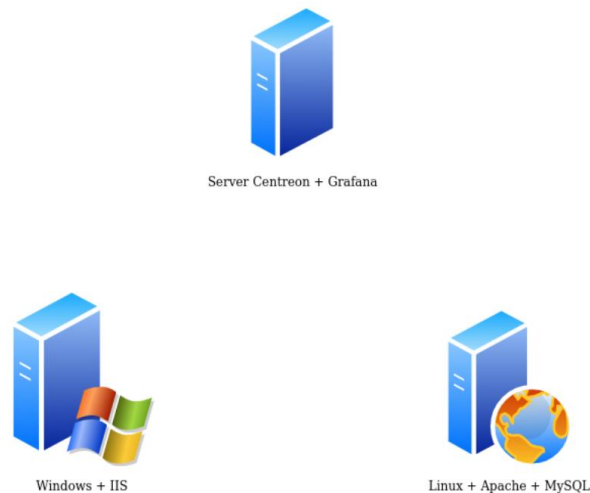


Table des matières

1. Architecture du réseau	1
2. Installation de Grafana	1
3. Installation d'influxdb.....	4
4. Mise en place de collecteurs Windows	8
5. Mise en place des collecteurs Linux	10
6. Affichage des données grâce à un tableau de bord	11

1. Architecture du réseau

L'architecture de notre réseau est la suivante :



2. Installation de Grafana

Pour installer Grafana nous nous aidons du site suivant : <https://www.how2shout.com/linux/how-to-install-grafana-on-almalinux-or-rocky-linux-8/>

Nous devons commencer par ajouter le répertoire RPM :

```
[root@localhost ~]# tee /etc/yum.repos.d/grafana.repo<<EOF
> [grafana]
> name=grafana
> baseurl=https://packages.grafana.com/oss/rpm
> repo_gpgcheck=1
> enabled=1
> gpgcheck=1
```

```
> gpgkey=https://packages.grafana.com/gpg.key
> sslverify=1
> sslcacert=/etc/pki/tls/certs/ca-bundle.crt
>
> EOF

[grafana]
name=grafana
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
```

Nous devons maintenant mettre à jour notre OS pour actualiser les paquets de Grafana :

```
[root@localhost ~]# dnf update -y

grafana
926 B/s | 454 B      00:00

grafana
12 kB/s | 1.7 kB     00:00

Import de la clef GPG 0x24098CB6 :
Utilisateur : « Grafana <info@grafana.com> »
Empreinte : 4E40 DDF6 D76E 284A 4A67 80E4 8C8C 34C5 2409 8CB6
Provenance : https://packages.grafana.com/gpg.key
... ..
... ..
```

Maintenant nous pouvons installer Grafana :

```
[root@localhost ~]# dnf install grafana

Dernière vérification de l'expiration des métadonnées effectuée il y a
0:04:08 le ven. 14 janv. 2022 02:21:39 EST.

Dépendances résolues.

=====
=====

  Paquet                                Architecture
Version                                Dépôt                                Taille
=====
=====
```

Installation:

```
grafana                                x86_64
8.3.3-1                                grafana                                69 M
... ..
```

Une fois notre Grafana installé nous devons le démarrer et activer le système :

```
[root@localhost ~]# systemctl start grafana-server
[root@localhost ~]# systemctl enable grafana-server

Synchronizing state of grafana-server.service with SysV service script with
/usr/lib/systemd/systemd-sysv-install.

Executing: /usr/lib/systemd/systemd-sysv-install enable grafana-server

Created symlink /etc/systemd/system/multi-user.target.wants/grafana-
server.service → /usr/lib/systemd/system/grafana-server.service.
```

