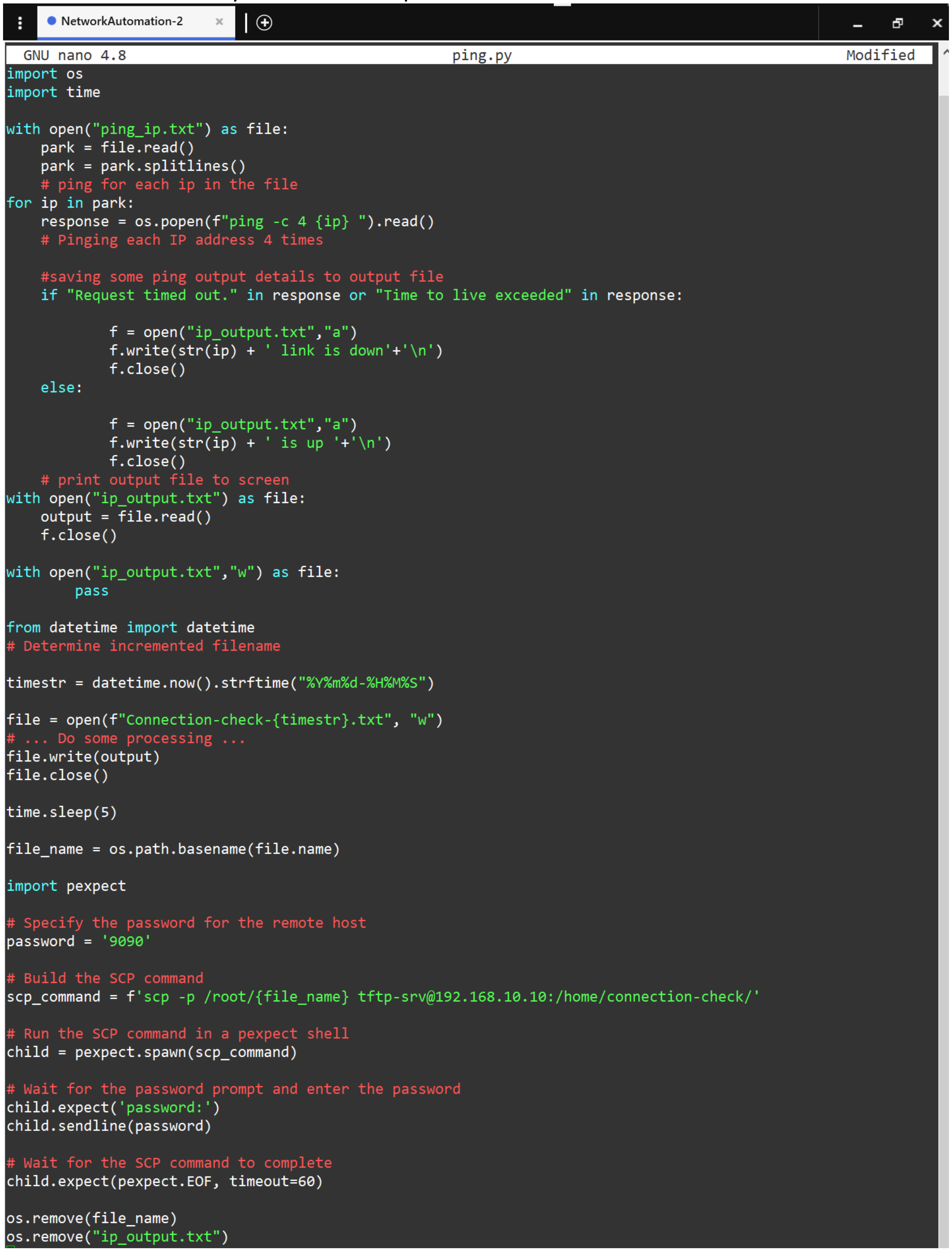
Automated Connectivity Assessment script



NetworkAutomation-2

```
  GNU nano 4.8                         ping.py                         Modified
import os
import time

with open("ping_ip.txt") as file:
    park = file.read()
    park = park.splitlines()
    # ping for each ip in the file
for ip in park:
    response = os.popen(f"ping -c 4 {ip} ").read()
    # Pinging each IP address 4 times

    #saving some ping output details to output file
    if "Request timed out." in response or "Time to live exceeded" in response:

            f = open("ip_output.txt","a")
            f.write(str(ip) + ' link is down'+'\n')
            f.close()
    else:

            f = open("ip_output.txt","a")
            f.write(str(ip) + ' is up '+'\n')
            f.close()
    # print output file to screen
with open("ip_output.txt") as file:
    output = file.read()
    f.close()

with open("ip_output.txt","w") as file:
        pass

from datetime import datetime
# Determine incremented filename

timestr = datetime.now().strftime("%Y%m%d-%H%M%S")

file = open(f"Connection-check-{timestr}.txt", "w")
# ... Do some processing ...
file.write(output)
file.close()

time.sleep(5)

file_name = os.path.basename(file.name)

import pexpect

# Specify the password for the remote host
password = '9090'

# Build the SCP command
scp_command = f'scp -p /root/{file_name} tftp-srv@192.168.10.10:/home/connection-check/'

# Run the SCP command in a pexpect shell
child = pexpect.spawn(scp_command)

# Wait for the password prompt and enter the password
child.expect('password:')
child.sendline(password)

# Wait for the SCP command to complete
child.expect(pexpect.EOF, timeout=60)

os.remove(file_name)
os.remove("ip_output.txt")
```