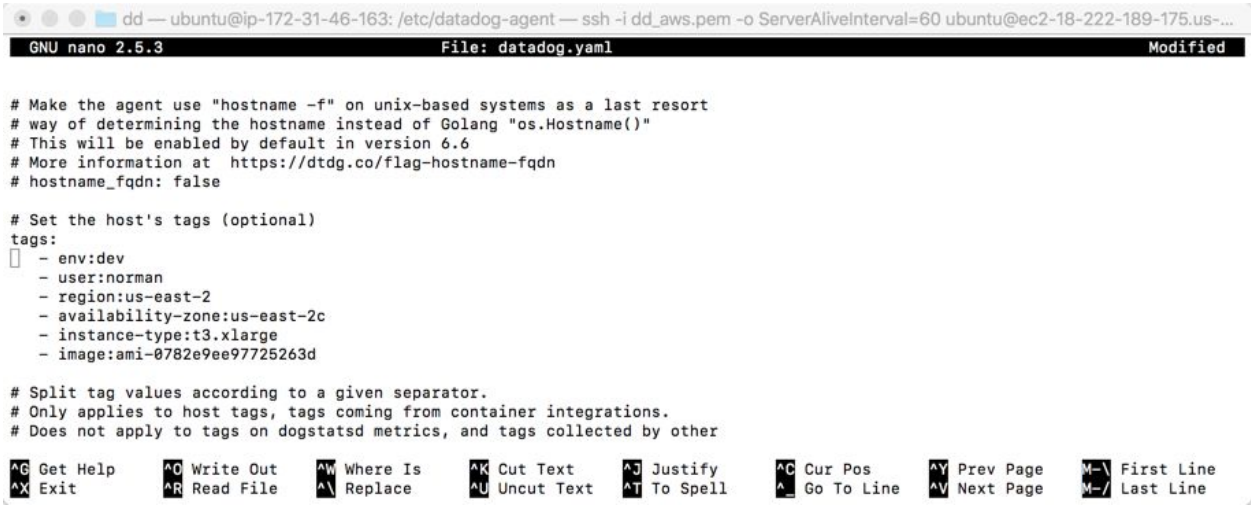
## Collecting Metrics

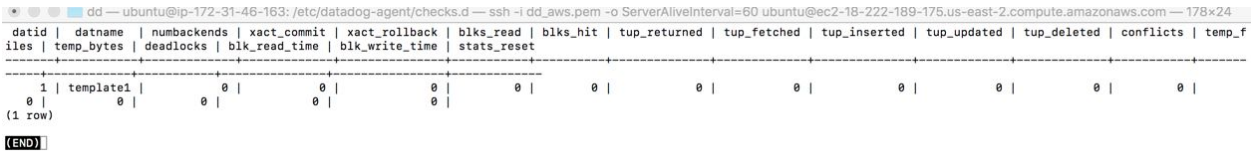
- Add tags in the Agent config file and show us a screenshot of your host and its tags on the Host Map page in Datadog.





- Install a database on your machine (MongoDB, MySQL, or PostgreSQL) and then install the respective Datadog integration for that database.

```
psql -h localhost -U datadog postgres -c \
"select * from pg_stat_database LIMIT(1);" \
&& echo -e "\e[0;32mPostgres connection - OK\e[0m" \
|| echo -e "\e[0;31mCannot connect to Postgres\e[0m"
```



- Create a custom Agent check that submits a metric named my_metric with a random value between 0 and 1000.

File: /etc/datadog-agent/checks.d/assignment.py

```
# the following try/except block will make the custom check compatible with any Agent
version
try:
   # first, try to import the base class from old versions of the Agent...
   from checks import AgentCheck
except ImportError:
   # ...if the above failed, the check is running in Agent version 6 or later
   from datadog_checks.checks import AgentCheck

# content of the special variable __version__ will be shown in the Agent status page
__version__ = "1.0.0"

from random import *

class HelloCheck(AgentCheck):
   def check(self, instance):
      self.gauge('assignment.my_metric', randint(1, 1000))
```

