**Practical Exercise: Cloud-Native Observability on a VM**

Training Objective

Learn how to set up and view dashboards to analyze statistics of WSO2 Micro Integrator.

High-level Steps

* Install and set up monitoring tools.
* View the dashboards to see and analyze service mediation statistics.

Prerequisites

* Download and install the following tools:
* Prometheus
* Fluent Bit
* Loki
* Jaeger
* Grafana

Setting up Cloud Native Observability on a VM

Follow the instructions given below to set up a cloud native observability solution for your Micro Integrator (MI) deployment in a VM environment.

You need to start with the [minimum deployment](https://apim.docs.wso2.com/en/latest/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm/#step-1-set-up-the-minimum-deployment), which enables metric monitoring. Once you have set up the minimum deployment, you can add [log processing](https://apim.docs.wso2.com/en/latest/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm/#step-2-optionally-integrate-the-log-processing-add-on) and [message tracing](https://apim.docs.wso2.com/en/latest/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm/#step-3-optionally-integrate-the-message-tracing-add-on) capabilities to your solution.

Step 1 - Set up the minimum deployment

The minimum cloud native observability deployment requires **Prometheus** and **Grafana**. The Micro Integrator uses Prometheus to expose its statistics to Grafana. Grafana is used to visualize the statistics.

Step 1.1 - Set up Prometheus

Follow the instructions below to set up the Prometheus server:

1. Download Prometheus from the [Prometheus site](https://prometheus.io/download/).

**Tip**

Select the appropriate operating system and the architecture based on your operating system and requirements.

1. Extract the downloaded file and navigate to that directory.

**Info**

This directory is referred to as <PROMETHEUS\_HOME> from hereon.

1. Open the <PROMETHEUS\_HOME>/prometheus.yml file, and in the scrape\_configs section, add a configuration as follows:

global:

scrape\_interval: 15s

evaluation\_interval: 15s

scrape\_configs:

- job\_name: 'prometheus'

static\_configs:

- targets: ['localhost:9090']

- job\_name: esb\_stats

metrics\_path: /metric-service/metrics

static\_configs:

- targets: ['localhost:9201']

**Note**

* + Do not add or remove spaces when you copy the above configuration to the prometheus.ymal file.
  + In the targets section, you need to add your IP address and the port in which you are running the Micro Integrator server.

1. To start the Prometheus server, open a terminal, navigate to <PROMETHEUS\_HOME>, and execute the following command:

./prometheus

When the Prometheus server is successfully started, you will see the following log:

*Server is ready to receive web requests.*

Step 1.2 - Set up Grafana

Follow the instructions below to set up the Grafana server:

1. Download and install [Grafana](https://grafana.com/grafana/download/7.1.1" \t "_blank).

**Tip**

Follow the instructions (for your operating system) on the Grafana website.

1. Start you Grafana server.

**Tip**

The procedure to start Grafana depends on your operating system and the installation process. For example, If your operating system is Mac OS and you have installed Grafana via Homebrew, you start Grafana by issuing the brew services start grafana command.

1. Access the Grafana UI from the localhost:3000 URL.
2. Sign in using admin as both the username and the password.

Step 1.3 - Import dashboards to Grafana

The Micro Integrator provides pre-configured Grafana dashboards in which you can visualize MI statistics.

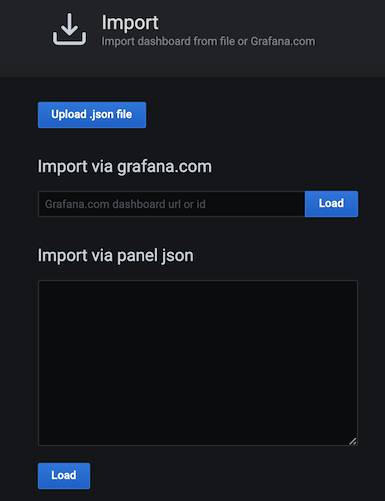
You can directly import the required dashboards to Grafana using the **dashboard ID**:

1. Go to [Grafana labs](https://grafana.com/orgs/wso2/dashboards" \t "_blank).
2. Select the required dashboard and copy the dashboard ID.
3. Provide this ID to Grafana and import the dashboard.
4. Repeat the above steps to import all other Micro Integrator dashboards.

These dashboards are provided as JSON files that can be manually imported to Grafana. To import the dashboards as JSON files:

1. Go to [Grafana labs](https://grafana.com/orgs/wso2/dashboards" \t "_blank), select the required dashboard and download the JSON file.
2. Sign in to Grafana, click the **+** icon in the left pane, and then click **Import**.

The **Import** dialog box opens as follows.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-import-dialog-box.png)

1. Click **Upload.json file**. Then browse for one of the dashboards that you downloaded as a JSON file.
2. Repeat the above two steps to import all the required dashboards that you downloaded and saved.

Step 1.4 - Set up the Micro Integrator

To enable observability for the Micro Integrator servers, add the following Synapse handler to the deployment.toml file (stored in the <MI\_HOME>/conf/ folder).

[[synapse\_handlers]]

name="CustomObservabilityHandler"

class="org.wso2.micro.integrator.observability.metric.handler.MetricHandler"

After applying the above change, you can start the Micro Integrator with the following JVM property:

-DenablePrometheusApi=true

Step 2 - Optionally, integrate the Log Processing add-on

Once you have successfully set up the [minimum deployment](https://apim.docs.wso2.com/en/latest/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm/#step-1-set-up-the-minimum-deployment), you need to set up the log processing add-on to process logs. To achieve this, you can use Grafana Loki-based logging stack.

A Loki-based logging stack consists of three components:

* **fluentBit** is the agent that gathers logs and sends them to Loki.
* **loki** is the main server that stores logs and processes queries.
* **Grafana** queries and displays the logs.

Follow the steps below to set up Fluent Bit and Grafana Loki:

Step 2.1 - Set up Fluent Bit

Follow the instructions below to set up Fluent Bit:

1. Download [Fluent Bit](https://fluentbit.io/download/).
2. Extract the downloaded file.

**Tip**

The directory is referred to as <FluentBit\_Home> from hereon.

1. Create the following files and save them with the given extension in a preferred location. You can use any text editor of your choice.

**Info**

In this example, the files are saved in the <HOME\_DIRECTORY>/conf directory.

