

Prometheus and Grafana

- Feature Summary, on page 1
- Feature Description, on page 1
- Managing the PCF Statistics, on page 2
- Bulk Statistics, on page 6

Feature Summary

Summary Data

Table 1: Summary Data

Applicable Product(s) or Functional Area	5G-PCF
Applicable Platform(s)	SMI
Feature Default Setting	Enabled – Always-on
Related Changes in this Release	Not Applicable
Related Documentation	Not Applicable

Feature Description

You can a monitor wide range of application and system statistics, and key performance indicators (KPI) within the PCF infrastructure. KPIs are useful to gain insight into the overall health of the PCF environment. Statistics offer a simplified representation of the PCF configurations and utilization-specific data.

The PCF integrates with Prometheus, a third-party monitoring and alerting solution to capture and preserve the performance data. This data is reported as statistics and can be viewed in the web-based dashboard. Grafana provides a graphical or text-based representation of statistics and counters, which the Prometheus database collects. The Grafana dashboard projects a comprehensive set of quantitative and qualitative data that encourages you to analyze PCF metrics in the reporting tool of your choice and take informed decisions.

By default, the monitoring solution is enabled, which indicates that Prometheus continually monitors your PCF environment and the Prometheus data source is associated with Grafana. You must have the administrative privileges to access Grafana. However, to view a specific dashboard, run the Prometheus queries. The queries are available in the built-in and custom format.

How it Works

KPIs constitute of metrics such as statistics and counters. These metrics represent the performance improvement or degradation. By default, Prometheus is enabled on the system where PCF is deployed, and configured with Grafana. Prometheus dynamically starts monitoring the data sources that are available on the system. For new dashboard panels, execute queries in Prometheus.

For more information about Prometheus, consult the Prometheus documentation at https://prometheus.io/docs/introduction/overview/.

Managing the PCF Statistics

This section describes how to view statistics within PCF.

Managing the PCF statistics involves the following:

1. Viewing the Statistics, on page 2
2. Accessing the Grafana Dashboard, on page 2
3. Viewing the PCF Dashboard, on page 3
4. Running a Query in Grafana, on page 4
5. Configuring Autorefresh, on page 5
6. Exporting and Importing Dashboards, on page 5

Viewing the Statistics

This section describes how to view the statistics information.

1. On the system where PCF is deployed, navigate to the following URL:

   `https://docs.`*`namespace`*`-product-documentation.`*`IP_address`*`.nip.io/`

   All the PCF-specific statistics and other generic statistics such as system-statistics derived from the SMI deployer get displayed on the HTML page.

Accessing the Grafana Dashboard

This section describes how to access Grafana to view the visual representation of KPIs.

1. On the system where PCF is deployed, navigate to the following URL to view the dashboard:

   `https://grafana.smi-cnat-monitoring.`*`IP_address`*`.nip.io`

2. Enter the administrative user's username and password.

For more information on Grafana's capabilities, consult the Grafana documentation available at http://docs.grafana.org.

Viewing the PCF Dashboard

This section how to view the PCF dashboard.

1. On the PCF Application dashboard, in the left pane, click the dashboard icon to open the menu and select **Manage**.
2. In the **Manage** tab, click the *namespace* folder.

   The folder name resembles the namespace in which PCF is installed. The available dashboards are listed.
3. Click **PCF Application**.

   The PCF Application dashboard displays the graphs. You can shuffle the location of the graphs by dragging the panels.

Note Cisco recommends configuring the panel options in the Grafana dashboard. With this option, you can view only the required graphs when the dashboard is loaded.

Deploy Database Monitoring Dashboard

Starting with release 2026.01.0, you can optionally configure a new Database (DB) Monitoring dashboard that provides dedicated monitoring capabilities for key PCF data stores, such as MongoDB replica sets.