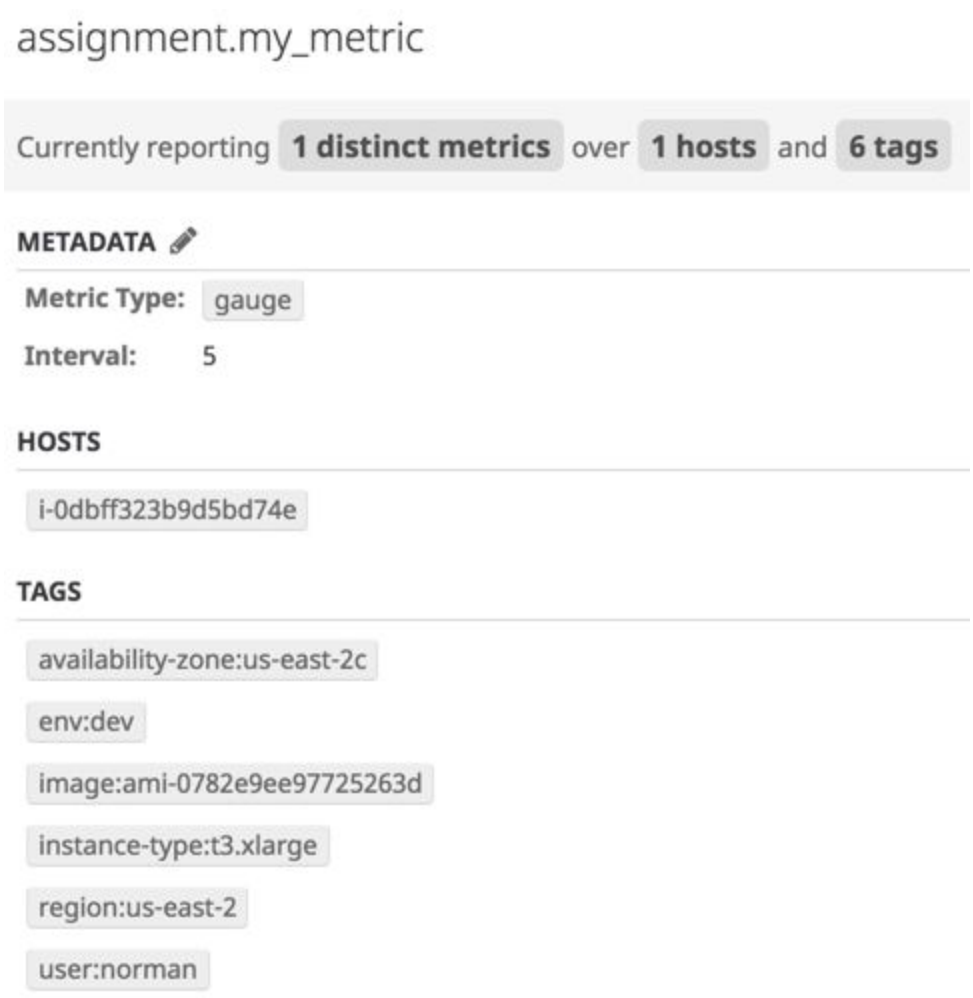
- Change your check's collection interval so that it only submits the metric once every 45 seconds.

File: /etc/datadog-agent/conf.d/assignment.yaml

```
instances:
   - min_collection_interval: 45
```

- Bonus Question Can you change the collection interval without modifying the Python check file you created?

```
curl -X PUT -H "Content-type: application/json" -d '{
     "type": "gauge",
"short_name": "my_metric",
"statsd_interval": "5"}'
"https://api.datadoghq.com/api/v1/metrics/assignment.my_metric?api_key=33aa0a665a5
44113052cf2efba7ac54f&application_key=d31a5814782cd258bda63d18b1db2839ab91f
ac7"
```

