

...Piotr...Dziwiewczyński
(podpis studenta)

Wyższa Szkoła Bankowa

Studia Podyplomowe

Programista Python

Michał Sybila

(imię i nazwisko studenta)

20846

(nr albumu)

Automatyzacja sieci IT

Praca dyplomowa

Promotor

Dr inż. Mariusz Mol

BYDGOSZCZ 2022

Spis treści

<u>Wstęp.....</u>	<u>3</u>
<u>I. Kod Programu.....</u>	<u>4</u>
1.1. Importujemy potrzebne biblioteki.....	4
1.2. Funkcja fping.....	4
1.3. Funkcja NTP.....	5
1.4. Funkcja rollback NTP.....	6
1.5. Funkcja backup configuration to TFTP.....	8
1.6. Menu programu.....	9
<u>II. Opis działania przebiegu programu.....</u>	<u>11</u>
<u>Zakończenie.....</u>	<u>16</u>

Wstęp

Pracując na stanowisku inżyniera do spraw sieci informacyjnych postanowiłem w końcu dołączyć do trendu w którym owy inżynier powinien umieć zautomatyzować swoją codzienną pracę wyzbywając się w ten sposób powtarzalnych zadań. Dlatego postawiłem na projekt, który zapozna mnie z podstawami komunikacji programu napisanego w programie Python z urządzeniami sieciowymi takimi jak routery, switchy czy firewalles.

I. Kod Programu

Popniżej przedstawiony został kod programu razem z komentarzami podzielonym na kilka głównych sekcji

1.1. Importujemy potrzebne biblioteki

```
import os #moduł OS – daje interfejs do systemu operacyjnego
from fileinput import close #umożliwia zamykanie plików
import paramiko #importujemy moduł PARAMIKO do obsługi SSH #implementacja
SSHv2 daje funkcjonalność klienta oraz servera SSH.
import time #obsługa czasu
from datetime import datetime #obsługa daty
```

1.2. Funkcja fping

Funkcja fping umożliwia nam przeprowadzenie szybkiego sprawdzenia komunikacji pomiędzy routerami których adresy IP zawarte są w pliku "routers" oraz serverów (wirtualizowanych instancji systemów linux). Funkcja przesyła za pośrednictwem modułu os komendy fping z parametrami odnoszącymi się do adresów IP routerów (plik routers) oraz IP serverów Linux (plik servers).

```
def fping(): #definicja funkcji fping
    routers = open("routers") #otwieramy plik routers zawierający adresy IP routerów
    for line in routers: #pętla for
        router_IP = line.strip()
        cmd = ('fping ' + router_IP) #przypisanie do zmiennej cmd komendy
systemowej systemu linux w postaci „fping + router_IP”
        os.system(cmd) #os.system wysyła komendę fping do systemu operacyjnego
        print(cmd, '\n') #wyświetlenie na ekranie komendy cmd

    servers = open("servers") #otwieramy plik servers z adresami IP serverów linux
    for line in servers: #pętla for
        server_IP = line #zmiennej server_IP przypisujemy adresy IP z pliku „servers”
        cmd = ('fping ' + server_IP) #przypisanie do zmiennej cmd komendy
systemowej systemu linux w postaci „fping + server_IP”
        os.system(cmd) #os.system wysyła komendę fping do systemu operacyjnego
        print(cmd, '\n') #wyświetlenie na ekranie komendy cmd
```

