

Памятка:

	ВЫПОЛНЯЕМ ОДИНАКОВО КОМАНДЫ НА ОБОИХ ВЕБ-СЕРВЕРАХ
	ВЫПОЛНЯЕМ КОМАНДУ ИНДИВИДУАЛЬНО НА СЕРВЕРЕ

I

Устанавливаем пакет мониторинга Prometheus из репозитория Ubuntu:

```
sudo apt update  
sudo apt install prometheus
```

И добавим Prometheus в автозагрузку при загрузке ОС:

```
sudo systemctl enable prometheus
```

Также нам необходим инструмент визуализации данных метрик и для этого нам понадобится произвести установку Grafana. Для этого сначала нужно установить несколько компонентов, которые позволят работать с http протоколом:

ВНИМАНИЕ: на момент написания мануала, доступ для пользователей из РФ заблокирован! Можете прокрутить ниже, для установки пакета в ручном режиме с нашего сервера.

=====НАЧАЛО 1 СПОСОБА=====

Офици. сайт по установке: <https://grafana.com/grafana/download?pg=get&plcmt=selfmanaged-box1-cta1>

```
sudo apt install -y apt-transport-https  
sudo apt install -y software-properties-common
```

Затем добавим ключ нового репозитория для установки Grafana с официального сайта разработчика:

```
curl -s https://packages.grafana.com/gpg.key | sudo gpg --no-default-keyring --keyring gnupg-ring:/etc/apt/trusted.gpg.d/grafana_key.gpg --import
```

```
Reading state information... Done  
software-properties-common is already the newest version (0.99.22.9).  
software-properties-common set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 49 not upgraded.  
user@server:~$ curl -s https://packages.grafana.com/gpg.key | sudo gpg --no-default-keyring --keyring gnupg-ring:/etc/apt/trusted.gpg.d/grafana_key.gpg --import  
gpg: keyring '/etc/apt/trusted.gpg.d/grafana_key.gpg' created  
gpg: no valid OpenPGP data found.  
gpg: Total number processed: 0  
user@server:~$
```

И сам репозиторий:

```
echo "deb https://packages.grafana.com/enterprise/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```

```
user@server:~$ echo "deb https://packages.grafana.com/enterprise/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list  
deb https://packages.grafana.com/enterprise/deb stable main  
user@server:~$
```

Далее снова обновляем список репозитариев и устанавливаем Grafana

```
sudo apt update
```

```
sudo apt install grafana
```

=====КОНЕЦ=====

2 СПОСОБ:

Или же скачиваем пакет с нашего сервера и устанавливаем его утилитой dpkg:

```
wget http://testus.yakit.ru/grafana.deb
```

```
sudo dpkg -i grafana.deb
```

Далее dpkg ругнется, что нету пакетов зависимостей для установки Grafana, тогда выполним поиск и установку и снова повторим установку после этого:

```
sudo apt install -f
```

```
sudo dpkg -i grafana.deb
```

```
Processing triggers for man-db (2.10.2-1) ...  
needrestart is being skipped since dpkg has failed  
user@web-server1:~$ sudo dpkg -i grafana.deb  
(Reading database ... 122788 files and directories currently installed.)  
Preparing to unpack grafana.deb ...  
Unpacking grafana-enterprise (11.4.0) over (11.4.0) ...  
Setting up grafana-enterprise (11.4.0) ...  
Restarting grafana-server service... OK  
user@web-server1:~$
```

Также обратите внимание на автозапуск Grafana при старте ОС:

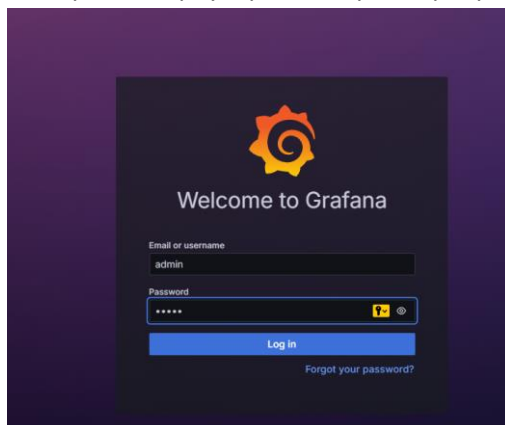
```
sudo /bin/systemctl daemon-reload  
sudo /bin/systemctl enable grafana-server
```

После установки проверим доступность в Графаны.

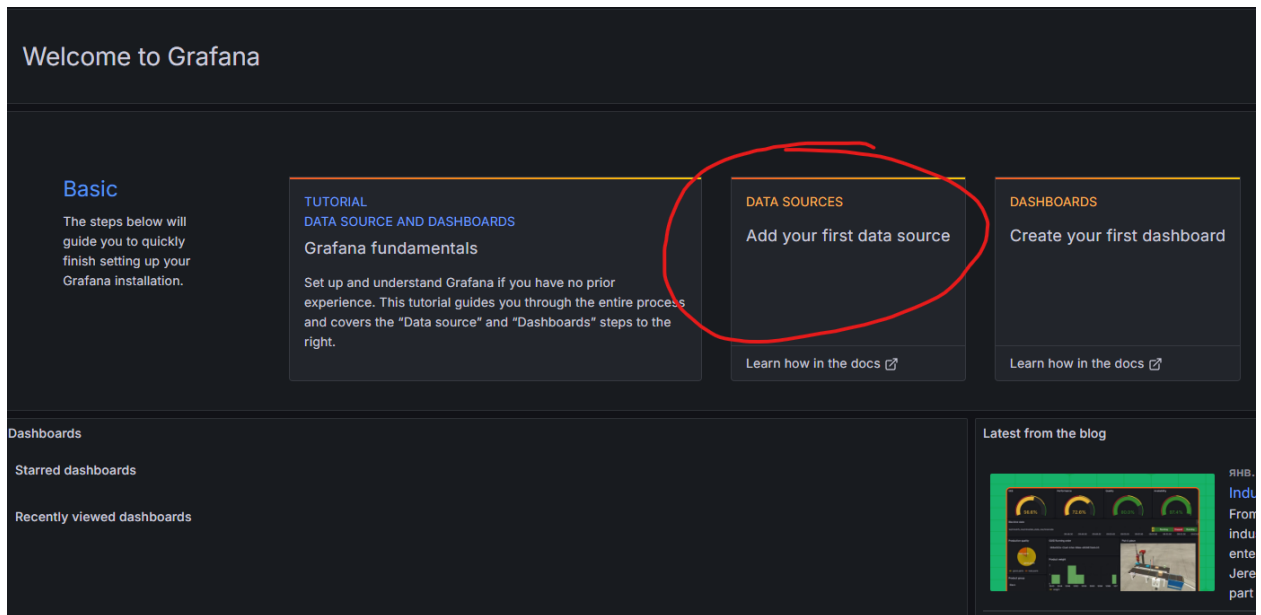
По умолчанию Grafana доступна на порту 3000. Логин и пароль при первом входе: **admin** и **admin**

После ввода пароля, приложение попросит ввести новый пароль. Можно оставить и старый admin

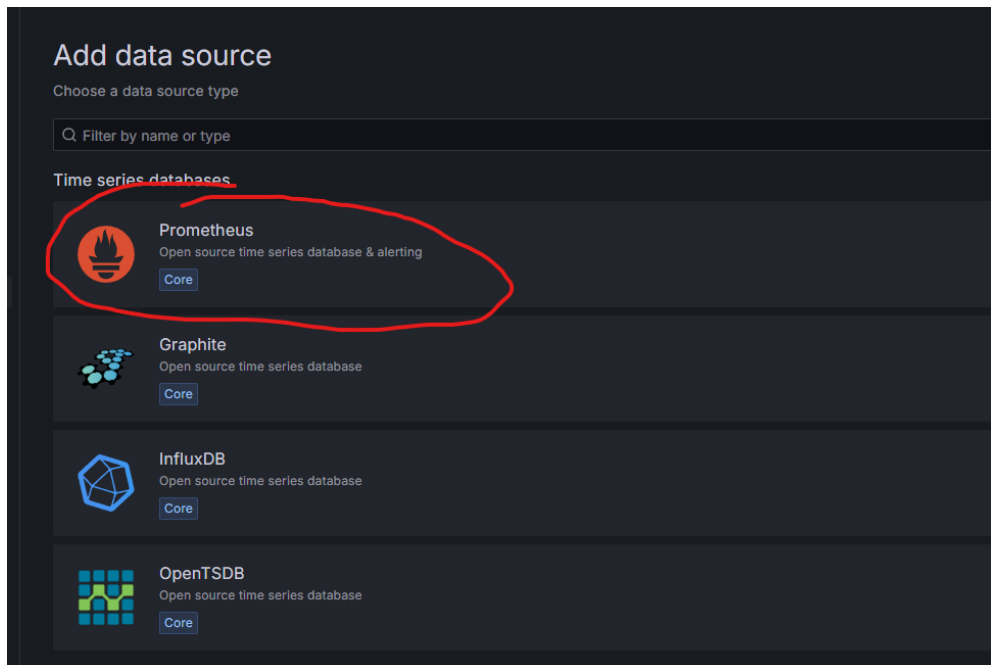
Набираем в браузере ип-адрес сервера, куда установили Grafana и порт 3000.



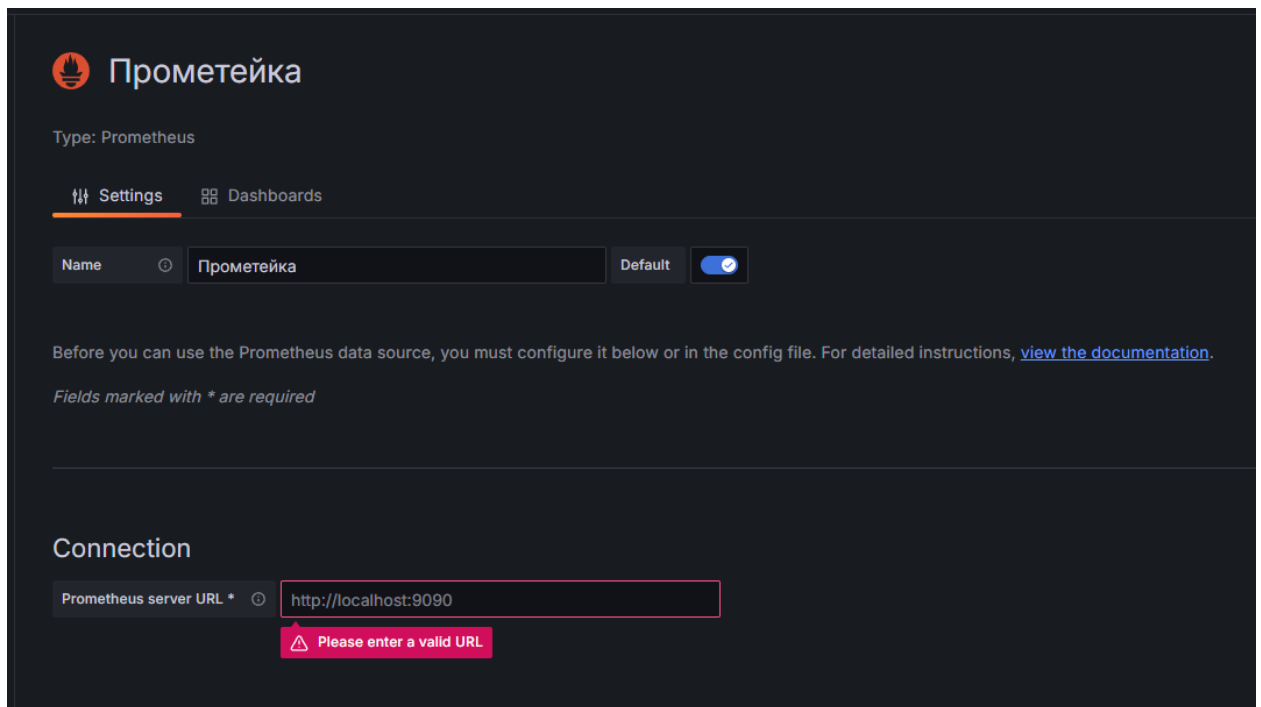
Теперь нам необходимо добавить источник данных для визуализации. В нашем случае это наш сервер с Prometheus. Нажимаем на Data Sources



И выбираем Prometheus:



Вводим адрес нашего Prometheus. Так как Grafana и Prometheus установлен на одном сервере, то в данном случае мы указываем вместо ип-адреса наш внутренний адрес и порт Prometheus – **http://localhost:9090**



Прометейка

Type: Prometheus

Settings Dashboards

Name ⓘ Прометейка Default ☒

Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, [view the documentation](#).

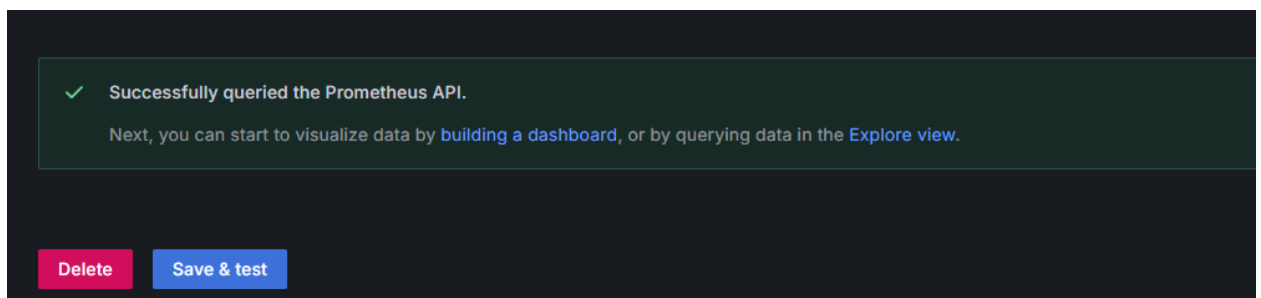
Fields marked with * are required

Connection

Prometheus server URL * ⓘ

Please enter a valid URL

И видим успешное уведомление, что Grafana увидела наш сервер Prometheus:



Далее нам необходимо повесить дашборд, который будет отображать метрики нашего сервера Prometheus. На сайте разработчика есть большой выбор дашбордов. Можно воспользоваться поиском по данной ссылке:

<https://grafana.com/grafana/dashboards/>

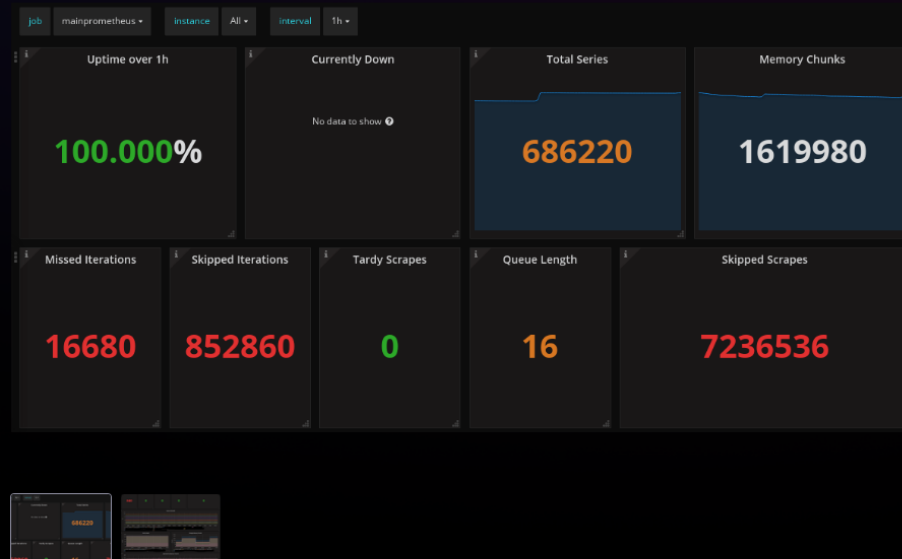
