Data Collector Reference - Services

Data Infrastructure Insights

NetApp
January 17, 2025

This PDF was generated from https://docs.netapp.com/us-en/data-infrastructure-insights/task_config_telegraf_node.html on January 17, 2025. Always check docs.netapp.com for the latest.

Table of Contents

Data Collector Reference - Services . . . 1
- Node Data Collection . . . 1
- ActiveMQ Data Collector . . . 3
- Apache Data Collector . . . 5
- Consul Data Collector . . . 8
- Couchbase Data Collector . . . 9
- CouchDB Data Collector . . . 11
- Docker Data Collector . . . 13
- Elasticsearch Data Collector . . . 21
- Flink Data Collector . . . 23
- Hadoop Data Collector . . . 30
- HAProxy Data Collector . . . 36
- JVM Data Collector . . . 42
- Kafka Data Collector . . . 46
- Kibana Data Collector . . . 49
- Kubernetes Monitoring Operator Installation and Configuration . . . 51
- Memcached Data Collector . . . 71
- MongoDB Data Collector . . . 74
- MySQL Data Collector . . . 76
- Netstat Data Collector . . . 81
- Nginx Data Collector . . . 82
- PostgreSQL Data Collector . . . 85
- Puppet Agent Data Collector . . . 87
- Redis Data Collector . . . 89

Data Collector Reference - Services

Node Data Collection

Data Infrastructure Insights gathers metrics from the node on which you install an agent.

Installation

1. From **Observability > Collectors**, choose an operating system/platform. Note that installing any integration data collector (Kubernetes, Docker, Apache, etc.) will also configure node data collection.
2. Follow the instructions to configure the agent. The instructions vary depending on the type of Operating System or Platform you are using to collect data.

Objects and Counters

The following objects and their counters are collected as Node metrics:

Object:	Identifiers:	Attributes:	Datapoints:
Node Filesystem	Node UUID Device Path Type	Node IP Node Name Node OS Mode	Free Inodes Free Inodes Total Inodes Used Total Used Total Used
Node Disk	Node UUID Disk	Node IP Node Name Node OS	IO Time Total IOPS In Progress Read Bytes (per sec) Read Time Total Reads (per sec) Weighted IO Time Total Write Bytes (per sec) Write Time Total Writes (per sec) Current Disk Queue Length Write Time Read Time IO Time
Node CPU	Node UUID CPU	Node IP Node Name Node OS	System CPU Usage User CPU Usage Idle CPU Usage Processor CPU Usage Interrupt CPU Usage DPC CPU Usage

Object:	Identifiers:	Attributes:	Datapoints:
Node	Node UUID	Node IP Node Name Node OS	Kernel Boot Time Kernel Context Switches (per sec) Kernel Entropy Available Kernel Interrupts (per sec) Kernel Processes Forked (per sec) Memory Active Memory Available Total Memory Available Memory Buffered Memory Cached Memory Commit Limit Memory Committed As Memory Dirty Memory Free Memory High Free Memory High Total Memory Huge Page Size Memory Huge Pages Free Memory Huge Pages Total Memory Low Free Memory Low Total Memory Mapped Memory Page Tables Memory Shared Memory Slab Memory Swap Cached Memory Swap Free Memory Swap Total Memory Total Memory Used Total Memory Used Memory Vmalloc Chunk Memory Vmalloc Total Memory Vmalloc Used Memory Wired Memory Writeback Total Memory Writeback Tmp Memory Cache Faults Memory Demand Zero Faults Memory Page Faults Memory Pages Memory Nonpaged Memory Paged Memory Cache Core Memory Standby Cache Normal Memory Standby Cache Reserve Memory Transition Faults Processes Blocked Processes Dead

Object:	Identifiers:	Attributes:	Datapoints:
Node Network	Network Interface Node UUID	Node Name Node IP Node OS	Bytes Received Bytes Sent Packets Outboud Discarded Packets Outboud Errors Packets Received Discarded Packets Received Errors Packets Received Packets Sent

Setup

Setup and Troubleshooting information can be found on the Configuring an Agent page.

ActiveMQ Data Collector

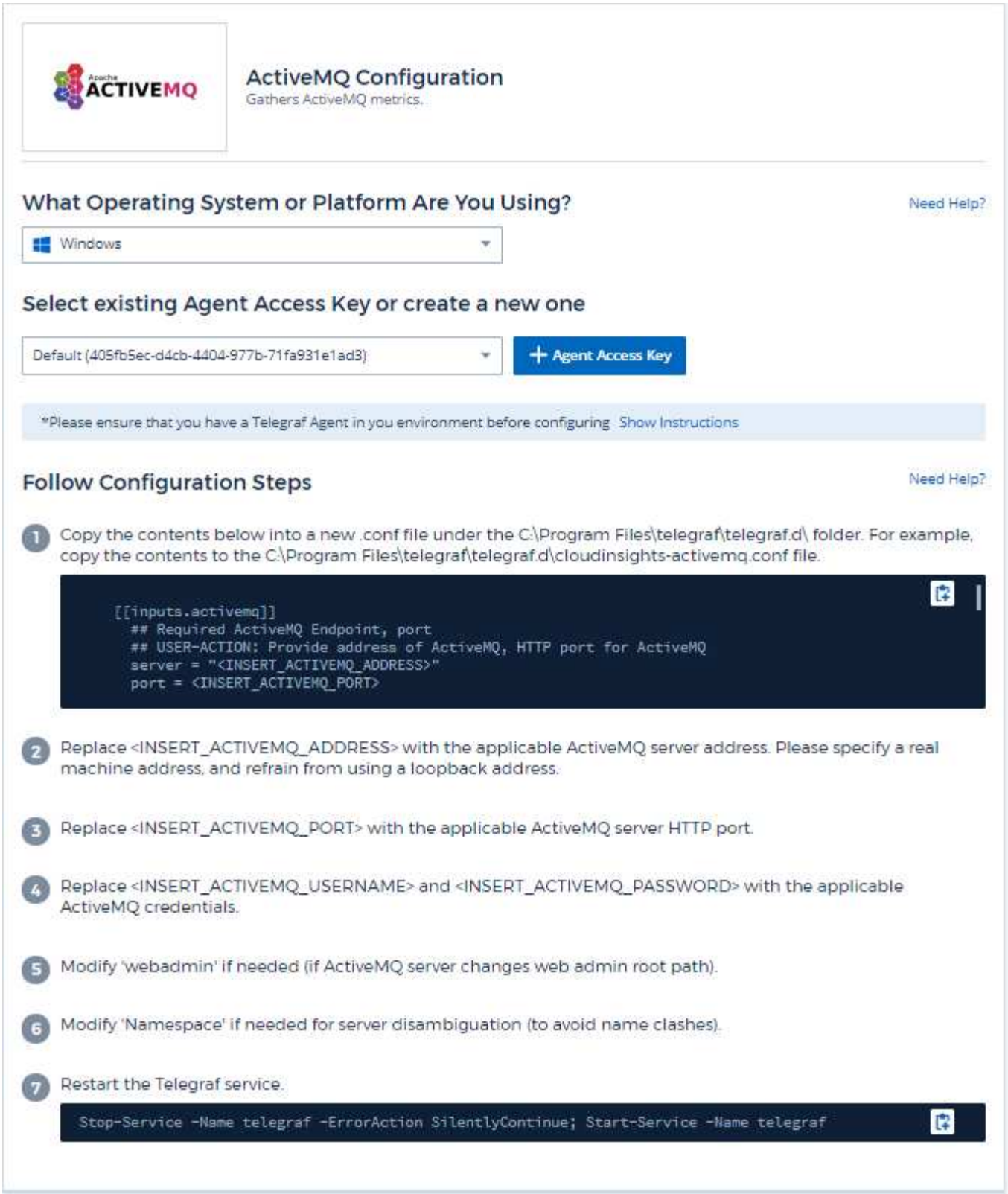
Data Infrastructure Insights uses this data collector to gather metrics from ActiveMQ.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose ActiveMQ.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the ActiveMQ documentation

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
ActiveMQ Queue	Namespace Queue Port Server	Node Name Node IP Node UUID	Consumer Count Dequeue Count Enqueue Count Queue Size
ActiveMQ Subscriber	Client ID Connection ID Port Server Namespace	Is Active Destination Node Name Node IP Node UUID Node OS Selector Subscription	Dequeue Count Dispatched Count Dispatched Queue Size Enqueue Count Pending Queue Size
ActiveMQ Topic	Topic Port Server Namespace	Node Name Node IP Node UUID Node OS	Consumer Count Dequeue Count Enqueue Count Size

Troubleshooting

Additional information may be found from the Support page.

Apache Data Collector

This data collector allows collection of data from Apache servers on your tenant.

Pre-requisites

- You must have your Apache HTTP Server set up and properly running
- You must have sudo or administrator permissions on your agent host/VM
- Typically, the Apache *mod_status* module is configured to expose a page at the '/server-status?auto' location of the Apache server. The *ExtendedStatus* option must be enabled in order to collect all available fields. For information about how to configure your server, see the Apache module documentation: https://httpd.apache.org/docs/2.4/mod/mod_status.html#enable

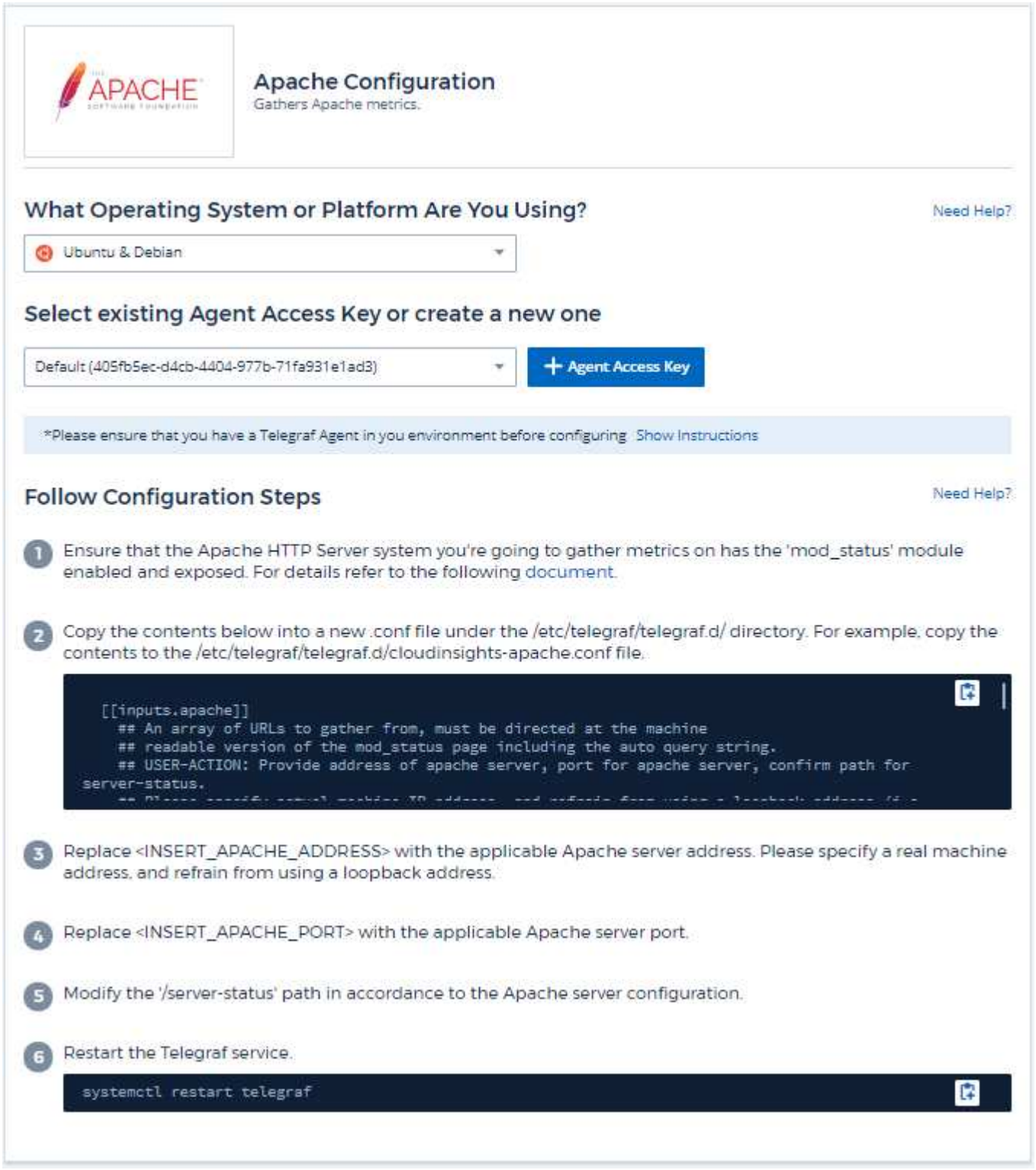
Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Apache.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.

4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Telegraf's plugin for Apache's HTTP Server relies on the 'mod_status' module to be enabled. When this is enabled, Apache's HTTP Server will expose an HTML endpoint that can be viewed on your browser or scraped for extraction of status of all Apache's HTTP Server configuration.

Compatibility:

Configuration was developed against Apache's HTTP Server version 2.4.38.

Enabling mod_status:

Enabling and exposing the 'mod_status' modules involves two steps:

- Enabling module
- Exposing stats from module

Enabling module:

The loading of modules is controlled by the config file under '/usr/local/apache/conf/httpd.conf'. Edit the config file and uncomment the following lines:

```
LoadModule status_module modules/mod_status.so
```

```
Include conf/extra/httpd-info.conf
```

Exposing stats from module:

The exposing of 'mod_status' is controlled by the config file under '/usr/local/apache2/conf/extra/httpd-info.conf'. Make sure you have the following in that configuration file (at least, other directives will be there):

```
# Allow server status reports generated by mod_status,
# with the URL of http://servername/server-status
<Location /server-status>
    SetHandler server-status
</Location>

#
# ExtendedStatus controls whether Apache will generate "full" status
# information (ExtendedStatus On) or just basic information
(ExtendedStatus
# Off) when the "server-status" handler is called. The default is Off.
#
ExtendedStatus On
```

For detailed instructions on the 'mod_status' module, see the Apache documentation

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Apache	Namespace Server	Node IP Node Name Port Parent Server Config Generation Parent Server MPM Generation Server Uptime Is Stopping	Busy Workers Bytes per Request Bytes per Second CPU Children System CPU Children User CPU Load CPU System CPU User Asynchronous Connections Closing Asynchronous Connections Keep Alive Asynchronous Connections Writing Connections Total Duration per Request Idle Workers Load Average (last 1m) Load Average (last 15m) Load Average (last 5m) Processes Requests per Second Total Accesses Total Duration Total KBytes Scoreboard Closing Scoreboard DNS Lookups Scoreboard Finishing Scoreboard Idle Cleanup Scoreboard Keep Alive Scoreboard Logging Scoreboard Open Scoreboard Reading Scoreboard Sending Scoreboard Starting Scoreboard Waiting

Troubleshooting

Additional information may be found from the Support page.

Consul Data Collector

Data Infrastructure Insights uses this data collector to gather metrics from Consul.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Consul.

   If you haven't configured an Agent for collection, you are prompted to install an agent on your tenant.

If you have an agent already configured, select the appropriate Operating System or Platform and click **Continue**.

2. Follow the instructions in the Consul Configuration screen to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.

Setup

Information may be found in the Consul documentation.

Objects and Counters for consul

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Consul	Namespace Check ID Service Node	Node IP Node OS Node UUID Node Name Service Name Check Name Service ID Status	Critical Passing Warning

Troubleshooting

Additional information may be found from the Support page.

Couchbase Data Collector

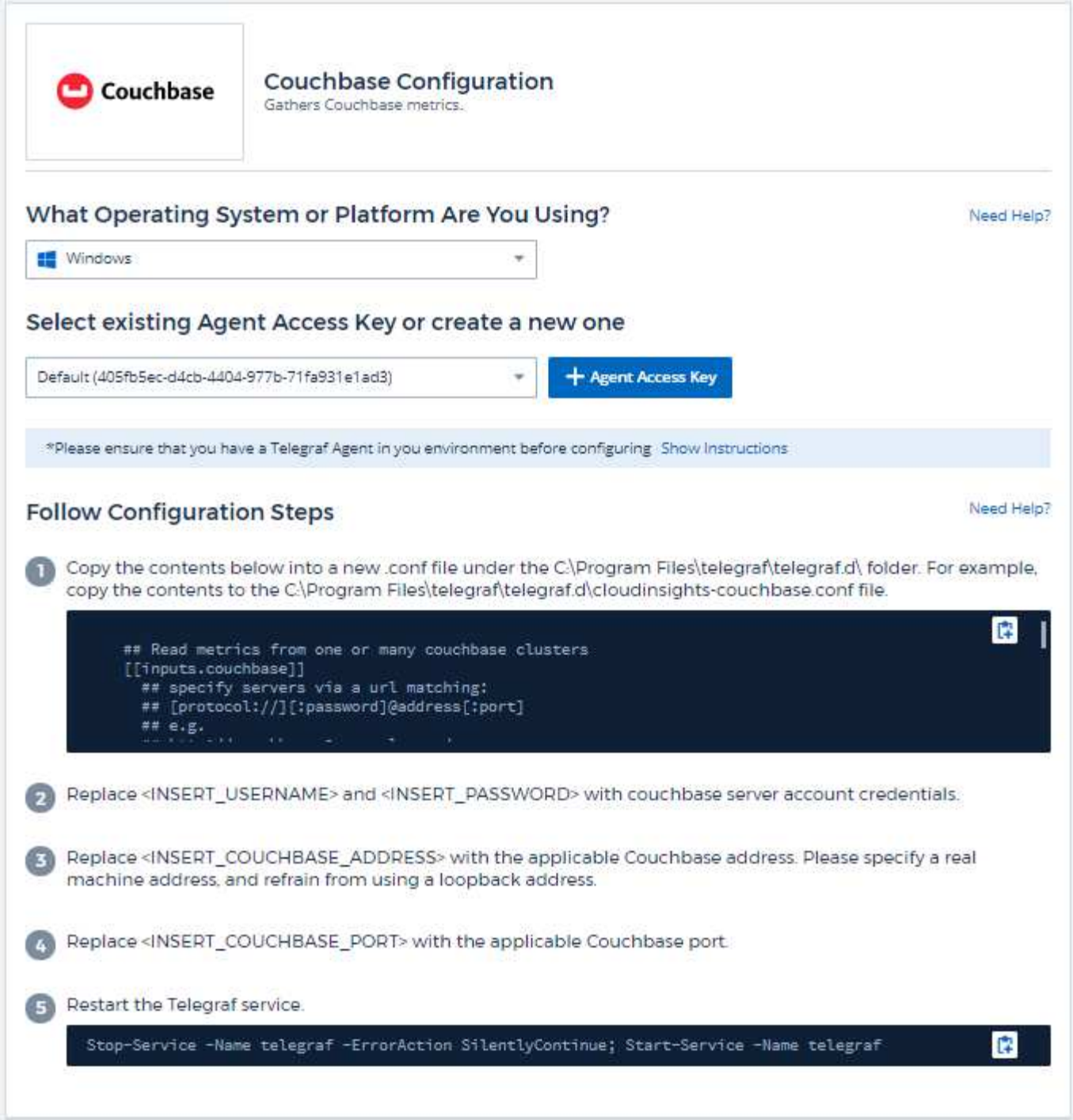
Data Infrastructure Insights uses this data collector to gather metrics from Couchbase.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Couchbase.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Couchbase documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Couchbase Node	Namespace Cluster Couchbase Node Hostname	Node Name Node IP	Memory Free Memory Total
Couchbase Bucket	Namespace Bucket Cluster	Node Name Node IP	Data Used Data Fetches Disk Used Item Count Memory Used Operations Per Second Quota Used

Troubleshooting

Additional information may be found from the Support page.

