|  |  |  |  |
| --- | --- | --- | --- |
|  | Follow Up Letter |  |  |
|  |  | |  |
|  | ### Objectif Global du Script  L'objectif du script est d'automatiser la gestion des adresses IP dans un réseau et de configurer des appareils, à la fois dans Cisco Packet Tracer et Ansible. Voici une vue d'ensemble de ce que fait chaque partie du script et comment elles s'intègrent pour atteindre cet objectif.  ### Structure du Script  Le script est maintenant divisé en modules pour améliorer la maintenabilité et la réutilisabilité. Voici les fichiers et leurs rôles respectifs :  1. \*\*`ip\_management/data.py`\*\* :  - Contient les données de configuration réseau pour chaque bureau et VLAN.  - Déclare un ensemble d'adresses IP déjà utilisées (`used\_ips`).  2. \*\*`ip\_management/ip\_functions.py`\*\* :  - Contient des fonctions pour gérer les adresses IP :  - `parse\_ip\_range(ip\_range)` : Parse une plage d'adresses IP et retourne les adresses de début et de fin.  - `generate\_ip\_list(start\_ip, end\_ip)` : Générer une liste d'adresses IP entre une adresse de début et une adresse de fin.  - `find\_available\_ip(bureau, vlan)` : Trouver une adresse IP disponible pour un bureau et un VLAN spécifiés.  3. \*\*`packet\_tracer/pt\_functions.py`\*\* :  - Contient une fonction pour configurer des appareils dans Packet Tracer :  - `configure\_packet\_tracer(router\_name, computer\_name, router\_ip, computer\_ip, subnet\_mask, gateway)` : Configure un routeur et un ordinateur avec les paramètres IP spécifiés.  4. \*\*`ansible/ansible\_functions.py`\*\* :  - Contient une fonction pour configurer des appareils via Ansible :  - `configure\_with\_ansible(host, playbook\_path)` : Configure un appareil en utilisant un playbook Ansible.  5. \*\*`main.py`\*\* :  - Point d'entrée principal du script. Utilise les fonctions des modules pour trouver une adresse IP disponible et configurer les appareils dans Packet Tracer et Ansible.  ### Détails des Fonctions et leur Rôle  ip\_management/data.py  - \*\*`data`\*\* : Une liste de dictionnaires contenant les informations réseau pour chaque bureau et VLAN, y compris la plage d'adresses IP.  - \*\*`used\_ips`\*\* : Un ensemble d'adresses IP déjà utilisées.  *ip\_management/ip\_functions.py*  - \*\*`parse\_ip\_range(ip\_range)`\*\* : Parse une chaîne de plage d'adresses IP et retourne les adresses de début et de fin.  - \*\*`generate\_ip\_list(start\_ip, end\_ip)`\*\* : Génère une liste d'adresses IP entre une adresse de début et une adresse de fin.  - \*\*`find\_available\_ip(bureau, vlan)`\*\* : Trouve une adresse IP disponible pour un bureau et un VLAN spécifiés.  **packet\_tracer/pt\_functions.py**  - \*\*`configure\_packet\_tracer(router\_name, computer\_name, router\_ip, computer\_ip, subnet\_mask, gateway)`\*\* : Configure un routeur et un ordinateur dans Packet Tracer avec les paramètres IP spécifiés.  *ansible/ansible\_functions.py`*  - \*\*`configure\_with\_ansible(host, playbook\_path)`\*\* : Exécute un playbook Ansible pour configurer un appareil réseau.  main.py`  - \*\*`main()`\*\* : Point d'entrée principal du script. Utilise `find\_available\_ip` pour trouver une adresse IP disponible et configure les appareils dans Packet Tracer et Ansible.  **Explication**  - \*\*Gestion des Adresses IP\*\* : Le script gère les adresses IP en utilisant `ip\_management/data.py` pour stocker les informations sur les sous-réseaux et les VLANs, et `ip\_management/ip\_functions.py` pour trouver une adresse IP disponible.  - \*\*Configuration des Appareils\*\* :  - \*\*Packet Tracer\*\* : Utilise `packet\_tracer/pt\_functions.py` pour configurer des appareils dans Packet Tracer.  - \*\*Ansible\*\* : Utilise `ansible/ansible\_functions.py` pour exécuter des playbooks Ansible et configurer des appareils.  **Conclusion**  Ce script est conçu pour automatiser la gestion des adresses IP et la configuration des appareils dans un réseau, en utilisant des outils comme Packet Tracer et Ansible. La modularité améliore la maintenabilité en séparant les différentes responsabilités dans des modules distincts. | |  |