Table 2: Feature History

Feature Name	Release Information	Description
Deploy Database Monitoring Dashboard	2026.01.0	You can enable or disable the deployment of a specific DB Monitoring dashboard (Grafana JSON) by setting a configuration value. As a result, the dashboard is deployed only when needed, which optimizes resource usage and provides critical insights into database health and usage. The Database Monitoring Dashboard provides dedicated monitoring capabilities for key PCF data stores, such as MongoDB replica sets. Command Introduced: **testing enable-optional-dashboards [true | false]**.

Information About Deployment of DB Monitoring Dashboard

This feature introduces a new DB Monitoring dashboard, presented as a Grafana JSON, which includes panels to display MongoDB health and usage metrics. This dashboard is designed to be deployed conditionally. The deployment process leverages the `pcf-dashboard` Helm chart, which incorporates conditional logic. The decision to include or exclude the DB dashboard JSON during deployment is driven by a specific configuration value within the `ops-center`.



Note The deployment of existing standard dashboards, such as application, JVM, Diameter, and KPI metrics, is not affected. These dashboards are always deployed, regardless of this new feature's configuration.

Benefits of Deployment of DB Monitoring Dashboard

- **Dedicated DB Monitoring:**

  Provides a specialized dashboard for monitoring key PCF data stores like MongoDB replica sets, offering insights into their health and usage.

  .

- **Resource Optimization:**

  Allows the DB Monitoring dashboard to be deployed only when explicitly enabled, potentially reducing resource consumption when it's not required.

- **Configurability:**

  Enhances the configurability of the deployment process, giving users control over which dashboards are active.

- **Zero Impact:**

  Ensures that the introduction and conditional deployment of this new dashboard have no impact on existing metrics, dashboards, or operational workflows.

Configure Conditional Deployment of DB Monitoring Dashboard

- **Enable the DB Monitoring Dashboard:**

  To deploy the DB Monitoring dashboard alongside the standard dashboards, set the `enable-optional-dashboards` configuration to `true`:

  ```
  testing enable-optional-dashboards true
  ```

- **Disable the DB Monitoring Dashboard:**

  To deploy only the standard dashboards and exclude the DB Monitoring dashboard, set the `enable-optional-dashboards` configuration to `false`:

  ```
  testing enable-optional-dashboards false
  ```

Running a Query in Grafana

This section describes how to execute a query in Grafana.

The PCF Dashboard creates a panel containing the graph that is based on the query that it ingests. Grafana brings up a panel to visualize data that is retrieved for one or more queries. You can run canned and custom queries from the dashboard. The canned queries are preexisting in the dashboard with the define syntax. Custom queries permit you to formulate queries that return specific information.

1. On the PCF Application Dashboard, in the left pane, click the explore icon to open the menu. On hovering over the icon, the tooltip text appears as **Explore**.
2. In the **Explore** pane, click the drop-down to choose the data source as **Prometheus**.
3. Do one of the following:
   - To execute a built-in query, click the **Metrics** drop-down and choose the query that you want to run.
   - To execute a custom query, enter the query in the corresponding field next to **Metrics**.
4. Click **Run Query**.

   The query retrieves the information from Prometheus and displays it in a graphical representation.

Configuring Autorefresh

This section describes how to configure autorefresh to ensure that you view the recent information on Grafana.

1. On the PCF Application dashboard, click the gear on the top-right corner to open the **Settings**. On hovering over this icon, the tooltip text displays **Dashboard settings**.
2. In the **General** pane, navigate to the **Time Options** section and enter the time range in the **Autorefresh** field. You can specify the range in seconds, minutes, hours, and days format.



Note If you opt not to specify range, then the dashboard gets refreshed at the default interval.

Exporting and Importing Dashboards

This section describes how to export and import Grafana dashboards between environments and share them.

Exporting Dashboards

To export a dashboard configuration to a file:

1. Log in as an administrative user.
2. Open the dashboard that you want to export.
3. Click the gear icon at the top of the page, and then click **Export** to save the dashboard configuration on your local system.
4. If prompted, browse to the location on your local system to save the dashboard template, then click OK.