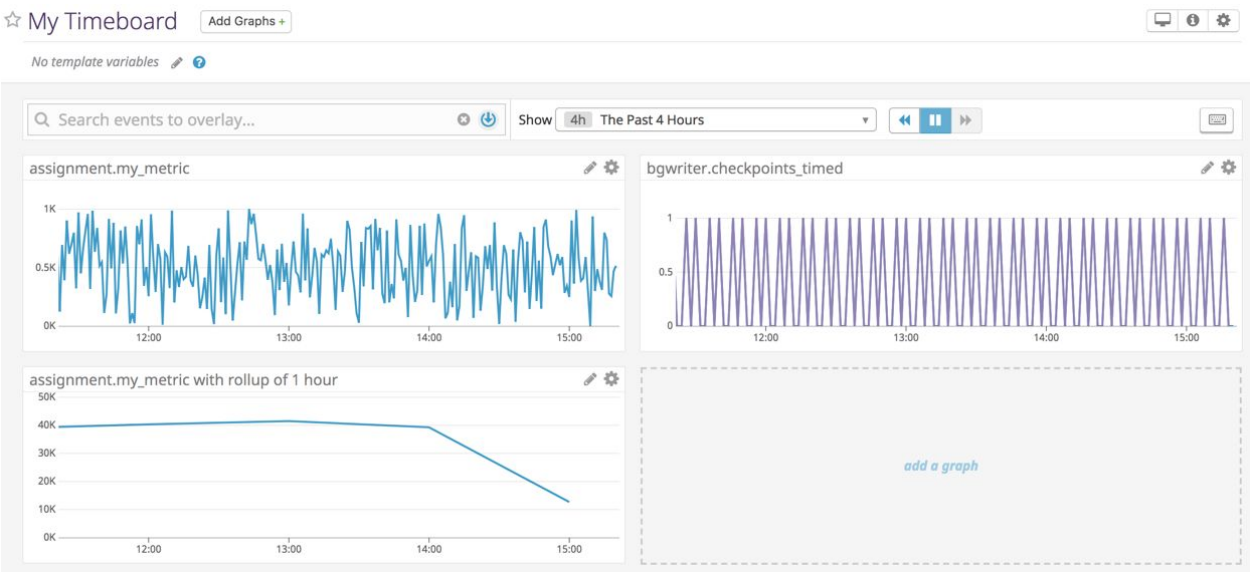



## Visualizing Data

Utilize the Datadog API to create a Timeboard that contains:

- Your custom metric scoped over your host.
- Any metric from the Integration on your Database with the anomaly function applied.
- Your custom metric with the rollup function applied to sum up all the points for the past hour into one bucket

See file create_timeboard.py.



