL

1. From Ops Manager, select the PCF Metrics tile.
2. Under the **Status** tab, record the IP of a **MySQL Server** node.
3. Use `bosh ssh` to access the VM from the previous step. For instructions, see Advanced Troubleshooting with the BOSH CLI.
4. Log in to mysql by running `mysql -u USERNAME -p PASSWORD`

> **Note**: If you do not know the username and password, you can run `cf env mysql-logqueue` with the `system` org and the `metrics-v1-3` space targeted.

5. Verify that the database was bootstrapped correctly:

a. Run `show databases` and check for a `metrics` database.

i. If there is no `metrics` database, the `migrate_db` errand of the BOSH release might not have run or succeeded. Ensure the errand is selected in the tile configuration and update the tile.

6. Run `use metrics` to select the `metrics` database:

```
mysql> use metrics;
```

7. Run `show tables` and ensure you see the following tables:

```
+----------------------------+
| Tables_in_metrics          |
+----------------------------+
| app_event                  |
| app_metric_rollup          |
| container_metric_1477353600 |
| container_metric_1477440000 |
| container_metric_1477526400 |
| container_metric_1477612800 |
| container_metric_1477699200 |
| container_metric_1477785600 |
| container_metric_1477872000 |
| container_metric_1477958400 |
| container_metric_1478044800 |
| container_metric_1478131200 |
| container_metric_1478217600 |
| container_metric_1478304000 |
| container_metric_1478390400 |
| container_metric_1478476800 |
| container_metric_1478563200 |
| container_metric_1478649600 |
| http_start_stop_1477353600  |
| http_start_stop_1477440000  |
| http_start_stop_1477526400  |
| http_start_stop_1477612800  |
| http_start_stop_1477699200  |
| http_start_stop_1477785600  |
| http_start_stop_1477872000  |
| http_start_stop_1477958400  |
| http_start_stop_1478044800  |
| http_start_stop_1478131200  |
| http_start_stop_1478217600  |
| http_start_stop_1478304000  |
| http_start_stop_1478390400  |
| http_start_stop_1478476800  |
| http_start_stop_1478563200  |
| http_start_stop_1478649600  |
| schema_version             |
+----------------------------+
```

8. Enter the following query several times to verify that the value returned does not decrease over time:

```
mysql> select count(*) from metrics.app_metric_rollup where timestamp > ((UNIX_TIMESTAMP() - 60) * POW(10, 3));
```

This command displays the rate at which metrics flow in over the last minute.

   a. If the command returns `0` or a consistently decreasing value, the problem is likely further up in ingestion. Proceed to Step 7: Check the MySQL Logqueue.

Step 7: Check the MySQL Logqueue

1. Run `cf apps` to see if the `mysql-logqueue` app instances are `started`.
2. If any instance of the app is `stopped`, run the following command to increase logging:

```
$ cf set-env mysql-logqueue LOG_LEVEL DEBUG
```

   a. Run the following command to stream logs:

```
$ cf logs mysql-logqueue
```

   b. In a different terminal window, run the following command:

```
$ cf restage mysql-logqueue
```

   c. Watch the logs emitted by the `mysql-logqueue` app for errors.

      - A common error is that the app cannot connect to MySQL because a user deleted the application security group (ASG) that PCF Metrics creates to allow the Logqueue app to connect to the MySQL VMs. You can run `cf security-group metrics-api` to see if the ASG exists. If not, see Creating Application Security Groups.

3. If the app is started and you do not find any errors, proceed to Step 8: Check the Metrics Aggregator.

© Copyright Pivotal Software Inc, 2013-2019

Step 8: Check the Metrics Aggregator

1. Run `cf apps` to see if the `metrics-aggregator` app instances are `started`.
2. If any instance of the app is `stopped`, run the following command to increase logging:

    ```
    $ cf set-env metrics-aggregator LOG_LEVEL DEBUG
    ```

    a. Run the following command to stream logs:

    ```
    $ cf logs metrics-aggregator
    ```

    b. In a different terminal window, run the following command:

    ```
    $ cf restage metrics-aggregator
    ```

    c. Watch the logs emitted by the `metrics-aggregator` app for errors.

    - A common error is that the app cannot connect to MySQL because a user deleted the application security group (ASG) that PCF Metrics creates to allow the aggregator app to connect to the MySQL VMs. You can run `cf security-group metrics-api` to see if the ASG exists. If not, see Creating Application Security Groups.

