Red Hat Virtualization 4.2

Metrics Store User Guide

Using Metrics Store with Red Hat Virtualization

Last Updated: 2018-10-15

Red Hat Virtualization 4.2 Metrics Store User Guide

Using Metrics Store with Red Hat Virtualization

Red Hat Virtualization Documentation Team
Red Hat Customer Content Services
rhev-docs@redhat.com

Legal Notice

Abstract

A comprehensive guide to understanding the metrics and logs collected by Metrics Store.

Table of Contents

CHAPTER 1. INTRODUCTION

The Metrics Store collects logs and metrics from Red Hat Virtualization. The data is transferred from Red Hat Virtualization to OpenShift where it is stored and aggregated in Elasticsearch and saved in indexes. Elasticsearch is a distributed, RESTful search and analytics engine that lets you perform and combine many types of searches.

Use Kibana to search, view, and interact with real-time data stored in Elasticsearch indexes. Kibana is an open source analytics and visualization platform designed to work with Elasticsearch. You can easily perform advanced data analysis and visualize your data in a variety of charts and tables.

Kibana enables you to view and preempt critical problems, track system usage and resources, and even plan for future growth.

1.1. ACCESSING KIBANA

1. Access Kibana at https://kibana.*<FQDN>*.

2. Log in by entering your username and password.

CHAPTER 2. ANALYZING METRICS

Kibana offers two ways of analyzing metrics:

- Build your own visualizations such as charts, graphs, and tables.
- Load and use predefined sets of visualizations

Red Hat suggests that you start off by using the predefined visualizations. Each set is known as a *dashboard*. Dashboards have the advantage of enabling you to quickly access a wide range of metrics while offering the flexibility of changing them to match your individual needs.

2.1. USING DASHBOARDS

A dashboard displays a set of saved visualizations. Dashboards have the advantage of enabling you to quickly access a wide range of metrics while offering the flexibility of changing them to match your individual needs.

You can use the **Dashboard** tab to create your own dashboards. Alternatively, Red Hat provides the following dashboard examples, which you can import into Kibana and use as is or customize to suit your specific needs:

- System dashboard
- Hosts dashboard
- VMs dashboard

Importing Dashboard Examples

1. Copy the `/etc/ovirt-engine-metrics/dashboards-examples` directory from the Manager virtual machine to your local machine.
2. Open Kibana and click the **Settings** tab.
3. Click the **Indices** tab.
4. Click the **Objects** tab.
5. Click **Import** and import **Searches** from your local copy of `/etc/ovirt-engine-metrics/dashboards-examples`.
6. Click **Import** and import **Visualizations**.

NOTE

If you see an error message while importing the visualizations, check your hosts to ensure that Collectd and Fluentd are running without errors.

7. Click **Import** and import **Dashboards**.
8. Select **project.ovirt-metrics-<ovirt-env-name>.<uuid>** in the **Index Patterns** pane and click the **Refresh field list**  button.

9. Select the **project.ovirt-logs-<ovirt-env-name>.<uuid>** index and click **`Refresh field list`**.
   The imported dashboards are now stored in the system.

Loading Saved Dashboards

Once you have created and saved a dashboard, or imported Red Hat's sample dashboards, you can display them in the **Dashboard** tab:

1. Click the **Dashboard** tab.
2. Click the **Load Saved Dashboard**  button to display a list of saved dashboards.
3. Click a saved dashboard to load it.

2.2. CREATING A NEW VISUALIZATION

Use the **Visualize** page to design data visualizations based on the metrics or log data collected by Metrics Store. You can save these visualizations, use them individually, or combine visualizations into a dashboard. A visualization can be based on one of the following data source types:

- A new interactive search
- A saved search
- An existing saved visualization

Visualizations are based on Elasticsearch's aggregation feature.

Creating a New Visualization

Kibana guides you through the creation process with the help of a visualization wizard.

1. To start the new visualization wizard, click the **Visualize** tab.
2. In step 1, **Create a new visualization table**, select the type of visualization you want to create.
3. In step 2, **Select a search source**, select whether you want to create a new search or reuse a saved search:
   - To create a new search, select **From a new search** and enter the indexes to use as the source. Use *project.ovirt-logs* prefix for log data or *project.ovirt-metrics* prefix for metric data.
   - To create a visualization from a saved search, select **From a saved search** and enter the name of the search.
     The visualization editor appears.