Importing Dashboards

To import a dashboard from a file:

1. Log in as an administrative user.
2. In the left pane, click the Dashboard icon to open the menu and click **Home**. The home pane opens.
3. Click the Home drop-down and click **Import dashboard**.
4. Specify the Grafana dashboard URL or ID that you want to import, provide the JSON details, or click **Upload.json File** and browse to the JSON file that you want to import.
5. Click **Load**.

   Make sure to save the dashboard to protect the changes that you made to the dashboard.

Exporting the Graph Data to CSV

This section describes how to export a dashboard in a CSV format.

1. On the Grafana dashboard, click the title of the graph to open the graph controls.
2. Click the rows button to open the menu.
3. To view the export option, click **More** and then click **Export CSV**.

   Your web browser downloads the *grafana_data_export.csv* file.

Filtering the Graphs

This section describes how to filter graphs on a dashboard.

You can narrow down the visualizations appearing on a dashboard by filtering them based on the specific time range.

1. On the PCF Application dashboard, in the top-right corner, click the clock icon.
2. Choose the range for which you want to view the graphs. Quick ranges provide the commonly used ranges that retrieve data in the shortest time. For specific range, provide the range under the Custom range heading.

Bulk Statistics

Bulk statistics are the statistics that are collected periodically and written to a set of CSV files. These statistics can be used by external analytic processes and/or network management systems. Bulk stats allows you to combine different KPIs into a unified query that fetches the custom statistical data. For the complete list of PCF KPIs, see Statistics and KPI Reference.

The SMI component handles the collection of the bulk stats from the nodes and PCF consumes these stats. The bulk stats are generated for the following components:

- Container: Includes the raw and rate of the change statistics.
- Pod: Includes the raw and rate of the change statistics.

- System: Consists of the system level KPIs.

Note The container and Pod statistics contain the predefined infrastructure outputs such as CPU and memory. You can also customize the query to fetch specific outputs as per your requirement.

There are two types of bulk statistics:

- Gauge - A snapshot value that shows the statistic at that reporting moment (for example, the number of current PDP contexts, simultaneous Active EPS Bearers). Gauge statistics can increment or decrement continuously.
- Counter - A historic value that shows the statistic that accumulated over time (for example, the total number of CSR requests received). Counter values can only increment except in two cases: rollover, where a counter exceeds its maximum value and rolls over to zero, and reset, where a counter is manually reset to zero.

Configuring the Bulk Statistics Collection

This section describes how to configure the bulk statistics collection feature.

You can optimize and control the bulk statistics collection by creating the Prometheus query that you configure on the SMI Ops Center. The cumulative result of the statistics query is available in a CSV file which is created on the node where you run the query.

1. Log in to the SMI Ops Center and run the following:

```
configure
  bulk-stats enable true
  bulk-stats query  kpi_name
  expression "sum(irate(kpi_name
{exported_application=~\".*\",command_code=~\".*\"}[1m])>0) by
(exported_application,command_code)"

  label  operation_name
  exit
  bulk-stats query   kpi_name
  expression "(sum(rate(kpi_name[duration])) by (operation_name))"
  label  operation_name
  exit
```

NOTES:

- **bulk-stats query** *kpi_name*: Specify the statistics name for which you want to generate stats in bulk. For example, inbound_request_total and diameter_request_total.

  See Statistics and KPI Reference for the list of KPIs.

- **expression** *"sum(irate(kpi_name {exported_application=~\".*\",command_code=~\".*\"}[1m])>0) by (exported_application,command_code)"*: Indicates the query format following which SMI collects the stats. For example: *"sum(irate(diameter_requests_total{exported_application=~\".*\",command_code=~\".*\"}[1m])>0) by (exported_application,command_code)"*

Note Based on the KPI tha you specify, manipulate the query. For instance, in case of the inbound_request_total KPI, add a parameter for specifying the duration as [5m]. This means that the SMI collects the stats for the total inbound requests that are processed in 5 minutes.

- **label** *operation_name*: Specify the operation that processes the KPI.

Sample Queries for Bulk Statistics

This section contains the sample Prometheus queries.