CouchDB Data Collector

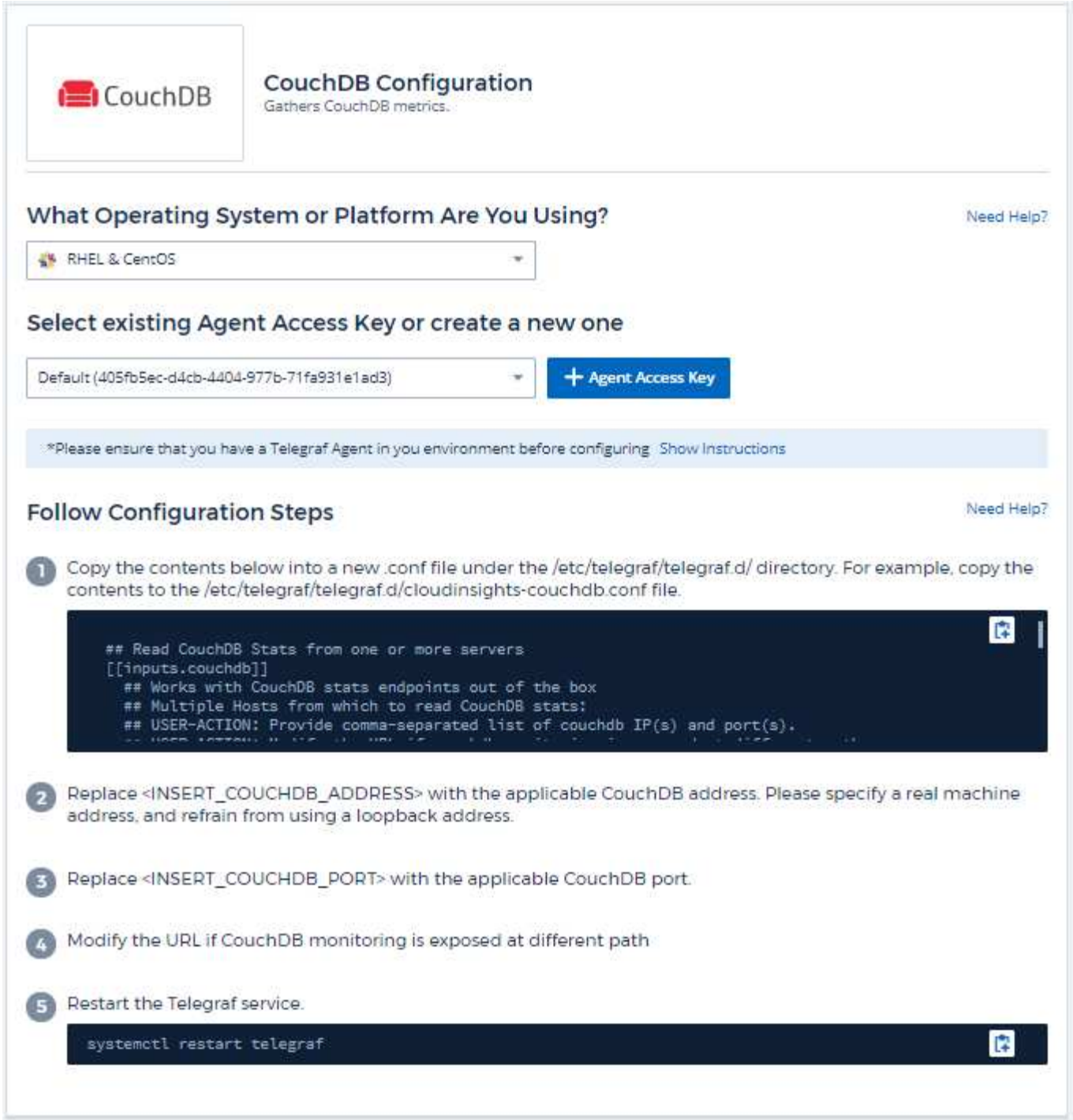
Data Infrastructure Insights uses this data collector to gather metrics from CouchDB.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose CouchDB.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the CouchDB documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
CouchDB	Namespace Server	Node Name Node IP	Authentication Cache Hits Authentication Cache Miss Database Reads Database Writes Databases Open Open OS Files Max Request Time Min Request Time Httpd Request Methods Copy Httpd Request Methods Delete Httpd Request Methods Get Httpd Request Methods Head Httpd Request Methods Post Httpd Request Methods Put Status Codes 200 Status Codes 201 Status Codes 202 Status Codes 301 Status Codes 304 Status Codes 400 Status Codes 401 Status Codes 403 Status Codes 404 Status Codes 405 Status Codes 409 Status Codes 412 Status Codes 500

Troubleshooting

Additional information may be found from the Support page.

Docker Data Collector

Data Infrastructure Insights uses this data collector to gather metrics from Docker.

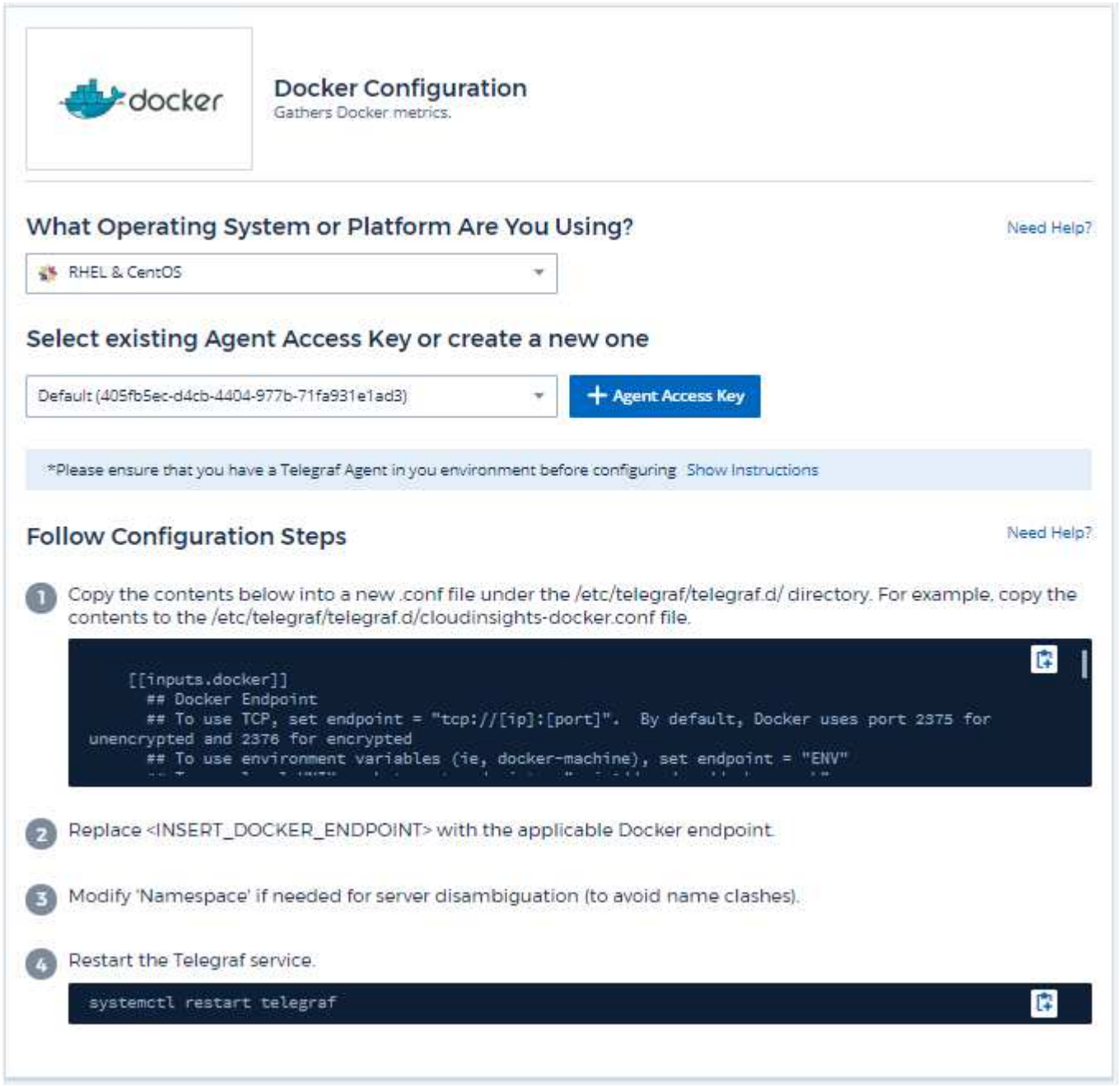
Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Docker.

   If you haven't configured an Agent for collection, you are prompted to install an agent on your tenant.

   If you have an agent already configured, select the appropriate Operating System or Platform and click **Continue**.

2. Follow the instructions in the Docker Configuration screen to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

The Telegraf input plugin for Docker collects metrics through a specified UNIX socket or a TCP endpoint.

Compatibility

Configuration was developed against Docker version 1.12.6.

Setting Up

Accessing Docker through a UNIX socket

If the Telegraf agent is running on baremetal, add the telegraf Unix user to the docker Unix group by running the following:

```
sudo usermod -aG docker telegraf
```

If the Telegraf agent is running within a Kubernetes pod, expose the Docker Unix socket by mapping the socket into the pod as a volume and then mounting that volume to /var/run/docker.sock. For example, add the following to the PodSpec:

```
volumes:
...
- name: docker-sock
hostPath:
path: /var/run/docker.sock
type: File
```

Then, add the following to the Container:

```
volumeMounts:
...
- name: docker-sock
mountPath: /var/run/docker.sock
```

Note that the Data Infrastructure Insights installer provided for the Kubernetes platform takes care of this mapping automatically.

Access Docker through a TCP endpoint

By default, Docker uses port 2375 for unencrypted access and port 2376 for encrypted access.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Docker Engine	Namespace Docker Engine	Node Name Node IP Node UUID Node OS Kubernetes Cluster Docker Version Unit	Memory Containers Containers Paused Containers Running Containers Stopped CPUs Go Routines Images Listener Events Used File Descriptors Data Available Data Total Data Used Metadata Available Metadata Total Metadata Used Pool Blocksize

Object:	Identifiers:	Attributes:	Datapoints:
Docker Container	Namespace Container Name Docker Engine	Kubernetes Container Hash Kubernetes Container Ports Kubernetes Container Restart Count Kubernetes Container Termination Message Path Kubernetes Container Termination Message Policy Kubernetes Pod Termination Grace Period Container Image Container Status Container Version Node Name Kubernetes Container Log Path Kubernetes Container Name Kubernetes Docker Type Kubernetes Pod Name Kubernetes Pod Namespace Kubernetes Pod UID Kubernetes Sandbox ID Node IP Node UUID Docker Version Kubernetes IO Config Seen Kubernetes IO Config Source OpenShift IO SCC Kubernetes Description Kubernetes Display Name OpenShift Tags Kompose Service Pod Template Hash Controller Revision Hash Pod Template Generation License Schema Build Date Schema License Schema Name Schema URL Schema VCS URL Schema Vendor Schema Version Schema Schema Version Maintainer Customer Pod	Memory Active Anonymous Memory Active File Memory Cache Memory Hierarchical Limit Memory Inactive Anonymous Memory Inactive File Memory Limit Memory Mapped File Memory Max Usage Memory Page Fault Memory Page Major Fault Memory Paged In Memory Paged Out Memory Resident Set Size Memory Resident Set Size Huge Memory Total Active Anonymous Memory Total Active File Memory Total Cache Memory Total Inactive Anonymous Memory Total Inactive File Memory Total Mapped File Memory Total Page Fault Memory Total Page Major Fault Memory Total Paged In Memory Total Paged Out Memory Total Resident Set Size Memory Total Resident Set Size Huge Memory Total Unevictable Memory Unevictable Memory Usage Memory Usage Percent Exit Code OOM Killed PID Started At Failing Streak

Object:	Identifiers:	Attributes:	Datapoints:
Docker Container Block IO	Namespace Container Name Device Docker Engine	Kubernetes Container Hash Kubernetes Container Ports Kubernetes Container Restart Count Kubernetes Container Termination Message Path Kubernetes Container Termination Message Policy Kubernetes Pod Termination Grace Period Container Image Container Status Container Version Node Name Kubernetes Container Log Path Kubernetes Container Name Kubernetes Docker Type Kubernetes Pod Name Kubernetes Pod Namespace Kubernetes Pod UID Kubernetes Sandbox ID Node IP Node UUID Docker Version Kubernetes Config Seen Kubernetes Config Source OpenShift SCC Kubernetes Description Kubernetes Display Name OpenShift Tags Schema Schema Version Pod Template Hash Controller Revision Hash Pod Template Generation Kompose Service Schema Build Date Schema License Schema Name Schema Vendor Customer Pod Kubernetes StatefulSet Pod Name Tenant Webconsole Build Date License Vendor	IO Service Bytes Recursive Async IO Service Bytes Recursive Read IO Service Bytes Recursive Sync IO Service Bytes Recursive Total IO Service Bytes Recursive Write IO Serviced Recursive Async IO Serviced Recursive Read IO Serviced Recursive Sync IO Serviced Recursive Total IO Serviced Recursive Write

Object:	Identifiers:	Attributes:	Datapoints:
Docker Container Network	Namespace Container Name Network Docker Engine	Container Image Container Status Container Version Node Name Node IP Node UUID Node OS K8s Cluster Docker Version Container ID	RX Dropped RX Bytes RX Errors RX Packets TX Dropped TX Bytes TX Errors TX Packets

Object:	Identifiers:	Attributes:	Datapoints:
Docker Container CPU	Namespace Container Name CPU Docker Engine	Kubernetes Container Hash Kubernetes Container Ports Kubernetes Container Restart Count Kubernetes Container Termination Message Path Kubernetes Container Termination Message Policy Kubernetes Pod Termination Grace Period Kubernetes Config Seen Kubernetes Config Source OpenShift SCC Container Image Container Status Container Version Node Name Kubernetes Container Log Path Kubernetes Container name Kubernetes Docker Type Kubernetes Pod Name Kubernetes Pod Namespace Kubernetes Pod UID Kubernetes Sandbox ID Node IP Node UUID Node OS Kubernetes Cluster Docker Version Kubernetes Description Kubernetes Display Name OpenShift Tags Schema Version Pod Template Hash Controller Revision Hash Pod Template Generation Kompose Service Schema Build Date Schema License Schema Name Schema Vendor Customer Pod Kubernetes StatefulSet Pod Name Tenant Webconsole Build Date	Throttling Periods Throttling Throttled Periods Throttling Throttled Time Usage In Kernel Mode Usage In User Mode Usage Percent Usage System Usage Total

Troubleshooting

Problem:	Try this:
I do not see my Docker metrics in Data Infrastructure Insights after following the instructions on the configuration page.	Check the Telegraf agent logs to see if it reports the following error: E! Error in plugin [inputs.docker]: Got permission denied while trying to connect to the Docker daemon socket If it does, take the necessary steps to provide the Telegraf agent access to the Docker Unix socket as specified above.

Additional information may be found from the Support page.

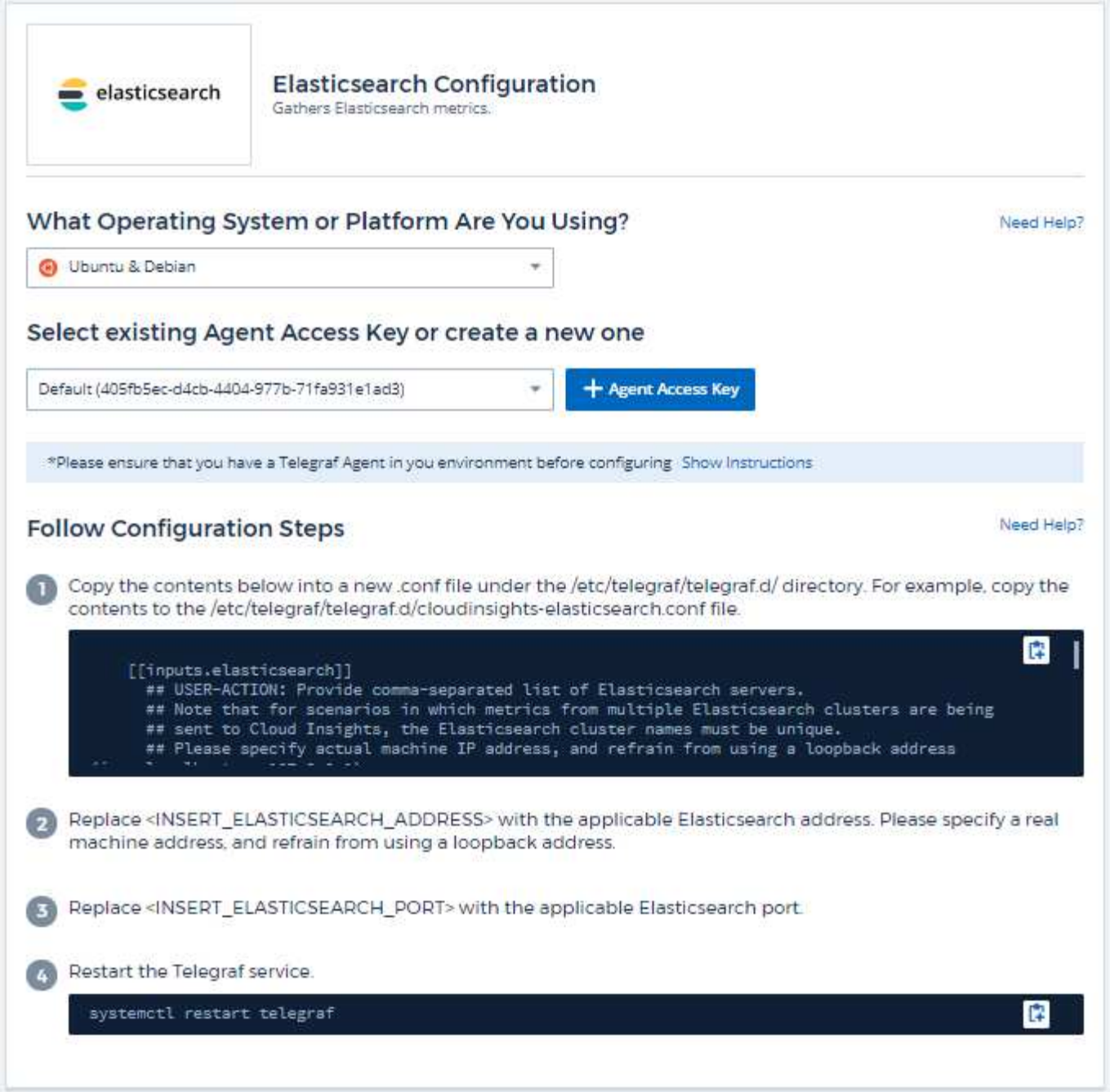
Elasticsearch Data Collector

Data Infrastructure Insights uses this data collector to gather metrics from Elasticsearch.

1. From **Observability > Collectors**, click **+Data Collector**. Choose Elasticsearch.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Elasticsearch documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:
Elasticsearch Cluster	Namespace Cluster	Node IP Node Name Cluster Status

Object:	Identifiers:	Attributes:
Elasticsearch Node	Namespace Cluster ES Node ID ES Node IP ES Node	Zone ID

Troubleshooting

Additional information may be found from the Support page.

