```
#################################################################
#                                                               #
#     Créateur : Arthur YANG - Responsable Documentation  #
#                         & Administratif       #

#     Date de création : 19/02/2025                         #
#                                                               #
#     Dernier modificateur : Arthur YANG            #
#     Date de modification : 19/02/2025             #
#                                                               #
#           Version actuelle : 1.0                  #
#                                                               #
#################################################################


----------------------- Configuration et sécurisation du serveur web prod
-----------------------

Sources : https://artheodoc.wordpress.com/wp-content/uploads/2022/01/1-
serveur_web_apache_debian_11_pour_wordpress_nom_de_domaine_-https.pdf
           https://les-enovateurs.com/debian-vps-apache-mysql-php-
wordpress


Actions effectuées dans la VM :

   => Connexion via root / mot de passe
   => apt update
   => apt upgrade


Installation de l'agent promtail (en root) :

      Installation de Promtail
             => Téléchargement et installation :
            On se place dans le chemin : /usr/local/bin/
            On télécharge le fichier zip de Promtail : wget
https://github.com/grafana/loki/releases/download/v3.4.2/promtail-linux-
amd64.zip
            Nous pouvons ensuite l'unzip : unzip (nom du fichier)
            C'est un fichier executable que l'on a.

      => Configuration de Promtail :

      Nous devons ici, avoir un fichier nommé "promtail-config.yaml" pour
la configuration de Promtail.
      Nous pouvons télécharger un modèle pré-existant : wget
https://github.com/grafana/loki/blob/main/clients/cmd/promtail/promtail-
local-config.yaml ou le créer nous même.
      Cela télécharge un modèle que l'on peut renommer comme nous le
voulions.


=> Config .yaml personnalisé que nous effectuons :
```

```
-------------------------------------------------------------------------
-------------------

server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /var/lib/promtail/positions.yaml

clients:
  - url: http://192.168.7.133:3100/loki/api/v1/push  # Utilisation de
l'IP du serveur Loki

scrape_configs:
  - job_name: SRV-WEB-AARD-SYSLOG
    static_configs:
      - targets:
          - localhost
        labels:
          job: SYSLOG-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/syslog

  - job_name: SRV-WEB-AARD-MAIL
    static_configs:
      - targets:
          - localhost
        labels:
          job: MAIL-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/mail.log

  - job_name: SRV-WEB-AARD-CRON
    static_configs:
      - targets:
          - localhost
        labels:
          job: CRON-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/cron.log

  - job_name: SRV-WEB-AARD-AUTHLOG
    static_configs:
      - targets:
          - localhost
        labels:
          job: AUTHLOG-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
```

```
          __path__: /var/log/auth.log

  - job_name: SRV-WEB-AARD-APACHE2_LOGS
    static_configs:
      - targets:
          - localhost
        labels:
          job: APACHE2_LOGS-SRV-WEB-AARD
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/apache2/*.log
```

=> Créer un dossier /var/lib/promtail/ : mkdir -p /var/lib/promtail
=> chown -R web:web /var/lib/promtail
=> chmod -R 750 /var/lib/promtail

=> Nous mettons l'utilisateur créée plus tôt "web" en tant que propriétaire et propriétaire groupe des fichiers de config de Loki.

=> chown -R web:web promtail-config.yaml promtail-linux-amd64
=> chmod 755 promtail-linux-amd64
=> chmod 640 promtail-config.yaml

=> Création du fichier : /etc/systemd/system/promtail.service

---

```
[Unit]
Description=Promtail Loki
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=web
Group=web
ExecStart=/usr/local/bin/promtail-linux-amd64 -
config.file=/usr/local/bin/promtail-config.yaml

SyslogIdentifier=web
Restart=always

[Install]
WantedBy=multi-user.target
```

Commandes système et visualisation des logs en direct :

```
=> systemctl enable promtail.service
=> systemctl start promtail.service
=> systemctl status promtail.service
=> journalctl -f -u promtail.service
```

Normalement le service promtail est lancé.