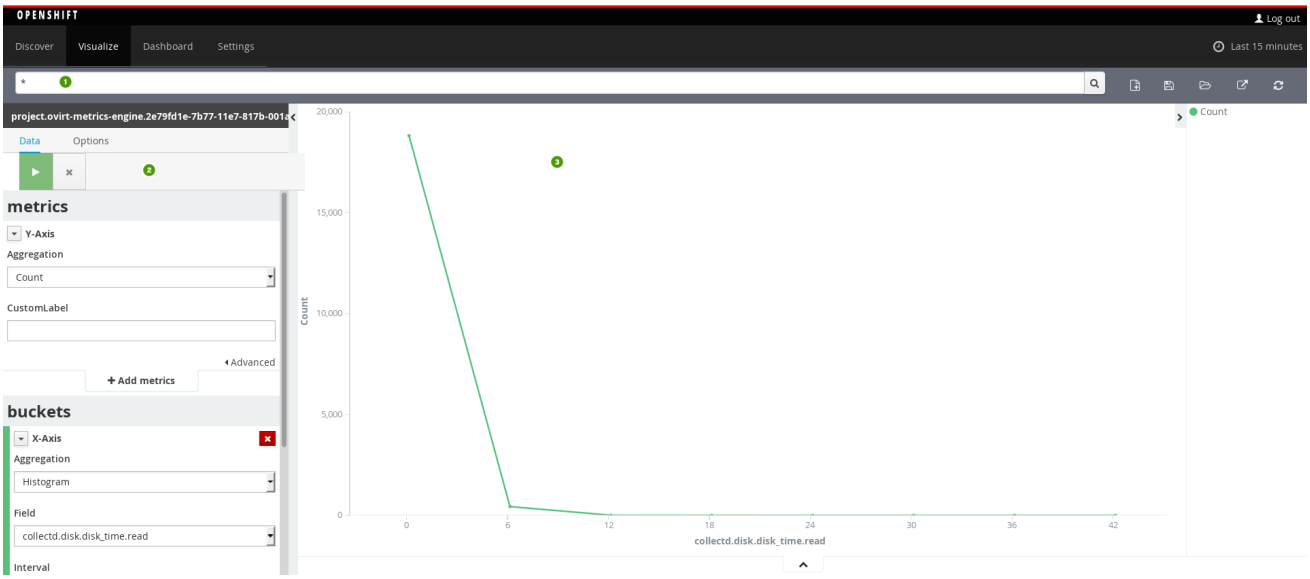
2.3. GRAPHIC USER INTERFACE ELEMENTS

The visualization editor consists of three main areas:

-  The toolbar
-  The aggregation builder

- 3 The preview pane

Visualization Editor



2.4. USING THE VISUALIZATION EDITOR

Use the visualization editor to create visualizations by:

- Submitting search queries from the toolbar
- Selecting metrics and aggregations from the aggregation builder

2.4.1. Submitting Search Queries

Use the toolbar to perform search queries based on the Lucene query parser syntax. For a detailed explanation of this syntax, see Apache Lucene - Query Parser Syntax.

2.4.2. Selecting Metrics and Aggregations

Use the aggregation builder to define which metrics to display, how to aggregate the data, and how to group the results.

The aggregation builder performs two types of aggregations, metric and bucket, which differ depending on the type of visualization you are creating:

- Bar, line, or area chart visualizations use **metrics** for the y-axis and **buckets** for the x-axis, segment bar colors, and row/column splits.
- Pie charts use **metrics** for the slice size and **buckets** to define the number of slices.

To define a visualization from the aggregation bar:

1. Select the metric aggregation for your visualization's y-axis from the **Aggregation** drop-down list in the **metrics** section, for example, count, average, sum, min, max, or unique count. For more information about how these aggregations are calculated, see Metrics Aggregation in the Elasticsearch Reference documentation.

2. Use the **buckets** area to select the aggregations for the visualization's x-axis, color slices, and row/column splits:

   a. Use the **Aggregation** drop-down list to define how to aggregate the bucket. Common bucket aggregations include date histogram, range, terms, filters, and significant terms.
   The order in which you define the buckets determines the order in which they will be executed, so the first aggregation determines the data set for any subsequent aggregations.
   For more information, see Aggregation Builder in the Kibana documentation.

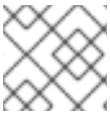
   b. Select the metric you want to display from the **Field** drop-down list. For details about each of the available metrics, see Metrics Schema.

   c. Select the required interval from the **Interval** field.

3. Click **Apply Changes**  .

2.5. METRICS SCHEMA

The following sections describe the metrics that are available from the **Field** menu when creating visualizations.



NOTE

All metric values are collected at 10 second intervals.

2.5.1. Aggregation Metrics

The Aggregation metric aggregates several values into one using aggregation functions such as sum, average, min, and max. It is used to provide a combined value for average and total CPU statistics.

The following table describes the aggregation metrics reported by the **Aggregation** plugin.

Metric Name	collectd.type_instance	Description
collectd.aggregation.percent	• interrupt • user • wait • nice • softirq • system • idle • steal	The average and total CPU usage, as an aggregated percentage, for each of the *collectd.type_instance* states.

Additional Values

- **collectd.plugin:** Aggregation

- **collectd.type_instance:** cpu-average / cpu-sum
- **collectd.plugin_instance:**
- **collectd.type:** percent
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.2. CPU Metrics

CPU metrics display the amount of time spent by the hosts' CPUs, as a percentage.

