UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea in INGEGNERIA ELETTRONICA

GESTIONE DELLA CONFIGURAZIONE DI UNO SWITCH TRAMITE LIBRERIE PYTHON PER NETWORK AUTOMATION

SWITCH CONFIGURATION MANAGEMENT USING PYTHON LIBRARIES FOR NETWORK AUTOMATION

Relatore:

Prof. **ENNIO GAMBI**

Tesi di Laurea di:

MICHELE MARMORÈ

Correlatore:

Ing. **ADELMO DE SANTIS**

A.A. **2020/ 2021**

Indice

Introduzione

La società in cui viviamo oggi è, da qualche decennio, proiettata sempre più in un mondo interconnesso dove quasi la totalità dei dispositivi di supporto all'uomo comunicano tra loro scambiando dati. Questo ha permesso a chiunque di approdare al centro di una rete dove è possibile fornire e ricevere informazioni di qualsiasi genere.

In principio la rete è stata concepita per uso professionale ed è stata sistematicamente aggiornata per accogliere man mano l'arrivo di nuovi dispositivi.

Oggi la società è totalmente dipendente dal network, sia in ambito professionale che privato.

Basti pensare che ogni dipendente di un'azienda possiede almeno un laptop che deve scambiare dati (ad esempio per e-mail, videoconferenze) con un destinatario che può banalmente essere un utente della stessa azienda, un server, una macchina per l'industria 4.0 che opera all'interno dell'azienda o il mondo esterno.

Allo stesso modo fuori dall'ambito professionale ogni persona possiede un PC, uno smartphone o un altro dispositivo domotico che ha bisogno di accedere al network per funzionare.

Vista l'enorme mole di traffico presente, si rende necessario un continuo aggiornamento dei link di connessione tra i vari nodi e dei dispositivi di rete che gestiscono ciò. La soluzione non consiste solo nell'incremento del numero dei dispositivi di gestione, ma anche nella configurazione degli stessi. Tuttavia, dato il numero dei dispositivi di rete è necessario l'impiego di automatismi.

Si introduce, quindi, il concetto di Network Automation. Automatizzare una rete significa consentire agli operatori di modificare, aggiornare, configurare una nuova rete con nuovi

dispositivi, non più manualmente come si è eseguito fino a pochi anni fa, ma automaticamente.

Così facendo si riducono o eliminano i problemi dovuti alla programmazione manuale quali velocità e suscettibilità ad errori.

Lo scopo del progetto è quello di gestire la configurazione di un dispositivo di rete mediante Network Automation tramite lo sviluppo di un codice Python e l'ausilio della libreria Netmiko.

Capitolo 1

Network Automation

1.1- Motivazioni e teoria del Network Automation

Uno dei maggiori problemi per i gestori di rete è la crescita dei costi IT (Information Technology) per le operazioni di manutenzione. Infatti, fino al 95% delle modifiche alla rete vengono eseguite manualmente, con costi operativi da 2 a 3 volte superiori al costo dell'infrastruttura fisica stessa. Una maggiore automazione IT, gestita centralmente e in remoto, è essenziale per le aziende per tenere il passo nel mondo digitale.

Il Network Automation è il processo di automazione della configurazione, gestione, test, distribuzione e funzionamento di dispositivi fisici e virtuali all'interno di una rete. Con le attività e le funzioni di rete quotidiane automatizzate e i processi ripetitivi controllati e gestiti automaticamente, la disponibilità dei servizi di rete migliora notevolmente[1].

Network Automation consente di accelerare la distribuzione delle applicazioni automatizzando il provisioning, la gestione della rete e della sicurezza[2].

Questa procedura è implementabile in qualsiasi rete. Le soluzioni basate su hardware e software consentono ai data center, ai fornitori di servizi e alle aziende di implementare l'automazione della rete per migliorare l'efficienza, minimizzare l'errore umano e ridurre le spese operative.[1]

Il Network Automation può essere implementato tramite codici di programmazione, come ad esempio Python (come nel caso di cui si parlerà in questo elaborato), oppure tramite API (Application Program Interface).

[1] https://www.cisco.com/c/en/us/solutions/automation/network-automation.html
[2] https://www.vmware.com/topics/glossary/content/network-automation

1.2-Network Automation in Python

Python è un linguaggio di programmazione di "alto livello" a differenza della maggior parte degli altri linguaggi, orientato a oggetti, adatto a sviluppare applicazioni distribuite, scripting, computazione numerica e system testing.

È un linguaggio multi-paradigma che ha tra i principali obiettivi dinamicità, semplicità e flessibilità. Supporta il paradigma object oriented, la programmazione strutturata e molte caratteristiche di programmazione funzionale e riflessione.

Le principali caratteristiche di Python sono le variabili non tipizzate e l'uso dell'indentazione per la sintassi delle specifiche, al posto delle più comuni parentesi.[3] Python permette di implementare il Network Automation tramite l'utilizzo di alcune librerie quali ad esempio "NETMIKO" o "telnet.lib".

Nel progetto eseguito si è utilizzata la libreria NETMIKO. Questa è una libreria open source progettata per semplificare la gestione SSH su un'ampia gamma di dispositivi di rete di vari fornitori, tra cui Cisco, Arista e Juniper Networks[4]. Poiché si è lavorato con dispositivi Huawei con protocollo di accesso SSH, per poter implementare Network Automation con Netmiko, si è dovuto ricorrere al protocollo di accesso linux SSH che è presente nella libreria. Netmiko permette anche di configurare con protocollo API e contiene inoltre una più vasta gamma di dispositivi per l'implementazione.

[3] https://it.wikipedia.org/wiki/Python
[4] https://www.networkcomputing.com/network-security/network-automation-netmiko

Capitolo 2

Cenni teorici sulle Virtual Local Area Network (VLAN)

2.1- VLAN

Per gestire il traffico dati tra molteplici utenti occorre una solida base strutturata in grado di organizzare e trasportare efficientemente l'informazione.

Le VLAN sono una applicazione molto importante nell'ambito del networking e di fatto trovano applicazione sia nelle architetture di rete semplici sia in quelle enterprise.

Una rete di notevole dimensione deve essere robusta ed organizzata.

Una VLAN è una tecnica che ci permette di realizzare tanti domini di broadcast differenti utilizzando solamente la configurazione di apparati di rete di livello 2 quali switch. Quello che di fatto si ottiene configurando una VLAN è che si consente la comunicazione tra alcune porte dei dispositivi di livello 2 e si bloccano le comunicazioni con le altre.

