Simulation of Layer 2 and Layer 3 layers of the Network using using the client-server socket programming paradigm

Ronast Subedi (RS22CE)

Abstract:

A Network Emulator is a virtual software that provides a reliable way to test the performance of network applications through quick configuration and deployment. In this work, we designed a network emulator to simulate the message passing between the source and the destination. We implemented fully functional bridges, routers, and stations in Python programming language to achieve this. The connection links between network components were created using connection-oriented TCP socket connections. The designed network emulator supports Layer 2 and Layer 3 of the OSI Model (Open Systems Interconnection Model). Specifically, our emulator supports ARP and IP protocols. We have also implemented functionality to remove stations that are connected to the bridge but have remained idle for more than the specified time interval. The designed network emulator has all the functionalities mentioned in the project description.

Introduction:

A Network Emulator assesses the performance of real-world applications in a virtual network. Distinguished from network simulation, which involves virtual models of various network components, a network emulator accurately evaluates application responsiveness, throughput, and end-user experience. Network emulators are available as both hardware and software solutions. They are used for quality assurance, troubleshooting, and proof of concept and aid network architects, engineers, and developers in making informed decisions about technological changes. In this project, we develop a network emulator to simulate the message-forwarding process from source to destination. The major components of the emulator are bridges, stations, routers, and links connecting them.

We leverage the Address Resolution Protocol (ARP) to discover the link layer's Media Access Control (MAC) addresses of the target station in the same LAN [3]. ARP plays a crucial role in establishing a correspondence between a dynamic Internet Protocol (IP) address and a static physical machine address, known as the MAC address, within a local-area network (LAN). The utilization of ARP is instrumental in transmitting messages, with an emphasis on broadcasting messages to nodes that are intended recipients. This approach ensures the efficient and targeted dissemination of messages within the network, enhancing the overall effectiveness of packet-forwarding mechanisms.

In our project, we also optimize the bridge's ports' utilization by identifying and removing idle stations. This ensures that network resources, particularly the ports of the bridge, are efficiently allocated and reserved for actively communicating entities. This contributes to the overall performance and responsiveness of the network, aligning with our project's objective of creating an effective and resource-efficient network simulator.

Related Works:

Ethernet switches are network hardware and have a significant role in networking and the internet. They relay Ethernet frames between connected devices and link traffic from individual network

connections into a broader Ethernet network[1, 2]. The switch identifies the station/router connected to it by examining the MAC address in the received frame. If the MAC address is not in its cache, the switch broadcasts the received frame to all the ports except the incoming port. The destination station/router accepts the frame while other stations drop it.

Every device connected to the Local Area Network(LAN) is assigned a unique IP address for communication and identification[5]. The sender needs to know the IP address of the destination before initiating the communication. Additionally, the MAC address of the receiver is also required. Consulting its routing table, the sender first determines if the destination is in the same LAN. Then, the sender checks if its ARP cache contains the entry for the destination IP address. The ARP cache table retains a comprehensive record of each IP address and its associated MAC address. If the ARP cache has the entry, then the sender sends the message to the destination or the gateway. Otherwise, the sender sends the request to obtain the MAC address by broadcasting the ARP request to all machines in the LAN. On recognizing its IP address, the receiver sends an ARP response containing its MAC address. The sender updates its ARP cache and then sends the IP packet in the pending queue.

Overview and Methodology:

In this network emulator, hosts/routers can directly connect to the bridge/switch through one of the bridge's ports. The number of stations that can be connected to the bridge depends on the ports available in the bridge. The main function of the bridge is to deliver the data frame to the host in the LAN. Bridge supports multicasting as well as broadcasting. The bridge maintains a bridge table and uses it to store the mapping between the MAC address of connected hosts and the port to which the particular host is connected to. The Ethernet frames contain a header including source and destination (next-hop) MAC addresses. Upon receiving the frame, if the bridge does not have mapping for the destination MAC address, then the bridge broadcasts the frame to all the ports except the incoming port. Bridge also adds the ***MAC:port*** mapping of the sender to its table when it first receives the frame from that sender. This phenomenon is known as self-learning. When forwarding frames, the bridge only sends them to the necessary segments of the LAN, which optimizes the communication process. Stations and routers accept data frames addressed to them and discard any frames not intended for their MAC address. Additionally, the bridge also periodically checks its bridge table and removes the entries that are inactive for a time period longer than the specified duration.