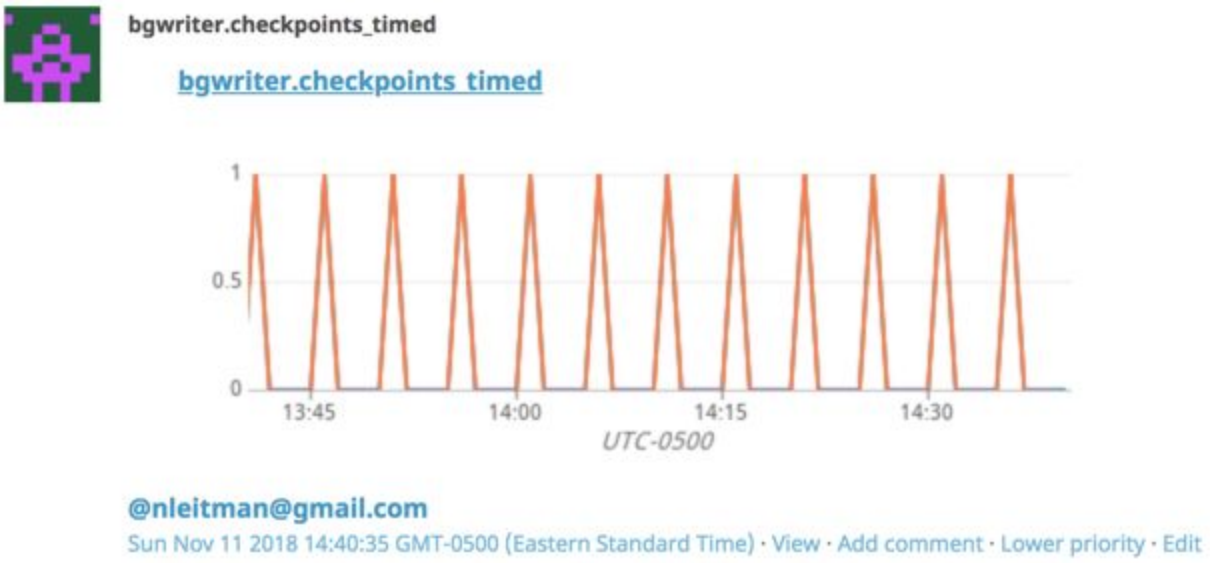

- Set the Timeboard's timeframe to the past 5 minutes

  There’s no option to set the timeframe of the Timeboard to 5 minutes:


  It seems that only Screenboards can have their timeframe set to 5 minutes:

  https://docs.datadoghq.com/graphing/dashboards/screenboard/#global-time-selector

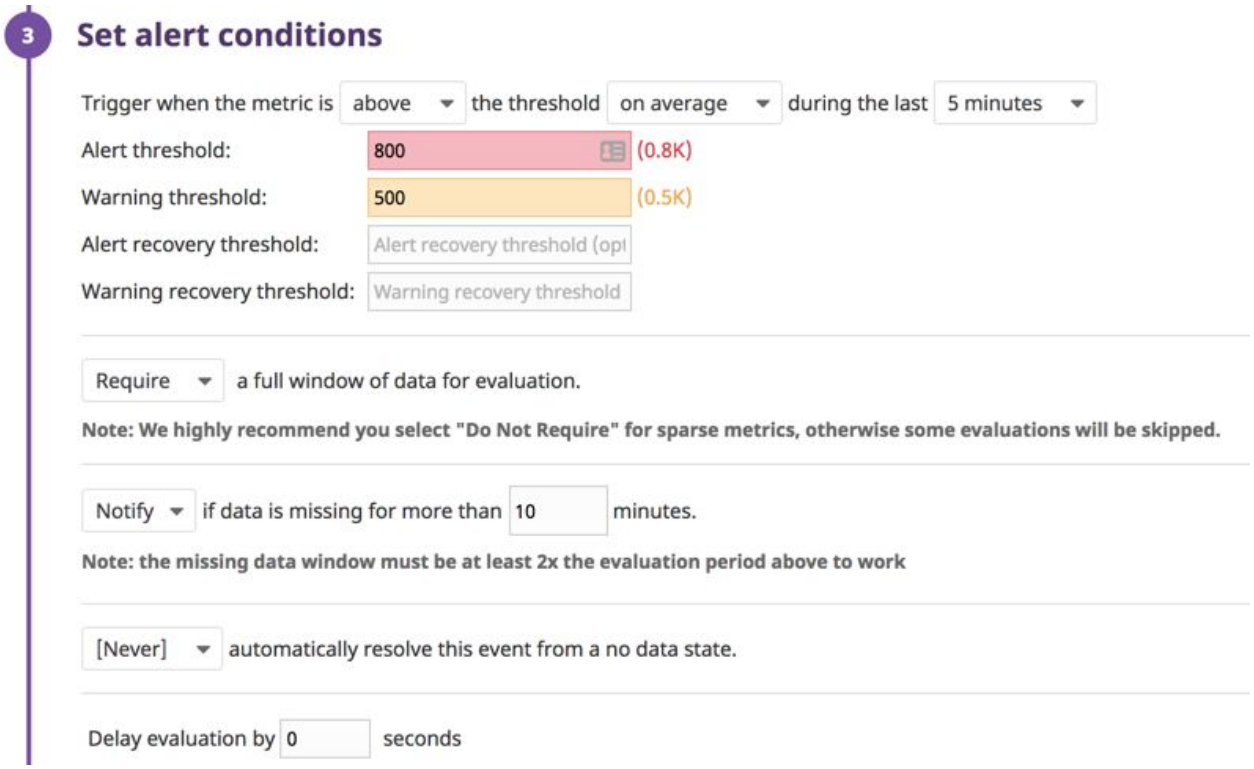
- Take a snapshot of this graph and use the @ notation to send it to yourself.




- Bonus Question: What is the Anomaly graph displaying?

