Grafana

Grafana is a common way to visualize information from multiple source systems, including Prometheus. It offers a user frien

Running Grafana

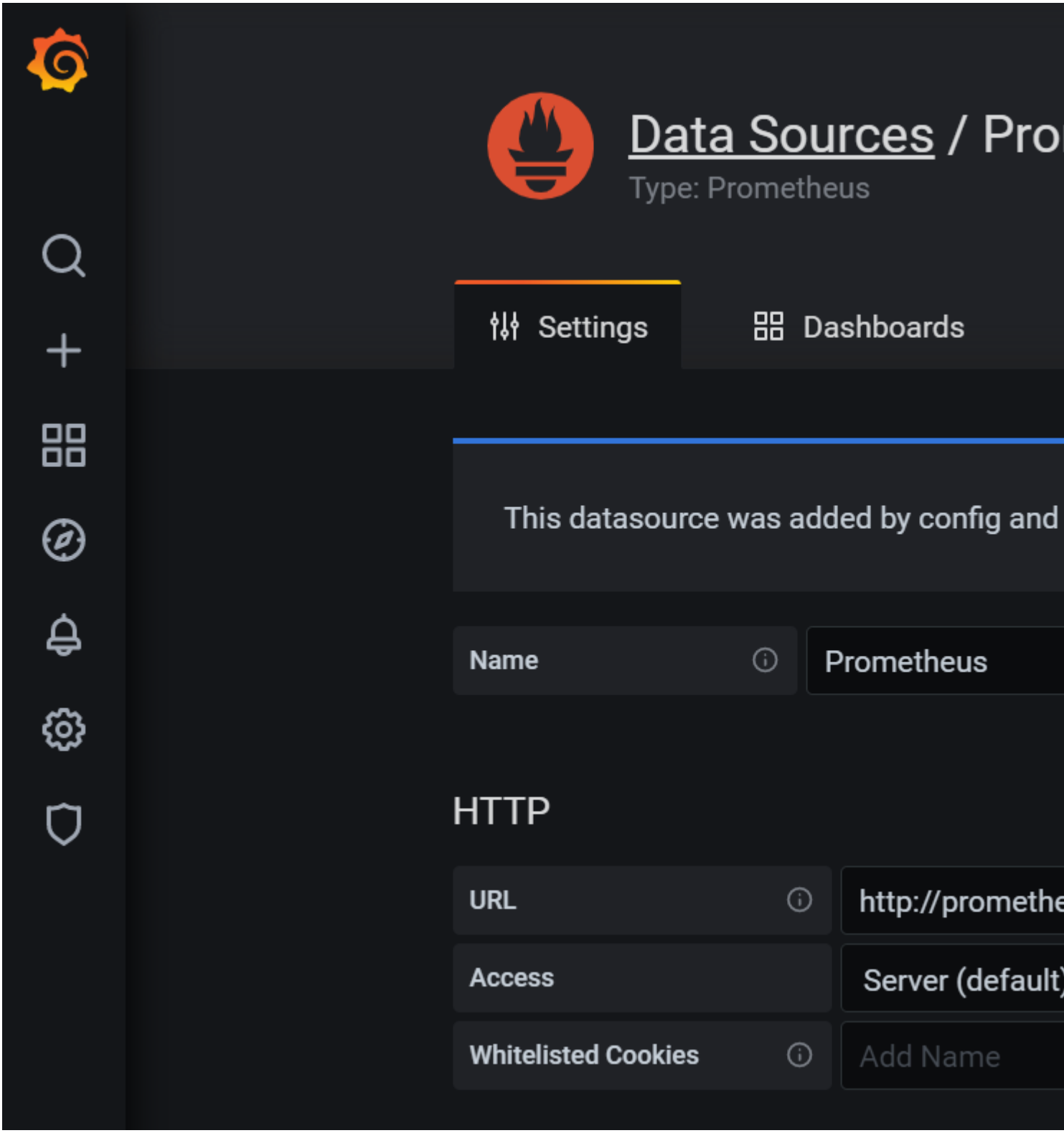
Like Prometheus before, Grafana can be run in a container:

```
docker run -d \
  -p 3000:3000 \
  grafana/grafana
```

This will launch Grafana on port 3000 of your node.

