

데이터 통신 Assignment 05

과제 제출일자 : 2019/06/02

학번 : 201502085, 이규봉

분반 : 00

과제 : File Transfer 구현

실습 개요, 목적

byte들의 전송으로, 파일을 전송할 수 있는 프로토콜을 만든다. 파일 전송 프로토콜은 채팅 프레임 전송 프레임과 별도로, 동시에 전송될 수 있어야 한다.

실습 시나리오 (채팅 전송 측)

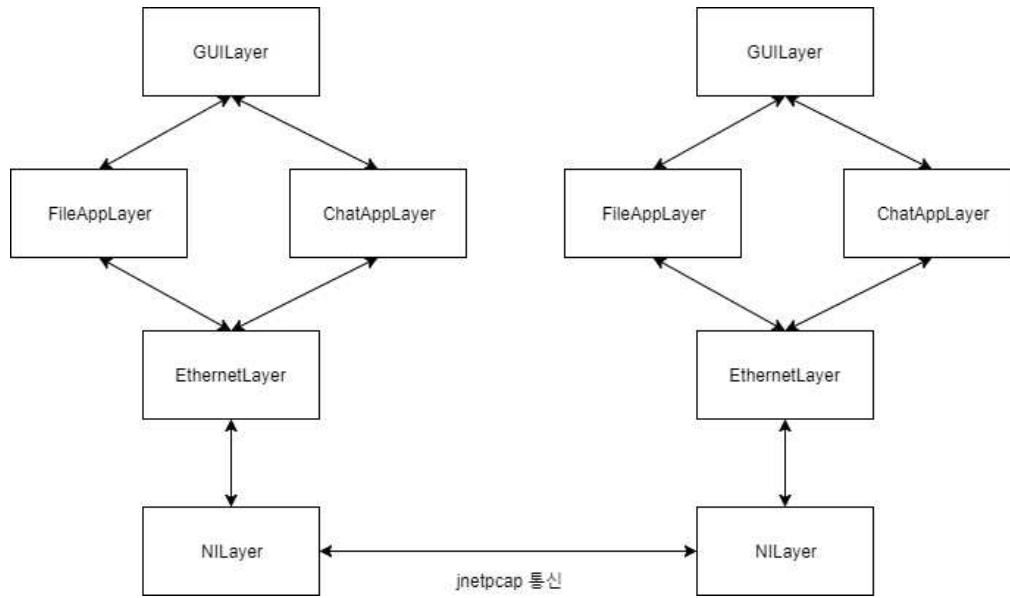
- 1) 프로그램을 실습할 두 대의 컴퓨터를 준비하고 랜선으로 연결함. 이더넷 네트워크 이외의 네트워크들을 모두 꺼줌.
- 2) Wireshark로 이더넷 드라이버에서 송수신 되는 패킷을 확인해, 컴퓨터끼리 연결이 잘 되었는지를 확인함. 패킷을 송신한 컴퓨터가 다시 상대 컴퓨터의 EthernetLayer에서 전송한 Ack 패킷을 받게 될 것임.
- 3) PC 1에서 Menu > Mac Address Setting 버튼을 클릭해, 주소 설정 창에서 NIC_ComboBox를 알맞게 설정하고, Destination Mac Address엔 상대 컴퓨터의 Source Mac Address를 직접 입력함.
- 4) PC 2에서도 똑같이 Mac Address를 설정한다. 서로 지정한 상대 컴퓨터의 Mac Address가 일치해야 한다.
- 5) 채팅을 전송할 프로세스에서 채팅창에 1456 바이트 이상의 길이의 String 데이터와 1456 바이트보다 작은 String 데이터를 키보드로 입력하고, 각각 Send 버튼을 클릭함
- 6) StopAndWaitDlg (GUI Layer)에서 ChatApp 레이어의 Send를 호출하고 String 데이터를 byte[] 로 인코딩 해, 전송함. 아래 레이어는 그 아래 레이어로 데이터를 전송하는데 이 때 해당하는 레이어의 정보를 헤더로 만들어 붙임.