Le VLAN sono un'implementazione delle LAN e hanno lo scopo di risolvere uno dei principali problemi che caratterizza quest'ultime, ossia che in esse è presente un solo dominio di broadcast, perciò, quando viene trasmesso un frame broadcast questo viene instradato in tutte le porte raggiungendo, quindi, tutti i nodi della rete.

Data la natura del frame trasmesso, tutti i nodi devono elaborarlo e ciò comporta un notevole dispendio di risorse. Inoltre, un eventuale virus che affligge un nodo si propaga in tutti gli altri nodi del dominio di broadcast e questo è chiaramente un fenomeno da evitare. Man mano che la rete cresce la situazione peggiora.

Per dividere i domini di broadcast si può procedere nei seguenti modi:

1) Con interfacce di livello 3. Tra queste troviamo il router, un dispositivo di rete usato come interfacciamento tra sottoreti diverse, eterogenee e non, che lavorando a livello logico come nodo interno di rete si occupa di instradare i pacchetti dati fra tali sottoreti permettendone l'interoperabilità a livello di indirizzamento.

   Affinché la comunicazione avvenga, in tale architettura, si devono utilizzare i protocolli IPv4/IPv6.Alcune criticità si riscontrano nel costo e nella complessità di debug. Infatti, i dispositivi di livello 3 sono più costosi, hanno bisogno di prestazioni maggiori e nel caso in cui incorra una problematica, non è facile eseguire il debug e identificarne la causa.

2) Interfacce di livello 2 con la tecnica delle VLAN. Questa tecnologia si basa sull'utilizzo di switch ossia dispositivi che gestiscono il flusso di dati attraverso una rete, trasmettendo un frame dati ricevuto solo a uno o più dispositivi per i quali esso è destinato. Ogni dispositivo collegato in rete a uno switch può essere identificato dal suo indirizzo MAC, il che permette di dirigere il flusso del traffico massimizzando la sicurezza e l'efficienza della rete.

   L'utilizzo di questa tecnica consente di configurare in maniera opportuna una serie di switch in modo da poter creare più domini di broadcast che si estendano su più switch anche geograficamente dislocati in punti diversi fra di loro.

   Di fatto si ottiene ciò che segue:

   a) ogni VLAN è un domino di broadcast differente;
   b) posso costruire VLAN con apparati geograficamente dislocati;
   c) gli apparati di livello 2 (switch) permettono lo scambio di frame solo all'interno delle VLAN e non tra VLAN diverse;

d) i router sono in grado di comunicare con le diverse VLAN e di far colloquiare tra di loro anche nodi che appartengono a VLAN diverse

Nella “Figura 1”[5] è riportato un esempio di rete costruita con le VLAN

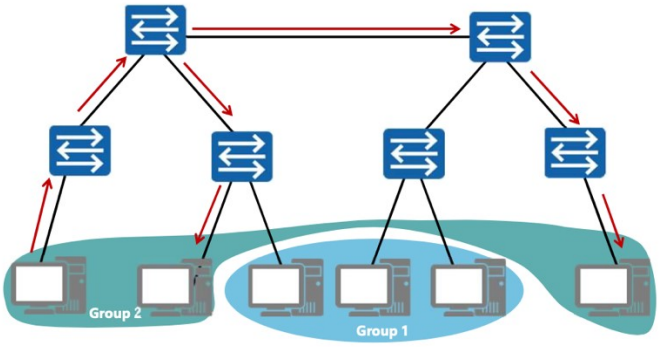

Figura 1. Topologia base di unaVLAN

In seguito in questo elaborato si approfondirà la seconda modalità di divisione dei domini di broadcast, VLAN.

Uno switch, per dividere i diversi domini di broadcast deve identificare un frame come appartenente ad una specifica VLAN piuttosto che ad un’altra.

Per comprendere meglio il meccanismo alla base di questa tecnica, di seguito nella “Figura 2” è riportata la struttura di un frame VLAN.

[5] Copyright © 2019 Huawei Technologies Co., Ltd. All rights reserved.

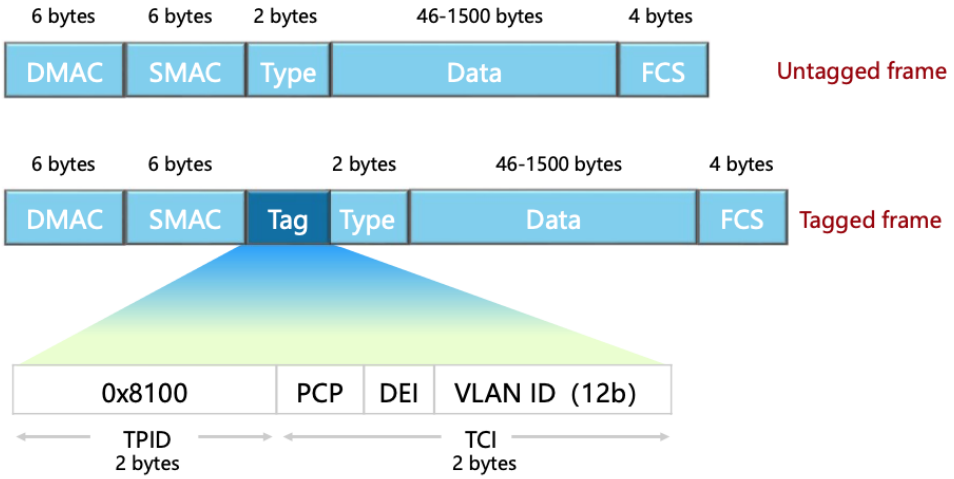

Figura 2. Frame Vlan

Si possono distinguere due tipi di frame che sono quelli tagged ed untagged. Tale distinzione è essenziale per descrivere ciò che accade alle porte di uno switch quando arriva un frame ethernet da un utente appartenente ad una specifica VLAN.

I frame tagged contengono un attributo che consente di identificare la loro VLAN di appartenenza e che prende il nome di "TAG". Il formato del TAG VLAN contiene un identificatore di protocollo TAG (TPID) e le informazioni di controllo TAG (TCI) associate. Il TPID viene utilizzato per identificare il frame come frame con TAG, che attualmente si riferisce solo al formato TAG IEEE 802.1Q, per il quale viene utilizzato un valore di 0x8100 per identificare questo formato. Il TCI contiene campi associati al tipo di formato TAG.