  It's showing that the values of 0 are considered to be within the normal range, and the values of 1 are classified as anomalies. Changing the bounds parameter has no effect on these classifications.

## Monitoring Data




{{#is_alert}}
Alert: value is > 800:
value = {{value}} on host:{{host.ip}}
{{/is_alert}}

{{#is_warning}}
Warning: value is > 500:
{{/is_warning}}

{{#is_no_data}}
Missing data
{{/is_no_data}}

@nleitman@gmail.com

To troubleshoot:
1) SSH into host

2) Search through PostgresSQL table "transactions" and see if there is any unusual data
3) Delete anomalous data and notify Engineering

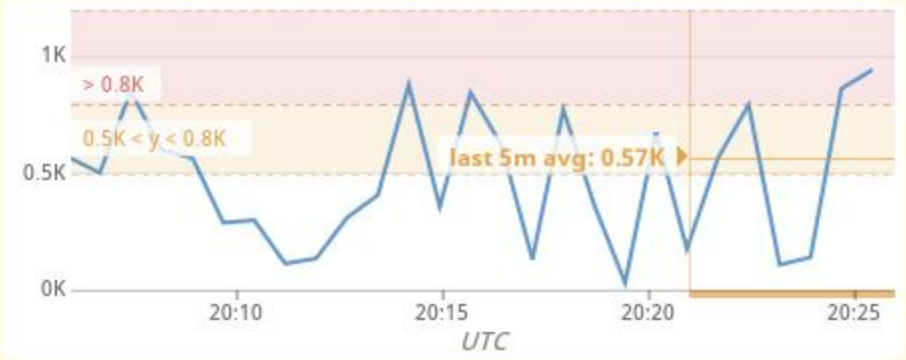
DATADOG

[Warn] assignment.my_metric is too high on i-0dbff323b9d5bd74e

Warning: value is > 500:

@nleitman@gmail.com

To troubleshoot:
1) SSH into host
2) Search through PostgresSQL table "transactions" and see if there is any unusual data
3) Delete anomalous data and notify Engineering




**assignment.my_metric** over **host:i-0dbff323b9d5bd74e** was **> 500.0** on average during the **last 5m**.

The monitor was last triggered at Tue Nov 13 2018 20:26:08 UTC (**4 secs ago**).

---

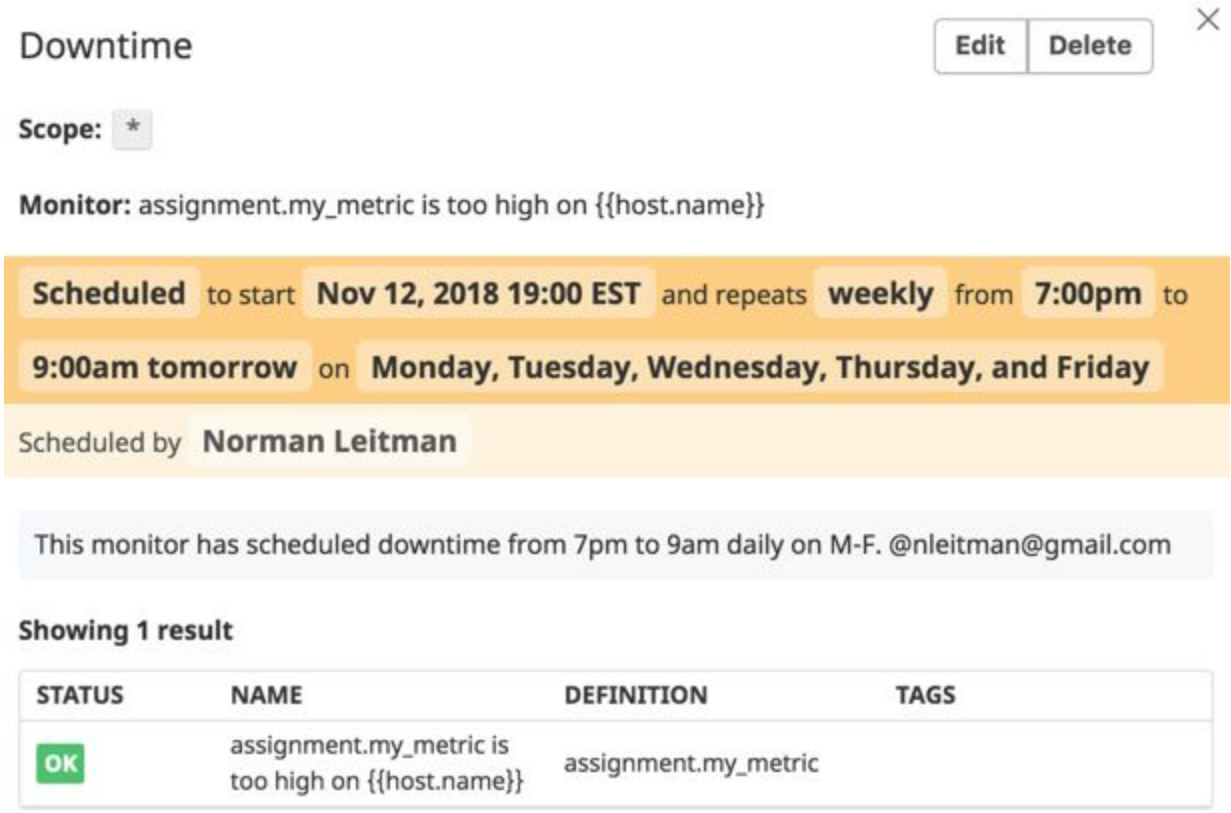
[Monitor Status] · [Edit Monitor] · [Show Processes]