Dans nos conteneurs LXC Debian, les logs semblent être dans le "journald" et non le syslog.

On installe "rsyslog" pour qu'ils puissent etre visible dans "syslog"

```
=> apt install rsyslog -y
=> systemctl enable rsyslog --now
=> tail -f /var/log/syslog
```

Pour que notre utilisateur "web" puissent avoir les droits de lecture de logs, on va lui ajouter dans le groupe adm :

```
=> usermod -aG adm web
=> systemctl restart promtail.service
```

En effet, Un problèmes d'accès aux fichiers /var/log/auth.log, /var/log/cron.log et /var/log/user.log par Grafana sera présent par le fait que l'on a paramétré le service promtail avec l'utilisateur "web".

Faire de même avec le fichier "syslog" :

```
=> chown web:root /var/log/syslog
=> chmod 640 /var/log/syslog
=> systemctl restart promtail.service
```

On peut rajouter :

```
=> chown web:utmp /var/log/btmp
```

De manière générale, regarder les permissions des fichiers selon l'utilisateur qui utilise le service.

Configurer les logs pour qu'ils soient visibles dans GRAFANA :

```
=> Les logs d'apache se trouvent dans le répertoire : /var/log/apache2
=> On donne les droits pour pouvoir les récupérer et envoyer vers LOKI et GRAFANA
=> chown -R web:adm /var/log/apache2
=> chmod -R 750 /var/log/apache2
```

=> Le répertoire et tout les fichiers .log donne le droit à promtail donc de les récuperer.

Nous devons maintenant configurer le fichier promtail-config.yaml :

```
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /var/lib/promtail/positions.yaml

clients:
  - url: http://192.168.7.133:3100/loki/api/v1/push  # Utilisation de l'IP du serveur Loki

scrape_configs:
  - job_name: SRV-WEB-AARD-SYSLOG
    static_configs:
      - targets:
          - localhost
        labels:
          job: SYSLOG-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/syslog

  - job_name: SRV-WEB-AARD-MAIL
    static_configs:
      - targets:
          - localhost
        labels:
          job: MAIL-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/mail.log

  - job_name: SRV-WEB-AARD-CRON
    static_configs:
      - targets:
          - localhost
        labels:
          job: CRON-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/cron.log

  - job_name: SRV-WEB-AARD-AUTHLOG
    static_configs:
      - targets:
          - localhost
```

```
        labels:
          job: AUTHLOG-AARD-WEB
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/auth.log

  - job_name: SRV-WEB-AARD-APACHE2_LOGS
    static_configs:
      - targets:
          - localhost
        labels:
          job: APACHE2_LOGS-SRV-WEB-AARD
          host: SRV-WEB-AARD
          stream: stdout
          __path__: /var/log/apache2/*.log
```

Nous pouvons recharger promtail puis s'apercevoir la réception des logs dans GRAFANA.

Maintenant nous voulons aussi les logs mais de Wordpress.
Pour cela, nous installons le plugin directement sur le wordpress admin, intitulé "WP Activity Log", l'activons et suivons le tuto d'install.
Nous modifions ce que nous avons mis comme MAIL a la base pour l'admin du WP. wizardsndice@outlook.fr => svg.wizardsndice@gmail.com

Une fois mis en place, on a un menu pour les logs sur wordpress.
Pour compléter le monitoring des logs de WP, il faut dans le menu du plugin installé, dans 'Réglages', puis 'Modif de fichiers', installé Melapress.

Pour la config de melapress, la freq. d'analyse est hebdomadaire et à 2h00 le lundi. La config de base déja coché est laissé et effectué. La purge des evenements se fera tout les 10 scans.
La config se termine pour MelaPress. Les changements de paramètres pour Melapress est a gauche dans le menu "File Monitoring".

