

**Automatisation De La Gestion
des Switches Aruba
Par Ansible**

Ershad Ramezani

Table des matières

Introduction.....	3
Les commutateurs Aruba dans le parc.....	3
Limite de la situation actuelle.....	3
Solution proposée.....	4
Terminologie de différents NOS (Network Operating System).....	4
ArubaOS-Switch.....	5
aos-switch-ansible-collection.....	5
Exigences.....	6
Le REST API.....	7
Configuration du REST API sur les commutateurs Aruba.....	7
Plugins de connexions.....	7
Installation d'ansible.....	8
Installation de la collection.....	8
Le répertoire du projet.....	8
Group_vars et host_vars.....	8
Les variable d'inventaire dans le group_vars.....	9
Création du fichier ansible.cfg.....	10
SSH/CLI modules.....	10
Création du dossier playbooks.....	10
Sauvegarde des configurations.....	11
Mise à jour des images des commutateurs.....	13
Récupération des informations système des commutateurs.....	16
Configuration du syslog sur les commutateurs Aruba.....	19
Vérifier la configuration syslog existante :.....	19
Retirer la configuration syslog précédente.....	20
Configurer le syslog.....	20
Conclusion.....	22

Introduction

Au sein de ce projet, il est présent un total de 31 commutateurs Aruba, qui sont soit autonomes soit empilés, et qui sont répartis dans différents bâtiments. Ces commutateurs sont issus de deux ou trois modèles distincts, à savoir les Aruba2530F et les Aruba2930F. Ma mission consistait à automatiser la gestion de configuration de l'ensemble de ces commutateurs Aruba présents dans le parc.

Les commutateurs Aruba dans le parc

En utilisant les données de l'inventaire de GLPi, j'ai produit un rapport détaillé sur les commutateurs présents dans le parc informatique de notre client, en fournissant leur modèle, leur numéro de version et leur version logicielle actuelle.

D'après ce rapport, il y a 31 commutateurs de la marque Aruba parmi les 41 examinés. Parmi ces commutateurs, certains ont été regroupés en empilement, ce qui nous donne un total de 18 commutateurs, qu'ils soient individuels ou faisant partie d'un empilement.

Dans le parc informatique de notre client, on trouve deux modèles de commutateurs Aruba : l'Aruba2530F et l'Aruba2930F :

N°	Name	Model	Firmware
1	SW [REDACTED]	Aruba 2530-48G (J9775A)	YA.15.18.0011 YA.15.17
2	SW [REDACTED]	Aruba 2530-48G (J9775A)	YA.15.18.0011 YA.15.17
3	SW [REDACTED]	Aruba 2530-24G (J9776A)	YA.15.17 YA.15.18.0011
4	SW [REDACTED]	Aruba 2530-48G (J9775A)	YA.15.18.0011 YA.15.17
5	SW [REDACTED]	Aruba 2530-48G (J9775A)	YA.15.18.0011 YA.15.17
6	SW [REDACTED]	Aruba 2530-24G (J9776A)	YA.15.17 YA.15.18.0011
7	SW [REDACTED]	Aruba 2530-24G (J9776A)	YA.15.17 YA.15.18.0011
8	SW [REDACTED]	Aruba 2530-8G-PoE+ (J9774A)	YA.15.17 YA.15.18.0013
9	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.03.0004
10	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.01.0003 WC.16.03.0004
11	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.01.0004 WC.16.07.0003
12	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.07.0003
13	SW [REDACTED]	Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	WC.16.07.0003
14	SW [REDACTED]	Aruba 2930F 24G 4SFP+ (JL253A)	WC.16.07.0003
15	SW [REDACTED]	Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	WC.16.07.0003
16	SW [REDACTED]	Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	WC.16.07.0003
17	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.07.0003
18	SW [REDACTED]	Aruba 2930F 48G 4SFP+ (JL254A)	WC.16.07.0003
		Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	
		Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	
		Aruba 2930F 48G PoE+ 4SFP+ (JL256A)	

Limite de la situation actuelle

Dans ce domaine, plusieurs problèmes ont été constatés :

