

Prometheus

Part 1

Agenda

1- Prometheus Basics

- **What Is Prometheus?**
- **Prometheus Architecture — Bird's-Eye View**
- **Prometheus Use Cases — Strengths and Limitations**

2- Installation and Configuration

- **Building a Prometheus Server**
- **Configuring Prometheus**
- **Configuring an Exporter**

3- Prometheus Data Model

- **What Is Time-Series Data?**
- **Metrics and Labels**
- **Metric Types**

4- Querying

- **Query Basics**
- **Query Operators**
- **Query Functions**
- **Prometheus HTTP API**

Prometheus Basics

Introduction to Prometheus

Prometheus is an open-source **monitoring** and **alerting** tool. It collects data about applications and systems and allows you to **visualize** the data and issue **alerts** based on the data.

Monitoring is an important component of DevOps automation. To manage a robust and complex infrastructure, you need to be able to quickly and easily understand what is happening in your systems!



High-Level Use Cases

- **Metric collection** - Collect important metrics about your systems and applications in one place.
- **Visualization** - Build dashboards that provide an overview of the health of your systems.
- **Alerting** - Receive an email when something is broken.

Introduction to Prometheus

Prometheus Background

- **Language** - Prometheus is primarily written in **Go**. Some components are written in **Java**, **Python**, and **Ruby**.
- **License** - Prometheus uses the open-source **Apache 2.0** license.
- **History** - Prometheus development was started by **Matt T. Proud** and **Julius Volz**. It was initially sponsored by **SoundCloud**. Today, it is a fully open-source project maintained by many individuals and organizations.
- **Website** - More information and full documentation can be found at **prometheus.io**.

Prometheus Pull Model

Prometheus collects metrics using a **pull model**. This means the Prometheus server pulls metric data from exporters — agents do not push data to the Prometheus server.

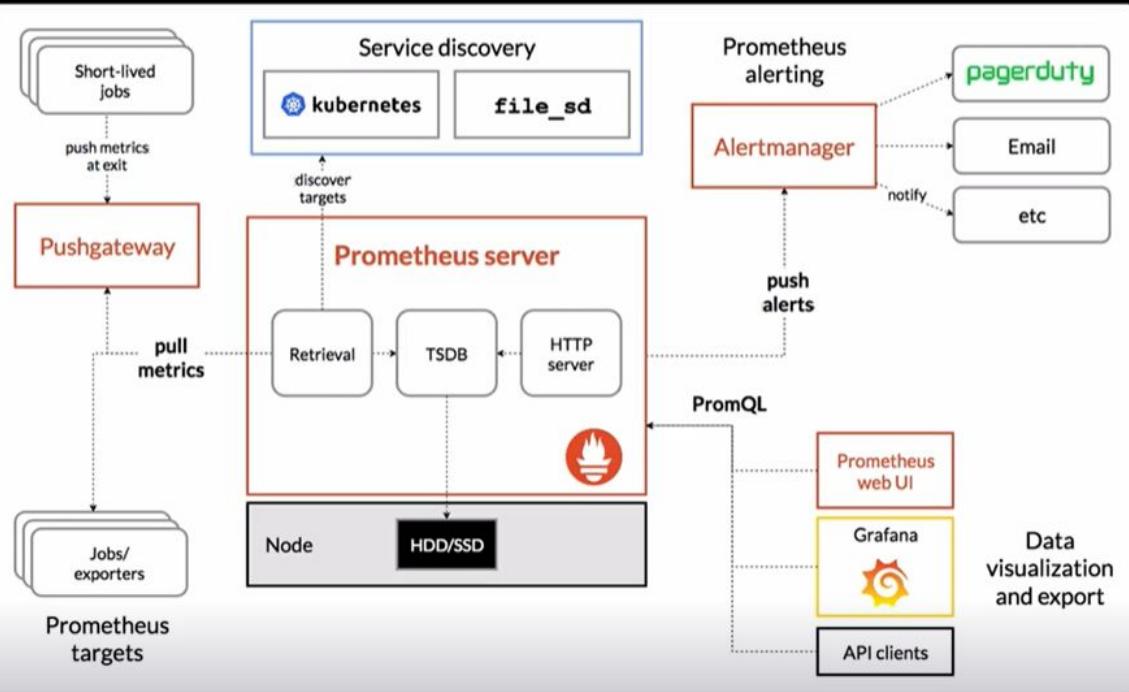
Introduction to Prometheus

Prometheus Architecture

Prometheus Pull Model

Prometheus collects metrics using a **pull model**. This means the Prometheus server pulls metric data from exporters — agents do not push data to the Prometheus server.

Introduction to Prometheus



- **Prometheus Server** - Collects metric data.
- **Exporters** - Provide metric data for Prometheus to consume.
- **Client Libraries** - Easily turn your custom application into an exporter that exposes metrics in a format Prometheus can consume.
- **Prometheus Pushgateway** - Allows pushing metrics to Prometheus for certain specific use cases.
- **Alertmanager** - Sends alerts triggered by metric data.
- **Visualization Tools** - Provide useful ways to view metric data. These are not necessarily part of Prometheus.

Prometheus Use Cases — Strengths and Limitations

Prometheus Use Case: Metric Collection

Judy is the administrator of a large infrastructure consisting of many servers, virtual machines, and components running in the cloud. She wants to be able to monitor the health of on-premises virtual machines as well as instances running in the cloud. Since all these components are interconnected, she does not want to have to search for data in multiple places when there is an outage.

Prometheus allows Judy to collect all this data in one place.

Prometheus Use Cases — Strengths and Limitations

Prometheus Use Case: Visualization

Mark's company has a complex IT infrastructure, and he is periodically on-call to respond to any outages that occur outside normal business hours. When an outage occurs, it often takes him a long time to discover which server or application is down. He wants to create a dashboard with graphs and charts that can show him what is happening at a glance so he can quickly diagnose problems and get back to sleep.

Prometheus allows Mark to build a dashboard that provides statistics on all his servers and applications. When something goes wrong, it is easy to spot where the problem lies.

Prometheus Use Cases — Strengths and Limitations

Prometheus Use Case: Alerting

Sarah's company has been having some problems with their network. She is frequently putting out fires when services become unavailable for customers. She is aware of ways to detect and fix these outages before they become serious, but it would be unrealistic to have someone constantly watch the data to determine when a problem is about to occur. She would like to build some automation to send her an email whenever the network issue is about to happen again.

With Prometheus, Sarah creates an alert that will trigger when the metric data indicates a problem is about to occur. Next time the issue is about to happen, she is able to take steps to prevent it.

Prometheus Use Cases — Strengths and Limitations

Prometheus Limitations

While Prometheus is a great tool for a variety of use cases, it is important to understand when it is not the best tool.

- **100% accuracy** (e.g., per-request billing) - Prometheus is designed to operate even under failure conditions. This means it will continue to provide data even if new data is not available due to failures and outages. If you need 100% up-to-the-minute accuracy, such as in the case of per-request billing, Prometheus may not be the best tool to use.
- **Non time-series data** (e.g., log aggregation) - Prometheus is built to monitor time-series metrics, especially data that is numeric. It is not the best choice for collecting more generic types of data, such as system logs.

Installation and Configuration

Building a Prometheus Server

Installation Options

There are multiple ways to install and run prometheus.

- Use pre-compiled binaries. Download from <https://prometheus.io/download/>.
- Build from source. Source code can be found on GitHub at <https://github.com/prometheus>.
- Run with Docker using pre-built images.

Building a Prometheus Server

```
cloud_user@wboyd1c:~$ sudo chown -R prometheus:prometheus /etc/prometheus
cloud_user@wboyd1c:~$ sudo chown prometheus:prometheus /var/lib/prometheus
cloud_user@wboyd1c:~$ prometheus --config.file=/etc/prometheus/prometheus.yml
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:295 msg="no time or size retention was set so using the default time retention" duration=15d
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:331 msg="Starting Prometheus" version="(version=2.16.0, branch=HEAD, revision=b90be6f32a33c03163d700e1452b54454ddce0ec)"
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:332 build_context="(go=go1.13.8, user=root@7ea0ae865f12, date=20200213-23:50:02)"
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:333 host_details="(Linux 4.15.0-1058-aws #60-Ubuntu SMP Wed Jan 15 22:35:20 UTC 2020 x86_64 wboyd1c.mylabserver.com (none))"
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:334 fd_limits="(soft=1024, hard=1048576)"
level=info ts=2020-02-18T18:31:30.892Z caller=main.go:335 vm_limits="(soft=unlimited, hard=unlimited)"
level=info ts=2020-02-18T18:31:30.903Z caller=main.go:661 msg="Starting TSDB ..."
level=info ts=2020-02-18T18:31:30.909Z caller=web.go:508 component=web msg="Start listening for connections" address=0.0.0.0:9090
level=info ts=2020-02-18T18:31:30.931Z caller=head.go:577 component=tsdb msg="replaying WAL, this may take awhile"
level=info ts=2020-02-18T18:31:30.931Z caller=head.go:625 component=tsdb msg="WAL segment loaded" segment=0 maxSegment=0
level=info ts=2020-02-18T18:31:30.937Z caller=main.go:676 fs_type=EXT4_SUPER_MAGIC
level=info ts=2020-02-18T18:31:30.937Z caller=main.go:677 msg="TSDB started"
level=info ts=2020-02-18T18:31:30.937Z caller=main.go:747 msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
level=info ts=2020-02-18T18:31:30.942Z caller=main.go:775 msg="Completed loading of configuration file" filename=/etc/prometheus/prometheus.yml
level=info ts=2020-02-18T18:31:30.943Z caller=main.go:630 msg="Server is ready to receive web requests."
```