1.3. Funkcja NTP

NTP jest funkcją która za pomocą modułu paramiko łączy się za pomocą protokołu SSH z routerami których adresy IP zostały pobrane z pliku routers. Zmienna czas do której przypisana została funkcja datetime.now służy do wyświetlania daty oraz czasu w którym funkcja została wykonana dla każdego z routerów.

```
def ntp(): #Funkcja NTP
    czas = datetime.now().strftime("%Y-%m-%d_%H-%M-%S") #znacznik daty oraz
    czasu

    routers = open("routers") #otwieramy plik routers z adresami IP routerów

    for line in routers: #pętla czytająca linie adresów IP w pliku routers
        print(czas)
        print("logujemy sie na router " + (line))
        router_IP = line.strip() #zmienna router_IP przechowuje adres IP oraz usuwa
        puste znaki za pomocą metody strip..
        login = open("hasla") #zmienna login odczytuje login oraz hasło do logowania na
        routery z pliku hasła.

        for line1 in login: #pętla sczytuje pierwszą linię pliku zawierającą login
            username = line1.strip()

            for line2 in login: #pętla sczytuje drugą linię pliku zawierającą hasło
                password = line2.strip()

##### Sekcja odpowiadająca za nawiązanie połączenia SSH przez paramik #####

        ssh_client = paramiko.SSHClient()

        ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        ssh_client.connect(hostname=router_IP, username=username,
        password=password)
        print("Successfull connection to " + (router_IP) + "\n")

        remote_connection = ssh_client.invoke_shell()
        output1 = remote_connection.recv(3000)

##### Koniec sekcji odpowiadającej za nawiązanie połączenia SSH przez paramik #####

        remote_connection.send("configure terminal\n") #funkcja sent
        paramiko wysyła komendę configure terminal do routera przecchodząc w tryb konfiguracji
```

```

        print("Configuring NTP server") #przechodzimy do sekcji
konfiguracji NTP
        remote_connection.send("ntp server 192.168.121.131\n")
#konfigurujemy adres servera NTP
        remote_connection.send("end\n") #komenda end umożliwia powrót
do trybu enable powodując wyjście z trybu konfiguracji
        remote_connection.send("write\n") #funkcja write powoduje
zapisanie zmodyfikowanej konfiguracji na routerach
        print() #wyświetlamy komunikację z urządzeniami

        time.sleep(3) #funkcja time sleep ustawiona na 3 sekundy daje czas
urządzeniu na wykonanie wszystkich operacji
        output2 = remote_connection.recv(65535) #konfiguracja
maksymalnej ilości znaków które możemy otrzymać z naszej funkcji
        print((output2).decode('ascii')) #decodowanie outputu do
standardu ASCII

        print(("Successfully configured your device &
Disconnecting from ") + (router_IP)) #wyświetlamy informację o poprawnym
wykonaniu funkcji

        ssh_client.close() #zamykamy połączenie SSH
        time.sleep(3) #odczekujemy 3 sekundy aby urządzenie miało czas na
poprawne zakończenie sesji

        routers.close() #zamykamy plik router
        login.close() #zamykamy plik logi

```

1.4. Funkcja rollback NTP

Rollback NTP jest funkcją usuwającą konfigurację NTP.

```

def ntp_rm(): #Funkcja ntp_rm
    czas = datetime.now().strftime("%Y-%m-%d_%H-%M-%S") #znacznik daty oraz
czasu

    routers = open("routers") #otwieramy plik routers z adresami IP routerów

    for line in routers: #pętla czytająca linie adresów IP w pliku routers
        print(czas) #wyświetlamy aktualny czas
        print("logujemy sie na router " + (line)) #wyświetlamy informację o
zalogowaniu się na urządzenie
        router_IP = line.strip() #zmienna router_IP przechowuje adres IP oraz usuwa
puste znaki za pomocą metody strip.

```

login = open("hasla") #zmienna login odczytuje login oraz hasło do logowania na routery z pliku hasła.

for line1 in login: #pętla czytuje pierwszą linię pliku zawierającą login
username = line1.strip()

for line2 in login: #pętla czytuje drugą linię pliku zawierającą hasło
password = line2.strip()

Sekcja odpowiadająca za nawiązanie połączenia SSH przez paramiko

ssh_client = paramiko.SSHClient()

ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh_client.connect(hostname=router_IP, username=username,
password=password)

print("Successfull connection to " + (router_IP) + "\n")

remote_connection = ssh_client.invoke_shell()
output1 = remote_connection.recv(3000)