Per quanto riguarda le porte, nel mondo delle VLAN abbiamo tre possibili configurazioni:

1) porte Access;
2) porte Trunk;
3) porte Hybrid.

Porte Access

Sono utilizzate per connettere gli host alla VLAN. Scambiano frame di tipo untagged sia in ingresso che in uscita. Di fatto, un frame viene taggato dallo switch per essere istradato, successivamente viene rimosso il TAG prima di essere inviato all'host di destinazione.

Porte Trunk

Sono utilizzate per eseguire connessioni tra gli switch.

Sul collegamento Trunk vengono instradati tutti i frame, anche appartenenti a VLAN differenti, distinguendoli attraverso il loro VLAN ID.

Porte Hybrid

Sono le configurazioni di default presenti negli apparati in questione quali switch Huawei. Sono porte che possono comportarsi come tipo "Access" o "Trunk" a seconda della configurazione che si effettua e consentono di definire in modo molto puntuale quali frame far transitare in modo tagged o untagged.

Il mondo VLAN è legato alle interfacce, ad ognuna delle quali è associato un "port VLAN ID" (PVID). Questo attributo è contenuto all'interno del TAG generato per i frame VLAN, precisamente nel campo VLAN ID (ultimi 12 bit).

Il PVID è fondamentale per il corretto funzionamento delle VLAN.

La topologia illustrata in "Figura 3"[6] mostra il protocollo di comunicazione di una rete VLAN con porte Access e Trunk.

[6]

Host A ed Host C appartengono alla VLAN 10 ed hanno quindi un PVID10, mentre Host B ed Host D alla VLAN 20 con PVID20, dunque, si hanno 2 domini di broadcast differenti.

Quando Host A invia un frame di tipo broadcast, questo è non taggato fino all'interfaccia PVID10, dove lo switch inserisce il TAG per la trasmissione. Ora SWA deve inoltrare il frame a SWB poiché Host C, che è nello stesso domino di broadcast di Host A, è collegato a SWB. Il collegamento Trunk tra i due switch è assegnato ad una VLAN nativa (in questo caso VLAN 10 con PVID10). Poiché il collegamento Trunk ha la VLAN nativa con stesso PVID del frame da trasmettere, SWA prima di trasmettere a SWB toglie il TAG al frame che verrà reinserito da SWB all'interfaccia PVID10. Quindi SWB vede che nella tabella dei MAC address della VLAN 10 è presente Host C ed inoltra il frame togliendo il TAG.

Quando invece Host B trasmette un frame di tipo broadcast, l'interfaccia access non cambia il funzionamento, contrariamente a quella Trunk.

In questo caso, poiché il PVID di Host B non coincide con il PVID nativo del Trunk, il frame verrà trasmesso a SWB taggato. Quindi SWB riceve il frame con TAG riferito alla interfaccia PVID 20 ed inoltra ad Host D.

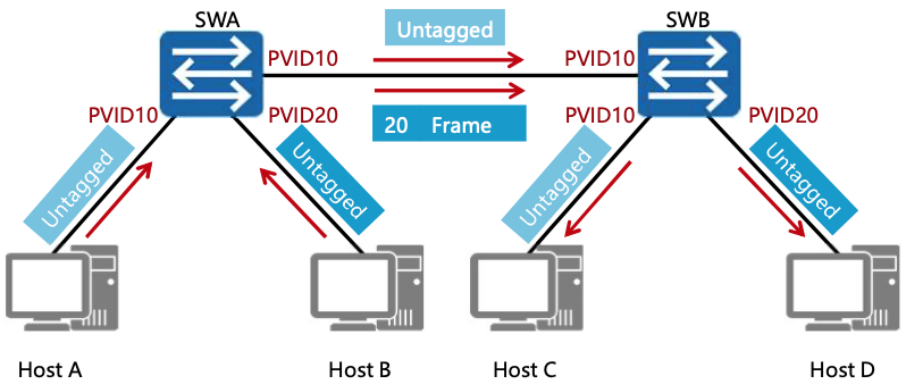

Figura 3. Implementazione delle VLAN

Nel progetto eseguito verranno utilizzate VLAN solo con porte di tipo access in quanto la topologia considerata non necessita di altro tipo. Ai fini del progetto, la scelta sarebbe potuta ricadere anche su porte di tipo Hybrid. Queste permettono di definire in modo puntuale quali frame trasmettere con il TAG e quali senza. Quindi in base alla configurazione si possono utilizzare le Hybrid come Access indicandole come "untagged" oppure come Trunk. Una importante differenza tra Trunk e Hybrid è che tutti i frame nella seconda viaggiano con il TAG indipendentemente dalla VLAN nativa.

Capitolo 3

Implementazione e verifica sul simulatore

3.1- Descrizione della topologia

Nel capitolo corrente si parla delle caratteristiche tecniche dello sviluppo del progetto. L'ambiente di lavoro attuale è di tipo virtuale, il software utilizzato è "eNSP", una piattaforma grafica di simulazione reti sviluppata da Huawei. Attraverso la simulazione di apparecchiature di rete reali, questa piattaforma consente ai professionisti e ai clienti ICT di acquisire rapidamente familiarità con i prodotti Huawei Datacom, comprendere e padroneggiare il funzionamento e la configurazione dei prodotti correlati e migliorare la rete di imprese ICT.[7]

Nella "Figura 4" è riportata la topologia di partenza e tutti i dispositivi utilizzati nella simulazione.

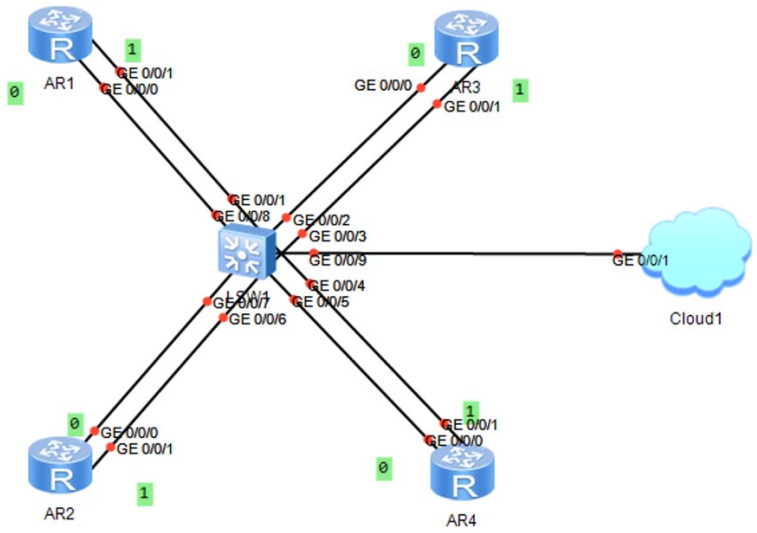

Figura 4. Topologia di partenza

[7] https://www.mindlessnerds.com/guide/huawei-ensp-download-install-fix-how-to/

Come visibile è stato utilizzato uno switch (Huawei S5700), quattro router (Huawei AR1220), ed il dispositivo cloud.

