Stack Telegraf / InfluxDB / Grafana

InfluxDB

Connexion à la BDD

Tout simplement (si on se trouve sur le serveur qui héberge la base de données et en supposant que le client influx est dans le PATH)

```
influx -precision rfc3339 -database 'influxdb' -username '<USER>' -password
influx -precision ns -database 'influxdb' -username '<USER>' -password
```

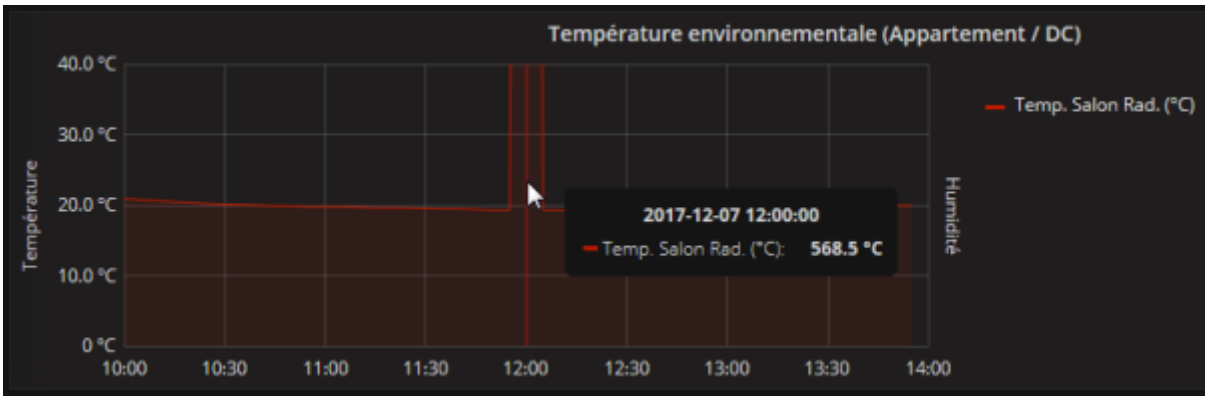
Correction d'un MEASUREMENT

Méthode subtile

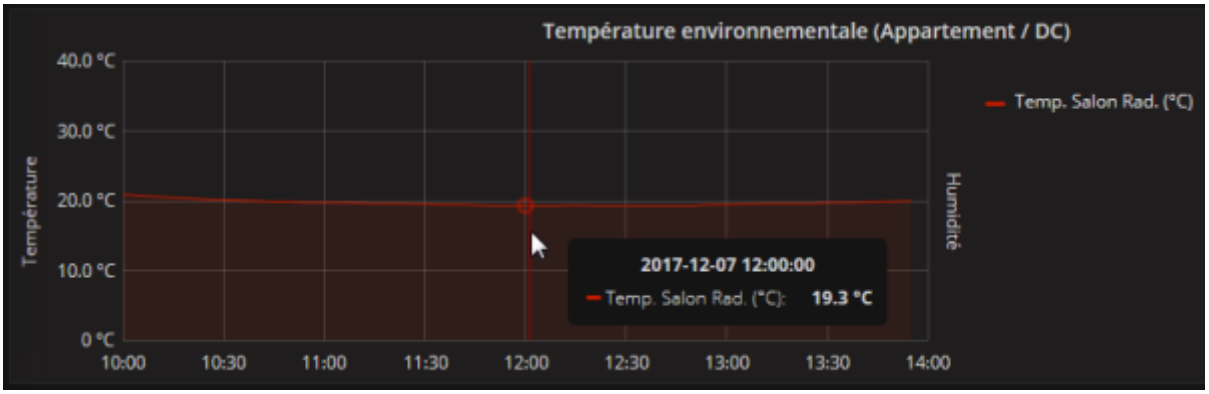
Contexte :
J'ai des erreurs de mesures[1] remontées par une de mes sondes de température, que je souhaite corriger à la méthode du garagiste Islandais[2] (Private Joke) :

- Nom de la base InfluxDB : `influxdb`
- La SERIE est stockée dans la MEASUREMENT **`Temperature_temper`**
- Fréquence de collecte : `5 min / 300 sec`
- Valeur à corriger :
  - Points de **température > 100°C** (la pièce à vivre n'est pas censée être un four)

Avant correction :



Après correction :



On commence par se connecter à la base de données qui va bien...

On affiche la structure de la **MEASUREMENT**

```
> SHOW SERIES FROM Temperature_temper
key
---
Temperature_temper,host=drouard.eu

> SHOW TAG KEYS FROM Temperature_temper
name: Temperature_temper
tagKey
------
host

> SHOW FIELD KEYS FROM Temperature_temper
name: Temperature_temper
fieldKey fieldType
-------- ---------
value    float
```

☛ Structure toute simple, contenant 1 SERIE avec :

- 1 tagKey `host`
- 1 fieldKey `value` de type Float

Commençons par évaluer l'ampleur de la tâche à réaliser (et voir si il ne serait pas plus efficace de créer un script...)

```
> SELECT COUNT("value") FROM Temperature_temper WHERE value > 100;
name: Temperature_temper
time                 count
----                 -----
1970-01-01T00:00:00Z 1
```

Bon ok, 1 point de mesure à corriger, *c'est pas si pire*...

