Installation von InfluxDB, auf dem Raspberry Pi 3

von Björn

Im Internet bin ich über folgendes schöne Grafana Dashboard gestolpert und wollte es natürlich gleich selbst ausprobieren.
Ich werde hier also beschreiben wie ich

- InfluxDB, als Datenbank
- Telegraf als Collector
- Grafana als Visualisierung

installiert habe.

1. InfluxDB installieren

Die Daten werden alle in der InfluxDB gesammelt. Die aktuelle Version findet ihr hier.

```
wget https://dl.influxdata.com/influxdb/releases/influxdb_1.5.1_armhf.deb
sudo dpkg -i influxdb_1.5.1_armhf.deb
sudo systemctl enable influxdb
sudo systemctl start influxdb
```

Damit installiert ihr InfluxDB in der Version 1.2.2, aktiviert den automatischen start und startet die Datenbank.

```
pi@raspberrypi:/etc/telegraf $ influx
Connected to http://localhost:8086 version 1.2.2
InfluxDB shell version: 1.2.2
> CREATE USER admin WITH PASSWORD '<password>' WITH ALL PRIVILEGES
> CREAT DATABASE telegraf
> exit
```

Damit legen wir einen User admin an und erzeigen die später verwendete Datenbank.

```
sudo vi /etc/influxdb/influxdb.conf
[http]
  enabled = true
  bind-address = ":8086"
  auth-enabled = true
```

Durch diese Anpassung in der influxdb.conf, aktivieren wir die Authentifizierung. Diese drei Parameter unter [http] sollten überprüft und entsprechend angepasst werden.

```
sudo systemctl restart influxdb
influx -username admin -password password
```

Nach einem Neustart des Services, ist der Zugang nur noch mittels Username und Passwort möglich.

Erste Schritte mit Python und InfluxDB

Von Noah Crowley / 30. März, 2018 / Community , Entwickler , InfluxData , InfluxDB / 13 Kommentare
10 Minuten

Eine aktualisierte version dieses blog-post ist verfügbar für diejenigen, die mit InfluxDB 2.0.

Wenn Sie schauen, um zu überwachen Ihre Infrastruktur oder Anwendungen von Drittanbietern, dann Telegraf ' s built-in plugins sind eine großartige option, ob Sie sich bei der Systemressourcen wie Festplatten-und Netzwerk-Auslastung oder die Leistung Ihrer MySQL-Datenbank.

Was ist, wenn Sie eine Anwendung erstellen, obwohl, wo Sie wollen, um Speicher Benutzer Daten in einem Zeitreihen-Datenbank ? Vielleicht ist es ein IoT-oder smart home-Anwendung, und jeder Benutzer benötigt Zugriff auf Lesungen von, sagen wir, Ihre intelligente Zahnbürste. Sie wollen zu speichern Sie die Zeit und Dauer der einzelnen Bürsten, senden Alarme zu erinnern Sie die Kinder zu Pinsel Ihre Zähne, und verfolgen Dinge wie Batterie Gesundheit und wie lange die aktuelle Pinsel Kopf war im Einsatz.

Sammeln Sie benutzerdefinierte Daten, ob für einen Benutzer-orientierte Anwendung oder für eine Infrastruktur Voraussetzung, dass Telegraf plugins nicht schon decken, ist wahrscheinlich zu verlangen, dass Sie neuen code schreiben.

Für die smart-Zahnbürste Beispiel, haben Sie vielleicht eine Basisstation, die ausgeführt wird, embedded Linux und kommuniziert mit der Zahnbürste mit Bluetooth. Sie habe bereits oben geschrieben code, hört für die eingehenden Daten, und es scheint gut zu funktionieren; jetzt müssen Sie es in InfluxDB.

Eine Methode ist das ausführen von Telegraf neben Ihrer Anwendung, und senden Sie Ihre Daten über eine Unix -, UDP-oder TCP-socket, Vermietung Telegraf Griff die Verbindung zur InfluxDB und Dosierungs-und das schreiben der Punkte.

Das ist großartig, wenn alles, was Sie brauchen, um Daten zu sammeln, aber wenn Sie brauchen, um Abfragen und abrufen, die Daten für Ihre Benutzer, die Sie wahrscheinlich wollen, zu nehmen Vorteil von einer der InfluxDB-Bibliotheken in verschiedenen Sprachen verfügbar zu Griff die Interaktion mit InfluxDB innerhalb der Anwendung selbst.