MySQL Node Failure

In some cases, a MySQL server node might fail to restart. The following two sections describe the known conditions that cause this failure as well as steps for diagnosing and resolving them. If neither of the causes listed apply, the final section provides instructions for re-deploying BOSH as a last resort to resolve the issue.

Cause 1: monit Timed Out

Diagnose

Follow these steps to see if a `monit` time-out caused the MySQL node restart to fail:

1. Use `bosh ssh` to access the failing node, using the IP address in the Ops Manager Director tile **Status** tab. For instructions, see Advanced Troubleshooting with the BOSH CLI.
2. Run `monit summary` and check the status of the `mariadb_ctrl` job.
3. If the status of the `mariadb_ctrl` job is `Execution Failed`, open the following file: `/var/vcap/sys/log/mysql/mariadb_ctrl.combined.log`.

    a. If the last line of the log indicates that MySQL started without issue, such as in the example below, `monit` likely timed out while waiting for the job to report healthy. Follow the steps below to resolve the issue.

    ```
    {"timestamp":"1481149250.288255692","source":"/var/vcap/packages/
    mariadb_ctrl/bin/mariadb_ctrl","message":"/var/vcap/packages/
    mariadb_ctrl/bin/mariadb_ctrl.mariadb_ctrl
     started","log_level":1,"data":{}}
    ```

Resolve

Run the following commands to return the `mariadb_ctrl` job to a healthy state:

1. Run `monit unmonitor mariadb`.
2. Run `monit monitor mariadb`.
3. Run `monit summary` and confirm that the output lists `mariadb_ctrl` as `running`.

Cause 2: Bin Logs Filled up the Disk

Diagnose

1. Use `bosh ssh` to access the failing node. For instructions, see Advanced Troubleshooting with the BOSH CLI.
2. Open the following log file: `/var/vcap/sys/log/mysql/mysql.err.log`.
3. If you see log messages that indicate insufficient disk space, the persistent disk is likely storing too many bin logs. Confirm insufficient disk space by doing the following:
    a. Run `df -h`.
        i. Ensure that you see the `/var/vcap/store` folder is at or over `90%` usage.
    b. Navigate to `/var/vcap/store/mysql` and run `ls -al`.
        i. Ensure that you see many files named with the format `mysql-bin.########`.

In MySQL for PCF, the server node does not make use of these logs and you can remove all except the most recent bin log. Follow the steps below to resolve the issue.

Resolve

1. Log in to mysql by running `mysql -u USERNAME -p PASSWORD`

    Note: If you do not know the username and password, you can run `cf env mysql-logqueue` with the `system` org and the `metrics-v1-3` space targeted.

2. Run `use metrics;`.
3. Run the following command:

```
mysql> PURGE BINARY LOGS BEFORE 'YYYY-MM-DD HH:MM:SS';
```

Re-deploy BOSH to Restart the Node

If troubleshooting based on the causes mentioned above did not resolve the issue with your failing MySQL node, you can follow the steps below to recover it. Pivotal recommends only using this procedure as a if there are no other potential solutions available.

warning: This procedure is extremely costly in terms of time and network resources. The cluster takes a significant amount of time to put the data replicated to the rest of the cluster back into the rebuilt node. This procedure consumes considerable network bandwidth as potentially hundreds of gigabytes of data needs to transfer.

Stop the Ingestor App

1. From Ops Manager, click the Elastic Runtime Tile.
    a. Click the **Credentials** tab.
    b. Under the **UAA** job, next to **Admin Credentials**, click **Link to Credential**.
    c. Record the username and password for use in the next step.
2. Log in to the cf CLI using the credentials from the previous step.

```
$ cf login -a https://api.YOUR-SYSTEM-DOMAIN -u admin -p PASSWORD
```

3. Target the `system` org and `metrics-v1-3` space of your PCF deployment:

```
$ cf target -o system -s metrics-v1-3
```

© Copyright Pivotal Software Inc, 2013-2019

4. Stop data flow into the Galera cluster:

```
$ cf stop metrics-ingestor
```

Edit Your Deployment Manifest

1. Follow the steps in the Log in to BOSH section of the Advanced Troubleshooting with the BOSH CLI topic to target and log in to your BOSH Director. The steps vary slightly depending on whether your PCF deployment uses internal authentication or an external user store.
2. Download the manifest of your PCF deployment:

```
$ bosh download manifest YOUR-PCF-DEPLOYMENT YOUR-PCF-MANIFEST.yml
```

> **Note**: You must know the name of your PCF deployment to download the manifest. To retrieve it, run `bosh deployments` to list your deployments and locate the name of your PCF deployment.