Building a Prometheus Server

The prometheus.service file

```
[Unit]
Description=Prometheus Time Series Collection and Processing Server
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
--config.file /etc/prometheus/prometheus.yml \
--storage.tsdb.path /var/lib/prometheus/ \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries

[Install]
WantedBy=multi-user.target
```

~

Building a Prometheus Server

```
cloud_user@wboyd1c:~$ sudo vi /etc/systemd/system/prometheus.service
```

```
cloud_user@wboyd1c:~$ sudo systemctl daemon-reload
```

```
cloud_user@wboyd1c:~$ sudo systemctl start prometheus
```

```
cloud_user@wboyd1c:~$ sudo systemctl enable prometheus
```

```
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
```

```
cloud_user@wboyd1c:~$ █
```

The screenshot shows the Prometheus web interface at the URL `34.239.103.209:9090/graph`. The top navigation bar includes links for Prometheus, Alerts, Graph, Status, and Help. Below the navigation is a search bar with the placeholder "Expression (press Shift+Enter for newlines)". A "Execute" button is highlighted in blue. To the right of the search bar is a green "G" icon. Below the search bar, there are tabs for "Graph" and "Console", with "Graph" currently selected. Under the "Graph" tab, there are controls for time selection: "Moment" and arrows for navigating between time periods. At the bottom, a table displays the results of the query, with columns for "Element" and "Value". The table shows the message "no data".

Configuring Prometheus

The Prometheus Configuration File

Prometheus can be configured using a configuration file. Run Prometheus with the `--config.file` flag to specify the location of this file. We set up our `systemd` service to use the configuration file at `/etc/prometheus/prometheus.yml`.

The configuration file used the `YAML` format.

Refer to the Prometheus documentation for detailed information on all the available configuration options.

Configuring Prometheus

Reloading the Configuration

Restarting Prometheus will load the new configuration.

Prometheus can also reload its configuration at runtime without the need for a restart. One way to do this is to send a **SIGHUP** signal to the Prometheus process (e.g., `sudo killall -HUP prometheus`).

Configuring Prometheus

/etc/prometheus/prometheus.yml

```
# my global config
global:
  scrape_interval:      15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
      - targets:
        # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.
```

Configuring Prometheus

```
cloud_user@wboyd1c:~$ curl localhost:9090/api/v1/status/config
{"status": "success", "data": {"yaml": "global:\n  scrape_interval: 10s\n  scrape_timeout: 10s\n  evaluation_interval: 15s\n  alerting:\n    alertmanagers:\n      - static_configs:\n          - targets: []\n            scheme: http\n            timeout: 10s\n  api_version: v1\n  scrape_configs:\n    - job_name: prometheus\n      honor_timestamps: true\n      scrape_interval: 10s\n      scrape_timeout: 10s\n      metrics_path: /metrics\n      scheme: http\n      static_configs:\n        - targets:\n            - localhost:9090\n"} }cloud_user@wboyd1c:~$ █
```

Prometheus Exporter

Exporters

A Prometheus exporter is any application that exposes metric data in a format that can be collected (or "scraped") by the Prometheus server.

For example, [Node Exporter](#) runs on a [*NIX](#) machine and collects a variety of system metrics. It then exposes them to Prometheus.

Prometheus Exporter

Scrape Config

The `scrape_config` section of the Prometheus config file provides a list of targets the Prometheus server will scrape, such as a Node Exporter running on a Linux machine.

Prometheus server will scrape these targets periodically to collect metric data.

Prometheus Data Model

What Is Time-Series Data?

Prometheus is built around storing **time-series data**.

Time-series data consists of a series of values associated with different points in time.

Single Data Point vs. Time Series

You could track a single data point, such as the current outdoor temperature.

- Outdoor temperature: 0C/31F

However, if you write down the temperature once every hour, that's a time series!

- 08:00AM__ -6C/21F
- 09:00AM__ -3C/26F
- 10:00AM__ -2C/28F
- 11:00AM__ 0C/31F

Time-Series Metric Example

Every metric in Prometheus tracks a particular value over time.

For example, Prometheus might track the available memory for a server, with an entry in the time series for every minute.

```
02-21-2020 09:55AM - node_memory_MemAvailable_bytes=3734269952
02-21-2020 09:56AM - node_memory_MemAvailable_bytes=3734276545
02-21-2020 09:57AM - node_memory_MemAvailable_bytes=3734295563
02-21-2020 09:58AM - node_memory_MemAvailable_bytes=3734263327
```

What Is Time-Series Data?

Time-Series Data in Prometheus

All Prometheus data is fundamentally stored as **time-series data**.

This means Prometheus not only tracks the current **value** of each metric but also **changes** to each metric over time.

Metrics and Labels

Metric Names

Every metric in Prometheus has a metric name. The metric name refers to the general feature of a system or application that is being measured.

An example of a metric name:

```
node_cpu_seconds_total
```

`node_cpu_seconds_total` measures the total amount of CPU time being used in CPU seconds.

Metrics and Labels

Metric Names

Note that the metric name merely refers to the feature being measured. Metric names do not point to a specific data value but potentially a collection of many values.

Simply querying `node_cpu_seconds_total` would likely return a list of multiple data points, such as CPU usage for multiple CPUs on a server, or even multiple servers.

Metrics and Labels

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	152850.3
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	82.6
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9505.94
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.32
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	521.6
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3157.67
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5037.74
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	153417.49
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	87.77
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9071.04
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.74
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	522.77
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3084.21

Metrics and Labels

Metric Labels

Prometheus uses **labels** to provide a **dimensional data model**. This means we can use labels to specify additional things, such as which node's CPU usage is being represented.

A unique combination of a **metric name** and a set of **labels** identifies a particular set of **time-series data**. This example uses a label called **cpu** to refer to usage of a specific CPU.

```
node_cpu_seconds_total{cpu="0"}
```

Metrics and Labels

Metric Labels

Most Prometheus metrics have multiple labels.

```
node_cpu_seconds_total{cpu="0",instance="10.0.1.102:9100",mode="idle"} 287.75  
node_cpu_seconds_total{cpu="0",instance="10.0.1.102:9100",mode="user"} 51.45
```

Using metric names and labels, you can write queries that do things like average CPU usage across a whole data center as well as drill down into the CPU usage of a single CPU on a single node.

Metrics and Labels

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{cpu="0"}

Execute - insert metric at cursor ⌂

Graph Console

<< Moment >>

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	152850.3
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	82.6
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9505.94
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.32
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	521.6
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3157.67
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5037.74
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	153417.49
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	87.77
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9071.04
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.74
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	522.77
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3084.21

Load time: 54ms
Resolution: 14s
Total time series: 16

Metric Types

Metric Types

Metric types refer to different ways in which exporters represent the metric data they provide.

Metric types are not represented in any special way in a Prometheus server, but it is important to understand them in order to properly interpret your metrics.

Metric Types

Counter

A **counter** is a single number that can only increase or be reset to zero. Counters represent cumulative values.

Total HTTP requests served:

- + - 0
- + - 12
- + - 85
- + - 276

Examples:

- + - Number of HTTP requests served by an application
- + - Number of records processed
- + - Number of application restarts
- + - Number of errors

Metric Types

Prometheus Time Series Collector X +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%5B5m%5D&g0.tab=1

☆ W

Prometheus Alerts Graph Status Help

node_cpu_seconds_total[5m]