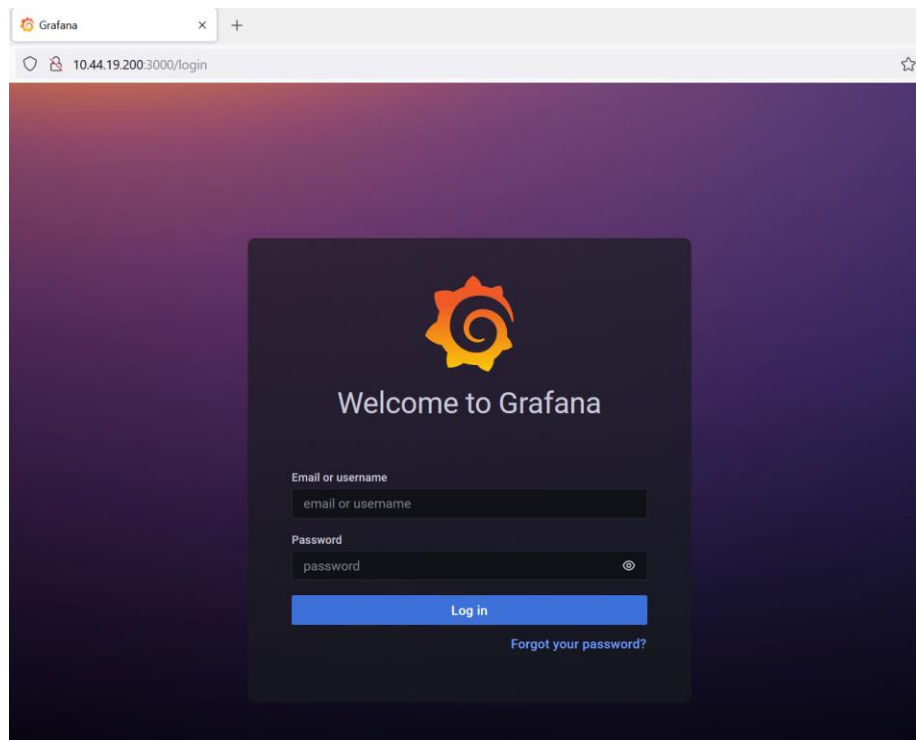
Nous devons aussi modifier le firewall pour autoriser l'accès à Grafana sur son port :

```
[root@localhost ~]# firewall-cmd --add-port=3000/tcp --permanent
success

[root@localhost ~]# firewall-cmd --reload
success
```

Si nous voulons utiliser un autre port que le 3000 nous devons modifier le fichier **/usr/share/grafana/conf/defaults.ini** et changer la ligne **"http_port = 3000"**

Si toute l'installation s'est bien passée nous devons avoir accès à notre tableau de bord en ouvrant un navigateur sur l'adresse de notre serveur avec le bon port (ici 3000) :



Nous pouvons donc nous connecter avec les identifiants admin/admin

Le mot de passe est à changer lors de la première connexion mais nous pouvons laisser admin/admin

Une fois le mot de passe changé nous arrivons sur les tableaux de bords de Grafana

3. Installation d'influxdb

Pour installer influxdb nous nous sommes aidés du site suivant :

<https://computingforgeeks.com/how-to-install-influxdb-on-rhel-8-centos-8/>

Nous devons aussi ajouter un répertoire RPM :

```
[root@localhost ~]# cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
> [influxdb]
> name = InfluxDB Repository - RHEL \${releasever}
> baseurl =
https://repos.influxdata.com/rhel/\${releasever}/\${basearch}/stable
> enabled = 1
> gpgcheck = 1
> gpgkey = https://repos.influxdata.com/influxdb.key
> EOF
[influxdb]
name = InfluxDB Repository - RHEL \${releasever}
baseurl = https://repos.influxdata.com/rhel/\${releasever}/\${basearch}/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
```

Nous devons mettre à jour le cache ou mettre à jour l'OS :

```
[root@localhost ~]# dnf makecache
Rocky Linux 8 - AppStream
12 kB/s | 4.8 kB      00:00
Rocky Linux 8 - BaseOS
13 kB/s | 4.3 kB      00:00
Rocky Linux 8 - Extras
9.7 kB/s | 3.5 kB      00:00
Rocky Linux 8 - PowerTools
13 kB/s | 4.8 kB      00:00
... ..
Cache des métadonnées créé.
```

Une fois le cache ou l'OS mis à jour nous pouvons installer influxdb :

```
[root@localhost ~]# dnf -y install influxdb
```

```
Dernière vérification de l'expiration des métadonnées effectuée il y a  
0:01:21 le ven. 14 janv. 2022 02:47:50 EST.
```

```
Dépendances résolues.
```

```
=====
=====

  Paquet                Architecture
Version                Dépôt
Taille

=====
=====

Installation:

  influxdb                x86_64
1.8.10-1                influxdb                52
M

... ..
... ..
```

