LIGNIERE
Timothée
BTS2 SIO
Date de création
12/01/2024
Date de maj
19/01/2024

NetShot

Sommaire

1 - Problématique

Lorsqu'une infrastructure devient de plus en plus importante, il peut devenir compliqué de s'assurer de la sauvegarde de toutes les configurations et de la version de chacune des machines par rapport aux mises à jour.

Netshot a la capacité de se connecter aux machines telles que les switchs et les routeurs afin d'en extraire des informations et de les traiter selon les besoins de l'utilisateur afin de ne pas avoir besoin de contrôler l'ensemble des machines individuellement.

Dans le cas de paramétrage spécifique/particulier, il est aussi possible d'écrire des scripts qui exécuteront des commandes pour vérifier ou appliquer des caractéristiques aux machines.

2 - Notice d’installation

En cas de problème, se référer à https://github.com/netfishers-onl/Netshot/wiki/Installing-Netshot-0.16-or-more-recent-on-Linux-server

2.1 – Installation PostGreSQL

Pour fonctionner, Netshot a besoin d’une base de données, nous allons donc installer PostgreSQL:

```
sudo apt install postgresql
```

Puis créer la base de données:

```
sudo -u postgres psql
CREATE USER netshot WITH ENCRYPTED PASSWORD « netshot » ;
CREATE DATABSE netshot01 WITH OWNER « netshot » ENCODING « UTF8 »
TEMPLATE template0;
\q
```

2.2 – Installation GraalVM JRE

Pour pouvoir interpréter du JS ou du Python, Netshot a besoin de GraalVM pour fonctionner ; selon la version de Netshot, la version de GraalVM peut être différente.

Au jour de la création de cette documentation, Netshot suit le tableau suivant concernant la version de GraalVM à utiliser :

Version de Netshot	Version de Graal
0.16.0 – 0.16.3	21.1.0
0.16.4 – 0.18.x	21.3.0
0.19.0 et plus récent	17.0.8-graal

L'installation de la version la plus récente suit donc les commandes suivantes :

```
GRAALVER="17.0.8"
    wget https://download.oracle.com/graalvm/17/archive/graalvm-jdk-${GRAALVER}_linux-x64_bin.tar.gz
tar xvzf graalvm-jdk-${GRAALVER}_linux-x64_bin.tar.gz
sudo mkdir /usr/lib/jvm
sudo mv graalvm-jdk-${GRAALVER}* /usr/lib/jvm/
sudo chown -R root:root /usr/lib/jvm/graalvm-jdk-${GRAALVER}*
sudo chmod 755 /usr/lib/jvm/graalvm-jdk-${GRAALVER}*
sudo ln -sfn /usr/lib/jvm/graalvm-jdk-${GRAALVER}* /usr/lib/jvm/graalvm
sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/graalvm/bin/java 92100
```

Désormais, vérifier que java pointe bien vers GraalVM JRE :

```
java -version 2>&1 | grep Environment | grep GraalVM
```

Cette commande devrait renvoyer le résultat suivant :

```
Java(TM) SE Runtime Environment Oracle GraalVM 17.0.8+9.1 (build 17.0.8+9-LTS-jvmci-23.0-b14)
```

Enfin, installer les langages JS et Python de GraalVM :

```
sudo /usr/lib/jvm/graalvm/bin/gu install js python
```

2.3 – Installation des paquets de Netshot

Tout d'abord, nous allons créer un utilisateur pour Netshot :

```
sudo adduser --system --home /usr/local/netshot --disabled-password --disabled-login netshot
```

Puis sélectionner votre version de Netshot sur la page Github du projet https://github.com/netfishers-onl/Netshot/releases :

Maintenant, suivre les commandes suivantes en remplaçant X.Y.Z par la version choisie (Par exemple, 0.19.3) :

```
NETSHOT_VERSION="X.Y.Z"
mkdir netshot_${NETSHOT_VERSION} && cd netshot_${NETSHOT_VERSION}
    wget https://github.com/netfishers-onl/Netshot/releases/download/v${NETSHOT_VERSION}/netshot_${NETSHOT_VERSION}.zip
unzip netshot_${NETSHOT_VERSION}.zip
sudo cp netshot.jar /usr/local/netshot
sudo mkdir /usr/local/netshot/drivers
sudo chown -R netshot /usr/local/netshot
sudo mkdir /var/local/netshot
sudo chown -R netshot /var/local/netshot
sudo mkdir /var/log/netshot
sudo chown -R netshot /var/log/netshot
sudo cp netshot.conf /etc/netshot.conf
sudo chown netshot /etc/netshot.conf
sudo chmod 400 /etc/netshot.conf
sudo cp systemd-netshot /etc/systemd/system/netshot.service
sudo systemctl daemon-reload
sudo systemctl enable netshot.service
sudo systemctl start netshot
```

Netshot devrait désormais être actif, vous pouvez donc vous connecter au port 8443 de votre machine dans un navigateur web de la manière suivante :

```
http://<Adresse du serveur>:8443
```

Les identifiants par défaut sont les suivants :

```
Account : admin
Password : netshot
```

Si vous souhaitez ajuster des paramètres tels que les identifiants de la base de données, dirigez-vous vers le fichier netshot.conf :

```
sudo nano /etc/netshot.conf
```

2.3 Addendum – Mise à jour de Netshot

Netshot étant un projet OpenSource, il est régulièrement mis à jour avec des corrections de bugs et parfois de nouvelles fonctionnalités, il est donc important de savoir comment faire ces mises à jour.