The following table describes CPU metrics as reported by the **CPU** plugin.

Table 2.1. CPU Metrics

Metric Name	collectd.type_instance	Description
collectd.cpu.percent	• interrupt • user • wait • nice • softirq • system • idle • steal	The percentage of time spent, per CPU, in the *collectd.type_instance* states.

Additional Values

- **collectd.plugin:** CPU
- **collectd.plugin_instance:** *The CPU's number*
- **collectd.type:** percent
- **ovirt.entity:** host

- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.3. CPU Load Average Metrics

CPU load represents CPU contention, that is, the average number of schedulable processes at any given time. This is reported as an average value for all CPU cores on the host. Each CPU core can only execute one process at a time. Therefore, a CPU load average above 1.0 indicates that the CPUs have more work than they can perform, and the system is overloaded.

CPU load is reported over short term (last one minute), medium term (last five minutes) and long term (last fifteen minutes). While it is normal for a host's short term load average to exceed 1.0 (for a single CPU), sustained load average above 1.0 on a host may indicate a problem.

On multi-processor systems, the load is relative to the number of processor cores available. The "100% utilization" mark is 1.00 on a single-core, 2.00 on a dual-core, 4.00 on a quad-core system.

Red Hat recommends looking at CPU load in conjunction with CPU Metrics.

The following table describes the CPU load metrics reported by the **Load** plugin.

Table 2.2. CPU Load Average Metrics

Metric Name	Description
collectd.load.load.longterm	Average number of schedulable processes per CPU core over the last 15 minutes. A value above 1.0 indicates the system was overloaded during the last 15 minutes.
collectd.load.load.midterm	Average number of schedulable processes per CPU core over the last five minutes. A value above 1.0 indicates the system was overloaded during the last 5 minutes.
collectd.load.load.shortterm	Average number of schedulable processes per CPU core over the last one minute. A value above 1.0 indicates the system was overloaded during the last minute.

Additional Values

- **collectd.plugin:** Load
- **collectd.type:** load

- **collectd.type_instance:** None
- **collectd.plugin_instance:** None
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.4. Disk Consumption Metrics

Disk consumption (DF) metrics enable you to monitor metrics about disk consumption, such as the used, reserved, and free space for each mounted file system.

The following table describes the disk consumption metrics reported by the **DF** plugin.

Metric Name	Description
collectd.df.df_complex	The amount of free, used, and reserved disk space, in bytes, on this file system.
collectd.df.percent_bytes	The amount of free, used, and reserved disk space, as a percentage of total disk space, on this file system.

Additional Values

- **collectd.plugin:** DF
- **collectd.type_instance:** free, used, reserved
- **collectd.plugin_instance:** *A mounted partition*
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.5. Disk Operation Metrics

Disk operation metrics are reported per physical disk on the host, and per partition.

The following table describes the disk operation metrics reported by the **Disk** plugin.

Table 2.3. Disk Operation Metrics

Metric Name	Description	collectd.dstypes
collectd.disk.disk_ops.read	The number of disk read operations.	Derive
collectd.disk.disk_ops.write	The number of disk write operations.	Derive
collectd.disk.disk_merged.read	The number of disk reads that have been merged into single physical disk access operations. In other words, this metric measures the number of instances in which one physical disk access served multiple disk reads. The higher the number, the better.	Derive
collectd.disk.disk_merged.write	The number of disk writes that were merged into single physical disk access operations. In other words, this metric measures the number of instances in which one physical disk access served multiple write operations. The higher the number, the better.	Derive
collectd.disk.disk_time.read	The average amount of time it took to do a read operation, in milliseconds.	Derive
collectd.disk.disk_time.write	The average amount of time it took to do a write operation, in milliseconds.	Derive
collectd.disk.pending_operations	The queue size of pending I/O operations.	Gauge
collectd.disk.disk_io_time.io_time	The time spent doing I/Os in milliseconds. This can be used as a device load percentage, where a value of 1 second of time spent represents a 100% load.	Derive

Metric Name	Description	collectd.dstypes
collectd.disk.disk_io_time.weighted_io_time	A measure of both I/O completion time and the backlog that may be accumulating.	Derive

Additional Values

- **collectd.plugin:** Disk
- **collectd.type_instance:** None
- **collectd.plugin_instance:** *The disk's name*
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10

2.5.6. Entropy Metrics

Entropy metrics display the available entropy pool size on the host. Entropy is important for generating random numbers, which are used for encryption, authorization, and similar tasks.

The following table describes the entropy metrics reported by the **Entropy** plugin.

Table 2.4. Entropy Metrics

Metric Name	Description
collectd.entropy.entropy	The entropy pool size, in bits, on the host.