**labelmap.json** file

{

"instance": "instance",

"log\_level": "log\_level",

"service": "service"

}

**parsers.conf** file

[PARSER]

Name observability

Format json

Time\_Key **time**

Time\_Format %Y-%m-%dT%H:%M:%S.%L

[PARSER]

Name wso2

Format regex

Regex \[(?<date>\d{2,4}\-\d{2,4}\-\d{2,4} \d{2,4}\:\d{2,4}\:\d{2,4}\,\d{1,6})\] (?<log\_level>[^\**s**]+) \{(?<class>[\**s**\S]\*)\} ([-]) (?<service>\{[\**s**\S]\*\})?(?<message>.\*)

Time\_Key date

Time\_Format %Y-%m-%d %H:%M:%S,%L

**fluentBit.conf** file

[SERVICE]

Flush 1

Daemon Off

Log\_Level info

Parsers\_File <Location/parsers.conf>

[INPUT]

Name tail

Path <MI\_HOME>/repository/logs/\*.log

Mem\_Buf\_Limit 500MB

Parser wso2

[OUTPUT]

Name loki

Match \*

Url http://localhost:3100/loki/api/v1/push

BatchWait 1

BatchSize 30720

Labels {job="fluent-bit"}

LineFormat json

LabelMapPath <Location/labelmap.json>

1. Follow the instructions below to build the Fluent Bit output plugin before starting Fluent Bit:
   * Clone the [grafana/loki git repository](https://github.com/grafana/loki" \t "_blank).
   * To build the Fluent Bit plugin, execute the following command.

make fluent-bit-plugin

For more details, see [Fluent Bit Output Plugin readme file](https://github.com/grafana/loki/blob/main/clients/cmd/fluent-bit/README.md#fluent-bit-output-plugin).

* + Copy and save the path of the out\_loki.so file.

1. Open a new terminal and navigate to the <FluentBit\_Home> directory.
2. Execute the following command:

**Tip**

Replace <location of out\_loki.so file> with the path that you copied and saved in the previous step.

fluent-bit -e <location of out\_loki.so file> -c <fluentbit.conf file path>

When Fluent Bit is successfully installed, you will see a log message.

Step 2.2 - Set up the Loki server

Grafana Loki aggregates and processes the logs from Fluent Bit.

Follow the instructions below to set up Grafana Loki:

1. Download Loki v1.6.1 from the [grafana/loki Git repository](https://github.com/grafana/loki/blob/v1.5.0/docs/installation/local.md" \t "_blank).

**Tip**

Be sure to select the appropriate OS version before downloading.

1. Create a configuration file named loki-local-config.yaml for Loki, similar to the sample given below, and save it in a preferred location.

**Tip**

* + You can use a text editor of your choice for this purpose.
  + You can change the given parameter values based on your requirement.

**auth\_enabled**: false

server:

http\_listen\_port: 3100

ingester:

lifecycler:

address: 127.0.0.1

ring:

kvstore:

store: inmemory

replication\_factor: 1

final\_sleep: 0s

chunk\_idle\_period: 5m

chunk\_retain\_period: 30s

max\_transfer\_retries: 0

schema\_config:

configs:

- from: 2018-04-15

store: boltdb

object\_store: filesystem

schema: v11

index:

prefix: index\_

period: 168h

storage\_config:

boltdb:

directory: /tmp/loki/index

filesystem:

directory: /tmp/loki/chunks

limits\_config:

enforce\_metric\_name: false

reject\_old\_samples: true

reject\_old\_samples\_max\_age: 168h

chunk\_store\_config:

max\_look\_back\_period: 0s

table\_manager:

retention\_deletes\_enabled: false

retention\_period: 0s

1. Unzip the file you downloaded in step 1.

The directory that is created as a result is referred to as <GrafanaLoki\_Home> from hereon.

1. Open a new terminal and navigate to <GrafanaLoki\_Home>.
2. Execute the following command:

./loki-darwin-amd64 -config.file=./loki-**local**-config.yaml

Step 2.3 - Configure Grafana to visualize logs

Follow the instructions below to add Loki as a datasource in Grafana:

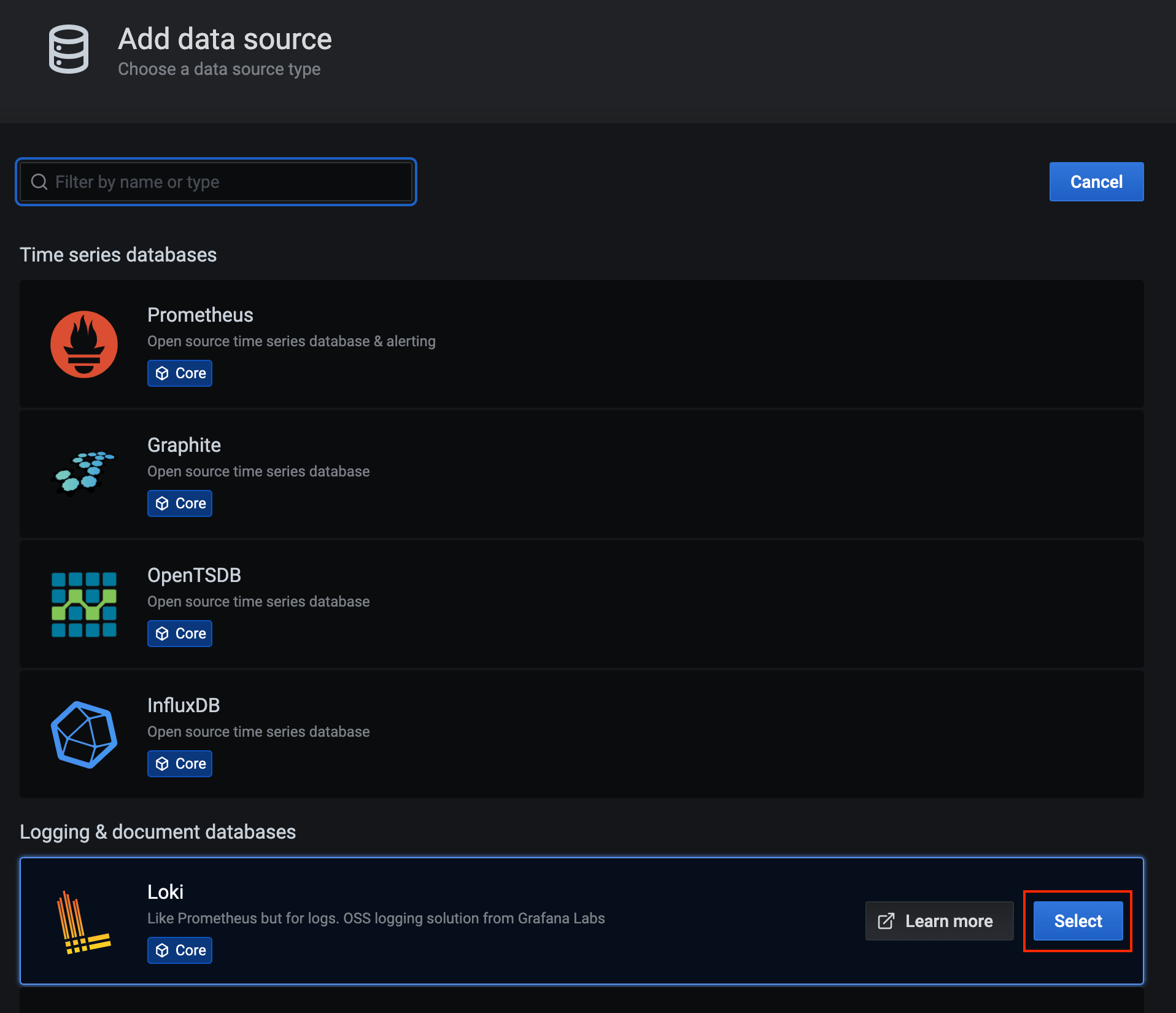
You need to do this in order to configure Grafana to display logs.

