

COMP416 Project-3

Network Layer Analysis and Routing Simulator

Ertan Can Güner 60610

Overview:

In this project we are asked to analyze ping and traceroute commands that use ICMP which is an error exchange protocol used on the network layer. We will analyze the communication between our machine and 'www.lehigh.edu', to observe what kind of information is being exchanged. In the second part of the project, we will implement 3 different routing algorithms and simulate it.

Routing:

Flooding Algorithm: Routers are allowed to only route once thus a boolean flag is introduced to keep track of the information. If the router has routed before an empty array list is returned. Otherwise, a lambda function is called to remove, where the packet came from, and return the neighbors.

```
public List<NeighborInfo> selectNeighbors(String origin, String destination,
                                          List<NeighborInfo> neighbors) {
    // Your code goes here.
    if(!routed){
        routed = true;
        neighbors.removeIf(n -> n.address.equalsIgnoreCase(previousHop));
        return neighbors;
    }else{
        return new ArrayList<>();
    }
}
```

Figure 1: Flooding Algorithm

Naive Minimum Cost Algorithm: This algorithm checks if the packet arrived from somewhere and then removes the 'previousHop' from the neighbors lists. Then chooses the least cost path from the neighbors list and returns it

```

public List<NeighborInfo> selectNeighbors(String origin, String destination, String previousHop,
                                         List<NeighborInfo> neighbors) {
    // Your code goes here.
    if(previousHop != null){
        neighbors.removeIf(n -> n.address.equalsIgnoreCase(previousHop));
    }
    List<NeighborInfo> chosen = new ArrayList<NeighborInfo>();
    chosen.add(neighbors.stream().min((n1,n2) -> Integer.compare(n1.cost, n2.cost) ).get());

    return chosen;
}

```

Figure 2: Naive Minimum Cost Algorithm

Minimum Cost Algorithm: We need to keep track of the paths that have been used and then return the least cost path. The algorithm uses an auxiliary array to store the possible paths. First, it checks if the packet came from somewhere. If it is, then that path is added to ‘excluded’ array then that path is removed from the auxiliary array. If not, then the lowest cost path is returned. After that the size of the auxiliary array is checked. If it has only one element, then that path is returned. If it is an empty array, then a random path is selected from neighbor’s array. If it does not satisfy these conditions, then the lowest cost path is selected from the auxiliary array. Finally, chosen path is added to the ‘excluded’ list.

```

public List<NeighborInfo> selectNeighbors(String origin, String destination, String previousHop,
                                         List<NeighborInfo> neighbors) {
    // Your code goes here.
    List<NeighborInfo> chosen = new ArrayList<NeighborInfo>();
    List<NeighborInfo> aux = neighbors.stream().collect(Collectors.toList());

    if(previousHop == null){
        chosen.add(aux.stream().min((n1,n2) -> Integer.compare(n1.cost, n2.cost) ).get());
    }else{
        excluded.add(previousHop);
        aux.removeIf(n -> excluded.contains(n.address));
        if(aux.size() == 1) chosen.add(aux.get(0));
        else if(aux.isEmpty()) chosen.add(neighbors.get((rand.nextInt()%neighbors.size())));
        else chosen.add(aux.stream().min((n1,n2) -> Integer.compare(n1.cost, n2.cost) ).get());
    }

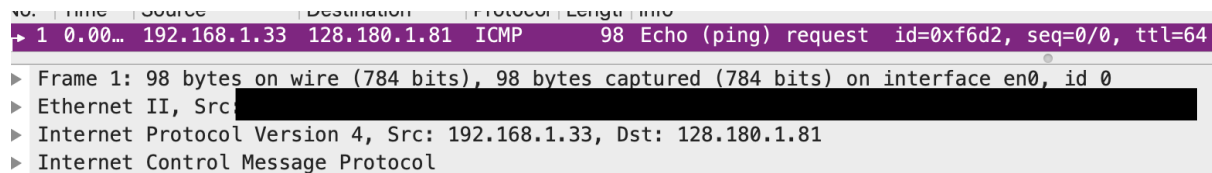
    chosen.forEach(n -> excluded.add(n.address));
    return chosen;
}

```

Figure 3: Minimum Cost Algorithm

Questions:

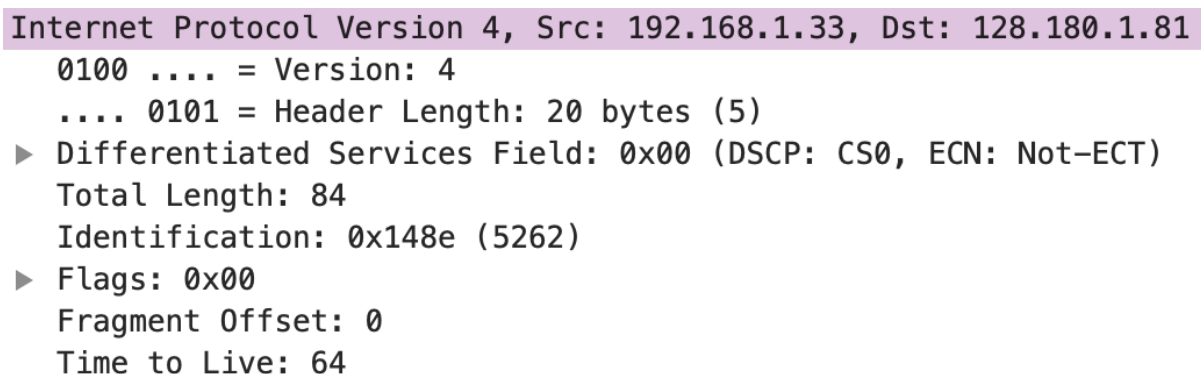
- 1) There are a total of 4 layers if you include the data. The first layer is Ethernet-II protocol which is a link layer protocol. Second layer is Internet Protocol (IP) which is in network layer. Third layer has the ICMP and the data. (Figure 4)



