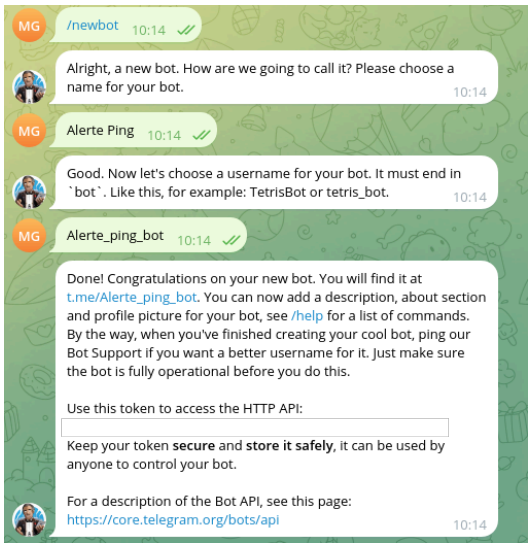
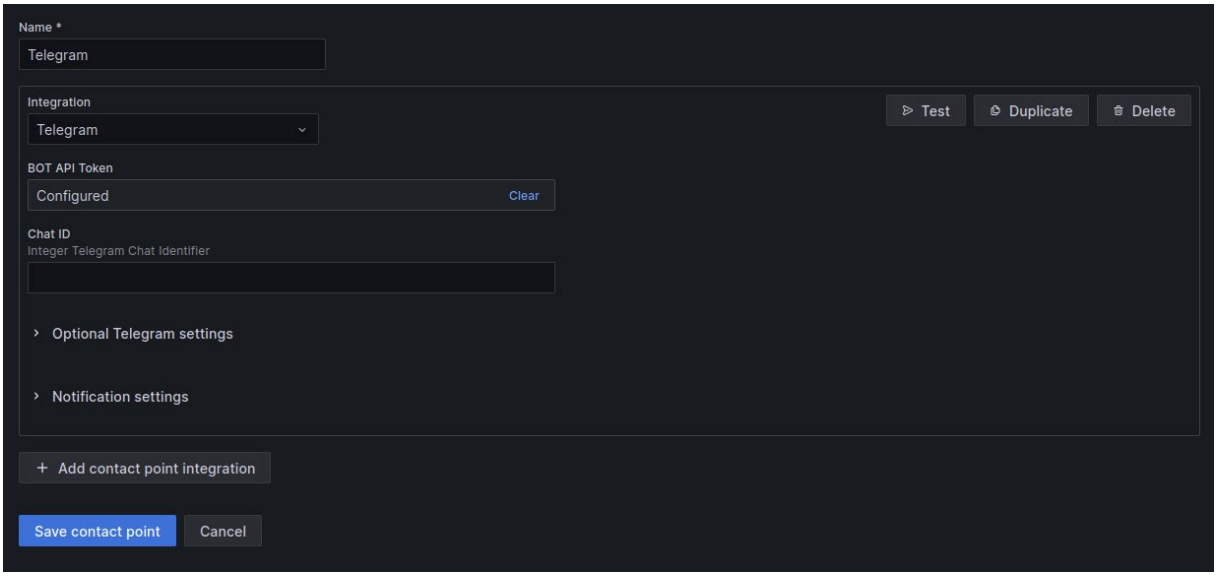
Envoie d'un message quand une ip n'est plus joignable à travers l'API de Telegram

Avant d'utiliser les alertes dans Grafana avec Telegram, il faut se créer un bot dans Telegram en ajoutant @BotFather. Pour créer un bot, il faut envoyer à @BotFather la commande /newbot pour commencer la procédure. Ensuite, @BotFather nous fournit les informations nécessaires pour créer notre bot. Une fois cette étape terminée, on reçoit un message nous indiquant que notre bot est créé, accompagné du token HTTP permettant d'accéder à l'API de Telegram.

