

# **Wireshark를 이용한 DHCP와 IPv6 활용 현황 분석**

제출자: 201421093 김민형

## 목차

1. 관찰 환경
2. DHCP 활용 현황 분석
3. IPv6 활용 현황 분석
4. 새로 알게 된 지식
5. 관찰 후 소감

## 1. 관찰 환경

사용 가능한 와이파이가 1 개인 기숙사에서는 DHCP의 관찰이 어렵다고 판단하여 다수의 와이파이를 제공하는 기숙사 식당에서 Ajou Univ 와이파이로 분석을 시작하여 분석 도중에 iptime 와이파이로 네트워크 환경을 바꾸면서 DHCP 패킷을 캡처하였다. 관찰 시간은 약 12 초로 짧은 시간이지만 분석에 필요한 패킷을 충분히 캡처할 수 있었다.

IPv6 분석을 하기 위해 다양한 종류의 프로토콜을 관찰하고자 두 번째 캡처는 약 100 초 정도 기숙사 와이파이를 이용하여 캡처하였다.

## 2. DHCP 활용 현황 분석

필자는 DHCP 프로토콜의 패킷을 한눈에 보기 위해 bootp 필터를 사용했다. Bootp란 서버에서 클라이언트로 네트워크 정보를 유동적으로 할당할 수 있는 프로토콜이다. Bootp 서버는 클라이언트로부터 request를 받게 되면 서버 내에 미리 설정되어 있는 목록에서 MAC 주소에 매핑되어 있는 네트워크 정보를 클라이언트에게 할당한다. 즉, bootp 서버는 사용자가 서버 내의 목록에 클라이언트의 MAC 정보를 미리 설정해놓고 클라이언트로부터 request를 받는다면 그 목록에 등록된 대로 매핑된 네트워크 정보를 할당해주는 것이다. DHCP는 이러한 bootp의 동작에 Dynamic한 기능이 확장된 통신 프로토콜이다

No.	Time	Source	Destination	Protocol	Length	Info
3974	4.739527	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x3c06f1f4
3977	4.751521	192.168.0.1	192.168.0.4	DHCP	598	DHCP Offer - Transaction ID 0x3c06f1f4
3984	5.757156	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x3c06f1f4
3985	5.770068	192.168.0.1	192.168.0.4	DHCP	598	DHCP ACK - Transaction ID 0x3c06f1f4

< 사진 1. Bootp 필터를 적용하여 관찰한 DHCP 패킷 >

필자가 수업에서 배운 내용대로 예상한 결과가 나왔다. DHCP는 총 4개의 메시지를 이용해 동작하는데 이는 Discover, Offer, Request 그리고 ACK이다. 여기서 Discover와 Offer는 필수 메시지는 아니다.

- ▶ Ethernet II, Src: Apple\_93:65:cf (8c:85:90:93:65:cf), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
- ▼ Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
  - 0100 .... = Version: 4
  - .... 0101 = Header Length: 20 bytes (5)
  - ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  - Total Length: 328
  - Identification: 0xaf84 (44932)
  - ▶ Flags: 0x00
  - Fragment offset: 0
  - Time to live: 255
  - Protocol: UDP (17)
  - Header checksum: 0xb21 [validation disabled]
  - [Header checksum status: Unverified]
  - Source: 0.0.0.0
  - Destination: 255.255.255.255
  - [Source GeoIP: Unknown]
  - [Destination GeoIP: Unknown]

Created by Paint X

< 사진 2. DHCP Discover 패킷 구조 및 내용 >

<사진 2>에서 주의 깊게 봐야 할 부분은 Destination의 맥 주소(ff:ff:ff:ff:ff:ff)와 IP 주소(255.255.255.255)이다. Discover 메시지는 클라이언트가 DHCP 서버를 찾기 위한 메시지다. 동일 subnet 상의 단말에게 브로드캐스팅을 하여 '혹시 DHCP 서버가 있다면 답을 주십시오'라는 클라이언트의 요청이다. 브로드캐스트란 동일 subnet 상에 있는 모든 단말에게 데이터를 전달하는 방식이다. 교수님은 수업 때 매우 혼잡하고 사람이 많은 놀이공원에서 스태프를 찾을 때 소리치는 게 효과적이라는 비유를 하시기도 했다.