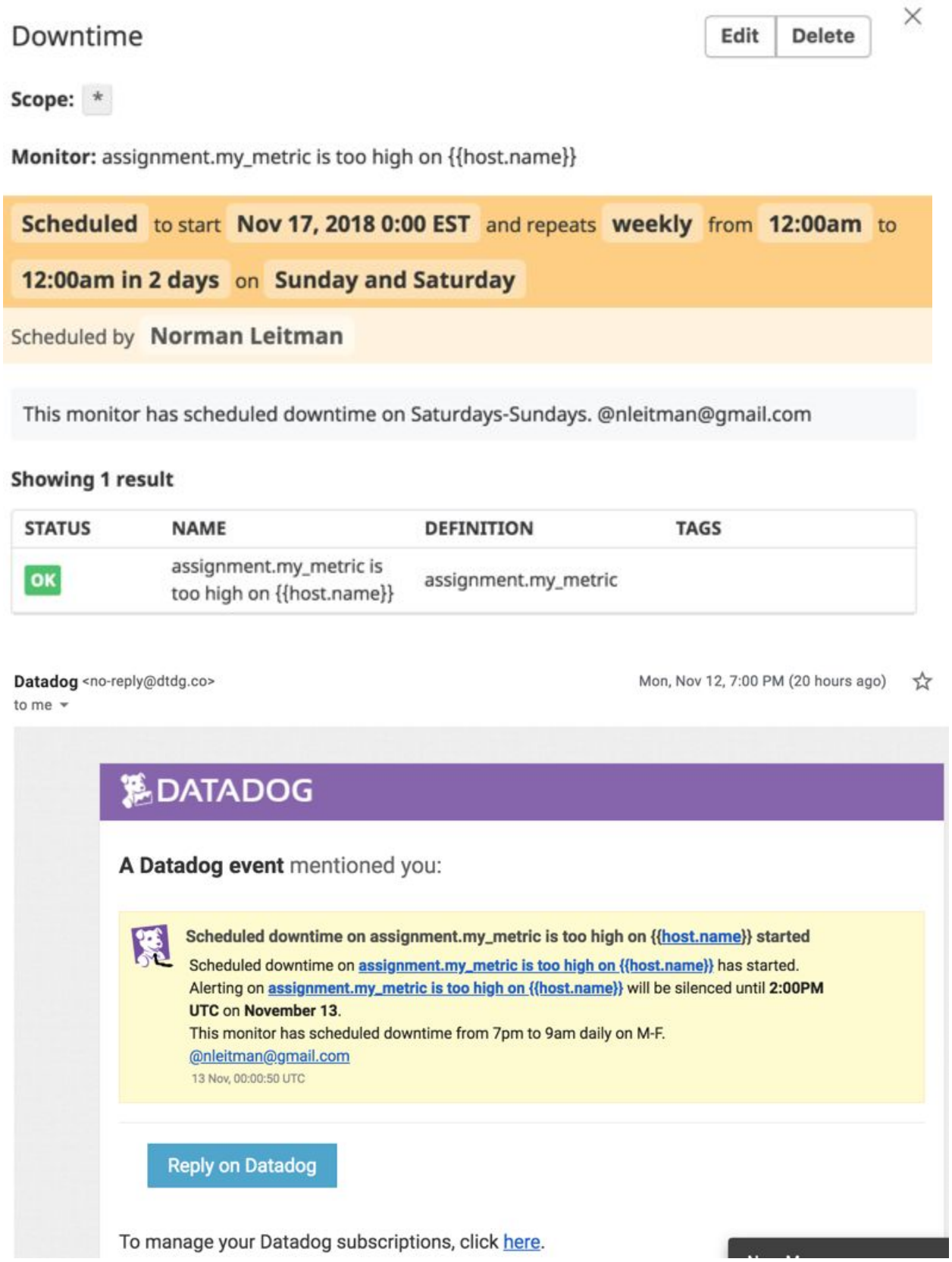
This alert was raised by account Datadog Recruiting Candidate