Load time: 81ms
Resolution: 14s
Total time series: 16

Execute - insert metric at cursor

Graph Console

<< Moment >>

Element	Value
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	153482.98 @1584114820.838 153491.43 @1584114830.838 153499.31 @1584114840.838 153507.12 @1584114850.838 153515 @1584114860.838 153522.7 @1584114870.838 153530.88 @1584114880.838 153539.11 @1584114890.838 153547.56 @1584114900.838 153555.7 @1584114910.838 153563.45 @1584114920.838 153571.83 @1584114930.838 153579.53 @1584114940.838 153587.3 @1584114950.838 153595.02 @1584114960.838 153602.73 @1584114970.839 153612.19 @1584114980.838 153621.87 @1584114990.838 153631.41 @1584115000.838 153641.02 @1584115010.838 153650.48 @1584115020.838 153660.27 @1584115030.838 153670.01 @1584115040.838 153679.74 @1584115050.838 153689.52 @1584115060.838 153699.30 @1584115070.838 153709.08 @1584115080.838 153718.86 @1584115090.838 153728.64 @1584115100.838 153738.42 @1584115110.838 153748.20 @1584115120.838 153757.98 @1584115130.838 153767.76 @1584115140.838 153777.54 @1584115150.838 153787.32 @1584115160.838 153797.10 @1584115170.838 153806.88 @1584115180.838 153816.66 @1584115190.838 153826.44 @1584115200.838 153836.22 @1584115210.838 153846.00 @1584115220.838 153855.78 @1584115230.838 153865.56 @1584115240.838 153875.34 @1584115250.838 153885.12 @1584115260.838 153894.90 @1584115270.838 153904.68 @1584115280.838 153914.46 @1584115290.838 153924.24 @1584115300.838 153934.02 @1584115310.838 153943.80 @1584115320.838 153953.58 @1584115330.838 153963.36 @1584115340.838 153973.14 @1584115350.838 153982.92 @1584115360.838 153992.70 @1584115370.838 154002.48 @1584115380.838 154012.26 @1584115390.838 154022.04 @1584115400.838 154031.82 @1584115410.838 154041.60 @1584115420.838 154051.38 @1584115430.838 154061.16 @1584115440.838 154070.94 @1584115450.838 154080.72 @1584115460.838 154090.50 @1584115470.838 154100.28 @1584115480.838 154110.06 @1584115490.838 154120.84 @1584115500.838 154130.62 @1584115510.838 154140.40 @1584115520.838 154150.18 @1584115530.838 154160.96 @1584115540.838 154170.74 @1584115550.838 154180.52 @1584115560.838 154190.30 @1584115570.838 154200.08 @1584115580.838 154210.86 @1584115590.838 154220.64 @1584115600.838 154230.42 @1584115610.838 154240.20 @1584115620.838 154250.98 @1584115630.838 154260.76 @1584115640.838 154270.54 @1584115650.838 154280.32 @1584115660.838 154290.10 @1584115670.838 154300.88 @1584115680.838 154310.66 @1584115690.838 154320.44 @1584115700.838 154330.22 @1584115710.838 154340.00 @1584115720.838 154350.78 @1584115730.838 154360.56 @1584115740.838 154370.34 @1584115750.838 154380.12 @1584115760.838 154390.90 @1584115770.838 154400.68 @1584115780.838 154410.46 @1584115790.838 154420.24 @1584115800.838 154430.02 @1584115810.838 154440.80 @1584115820.838 154450.58 @1584115830.838 154460.36 @1584115840.838 154470.14 @1584115850.838 154480.92 @1584115860.838 154490.70 @1584115870.838 154500.48 @1584115880.838 154510.26 @1584115890.838 154520.04 @1584115900.838 154530.82 @1584115910.838 154540.60 @1584115920.838 154550.38 @1584115930.838 154560.16 @1584115940.838 154570.94 @1584115950.838 154580.72 @1584115960.838 154590.50 @1584115970.838 154600.28 @1584115980.838 154610.06 @1584115990.838 154620.84 @1584116000.838 154630.62 @1584116010.838 154640.40 @1584116020.838 154650.18 @1584116030.838 154660.96 @1584116040.838 154670.74 @1584116050.838 154680.52 @1584116060.838 154690.30 @1584116070.838 154700.08 @1584116080.838 154710.86 @1584116090.838 154720.64 @1584116100.838 154730.42 @1584116110.838 154740.20 @1584116120.838 154750.98 @1584116130.838 154760.76 @1584116140.838 154770.54 @1584116150.838 154780.32 @1584116160.838 154790.10 @1584116170.838 154800.88 @1584116180.838 154810.66 @1584116190.838 154820.44 @1584116200.838 154830.22 @1584116210.838 154840.00 @1584116220.838 154850.78 @1584116230.838 154860.56 @1584116240.838 154870.34 @1584116250.838 154880.12 @1584116260.838 154890.90 @1584116270.838 154900.68 @1584116280.838 154910.46 @1584116290.838 154920.24 @1584116300.838 154930.02 @1584116310.838 154940.80 @1584116320.838 154950.58 @1584116330.838 154960.36 @1584116340.838 154970.14 @1584116350.838 154980.92 @1584116360.838 154990.70 @1584116370.838 155000.48 @1584116380.838 155010.26 @1584116390.838 155020.04 @1584116400.838 155030.82 @1584116410.838 155040.60 @1584116420.838 155050.38 @1584116430.838 155060.16 @1584116440.838 155070.94 @1584116450.838 155080.72 @1584116460.838 155090.50 @1584116470.838 155100.28 @1584116480.838 155110.06 @1584116490.838 155120.84 @1584116500.838 155130.62 @1584116510.838 155140.40 @1584116520.838 155150.18 @1584116530.838 155160.96 @1584116540.838 155170.74 @1584116550.838 155180.52 @1584116560.838 155190.30 @1584116570.838 155200.08 @1584116580.838 155210.86 @1584116590.838 155220.64 @1584116600.838 155230.42 @1584116610.838 155240.20 @1584116620.838 155250.98 @1584116630.838 155260.76 @1584116640.838 155270.54 @1584116650.838 155280.32 @1584116660.838 155290.10 @1584116670.838 155300.88 @1584116680.838 155310.66 @1584116690.838 155320.44 @1584116700.838 155330.22 @1584116710.838 155340.00 @1584116720.838 155350.78 @1584116730.838 155360.56 @1584116740.838 155370.34 @1584116750.838 155380.12 @1584116760.838 155390.90 @1584116770.838 155400.68 @1584116780.838 155410.46 @1584116790.838 155420.24 @1584116800.838 155430.02 @1584116810.838 155440.80 @1584116820.838 155450.58 @1584116830.838 155460.36 @1584116840.838 155470.14 @1584116850.838 155480.92 @1584116860.838 155490.70 @1584116870.838 155500.48 @1584116880.838 155510.26 @1584116890.838 155520.04 @1584116900.838 155530.82 @1584116910.838 155540.60 @1584116920.838 155550.38 @1584116930.838 155560.16 @1584116940.838 155570.94 @1584116950.838 155580.72 @1584116960.838 155590.50 @1584116970.838 155600.28 @1584116980.838 155610.06 @1584116990.838 155620.84 @1584117000.838 155630.62 @1584117010.838 155640.40 @1584117020.838 155650.18 @1584117030.838 155660.96 @1584117040.838 155670.74 @1584117050.838 155680.52 @1584117060.838 155690.30 @1584117070.838 155700.08 @1584117080.838 155710.86 @1584117090.838 155720.64 @1584117100.838 155730.42 @1584117110.838 155740.20 @1584117120.838 155750.98 @1584117130.838 155760.76 @1584117140.838 155770.54 @1584117150.838 155780.32 @1584117160.838 155790.10 @1584117170.838 155800.88 @1584117180.838 155810.66 @1584117190.838 155820.44 @1584117200.838 155830.22 @1584117210.838 155840.00 @1584117220.838 155850.78 @1584117230.838 155860.56 @1584117240.838 155870.34 @1584117250.838 155880.12 @1584117260.838 155890.90 @1584117270.838 155900.68 @1584117280.838 155910.46 @1584117290.838 155920.24 @1584117300.838 155930.02 @1584117310.838 155940.80 @1584117320.838 155950.58 @1584117330.838 155960.36 @1584117340.838 155970.14 @1584117350.838 155980.92 @1584117360.838 155990.70 @1584117370.838 156000.48 @1584117380.838 156010.26 @1584117390.838 156020.04 @1584117400.838 156030.82 @1584117410.838 156040.60 @1584117420.838 156050.38 @1584117430.838 156060.16 @1584117440.838 156070.94 @1584117450.838 156080.72 @1584117460.838 156090.50 @1584117470.838 156100.28 @1584117480.838 156110.06 @1584117490.838 156120.84 @1584117500.838 156130.62 @1584117510.838 156140.40 @1584117520.838 156150.18 @1584117530.838 156160.96 @1584117540.838 156170.74 @1584117550.838 156180.52 @1584117560.838 156190.30 @1584117570.838 156200.08 @1584117580.838 156210.86 @1584117590.838 156220.64 @1584117600.838 156230.42 @1584117610.838 156240.20 @1584117620.838 156250.98 @1584117630.838 156260.76 @1584117640.838 156270.54 @1584117650.838 156280.32 @1584117660.838 156290.10 @1584117670.838 156300.88 @1584117680.838 156310.66 @1584117690.838 156320.44 @1584117700.838 156330.22 @1584117710.838 156340.00 @1584117720.838 156350.78 @1584117730.838 156360.56 @1584117740.838 156370.34 @1584117750.838 156380.12 @1584117760.838 156390.90 @1584117770.838 156400.68 @1584117780.838 156410.46 @1584117790.838 156420.24 @1584117800.838 156430.02 @1584117810.838 156440.80 @1584117820.838 156450.58 @1584117830.838 156460.36 @1584117840.838 156470.14 @1584117850.838 156480.92 @1584117860.838 156490.70 @1584117870.838 156500.48 @1584117880.838 156510.26 @1584117890.838 156520.04 @1584117900.838 156530.82 @1584117910.838 156540.60 @1584117920.838 156550.38 @1584117930.838 156560.16 @1584117940.838 156570.94 @1584117950.838 156580.72 @1584117960.838 156590.50 @1584117970.838 156600.28 @1584117980.838 156610.06 @1584117990.838 156620.84 @1584118000.838 156630.62 @1584118010.838 156640.40 @1584118020.838 156650.18 @1584118030.838 156660.96 @1584118040.838 156670.74 @1584118050.838 156680.52 @1584118060.838 156690.30 @1584118070.838 156700.08 @1584118080.838 156710.86 @1584118090.838 156720.64 @1584118100.838 156730.42 @1584118110.838 156740.20 @1584118120.838 156750.98 @1584118130.838 156760.76 @1584118140.838 156770.54 @1584118150.838 156780.32 @1584118160.838 156790.10 @1584118170.838 156800.88 @1584118180.838 156810.66 @1584118190.838 156820.44 @1584118200.838 156830.22 @1584118210.838 156840.00 @1584118220.838 156850.78 @1584118230.838 156860.56 @1584118240.838 156870.34 @1584118250.838 156880.12 @1584118260.838 156890.90 @1584118270.838 156900.68 @1584118280.838 156910.46 @1584118290.838 156920.24 @1584118300.838 156930.02 @1584118310.838 156940.80 @1584118320.838 156950.58 @1584118330.838 156960.36 @1584118340.838 156970.14 @1584118350.838 156980.92 @1584118360.838 156990.70 @1584118370.838 157000.48 @1584118380.838 157010.26 @1584118390.838 157020.04 @1584118400.838 157030.82 @1584118410.838 157040.60 @1584118420.838 157050.38 @1584118430.838 157060.16 @1584118440.838 157070.94 @1584118450.838 157080.72 @1584118460.838 157090.50 @1584118470.838 157100.28 @1584118480.838 157110.06 @1584118490.838 157120.84 @1584118500.838 157130.62 @1584118510.838 157140.40 @1584118520.838 157150.18 @1584118530.838 157160.96 @1584118540.838 157170.74 @1584118550.838 157180.52 @1584118560.838 157190.30 @1584118570.838 157200.08 @1584118580.838 157210.86 @1584118590.838 157220.64 @1584118600.838 157230.42 @1584118610.838 157240.20 @1584118620.838 157250.98 @1584118630.838 157260.76 @1584118640.838 157270.54 @1584118650.838 157280.32 @1584118660.838 157290.10 @1584118670.838 157300.88 @1584118680.838 157310.66 @1584118690.838 157320.44 @1584118700.838 157330.22 @1584118710.838 157340.00 @1584118720.838 157350.78 @1584118730.838 157360.56 @1584118740.838 157370.34 @1584118750.838 157380.12 @1584118760.838 157390.90 @1584118770.838 157400.68 @1584118780.838 157410.46 @1584118790.838 157420.24 @1584118800.838 157430.02 @1584118810.838 157440.80 @1584118820.838 157450.58 @1584118830.838 157460.36 @1584118840.838 157470.14 @1584118850.838 157480.92 @1584118860.838 157490.70 @1584118870.838 157500.48 @1584118880.838 157510.26 @1584118890.838 157520.04 @1584118900.838 157530.82 @1584118910.838 157540.60 @1584118920.838 157550.38 @1584118930.838 157560.16 @1584118940.838 157570.94 @1584118950.838 157580.72 @1584118960.838 157590.50 @1584118970.838 157600.28 @1584118980.838 157610.06 @1584118990.838 157620.84 @1584119000.838 157630.62 @1584119010.838 157640.40 @1584119020.838 157650.18 @1584119030.838 157660.96 @1584119040.838 157670.74 @1584119050.838 157680.52 @1584119060.838 157690.30 @1584119070.838 157700.08 @1584119080.838 157710.86 @1584119090.838 157720.64 @1584119100.838 157730.42 @1584119110.838 157740.20 @1584119120.838 157750.98 @1584119130.838 157760.76 @1584119140.838 157770.54 @1584119150.838 157780.32 @1584119160.838 157790.10 @1584119170.838 157800.88 @1584119180.838 157810.66 @1584119190.838 157820.44 @1584119200.838 157830.22 @1584119210.838 157840.00 @1584119220.838 157850.78 @1584119230.838 157860.56 @1584119240.838 157870.34 @1584119250.838 157880.12 @1584119260.838 157890.90 @1584119270.838 157900.68 @1584119280.838 157910.46 @1584119290.838 157920.24 @1584119300.838 157930.02 @1584119310.838 157940.80 @1584119320.838 157950.58 @1584119330.838 157960.36 @1584119340.838 157970.14 @1584119350.838 157980.92 @1584119360.838 157990.70 @1584119370.838 158000.48 @1584119380.838 158010.26 @1584119390.838 158020.04 @1584119400.838 158030.82 @1584119410.838 158040.60 @1584119420.838 158050.38 @1584119430.838 158060.16 @1584119440.838 158070.94 @1584119450.838 158080.72 @1584119460.838<br