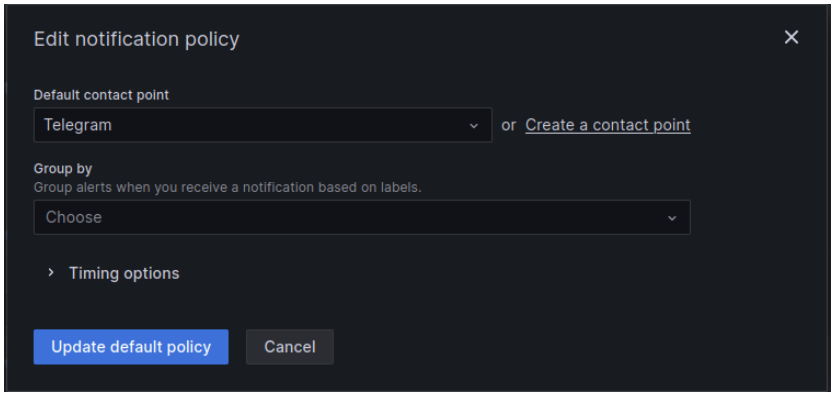


Donc, j'ai ma clé API, mais pour que mon bot envoie un message, il faut créer un nouveau canal ou utiliser un canal existant, puis inviter notre bot à l'intérieur. On utilise un autre bot, @RawDataBot, pour obtenir l'ID de la discussion. Avec ce bot, on sélectionne le type de discussion et on choisit le canal dans lequel est notre bot.

Maintenant, j'ai tous les prérequis pour utiliser l'alerting sur Grafana. Dans le menu de Grafana, on choisit 'Alerting', puis 'Contact Points' pour choisir l'endroit où le message va arriver. On choisit Telegram, on renseigne le token API de son bot ainsi que l'ID de la discussion, puis on teste la connexion.



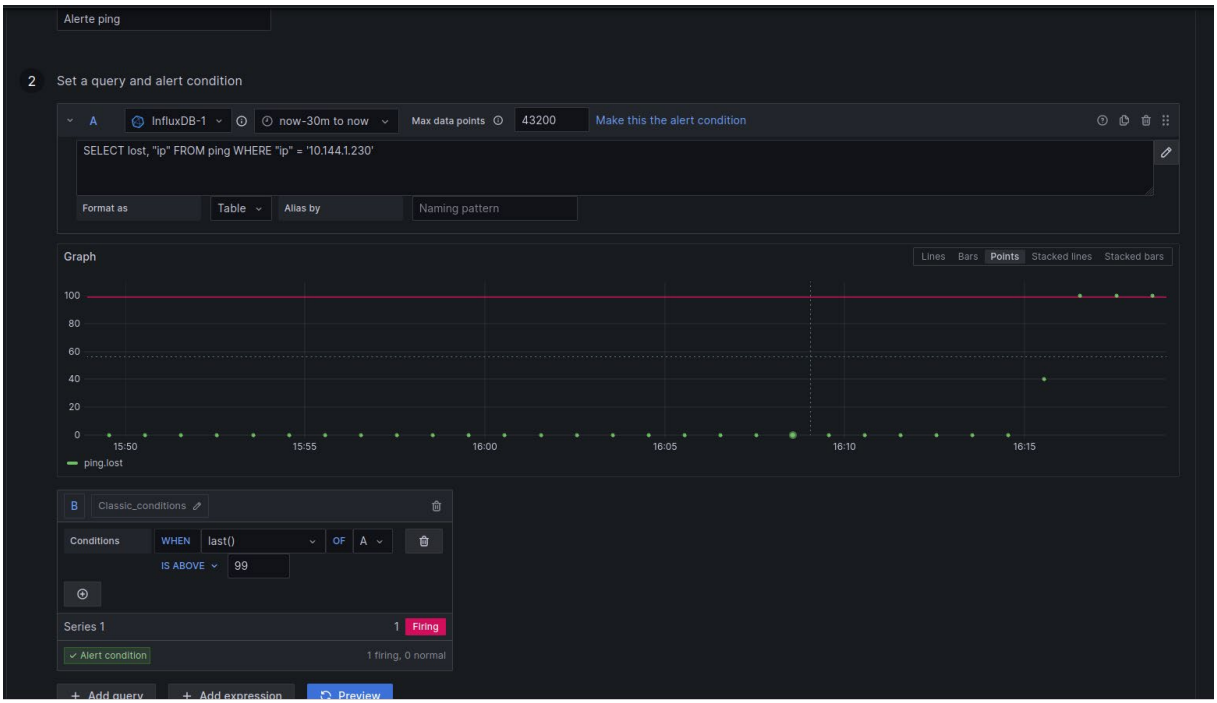
Si on reçoit bien le message de test, on peut sauvegarder le point de contact et aller dans le menu 'Notification Policies' dans 'Alerting', pour définir le point de contact que je viens de créer comme point de contact par défaut.