Pour appliquer la 2ème méthode, qui implique d'utiliser le timestamp au format **epoch** (précision à la nanoseconde), je ne saurais que conseiller aussi de lister les points à corriger avec le format de précision `ns`

Liste des points de **température** à corriger

```
> PRECISION rfc3339
> SELECT * FROM Temperature_temper WHERE value > 100 LIMIT 10;
name: Temperature_temper
time                 host       value
----                 ----       -----
2017-12-07T11:00:04Z drouard.eu 568.5

> PRECISION ns
> SELECT * FROM Temperature_temper WHERE value > 100 LIMIT 10;
name: Temperature_temper
time                 host       value
----                 ----       -----
1512644404000000000 drouard.eu 568.5
```

Passons à la correction !
En fonction de la nature, criticité de la métrique, 2 possibilités pour corriger l'erreur :

- Méthode 1 ☛ **Supprimer la valeur** erronée :
  - Il est possible de corriger l'apparence de la courbe à la volée dans Grafana avec la clause `GROUP BY fill(0/null/none/etc.)`... c'est fourbe, mais esthétique... inconvénient, ça masquerait les erreurs de collectes trop récurrentes 😕
- Méthode 2 ☛ **Maquiller la valeur** erronée (Ce n'est qu'une extension de la méthode 1) :
  - Cette stratégie consiste sournoisement à faire une moyenne (par ex.) entre les 2 valeurs entourant le pic puis à inscrire cette valeur à la place de la valeur du pic

On commence par identifier le laps de temps incriminé via une petite requête (±1 période de collecte) :

```
> PRECISION rfc3339
> SELECT * FROM Temperature_temper WHERE time >= '2017-12-07T10:54:04Z' AND
time <= '2017-12-07T11:06:04Z'
name: Temperature_temper
time                 host       value
----                 ----       -----
2017-12-07T10:55:01Z drouard.eu 19.29
2017-12-07T11:00:04Z drouard.eu 568.5
2017-12-07T11:05:01Z drouard.eu 19.31
```

On supprime la valeur erronée

```
DELETE FROM Temperature_temper WHERE "host" = 'drouard.eu' AND time =
'2017-12-07T11:00:04Z'
```

On vérifie que la mesure à bien été supprimée

```
> SELECT * FROM Temperature_temper WHERE time >= '2017-12-07T10:54:04Z' AND
time <= '2017-12-07T11:06:04Z'
name: Temperature_temper
```

```
time                 host       value
----                 ----       -----
2017-12-07T10:55:01Z drouard.eu 19.29
2017-12-07T11:05:01Z drouard.eu 19.31
```

La **méthode 1** consiste à en rester là et éventuellement maquiller légèrement la courbe dans Grafana...

Maintenant, on va aller un peu plus loin avec la **méthode 2**.

Note : il faut savoir que si l'on insert une valeur avec un **timestamp existant**, alors l'`INSERT` agira comme un `UPDATE` (qui d'ailleurs, n'existe pas en tant que tel dans influxDB). Ce qui implique que pour la méthode 2, il n'est pas nécessaire de supprimer la valeur précédente 😎

On va devoir injecter une nouvelle valeur avec comme paramètres :

- **`value`** = Moyenne des valeurs encadrant
  - Dans cet ex : `(19.29 + 19.31) / 2` = **19.30**
- **`timestamp`** = Timestamp au format epoch, précision à la nanoseconde
  - Dans cet ex : `timestamp` = **1512644404000000000**

A noter qu'il est possible de convertir une date au format RFC-3339 vers le format epoch avec une précision à la nanoseconde avec la commande `date` :

```
date -d 2017-12-07T11:00:04Z +%s%N
```

Syntaxe générique d'insertion de donnée dans InfluxDB (Docs de référence sur https://docs.influxdata.com : Writing and exploring data et Line Protocol) :

```
<measurement>[,<tag_key>=<tag_value>[,<tag_key>=<tag_value>]]
<field_key>=<field_value>[,<field_key>=<field_value>] [<timestamp>]
```

Ensuite, on peut forger la requête d'insertion soit :

- En mode **CLI** via le client `influx`

```
INSERT Temperature_temper,host=drouard.eu value=19.30 1512644404000000000
```

- En mode **HTTP API** (C'est à la mode... et et c'est pratique aussi) via `curl`

```
curl -i -XPOST -u USERNAME:PASSWORD
"http://localhost:8086/write?db=db_drouard&precision=ns" --data-binary
'Temperature_temper,host=drouard.eu value=19.30 1512644404000000000'
```