Flink Data Collector

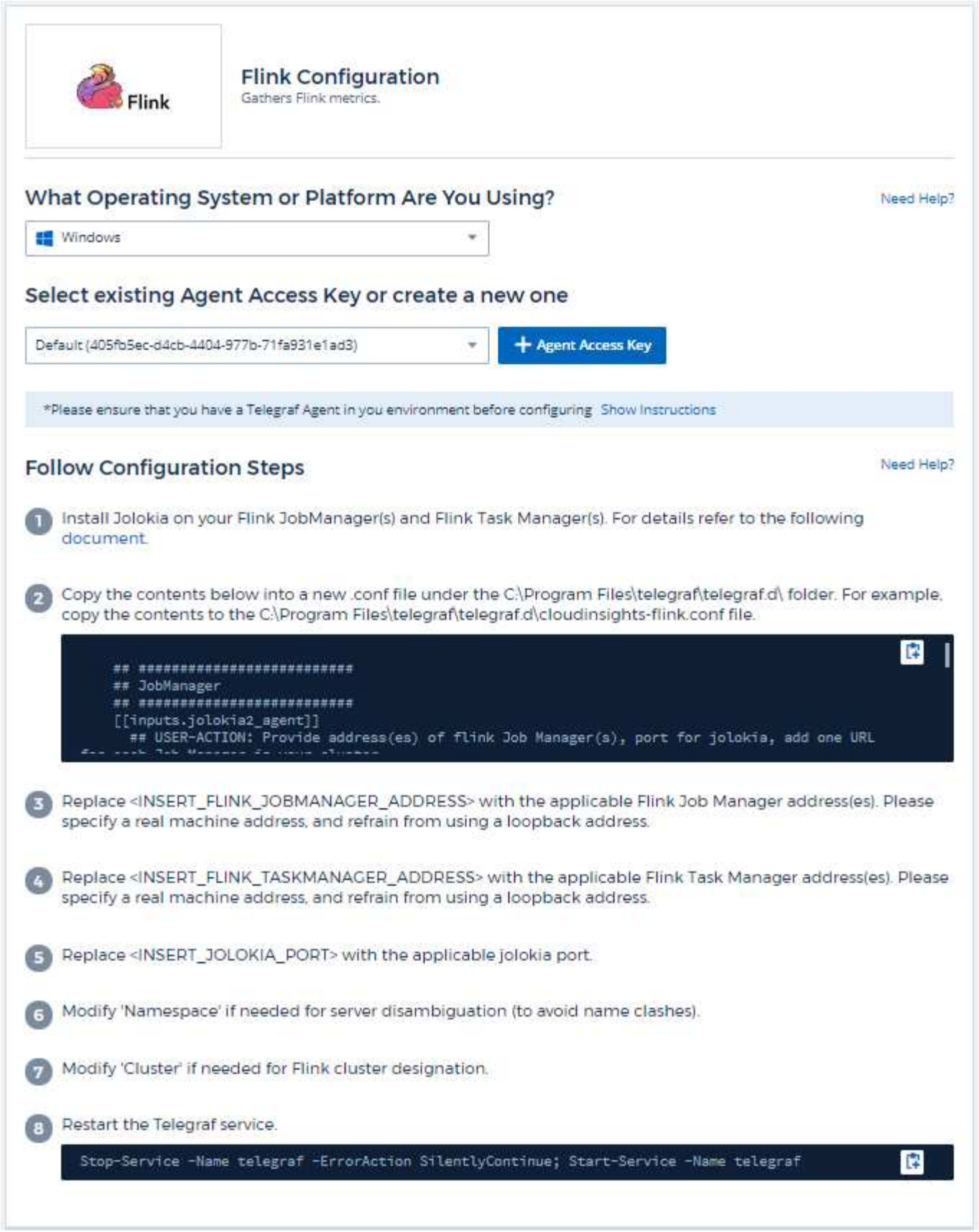
Data Infrastructure Insights uses this data collector to gather metrics from Flink.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Flink.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

A full Flink deployment involves the following components:

JobManager: The Flink primary system. Coordinates a series of TaskManagers. In a High Availability setup, system will have more than one JobManager.
TaskManager: This is where Flink operators are executed.
The Flink plugin is based on the telegraf's Jolokia plugin. As such as a requirement to gather info from all Flink components, JMX needs to be configured and exposed via Jolokia on all components.

Compatibility

Configuration was developed against Flink version 1.7.0.

Setting Up

Jolokia Agent Jar

For all individual components, a version the Jolokia agent jar file must be downloaded. The version tested against was Jolokia agent 1.6.0.

Instructions below assume that downloaded jar file (jolokia-jvm-1.6.0-agent.jar) is placed under location '/opt/flink/lib/'.

JobManager

To configure JobManager to expose the Jolokia API, you can setup the following environment variable on your nodes then restart the JobManager:

```
export FLINK_ENV_JAVA_OPTS="-javaagent:/opt/flink/lib/jolokia-jvm-1.6.0-agent.jar=port=8778,host=0.0.0.0"
```

You can choose a different port for Jolokia (8778). If you have an internal IP to lock Jolokia onto you can replace the "catch all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from the telegraf plugin.

TaskManager

To configure TaskManager(s) to expose the Jolokia API, you can setup the following environment variable on your nodes then restart the TaskManager:

```
export FLINK_ENV_JAVA_OPTS="-javaagent:/opt/flink/lib/jolokia-jvm-1.6.0-agent.jar=port=8778,host=0.0.0.0"
```

You can choose a different port for Jolokia (8778). If you have an internal IP to lock Jolokia onto you can replace the "catch all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from the telegraf plugin.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Flink Task Manager	Cluster Namespace Server	Node Name Task Manager ID Node IP	Network Available Memory Segments Network Total Memory Segments Garbage Collection PS MarkSweep Count Garbage Collection PS MarkSweep Time Garbage Collection PS Scavenge Count Garbage Collection PS Scavenge Time Heap Memory Committed Heap Memory Init Heap Memory Max Heap Memory Used Thread Count Daemon Thread Count Peak Thread Count Thread Count Total Started
Flink Job	Cluster Namespace server Job ID	Node Name Job Name Node IP Last Checkpoint External Path Restarting Time	Downtime Full Restarts Last Checkpoint Alignment Buffered Last Checkpoint Duration Last Checkpoint Size Number of Completed Checkpoints Number of Failed Checkpoints Number of in Progress Checkpoints Number of Checkpoints Uptime

Object:	Identifiers:	Attributes:	Datapoints:
Flink Job Manager	Cluster Namespace Server	Node Name Node IP	Garbage Collection PS MarkSweep Count Garbage Collection PS MarkSweep Time Garbage Collection PS Scavenge Count Garbage Collection PS Scavenge Time Heap Memory Committed Heap Memory Init Heap Memory Max Heap Memory Used Number Registered Task Managers Number Running Jobs Task Slots Available Task Slots Total Thread Count Daemon Thread Count Peak Thread Count Thread Count Total Started

Object:	Identifiers:	Attributes:	Datapoints:
Flink Task	Cluster Namespace Job ID Task ID	Server Node Name Job Name Sub Task Index Task Attempt ID Task Attempt Number Task Name Task Manager ID Node IP Current Input Watermark	Buffers In Pool Usage Buffers In Queue Length Buffers Out Pool Usage Buffers Out Queue Length Number Buffers In Local Number Bufffers In Local Per Second Count Number Buffers in Local Per Second Rate Number Buffers In Remote Number Buffers In Remote Per Second Count Number Buffers In Remote Per Second Rate Number Buffers Out Number Buffers Out Per Second Count Number Buffers Out Per Second Rate Number Bytes In Local Number Bytes In Local Per Second Count Number Bytes In Local Per Second Rate Number Bytes In Remote Number Bytes In Remote Per Second Count Number Bytes In Remote Per Second Rate Number Bytes Out Number Bytes Out Per Second Count Number Bytes Out Per Second Rate Number Records In Number Records In Per Second Count Number Records In Per Second Rate Number Records Out Number Records Out Per Second Count Number Records Out Per Second Rate

Object:	Identifiers:	Attributes:	Datapoints:
Flink Task Operator	Cluster Namespace Job ID Operator ID Task ID	Server Node Name Job Name Operator Name Sub Task Index Task Attempt ID Task Attempt Number Task Name Task Manager ID Node IP	Current Input Watermark Current Output Watermark Number Records In Number Records In Per Second Count Number Records In Per Second Rate Number Records Out Number Records Out Per Second Count Number Records Out Per Second Rate Number Late Records Dropped Assigned Partitions Bytes Consumed Rate Commit Latency Avg Commit Latency Max Commit Rate Commits Failed Commits Succeeded Connection Close Rate Connection Count Connection Creation Rate Count Fetch Latency Avg Fetch Latency Max Fetch Rate Fetch Size Avg Fetch Size Max Fetch Throttle Time Avg Fetch Throttle Time Max Heartbeat Rate Incoming Byte Rate IO Ratio IO Time Avg (ns) IO Wait Ratio IO Wait Time Avg (ns) Join Rate Join Time Avg Last Heartbeat Ago Network IO Rate Outgoing Byte Rate Records Consumed Rate Records Lag Max Records per Request Avg Request Rate Request Size Avg Request Size Max Response Rate Select Rate Sync Rate Sync Time Avg Heartbeat Response Time

Troubleshooting

Additional information may be found from the Support page.

Hadoop Data Collector

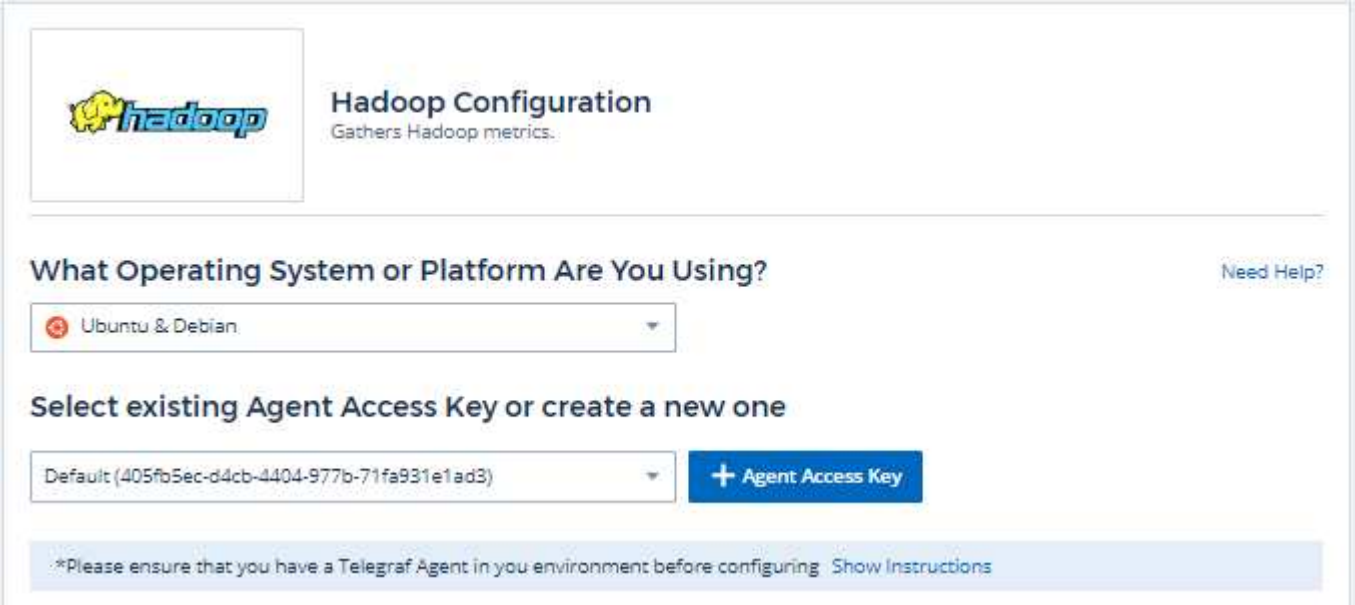
Data Infrastructure Insights uses this data collector to gather metrics from Hadoop.

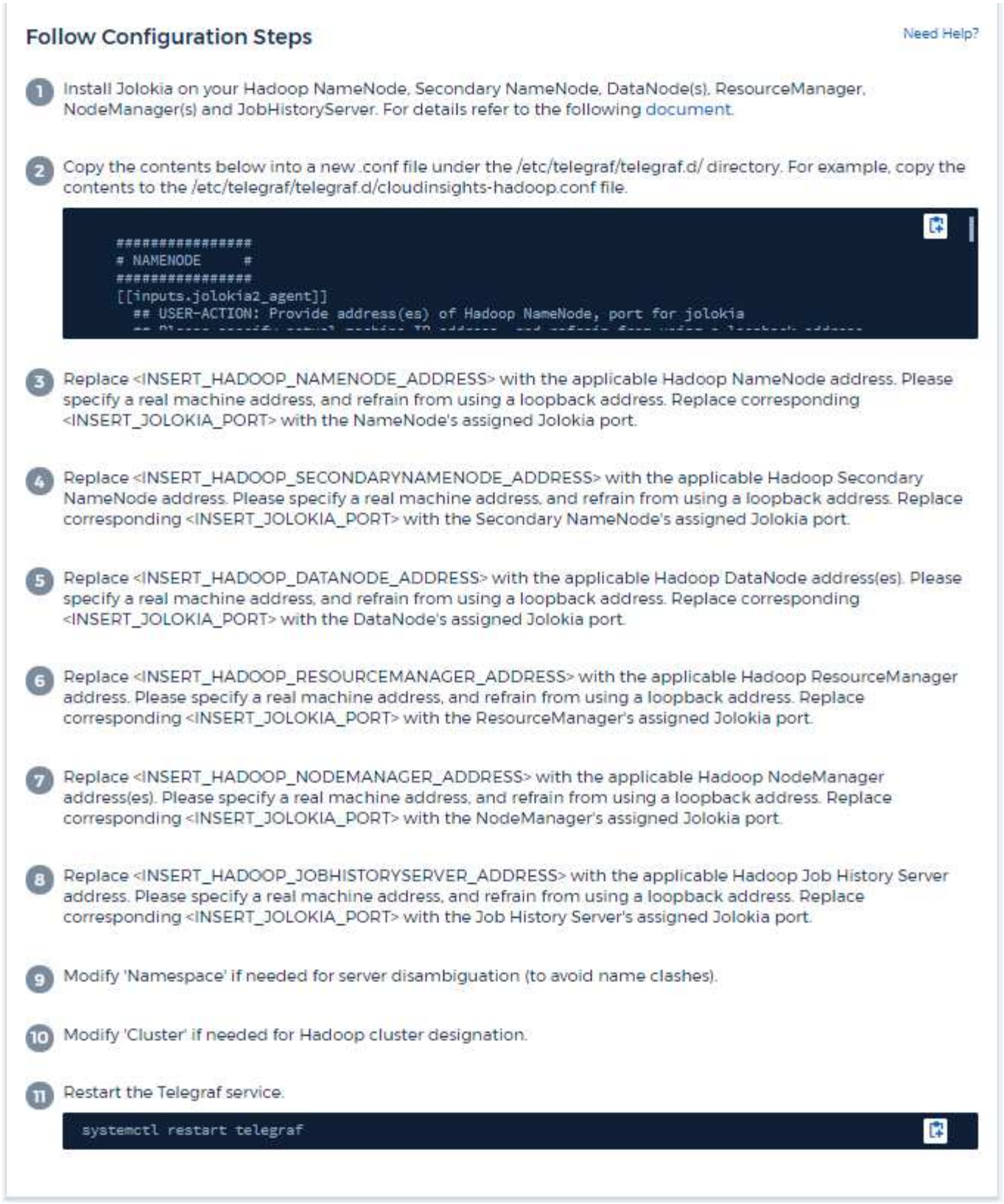
Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Hadoop.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.





Setup

A full Hadoop deployment involves the following components:

- NameNode: The Hadoop Distributed File System (HDFS) primary system. Coordinates a series of DataNodes.

- Secondary NameNode: a warm failover for the main NameNode. In Hadoop the promotion to NameNode does not occur automatically. Secondary NameNode gathers information from NameNode to be ready to be promoted when needed.
- DataNode: Actual owner for data.
- ResourceManager: The compute primary system (Yarn). Coordinates a series of NodeManagers.
- NodeManager: The resource for compute. Actual location for running of applications.
- JobHistoryServer: Responsible for servicing all job history related requests.

The Hadoop plugin is based on the telegraf's Jolokia plugin. As such as a requirement to gather info from all Hadoop components, JMX needs to be configured and exposed via Jolokia on all components.

Compatibility

Configuration was developed against Hadoop version 2.9.2.

Setting Up

Jolokia Agent Jar

For all individual components, a version the Jolokia agent jar file must be downloaded. The version tested against was Jolokia agent 1.6.0.

Instructions below assume that downloaded jar file (jolokia-jvm-1.6.0-agent.jar) is placed under location '/opt/hadoop/lib/'.

NameNode

To configure NameNode to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export HADOOP_NAMENODE_OPTS="$HADOOP_NAMENODE_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7800,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8000
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8000 above) and Jolokia (7800).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

Secondary NameNode

To configure the Secondary NameNode to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export HADOOP_SECONDARYNAMENODE_OPTS="$HADOOP_SECONDARYNAMENODE_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7802,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8002
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8002 above) and Jolokia (7802).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

DataNode

To configure the DataNodes to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export HADOOP_DATANODE_OPTS="$HADOOP_DATANODE_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7801,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8001
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8001 above) and Jolokia (7801).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

ResourceManager

To configure the ResourceManager to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export YARN_RESOURCEMANAGER_OPTS="$YARN_RESOURCEMANAGER_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7803,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8003
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8003 above) and Jolokia (7803).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

NodeManager

To configure the NodeManagers to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export YARN_NODEMANAGER_OPTS="$YARN_NODEMANAGER_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7804,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8004
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8004 above) and Jolokia (7804).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

JobHistoryServer

To configure the JobHistoryServer to expose the Jolokia API, you can setup the following in <HADOOP_HOME>/etc/hadoop/hadoop-env.sh:

```
export HADOOP_JOB_HISTORYSERVER_OPTS="$HADOOP_JOB_HISTORYSERVER_OPTS
-javaagent:/opt/hadoop/lib/jolokia-jvm-1.6.0
-agent.jar=port=7805,host=0.0.0.0 -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=8005
-Dcom.sun.management.jmxremote.password.file=$HADOOP_HOME/conf/jmxremote.p
assword"
You can choose a different port for JMX (8005 above) and Jolokia (7805).
If you have an internal IP to lock Jolokia onto you can replace the "catch
all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from
the telegraf plugin. You can use the option '-
Dcom.sun.management.jmxremote.authenticate=false' if you don't want to
authenticate. Use at your own risk.
```

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:
Hadoop Secondary NameNode	Cluster Namespace Server	Node Name Node IP Compile Info Version
Hadoop NodeManager	Cluster Namespace Server	Node Name Node IP
Hadoop ResourceManager	Cluster Namespace Server	Node Name Node IP
Hadoop DataNode	Cluster Namespace Server	Node Name Node IP Cluster ID Version
Hadoop NameNode	Cluster Namespace Server	Node Name Node IP Transaction ID Last Written Time Since Last Loaded Edits HA State File System State Block Pool ID Cluster ID Compile Info Distinct Version Count Version
Hadoop JobHistoryServer	Cluster Namespace Server	Node Name Node IP

Troubleshooting

Additional information may be found from the Support page.

HAProxy Data Collector

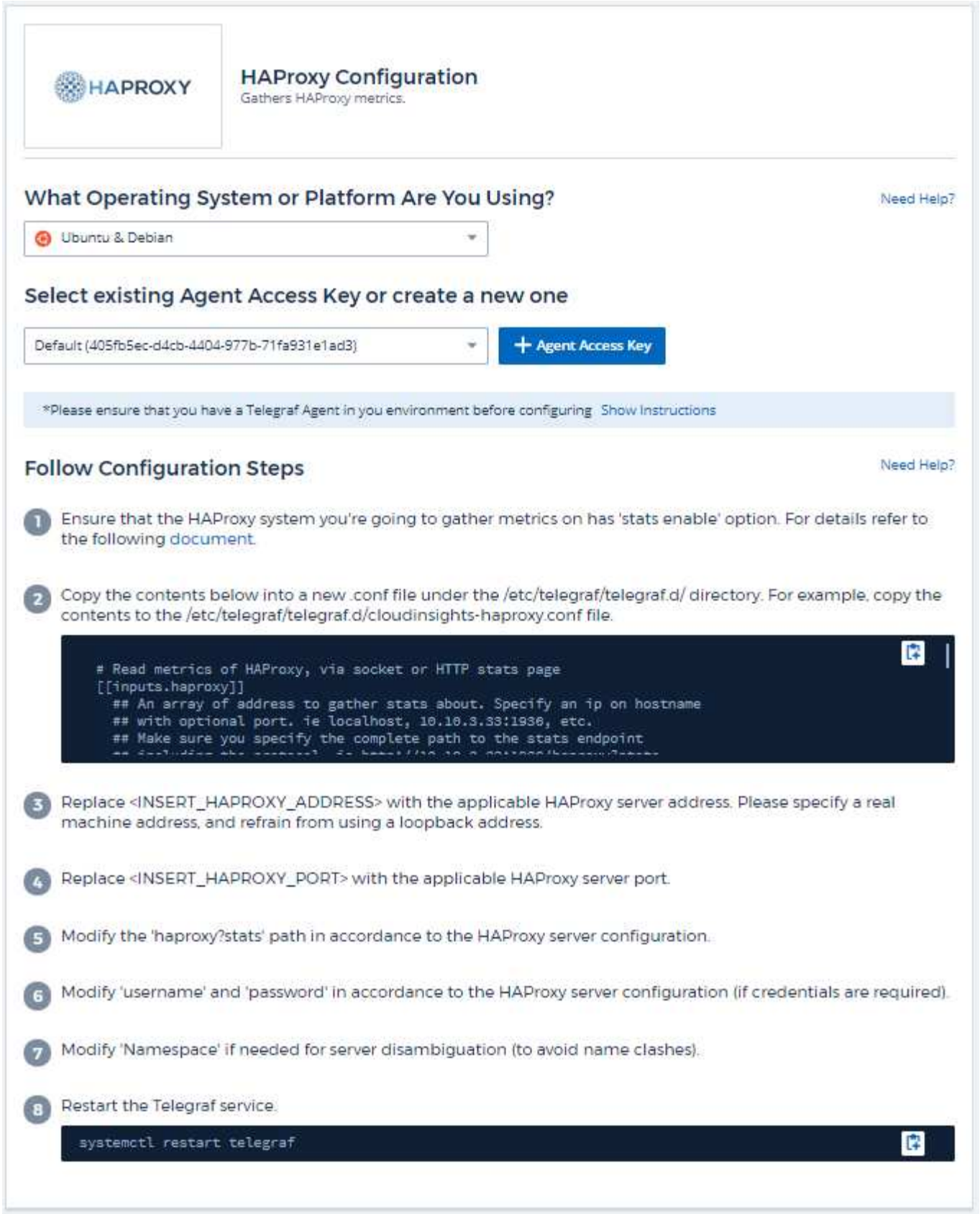
Data Infrastructure Insights uses this data collector to gather metrics from HAProxy.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose HAProxy.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Telegraf's plugin for HAProxy relies on HAProxy Stats enablement. This is a configuration built into HAProxy but it is not enabled out of the box. When enabled, HAProxy will expose an HTML endpoint that can be viewed

on your browser or scraped for extraction of status of all HAProxy configurations.