Additional Values

- **collectd.plugin:** Entropy
- **collectd.type_instance:** None
- **collectd.plugin_instance:** None
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*

- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.7. Network Interface Metrics

The following types of metrics are reported from physical and virtual network interfaces on the host:

- Bytes (octets) transmitted and received (total, or per second)
- Packets transmitted and received (total, or per second)
- Interface errors (total, or per second)

The following table describes the network interface metrics reported by the **Interface** plugin.

Table 2.5. Network Interface Metrics

collectd.type	Metric Name	Description
if_octets	collectd.interface.if_octets.rx	A count of the bytes received by the interface. You can view this metric as a Rate/sec or a cumulative count (Max): * Rate/sec: Provides the current traffic level on the interface in bytes/sec. * Max: Provides the cumulative count of bytes received. Note that since this metric is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.

collectd.type	Metric Name	Description
if_octets	collectd.interface.if_octets.tx	A count of the bytes transmitted by the interface. You can view this metric as a Rate/sec or a cumulative count (Max): * Rate/sec: Provides the current traffic level on the interface in bytes/sec. * Max: Provides the cumulative count of bytes transmitted. Note that since this metric is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.
if_packets	collectd.interface.if_packets.rx	A count of the packets received by the interface. You can view this metric as a Rate/sec or a cumulative count (Max): * Rate/sec: Provides the current traffic level on the interface in bytes/sec. * Max: Provides the cumulative count of packets received. Note that since this metric is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.

collectd.type	Metric Name	Description
if_packets	collectd.interface.if_packets.tx	A count of the packets transmitted by the interface. You can view this metric as a Rate/sec or a cumulative count (Max): * Rate/sec: Provides the current traffic level on the interface in packets/sec. * Max: Provides the cumulative count of packets transmitted. Note that since this metric is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.
if_errors	collectd.interface.if_errors.rx	A count of errors received on the interface. You can view this metric as a Rate/sec or a cumulative count (Max). * Rate/sec rollup provides the current rate of errors received on the interface in errors/sec. * Max rollup provides the total number of errors received since the beginning. Note that since this is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.

collectd.type	Metric Name	Description
if_errors	collectd.interface.if_errors.tx	A count of errors transmitted on the interface. You can view this metric as a Rate/sec or a cumulative count (Max). * Rate/sec rollup provides the current rate of errors transmitted on the interface in errors/sec. * Max rollup provides the total number of errors transmitted since the beginning. Note that since this is a cumulative counter, its value will periodically restart from zero when the maximum possible value of the counter is exceeded.
if_dropped	collectd.interface.if_dropped.rx	
if_dropped	collectd.interface.if_dropped.tx	

Additional Values

- **collectd.plugin:** Interface
- **collectd.type_instance:** None
- **collectd.plugin_instance:** *The network's name*
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Derive

2.5.8. Memory Metrics

Metrics collected about memory usage.

The following table describes the memory usage metrics reported by the **Memory** plugin.

Table 2.6. Memory Metrics

Metric Name	collectd.type	collectd.type_instance	Description
collectd.memory.memory	memory	used	The total amount of memory used.
		free	The total amount of unused memory.
		cached	The amount of memory used for caching disk data for reads, memory-mapped files, or tmpfs data.
		buffered	The amount of memory used for buffering, mostly for I/O operations.
		slab_recl	The amount of reclaimable memory used for slab kernel allocations.
		slab_unrecl	Amount of unreclaimable memory used for slab kernel allocations.
collectd.memory.percent	percent	used	The total amount of memory used, as a percentage.
		free	The total amount of unused memory, as a percentage.
		cached	The amount of memory used for caching disk data for reads, memory-mapped files, or tmpfs data, as a percentage.
		buffered	The amount of memory used for buffering I/O operations, as a percentage.

Metric Name	collectd.type	collectd.type_instance	Description
		slab_recl	The amount of reclaimable memory used for slab kernel allocations, as a percentage.
		slab_unrecl	The amount of unreclaimable memory used for slab kernel allocations, as a percentage.

Additional Values

- **collectd.plugin:** Memory
- **collectd.plugin_instance:** None
- **ovirt.entity:** Host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.9. NFS Metrics

NFS metrics enable you to analyze the use of NFS procedures.

The following table describes the NFS metrics reported by the **NFS** plugin.

Metric Name	collectd.type_instance	Description

Metric Name	collectd.type_instance			Description
collectd.nfs.nfs_procedure	null / getattr / lookup / access / readlink / read / write / create / mkdir / symlink / mknod / rename / readdir / remove / link / fsstat / fsinfo / readdirplus / pathconf / rmdir / commit / compound / reserved / access / close / delegpurge / putfh / putpubfh putrootfh / renew / restorefh / savefh / secinfo	/ setattr / setclientid / setcltid_confirm / verify / open / openattr / open_confirm / exchange_id / create_session / destroy_session / bind_conn_to_session / delegreturn / getattr / getfh / lock / lockt / locku / lookupp / open_downgrade / nverify	/ release_lockowner / backchannel_ctl / free_stateid / get_dir_delegation / getdeviceinfo / getdevicelist / layoutcommit / layoutget / layoutreturn / secinfo_no_name / sequence / set_ssv / test_stateid / want_delegation / destroy_clientid / reclaim_complete	The number of processes per *collectd.type_instance* state.

