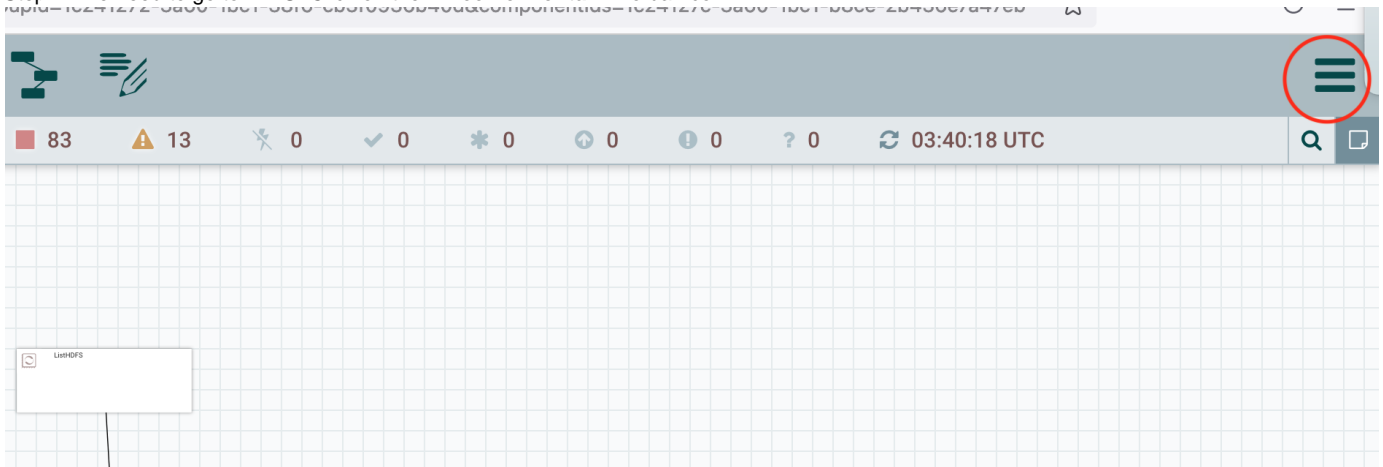













## Nifi Processors Monitoring

Step1:- We need to go to Nifi UI Click on the **Three horizontal Line** bar icon



Step2:- Click on **Controller Settings**

 03:45:25 UTC

- 
-  Summary
  -  Counters
  -  Bulletin Board
  -  Data Provenance
  -  Controller Settings
  - Parameter Contexts
  -  Flow Configuration History
  -  Node Status History
  -  Templates
  -  Help
  -  About

Step3:- Click on **REPORTING TASKS**

## NiFi Settings

GENERAL

REPORTING TASK CONTROLLER SERVICES

REPORTING TASKS

REGISTRY CLIENTS

Maximum Timer Driven Thread Count ?

10

Maximum Event Driven Thread Count ?

1

APPLY

Step4:- Create new reporting task by clicking on '+' button

ING TASK CONTROLLER SERVICES	REPORTING TASKS	REGISTRY CLIENTS
------------------------------	-----------------	------------------

+

	Type	Bundle	Run Status	
usReportingTask	PrometheusReportingTask 1.13.2	org.apache.nifi - nifi-prometheus-nar	▶ Running	■
usReportingTask - new test	PrometheusReportingTask 1.13.2	org.apache.nifi - nifi-prometheus-nar	▶ Pending	■

#### Step5:- Search for **PrometheusReportingTask**

## Add Reporting Task

Source

Displaying 1 of 15

prom

all groups ▾	Type ▲	Version	Tags
ambari azure datadog disk garbage collection gc heap jvm log log analytics memory <b>metrics</b> monitor monitoring prometheus provenance repo <b>reporting</b> restricted site site to site stats status storage warning	PrometheusReportingTask	1.13.2	prometheus, metrics, time serie...

**PrometheusReportingTask 1.13.2** org.apache.nifi - nifi-prometheus-nar

Reports metrics in Prometheus format by creating a /metrics HTTP(S) endpoint which can be used for external monitoring of the application. The reporting task reports a set of metrics regarding the JVM (optional) and the NiFi instance. Note that if the underlying Jetty server (i.e. the Prometheus endpoint) cannot be started (for example if two PrometheusReportingTask instances are started ...

CANCEL

ADD

Step6:-select on configure reporting task

	PrometheusReportingTask	PrometheusReportingTask 1.13.2	org.apache.nifi - nifi-prometheus-nar	running
	PrometheusReportingTask	PrometheusReportingTask 1.13.2	org.apache.nifi - nifi-prometheus-nar	Stopped

Step7:- Go to **PROPERTIES** for metrics, select the port no on which metrics to be emitted

## Configure Reporting Task

SETTINGS

PROPERTIES

COMMENTS

Required field



Property		Value
Prometheus Metrics Endpoint Port		9092
Instance ID		\${hostname(true)}
Metrics Reporting Strategy		All Components
Send JVM metrics		false
SSL Context Service		No value set
Client Authentication		No Authentication

CANCEL

APPLY

Step8:- We can also select different variables which we can monitor

Property		Value
Prometheus Metrics Endpoint Port		9092
Instance ID		
Metrics Reporting Strategy		All Components
Send JVM metrics		Root Process Group
SSL Context Service		All Process Groups
Client Authentication		All Components

Step9:- We can also set JVM metrics to true if we need JVM metrics also

## Configure Reporting Task

SETTINGS

PROPERTIES

COMMENTS

Required field



Property		Value
Prometheus Metrics Endpoint Port	?	9092
Instance ID	?	\${hostname(true)}
Metrics Reporting Strategy	?	All Components
Send JVM metrics	?	false
SSL Context Service	?	No value set
Client Authentication	?	No Authentication

