Homework 2 Manual: Ping and Traceroute

Due date: 28th February, 2025

1. Goal

Write the ping and traceroute programs that work like real ones in Linux. We use ping to check network connectivity and to see whether a remote server is up and running. Ping sends an ICMP echo request packet to the server, which will in turn reply with an ICMP echo reply packet to the sender. The sender then knows the server is up and running and measures the elapsed time for the communication. Traceroute is a network diagnostic tool often used to find out the routes from the sender to the receiver. In traceroute, the sender generates a sequence of UDP packets destined for the receiver with TTL gradually increasing starting at 1 in order to discover the intermediate routers. An intermediate router issues an ICMP error message when TTL becomes 0 and returns the message back to the sender.This is how the sender discovers the intermediate routers on the route to the destination.

IMPORTANT: The ICMP packets generated from your ping and traceroute should be acceptable by actual routers or servers. You need to use raw sockets to compose ICMP packets. Make your program display the output as the real ping and traceroute programs.

Make sure to read the following instructions carefully before starting to write your code, more specific coding details are outlined later in the document.

2. Instructions

1. **Programming Language:**
    a. You must use **Python** to implement your code (Useful link: https://www.python.org)
    b. Follow the PEP 8 coding guidelines for writing clean and professional Python code. (Here is a link: https://peps.python.org/pep-0008/)
    c. Use `git` for version control of your code
2. **Code Documentation and Comments:**
    a. Properly comment your code to make it clear and understandable

   b. Generate a PDF documentation for your code using a tool like Sphinx that automatically generates it for you. (see this link: https://www.sphinx-doc.org/en/master/). There are other tools that provide similar functionality (i.e. automatically generates documentation for you). You can use any of them if you dont like sphinx

3. **What to Submit:** All files must be submitted in a **single zip file**. Name the zip file <firstname>_<lastname>_hw1.zip (replace <firstname> and <lastname> with your name). The zip will have **atleast 7 files**:
   a. **Python code**: The main `my_ping.py` and `my_traceroute.py` along with any addition Python files required to run your code.
   b. **Requirements File**: Submit a `requirements.txt` file listing all the Python dependencies required to run your code. (See this link: https://www.geeksforgeeks.org/how-to-create-requirements-txt-file-in-python/)
   c. **Readme**: A `README.md` file that clearly explains: 1- How to compile and run your code. 2- Examples of command-line usage
   d. **Report (PDF)**: The report must be in pdf format. It should include:
      i. Screenshots of the output of `my_ping.py` ran with all the different flags (command line options) you implement.
      ii. Screenshots of the output of `my_traceroute.py` ran with all the different flags (command line options) you implement.
   e. **Code Documentation**: It must be in PDF format. Use a tool like Sphinx to automatically generate documentation
   f. **GIT Log**: Use Git for version control and make commits at appropriate points during development. Submit a git log file showing the commit history. This file should be name `revisions.txt`

2. Ping

Learn about ping command. See link in the references section. Build your own ping named `my_ping.py` with the following options:

- `-c count` : Stop after sending (and receiving) count ECHO_RESPONSE packets. If this option is not specified, ping will operate until interrupted.
- `-i wait` : Wait for `wait` seconds between sending each packet. Default is one second.
- `-s packetsize` : Specify the number of data bytes to be sent. Default is 56 (64 ICMP data bytes including the header).
- `-t timeout` : Specify a timeout in seconds before ping exits regardless of how many packets have been received.

3. Traceroute

Learn about traceroute command. See link in the references section. Build your own traceroute named `my_traceroute.py` with the following options:

- `-n` : Print hop addresses numerically rather than symbolically and numerically.
- `-q nqueries` : Set the number of probes per TTL to `nqueries` .
- `-S` : Print a summary of how many probes were not answered for each hop.

Tips

- If using Windows, ensure Windows Defender Firewall allows ICMP traffic:
  - Turn Firewall off (not recommended, turn it back on after testing).
  - Create an Inbound ICMP Rule. (see this link: Link)
  - Enable existing inbound ICMP rule. Look for advanced setting in printer and sharing settings and find inbound ICMP echo request.
  - Use the **argparse** module in Python to parse command-line arguments.

References

- Create an Inbound ICMP Rule
- Python argparse Documentation
- Ping Documentation
- Traceroute Documentation