Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

CSU33031 Computer Networks
Flow Forwarding

November 1, 2023

1 Introduction

The focus of this assignment is to learn about decisions to forward flows of packets and the information that is kept at network elements that make forwarding decisions. The design of forwarding mechanisms aims to reduce the reduce the communication necessary to establish forwarding information while providing flexibility and fault-tolerance.

2 Protocol Details

Assume that you have been tasked to develop a reactive routing approach as an overlay that estashlishes routing information by broadcasting requests for locations of destinations that are not listed in current routing information (see figure 1). Examples for such an approach are ad hoc networks that are established in regions where a traditional infrastructure is not available, for example due to natural disasters [7, 1, 5].

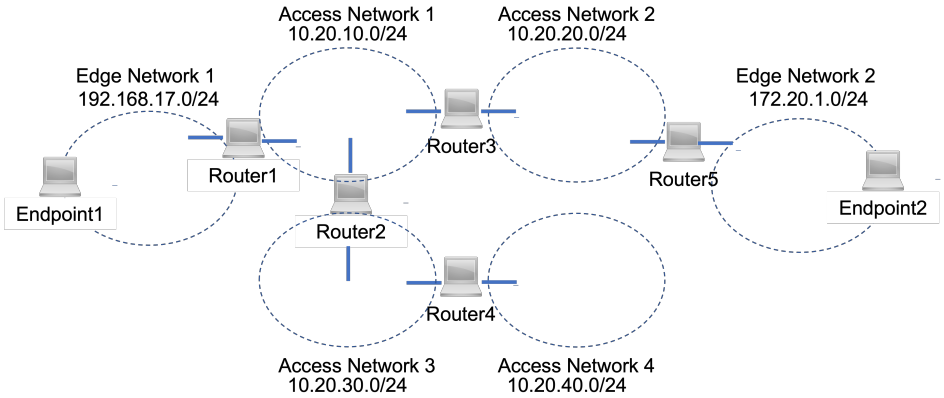

Figure 1: You will have a topology that consists of a number of networks and network elements. The networks at the edge of the topology can have endpoints of communication and the networks are connected by nodes that have multiple interfaces, one for every network they are connected to.

The network elements in this overlay are identified by addresses that are not tied to IP addresses. in the examples below, I used 4-byte addresses; however, you could also choose to use 2-, 6- or 8-byte addresses. These addresses are randomly assigned to network elements and a network element can be located anywhere in the topolgy i.e. the networks are not hierarchically organised and addresses may be based for example on the hardware address of a network interface.
For this assignment, an application executing on an endpoint will connect to an implementation of routing mechanism that is located on another network element that acts as default gateway. The socket address

for this routing mechanism may be hardcoded in the application, read from a configuration file, etc. The communication component of the application receive a series of bytes and a destination address. The routing mechanism will broadcast[1] on networks that it is attached to (see figure 2).

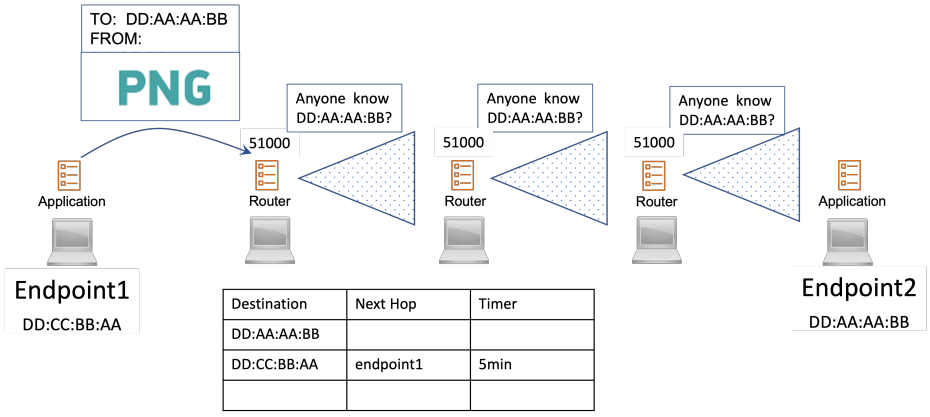

Figure 2: Initial contact by an application to a gateway and broadcasts of requests between routing mechanisms to find a destination specified in the information provided by the application.

An application on an endpoint receiving a broadcast containing its address will reply to the routing mechanism on its network, which will then reply to the routing mechanism that it received the request from it (see figure 3). The initial routing mechanism will transmit the packet that it received from the endpoint when it receives a reply to its broadcast for the location of the destination of packet from the endpoint (see figure 4). Routers along the path will learn the location of endponts from the traffic that passes through them. For example, routers that forward content to a destination, will learn from the header information the source of the content and integrate this knowledge in their forwarding table. This information will allow flows of packets back- and forwards between the original source and the destination. The information regarding flows at every router should have a livetime associated with it; so, that unused information can be removed by the routing mechanism from the forwarding tables.