### ▼ Bootstrap Protocol (Offer)

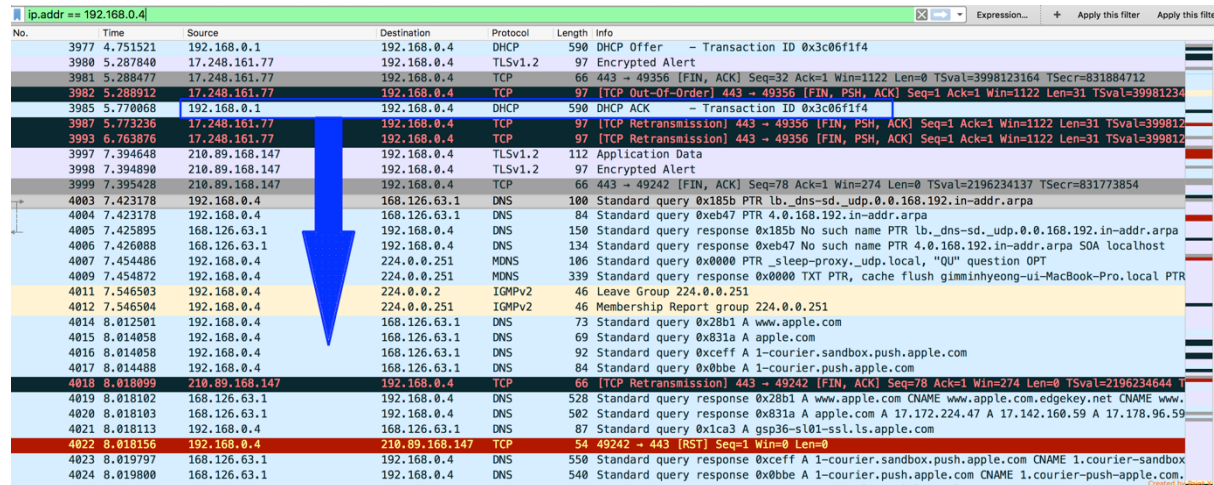
[illegible]

### < 사진 3. DHCP Offer 패킷 구조 및 내용 >

<사진 3>에서 주의 깊게 봐야 할 부분은 Your IP address 와 IP Address Lease Time 이다. Your IP address 항목은 단말에 할당할 IP 주소이고 IP Address Lease Time 은 단말이 IP 주소를 임대할 수 있는 시간이 설정되어있다. DHCP Offer 는 서버가 앞서 받은 요청에 대해 '여기 서버 있습니다' 라고 대답하는 메시지이다. 그러므로 현재 필자에게 192.168.0.4 에 해당 하는 IP 주소를 2 시간 동안 임대받을 수 있는 오퍼가 들어온 것이다.

그다음 보이는 DHCP Request 는 Offer 로부터 받은 네트워크 정보들을 사용하겠다고 보내는 메시지다. 즉, DHCP 서버가 존재한다는 것을 인지했고, 서버로부터 할당받을 네트워크 정보를 받았으니 클라이언트는 DHCP 서버로 오퍼 받은 네트워크 정보를 사용하겠다고 다시 요청하는 것이다.

ACK 메시지는 DHCP 의 마지막 절차로, Request 에 의해서 DHCP 서버가 Offer 응답에서 보낸 네트워크 정보대로 단말에게 네트워크 정보를 할당해주는 메시지다.