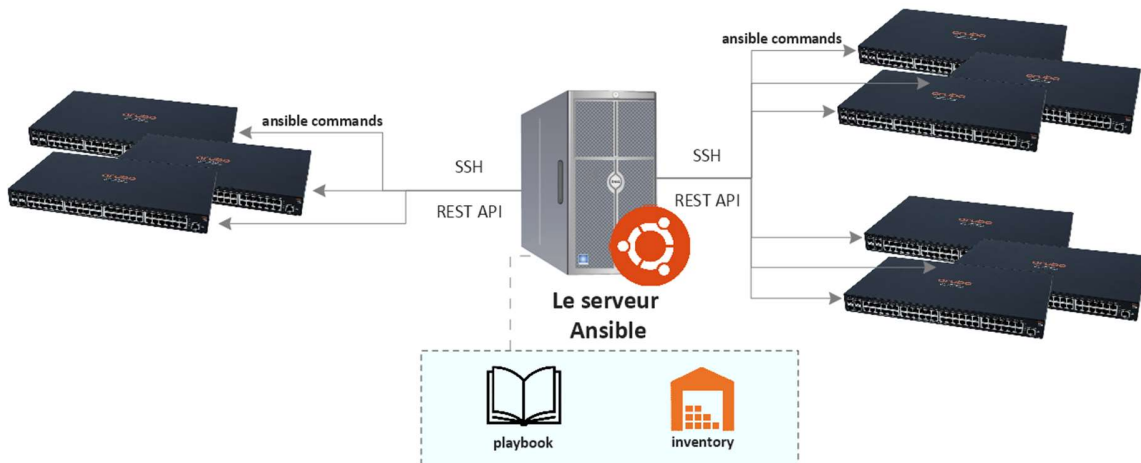
- Un manque de sauvegarde centralisée des configurations des commutateurs
- Une absence de documentation présentant les spécifications des matériels

- Des images obsolètes des commutateurs, datant de plus de 6 ans
- Le besoin d'un système d'automatisation pour appliquer des configurations sur l'ensemble des commutateurs.

Solution proposée

Lors de ma proposition, j'ai suggéré l'utilisation d'Ansible, un logiciel open source qui permet d'automatiser diverses tâches informatiques, telles que le provisionnement, la configuration et le déploiement d'applications.

Ensuite, j'ai planifié l'installation d'un serveur Ansible dans le réseau de notre client, ce qui m'a permis d'avoir accès à plusieurs fonctionnalités, telles que la mise en place d'un inventaire centralisé des équipements réseau, l'automatisation de la sauvegarde des configurations réseau, la sauvegarde automatique des fichiers OS des équipements réseau, ainsi que la mise à jour et l'homogénéisation des OS des équipements réseau.



Terminologie de différents NOS (Network Operating System)

Différents systèmes d'exploitation de réseau (NOS) sont utilisés pour les commutateurs chez HPE :

- Les systèmes d'exploitation pour les commutateurs réseau sont Comware, ProVision et ArubaOS.
- Les versions 5 et 7 de Comware sont actuellement utilisées dans les Datacenters et sur les commutateurs FlexNetwork/FlexFabric fabriqués par 3COM/H3C (appartenant à HPE).
- HP ProVision est le système d'exploitation réseau (NOS) utilisé dans tous les commutateurs HP ProCurve fabriqués par HP, maintenant connu sous le nom de HPE.
- Les commutateurs HP ProCurve ont été renommés en Aruba Switches après la construction de la nouvelle division Aruba par HPE. ArubaOS-Switch est une version améliorée et renommée du système d'exploitation ProVision pour ces nouveaux commutateurs construits par Aruba.
- ArubaOS-CX est un système d'exploitation récent, développé de manière indépendante, utilisé sur des plateformes telles que l'Aruba 8400X et 8320.