Koniec sekcji odpowiadającej za nawiązanie połączenia SSH przez paramiko

remote_connection.send("configure terminal\n") #funkcja sent
paramiko wysyła komendę configure terminal do routera przecchodząc w tryb konfiguracji
print("Configuring NTP server") #przechodzimy do sekcji
konfiguracji NTP

remote_connection.send("no ntp server 192.168.121.131\n")
#usuwamy adres servera NTP

remote_connection.send("end\n") #komenda end umożliwia powrót
do trybu enable powodując wyjście z trybu konfiguracji

remote_connection.send("write\n") #funkcja write powoduje
zapisanie zmodyfikowanej konfiguracji na routerach
print() #wyświetlamy komunikację z urządzeniami

time.sleep(3) #funkcja time sleep ustawiona na 3 sekundy daje czas
urządzeniu na wykonanie wszystkich operacji

output2 = remote_connection.recv(65535) #konfiguracja
maksymalnej ilości znaków które możemy otrzymać z naszej funkcji
print((output2).decode('ascii')) #decodowanie outputu do
standardu ASCII

print(("Successfully configured your device &
Disconnecting from ") + (router_IP)) #wyświetlamy informację o poprawnym
wykonaniu funkcji

ssh_client.close() #zamykamy połączenie SSH

```
        time.sleep(3) #odczekujemy 3 sekundy aby urządzenie miało czas na
poprawne zakończenie sesji
```

```
    routers.close() #zamykamy plik router
    login.close() #zamykamy plik logi
```

1.5. Funkcja backup configuration to TFTP

Jest to funkcją która zapisuje aktualną konfigurację routerów oraz przesyła jej kopię na server TFTP.

```
def backup(): #Funkcja backup
    czas = datetime.now().strftime("%Y-%m-%d_%H-%M-%S") #znacznik daty oraz
    czasu

    routers = open("routers") #otwieramy plik routers z adresami IP routerów

    for line in routers: #pętla czytająca linie adresów IP w pliku routers
        print(czas) #wyświetlamy aktualny czas
        print("logujemy się na router " + (line)) #wyświetlamy informację o
        zalogowaniu się na urządzenie
        router_IP = line.strip() #zmienna router_IP przechowuje adres IP oraz usuwa
        puste znaki za pomocą metody strip.
        login = open("hasla") #zmienna login odczytuje login oraz hasło do logowania na
        routery z pliku hasła.

        for line1 in login: #pętla czytuje pierwszą linię pliku zawierającą login
            username = line1.strip()

            for line2 in login: #pętla czytuje drugą linię pliku zawierającą hasło
                password = line2.strip()

##### Sekcja odpowiadająca za nawiązanie połączenia SSH przez paramik #####

        ssh_client = paramiko.SSHClient()

        ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        ssh_client.connect(hostname=router_IP, username=username,
        password=password)
        print("Successfull connection to " + (router_IP) + "\n")

        remote_connection = ssh_client.invoke_shell()
        output1 = remote_connection.recv(3000)

##### Koniec sekcji odpowiadającej za nawiązanie połączenia SSH przez paramik #####
```



```

        print("Now making running-config backup of " + (router_IP)
+ "\n") #Wyświetlamy informację o nazwie routera dla którego tworzona jest kopia zapasowa
konfiguracji do przesłania na server TFTP
        time.sleep(3) #funkcja time sleep ustawiona na 3 sekundy daje czas
urządzeniu na wykonanie wszystkich operacji
        remote_connection.send("copy running-config tftp\n")
#funkcja send paramiko wysyła komendę "copy running-config tftp" do routera powodując zapisanie
konfiguracji i wysłanie jej na server podany w następnym kroku
        remote_connection.send("192.168.121.131\n") #podajemy adres
servera do wykonania kopii zapasowej konfiguracji
        remote_connection.send((router_IP)+ ".bak@" + (czas+
"\n")) #podajemy nazwę pliku w formacie IP.bak@data_godzina – przykład poniżej
#192.168.121.136.bak@2022-09-08_18-07-13
        time.sleep(3) #funkcja time sleep ustawiona na 3 sekundy daje czas
urządzeniu na wykonanie wszystkich operacji
        print() #wyświetlamy komunikację z urządzeniami
        time.sleep(3) #funkcja time sleep ustawiona na 3 sekundy daje czas
urządzeniu na wykonanie wszystkich operacji

        output2 = remote_connection.recv(65535) #konfiguracja
maksymalnej ilości znaków które możemy otrzymać z naszej funkcji
        print((output2).decode('ascii')) #decodowanie outputu do
standardu ASCII

        print(("Successfully configured your device &
Disconnecting from ") + (router_IP)) #wyświetlamy informację o poprawnym
wykonaniu funkcji

        ssh_client.close() #zamykamy plik router
        time.sleep(3) #zamykamy plik logi

    routers.close() #zamykamy plik router
    login.close() #zamykamy plik logi
    return()