Table 3: PCF Sample Queries

Query	Description
incoming_request_total{interface_name="N7""N5", command="Create"}	Fetches the total incoming create request for the N7 and N5 interfaces.
incoming_request_total{interface_name="N7""N5", command="Update"}	Collects the total incoming update request for the N7 and N5 interfaces.
diameter_requests_total{command_code="AAR"}	Collects the total number of Diameter requests that PCF has processed.
message_total{type=~"ldap_change-res_success"}	Fetches the total number of messages that are triggered for the changes that are successfully completed in the LDAP.
outgoing_request_total{interface_name="NRF""N5", command="DeRegister"}	Collects the total count of the outgoing deregistration requests that NRF and N5 processed.

Sample Configuration

The following is a sample bulk statistic:

```
 cee(config)#
bulk-stats enable true
bulk-stats query diameter_request_total
expression "sum(irate(diameter_requests_total{exported_application=~\".*\",command_code=~\
".*\"}[1m])>0) by (exported_application,command_code)"
label command_code
exit
bulk-stats query inbound_request_total
expression "(sum(rate(inbound_request_total[5m])) by (operation_name))"
label operation_name
exit
bulk-stats query outgoing_request_total
expression "(sum(rate(outgoing_request_total[5m])) by (operation_name))"
label operation_name
exit
```

Sample Bulk Statistics Configuration

This section provides sample bulk statistics configurations that are defined in PCF.

LDAP_Max_Avail_Connections

```
bulk-stats query LDAP_Max_Avail_Connections
expression sum(ldap_connections_total{type=~\"MaximumAvailableConnections.*\",
server_set=~\".*\"} OR on () vector (0)) by (namespace)
label (server_set)
exit
```

LDAP_Num_Avail_Connections

```
bulk-stats query LDAP_Num_Avail_Connections
expression sum(ldap_connections_total{type=~\"MaximumAvailableConnections.*\",
server_set=~\".*\"} OR on () vector (0)) by (namespace)
label (server_set)
exit
```

N28_Notify_Failure

```
bulk-stats query N28_Notify_Failure
expression sum(incoming_request_total{interface_name=\"N28\", command=\"Notify\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Notify_Success

```
bulk-stats query  N28_Notify_Success
expression
sum(incoming_request_total{interface_name=\"N28\",command=\"Notify\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Notify_Total

```
bulk-stats query  N28_Notify_Total
expression sum(incoming_request_total{interface_name=\"N28\",command=\"Notify\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Sessions

```
bulk-stats query N28_Sessions
expression sum(db_records_total{session_type=\"N28_TGPP\"} OR on () vector (0)) by
(namespace)
label (session_type)
exit
```

N28_Subscribe_Failure

```
bulk-stats query N28_Subscribe_Failure
expression sum(outgoing_request_total{interface_name=\"N28\", command=\"Subscribe\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Subscribe_Failure

```
bulk-stats query N28_Subscribe_Failure
expression sum(outgoing_request_total{interface_name=\"N28\", command=\"Subscribe\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Subscribe_Success

```
bulk-stats query N28_Subscribe_Success
expression sum
(outgoing_request_total{interface_name=\"N28\",command=\"Subscribe\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Subscribe_Total

```
bulk-stats query N28_Subscribe_Total
expression sum(outgoing_request_total{interface_name=\"N28\",command=\"Subscribe\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Terminate_Failure

```
bulk-stats query  N28_Terminate_Failure
expression sum(incoming_request_total{interface_name=\"N28\", command=\"Terminate\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Terminate_Success

```
bulk-stats query N28_Terminate_Success
expression
sum(incoming_request_total{interface_name=\"N28\",command=\"Terminate\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Terminate_Total

```
bulk-stats query N28_Terminate_Total
expression sum(incoming_request_total{interface_name=\"N28\",command=\"Terminate\"}
OR on () vector (0)) by (namespace)
```

```
label (interface_name)
exit
```

N28_Unsubscribe_Failure

```
bulk-stats query N28_Unsubscribe_Failure
expression sum(outgoing_request_total{interface_name=\"N28\",
command=\"Unsubscribe\", result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Unsubscribe_Success

```
bulk-stats N28_Unsubscribe_Success
expression
sum(outgoing_request_total{interface_name=\"N28\",command=\"Unsubscribe\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N28_Unsubscribe_Total