Nous pouvons vérifier la bonne installation :

```
[root@localhost ~]# rpm -qi influxdb

Name           : influxdb
Version        : 1.8.10
Release        : 1
Architecture   : x86_64
Install Date:  ven. 14 janv. 2022 02:49:20 EST
... ..
```

Nous pouvons maintenant démarrer et activer le système :

```
[root@localhost ~]# systemctl start influxdb
[root@localhost ~]# systemctl enable influxdb
```

Nous pouvons vérifier que le système est bien démarré :

```
[root@localhost ~]# systemctl status influxdb

● influxdb.service - InfluxDB is an open-source, distributed, time series
database

   Loaded: loaded (/usr/lib/systemd/system/influxdb.service; enabled;
   vendor preset: disabled)

   Active: active (running) since Fri 2022-01-14 02:52:09 EST; 44s ago
     Docs: https://docs.influxdata.com/influxdb/

  Main PID: 26207 (influxd)

    Tasks: 8 (limit: 11408)

   Memory: 8.4M
```

```
CGroup: /system.slice/influxdb.service
└─26207 /usr/bin/influxd -config /etc/influxdb/influxdb.conf
... ..
```

Nous devons configurer le firewall pour autoriser la connexion entre le client et le serveur sur le port 8086 :

```
[root@localhost ~]# firewall-cmd --add-port=8086/tcp --permanent
success
[root@localhost ~]# firewall-cmd --reload
success
```

Nous pouvons aussi changer le port utilisé par influxdb dans `/etc/influxdb/influxdb.conf`

Si nous voulons activer l'authentification http nous devons le faire dans `/etc/influxdb/influxdb.conf` :

```
[http]

# Determines whether user authentication is enabled over HTTP/HTTPS.
auth-enabled = true
```

Puis redémarrer le service :

```
[root@localhost ~]# systemctl restart influxdb
```

Et créer un utilisateur avec un mot de passe :

```
[root@localhost ~]# curl -XPOST "http://localhost:8086/query" --data-urlencode "q=CREATE USER \
> $USER$ WITH PASSWORD '$PASSWD' WITH ALL PRIVILEGES"
{"results":[{"statement_id":0}]}
```

! \ Attention à bien remplacer les deux macros dans la commande par le nom d'utilisateur et le mot de passe

Maintenant si nous voulons lancer influxdb dans notre terminal nous devons lancer la commande suivante en remplaçant bien le nom d'utilisateur et le mot de passe :

```
influx -username 'username' -password 'password'
```

Pour une utilisation avec curl :

```
curl -G http://localhost:8086/query -u username:password --data-urlencode "q=SHOW DATABASES"
```

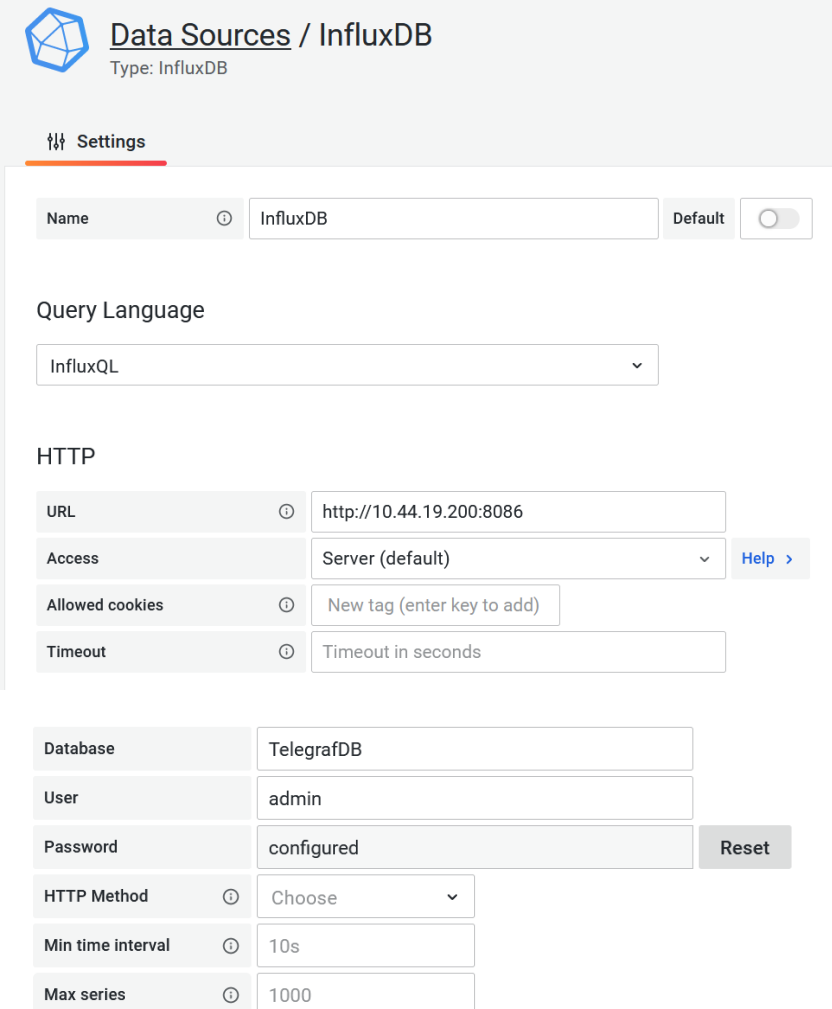
Nous pouvons vérifier qu'influxdb écoute bien sur le port 8086 :

```
[root@localhost ~]# ss -tunelp | grep 8086
tcp    LISTEN 0          128          *:8086          *:*
users: (("influxd",pid=27078,fd=14)) uid:982 ino:229448 sk:13 v6only:0 <->
```