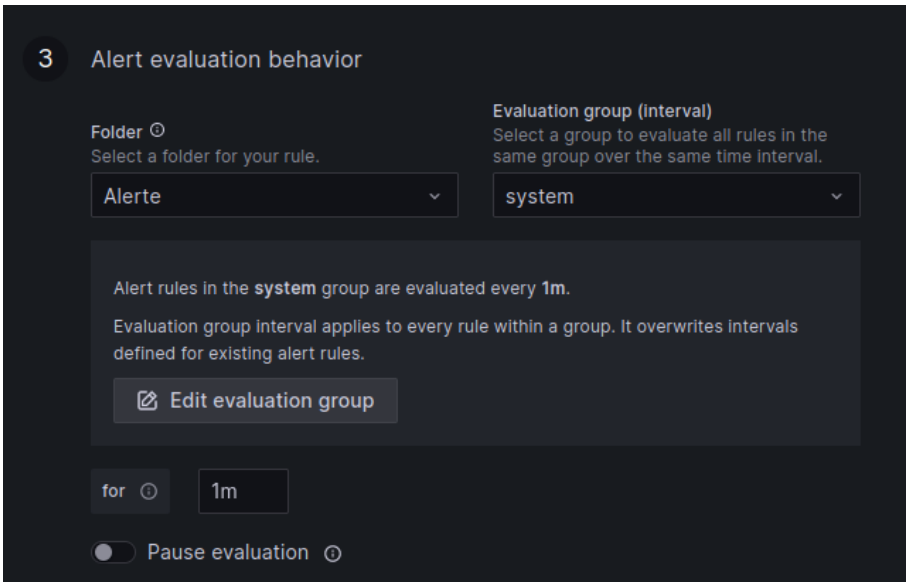
Pour créer la règle d'alerte, il faut aller dans le menu 'Alert Rules' et cliquer sur 'Create Alert Rule'. On sélectionne dans la base de données les valeurs que l'on veut surveiller, puis il faut ajouter des conditions pour déterminer quand déclencher l'alerte. Voici un exemple avec le routeur de test :

Dans un premier temps, on sélectionne le dernier nombre de paquets perdus pour le routeur test.

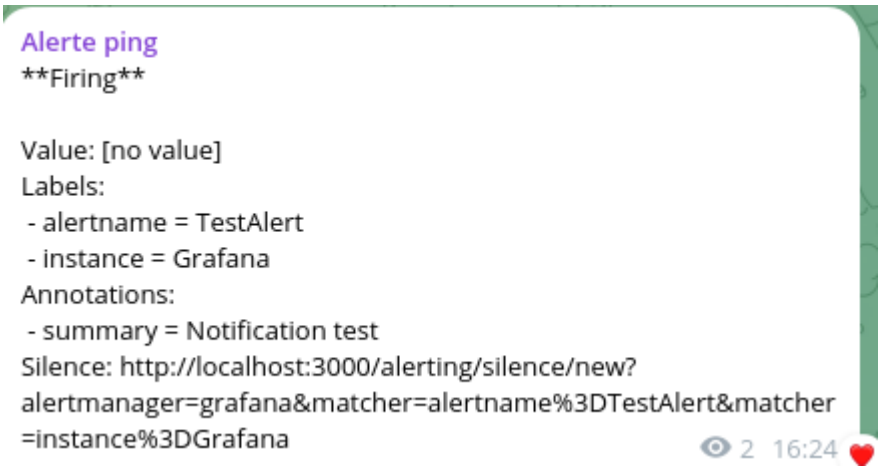
Dans un second temps, on sélectionne le résultat de la requête précédente et on configure l'alerte pour envoyer un message lorsque le nombre de paquets perdus est supérieur à 99.



