

[Open in app](#)

★ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



How to Send Email Alerts using Prometheus AlertManager

Get mail alerts from Prometheus about server resource usage.



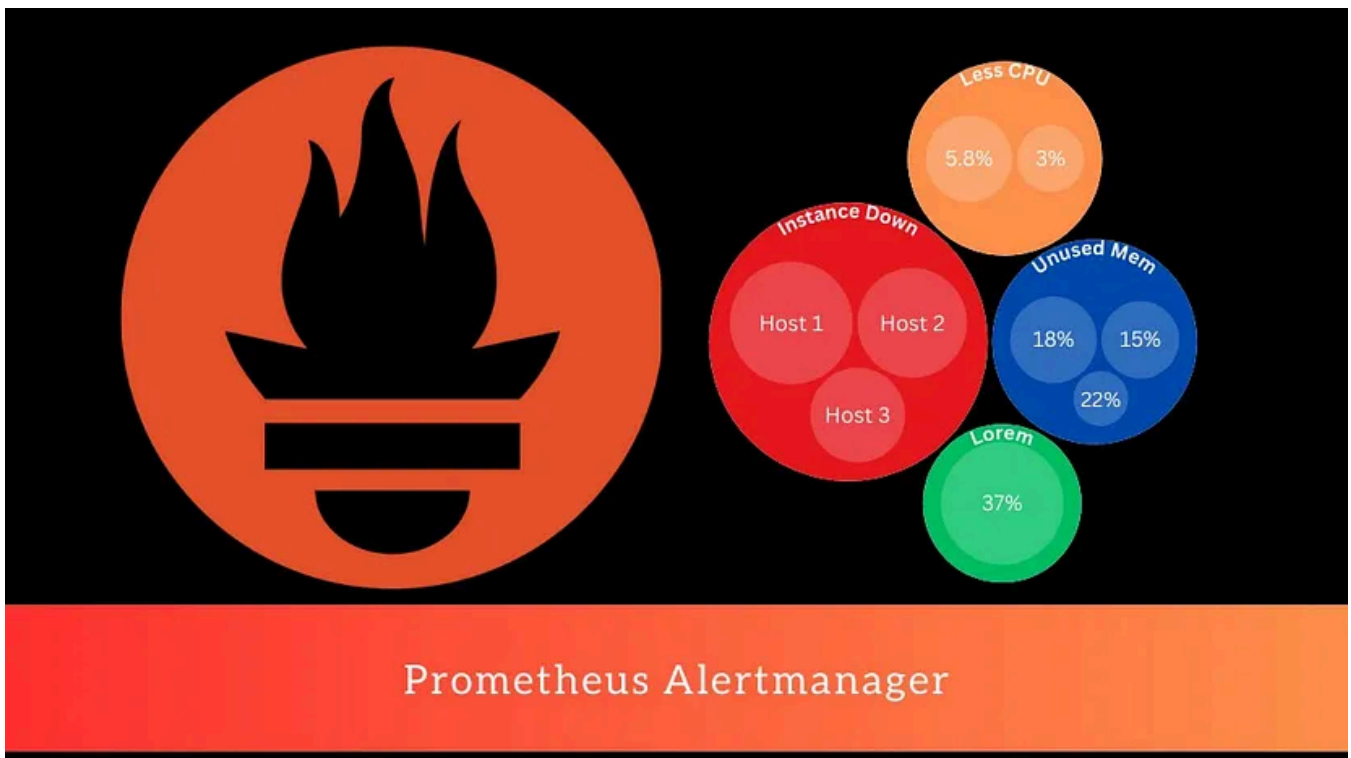
Sagar #OpenToWork · [Follow](#)

Published in DevOps.dev

4 min read · Sep 8, 2023



... More



This is the 2nd article, to setup Prometheus with Node exporter and Grafana refer below article.

You can find the repository on [Github](#).

Setup Monitoring — Prometheus and Grafana.

The Prometheus Alertmanager is a tool for sending alerts via emails, slack and various other mediums.

In this article we'll configure it to send Server Status alerts via email.

Let's get started.