Des notifs via emails peuvent être envoyés pour des rapport des ce logs par exemple. Nous autorisons sur le FW l'envoi de mail et nous pouvons normalement recevoir sur les boites mails. Nous enverrons les mails sur la boite 'svg.wizardsndice@gmail.com' donc nous avons paramétré les plugins afin qu'ils envoient sur notre boite mail SVG.

Pour l'envoi des mails, nous installons un plugin / client smtp sur wordpress. 'WP Mail SMTP' est le nom du plugin.
Nous ne suivons pas l'assistant de config : on choisit d'aller dans les réglages de WP mail SMTP, et de choisir dans les services d'envoi, l'option "Autre SMTP"
Nous pouvons ainsi ici configurer notre compte GMAIL "SVG" pour pouvoir envoyer des mails depuis wordpress.

La config est simple :

=> Hébergeur SMTP : smtp.gmail.com
=> Cryptage : TLS
=> Port : 587
=> Authentification activé
=> ID SMTP : svg.wizardsndice@gmail.com
=> Mdp smtp : Le mot de passe d'application créer dans les paramètres de la boite mail SVG

Sauvegarder ensuite, puis nous pouvons tester l'envoi de mail depuis les plugins installés, tout fonctionne normalement.
Les plugins étant gratuit, ne donnent pas vraiment d'informations sur les logs dans le rapport, néanmoins ils pourraient être utile et plus largement détaillés avec une version payante.

Nous configurons aussi des filtres et étiquettes pour la réception de ces mails sur la boite GMAIL SVG. Selon si c'est de DRYN ou AARD en fonction de l'ip source pour le moment.

Nous avons donc maintenant pour WordPress le monitoring des logs sur WordPress directement. Il faudrait faire des configurations plus "avancées" pour pouvoir avoir ces logs sur GRAFANA, mais demande plus d'investigation. Néanmoins les logs d'apache sont bien dispo sur GRAFANA.

Après cette phase de config des logs / monitoring, on peut passer au configs suivantes.

Nous allons ajouter la possibilité de pouvoir se connecter avec des identifiant ldap sur le wordpress.
Pour cela, nous installons le plugin "Active Directory / LDAP Intergration".

Il faut aussi ajouter l'extension PHP LDAP.

=> Étape 1
Pour Ubuntu/Debian, la commande d'installation serait sudo apt-get -y install php-Idap.
Recherchez extension=php_ldap.so dans le fichier /etc/php/8.2/apache2/php.ini. Décommentez cette ligne, si elle n'est pas présente, ajoutez cette ligne dans le fichier et enregistrez le fichier
Dans notre cas, on ajoute cette ligne.

Étape 3
Redémarrez le serveur. Ensuite, actualisez la page de configuration du plugin LDAP/AD

On peut passer ensuite a la config :

=> On ouvre les ports 389 du SRV DC vers le serveur WEBPREPROD

=> On créer un compte LDAP pour la liaison : wordpress.wp (MDP sur bitwarden)
=> Service Account Username : wordpress.wp@wnd.local
=> Teste la liaison : OK
=> Search Base : dc=wnd,dc=local
=> SAMAccountName = wnd.local\user
=> On peut tester avec nos user : OK donc config de base FINI

Après on peut personnaliser les rôles etc ...
=> faire du role mapping (Le mappage de rôles permet d'attribuer un rôle WordPress en fonction du groupe LDAP auquel appartient un utilisateur sur le serveur LDAP).
=> Néanmoins, comme nous avons la version gratuite, nous pourrons pas le faire.
=> Donc par défaut nos comptes LDAP sont des abonnées du WP. Nous pourrons modifier nos rôles a nous une fois connecté une fois sur WP pour etre admin.

Dans le menu "Attribute Mapping",

=> Infos sur cette partie : https://faq.miniorange.com/knowledgebase/configure-attribute-mapping-in-ldap/
=> nous laissons "username@email_domain" dans la premiere case, rien dans la 2eme et les autres rien car abonnement gratuit

Ensuite dans le dernier menu, nous autorisons tous ce que nous pouvons jusqu'aux options au abonnés.

