**CREATION PROXY SYSLOG UBUNTU**

Il s’agit de créer un server ayant 2 cartes réseaux : 1 sur un réseau de machines (dont il faut récupérer les syslog) et 1 vers un server Syslog distant.

Process :

- Rsyslog : logiciel récupérant les syslog des équipements sur le port UDP 6666 et les écrivant sur un fichier buffer sur le disque dur (après avoir mis à jour la date de l’évènement).

- Synchronisation du server via NTP.

- Création d’un scripte python afin de lire le fichier buffer, effacer les syslog trop vieux, tenter d’envoyer les syslog vers le serveur distant sur le port TCP6666 et effacé les log dont l’envoi est un succès.

- Modification du server pour lancer le scripte python toutes les X minutes.

1. **Installation et configuration de Rsyslog :**

* Installation rsyslog :

*Sudo apt update*

*Sudo apt install rsyslog*

* Configuration de rsyslog :

Nous voulons que rsyslog écoute les logs sur le port UDP 6666, modifie la date et l’heure du message (au cas où l’appareil ne serait pas à jour) et l’enregistre ligne par ligne dans un fichier buffer sur le disque dur ( / opt/syslog-relay/buffer-relay.log).

* Edition du fichier de configuration rsyslog :

*sudo nano /etc/rsyslog.d/66-udp-buffer-relay.conf*

* Contenu du fichier :

###############################

# Proxy syslog UDP vers buffer

# - Remplace la date par celle de réception

# - Enlève le timestamp original de l’émetteur

###############################

# Charge le module UDP

module(load="imudp")

input(type="imudp" port="6666" name="udp-buffer")

# Template personnalisée : format BSD classique

template(name="CleanBufferFormat" type="string"

string="<%pri%>%timegenerated:::date-bsd% %hostname% %app-name%: %msg%\n")

# Application : uniquement pour les messages venant du port 6666 UDP

if $inputname == 'udp-buffer' then {

action(type="omfile"

file="/opt/syslog-relay/buffer-relay.log"

template="CleanBufferFormat")

stop

}

* Redémarrage sur service rsyslog :

*sudo systemctl restart rsyslog*

* Creation du dossier accueillant les log et autorisations :

*sudo mkdir -p /opt/syslog-relay*

*(creation du dossier)*

*Sudo nano /opt/syslog-relay/buffer-relay.log*

*(creation du fichier buffer vierge pour modification des propriétés)*

*sudo chown syslog:syslog /opt/syslog-relay /opt/syslog-relay/buffer-relay.log*

*(Définit que syslog est propriétaire et groupe pour le dossier et le fichier)*

*sudo chmod 775 /opt/syslog-relay*

*(Donne les droits lecture/écriture/exécution au propriétaire et au groupe et lecture, écriture aux autres)*

*sudo chmod 664 /opt/syslog-relay/buffer-relay.log*

*(Donne les droits lecture et écriture au propriétaire, et lecture aux autres)*

* Vérifier l’état du buffer :

*tail -f /opt/syslog-relay/buffer-relay.log*

1. **Copie et configuration du scripte python**

* Creation server tftp sur le pc windows ayant les fichiers python :

Installation du logiciel TFTP64 et pointer du dossier de partage des fichiers à transférer.

* Récupération des fichiers sur le server :

*sudo apt install tftp*

*sudo chown $USER:$USER /opt/syslog-relay/*

*cd /opt/syslog-relay/*

*tftp <IP du server tftp windows>*

*tftp> binary*

*tftp> get syslog-relay.py*

*tftp> get syslog-relay-config.txt*

*tftp> quit*

* Modification des paramètres du scripte :

Il suffit d’éditer le fichier « syslog-relay-config.txt » avec nano pour modifier l’adresse IP du server syslog distant, le temps avant que les logs soient considérés comme périmés.

* Rendre le scripte appelable de n’importe où sur le server :

Il faut mettre un lien dans le dossier /usr/local/bin/

*sudo ln -s /opt/syslog-relay/syslog-relay.py /usr/local/bin/syslog-relay*

* Verifier si des logs sont présents dans le buffer :

*tail -f /opt/syslog-relay/buffer-relay.log*

On peut arrêter cet affichage avec Ctrl + C

* Tester le scripte temporairement :

*sudo python3 /opt/syslog-relay/syslog-relay.py*

* Automatiser le lancement du scripte toutes les 5 minutes dès allumage sur server :

Il faut ajouter une ligne dans le scripte d’automatisation crontab :

*sudo crontab -e*

Puis ajouter une ligne à la fin du contenu :

*\*/2 \* \* \* \* /usr/bin/python3 /opt/syslog-relay/syslog-relay.py >> /var/log/syslog-relay-cron.log 2>&1*

Enregistrer et quitter.

1. **Synchroniser le server Proxy syslog sur un server NTP local** :

* Installation de l’application chrony

sudo apt update && sudo apt install chrony

* Edition du fichier de config de chrony :

sudo nano /etc/chrony/chrony.conf

Contenu exemple avec les servers NTP locaux 10.1.1.253 :

*# chrony.conf — configuration de Chrony avec priorité au serveur local NTP*

*# Inclure les fichiers de configuration supplémentaires (conservé par défaut)*

*confdir /etc/chrony/conf.d*

*# Serveur NTP local — utilisé en priorité*

*server 10.10.1.91 iburst prefer*

*# Désactiver l'ajout dynamique de sources via DHCP ou fichiers*

*# sourcedir /run/chrony-dhcp*

*# sourcedir /etc/chrony/sources.d*

*# Fichier de clés d'authentification NTP (optionnel ici)*

*keyfile /etc/chrony/chrony.keys*

*# Fichier de dérive pour ajustement du taux d'horloge*

*driftfile /var/lib/chrony/chrony.drift*

*# Répertoire de sauvegarde pour les clés NTS*

*ntsdumpdir /var/lib/chrony*

*# Activer le logging si nécessaire*

*# log tracking measurements statistics*

* Redémarrage de chrony:

*sudo systemctl restart chrony*

* Vérification de la synchro:

*chronyc tracking*

*chronyc sources*