Перейдем для теста по ссылке ниже и установим дашборд:

<https://grafana.com/grafana/dashboards/3662-prometheus-2-0-overview/>

Зайдя на сайт и найдя необходимый нам дашборд, нам необходимо лишь указать в Grafana идентификатор этого дашборда для импорта в систему:

Prometheus 2.0 Overview

Get started faster with Grafana Cloud then easily build these dashboards. <https://grafana.com/products/cloud/>
Overview of metrics from Prometheus 2.0. Useful for using prometheus to monitor your prometheus. Revisions welcome!



Prometheus server provides its own metrics on /metrics. This dashboard graphs some of them, with intelligent templating (ie for when you have different jobs of prometheus servers).

Please note: this graph works for metrics from Prometheus 2.0, *not* 1.0.0 as the Dependencies state.

Get started faster with a Grafana Cloud trial: <https://grafana.com/products/cloud/>

Revisions

Revision	Description	Created	
2	Overview of metrics from Prometheus 2.0. Useful for using prometheus to monitor your prometheus. Revisions welcome!	2017-11-24T14:27:03	Download
1		2017-11-09T14:26:13	Download

[Upload revision](#)



Metrics Endpoint (Prometheus)

[Grafana Labs solution](#)

Easily monitor any Prometheus-compatible and publicly accessible metrics URL with Grafana Cloud's out-of-the-box monitoring solution.

[Learn more](#)



Get this dashboard

Import the dashboard template

[Copy ID to clipboard](#)

or

[Download JSON](#)

ID
3662

Datasource
Prometheus

Dependencies
grafana 4.5.0-beta1 Graph (old)

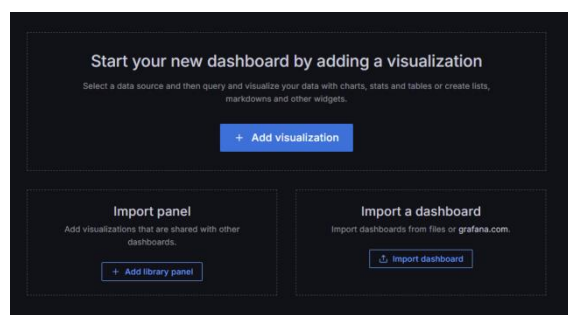
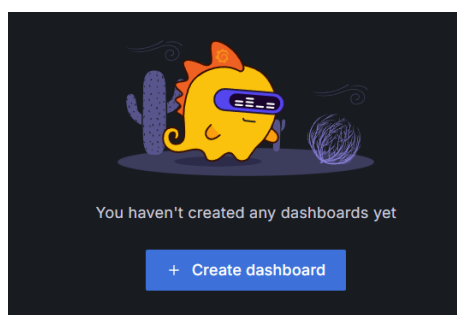
Singlestat Table

Published by
jeremy b

Last update
2020-12-23T22:47:42

[Edit](#) →

В интерфейсе Grafana откройте Dashboards -> Create Dashboard -> Import a dashboard и введите идентификатор дашборда и нажмите кнопку Load:



Import dashboard
Import dashboard from file or Grafana.com

Upload dashboard JSON file
Drag and drop here or click to browse
Accepted file types: json, .txt

Find and import dashboards for common applications at grafana.com/dashboards

Grafana.com dashboard URL or ID Load

Import via dashboard JSON model