Compatibility:

Configuration was developed against HAProxy version 1.9.4.

Setting Up:

To enable stats, edit your haproxy configuration file and add the the following lines after the 'defaults' section, using your own user/password and/or haproxy URL:

```
stats enable
stats auth myuser:mypassword
stats uri  /haproxy?stats
```

The following is a simplified example configuration file with stats enabled:

```
global
  daemon
  maxconn 256

defaults
  mode http
  stats enable
  stats uri /haproxy?stats
  stats auth myuser:mypassword
  timeout connect 5000ms
  timeout client 50000ms
  timeout server 50000ms

frontend http-in
  bind *:80
  default_backend servers

frontend http-in9080
  bind *:9080
  default_backend servers_2

backend servers
  server server1 10.128.0.55:8080 check ssl verify none
  server server2 10.128.0.56:8080 check ssl verify none

backend servers_2
  server server3 10.128.0.57:8080 check ssl verify none
  server server4 10.128.0.58:8080 check ssl verify none
```

For complete and up to date instructions, see the HAProxy documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
HAProxy Frontend	Namespace Address Proxy	Node IP Node Name Proxy ID Mode Process id Sessions Rate Limit Server id Sessions Limit Status	Bytes In Bytes Out Cache Hits Cache Lookups Compression Bytes Bypassed Compression Bytes In Compression Bytes Out Compression Responses Connection Rate Connection Rate Max Connections Total Requests Denied by Connection Rule Requests Denied by Security Concerns Responses Denied by Security Concerns Requests Denied by Session Rule Requests Errors Responses 1xx Responses 2xx Responses 3xx Responses 4xx Responses 5xx Responses Other Requests Intercepted Sessions Rate Sessions Rate Max Requests Rate Requests Rate Max Requests Total Sessions Sessions Max Sessions Total Requests Rewrites

Object:	Identifiers:	Attributes:	Datapoints:
HAProxy Server	Namespace Address Proxy Server	Node IP Node Name Check Time to Finish Check Fall Configuration Check Health Value Check Rise Configuration Check Status Proxy ID Last Change Time Last Session Time Mode Process id Server id Status Weight	Active Servers Backup Servers Bytes In Bytes Out Check Downs Check Fails Client Aborts Connections Connection Average Time Downtime Total Denied Responses Connection Errors Response Errors Responses 1xx Responses 2xx Responses 3xx Responses 4xx Responses 5xx Responses Other Server Selected Total Queue Current Queue Max Queue Average Time Sessions per Second Sessions per Second Max Connection Reuse Response Time Average Sessions Sessions Max Server Transfer Aborts Sessions Total Sessions Total Time Average Requests Redispatches Requests Retries Requests Rewrites

Object:	Identifiers:	Attributes:	Datapoints:
HAProxy Backend	Namespace Address Proxy	Node IP Node Name Proxy ID Last Change Time Last Session Time Mode Process id Server id Sessions Limit Status Weight	Active Servers Backup Servers Bytes In Bytes Out Cache Hits Cache Lookups Check Downs Client Aborts Compression Bytes Bypassed Compression Bytes In Compression Bytes Out Compression Responses Connections Connection Average Time Downtime Total Requests Denied by Security Concerns Responses Denied by Security Concerns Connection Errors Response Errors Responses 1xx Responses 2xx Responses 3xx Responses 4xx Responses 5xx Responses Other Server Selected Total Queue Current Queue Max Queue Average Time Sessions per Second Sessions per Second Max Requests Total Connection Reuse Response Time Average Sessions Sessions Max Server Transfer Aborts Sessions Total Sessions Total Time Average Requests Redispatches Requests Retries Requests Rewrites

Troubleshooting

Additional information may be found from the Support page.

JVM Data Collector

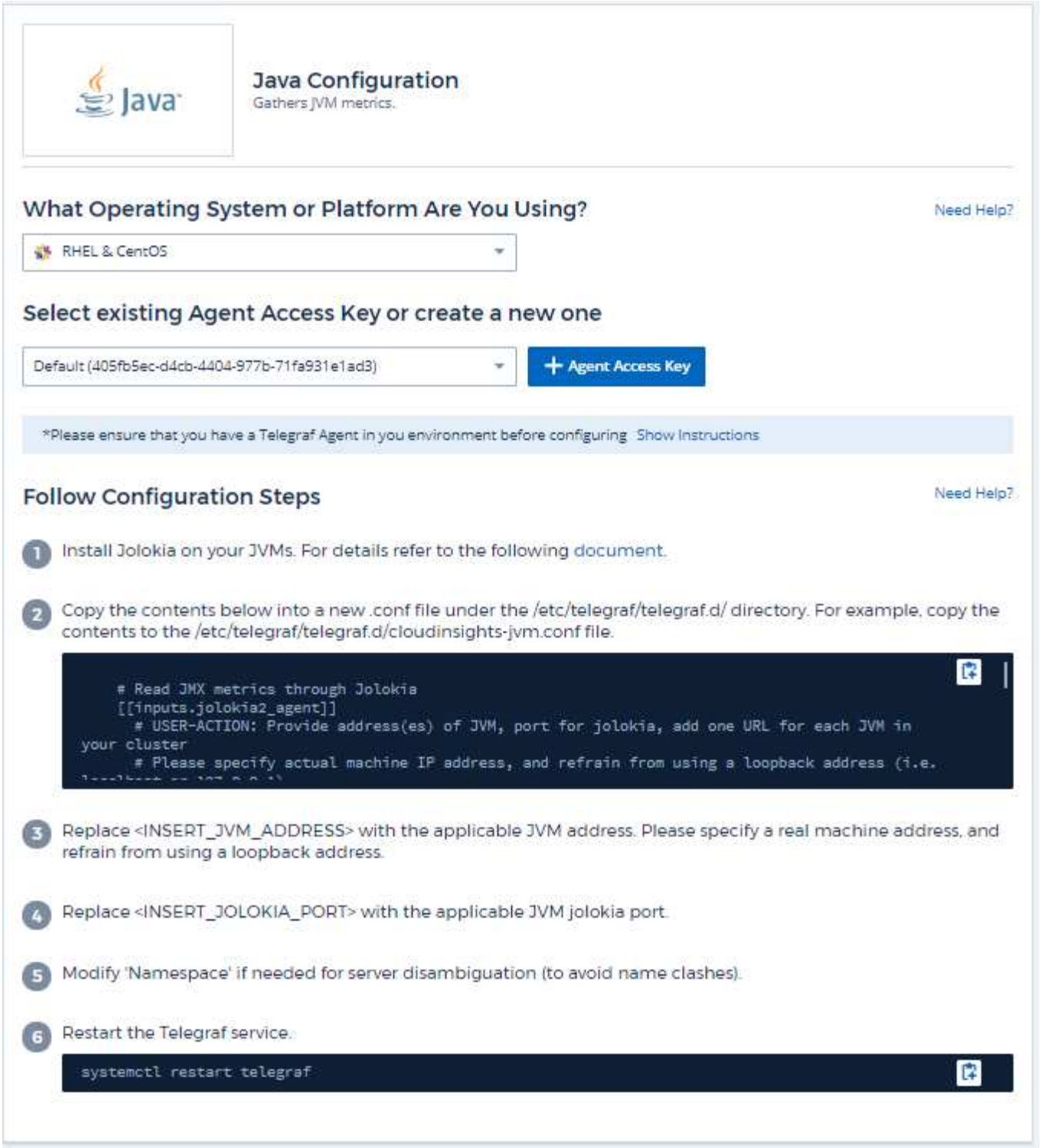
Data Infrastructure Insights uses this data collector to gather metrics from JVM.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose JVM.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in JVM documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
JVM	Namespace JVM	OS Architecture OS Name OS Version Runtime Specification Runtime Specification Vendor Runtime Specification Version Uptime Runtime VM Name Runtime VM Vendor Runtime VM Version Node Name Node IP	Class Loaded Class Loaded Total Class Unloaded Memory Heap Committed Memory Heap Init Memory Heap Used Max Memory Heap Used Memory Non Heap Committed Memory Non Heap Init Memory Non Heap Max Memory Non Heap Used Memory Objects Pending Finalization OS Processors Available OS Committed Virtual Memory Size OS Free Physical Memory Size OS Free Swap Space Size OS Max File Descriptor Count OS Open File Descriptors Count OS Processor CPU Load OS Processor CPU Time OS System CPU Load OS System Load Average OS Total Physical Memory Size OS Total Swap Space Size Thread Daemon Count Thread Peak Count Thread Count Thread Total Started Count Garbage Collector Copy Collection Count Garbage Collector Copy Collection Time Garbage Collector Mark-sweep Collection Count Garbage Collector Mark-sweep Collection Time Garbage Collector G1 Old Generation Collection Count Garbage Collector G1 Old Generation Collection Time Garbage Collector G1 Young Generation

Troubleshooting

Additional information may be found from the Support page.

Kafka Data Collector

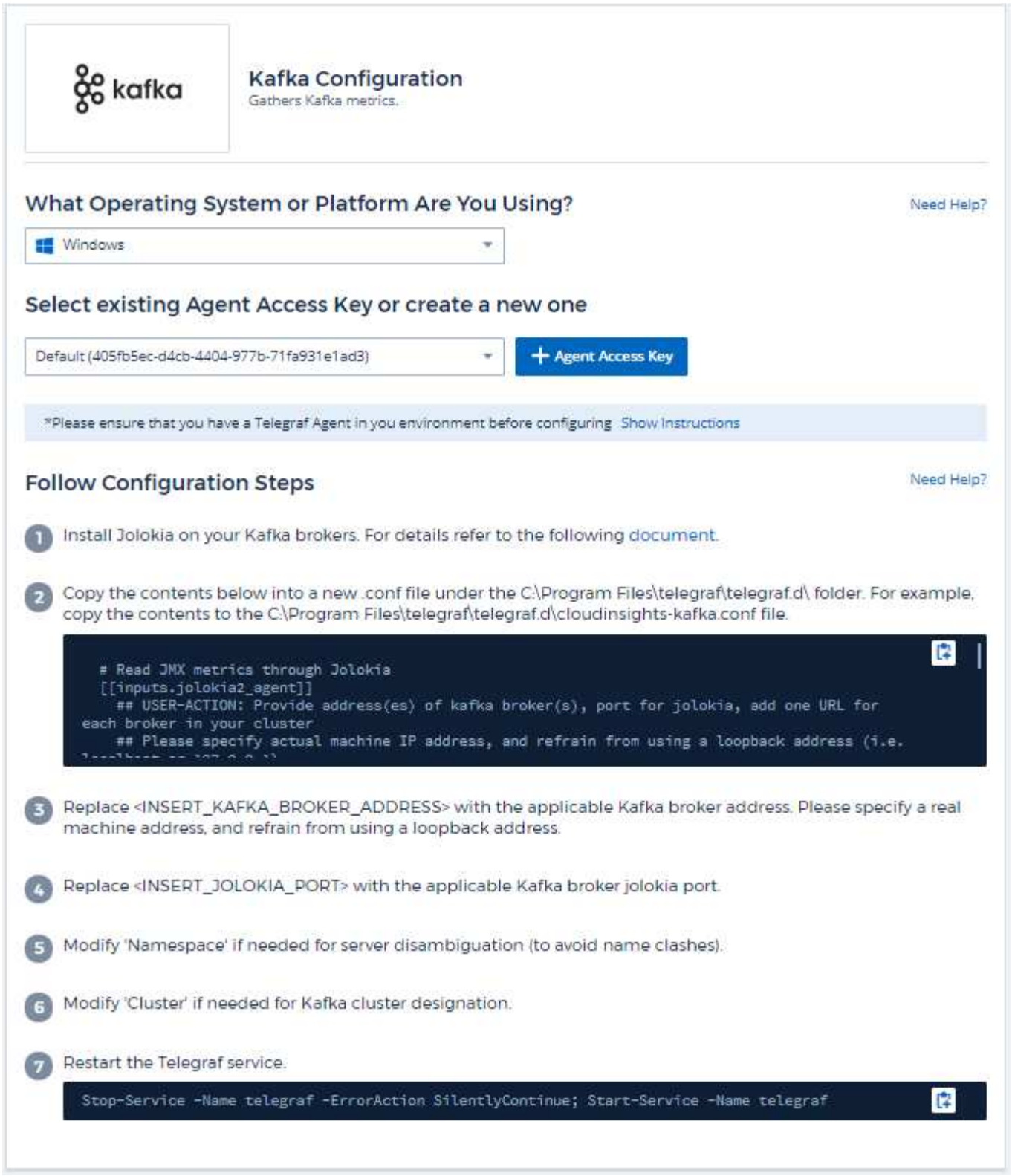
Data Infrastructure Insights uses this data collector to gather metrics from Kafka.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Kafka.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

The Kafka plugin is based on the telegraf's Jolokia plugin. As such as a requirement to gather info from all Kafka brokers, JMX needs to be configured and exposed via Jolokia on all components.

Compatibility

Configuration was developed against Kafka version 0.11.0.2.

Setting up

All the instructions below assume your install location for kafka is '/opt/kafka'. You can adapt instructions below to reflect your install location.

Jolokia Agent Jar

A version the Jolokia agent jar file must be downloaded. The version tested against was Jolokia agent 1.6.0.

Instructions below assume that the downloaded jar file (jolokia-jvm-1.6.0-agent.jar) is placed under the location '/opt/kafka/libs/'.

Kafka Brokers

To configure Kafka Brokers to expose the Jolokia API, you can add the following in <KAFKA_HOME>/bin/kafka-server-start.sh, just before the 'kafka-run-class.sh' call:

```
export JMX_PORT=9999
export RMI_HOSTNAME=`hostname -I`
export KAFKA_JMX_OPTS="-javaagent:/opt/kafka/libs/jolokia-jvm-1.6.0-
agent.jar=port=8778,host=0.0.0.0
-Dcom.sun.management.jmxremote.password.file=/opt/kafka/config/jmxremote.p
assword -Dcom.sun.management.jmxremote.ssl=false
-Djava.rmi.server.hostname=$RMI_HOSTNAME
-Dcom.sun.management.jmxremote.rmi.port=$JMX_PORT"
```

Note that example above is using 'hostname -I' to setup the 'RMI_HOSTNAME' environment variable. In multiple IP machines, this will need to be tweaked to gather the IP you care about for RMI connections.

You can choose a different port for JMX (9999 above) and Jolokia (8778). If you have an internal IP to lock Jolokia onto you can replace the "catch all" 0.0.0.0 by your own IP. Notice this IP needs to be accessible from the telegraf plugin. You can use the option '-Dcom.sun.management.jmxremote.authenticate=false' if you don't want to authenticate. Use at your own risk.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:
Kafka Broker	Cluster Namespace Broker	Node Name Node IP

Troubleshooting

Additional information may be found from the Support page.

Kibana Data Collector

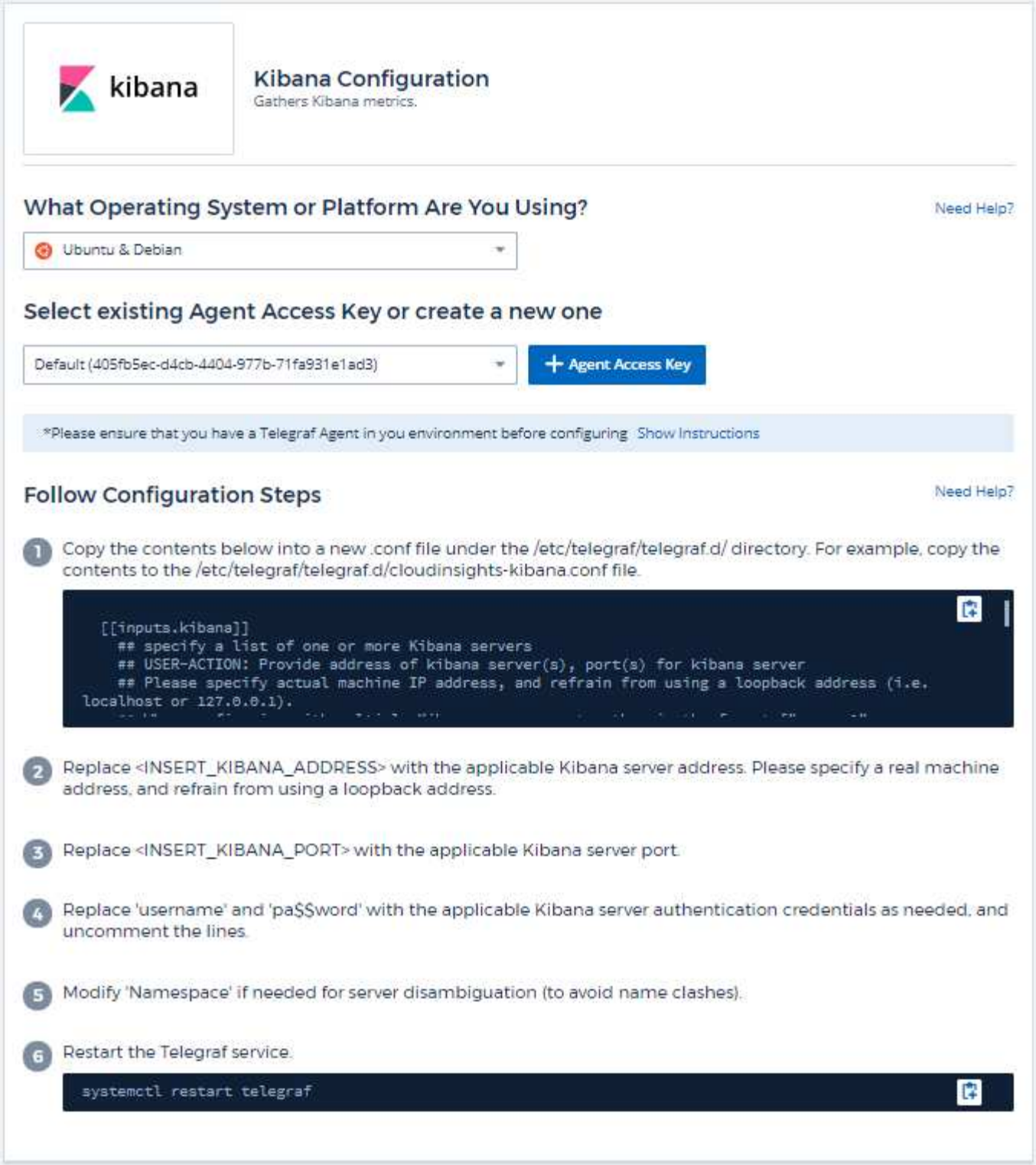
Data Infrastructure Insights uses this data collector to gather metrics from Kibana.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Kibana.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Kibana documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Kibana	Namespace Address	Node IP Node Name Version Status	Concurrent Connections Heap Max Heap Used Requests per Second Response Time Average Response Time Max Uptime

Troubleshooting

Additional information may be found from the Support page.

