

Support Engineer Hiring Challenge

Author: Chris Matteri

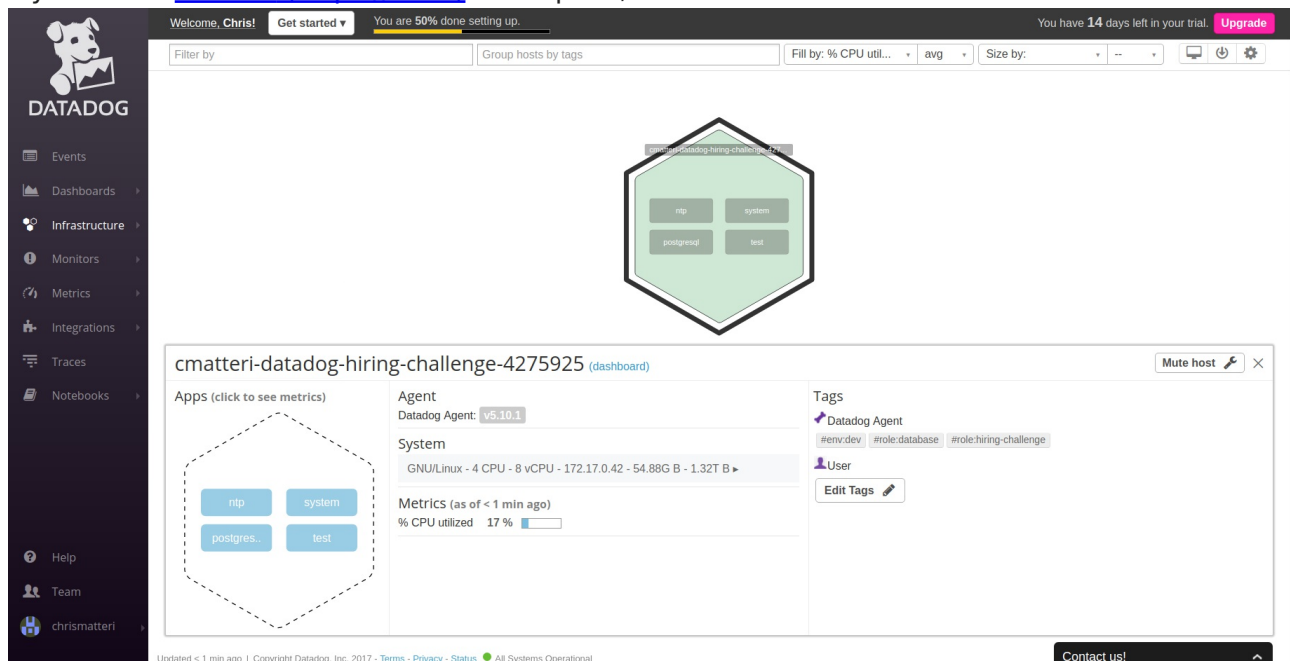
If you aren't using a Markdown previewer, please view [answers.pdf](#)

In your own words, what is the Agent?

The Agent is an application that runs on a host that is being monitored. It collects data from the system, and typically also from various applications and services. The Agent transmits the data, which consists of metrics and events, to Datadog's servers to be archived, processed, and displayed.

Host Map

My host is a [Cloud 9 \(https://c9.io\)](https://c9.io) Workspace, which is a Docker container.



Random Agent check

Creating a custom Agent check required two files, `random.py` and `random.yaml`, which were placed in `/etc/dd-agent/checks.d/` and `/etc/dd-agent/conf.d/`, respectively. A gauge was the most appropriate metric type for sampling a random signal at a fixed sample rate.

```
# random.py
import random
from checks import AgentCheck

class RandomCheck(AgentCheck):
    def check(self, instance):
        self.gauge('test.support.random', random.random())
```

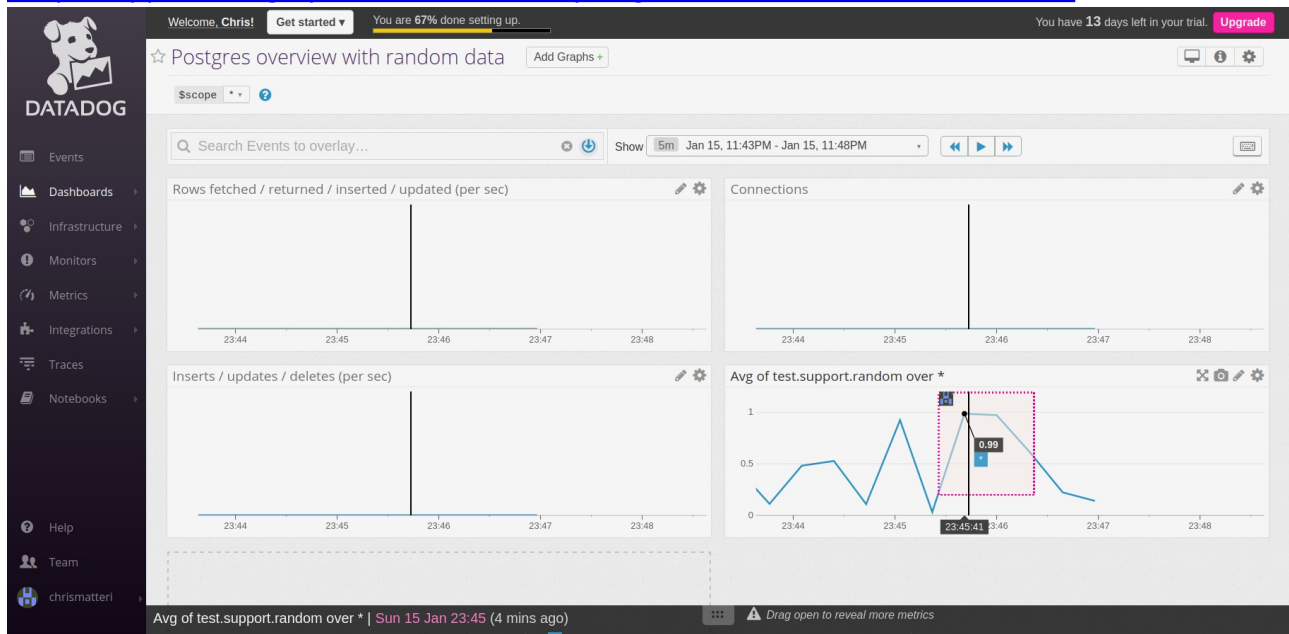
```
# random.yaml
init_config:

instances:
  [{}]
```