Nous pouvons tester le ldap sur la page d'accueil de connexion => Ca fonctionne
Mais nous sommes juste lecteur, donc avec le compte "superadmin", nous pouvons modifier le role et mettre admin dans le menu des comptes.
Il faut obligatoirment mettre un mail aux users, j'ai inventé pour le moment.

Voila pour le LDAP intégré sur Wordpress.

Nous pouvons suivre les recommendés de WP dans "l'état de santé du site" dans le menu "tableau de bord" qui permet de savoir au niveau conf perf,sécu ce que nous pouvons faire.
Au niveau des conf performances, nous mettre en place les modules PHP importants qui ne sont pas forcément présent mais qui servent pour faire tourner le site.

Les modules nécessaires sont :
=> imagick
=> intl

Nous installons donc ces modules sur le serveur web :
=> apt install php-imagick -y
=> apt install php-intl -y

Puis on restart le service apache2.
Les modules sont installés et serviront pour plus tard.

Il faut aussi voir la mise en cache des pages web/wordpress.
(Elles permettent l'upgrade de la vitesse et perf du site en servant des pages statiques au client au lieu de tout le temps générer dynamiquement à chaque visite)

Pour cela, nous installons un plugin : "W3 Super Cache"
Après l'installation, il va falloir autoriser plus de droits sur le serveur WEB et notamment le fichier wp-config.php (Actuellement en 400)

=> chmod 600 /var/www/html/wp-config.php
=> chmod -R 777 /var/www/html/wp-content
Attention : Ce 2ème réglage est temporaire. Une fois WP Super Cache configuré, il faudra remettre des permissions plus sécurisées.

Puis restart du service apache2, rafraichir la page WP sur le web.
Maintenant on repasse à des droits plus restrictifs :

=> chmod -R 755 /var/www/html/wp-content

Apres restart encore du service apache2 et page admin web de WP
On voit que la page réglage est enlevé au final apres nos parametrages.
L'installation est ainsi finit.
Maintenant il faut continuer à configurer ce plugin. Mais nous le ferons dans un autre temps.

Voila pour les reco de WP mais il pourrait en avoir d'autres bien sur.

Pour mettre en place Woocommerce, il nous faudrais les prérequis suivant :

=> Un nom de domaine (Pas encore obtenue / wizardsndice.fr)
=> Hébergement Web (obtenue)
=> Téléchargement de WordPress (obtenue)
=> Installation d'un certificat SSL (obtenue, c'est le reverse proxy qui s'occupe du certificat SSL)

Donc comme nous n'avons pas tout sous la main pour l'instant, nous installons et configurons au maximum WooCommerce pour le moment. Cela fonctionne quand meme, meme sans le nom de domaine. c'est juste que la config à 100% ne sera pas faite.

Pour cela, nous installons le plugin Woocommerce sur WP :
=> Puis nous commençons l'installation avec l'assistance
=> Nous mettons l'adresse wizardsndice@outlook.com
=> Vêtements et accessoires
=> On laisse cocher les fonctionnalités sauf pour l'IA

Maintenant c'est que de la config qu'il faut faire, par exemple :

=> Il y a parfois des configs vraiment obligatoire, par exemple le fait que le site soit juste privé, il faut configuré le module jetpack de WooCommerce etc …

=> On supprime le plugin MailPoet qui pour l'instant ne servira dans les extensions

Nous allons juste faire en sorte que des gens peuvent s'authentifier / créer des comptes.

=> Il faut juste dans les paramètres/réglages de WooCommerce => puis le menu 'comptes et condidentialité' , on peut activer la création de compte.