Voici une comparaison entre NOS (Network Operating System), le produit et le mark de HPE et Cisco :

NOS	Switch Portfolio Name Product Line	Brand Name
-----	------------------------------------	------------

CatOS	Catalyst	Cisco
IOS	Catalyst	Cisco
NX-OS	Nexus	Cisco
Comware	FlexNetwork/FlexFabric	3COM/H3C
Provision	ProCurve	HP (rebranded to ArubaOS)
ArubaOS-Switch	Aruba Switches	HPE
ArubaOS-CX	Aruba Swiches	HPE

ArubaOS-Switch

Les versions d'ArubaOS-Switch sont disponibles sur le site web officiel d'Aruba. Les versions commençant par "YA" sont destinées aux commutateurs 2530F, tandis que les versions commençant par "WC" sont destinées aux commutateurs 2930F.

<https://asp.arubanetworks.com/downloads>

The screenshot shows the Aruba Support Portal interface. At the top, there are navigation links for 'CONTACT US' and 'LOGIN'. Below the search bar, the page title is 'Software and Documents'. A search filter 'WC' is applied, showing 203 results. The results are sorted by 'Relevance'. Two software versions are displayed:

- WC.16.09.0022**: Released on Jan 10, 2023. Product: Aruba Switches (v16.09). File size: 28.5 MB.
- WC.16.11.0008**: Released on Nov 12, 2022. Product: Aruba Switches (v16.11). File size: 28.8 MB.

aos-switch-ansible-collection

Si l'on visite la page officielle d'Aruba sur GitHub, on peut y trouver des collections ainsi que des rôles spécifiques pour les produits Aruba. La collection "Aos-switch-ansible-collection" a été créée dans le but d'automatiser la gestion de la configuration des commutateurs Aruba qui sont équipés d'ArubaOS-switch.

<https://github.com/aruba>

Aruba, a Hewlett Packard Enterprise company

61 followers Santa Clara, California https://arubanetworks.com aruba-github@hpe.com Verified

Overview Repositories 42 Projects Packages People 4

Pinned

- aos-switch-ansible-collection** (Public)
 - Ansible collection for AOS-Switch switches.
 - Python 40 stars 8 forks
- aoscx-ansible-collection (Public)
 - Ansible collections for AOS-CX switches.
 - Python 17 stars 19 forks
- pycentral (Public)
 - Aruba Central Python SDK
 - Python 9 stars 17 forks
- aoscx-ansible-dcn-workflows (Public)
 - repository to hold Ansible workflows for AOS-CX Data Centers
 - Python 7 stars 4 forks
- aruba-central-ansible-role (Public)
 - Ansible Role for Aruba Central
 - Python 10 stars 7 forks
- clearpass-exchange-snippets (Public)
 - ClearPass Exchange integration snippets
 - 14 stars 13 forks

Il est proposé dans cette collection une sélection de 36 modules de configuration spécifiquement adaptés aux commutateurs AOS-Switch :

..	
arubaoss_aaa_accounting.py	Fix sanity pep8 (#51)
arubaoss_aaa_authentication.py	Fix sanity pep8 (#51)
arubaoss_aaa_authorization.py	Fix sanity pep8 (#51)
arubaoss_acl_policy.py	Fix sanity pylint (#44)
arubaoss_anyccli.py	Fix sanity pep8 (#51)
arubaoss_captive_portal.py	Fix sanity pep8 (#51)
arubaoss_command.py	Fix sanity pep8 (#51)
arubaoss_config.py	Fix sanity pep8 (#51)
arubaoss_config_bkup.py	Fix sanity pylint (#44)
arubaoss_dns.py	Fix sanity pep8 (#51)

Exigences

Voici les exigences à respecter pour utiliser cette collection :

- Avoir installé Python 2.7 ou 3.5+.
- Avoir installé Ansible 2.9.0 ou une version plus récente.

- Si vous souhaitez utiliser la collection avec l'image ArubaOS-Switch, vous devez disposer de la version 16.08 ou plus récente. Toutefois, cette exigence ne s'applique pas aux modules SSH/CLI (arubaoss_commad et arubaoss_config).
- Le REST doit être activé sur l'appareil AOS-Switch pour que la collection fonctionne correctement.