Es gibt eine Reihe von Sprachen gibt, die bereits InfluxDB Bibliotheken, die viele von Ihnen in der Trägerschaft der Gemeinde. Wir nehmen einen genaueren Blick auf die Verwendung der influxdb-python - Bibliothek in diesem post, aber, wenn Python nicht Ihr Stil ist, Sie finden eine Liste der Bibliotheken auf die InfluxDB-API-client-Bibliotheken Seite

InfluxDB-Python-Client-Bibliothek

Während die influxdb-python - Bibliothek wird gehostet von InfluxDB ' s GitHub-account, es wird verwaltet von einem trio von community-Freiwilligen, @aviau , @xginn8 und @sebito91 . Vielen Dank Ihnen für Ihre harte Arbeit und Ihren Einsatz für die Gemeinschaft.

Was Brauchen Sie?

Die folgenden Beispiele wurden getestet und mit einem MacOS-system mit Python 3 installiert über Homebrew (Anleitung hier) und ein Ubuntu 16.04-system mit der Standard-Python 3-installation.

Installationen von Python erhalten kann ein bisschen knifflig sein, verschiedene Versionen der Sprache, sowie Projekte, die erfordern verschiedene Versionen der Bibliotheken installiert, kann das schnell zu Konflikten führen. Während wir gehen nicht in die details der Python-Installationen hier, zu verstehen, wie verschiedene Versionen installiert sind und miteinander interagieren können, und ein Blick in zusätzliche Werkzeuge wie virtualenv oder pyenv nützlich sein könnten.

Hier finden Sie einige weitere Artikel über die Installation von Python und zusätzliche Werkzeuge an Die Hitchhiker ' s Guide to Python (Mac , Linux).

Wir werden auch senden von Daten an eine lokale Instanz von InfluxDB. Wenn Sie nicht bereits eine haben, können Sie Folgen Sie den Installationsanweisungen auf unserer Dokumentations-Seite , oder verwenden Sie die sandbox-Skripten , um eine vollständige TICK-Stack im Andockfenster.

Installieren der Bibliothek

Wie viele Python-Bibliotheken, der einfachste Weg, um aufzustehen und zu laufen ist um die Bibliothek zu installieren mit `pip`.

Wir gehen laufen `pip` mit der `-m` argument der Python-Befehl, um sicher zu sein, das Python ist das install-Ziel (nach dieser Tipp von Raymond Hettinger).

```
$ python3 -m pip install influxdb
```

Sie sollten sehen, eine Ausgabe, die angibt Erfolg.

Wir arbeiten über einige der Funktionen der Python-Bibliothek, die mit einer REPL , so, dass wir können geben Sie Befehle ein und sofort sehen Sie Ihre Leistung. Let ' s start REPL jetzt, und importieren Sie die `InfluxDBClient` aus der `python-influxdb` Bibliothek zu machen sicher, dass es installiert wurde:

```
$ python3
Python 3.6.4 (default, Mar  9 2018, 23:15:03)
[GCC 4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.39.2)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from influxdb import InfluxDBClient
>>>
```

Keine Fehler—sieht aus wie wir sind bereit zu gehen!

Herstellen einer Verbindung

Der nächste Schritt wird sein, erstellen Sie eine neue Instanz der InfluxDBClient (API-docs), mit Informationen über den server, die wir zugreifen möchten. Geben Sie den folgenden Befehl in der REPL, ersetzen die Werte von `host` und `port` mit der entsprechenden URL/IP-Adresse und den port Ihres InfluxDB host. In diesem Fall sind wir vor Ort läuft auf dem Standard-port:

```
>>> client = InfluxDBClient(host='localhost', port=8086)
```

Es gibt einige zusätzliche Parameter zur Verfügung, um die `InfluxDBClient` Konstruktor, einschließlich Benutzername und Passwort, die Datenbank zu verbinden, ob oder nicht für die Verwendung von SSL-und UDP-timeout-Parameter.