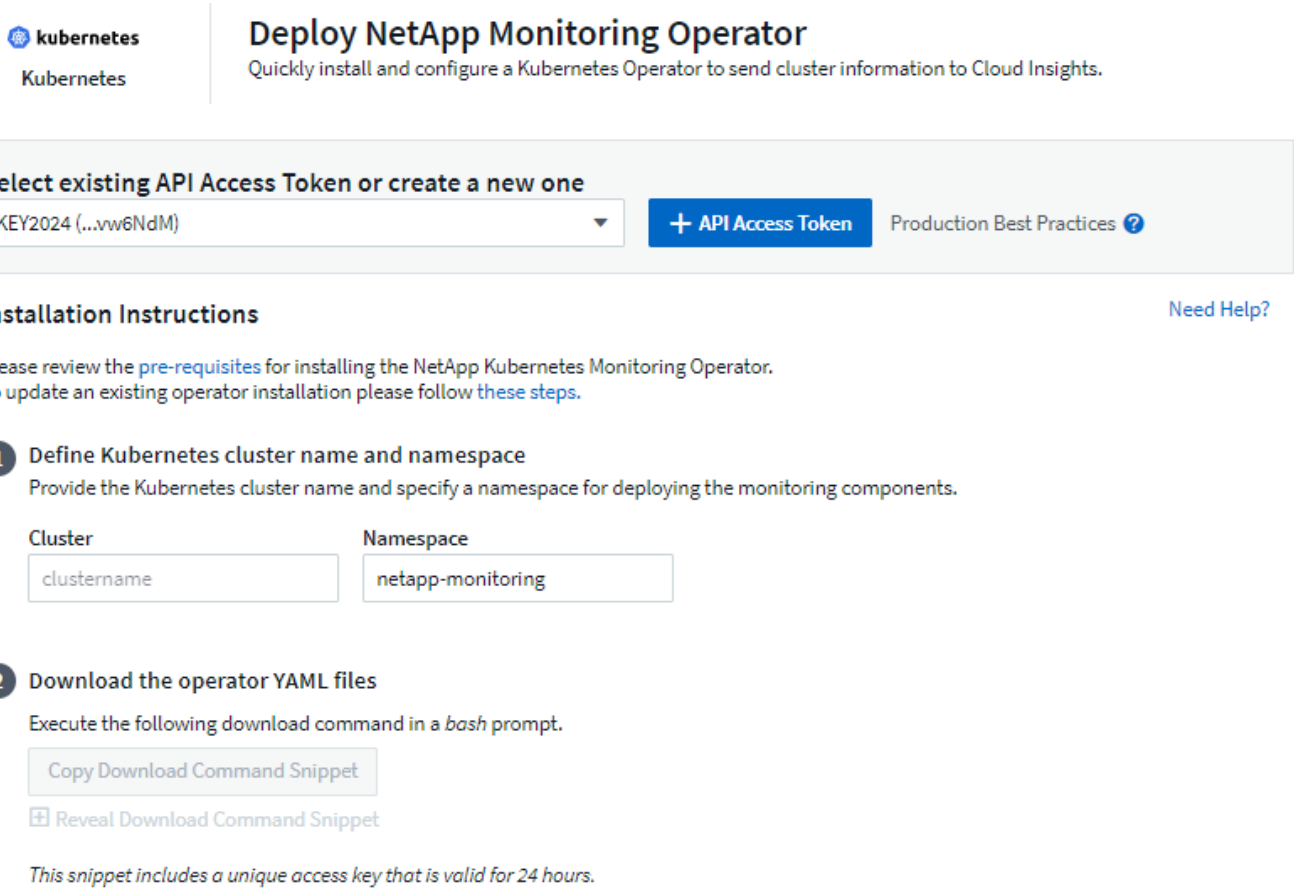
Kubernetes Monitoring Operator Installation and Configuration

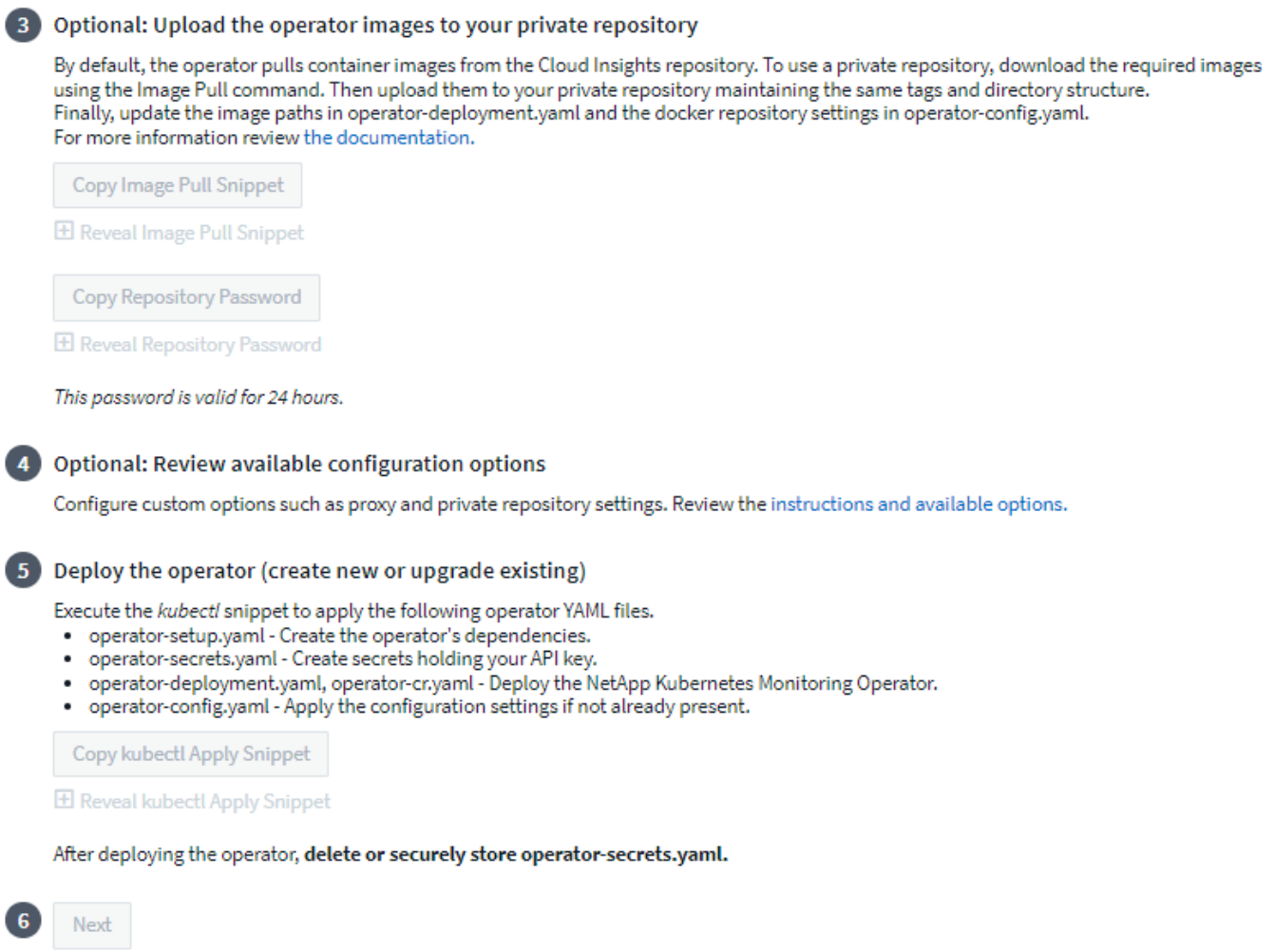
Data Infrastructure Insights offers the **Kubernetes Monitoring Operator** for Kubernetes collection. Navigate to **Kubernetes > Collectors > +Kubernetes Collector** to deploy a new operator.

Before installing the Kubernetes Monitoring Operator

See the Pre-requisites documentation before installing or upgrading the Kubernetes Monitoring Operator.

Installing the Kubernetes Monitoring Operator


kubernetes
Kubernetes
Deploy NetApp Monitoring Operator
Quickly install and configure a Kubernetes Operator to send cluster information to Cloud Insights.
Select existing API Access Token or create a new one
KEY2024 (...vw6NdM)
+ API Access Token
Production Best Practices
Installation Instructions
Need Help?
Please review the pre-requisites for installing the NetApp Kubernetes Monitoring Operator.
To update an existing operator installation please follow these steps.
1 Define Kubernetes cluster name and namespace
Provide the Kubernetes cluster name and specify a namespace for deploying the monitoring components.
Cluster
clustername
Namespace
netapp-monitoring
2 Download the operator YAML files
Execute the following download command in a bash prompt.
Copy Download Command Snippet
Reveal Download Command Snippet
This snippet includes a unique access key that is valid for 24 hours.



Steps to install Kubernetes Monitoring Operator agent on Kubernetes:

1. Enter a unique cluster name and namespace. If you are upgrading from a previous Kubernetes Operator, use the same cluster name and namespace.
2. Once these are entered, you can copy the Download Command snippet to the clipboard.
3. Paste the snippet into a *bash* window and execute it. The Operator installation files will be downloaded. Note that the snippet has a unique key and is valid for 24 hours.
4. If you have a custom or private repository, copy the optional Image Pull snippet, paste it into a *bash* shell and execute it. Once the images have been pulled, copy them to your private repository. Be sure to maintain the same tags and folder structure. Update the paths in *operator-deployment.yaml* as well as the docker repository settings in *operator-config.yaml*.
5. If desired, review available configuration options such as proxy or private repository settings. You can read more about configuration options.
6. When you are ready, deploy the Operator by copying the kubectl Apply snippet, downloading it, and executing it.
7. The installation proceeds automatically. When it is complete, click the *Next* button.
8. When installation is complete, click the *Next* button. Be sure to also delete or securely store the *operator-secrets.yaml* file.

If you are using a proxy, read about configuring proxy.

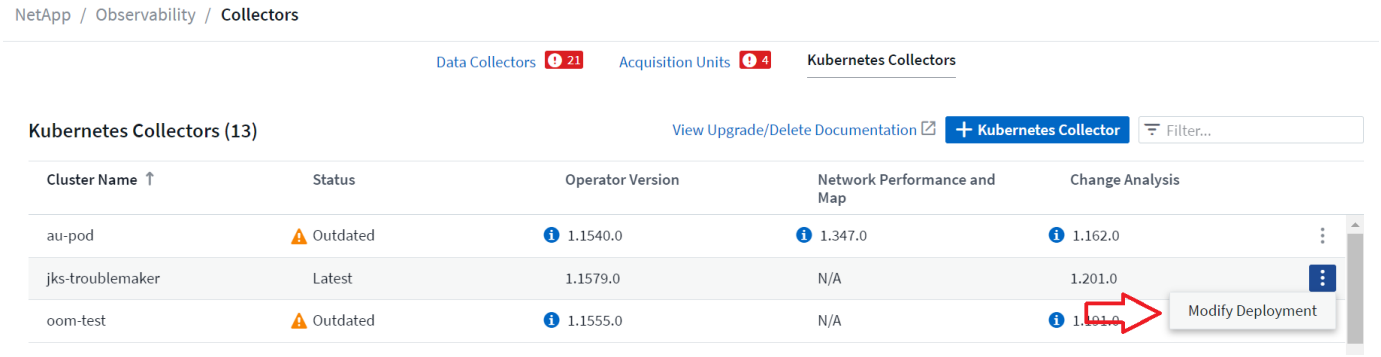
If you are have a custom repository, read about using a custom/private docker repository.

Kubernetes Monitoring Components

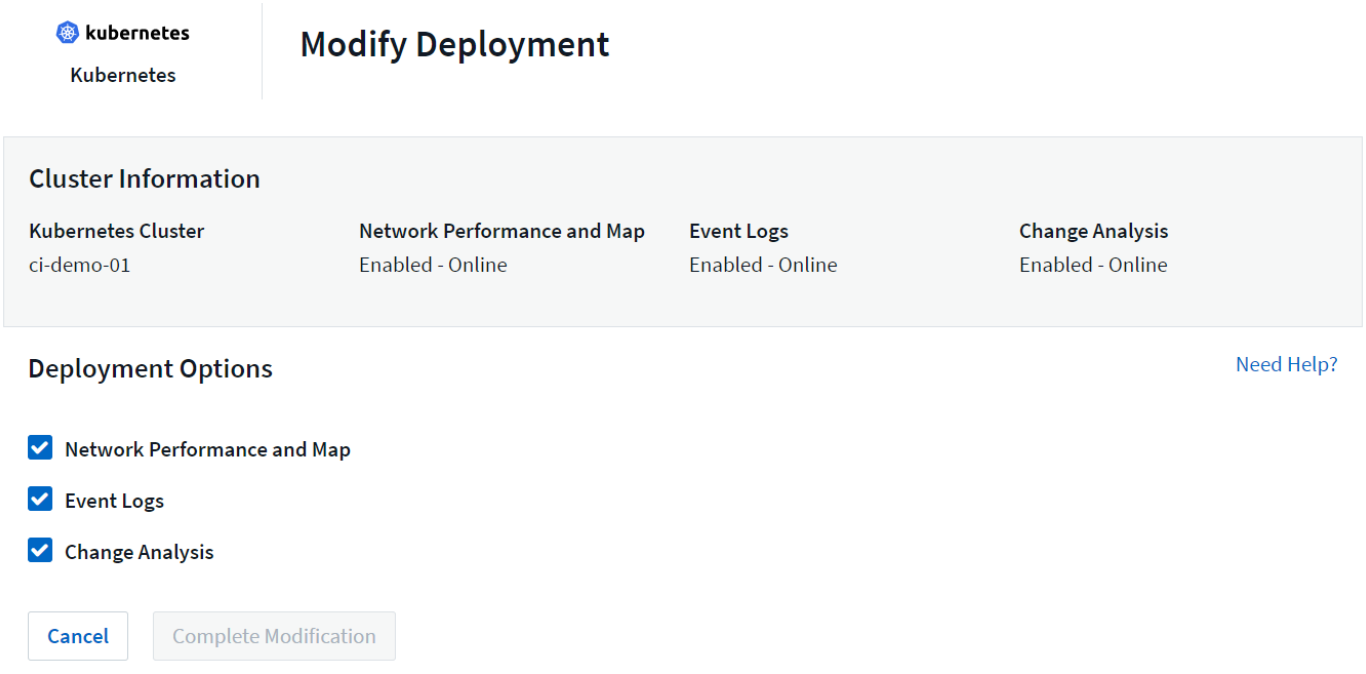
Data Infrastructure Insights Kubernetes Monitoring is comprised of four monitoring components:

- Cluster Metrics
- Network Performance and Map (optional)
- Event Logs (optional)
- Change Analysis (optional)

The optional components above are enabled by default for each Kubernetes collector; if you decide you don't need a component for a particular collector, you can disable it by navigating to **Kubernetes > Collectors** and selecting *Modify Deployment* from the collector's "three dots" menu on the right of the screen.



The screen shows the current state of each component and allows you to disable or enable components for that collector as needed.



Upgrading to the latest Kubernetes Monitoring Operator

Determine whether an AgentConfiguration exists with the existing Operator (if your namespace is not the default *netapp-monitoring*, substitute the appropriate namespace):

```
kubectl -n netapp-monitoring get agentconfiguration netapp-monitoring-
configuration
```

If an AgentConfiguration exists:

- Install the latest Operator over the existing Operator.
  - Ensure you are pulling the latest container images if you are using a custom repository.

If the AgentConfiguration does not exist:

- Make note of your cluster name as recognized by Data Infrastructure Insights (if your namespace is not the default netapp-monitoring, substitute the appropriate namespace):

  ```
  kubectl -n netapp-monitoring get agent -o
  jsonpath='{.items[0].spec.cluster-name}'
  ```

- Create a backup of the existing Operator (if your namespace is not the default netapp-monitoring, substitute the appropriate namespace):

  ```
  kubectl -n netapp-monitoring get agent -o yaml > agent_backup.yaml
  ```

- Uninstall the existing Operator.
- Install the latest Operator.
  - Use the same cluster name.
  - After downloading the latest Operator YAML files, port any customizations found in agent_backup.yaml to the downloaded operator-config.yaml before deploying.
  - Ensure you are pulling the latest container images if you are using a custom repository.

Stopping and Starting the Kubernetes Monitoring Operator

To stop the Kubernetes Monitoring Operator:

```
kubectl -n netapp-monitoring scale deploy monitoring-operator --replicas=0
```

To start the Kubernetes Monitoring Operator:

```
kubectl -n netapp-monitoring scale deploy monitoring-operator --replicas=1
```

Uninstalling

To remove the Kubernetes Monitoring Operator

Note that the default namespace for the Kubernetes Monitoring Operator is "netapp-monitoring". If you have set your own namespace, substitute that namespace in these and all subsequent commands and files.

Newer versions of the monitoring operator can be uninstalled with the following commands:

```
kubectl -n <NAMESPACE> delete agent -l installed-by=nkmo-<NAMESPACE>
kubectl -n <NAMESPACE> delete
clusterrole,clusterrolebinding,crd,svc,deploy,role,rolebinding,secret,sa
-l installed-by=nkmo-<NAMESPACE>
```

If the monitoring operator was deployed in its own dedicated namespace, delete the namespace:

```
kubectl delete ns <NAMESPACE>
```

If the first command returns “No resources found”, use the following instructions to uninstall older versions of the monitoring operator.

Execute each of the following commands in order. Depending on your current installation, some of these commands may return ‘object not found’ messages. These messages may be safely ignored.

```
kubectl -n <NAMESPACE> delete agent agent-monitoring-netapp
kubectl delete crd agents.monitoring.netapp.com
kubectl -n <NAMESPACE> delete role agent-leader-election-role
kubectl delete clusterrole agent-manager-role agent-proxy-role agent-
metrics-reader <NAMESPACE>-agent-manager-role <NAMESPACE>-agent-proxy-role
<NAMESPACE>-cluster-role-privileged
kubectl delete clusterrolebinding agent-manager-rolebinding agent-proxy-
rolebinding agent-cluster-admin-rolebinding <NAMESPACE>-agent-manager-
rolebinding <NAMESPACE>-agent-proxy-rolebinding <NAMESPACE>-cluster-role-
binding-privileged
kubectl delete <NAMESPACE>-psp-nkmo
kubectl delete ns <NAMESPACE>
```

If a Security Context Constraint was previously-created:

```
kubectl delete scc telegraf-hostaccess
```

About Kube-state-metrics

The NetApp Kubernetes Monitoring Operator installs its own kube-state-metrics to avoid conflict with any other instances.

For information about Kube-State-Metrics, see this page.

Configuring/Customizing the Operator

These sections contain information on customizing your operator configuration, working with proxy, using a custom or private docker repository, or working with OpenShift.

Configuration Options

Most commonly modified settings can be configured in the *AgentConfiguration* custom resource. You can edit this resource before deploying the operator by editing the *operator-config.yaml* file. This file includes commented-out examples of settings. See the list of available settings for the most recent version of the operator.

You can also edit this resource after the operator has been deployed by using the following command:

```
kubectl -n netapp-monitoring edit AgentConfiguration
```

To determine if your deployed version of the operator supports AgentConfiguration, run the following command:

```
kubectl get crd agentconfigurations.monitoring.netapp.com
```

If you see an “Error from server (NotFound)” message, your operator must be upgraded before you can use the AgentConfiguration.