3. Open the manifest and set the number of instances of the failed server node to `0`.
4. Run `bosh deployment YOUR-PCF-MANIFEST.yml` to specify your edited manifest.
5. Run `bosh deploy` to deploy with your manifest.
6. Run `bosh disks --orphaned` to see the persistent disk or disks associated with the failed node.
    a. Record the `CID` of each persistent disk.
    b. Contact Pivotal Support to walk through re-attaching the orphaned disks to new VMs to preserve their data.
7. Open the manifest and set the number of instances of the failed server node to `1`.
8. Run `bosh deploy` to deploy with your edited manifest.
9. Wait for BOSH to rebuild the node.

MySQL SST Disabled Error

If you see the message below on a failing node in `/var/vcap/sys/log/mysql/mysql.err.log`, you can resolve the error by following the instructions in the Interruptor Logs section of the MySQL for PCF documentation.

```
WSREP_SST: [ERROR] ############################################################################## (20160610 04:33:21.338)
WSREP_SST: [ERROR] SST disabled due to danger of data loss. Verify data and bootstrap the cluster (20160610 04:33:21.340)
WSREP_SST: [ERROR] ############################################################################## (20160610 04:33:21.341)
```

Log Errors

Error	The PCF Metrics UI does not show any new logs from Elasticsearch.
Cause	The tile deployed with the `Push PCF Metrics Data Components` errand deselected

Solution	Restart the Elasticsearch Logqueue using the cf CLI as follows:
	1. Target the `system` org and `metrics-v1-3` space of your PCF deployment:
	`$ cf target -o system -s metrics-v1-3`
	2. Run the following command to restart the Logqueue application:
	`$ cf restart elasticsearch-logqueue`
	Note: To avoid having to apply this fix in the future, select the checkbox to enable the `Push PCF Metrics Data Components` errand before your next tile update.

© Copyright Pivotal Software Inc, 2013-2019

503 Errors

Error	You encounter `503` errors when accessing the PCF Metrics UI in your browser.
Cause	Your Elasticsearch nodes might have become unresponsive.
Solution	Check the Elasticsearch index health by following the procedure below, and consider adding additional Elasticsearch nodes. 1. Retrieve the IP address of your Elasticsearch master node by navigating to the **Metrics** tile in the Ops Manager Installation Dashboard, clicking the **Status** tab, and recording the IP address next to **ElasticSearchMaster**.  2. SSH into the Ops Manager VM by following the instructions in SSH into Ops Manager. 3. From the Ops Manager VM, use `curl` to target the IP address of your Elasticsearch master node. Follow the instructions in the Cluster Health topic of the Elasticsearch documentation.

Failed to Fetch Apps

Error	Even though you entered the correct UAA credentials, the metrics app fails to fetch the list of apps.
Cause	The browser plugins or cookies inject extraneous content in requests to Cloud Controller API, causing it to reject the request.
Solution	Confirm the problem and clear the browser, as follows: 1. Try the browser's incognito mode to see if the metrics app is able to fetch the list of apps. If this works, the problem is likely cookies or plugins. 2. Clear your browser cookies and plugins.