1. Start you Grafana server.

**Tip**

The procedure to start Grafana depends on your operating system and the installation process. For example, If your operating system is Mac OS and you have installed Grafana via Homebrew, you start Grafana by issuing the brew services start grafana command.

1. Access Grafana via http://localhost:3000/.
2. In the **Data Sources** section, click **Add your first data source**. In the **Add data source** page that appears, click **Select** for **Loki**.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-select-datasource.png)

1. In the **Add data source** page -> **Settings** tab, update the configurations for Loki.
2. Click **Save & Test**.

If the datasource is successfully configured, it is indicated via a message.

Step 3 - Optionally, integrate the Message Tracing add-on

Once you have successfully set up the [minimum deployment](https://apim.docs.wso2.com/en/latest/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm/#step-1-set-up-the-minimum-deployment), you need to set up the message tracing add-on using Jaeger.

**Note**

It is not essential to set up minimum deployment to observe the traces through Jaeger. You can do this without the minimum deployment as well.

Step 3.1 - Set up Jaeger

Download and install [Jaeger](https://www.jaegertracing.io/download/).

**Note**

* There are some limitations in the Jaeger client, which by default uses a UDP sender as mentioned in [the Jaeger documentation](https://www.jaegertracing.io/docs/1.22/client-libraries/). If the payload size exceeds 65 KB, spans might get lost in the Jaeger console.
* Jaeger [sampler types](https://www.jaegertracing.io/docs/1.22/sampling/) can also play a major role in tracing. Depending on the TPS, the sampler type should be carefully chosen.
* In general, before including tracing in production deployments, it is essential to look into performance tests and scaling requirements. For details on how to achieve better performance, see the [Jaeger performance tuning guide](https://www.jaegertracing.io/docs/1.22/performance-tuning/).

Step 3.2 - Set up the Micro Integrator

Follow the instructions below to configure the Micro Integrator to publish tracing information:

Add the following configurations to the deployment.toml file (stored in the <MI\_HOME>/conf/).

```toml

[mediation]

flow.statistics.capture\_all= true

stat.tracer.collect\_payloads= true

stat.tracer.collect\_mediation\_properties= true

[opentracing]

enable = true

logs = true

manager\_host = "localhost"

agent\_host = "localhost"

```

**Note**

The service name used to initialize the JaegerTracer can be configured using the environment variable SERVICE\_NAME as shown below.

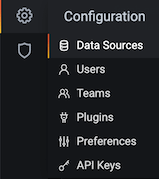
export SERVICE\_NAME=customServiceName

SERVICE\_NAME is set to WSO2-SYNAPSE by default.

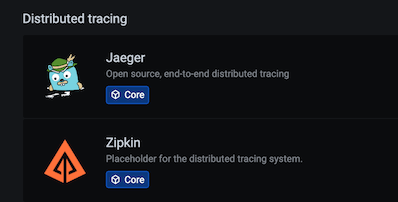
Step 3.3 - Configure Grafana to visualize tracing data

In order to configure Grafana to display tracing information, follow the steps given below.

1. Add Jaeger as a datasource:
   1. Access Grafana via localhost:3000 and sign in.
   2. Click the **Configuration** icon in the left menu and then click **Data Sources**.

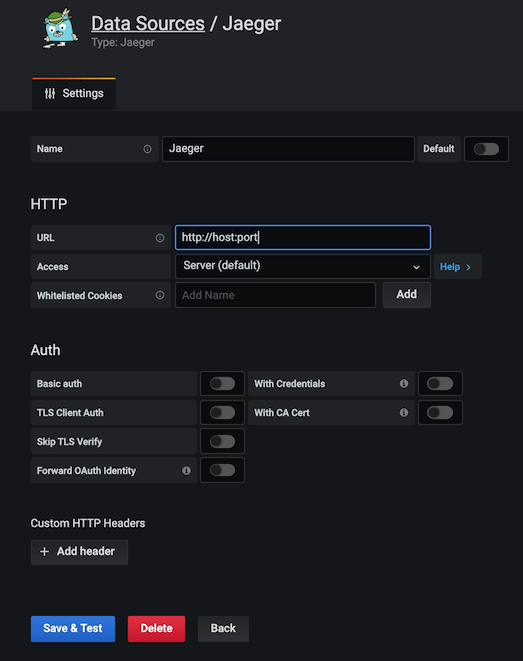
[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/open-datasources.png)

* 1. Click **Add data source** to open the **Add data source** page where all the available data source types are displayed. Here, click **Jaeger**.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/select-jaeger.png)

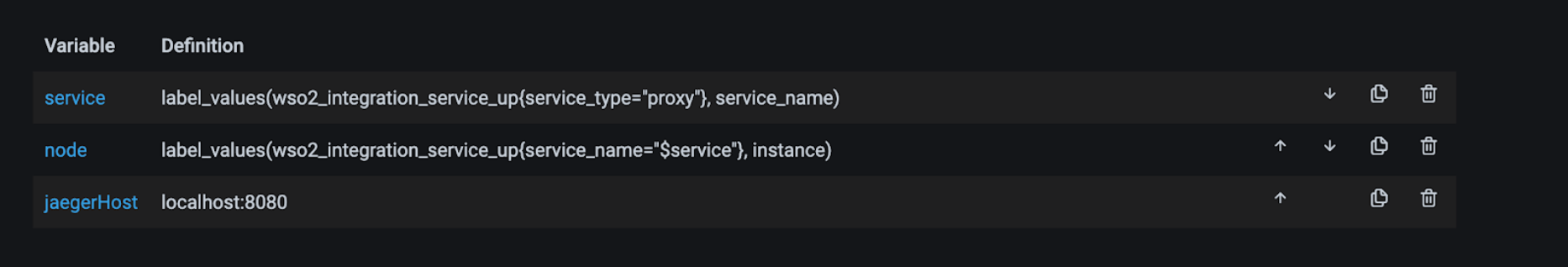
This opens the **Data Sources/Jaeger** dialog box.