Configuring Proxy Support

There are two places where you may use a proxy on your tenant in order to install the Kubernetes Monitoring Operator. These may be the same or separate proxy systems:

- Proxy needed during execution of the installation code snippet (using "curl") to connect the system where the snippet is executed to your Data Infrastructure Insights environment
- Proxy needed by the target Kubernetes cluster to communicate with your Data Infrastructure Insights environment

If you use a proxy for either or both of these, in order to install the Kubernetes Operating Monitor you must first ensure that your proxy is configured to allow good communication to your Data Infrastructure Insights environment. If you have a proxy and can access Data Infrastructure Insights from the server/VM from which you wish to install the Operator, then your proxy is likely configured properly.

For the proxy used to install the Kubernetes Operating Monitor, before installing the Operator, set the *http_proxy/https_proxy* environment variables. For some proxy environments, you may also need to set the *no_proxy environment* variable.

To set the variable(s), perform the following steps on your system **before** installing the Kubernetes Monitoring Operator:

1. Set the *https_proxy* and/or *http_proxy* environment variable(s) for the current user:
    a. If the proxy being setup does not have Authentication (username/password), run the following

command:

```
export https_proxy=<proxy_server>:<proxy_port>
```

b. If the proxy being setup does have Authentication (username/password), run this command:

```
export
http_proxy=<proxy_username>:<proxy_password>@<proxy_server>:<proxy_po
rt>
```

For the proxy used for your Kubernetes cluster to communicate with your Data Infrastructure Insights environment, install the Kubernetes Monitoring Operator after reading all of these instructions.

Configure the proxy section of AgentConfiguration in operator-config.yaml before deploying the Kubernetes Monitoring Operator.

```
agent:
  ...
  proxy:
    server: <server for proxy>
    port: <port for proxy>
    username: <username for proxy>
    password: <password for proxy>

    # In the noproxy section, enter a comma-separated list of
    # IP addresses and/or resolvable hostnames that should bypass
    # the proxy
    noproxy: <comma separated list>

    isTelegrafProxyEnabled: true
    isFluentbitProxyEnabled: <true or false> # true if Events Log enabled
    isCollectorsProxyEnabled: <true or false> # true if Network
Performance and Map enabled
    isAuProxyEnabled: <true or false> # true if AU enabled
  ...
...
```

Using a custom or private docker repository

By default, the Kubernetes Monitoring Operator will pull container images from the Data Infrastructure Insights repository. If you have a Kubernetes cluster used as the target for monitoring, and that cluster is configured to only pull container images from a custom or private Docker repository or container registry, you must configure access to the containers needed by the Kubernetes Monitoring Operator.

Run the “Image Pull Snippet” from the NetApp Monitoring Operator install tile. This command will log into the

Data Infrastructure Insights repository, pull all image dependencies for the operator, and log out of the Data Infrastructure Insights repository. When prompted, enter the provided repository temporary password. This command downloads all images used by the operator, including for optional features. See below for which features these images are used for.

Core Operator Functionality and Kubernetes Monitoring

- netapp-monitoring
- ci-kube-rbac-proxy
- ci-ksm
- ci-telegraf
- distroless-root-user

Events Log

- ci-fluent-bit
- ci-kubernetes-event-exporter

Network Performance and Map

- ci-net-observer

Push the operator docker image to your private/local/enterprise docker repository according to your corporate policies. Ensure that the image tags and directory paths to these images in your repository are consistent with those in the Data Infrastructure Insights repository.

Edit the monitoring-operator deployment in operator-deployment.yaml, and modify all image references to use your private Docker repository.

```
image: <docker repo of the enterprise/corp docker repo>/ci-kube-rbac-
proxy:<ci-kube-rbac-proxy version>
image: <docker repo of the enterprise/corp docker repo>/netapp-
monitoring:<version>
```

Edit the AgentConfiguration in operator-config.yaml to reflect the new docker repo location. Create a new imagePullSecret for your private repository, for more details see *https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/*

```
agent:
  ...
  # An optional docker registry where you want docker images to be pulled
from as compared to CI's docker registry
  # Please see documentation link here:
xref:{relative_path}task_config_telegraf_agent_k8s.html#using-a-custom-or-
private-docker-repository
  dockerRepo: your.docker.repo/long/path/to/test
  # Optional: A docker image pull secret that maybe needed for your
private docker registry
  dockerImagePullSecret: docker-secret-name
```

OpenShift Instructions

If you are running on OpenShift 4.6 or higher, you must edit the AgentConfiguration in *operator-config.yaml* to enable the *runPrivileged* setting:

```
# Set runPrivileged to true SELinux is enabled on your kubernetes nodes
runPrivileged: true
```

Openshift may implement an added level of security that may block access to some Kubernetes components.

Tolerations and Taints

The *netapp-ci-telegraf-ds*, *netapp-ci-fluent-bit-ds*, and *netapp-ci-net-observer-l4-ds* DaemonSets must schedule a pod on every node in your cluster in order to correctly collect data on all nodes. The operator has been configured to tolerate some well known **taints**. If you have configured any custom taints on your nodes, thus preventing pods from running on every node, you can create a **toleration** for those taints in the *AgentConfiguration*. If you have applied custom taints to all nodes in your cluster, you must also add the necessary tolerations to the operator deployment to allow the operator pod to be scheduled and executed.

Learn More about Kubernetes Taints and Tolerations.

Return to the **NetApp Kubernetes Monitoring Operator Installation** page

A Note About Secrets

To remove permission for the Kubernetes Monitoring Operator to view secrets cluster-wide, delete the following resources from the *operator-setup.yaml* file before installing:

```
ClusterRole/netapp-ci-<namespace>-agent-secret-clusterrole
ClusterRoleBinding/netapp-ci-<namespace>-agent-secret-clusterrolebinding
```

If this is an upgrade, also delete the resources from your cluster:

```
 kubectl delete ClusterRole/netapp-ci-<namespace>-agent-secret-clusterrole
 kubectl delete ClusterRoleBinding/netapp-ci-<namespace>-agent-secret-
clusterrolebinding
```

If Change Analysis is enabled, modify the *AgentConfiguration* or *operator-config.yaml* to uncomment the change-management section and include *kindsToIgnoreFromWatch: '"secrets"'* under the change-management section. Note the presence and position of single and double quotes in this line.

```
# change-management:
  ...
  # # A comma separated list of kinds to ignore from watching from the
default set of kinds watched by the collector
  # # Each kind will have to be prefixed by its apigroup
  # # Example: '"networking.k8s.io.networkpolicies,batch.jobs",
"authorization.k8s.io.subjectaccessreviews"'
  kindsToIgnoreFromWatch: '"secrets"'
  ...
```

Verifying Kubernetes Monitoring Operator Image Signatures

The image for the operator and all related images it deploys are signed by NetApp. You can manually verify the images before installation using the cosign tool, or configure a Kubernetes admission controller. For more details please see the Kubernetes documentation.

The public key used to verify the image signatures is available in the Monitoring Operator install tile under *Optional: Upload the operator images to your private repository > Image Signature Public Key*

To manually verify an image signature, perform the following steps:

1. Copy and run the Image Pull Snippet
2. Copy and enter the Repository Password when prompted
3. Store the Image Signature Public Key (dii-image-signing.pub in the example)
4. Verify the images using cosign. Refer to the following example of cosign usage

```
$ cosign verify --key dii-image-signing.pub --insecure-ignore-sct
--insecure-ignore-tlog <repository>/<image>:<tag>
Verification for <repository>/<image>:<tag> --
The following checks were performed on each of these signatures:
  - The cosign claims were validated
  - The signatures were verified against the specified public key
[{"critical":{"identity":{"docker-
reference":"<repository>/<image>"},"image":{"docker-manifest-
digest":"sha256:<hash>"},"type":"cosign container image
signature"},"optional":null}]
```

Troubleshooting

Some things to try if you encounter problems setting up the Kubernetes Monitoring Operator:

Problem:	Try this:
I do not see a hyperlink/connection between my Kubernetes Persistent Volume and the corresponding back-end storage device. My Kubernetes Persistent Volume is configured using the hostname of the storage server.	Follow the steps to uninstall the existing Telegraf agent, then re-install the latest Telegraf agent. You must be using Telegraf version 2.0 or later, and your Kubernetes cluster storage must be actively monitored by Data Infrastructure Insights.

Problem:	Try this:
I’m seeing messages in the logs resembling the following: E0901 15:21:39.962145 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: Failed to list *v1.MutatingWebhookConfiguration: the server could not find the requested resource E0901 15:21:43.168161 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: Failed to list *v1.Lease: the server could not find the requested resource (get leases.coordination.k8s.io) etc.	These messages may occur if you are running kube-state-metrics version 2.0.0 or above with Kubernetes versions below 1.20. To get the Kubernetes version: *kubectl version* To get the kube-state-metrics version: *kubectl get deploy/kube-state-metrics -o jsonpath='{..image}'* To prevent these messages from happening, users can modify their kube-state-metrics deployment to disable the following Leases: *mutatingwebhookconfigurations validatingwebhookconfigurations volumeattachments resources* More specifically, they can use the following CLI argument: resources=certificatesigningrequests,configmaps,cronjobs,daemonsets, deployments,endpoints,horizontalpodautoscalers,ingresses,jobs,limitranges, namespaces,networkpolicies,nodes,persistentvolumeclaims,persistentvolumes, poddisruptionbudgets,pods,replicasets,replicationcontrollers,resourcequotas, secrets,services,statefulsets,storageclasses The default resource list is: "certificatesigningrequests,configmaps,cronjobs,daemonsets,deployments, endpoints,horizontalpodautoscalers,ingresses,jobs,leases,limitranges, mutatingwebhookconfigurations,namespaces,networkpolicies,nodes, persistentvolumeclaims,persistentvolumes,poddisruptionbudgets,pods,replicasets, replicationcontrollers,resourcequotas,secrets,services,statefulsets,storageclasses, validatingwebhookconfigurations,volumeattachments"

Problem:	Try this:
I see error messages from Telegraf resembling the following, but Telegraf does start up and run: Oct 11 14:23:41 ip-172-31-39-47 systemd[1]: Started The plugin-driven server agent for reporting metrics into InfluxDB. Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="failed to create cache directory. /etc/telegraf/.cache/snowflake, err: mkdir /etc/telegraf/.ca che: permission denied. ignored\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="failed to open. Ignored. open /etc/telegraf/.cache/snowflake/ocsp_response_cache.json: no such file or directory\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: 2021-10-11T14:23:41Z I! Starting Telegraf 1.19.3	This is a known issue. Refer to This GitHub article for more details. As long as Telegraf is up and running, users can ignore these error messages.
On Kubernetes, my Telegraf pod(s) are reporting the following error: "Error in processing mountstats info: failed to open mountstats file: /hostfs/proc/1/mountstats, error: open /hostfs/proc/1/mountstats: permission denied"	If SELinux is enabled and enforcing, it is likely preventing the Telegraf pod(s) from accessing the /proc/1/mountstats file on the Kubernetes node. To overcome this restriction, edit the agentconfiguration, and enable the runPrivileged setting. For more details, refer to the OpenShift Instructions.
On Kubernetes, my Telegraf ReplicaSet pod is reporting the following error: [inputs.prometheus] Error in plugin: could not load keypair /etc/kubernetes/pki/etcd/server.crt:/etc/kubernetes/pki/etcd/server.key: open /etc/kubernetes/pki/etcd/server.crt: no such file or directory	The Telegraf ReplicaSet pod is intended to run on a node designated as a master or for etcd. If the ReplicaSet pod is not running on one of these nodes, you will get these errors. Check to see if your master/etcd nodes have taints on them. If they do, add the necessary tolerations to the Telegraf ReplicaSet, telegraf-rs. For example, edit the ReplicaSet… kubectl edit rs telegraf-rs …and add the appropriate tolerations to the spec. Then, restart the ReplicaSet pod.

Problem:	Try this:
I have a PSP/PSA environment. Does this affect my monitoring operator?	If your Kubernetes cluster is running with Pod Security Policy (PSP) or Pod Security Admission (PSA) in place, you must upgrade to the latest Kubernetes Monitoring Operator. Follow these steps to upgrade to the current Operator with support for PSP/PSA: 1. Uninstall the previous monitoring operator: kubectl delete agent agent-monitoring-netapp -n netapp-monitoring kubectl delete ns netapp-monitoring kubectl delete crd agents.monitoring.netapp.com kubectl delete clusterrole agent-manager-role agent-proxy-role agent-metrics-reader kubectl delete clusterrolebinding agent-manager-rolebinding agent-proxy-rolebinding agent-cluster-admin-rolebinding 2. Install the latest version of the monitoring operator.
I ran into issues trying to deploy the Operator, and I have PSP/PSA in use.	1. Edit the agent using the following command: kubectl -n <name-space> edit agent 2. Mark 'security-policy-enabled' as 'false'. This will disable Pod Security Policies and Pod Security Admission and allow the Operator to deploy. Confirm by using the following commands: kubectl get psp (should show Pod Security Policy removed) kubectl get all -n <namespace> | grep -i psp (should show that nothing is found)
"ImagePullBackoff" errors seen	These errors may be seen if you have a custom or private docker repository and have not yet configured the Kubernetes Monitoring Operator to properly recognize it. Read more about configuring for custom/private repo.

Problem:	Try this:

I am having an issue with my monitoring-operator deployment, and the current documentation does not help me resolve it.	Capture or otherwise note the output from the following commands, and contact the Technical Support team.
	
	`kubectl -n netapp-monitoring get all`
	`kubectl -n netapp-monitoring describe all`
	`kubectl -n netapp-monitoring logs <monitoring-operator-pod> --all-containers=true`
	`kubectl -n netapp-monitoring logs <telegraf-pod> --all-containers=true`

net-observer (Workload Map) pods in Operator namespace are in CrashLoopBackOff	These pods correspond to Workload Map data collector for Network Observability. Try these: • Check the logs of one of the pods to confirm minimum kernel version. For example: --- {"ci-tenant-id":"your-tenant-id","collector-cluster":"your-k8s-cluster-name","environment":"prod","level":"error","msg":"failed in validation. Reason: kernel version 3.10.0 is less than minimum kernel version of 4.18.0","time":"2022-11-09T08:23:08Z"} --- • Net-observer pods requires the Linux kernel version to be at least 4.18.0. Check the kernel version using the command “uname -r” and ensure they are >= 4.18.0
Pods are running in Operator namespace (default: netapp-monitoring), but no data is shown in UI for workload map or Kubernetes metrics in Queries	Check the time setting on the nodes of the K8S cluster. For accurate audit and data reporting, it is strongly recommended to synchronize the time on the Agent machine using Network Time Protocol (NTP) or Simple Network Time Protocol (SNTP).
Some of the net-observer pods in Operator namespace are in Pending state	Net-observer is a DaemonSet and runs a pod in each Node of the k8s cluster. • Note the pod which is in Pending state, and check if it is experiencing a resource issue for CPU or memory. Ensure the required memory and CPU is available in the node.

Problem:	Try this:
I'm seeing the following in my logs immediately after installing the Kubernetes Monitoring Operator: [inputs.prometheus] Error in plugin: error making HTTP request to http://kube-state-metrics.<namespace>.svc.cluster.local:8080/metrics: Get http://kube-state-metrics.<namespace>.svc.cluster.local:8080/metrics: dial tcp: lookup kube-state-metrics.<namespace>.svc.cluster.local: no such host	This message is typically only seen when a new operator is installed and the *telegraf-rs* pod is up before the *ksm* pod is up. These messages should stop once all pods are running.
I do see not any metrics being collected for the Kubernetes CronJobs that exist in my cluster.	Verify your Kubernetes version (i.e. `kubectl version`). If it is v1.20.x or below, this is an expected limitation. The kube-state-metrics release deployed with the Kubernetes Monitoring Operator only supports v1.CronJob. With Kubernetes 1.20.x and below, the CronJob resource is at v1beta.CronJob. As a result, kube-state-metrics cannot find the CronJob resource.
After installing the operator, the telegraf-ds pods enter CrashLoopBackOff and the pod logs indicate "su: Authentication failure".	Edit the telegraf section in *AgentConfiguration*, and set *dockerMetricCollectionEnabled* to false. For more details, refer to the operator's configuration options. … spec: … telegraf: … - name: docker run-mode: - DaemonSet substitutions: - key: DOCKER_UNIX_SOCK_PLACEHOLDER value: unix:///run/docker.sock … …
I see repeating error messages resembling the following in my Telegraf logs: E! [agent] Error writing to outputs.http: Post "https://<tenant_url>/rest/v1/lake/ingest/influxdb": context deadline exceeded (Client.Timeout exceeded while awaiting headers)	Edit the telegraf section in *AgentConfiguration*, and increase *outputTimeout* to 10s. For more details, refer to the operator's configuration options.
I'm missing *involvedobject* data for some Event Logs.	Be sure you have followed the steps in the Permissions section above.

Problem:	Try this:
Why am I seeing two monitoring operator pods running, one named netapp-ci-monitoring-operator-<pod> and the other named monitoring-operator-<pod>?	As of October 12, 2023, Data Infrastructure Insights has refactored the operator to better serve our users; for those changes to be fully adopted, you must remove the old operator and install the new one.

My kubernetes events unexpectedly stopped reporting to Data Infrastructure Insights.	Retrieve the name of the event-exporter pod:
	`` `kubectl -n netapp-monitoring get pods |grep event-exporter |awk '{print $1}' |sed 's/event-exporter./event-exporter/'` ``
	It should be either "netapp-ci-event-exporter" or "event-exporter". Next, edit the monitoring agent `kubectl -n netapp-monitoring edit agent`, and set the value for LOG_FILE to reflect the appropriate event-exporter pod name found in the previous step. More specifically, LOG_FILE should be set to either "/var/log/containers/netapp-ci-event-exporter.log" or "/var/log/containers/event-exporter*.log"
	`fluent-bit:`
	`...`
	`- name: event-exporter-ci`
	`substitutions:`
	`- key: LOG_FILE`
	`values:`
	`- /var/log/containers/netapp-ci-event-exporter*.log`
	`...`
	Alternatively, one can also uninstall and reinstall the agent.

I'm seeing pod(s) deployed by the Kubernetes Monitoring Operator crash because of insufficient resources.	Refer to the Kubernetes Monitoring Operator configuration options to increase the CPU and/or memory limits as needed.
A missing image or invalid configuration caused the netapp-ci-kube-state-metrics pods to fail to startup or become ready. Now the StatefulSet is stuck and configuration changes are not being applied to the netapp-ci-kube-state-metrics pods.	The StatefulSet is in a broken state. After fixing any configuration problems bounce the netapp-ci-kube-state-metrics pods.

Problem:	Try this:
netapp-ci-kube-state-metrics pods fail to start after running a Kubernetes Operator upgrade, throwing ErrImagePull (failing to pull the image).	Try resetting the pods manually.
"Event discarded as being older then maxEventAgeSeconds" messages are being observed for my Kubernetes cluster under Log Analysis.	Modify the Operator *agentconfiguration* and increase the *event-exporter-maxEventAgeSeconds* (i.e. to 60s), *event-exporter-kubeQPS* (i.e. to 100), and *event-exporter-kubeBurst* (i.e. to 500). For more details on these configuration options, see the configuration options page.
Telegraf warns of, or crashes because of, insufficient lockable memory.	Try increasing the limit of lockable memory for Telegraf in the underlying operating system/node. If increasing the limit is not an option, modify the NKMO agentconfiguration and set *unprotected* to *true*. This will instruct Telegraf to no attempt to reserve locked memory pages. While this can pose a security risk as decrypted secrets might be swapped out to disk, it allows for execution in environments where reserving locked memory is not possible. For more details on the *unprotected* configuration options, refer to the configuration options page.
I see warning messages from Telegraf resembling the following: *W! [inputs.diskio] Unable to gather disk name for "vdc": error reading /dev/vdc: no such file or directory*	For the Kubernetes Monitoring Operator, these warning message are benign and can be safely ignored. Alternatively, edit the telegraf section in AgentConfiguration, and set *runDsPrivileged* to true. For more details, refer to the operator's configuration options.

Problem:	Try this:

My fluent-bit pod is failing with the following errors:
 	
[2024/10/16 14:16:23] [error] [/src/fluent-bit/plugins/in_tail/tail_fs_inotify.c:360 errno=24] Too many open files
[2024/10/16 14:16:23] [error] failed initialize input tail.0
[2024/10/16 14:16:23] [error] [engine] input initialization failed
	Try to change your *fsnotify* settings in your cluster:
	
	`sudo sysctl fs.inotify.max_user_instances (take note of setting)`
	
	`sudo sysctl fs.inotify.max_user_instances=<something larger than current setting>`
	
	`sudo sysctl fs.inotify.max_user_watches (take note of setting)`
	
	`sudo sysctl fs.inotify.max_user_watches=<something larger than current setting>`
	
	Restart Fluent-bit.
	
