24-1 IoT실습 Lab2

22100579 이진주

실습#1

세팅완료!

실습#2

명령어의 의미

```
leeejjju@thxuberrypi:~ $ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
fault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN gr
oup default qlen 1000
    link/ether e4:5f:01:fd:79:21 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP group default qlen 1000
    link/ether e4:5f:01:fd:79:23 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.24/24 brd 192.168.0.255 scope global dynamic noprefixrout
e wlan0
       valid_lft 6902sec preferred_lft 6902sec
    inet6 fe80::fa04:e8c1:df2a:cd3c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

현재 시스템에서 사용 가능한 모든 네트워크 인터페이스와 그들의 IP주소를 확인한다.

```
leeejjju@thxuberrypi:~ $ ip addr show dev eth0
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN gr
oup default qlen 1000
    link/ether e4:5f:01:fd:79:21 brd ff:ff:ff:ff:ff:ff
```

위와 같으나, eth0라는 인터페이스의 정보만을 필터링하여 보여준다.

```
leeejjju@thxuberrypi:~ $ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEF
AULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN mo
de DEFAULT group default qlen 1000
    link/ether e4:5f:01:fd:79:21 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP mode DORMANT group default qlen 1000
    link/ether e4:5f:01:fd:79:23 brd ff:ff:ff:ff:ff:ff
```

시스템에서 사용 가능한 네트워크 인터페이스의 상태, 링크 상태, MAC 주소 등을 확인한다.

```
leeejjju@thxuberrypi:~ $ sudo ip link set eth0 up
```

Eth0 인터페이스를 활성화한다. (데이터 송수신 가능)

```
leeejjju@thxuberrypi:~ $ sudo ip link set eth0 down
```

Eth0 인터페이스를 비활성화한다. (데이터 송수신 불가)

```
leeejjju@thxuberrypi:~ $ sudo ip link set eth0 promisc on
```

Eth0 인터페이스를 Promiscuous 모드로 설정한다. (네트워크 패킷을 모두 수신)

```
leeejjju@thxuberrypi:~/IoTLab $ ip route add default via 192.168.1.1 dev eth
0
```

Eth0 인터페이스를 통해 전송될 모든 트래픽의 게이트웨이의 ip를 192.168.1.1로 설정한다.

```
leeejjju@thxuberrypi:~/IoTLab $ sudo ip route add default via 192.168.1.1 de
v eth0
```

목적지 192.168.1.0/24 네트워크에 대한 경로를 추가한다. 이 때 전송될 게이트웨이는 192.168.1.1 이다.

```
leeejjju@thxuberrypi:~/IoTLab $ sudo ip route add 192.168.1.0/24 via 192.168
.1.1
```

이전에 추가한 경로를 삭제한다. 더 이상 해당 네트워크로의 트래픽이 해당 게이트웨이를 통해 전송되지 않도록 한다.

```
leeejjju@thxuberrypi:~/IoTLab $ ip neigh
192.168.0.25 dev wlan0 lladdr 60:f6:77:66:26:96 DELAY
192.168.0.1 dev wlan0 lladdr 58:86:94:55:40:63 REACHABLE
```

현재 네트워크 인터페이스의 이웃 정보, ARP 캐시에 저장된 정보를 확인한다. (ip와 MAC 매핑)

```
leeejjju@thxuberrypi:~/IoTLab $ ip neigh show dev eth0
```

특정 네트워크 인터페이스(eth0)에 대한 이웃 정보를 보여준다. 없네...

```
leeejjju@thxuberrypi:~/IoTLab $ sudo ip neigh flush all
```

모든 네트워크 인터페이스의 이웃 정보를 삭제한다. (ARP캐시를 비운다)

실습#3

위에서 해봤던 `sudo ip neigh flush all` 명령어를 이용해 ARP 캐시를 비운다.

그리고 다시 패킷을 캡쳐하면 다음과 같은 ARP 패킷들이 캡쳐된다.

arp

No.	Time	tcp stream	Source	Destination	Protocol	Length	Info
15	20.796589147		RaspberryPiT_fd:79:…	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.24
16	20.800703569		EFMNetworks_55:40:63	RaspberryPiT_fd:79:…	ARP	42	192.168.0.1 is at 58:86:94:55:40:63
19	25.906514696		EFMNetworks_55:40:63	RaspberryPiT_fd:79:…	ARP	42	Who has 192.168.0.24? Tell 192.168.0.1
20	25.906550436		RaspberryPiT_fd:79:…	EFMNetworks_55:40:63	ARP	42	192.168.0.24 is at e4:5f:01:fd:79:23
25	70.962679740		EFMNetworks_55:40:63	RaspberryPiT_fd:79:…	ARP	42	Who has 192.168.0.24? Tell 192.168.0.1
26	70.962727591		RaspberryPiT_fd:79:…	EFMNetworks_55:40:63	ARP	42	192.168.0.24 is at e4:5f:01:fd:79:23
27	70.976555045		RaspberryPiT_fd:79:…	EFMNetworks_55:40:63	ARP	42	Who has 192.168.0.1? Tell 192.168.0.24
28	70.977373402		EFMNetworks_55:40:63	RaspberryPiT_fd:79:…	ARP	42	192.168.0.1 is at 58:86:94:55:40:63

여기서 내 컴퓨터에서 request한 패킷과 그에 대한 reply를 찾기 위해, 내 컴퓨터의 IP주소와 MAC 주소를 확인해보자.