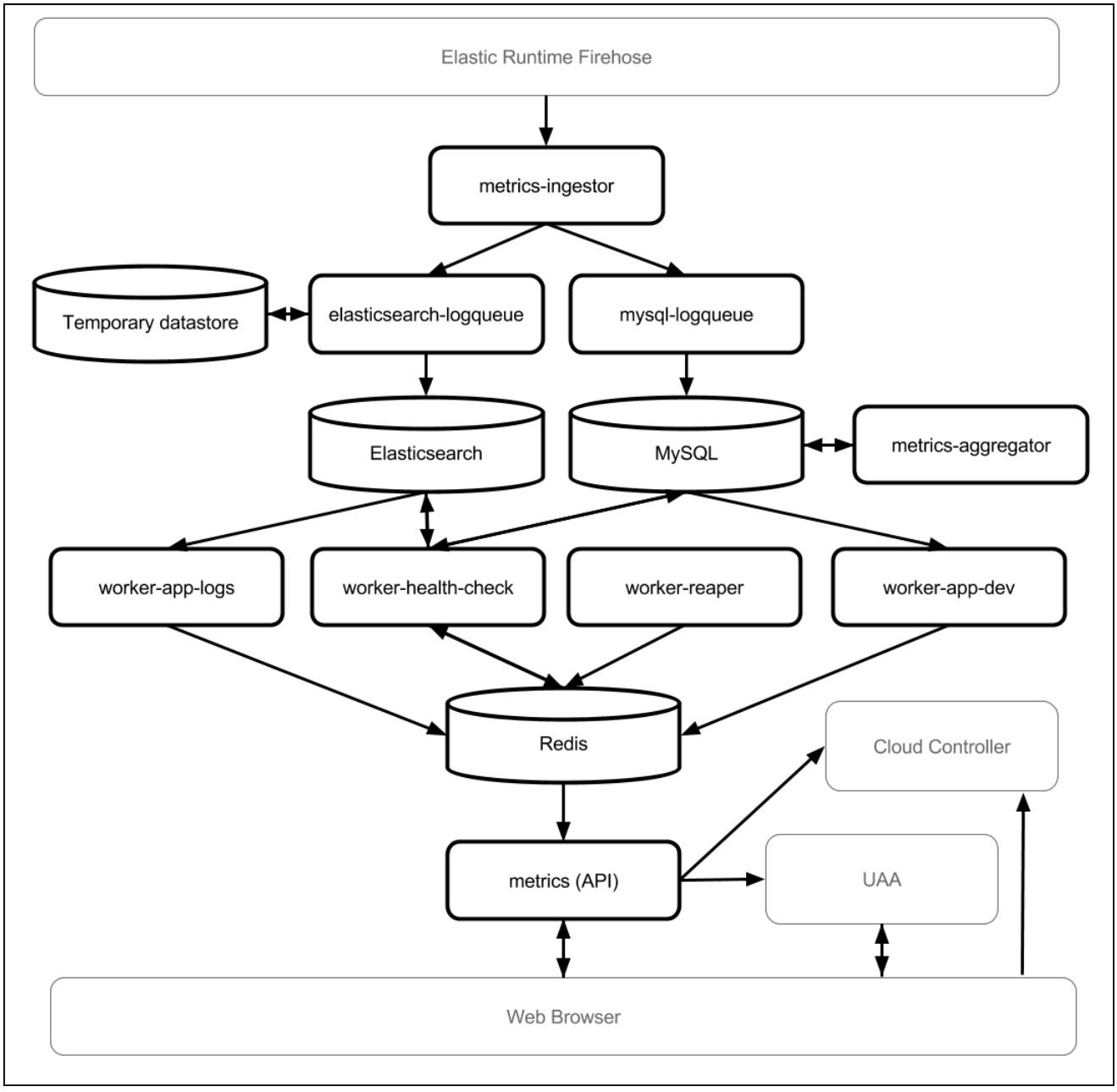
PCF Metrics Product Architecture

This topic describes the product architecture of Pivotal Cloud Foundry (PCF) Metrics.

Overview

The diagram below displays the components of PCF Metrics in bold, as well as the Cloud Foundry components that the PCF Metrics system interacts with.

PCF Metrics deploys several Cloud Foundry apps as part of the install process. These components are the bold rectangles in the diagram. The cylinders represent the data storage components of PCF Metrics.



See the following sections to understand the several processes that happen within the PCF Metrics system.