- 7) ChatAppLayer의 송신 쓰레드에서 1456 바이트보다 작은 데이터는 (메시지큐를 거쳐) 바로 전송하고, 1456 바이트보다 큰 데이터는 메시지큐에 넣어놓았다가, 1456 바이트 씩 잘라 전송하고, 상대 프로세스가 전송한 Ack 신호를 받을 때 마다 다음 프레임을 전송함.
- 8) NILayer에서 jnetPcap을 이용해 랜선으로 데이터를 전송함
- 9) 수신하는 쪽 프로세스의 NILayer에선 랜선으로 들어온 데이터는 모두 들어오며, 위 레이어의 Receive를 호출.
- 10) Ethernet Layer 쪽 Receive 함수에서, Ethernet Frame을 확인해 들어온 데이터의 Dst Address가 본인 프로세스가 갖고 있는 Source Address와 일치하는지, 또는 들어온 데이터가 브로드캐스팅인지 확인해, 일치하는 경우 필요 없어진 Ethernet 헤더를 제거하고, 위 쪽 레이어로 남은 데이터를 올려 보냄. 데이터가 위 레이어로 성공적으로 올려 보내지면, 데이터를 정상적으로 수신했다는 의미로 Ack 패킷을 만들어 보냄.
- 11) ChatAppLayer에서 단편화 되지 않은 패킷은 바로 위 레이어로 올려 보내고, 단편화 되어 있는 패킷은 버퍼에 저장해 놓았다가 마지막 패킷을 수신했을 때 위 레이어로 버퍼의 내용을 올려보내고, 버퍼를 비움.
- 12) 결과적으로 최상위 레이어 (GUI) 측에선, 메시지를 송신한 프로세스 측이 보낸 데이터를 그대로 전달받게 됨. byte[]을 다시 String으로 UTF-8 로 인코딩한 후, Chatting Area 쪽에 append 해 수신한 메시지를 표시함.
- 13) 채팅 쓰레드는 파일 전송 쓰레드 및 GUI 쓰레드 (메인 쓰레드) 와 별개로 동작한다. 따라서, 채팅 큐에 메시지가 차 있어도 새 채팅 프레임을 만들 수 있으며, 파일 전송 중에도 채팅 프레임을 전송하는데 문제가 없다.