Metric Types

Gauge

A **gauge** is a single number that can increase and decrease over time.

Current HTTP requests active:

- +- 76
- +- 82
- +- 24
- +- 56

Examples:

- +- Number of concurrent HTTP requests
- +- CPU usage
- +- Memory usage
- +- Current active threads

Metric Types

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_memory_MemAvailable_bytes%5B5m%5D&g0.tab=1 ☆ W

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_memory_MemAvailable_bytes[5m]

Execute - insert metric at cursor ↗

Graph Console

◀ Moment ▶

Element	Value
node_memory_MemAvailable_bytes{instance="172.31.110.170:9100",job="Linux Server"}	1589780480 @1584115120.838 1589809152 @1584115130.838 1589555200 @1584115140.838 1589555200 @1584115150.838 1589555200 @1584115160.838 1588375552 @1584115170.838 1589686272 @1584115180.838 1589690368 @1584115190.838 1589690368 @1584115200.838 1588785152 @1584115210.838 1580937216 @1584115220.838 1581064192 @1584115230.838 1581064192 @1584115240.838 1581322240 @1584115250.838 1581322240 @1584115260.838 1581322240 @1584115270.838 1581576192 @1584115280.838 1581576192 @1584115290.838 1581576192 @1584115300.838 1581322240 @1584115310.838

Metric Types

Histogram

A **histogram** counts the number of observations/events that fall into a set of configurable buckets, each with its own separate time series. A histogram will use labels to differentiate between buckets. The below example provides the number of HTTP requests whose duration falls into each bucket.

```
http_request_duration_seconds_bucket{le="0.3"}  
http_request_duration_seconds_bucket{le="0.6"}  
http_request_duration_seconds_bucket{le="1.0"}
```

Histograms also include separate metric names to expose the **_sum** of all observed values and the total **_count** of events.

```
http_request_duration_seconds_sum  
http_request_duration_seconds_count
```

Metric Types

Prometheus Time Series Collector | Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=prometheus_http_request_duration_seconds_bucket&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

prometheus_http_request_duration_seconds_bucket

Load time: 86ms Resolution: 14s Total time series: 100

Execute - insert metric at cursor

Graph Console

Moment

Element	Value
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="0.1")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="0.2")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="0.4")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="1")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="3")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="8")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="20")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="60")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="120")	160672
prometheus_http_request_duration_seconds_bucket(handler="/metrics",instance="localhost:9090",job="prometheus",le="+Inf")	160672
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="0.1")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="0.2")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="0.4")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="1")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="3")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="8")	161
prometheus_http_request_duration_seconds_bucket(handler="/api/v1/query",instance="localhost:9090",job="prometheus",le="20")	161

Metric Types

Histogram

A **histogram** counts the number of observations/events that fall into a set of configurable buckets, each with its own separate time series. A histogram will use labels to differentiate between buckets. The below example provides the number of HTTP requests whose duration falls into each bucket.

```
http_request_duration_seconds_bucket{le="0.3"}  
http_request_duration_seconds_bucket{le="0.6"}  
http_request_duration_seconds_bucket{le="1.0"}
```

Histograms also include separate metric names to expose the **_sum** of all observed values and the total **_count** of events.

```
http_request_duration_seconds_sum  
http_request_duration_seconds_count
```

Metric Types

Summary

A **summary** is similar to a histogram, but it exposes metrics in the form of **quantiles** instead of buckets. While buckets divide values based on specific boundaries, quantiles divide values based on the percentiles into which they fall.

This value represents the number of HTTP requests whose duration falls within the 95th percentile of all requests or the top 5% longest requests.

```
http_request_duration_seconds{quantile="0.95"})
```

Like histograms, summaries also expose the **_sum** and **_count** metrics.

Metric Types

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=go_gc_duration_seconds&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

go_gc_duration_seconds

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
go_gc_duration_seconds{instance="localhost:9090",job="prometheus",quantile="0"}	0.000007166
go_gc_duration_seconds{instance="localhost:9090",job="prometheus",quantile="0.25"}	0.000014016
go_gc_duration_seconds{instance="localhost:9090",job="prometheus",quantile="0.5"}	0.000016595
go_gc_duration_seconds{instance="localhost:9090",job="prometheus",quantile="0.75"}	0.000033455
go_gc_duration_seconds{instance="localhost:9090",job="prometheus",quantile="1"}	0.001832717
go_gc_duration_seconds{instance="172.31.110.170:9100",job="Linux Server",quantile="0"}	0.000007363
go_gc_duration_seconds{instance="172.31.110.170:9100",job="Linux Server",quantile="0.25"}	0.000011912
go_gc_duration_seconds{instance="172.31.110.170:9100",job="Linux Server",quantile="0.5"}	0.000025525
go_gc_duration_seconds{instance="172.31.110.170:9100",job="Linux Server",quantile="0.75"}	0.000043627
go_gc_duration_seconds{instance="172.31.110.170:9100",job="Linux Server",quantile="1"}	0.004201534

Add Graph Remove Graph

Querying

Querying

Querying allows you to access and work with your metric data in Prometheus.