Quest'ultimo è fondamentale per l'esito del progetto, è un dispositivo che consente la connessione con la rete fisica ed è stato utilizzato per effettuare l'accesso da remoto eseguendo così la configurazione automatica dello switch.

Tutti i collegamenti sono di tipo Gigabit-Ethernet, compresa la connessione al cloud.

Dalla topologia di partenza (Figura 4) si vogliono ottenere le due topologie illustrate nella Figura 5.

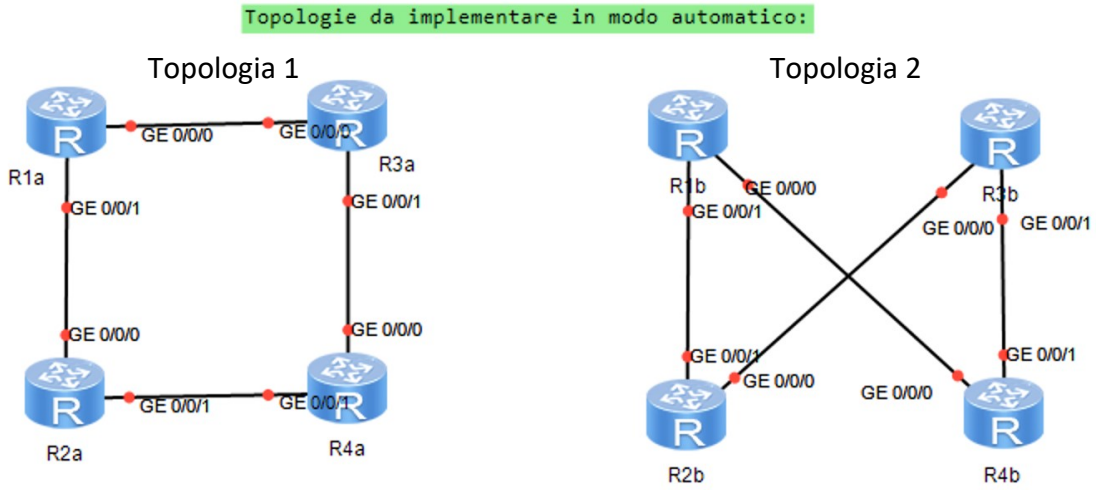

Figura 5. Topologie da implementare

3.2-Descrizione del codice sorgente

Con tale obiettivo è stato deciso di procedere come illustrato di seguito.

È necessario, per poter eseguire una configurazione automatica, che l'utente programmatore riesca ad accedere al dispositivo. Per fare ciò è stata assegnata una configurazione di base allo switch (Huawei S5700). È stato creato un utente "SSH" così da permettere l'accesso da remoto togliendo il vincolo dalla porta console.

Successivamente, per proiettare lo switch sulla rete esterna, è stata configurata una VLAN (VLAN2), assegnata alla porta Gigabit-Ethernet 0/0/9 che collega direttamente il dispositivo al cloud. L'assegnazione dell'indirizzo IP è stata fatta tramite DHCP.

Di seguito è riportato il codice che illustra la procedura della configurazione appena descritta.

```
<Huawei>undo terminal monitor
<Huawei>system-view
Enter system view, return user view with Ctrl+Z.
[Huawei]stelnet server enable
[Huawei]rsa local-key-pair create
The key name will be: Huawei_Host
The range of public key size is (2048 ~ 2048).
NOTES: If the key modulus is greater than 512,
       it will take a few minutes.
Input the bits in the modulus[default = 2048]:2048
Generating keys...
..............+++++
.........................++
....++++
...........++

[Huawei]aaa
[Huawei-aaa]local-user admin password cipher 123456
[Huawei-aaa]local-user admin privilege level 3
```

```
[Huawei-aaa]local-user admin service-type ssh
[Huawei-aaa]
[Huawei-aaa]quit
[Huawei]ssh user admin authentication-type password
[Huawei]user-interface vty 0 4
[Huawei-ui-vty0-4]authentication-mode aaa
[Huawei-ui-vty0-4]screen-length 40
[Huawei-ui-vty0-4]idle-timeout 10
[Huawei-ui-vty0-4]user privilege level 3
[Huawei-ui-vty0-4]history-command max-size 200
[Huawei-ui-vty0-4]quit
[Huawei]ssh user admin service-type all
[Huawei]ssh user admin authorization-cmd aaa
[Huawei] Scp server enable
[Huawei]dhcp enable
[Huawei]Vlan 2
[Huawei]int Vlanif 2
[Huawei-Vlanif2]ip address dhcp-alloc
[Huawei-Vlanif2]quit
[Huawei]dis int Vlanif 2
Vlanif2 current state : UP
Line protocol current state : UP
Last line protocol up time : --
Description:
Route Port,The Maximum Transmit Unit is 1500
Internet Address is allocated by DHCP, 10.211.55.8/24
Current system time: --
Input bandwidth utilization  : --
Output bandwidth utilization : --
[Huawei-Vlanif2]quit
[Huawei]int gig 0/0/9
[huawei-GigabitEthernet0/0/9]port link-type access
[huawei-GigabitEthernet0/0/9]port default vlan 2
```

3.3- Descrizione del codice Python

Assegnata la configurazione base, si è passati allo sviluppo del codice di programmazione automatica.

A tale scopo è stato utilizzato il linguaggio di programmazione Python con l'ausilio delle librerie Netmiko (principalmente) e loggin.

L'ambiente di sviluppo è PyCharm, un ambiente integrato (IDE) Python dedicato, che fornisce una vasta gamma di strumenti essenziali per gli sviluppatori, che permettono di creare un ambiente conveniente per lo sviluppo produttivo di Python, web e data science.[8].