Routers are the layer 3 devices, and they interconnect two or more LANS. Stations communicate with each other using the Internet Protocol. Every station and router has unique IP addresses. The station/router forwards IP packets hop-by-hop based on its routing table. Stations accept IP packets addressed to them and discard others, while routers forward IP packets to their next hops based on destination IP addresses. Each router and station maintains its ARP(Address Resolution Protocol) cache and routing table. ARP is used to identify the MAC address associated with an IP address. If a mapping is not in the cache, then the station/router adds the IP packet to the queue and sends an ARP request. Upon receiving the ARP response, the station/router processes the packets in its pending queue. If an IP packet is destined for the local station, it is displayed; otherwise, it is dropped.

In this project, we implement a bridge as a TCP socket server. When the bridge first starts, it stores its IP address and port number(of a socket) in a file and is ready to listen to the connections. Bridges wait for connection setup requests from stations. Routers and stations are realized as TCP socket

clients. A station/router is connected to a LAN. So, when the station/router is initialized, it reads the socket IP and port of the corresponding bridge and tries to connect to the bridge's socket. If the connection is successfully established, the bridge sends a message indicating that the connection is successful to the station/router. Otherwise, a message mentioning connection failure is sent.

When a station/router starts, it loads 3 files containing its information, routing table, and LAN information. It then initializes the necessary data structures, connects to LANs specified in the interface file, and waits for bridge connection setup requests. The main tasks of the station are sending and accepting messages and handling control messages such as ARP. Stations support two layers: the IP layer and the MAC layer. IP packets are encapsulated in the IP layer, which consults the routing table to determine the next-hop IP address. The MAC layer uses ARP protocol to obtain MAC addresses corresponding to the IP addresses for frame transmission. In this project, a router and a station have similar functionalities, except the router is connected to multiple bridges and forwards packets not destined for it.

Experiments and Results:

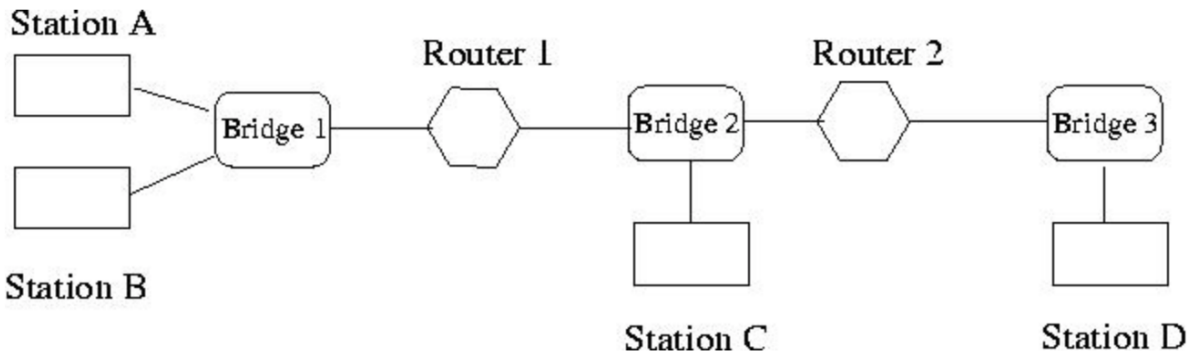

Fig1: Overview of the network simulator.

We implemented the network emulator using Python3.9 programming language. In our experiment, we designed the network topology, as shown in Figure 1. The topology consists of 2 routers, 3 LANs, and 4 stations. A LAN is realized through a bridge. Stations A and B are in LAN1, C is in LAN2, and D is in LAN3. Router 1 connects LAN1 and LAN2, whereas Router 2 connects LAN2 and LAN3. TCP socket connections were created to implement the physical links among the network components (stations, bridges, routers). Below, we show the complete process when two stations on different LANs communicate. We consider a scenario when Station A sends data to Station D.