You can use **PromQL** (Prometheus Query Language) to write queries and retrieve useful information from the metric data collected by Prometheus.

You can use Prometheus queries in a variety of ways to obtain and work with data.

- Expression browser
- Prometheus HTTP API
- Visualization tools such as Grafana

Query Basics

Selectors

The most basic component of the PromQL query is a **time-series selector**. A time-series selector is simply a metric name, optionally combined with labels.

Both of the following are valid time-series selectors.

```
node_cpu_seconds_total
```

```
node_cpu_seconds_total{cpu="0"}
```

When multiple values exist for a label in the Prometheus data and you do not specify the label in your query, Prometheus will simply return data for all the values of that label.

Query Basics

Prometheus Time Series Collector | Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156172.46
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	84.08
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9953.8
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.4
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	546.49
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3257.4
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5117.32
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156740.35
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	89.04
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9497.65
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.87
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	552.83
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3184.02
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	4971.16

Load time: 80ms
Resolution: 14s
Total time series: 16

Query Basics

Prometheus Time Series Collector < + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bcpu%3D"0"%7D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{cpu="0"}

Execute - insert metric at cursor · [Help](#)

Graph Console

Moment

Element Value

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156195.76
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	84.08
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9958.35
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.41
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	546.5
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3258.94
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5117.77

Add Graph Remove Graph

Query Basics

Label Matching

You can use a variety of operators to perform advanced matches based upon label values.

- + - `=`: Equals
- + - `!=`: Does not equal
- + - `=~`: Regex match
- + - `!~`: Does not regex match

```
node_cpu_seconds_total{cpu="0"}
```

```
node_cpu_seconds_total{cpu!="0"}
```

```
node_cpu_seconds_total{cpu=~"1*"}  
node_cpu_seconds_total{cpu!~"1*"}
```

Query Basics

Prometheus Time Series Collector x +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bcpu!%3D"0"%7D&g0.tab=1

☆ 🌐 🔍 🌐

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{cpu!="0"}

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	156832.26
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	89.05
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	9505.88
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	2.88
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	553.28
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system")	3189.37
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user")	4974.88

Remove Graph

Add Graph

Query Basics

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bcpu%3D"0"%7D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total(mode=~"s.*")

Load time: 49ms
Resolution: 14s
Total time series: 8

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	156832.26
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	89.05
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	9505.88
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	2.88
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	553.28
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system")	3189.37
node_cpu_seconds_total(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user")	4974.88

Add Graph Remove Graph

Query Basics

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total{model=~"user|system"}&g0.tab=1

☆ 🌐 W 🔍

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{model=~"user|system"}

Load time: 50ms
Resolution: 14s
Total time series: 12

Execute - insert metric at cursor •

Graph Console

Moment

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156377.27
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	84.14
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9969.46
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.42
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	547.38
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156950.71
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	89.05
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9506.3
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.88
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	553.57

Remove Graph

Query Basics

Range Vector Selectors

Since Prometheus data is fundamentally time-series data, you can select data points over a particular time range.

This query selects all values for the metric name and label recorded over the last two minutes.

```
node_cpu_seconds_total{cpu="0"}[2m]
```

Query Basics

Prometheus Time Series Collector | Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total{cpu="0"}&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{cpu="0"}

Execute - insert metric at cursor

Graph Console

Moment

Element Value

node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	156545.88
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	84.16
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	9969.57
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	3.42
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	54.758
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system")	3264.34
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user")	5119.91

Remove Graph

Query Basics

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bcpu%3D"0"%7D%5B2m%5D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{cpu="0"}[2m]

Load time: 52ms
Resolution: 14s
Total time series: 8

Execute - insert metric at cursor

Graph Console

Moment

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	156446.7 @1584118160.838 156456.62 @1584118170.838 156466.55 @1584118180.838 156476.48 @1584118190.838 156486.38 @1584118200.838 156496.3 @1584118210.838 156506.22 @1584118220.838 156516.15 @1584118230.838 156526.04 @1584118240.838 156535.95 @1584118250.838 156545.88 @1584118260.838 156555.8 @1584118270.838
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	84.15 @1584118160.838 84.15 @1584118170.838 84.15 @1584118180.838 84.15 @1584118190.838 84.15 @1584118200.838 84.15 @1584118210.838 84.15 @1584118220.838 84.15 @1584118230.838 84.15 @1584118240.838 84.16 @1584118250.838 84.16 @1584118260.838 84.17 @1584118270.838

Query Basics

Offset Modifier

You can use an **offset modifier** to provide a time offset to select data from the past, with or without a range selector.

Select the CPU usage from one hour ago:

```
node_cpu_seconds_total offset 1h
```

Select CPU usage values over a five-minute period one hour ago:

```
node_cpu_seconds_total[5m] offset 1h
```

Query Basics

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total[5m] offset 1h&g0.tab=1