Le REST API

Les API REST sont un style architectural qui permet aux logiciels de communiquer entre eux sur un réseau. Elles utilisent des méthodes HTTP pour récupérer et publier des données entre un périphérique client et un serveur.

Grâce au protocole HTTP, les API REST facilitent la communication entre logiciels sur des appareils différents, même s'ils ont des systèmes d'exploitation et des architectures différents. Le client peut demander des ressources dans un langage que le serveur comprend et le serveur renvoie la ressource dans un format que le client accepte. Les formats de réponse courants sont JSON (JavaScript Object Notation), XML (Extensible Markup Language) ou texte, bien que de nombreuses API prennent en charge d'autres formats.

Configuration du REST API sur les commutateurs Aruba

Ces commandes permettent d'activer et visualiser la configuration par rapport à REST API :

```
SW- [redacted] (config)# web-management ssl
SW- [redacted] (config)# web-management plaintext
SW- [redacted] (config)# rest-interface
SW- [redacted] (config)#
SW- [redacted] (config)# show web-management

Web Management - Server Configuration

HTTP Access      : Enabled
HTTPS Access     : Enabled
SSL Port         : 443
Idle Timeout     : 600 seconds
Management URL   : http://h17007.www1.hp.com/device_help
Support URL      : http://www.arubanetworks.com/products/networking/
User Interface   : Improved

SW- [redacted] (config)# show rest-interface

REST Interface - Server Configuration

REST Interface      : Enabled
REST Operational Status : Up
REST Session Idle Timeout : 600 seconds
HTTP Access        : Enabled
HTTPS Access       : Enabled
SSL Port           : 443
```

Plugins de connexions

Les plugins de connexion d'Ansible sont utilisés pour se connecter aux hôtes cibles afin d'exécuter des tâches sur eux. Bien qu'Ansible possède plusieurs plugins de connexion, chaque hôte ne peut utiliser qu'un seul plugin à la fois. Dans le cadre de ce projet, nous allons utiliser les plugins de connexion network_cli et local d'Ansible.

Le plugin de connexion local d'Ansible, "Ansible.builtin.local", permet à Ansible d'exécuter des tâches sur le contrôleur Ansible lui-même plutôt que sur un hôte distant.

Le plugin de connexion "Ansible.netcommon.network_cli" fournit une connexion à des périphériques distants via SSH et implémente un shell CLI. Ce plugin de connexion est généralement utilisé pour envoyer et recevoir des commandes CLI aux périphériques réseau.

Installation d'ansible

Il est prévu que j'installe la version 2.3 du noyau Ansible sur une machine Ubuntu. L'installation d'Ansible peut être effectuée simplement en utilisant les trois commandes suivantes (ainsi que les commandes "apt update" et "apt upgrade") :

```
sudo apt update & sudo apt upgrade
sudo apt install software-properties-common
sudo apt-add-repository --yes --update ppa:ansible/ansible
sudo apt update
sudo apt install ansible -y
```

Le répertoire racine d'Ansible sera créé dans /etc/ansible, contenant par défaut le fichier ansible.cfg, le fichier hosts, et le dossier rôles vide. Le fichier ansible.cfg est utilisé pour stocker les paramètres de configuration d'Ansible. Le fichier hosts est l'inventaire par défaut de tous les hôtes pouvant être gérés par Ansible.

Installation de la collection

Pour installer la collection, il suffit d'utiliser la commande suivante :

```
ansible-galaxy collection install arubanetworks.aos_switch
```

Le répertoire du projet

Nous avons la possibilité de créer notre projet dans n'importe quel répertoire de notre choix. Dans ce cas, je vais le créer dans mon répertoire personnel en utilisant les commandes suivantes :

```
cd /home/ansible
mkdir nom_de_mon_projet
```

Le fichier hosts utilisé par Ansible est généralement au format .ini, et il contient tous les hôtes pouvant être gérés par Ansible. Les hôtes peuvent être spécifiés de deux manières : soit en utilisant un nom d'hôte unique (alias), soit en les regroupant dans un groupe.