Wenn Sie wollte, um eine Verbindung zu einem remote-host zu `mydomain.com` port `8086` mit Benutzernamen `myuser` und Kennwort `mypass` und mit SSL verwenden, könnten Sie den folgenden Befehl statt, die es ermöglicht, SSL-und SSL-überprüfung mit zwei zusätzliche Argumente, `ssl=True` und `ssl_verify=True`:

```
>>> client = InfluxDBClient(host='mydomain.com', port=8086, username='myuser',
password='mypass' ssl=True, verify_ssl=True)
```

Nun erstellen wir eine neue Datenbank namens `pyexample` zum speichern unserer Daten:

```
>>> client.create_database('pyexample')
```

Wir können überprüfen, ob die Datenbank ist es, mithilfe der `get_list_database()` Funktion client:

```
>>> client.get_list_database()
[{'name': 'telegraf'}, {'name': '_internal'}, {'name': 'pyexample'}]
```

Dort ist es, neben der `telegraf` und `_internal` Datenbanken habe ich auf meinem installieren. Schließlich setzen wir den client um diese Datenbank zu verwenden:

```
>>> client.switch_database('pyexample')
```

Einfügen Von Daten

Nun, wir haben eine Datenbank zu schreiben Daten, und unsere client ordnungsgemäß konfiguriert ist, ist es Zeit, um einige Daten eingefügt werden! Wir verwenden unsere Kunden `write_points()` Methoden, dies zu tun (API docs). Diese Methode akzeptiert eine Liste von Punkten und einige zusätzliche Parameter, darunter "die batch-Größe", die gibt uns die Fähigkeit zum einfügen von Daten in batches im Gegensatz zu allen auf einmal. Dies kann nützlich sein, wenn Sie das einfügen großer Datenmengen.

Die `write_points()` Methode ein argument genannt `points`, die ist eine Liste von Wörterbüchern, und enthält die Punkte werden in die Datenbank geschrieben. Lassen Sie uns einige Beispieldaten jetzt und fügen Sie ihn ein. Erste, fügen wir drei Punkte im JSON-format, um eine variable mit dem Namen `json_body`:

```
>>> json_body = [
```

```
    {
        "measurement": "brushEvents",
        "tags": {
            "user": "Carol",
            "brushId": "6c89f539-71c6-490d-a28d-6c5d84c0ee2f"
        },
        "time": "2018-03-28T8:01:00Z",
        "fields": {
            "duration": 127
        }
    },
    {
        "measurement": "brushEvents",
        "tags": {
            "user": "Carol",
            "brushId": "6c89f539-71c6-490d-a28d-6c5d84c0ee2f"
        },
        "time": "2018-03-29T8:04:00Z",
        "fields": {
            "duration": 132
        }
    },
    {
        "measurement": "brushEvents",
        "tags": {
            "user": "Carol",
            "brushId": "6c89f539-71c6-490d-a28d-6c5d84c0ee2f"
        },
        "time": "2018-03-30T8:02:00Z",
        "fields": {
            "duration": 129
        }
    }
]
```

Diese zeigen "brush Ereignisse" für unsere smart Zahnbürste, jeder von Ihnen geschieht um 8 Uhr morgens, mit Tags versehen ist, mit dem Benutzernamen der person, die mit der Zahnbürste und eine ID an der Bürste selbst (so dass wir können verfolgen, wie lange jeder Pinsel Kopf verwendet wurde), und hat ein Feld, in dem, wie lange der Benutzer gebürstet für in Sekunden.

Da wir bereits haben unsere Datenbank eingestellt und der Standard-Eingang für `write_points()` ist JSON, können wir die Methode aufruft Nutzung unserer `json_body` variable als einziges argument wie folgt:

```
>>> client.write_points(json_body)
True
```

Sollten Sie die Antwort `True` wird von der Funktion zurückgegeben wird, wenn der Schreibvorgang erfolgreich war. Wenn Sie eine Anwendung erstellen, würden Sie wollen diese Sammlung von Daten, um die automatische, hinzufügen von Punkten, um die Datenbank jedes mal, wenn ein Benutzer interagiert mit der Zahnbürste.

Abfragen Von Daten

Nun, wir haben einige Daten in der Datenbank, lassen Sie uns versuchen, einige Abfragen, um es

wieder heraus. Wir verwenden die gleichen client-Objekt, das wir verwendet, um Daten zu schreiben, außer dieser Zeit, wir werden ausführen einer Abfrage auf InfluxDB und die Ergebnisse mit unseren Kunden `query()` function (API docs).

```
>>> client.query('SELECT "duration" FROM "pyexample"."autogen"."brushEvents"
WHERE time > now() - 4d GROUP BY "user"')
>>>
```