Nous devons sur notre serveur créer une base de données avec influxdb :

```
[root@localhost ~]# influx -username 'admin' -password 'admin' -execute  
"create database TelegrafDB"  
  
[root@localhost ~]# influx -username 'admin' -password 'admin' -execute  
"show databases"  
  
name: databases  
  
name  
----  
_internal  
TelegrafDB
```

Une fois influxdb installé et sa base de données créée nous pouvons ajouter une datasource dans Grafana lié à notre base de données influxdb :



The screenshot shows the 'Data Sources / InfluxDB' configuration page in Grafana. The page has a header with the InfluxDB logo and the title 'Data Sources / InfluxDB' with a subtitle 'Type: InfluxDB'. Below the header is a 'Settings' tab. The main configuration area includes a 'Name' field set to 'InfluxDB' with a 'Default' toggle switch. Under 'Query Language', a dropdown menu is set to 'InfluxQL'. The 'HTTP' section contains fields for 'URL' (http://10.44.19.200:8086), 'Access' (Server (default)), 'Allowed cookies' (New tag (enter key to add)), and 'Timeout' (Timeout in seconds). At the bottom, there are fields for 'Database' (TelegrafDB), 'User' (admin), 'Password' (configured), 'HTTP Method' (Choose), 'Min time interval' (10s), and 'Max series' (1000). A 'Reset' button is located next to the 'Password' field.

Field	Value
Name	InfluxDB
Query Language	InfluxQL
URL	http://10.44.19.200:8086
Access	Server (default)
Allowed cookies	New tag (enter key to add)
Timeout	Timeout in seconds
Database	TelegrafDB
User	admin
Password	configured
HTTP Method	Choose
Min time interval	10s
Max series	1000

Nous aurions aussi pu le faire en ligne de commande :

```
[root@localhost ~]# curl -s -H "Content-Type: application/json" -XPOST
http://admin:admin@localhost:3000/api/datasources -d @- << EOF

{
    "name": "Telegraf",
    "type": "influxdb",
    "url": "http://10.44.19.200:8086",
    "access": "proxy",
    "basicAuth": false,
    "database": "TelegrafDB",
    "isDefault": true,
    "jsonData": {
        "timeInterval": "10s"
    }
}

EOF
```

4. Mise en place de collecteurs Windows

Installation de telegraf sous windows :

Nous devons d'abord installer chocolatey en ouvrant un powershell :

```
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-
Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/inst
all.ps1'))
```

Une fois installé nous pouvons installer telegraf :

```
choco install telegraf -y
```

Nous pouvons vérifier son état :

```
Get-Service telegraf
```

Nous devons maintenant modifier le fichier de configuration de telegraf pour donner l'IP de la base de données pour l'envoi des données et son nom :

Pour le modifier en CLI nous devons nous placer dans le répertoire contenant le fichier de configuration telegraf et lancer la commande suivante :

```
notepad .\telegraf.conf
```



```
urls = [ "http://monserveur.exemple.com:8086" ] # required
# The target database for metrics (telegraf will create it if not exists)
database = "TelegrafDB" # required
## HTTP Basic Auth
    username = "admin"
    password = "admin"
```