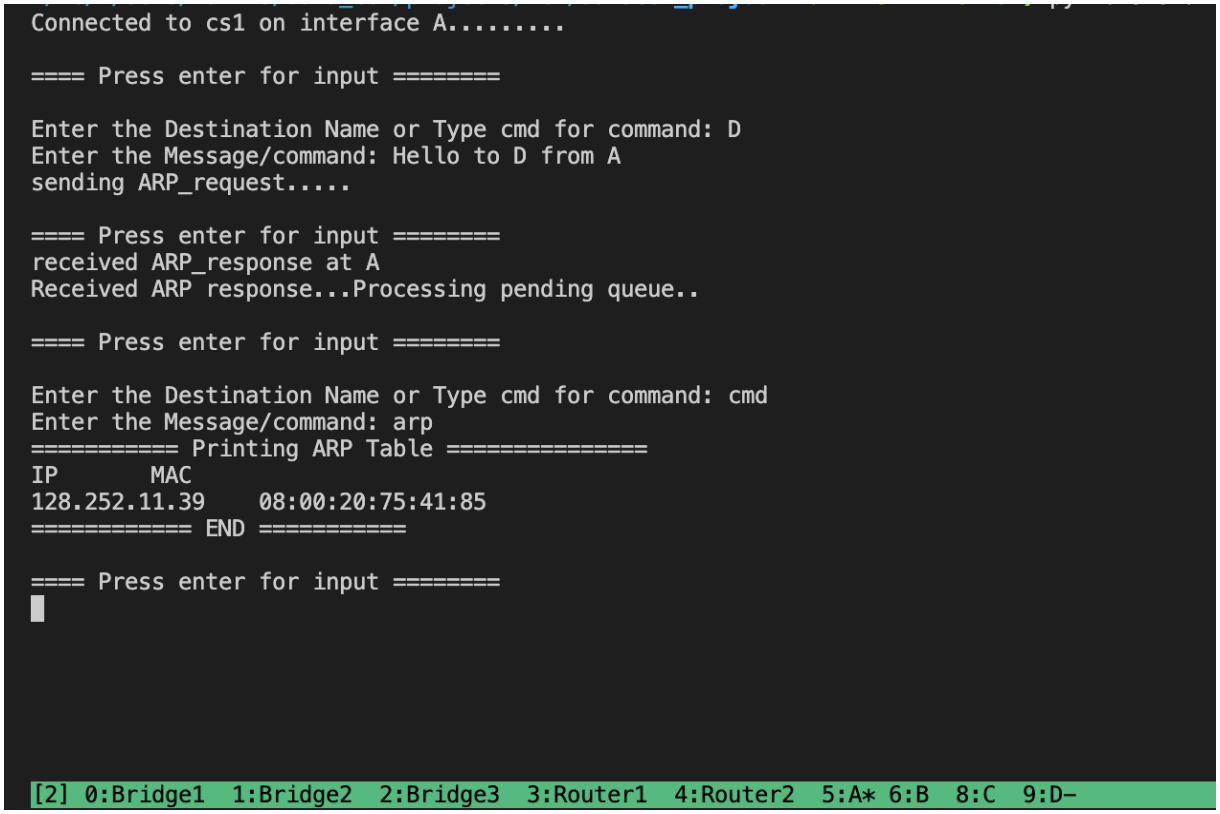

Fig2: Station A

Figure 2 shows the events happening in station A when it sends the message to station D. From its routing table A finds that D is in a different LAN. It then obtains the next hop IP (of Router 1) through which D can be reached with the help of its routing table. Initially, the ARP table of A does not contain the IP: MAC mapping of the corresponding interface of Router 1. So, A enqueues the packet in the pending queue and sends the ARP request to Router 1. In ARP request, A sets the destination MAC to be ***ff:ff:ff:ff:ff:ff***. The message gets to bridge 1, which does not have ***MAC:port*** mapping. So, it broadcasts the message to Router 1 and Station B.