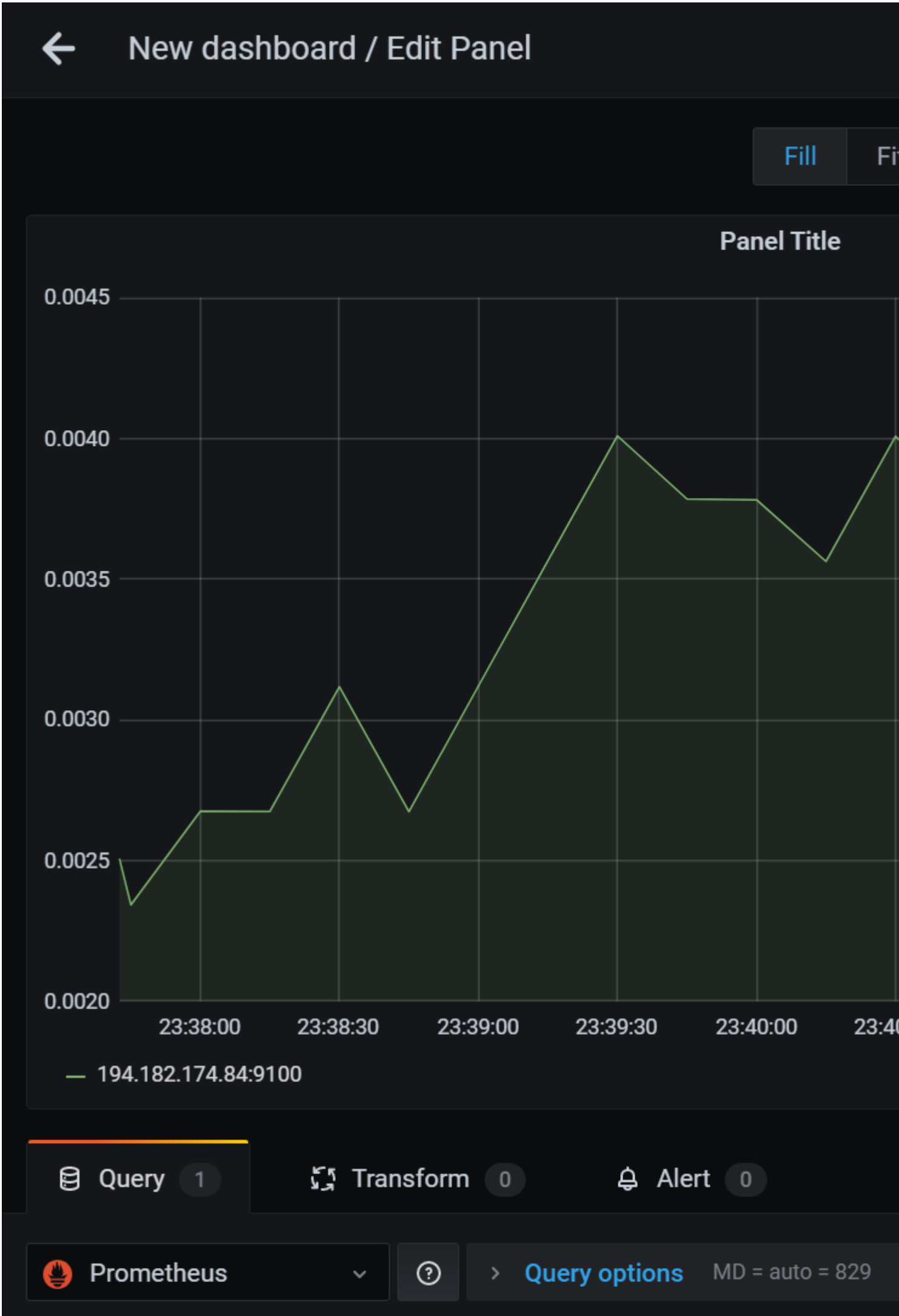
Setting up Grafana with Prometheus

Grafana is an aggregator for information from many different systems. Data sources can be configured in the Configuration
the previous exercise you can leave everything else on default settings.



Creating a CPU graph

Once you have the data source set up you can create a new dashboard and add a panel. The panel can have different visu
In addition, you can also provide sophisticated legend information, such as `` to list the instance name.


New dashboard / Edit Panel
Fill
Panel Title
0.0045
0.0040
0.0035
0.0030
0.0025
0.0020
23:38:00
23:38:30
23:39:00
23:39:30
23:40:00
194.182.174.84:9100
Query 1
Transform 0
Alert 0
Prometheus
Query options
MD = auto = 829

On the right hand side you can adjust various display options, for example what range and display format the Axis' should h