La libreria Netmiko è un modulo che crea interfacce e metodi per interagire con i dispositivi di rete e supporta al suo interno un'ampia gamma di dispositivi. Nel caso in questione, è stata utilizzata per sostenere una connessione di tipo SSH.[9]

La libreria Logging, invece, è stata utilizzata per trasferire i dati di accesso alla funzione implementata da Netmiko.

Di seguito è illustrato il codice sviluppato per la configurazione automatica delle VLAN.

Nella "Figura 6" è illustrata la parte iniziale del codice, dove viene implementata la procedura di accesso allo switch da configurare.

Nelle righe 1-3 vi è la dichiarazione delle librerie utilizzate quali "Netmiko" e "Logging".

[8] https://www.jetbrains.com/help/pycharm/quick-start-guide.html
[9] https://github.com/ktbyers/netmiko

In seguito, è stato deciso di creare un file (“logger.txt”) all’interno del quale è salvato lo storico dei comandi eseguiti dal programma, così da avere un feedback visivo della configurazione effettuata.

Nelle righe 6-17 è dichiarata una variabile “host1” la quale contiene le credenziali di accesso dello User SSH configurato di base come indicato sopra.

Infine, si è deciso di dare un feedback della avvenuta connessione al dispositivo (riga 20).

```
1  from netmiko import Netmiko
2
3  import logging
4  logging.basicConfig(filename="test.log", level=logging.DEBUG)   # It will log all reads and writes on the SSH channel
5  logger = logging.getLogger("netmiko")
6  host1 = {                                                       # Enter Device information
7
8      "host": "10.211.55.8",
9
10     "username": "admin",
11
12     "password": "123456",
13
14     "device_type": "huawei",
15
16     "global_delay_factor": 0.1,                                   # Increase all sleeps by a factor of 1
17 }
18 net_connect = Netmiko(**host1)
19
20 print("Connected to:", net_connect.find_prompt())               # Display hostname
```

Figura 6. Comando di accesso allo switch

Nella “Figura 7” è invece indicata la procedura di configurazione automatica della rete.

Per fare ciò è stato deciso di procedere in maniera iterativa. Tale modo permette di raggiungere, dalla topologia di partenza, oltre che le 2 topologie richieste, anche qualsiasi altra si desideri. Tale decisione è giustificata dal fatto che la scelta di più topologie finali di arrivo implica l’applicazione di codici Python differenti.

La procedura di configurazione si può definire “step by step”, infatti, partendo dalla riga 24 si procede in ordine, come segue:

1) definire la volontà di creare o configurare VLAN;

2) numerazione della VLAN;

3) definizione del numero di porte da assegnare alla VLAN corrente;

4) definizione del numero di porta da assegnare (es 0/0/1 o 0/0/2 etc.).

Il tutto è contenuto in un ciclo iterativo “WHILE” che consente, una volta finita la prima configurazione, di iniziarne una nuova o terminare con il logout dal dispositivo.

Inizialmente viene richiesto se creare o configurare una VLAN. L’obiettivo è quello di configurarne una già esistente, ad esempio aggiungendo nuove porte, oppure crearne un’altra generando, quindi, un nuovo dominio di broadcast.

In seguito, si richiede il numero della VLAN su cui si vuole eseguire la configurazione. Dopodiché è possibile dichiarare il numero di porte che si vuole assegnare alla VLAN corrente e successivamente tramite un ciclo iterativo di tipo “FOR” si inseriscono i numeri delle porte da assegnare.

Nelle righe 46-48 avviene la decisione di proseguire la configurazione o terminare con l’operazione di logout.

```
22  print("starting config")
23
24  while True:
25      print("do you want to create Vlan?")                          #Creation vlan
26      risposta= str(input("yes or no?"))
27
28      if risposta == "yes":
29          vlan = int(input("insert a Vlan to create "))
30          comando_vlan = "vlan " + str(vlan)
31          output = net_connect.send_config_set(comando_vlan, delay_factor=1.5)  # Run set of commands
32          vlan_N= vlan
33      else:
34          print("do you want to config a Vlan?")                    #Select vlan to config
35          risp2 = str(input("yes or no"))
36          if risp2 == "yes":
37              vlan_N= int(input("insert Vlan number"))
38          else:
39              break
40      N = int(input("how much port do you want config?"))                    #Define N° port
41      for x in range(N):                                                     #Port assignment vlan
42          port= int(input("insert port number"))
43          command_loop = "int gig 0/0/" + str(port), "port link-type access", "port default vlan " + str(vlan_N) #Command
44
45          output = net_connect.send_config_set(command_loop, delay_factor=.5)  # Run set of commands
46      exit_mode=str(input("do you want to exit config?(yes or no)"))
47      if exit_mode=="yes":
48          break
```

Figura 7. Implementazione della configurazione (step by step)

Nella “Figura 8” è illustrata la fine della configurazione con la richiesta del feedback visivo riguardo la lista di configurazione delle VLAN (righe 49-54) e l’operazione di logout (riga 57).

```
49  print("do you want display VLAN config?")                  #Display vlan configuration
50  risp3= str(input("yes or no"))
51  if risp3 == "yes":
52      dis_cmd= "dis port vlan"
53      output = net_connect.send_config_set(dis_cmd, delay_factor=.5)
54      print(output)
55  else:
56      print("END CONFIGURATION")
57      net_connect.disconnect()                                # Disconnect from Session
```

Figura 8. Richiesta del display vlan e logout

3.4-Simulazione Topologia 1

La “Figura 9” illustra l’output monitor della IDE PyCharm dove vengono conferiti gli input necessari alla configurazione. Quest’ultima inizia con l’accesso conferendo un feedback dell’effettivo login. Successivamente, fino al termine della configurazione, vengono create le rispettive VLAN (numerate 10 20 30 40) ed assegnate le porte corrispondenti alla topologia da raggiungere.

