Documentation script OCprojet6

Liste scripts et fonctions :

config.py :
Contient les différentes fonctions créés pour le programme.
generate_config.py :
Programme principale à exécuter.
menu.py :
Menu du programme, utilisé uniquement pour l'affichage du menu.
script_save.py :
Sous-programme pour la sauvegarde des devices (switchs et routeurs des différents sites), peut être exécuté depuis le programme principale ou en autonome pour l'automatisation de la sauvegarde.

Fonctionnement des scripts :

generate_config.py & menu.py :

Le programme et le menu ne font qu'afficher et gérer le choix de ce que l'on veut faire.
- créer un site
- créer un routeur
- créer un switch
- lancer un backup

config.py :

Contient les différentes fonctions que lance le programme "generate_config.py", ainsi que ces propres fonctions.

Les variables que l'on aura besoin dans tout le programme et qui seront modifiées aux besoin de chacun.

```
# Définition des variables globales
ip_serv_ftp = "10.1.2.100"          # adresse ip du serveur FTP
dir_ftp = "/home/share/"            # répertoire de base du serveur FTP
dir_site = "site/"                  # répertoire pour les fichiers sites
dir_config ="config/"               # répertoire pour les fichiers des config
dir_sav ="sav/"                     # répertoire pour les fichiers de backup
dir_template = "template/           # répertoire pour les templates de config

file_device = "list_device.csv"     # fichier qui liste les devices existants
                                    ficher de type CVS qui contient le nom du device et son
                                    adresse ip.
                                    Exemple : ros00101;10.1.99.30
```

Les Fonctions :

conf_site() "CREATION DES SITES" :

Saisir et définition du N° de site
demande d'entrée du numéro du site à créer
demande de l'adresse IP WAN du site à créer

définition des VLANs du site
créer les adresses IP des différents VLANs du site
le N° du site donnera le sous réseau qui différera pour chaque site
Exemple : vlan2 = 10 . n° site saisie . 2 . 0

création du fichier CSV du site
création d'un fichier CSV qui contient les adresses IP des différents VLANs et IP WAN.
Construction d'une ligne du fichier :
nom du vlan ; description du vlan ; ip du vlan ; masque du vlan ; gateway du vlan
Exemple d'une ligne du fichier :
vlan2;vlan2;10.1.2.0;255.255.255.0;10.1.2.254
IP;ip-wan;192.168.122.10;255.255.255.0

write_list_device() "Ecrire le fichier "liste des devices existants" :

Permet d'ajouter un device au fichier list_device.csv.
Ce fichier permet à la sauvegarde de savoir quel device existe et doit être sauvegardé.
Le fichier contient nom du device et son adresse ip.
Exemple : ros00101;10.1.99.30

read_site() "Lire et écrire la configuration d'un site dans une liste" :

Permet de lire le fichier CSV d'un site et met dans une liste et retourne cette liste.
Cette liste contient toutes les informations (vlan, ip, ...) du site

conf_ro() "Création des routeurs" :

Permet de créer un fichier de configuration pour les routeurs des sites.
demande du N° du site pour lequel on veut créer le routeur.
!!! ATTENTION sauf le site 1, site principale géré à part, car point de sortie vers Internet et qui doit connaître tout les autres sites. !!!

Saisir le n° du site à configurer
demande du N° du site à configurer

Création du nom du routeur
définition du nom du routeur suivant le N° du site : rosxxx01 où xxx est le N° du site

Récupération du template
on récupère le template du routeur dans une liste

récupération des valeurs du sites
on récupère les valeurs du site à configurer

Récupération des valeurs nécessaires à la modification du template
on récupère les valeurs du site dans des variables

Génération de la configuration
on met à jour les valeurs du template à adapter suivant les valeurs du site
Exemple : hostname, vlan2, vlan99 etc.…

Ecrire le fichier dans "config"
on écrit le fichier de configuration dans le dossier "config" du serveur FTP.