Die `query()` die Funktion gibt ein `ResultSet` object (API-Docs), die alle Daten enthält der Folge zusammen mit einigen Komfort-Methoden. Unserer Abfrage anfordert, alle Messungen in unsere `pyexample` Datenbank, gruppiert nach Benutzer. Sie verwenden können die `.raw` parameter für den Zugriff auf die raw-JSON-Antwort von InfluxDB:

```
>>> results.raw
{'statement_id': 0, 'series': [{'name': 'brushEvents', 'tags': {'user':
'Carol'}, 'columns': ['time', 'duration'], 'values': [['2018-03-28T08:01:00Z',
127], ['2018-03-29T08:04:00Z', 132], ['2018-03-30T08:02:00Z', 129]]}]}
```

In den meisten Fällen werden Sie nicht brauchen, um auf die JSON direkt, aber. Verwenden Sie stattdessen die `get_points()` Methode der `ResultSet` um die Messungen aus der Anforderung, eine Filterung nach tag oder Feld. Wenn Sie wollte zu Durchlaufen aller Carol ' s Bürsten Sitzungen; Sie könnten erhalten alle Punkte, die zusammengefasst unter der Bezeichnung "Benutzer" mit dem Wert "Carol", mit diesem Befehl:

```
>>> points = results.get_points(tags={'user':'Carol'})
```

`points` in diesem Fall ist ein Python-Generator, die ist eine Funktion, die funktioniert ähnlich wie ein Iterator; Sie können iterieren über ihn mit ein `for x in y` Schleife wie folgt:

```
>>> points = results.get_points(tags={'user': 'Carol'})
>>> for point in points:
...     print("Time: %s, Duration: %i" % (point['time'], point['duration']))
...
Time: 2018-03-28T08:01:00Z, Duration: 127
Time: 2018-03-29T08:04:00Z, Duration: 132
Time: 2018-03-30T08:02:00Z, Duration: 129
```

Je nach Anwendung, die Sie möglicherweise Durchlaufen Sie diese Punkte, um berechnen Sie die Durchschnittliche Putzzeit für Ihre Benutzer, oder einfach nur, um zu überprüfen, dass X Anzahl von Bürsten Veranstaltungen pro Tag.

Wenn Sie daran interessiert waren, die Verfolgung der Menge der Zeit, die ein einzelner Pinsel Kopf, verwendet wurde, Sie könnte ersetzen eine neue Abfrage, die Gruppen Punkte, die auf `brushId`, dann nehmen Sie die Dauer jeder dieser Punkte und fügen Sie es hinzu, um eine Summe. An einem bestimmten Punkt konnten Sie aufmerksam Ihre Benutzer, dass es Zeit zu ersetzen Ihre Pinsel Kopf:

```
>>> results = client.query('SELECT "duration" FROM
"pyexample"."autogen"."brushEvents" WHERE time > now() - 4d GROUP BY "brushId"')
>>> points = results.get_points(tags={'brushId': '6c89f539-71c6-490d-a28d-
6c5d84c0ee2f'})
>>> brush_usage_total = 0
>>> for point in points:
...     brush_usage_total = brush_usage_total + point['duration']
...
>>> if brush_usage_total > 350:
...     print("You've used your brush head for %s seconds, more than the
recommended amount! Time to replace your brush head!" % brush_usage_total)
...
You've used your brush head for 388 seconds, more than the recommended amount!
Time to replace your brush head!
>>>
```

Zusätzliche Dokumentation und Funktionalität

Die `influx-python` Bibliothek enthält eine ganze Menge von zusätzlichen Funktionen, die wir noch nicht abdecken, die in dem Artikel oben. Es gibt weitere administrative Funktionen in der client-wie das hinzufügen von Benutzern, das verwalten von Datenbanken und löschen von Messungen, sowie weitere Objekte wie `SeriesHelper` bietet einige Komfort-Funktionen zum schreiben von Punkten in bulk, und `DataFrameClient`, die erleichtert die integration mit PANDAS und DataFrames.

Wenn Sie interessiert sind, die diese Bibliothek nutzen, in Ihren Projekten, macht es Sinn, verbringen einige Zeit mit der API-Dokumentation und source-code , verstehen nicht nur die Funktionalität bereitgestellt wird, aber die Art und Weise es funktioniert hinter den kulissen.

Und wenn Sie etwas bauen, cool mit InfluxDB, wir würden uns freuen, Sie auf unserem blog, so teilen Sie es mit uns auf Twitter @InfluxDB !