No.	Time	Source	Destination	Protocol	Length	Info
3977	4.751521	192.168.0.1	192.168.0.4	DHCP	590	DHCP Offer - Transaction ID 0x3c86f1f4
3980	5.287840	17.248.161.77	192.168.0.4	TLSv1.2	97	Encrypted Alert
3981	5.288477	17.248.161.77	192.168.0.4	TCP	66	443 → 49356 [FIN, ACK] Seq=32 Ack=1 Win=1122 Len=0 TSval=3998123164 TSecr=831884712
3982	5.288912	17.248.161.77	192.168.0.4	TCP	97	[TCP Out-Of-Order] 443 → 49356 [FIN, PSH, ACK] Seq=1 Ack=1 Win=1122 Len=31 TSval=39981234
3985	5.770068	192.168.0.1	192.168.0.4	DHCP	590	DHCP ACK - Transaction ID 0x3c86f1f4
3987	5.773236	17.248.161.77	192.168.0.4	TCP	97	[TCP Retransmission] 443 → 49356 [FIN, PSH, ACK] Seq=1 Ack=1 Win=1122 Len=31 TSval=399812
3993	6.763876	17.248.161.77	192.168.0.4	TCP	97	[TCP Retransmission] 443 → 49356 [FIN, PSH, ACK] Seq=1 Ack=1 Win=1122 Len=31 TSval=399812
3997	7.394648	210.89.168.147	192.168.0.4	TLSv1.2	112	Application Data
3998	7.394890	210.89.168.147	192.168.0.4	TLSv1.2	97	Encrypted Alert
3999	7.395428	210.89.168.147	192.168.0.4	TCP	66	443 → 49242 [FIN, ACK] Seq=78 Ack=1 Win=274 Len=0 TSval=2196234137 TSecr=831773854
4003	7.423178	192.168.0.4	168.126.63.1	DNS	100	Standard query 0x185b PTR lb_dns-sd_udp.0.0.168.192.in-addr.arpa
4004	7.423178	192.168.0.4	168.126.63.1	DNS	84	Standard query response 0xeb47 PTR 4.0.168.192.in-addr.arpa
4005	7.425895	168.126.63.1	192.168.0.4	DNS	150	Standard query response 0x185b No such name PTR lb_dns-sd_udp.0.0.168.192.in-addr.arpa
4006	7.426888	168.126.63.1	192.168.0.4	DNS	134	Standard query response 0xeb47 No such name PTR 4.0.168.192.in-addr.arpa SOA localhost
4007	7.454486	192.168.0.4	224.0.0.251	MDNS	106	Standard query 0x0000 PTR _sleep-proxy_udp.local, "QU" question OPT
4009	7.454872	192.168.0.4	224.0.0.251	MDNS	339	Standard query response 0x0000 TXT PTR, cache flush gimminhyeong-ui-MacBook-Pro.local PTR
4011	7.546503	192.168.0.4	224.0.0.2	IGMPv2	46	Leave Group 224.0.0.251
4012	7.546504	192.168.0.4	224.0.0.251	IGMPv2	46	Membership Report group 224.0.0.251
4014	8.012501	192.168.0.4	168.126.63.1	DNS	73	Standard query 0x28b1 A www.apple.com
4015	8.014058	192.168.0.4	168.126.63.1	DNS	69	Standard query 0x831a A apple.com
4016	8.014058	192.168.0.4	168.126.63.1	DNS	92	Standard query 0xceff A 1-courier.sandbox.push.apple.com
4017	8.014488	192.168.0.4	168.126.63.1	DNS	84	Standard query 0x0bbe A 1-courier.push.apple.com
4018	8.018099	210.89.168.147	192.168.0.4	TCP	66	[TCP Retransmission] 443 → 49242 [FIN, ACK] Seq=78 Ack=1 Win=274 Len=0 TSval=2196234644 T
4019	8.018102	168.126.63.1	192.168.0.4	DNS	528	Standard query response 0x28b1 A www.apple.com CNAME www.apple.com.edgekey.net CNAME ww
4020	8.018103	168.126.63.1	192.168.0.4	DNS	582	Standard query response 0x831a A apple.com A 17.172.224.47 A 17.142.160.59 A 17.178.96.59
4021	8.018113	192.168.0.4	168.126.63.1	DNS	87	Standard query 0x1ca3 A gsp36-sl01-ssl.ls.apple.com
4022	8.018156	192.168.0.4	210.89.168.147	TCP	54	49242 → 443 [RST] Seq=1 Win=0 Len=0
4023	8.019797	168.126.63.1	192.168.0.4	DNS	550	Standard query response 0xceff A 1-courier.sandbox.push.apple.com CNAME 1.courier-sandbox
4024	8.019800	168.126.63.1	192.168.0.4	DNS	540	Standard query response 0x0bbe A 1-courier.push.apple.com CNAME 1.courier-push-apple.com
4025	8.021820	168.126.63.1	192.168.0.4	DNS	363	Standard query response 0x1ca3 A gsp36-sl01-ssl.ls.apple.com A 17.172.224.47 A 17.142.160.59 A 17.178.96.59