First update the docker-compose.yml to include alertmanager image.

```
version: '3.8'
networks:
  monitoring:
    driver: bridge
volumes:
  prometheus_data: {}
  grafana_data: {}
  alertmanager-data: {}

services:
  node-exporter:
    image: prom/node-exporter:latest
    container_name: node-exporter
    restart: unless-stopped
    volumes:
      - /proc:/host/proc:ro
      - /sys:/host/sys:ro
      - /:/rootfs:ro
    command:
      - '--path.procfs=/host/proc'
      - '--path.rootfs=/rootfs'
      - '--path.sysfs=/host/sys'
      - '--collector.filesystem.mount-points-exclude=^/(sys|proc|dev|host|etc)('
    ports:
      - 9100:9100
    networks:
      - monitoring

  prometheus:
    image: prom/prometheus:latest
    container_name: prometheus
    restart: unless-stopped
```

```
volumes:
  - ./prometheus:/etc/prometheus/
  - prometheus_data:/prometheus
command:
  - '--config.file=/etc/prometheus/prometheus.yml'
  - '--storage.tsdb.path=/prometheus'
  - '--web.console.libraries=/etc/prometheus/console_libraries'
  - '--web.console.templates=/etc/prometheus/consoles'
  - '--web.enable-lifecycle'
ports:
  - 9090:9090
networks:
  - monitoring

alertmanager:
  image: prom/alertmanager:latest
  restart: unless-stopped
  container_name: alertmgr
  ports:
    - "9093:9093"
  volumes:
    - "./alertmanager:/config"
    - alertmanager-data:/data
  command: --config.file=/config/alertmanager.yml --log.level=debug
  networks:
    - monitoring

grafana:
  image: grafana/grafana:latest
  container_name: grafana
  user: '0'
  ports:
    - 3000:3000
  restart: unless-stopped
  volumes:
    - ./grafana/provisioning:/etc/grafana/provisioning
    - /grafana_data:/var/lib/grafana

  networks:
    - monitoring
```

Create an app password in your Google account. Follow the official guide by Google.

Sign in with app passwords

support.google.com

Now create alert manager configuration using below:

```
# alertmanager.yml

route:
  receiver: 'Mail Alert'
  # group_by: [ alertname ]
  repeat_interval: 30s
  group_wait: 15s
  group_interval: 15s

receivers:
- name: 'Mail Alert'
  email_configs:
  - smarthost: 'smtp.gmail.com:587'
    auth_username: '<your email id here>'
    auth_password: "<your app password here>"
    from: '<your email id here>'
    to: '<receiver's email id here>'
    headers:
      subject: 'Prometheus Mail Alerts'
```

Now, will create rules based on which we'll get alerts.

```
#alerts.yml

groups:
- name: Example
  rules:

  - alert: InstanceDown
    expr: up == 0
    for: 15s
    labels:
      severity: critical
    annotations:
      summary: "Instance [{{ $labels.instance }}] down"
      description: "[{{ $labels.instance }}] of job {{ $labels.job }} has been down"

# - name: NodeMemUsage
# rules:
# - alert: HostMemoryIsUnderutilized
#   expr: ({{__name__}}="node_memory_MemAvailable_bytes", instance="node-exporter")
#   for: 15s
#   labels:
#     severity: critical
```

```
annotations:
  summary: Host Memory is underutilized (instance {{ $labels.instance }})
  description: "Node memory is < 20% for 1 week. Consider reducing memory
```

We need to modify Prometheus configuration to include alertmanager and the rules configuration. I have added a node which does not exist for this demo. This is for getting the InstanceDown alert.

```
#prometheus.yml

global:
  scrape_interval: 30s

rule_files:
  - alert.yml

alerting:
  alertmanagers:
    - scheme: http
      static_configs:
        - targets: [ 'alertmanager:9093' ]

scrape_configs:
  - job_name: 'prometheus'
    scrape_interval: 1m
    static_configs:
      - targets: ['localhost:9090']

  - job_name: 'node'
    static_configs:
      - targets: ['node-exporter:9100']

  - job_name: 'dummy_node'
    static_configs:
      - targets: ['localhost:1234']
```

So the directory structure looks like this:

```
[root@labputer Monitor_with_Prometheus]# tree
.
├── alertmanager
└── alertmanager.yml
```

```
├── docker-compose.yml
├── grafana
│   ├── provisioning
│   │   ├── dashboards
│   │   │   ├── dashboard.yml
│   │   │   └── node_exporter.json
│   │   └── datasources
│   │       └── datasource.yml
└── prometheus
    ├── alert.yml
    └── prometheus.yml
```

6 directories, 7 files

Now start the containers with “docker compose up -d”. We can verify alertmanager at <http://ip:9093> and will also be able to view the alerts in Prometheus. At the beginning everything seems to be fine. We don’t see any alerts being active. But a few seconds and we shall see alerts firing and we’ll get emails.

