PKU-CompNet (H) Fall'22
Lab Assignment (Premium): Protocol Stack
Report

Jiaheng Li (2000013054)

September 26, 2022

1 Lab 1: Link-layer

1.1 Writing Task 1 (WT1)

The third frame in the results is:

```
No.   Time       Source    Destination       Protocol   Length   Info
12    1.068164   0.0.0.0   255.255.255.255   DHCP       342      DHCP Discover - Transaction ID 0x13699715
```

1. There are 827 frames in the filtered result. (`Dislayed: 827`)
2. The destination address of the Ethernet frame is `ff:ff:ff:ff:ff:ff`, which is the broadcast address.
3. The 71st byte is `0x15`.

1.2 Programming Task 1 (PT1)

The required functions are implemented in `src/lib/device.c`, which maintains a linear structure of `Device` objects declared in `src/include/internal/device.h`, containing basic information of the devices and their runtime handlers for management.

1.3 Programming Task 2 (PT2)

The required functions are implemented in `src/lib/packetio.c`.

For a `sendFrame()` call, it simply get information from the `devices` structure, allocate space to put each element of an Ethernet II frame in order, and then call `pcap_sendpacket()` to send it out.

For `setFrameReceiveCallback()`, it simply modifies the global pointer to the callback.

To receive frames, a thread is created in each successful `addDevice()`, continously running `pcap_loop()` to call the receiving callback for that device.

The global data about devices and the callback are protected through atomic variables or mutexes.

1.4 Checkpoint 1 (CP1)

The typescript record is at `checkpoints/CP1/{typescript, timing.log}`.

For convenient testing, a `console` tool is implemented in `src/tools/console.c` and will be built to `build/tools/console`, which enables us to call the functions above interactively.

In the script, several calls to `addDevice()` and `findDevice()` in a row shows that they can correctly add (only) the valid Ethernet devices on the host and correctly find the added devices.

1.5 Checkpoint 2 (CP2)

The typescript record is at `checkpoints/CP2/{typescript, timing.log}`.

A testing tools is implemented in `src/tools/eth_test.c`. It receives several devices and their destination address (the MAC address on their other sides) as arguments. It sends 10000 random raw Ethernet frames through each device (in a new thread for each device), and receives the frames from the other side at the same time. (To prevent losses, it will delay for a short time after each batch of frames.) In the end, it will print a checksum of frames it sends, and another of frames it receives.

In the script, the `eth_test` tool above is simultaneously run on each of the 5 hosts in the example network provided in `vnetUtils`, using all of the 8 devices and 4 links among them. After the run, we can see that the sending and receiving checksum of each link matches, which shows the (basic) correctness of our sending and receiving implementation.