Comment in Datadog

Bonus Question: Since this monitor is going to alert pretty often, you don't want to be alerted when you are out of the office. Set up two scheduled downtimes for this monitor:

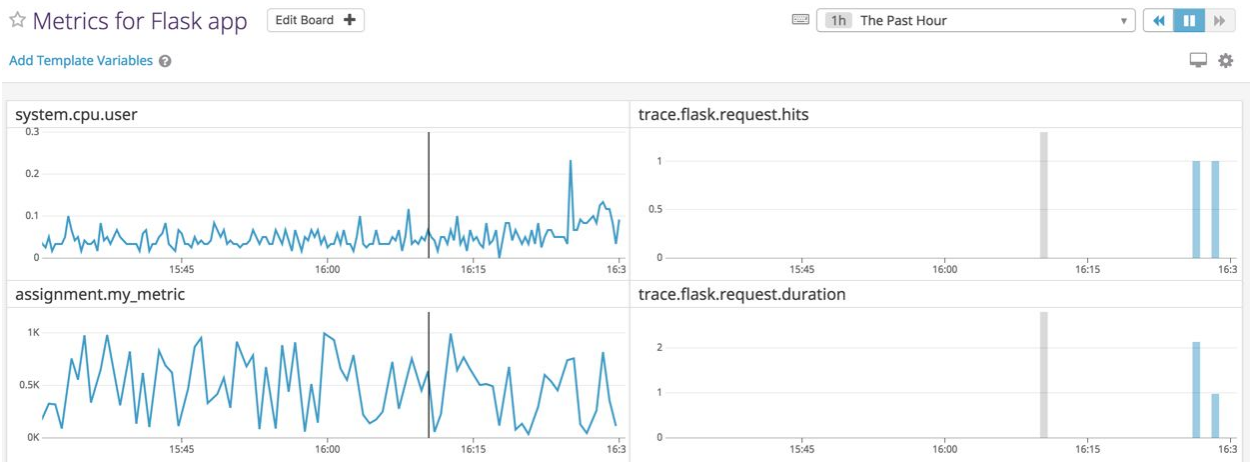
- One that silences it from 7pm to 9am daily on M-F,
- And one that silences it all day on Sat-Sun.
- Make sure that your email is notified when you schedule the downtime and take a screenshot of that notification.