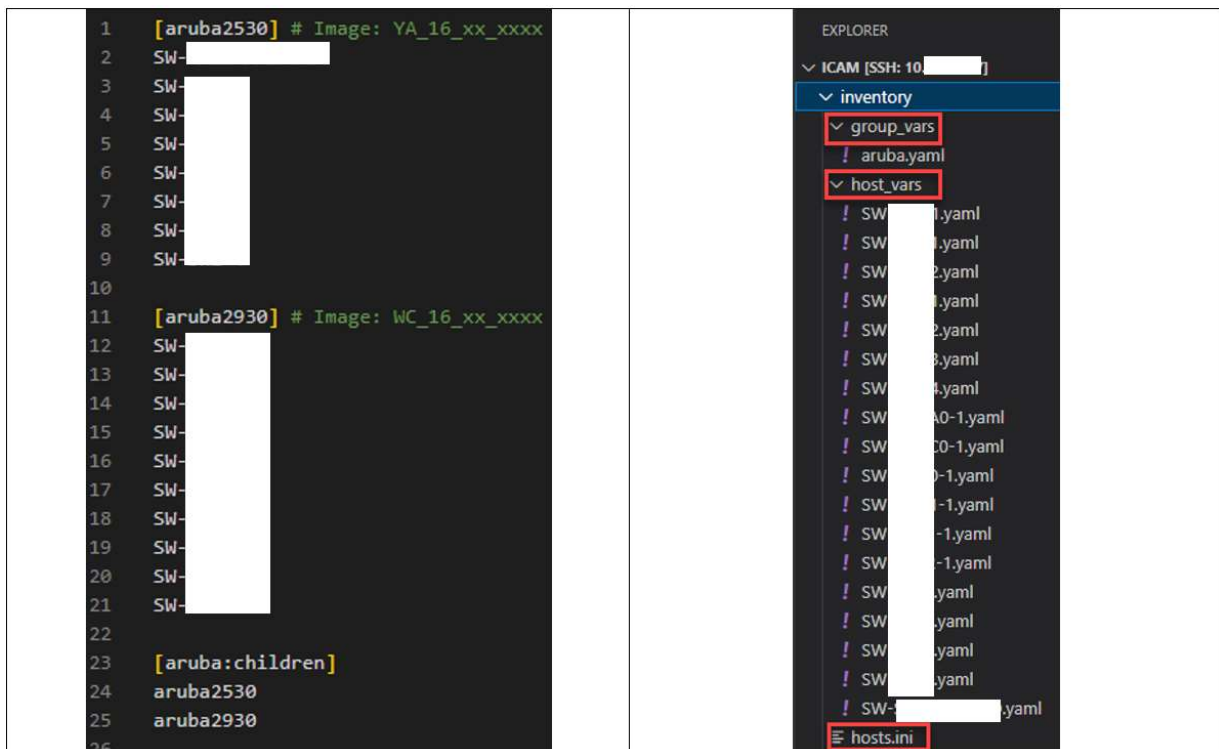
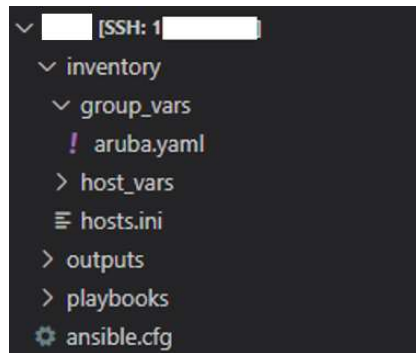
Dans mon cas, j'ai créé deux groupes pour les commutateurs Aruba dans mon parc, en fonction des deux modèles différents, ainsi qu'un groupe parent qui contient ces deux groupes comme enfants.

Group_vars et host_vars

En plus du fichier hosts qui contient les informations sur les hôtes et les groupes, j'ai créé deux répertoires nommés group_vars et host_vars. Le répertoire group_vars contient des fichiers pour les variables de groupe, tandis que le répertoire host_vars contient des fichiers pour les variables spécifiques à chaque hôte. Ces fichiers sont au format YAML et sont nommés en fonction des noms d'hôtes ou de groupes correspondants dans le fichier hosts.