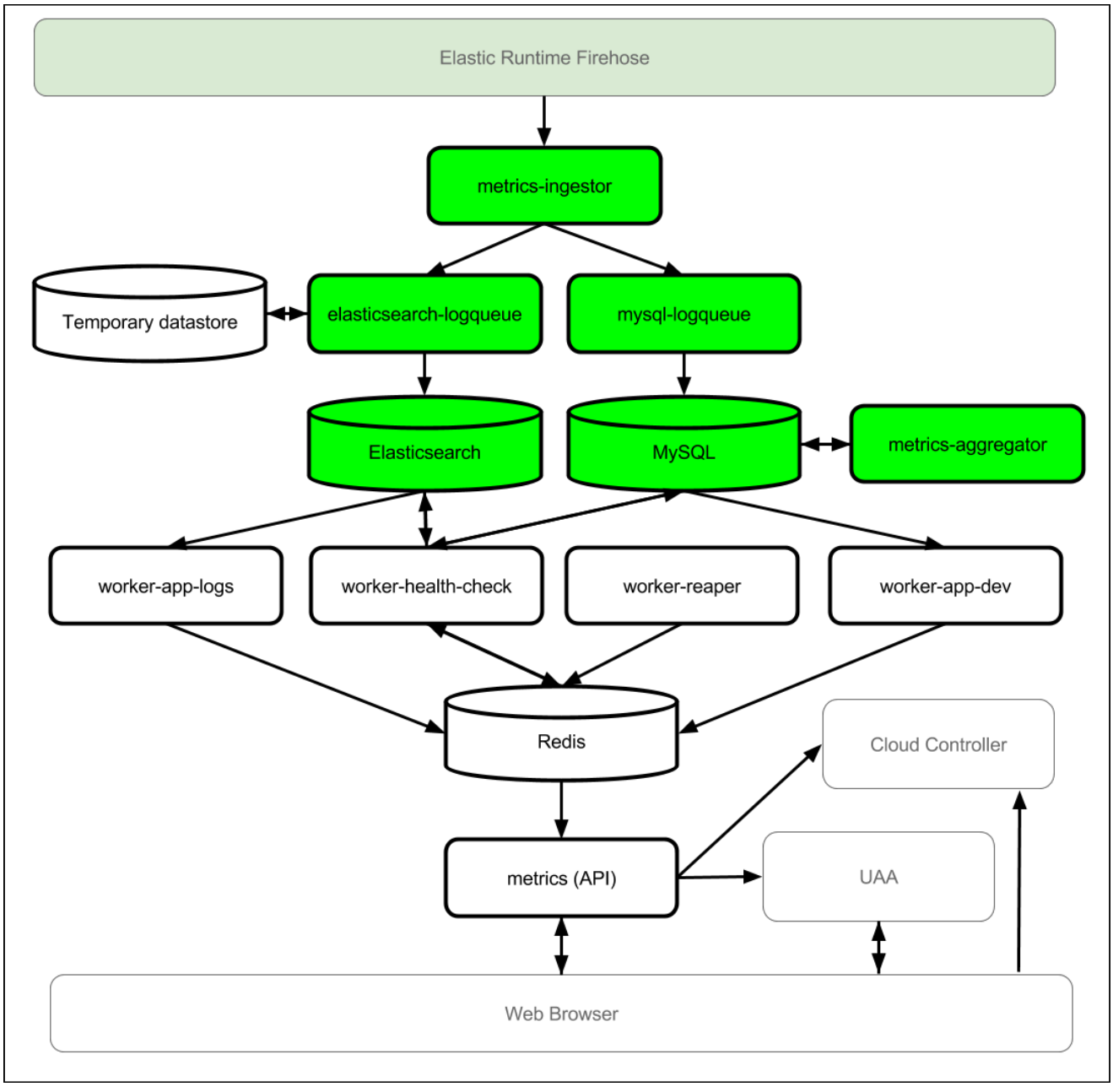
How Data Flows from the Firehose to the Datastores

This section describes how PCF Metrics fills its datastores. PCF Metrics uses two datastores:

- The MySQL component stores metric and event data from the apps running on your PCF deployment.
  - Examples of events are `start` and `stop`.
  - Examples of metrics are *container metrics* such as CPU and *network metrics* such as Requests.
- The Elasticsearch component stores logs from the apps running on your PCF deployment.

Components

The diagram below highlights the components involved in the process of getting metric and log data into the Elasticsearch and MySQL datastore.



Process

The following table describes how the components act during each stage.

Stage	Description
1	The `metrics-ingestor` app does the following: • Receives app logs from the Firehose and forwards them to both the `elasticsearch-logqueue` and `mysql-logqueue` apps