```
bulk-stats query N28_Unsubscribe_Total
expression sum(outgoing_request_total{interface_name=\"N28\",command=\"Unsubscribe\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Create_Failure

```
bulk-stats query N7_Create_Failure
expression sum(incoming_request_total{interface_name=\"N7\", command=\"Create\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Create_Success

```
bulk-stats query N7_Create_Success
expression
sum(incoming_request_total{interface_name=\"N7\",command=\"Create\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Create_Total

```
bulk-statsquery N7_Create_Total
expression sum(incoming_request_total{interface_name=\"N7\",command=\"Create\"} OR
on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Delete_Failure

```
bulk-stats query N7_Delete_Failure
expression sum(incoming_request_total{interface_name=\"N7\", command=\"Delete\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Delete_Success

```
bulk-stats query N7_Delete_Success
expression
sum(incoming_request_total{interface_name=\"N7\",command=\"Delete\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Delete_Total

```
bulk-stats query  N7_Delete_Total
expression sum(incoming_request_total{interface_name=\"N7\",command=\"Delete\"} OR
on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Notify_Failure

```
bulk-stats query  N7_Notify_Failure
expression sum(outgoing_request_total{interface_name=\"N7\", command=\"Notify\",
result_code!~\"20.*\"} OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Notify_Success

```
bulk-stats query  N7_Notify_Success
expression
sum(outgoing_request_total{interface_name=\"N7\",command=\"Notify\",result_code=~\"20.*\"}
 OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

N7_Notify_Total

```
bulk-stats query  N7_Notify_Total
expression sum(outgoing_request_total{interface_name=\"N7\",command=\"Notify\"} OR
on () vector (0)) by (namespace)
label (interface_name)
exit
```

NAP_Total

```
bulk-stats query NAP_Total
expression sum(message_total{type=~\"ldap_change-res.+\"} OR on () vector (0)) by
(namespace)
```

```
label (ldap)
exit
```

NRF_Deregister_Failure

```
bulk-stats query NRF_Deregister_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"DeRegister\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Deregister_Success

```
bulk-stats query NRF_Deregister_Success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"DeRegister\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Deregister_Total

```
bulk-stats query NRF_Deregister_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"DeRegister\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Discovery_Failure

```
bulk-stats query NRF_Discovery_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Discovery\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label(interface_name)
exit
```

NRF_Discovery_Success

```
bulk-stats query NRF_Discovery_Success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Discovery\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Discovery_Total

```
bulk-stats query NRF_Discovery_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"Discovery\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Heartbeat_Failure

```
bulk-stats query NRF_Heartbeat_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Heartbeat\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Heartbeat_Success

```
bulk-stats query NRF_Heartbeat_Success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Heartbeat\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Heartbeat_Total

```
bulk-stats query NRF_Heartbeat_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"Heartbeat\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Notify_Failure

```
bulk-stats query NRF_Notify_Failure
expression
sum(incoming_request_total{interface_name=\"NRF\",command=\"Notify\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Notify_Success

```
bulk-stats query NRF_Notify_Success
expression
sum(incoming_request_total{interface_name=\"NRF\",command=\"Notify\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Notify_Total

```
bulk-stats query NRF_Notify_Total
expression sum(incoming_request_total{interface_name=\"NRF\",command=\"Notify\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Register_Failure

```
bulk-stats query NRF_Register_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Register\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Register_Total

```
bulk-stats query NRF_Register_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"Register\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Register_success

```
bulk-stats query NRF_Register_success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Register\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Subscribe_Failure

```
bulk-stats query NRF_Subscribe_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Subscribe\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Subscribe_Success

```
bulk-stats query NRF_Subscribe_Success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Subscribe\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Subscribe_Total

```
bulk-stats query NRF_Subscribe_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"Subscribe\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Unsubscribe_Failure