Additional Values

- **collectd.plugin:** NFS
- **collectd.plugin_instance:** *File system + server or client (for example: v3client)*
- **collectd.type:** nfs_procedure
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Derive

2.5.10. PostgreSQL Metrics

PostgreSQL data collected by executing SQL statements on a PostgreSQL database.

The following table describes the PostgreSQL metrics reported by the **PostgreSQL** plugin.

Table 2.7. PostgreSQL Metrics

Metric Name	collectd.type_instance	Description
collectd.postgresql.pg_numbackends	N/A	How many server processes this database is using.
collectd.postgresql.pg_n_tup_g	live	The number of live rows in the database.
	dead	The number of dead rows in the database. Rows that are deleted or obsoleted by an update are not physically removed from their table; they remain present as dead rows until a VACUUM is performed.
collectd.postgresql.pg_n_tup_c	del	The number of delete operations.
	upd	The number of update operations.
	hot_upd	The number of update operations that have been performed without requiring an index update.
	ins	The number of insert operations.
collectd.postgresql.pg_xact	num_deadlocks	The number of deadlocks that have been detected by the database. Deadlocks are caused by two or more competing actions that are unable to finish because each is waiting for the other's resources to be unlocked.
collectd.postgresql.pg_db_size	N/A	The size of the database on disk, in bytes.
collectd.postgresql.pg_blks	heap_read	How many disk blocks have been read.
	heap_hit	How many read operations were served from the buffer in memory, so that a disk read was not necessary. This only includes hits in the PostgreSQL buffer cache, not the operating system's file system cache.
	idx_read	How many disk blocks have been read by index access operations.

Metric Name	collectd.type_instance	Description
	idx_hit	How many index access operations have been served from the buffer in memory.
	toast_read	How many disk blocks have been read on TOAST tables.
	toast_hit	How many TOAST table reads have been served from buffer in memory.
	tidx_read	How many disk blocks have been read by index access operations on TOAST tables.

Additional Values

- **collectd.plugin:** Postgresql
- **collectd.plugin_instance:** *Database's Name*
- **ovirt.entity:** engine
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10
- **collectd.dstypes:** Gauge

2.5.11. Process Metrics

The following table describes the process metrics reported by the **Processes** plugin.

Metric Name	collectd.type	collectd.dstypes
collectd.processes.ps_state	ps_state	Gauge
collectd.processes.ps_disk_ops.read	ps_disk_ops	Derive
collectd.processes.ps_disk_ops.write	ps_disk_ops	Derive

Metric Name	collectd.type	collectd.dstypes
collectd.processes.ps_vm	ps_vm	Gauge
collectd.processes.ps_rss	ps_rss	Gauge
collectd.processes.ps_data	ps_data	Gauge
collectd.processes.ps_code	ps_code	Gauge
collectd.processes.ps_stacksize	ps_stacksize	Gauge
collectd.processes.ps_cputime.syst	ps_cputime	Derive
collectd.processes.ps_cputime.user	ps_cputime	Derive
collectd.processes.ps_count.processes	ps_count	Gauge
collectd.processes.ps_count.threads	ps_count	Gauge
collectd.processes.ps_pagefaults.majfltadd	ps_pagefaults	Derive
collectd.processes.ps_pagefaults.minflt	ps_pagefaults	Derive
collectd.processes.ps_disk_octets.write	ps_disk_octets	Derive
collectd.processes.ps_disk_octets.read	ps_disk_octets	Derive
collectd.processes.fork_rate	fork_rate	Derive

Additional Values

- **collectd.plugin:** Processes
- **collectd.plugin_instance:** _The process's name (except for collectd.processes.fork_rate=N/A) collectd.type_instance:* N/A (except for collectd.processes.ps_state=running/ zombies/ stopped/ paging/ blocked/ sleeping)
- **ovirt.entity:** host
- **ovirt.cluster.name.raw:** *The cluster's name*

- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10

2.5.12. Swap Metrics

Swap metrics enable you to view the amount of memory currently written onto the hard disk, in bytes, according to available, used, and cached swap space.

The following table describes the Swap metrics reported by the **Swap** plugin.

Table 2.8. Swap Metrics

Metric Name	collectd.type	collectd.type_instance	collectd.dstypes	Description
collectd.swap.swap	swap	used / free / cached	Gauge	The used, available, and cached swap space (in bytes).
collectd.swap.swap_io	swap_io	in / out	Derive	The number of swap pages written and read per second.
collectd.swap.percent	percent	used / free / cached	Gauge	The percentage of used, available, and cached swap space.