Les fichiers host_vars contiennent la variable "ansible_host", qui permet de spécifier l'adresse IP de chaque hôte et de s'assurer qu'il est accessible par Ansible.

Le fichier "aruba.yaml" contient les variables communes à tous les commutateurs appartenant au groupe "aruba", telles que le nom d'utilisateur et le mot de passe pour la connexion SSH.



Les variable d'inventaire dans le group_vars

La seule variable qui doit être définie par l'hôte est "ansible_host". Toutes les autres variables sont définies dans le fichier "aruba.yaml", qui contient les variables communes à tous les commutateurs appartenant au groupe "aruba". Les variables dans ce fichier sont les suivantes :

- "ansible_user" : le nom d'utilisateur pour le commutateur au format texte brut.
- "ansible_password" : le mot de passe pour le commutateur au format texte brut.
- "ansible_connection" : doit être défini sur "local" pour utiliser les modules API REST, ou sur "network_cli" pour utiliser les modules SSH/CLI.
- "ansible_network_os" : doit toujours être défini sur "arubanetworks.aos_switch.arubaoss".

```
inventory > host_vars > ! SW[redacted].yaml
1  ansible_host: [redacted]

inventory > group_vars > ! aruba.yaml
1  ansible_user: [redacted]
2  ansible_password: [redacted]
3  ansible_connection: local
4  ansible_network_os: arubanetworks.aos_switch.arubaoss
```

Création du fichier ansible.cfg

Le fichier ansible.cfg est utilisé pour définir certaines configurations par défaut d'Ansible. Voici quelques exemples de paramètres que l'on peut définir dans ce fichier :

- "inventory" : permet de définir le chemin d'accès au fichier hosts.
- "host_key_checking" : permet de désactiver la vérification de la clé de l'hôte pour éviter que Ansible ne vérifie la clé publique de l'appareil. Cette vérification est mise en œuvre pour empêcher les attaques man-in-the-middle.
- "network_group_modules" : permet de modifier l'environnement avec NETWORK_GROUP_MODULES pour que Ansible reconnaisse le système d'exploitation des commutateurs Aruba.
- "command_timeout" : permet d'ajouter le paramètre "command_timeout" pour augmenter le délai d'expiration des commandes Ansible. Cette option peut s'avérer utile dans des situations où une commande peut prendre un certain temps pour s'exécuter, par exemple lors du téléchargement de fichiers sur de nombreux commutateurs.

Le fichier ansible.cfg doit être placé dans le répertoire de notre projet (par exemple, /home/ansible/nom_de_mon_projet).

```
ansible.cfg
1  [defaults]
2  inventory=/home/ansible/ICAM/inventory/hosts.ini
3  NETWORK_GROUP_MODULES=arubaoss
4  host_key_checking=false
5
6  [persistent_connection]
7  command_timeout=300
8
```

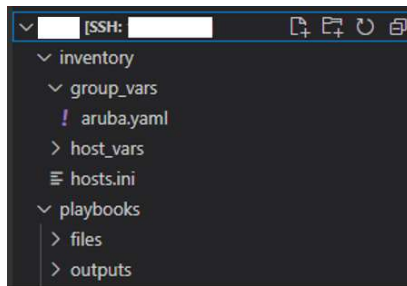
SSH/CLI modules

Pour pouvoir utiliser les modules "arubaoss_config" et "arubaoss_command" dans un playbook Ansible, l'accès SSH doit être activé sur les commutateurs. Cela diffère des autres modules qui utilisent plutôt la connexion par l'API REST.

Comme tous les commutateurs ont une connexion par défaut définie dans le fichier "group_vars" (avec "ansible_connection: local" dans le fichier "aruba.yaml"), il est nécessaire de spécifier la connexion "network_cli" dans chaque playbook où nous utilisons ces deux modules. Cela peut être fait en utilisant la variable "ansible_connection: network_cli".