Pour commencer, télécharger le fichier zip de la version que vous souhaitez utiliser à la place de l'ancienne :

```
NETSHOT_VERSION="X.Y.Z"
mkdir netshot_${NETSHOT_VERSION} && cd netshot_$
{NETSHOT_VERSION}
    wget https://github.com/netfishers-onl/Netshot/releases/download/v$
{NETSHOT_VERSION}/netshot_${NETSHOT_VERSION}.zip
unzip netshot_${NETSHOT_VERSION}.zip
```

Une fois le fichier décompressé, faire une copie des fichiers suivants par sécurité si vous aviez déjà des données avant la mise à jour :

```
/usr/local/netshot
/usr/local/netshot/drivers #Si vous avez créé des drivers personnalisés
/etc/netshot.conf
```

Ainsi que la base de données à l'aide de la commande suivante :

```
sudo -u postgres pg_dump netshot01 > Netshot_backup_database
```

Si cette sauvegarde a besoin d'être utilisée, faire :

```
sudo -u postgres pg_restore netshot01 < Netshot_backup_database
```

Puis copier le paquet mis à jour après avoir arrêté le service :

```
sudo systemctl stop netshot
sudo cp netshot.jar /usr/local/netshot
```

Dans les notes de mise à jour sur la page du projet, vérifier s'il n'y a pas des actions supplémentaires à faire (Tel que mettre à jour GraalVM) ; Vérifier la compatibilité de vos drivers personnalisés avec la nouvelle version si vous en aviez puis redémarrer le service :

```
sudo systemctl restart netshot
```

En cas de problème vis à vis de certaines données, arrêter à nouveau le service puis utiliser vos sauvegardes pour remplacer les potentielles données corrompues.

3 - Notice d’utilisation

L’interface de Netshot ayant tendance à être modifiée selon les versions, j’éviterais dans la mesure du possible toute capture d’écran.

La notice d’utilisation suivante a été faite sur la version 0.19.3 de Netshot, certaines fonctionnalités peuvent avoir légèrement changé si vous utilisez une version plus récente.

L’ordre des fonctionnalités présentées dans les futures paragraphes repose sur la présentation de Netshot, ce n’est donc pas forcément l’ordre logique d’utilisation des fonctionnalités (Par exemple, les informations extraites des machines sont présentées avant la connexion des machines elles-mêmes).

3.1 - Reports

Reports est l'endroit où les données traitées des switchs sont centralisées, on a donc ainsi ici un récapitulatif des derniers changements ou des résumés des fichiers de configuration.

3.1.1 – Configuration Changes

Toutes les modifications détectées sur un fichier de configuration sur une machine sont répertoriées ici sous la forme d'un tableau afin de savoir combien de modifications ont été faites sur une période donnée.

3.1.2 – Device Access Failures

Tous les appareils que Netshot n'a pas réussi à contacter depuis plusieurs jours sont répertoriés ici sous la forme d'une liste avec leur nom d'hôte, leur adresse IP, le type d'appareil et la date des dernières captures de fichiers de configurations réussies.

3.1.3 – Configuration Compliance

Les retours de compliances sont renvoyés ici afin de savoir combien de machines par groupes sont considérées comme conformes vis à vis des compliances créées (Plus d'information dans 3.4 – Compliance), en cliquant sur un nom de groupe, les noms des machines non conformes apparaissent sous forme de liste sous le tableau.

3.1.4 – Software compliance

Les retours de vérification de version des machines sont centralisés ici par groupe sous la forme de camemberts. Les versions doivent avoir été déclarées au préalable dans les compliances car Netshot ne va pas chercher les dernières versions en date des firmwares, c'est à l'utilisateur d'indiquer à NetShot quelles versions sont les plus récentes. (Plus d'information dans 3.4 – Compliance).

Selon ce qui a été définit au préalable, chaque machine aura soit le statut « Gold », « Silver », « Bronze » ou encore « Non Compliant ».

Si une machine est « non compliant », c'est que la version du firmware de cette machine n'a pas été déclarée.

En cliquant sur chacun des statuts de chaque groupe, on peut déterminer les machines qui font parties de chaque catégorie.

3.1.5 – Hardware support status

Le résumé des dates de fin de vente et de support des machines se trouve ici. Pour la même raison que pour les versions de firmware, l'utilisateur doit lui-même définir au préalable les dates de fin de support de chaque modèle afin que NetShot l'avertisse (Plus d'information dans 3.4 - Compliance).

3.1.6 – Data export

A partir de cette fenêtre, il est possible d'exporter des données générales sur un ensemble de machines sous la forme d'un tableau excel afin d'avoir une trace s'il y a besoin d'un archivage des données générales.