=> Néanmoins, la création fonctionne mais lorsque nous nous déconnectons de l'utilisateur et que nous voulons nous reconnecter avec, celui-ci ne fonctionne plus alors qu'il est toujours bel et bien présent dans la BDD. Je ne sais pas l'erreur actuelle mais elle est peut être lié au fait que n'avons pas fini à l'heure actuelle, de configurer WooCommerce ou Wordpress de manière générale.

Mais voila, a partir de la et comme tout a l'heure, il faut juste continuer à configurer WooCommerce. Donc on a un déploiement et une configuration vraiment légère.

=> Il faudrait voir plus tard comment enlever la page d'authentification de Wordpress d'origine pour éviter que client accède dessus. (surement enlever le wp-admin.php)

Avant de passer aux fichiers de configs apache ou php sur le serveur directement, nous devrons installer un autre plugin qui permettra la redirection de la page "wp-login.php" vers la page web classique, pour permettre l'authentification dessus et pas depuis la page "wp-login.php"

De même pour la page "wp-admin" qui les redirige autre part si ils n'ont pas le droit ou veulent y acceder directement.

Donc cette partie qui serait plus dans la partie sécu, sera à pensé lorsque tout sera mis en place sur les 2 sites.

On repasse maintenant coté serveur pour faire un tour de la configuration APACHE/PHP :

=> Vérification des modules Apache activés

=> a2enmod rewrite headers expires

=> systemctl restart apache2

- rewrite : Nécessaire pour les permaliens WordPress.
- headers : Utile pour la sécurité et les réglages de cache.
- expires : Aide à la gestion du cache des ressources statiques.

Après cela, on va faire en sorte d'utiliser un virtualhost différent de celui-par défaut

```
      Celui qu'on utilise est le 000...
            => on peut le voir avec la commande : ls -l
/etc/apache2/sites-enabled/

Il sera pas si différent mais au moins on aura un "personnalisé"
      on copie le virtualhost de base vers un autre fichier :
            => cp /etc/apache2/sites-available/000-default.conf
/etc/apache2/sites-available/wizardsndice.conf

Maintenant on peut le personnaliser "wizardsndice.conf" :

--------------------------------------------------------------------------
-----

<VirtualHost *:80>
    ServerName wizardsndice.fr
    ServerAlias 172.16.7.1 wizardsndice.wnd.local
    ServerAdmin svg.wizardsndice@gmail.com
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/html>
        AllowOverride All
        Require all granted
    </Directory>

    # Empêcher l’exécution de scripts PHP dans /uploads et /wp-includes
    <Directory "/var/www/html/wp-content/uploads">
        Require all denied
    </Directory>

    <Directory "/var/www/html/wp-includes">
        Require all denied
    </Directory>

    # Bloquer l’accès aux fichiers sensibles
    <FilesMatch "(wp-config.php|xmlrpc.php|.htaccess|.htpasswd)">
        Require all denied
    </FilesMatch>

    # Empêcher l'affichage des fichiers d'un dossier (évite de voir la
liste des fichiers)
    Options -Indexes

    # Headers utiles pour la sécurité et la compatibilité avec le reverse
proxy
      Header set X-Forwarded-Proto "https"
      Header set X-Frame-Options "SAMEORIGIN"
      Header set X-XSS-Protection "1; mode=block"
      Header set X-Content-Type-Options "nosniff"
      Header set Content-Security-Policy "default-src 'self'"
```

```
</VirtualHost>
```

Ensuite on regarde la config de "/var/www/html/.htaccess" si elle est bien configuré :

```
      => nano /var/www/html/.htaccess
```

Il faut qu'elle soit comme ceci :

```
# BEGIN WordPress
# Les directives (lignes) entre     BEGIN WordPress     et     END
WordPress     sont g  n  r  es
# dynamiquement, et doivent   tre modifi  es uniquement via les filtres
WordPress.
# Toute modification des directives situ  es entre ces marqueurs sera
surcharg  e.
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
</IfModule>

# END WordPress
```

Puis, on s'assure que le fichier a les bonnes permissions :

```
      => sudo chown www-data:www-data /var/www/html/.htaccess
      => sudo chmod 644 /var/www/html/.htaccess
```

Enfin on restart le service apache et charge le nouvelle hote virtuel "wizardsndice.conf" :

```
      => a2ensite wizardsndice.conf (se mettre dans le bon répertoire)
      => systemctl restart apache2
      => a2dissite 000-default.conf
```

On peut aussi vérif quels sites sont activés avec : => apachectl -S

Normalement cela utilise maintenant notre config créer plus tot sur le site web.