* 1. In the **Data Sources/Jaeger** dialog box, enter the URL of the Jaeger query component in the **URL** field in the http://host:port format as shown below.

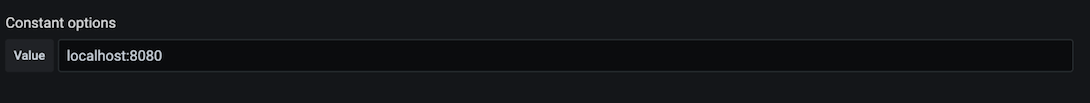
[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/enter-basic-jaeger-information.png)

* 1. Click **Save and Test**. If the data source is successfully configured, it is indicated via a message.

1. Set up drill-down links to the Jaeger UI in service-level dashboards.
   1. Navigate to the settings section of the service-level dashboard by clicking the cogwheel icon in the upper-right corner.
   2. Click **Variable**. This opens the following view.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/variables.png)

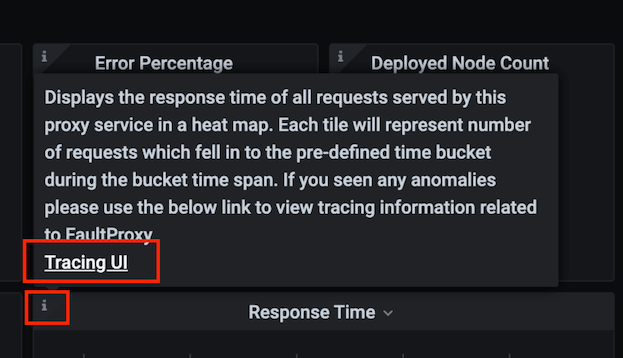
* 1. Edit the JaegerHost variable and provide your Jaeger query component hostname and port in the host:port syntax as shown below.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/constant-options.png)

* 1. Click **Save**

You need to perform the above steps for all the service-level dashboards (i.e., Proxy Service dashboard, API Service Dashboard, and Inbound Endpoint dashboard).

Once Grafana is successfully configured to visualize statistics, you should be correctly redirected to the Jaeger UI from the Response Time widget of each service-level dashboard as shown below.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/jaeger-ui.png)

Detailed Instructions

View and analyze dashboards:

Viewing Cloud Native Observability Statistics

Let's use the **dashboards** from the cloud-native observability deployment to monitor **statistics** from your integration artifacts.

Before you begin

Set up the suitable cloud-native observability deployment. The dashboards described in this section apply to all the cloud-native deployment strategies.

See the following topics for information and instructions:

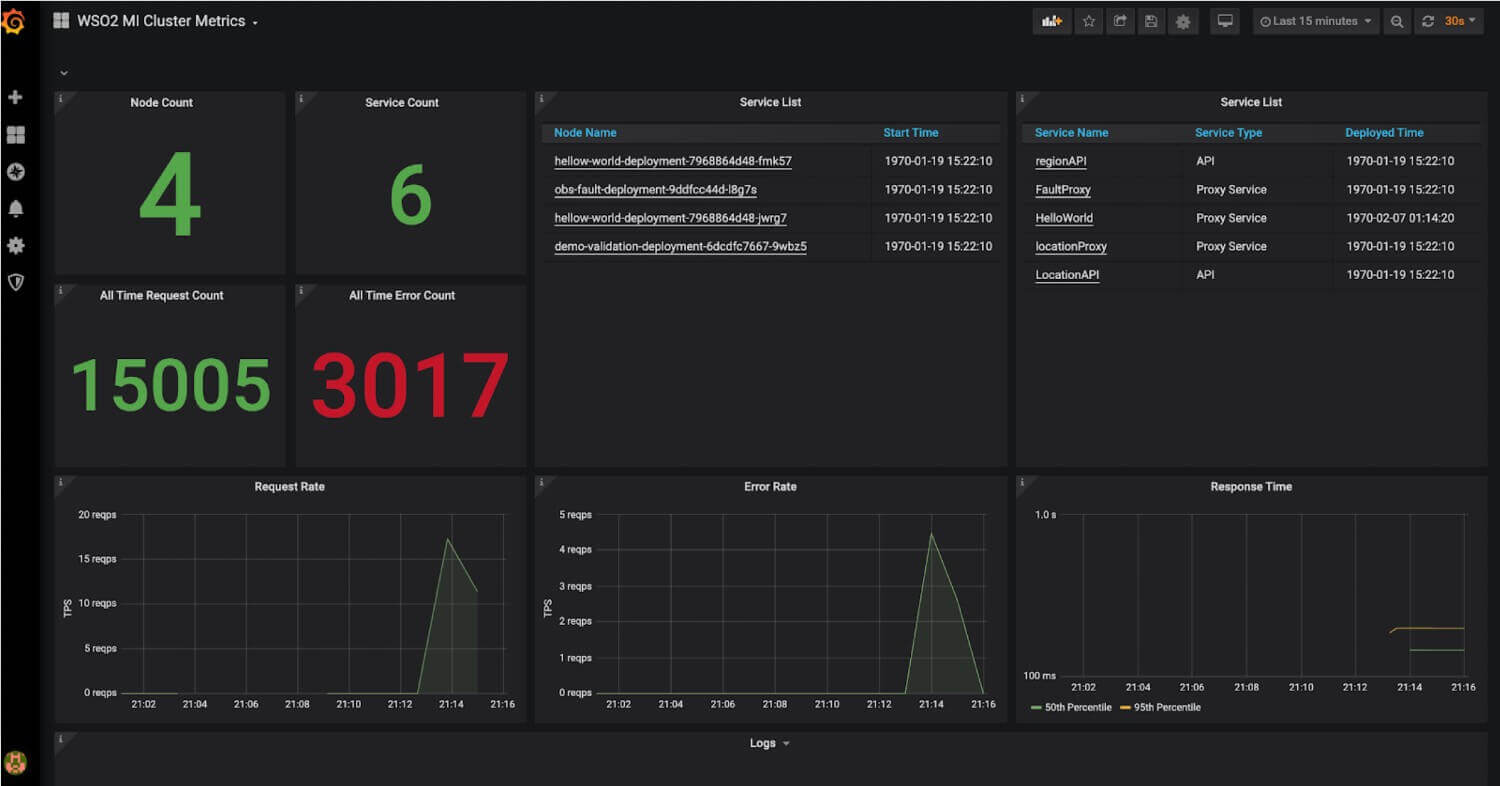
**Tip**

If you do not know which dashboard to download when setting up cloud-native observability, check the "Downloading the dashboard" section in the respective sub-sections below for details on the dashboard.