Poiché la seguente è la prima configurazione in assoluto, eccetto quella di base, non è visibile la procedura di modifica della configurazione.

```
Connected to: <SWITCH1>
starting config
do you want to create Vlan?
yes or no?yes
insert a Vlan to create 10
how much port do you want config?2
insert port number1
insert port number2
do you want to exit config?(yes or no)no
do you want to create Vlan?
yes or no?yes
insert a Vlan to create 20
how much port do you want config?2
insert port number3
insert port number4
do you want to exit config?(yes or no)no
do you want to create Vlan?
yes or no?yes
insert a Vlan to create 30
how much port do you want config?2
insert port number5
insert port number6
do you want to exit config?(yes or no)no
do you want to create Vlan?
yes or no?yes
insert a Vlan to create 40
how much port do you want config?2
insert port number7
insert port number8
do you want to exit config?(yes or no)yes
```

Figura 9. Esecuzione su software

Al termine di ogni porzione viene richiesto se continuare o meno la configurazione, in caso di fine si può scegliere se visionare l'assegnazione delle porte appena eseguita oppure eseguire il logout. La funzione appena descritta è riportata in "Figura 10".

```
do you want to exit config?(yes or no)yes
do you want display VLAN config?
yes or noyes
system-view
Enter system view, return user view with Ctrl+Z.
[SWITCH1]dis port vlan
Port                    Link Type    PVID  Trunk VLAN List
--------------------------------------------------------------------------------
GigabitEthernet0/0/1    access       10    -
GigabitEthernet0/0/2    access       10    -
GigabitEthernet0/0/3    access       20    -
GigabitEthernet0/0/4    access       20    -
GigabitEthernet0/0/5    access       30    -
GigabitEthernet0/0/6    access       30    -
GigabitEthernet0/0/7    access       40    -
GigabitEthernet0/0/8    access       40    -
GigabitEthernet0/0/9    access       2     -
GigabitEthernet0/0/10   hybrid       1     -
GigabitEthernet0/0/11   hybrid       1     -
GigabitEthernet0/0/12   hybrid       1     -
GigabitEthernet0/0/13   hybrid       1     -
GigabitEthernet0/0/14   hybrid       1     -
GigabitEthernet0/0/15   hybrid       1     -
GigabitEthernet0/0/16   hybrid       1     -
GigabitEthernet0/0/17   hybrid       1     -
GigabitEthernet0/0/18   hybrid       1     -
GigabitEthernet0/0/19   hybrid       1     -
GigabitEthernet0/0/20   hybrid       1     -
GigabitEthernet0/0/21   hybrid       1     -
GigabitEthernet0/0/22   hybrid       1     -
GigabitEthernet0/0/23   hybrid       1     -
GigabitEthernet0/0/24   hybrid       1     -
[SWITCH1]return
<SWITCH1>
```

Figura 10. Fine della configurazione con feedback e logout.

Come si può notare ora lo switch ha una configurazione al suo interno che rispecchia la topologia riportata di seguito (Figura 11).

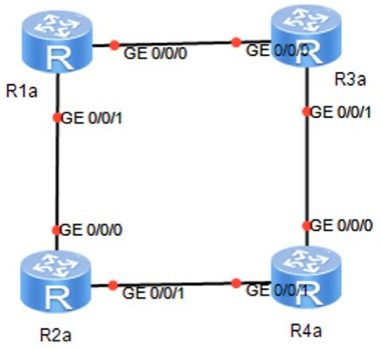

Figura 11. Topologia 1

3.5-Simulazione Topologia 2

La seconda configurazione, illustrata in “Figura13”, è volta a raggiungere la topologia 2 “Figura 12” partendo dalla topologia 1, implementando, quindi, la funzione di modifica delle VLAN.

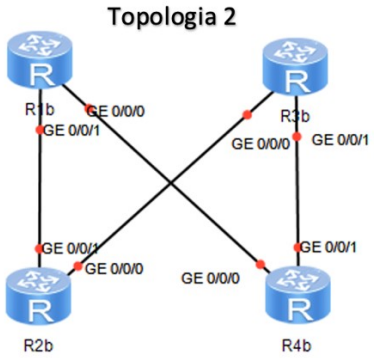

Figura 12. Topologia 2

```
Connected to: <SWITCH1>
starting config
do you want to create Vlan?
yes or no?no
do you want to config a Vlan?
yes or noyes
insert Vlan number10
how much port do you want config?2
insert port number1
insert port number5
do you want to exit config?(yes or no)no
do you want to create Vlan?
yes or no?no
do you want to config a Vlan?
yes or noyes
insert Vlan number30
how much port do you want config?1
insert port number2
do you want to exit config?(yes or no)yes
do you want display VLAN config?
yes or noyes
system-view
Enter system view, return user view with Ctrl+Z.
[SWITCH1]dis port vlan
Port                    Link Type    PVID  Trunk VLAN List
--------------------------------------------------------------------------------
GigabitEthernet0/0/1    access       10    -
GigabitEthernet0/0/2    access       30    -
GigabitEthernet0/0/3    access       20    -
GigabitEthernet0/0/4    access       20    -
GigabitEthernet0/0/5    access       10    -
GigabitEthernet0/0/6    access       30    -
GigabitEthernet0/0/7    access       40    -
GigabitEthernet0/0/8    access       40    -
GigabitEthernet0/0/9    access       2     -
GigabitEthernet0/0/10   hybrid       1     -
```

Figura 13. Modifica assegnazioni delle Vlan esistenti per TOPO2

Capitolo 4

Test su hardware reale

4.1-Caratteristiche del laboratorio:

Il test su hardware reale è stato eseguito nel laboratorio Huawei presente nel dipartimento di ingegneria dell'Università Politecnica delle Marche.

Il laboratorio Huawei comprende un insieme di apparecchiature di rete multivendor, con un focus specifico su Huawei. Sono presenti 6 router di fascia media, 3 switch (Huawei S5700), access controller, firewall e tutto il necessario per consentire agli studenti di sperimentare in un ambiente protetto e separato dalla rete del dipartimento.

Gli apparati di rete sono accessibili attraverso un terminal server per la configurazione per ottenere la massima versatilità di interazione. Sono presenti dispositivi ausiliari che consentono di implementare servizi come freeradius o tftp server.

Tutti i dispositivi hardware sono configurabili dall'utente, che ha a disposizione tutto il necessario per costruire una qualsivoglia rete in modo completamente libero e sicuro.

4.2- Implementazione su hardware reale

Divincolandosi dall'ambiente di sviluppo virtuale, si è deciso di effettuare una configurazione con Network Automation con apparati reali.

Con tale scopo sono stati utilizzati quattro router Huawei di fascia media, e uno switch Huawei S5700. Come nel caso virtuale, è stato configurato un utente SSH per permettere l'accesso da una porta (nel caso in questione la porta 26 dello switch 3) e successivamente è stato lanciato il codice Python che ha effettuato la configurazione.