Ajouter le device à la liste de devices existant pour le backup
on écrit le nouveau routeur créé dans le fichier "liste_device.csv" pour la prochaine sauvegarde.

conf_sw_level1() et **conf_sw_level2()**

2 fonctions créées pour différencier les 2 niveau de switch, la structure est la même (idem aussi du routeur), on va juste chargé un template différent et récupérer l'adresse IP du nouveau switch.

conf_sw() "Création des configs des switchs" :

Permet de créer la configuration des switchs suivant un site et suivant le niveau du swith.

Saisir le N° du site à configurer
on de mande le N° du site à configurer

Saisir quel niveau de switch doit être créé
on demande le niveau du switch (1 ou 2)

Saisir quel numéro de switch doit être créé
si switch de niveau 2, alors on demande le n° du switch à créer (1 à 20)
on appelle soit conf_sw_level1 soit conf_sw_level2

script_save.py :

Permet de faire une sauvegarde de toutes les configurations des routeurs et switchs en production.
Suivant le fichier "list_device.csv", le script fera une sauvegarde sur le serveur FTP via une commande Cisco envoyé aux devices.

Les variables que l'on aura besoin dans tout le programme et qui seront modifiées aux besoin de chacun.

```
# Définition des variables globales (identique à celles de config.py)
ip_serv_ftp = "10.1.2.100"          # adresse ip du serveur FTP
dir_ftp = "/home/share/"            # répertoire de base du serveur FTP
dir_sav ="sav/"                     # répertoire pour les fichiers de backup
dir_log ="log/"                     # répertoire pour les fichiers de log

file_device = "list_device.csv"     # fichier qui liste les device existants
```

<u>Les Fonctions :</u>

test_ping() # TEST PING SI DEVICE ONLINE
Permet de tester si un device répond sur le réseau et donc si on peut lancer la sauvegarde.

executer_commande_cisco() # COMMANDE CISCO A EXECUTER

Permet d'envoyer une commande Cisco. Utilise la librairie NETMIKO.

(on peut utiliser Netmiko pour Juniper, Aruba ou autre voir
https://reseau.network/automatisation_netmiko/)

Définition paramètre connexion Cisco
On défini le device sur lequel sera envoyé la commande

Test et envoie de la commande Cisco
On envoie la commande et on retourne si OK ou non.

SCRIPT DE BACKUP DES SWITCHS ET ROUTEURS INSTALLES"
Corps du programme :

Créer la liste des devices à backuper
On récupère depuis list_device.csv tous les devices à sauvegarder

Récupérer l'heure
On récupère l'heure pour dater notre fichier de log de la sauvegarde.

Définition du nom de fichier de log
On défini le nom du fichier de log.

Pour chaque devices faire un backup FTP / SSH
On va pour tous les devices de la liste, vérifier si le devices est bien "on line" et envoyer la commande Cisco suivant son adresse ip .
En retour on met à jour le fichier de log.

Modification pour votre usage :

Dans mon exemple, le VLAN99 est le "VLAN de management" des switchs et routeurs, se sera avec ce VLAN que sera configuré les adresses IP des swicths.
J'utilise aussi le VLAN99 des routeurs pour faire la sauvegarde.

Le plus important :
- Créer vos propres templates devices (routeur et switchs) (exemples dans le dossier "template")
- Créer votre structure pour les sites (vlans, vpn, ip-wan…)

Dans le script "config.py" :
=> Modifier les variables globales suivant vos différents dossiers.

#CREATION DES SITE :

=> adapter, suivant votre structure de site, la section *# Saisir et définition du n° site*
=> adapter, suivant vos VLANs et autres, la section *# définition des VLANs du site*
=> ajouter les lignes à écrire dans le fichier du site dans la section *# création du fichier CSV du site.*
Techniquement :
On crée et construit des variables "string" pour chaque valeur de vlans, adresses, etc.
Exemple : vlan2_site = "10."+**site_number**+".2.0" *# définition du lan du VLAN2*
=> donc pour chaque VLANs seul "**site_number**" changera suivant notre saisie.
=> dans cette simple configuration on pourra avoir 255 sites différents.