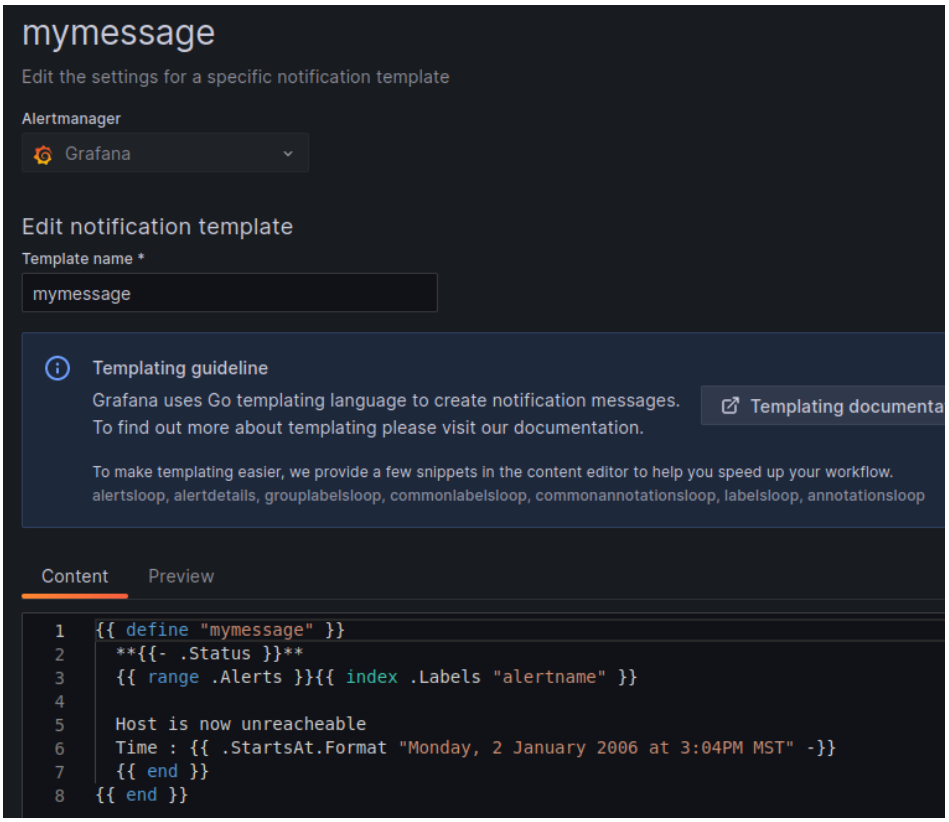
Pour l'alerte, on est obligé d'utiliser un autre dossier que le dossier général et de le placer dans un groupe. Si on ne possède que le dossier général sans groupes, on peut les créer ici. On définit également quand le script va s'exécuter .



Si je débranche le routeur je reçois le message par défaut par défaut qui donne des informations pas utiles dans mon cas :

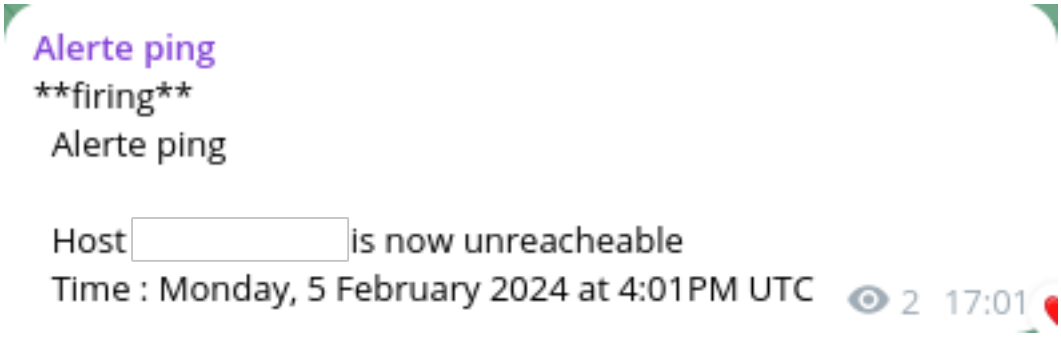


On peut choisir le message à envoyer dans 'Contact Points', puis 'Template' si on veut utiliser des variables, ou dans son point de contact Telegram dans le menu déroulant 'Optional Telegram settings' si on veut simplement mettre une description de l'alerte à afficher dans le message. Dans mon cas, j'utiliserai 'Template'.