Il est possible d’exporter les données suivantes :

- Les détails de groupes
- Liste des appareils et de leurs particularités
- Liste des interfaces en incluant les adresses IP et MAC (Inclut les VLANs)
- Liste des modules de chaque appareil (Avec les numéros de séries)
- Liste des appareils avec leurs positions et contacts
- Liste des compliances des appareils

Il est nécessaire de préciser, que NetShot est tout à fait capable d’exporter toutes ces données sur un seul fichier excel, il suffit de cocher chaque option nécessaire avant d’exporter.

3.2 – Devices

L’ensemble des machines et des groupes présents sur NetShot sont tous répertoriés à cet emplacement, c’est par conséquent aussi ici qu’il est possible d’interagir avec les machines.

3.2.1 – Add a device

Cette fonctionnalité se rajoute à la barre en haut du site de NetShot quand l’utilisateur se trouve sur la page de **Devices.** Comme son nom l’indique, elle permet de relier les machines à NetShot

3.2.1 Addendum – Permettre à une machine d’être détectée par NetShot

Pour détecter les machines lors de la première interaction avec NetShot, NetShot se base sur les paquets SNMP, il est donc nécessaire de les activer sur la machine à connecter, vous pouvez le faire à l’aide des commandes suivantes en terminal de configuration sur vos machines cisco :

```
snmp-server community public RO
snmp-server community private RW
```

Penser à sauvegarder votre configuration après avoir fait cette modification.

3.2.1.1 – Add a simple device

Permet de rajouter un seul appareil en indiquant son adresse IP, NetShot cherchera par défaut à savoir quel type d’appareil sera relié.

3.2.1.2 – Scan subnet(s) for devices

Permet de chercher un ensemble d’appareil soit en cherchant par un ou plusieurs sous réseaux et/ou plusieurs adresses de machines pour ne pas avoir besoin d’utiliser plusieurs fois l’option « Add a simple device »

3.2.1.3 – Add a group

Permet de créer un groupe à partir d’un nom, il est aussi possible de créer des dossiers de groupes. Il existe deux types de groupes, les groupes statiques et les groupes dynamiques.

3.2.1.3.1 – Static Group

Dans un groupe statique, il est nécessaire de rajouter manuellement chaque machine au groupe à partir des paramètres de groupe.

3.2.1.3.2 – Dynamic group

Dans un groupe dynamique, il est simplement nécessaire de créer une expression simple qui cherchera un critère particulier sur toutes les machines, les rajoutant ainsi automatiquement si elles remplissent cette condition.

3.2.2 – Schedule a Task

Cette fonctionnalité étant présente à la fois sur la page de « Devices » et de « Tasks », elle ne sera présentée qu’une seule fois ici.

Elle permet donc de prévoir une tâche sur l’instant ou dans un temps donné.

A chaque fois que le texte « The task will run once, now (click for more options) » apparaît, cliquer dessus pour définir un temps précis pour la tâche et si cette même tâche doit être un événement récurrent.

Les événements définis comme des événements récurrents peuvent être retrouvés dans la case « Scheduled » des « Tasks »

3.2.2.1 – Take a snapshot of a group of devices

Cette tâche prendra une snapshot sur un ensemble de machines faisant partie d'un groupe, il est donc possible de préciser sur quel groupe elle doit s'exécuter selon certains critères.

Par défaut, NetShot cherchera ensuite à faire les diagnostics et à vérifier les compliances des machines ayant reçu cette tâche.

3.2.2.2 – Run diagnostics on a group of devices

Cette tâche appliquera tous les diagnostics ayant un rapport avec le groupe qui recevra la tâche (Plus d'informations sur les diagnostics dans 3.3 – Diagnostics) et cherchera par défaut à vérifier les compliances des machines.

3.2.2.3 – Check the config compliance of a group of devices

Cette tâche appliquera toutes les compliances ayant un rapport avec le groupe qui recevra la tâche (Plus d'informations dans 3.4 - Compliance).

3.2.2.4 – Check the software compliance and hardware support status of a group of devices

Cette tâche vérifiera la version du firmware et si le matériel est encore sous support selon ce qui a été définit au préalable dans les compliances (Plus d'informations dans 3.4 - Compliance)

3.2.2.5 – Scan subnets to discover devices

Scannera des sous réseaux pour découvrir de nouveaux appareils, peut être fortement intéressant combiné à l'option de faire de cette tâche un événement récurrent si vous rajoutez très souvent de nouvelles machines à votre infrastructure.

3.2.2.6 – Purge old entries from the database

Détruira les anciennes informations de la base de données si elles remplissent certaines conditions telles que le temps d'existence de ces configurations.

3.2.3 – Fonctionnalités lorsqu'une machine est sélectionnée

Lorsque vous sélectionnez une machine, vous obtenez des options supplémentaires afin de connaître un grand nombre d'informations la concernant.

3.2.3.1 - Général

C'est ici que vous retrouverez les informations les plus basiques sur votre machine tels que :

- Le nom d'hôte
- L'adresse IP
- Le type d'appareil
- La famille de l'appareil (si elle reconnue (E.g Cisco Catalyst 2900))

3.2.3.2 – Configuration

L'historique des configurations enregistrées par NetShot se trouve à cet emplacement.