```

1.6. Menu programu

menu=True #W celu utworzenia menu programu przypisujemy zmiennej menu wartość True
while menu: #Pętla while umożliwi nam wybieranie funkcji programu z menu aż do momentu
wybrania funkcji Exit

```

print (""" #wyświetlamy menu programu
1.connectivity check
2.configure NTP
3.rollback NTP

```

```

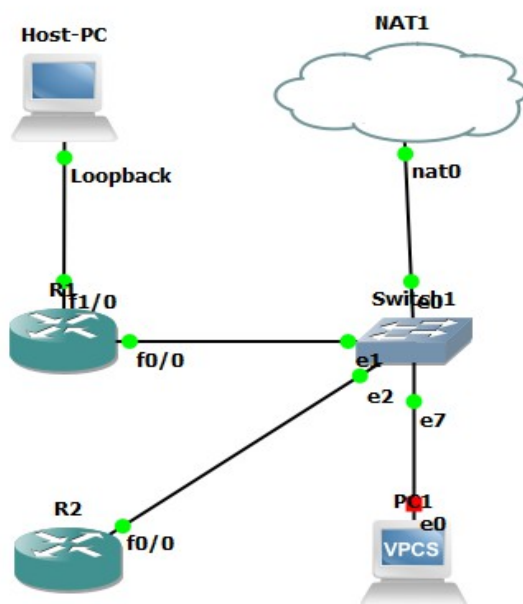
4.backup configuration to TFTP
5.Exit/Quit
""")
menu=input("What would you like to do? ")
if menu=="1": #przypisujemy wartosci 1 funkcję "fping"
    print("\n fping started")
    fping()
elif menu=="2": #przypisujemy wartosci 2 funkcję "NTP"
    print("\n NTP started")
    ntp()
elif menu=="3": #przypisujemy wartosci 3 funkcję "NTP removing"
    print("\n NTP removing")
    ntp_rm()
elif menu=="4": #przypisujemy wartosci 4 funkcję "backup"
    print("\n backing up configuration")
    backup()
elif menu=="5": #przypisujemy wartosci 5 funkcję "exit"
    print("\n EXIT")
    break
elif menu!="":
    print("\n Try again")

```

II. Opis działania przebiegu programu

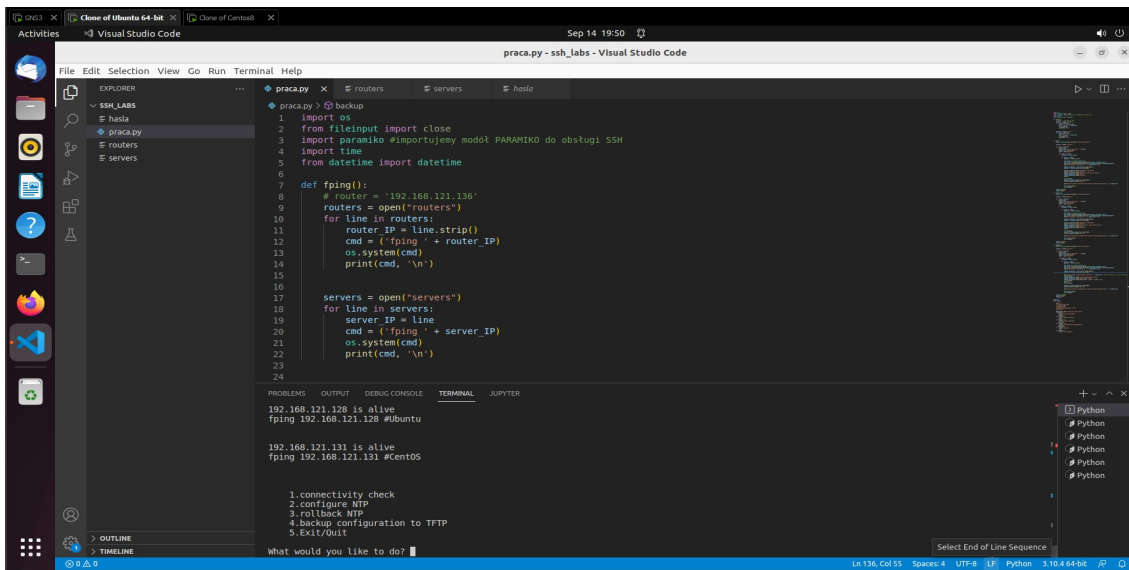
Program umożliwia wykonanie kilku prostych operacji związanych z automatyzacją sieci i urządzeń IP.