Nous devons relancer le service afin de prendre en compte la modification du fichier :

```
Get-Service telegraf | Restart-Service
```

Nos collecteurs sont normalement mis en place et nous pouvons vérifier cela sur notre serveur Rocky dans la base de données :

```
[root@localhost etc]# influx --username admin --password admin
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
> show databases
name: databases
name
----
_internal
TelegrafDB
telegraf
> use TelegrafDB
Using database TelegrafDB
> show measurements
name: measurements
name
----
apache
cpu
disk
... ..
swap
system
> SHOW TAG VALUES FROM system WITH KEY=host
name: system
key  value
```

```
---  -----  
host WIN-QLJBJDQ0KEL  
host debian10.linuxvmimages.local
```

Nous retrouvons bien notre hôte Windows Serveur donc cela veut dire que nous recevons bien les données de la machine

5. Mise en place des collecteurs Linux

Pour le collecteur à mettre en place nous avons choisit Telegraf et pour le mettre en place nous nous sommes aidés du site suivant : <https://www.aukfood.fr/grafana-avec-influxdb-et-telegraf/>

Tout d'abord nous devons récupérer les dépôts d'influxdb qui contiennent Telegraf

```
apt-get install curl apt-transport-https  
curl -sL https://repos.influxdata.com/influxdb.key | apt-key add -  
echo "deb https://repos.influxdata.com/debian jessie stable" >  
/etc/apt/sources.list.d/influxdb.list  
apt-get update
```

Maintenant nous pouvons installer Telegraf :

```
apt-get install telegraf
```

Comme pour Windows nous devons maintenant modifier le fichier de configuration de telegraf dans **/etc/telegraf/telegraf.conf** :

```
[[outputs.influxdb]]  
    urls = ["http://10.44.19.200:8086"]  
    database = "TelegrafDB"  
    ## HTTP Basic Auth  
    username = "admin"  
    password = "admin"
```

Nous devons aussi redémarrer le service pour prendre en compte les changements :

```
systemctl restart telegraf
```

Nous pouvons vérifier dans la base de données influxdb que notre machine envoie bien des données :

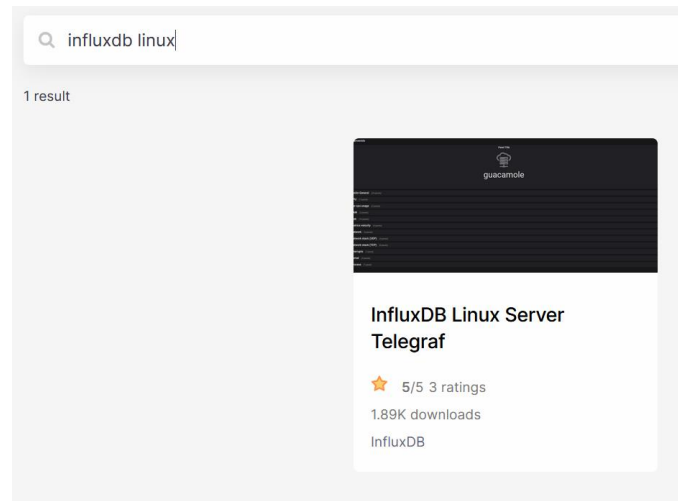
```
> SHOW TAG VALUES FROM system WITH KEY=host  
name: system  
key  value  
---  -----  
host WIN-QLJBJDQ0KEL  
host debian10.linuxvmimages.local
```

6. Affichage des données grâce à un tableau de bord

Pour afficher les données nous allons sur le site suivant : <https://grafana.com/grafana/dashboards/>