Additional Fields

- **collectd.plugin:** Swap
- **collectd.plugin_instance:** None
- **ovirt.entity:** host or Manager
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10

2.5.13. Virtual Machine Metrics

The following table describes the virtual machine metrics reported by the **Virt** plugin.

Metric Name	collectd.type	collectd.type_instance	collectd.dstypes
collectd.virt.ps_cputime.syst	ps_cputime.syst	N/A	Derive
collectd.virt.percent	percent	virt_cpu_total	Gauge
collectd.virt.ps_cputime.user	ps_cputime.user	N/A	Derive
collectd.virt.virt_cpu_total	virt_cpu_total	CPU number	Derive
collectd.virt.virt_vcpu	virt_vcpu	CPU number	Derive
collectd.virt.disk_octets.read	disk_octets.read	disk name	Gauge
collectd.virt.disk_ops.read	disk_ops.read	disk name	Gauge
collectd.virt.disk_octets.write	disk_octets.write	disk name	Gauge
collectd.virt.disk_ops.write	disk_ops.write	disk name	Gauge
collectd.virt.if_octets.rx	if_octets.rx	network name	Derive
collectd.virt.if_dropped.rx	if_dropped.rx	network name	Derive
collectd.virt.if_errors.rx	if_errors.rx	network name	Derive
collectd.virt.if_octets.tx	if_octets.tx	network name	Derive
collectd.virt.if_dropped.tx	if_dropped.tx	network name	Derive
collectd.virt.if_errors.tx	if_errors.tx	network name	Derive
collectd.virt.if_packets.rx	if_packets.rx	network name	Derive
collectd.virt.if_packets.tx	if_packets.tx	network name	Derive

Metric Name	collectd.type	collectd.type_instance	collectd.dstypes
collectd.virt.memory	memory	rss / total /actual_balloon / available / unused / usable / last_update / major_fault / minor_fault / swap_in / swap_out	Gauge
collectd.virt.total_requests	total_requests	flush-DISK	Derive
collectd.virt.total_time_in_ms	total_time_in_ms	flush-DISK	Derive
collectd.virt.total_time_in_ms	total_time_in_ms	flush-DISK	Derive

Additional Values

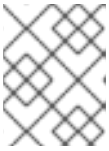
- **collectd.plugin:** virt
- **collectd.plugin_instance:** The virtual machine's name
- **ovirt.entity:** vm
- **ovirt.cluster.name.raw:** *The cluster's name*
- **ovirt.engine_fqdn.raw:** *The Manager's FQDN*
- **hostname:** *The host's FQDN*
- **ipaddr4:** *IP address*
- **interval:** 10

2.5.14. Gauge and Derive Data Source Types

Each metric includes a *collectd.dstypes* value that defines the data source's type:

- **Gauge**: A gauge value is simply stored as-is and is used for values that may increase or decrease, such as the amount of memory used.
- **Derive**: These data sources assume that the change of the value is interesting, i.e., the derivative. Such data sources are very common for events that can be counted, for example the number of disk read operations. The total number of disk read operations is not interesting, but rather the change since the value was last read. The value is therefore converted to a rate using the following formula:

```
rate = value(new)-value(old)\
       time(new)-time(old)
```



NOTE

If value(new) is less than value (old), the resulting rate will be negative. If the minimum value to zero, such data points will be discarded.

2.6. WORKING WITH METRICS STORE INDEXES

Metrics Store creates the following two indexes per day:

- project.ovirt-metrics-<ovirt-env-name>.uuid.yyyy.mm.dd
- project.ovirt-logs-<ovirt-env-name>.uuid.yyyy.mm.dd

When using the **Discover** page, select the index named project.ovirt-logs-<ovirt-env-name>.uuid.

In the **Visualization** page select project.ovirt-metrics-<ovirt-env-name>.uuid for metrics data or project.ovirt-logs-<ovirt-env-name>.uuid for log data.

CHAPTER 3. ANALYZING LOGS

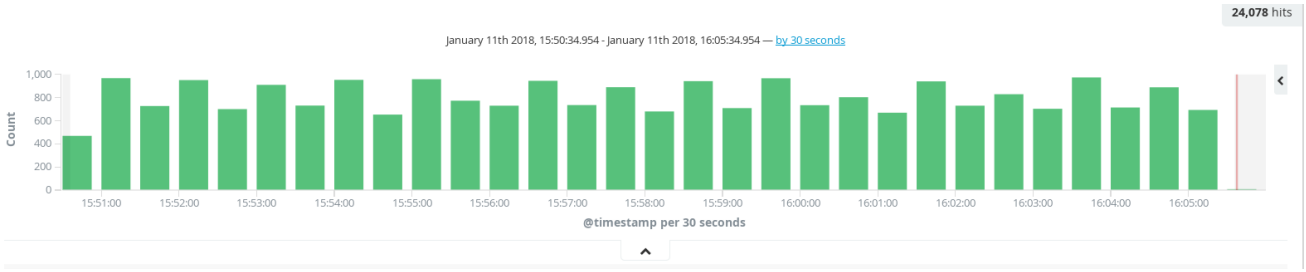
Use the **Discover** page to interactively explore the data collected from Red Hat Virtualization. Each set of results that is collected is referred to as a *document*. Documents are collected from the following log files:

- **engine.log** contains all Red Hat Virtualization Manager UI crashes, Active Directory lookups, database issues, and other events.
- **vdsm.log** is the log file for VDSM, the Manager's agent on the virtualization host(s), and contains host-related events.