La configurazione dell'utente SSH è necessariamente avvenuta tramite la porta console del dispositivo. Il laboratorio Huawei dispone di un terminal server che permette, collegandosi con protocollo telnet, di accedere alla porta console di tutti i dispositivi presenti, rispettivamente selezionando la porta di comunicazione.

Il terminal server è ospitato da un router Cisco 3640 1che isola, oltretutto, la rete del laboratorio dalla rete del dipartimento permettendo qualsiasi configurazione senza rischio di mandare in down la rete istituzionale.

La configurazione base conferita a "SW3" è la medesima dello switch virtuale. Nella seguente "Figura 14" sono visibili i dispositivi presenti nel laboratorio.

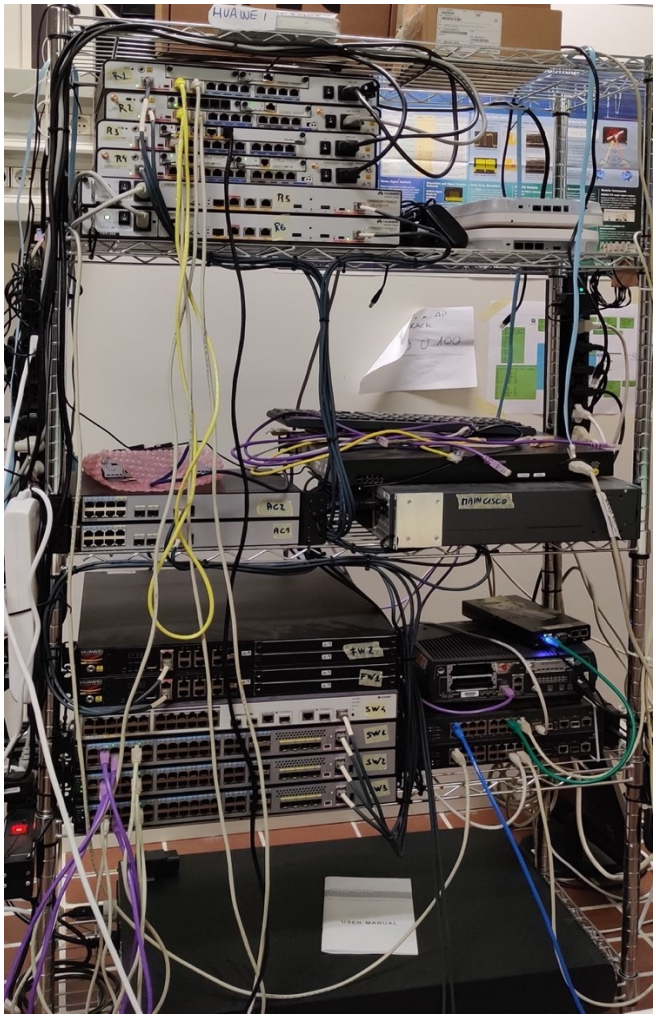

Figura 14. Laboratorio Huawei

Capitolo 5
Conclusione

5.1-Conclusione

L'elaborato tratta un esempio di Network Automation, volto alla configurazione delle VLAN di un apparato di "livello 2", permettendo la creazione da remoto di una rete specifica. Si può concludere che il progetto ha avuto esito positivo e di fatto il codice sviluppato consente di costruire una qualsivoglia topologia di rete, più o meno complessa. Quanto ottenuto, seppur una minuscola applicazione di Network Automation, è notevole; infatti, la possibilità di eseguire una configurazione automatica è un grande vantaggio, poiché riduce notevolmente l'impiego di manodopera per la programmazione, causa di tempi e costi elevati, e suscettibilità ad errori. Inoltre, con questa tecnologia non è necessario che un operatore che debba intervenire per la modifica di una configurazione, conosca il codice di programmazione del dispositivo che sta maneggiando. Di fatto ogni casa produttrice di dispositivi di rete, come Cisco piuttosto che Huawei, Arista e Mikrotik adotta un proprio linguaggio di configurazione. Grazie al Network Automation è stato possibile unificare il tutto dal lato programmazione, poiché sarà la parte software che convertirà questa, fatta ora con comandi standard, nel linguaggio adatto al dispositivo che si sta configurando.

5.2-Osservazioni e criticità riscontrate

Parlando delle criticità occorre fare un focus sul simulatore "eNSP" e più precisamente sulla notevole dipendenza dalla disponibilità di risorse del calcolatore su cui si lavora. Di fatto l'esito della simulazione di una rete dipende dalla capacità di elaborazione. Durante lo sviluppo del progetto, si sono riscontrati diversi problemi dovuti alla limitatezza delle

risorse computazionali a disposizione, causa del down di alcune porte simulate e con conseguente esito negativo della simulazione. Nel caso in questione, il problema è dovuto anche al fatto che, avendo a disposizione un sistema operativo Mac OS ed essendo eNSP un software Windows, si è dovuto ricorrere alla virtualizzazione del sistema operativo in questione limitando notevolmente le risorse computazionali.

Un'altra osservazione riguarda il dispositivo Cloud disponibile per la connessione da un dispositivo esterno alla rete in eNSP. Per utilizzare il cloud è necessario mappare opportunamente la rete definendo un percorso ben chiaro per il passaggio dei dati.

Ciò non è sufficiente, infatti il cloud necessita anche di un adattatore di rete (facilmente reperibile in commercio) e non funziona con una rete wireless (wi-fi). Tuttavia, nel caso in questione, lavorando con un sistema operativo virtualizzato, è stato possibile eseguire la mappatura tramite la rete virtuale creata dal virtualizzatore di Windows. Ecco spiegato il motivo per il quale, come visibile dal codice di configurazione, allo switch virtuale è assegnato un indirizzo IP 10.211.55.8 .

Il laboratorio Huawei è quindi essenziale per eseguire le simulazioni di rete in quanto non si hanno influenze dovute alla disponibilità di risorse. Di fatto, si ha a che fare con apparati reali che rispondono a problemi reali, a latenze reali e completamente incorrelate dal sistema operativo che si utilizza, piuttosto che alle risorse disponibili e all’efficienza della scheda di rete presente.