CANCEL

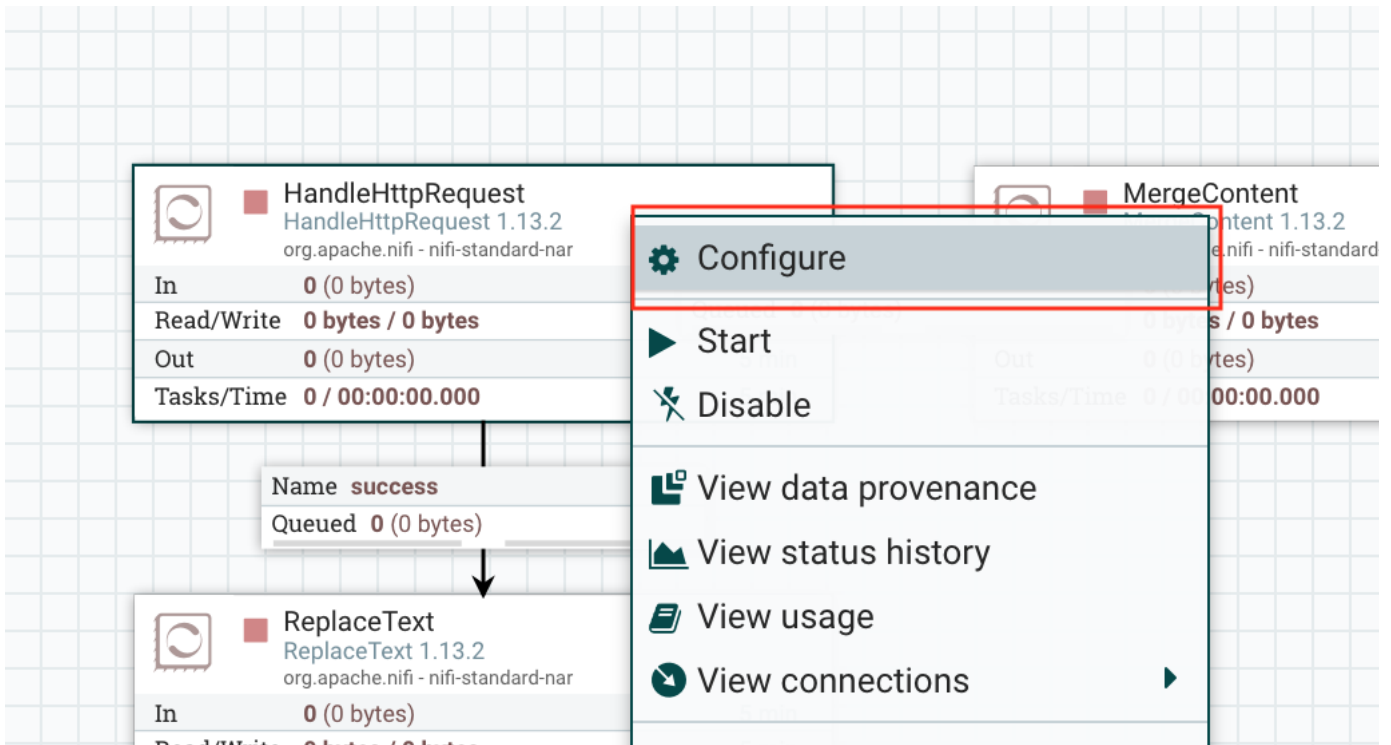
APPLY

Step10:- Check whether we are able to get metrics or not

<http://<hostip>:9092/metrics/>

Step11:- Check for the component id which component you want to monitor

For every component, they have a separate id in properties as shown below



**Configure Processor**

**Stopped**

**SETTINGS** | SCHEDULING | PROPERTIES | COMMENTS

Name: HandleHttpRequest ☒ Enabled

Id: 67c0e872-c8ee-3419-24a2-7d6ba6670ce3

Type: HandleHttpRequest 1.13.2

Bundle: org.apache.nifi - nifi-standard-nar

Penalty Duration: 30 sec Yield Duration: 1 sec

Bulletin Level: WARN

Automatically Terminate Relationships ☒ success

All content that is received is routed to the 'success' relationship

**CANCEL** **APPLY**

Step12:- We can select the required monitoring metrics accordingly from the available metrics

☐ Enable query history

nifi\_amo

**nifi\_amount\_bytes\_read**

**nifi\_amount\_bytes\_received**

**nifi\_amount\_bytes\_sent**

**nifi\_amount\_bytes\_transferred**

**nifi\_amount\_bytes\_written**

**nifi\_amount\_flowfiles\_received**

**nifi\_amount\_flowfiles\_removed**

**nifi\_amount\_flowfiles\_sent**

**nifi\_amount\_flowfiles\_transferred**

**nifi\_amount\_items\_input**

**nifi\_amount\_items\_output**

**nifi\_amount\_items\_queued**

**nifi\_amount\_threads\_active**

**nifi\_amount\_threads\_terminated**

Step13:- If we want to ensure a process ran successfully then we need to monitor ending components like nifi\_amount\_bytes\_recieved in a particular process so that we can figure out whether the whole job is completed or not.

Step14:- Set alerts in Prometheus rules accordingly  
An example alert rule for a component

```
alert: SaaS-Stage1(Common) warning stageazadeaingestioin FlowFile Processor is down
expr: rate(nifi_amount_bytes_received{component_id="05c00a-0171-4dd5-9271-430475b1"}[6h]) < 10
for: 1m
labels:
  severity: "critical"
annotations:
  summary: ""
```