* Setting up [cloud-native observability for a VM environment](https://apim.docs.wso2.com/en/4.1.0/observe/micro-integrator/setting-up-cloud-native-observability-on-a-vm).
* Setting up [cloud-native observability for a Kubernetes environment](https://apim.docs.wso2.com/en/4.1.0/observe/micro-integrator/setting-up-cloud-native-observability-in-kubernetes).

Cluster dashboard

Cluster dashboards visualize the overall statistics relating to your Micro Integrator cluster. You can view information related to your MI cluster.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-cluster-dashboard.jpg)

Downloading the dashboard

You can download the dashboard from the [Grafana Labs - WSO2 Integration Cluster Metrics](https://grafana.com/grafana/dashboards/12783" \t "_blank).

Statistic types

The following is the list of widgets displayed in this dashboard.

| **Widget** | **Description** |
| --- | --- |
| **Node Count** | The total number of nodes in the cluster. |
| **Service Count** | The total number of services deployed in the cluster. |
| **Node List** | The list of nodes in the cluster. The time at which the node started is displayed together with the node name. You can click on a node to open the **MI Node Metrics** dashboard, which displays statistics specific to the selected node. |
| **Service List** | The list of services deployed in the cluster. The service type and the deployment time are displayed for each service. The service can be a proxy service or a REST API.  You can click on a proxy service to view statistics specific to it in the **WSO2 Proxy Service Metrics** dashboard.  You can click on a REST API service to view statistics specific to it in the **WSO2 API Metrics** dashboard. |
| **All Time Request Count** | The total number of requests handled by the cluster. |
| **All Time Error Count** | The total number of errors that have occurred for requests handled by the cluster. |
| **Request Rate** | This is a graphical representation of the number of requests handled by the cluster against time. |
| **Error Rate** | This is a graphical representation of the number of errors that have occurred for the cluster against time. |
| **Response Time** | The amount of time taken by the cluster to respond to a request against time. |

Purpose

This dashboard serves the following purposes:

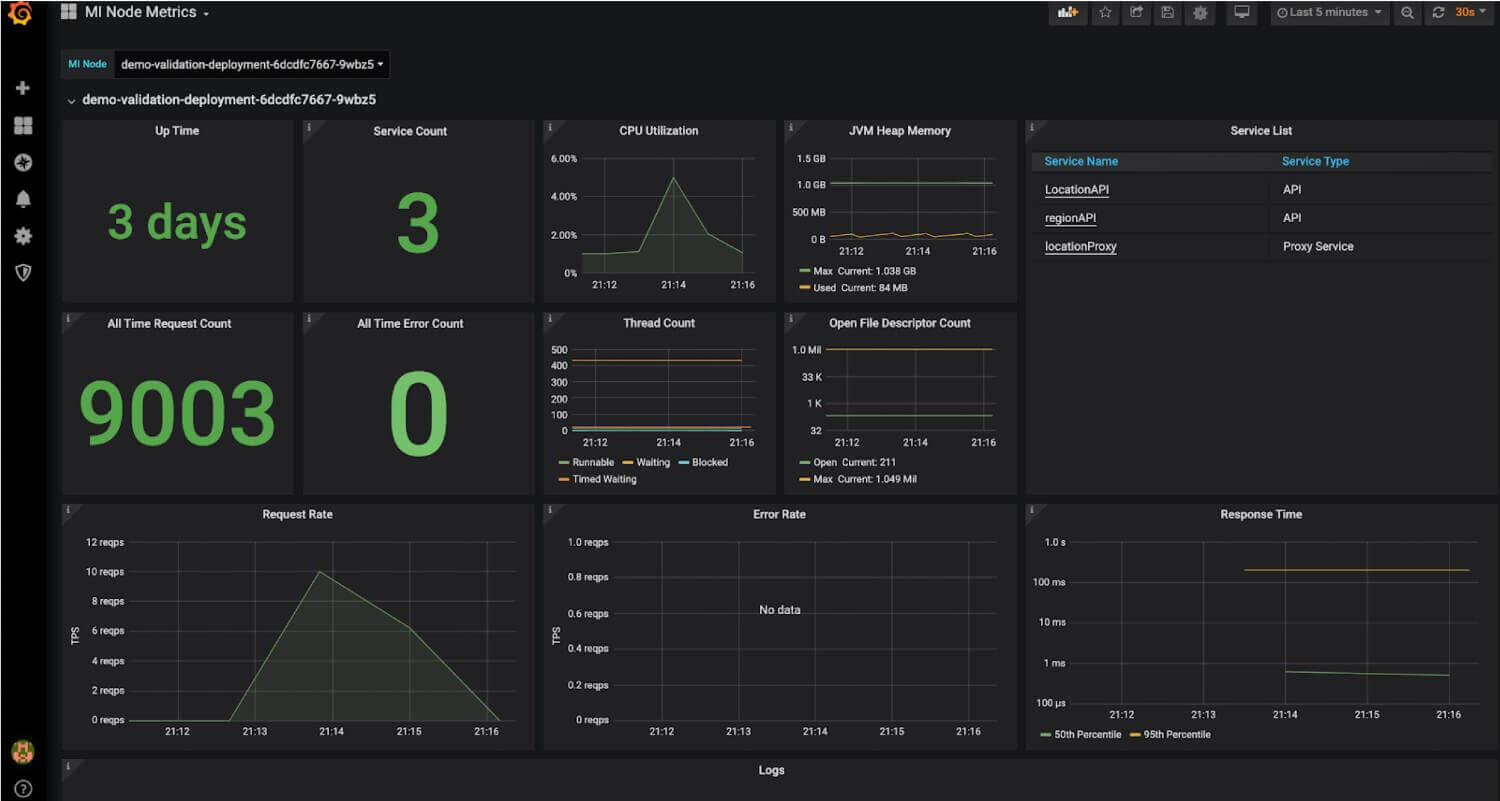
* It provides an overview of how the cluster as a whole performs in terms of the successful execution of requests and the response time.
* It also provides the basic details of the nodes and services deployed in the cluster. This can indicate how each node/service affects the overall cluster performance. e.g., If the **Error Rate** widget indicates a surge in the error rate at a particular time, you can identify a node/service that started at around the same time (as shown by the **Node List** and **Service List** widgets) as a possible cause of it.
* It provides access to other dashboards that display statistics related to specific nodes and services so that you can carry out further analysis relating to the performance of your Micro Integrator setup.