```
leeejjju@thxuberrypi:~/IoTLab $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e4:5f:01:fd:79:21  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 121  bytes 9883 (9.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 121  bytes 9883 (9.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.24  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::fa04:e8c1:df2a:cd3c  prefixlen 64  scopeid 0x20<link>
        ether e4:5f:01:fd:79:23  txqueuelen 1000  (Ethernet)
        RX packets 2704  bytes 1402796 (1.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1133  bytes 176296 (172.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Wlan0의 정보를 보면, IP는 `192.168.0.24`, mac주소는 `ether e4:5f:01:fd:79:23` 이다.

이제 다시 캡쳐된 패킷을 보자.

Source	Destination	Protocol	Length	Info
RaspberryPiT_fd:79:…	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.24
EFMNetworks_55:40:63	RaspberryPiT_fd:79:…	ARP	42	192.168.0.1 is at 58:86:94:55:40:63

Info의 정보를 보면 broadcast를 통해 내 컴퓨터 IP(192.168.0.24)에게 192.168.0.1의 정체를 알려달라는 request를 뿌리고, 이어서 그 응답이 오는 부분이 위와 같이 나온다.

Request 패킷의 Ethernet frame은 다음과 같다.

```
Frame 15: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlan0,
    Section number: 1
  > Interface id: 0 (wlan0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Mar 22, 2024 17:17:25.728674013 대한민국 표준시
    UTC Arrival Time: Mar 22, 2024 08:17:25.728674013 UTC
    Epoch Arrival Time: 1711095445.728674013
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 0.415915208 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 20.796589147 seconds]
    Frame Number: 15
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
```

인터페이스는 wlan0, Encapsulation type은 대부분의 LAN 통신이 그러하듯 이더넷이다.

시간에 대한 정보가 다양한 형태로 표시되고, Frame number와 길이가 함께 표시된다. Frame number는 Wireshark의 캡처된 패킷 넘버와 동일하다. 15번째로 캡처된 패킷이라는 의미 같다.

mark되거나 ignore되지 않은 것을 확인할 수 있다.

Coloring Rule은 와이어샤크가 컬러링에 사용한 규칙 이름과 필터라고 한다.

```
Ethernet II, Src: RaspberryPiT_fd:79:23 (e4:5f:01:fd:79:23), Dst: Broadcast (ff:ff:ff:f
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: RaspberryPiT_fd:79:23 (e4:5f:01:fd:79:23)
    Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: RaspberryPiT_fd:79:23 (e4:5f:01:fd:79:23)
    Sender IP address: 192.168.0.24
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.1
```

이 구간에는 패킷의 목적지 MAC 주소와 출발지 MAC 주소가 표시된다. 이 request 패킷의 목적지는 broadcast를 의미하는 ff:ff:ff:ff:ff:ff이고, 출발지는 위에서 확인한 내 rpi의 MAC 주소이다. 패킷의 타입 역시 ARP이며, 내용물에는 sender와 target의 IP, MAC 주소가 담겨 있다. 이 때, target의 MAC 주소는 요구하는 것이고 아직 모르는 상태이므로 0으로 채워져 있고, 이미 알고 있는 IP 주소만 입력되어 있다. 이 부분의 sender 주소들은 target의 IP를 가진 사람이 확인한 뒤 자신의 MAC 정보를 채워 보낼 목적지가 될 것이다.

이 request에 대한 response는 다음과 같다.

```
Frame 16: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlan0,
    Section number: 1
  > Interface id: 0 (wlan0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Mar 22, 2024 17:17:25.732788435 대한민국 표준시
    UTC Arrival Time: Mar 22, 2024 08:17:25.732788435 UTC
    Epoch Arrival Time: 1711095445.732788435
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 0.004114422 seconds]
    [Time delta from previous displayed frame: 0.004114422 seconds]
    [Time since reference or first frame: 20.800703569 seconds]
    Frame Number: 16
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
```

위와 같이 와이어샤크 차원에서의 설명이 기록되어 있다. 패킷에 대한 내용은 다음과 같다.

```
Ethernet II, Src: EFMNetworks_55:40:63 (58:86:94:55:40:63), Dst: RaspberryPiT_fd:79:23
  > Destination: RaspberryPiT_fd:79:23 (e4:5f:01:fd:79:23)
  > Source: EFMNetworks_55:40:63 (58:86:94:55:40:63)
    Type: ARP (0x0806)
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: EFMNetworks_55:40:63 (58:86:94:55:40:63)
    Sender IP address: 192.168.0.1
    Target MAC address: RaspberryPiT_fd:79:23 (e4:5f:01:fd:79:23)
    Target IP address: 192.168.0.24
```

목적지는 나의 rpi MAC 주소이고, 출발지는 알 수 없는 MAC 주소이다. 아마 요청했던 IP를 가진 친구일 것이다. 타입은 물론 ARP이다.

내용에는 아까와 마찬가지로 sender, target의 IP, MAC 주소가 들어있다. Sender는 broadcast로 전달된 request를 보고 "오잉 내 IP네? 내거네?" 한 사람이다. 자신의 IP, MAC 정보를 sender 부분에 채우고, target에는 자신의 정보를 원하던(requset에서 sender였던) 그 친구의 정보를 채워 보낸 것이다. 실제로 요청했던 내 rpi의 MAC, IP가 잘 채워져 있다.

즉, 내 rpi가 192.168.0.1의 MAC을 알아내기 위해 사방팔방에 broadcast로 "이 IP 응답바람!! 난 뭐시깽이임!!"하고 개인정보와 상대 IP를 뿌리고 다녔고, 이 패킷을 받은 그 IP의 주인공이 "어레레? 이거 난데?" 하고 정보를 채워서 답장을 준 것이다. 이렇게 ARP 과정을 확인해볼 수 있었다!