Creating an alert

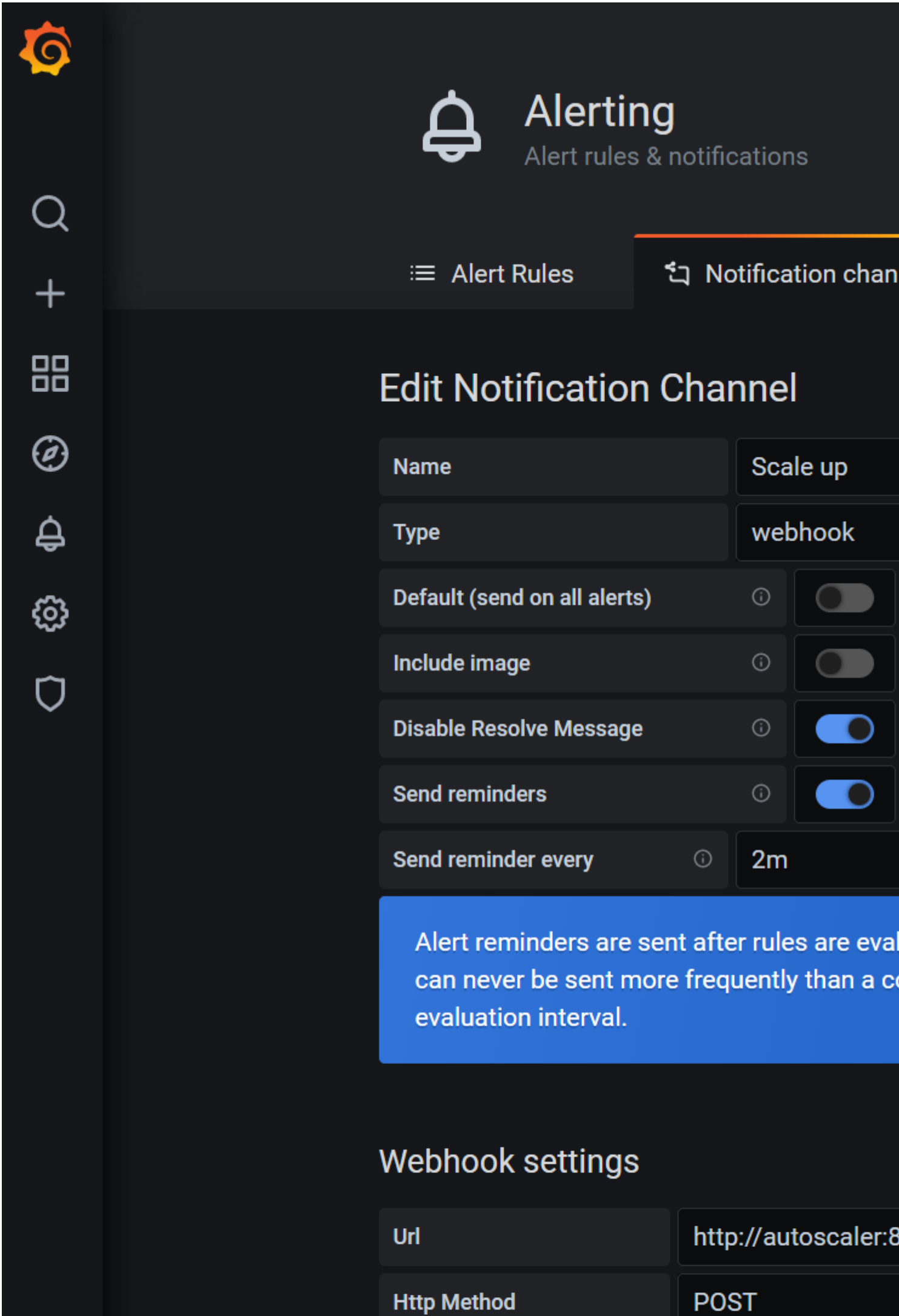
In the same interface as above you can create an alert. The alert will allow you to set an evaluation period to determine how
triggered.


New dashboard / Edit Panel
Fill
Panel Title
1.0
0.8
0.6
0.4
0.2
0
23:38:00
23:38:30
23:39:00
23:39:30
23:40:00
23:40:30
194.182.174.84:9100
Query 1
Transform 0
Alert 0
Rule
Name
CPU usage

You can also set up notifications to send the alert to various notification channels. This functionality can be used to trigger th

Setting up a notification channel

Before you can set up an alert notification you have to create a notification channel. This can be done from the left hand me
Exoscale to increase the instance pool size. Finally, you should also set up the "Send reminders" option to keep triggering th


Alerting
Alert rules & notifications
Alert Rules
Notification chan
Edit Notification Channel
Name
Scale up
Type
webhook
Default (send on all alerts)
Include image
Disable Resolve Message
Send reminders
Send reminder every
2m
Alert reminders are sent after rules are eva
can never be sent more frequently than a c
evaluation interval.
Webhook settings
Url
http://autoscaler:8
Http Method
POST

Deploying Grafana in an automated fashion

Since our project work revolves around Terraform we need a way to deploy Grafana with all settings in an automated fashio

Provisioning data sources

To provision data sources we must place or mount the data source configuration file in the `/etc/grafana/provisioning/da`

```
apiVersion: 1
datasources:
- name: Prometheus
  type: prometheus
  access: proxy
  orgId: 1
  url: http://prometheus:9090
  version: 1
  editable: false
```

Provisioning a notification channel

Notification channels can also be provisioned by placing the appropriate YAML file in `/etc/grafana/provisioning/notifie`

```
notifiers:
  - name: Scale up
    type: webhook
    uid: scale-up
    org_id: 1
    is_default: false
    send_reminder: true
    disable_resolve_message: true
    frequency: "2m"
    settings:
      autoResolve: true
      httpMethod: "POST"
      severity: "critical"
      uploadImage: false
      url: "http://autoscaler:8090/up"
```

Note, that the `uid` field of the notifier matters as this will be referenced from the dashboard.

Provisioning dashboards

Provisioning dashboards is slightly more complex. As a first step we must tell Grafana to look in a certain directory for the da

```
apiVersion: 1

providers:
- name: 'Home'
  orgId: 1
  folder: ''
  type: file
  updateIntervalSeconds: 10
  options:
    path: /etc/grafana/dashboards
```

We can then place the dashboard JSON file in the specified directory. The easiest way to create a JSON file is to manually