실습 시나리오 (파일 전송 측)

- 1) 프로그램을 실행할 두 대의 컴퓨터를 준비하고 랜선으로 연결함. 이더넷 네트워크 이외의 네트워크들을 모두 꺼줌.
- 2) Wireshark로 이더넷 드라이버에서 송수신 되는 패킷을 확인해, 컴퓨터끼리 연결이 잘 되었는지를 확인함. 패킷을 송신한 컴퓨터가 다시 상대 컴퓨터의 EthernetLayer에서 전송한 Ack 패킷을 받게 될 것임.
- 3) PC 1에서 Menu > Mac Address Setting 버튼을 클릭해, 주소 설정 창에서 NIC_ComboBox를 알맞게 설정하고, Destination Mac Address엔 상대 컴퓨터의 Source Mac Address를 직접 입력함.
- 4) PC 2에서도 똑같이 Mac Address를 설정한다. 서로 지정한 상대 컴퓨터의 Mac Address가 일치해야 한다.
- 5) PC 1에서 Menu > File Transfer 버튼을 클릭해, 파일을 전송할 용도로 만든 창을 띄운다.
- 6) PC 1의 File transfer 창의 File Select 버튼을 클릭해, 파일 선택 JFileChooser 객체를 만들어, 파일 선택 창을 띄운다.
- 7) 파일 선택 창에서 전송할 파일을 선택하고 열기 버튼을 클릭한다.
- 8) File Transfer 창 내 Selected File Path에 선택한 파일의 경로가 제대로 들어가 있는지 확인하고, Transfer 버튼을 클릭한다.
- 9) 파일의 이름을 담은 첫 번째 프레임이 상대방에게 전송된다. 상대 측 컴퓨터는 프레임을 받고 데이터를 String으로 만들어 파일을 저장할 때 이 String 데이터를 파일의 이름으로 쓰게 된다.
- 10) FileInputStream 객체를 생성해 선택된 파일을 읽어와 byte[] 에 저장한다.
- 12) 파일 전송 큐에 읽어온 파일 byte[]을 add 하고, 전송 쓰레드에 send_lock 객체에 notify 해, 파일 전송 쓰레드를 깨운다. (없으면 이 때 쓰레드 객체 생성)
- 13) 파일 전송 쓰레드는 1448 바이트보다 작은 데이터는 (파일 전송 큐를 거쳐) 바로 전송하고, 1448 바이트보다 큰 데이터는 파일 전송 큐에 넣어놓았다가, 1448 바이트 씩 잘라 전송하고, Thread.Sleep 으로 일정 시간 기다렸다가, 한 프레임 씩 상대 컴퓨터로 전송함. 아래 레이어는 그 아래 레이어로 데이터를 전송하는데 이 때 해당하는 레이어의 정보를 헤더로 만들어 붙임.
- 14) NILayer에서 jnetPcap을 이용해 랜선으로 데이터를 전송함
- 15) 수신하는 쪽 프로세스의 NILayer에선 랜선으로 들어온 데이터는 모두 들어오며, 위 레이어의 Receive를 호출.
- 16) Ethernet Layer 쪽 Receive 함수에서, Ethernet Frame을 확인해 들어온 데이터의 Dst Address가 본인 프로세스가 갖고 있는 Source Address와 일치하는지, 또는 들어온 데이터가 브로드캐스팅인지 확인해, 일치하는 경우 필요 없어진 Ethernet 헤더를 제거하고, 위 쪽 레이어로 남은 데이터를 올려 보냄.
- 17) 수신 컴퓨터 FileAppLayer에서 단편화 되지 않은 패킷은 바로 위 레이어로 올려 보내, 파일을 생성하고, 단편화 되어 있는 패킷은 버퍼에 FileFrame이란 객체 (순서와 바이트 데이터를 지님) 의 형태로 저장해 놓았다가 마지막 패킷을 수신했을 때 FileFrame 내 index 값, 즉 fapp_seq_number로 Collections.sort()를 이용해 정렬해 원래 파일의 데이터가 되게 한다. 그 후 FileFrame의 리스트를 배열 형태로 고치고, 위 레이어로 올려보낸다.
- 18) 결과적으로 최상위 레이어 (GUI) 측에선, 파일을 송신한 프로세스 측이 보낸 파일의 byte[]을 전달받게 됨. byte[]을 Files.write을 이용해 파일의 형태로 복원한다. 파일 종류나 확장자에 상관 없이 파일이 복원 (전송) 된다.
- 19) 파일 전송 쓰레드는 채팅 쓰레드 및 GUI 쓰레드 (메인 쓰레드) 와 별개로 동작한다. 따라서, File Transfer 중에도 새 채팅 프레임을 만들 수 있으며, 채팅 프레임 전송 중에도 파일을 전송하는데 문제가 없다.

프로토콜 스택의 구조, 각 프로토콜 (layer) 의 역할

제가 구현한 프로토콜 스택의 구조는 아래와 같습니다.