Autre configurations que l'on va faire meme si on pourrait le mettre dans la partie "Sécu" :

=> Cacher les informations de version d'Apache et Debian sur les pages d'erreur
=> Éditer le fichier /etc/apache2/conf-available/security.conf :
=> Ajoute ou modifie ces lignes :
=> ServerTokens Prod
=> ServerSignature Off
=> on restart apache

Pour finir la partie configuration (qui pourrait être encore plus longue), on personnaliser le fichier /etc/apache2/apache2.conf :

=> Timeout 30
=> Ligne à ajouter :
=> <IfModule mod_deflate.c>
AddOutputFilterByType DEFLATE text/plain text/css text/javascript application/javascript text/xml application/xml application/json
</IfModule>
=> Modif de la ligne : <FilesMatch "^\.ht"> PAR <FilesMatch "(^\.ht|\.git|\.env|composer.json|composer.lock)">

=> Puis dans cette partie :

```
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>
```

AVEC

```
<Directory /var/www/>
    Options -Indexes +FollowSymLinks
    AllowOverride All
    Require all granted
</Directory>
```

=> La suite dans ce fichier de conf pourra se faire dans la partie "Sécu" avec les .htaccess. etc ... et plus si besoin.

Voila pour la partie configuration. Comme dit plus tot, il pourrait y avoir encore plein de configuration possible, mais ceci demanderait plus de temps et de reflexion.

On va installer pour la fin, NODE EXPORTER pour récupérer des informations plus précises que avec SNMP, qui surveillera les serveurs directs et non les équipements réseaux.

=> Dans /usr/local/bin/ , on télécharge NODE Exporter qui va nous servir à récupérer les métriques du serveur web ici.

```
=> wget https://github.com/prometheus/node_exporter/releases/download/v1.9.0/node_exporter-1.9.0.linux-amd64.tar.gz
=> tar -xvzf node_exporter-1.9.0.linux-amd64.tar.gz
=> rm node_exporter-1.9.0.linux-amd64.tar.gz
=> mv /usr/local/bin/node_exporter-1.9.0.linux-amd64/node_exporter /usr/local/bin/
=> chown -R web:web /usr/local/bin/node_exporter
=> chmod 755 /usr/local/bin/node_exporter
=> rm -r /usr/local/bin/node_exporter-1.9.0.linux-amd64/
```

=> Création du fichier : nano /etc/systemd/system/node_exporter.service

---

```
[Unit]
Description=Prometheus NODE Exporter
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=web
Group=web
ExecStart=/usr/local/bin/node_exporter \
  --web.listen-address=0.0.0.0:XXXX \
  --collector.systemd \
  --collector.processes

SyslogIdentifier=web
Restart=always

[Install]
WantedBy=multi-user.target
```

On a changé ici le port d'écoute de SNMP EXPORTER de Prometheus ici, dans le fichier du service. Ca ne se fait pas dans le .yml car il n'y pas de fichier .yml pour NODE Exporter .

Commandes système et visualisation des logs en direct :

```
=> systemctl daemon-reexec
=> systemctl daemon-reload
=> systemctl enable node_exporter.service
=> systemctl start node_exporter.service
=> systemctl status node_exporter.service
```

=> journalctl -f -u node_exporter.service

Voila pour l'installation de NODE EXPORTER.