Program komunikuje się oraz zarządza wirtualizowaną siecią IP odpaloną w programie GNS3 który symuluje 2 routery cisco R1 (192.168.121.136), R2 (192.168.121.138), host PC, NAT oraz systemowy Loopback w systemie Windows (7.7.7.1) tworząc prostą topologię sieci którą obrazuje rys1.



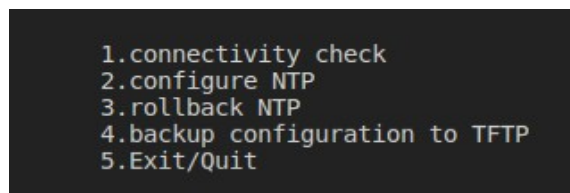
Rys1 – topologia sieci w programie GNS3

Program komunikuje się również z emulowanymi serwerami systemu Linux w Dystrybucji Ubuntu z której zarządzamy siecią oraz Centos który obsługuje server tftp na który możemy przysyłać kopię zapasową konfiguracji urządzeń sieciowych. W naszym przypadku program jest uruchamiany w wirtualizowanej dystrybucji systemu linux na której zainstalowany został program Visual Studio Code, system Ubuntu posiada dostępny program fping uruchamiany z terminala oraz funkcją ssh.



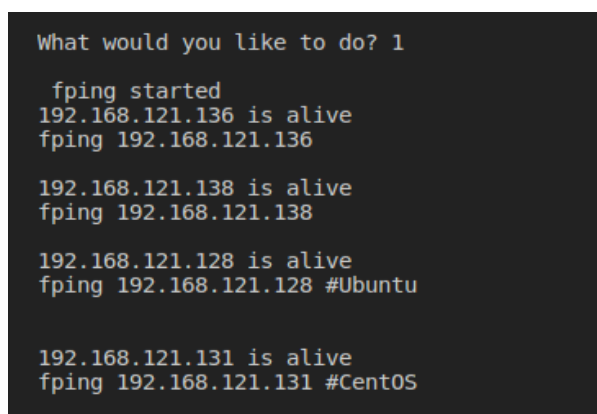
Rys2 – Zrzut programu Visual Studio Code uruchomionego w wirtualnym systemie Linux

Po uruchomieniu programu wyświetla się nam pełne menu ukazane poniżej:



Rys3 – MENU programu

Funkcja numer 1 – connectivity check - funkcja fping umożliwia nam przeprowadzenie szybkiego sprawdzenia komunikacji pomiędzy routerami których adresy IP zawarte są w pliku "routers" oraz serwerów (wirtualizowanych instancji systemów linux). Funkcja przesyła za pośrednictwem modułu os komendy fping z parametrami odnoszącymi się do adresów IP routerów (plik routers) oraz IP serwerów Linux (plik servers).



Rys4 – Funkcja fping potwierdza prawidłową komunikację z routerami oraz serverami linux

Funkcja numer 2 – Configure NTP - ntp jest funkcją która za pomocą modułu paramiko łączy się za pomocą protokołu SSH z routerami których adresy IP zostały pobrane z pliku routers. Zmienna czas do której przypisana została funkcja datetime.now służy do wyświetlania daty oraz czasu w którym

funkcja została wykonana dla każdego z routerów. Protokół NTP umożliwia precyzyjną synchronizację czasu pomiędzy komputerami dzięki niemu podczas troublehootingu urządzeń sieciowych jesteśmy w stanie śledzić zdarzenia zależnych od siebie urządzeń mając pewność że każdy z nich posługuje się tym samym czasem. Program konfiguruje adres 192.168.121.131 jako server NTP na wszystkich routerach których adresy znajdują się w pliku routers.

```
R1#  
R1#  
R1#show run | i ntp  
R1#
```