Recommended action

* Identify the times at which the error rate and/or the response time has been rising. Depending on the time, you can investigate the cause of if (e.g., a node/service that started around the same time).
* Click on the nodes/services that you have identified as nodes/services to be further analyzed to improve the performance of your Micro Integrator setup, and view the visualizations specific to them.
* Based on the request count, make the appropriate decisions with regard to the resource allocation (i.e., whether to add or reduce the number of nodes to leave the present number unchanged).
* Identify the popular services and make business decisions accordingly. For example, if there is a surge in the request rate, you can identify the services that were active during that time. You can analyze such services in more detail by viewing information specific to them and decide whether to invest more in them.

Node dashboard

This displays statistics specific to a selected node.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-node-metrics.jpg)

Downloading the dashboard

You can download the dashboard from the [Grafana Labs - WSO2 Integration Node Metrics](https://grafana.com/grafana/dashboards/12887" \t "_blank).

Statistic types

The following is the list of widgets displayed in this dashboard.

| **Widget** | **Description** |
| --- | --- |
| **Up Time** | The time duration that has elapsed since the node became active for the current session. |
| **Service Count** | The number of services (i.e., proxy services and REST API services) that are currently deployed in the node. |
| **All Time Request Count** | The total number of requests received by the node after it became a member of the current Micro Integrator setup. |
| **All Time Error Count** | The total number of requests handled by the node that have resulted in errors. |
| **CPU Utilization** | A visualization of the node's CPU consumption over time. |
| **JVM Heap Memory** | A visualization of the amount of JVM heap memory consumed by the node over time. |
| **Thread Count** | A visualization of the number of threads allocated to the node over time. |
| **Open File Descriptor Count** |  |
| **Services List** | The complete list of services (i.e., proxy services and REST API services) that are currently deployed in the node. |
| **Request Rate** | A visualization of the total number of requests received by the node over time. |
| **Error Rate** | A visualization of the total number of requests handled by the node that has resulted in errors over time. |
| **Response Time** | A visualization of the amount of time taken by the node to respond to requests over time. |

Purpose

The purposes of this dashboard are as follows:

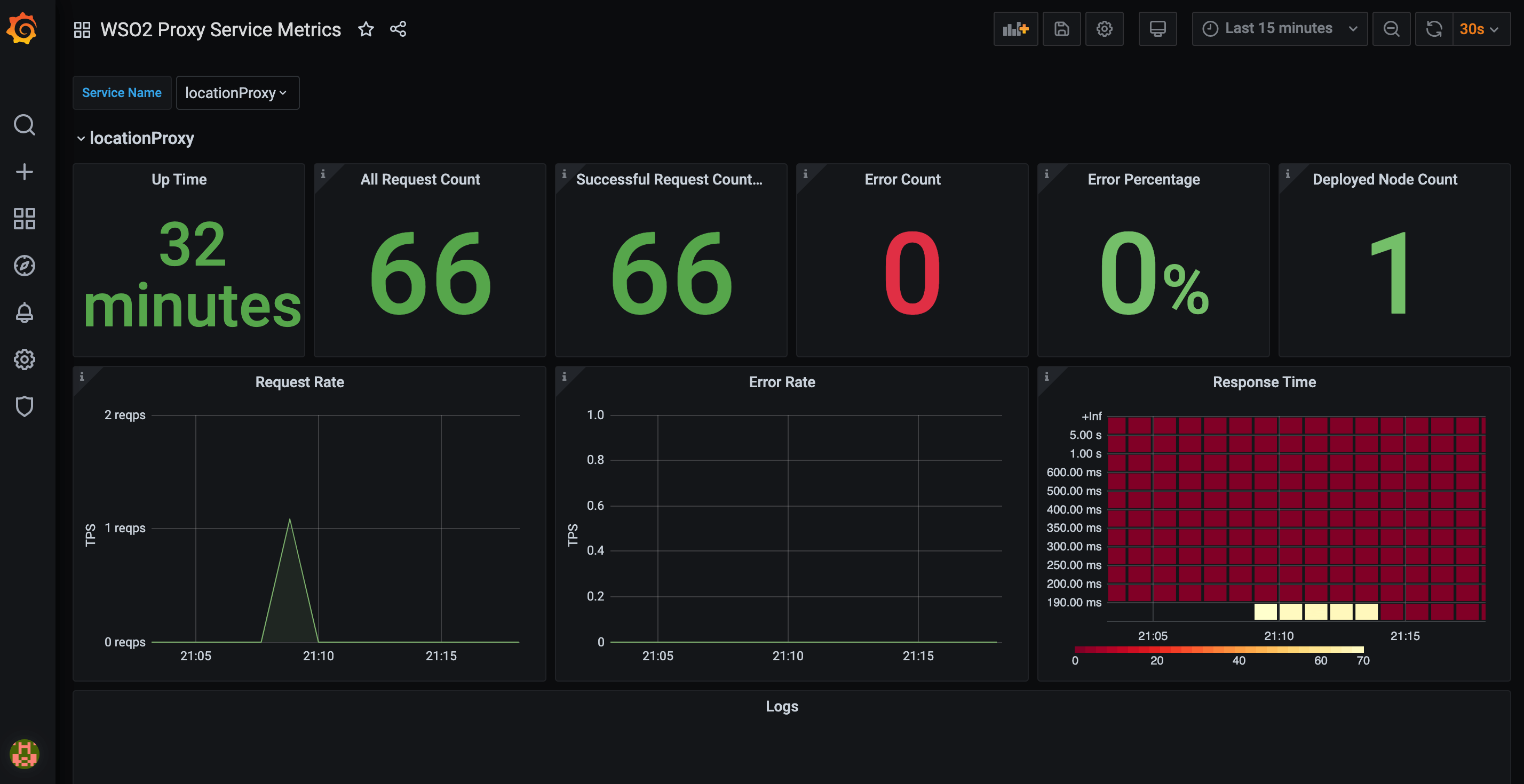
* It shows the performance of individual nodes in terms of the error count and the response time.
* It allows you to track the resource consumption of individual nodes and make decisions accordingly (e.g., to allocate more CPU cores, undeploy services with a high throughput if the node does not have sufficient system resources to run them etc.).
* By clicking on the name of a service deployed in the selected node, you can open the **Proxy Service Dashboard** and the **API Dashboard** dashboard (depending on the type of the service) to view statistics specific to the selected service.

Recommended action

* Evaluate whether the resources allocated to the node (i.e., system memory, CPU cores, etc.) are sufficient/excessive in proportion to the throughput it handles (i.e., the number of requests within a specific duration of time), and make changes accordingly. For example, suppose the number of requests that are handled is less in proportion to the node's capacity in terms of system resources. In that case, you can either reduce the number of resources to reduce your cost or deploy more services in the node to utilize the existing resources in a more optimum manner.
* Click on the services deployed in the node to view statistics specific to those services. This allows you to evaluate the throughput of each service to analyze further and make decisions on how to deploy the available services in the available nodes in a manner that optimizes the use of resources. It also allows you to identify the services that contribute to the total error count of the node and take appropriate action.