Vous pouvez comparer deux versions du fichier de configuration en glissant le 'nom' (E.g. 2024-01-16 X:X) du fichier de configuration vers le bouton « Drop to Diff x », après avoir glissé deux fichiers vers les deux emplacements, le bouton « Compare » apparaitra et vous permettra de les comparer.

Vous pouvez aussi voir directement le fichier de configuration ou le télécharger à l'aide des boutons « Views » et « Download » après avoir cliqué sur l'une des versions.

Il est bon de noter que NetShot stocke les fichiers de configurations uniquement s'ils sont différents de l'ancien et que les fichiers de configuration ne correspondent qu'aux fichiers lors de la snapshot. Cela signifie donc que seule la dernière version de la configuration sera prise en compte lors de la Snapshot, des versions du fichier de configuration peuvent donc se retrouver ignorées.

3.2.3.3 - Interfaces

Cette fenêtre vous permet de voir l'ensemble des interfaces, VLANs inclus ; Vous pouvez aussi récupérer l'adresse IP et MAC de chaque interface.

Les interfaces marquées en « Disable » signifient qu'elles sont en « shutdown » et non pas en « no shutdown » sur la running-config.

3.2.3.4 – Modules

Permet d'avoir une liste des modules présents sur l'appareil et sur chaque machine de l'appareil (Dans le cas d'un stack par exemple). Si des modules sont retirés / non détectés, alors, NetShot le garde en tête mais le place dans l'historique des modules.

L'historique peut être ignoré en décochant la case « Show history » en bas du tableau.

3.2.3.5 - Diagnostics

Renvoie les résultats des différents diagnostics faits par Netshot sur la machine (Plus d'information dans 3.3 – Diagnostics).

Si les résultats des diagnostics ne vous paraissent pas à jour, vous pouvez utiliser le bouton « Run diagnostics on this device » pour lancer une tâche qui va appliquer à nouveau les diagnostics.

3.2.3.6 – Compliance

Renvoie les résultats de l'ensemble des compliances liés à la machine ; Par défaut, Netshot montrera uniquement les policy compliances qui sont en Non conforme, il est possible de voir aussi celles qui sont en conforme en décochant la case « Non conforming only »

Dans le cas où le retour des compliances ne vous paraît pas à jour, vous pouvez utiliser le bouton « Recheck Compliance » pour refaire une tâche de vérification des compliances

3.2.3.7 - Tasks

Liste les vingt dernières tâches qui ont eu lieu sur la machine, qu'elles soient réussies ou échouées.

3.2.3.8 – Edit

C'est à cet emplacement que vous pouvez modifier les paramètres de connexion vers une machine pour indiquer des précisions si nécessaire :

- Connexion avec identifiants globaux (Plus d'information dans 3.6 – Admin)
- Connexion avec compte ssh spécifique et mot de passe spécifique
- Connexion avec compte ssh spécifique et clé ssh spécifique
- Connexion avec compte telnet spécifique et mot de passe spécifique

Dans tous les cas, NetShot vous demandera le mot de passe pour utiliser le mode « enable » du switch.

3.2.3.9 – Delete

Permet de supprimer l'intégralité des informations lié à une machine, rien ne sera conservé.

3.2.3.10 – Enable/Disable

« Active » ou « Désactive » la machine du point de vue de NetShot, si elle est en « Disable » ça ne l'éteindra pas, cela signifiera que toutes les tâches la concernant seront ignorées.

3.2.3.11 – Run Script

Permet de lancer des scripts en Javascript pour interagir avec la machine à distance. Ces scripts peuvent être sauvegardés pour être réutilisés plus tard, **cependant, si vous essayez de sauvegarder une modification sur un script déjà existant, vous devez le supprimer avant de le sauvegarder.** (Une requête a été faite sur le projet Github pour améliorer ce point et simplement écraser les anciennes sauvegardes).

Ce type de script ne vous renverra pas particulièrement d'information, il vous indiquera s'il a réussi la connexion et si les commandes demandées ont été appliquées.

Pour recevoir des informations depuis la machine par un script, cela se passe au niveau des compliances et des diagnostics (Plus d'informations dans 3.3 – Diagnostics et 3.4 - Compliance).

Plus de détail sur les scripts :

Pour l'ensemble des scripts, la structure reste très similaire à chaque fois sous la forme suivante :

```
Function run(cli,device) {
        cli.macro(«<Macro>») ;
        cli.command(«<Command>») ;
}
```

Pour Cisco IOS, <Macro> peut être remplacé par les termes suivants :

- enable (Qui vous mettra dans le terminal d'enable)
- configure (Qui vous mettra dans le terminal de configuration)
- save (Pour sauvegarder running-config sur startup-config)
- end (Pour retourner sur le terminal d'enable quand vous étiez dans le terminal de configuration)

Les macros « save » et « end » ne semblent pas fiables, essayer de les ignorer en utilisant les commandes normales (E.g. « wr » et « end »).