```
{
  "title": "Example - Repeating Dictionary variables",
  "uid": "._0HnEoN4z",
  "panels": [...],
  ...
}
```

Load Cancel

После чего дайте имя дашборду и выберите из списка наш добавленный в Grafana сервер Prometheus, после чего нажмите кнопку **import**

Import dashboard
Import dashboard from file or Grafana.com

Importing dashboard from [Grafana.com](https://grafana.com)

Published by: jeremy b
Updated on: 2020-12-24 07:47:42

Options

Name: Prometheus 2.0 Overview

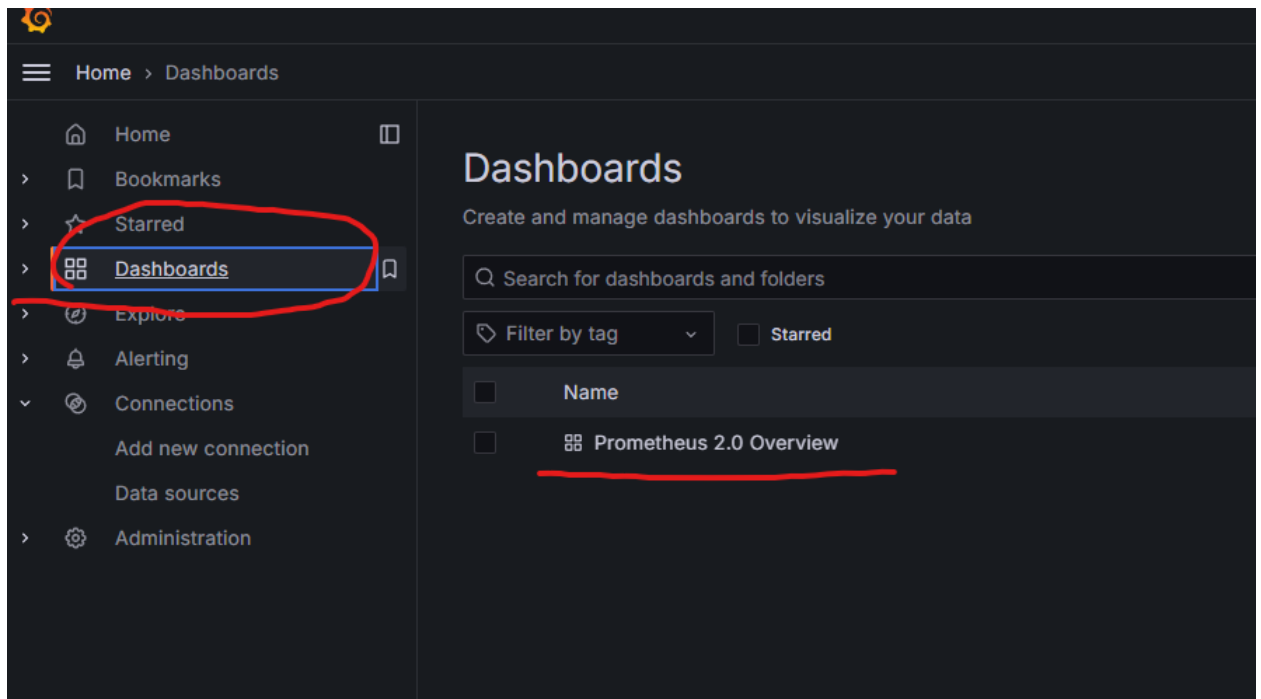
Folder: Dashboards

Unique identifier (UID)
The unique identifier (UID) of a dashboard can be used for uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.
Change uid

prometheus
Прометейка

Import Cancel

После того, как мы добавили дашборд сервера Prometheus, снова кликните в левой части интерфейса Dashboard и выберите по имени наш дашборд:



Где мы можем видеть различные показатели:



Мы протестировали работоспособность нашей системы и теперь приступим к сбору данных с других серверов.

Данные в Prometheus могут поступать из так называемых экспортёров, программ, которые делают различные метрики доступными по HTTP адресу `/metrics`. Сервер Prometheus регулярно опрашивает такие экспортёры и собирает с них данные. Экспортёры существуют для различных служб и сервисов, например, для Nginx, MySQL, PHP-FPM, Apache и естественно экспортёр общих сведений о сервере.

Теперь приступим к установке экспортеров нод (серверов в нашем кластере):

Выполняем установку Node Exporter на все наши сервера: два сервера с СУБД MySQL и веб-сервер 2. Мы ставили мониторинг через `sudo apt install Prometheus`, поэтому установка затянула за собой пакет и `node exporter`. Если мы бы добавляли в ручном режиме ключ репозитория или ставили приложение из исходников, то необходимо было поставить и на веб-сервер 1.