Postgres/Random Metric Dashboard

[Postgres dashboard with random data metric](#)


<https://app.datadoghq.com/dash/234086/postgres-overview-with-random-data>




What is the difference between a timeboard and a screenboard?

Timeboards have a rigid grid structure, and all of their graphs show the same interval of time. This makes them useful for viewing the relationship of many metrics in a certain time period. Screenboards are more flexible. Graphs on a screenboard can have different sizes and timespans. Screenboards can also show event streams, images, and even custom HTML. They allow a variety of heterogenous information to be viewed immediately in a single location.

I emailed myself this snapshot with an @ notification:

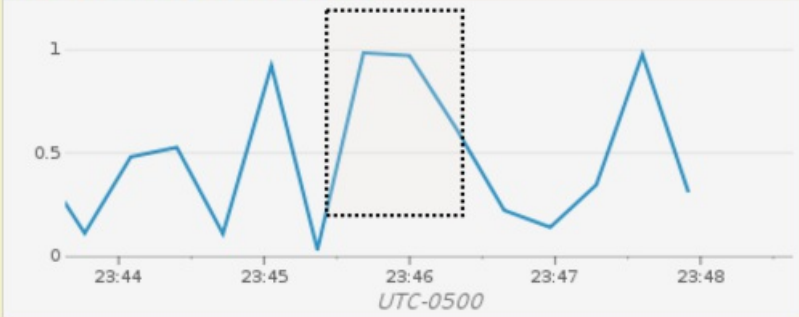


Chris Matteri (@chrismatter@gmail.com) mentioned you in a comment:



Chris Matteri

[Avg of test.support.random over *](#)



@mail@chrismatter.com Random data rises above 0.9!
16 Jan, 04:48:09 UTC

Reply to @chrismatter@gmail.com


To manage your Datadog subscriptions, click [here](#).

High Random Value Monitor

I received an email when my high value monitor was triggered:

[Monitor Alert] Triggered: High Random Data: Random data was high on cmatteri-datadog-hiring-challenge-4275925 Inbox x

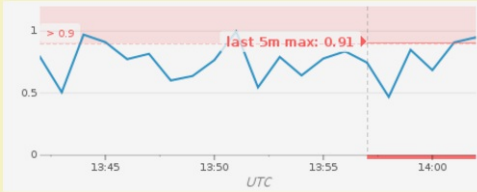
Datadog Alerting alert@datadoghq.com via outbound.dtdg.co to me 9:03 AM (1 hour ago)



[Triggered on {host:cmatteri-datadog-hiring-challenge-4275925}] High Random Data: Random data was high on cmatteri-datadog-hiring-challenge-4275925

test.support.random was over 0.9 during the last five minutes on cmatteri-datadog-hiring-challenge-4275925.

See history here:
[@chrismatteri@gmail.com](https://app.datadoghq.com/dash/234086)




test.support.random over host:cmatteri-datadog-hiring-challenge-4275925 was > 0.9 at least once during the last 5m.

The monitor was last triggered at Mon Jan 16 2017 14:03:00 UTC (27 secs ago).


[\[Monitor Status\]](#) · [\[Edit Monitor\]](#) · [\[View cmatteri-datadog-hiring-challenge-4275925\]](#)

This alert was raised by account Datadog Recruiting Candidate

I also received an email when I scheduled downtime:



A Datadog event mentioned you:



Scheduled downtime on High Random Data: Random data was high on {{host.name}} started

Scheduled downtime on [High Random Data: Random data was high on {{host.name}}](#) has started.

Alerting on [High Random Data: Random data was high on {{host.name}}](#) will be silenced until **2:00PM UTC on January 16**.

@chrismatteri@gmail.com To allow staff to sleep, high random data will not generate alerts between 7 PM and 9 AM EST, effective immediately.

16 Jan, 00:00:32 UTC

Reply on Datadog

To manage your Datadog subscriptions, click [here](#).

Instrumenting a Flask App with Dogstatsd

To learn more about Datadog, I decided to instrument a web app I created. I wanted to know how much time the backend required to process requests. A Flask extension already exists (flask-datadog) that records these metrics and sends them to dogstatsd, so doing this should have been trivial. However, there was an error in its example usage ([I opened an issue and it](#)

was fixed (<https://github.com/50onRed/flask-datadog/issues/10>)), that made it send data to a non-existent host. Due to the lack of acknowledgements in the UDP statsd protocol, using a nonexistent host didn't cause any errors, so I had a fun time figuring out what was wrong!

[WXPlot backend dashboard \(https://app.datadoghq.com/dash/234087/wxplot-backend-dashboard\)](https://app.datadoghq.com/dash/234087/wxplot-backend-dashboard)