Encore sur Cisco IOS, <Command> peut être remplacé par n'importe quelle commande que vous utiliseriez sur le terminal, mais il faut prendre en compte la macro vous avez utilisée en dernier dans votre script pour une pas demander une commande du terminal de configuration sur celui d'enable par exemple.

Il est aussi possible de demander des valeurs de variable à l'utilisateur et de les utiliser de la manière suivante (Pour montrer l'utilisation, nous allons prendre un script qui change l'hostname de la machine) :

```
const Input = {
        Hostname: {
                        label : « Hostname »,
                        description : « Hostname qui sera appliquer sur la machine »,
        }
} ;
function run(cli,device) {
        const { Hostname } = cli.userInputs ;
        cli.macro(«configure») ;
        cli.command(‘hostname ${Hostname}‘) ;
}
```

Toutes commandes utilisant une variable dans le même type de cas que pour « Hostname » doit utiliser « ` » à la place des guillemets, sans ceci, la variable sera simplement vue comme du texte normal et ce ne sera pas le contenu de la variable mais simplement son nom qui sera utilisé.

Vous pouvez tout à fait récupérer plusieurs variables de cette manière d'un seul coup :

```
const Input = {
        VAR1: {
                        label : « VAR1 »,
                        description : « Description de la VAR1 »,
        },
        VAR2 : {
                        label : « VAR2 »,
                        description : « Description de la VAR2 »,
} ;
function run(cli,device) {
        const { VAR1, VAR2 } = cli.userInputs ;
        cli.macro(«<Macro>») ;
        cli.command(`<Command> ${VAR1} ${VAR2} `) ;
}
```

Récupérer des variables depuis l'utilisateur est intéressant mais vous pouvez aussi récupérer des variables depuis la machine :

```
function run(cli,device) {
        cli.macro(«enable») ;
        let config = cli.command(« show running-config »)
}
```

Puis à l'aide de Javascript, vous pouvez traiter cette variable de la manière désirée.

Dans le cas d'une commande qui demande une confirmation telle que « reload » sur Cisco, il est possible d'utiliser la forme suivante (Exemple avec la commande « reload ») :

```
function run(cli,device) {
	cli.macro(«enable») ;
	cli.command(« reload », {
			mode : {
				prompt : /Proceed with reload\? \[confirm\]/,
			},
			timeout:300000,
	});
	cli.command(« y ») ;
}
```

Il est important de noter que la phrase en face de « prompt » est une RegEx, elle doit donc exactement correspondre à la ligne de l'IOS demandant une confirmation. La commande suivante sera utiliser lorsque la Regex correspondra à une ligne du terminal (Par conséquent, si la regex ne correspond à rien, le script sera bloqué).

En cas de problème pour rédiger la regex, se référer à regex101.com

Lors de la création d'un script, il faut prendre en compte que vous manipulez un programme qui va lui-même interagir avec l'IOS de votre machine. Si vous vous retrouvez bloqué sur une commande, chercher à le contourner à l'aide de l'autre langage (Soit JS, soit l'IOS dans ce sens).

Par exemple, lors d'une copie depuis un serveur TFTP, Cisco IOS vous demandera de confirmer le nom du fichier et l'adresse du serveur ; or, NetShot n'est pas encore capable d'utiliser simplement la touche « Entrée » du clavier pour confirmer. Ce problème peut être ignoré à l'aide de la commande « file prompt quiet » qui empêchera Cisco IOS de vous demander une confirmation.

Il faut essayer de toujours chercher les limites de l'IOS et les capacités de NetShot à interagir avec la machine selon vos besoins.

3.2.3.12 – Snapshot

Prendra une snapshot de la machine à un moment désiré.

Les snapshot sont les captures de fichier de configuration de NetShot, si aucune modification n'est détectée avec l'ancienne snapshot, alors NetShot ignore le fichier de configuration actuelle l'ayant déjà techniquement. Or, s'il y a une différence entre les deux, qu'importe l'ampleur, NetShot sauvegardera la configuration.

3.2.3 Addendum – Plusieurs machines sélectionnées

Quand vous sélectionnez plusieurs machines en utilisant le clic gauche de votre souris et bouton « Ctrl », vous obtiendrez un menu plus restreint permettant d'utiliser des fonctionnalités sur les machines sélectionnées en même temps.

Ces fonctionnalités sont :

- Edit
- Delete
- Enable/Disable
- Run a script
- Take a snapshot
- Check Compliance
- Run diagnostics

Les modifications apportées dans ce type de cas seront appliquées à toutes les machines sélectionnées.

3.3 – Diagnostics

Par défaut, NetShot ne propose aucun diagnostic, il faudra donc les créer.

Les diagnostics sont des scripts/requêtes faits aux machines pour obtenir facilement des données qui sont ensuite accessibles sur la page « Diagnostics » dans « Devices ».

3.3.1 – Create Diagnostics

« Create Diagnostics » est uniquement accessible en étant sur la page de « Diagnostics » et devient visible dans la barre du haut du site de NetShot.