5.3-Idee per sviluppi futuri

5.3.1-link-aggregation

Nel capitolo 1 si è approfondito il significato di Network Automation. In questo elaborato, di fatto, si è eseguito un progetto basilare di network automation volto alla configurazione automatica delle VLAN. Il progetto può essere proiettato verso un nuovo orizzonte partendo dal punto di arrivo in questione. Per fare ciò è bene parlare del concetto di "link-aggregation", che consente di costruire architetture in grado di migliorare notevolmente la capacità di elaborazione della rete. Il link-aggregation è una procedura che permette di aumentare la portata in banda di una linea di collegamento tra dispositivi, separando il traffico dati su più linee filari. Nella "Figura 15" è illustrata la topologia prima(a) e dopo link aggregation(b) [10].

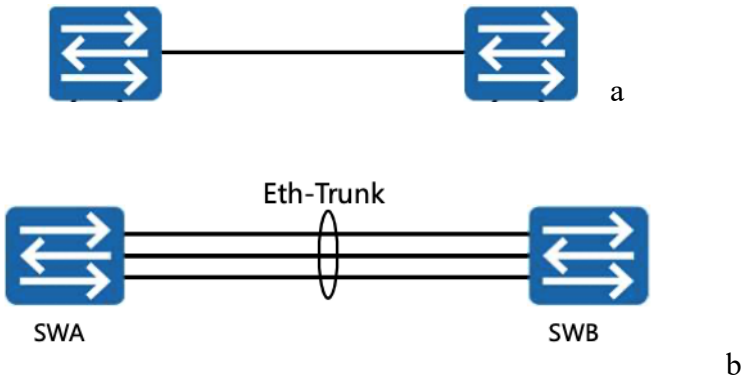

Figura 15. Topologia prima e dopo link aggregation

Estendendo il ragionamento alla topologia di partenza "Figura 16", si può pensare di fare link-aggregation sui due collegamenti verso i singoli router, consentendo nel peggiore dei casi prestazioni identiche alla topologia senza link-aggregation, ed in tutti gli altri casi

[10]

una portata maggiore rispetto al precedente, con il limite che tende al doppio della banda disponibile in precedenza. Ovviamente la programmazione non è banale poiché oltre che gestire la configurazione dello switch (apparato di livello 2) occorre gestire anche la configurazione dei router (apparati di livello 3) per consentire la procedura di link-aggregation.

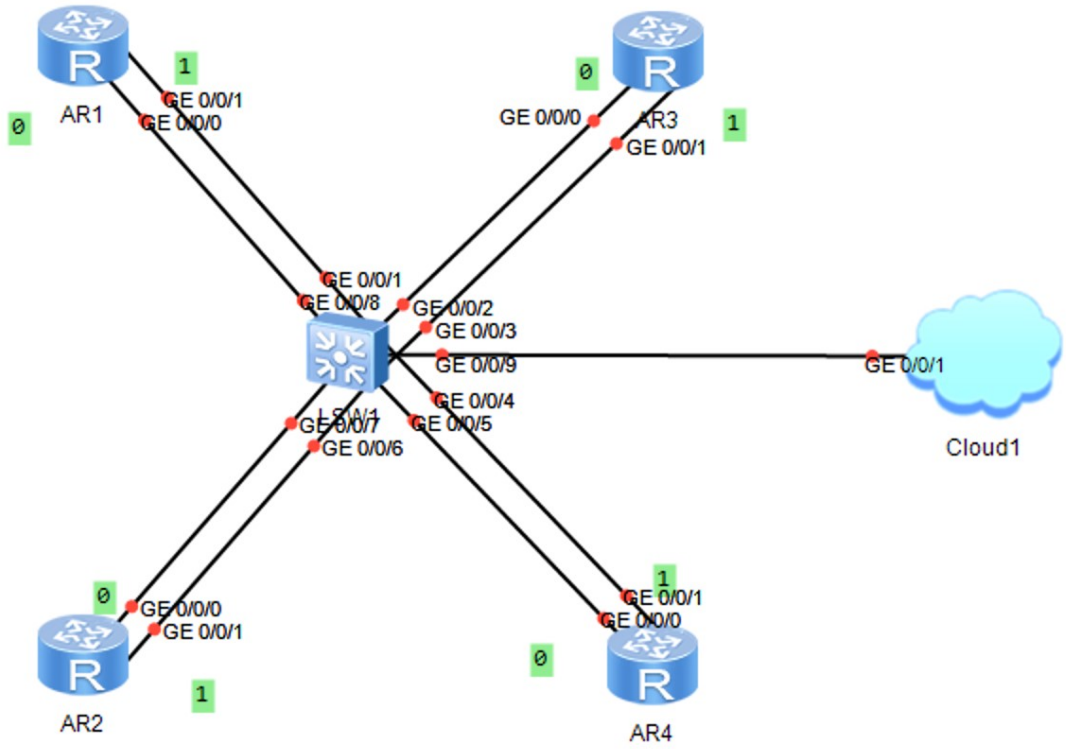

Figura 16. Topologia di partenza

5.3.2-protocollo API (Application Program Interface)

Un altro possibile sviluppo è eseguire la configurazione con protocollo API (Application Program Interface). L'automazione basata su API sostituisce le istruzioni manuali della riga di comando per configurare ciascun dispositivo di rete. Le API possono essere invocate direttamente o passare attraverso un linguaggio di programmazione, ad esempio Python, Java o Go. Gli script sono solo un aspetto dell'automazione della rete: una moderna piattaforma di automazione monitora le risorse di rete durante il provisioning e verifica che una rete sarà in grado di gestire una richiesta di configurazione prima di implementarla[11]. L'impiego delle API proietta l'operazione di Network Automation su un livello più alto; infatti, tramite questa tecnologia è possibile configurare una rete con comandi standard ed elementari, che verranno poi convertiti in righe di codice adatte al dispositivo che si sta configurando. L'operazione appena descritta consente al personale competente di lavorare con dispositivi di molteplici case produttrici con poche difficolta dal lato programmazione.

[11] https://www.vmware.com/topics/glossary/content/network-automation

Bibliografia

[1] https://www.cisco.com/c/en/us/solutions/automation/network-automation.html

[2] https://www.vmware.com/topics/glossary/content/network-automation

[3] https://it.wikipedia.org/wiki/Python

[4] https://www.networkcomputing.com/network-security/network-automation-netmiko

[5] Copyright © 2019 Huawei Technologies Co., Ltd. All rights reserved.

[6] https://www.mindlessnerds.com/guide/huawei-ensp-download-install-fix-how-to/

[7] https://www.jetbrains.com/help/pycharm/quick-start-guide.html

[8] https://github.com/ktbyers/netmiko

[9] https://www.vmware.com/topics/glossary/content/network-automation