Meanwhile, bridge 1 also adds the entry for station A in its bridge table. As the corresponding interface of Router 1 is the destination, Router 1 replies back with the ARP response, while station B drops the message. After station A receives the ARP response, it updates its ARP cache, checks its pending queue, and forwards the necessary packet, adding the correct destination MAC. The packet reaches Router 1, and Router 1 also follows a similar process. In this way, through hop-by-hop forwarding, the message reaches destination D. Figures 3,4,5,6,7, and 8 show the actions performed in every other network component when they receive the message.

```
Server started!!
=== Hostname: 127.0.0.1 Listening on Port: 50546 ===
Press enter for input:
Station connected at port 0.......
Press enter for input:
Station connected at port 1.......
Press enter for input:
Station connected at port 2.......
Press enter for input:
Destination mac not in the bridge table...broadcast
Press enter for input:
Destination mac in the bridge table...unicast
Press enter for input:
Destination mac in the bridge table...unicast
Press enter for input:

Enter the Destination Name or Type cmd for command: cmd
Enter the Message/command: bt
        MAC             Port
00:00:0C:04:52:27               1
08:00:20:75:41:85               0
Press enter for input:
[2] 0:Bridge1* 1:Bridge2  2:Bridge3  3:Router1  4:Router2  5:A  6:B  8:C  9:D-
```

Fig3: Bridge 1

```
Connected to cs1 on interface R1-cs1.........
Connected to cs2 on interface R1-cs2.........

==== Press enter for input ========
received ARP_request at R1-cs1
sending ARP_response.....

==== Press enter for input ========
Packet for host in different lan
sending ARP_request.....

==== Press enter for input ========
received ARP_response at R1-cs2
Received ARP response...Processing pending queue..

==== Press enter for input ========

Enter the Destination Name or Type cmd for command: cmd
Enter the Message/command: arp
=========== Printing ARP Table ===============
IP       MAC
128.252.13.38    08:00:20:75:41:88
============ END ===========

==== Press enter for input ========
[2] 0:Bridge1- 1:Bridge2  2:Bridge3  3:Router1* 4:Router2  5:A  6:B  8:C  9:D
```

Fig4: Router 1

```
Server started!!
=== Hostname: 127.0.0.1 Listening on Port: 50547 ===
Press enter for input:
Station connected at port 0.......
Press enter for input:
Station connected at port 1.......
Press enter for input:
Station connected at port 2.......
Press enter for input:
Destination mac not in the bridge table...broadcast
Press enter for input:
Destination mac in the bridge table...unicast
Press enter for input:
Destination mac in the bridge table...unicast
Press enter for input:

Enter the Destination Name or Type cmd for command: cmd
Enter the Message/command: bt
        MAC             Port
00:00:0C:04:52:27               0
08:00:20:75:41:88               1
Press enter for input:
[2] 0:Bridge1  1:Bridge2* 2:Bridge3  3:Router1- 4:Router2  5:A  6:B  8:C  9:D
```

Fig5: Bridge 2

```
Connected to cs2 on interface R2-cs2.........
Connected to cs3 on interface R2-cs3.........

==== Press enter for input ========
received ARP_request at R2-cs2
sending ARP_response.....

==== Press enter for input ========
Packet is destined for network in the next interface....
Destination ip not in the arp table....
sending ARP_request.....

==== Press enter for input ========
received ARP_response at R2-cs3
Received ARP response...Processing pending queue..

==== Press enter for input ========

Enter the Destination Name or Type cmd for command: cmd
Enter the Message/command: arp
=========== Printing ARP Table ===============
IP       MAC
128.252.13.67    00:00:0C:04:52:67
============ END ===========

==== Press enter for input ========
[2] 0:Bridge1  1:Bridge2- 2:Bridge3  3:Router1  4:Router2* 5:A  6:B  8:C  9:D
```