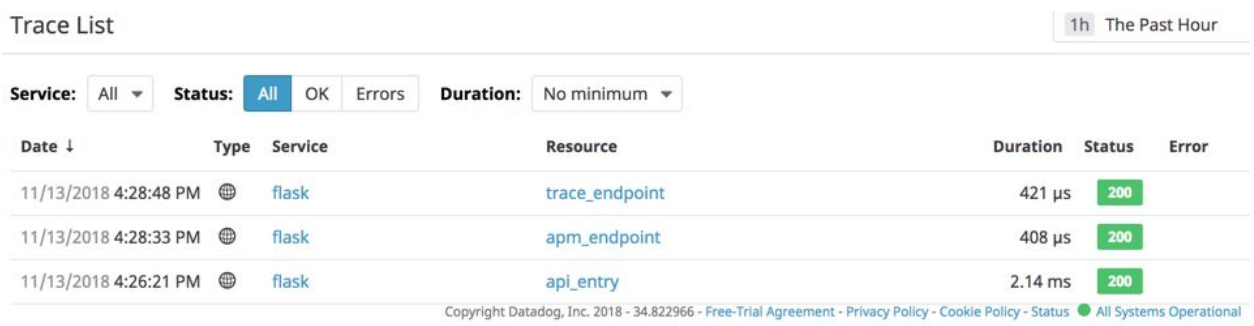



Downtime
Edit
Delete
Scope: *
Monitor: assignment.my_metric is too high on {{host.name}}
Scheduled to start Nov 17, 2018 0:00 EST and repeats weekly from 12:00am to 12:00am in 2 days on Sunday and Saturday
Scheduled by Norman Leitman
This monitor has scheduled downtime on Saturdays-Sundays. @nleitman@gmail.com
Showing 1 result
STATUS
NAME
DEFINITION
TAGS
OK
assignment.my_metric is too high on {{host.name}}
assignment.my_metric
Datadog <no-reply@dtdg.co>
to me
Mon, Nov 12, 7:00 PM (20 hours ago)
DATADOG
A Datadog event mentioned you:
Scheduled downtime on assignment.my_metric is too high on {{host.name}} started
Scheduled downtime on assignment.my_metric is too high on {{host.name}} has started.
Alerting on assignment.my_metric is too high on {{host.name}} will be silenced until 2:00PM UTC on November 13.
This monitor has scheduled downtime from 7pm to 9am daily on M-F.
@nleitman@gmail.com
13 Nov, 00:00:50 UTC
Reply on Datadog
To manage your Datadog subscriptions, click here.

## Collecting APM Data

https://p.datadoghq.com/sb/ec7a3a39b-9c8801d356cf85f71270e43e54055f81




- Bonus Question: What is the difference between a Service and a Resource?

  A service is a set of processes that work together to provide a feature set, such as a web or database service. The service provides methods, or verbs, for operating on resources. For example, RESTful web services include the following methods: GET, PUT, PATCH, POST and DELETE.

  Resources are nouns and include everything that can be accessed by a service. Some examples are URLs such as */user/home* or SQL statements such as *select * from users where id = ?*. For comparison, a resource in REST is similar to an Object in Object Oriented Programming or an Entity in a Database.

## Final Question

Datadog has been used in a lot of creative ways in the past. We've written some blog posts about using Datadog to monitor the NYC Subway System, Pokemon Go, and even office restroom availability!

Is there anything creative you would use Datadog for?

> Similar to NYC subway monitoring, I would use Datadog to monitor traffic conditions: https://511ny.org/
>
> The 511 NY API (https://www.programmableweb.com/api/511ny) provides the following resources: https://511ny.org/developers/help
>
> To set this up, I would first ingest the list of roadways from here:
>
> https://511ny.org/developers/help/api/get-api-getroadways_key_format
>
> From the list, I would choose which ones I wanted to explicitly monitor.
>
> Then I would call this endpoint on a regular basis, perhaps once per minute, to get traffic events:
>
> https://511ny.org/developers/help/api/get-api-getevents_key_format
>
> I would search the response on the field `RoadwayName`. For each match, I would post metrics to Datadog via the API:
>
> https://docs.datadoghq.com/api/?lang=python#metrics
>
> I would add tags for the borough, and neighborhoods in Manhattan.
>
> I would ingest traffic alerts from here:
>
> https://511ny.org/developers/help/api/get-api-getalerts_key_format
>
> And post them to Datadog using this endpoint:
>
> https://docs.datadoghq.com/api/?lang=python#events