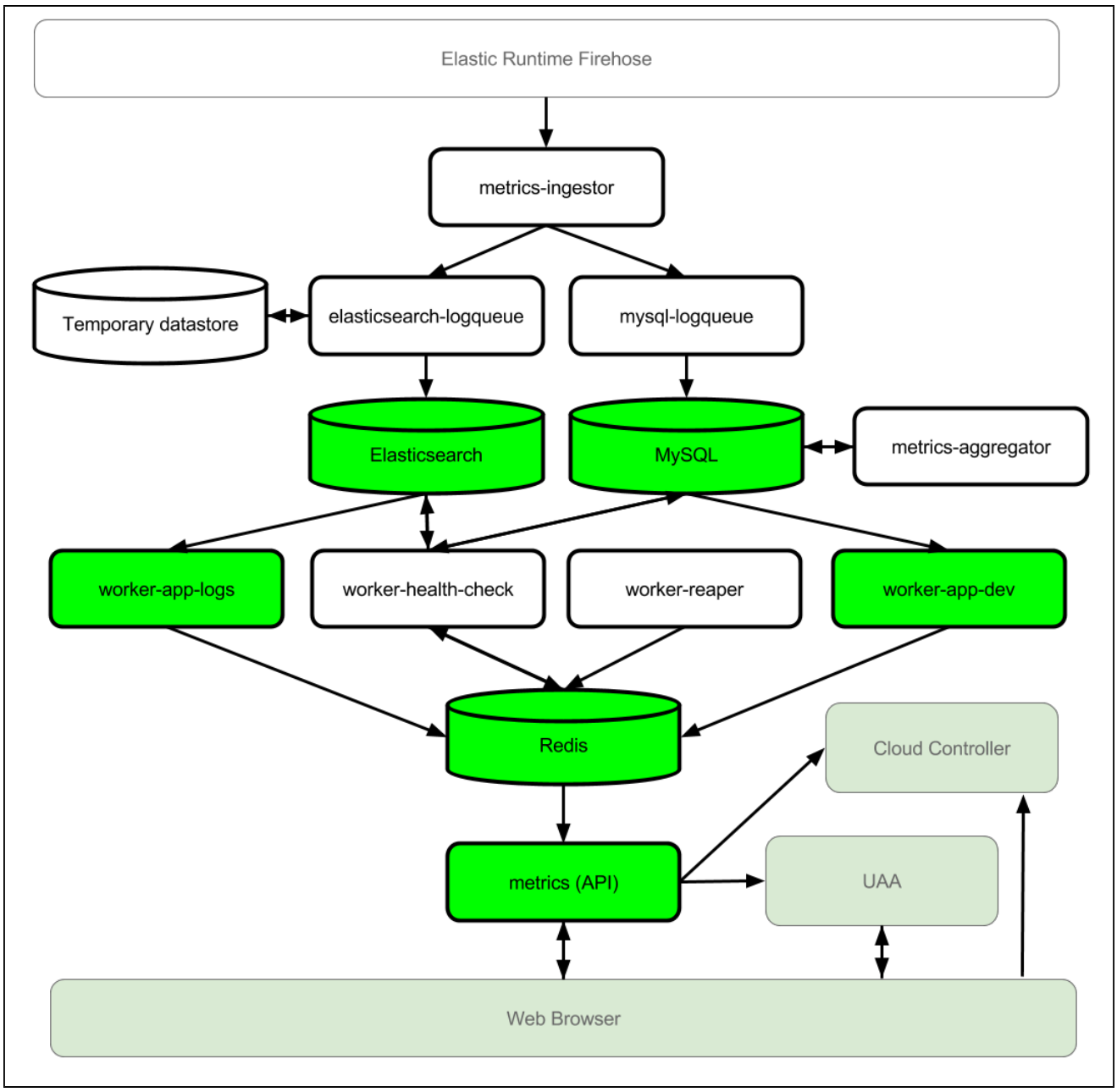
	• Receives container metrics and network metrics (HTTPStartStop events) from the Firehose and forwards them to the `mysql-logqueue` app
2	Each of the logqueues act independently, writing information to the datastores: **Elasticsearch logqueue** The `elasticsearch-logqueue` app checks whether the Elasticsearch datastore is available and does the following: • **If Elasticsearch is available**: Buffers logs and writes them to the Elasticsearch datastore • **If Elasticsearch is NOT available**: 1. Buffers logs and writes them to the Temporary datastore until Elasticsearch becomes available 2. After the Elasticsearch becomes available: ▪ Retrieves logs from the Temporary datastore in groups of 1000, buffers them, and writes them to the Elasticsearch datastore ▪ Continues buffering logs from the ingestor and writing them to the Elasticsearch datastore **MySQL logqueue** The `mysql-logqueue` app buffers logs and writes each data type to MySQL as follows: • **Container metrics**: Inserts messages into the `container_metric` table of MySQL • **Network**: Inserts messages into the `http_start_stop` table of MySQL • **App logs**: Parses log messages for an app event name and inserts the message into the `app_event` table of MySQL
3	The `metrics-aggregator` app, which runs according to an `AGGREGATE_FREQUENCY` property, does the following to aggregate the data stored in MySQL: 1. Retrieves container and network metrics from MySQL 2. Aggregates the data for each app over the last four minutes, grouped by one minute intervals 3. Inserts the aggregated data into the `app_metric_rollup` table of the MySQL component

How the PCF Metrics UI Retrieves Data from the Datastores

This section describes the flow of data through the system when you interact with the PCF Metrics UI.