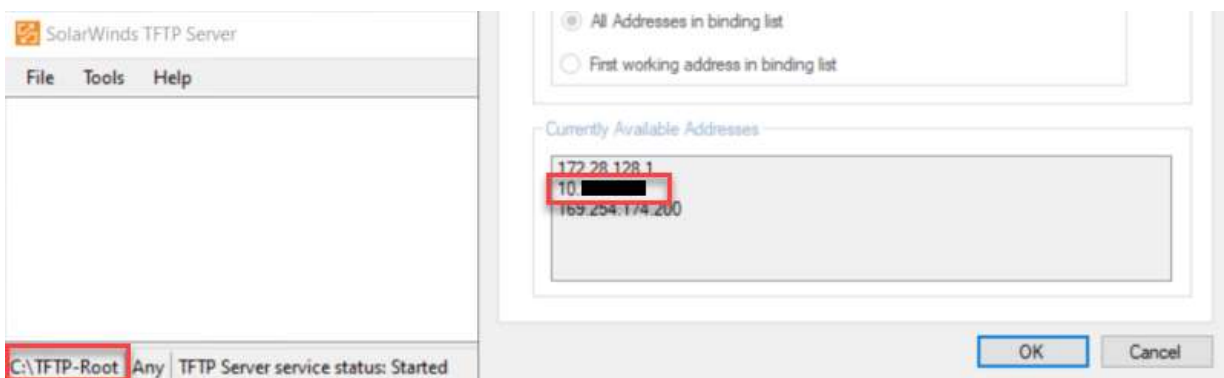
Création du dossier playbooks

J'ai créé un répertoire nommé "playbooks" à la racine de mon projet (nom_de_mon_projet/playbook) pour stocker les différents playbooks. En outre, j'ai créé deux autres répertoires à l'intérieur du répertoire "playbooks", "files" et "outputs", pour conserver les fichiers qui ne sont pas des playbooks.



Sauvegarde des configurations

Nous pouvons sauvegarder les configurations en utilisant le module `arubaoss_command`. Pour ce faire, il est nécessaire d'installer un serveur TFTP. J'ai installé le serveur SolarWinds TFTP sur un PC :



La commande pour sauvegarder la configuration en cours d'exécution depuis un commutateur Aruba vers un serveur TFTP est :

```
copy running-config tftp [adresse IP du serveur] [nom du fichier]
```

Il est possible de mettre en place une automatisation pour cette tâche en utilisant le playbook mentionné ci-dessous :