Rys5 – brak konfiguracji servera NTP na routerach R1 oraz R2

```
NTP started  
2022-09-14_20-00-38  
logujemy się na router 192.168.121.136  
  
Successfull connection to 192.168.121.136  
  
Configuring NTP server  
  
configure terminal  
Enter configuration commands, one per line. End with CNTL/Z.  
R1(config)#ntp server 192.168.121.131  
R1(config)#end  
R1#write  
Warning: Attempting to overwrite an NVRAM configuration previously written  
by a different version of the system image.  
Overwrite the previous NVRAM configuration?[confirm]  
Successfully configured your device & Disconnecting from 192.168.121.136  
2022-09-14_20-00-38  
logujemy się na router 192.168.121.138  
  
Successfull connection to 192.168.121.138  
  
Configuring NTP server  
  
configure terminal  
Enter configuration commands, one per line. End with CNTL/Z.  
R2(config)#ntp server 192.168.121.131  
R2(config)#end  
R2#write  
Warning: Attempting to overwrite an NVRAM configuration previously written  
by a different version of the system image.  
Overwrite the previous NVRAM configuration?[confirm]  
Successfully configured your device & Disconnecting from 192.168.121.138  
  
1.connectivity check  
2.configure NTP  
3.rollback NTP  
4.backup configuration to TFTP  
5.Exit/Quit
```

Rys6 – dane prezentowane podczas wykonywania funkcji NTP

```

R1#
R1#show run | i ntp
ntp server 192.168.121.131
R1#show ntp associations

  address          ref clock      st   when   poll reach  delay  offset  disp
~192.168.121.131 93.94.224.67    3    50    64    17 11.875 611.458 938.45
* sys.peer, # selected, + candidate, - outlier, x falseticker, ~ configured
R1#

```

Rys7 – widoczny skonfigurowany server NTP oraz zrzut funkcji "show ntp association" z routera R1 który pokazuje komunikację z serverem NTP

Funkcja numer 3 – rollback NTP usuwa konfigurację dodaną do routerów podczas wykonywanie funkcji NTP

Funkcja numer 4 – backup configuration to TFTP – funkcja zapisuje bieżącą konfigurację routerów których adresy IP znajdują się w pliku "routers" na server TFTP skonfigurowany na wirtualnym serverze linux (CentOS).

Nazwa pliku	Rozmiar p...	Typ pliku	Data modyfikacji	Prawa dost...	Właści
192.168.121.138.bak@2022-09-09_16-16-10	1 316	Plik BAK@...	09.09.2022 16:16:28	-rw-rw-r--	tftpuse
192.168.121.136.bak@2022-09-09_16-16-10	1 361	Plik BAK@...	09.09.2022 16:16:15	-rw-rw-r--	tftpuse
192.168.121.138.bak@2022-09-08_19-24-37	1 316	Plik BAK@...	08.09.2022 19:24:55	-rw-rw-r--	tftpuse
192.168.121.136.bak@2022-09-08_19-24-37	1 361	Plik BAK@...	08.09.2022 19:24:42	-rw-rw-r--	tftpuse
192.168.121.138.bak@2022-09-08_18-12-42	1 343	Plik BAK@...	08.09.2022 18:12:59	-rw-rw-r--	tftpuse
192.168.121.136.bak@2022-09-08_18-12-42	1 388	Plik BAK@...	08.09.2022 18:12:47	-rw-rw-r--	tftpuse
192.168.121.138.bak@2022-09-08_18-07-13	1 343	Plik BAK@...	08.09.2022 18:07:32	-rw-rw-r--	tftpuse
192.168.121.136.bak@2022-09-08_18-07-13	1 388	Plik BAK@...	08.09.2022 18:07:19	-rw-rw-r--	tftpuse
c3745-adventerprisek9-mz.124-25c.bin	82 026 148	Plik BIN	06.09.2022 21:38:00	-rw-rw-r--	kobe k
running-config	1 235	Plik	06.09.2022 21:11:47	-rw-rw-r--	tftpuse
transfer_file77	237	Plik	20.08.2022 18:00:13	-rw-rw-r--	root ro
transfer_file01	237	Plik	20.08.2022 17:59:33	-rw-rw-r--	tftpuse