3.1. GRAPHIC USER INTERFACE ELEMENTS

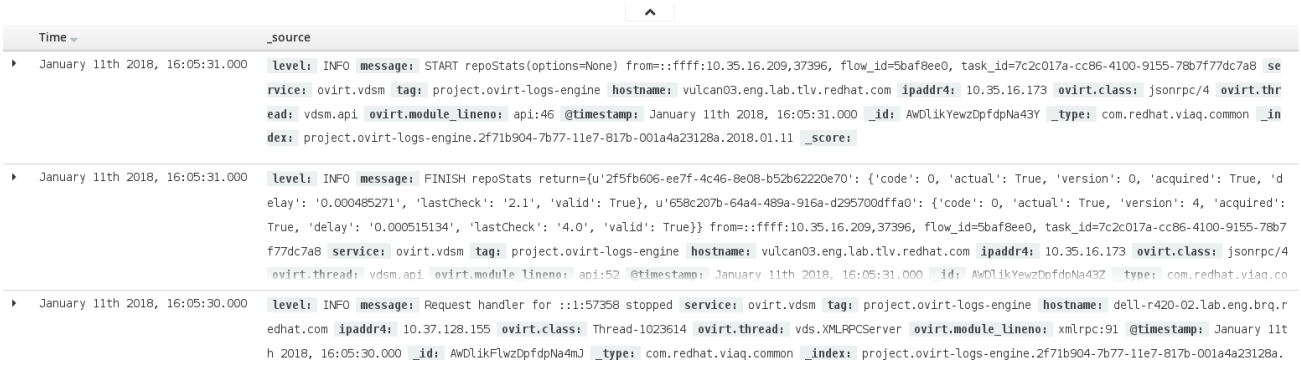
The distribution of documents over time is displayed in a histogram at the top of the page. By default the information is grouped into 30 second intervals, but this can be changed by clicking the time drop-down list that appears above the histogram.

Figure 3.1. Histogram



The bottom of the page displays the documents in a table, sorted according to time.

Figure 3.2. Documents Table



Time	_source
January 11th 2018, 16:05:31.000	level: INFO message: START repoStats(options=None) from=::ffff:10.35.16.209,37396, flow_id=5baf8ee0, task_id=7c2c017a-cc86-4100-9155-78b7f77dc7a8 service: ovirt.vdsm tag: project.ovirt-logs-engine hostname: vulcan03.eng.lab.tlv.redhat.com ipaddr4: 10.35.16.173 ovirt.class: jsonrpc/4 ovirt.thread: vdsm.api ovirt.module_lineno: api:46 @timestamp: January 11th 2018, 16:05:31.000 _id: AWDlikYewzDpfdpNa43Y _type: com.redhat.viaq.common _index: project.ovirt-logs-engine.2f71b904-7b77-11e7-817b-001a4a23128a.2018.01.11 _score:
January 11th 2018, 16:05:31.000	level: INFO message: FINISH repoStats return={u'2f5fb606-ee7f-4c46-8e08-b52b62220e70': {'code': 0, 'actual': True, 'version': 0, 'acquired': True, 'delay': '0.000485271', 'lastCheck': '2.1', 'valid': True}, u'658c207b-64a4-489a-916a-d295700dffa0': {'code': 0, 'actual': True, 'version': 4, 'acquired': True, 'delay': '0.000515134', 'lastCheck': '4.0', 'valid': True}} from=::ffff:10.35.16.209,37396, flow_id=5baf8ee0, task_id=7c2c017a-cc86-4100-9155-78b7f77dc7a8 service: ovirt.vdsm tag: project.ovirt-logs-engine hostname: vulcan03.eng.lab.tlv.redhat.com ipaddr4: 10.35.16.173 ovirt.class: jsonrpc/4 ovirt.thread: vdsm.api ovirt.module_lineno: api:52 @timestamp: January 11th 2018, 16:05:31.000 _id: AWDlikYewzDpfdpNa43Z _type: com.redhat.viaq.co
January 11th 2018, 16:05:30.000	level: INFO message: Request handler for ::1:57358 stopped service: ovirt.vdsm tag: project.ovirt-logs-engine hostname: dell-r420-02.lab.eng.brq.redhat.com ipaddr4: 10.37.128.155 ovirt.class: Thread-1023614 ovirt.thread: vds.XMLRPCServer ovirt.module_lineno: xmlrpc:91 @timestamp: January 11th 2018, 16:05:30.000 _id: AWDlikFlwzDpfdpNa4mJ _type: com.redhat.viaq.common _index: project.ovirt-logs-engine.2f71b904-7b77-11e7-817b-001a4a23128a.