UPDATE :

=> On a remis le 000-defaut.conf comme hote virtuel pour le sites-enable.
=> Le fichier conf "wizardsndice.conf" est pas bon car il n'affiche rien lorsque on créer les pages on modifie des pages etc...

De plus, on quasiment tout uninstall les plugins sur WP, donc il faudra les remettre de nouveau.

(partie sécu (par le site, le plugin pour le cache, autres), mettre plus tard wordpress derrière un rproxy)

=> Ajout du MOTD personnalisé sur la VM (et toutes) du NAS.
/etc/motd

Wizards & Dice | Shell Access for Debian GNU/Linux

****
****
*
LEGAL NOTICE
* This system is for authorized use only. All activities on this system may be * monitored and recorded. Unauthorized access or use is strictly prohibited * and may result in disciplinary action, criminal prosecution, or both. By * continuing to use and access this system, you consent to such monitoring.*

****
****

=> Ajout des éléments pour le monitoring des machines et de leurs MAJS.

=> wget https://raw.githubusercontent.com/labmonkeys-space/apt-prometheus/main/script/apt-metrics.sh -O /usr/local/bin/apt-metrics.sh
=> chmod 750 /usr/local/bin/apt-metrics.sh
=> chown web /usr/local/bin/apt-metrics.sh
=> Puis on change tout le fichier :


```
#!/bin/bash

OUT_FILE="/var/lib/node_exporter/textfile_collector/apt_updates.prom"

# Total updates
```

```
TOTAL_UPDATES=$(apt list --upgradeable 2>/dev/null | grep -v "Listing" |
wc -l)

# Security updates (si 'unattended-upgrades' est installé)
SEC_UPDATES=$(apt list --upgradeable 2>/dev/null | grep security | wc -l)

# Reboot required (flag fichier)
if [ -f /var/run/reboot-required ]; then
  REBOOT_NEEDED=1
else
  REBOOT_NEEDED=0
fi

# Export en format Prometheus
echo "# HELP debian_apt_updates Nombre total de mises à jour APT" >
"$OUT_FILE"
echo "# TYPE debian_apt_updates gauge" >> "$OUT_FILE"
echo "debian_apt_updates $TOTAL_UPDATES" >> "$OUT_FILE"

echo "# HELP debian_security_updates Nombre de mises à jour de sécurité"
>> "$OUT_FILE"
echo "# TYPE debian_security_updates gauge" >> "$OUT_FILE"
echo "debian_security_updates $SEC_UPDATES" >> "$OUT_FILE"

echo "# HELP debian_reboot_required Système nécessite un redémarrage" >>
"$OUT_FILE"
echo "# TYPE debian_reboot_required gauge" >> "$OUT_FILE"
echo "debian_reboot_required $REBOOT_NEEDED" >> "$OUT_FILE"
```