← → C ⚠ Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total[5m] offset 1h

Execute - insert metric at cursor ↴

Graph Console

Moment

Element

Element	Value
node_cpu_seconds_total(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	153258.62 @1584114540.838 153266.89 @1584114550.838 153274.87 @1584114560.838 153282.61 @1584114570.838 153290.42 @1584114580.838 153298.47 @1584114590.838 153306.27 @1584114600.838 153314.23 @1584114610.838 153322.3 @1584114620.838 153330.54 @1584114630.838 153338.37 @1584114640.838 153346.63 @1584114650.838 153354.68 @1584114660.838 153362.63 @1584114670.838 153370.67 @1584114680.838 153379.16 @1584114690.838 153387.44 @1584114700.838 153395.34 @1584114710.838 153403.64 @1584114720.838 153411.7 @1584114730.838 153419.52 @1584114740.838 153427.26 @1584114750.838 153435.12 @1584114760.838 153442.89 @1584114770.838 153450.66 @1584114780.838 153458.43 @1584114790.838 153466.2 @1584114800.838 153473.97 @1584114810.838 153481.74 @1584114820.838 153489.51 @1584114830.838 153497.28 @1584114840.838 153505.05 @1584114850.838 153512.82 @1584114860.838 153520.59 @1584114870.838 153528.36 @1584114880.838 153536.13 @1584114890.838 153543.9 @1584114900.838 153551.67 @1584114910.838 153559.44 @1584114920.838 153567.21 @1584114930.838 153574.98 @1584114940.838 153582.75 @1584114950.838 153590.52 @1584114960.838 153598.29 @1584114970.838 153606.06 @1584114980.838 153613.83 @1584114990.838 153621.6 @1584115000.838 153629.37 @1584115010.838 153637.14 @1584115020.838 153644.91 @1584115030.838 153652.68 @1584115040.838 153660.45 @1584115050.838 153668.22 @1584115060.838 153675.99 @1584115070.838 153683.76 @1584115080.838 153691.53 @1584115090.838 153699.3 @1584115100.838 153707.07 @1584115110.838 153714.84 @1584115120.838 153722.61 @1584115130.838 153730.38 @1584115140.838 153738.15 @1584115150.838 153745.92 @1584115160.838 153753.69 @1584115170.838 153761.46 @1584115180.838 153769.23 @1584115190.838 153776.99 @1584115200.838 153784.76 @1584115210.838 153792.53 @1584115220.838 153800.3 @1584115230.838 153808.07 @1584115240.838 153815.84 @1584115250.838 153823.61 @1584115260.838 153831.38 @1584115270.838 153839.15 @1584115280.838 153846.92 @1584115290.838 153854.69 @1584115300.838 153862.46 @1584115310.838 153870.23 @1584115320.838 153877.99 @1584115330.838 153885.76 @1584115340.838 153893.53 @1584115350.838 153901.3 @1584115360.838 153909.07 @1584115370.838 153916.84 @1584115380.838 153924.61 @1584115390.838 153932.38 @1584115400.838 153940.15 @1584115410.838 153947.92 @1584115420.838 153955.69 @1584115430.838 153963.46 @1584115440.838 153971.23 @1584115450.838 153978.99 @1584115460.838 153986.76 @1584115470.838 153994.53 @1584115480.838 154002.3 @1584115490.838 154010.07 @1584115500.838 154017.84 @1584115510.838 154025.61 @1584115520.838 154033.38 @1584115530.838 154041.15 @1584115540.838 154048.92 @1584115550.838 154056.69 @1584115560.838 154064.46 @1584115570.838 154072.23 @1584115580.838 154080.0 @1584115590.838 154087.77 @1584115600.838 154095.54 @1584115610.838 154103.31 @1584115620.838 154111.08 @1584115630.838 154118.85 @1584115640.838 154126.62 @1584115650.838 154134.39 @1584115660.838 154142.16 @1584115670.838 154150.93 @1584115680.838 154158.7 @1584115690.838 154166.47 @1584115700.838 154174.24 @1584115710.838 154182.01 @1584115720.838 154189.78 @1584115730.838 154197.55 @1584115740.838 154205.32 @1584115750.838 154213.09 @1584115760.838 154220.86 @1584115770.838 154228.63 @1584115780.838 154236.4 @1584115790.838 154244.17 @1584115800.838 154251.94 @1584115810.838 154259.71 @1584115820.838 154267.48 @1584115830.838 154275.25 @1584115840.838 154283.02 @1584115850.838 154290.79 @1584115860.838 154298.56 @1584115870.838 154306.33 @1584115880.838 154314.1 @1584115890.838 154321.87 @1584115900.838 154329.64 @1584115910.838 154337.41 @1584115920.838 154345.18 @1584115930.838 154352.95 @1584115940.838 154360.72 @1584115950.838 154368.49 @1584115960.838 154376.26 @1584115970.838 154384.03 @1584115980.838 154391.8 @1584115990.838 154400.57 @1584116000.838 154408.34 @1584116010.838 154416.11 @1584116020.838 154423.88 @1584116030.838 154431.65 @1584116040.838 154439.42 @1584116050.838 154447.19 @1584116060.838 154454.96 @1584116070.838 154462.73 @1584116080.838 154470.5 @1584116090.838 154478.27 @1584116100.838 154485.04 @1584116110.838 154492.81 @1584116120.838 154500.58 @1584116130.838 154508.35 @1584116140.838 154516.12 @1584116150.838 154523.89 @1584116160.838 154531.66 @1584116170.838 154539.43 @1584116180.838 154547.2 @1584116190.838 154554.97 @1584116200.838 154562.74 @1584116210.838 154570.51 @1584116220.838 154578.28 @1584116230.838 154586.05 @1584116240.838 154593.82 @1584116250.838 154601.59 @1584116260.838 154609.36 @1584116270.838 154617.13 @1584116280.838 154624.9 @1584116290.838 154632.67 @1584116300.838 154640.44 @1584116310.838 154648.21 @1584116320.838 154655.98 @1584116330.838 154663.75 @1584116340.838 154671.52 @1584116350.838 154679.29 @1584116360.838 154687.06 @1584116370.838 154694.83 @1584116380.838 154702.6 @1584116390.838 154710.37 @1584116400.838 154718.14 @1584116410.838 154725.91 @1584116420.838 154733.68 @1584116430.838 154741.45 @1584116440.838 154749.22 @1584116450.838 154756.99 @1584116460.838 154764.76 @1584116470.838 154772.53 @1584116480.838 154780.3 @1584116490.838 154788.07 @1584116500.838 154795.84 @1584116510.838 154803.61 @1584116520.838 154811.38 @1584116530.838 154819.15 @1584116540.838 154826.92 @1584116550.838 154834.69 @1584116560.838 154842.46 @1584116570.838 154850.23 @1584116580.838 154857.0 @1584116590.838 154864.77 @1584116600.838 154872.54 @1584116610.838 154880.31 @1584116620.838 154888.08 @1584116630.838 154895.85 @1584116640.838 154903.62 @1584116650.838 154911.39 @1584116660.838 154919.16 @1584116670.838 154926.93 @1584116680.838 154934.7 @1584116690.838 154942.47 @1584116700.838 154950.24 @1584116710.838 154957.01 @1584116720.838 154964.78 @1584116730.838 154972.55 @1584116740.838 154980.32 @1584116750.838 154988.09 @1584116760.838 154995.86 @1584116770.838 155003.63 @1584116780.838 155011.4 @1584116790.838 155019.17 @1584116800.838 155026.94 @1584116810.838 155034.71 @1584116820.838 155042.48 @1584116830.838 155050.25 @1584116840.838 155058.02 @1584116850.838 155065.79 @1584116860.838 155073.56 @1584116870.838 155081.33 @1584116880.838 155089.1 @1584116890.838 155096.87 @1584116900.838 155104.64 @1584116910.838 155112.41 @1584116920.838 155120.18 @1584116930.838 155127.95 @1584116940.838 155135.72 @1584116950.838 155143.49 @1584116960.838 155151.26 @1584116970.838 155159.03 @1584116980.838 155166.8 @1584116990.838 155174.57 @1584117000.838 155182.34 @1584117010.838 155190.11 @1584117020.838 155197.88 @1584117030.838 155205.65 @1584117040.838 155213.42 @1584117050.838 155221.19 @1584117060.838 155228.96 @1584117070.838 155236.73 @1584117080.838 155244.5 @1584117090.838 155252.27 @1584117100.838 155260.04 @1584117110.838 155267.81 @1584117120.838 155275.58 @1584117130.838 155283.35 @1584117140.838 155291.12 @1584117150.838 155298.89 @1584117160.838 155306.66 @1584117170.838 155314.43 @1584117180.838 155322.2 @1584117190.838 155330.97 @1584117200.838 155338.74 @1584117210.838 155346.51 @1584117220.838 155354.28 @1584117230.838 155362.05 @1584117240.838 155369.82 @1584117250.838 155377.59 @1584117260.838 155385.36 @1584117270.838 155393.13 @1584117280.838 155400.9 @1584117290.838 155408.67 @1584117300.838 155416.44 @1584117310.838 155424.21 @1584117320.838 155431.98 @1584117330.838 155439.75 @1584117340.838 155447.52 @1584117350.838 155455.29 @1584117360.838 155463.06 @1584117370.838 155470.83 @1584117380.838 155478.6 @1584117390.838 155486.37 @1584117400.838 155494.14 @1584117410.838 155501.91 @1584117420.838 155509.68 @1584117430.838 155517.45 @1584117440.838 155525.22 @1584117450.838 155532.99 @1584117460.838 155540.76 @1584117470.838 155548.53 @1584117480.838 155556.3 @1584117490.838 155564.07 @1584117500.838 155571.84 @1584117510.838 155579.61 @1584117520.838 155587.38 @1584117530.838 155595.15 @1584117540.838 155602.92 @1584117550.838 155610.69 @1584117560.838 155618.46 @1584117570.838 155626.23 @1584117580.838 155633.9 @1584117590.838 155641.67 @1584117600.838 155649.44 @1584117610.838 155657.21 @1584117620.838 155664.98 @1584117630.838 155672.75 @1584117640.838 155680.52 @1584117650.838 155688.29 @1584117660.838 155696.06 @1584117670.838 155703.83 @1584117680.838 155711.6 @1584117690.838 155719.37 @1584117700.838 155727.14 @1584117710.838 155734.91 @1584117720.838 155742.68 @1584117730.838 155750.45 @1584117740.838 155758.22 @1584117750.838 155765.99 @1584117760.838 155773.76 @1584117770.838 155781.53 @1584117780.838 155789.3 @1584117790.838 155797.07 @1584117800.838 155804.84 @1584117810.838 155812.61 @1584117820.838 155820.38 @1584117830.838 155828.15 @1584117840.838 155835.92 @1584117850.838 155843.69 @1584117860.838 155851.46 @1584117870.838 155859.23 @1584117880.838 155866.9 @1584117890.838 155874.67 @1584117900.838 155882.44 @1584117910.838 155890.21 @1584117920.838 155897.98 @1584117930.838 155905.75 @1584117940.838 155913.52 @1584117950.838 155921.29 @1584117960.838 155929.06 @1584117970.838 155936.83 @1584117980.838 155944.6 @1584117990.838 155952.37 @1584118000.838 155960.14 @1584118010.838 155967.91 @1584118020.838 155975.68 @1584118030.838 155983.45 @1584118040.838 155991.22 @1584118050.838 156000.0 @1584118060.838 156007.77 @1584118070.838 156015.54 @1584118080.838 156023.31 @1584118090.838 156031.08 @1584118100.838 156038.85 @1584118110.838 156046.62 @1584118120.838 156054.39 @1584118130.838 156062.16 @1584118140.838 156070.93 @1584118150.838 156078.7 @1584118160.838 156086.47 @1584118170.838 156094.24 @1584118180.838 156102.01 @1584118190.838 156109.78 @1584118200.838 156117.55 @1584118210.838 156125.32 @1584118220.838 156133.09 @1584118230.838 156140.86 @1584118240.838 156148.63 @1584118250.838 156156.4 @1584118260.838 156164.17 @1584118270.838 156171.94 @1584118280.838 156179.71 @1584118290.838 156187.48 @1584118300.838 156195.25 @1584118310.838 156203.02 @1584118320.838 156210.79 @1584118330.838 156218.56 @1584118340.838 156226.33 @1584118350.838 156234.1 @1584118360.838 156241.87 @1584118370.838 156249.64 @1584118380.838 156257.41 @1584118390.838 156265.18 @1584118400.838 156272.95 @1584118410.838 156280.72 @1584118420.838 156288.49 @1584118430.838 156296.26 @1584118440.838 156304.03 @1584118450.838 156311.8 @1584118460.838 156319.57 @1584118470.838 156327.34 @1584118480.838 156335.11 @1584118490.838 156342.88 @1584118500.838 156350.65 @1584118510.838 156358.42 @1584118520.838 156366.19 @1584118530.838 156373.96 @1584118540.838 156381.73 @1584118550.838 156389.5 @1584118560.838 156397.27 @1584118570.838 156405.04 @1584118580.838 156412.81 @1584118590.838 156420.58 @1584118600.838 156428.35 @1584118610.838 156436.12 @1584118620.838 156443.89 @1584118630.838 156451.66 @1584118640.838 156459.43 @1584118650.838 156467.2 @1584118660.838 156474.97 @1584118670.838 156482.74 @1584118680.838 156490.51 @1584118690.838 156498.28 @1584118700.838 156506.05 @1584118710.838 156513.82 @1584118720.838 156521.59 @1584118730.838 156529.36 @1584118740.838 156537.13 @1584118750.838 156544.9 @1584118760.838 156552.67 @1584118770.838 156560.44 @1584118780.838 156568.21 @1584118790.838 156575.98 @1584118800.838 156583.75 @1584118810.838 156591.52 @1584118820.838 156599.29 @1584118830.838 156607.06 @1584118840.838 156614.83 @1584118850.838 156622.6 @1584118860.838 156630.37 @1584118870.838 156638.14 @1584118880.838 156645.91 @1584118890.838 156653.68 @1584118900.838 156661.45 @1584118910.838 156669.22 @1584118920.838 156676.99 @1584118930.838 156684.76 @1584118940.838 156692.53 @1584118950.838 156700.3 @1584118960.838 156708.07 @1584118970.838 156715.84 @1584118980.838 156723.61 @1584118990.838 156731.38 @1584119000.838 156739.15 @1584119010.838 156746.92 @1584119020.838 156754.69 @1584119030.838 156762.46 @1584119040.838 156770.23 @1584119050.838 156777.0 @1584119060.838 156784.77 @1584119070.838 156792.54 @1584119080.838 156800.31 @1584119090.838 156808.08 @1584119100.838 156815.85 @1584119110.838 156823.62 @1584119120.838 156831.39 @1584119130.838 156839.16 @1584119140.838 156846.93 @1584119150.838 156854.7 @1584119160.838 156862.47 @1584119170.838 156870.24 @1584119180.838 156878.01 @1584119190.838 156885.78 @1584119200.838 1