Components

The diagram below highlights the components involved in this process.



Process

The following table describes how the components act during each stage.

Stage	Description
1	A user launches `metrics.SYSTEM-DOMAIN` in a browser and enters her UAA credentials.
2	After the UAA authorizes the user, the browser does the following: 1. Retrieves through the Cloud Controller API a list of apps that the user can access 2. Displays a page in which the user can select any app returned by the Cloud Controller API
3	A user selects an app from the dropdown menu, which does the following: 1. Opens a Server-Sent Events (SSE) connection to the `metrics` app (metrics API) 2. Sends HTTP Put requests to the metrics API to retrieve metrics and logs for the specified time frame
4	The metrics API receives the requests from the browser and does the following:

© Copyright Pivotal Software Inc, 2013-2019

	1. Communicates with the UAA and Cloud Controller to confirm that the user can access data for the requested app 2. Creates jobs on Redis channels that describe the type of metric, log, or event requested, as well as the time period **Note**: PCF Metrics uses Redis as a pub-sub mechanism between the metrics API and worker apps to marshal metrics and logs.
5	The `worker-app-dev` and `worker-app-logs` apps, which subscribe to the job channels on Redis, recognize the jobs created by the metrics API. The apps remove their corresponding jobs and do the following: 1. Retrieve data from the datastores: a. `worker-app-dev` queries MySQL to retrieve any metrics and events requested for the time period. b. `worker-app-logs` queries Elasticsearch to retrieve the logs for the time period requested. 2. Publish the data to Redis
6	Redis forwards the data to the metrics API.
7	The metrics API streams the data to the browser over SSE, and the PCF Metrics UI displays the data requested by the user.

How Worker Apps Monitor the System

The following table describes the two worker components that PCF Metrics uses to monitor other components in the system.

Worker Component	Function
`worker-health-check`	The health-check worker is an app that does the following every minute: • Checks whether the apps deployed by PCF Metrics can reach the MySQL, Elasticsearch, and Redis datastores • Records the number of MySQL connections and Redis channels
`worker-reaper`	The reaper worker is an app that removes orphaned connections from the `worker-app-dev` and `worker-app-logs` apps to Redis. PCF Metrics requires the reaper worker because Redis does not remove its connections to `worker-app-dev` and `worker-app-logs` if they restart.

© Copyright Pivotal Software Inc, 2013-2019

PCF Metrics Release Notes and Known Issues

This topic contains release notes for Pivotal Cloud Foundry (PCF) Metrics.

v1.3.8

Release Date: August 25, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.8:

- **Bug Fix in Push PCF Metrics Components Errand**: PCF Metrics v1.3.8 fixes a bug in the Push PCF Metrics Components Errand that emitted too many logs and fails if run on an old BOSH Director.

v1.3.7

Release Date: June 19, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.7:

- **Running Errands by Default**: Push apps errand in PCF Metrics tile now defaults to always run, which fixes the bug where stemcell upgrades caused Elasticsearch to go into an unhealthy state.

v1.3.6

Release Date: June 2, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.6:

- **Intermediate Certs**: PCF Metrics v1.3.6 includes a bug fix that allows deployments to use certificates signed by intermediate certificate authorities.
- **Stemcell Bump**: Major stemcell version bump from 3263.x to 3363.x.

Known Issues

See the Known Issues section for the previous release.

v1.3.5

Release Date: May 9, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.5:

- **Reduced Elasticsearch VM Footprint**: PCF Metrics v1.3.5 removes extraneous Elasticsearch VMs, greatly reducing the resource cost of the tile.
- **Simplified Tile Installation**: Several fields in the tile config on Ops Manager have been removed to simplify the tile installation process.
- **Bug Fixes**: Users can now download logs when there is a filter applied.

Known Issues

See the Known Issues section for the previous release.

v1.3.4

Release Date: April 24, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.4:

- **Dependency Graphs and Span ID Filtering**: PCF Metrics v1.3.4 re-enables dependency graphs and span id filtering on the trace explorer page.

