CSCI-4220 Lab 6: Mini Traceroute with getaddrinfo()

Overview

This lab combines two important concepts from the IP layer: 1. **Name resolution** using `getaddrinfo()`. 2. **Traceroute logic** using IP's Time-To-Live (TTL) and ICMP responses.

You will implement a simplified version of the `traceroute` command that sends UDP probe packets with increasing TTL values and listens for ICMP responses from routers along the path.

Learning Objectives

After completing this lab, you will be able to:

- Use `getaddrinfo()` to resolve hostnames to IP addresses.
- Adjust the IP TTL field with `setsockopt()`.
- Use UDP and ICMP sockets to measure per-hop latency.
- Interpret ICMP Time Exceeded and Port Unreachable messages.
- Compare your traceroute output to the system traceroute.

Background

`traceroute` exploits the **IP Time-To-Live (TTL)** field and **ICMP error reporting**.

TTL Value	What Happens
1	First router decrements TTL to 0 and replies with ICMP Time Exceeded.
2	Second router replies, revealing the next hop.
…	…
n	Destination replies with ICMP Port Unreachable (UDP target port unused).

Lab Tasks

Step 1. Resolve Hostname

Use `getaddrinfo()` to look up the IPv4 address of the destination hostname provided as a command-line argument. Print the resolved IP address before starting traceroute.

Step 2. Create Sockets

- **UDP socket** for sending probes.
- **Raw ICMP socket** for receiving replies.

Step 3. TTL Probing Loop

For each TTL value from 1 to 30:

1. Set `IP_TTL` via `setsockopt()`.
2. Record the send timestamp (`gettimeofday()`).

3. Send an empty UDP datagram to the destination on port 33434. Port 33434 is well above the range of well-known ports. Traceroute traditionally sends UDP probes to high-numbered destination ports (starting at 33434) because:
   - Those ports are unlikely to be in use.
   - The final destination will respond with an ICMP Port Unreachable message when no application is listening there.
   - This behavior was first used in Van Jacobson's original traceroute (1988).
4. Wait up to **1 second** for an ICMP response.
   - If you receive a packet: process it to determine the ICMP type and code.
   - If no packet arrives within the timeout, assume the router or firewall dropped the ICMP reply.
5. When a reply arrives:
   - ICMP type 11 → Time Exceeded → the packet expired at this router (print that router's IP and RTT).
   - ICMP type 3 code 3 → Destination Unreachable (Port Unreachable) → you reached the final destination (print IP, RTT, and a "Destination reached" message).
   - Any other ICMP type/code can be printed for debugging (optional).
6. If no ICMP reply is received (timeout):
   - Print an asterisk (*) for that hop, following standard traceroute convention.
   - Do not exit the loop — continue to the next TTL value.
7. After each hop:
   - Print the hop number, router IP (or *), and RTT in milliseconds.
   - Stop the loop once you reach the destination (ICMP type 3, code 3) or after TTL=30.

Step 4. Stop Condition

Terminate when you receive an ICMP Port Unreachable (type 3, code 3), indicating you reached the destination. However, sometimes you'll never get the final "Port Unreachable" message from the destination. Traceroute stops increasing the TTL after a maximum hop count (often 30 by default) Common reasons include that Firewalls and filtering block ICMP responses, or routers may throttle ICMP messages to avoid being overloaded, or the target host itself might be down or not reachable at all.

Example output

```
lei@@Leis-MacBook-Pro~/Labs sudo ./lab6_traceroute www.google.com
Tracing route to www.google.com (142.251.35.164)
 1  192.168.50.1     1.46 ms
 2  142.254.211.225  11.51 ms
 3  24.58.201.201    26.70 ms
 4  24.58.33.138     19.06 ms
 5  24.58.32.56      11.06 ms
 6  24.30.201.130    18.68 ms
 7  *
 8  *
 9  142.251.60.238   21.24 ms
10  142.251.64.7     18.84 ms
11  142.251.35.164   21.34 ms  Destination reached.
```

Hints

- Requires `sudo` to create a raw socket.
- Use `inet_ntop()` to print IPs.
- Be sure to free results from `getaddrinfo()`.
- Use two sockets:
  - UDP socket for sending probes.
  - Raw ICMP socket (socket(AF_INET, SOCK_RAW, IPPROTO_ICMP)) for receiving replies.
- ICMP response Timeouts
  - Using setsockopt() with SO_RCVTIMEO on recv socket. When that timer expires, recvfrom() returns -1.
- For each probe, you will measure the round-trip time (RTT) — the total time it takes for your UDP packet to reach a router (or the destination) and for the corresponding ICMP reply to come back.
- When you receive data from the raw ICMP socket using recvfrom(), the buffer (buf) contains the entire IPv4 packet — starting with an IP header, followed by an ICMP header and payload. You'll need to locate the ICMP header within that buffer.
  - You can get the ip header struct and the len field by `struct ip *ip_hdr = (struct ip *)buf;` and `ip_hdr->ip_hl`.. check <netinet/ip.h> Structure of an internet header, naked of options.
  - Remember ip_hl counts 32-bit words.
  - ICMP header starts right after the IP header. See our slides.
    - The first byte of the ICMP header is the ICMP type, the second is the ICMP code.

Wireshark Exercise

After completing your program:

1. Start Wireshark capture on your active network interface.
2. Apply filter: `icmp || udp.port == 33434`
3. Run your traceroute program.
4. Observe:
   - Outgoing UDP packets with incrementing TTL values.
   - Corresponding ICMP Time Exceeded messages.
   - The final ICMP Port Unreachable reply from the destination.

Screenshot Checklist

☐ UDP packet with TTL=1 and ICMP Time Exceeded.
☐ TTL=2 and next-hop router response.
☐ Final ICMP Port Unreachable from destination.

Questions to Answer

1. Which ICMP type and code are used by intermediate routers?
2. What happens when TTL=30 but no ICMP reply is received?
3. How do RTTs vary across hops? Briefly explain why with possible reasons.

What to Submit

Submit the following files on Submitty:

- `lab6_traceroute.c` (source code)
- One-page report including:
  - Screenshot of traceroute output.
  - Wireshark screenshots (TTL increments, ICMP replies).
  - Your answers to three questions.

Compilation

```
gcc -o lab6_traceroute lab6_traceroute.c
sudo ./lab6_traceroute www.google.com
```