Query Operators

Operators

`Operators` allow you to perform calculations based upon your metric data.

Arithmetic Binary Operators

The following operators perform basic arithmetic on numeric values.

```
+  
+- +: Addition  
+- -: Subtraction  
+- *: Multiplication  
+- /: Division  
+- %: Modulo  
+- ^: Exponentiation
```

```
node_cpu_seconds_total * 2
```

Query Operators

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total

Load time: 67ms
Resolution: 14s
Total time series: 16

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	157676.99
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	84.92
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9974.86
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.44
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	551.05
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3270.54
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5123.7
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	158248.28
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	89.95
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9512.8
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.89
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	557.42
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3197.63
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	4979.02

Query Operators

Prometheus Time Series Collector +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20*g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total * 2

Execute - insert metric at cursor ↴

Graph Console

Moment

Element

Element	Value
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	315412.8
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	169.84
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	19949.76
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	6.88
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	1102.26
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system")	6541.64
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user")	10247.92
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	316555.84
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	179.94
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	19025.86
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	5.78
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	1115.06
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system")	6395.42
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user")	9958.14

Remove Graph

Query Operators

Matching Rules

When using operators, Prometheus uses matching rules to determine how to combine or compare records from two sets of data. By default, records only match if all of their labels match.

Use the following keywords to control matching behavior:

`ignoring(<label list>)` - Ignore the specified labels when matching.
`on(<label list>)` - Use only the specified labels when matching.

```
node_cpu_seconds_total{mode="system"} + ignoring(mode) node_cpu_seconds_total{mode="user"}
```

```
node_cpu_seconds_total{mode="system"} + on(cpu) node_cpu_seconds_total{mode="user"}
```

Query Operators

Prometheus Time Series Collector | Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bmode%3D"system"%7D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

```
node_cpu_seconds_total{mode="system"}
```

Execute - insert metric at cursor ↴

Graph Console

◀ Moment ▶

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.10.170:9100",job="Linux Server",mode="system"}	3272.66
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3199.08

Remove Graph

Query Operators

Prometheus Time Series Collector

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total{mode="system"}+node_cpu_seconds_total{mode="user"}&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{mode="system"} + node_cpu_seconds_total{mode="user"}

Execute - insert metric at cursor

Graph Console

Moment

Element Value

no data

Add Graph Remove Graph

Load time: 75ms
Resolution: 14s
Total time series: 0

Query Operators

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bmode%3D"system"%7D%20%2B%20ignoring(mode)%20node_cpu_seconds_total%7Bmode%3D"user"%7D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{mode="system"} + ignoring(mode) node_cpu_seconds_total{mode="user"}

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

{cpu="0",instance="172.31.110.170:9100",job="Linux Server"}	8401.150000000001
{cpu="1",instance="172.31.110.170:9100",job="Linux Server"}	8181.42

Remove Graph Add Graph

The screenshot shows the Prometheus web interface with a query in the main input field: `node_cpu_seconds_total{mode="system"} + ignoring(mode) node_cpu_seconds_total{mode="user"}`. The results table displays two data points:

Element	Value
{cpu="0",instance="172.31.110.170:9100",job="Linux Server"}	8401.150000000001
{cpu="1",instance="172.31.110.170:9100",job="Linux Server"}	8181.42

Query Operators

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%7Bmode%3D"system"%7D%20%2B%20on(cpu)%20node_cpu_seconds_total%7Bmode%3D"user"%7D&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total{mode="system"} + on(cpu) node_cpu_seconds_total(mode="user")

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
{cpu="0"}	8401.42
{cpu="1"}	8181.72

Remove Graph Add Graph

Load time: 50ms
Resolution: 14s
Total time series: 2

Query Operators

Comparison Binary Operators

By default, comparison operators filter results to only those where the comparison evaluates as true.

```
+-- ==: Equal  
+-- !=: Not equal  
+-- >: Greater than  
+-- <: Less than  
+-- >=: Greater than or equal  
+-- <=: Less than or equal
```

```
node_cpu_seconds_total == 0
```

Use the `bool` modifier to instead return the boolean result of the comparison.

```
node_cpu_seconds_total == bool 0
```

Query Operators

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20%3D%200&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total == 0

Execute - insert metric at cursor ↴

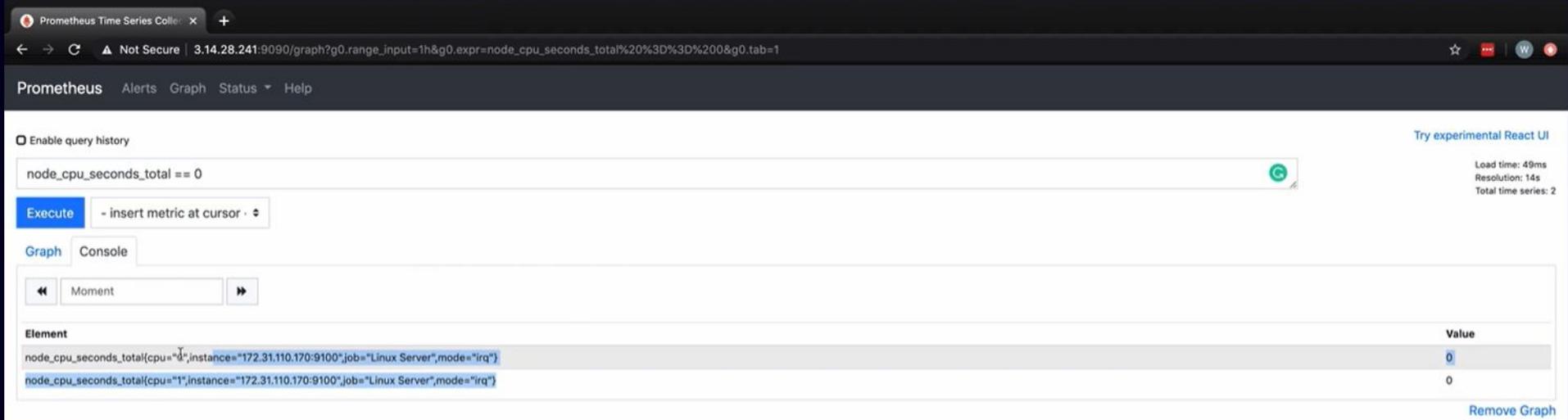
Graph Console

Moment

Element Value

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0

Remove Graph



Query Operators

Prometheus Time Series Collector × + Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20!%3D%200&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total != 0

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	158216.58
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	85.01
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9975.89
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.46
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	552.98
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3274.45
node_cpu_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	5127.67
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	158788.49
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	90.05
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	9515.07
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	2.89
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	559.57
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	3201.02
node_cpu_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	4981.5

Remove Graph

Query Operators

Prometheus Time Series Collector < + ▲ Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20%3D%3D%20bool%200&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total == bool 0

G Load time: 52ms
Resolution: 14s Total time series: 16

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	1
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	1
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	0
{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	0

Remove Graph

Query Operators

Logical/Set Binary Operators

These operators combine sets of results in various ways.

- +**- and**: Intersection
- +**- or**: Union
- +**- unless**: Complement

These operators use labels to compare records. For example, **and** returns only records where the set of labels in one set of results is matched by labels in the other set.