```
bulk-stats query NRF_Unsubscribe_Failure
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Unubscribe\",result_code!~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Unsubscribe_Success

```
bulk-stats query NRF_Unsubscribe_Success
expression
sum(outgoing_request_total{interface_name=\"NRF\",command=\"Unubscribe\",result_code=~\"20.*\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

NRF_Unsubscribe_Total

```
bulk-stats query NRF_Unsubscribe_Total
expression sum(outgoing_request_total{interface_name=\"NRF\",command=\"Unsubscribe\"}
OR on () vector (0)) by (namespace)
label (interface_name)
exit
```

PLF_Failure

```
bulk-stats query PLF_Failure
expression sum(message_total{type!~\"ldap_search-res_norecord|ldap_search-res_success\"}
 OR on ()
vector (0)) by (namespace)
label (ldap)
exit
```

PLF_Success

```
bulk-stats query PLF_Success
expression sum(message_total{type=~\"ldap_search-res_norecord|ldap_search-res_success\"}
 OR on ()
vector (0)) by (namespace)
label (ldap)
exit
```

PLF_Total

```
bulk-stats query PLF_Total
expression sum(message_total{type=~\"ldap_search.+\"} OR on () vector (0)) by
(namespace)
label (ldap)
exit
```

Rx_AAR_Failure

```
bulk-stats query Rx_AAR_Failure
expression sum(diameter_responses_total{command_code=\"AAA\",result_code!=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_AAR_Success

```
bulk-stats query Rx_AAR_Success
expression sum(diameter_responses_total{command_code=\"AAA\",result_code=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_AAR_Total

```
bulk-stats query Rx_AAR_Total
expression sum(diameter_requests_total{command_code=\"AAR\"} OR on () vector (0))
by (namespace)
label (command_code)
exit
```

Rx_ASR_Failure

```
bulk-stats query Rx_ASR_Failure
expression sum(diameter_responses_total{command_code=\"ASA\",result_code!=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_ASR_Success

```
bulk-stats query Rx_ASR_Success
expression sum(diameter_responses_total{command_code=\"ASA\",result_code=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code))
exit
```

Rx_ASR_Total

```
bulk-stats query Rx_ASR_Total
expression sum(diameter_requests_total{command_code=\"ASR\"} OR on () vector (0))
by (namespace)
label (command_code)
exit
```

Rx_RAR_Failure

```
bulk-stats queryRx_RAR_Failure
expression sum(diameter_responses_total{command_code=\"RAA\",result_code!=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_RAR_Success

```
bulk-stats query Rx_RAR_Success
expression sum(diameter_responses_total{command_code=\"RAA\",result_code=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_RAR_Total

```
bulk-stats query Rx_RAR_Total
expression sum(diameter_requests_total{command_code=\"RAR\"} OR on () vector (0))
by (namespace)
label (command_code)
exit
```

Rx_STR_Failure

```
bulk-stats query Rx_STR_Failure
expression sum(diameter_responses_total{command_code=\"STA\",result_code!=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_STR_Success

```
bulk-stats query Rx_STR_Success
expression sum(diameter_responses_total{command_code=\"STA\",result_code=\"2001\"}
OR on () vector (0)) by (namespace)
label (command_code)
exit
```

Rx_STR_Total

```
bulk-stats query Rx_STR_Total
expression sum(diameter_requests_total{command_code=\"STR\"} OR on () vector (0)) by
(namespace)
label (command_code)
exit
```

Rx_Sessions

```
bulk-stats query Rx_Sessions
expression sum(db_records_total{session_type=\"RX_5G_TGPP\"} OR on () vector (0)) by
(namespace)
label (session_type)
exit
```

N5_Sessions

```
bulk-stats query N5_Sessions
expression sum(db_records_total{session_type=\"N5_5G_TGPP\"} OR on () vector (0)) by
(namespace)
label (session_type)
exit
```

APN_Sessions

```
bulk-stats query APN_Sessions
expression sum(db_records_total{session_type=~"APN-. *"} OR on () vector (0)) by
(namespace, session_type)
label (session_type)
exit
```

DNN_Sessions