The screenshot shows the Prometheus Alertmanager web interface. At the top, a red banner indicates "2 alerts for". Below this, a blue button labeled "View In Alertmanager" is visible. The main content area displays two alerts. The first alert is titled "[2] Firing" and has the following details: Labels (alertname = InstanceDown, instance = localhost:1234, job = dummy_node, severity = critical) and Annotations (description = [localhost:1234] of job dummy_node has been down for more than 15 seconds, summary = Instance [localhost:1234] down, and a Source link). The second alert is titled "HostMemoryIsUnderutilized" and has the following details: Labels (alertname = HostMemoryIsUnderutilized, instance = node-exporter:9100, job = node, severity = critical) and Annotations (description = Node memory is < 20% for 1 week. Consider reducing memory space. (instance node-exporter:9100) VALUE = 2.582114304e+09 LABELS = map[__name__:node_memory_MemAvailable_bytes instance:node-exporter:9100 job:node], summary = Host Memory is underutilized (instance node-exporter:9100), and a Source link).

Here, both of the alerts were combined and sent via a single mail. If you want to receive separate mails based on the alert types, enable the **group_by** in **alertmanager.yml**.

To view the alerts and Prometheus, click on the alert tab. Expand the alerts to view the rule. Check the annotations box to view the target where the rule is being applied.

The screenshot shows the Prometheus Alerting interface. At the top, there are tabs for Alerts, Graph, Status, and Help. Below the tabs, there are filters for Inactive (0), Pending (0), and Firing (2). A search bar is present with the text 'Filter by name or labels'. The main content area shows two active alerts:

- InstanceDown (1 active)**:
 - name: InstanceDown
 - expr: up == 0
 - for: 15s
 - labels: severity: critical
 - annotations:
 - description: {{{ \$labels.instance }}} of job {{{ \$labels.job }}} has been down for more than 15 seconds.
 - summary: Instance {{{ \$labels.instance }}} down
- HostMemoryIsUnderutilized (1 active)**:
 - name: HostMemoryIsUnderutilized
 - expr: (((name == "node_memory_peakavailable_bytes", instance="node-exporter:9100", job="node") > 80))
 - for: 15s
 - labels: severity: critical
 - annotations:
 - description: Node memory is < 20% for 1 week. Consider reducing memory space. (instance {{{ \$labels.instance }}})
 - VALUE = {{{ \$value }}}
 - LABELS = {{{ \$labels }}}
 - summary: Host Memory is underutilized (instance {{{ \$labels.instance }}})

Below the alerts, there are tables showing the state and active since information for each alert.

Labels	State	Active Since	Value
alertname=InstanceDown instance=localhost:1234 job=dummy_node severity=critical	FIRING	2023-09-08T08:38:33.01420324Z	0

Labels	State	Active Since	Value
alertname=HostMemoryIsUnderutilized instance=node-exporter:9100 job=node severity=critical	FIRING	2023-09-08T08:38:33.01420324Z	2680832000

The alerts and rules can also be viewed from alert manager. The status tab shows the status of alert manager as well as the rules we defined.