X

```
node_cpu_seconds_total and node_cpu_guest_seconds_total
```

Query Operators

Prometheus Time Series Collector | Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20and%20node_cpu_guest_seconds_total&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

node_cpu_seconds_total and node_cpu_guest_seconds_total

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
node_cpu_guest_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	0
node_cpu_guest_seconds_total{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	0
node_cpu_guest_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	0
node_cpu_guest_seconds_total{cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	0

Remove Graph

The screenshot shows the Prometheus Time Series Collector interface. At the top, there's a browser header with the URL '3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=node_cpu_seconds_total%20and%20node_cpu_guest_seconds_total&g0.tab=1'. Below it is the Prometheus navigation bar with links for Prometheus, Alerts, Graph, Status, and Help. The main area has a search bar containing 'node_cpu_seconds_total and node_cpu_guest_seconds_total' with a green 'Execute' button. Below the search bar are tabs for 'Graph' (which is selected) and 'Console'. A 'Moment' input field is present. The results section is titled 'Element Value' and contains four rows of data from the Prometheus database. The first two rows belong to 'cpu="0"' and the last two to 'cpu="1"'. Each row includes labels for 'cpu', 'instance', 'job', and 'mode'. The 'Value' column for all entries is '0'. In the bottom right corner of the results table, there's a blue 'Remove Graph' link.

Query Operators

Aggregation Operators

Aggregation operators combine multiple values into a single value.

```
+-- sum: Add all values together.  
+-- min: Select the smallest value.  
+-- max: Select the largest value.  
+-- avg: Calculate the average of all values.  
+-- stddev: Calculate population standard deviation over all values.  
+-- stdvar: Calculate population standard variance over all values.  
+-- count: Count number of values.  
+-- count_values: Count the number of values with the same value.  
+-- bottomk: Smallest number (k) of elements.  
+-- topk: Largest number (k) of elements.  
+-- quantile: Calculate the quantile for a particular dimension.
```

This query uses `avg` to get the average idle time between all CPUs.

```
avg(node_cpu_seconds_total{mode="idle"})
```

Query Operators

Prometheus Time Series Collector x +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=avg(node_cpu_seconds_total{mode="idle"})&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history

avg(node_cpu_seconds_total{mode="idle"})

Try experimental React UI

Load time: 53ms
Resolution: 14s
Total time series: 1

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

0	158957.29499999998
---	--------------------

Remove Graph

The screenshot shows the Prometheus web interface with a single data point for average idle CPU time. The query entered is 'avg(node_cpu_seconds_total{mode="idle"})'. The result table shows one element with a value of 158957.29499999998. The interface includes tabs for Graph and Console, and a 'Moment' selector. Top navigation includes Prometheus, Alerts, Graph, Status, and Help. A message at the top right indicates a total of 1 time series found.

Query Functions

Functions

Functions provide a wide array of built-in functionality to aid in the process of writing queries.

Consult the Prometheus documentation for a full list of available functions.

For example:

- +-- `abs()`: Calculates absolute value
- +-- `clamp_max()`: Returns values, but replaces them with a maximum value if they exceed that value
- +-- `clamp_min()`: Returns values, but replaces them with a minimum value if they are less than that value

Query Functions

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=clamp_max(node_cpu_seconds_total{cpu="0"}>0)*10000&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

clamp_max(node_cpu_seconds_total{cpu="0"}, 1000)

Execute - insert metric at cursor ↴

Graph Console

Moment

Element Value

Element	Value
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle"}	1000
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait"}	85.8
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq"}	0
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice"}	1000
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq"}	3.49
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal"}	557.57
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system"}	1000
{cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user"}	1000

Remove Graph

Query Functions

Prometheus Time Series Collector × +

Not Secure | 3.14.28.241:9090/graph?g0.range_input=1h&g0.expr=rate(node_cpu_seconds_total[5B1h%5D)&g0.tab=1

Prometheus Alerts Graph Status Help

Enable query history Try experimental React UI

rate(node_cpu_seconds_total[1h])

Load time: 65ms
Resolution: 14s
Total time series: 16

Execute - insert metric at cursor ↴

Graph Console

Moment

Element	Value
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	0.9605348189415062
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	0.0004456824512534803
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	0.009473537604456884
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	0.00001949860724233991
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	0.0029052924791086528
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="system")	0.010286908077994383
(cpu="0",instance="172.31.110.170:9100",job="Linux Server",mode="user")	0.01698328690807781
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="idle")	0.9618941504178304
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="iowait")	0.0002729805013927548
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="irq")	0
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="nice")	0.012222841225627024
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="softirq")	0.000008356545961002855
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="steal")	0.0032618384401114303
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="system")	0.009239554317548766
(cpu="1",instance="172.31.110.170:9100",job="Linux Server",mode="user")	0.013339832869080871

Prometheus HTTP API

HTTP API

Prometheus provides an **HTTP API** you can use to execute queries and obtain results using HTTP requests.

This API is a useful way to interact with Prometheus, especially if you are building your own custom tools that require access to Prometheus data.

Prometheus HTTP API

```
cloud_user@wboyd1c:~$ curl localhost:9090/api/v1/query?query=node_cpu_seconds_total
{"status":"success","data":{"resultType":"vector","result":[{"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "idle"}, "value": [1584122849.099, "160952.75"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "iowait"}, "value": [1584122849.099, "85.93"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "irq"}, "value": [1584122849.099, "0"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "nice"}, "value": [1584122849.099, "10011.58"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "softirq"}, "value": [1584122849.099, "3.5"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "steal"}, "value": [1584122849.099, "561.72"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "system"}, "value": [1584122849.099, "3309.92"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "user"}, "value": [1584122849.099, "5192.94"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "idle"}, "value": [1584122849.099, "161531.12"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "iowait"}, "value": [1584122849.099, "91.3"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "irq"}, "value": [1584122849.099, "0"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "nice"}, "value": [1584122849.099, "9563.83"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "softirq"}, "value": [1584122849.099, "2.92"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "steal"}, "value": [1584122849.099, "569.27"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "system"}, "value": [1584122849.099, "3230.92"]}, {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "1", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "user"}, "value": [1584122849.099, "5029.66"]}]}}cloud_user@wboyd1c:~$ █
```

Prometheus HTTP API

```
cloud_user@wboyd1c:~$ curl localhost:9090/api/v1/query --data-urlencode "query=node_cpu_seconds_total{cpu=\"0\"}"
{
  "status": "success",
  "data": {
    "resultType": "vector",
    "result": [
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "idle"}, "value": [1584122990.425, "161087.98"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "iowait"}, "value": [1584122990.425, "85.94"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "irq"}, "value": [1584122990.425, "0"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "nice"}, "value": [1584122990.425, "10013.84"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "softirq"}, "value": [1584122990.425, "3.5"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "steal"}, "value": [1584122990.425, "562.92"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "system"}, "value": [1584122990.425, "3310.24"]},
      {"metric": {"__name__": "node_cpu_seconds_total", "cpu": "0", "instance": "172.31.110.170:9100", "job": "Linux Server", "mode": "user"}, "value": [1584122990.425, "5194.69"]}
    ]
  }
}
cloud_user@wboyd1c:~$ █
```

Prometheus HTTP API

```
cloud_user@wboyd1c:~$ start=$(date --date '-5 min' +'%Y-%m-%dT%H:%M:%SZ')
cloud_user@wboyd1c:~$ end=$(date +'%Y-%m-%dT%H:%M:%SZ')
cloud_user@wboyd1c:~$ curl "localhost:9090/api/v1/query_range?query=node_cpu_seconds_total&start=$start&end=$end&step=1m"
{"status":"success","data":{"resultType":"matrix","result":[{"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"idle"},"values":[[1584122783,"160894.5"],[1584122843,"160952.75"],[1584122903,"161010.58"],[1584122963,"161068.83"],[1584123023,"161126.79"],[1584123083,"161184.48"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"iowait"},"values":[[1584122783,"85.93"],[1584122843,"85.93"],[1584122903,"85.93"],[1584122963,"85.94"],[1584123023,"85.95"],[1584123083,"85.95"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"irq"},"values":[[1584122783,"0"],[1584122843,"0"],[1584122903,"0"],[1584122963,"0"],[1584123023,"0"],[1584123083,"0"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"nice"},"values":[[1584122783,"10011"],[1584122843,"10011.58"],[1584122903,"10012.75"],[1584122963,"10013.32"],[1584123023,"10014.25"],[1584123083,"10015.58"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"softirq"},"values":[[1584122783,"3.5"],[1584122843,"3.5"],[1584122903,"3.5"],[1584122963,"3.5"],[1584123023,"3.5"],[1584123083,"3.5"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"steal"},"values":[[1584122783,"561.27"],[1584122843,"561.72"],[1584122903,"562.21"],[1584122963,"562.72"],[1584123023,"563.21"],[1584123083,"563.75"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"system"},"values":[[1584122783,"3309.77"],[1584122843,"3309.92"],[1584122903,"3310.06"],[1584122963,"3310.2"],[1584123023,"3310.36"],[1584123083,"3310.52"]]}, {"metric":{"__name__":"node_cpu_seconds_total","cpu":"0","instance":"172.31.110.170:9100","job":"Linux Server","mode":"user"},"values":[[1584122783,"5192.04"],[1584122843,"5192.94"],[1584122903,"5193.86"]]}]}
```

Thank You!