```
sudo apt install prometheus-node-exporter -y
```

Далее запускаем как службу и в автостарт:

```
sudo systemctl start prometheus-node-exporter
sudo systemctl enable prometheus-node-exporter
```

Теперь нам необходимо прописать адреса всех наших нод (ВМ), где установили `node exporter` на сервере Prometheus. Открываем конфиг файл Prometheus на сервере, где установлен Prometheus (это веб-сервер 1)

```
sudo nano /etc/prometheus/prometheus.yml
```

В самом конце где начинается строка **scrape configs** дописываем наши новые джобы (задачи):

```
# 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'

    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 5s
    scrape_timeout: 5s

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

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

  - job_name: node
    # If prometheus-node-exporter is installed, grab stats about the local
    # machine by default.
    static_configs:
      - targets: ['localhost:9100']
```

Разберем последнюю джобу:

```
- job_name: node
  # If prometheus-node-exporter is installed, grab stats about the local
  # machine by default.
  static_configs:
    - targets: ['localhost:9100']
```

Имя джобы `node`, это наша нода, на которой установлен Prometheus, другим джабам мы будем давать имена допустим `mysql1`, `mysql2`, `web2`, а строка `targets` будет содержать вместо `localhost`, ип-адрес LAN сети наших серверов. Также можно добавить интервал опроса в 5 секунд:

```
scrape_interval: 5s
```

Допустим джоба MySQL сервера 1, чтобы добавить наш установленный на нем Node exporter:

```
- job_name: mysql1
  # If prometheus-node-exporter is installed, grab stats about the local
```



```
# machine by default
scrape_interval: 5s
static_configs:
  - targets: ['192.168.0.2:9100']
```

Добавили один веб-сервер2 для примера (далее по инструкции он будет называться сервер mysql)

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

- job_name: node
  # If prometheus-node-exporter is installed, grab stats about the local
  # machine by default.
  static_configs:
    - targets: ['localhost:9100']

- job_name: web2
  scrape_interval: 5s
  static_configs:
    - targets: ['192.168.56.101:9100']
```

ТАК КАК ЭТО ДЕКЛАРАТИВНЫЙ ФАЙЛ ОБРАТИТЕ ВНИМАНИЕ НА ОТСТУПЫ!

После внесения изменений в конфиг файл, перезапустите службу

```
sudo systemctl restart prometheus
```

Не забудьте проверить статус Prometheus !

Теперь по адресу Prometheus нашего первого веб-сервера можно увидеть следующее:

<http://ип-адрес:9090/classic/targets>

Сначала статус точки входа Down

Prometheus Alerts Graph Status ▾ Help		
Targets		
<div> <div>All</div> <div>Unhealthy</div> <div>Collapse All</div> </div>		
node (1/1 up) show less		
Endpoint	State	Labels
http://localhost:9100/metrics	UP	instance="localhost:9100" job="node"
prometheus (1/1 up) show less		
Endpoint	State	Labels
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"
web2 (0/1 up) show less		
Endpoint	State	Labels
http://192.168.56.101:9100/metrics	DOWN	instance="192.168.56.101:9100" job="web2"

Спустя некоторое время начались передаваться метрики:

Prometheus Alerts Graph Status Help

Targets

All Unhealthy Collapse All

node (1/1 up) show less

Endpoint	State	Labels
http://localhost:9100/metrics	UP	instance="localhost:9100" job="node"

prometheus (1/1 up) show less

Endpoint	State	Labels
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"

web2 (1/1 up) show less

Endpoint	State	Labels
http://192.168.56.101:9100/metrics	UP	instance="192.168.56.101:9100" job="web2"

Работоспособность node exporter на серверах можно проверить зайдя по адресу:

ип-адрес –сервера:9100

И увидим метрики:

```
# HELP apt_autoremove_pending Apt package pending autoremove.
# TYPE apt_autoremove_pending gauge
apt_autoremove_pending 0
# HELP apt_upgrades_pending Apt package pending updates by origin.
# TYPE apt_upgrades_pending gauge
apt_upgrades_pending{arch="all",origin="Ubuntu:22.04/jammy"} 1
apt_upgrades_pending{arch="amd64",origin="Ubuntu:22.04/jammy-updates"} 3
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0.006574255
go_gc_duration_seconds{quantile="1"} 0.736632025
go_gc_duration_seconds_sum 2.435691704
go_gc_duration_seconds_count 143
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 8
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.18.1"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 2.523928e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 5.16423832e+08
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.541765e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 1.3038547e+07
# HELP go_memstats_gc_cpu_fraction The fraction of this program's available CPU time used by the GC since the program started.
# TYPE go_memstats_gc_cpu_fraction gauge
go_memstats_gc_cpu_fraction 0.00925153165205992
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 6.048004e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 2.523928e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 6.1513728e+07
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
```

Теперь вернемся в Grafana и добавим дашборд для Node exporter.

Заходим на сайт <https://grafana.com/grafana/dashboards/>

Или пользуемся поиском (чекбокс на выдачу инфы дашбордов) и выбираем нужный нам дашборд. В данном случае мы искали prometheus node exporter и выбираем желаемый дашборд, который действительно заработает:

Dashboard

Grafana-Prometheus-Node_Exporter_Host_Metrics_Dashboard

This is a simple Grafana Dashboard to display Host Metrics such CPU, Memory, DISK IO, Load Average, collected using the Node Exporter

Dashboard

Grafana-Prometheus-Node_Exporter_Host_Metrics_Dashboard

This is a simple Grafana Dashboard to display Host Metrics such CPU, Memory, DISK IO, Load Average, collected using the Node Exporter

Dashboard

Host Stats - Prometheus Node Exporter 0.16.0

Basic host stats: CPU, Memory Usage, Disk Utilisation, Filesystem usage and Predicted time to filesystems filling. This for the Node Exporter version 0.16.0 or later.

Dashboard

Host Stats - Prometheus Node Exporter - pre 0.16.0

Basic host stats: CPU, Memory Usage, Disk Utilisation, Filesystem usage and Predicted time to filesystems filling

Dashboard

Node Exporter Full

Nearly all default values exported by Prometheus node exporter graphed. Only requires the default job_name: node, add as many tar...