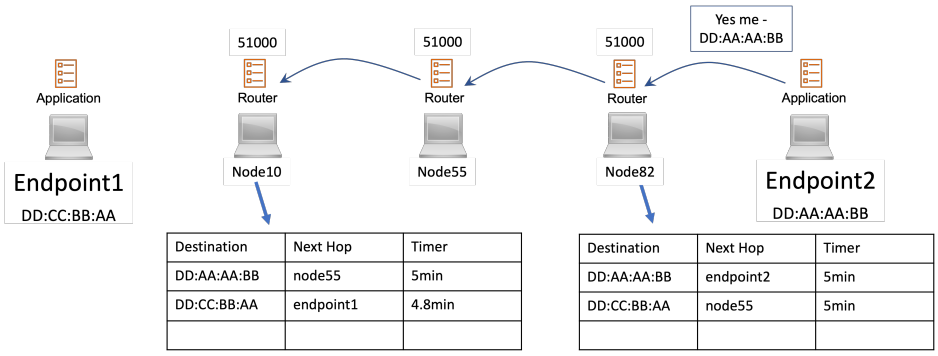

Figure 3: Reply from a destination and establishment of path.

In a first step, solve the communication betwee an endpoint and a router, then the broadcast of requests to resolve an address e.g. with 1 or 2 routers in your topology, then the response to such a request, then the forwarding of the original information. After your implementation can deliver content to a destionation, I would suggest to increase the number of packets between endpoints and then the number of endpoints and routers in your topology.
One of the challenges of the assignment is handling exceptional cases e.g. when an address given as destination of a packet does not exist in the topology or a routing mechanism receiving a large number of packets while

[1] In some socket implementations, you will need to set a socket option to allow broadcast for UDP datagrams e.g. $s.setsockopt(scoket.SOL\_SOCKET, socket.SO\_BROADCAST, 1) for Python.$

attempting to establish a route to the destination of the packets. I would suggest to solve the establishing of a path for the flow of packets as a first step, then to address the delivery of content to a destination, then to increase the number of network elements, and then to address the edge cases. Part 1 of the assignment will focus on the establishing on a path for the flow of packets, part 2 of the assignment will focus on increasing the complexity of the topology and for those who have time, may be the handling of edge cases.

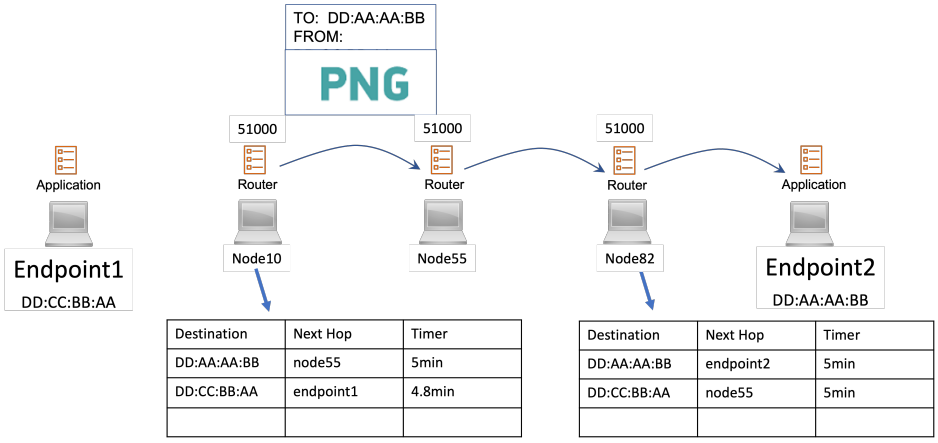

Figure 4: Delivery of content over an establish path.

The final implementation should allow for a number of endpoints to be started in a topology and communicate concurrently i.e. your routing mechanism needs to be able to handle a number of concurrent flows originating from endpoints and passing through your routing mechanisms. The initial implementation of the network elements for this assignment can be based on your implementation for assignment 1 i.e. the endpoints will have functionalities similar to the implementations of Producers and Consumers from assignment 1 and the routing mechanism will have functionalities similiar to those implemented in Brokers from assignment 1. The payload created by applications as part of this assignment is your choice e.g. you could reuse code from assignment 1 that transferred images as payload of your protocol.
The following flow diagrams, figure 5 is an example of visualizations of network traffic between components. They show the sequence of messages exchanged between components with the start of their transmission at the sender and their arrival at the receiver. Please use Flow Diagrams like these to document the communication between components of your implementation in your videos and in your final report.

