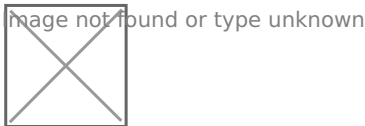


Monitor Your System with Grafana using Netdata and Prometheus

Prometheus is quite amazing! Here I show you how to use it to pull metrics from the Netdata API to display in a custom dashboard with Grafana.

The idea behind this project is to have a custom dashboard that displays only things you really want to see at a glance. You can choose to dig through the Prometheus Netdata metrics more to display anything you wish but below is the example I setup for this tutorial.

Prometheus does all the work. The only thing required to be installed on the machine being monitored is Netdata. No other agents or workers are needed that could utilize more system memory on your servers. That's the beauty of Prometheus. Rather than coming to your machine to collect data, Prometheus waits for data to come to it instead, using API calls.



Netdata is required to be installed on any machine that will be monitored. Please see [this video](#) before moving forward here.

I chose to host Grafana and Prometheus on their own separate LXCcontainer using Proxmox. You can use a RPi or any host you wish. I do this so I can use one central host to keep things more organized. You only need one host for Grafana and Prometheus.

Install Grafana

I use a docker stack through Portainer.

```
version: '3.3'
services:
  grafana:
```

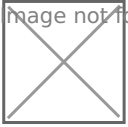
```
ports:
  - '3000:3000'

container_name: grafana

image: grafana/grafana
```

Grafana will then be accessible on port 3000. Login with admin/admin then choose a stronger password.

image not found or type unknown



Install Prometheus

Again, I use a docker stack through Portainer.

```
version: '3.3'

services:
  prometheus:
    ports:
      - '9090:9090'

    volumes:
      - '/docker/prometheus:/etc/prometheus'

    image: prom/prometheus
```

This will create a folder on your host machine at /docker/prometheus

Prometheus will then be accessible on port 9090.

image not found or type unknown



Create the prometheus.yml file

```
cd /docker/prometheus
```

```
touch prometheus.yml
```

Edit the prometheus.yml file

```
nano /docker/prometheus/prometheus.yml
```

Below is my example prometheus.yml file. You should change the IP to match that of your Netdata web UI. You should not have to change anything above the pound line, only that in between.

```
# 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'.

    static_configs:
      - targets: ['localhost:9090']

#####
###
```

```

- job_name: 'netdata-scrape'

metrics_path: '/api/v1/allmetrics'
params:
  # format: prometheus | prometheus_all_hosts
  # You can use `prometheus_all_hosts` if you want Prometheus to set the `instance` to
your hostname instead of IP
  format: [prometheus]
  #
  # source: as-collected | raw | average | sum | volume
  # default is: average
  #source: [as-collected]
  #
  # server name for this prometheus - the default is the client IP
  # for Netdata to uniquely identify it
  #server: ['prometheus1']
honor_labels: true
static_configs:
  - targets: ['192.168.1.13:19999']

#####
###

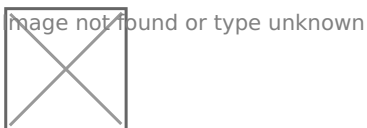
```

If you decide to monitor more than one Netdata instance just copy and paste from each pound line then edit the job name and target IP.

Prometheus Netdata Metrics

You can browse the metrics explorer by pressing the small globe icon next to the search bar or use the metrics I used in the video to start your dashboard. You can see those below.

On the Prometheus webui, copy and paste the following metrics in the finder then copy the query you wish to display in Grafana.



System Uptime

```
netdata_system_uptime_seconds_average
```

When adding this to Grafana be sure to select "seconds (s)" as the unit of measurement under Standard options.

CPU Temperature (requires sensors to be installed on the server using lm-sensors)

```
netdata_sensors_temperature_Celsius_average
```

When adding this to Grafana be sure to select "Celsius (°C)" as the unit of measurement under Standard options.

CPU Usage

```
netdata_cpu_cpu_percentage_average
```

When adding this to Grafana be sure to select "Percent (0-100)" as the unit of measurement under Standard options.

Memory Used

This was a tricky one. I had to take total memory and subtract the available memory from it to get an accurate number.

```
31970 - netdata_mem_available_MiB_average{instance="192.168.1.13:19999", job="netdata-scrape"}
```

When adding this to Grafana be sure to select "mebibytes" as the unit of measurement under Standard options.

As you can see, I have 32GB of RAM. I had to play with the numbers and used [bpytop](#) to compare. I was able to get the RAM spot on to match the same output in bpytop.

image not found or type unknown



image not found or type unknown



Hard Drive Space

```
netdata_disk_space_GiB_average
```

When adding this to Grafana be sure to select "gibibytes" as the unit of measurement under Standard options.

If you'd like to look at the JSON for my Grafana dashboard, please [view it here](#).

Video Tutorial

<https://www.youtube.com/embed/uimGcQVRaql>

Did you find this helpful? [Subscribe to me on Youtube](#) for more content!

Revision #23

Created 21 September 2021 14:04:06 by Jeremy

Updated 4 October 2021 13:18:43 by Jeremy