WSO2 Proxy Service Metrics dashboard

In the Proxy service dashboard, you can view information related to a specific Proxy service.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-proxy-services-dashboard.png)

Downloading the dashboard

You can download the dashboard from the [Grafana Labs - WSO2 Proxy Service Metrics](https://grafana.com/grafana/dashboards/12889" \t "_blank).

Statistic types

The following is the list of widgets displayed in this dashboard.

| **Widget** | **Description** |
| --- | --- |
| **Up Time** | The time duration that has elapsed since the proxy service started running during the current session. |
| **All Request Count** | The total number of requests received and handled by the proxy service during the selected time interval. |
| **Successful Request Count** | The total number of requests that were successfully executed by the proxy service during the selected time interval. |
| **Error Count** | The total number of requests handled by the proxy service during the selected time interval that have resulted in errors. |
| **Error Percentage** | The requests handled by the proxy service during the selected time interval that have resulted in errors, as a percentage of the total number of requests received by the proxy service during that same time interval. |
| **Deployed Node Count** | The number of nodes in which the proxy service is deployed. |
| **Request Rate** | A visualization of the total number of requests handled by the proxy service during the selected time interval. |
| **Error Rate** | A visualization of the total number of errors that have occurred for the proxy service during the selected time interval. |
| **Response Time** | A visualization of the time taken by the proxy service to respond to requests during the selected time interval. |

Purpose

The purposes of this dashboard are as follows:

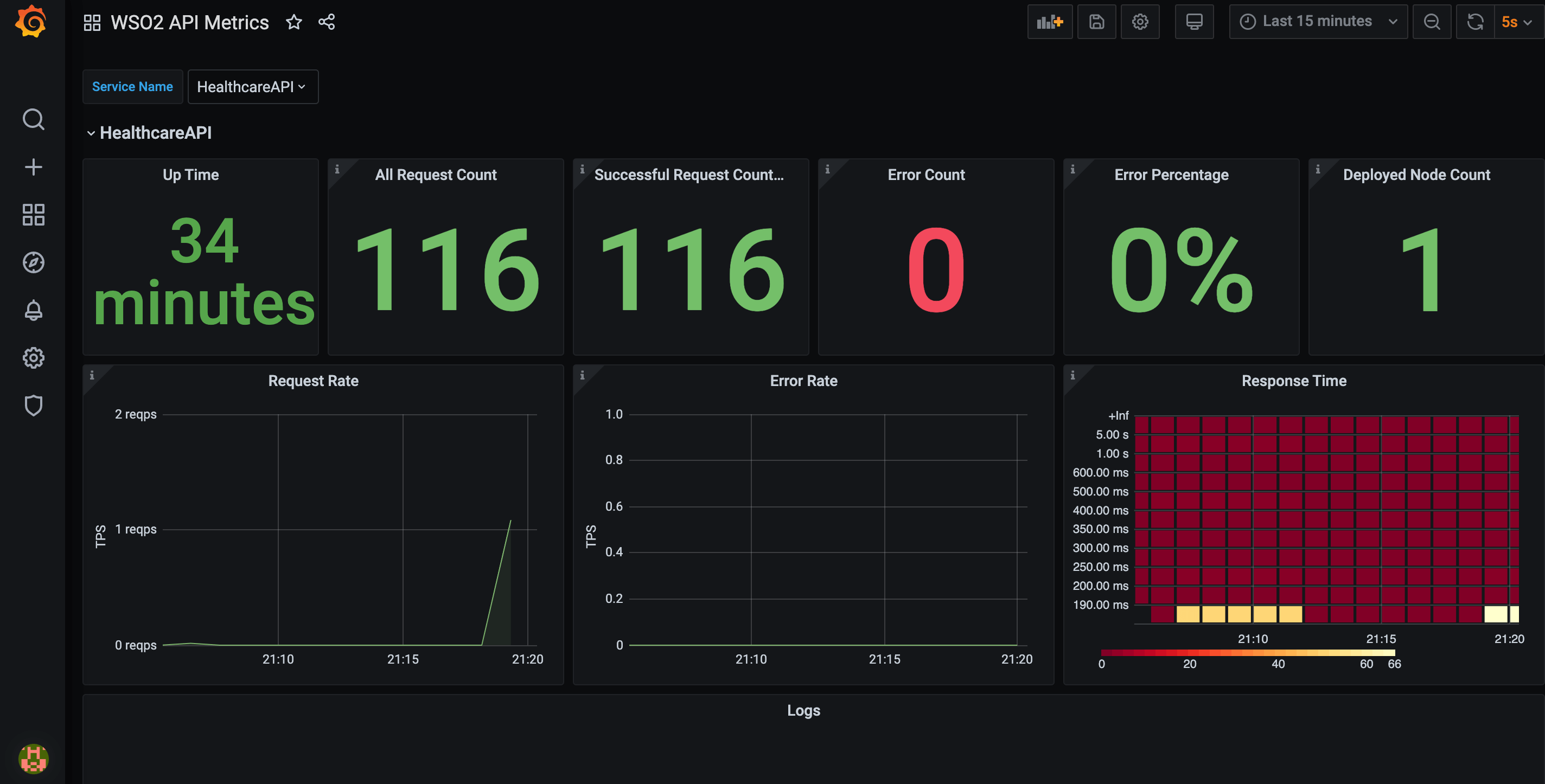
* To understand the performance of a selected proxy service in terms of the number of requests it processes within a given time duration, the number/percentage of errors that have resulted, and the time taken by the proxy service to respond to requests.
* To understand the client demand for the related business based on the number of requests received by the proxy service.

Recommended action

* If the number of requests/response time is too high, deploy the proxy service in more nodes in the cluster so that the throughput is divided.
* If there are errors, check the mediation flow of the proxy service and make changes to prevent the errors.

WSO2 API Metrics dashboard

This dashboard displays overall statistics related to a specific API.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-api-services-dashboard.png)

Downloading the dashboard

You can download the dashboard from the [Grafana Labs - WSO2 API Metrics](https://grafana.com/grafana/dashboards/12888" \t "_blank).