  Note: For the dependency graph and span id filtering to work correctly, you must have a version of ERT that is v1.9.16+, v1.10.3+, or v1.11.0+.

- **Compatibility with Azure/OpenStack**: This version of PCF Metrics can be successfully installed on Azure and OpenStack.

Known Issues

See the Known Issues section for the previous release.

v1.3.3

Release Date: April 12, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.3:

- **Internetless Installations**: PCF Metrics v1.3.3 removes multiple unnecessary dependencies that prevented the tile from being installed in an internetless environment.
- **Reduced MySQL Disk Usage**: Raw data in MySQL is now pruned after 2 days, greatly reducing the amount of disk space required to store metrics in MySQL.
- **Metrics Homepage Loads Faster**: Loading apps onto Metrics homepage is now cached. Subsequent loads of homepage should be considerably faster.
- **Bug Fixes:** Fixed stability issues and UI tweaks.

Known Issues

See the Known Issues section for the previous release.

v1.3.0

Release Date: February 23, 2017

Notes

The following list describes what's new in PCF Metrics v1.3.0:

- **Reduced Log Loss During Upgrades**: PCF Metrics uses a temporary datastore during Elasticsearch downtime, including upgrades, to significantly reduce log loss by continuing to store app logs from the Loggregator Firehose. The temporary datastore is a new Redis component deployed with PCF Metrics that operators must size according to the needs of their system. See Configuring the Temporary Datastore for more information.

  Note: PCF Metrics only uses the temporary datastore when upgrading from v1.3 or later.

- **The Trace Explorer**: PCF Metrics provides an interactive graph that allows you to trace requests as they flow through your apps and their endpoints, along with the corresponding logs. See the Trace App Requests section of *Monitoring and Troubleshooting Apps with PCF Metrics*.
- **Improved UI**: PCF Metrics v1.3.0 includes several UI enhancements such as a new time selector and an improved UX for collapsing and expanding which views you are interested in. To view the UI and understand the new functionality, see Monitoring and Troubleshooting Apps with PCF Metrics.
- **Events**: The **Events** graph now includes the following events:
  - **SSH**: This event corresponds to someone successfully using SSH to access a container that runs an instance of the app.
  - **STG Fail**: This event corresponds to your app failing to stage in PCF.

Known Issues

The following sections describe the known issues in PCF Metrics v1.3.0

Compatibility with Elastic Runtime

PCF Metrics v1.3.x requires a version of Elastic Runtime between v1.9.0 and v1.11.x.

Installing Metrics on Azure/OpenStack

PCF Metrics v1.3.0 and v1.3.3 will not install correctly if you have ERT v1.10.x installed on Azure or OpenStack. Upgrade to v1.3.4 if you wish to use PCF Metrics with ERT v1.10.x on either of these.

Metrics and Log Loss when Upgrading from v1.2 to v1.3

The upgrade process from v1.2 to v1.3 acts in the following sequence:

1. Removes the data storage components of v1.2
2. Deploys v1.3 data storage and ingestion components

The upgrade process does not save any v1.2 data and the new components do not begin ingesting and storing log or metrics data until they successfully deploy.

Smoke Test Failure

The PCF Metrics **Smoke Test** errand may fail if your deployment authenticates user sign-ons with an external SAML identity provider or an external LDAP server. In some cases, these external user stores have an additional login procedure that prevents the errand from authenticating with the deployment and validating against the Metrics API.

If you experience this issue, disable the **Smoke Test** errand in the PCF Metrics tile and click **Apply Changes** to run the install again.

See the Configure Authentication and Enterprise SSO section of the *Configuring Elastic Runtime* topic for more information on what configurations can lead to this failure.

© Copyright Pivotal Software Inc, 2013-2019

For Operators who Deploy PCF Metrics using BOSH

If both of the following are true, you may experience issues while using PCF Metrics:

- You deploy PCF Metrics using BOSH instead of using the PCF Metrics tile in Ops Manager.
- You use self-signed certificates.

Pivotal recommends using certificates issued by a Certificate Authority for BOSH deployments of this product.

Past Minor v1.2.x

Release Notes for v1.1.x releases can be found here.

Past Minor v1.1.x

Release Notes for v1.1.x releases can be found here.

Past Minor v1.0.x

Release Notes for v1.0.x releases can be found here.

© Copyright Pivotal Software Inc, 2013-2019