Enfin, on vérifie la correction

```
> SELECT * FROM Temperature_temper WHERE time >= '2017-12-07T10:54:04Z' AND time <= '2017-12-07T11:06:04Z'
name: Temperature_temper
time                host       value
----                ----       -----
1512644101000000000 drouard.eu 19.29
1512644404000000000 drouard.eu 19.3
1512644701000000000 drouard.eu 19.31
```

Méthode (moins) subtile

Contexte :
J'ai un MEASUREMENT que je dois renommer dans les lesquels, j'ai eu des TAG ou des FIELD crée à tord, avec certains TAG que je vais devoir fusionner :

- Nom de la base InfluxDB : `influxdb`
- La SERIE est stockée dans la MEASUREMENT **`Temperature_temper`**
- Éléments à corriger :
  - Le nom du MEASUREMENT **`Temperature_temper`** devient **`nagios_temper`**
  - Le TAG **`perfdata`** doit être supprimé
  - Les TAG **`host`** et **`host_1`** doivent fusionner pour devenir **`host`**

La structure du MEASUREMENT étant :

```
name: Temperature_temper
time                host       host_1     perfdata    value
----                ----       ------     --------    -----
1512659404000000000 drouard.eu                        20.06
1512659702000000000 drouard.eu                        20.06
1512660004000000000 drouard.eu                        20.06
...
1541081404000000000            drouard.eu Temperature 20.12
1541081703000000000            drouard.eu Temperature 20.12
1541082004000000000            drouard.eu Temperature 20.12
...
```

On commence par exporter les données

```
influx -database 'influxdb' -username '<USER>' -password -execute "SELECT * FROM Temperature_temper" > /tmp/export_Temperature_temper.txt
```

Ensuite je supprime les entêtes (les 3 premières lignes)

```
sed -i '1,3d' /tmp/export_Temperature_temper.txt
```

Je supprime les données contenu dans le TAG **perfdata** (cela reviendra à supprimer la colonne par la suite)

```
sed -i 's/ Temperature//g' /tmp/export_Temperature_temper.txt
```

Ensuite, je formate avec awk le fichier à injecter dans la base InfluxDB, pour rappel, la syntaxe générique d'insertion de donnée dans InfluxDB (Docs de référence sur https://docs.influxdata.com : Writing and exploring data et Line Protocol) :

```
<measurement>[,<tag_key>=<tag_value>[,<tag_key>=<tag_value>]]
<field_key>=<field_value>[,<field_key>=<field_value>] [<timestamp>]
```

```
awk '{print "nagios_temper,host="$2" value="$3" "$1}'
/tmp/export_Temperature_temper.txt | head -5
```

Si le format est correct, je génère le fichier d'import (c'est ici que l'on définit le nouveau MEASUREMENT)

```
awk '{print "nagios_temper,host="$2" value="$3" "$1}'
/tmp/export_Temperature_temper.txt > /tmp/import_Temperature_temper.txt
```

On aurait pu tout à fait manipuler les données manuellement dans un éditeur de texte (vim, notepad++, etc.) ou dans un tableur tel que LibreOffice au format CSV... Mais ce n'est pas forcément toujours plus simple et dépend du nombre et du type de modification 

Et je l'importe avec `curl` dans la base de données InfluxDB

```
curl -i -XPOST -u USERNAME:PASSWORD
"http://localhost:8086/write?db=db_drouard&precision=ns" --data-binary
@/tmp/import_Temperature_temper.txt
```

L'import est complet si curl nous retourne le **code HTTP 204**

```
HTTP/1.1 100 Continue

HTTP/1.1 204 No Content
Content-Type: application/json
Request-Id: 7099242b-dde8-11e8-9845-000000000000
X-Influxdb-Build: OSS
X-Influxdb-Version: 1.x.x
X-Request-Id: 7099242b-dde8-11e8-9845-000000000000
Date: Thu, 01 Nov 2018 15:11:45 GMT
```

Copier un MEASUREMENT

On utilise la commande `SELECT * INTO …` (La subtilité étant le paramètre **`GROUP BY *`** qui permet de garder les TAG du MEASUREMENT)

```
SELECT * INTO openweathermapNEW FROM openweathermap GROUP BY *
```

Supprimer un TAG / FIELD d'un MEASUREMENT

On reprend la copie d'un MEASUREMENT mais on définit les FIELD a copier dans le SELECT et les TAG dans le GROUP BY.

```
SELECT time,main_humidity,main_temp INTO openweathermapNEW FROM
openweathermap GROUP BY host
```

On surprime l'ancien MEASUREMENT

```
DROP MEASUREMENT openweathermap
```

Et on recopie intégralement le nouveau MEASUREMENT avec l'ancien nom

```
SELECT * INTO openweathermap FROM openweathermapNEW GROUP BY *
```

1)
Problème corrigé depuis… causé par une concurrence d'accès à la sonde entre Nagios et Telegraf.
2)
Fix It or Remove It :p