Dashboard

OpenWRT


Dashboard for OpenWRT routers with installed lua scripts.

С идентификатором 1860

All dashboards

EN

Node Exporter Full




Nearly all default values exported by Prometheus node exporter graphed.

Only requires the default job_name: node, add as many targets as you need in 'etc/prometheus/prometheus.yml'.

```
- job_name: node
  static_configs:
    - targets: ['localhost:9100']
```

Recommended for prometheus-node-exporter the arguments '-collector.systemd -collector.processes' because the graph uses some of their metrics.

Since revision 16, for prometheus-node-exporter v0.18 or newer. Since revision 12, for prometheus-node-exporter v0.16 or newer.



Linux Server

Grafana Labs solution

Monitor Linux with Grafana. Easily monitor your Linux deployment with Grafana Cloud's out-of-the-box monitoring solution.

[Learn more](#)

Get this dashboard

1

Sign up for Grafana Cloud

[Create free account](#)

2

Import the dashboard template

[Copy ID to clipboard](#)

or

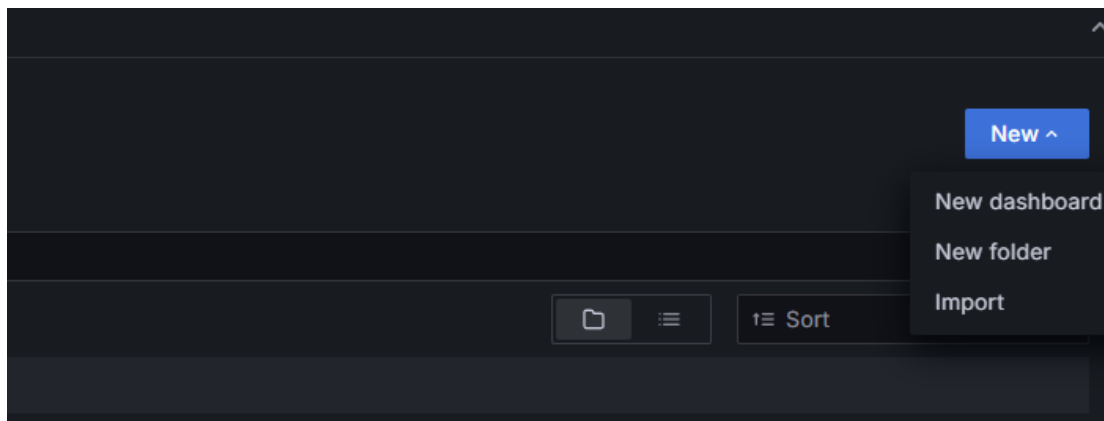
[Download JSON](#)

ID

1860

Datasource

После чего заходим в веб-интерфейс Grafana > Dashboards > **new** > **import** > **load** и добавляем дашборд.



Import dashboard

Import dashboard from file or Grafana.com

Importing dashboard from Grafana.com

Published by

Updated on

Options

Name

Node Exporter серверов

Folder

Dashboards

Unique identifier (UID)

The unique identifier (UID) of a dashboard can be used for uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