Dans le node_exporter.service

```
[Unit]
Description=Prometheus NODE Exporter
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=web
Group=web
ExecStart=/usr/local/bin/node_exporter --web.listen-address=0.0.0.0:XXXX
--collector.systemd --collector.processes --collector.textfile --
collector.textfile.directory=/var/lib/node_exporter/textfile_collector

SyslogIdentifier=web
Restart=always

[Install]
```

```
WantedBy=multi-user.target

-------------------------------------------------------------------------------
-----------------------------------------------

=> mkdir -p /var/lib/node_exporter/textfile_collector
=> chown -R web:root /var/lib/node_exporter/textfile_collector
=> chmod -R 770 /var/lib/node_exporter/textfile_collector

           => systemctl daemon-reexec
           => systemctl daemon-reload
           => systemctl restart node_exporter.service

Il faut maintenant faire un cron, pour executer le script "apt-
metrics.sh" régulièrement :

-------------------------------------------------------------------------------
-----------------------------------------------
=> crontab -e

30 7 * * * /usr/local/bin/apt-metrics.sh

-------------------------------------------------------------------------------
-----------------------------------------------

Maintenant il suffit sur grafana avec le dashboard, de regarder ce que
nous voulons voir.


Mise en place de UFW :

     => apt install ufw

Politique par défaut

     => ufw default deny incoming
     => ufw default deny outgoing

Maintenant les règles précises :

=> ufw allow from 172.16.14.1 to any port 14714 proto tcp comment 'SSH
depuis bastion'
=> ufw allow from 192.168.7.132 to any port 14714 proto tcp comment 'SSH
depuis SRV-RDS-AARD'
=> ufw allow from 192.168.7.128/28 to any port 9080 proto tcp comment
'VUE WEB DES TARGETS PROMTAIL'
=> ufw allow from 192.168.7.134 to any port 59051 proto tcp comment
'Prometheus avec node_exporter'
=> ufw allow from 172.16.7.129 to any port 80 proto tcp comment 'HTTP
depuis PROXY'
=> ufw allow from 172.16.7.129 to any port 443 proto tcp comment 'HTTPS
depuis PROXY'
=> ufw allow from 192.168.7.132 to any port 80 proto tcp comment 'HTTP
depuis SRV-RDS-AARD'
```

```
=> ufw allow from 192.168.7.132 to any port 443 proto tcp comment 'HTTPS
depuis SRV-RDS-AARD'
=> ufw allow out to 192.168.7.133 port 3100 proto tcp comment "Accès à
Loki pour logs"
=> ufw allow out 80/tcp comment 'Sortie HTTP vers dépôts Linux'
=> ufw allow out 443/tcp comment 'Sortie HTTPS vers dépôts Linux'
=> ufw allow out to 1.1.1.1 port 53 proto udp comment 'Sortie DNS vers
1.1.1.1'
=> ufw allow out 123/udp comment 'NTP'
=> ufw allow out 587/tcp comment 'Envoi de mails via WORDPRESS'
=> ufw allow out to 192.168.7.145 port 29590 proto tcp comment 'Requetes
vers BDD'
=> ufw enable

Pour ajouter la possibilité de PING, il faut modifier le fichier
/etc/ufw/before.rules et ajouter :

-------------------------------------------------------------------------
---------

# allow ICMP outbound (ping vers l'extérieur)
-A ufw-before-output -p icmp --icmp-type echo-request -j ACCEPT
-A ufw-before-output -p icmp --icmp-type destination-unreachable -j
ACCEPT
-A ufw-before-output -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-output -p icmp --icmp-type echo-reply -j ACCEPT

-------------------------------------------------------------------------
---------

Voila pour UFW, il faudra voir si d'autres problèmes sont présent, comme
ça on ajoute ou enleve des règles.


Installation lynis + fail2ban


=> Executer lynis dans syslog : lynis audit system | logger -t lynis

OU

=> Executer le script dans /usr/local/bin/lynis-syslog.sh

nano /usr/local/bin/lynis-syslog.sh

Script :

---------------------------------------

#!/bin/bash
lynis audit system | logger -t lynis

---------------------------------------
```

```
=> chmod 750 /usr/local/bin/lynis-syslog.sh
=> chown web /usr/local/bin/lynis-syslog.sh
```

Pas de cron.

On peut voir depuis GRAFANA dans les logs, le résultat.

Fail2Ban :

```
=> apt install fail2ban -y
=> systemctl enable fail2ban
=> systemctl start fail2ban
```

On ne modifie jamais directement jail.conf. On crée un fichier jail.local :

```
=> cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
=> nano /etc/fail2ban/jail.local
```

```
[sshd]
enabled = true
port    = ssh
logpath = %(sshd_log)s
backend = %(sshd_backend)s
maxretry = 5
bantime = 500
findtime = 600
```

```
=> systemctl restart fail2ban
=> fail2ban-client status
=> fail2ban-client status sshd
```

Fail2Ban lit les logs /var/log/auth.log (par défaut) pour détecter les tentatives SSH.