fapp_seq_number 로 정렬

저는 프레임들을 fapp_seq_number로 정렬해 GUILayer로 올려보내기 위해, 아래와 같이 FileFrame 클래스를 만들고, Comparable, compareTo를 구현해 Collections.sort()로 정렬하면 fapp_seq_number로 정렬하도록 만들었습니다.

```
private class FileFrame implements Comparable<FileFrame>{
    private byte[] bytes;
    private int index;

    public FileFrame(byte[] _bytes, int _index) {
        this.bytes = _bytes;
        this.index = _index;
    }

    @Override
    public int compareTo(FileFrame otherFrame) {
        if(this.index > otherFrame.index) {
            return 1;
        }
        else if(this.index < otherFrame.index) {
            return -1;
        }
        else {
            return 0;
        }
    }
}

// 한 개의 파일이 한 바이트 배열안에 모두 들어간다고 가정함 (파일의 크기가 2기가 이하라고 가정)
private byte[] sortBytesList(List<FileFrame> byteList, int fileLength) {

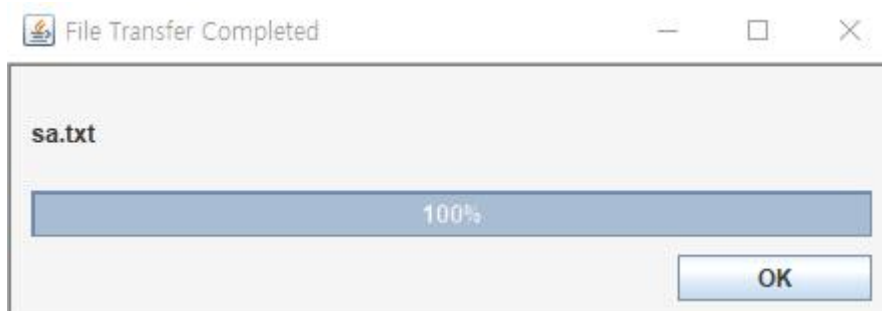
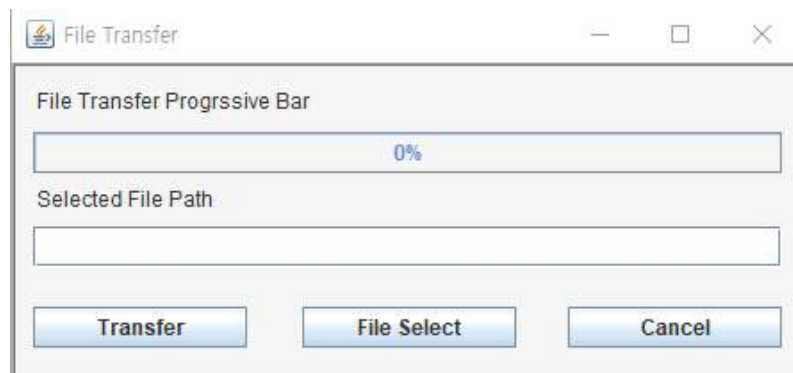
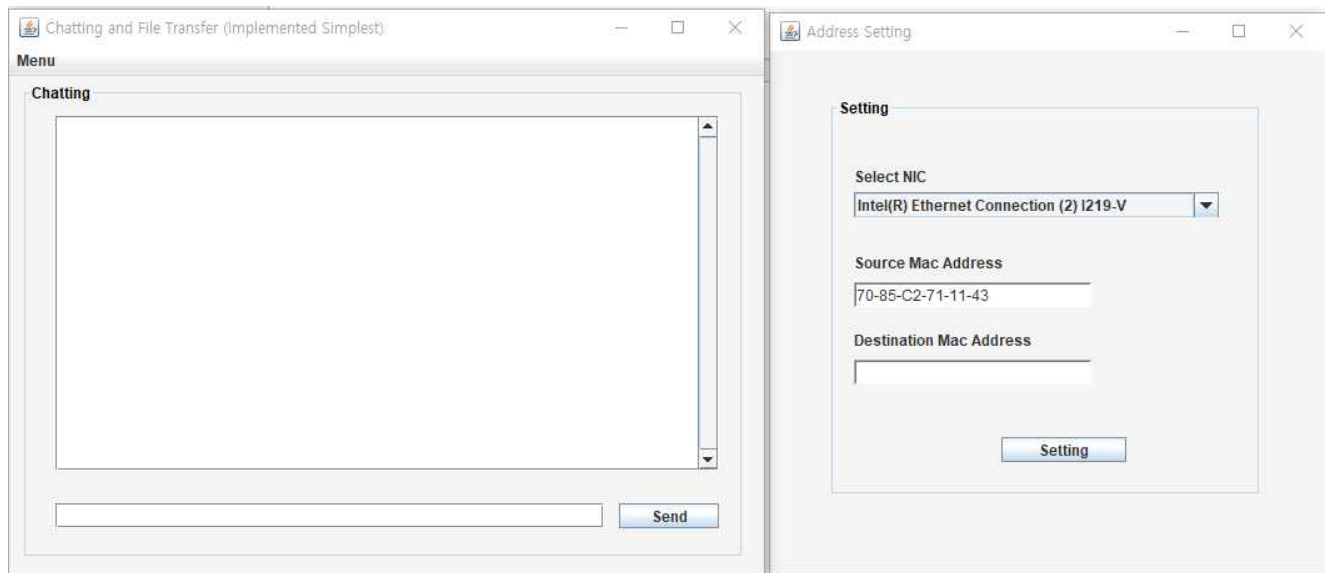
    byte[] result = new byte[fileLength];

    Collections.sort(byteList);

    for (int i =0; i < byteList.size(); i++) {
        FileFrame frame = byteList.get(i);
        for(int j = 0; j< frame.bytes.length; j++) {
            result[i * FILE_FRAGMENTATION_CRITERIA + j] = frame.bytes[j];
        }
    }

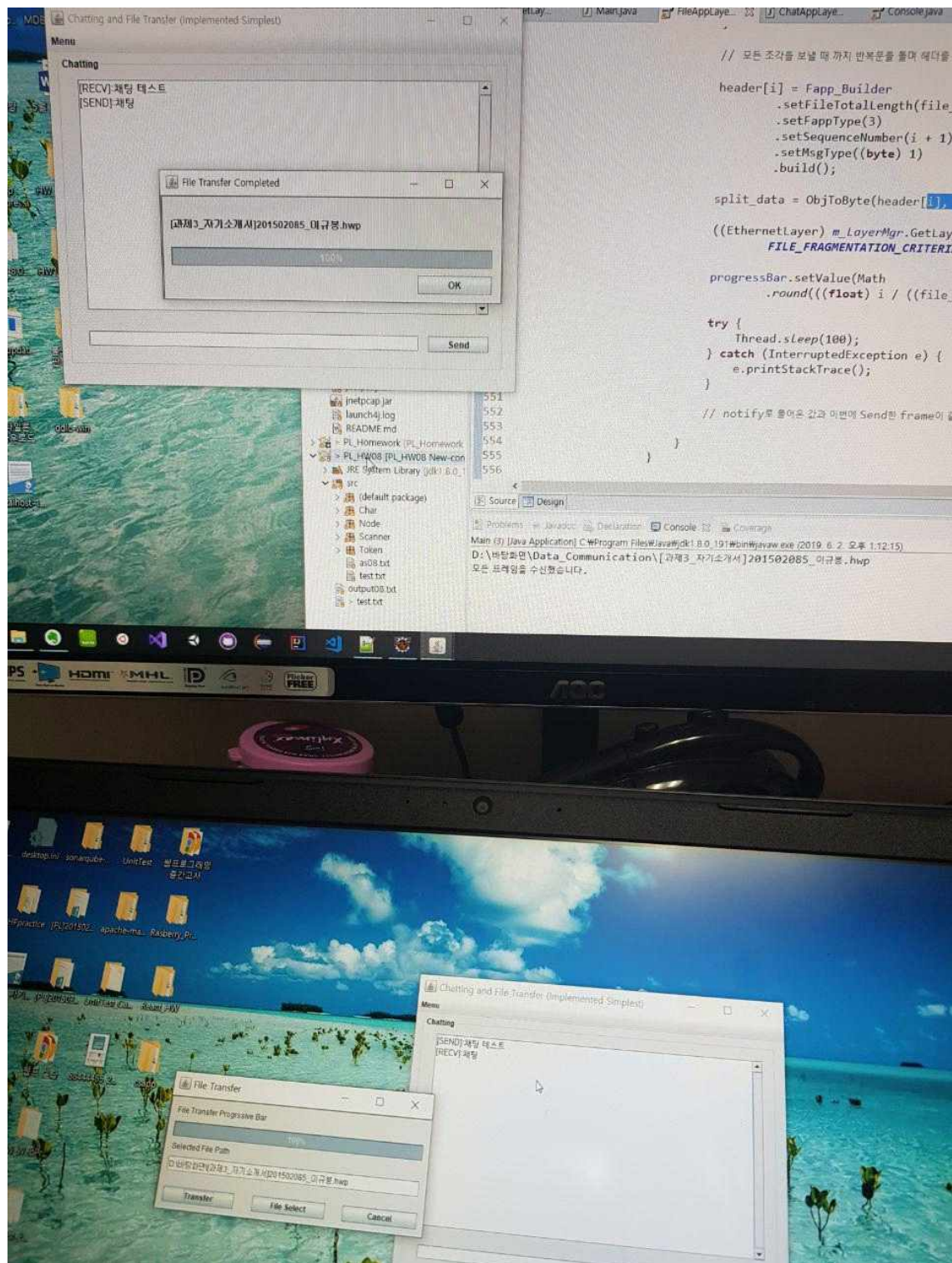
    return result;
}
```