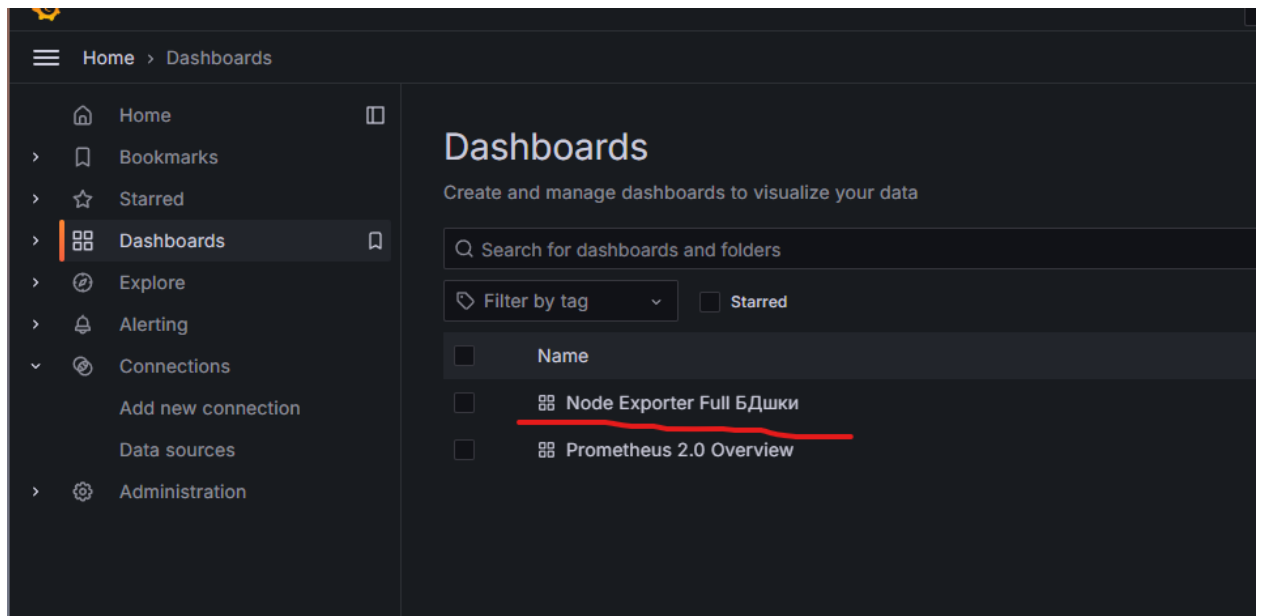
Change uid

Prometheus

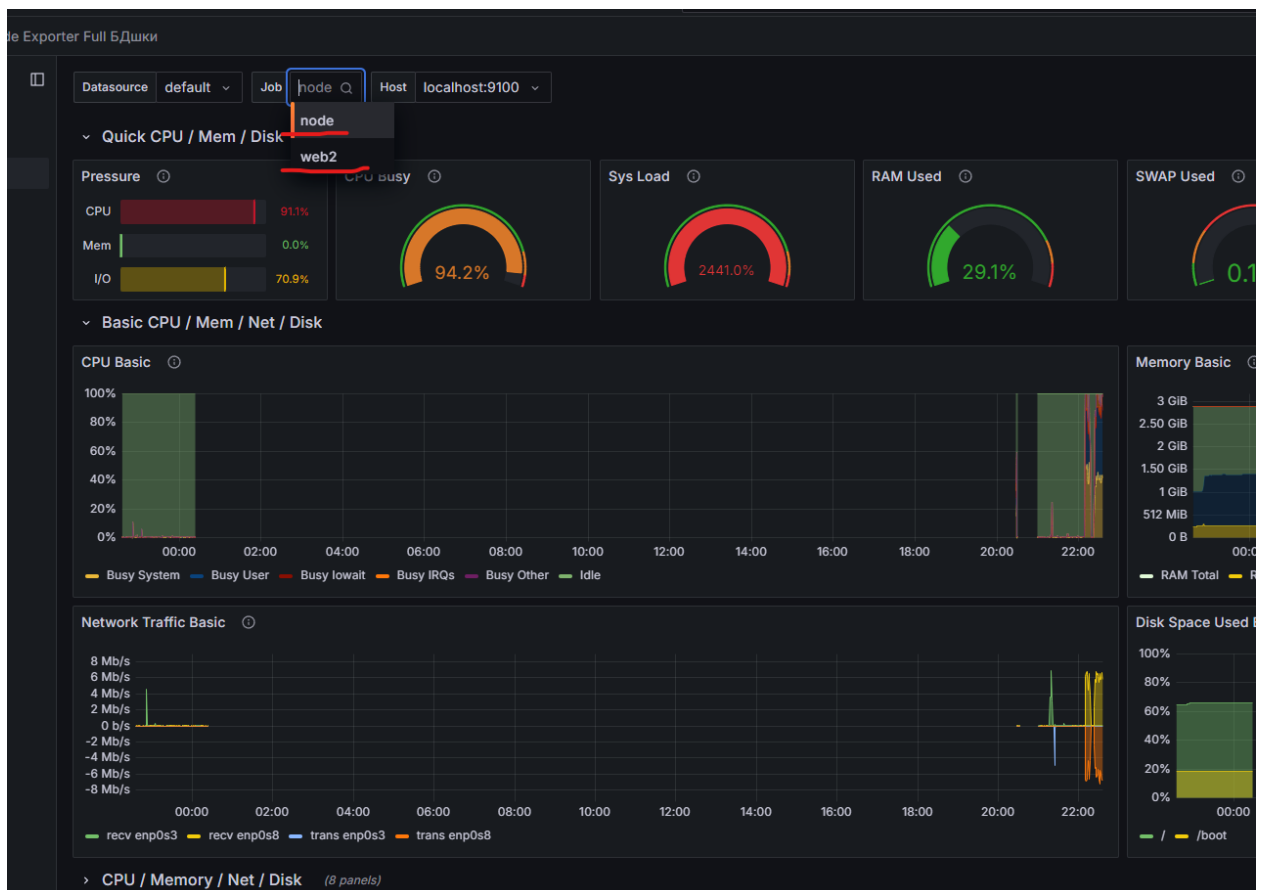
Прометейка

Import Cancel

Переходим в Dashboards и видим новый дашборд (БДшка просто названо):



Теперь мы можем настроить данный дашборд по своему вкусу, чему также целая история и мануал. В джобах мы видим возможность выбора наших нод, которые указали в `/etc/prometheus/prometheus.yml`



ПРОДЕЛАЙТЕ УСТАНОВКУ PROMETHEUS NODE EXPORTER НА ВСЕХ НЕОБХОДИМЫХ СЕРВЕРАХ, ДОБАВЬТЕ В ФАЙЛ КОНФИГУРАЦИИ PROMETHEUS ДЖОБЫ С АДРЕСАМИ ЭТИ СЕРВЕРОВ. РЕСТАРТАНИТЕ ПРОМЕТЕЙ И НАСТРОЙКА МОНИТОРИНГА ЗАКОНЧЕНА!