No.	Time	Source	Destination	Protocol	Length	Info
1	0.00...	192.168.1.33	128.180.1.81	ICMP	98	Echo (ping) request id=0xf6d2, seq=0/0, ttl=64
▶ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface en0, id 0						
▶ Ethernet II, Src: [REDACTED]						
▶ Internet Protocol Version 4, Src: 192.168.1.33, Dst: 128.180.1.81						
▶ Internet Control Message Protocol						

Figure 4: Three Layers of an ICMP Datagram

- 2) Time-to-Live flag specifies the maximum number of routers that a packet can transit. Its amount is decreased by one every 'hop' and the packet is dropped if it reached zero. It resides in the IP layer (Figure 5) and the value is constant if it is not specified from the user.



```
Internet Protocol Version 4, Src: 192.168.1.33, Dst: 128.180.1.81
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 84
  Identification: 0x148e (5262)
  ▶ Flags: 0x00
  Fragment Offset: 0
  Time to Live: 64
```

Figure 5: Internet Protocol header

- 3) That data is stored in the Internet Protocol layer because it is not designed to be used for communication between application process'. It is used alongside with IP and used for communication between hosts and routers.
- 4) The length of the whole ICMP part is 64 bytes but data only occupies the 48 bytes of it. The data holds all the information about the ICMP header.
- 5) The minimum value to reach the destination is 11.
- 6) Identifier and Sequence numbers are used to identify the reply of the ping request. If the values match with the reply, then that is the reply.

- 7) It is 8 bytes long. The ICMP packet consists of the IP and UDP parts of the request that we've send. (Figure 6)

```
5 1.55... 192.168.1.1 192.168.1.33 ICMP 94 Time-to-live exceeded (Time to live exceeded in transit)
Frame 5: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface en0, id 0
Ethernet II, Src: [REDACTED]
Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.33
Internet Control Message Protocol
  Type: 11 (Time-to-live exceeded)
  Code: 0 (Time to live exceeded in transit)
  Checksum: 0x3900 [correct]
  [Checksum Status: Good]
  Unused: 00000000
  ▶ Internet Protocol Version 4, [REDACTED]
  ▼ User Datagram Protocol, Src Port: 54009, Dst Port: 33435
    Source Port: 54009
    ▼ Destination Port: 33435
      ▶ [Expert Info (Chat/Sequence): Possible traceroute: hop #1, attempt #1]
      Length: 32
      Checksum: 0x664a [unverified]
      [Checksum Status: Unverified]
      [Stream index: 0]
      UDP payload (24 bytes)
    Data (24 bytes)
```

Figure 6: ICMP Exceeded Packet

- 8) If the server that we are pinging implemented the protocol than the probing is done by sending an empty packet with ttl=1 and the value is incremented by one after we receive a reply from the server. After we receive the reply the IP header shows the address of the intermediate routers. We keep on sending the packets till we reach the destination.
- 9) 3 packets are sent to probe each router along the path.
- 10) Delay from router 8 is significantly higher than 9. The only difference that can be observed is that their TTL field. Reply from 8 has a TTL value of 224 which is much higher than default, which is generally between 50-100.

```
tracert to www.lehigh.edu (128.180.1.81), 64 hops max, 52 byte packets
 1 192.168.1.1 (192.168.1.1) 2.262 ms 2.268 ms 1.945 ms
 2 212.156.201.191.static.turktelekom.com.tr (212.156.201.191) 15.415 ms 17.213 ms 10.350 ms
 3 81.212.2.147.static.turktelekom.com.tr (81.212.2.147) 10.572 ms 12.406 ms 9.489 ms
 4 00-gayrettepe-xrs-t2-2---00-buyukcekmece-t3-2.statik.turktelekom.com.tr (81.212.215.12) 9.256 ms 9.768 ms 9.271 ms
 5 10-tokmakrl-ess1-t4-1---10-beyduragirl-ess1-t4-1.statik.turktelekom.com.tr (81.212.26.71) 10.001 ms
   00-ebgp-gayrettepe-k---00-gayrettepe-xrs-t2-2.statik.turktelekom.com.tr (81.212.202.19) 9.672 ms
   10-tokmakrl-ess1-t4-1---10-beyduragirl-ess1-t4-1.statik.turktelekom.com.tr (81.212.26.71) 11.123 ms
 6 301-fra-col-2---00-gayrettepe-xrs-t2-2.statik.turktelekom.com.tr (212.156.101.196) 55.744 ms 55.860 ms 53.842 ms
 7 * * *
 8 ae-0-11.bar1.syracuse2.level3.net (4.69.141.241) 349.566 ms 200.840 ms 139.762 ms
 9 lehigh-univ.bar1.syracuse2.level3.net (4.26.25.78) 145.934 ms 148.531 ms 146.838 ms
10 * * *
11 * * *
```

Figure 7: Traceroute Log

- 11) Total communication cost represents all of the packets that have transmitted. Meaning that the packets have travelled 22 units and kept the lines busy.
- 12) Protocol does not converge because there are a total of 2 loops in the system, thus the packet transmitted indefinitely.
- 13) The path is expected but the total communication cost should be 24 which the value displayed is 28.
- 14) Because the algorithm doesn't converge. It gets stuck in a loop from 1->3->2. The algorithm always looks for the minimum cost path so the paths that connect 4 are much larger than the paths that intermediate nodes have (2-3).
- 15) This data is extracted using debug mode.
- Node1: 3,2,3
 - Node2: 3,1
 - Node3: 1,2,1,4
 - Node4: None