Il permet de créer un diagnostic selon différents paramètres :

- Le nom du diagnostic
- L'activation du diagnostic
- Le type de résultat (Soit texte, soit numérique ou binaire)
- Application à un groupe (Au jour de la version 0.19.3 de NetShot, il n'est pas possible d'appliquer un diagnostic à plusieurs groupes, une requête à ce sujet a été faite sur GitHub)
- Le type de diagnostic (Simple, Javascript ou Python. Plus d'informations dans les sous chapitres suivants)
- **Les paramètres suivants dépendent du type de diagnostic**

3.3.1.1 - Simple

Un diagnostic de type « simple » est un diagnostic qui prend donc la forme d'une requête simple en demandant les paramètres suivants :

- Le type d'appareil (Cisco IOS ou Avaya ERS par exemple)
- Le mode du terminal pour faire la requête (enable ou configure)
- La commande qui sera faite dans le terminal pour récupérer les données
- Un paterne Regex (Un terme que vous souhaitez remplacer par un autre)
- Détermination du remplacement du paterne Regex

3.3.1.2 - JavaScript-based

Un diagnostic de type « Javascript-based » est donc un diagnostic qui se basera sur un script écrit en JS pour récupérer des données.

L'exemple donné par NetShot vis à vis d'un tel diagnostic est le suivant :

```
function diagnose(cli,device,diagnostic) {
        cli.macro(«enable») ;
        var output = cli.command (« show something ») ;
        // Process output somewhat
        diagnostic.set(output) ;
}
```

3.3.1.3 - Python-based

Un diagnostic de type « Python-based » est donc un diagnostic qui se basera sur un script écrit en Python pour récupérer des données.

L'exemple donné par NetShot vis à vis d'un tel diagnostic est le suivant :

```
def diagnose(cli,device,diagnostic) :
        cli.macro(« enable »)
        output = cli.command(« show something »)
        diagnostic.set(output)
```

3.4 - Compliance

Les « compliances » sont des vérifications de paramètres ou de version d'hardware/software afin de vérifier si tout est conforme.

3.4.1 – Create policy

« Create policy » apparaît lorsque vous êtes sur la page de « Compliance » dans la barre en haut du site de NetShot.

Il vous permet donc de créer des politiques de compliances personnalisées.

Lors de la création d'une politique, NetShot vous demandera un nom de politique et sur quels groupes il doit s'appliquer.

Une fois la politique créée, vous devrez créér des règles de compliance.

3.4.1.1 - Edit

« Edit » permet de mettre à jour les paramètres que vous aviez définis lors de la création de la politique.

3.4.1.2 - Delete

« Delete » supprimera la politique et toutes les règles qui sont liées à elle.

3.4.1.3 – Add rule

« Add rule » vous permet de rajouter une règle de compliance à partir d'un simple nom, il en existe trois types présentés dans les sous chapitres suivants.

Pensez bien à utiliser le bouton « Add » en rajoutant la règle, le bouton « Entrée » du clavier ne la rajoutera pas.

3.4.1.3.1 – Simple text (or regular expression) rule

Ce type de règle vous demandera les paramètres suivants pour être fonctionnel :

- Le type d'appareil (Cisco ou Avaya par exemple)
- Où aller chercher les données dans la machine ou dans ce que contient déjà Cisco (Je n'irais pas dans les détails étant donné que cela peut être différent pour chaque type d'appareil)
- Tous les blocs doivent être valides / Au moins un bloc est valide
- Le texte doit exister / Le texte ne doit pas exister
- Le texte fourni est une expression régulière
- Comparer le texte à la section entière
- Normaliser le texte
- Texte ou paterne fourni.

Une fois votre règle créée, vous pouvez la tester grâce au bouton « Select a Device » et « Test... » afin de vérifier si le résultat est bien celui attendu selon les appareils et ce qui est vérifié.

3.4.1.3.2 – Javascript rule

Une règle en Javascript repose sur le script que vous lui donnez pour déterminer si une machine est conforme ou non.

Par exemple, pour vérifier que le site web de vos machines cisco est bien désactivé, vous utiliseriez la règle suivante :

```
function check(device) {
	var config = device.get(« runningConfig ») ;
	if ((config.includes(« no ip http server »)) && (config.includes(« no ip http secure-server))){
			return CONFORMING ;
}
	else {
			return NONCONFORMING ;
	}
}
```

3.4.1.3.3 – Python rule

Une règle en Python repose aussi sur le script que vous lui donnez pour déterminer si une machine est conforme ou non.

Encore une fois, si vous voulez vérifier que le site web de vos machines cisco est bien désactivé, vous utiliseriez la règle suivante :

```
Def check(device) :
	config = device.get(« runningConfig »)
	if (« no ip http server » in config) and (« no ip http secure-server » in config) :
			return result_option.CONFORMING
	else :
			return result_option.NONCONFORMING
```

3.4.1.3 Addendum – Paramètres des règles

Les règles sont certes appliquées à des groupes à l'aide des politiques, mais vous pouvez aussi éviter qu'une machine ne subisse une vérification à l'aide des paramètres des règles symbolisé par la clé à molette à côté du nom de la règle.

C'est aussi ici que vous pouvez activer ou désactiver la règle selon vos besoins.

Vous pouvez aussi supprimer une règle à l'aide du bouton en forme de corbeille au niveau du nom de la règle.