```
bulk-stats query DNN_Sessions
expression sum(db_records_total{session_type=~"DNN-. *"} OR on () vector (0)) by
(namespace, session_type)
label (session_type)
exit
```

Total_Diameter_Peer

```
bulk-stats query Total_Diameter_Peer
expression count(diameter_peer_status) OR on() vector(0)
exit
```

Total_Diameter_Peer_Connected

```
bulk-stats query Total_Diameter_Peer_Connected
expression count(diameter_peer_status==0) OR on() vector(0)
exit
```

USD_Modify_Failure

```
bulk-stats query USD_Modify_Failure
expression
sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_Modify\",status!=\"success\"} OR
on () vector (0)) by (namespace)
label (ldap)
exit
```

USD_Modify_Success

```
bulk-stats query USD_Modify_Success
expression
sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_Modify\",status=\"success\"} OR on
() vector (0)) by (namespace)
label (ldap)
exit
```

USD_Modify_Total

```
bulk-stats query USD_Modify_Total
expression sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_Modify\"} OR on
() vector (0)) by (namespace)
label (ldap)
exit
```

USD_Query_Failure

```
bulk-stats query USD_Query_Failure
expression
sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_query\",status!=\"success\"} OR on
() vector (0)) by (namespace)
label (ldap)
exit
```

USD_Query_Success

```
bulk-stats query USD_Query_Success
expression
sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_query\",status=\"success\"} OR on
() vector (0)) by (namespace)
label (ldap)
exit
```

USD_Query_Total

```
bulk-stats query USD_Query_Total
expression sum(message_total{component=\"ldap-ep\",type=~\".*_ldap_query\"} OR on
() vector (0)) by (namespace)
label (ldap)
exit
```

active-alerts

```
bulk-stats query active-alerts
expression sum(ALERTS{alertstate=\"firing\"})"
label (alertname)
exit
```

config-query-memory-used

```
bulk-stats query config-query-memory-used
expression sum(node_memory_MemTotal_bytes)-sum(node_memory_MemFree_bytes
label (hostname)
exit
```

query cpu-idle

```
bulk-stats query query cpu-idle
expression avg(rate(node_cpu_seconds_total{mode=\"idle\"}[1m]))
label (hostname)
exit
```

cpu-iowait

```
bulk-stats query cpu-iowait
expression avg(rate(node_cpu_seconds_total{mode=\"iowait\"}[1m]))*100.00
label (hostname)
exit
```

cpu-softirq

```
bulk-stats query cpu-softirq
expression avg(rate(node_cpu_seconds_total{mode=\"softirq\"}[1m]))*100.00
label (hostname)
exit
```

cpu-steal

```
bulk-stats query cpu-steal
expression avg(rate(node_cpu_seconds_total{mode=\"steal\"}[1m]))*100.00
label (hostname)
exit
```

cpu-system

```
bulk-stats query cpu-system
expression avg(rate(node_cpu_seconds_total{mode=\"system\"}[1m]))*100.00
label (hostname)
exit
```

cpu-user

```
bulk-stats query cpu-user
expression avg(rate(node_cpu_seconds_total{mode=\"user\"}[1m]))*100.00
label (hostname)
exit
```

daemonset-ready-percent

```
bulk-stats query daemonset-ready-percent
expression
kube_daemonset_status_number_ready/kube_daemonset_status_desired_number_scheduled*100
label (daemonset)
exit
```

datastore_failures

```
bulk-stats query datastore_failures
expression sum(datastore_request_total{error_code!~\"0|409\"})
label (error_code)
exit
```

deployment-ready-percent

```
bulk-stats query deployment-ready-percent
expression kube_deployment_status_replicas_available/kube_deployment_status_replicas*100
label (deployment)
exit
```

diameter_peer_status

```
bulk-stats query diameter_peer_status
expression avg(diameter_peer_status{host=~\".*\"} OR on () vector (0)) by (namespace)
```

