La recette pour avoir InfluxDB2 en HTTPS

DevOps

Bonjour à tous, après deux heures de recherches pour mettre en place ma base de données InfluxDB dans une instance et pouvoir communiquer de manière sécurisée avec elle, j’ai décidé d’écrire cet article pour aider ceux d’entre vous qui voudraient faire de même en beaucoup moins de temps.

Tout d’abord, qu’est-ce que InfluxDB ? InfluxDB est une base de données orientée *time series*, optimisée pour stocker une série de points, eux-mêmes caractérisés par une valeur et une date. C’est généralement utilisé dans le domaine de la finance pour modéliser l’évolution du prix d’une action au cours du temps, ou alors lorsqu’on a un capteur qui envoie de la donnée en temps réel. Je vous invite à regarder le site de [influxdata](https://www.influxdata.com/) pour plus de précisions.

Pour le reste de cet article, je vais utiliser AWS pour mettre en place mon serveur et mon DNS. Bien évidemment, ceci est parfaitement possible avec n’importe quel autre *cloud provider* ou même *on premise*.

Prérequis

* Avoir un nom de domaine.
* Avoir un compte AWS ou alors un serveur et un DNS à disposition.

Installer InfluxDB sur notre instance EC2

La première chose à faire c’est d’installer influxDB. Pour le coup, rien de plus facile, il suffit d’aller sur [cette page](https://portal.influxdata.com/downloads/), sélectionner la dernière version de InfluxDB, pour ma part v2.0.3, et choisir la méthode d’installation adaptée à votre cas. Pour mon cas, j’ai pris l’AMI de base d’Amazon et j’ai utilisé ces lignes de commandes:

wget https://dl.influxdata.com/influxdb/releases/influxdb2-2.0.3.x86\_64.rpm

sudo yum localinstall influxdb2-2.0.3.x86\_64.rpm

Une fois cela fait, on peut lancer notre base de données. Rien de plus simple, un simple influxd suffit. Si vous avez bien setup votre security group (n’oubliez pas le port 8086) et attribué une public address IP, alors vous pourrez accéder à votre database via: mon-ip:8086 dans mon cas 34.249.22.224:8086.

Tout fonctionne! Malheureusement, nous ne sommes pas arrivés au bout du voyage.

[](https://dylan.doamaral.dev/static/3cf94b4f607493cf6c1ddea47ae19003/4c0ad/influx-http.png)

Le site n’est pas sécurisé et les communications ne sont pas chiffrées. C’est à ce moment-là qu’on regarde la documentation et qu’on tombe sur ceci: <https://docs.influxdata.com/influxdb/v2.0/security/enable-tls/>. C’est exactement ce que l’on recherche, mais voilà cela ne m’a pas convenu et ce pour plusieurs raisons:

* OpenSSL est trop encombrant et dur à utiliser pour un néophyte des certificats SSL.
* Ils nous donnent une solution en lançant la database via une ligne de commande alors que j’aimerais utiliser un service.

Ces deux points m’ont pris beaucoup de temps car j’ai dû fouiller à droite et à gauche pour trouver mon bonheur. Ainsi, voici une possibilité pour avoir son serveur influxDB en HTTPS.

Créer son certificat SSL

Premièrement, il nous faut un certificat SSL (pour plus d’informations, [cet article](https://www.globalsign.com/fr/centre-information-ssl/definition-certificat-ssl) en parle largement). La documentation conseille l’utilisation d’OpenSSL. Je vais personnellement vous conseiller le [Certbot](https://certbot.eff.org/) de [Let’s Encrypt](https://letsencrypt.org/fr/). Malheureusement, Certbot n’est pas installable depuis yum directement. On doit rajouter l’EPEL repository grâce aux commandes suivantes:

cd /tmp

wget -O epel.rpm -nv \

https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

sudo yum install -y ./epel.rpm

Pour ensuite installer Certbot:

sudo yum install python2-certbot-apache.noarch

Avant de lancer Certbot, on va se rendre dans route 53 ou dans n’importe quel autre DNS et on va créer un Alias record qui va mapper le nom de domaine à utiliser par l’instance vers l’ip de notre instance, cela va dans un premier temps permettre à Certbot de s’assurer que vous possédez bel et bien ce nom de domaine (n'oubliez pas d'activer les ports 80 et 443). Une fois cela fait, vous pouvez lancer la commande certbot en root. Suivez les instructions et n’oubliez pas de préciser le nom de domaine que vous voulez attribuer à la database au cours du processus.

Cela va générer un certificat qui se renouvelle automatiquement. On retrouve les différents fichiers qui vont nous servir à configurer influxDB.

Il y a en tout quatre fichiers générés dans le chemin /etc/letsencrypt/live/yourdomain.com/ mais seulement deux fichiers nous intéressent ici, le fichier fullchain.pem et le fichier privkey.pem respectivement l’https-certificate et l’https-private-key.

On ne veut pas forcément utiliser root pour lancer notre database, ainsi le mieux à faire est d’attribuer des permissions à notre user. Nous allons d’abord expliciter le fait que ce certificat appartient à notre user:

sudo chown ec2-user: /etc/letsencrypt/live/yourdomain.com/\*

Puis nous allons changer les permissions en prenant en considération les commandes chmod données par la documentation de InfluxDB.

sudo chmod 644 /etc/letsencrypt/live/yourdomain.com/fullchain.pem

sudo chmod 600 /etc/letsencrypt/live/yourdomain.com/privkey.pem

Une fois cela obtenu, il nous suffit de lancer la commande suivante:

influxd --tls-cert="/etc/letsencrypt/live/yourdomain.com/fullchain.pem" --tls-key="/etc/letsencrypt/live/yourdomain.com/privkey.pem" --http-bind-address=":443"

Et voilà:

[](https://dylan.doamaral.dev/static/20a9cc5b879e1370afba0b9fe0a65c99/5e02b/influx-https.png)

Lancer influx en tant que service

Notre connexion est enfin sécurisée et en prime nous avons un joli nom de domaine au lieu d’une adresse IP qui peut changer et être dure à retenir. J’ai fait le choix d’utiliser le port 443 pour ne pas avoir le :8086 derrière mon nom de domaine. Maintenant, il ne nous reste qu’un problème. On lance la commande manuellement, ainsi que se passe-t-il si l’instance reboot ? Vous l’aurez compris, cela va stopper notre base de données, ainsi nous allons utiliser le service influx. Il suffit de rentrer la commande suivante pour lancer influxdb en tant que service:

sudo service influxdb start

Malheureusement, on n’a plus le port, ni l’https puisque nous passions les arguments via des flags de notre commande influxd. Fort heureusement, il existe une autre manière d'ajouter une configuration pour notre base de données. Pour ce faire, nous allons créer un fichier de configuration.

Pour cela, nous allons setup la localisation de ce fameux fichier via une variable d’environnement:

export INFLUXD\_CONFIG\_PATH=/etc/influxdb

Puis à l'intérieur de ce chemin, nous allons créer un fichier config.toml. Ceci diffère de influxDB post version 2, pour plus d’informations: <https://docs.influxdata.com/influxdb/v2.0/reference/config-options/>.

Il nous suffit juste de retranscrire la même configuration qu’en ligne de commande:

# The SSL certificate to use when HTTPS is enabled.

tls-cert = "/etc/letsencrypt/live/yourdomain.com/fullchain.pem"

# Use a separate private key location.

tls-key = "/etc/letsencrypt/live/yourdomain.com/privkey.pem"

# The port to use.

http-bind-address = ":443"

Le tour est joué, vous pouvez vous amuser à interagir avec ce merveilleux outil en toute sécurité !