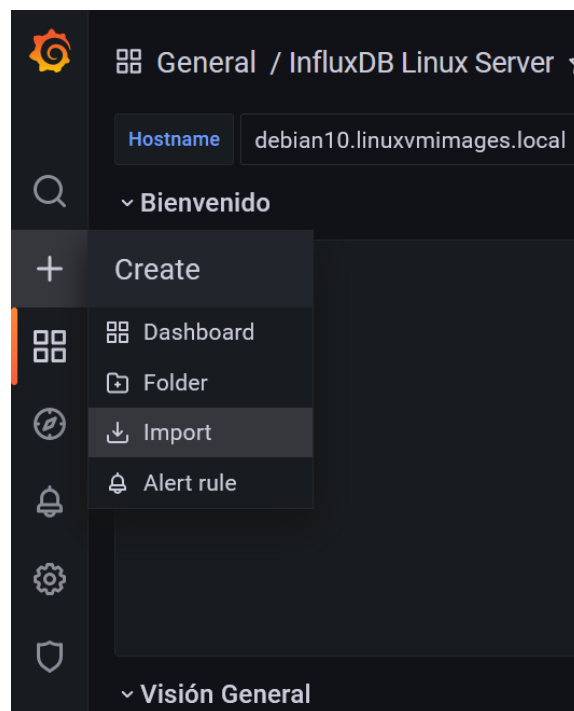
Nous recherchons un Dashboard qui correspond à notre machine et qui remonte les données que nous avons besoin. Nous pourrions créer notre propre Dashboard mais cela nous prendrait beaucoup plus de temps

Par exemple pour la machine Linux :

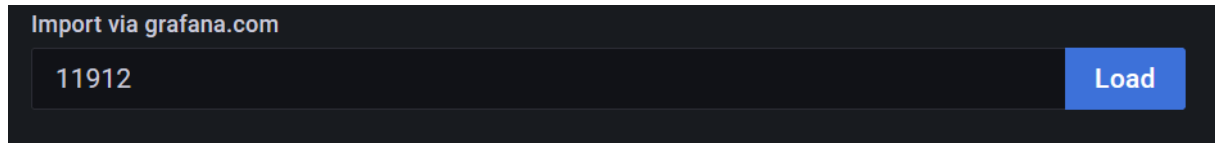


Nous prenons donc ce Dashboard en allant dessus et en copiant son ID (ici 11912)

Une fois l'ID copié nous retournons sur notre grafana et nous allons dans l'onglet "+" puis **"import"** :



Nous saisissons l'ID copié précédemment dans la fenêtre "**import via grafana.com**" puis nous appuyons sur le bouton load :

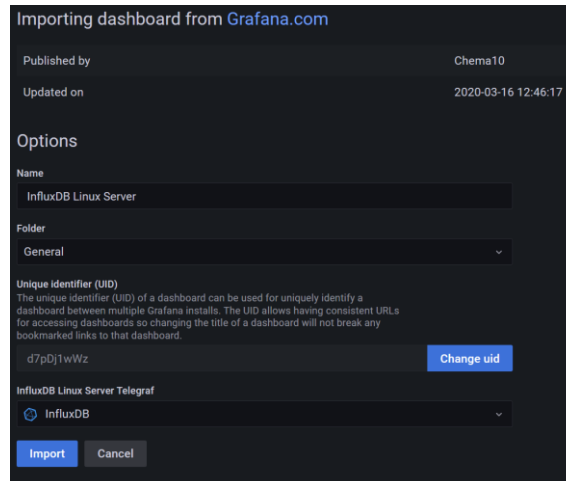


Import via grafana.com

11912

Load

Après avoir appuyé sur le bouton nous arrivons dans un menu qui nous permet de modifier les options de notre Dashboard (ici nous changeons son nom et nous le lions à notre data source) :



Importing dashboard from Grafana.com

Published by Chema10

Updated on 2020-03-16 12:46:17

Options

Name InfluxDB Linux Server

Folder General

Unique Identifier (UID)
The unique identifier (UID) of a dashboard can be used to uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

