CSE 452: Computer Networks Sessional

January 2022

Assignment 3

ARP

1 Introduction

ARP is a widely used protocol in network communication. It is used for finding the MAC address of a network device. In this assignment, first, we will learn the details of the ARP protocol then we will implement it using c++.

1.1 Definition

ARP (Address Resolution Protocol) is used for finding the MAC (Media Access Control) address of a network device. In a LAN (Local Area Network) all the communication is performed by using a MAC address. Let us imagine a scenario where we know a device's IP address but we do not know the MAC address of the device. In this case, we use ARP to get the MAC address of that device. Every end device has an ARP cache in which they store the IP address and its corresponding MAC address. If our desired IP address and MAC address pair are present in the ARP cache then we use the MAC address from the cache otherwise we use ARP messages to get the MAC address and store it in the ARP cache. Normally, the ARP cache is periodically flushed. But for this assignment, we will not implement the flushing mechanism of the ARP cache. Let us see the two ARP messages-

1. **ARP Request:** The source device will broadcast the ARP request message to the local network by specifying the requested IP and giving ff:ff:ff:ff:ff:ff as the destination MAC address. ff:ff:ff:ff:ff:ff is a broadcast MAC address so it will be received by all the other devices that are connected to the LAN network.
2. **ARP Reply:** Each device will compare the requested IP address with its IP address. If it finds matches then the device will send an ARP Reply message otherwise drop the message.

The messages are summarized in Figure [1]. For more details please read this article.

Prerequisite:

You need to understand the previous assignments to do this assignment. You can review the server, client architecture and socket programming. Do not worry we will provide so many functions to make this assignment easier for you.

2 Assignment Description

For simplicity, we will not develop the actual scenario here. We will develop ARP in a reduced scenario. Let us think of a server that will simulate the work of a switch. We can think of all the end devices such as laptops, desktops, and mobile phones as the clients. In the client part, you need to implement the ARP protocol so that one client can resolve a MAC address by using the specified IP address of the other client.

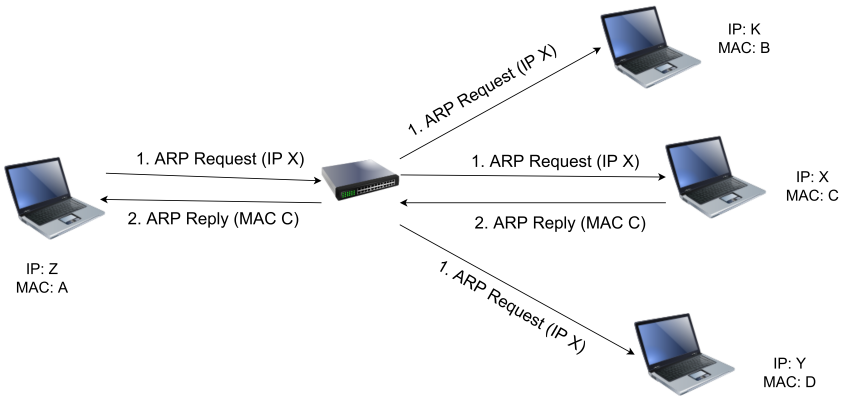

Figure 1: Schematic diagram of ARP messages.

Figure [2] shows a standard Ethernet frame header and Figure [3] shows the a standard ARP payload. But we do not use the whole frame format for this assignment. We will use two struct of c that will simulate Ethernet frame header and ARP payload. Figure [4] and [5] show two struct that will be used in this assignment. All the tasks are described in detail below-

802.3 Ethernet packet and frame structure

Layer	Preamble	Start frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)	Payload	Frame check sequence (32-bit CRC)	Interpacket gap
	7 octets	1 octet	6 octets	6 octets	(4 octets)	2 octets	46-1500 octets	4 octets	12 octets
Layer 2 Ethernet frame			← 64–1522 octets →						
Layer 1 Ethernet packet & IPG	← 72–1530 octets →								← 12 octets →

Figure 2: Ethernet Header.

1. **MAC Address and IP address generation:** We have provided two functions to generate the MAC address and IP address randomly. Use them to generate the MAC address and IP address randomly when the client starts up and shows in the console.

2. **Server as a Switch:** In this assignment, the server will work as a switch. It will receive an Ethernet frame and based on the destination MAC address present in the frame it will forward the message to the appropriate device. If the destination MAC address is a broadcast message then the server will forward the message to all the devices connected in the network except the sender. The server also needs to track the MAC address that is associated with client SOCKET. In practice, the server has a MAC address table where it stores the port as well as the corresponding MAC address. In our assignment, we will also store the client socket to send frame easily. You need to implement this MAC address table. For populating the MAC address table, when the server receives a message it will save the source MAC address from the Ethernet frame along with its SOCKET, and assign a port for that client. You can give port number 0 for the first client, port number 1 for the second client, and so on. When the server gets an Ethernet frame, it first inspects the destination MAC address field. If the destination MAC address is the broadcast address then it will broadcast to all the ports except the