Rys8 – widok programu filezilla pokazuje pliki dostępne na serverze TFTP

```

What would you like to do? 4

backing up configuration
2022-09-14_20-30-44
logujemy się na router 192.168.121.136

Successfull connection to 192.168.121.136

Now making running-config backup of 192.168.121.136

copy running-config tftp
Address or name of remote host []? 192.168.121.131
Destination filename [r1-config]? 192.168.121.136.bak@2022-09-14_20-30-44
!!
1389 bytes copied in 1.736 secs (800 bytes/sec)

R1#
Successfully configured your device & Disconnecting from 192.168.121.136
2022-09-14_20-30-44
logujemy się na router 192.168.121.138

Successfull connection to 192.168.121.138

Now making running-config backup of 192.168.121.138

copy running-config tftp
Address or name of remote host []? 192.168.121.131
Destination filename [r2-config]? 192.168.121.138.bak@2022-09-14_20-30-44
!!
1344 bytes copied in 1.820 secs (738 bytes/sec)

R2#
Successfully configured your device & Disconnecting from 192.168.121.138

```

Rys9 – dane prezentowane podczas wykonywania funkcji "backing up configuration to TFTP"

Nazwa pliku	Rozmiar p...	Typ pliku	Data modyfikacji	Prawa dost...	Wła
..					
192.168.121.138.bak@2022-09-14_20-30-44	1 344	Plik BAK@...	14.09.2022 20:31:02	-rw-rw-r--	tftp
192.168.121.136.bak@2022-09-14_20-30-44	1 389	Plik BAK@...	14.09.2022 20:30:49	-rw-rw-r--	tftp
192.168.121.138.bak@2022-09-09_16-16-10	1 316	Plik BAK@...	09.09.2022 16:16:28	-rw-rw-r--	tftp
192.168.121.136.bak@2022-09-09_16-16-10	1 361	Plik BAK@...	09.09.2022 16:16:15	-rw-rw-r--	tftp
192.168.121.138.bak@2022-09-08_19-24-37	1 316	Plik BAK@...	08.09.2022 19:24:55	-rw-rw-r--	tftp
192.168.121.136.bak@2022-09-08_19-24-37	1 361	Plik BAK@...	08.09.2022 19:24:42	-rw-rw-r--	tftp
192.168.121.138.bak@2022-09-08_18-12-42	1 343	Plik BAK@...	08.09.2022 18:12:59	-rw-rw-r--	tftp
192.168.121.136.bak@2022-09-08_18-12-42	1 388	Plik BAK@...	08.09.2022 18:12:47	-rw-rw-r--	tftp
192.168.121.138.bak@2022-09-08_18-07-13	1 343	Plik BAK@...	08.09.2022 18:07:32	-rw-rw-r--	tftp
192.168.121.136.bak@2022-09-08_18-07-13	1 388	Plik BAK@...	08.09.2022 18:07:19	-rw-rw-r--	tftp
c3745-adventerprisek9-mz.124-25c.bin	82 026 148	Plik BIN	06.09.2022 21:38:00	-rw-rw-r--	koł
running-config	1 235	Plik	06.09.2022 21:11:47	-rw-rw-r--	tftp
transfer file77	237	Plik	20.08.2022 18:00:13	-rw-rw-r--	roo

Rys10 – widok programu filezilla pokazuje dwa nowe pliki z konfiguracją dostępne na serwerze TFTP

Funkcja numer 4 – Exit/Quit – wyjście z programu

Zakończenie

Pracując nad projektem zapoznałem się z podstawami automatyzacji sieci które otworzą mi drogę do poszerzania wiedzy i praktyki w tym kierunku nakierunkowując moją karierę zawodową na nowe „szybsze” tory automatyzacji z wykorzystaniem języka programowania Python.