d7pDj1wWz

Change uid

InfluxDB Linux Server Telegraf

InfluxDB

Import Cancel

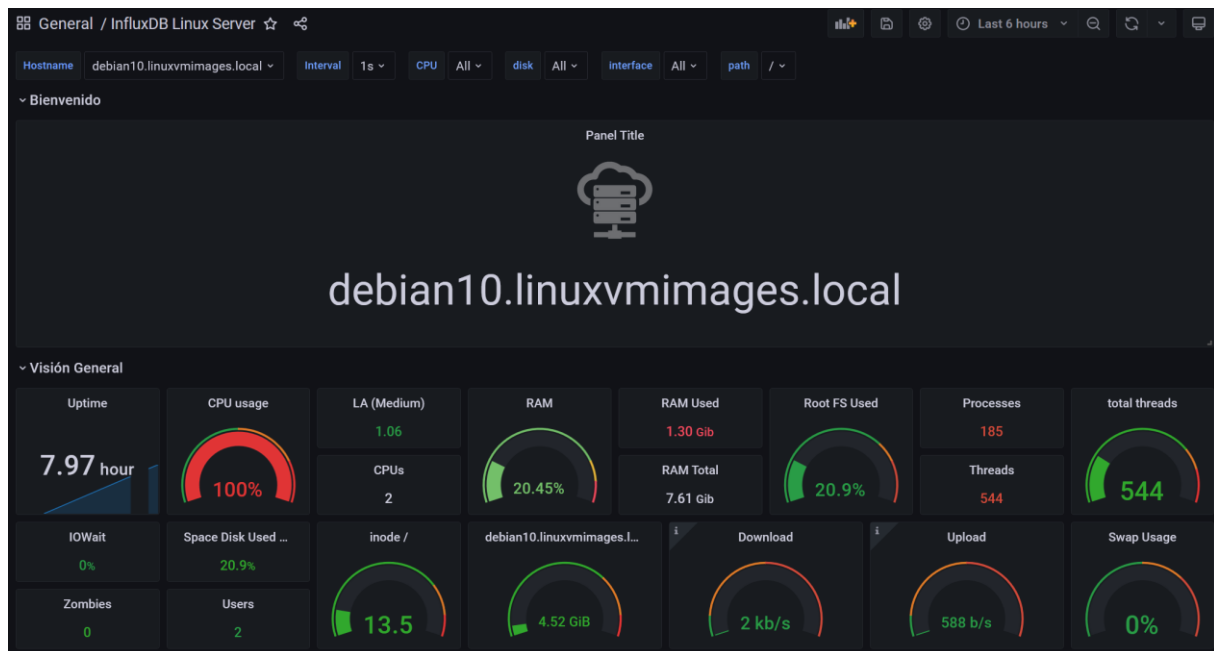
Une fois les options données nous pouvons faire import et nous arrivons sur le Dashboard :



Nous pouvons tester son bon fonctionnement en faisant tourner les CPU de la machine au maximum par exemple :

```
root@debian10:/home/debian# stress -c 2
stress: info: [8665] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
```

Nous regardons dans le Dashboard l'évolution des CPU :



Nous voyons bien les CPU tourner au maximum pour notre machine debian

Pour notre Dashboard Windows, nous avons récupéré celui-ci :