Internet Protocol (IPv4) over Ethernet ARP packet

Octet offset	0	1
0	Hardware type (HTYPE)	
2	Protocol type (PTYPE)	
4	Hardware address length (HLEN)	Protocol address length (PLEN)
6	Operation (OPER)	
8	Sender hardware address (SHA) (first 2 bytes)	
10	(next 2 bytes)	
12	(last 2 bytes)	
14	Sender protocol address (SPA) (first 2 bytes)	
16	(last 2 bytes)	
18	Target hardware address (THA) (first 2 bytes)	
20	(next 2 bytes)	
22	(last 2 bytes)	
24	Target protocol address (TPA) (first 2 bytes)	
26	(last 2 bytes)	

Figure 3: ARP packe.t

```
// Ethernet frame
struct ethernet_frame
{
    char preamble[7];                   // we will not use this field.
    char start_frame_delimiter;         // we will not use this field.

    char mac_destination[7];            // destination mac address
    char mac_source[7];                 // source      mac address

    char ethernet_type[2];              // we will not use this field.

    arp_payload payload;                // ARP         payload.
};
```

Figure 4: Ethernet Header Struct in c.

```
// ARP payload
struct arp_payload
{
    short hardware_type;                // we will not use this field.
    short protocol_type;                // we will not use this field.
    char hardware_address_length;       // we will not use this field.
    char protocol_address_length;       // we will not use this field.

    short operation;                    // put 0 for arp request and 1 for arp reply.
                                        // Please see the defined value in the top

    char sender_hardware_address[7];    // MAC address of sending machine
    char sender_protocol_address[5];    // ip of sender machine

    char target_hardware_address[7];    // MAC address of receiving machine
                                        // (Ignore this field in ARP request)
    char target_protocol_address[5];    // ip of receiver machine
};
```

Figure 5: ARP Payload Struct in c.

port from which it receives the frame. If the MAC address is not a broadcast MAC then it searches in the MAC address table. If it finds a port that has the same MAC then forward the message to that port, otherwise broadcast that frame. So, an Ethernet frame that does not contain a broadcast MAC in its destination MAC may also become a broadcast frame. Furthermore, You need to implement command executing functionality in the server. The server will wait for command and based on the command it will show messages. Commands are discussed in detail below-

(a) **Show MAC Address Table:** When a user types *sh mac address-table* server will print the whole MAC address table.

(b) **Remove all entries from MAC Address Table:** When a user type *clear mac address-table* server will delete the whole ARP table.

(c) **Exit:** When a user types *exit* server will terminate.

Note: When the server receives a frame, print the source MAC address and destination MAC address of that frame. The server can not access the payload of the Ethernet frame.

3. **ARP Resolution:** First, a client sends an ARP Request message to the server with the broadcast address in the destination MAC address. When a client receives an ARP Request message it checks the requested IP address with its IP address. If matches then it will send an ARP Reply message otherwise drop it. Each client also needs to maintain an ARP table. For populating the ARP table, when an ARP Request or an ARP Reply message is received by the client, it will save the source MAC address along with the source IP address. Even when a client drops an ARP Request, it will update its ARP table using the source IP and MAC address.

4. **Command in Client Part:** In this assignment, we will implement some commands to demonstrate the ARP protocol. There are in total four commands. When a client starts up it will wait for the command from the user and based on the command it will perform some tasks. The commands are described below-

(a) **Find MAC:** When a user types *find_mac ip_address* the client will first check whether it has a MAC address corresponding to this IP in its table. If it finds then print the MAC in the console, otherwise sends an ARP Request message. Example- *find_mac 192.168.10.1*
Note: This is not an actual command. So, you can not give it in any device for starting ARP request. It is taken for initiating the ARP messages in this assignment.

(b) **Show ARP Table:** When a user types *arp -a* client will print the whole ARP table.

(c) **Remove all entries from ARP Table:** When a user type *arp -d* client will delete the whole ARP table.

(d) **Exit:** When a user type *exit* client will terminate.

You need to print all the messages in both the server and client console. We have provided $server.exe$ **and** $client.exe$ **files. Please run them to see the print message formats.** We have also provided a $helper.cpp$ file. You can also find some other necessary functions in template files for doing this assignment.

3 Submission Guidelines

Make a folder by using your ID. Example 1706XYZ. Put all the files inside this folder. Please rename the server file as 1706XYZ_server.cpp and the client file as 1706XYZ_client.cpp then zip the folder and submit the zip file in teams. **Do not forget to Turn In after uploading the file**.

4 Deadline

August 22, 2022, 11:55 pm (Strict)

5 Tentative Mark Distribution

The tentative mark Distribution is given below:

Task	Mark
Task-1: Server as a Switch	15
Task-2: ARP Resolution	20
Task-3: Command in Client Part	15
Total	**50**

Table 1: Tentative Mark Distribution

6 Plagiarism

Do not copy from any web source or friends. Students involved in such activities will be severely penalized by awarding **-100%** of the total marks.

Prepared by Md. Ashraful Islam. Verified by Md. Toufikuzzaman and Sukarna Barua. Figure [2] and Figure [3] are taken from this article and this article respectively.