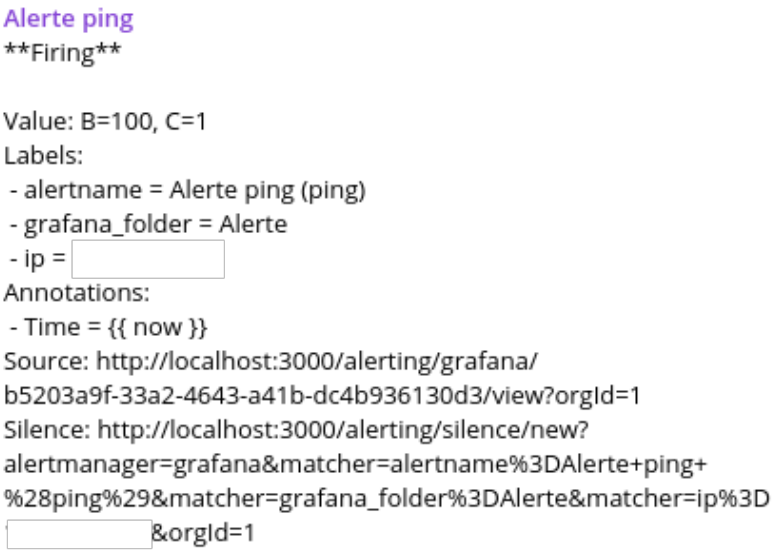


Mon message aura le statut de l'alerte, le nom de l'alerte, un message indiquant que l'hôte est injoignable avec l'adresse IP qui est devenue injoignable, ainsi que la date et l'heure à laquelle cela s'est produit.

Si le format de la date est du 2 janvier 2006 à 15h04, c'est en fait le format de référence dans le langage Go. Le MST permet d'afficher le temps universel coordonné UTC. Cependant, le problème de cette technique est qu'elle ne permet pas d'avoir l'adresse IP dynamiquement.



Pour récupérer l'IP de façon dynamique, on enlève le modèle dans le contact point pour que le message par défaut soit celui défini dans les règles d'alerte. Car il inclut un label avec l'IP. Cette méthode fournit trop d'informations inutiles, et on ne peut pas obtenir l'heure même si désormais on a l'adresse IP de façon dynamique.



L'outil d'alerting de Grafana, dans ce cas d'utilisation, n'est pas le meilleur moyen de savoir si une IP est injoignable. Donc, j'ai utilisé un script Bash pour quand même envoyer un message.

Pour comprendre comment fonctionne l'envoi de messages avec Bash à Telegram, j'ai utilisé un modèle :

https://dev.to/nathannosudo/bash-script-to-send-a-message-in-telegram-from-your-terminal-20c7

Here is a simple Bash script that you can use to send a message in Telegram:

```
# Set the API token and chat ID
API_TOKEN="<your_api_token>"
CHAT_ID="<your_chat_id>"

# Set the message text
MESSAGE="This is a test message"

# Use the curl command to send the message
curl -s -X POST https://api.telegram.org/bot$API_TOKEN/sendMessage
```

Pour la base de mon script, j'ai réutilisé un script que j'avais déjà fait, qui permet de récupérer les adresses IP d'une base de données, de faire un ping et de mettre le résultat dans une base de données InfluxDB. Pour savoir quand une adresse est injoignable, j'ai ajouté un système booléen dans une nouvelle colonne de la base de données. Lorsque le résultat d'un ping est de 100 % de paquets perdus, la valeur booléenne est de 1. Si la valeur de 'lost' est inférieure à 100 %, alors la valeur est de 0. Si une ip passe de KO à UP un message est envoyé et inversement lorsque l'ip passe de UP à KO.

Il était nécessaire de savoir si une adresse est toujours injoignable, pour envoyer un message au bout d'une heure. Donc j'ai modifié le fonctionnement du booléen en modifiant la valeur de bool pour la mettre à 2 quand le message est envoyé et quand l'ip passe de UP à KO. Puis le script récupère l'heure du dernier message envoyé et compare avec l'heure actuel. L'heure dans influxDB est automatiquement enregistrée en unix timestamp au format nanosecondes, il faut s'adapter à se format pour afficher le temps de la machine dans le même format. Pour que le format soit plus compréhensible par une personne j'ai convertie la différence de temps en secondes. Le message sera envoyé quand la différence de temps est supérieure ou égale à 3600 secondes.

En regardant les commandes que faisait le script avec set -x, j'ai vue qu'il y avait un problème avec fping car il voulait effectuer plusieurs fois les pings pour une ip et sortait un message d'erreur comme celui si-dessous. Mais la réponse de fping est quand même renseignée dans le fichier résultat. Donc pour éviter les problèmes dans la boucle et dans la base de données, si la ligne contient « duplicate for » le script passe à la ligne suivante.