Si vous souhaitez apporter des modifications à une règle, utiliser le bouton « Edit the script » ou « Edit the content » (Le terme change selon si c'est un diagnostic par expression ou par script).

3.4.2 - Software

Vous pouvez rajouter des règles vérifiant la version des firmwares de vos appareils afin de déterminer lesquels doivent être mis à jour. Pour cela, vous pouvez utiliser le bouton en forme de plus qui vous demandera les paramètres suivants :

- Sur quel groupe appliquer la règle ? (Vous pouvez utiliser [any] pour l'appliquer sur tous)
- Le type d'appareil (Avaya ou Cisco par exemple)
- La famille de l'appareil (Cisco catalyst 2900 par exemple)
- Le numéro de partie si vous voulez vérifier sur un hardware bien spécifique
- La version du firmware
- Le résultat, soit Gold ou Silver ou encore Bronze. (Dans une optique idéale, un appareil en Gold a la dernière version, Silver n'est pas à la dernière mais n'est pas à un point critique et Bronze doit être mis à jour le plus vite possible)

3.4.3 - Hardware

Très similaire au software compliance, celui-ci permet de contrôler les dates de fin de support d'appareils et de fin de ventes de ces appareils. Encore une fois, il faut les définir soit-même d'une manière similaire aux software :

- Sur quel groupe appliquer la règle ? (Vous pouvez utiliser [any] pour l'appliquer sur tous)
- Le type d'appareil (Avaya ou Cisco par exemple)
- La famille de l'appareil (Cisco catalyst 2900 par exemple)
- Le numéro de partie si vous voulez vérifier sur un hardware bien spécifique
- La date de fin de vente du modèle en question
- La date de fin de support du modèle en question

3.5 - Tasks

L'historique des différentes tâches sur l'ensemble peut être retrouvé ici ainsi que les tâches en cours et celles prévues. La différence avec l'onglet « Tasks » de « Devices » est que celui-ci représente véritablement l'ensemble des tâches là où dans « Devices », seules les tâches liées à la machine étaient visibles.

Vous pouvez obtenir des informations basiques sur une tâche à l'aide du bouton ressemblant à deux fenêtres d'ordinateurs se chevauchant, vous indiquant ainsi l'ID de la tâche, la description de la tâche, le commentaire sur la tâche, qui l'a créée quand, à quelle date et heure elle doit s'exécuter et quel est son statut.

Vous pouvez aussi définir sur quelle journée vous souhaitez avoir l'historique avec la date à côté de la case « All »

3.5.1 – Schedule a task

Ce « Schedule a task » est identique à celui présenté dans 3.2.2 – Schedule a task.

3.5.2 – Running

La case « Running » du tableau montre toutes les tâches qui sont en cours d'exécution, si une tâche prend plus de temps que d'habitude, n'hésitez pas à redémarrer le serveur.

Cela reste rare mais le risque n'est pas de 0 %.

3.5.3 - Scheduled

La case « Scheduled » liste toutes les tâches qui sont prévues dans le futur, vous pouvez les annuler à l'aide du bouton suivant afin d'éviter leur exécution :

Les tâches ayant une forme de flèche en spirale sont des événements récurrents, elles ont donc été réglées pour se répéter périodiquement.

3.5.4 - Succeeded

La case « Scheduled » liste toutes les tâches qui ont réussi leur exécution.

3.5.5 – Failed

La case « Failed » liste les tâches ayant échoué leur exécution, si une snapshot ou un script a potentiellement échoué, vous pouvez tout à fait le retrouver ici.

3.5.6 - Cancelled

La case « Cancelled » liste les tâches qui ont été annulées par un utilisateur pour X raisons, c'est donc ici qu'il faut chercher si vous pensez avoir annulé une tâche par accident.

3.5.7 - All

L'historique de toutes les tâches confondues, quels que soient leurs statuts, se trouve à cet emplacement.

3.6 - Admin

L’ensemble des paramètres liés aux droits de connexions et accès se trouve ici, pour la grande majorité du temps, il n’est pas nécessaire d’utiliser cette section ; néanmoins, ça peut le devenir dans certains cas.

3.6.1 – Users

Liste des utilisateurs de NetShot et leurs droits afin de contrôler les accès. Il est possible de rajouter des utilisateurs à l’aide du bouton « Add » en dessous de cette liste, auquel cas, NetShot vous demandera les paramètres suivants :

- Nom d’utilisateur
- Le type de compte (Visitor/Operator/Read-write/Read-write and commands/Admin)
- Utilisateur distant ou non
- Mot de passe

3.6.2 – Device Domains

Les domaines auxquels NetShot fait partie, vous pouvez en rajouter avec le bouton « Add », NetShot vous demandera ensuite les paramètres suivants :

- Nom du domaine
- Description
- Adresse du serveur de domaine

3.6.3 – Device Credentials

Vous pouvez enregistrer des identifiants pour les différents moyens de connexion vers les machines à partir de ce point, vous évitant ainsi de devoir toujours réécrire les identifiants pour chaque machine s’ils sont toujours les mêmes et de pouvoir donc les modifier depuis un seul point si nécessaire.