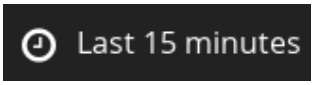
3.2. USING THE DISCOVER PAGE

From the **Discover** page you can:

- Set the time filter
- Submit search queries
- Filter the search results
- View the results in the Visualization page
- Customize the Documents table

3.2.1. Setting the Time Filter

By default, data from the last 15 minutes is displayed. There are several ways to change the time filter:

- Click the time filter 

and either select a predefined time filter or define a time range from the **Relative** or **Absolute** menus.
- Define a filter directly from the histogram by clicking a bar or click and drag over several bars. For more information, see Setting the Time Filter in the Kibana documentation.

3.2.2. Searching Your Data

Use the search field at the top of the page to filter the results according to a specific value. For example, to display results containing the word "login", type **`*login*`** in the search field. For more information about searches, see Searching Your Data in the Kibana documentation.

3.2.3. Filtering By Field

Filtering log data by field enables you to focus on the specific error that interests you.

To filter the log data by field:

1. Click the name of the field you want to filter on from the **Available Fields** pane. This displays the top five values for that field. To the right of each value, there are two magnifying glass buttons, one for adding a regular (positive) filter, and one for adding a negative filter.

Table 3.1. Available Fields

Available Field	Description
_id	The unique ID of the document.
_index	The ID of the index to which the document belongs. The index with the *project.ovirt-logs* prefix is the only relevant index in the **Discover** page.
hostname	For the engine.log this is the hostname of the Manager. For the vdsm.log this is hostname of the host.
level	The log record's severity: TRACE, DEBUG, INFO, WARN, ERROR, FATAL.
message	The body of the document's message.
ovirt.class	The name of a Java class that produced this log.
ovirt.correlationid	For the engine.log only. This ID is used to correlate the multiple parts of a single task performed by the Manager.

Available Field	Description
ovirt.thread	The name of a Java thread inside which the log record was produced.
tag	Predefined sets of metadata that can be used to filter the data.
@timestamp	The time that the record was issued.
_score	N/A
_type	N/A
ipaddr4	The machine's IP address.
ovirt.cluster_name	For the vdsm.log only. The name of the cluster to which the host belongs.
ovirt.engine_fqdn	The Manager's FQDN.
ovirt.module_lineno	The file and line number within the file that ran the command defined in *ovirt.class*.
pipeline_metadata.collector.inputname	N/A
pipeline_metadata.collector.ipaddr4	N/A
pipeline_metadata.collector.ipaddr6	N/A
pipeline_metadata.collector.name	N/A
pipeline_metadata.collector.received_at	N/A
pipeline_metadata.collector.version	N/A
service	The log file from which the document was extracted.

2. To add a positive filter, click the **Positive Filter** button  . This filters out results that do not contain that value in the field.

3. To add a negative filter, click the **Negative Filter** button  . This excludes results that contain that value in the field.
   For more information about working with filters, see Working with Filters in the Kibana documentation.

3.2.4. Visualizing Log Data

You can visualize and aggregate log data in the **Visualization** page by selecting a specific field from within the **Discover** page.

To visualize log data:

1. Click the name of the field you want to visualize from the **Available Fields** pane (see Section 3.2.3, “Filtering By Field”).
2. Click the **Visualize** button that appears beneath the top five values.
   You are transferred to the **Visualize** page where you can view the filtered value in a graphical format.

3.2.5. Customizing the Documents Table

You can customize the way that the data is displayed in the Documents table by adding fields to the table as columns and changing the display order.

To add fields to the table as columns:

1. Hover over the name of the field you want to add to the documents table from the **Available Fields** pane (see Section 3.2.3, “Filtering By Field”).
2. Click **add**. The field is added to the table.
3. Optionally click the **Sort by** arrow that appears next to the column title to sort the results by that column.

CHAPTER 4. TROUBLESHOOTING

The following sections explain how to resolve issues that may occur in Metrics Store.

4.1. INFORMATION IS MISSING FROM KIBANA

If Kibana is not displaying metric or log information as expected, you can use **`journalctl`** to investigate the collectd and fluentd log files as follows:

- If only metrics information is missing, check the collectd log files.
- If only log information is missing, check the fluentd log files.
- If both metrics and logs information are missing, check both log files.

1. To investigate collectd log files:

```
# journalctl -u collectd
```

2. To investigate fluentd log files:

```
# journalctl -u fluentd
```

To learn about the other journalctl options refer to the man page.