<https://grafana.com/grafana/dashboards/1902>

Il nous faut copier la configuration du fichier telegraf.conf de cette page et la coller dans notre fichier de configuration (/ ! \ **en remplaçant seulement les inputs et non le reste**) :

```
#####  
####  
  
#                               INPUTS  
#  
  
#####  
####  
  
[[inputs.win_perf_counters]]  
  [[inputs.win_perf_counters.object]]  
    # Processor usage, alternative to native, reports on a per core.  
    ObjectName = "Processor"  
    Instances = ["*"]  
    Counters = [  
      "% Idle Time",  
      "% Interrupt Time",  
      "% Privileged Time",
```

```
    "% User Time",  
    "% Processor Time"  
]  
  
Measurement = "win_cpu"  
  
# Set to true to include _Total instance when querying for all (*).  
#IncludeTotal=false  
  
[[inputs.win_perf_counters.object]]  
  
# Disk times and queues  
ObjectName = "LogicalDisk"  
Instances = ["*"]  
Counters = [  
    "% Idle Time",  
    "% Disk Time",  
    "% Disk Read Time",  
    "% Disk Write Time",  
    "% User Time",  
    "% Free Space",  
    "Current Disk Queue Length",  
    "Free Megabytes",  
    "Disk Read Bytes/sec",  
    "Disk Write Bytes/sec"  
]  
  
Measurement = "win_disk"  
  
# Set to true to include _Total instance when querying for all (*).  
#IncludeTotal=false  
  
[[inputs.win_perf_counters.object]]  
  
ObjectName = "System"  
Counters = [  
    "Context Switches/sec",  
    "System Calls/sec",  
    "Processor Queue Length",  
    "Threads",  
    "System Up Time",
```

```
"Processes"

]

Instances = ["-----"]

Measurement = "win_system"

# Set to true to include _Total instance when querying for all (*).
#IncludeTotal=false

[[inputs.win_perf_counters.object]]

# Example query where the Instance portion must be removed to get data
back,

# such as from the Memory object.

ObjectName = "Memory"

Counters = [

    "Available Bytes",

    "Cache Faults/sec",

    "Demand Zero Faults/sec",

    "Page Faults/sec",

    "Pages/sec",

    "Transition Faults/sec",

    "Pool Nonpaged Bytes",

    "Pool Paged Bytes"

]

# Use 6 x - to remove the Instance bit from the query.

Instances = ["-----"]

Measurement = "win_mem"

# Set to true to include _Total instance when querying for all (*).
#IncludeTotal=false

[[inputs.win_perf_counters.object]]

# more counters for the Network Interface Object can be found at
# https://msdn.microsoft.com/en-us/library/ms803962.aspx

ObjectName = "Network Interface"

Counters = [

    "Bytes Received/sec",

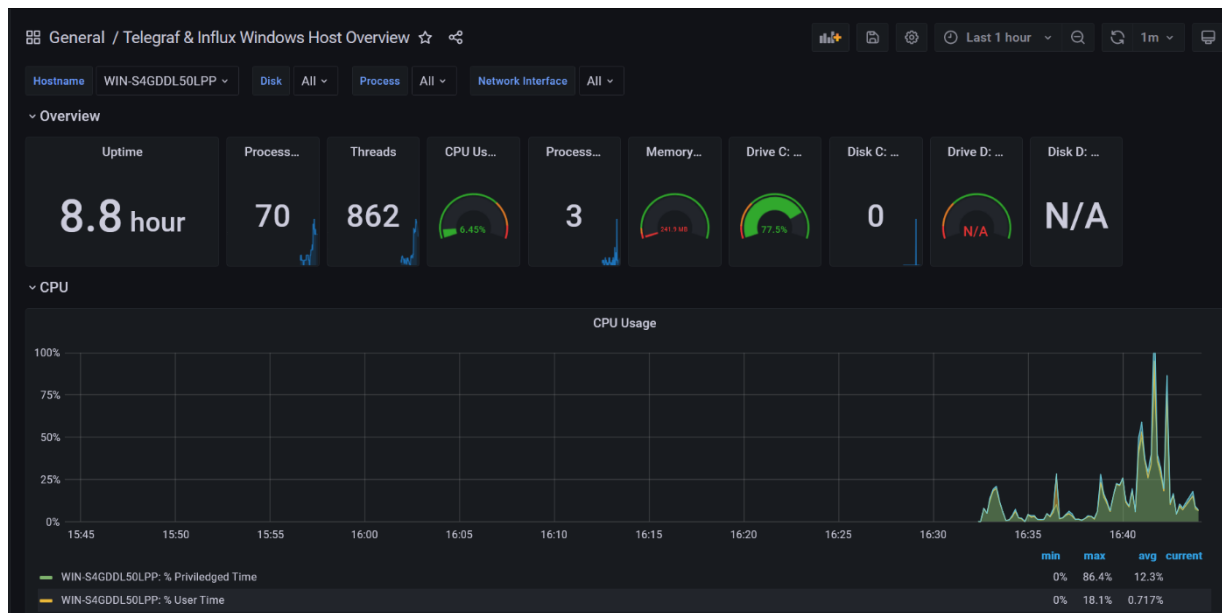
    "Bytes Sent/sec",
```

```
        "Packets Received/sec",  
        "Packets Sent/sec"  
    ]  
    Instances = ["*"] # Use 6 x - to remove the Instance bit from the  
query.  
    Measurement = "win_net"  
    #IncludeTotal=false #Set to true to include _Total instance when  
querying for all (*).  
  
[[inputs.win_perf_counters.object]]  
    # Process metrics  
    ObjectName = "Process"  
    Counters = [  
        "% Processor Time",  
        "Handle Count",  
        "Private Bytes",  
        "Thread Count",  
        "Virtual Bytes",  
        "Working Set"  
    ]  
    Instances = ["*"]  
    Measurement = "win_proc"  
    #IncludeTotal=false #Set to true to include _Total instance when  
querying for all (*).
```

Une fois la configuration mise en place nous devons relancer le service telegraf

La manipulation pour l'import de ce Dashboard est la même que pour le Dashboard du Linux

Nous avons donc un Dashboard pour notre machine Windows :



Dashboard de visualisation d'Apache : <https://grafana.com/grafana/dashboards/9675>