프로그램 GUI



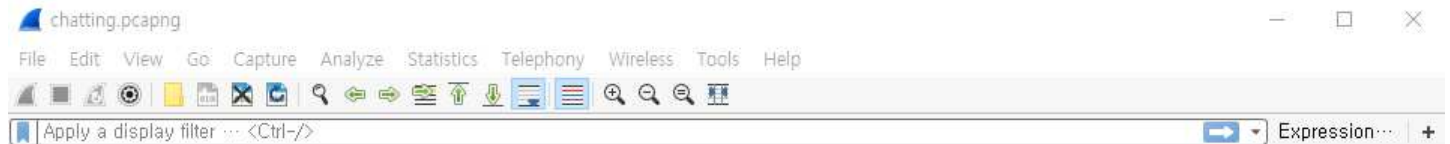
실행 결과

아래 화면은 자기소개서 hwp 파일을 이 프로그램을 통해 전송하는 화면 입니다.



패킷 캡처 화면


Chatting Transfer 패킷 - 이전 과제에서 제출했던 Chatting 레이어와 동일하게 작동합니다.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Clevo_58:df:41	AsrockIn_71:11:43	0x2008	60	Ethernet II
2	3.718100	AsrockIn_71:11:43	Clevo_58:df:41	0xd007	14	Ethernet II
3	3.718124	AsrockIn_71:11:43	Clevo_58:df:41	0xd007	14	Ethernet II

File Transfer 패킷

아래의 패킷 중 1번 패킷이 파일 이름 패킷에 해당합니다. 나머지 패킷들은 단편화된 파일 바이너리 정보에 해당하는 패킷들입니다. 전송한 파일의 크기는 18,944 바이트입니다. 파일 앱 패킷의 단편화 기준이 1448 바이트이므로, 파일 앱의 헤더 12바이트와 이더넷 레이어 헤더 14바이트를 더해, 한 개의 패킷이 1474 바이트 크기를 갖는 것을 알 수 있습니다.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	74	Ethernet II
2	0.000741	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
3	0.101356	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
4	0.202401	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
5	0.304063	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
6	0.404557	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
7	0.506104	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
8	0.606734	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
9	0.707047	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
10	0.807168	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
11	0.908698	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
12	1.009071	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
13	1.109768	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
14	1.210761	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	1474	Ethernet II
15	1.311207	Clevo_58:df:41	AsrockIn_71:11:43	0x2a08	146	Ethernet II

참고 웹페이지, 자료 URL

※ 이전 과제에 참고했던 페이지들은 아래 리스트에 포함하지 않았습니다.

1 - File transfer modal 창

<https://www.mkyong.com/swing/java-swing-jfilechooser-example/>

2 - JProgressive bar 관련

<https://www.geeksforgeeks.org/java-swing-jprogressbar/>

3 - byte[] File 형태로 쓰기

<https://stackoverflow.com/questions/2885173/how-do-i-create-a-file-and-write-to-it-in-java>

4 - JOptionPane 관련

<https://stackoverflow.com/questions/8689122/joptionpane-yes-no-options-confirm-dialog-box-issue>