<ip> : duplicate for [2], 64 bytes, 18.3 ms

#!/bin/bash

 # Assignez vos valeurs par defaut a ces variables si elles sont vides
min_default="0"
avg_default="0"
max_default="0"

```
# Execute la requete pour obtenir toutes les adresses IP dans un fichier texte
curl -k -s https://app.amanela.com/voganet/api/api.php --data
'k=xcvxvrzeskljmkl&wl=1&q=getResults&p={"1":{"app":"P9",
"obj":"P9_MainManager","id":"","fct":"getActiveObjectOfProductId","param":[{"obj":1,"value":"157"
}],"postfct":""}}' | sed 's/,/\n/g' | sed 's/"//g' | sed 's/}//g' | sed 's/\\t//g' | cut -d' ' -f1 | sed
's/{1:\[//g' | sed 's/{1:\[//g' | sed 's/\]//g' | cut -d':' -f1 > IPa.txt

# Execute fping a partir du fichier IPa.txt pour toutes les adresses IP avec 10 pings chacune, les
resultats sont stockes dans un fichiers
rm resulto
/usr/bin/fping -c 10 -q -f IPa.txt >> resulto 2>&1

# Recupere les informations que l'on souhaite dans le resultat du fping et les inserent dans la base de
donnee
while read -r line; do#
        #Vérifiez si la ligne contient "duplicate for", si oui, passez à la prochaine ligne
    if [[ "$line" == *"duplicate for"* ]]; then
        continue
    fi

    ip=$(echo $line | cut -d' ' -f 1)
    lost=$(echo $line | cut -d' ' -f 5 | cut -d'/' -f 3 | cut -d'%' -f 1 | cut -d'/' -f 3)
    min=$(echo $line | cut -d' ' -f 8 | cut -d'/' -f 1)
    avg=$(echo $line | cut -d' ' -f 8 | cut -d'/' -f 2)
    max=$(echo $line | cut -d' ' -f 8 | cut -d'/' -f 3)

    # Verifiez si les variables sont vides et affectez les valeurs par defaut si necessaire
    min="${min:-$min_default}"
    avg="${avg:-$avg_default}"
    max="${max:-$max_default}"

    # Construise# Ajoutez la colonne booleenne en fonction de la valeur de lost
    if [ $lost -eq 100 ]; then
        bool=1
    else
        bool=0
    fi

    # Interroger la base de donnees pour recuperer la valeur precedente de bool pour une adresse IP
    prev_bool=$(influx -username "$dbuser" -password "$INFLUXDB_PASSWORD" -database
"$dbname" -execute "SELECT last(\"bool\") FROM $dbname WHERE \"ip\" = '$ip'" | awk 'NR > 3 {print
$2}')

    # Envoie un message si l'etat change de UP a KO
    if ((prev_bool == 0  && bool == 1)); then
        MESSAGE_KO="Host $ip est maintenant KO"
```

```
            curl -s -X POST "https://api.telegram.org/bot$API_TOKEN/sendMessage" -d
"chat_id=$CHAT_ID" -d "text=$MESSAGE_KO"
            bool=2
        fi

        # Envoie un message si l'etat change de KO a UP
        if ((prev_bool == 1 || prev_bool == 2)) && ((bool == 0)); then
            MESSAGE_UP="Host $ip est maintenant UP"
            curl -s -X POST "https://api.telegram.org/bot$API_TOKEN/sendMessage" -d
"chat_id=$CHAT_ID" -d "text=$MESSAGE_UP"
        fi

        # Si l'etat reste KO
        if ((prev_bool == 1 && bool == 1)); then
            # Verifier le temps depuis le dernier message KO
            last_ko_time=$(influx -username "$dbuser" -password "$INFLUXDB_PASSWORD" -database
"$dbname" -execute "SELECT last("bool") FROM ping WHERE ip='$ip' AND bool=2" | awk 'NR > 3
{print $1}')
            current_time=$(date +%s%N)
            time_diff=$((current_time - last_ko_time))
            # Convertir la difference de temps en secondes
            time_diff=$((time_diff / 1000000000))

            # Si le temps depuis le dernier message KO est superieur ou egal a 1 heure, envoyer un
nouveau message KO
            if ((time_diff >= 3600)); then
                MESSAGE_DOWN="Host $ip est toujours KO"
                curl -s -X POST "https://api.telegram.org/bot$API_TOKEN/sendMessage" -d
"chat_id=$CHAT_ID" -d "text=$MESSAGE_DOWN"
                bool=2
            fi
        fiz la commande curl avec la date et des conditions pour chaque variable (l espace entre les
variables permet d enregistrer ip en tant que tag et les autres donnees en field)
        curl -i -XPOST "http://localhost:8086/write?db=$dbname" -u
"$dbuser:${INFLUXDB_PASSWORD}" --data-binary "test,ip=$ip
lost=$lost,min_ms=$min,avg_ms=$avg,max_ms=$max,bool=$bool"

done < resulto
```