	Note: to make these settings persistent across node restarts, you need to put the following lines in */etc/sysctl.conf*
	
	`fs.inotify.max_user_instances=<something larger than current setting>`
	`fs.inotify.max_user_watches=<something larger than current setting>`

The telegraf DS pods are reporting errors pertaining to the kubernetes input plugin failing to make HTTP requests due to the inability to validate the TLS certificate. For example: E! [inputs.kubernetes] Error in plugin: error making HTTP request to "https://<kubelet_IP>:10250/stats/summary": Get "https://<kubelet_IP>:10250/stats/summary": tls: failed to verify certificate: x509: cannot validate certificate for <kubelet_IP> because it doesn't contain any IP SANs	This will occur if the kubelet is using self-signed certificates, and/or the specified certificate does not include the <kubelet_IP> in the certificates *Subject Alternative Name* list. To resolve this, the user can modify the agent configuration, and set *telegraf:insecureK8sSkipVerify* to *true*. This will configure the telegraf input plugin to skip verification. Alternatively, the user can configure the kubelet for serverTLSBootstrap, which will trigger a certificate request from the 'certificates.k8s.io' API.

Additional information may be found from the Support page or in the Data Collector Support Matrix.

Memcached Data Collector

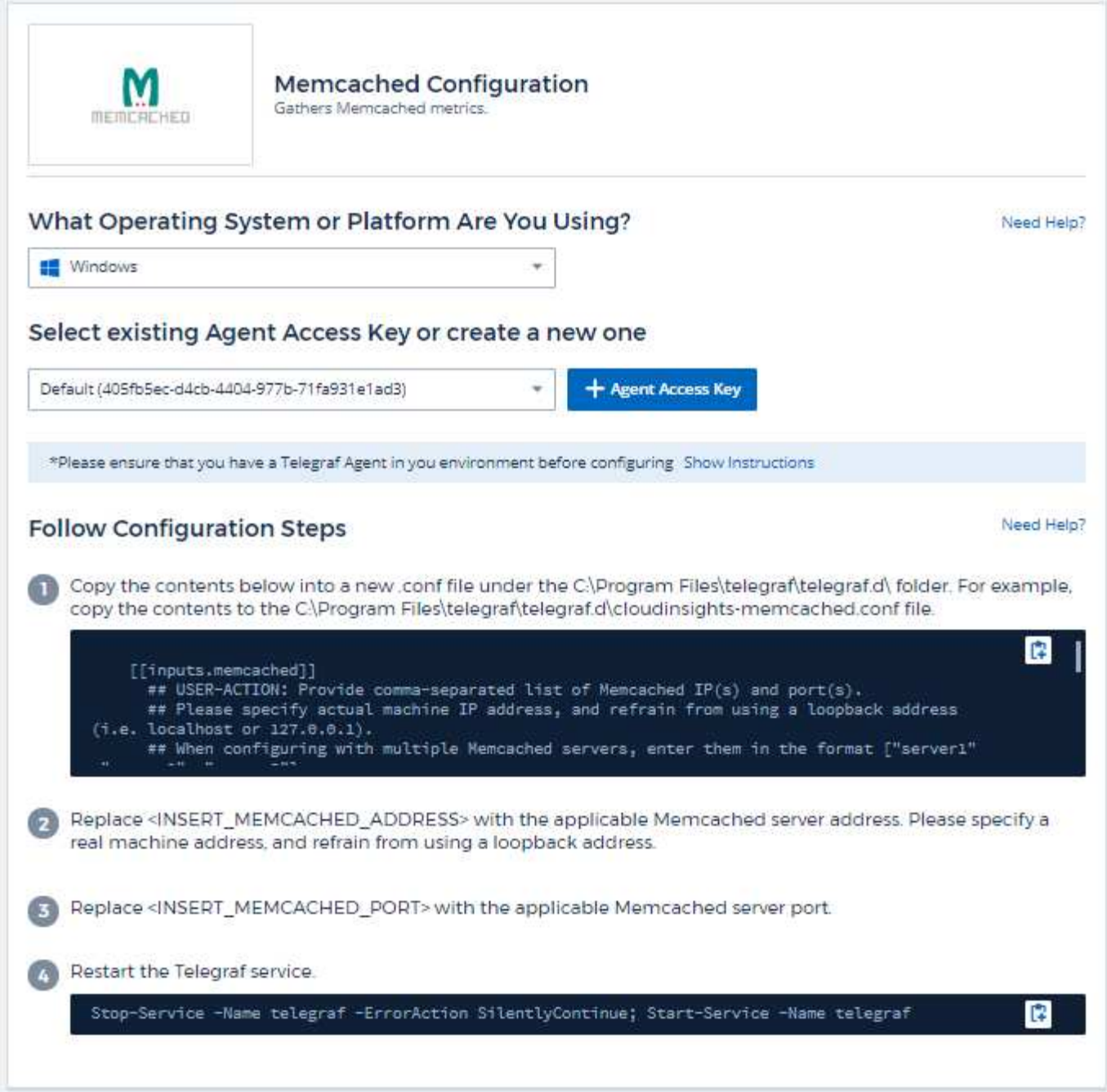
Data Infrastructure Insights uses this data collector to gather metrics from Memcached.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Memcached.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Memcached wiki.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Memcached	Namespace Server	Node IP Node Name	Accepting Connections Handled Authentication Requests Failed Authentications Bytes Used Bytes Read (per sec) Bytes Written (per sec) CAS Badval CAS Hits CAS Misses Flush Reqs (per sec) Get Reqs (per sec) Set Reqs (per sec) Touch Reqs (per sec) Connection Yields (per sec) Connection Structures Open Connections Current Stored Items Decr Requests Hits (per sec) Decr Requests Misses (per sec) Delete Requests Hits (per sec) Delete Requests Misses (per sec) Items Evicted Valid Evictions Expired Items Get Hits (per sec) Get Misses (per sec) Used Hash Bytes Hash Is Expanding Hash Power Level Incr Requests Hits (per sec) Incr Requests Misses (per sec) Server Max Bytes Listen Disabled Num Reclaimed Worker Threads Count Total Opened Connections Total Items Stored Touch Hits Touch Misses Server Uptime

Troubleshooting

Additional information may be found from the Support page.

MongoDB Data Collector

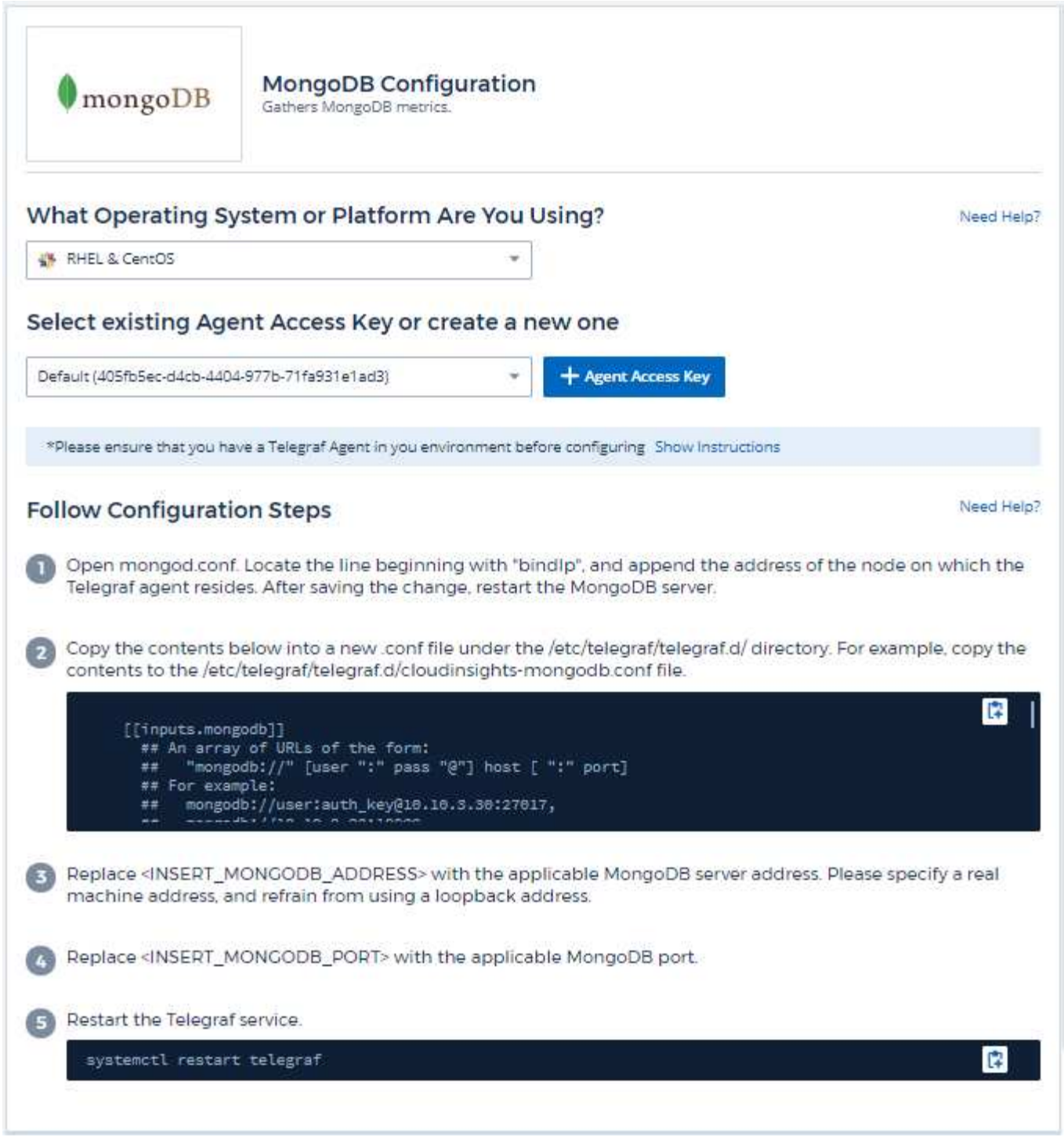
Data Infrastructure Insights uses this data collector to gather metrics from MongoDB.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose MongoDB.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the MongoDB documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
MongoDB	Namespace Hostname		
MongoDB Database	Namespace Hostname Database name		

Troubleshooting

Information may be found from the Support page.

MySQL Data Collector

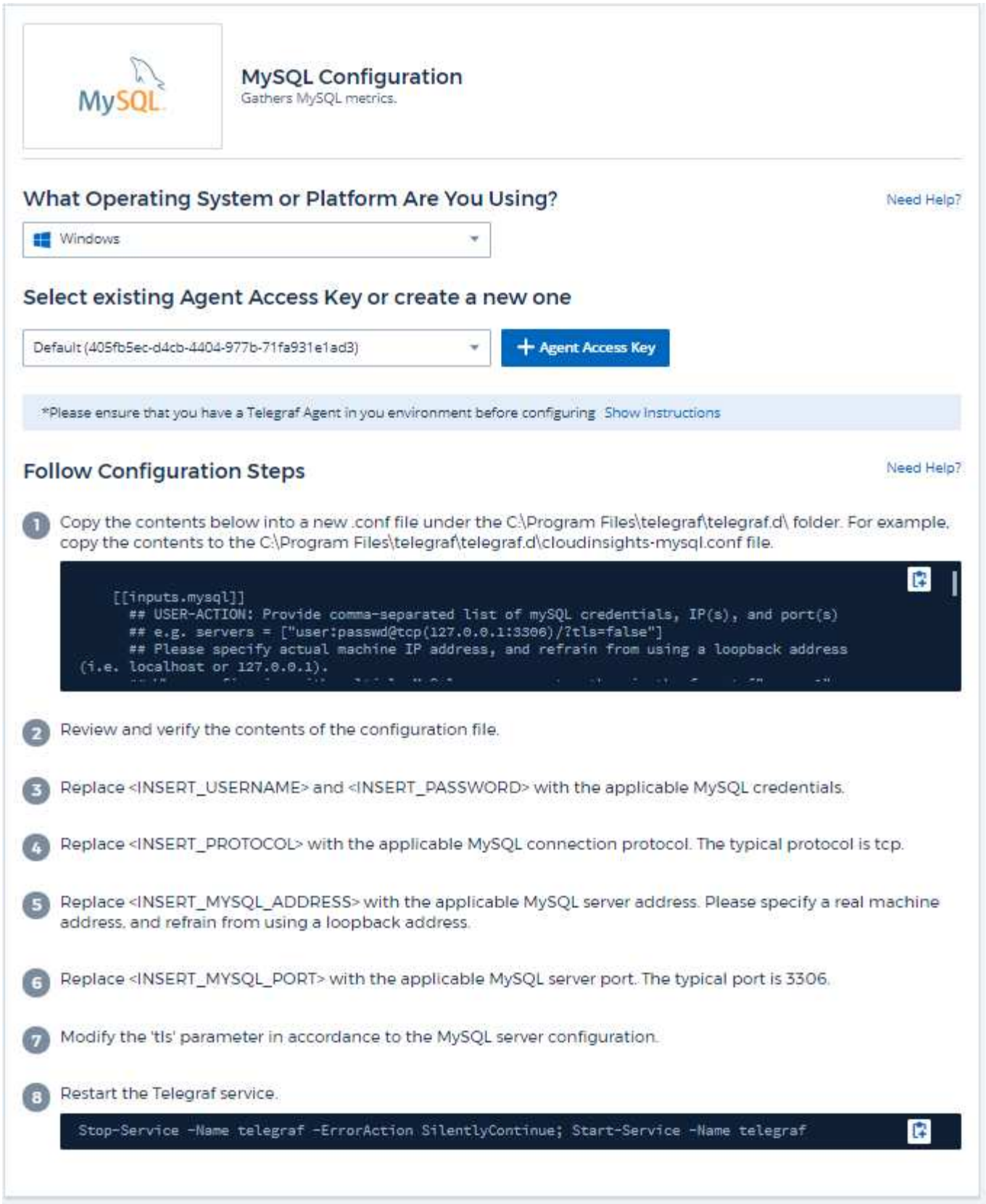
Data Infrastructure Insights uses this data collector to gather metrics from MySQL.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose MySQL.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the MySQL documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
MySQL	Namespace MySQL Server	Node IP Node Name	Aborted Clients (per sec) Aborted Connects (per sec) RX Bytes (per sec) TX Bytes (per sec) Commands Admin (per sec) Commands Alter Event Commands Alter Function Commands Alter Instance Commands Alter Procedure Commands Alter Server Commands Alter Table Commands Alter Tablespace Commands Alter User Commands Analyze Commands Assign To Keycache Commands Begin Commands Binlog Commands Call Procedure Commands Change DB Commands Change Master Commands Change Repl Filter Commands Check Commands Checksum Commands Commit Commands Create DB Commands Create Event Commands Create Function Commands Create Index Commands Create Procedure Commands Create Server Commands Create Table Commands Create Trigger Commands Create UDF Commands Create User Commands Create View Commands Dealloc SQL Connection Errors Accept Created Tmp Disk Tables Delayed Errors Flush Commands Handler Commit Innodb Buffer Pool Bytes Data Key Blocks Not Flushed

Troubleshooting

Additional information may be found from the Support page.

Netstat Data Collector

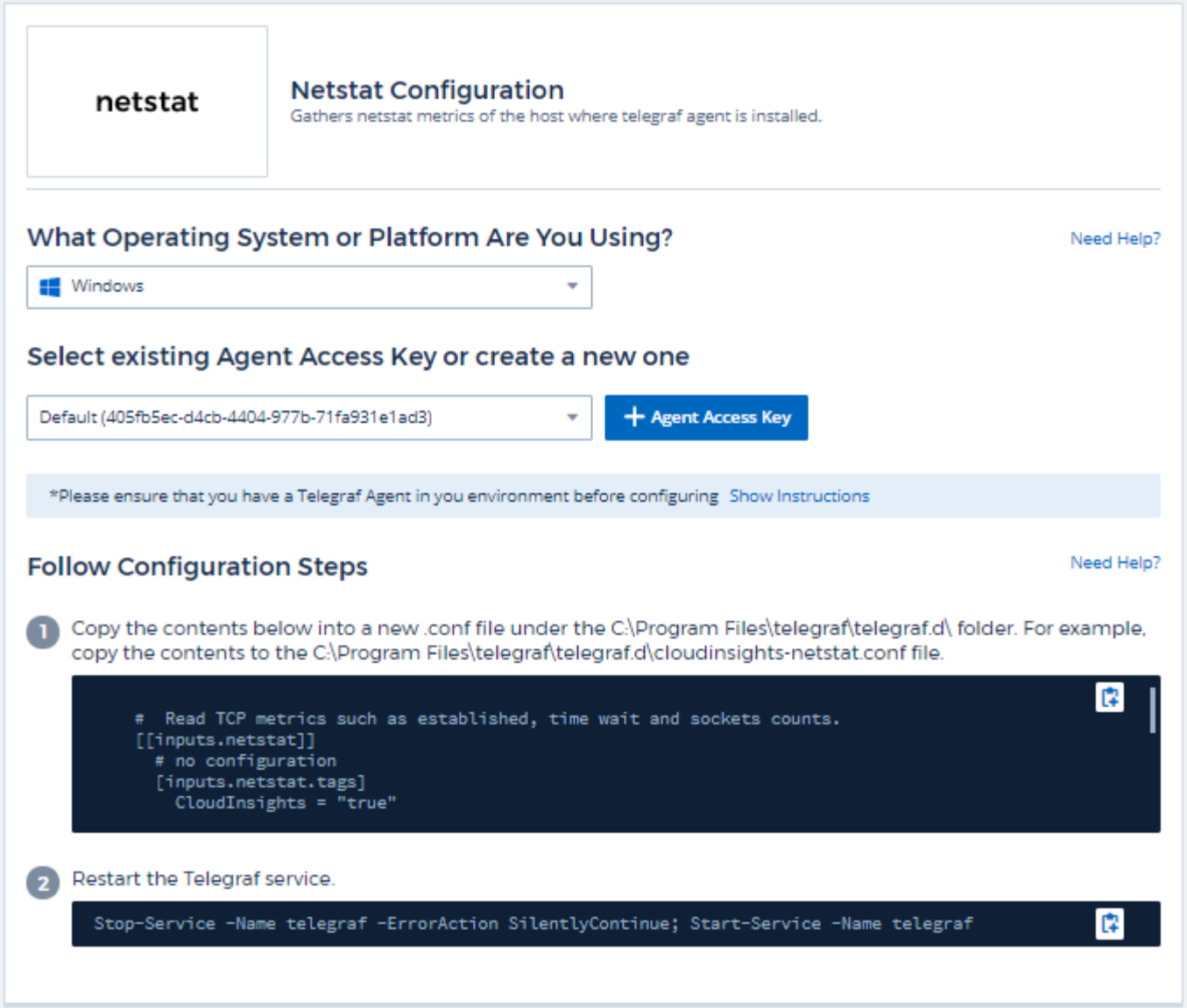
Data Infrastructure Insights uses this data collector to gather Netstat metrics.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Netstat.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Netstat	Node UUID	Node IP Node Name	

Troubleshooting

Additional information may be found from the Support page.