The screenshot shows the Alertmanager interface. At the top, there are tabs for Alertmanager, Alerts, Silences, Status, Settings, and Help. A 'New Silence' button is visible. Below the tabs, there are filters for Receiver: All, Silenced, and Inhibited. A search bar is present with the text 'Custom matcher, e.g. env=production'. Below the search bar, there is a '+ Expand all groups' button. The main content area shows a list of alerts:

- Mail Alert**: Not grouped, 2 alerts

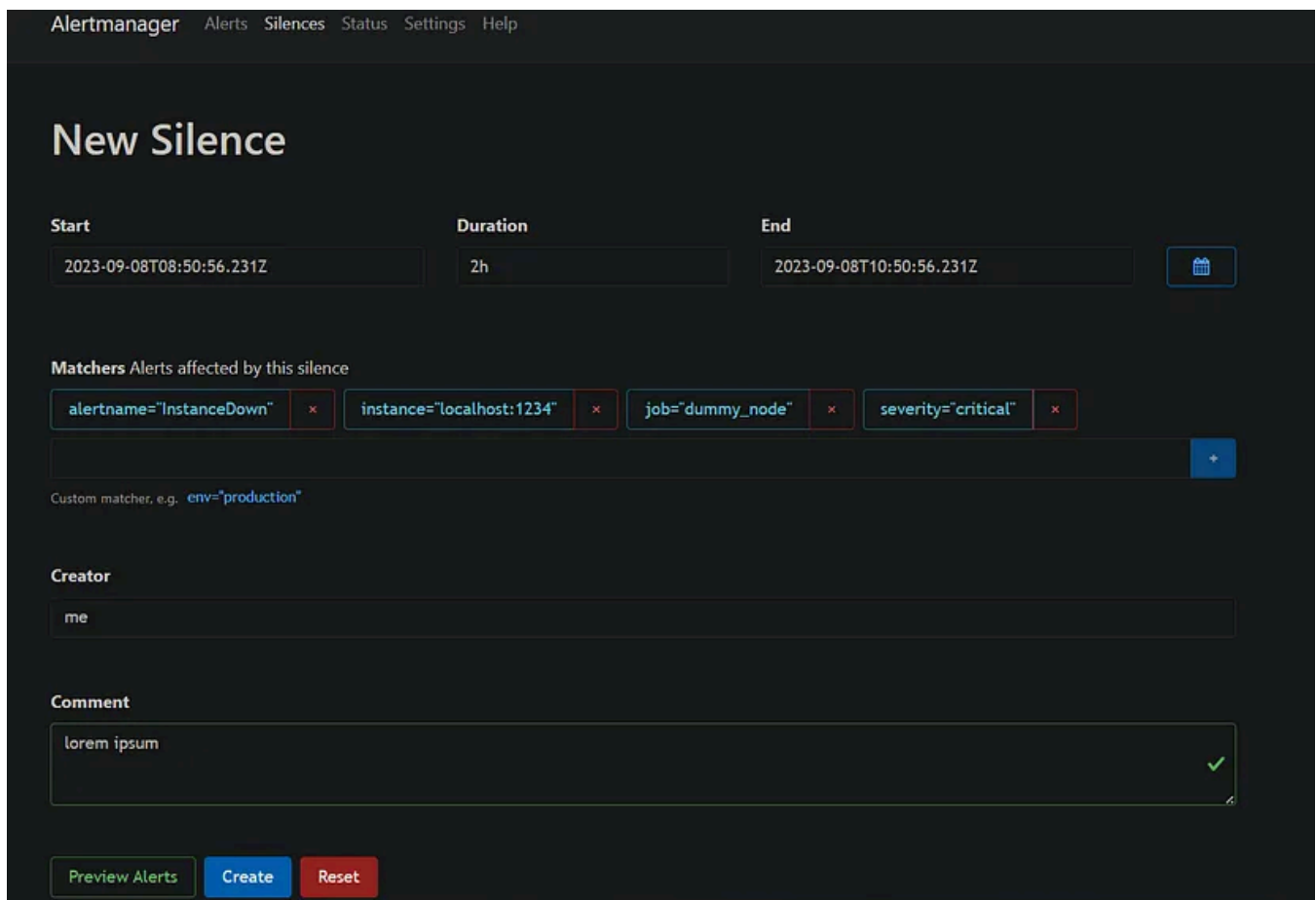
The alerts are listed with their timestamps and actions (Info, Source, Silence, Link). The first alert is:

- 2023-09-08T08:39:33.014Z
- Alert: alertname="InstanceDown" instance="localhost:1234" job="dummy_node" severity="critical"

The second alert is:

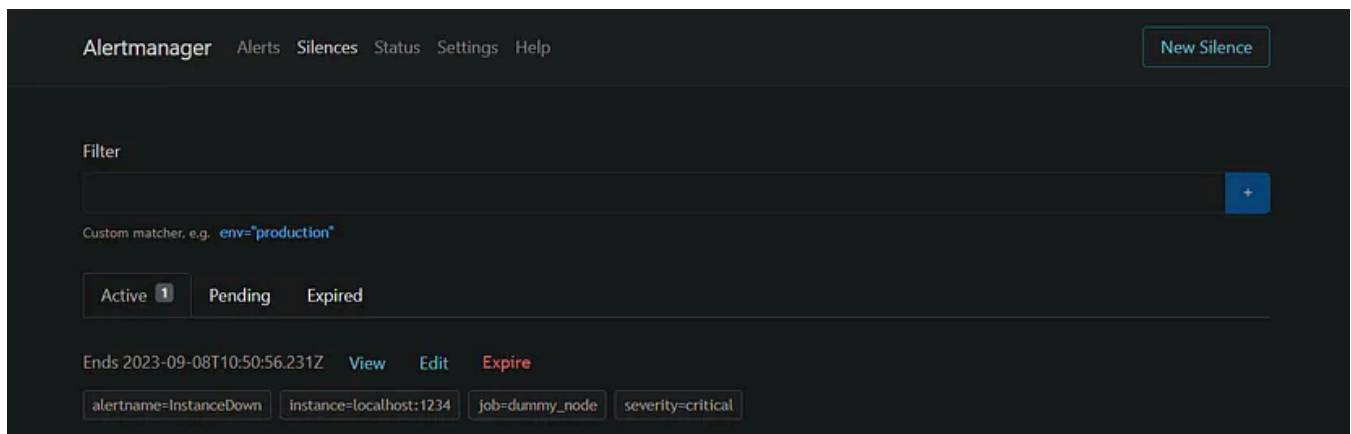
- 2023-09-08T08:39:33.014Z
- Alert: alertname="HostMemoryIsUnderutilized" instance="node-exporter:9100" job="node" severity="critical"

You can silence any alert for a particular period of time. Click on the silence button above the alert. A new window appears to choose duration and etc. Fill all mandatory fields and click create.



The screenshot shows the 'New Silence' form in the Prometheus AlertManager web interface. The form includes fields for 'Start' (2023-09-08T08:50:56.231Z), 'Duration' (2h), and 'End' (2023-09-08T10:50:56.231Z). Below these is a 'Matchers' section with four matchers: 'alertname="InstanceDown"', 'instance="localhost:1234"', 'job="dummy_node"', and 'severity="critical"'. A 'Custom matcher' field is also present with the example 'env="production"'. The 'Creator' field is set to 'me'. The 'Comment' field contains 'lorem ipsum'. At the bottom are buttons for 'Preview Alerts', 'Create', and 'Reset'.

You can make a silenced alert expire the duration forcefully and receive the alerts again.



The screenshot shows the 'New Silence' form in the Prometheus AlertManager web interface. The 'Filter' section is visible, showing a custom matcher 'env="production"'. Below the filter are tabs for 'Active' (1), 'Pending', and 'Expired'. The 'Ends' field shows '2023-09-08T10:50:56.231Z' with buttons for 'View', 'Edit', and 'Expire'. The 'Matchers' section is also visible, showing the same four matchers as in the previous screenshot.

Hope you found this article helpful. In the next article we'll discuss monitoring your website using Blackbox exporter.

Thanks for reading.

Reference: Prometheus.io

Read Next:

Visualize Your Website Metrics — Blackbox Exporter and Grafana

Configure Blackbox Exporter to View Site details in Grafana, get alerts for TLS expiration and other details.

sagarkrp.medium.com

SSL for Everyone: A Guide to configure Let's Encrypt using Certbot

Setup SSL certificate for free. Secure your application with Let's Encrypt.

blog.devops.dev

DevOps

Prometheus

Devops Tool

Devops Practice

Docker



Follow



Written by Sagar #OpenToWork

213 Followers · Writer for DevOps.dev

DevOps Engineer | sagarpanda.com | github.com/sagarkpanda | <https://www.linkedin.com/in/sagarkpanda>. #opentowork

More from Sagar #OpenToWork and DevOps.dev