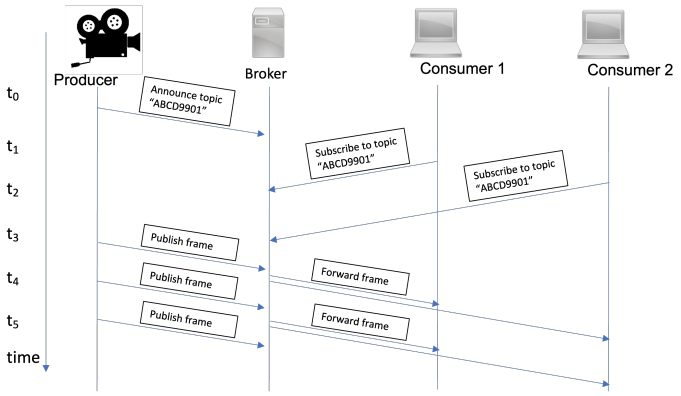

Figure 5: Taken from Assignment 1: The producer would contact the broker to announce a specific stream. The consumers send subscriptions to the broker for a stream and when the producer transmits frames of a stream to a broker, the broker will forward these to consumers that have subscribed

Following on from assignment 1, you should be familiar with virtual infrastructures such as Docker [2], Mininet [6], etc. Your final solution for assignment 2 should be demonstrated on a topology in these virtual

infrastructures consisting of a number of seperate broadcast domains.The protocol can be implemented in a programming language of your choosing. One of the conditions is that the communication between the processes is realized using **UDP sockets and UDP Datagrams**. Please avoid Python 2.7 because the implementation of Datagram sockets in the obsolete versions is based the transfer of strings instead of binary arrays.
You can use Generative AI tools such as GitHub Copilot[2] or OpenAI's ChatGPT[3]. You have to indicate in your videos and reports material that has been created with the help of these tools and you are responsible for any potential inaccuracies that this may introduce.

3 Deliverables & Submission Details

The deliverables for this assignment are split into 3 parts: 2 videos, a report describing your solution and the files making up the implementation of your solution. The deadline for the submission of these deliverables are given in Blackboard.
One component of the deliverables at every step is the submission of captures of network traffic. These captures should be in the form of PCAP files [3]. Programs such as Wireshark [4], tshark, etc offer functionality to capture network traffic from interfaces. The videos and reports should NOT contain your student ID.

3.1 Part 1: Video & PCAP file

The video of part 1 should demonstrate the initial design of your solution and a capture of network traffic between the components of your solution and the files of your traffic capture in pcap format. In the video, you should explain the setup of the topology that you are using and the information that makes up the header information in your traffic captures.
The submission process for this part consists of two steps: 1) Submitting the PCAP file or files that you captured from your network traffic, and 2) submitting the video for this step.
The video is to be no longer than 4 minutes; content past the 4 minute-mark will be ignored during marking. Videos with voice-over using text-to-speech (TTS) will not be accepted and marked with 0.

3.2 Part 2: Video & PCAP file

The video of part 2 should demonstrate the current state of your solution, the functionality that you have implemented so far, and a capture of network traffic between the components of your solution and the files of your traffic capture in pcap format. In the video, you should explain the basic implementation fo your protocol and the information that is being exchanged between the components of your solution.
The submission process for this part consists of two steps: 1) Submitting the PCAP file or files that you captured from your network traffic, and 2) submitting the video for this step.
Videos with voice-over using text-to-speech (TTS) will not be accepted and marked with 0. The video is to be no longer than 4 minutes; content past the 4 minute-mark may be ignored during marking.

3.3 Final Deliverable

The final deliverable should include a report that describes the components of your solution and their functionality, the protocol that you implemented and the communication between the components of your solution, the topology that you used to run you solution and how your solution was executed.
The submission process for this part consists of three steps: 1) Submitting the PCAP file or files that you captured from your network traffic, 2) submitting the source code and any files that may be necessary to execute your solution, and 3) submitting a PDF copy of your report about your solution.
The files that contain the implementation and the report should be submitted through Blackboard. Every source file should contain the name of the author. The source files of the implementation should be submitted as an archived file e.g. ".zip" or ".tar.gz". The report should be submitted as a PDF document. The deadline for the submission is given in Blackboard.
The report may be submitted to services such as TurnItIn for plagiarism checks.

[2] `https://github.com/features/copilot`
[3] `https://openai.com/chatgpt`

4 Marking Scheme

The contribution of the assignment to the overall mark for the module is 30% or 30 points. The submission for part 1 and 2 will be each marked out of 5 points and the submission for the final part will be marked out of 20 points. The mark for the final deliverable will be split into 50% for the functionality of your solution and 50% for the documentation through the report.

References

[1] Samir R. Das, Charles E. Perkins, and Elizabeth M. Belding-Royer. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.

[2] Docker. Docker project page. https://www.docker.com, visited Sep 2021.

[3] Wikipedia Editors. pcap - wikipedia page. https://en.wikipedia.org/wiki/Pcap, visited Aug 2021.

[4] Wireshark Foundation. Wireshark project page. https://www.wireshark.org, visited Sep 2021.

[5] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *Proceedings of the IEEE International Multi Topic Conference (IEEE INMIC 2001)*, pages 62–68, Lahore, Pakistan, December 2001.

[6] Mininet. Mininet project page. http://mininet.org, visited Sep 2021.

[7] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, pages 90–100, New Orleans, LA, USA, February 1999.