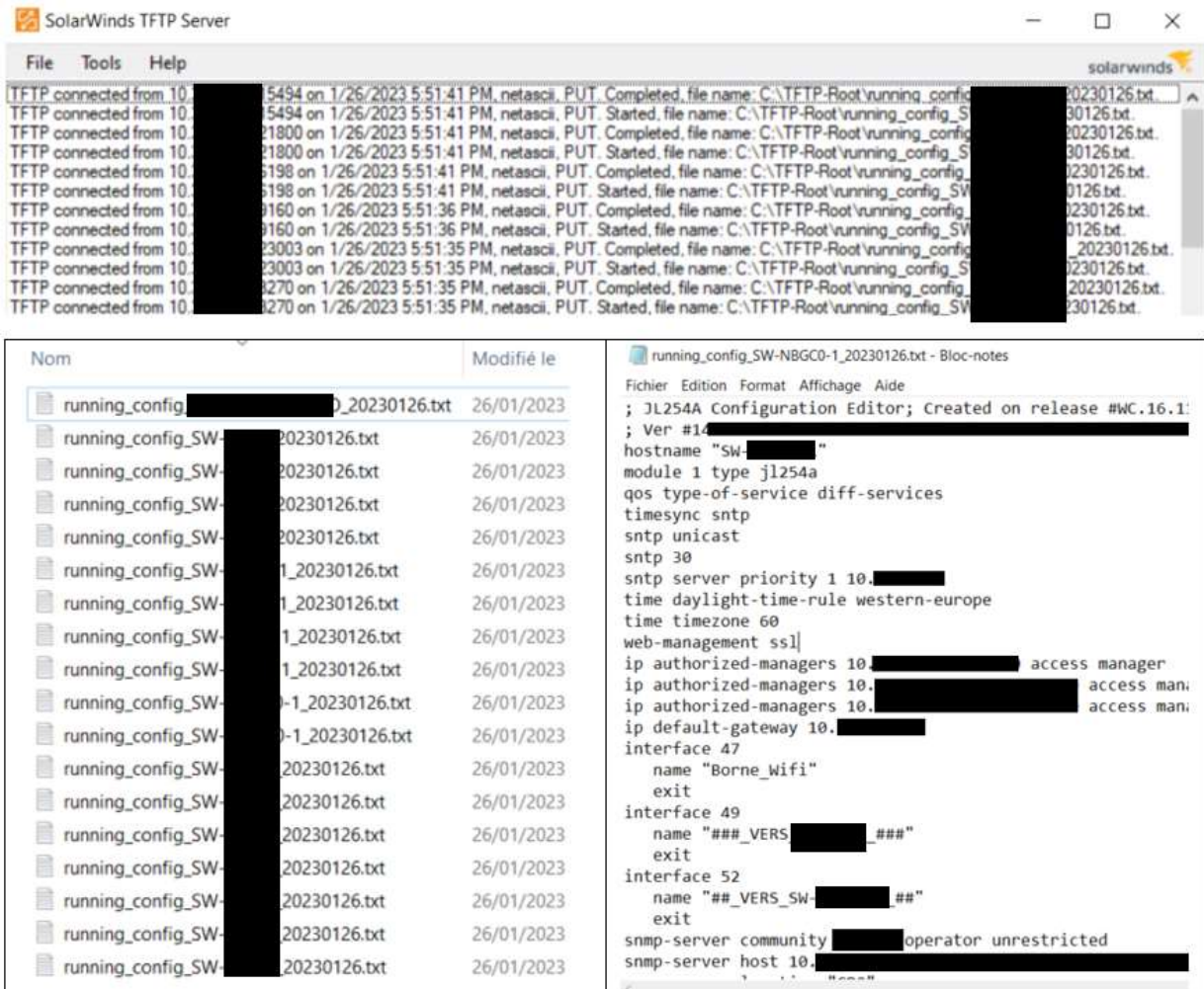
```

playbooks > ! 5.backup_running_config.yaml
1  - hosts: aruba
2    gather_facts: false
3    collections:
4      - arubanetworks.aos_switch
5    vars:
6      ansible_connection: network_cli
7      date: "{{ lookup('pipe', 'date +%Y%m%d') }}"
8      filename: "running_config_{{ inventory_hostname }}_{{ date }}.txt"
9      tftp_server: "10.172.28.128"
10   tasks:
11     - name: Backup running configurations
12       arubaoss_command:
13         commands:
14           - command: "copy running-config tftp {{ tftp_server }} {{ filename }}"
15

```

- Le code spécifie qu'il s'exécutera sur le groupe d'hôtes `aruba`, et il désactive la fonction `gather_facts`.
- Il définit certaines variables, telles que `ansible_connection`, `date`, `filename` et `tftp_server`.

Voici les enregistrements du log du serveur TFTP ainsi que les fichiers de sauvegarde :



Mise à jour des images des commutateurs

Toutes les images des équipements Aruba dans le parc sont inférieures à la version 16.08 et datent de plus de 6 ans. Elles ne sont pas compatibles avec les modules REST API car ceux-ci ne sont pas pris en charge sur ces versions.

J'ai utilisé le module arubaoss_command pour mettre à jour tous les commutateurs du parc afin de résoudre ce problème :

- Télécharger la version la plus récente des images et placez-les dans le répertoire principal du serveur TFTP.
- Les commutateurs Aruba disposent tous de deux mémoires flash qui permettent une récupération après une situation d'urgence. Il est possible de mettre à jour l'image de modèle Aruba2530F en utilisant la commande suivante :

**La version complète est disponible aussi
mais protéger par un mot de passe.**

**Merci de me contacter par email :
Ershad.ra@gmail.com**