Vous pouvez rajouter plusieurs comptes pour un même moyen de connexion si nécessaire à l’aide du bouton « Add », nous n’irons pas dans les détails, les paramètres étant différents pour chaque protocole (E.g Telnet et SSH.).

3.6.4 – Device Drivers

Les différents drivers qu’utilisent NetShot pour interagir avec les machines, s’il manque un driver pour interagir avec l’une de vos machines, vous pouvez le coder vous-même, malheureusement, la documentation sur ce point est assez vide (Une requête sur ce point sur GitHub a déjà été fait, vous pouvez néanmoins essayer de demander une issue si vous êtes véritablement bloqués)

3.6.5 – API Tokens

La liste des tokens de connexion qui ont été créés pour accéder à l’API de NetShot, je ne me concentrerais pas sur ce point car c’est un point différent de l’utilisation de NetShot.

3.6.6 – Web Hooks

Liste les Web Hooks qui ont été créés, je ne m’étendrais pas non plus sur cette utilisation des capacités de NetShot.

3.6.7 – Clustering

Liste les autres serveurs de NetShot qui forment un cluster, je n’ai pas eu l’occasion de le mettre en place pour le tester, je n’ai donc pas de documentation sur ce point.

4 - Annexes

4.1 – Fiche Recette

Fiche de recette

Vérification de l'opérationnalité de la solution mise en œuvre : *NetShot*

Description du test :

1. *Vérification du statut du service de NetShot*
2. *Tentative de connexion au site web*
3. *Rajout d'une machine*
4. *Tentative de snapshot sur la machine rajouter*

Résultats Attendus :

1. *Statut du service en active(running)*
2. *Connexion au site réussie*
3. *Rajout de la machine réussi*
4. *Snapshot réussi*

Réception Globale : NetShot **Date: 18/01/24** **Auteurs: Timothée Lignière**
Reçu : Reçu avec réserve : …… Refusé : …… Commentaire :

Recette étape par étape *	***** *(pour chaque étape, vous devez élaborer dans un fichier distinct un scénario détaillé à faire appliquer au « client » venant valider votre solution)*

Réception Etape 1 : Vérification du statut du service à l'aide de la commande « systemctl status netshot » qui doit donc renvoyer le statut active(running). Si ce test a échoué, faire un « systemctl restart netshot » puis revérifier le statut, si le test est toujours un échec, ne pas recommencer.
Reçu : Reçu avec réserve : …… Refusé : …… Commentaire :

Réception Etape 2 : En se connectant au port 8443 de l'adresse IP de la machine, on a bien la page de connexion de NetShot, et il est possible de s'y connecter avec les identifiants fournis (Par défaut : Username – Admin , Password : netshot)
Reçu : Reçu avec réserve : …… Refusé : …… Commentaire :

Réception Etape 3 : A partir du menu « Devices » de NetShot et du bouton « Add a device » puis de l'option « Add a simple device » NetShot réussi à rajouter un appareil (Un problème de connexion pourrait aussi venir de l'appareil en question, veuillez assurer sa capacité de connexion au préalable)
Reçu : Reçu avec réserve : …… Refusé : ……

Commentaire :

Réception Etape 4 : Toujours dans le menu « Devices », utiliser le bouton « Snapshot » après avoir sélectionné la machine, (Veuillez encore vérifier les options de connexion au préalable afin d'assurer le bon fonctionnement du logiciel)
Reçu : Reçu avec réserve : …… Refusé : …… Commentaire :

4.2 – Activer le SSH sur une machine Cisco

Pour pouvoir interagir correctement avec une machine, NetShot a besoin que cette dernière ait son serveur SSH d'actif et qu'il ait ses identifiants.

La partie des identifiants a déjà été énoncée dans les chapitres précédents, voici donc les commandes à utiliser pour activer le serveur SSH sur une machine vierge (De légères différences pourraient être nécessaires dans le cas d'une machine déjà configurée) :

```
(config)line vty 0 4
(config-line)login local
(config-line)transport input ssh
(config-line)exit
(config)ip domain-name <Domaine>
(config)crypto key generate rsa #Puis choisissez la longueur de la clé
(config)username <Name> password <Password>
(config)enable password level 15 <Password>
```

Penser bien à sauvegarder après cela.

4.3 – Activer le SNMP sur une machine Cisco

Tout comme pour le SSH, NetShot a parfois besoin du SNMP pour détecter la machine, voici donc les commandes pour l'activer :

```
(config)snmp-server community public RO
(config)snmp-server community private RW
```

Penser à sauvegarder après avoir terminé.

4.4 – Obtenir les logs de communication entre NetShot et une machine

Parfois, lors de la création d'un script, il peut être compliqué de déterminer où est le problème en recevant une failure de script, et les logs d'un script n'indiquent pas toujours des informations véritablement utiles.

Il est donc possible d'obtenir les communications entre la machine et NetShot en rajoutant la ligne suivante dans **/etc/netshot.conf** :

```
netshot.log.class.onl.netfishers.netshot.device.access.Cli = ALL
```

L'ensemble des communications sera ensuite visible dans le fichier **/var/log/netshot/netshot.log .**