Fig6: Router 2

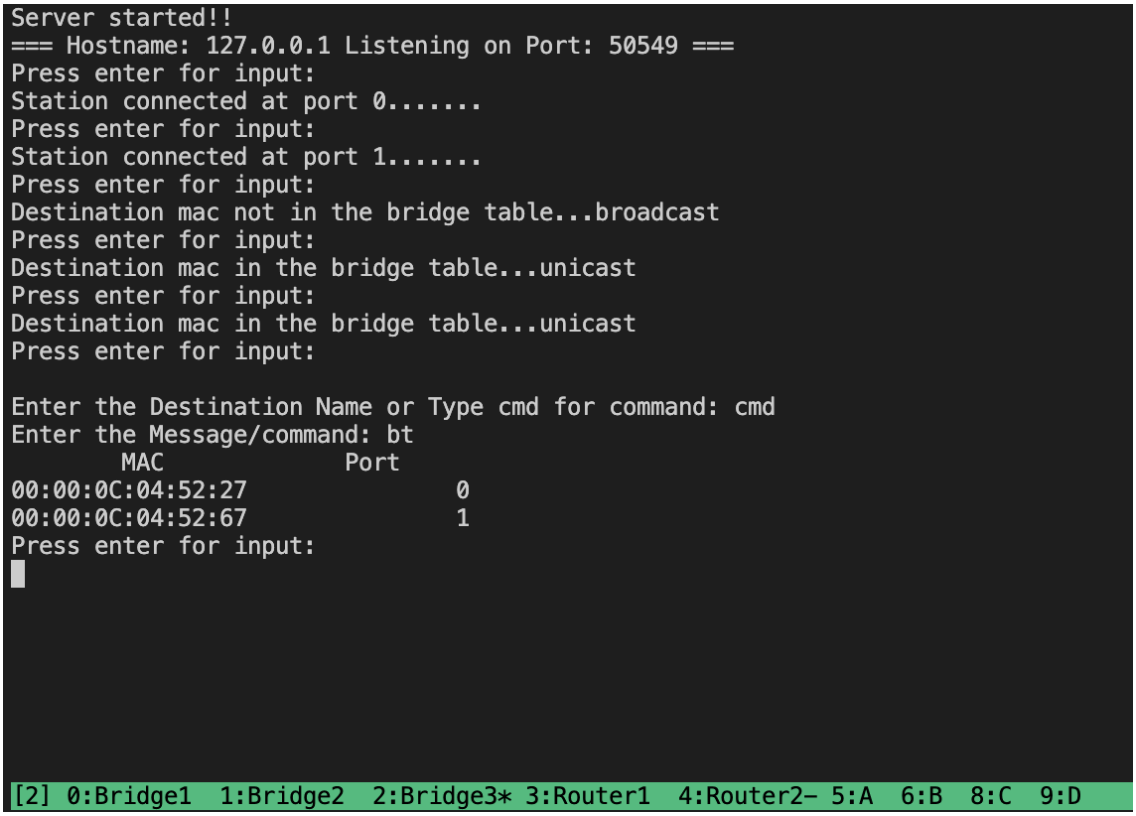

Fig7: Bridge 3

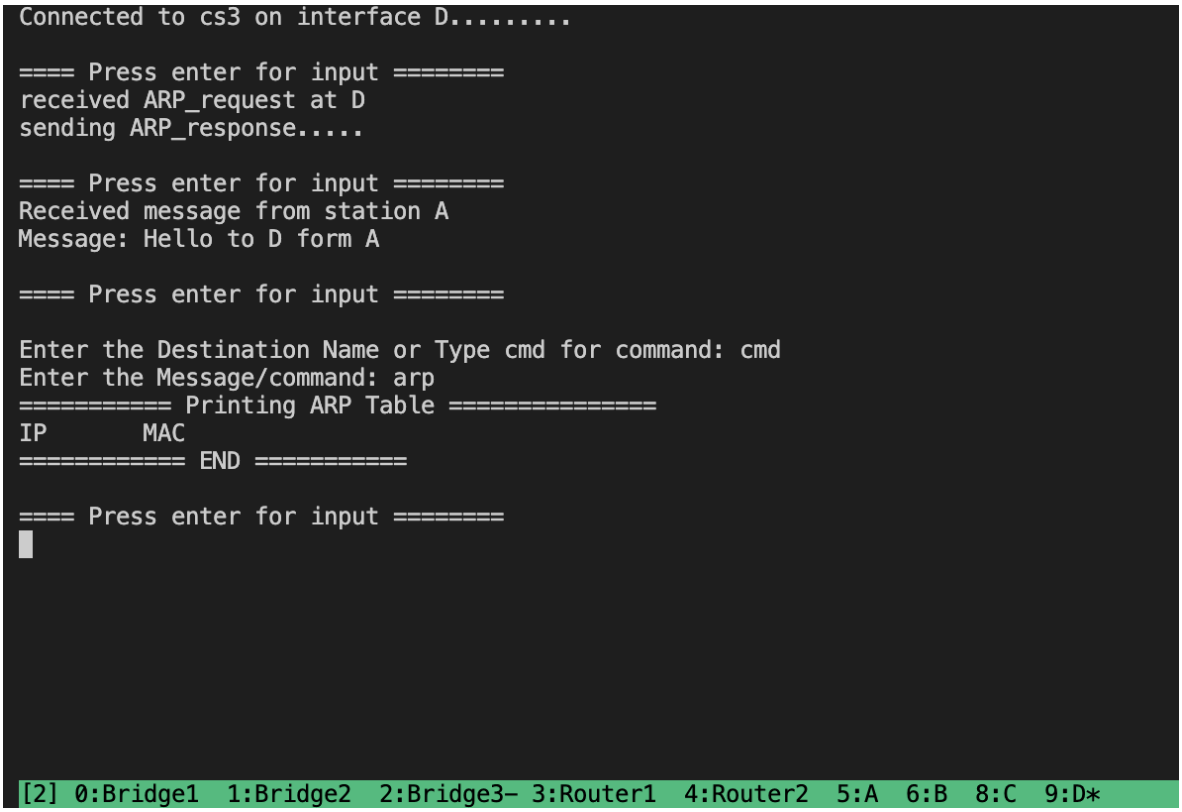

Fig8: Station D

From the simulation above, we verify that the network components are functioning as expected. Message from sender A was successfully delivered to the receiver D.

Conclusion:

To summarize, we have developed a network emulator to study the message-passing process from the source to the destination. The emulator supports any number of bridges, routers, and stations. The I/O multiplexing in the network components is achieved by using ***select()*** function. A station supports two layers: the MAC Layer and the IP Layer. The designed emulator performs hop-by-hop message passing and adheres to Address Resolution Protocol(ARP) for obtaining the MAC address of the destination in the local LAN. Additionally, the bridge periodically checks the inactive stations connected to it and removes them to free the bridge port after the specific time interval. Moreover, the ARP cache is also periodically monitored, and inactive entries are terminated after a timeout. Hence, the emulator is fully functional and supports Layer 2 and Layer 3 of the OSI network model.

References:

1. Andrew S. Tanenbaum, Nick Feamster, and David Wetherall, “Computer Networks,” Prentice Hall, 6th edition, 2020
2. Joann Zimmerman, Charles E. Spurgeon, “Ethernet Switches,” O'Reilly Media, April 2013
3. Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <https://www.rfc-editor.org/info/rfc826>.
4. B. Issac and L. A. Mohammed, "Secure unicast address resolution protocol (S-UARP) by extending DHCP," 2005 13th IEEE International Conference on Networks Jointly held with the 2005 IEEE 7th Malaysia International Conf on Communic, Kuala Lumpur, Malaysia, 2005, pp. 6 pp.-, doi: 10.1109/ICON.2005.1635503.
5. https://www.techtarget.com/searchnetworking/definition/Address-Resolution-Protocol-ARP
6. Wikipedia: https://www.wikipedia.org/