Puis, on écrit ces variables une à une dans un fichier CSV, il faut alors ajouter autant de ligne que l'on aura de vlans. On finit toujours par un saut de ligne, sans oublier le " ;" pour la séparation du fichier CSV.

Exemple :

```
file.write("vlan2;vlan2;")  # nom du vlan + description
file.write(vlan2_site)   # ip du vlan
file.write(";255.255.255.0;")   # masque du vlan
file.write(vlan2_site_gw)  # gw du vlan
file.write("\n")   # saut de ligne
!!! pas de saut de ligne en fin de fichier !!!
```

 # CREATION DES ROUTEURS :

=> adapter, suivant vos VLANs et autres, la section *# Récupération des valeurs nécessaires à la modification du template*

=> ajouter les lignes à écrire dans le fichier du site dans la section *# Génération de la configuration*

Techniquement :

Récupération du template : on récupère le template du routeur que l'on met dans une liste.

=> config[] est une liste qui contient chaque ligne du template.

Récupération des valeurs du sites : on récupère chaque ligne du fichier site dans une liste.

=> valeur_site[] est une liste qui contient le fichier csv du site.

=> chaque indice de valeur_site[] correspond à une ligne du fichier csv du site

Récupération des valeurs nécessaires à la modification du template :

on crée des variables qui vont contenir 1 ligne du fichier "csv"

Exemple :

valeur_vlan2 = valeur_site[0]

=> valeur_vlan2 est une liste qui contient la ligne correspondante à l'indice de valeur_site[0]

=> ainsi chaque indice de valeur_vlan2[] va contenir les différentes valeurs nécessaire au vlan2.

Génération de la configuration : on va modifier les ligne du template suivant l'indice de la liste config[].

Exemple :

config[23]="ip address "+valeur_vlan2[4]+" "+valeur_vlan2[3]+"\n" # ip de la gw du vlan2

=> on modifie la ligne 24 du template (indice commençant à 0), en ajoutant du texte et les valeurs du vlan2, encore un fois, en suivant l'indice de valeur_vlan2[].

 # CREATION DES SWICTHS

Comme pour les routeurs, on lit les fichiers site.csv ou template que l'on met dans des listes.

Récupération du template du switch niveau1 *ou* # Récupération du template du switch niveau2

On récupère le template suivant le niveau du switch => config[]

Récupération des valeurs du sites

On récupère le fichier csv du site pour lequel on veut configurer le switch => valeur_site[]

On récupère les valeurs du site pour configurer l'adresse IP du switch => valeur_vlan99[]

Petite différence on construit l'adresse ip du swicth suivant son niveau et son numéro.

Exemple pour switch level2:

```
ip_lan_vlan99 = valeur_vlan99[2]          # récupération adresse ip du vlan99
ip_sw_vlan99=ip_lan_vlan99.split(".")     # découpage de l'adresse IP dans une liste
ip_sw_vlan99[3]=num_sw+1
ip_sw_vlan99[3]=str(ip_sw_vlan99[3])
```

#reconstitution adresse IP du switch
ip_sw = ip_sw_vlan99[0]+"."+ip_sw_vlan99[1]+"."+ip_sw_vlan99[2]+"."+ip_sw_vlan99[3]

Génération de la configuration

On génère la configuration

Exemple :

config[107]=" ip address "+ip_sw+" "+valeur_vlan99[3]+"\n" # config IP dans vlan99 management

Dans le script "script_save.py" :

=> Modifier les variables globales suivant vos différents dossiers.

=> Le script fait la sauvegarde via le fichier "list_device.csv", si ce fichier est modifié alors il faudra adapter la boucle sous *# Créer la liste des devices à backuper*, qui sert a extraire les adresses IP du fichier.

La sauvegarde se fait via une librairie externe "NETMIKO".
Sinon on peut faire la même chose via SSH sur un serveur Linux, pas de solution Windows testé.
(voir ligne de commande en commentaire)

Pour SSH, il faut aussi installer "SSHPASS" pour Linux (voir https://linux.die.net/man/1/sshpass).