Statistic types

The following is the list of widgets displayed in this dashboard.

| **Widget** | **Description** |
| --- | --- |
| **Up Time** | The time duration that has elapsed since the API service started running during the current session. |
| **All Request Count** | The total number of requests received and handled by the API during the selected time interval. |
| **Successful Request Count** | The total number of requests that were successfully executed by the API during the selected time interval. |
| **Error Count** | The total number of requests handled by the API during the selected time interval that have resulted in errors. |
| **Error Percentage** | The requests handled by the API during the selected time interval that have resulted in errors, as a percentage of the total number of requests received by the API during that same time interval. |
| **Deployed Node Count** | The number of nodes in which the API service is deployed. |
| **Request Rate** | A visualization of the total number of requests handled by the API service during the selected time interval. |
| **Error Rate** | A visualization of the total number of errors that have occurred for the API service during the selected time interval. |
| **Response Time** | A visualization of the time taken by the API service to respond to requests during the selected time interval. |

Purpose

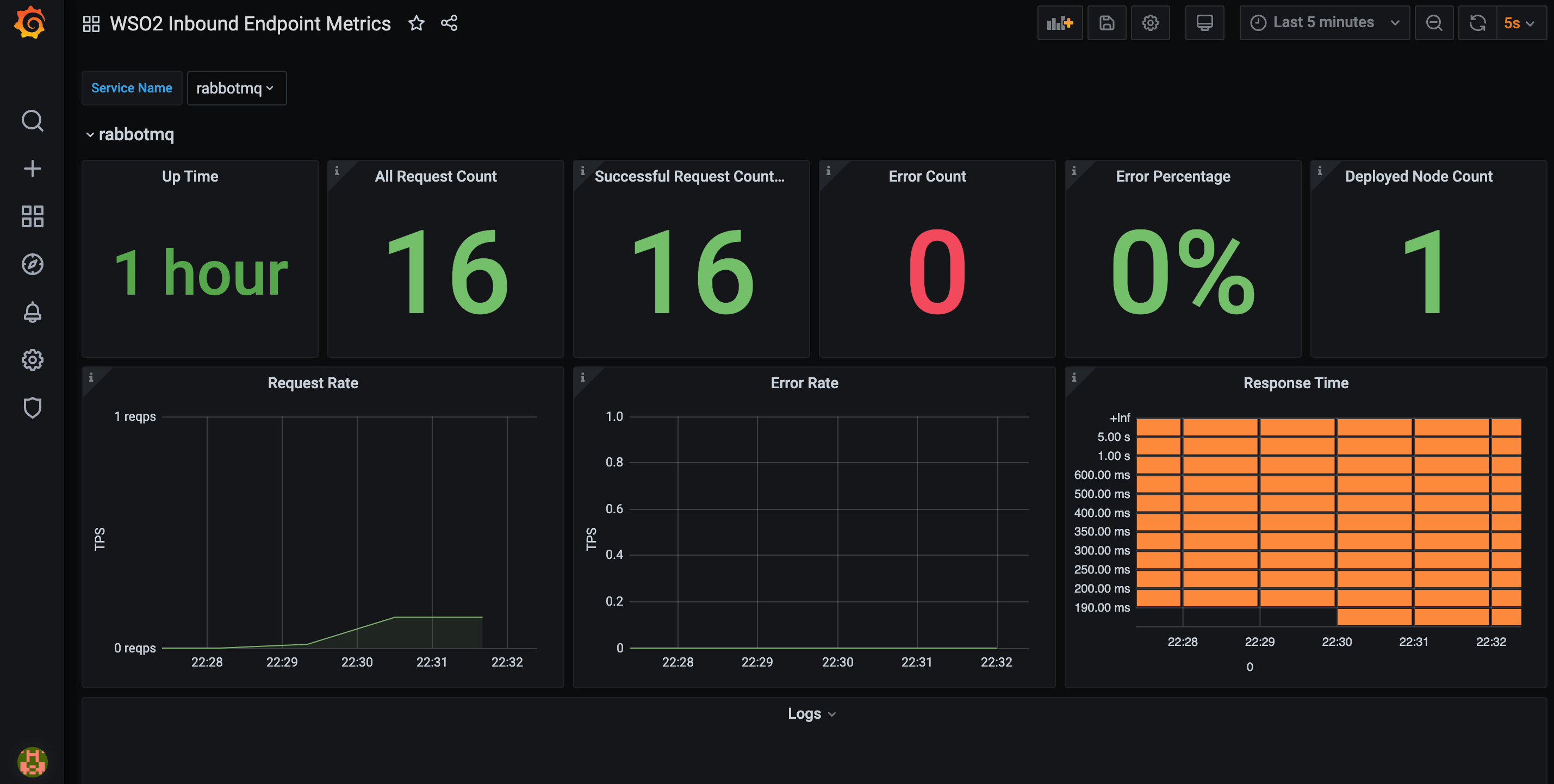
* To understand the performance of a selected API service in terms of the number of requests it processes within a given time duration, the number/percentage of errors that have resulted, and the time taken by the API service to respond to requests.
* To understand the client demand for the related business based on the number of requests received by the API service.

Recommended action

* If the number of requests/response time is too high, deploy the API service in more nodes in the cluster so that the throughput is divided.
* If there are errors, check the mediation flow of the API service and make changes to prevent the errors.

WSO2 Inbound Endpoint Metrics dashboard

At a given time, the Inbound endpoint dashboard displays the overall statistics related to a selected endpoint. You can view information related to a specific Inbound endpoint. We can download this dashboard from here. In this dashboard, it will show us Up Time, All Request Count, Successful Request Count, Error Count, Error Percentage, Deployed Node Count, Request Rate, Error Rate and Response Time.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/grafana-inbound-endpoint-metrics-dashboard.png)

Downloading the dashboard

You can download the dashboard from the [Grafana Labs - WSO2 Inbound Endpoint Metrics](https://grafana.com/grafana/dashboards/12890" \t "_blank).

Statistic types

The following is the list of widgets displayed in this dashboard.

| **Widget** | **Description** |
| --- | --- |
| **Up Time** | The time duration that has elapsed since the inbound endpoint became active during the current session. |
| **All Request Count** | The total number of requests received and handled by the inbound endpoint during the selected time interval. |
| **Successful Request Count** | The total number of requests that were successfully executed by the inbound endpoint during the selected time interval. |
| **Error Count** | The total number of requests handled by the inbound endpoint during the selected time interval that have resulted in errors. |
| **Error Percentage** | The requests handled by the inbound endpoint during the selected time interval that have resulted in errors, as a percentage of the total number of requests received by the endpoint during that same time interval. |
| **Deployed Node Count** | The number of nodes in which the inbound endpoint is deployed. |
| **Request Rate** | A visualization of the total number of requests handled by the inbound endpoint during the selected time interval. |
| **Error Rate** | A visualization of the total number of errors that have occurred for the inbound endpoint during the selected time interval. |
| **Response Time** | A visualization of the time taken by the inbound endpoint to respond to requests during the selected time interval. |