```
label (host)
exit
```

entitlement-status

```
bulk-stats query entitlement-status
expression entitlement_status{enforce_mode!=\"InCompliance\"}
label (hostname)
exit
```

filesystem-data-avail-bytes

```
bulk-stats query filesystem-data-avail-bytes
expression avg(node_filesystem_avail_bytes{device=\"/dev/vda1\"})
label (hostname)
exit
```

filesystem-root-avail-bytes

```
bulk-stats query filesystem-root-avail-bytes
expression avg(node_filesystem_avail_bytes{device=\"/dev/sda1\"})
label (hostname)
exit
```

k8s-pods-status

```
bulk-stats query  k8s-pods-status
expression sum(kube_pod_status_phase)
label (phase)
exit
```

kubelet-node-status

```
bulk-stats query kubelet-node-status
expression sum(kube_node_status_condition{status=\"true\"})
label (condition)
exit
```

kublet-running-pod-count

```
bulk-stats query kublet-running-pod-count
expression kubelet_running_pod_count
label (hostname)
exit
```

memory-used

```
bulk-stats query query memory-used
expression sum(node_memory_MemTotal_bytes)
label (hostname)
exit
```

network-carrier-bond-changes-total

```
bulk-stats query network-carrier-bond-changes-total
expression sum(node_network_carrier_changes_total{device=~\"bond[0-9]\"} OR on ()
vector (0)) by (namespace)
label (hostname)
exit
```

network-carrier-ens-changes-total

```
bulk-stats query network-carrier-ens-changes-total
expression sum(node_network_carrier_changes_total{device=~\"ens.*\"} OR on () vector
(0)) by (namespace)
label (hostname)
exit
```

network-errors-total

```
bulk-stats query network-errors-total
expression sum(node_network_receive_errs_total)
label (hostname)
exit
```

network-receive-bond-bytes-total

```
bulk-stats query network-receive-bond-bytes-total
expression sum(node_network_receive_bytes_total{device=~\"bond[0-9]\"})
label (hostname)
exit
```

network-receive-ens-bytes-total

```
bulk-stats query network-receive-ens-bytes-total
expression sum(node_network_receive_bytes_total{device=~\"ens.*\"})
label (hostname)
exit
```

network-transmit-bond-bytes-total

```
bulk-stats query network-transmit-bond-bytes-total
expression sum(node_network_transmit_bytes_total{device=~\"bond[0-9]\"})
label (hostname)
exit
```

network-transmit-ens-bytes-total

```
bulk-stats query network-transmit-ens-bytes-total
expression sum(node_network_transmit_bytes_total{device=~\"ens.*\"})
label (hostname)
exit
```

node-disk-rate-read-bytes

```
bulk-stats query node-disk-rate-read-bytes
expression sum(rate(node_disk_read_bytes_total[5m]))
label (hostname)
exit
```

node-disk-write-read-bytes

```
bulk-stats query node-disk-write-read-bytes
expression sum(rate(node_disk_written_bytes_total[5m]))
label (hostname)
exit
```

node-load-15

```
bulk-stats query node-load-15
expression node_load15
label (hostname)
exit
```

node-memory-free-bytes

```
bulk-stats query node-memory-free-bytes
expression sum(node_memory_MemTotal_bytes)
label (hostname)
exit
```

record_conflict

```
bulk-stats query record_conflict
expression sum(datastore_notify_total{notification_type=\"RECORD_CONFLICT\"} OR on
() vector (0)) by (namespace)
label (notification_type)
exit
```

statefulset-ready-percent

```
bulk-stats query statefulset-ready-percent
expression kube_statefulset_status_replicas_ready/kube_statefulset_status_replicas*100
label (statefulset)
exit
```

timer_expiry

```
bulk-stats query timer_expiry
expression sum(datastore_notify_total{notification_type=\"TIMER_EPIRED\"} OR on ()
vector (0)) by (namespace)
label (notification_type)
exit
```

version_mismatch_retries

```
bulk-stats query version_mismatch_retries
expression sum(datastore_request_total{error_code=\"409\"} OR on () vector (0)) by
(namespace)
label (error_code)
exit
```