Nginx Data Collector

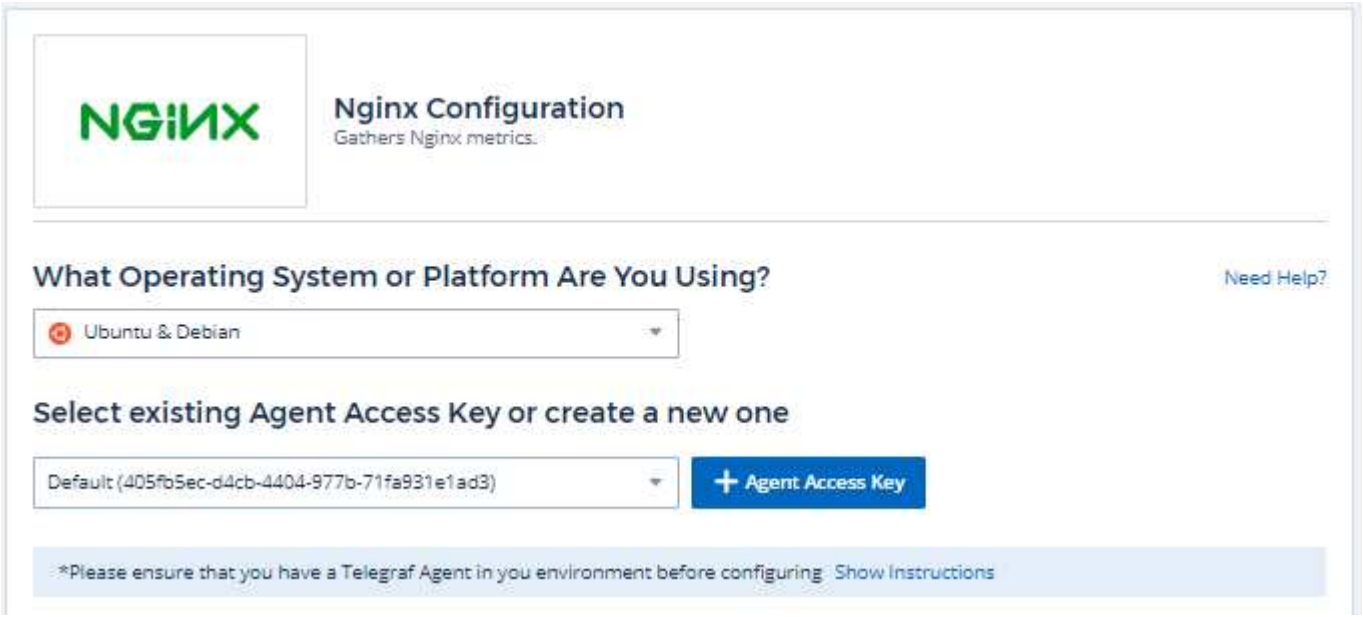
Data Infrastructure Insights uses this data collector to gather metrics from Nginx.

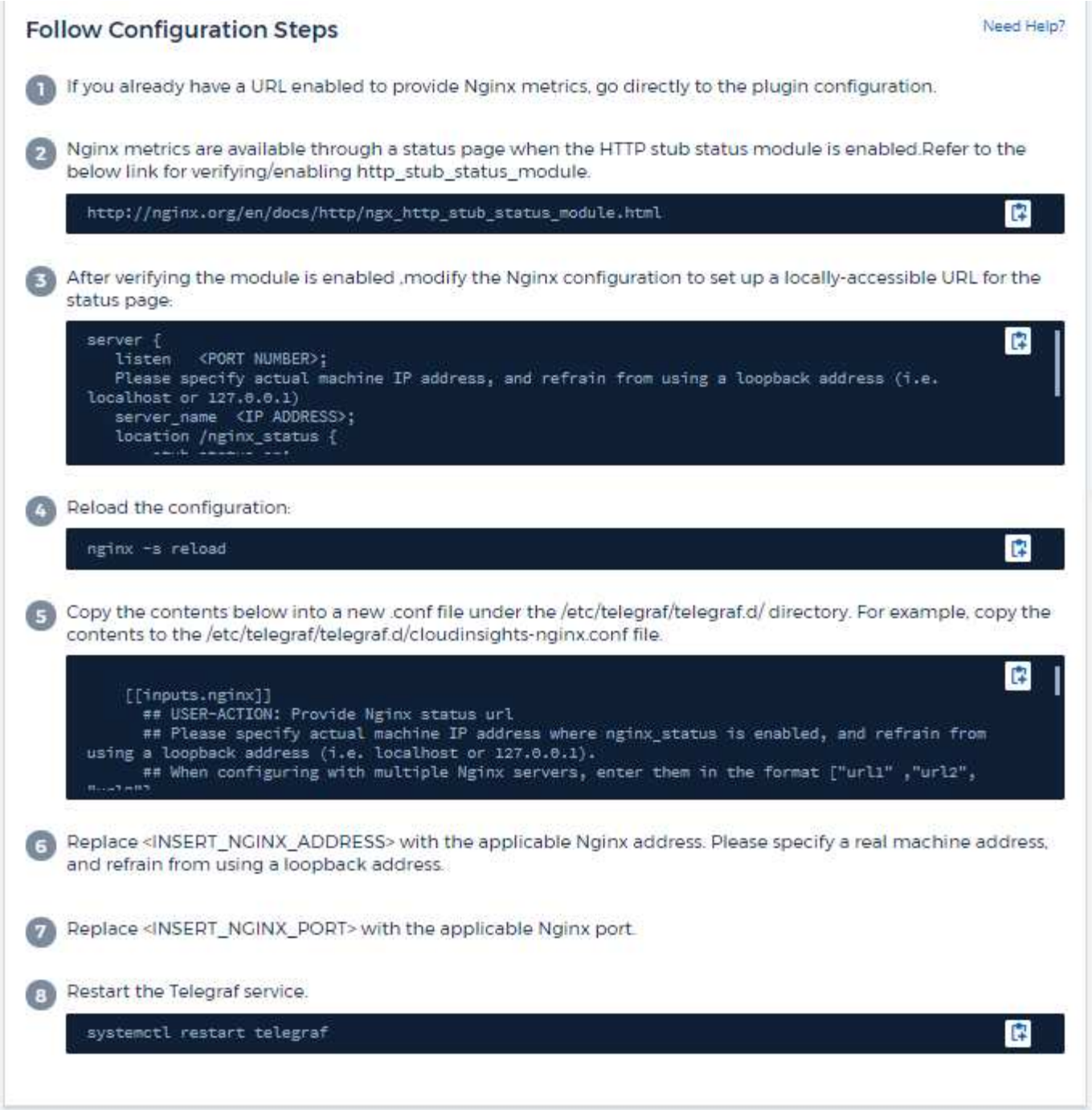
Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Nginx.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.





Setup

Nginx metric collection requires that Nginx http_stub_status_module be enabled.

Additional information may be found in the Nginx documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Nginx	Namespace Server	Node IP Node Name Port	Accepts Active Handled Reading Requests Waiting Writing

Troubleshooting

Additional information may be found from the Support page.

PostgreSQL Data Collector

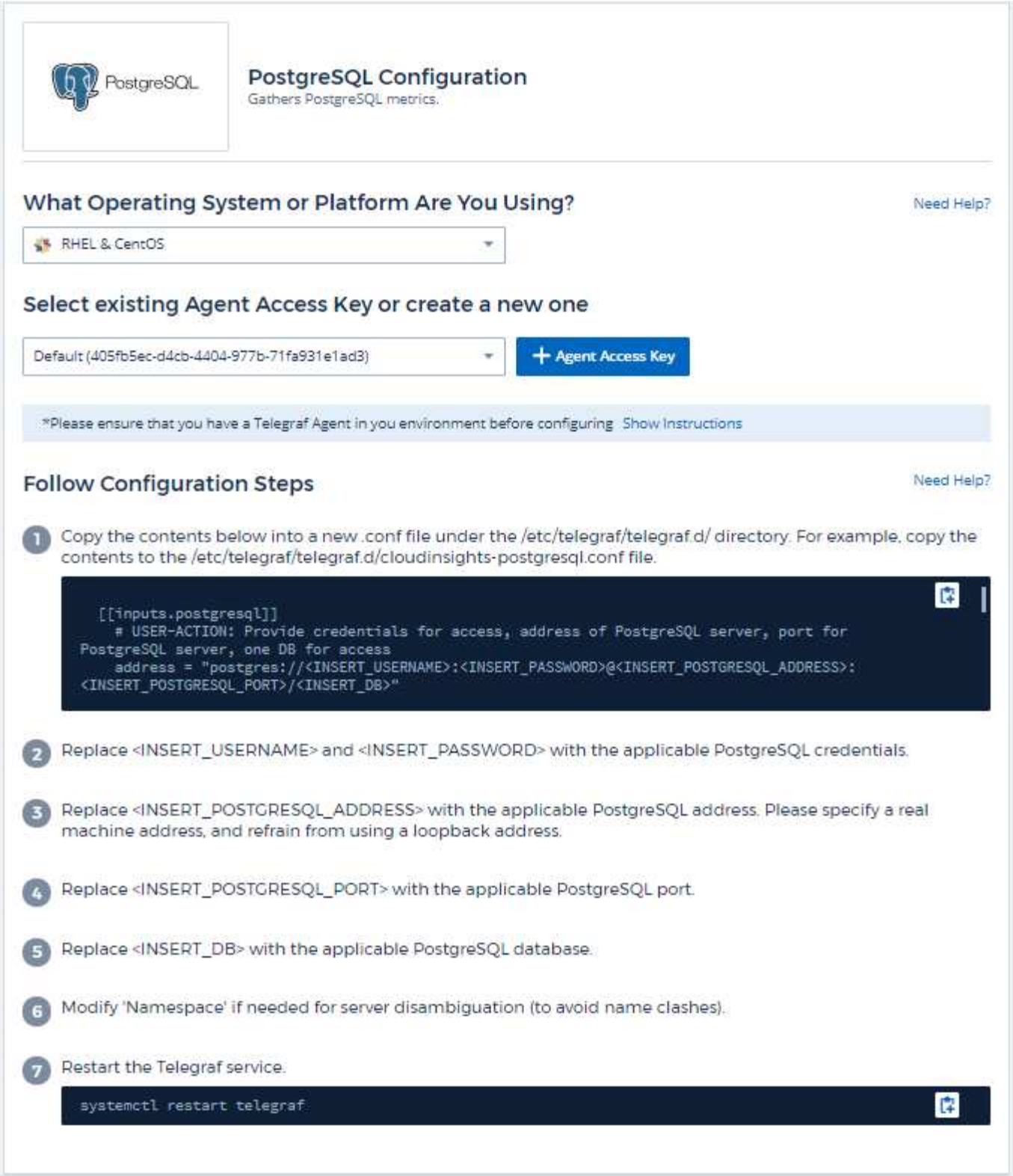
Data Infrastructure Insights uses this data collector to gather metrics from PostgreSQL.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose PostgreSQL.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the PostgreSQL documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
PostgreSQL Server	Namespace Database Server	Node Name Node IP	Buffers Allocated Buffers Backend Buffers Backend File Sync Buffers Checkpoint Buffers Clean Checkpoints Sync Time Checkpoints Write Time Checkpoints Requests Checkpoints Timed Max Written Clean
PostgreSQL Database	Namespace Database Server	Database OID Node Name Node IP	Blocks Read Time Blocks Write Time Blocks Hits Blocks Reads Conflicts Deadlocks Client Number Temp Files Bytes Temp Files Number Rows Deleted Rows Fetched Rows Inserted Rows Returned Rows Updated Transactions Committed Transactions Rollbacked

Troubleshooting

Additional information may be found from the Support page.

Puppet Agent Data Collector

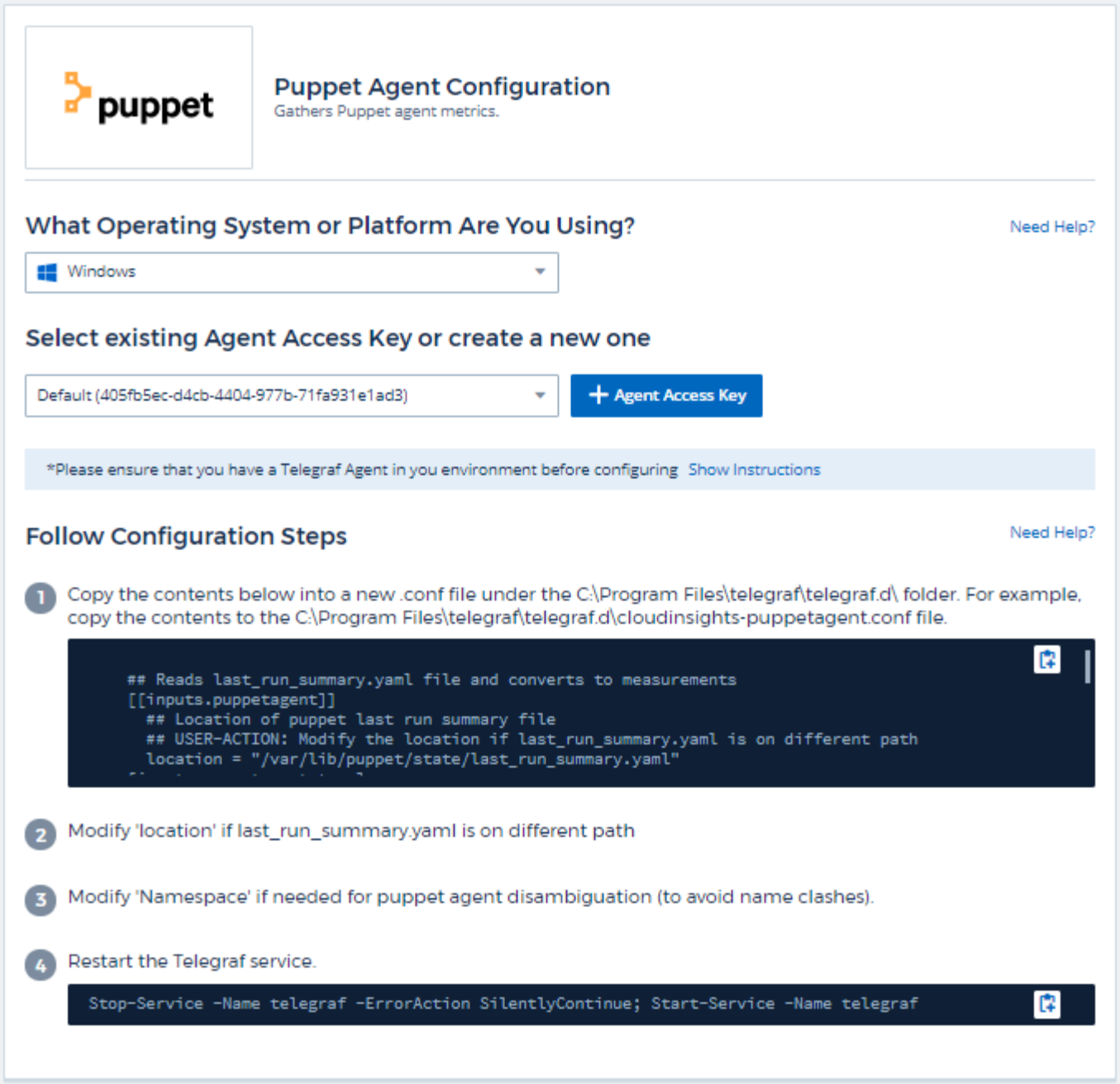
Data Infrastructure Insights uses this data collector to gather metrics from Puppet Agent.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Puppet.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Puppet documentation

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:

Puppet Agent	Namespace Node UUID	Node Name Location Node IP Version Configstring Version Puppet	Changes Total Events Failure Events Success Events Total Resources Changed Resources Failed Resources Failed To Restart Resources Outofsync Resources Restarted Resources Scheduled Resources Skipped Resources Total Time Anchor Time Configretrieval Time Cron Time Exec Time File Time Filebucket Time Lastrun Time Package Time Schedule Time Service Time Sshauthorizedkey Time Total Time User

Troubleshooting

Additional information may be found from the Support page.

Redis Data Collector

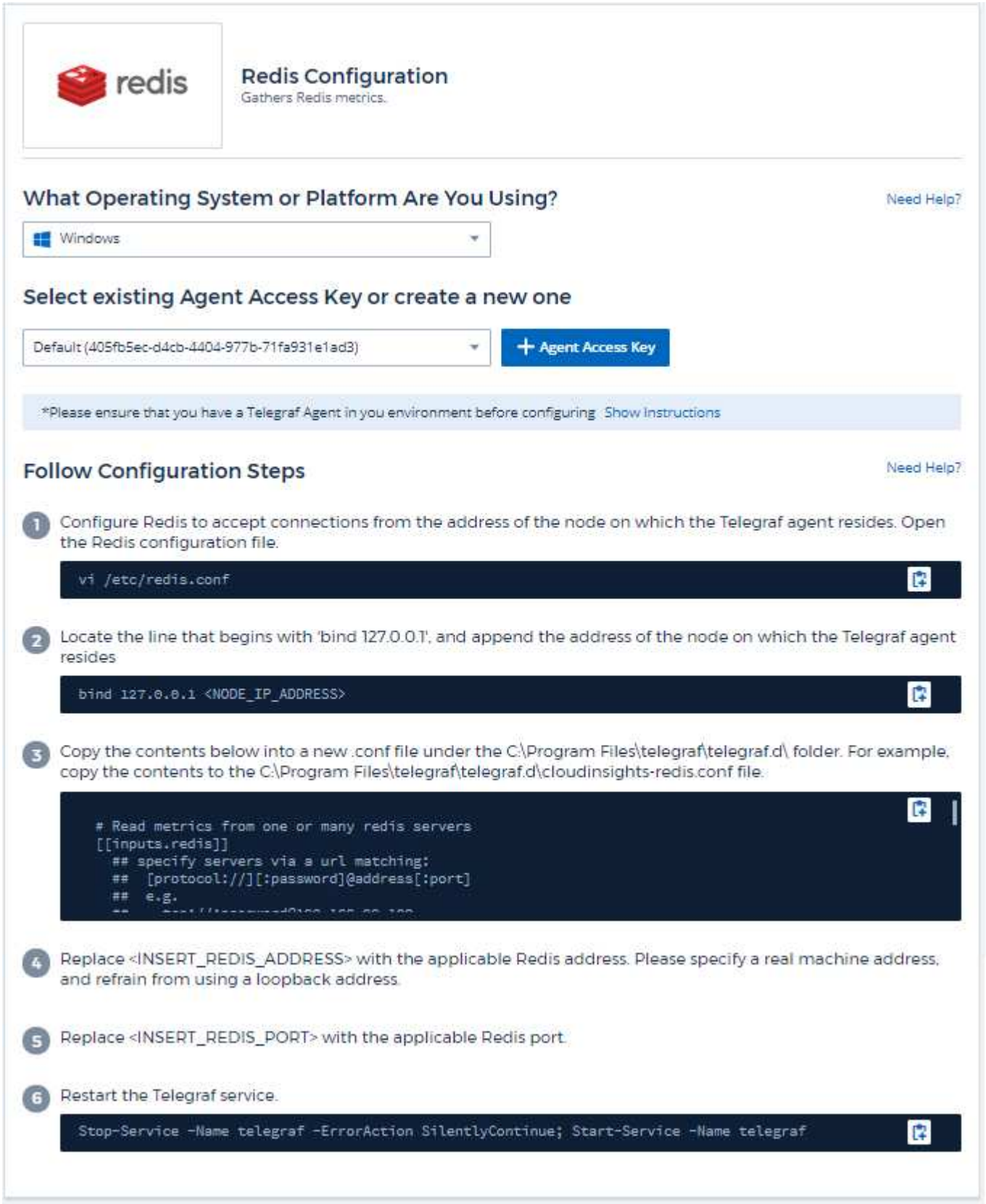
Data Infrastructure Insights uses this data collector to gather metrics from Redis. Redis is an open source, in-memory data structure store used as a database, cache, and message broker, supporting the following data structures: strings, hashes, lists, sets, and more.

Installation

1. From **Observability > Collectors**, click **+Data Collector**. Choose Redis.

   Select the Operating System or Platform on which the Telegraf agent is installed.

2. If you haven't already installed an Agent for collection, or you wish to install an Agent for a different Operating System or Platform, click *Show Instructions* to expand the Agent installation instructions.
3. Select the Agent Access Key for use with this data collector. You can add a new Agent Access Key by clicking the **+ Agent Access Key** button. Best practice: Use a different Agent Access Key only when you want to group data collectors, for example, by OS/Platform.
4. Follow the configuration steps to configure the data collector. The instructions vary depending on the type of Operating System or Platform you are using to collect data.



Setup

Information may be found in the Redis documentation.

Objects and Counters

The following objects and their counters are collected:

Object:	Identifiers:	Attributes:	Datapoints:
Redis	Namespace Server		

Troubleshooting

Additional information may be found from the Support page.

Copyright information

Copyright © 2025 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at http://www.netapp.com/TM are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.