< 사진 4. DHCP ACK 이후의 패킷들 >

<사진 4>를 통해 실제로 DHCP 과정이 모두 끝난 이후의 패킷들을 살펴보면 필자의 IP 주소가 192.168.0.4 로 바뀌었음을 알 수 있다.

### 3. IPv6 활용 현황 분석

현재의 네트워크 세대는 IPv4의 시대에서 IPv6의 시대로 전환하려는 시기이다. 이 시점에서 필자가 캡처한 패킷 중 IPv6를 사용하는 패킷이 전체 트래픽 대비 어느 정도 차지하는지 알아볼 것이다.

Packets: 1206 · Displayed: 55 (4.6%) · Load time: 0:0.65	Profile: Default
--	------------------

< 사진 5. 전체 트래픽 대비 IPv6의 비중 >

<사진5>는 'ipv6' 필터를 적용한 후 화면 우측 하단에 표시되는 필터를 적용한 패킷의 비중에 대한 설명이다. 약 100초간 트래픽을 캡처한 결과 IPv6를 사용하는 패킷은 전체 패킷 1,206개 중 55개, 약 4.6%에 불과했다.

Source	Destination
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::16
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::16
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::fb
fe80::1401:e16d:e222:510d	ff02::1:ff94:7a56
fe80::1401:e16d:e222:510d	ff02::1:ff94:7a56
fe80::1401:e16d:e222:510d	ff02::fb

< 사진 6. IPv6 패킷의 source 와 destination 주소 >

필자가 캡처한 IPv6 패킷의 모든 source 주소는 <사진 6>과 같이 fe80으로 시작한다. 이는 IPv6의 stateless 한 특성 때문인데, 어떠한 상태에서도 자동으로 주소를 설정할 수 있다는 의미이다. 일반적으로는 주소를 할당받기 위해서는 메시지를 기다려서 주소를 구성해야 하지만, 해당 링크의 고유성을 보장된 주소가 예약된 알고리즘에 의해서 자동으로 구성할 수 있는 기능이다. 링크 로컬 주소는 대개 fe80 식별자로 자동 구성된다.

반면, destination의 주소는 몇몇을 제외하고는 ff02 식별자로 구성되어 있다. 그중에서도 ff02::fb의 비중이 높은데, 이는 Multicast DNS(MDNS) 트래픽에 쓰이는 주소이다. ff02::1:ffxx:xxxx의 주소는 Solicited-Node Multicast address에 쓰이는 것으로 IPv6 노드가 유니캐스트 혹은 애니캐스트 주소를 생성할 때 자동으로 생성되는 멀티캐스트 주소이다. 그리고 ff02::1:3 주소는 Link-Local Multicast Name Resolution(LLMNR) 트래픽에 쓰이는 주소이다.

필자는 IPv6를 사용하는 프로토콜에는 어떤 것이 있을지를 필자가 캡처한 패킷을 통해 알아보고 그에 대한 헤더 구조를 살펴볼 생각이다. 필자의 trace 목록에는 MDNS, LLMNR 그리고 ICMPv6가 캡처되었다.

## 1) Multicast Domain Name System (MDNS)

MDNS는 로컬 네트워크 영역에서 설정 없이 호스트 이름을 찾기 위해 사용하는 서비스다. Domain Name System (DNS)와 유사한 패킷 형식을 사용한다. 소형 네트워크 환경에서 별도의 네임서버를 사용하지 않고 호스트를 찾을 수 있다. MDNS 클라이언트는 호스트 이름을 알아야 할 경우 IP 멀티캐스트 쿼리 메시지를 보낸다. 이때 자신의 호스트 이름과 IP 주소 등, 자신을 식별할 수 있는 정보들을 함께 보낸다. 멀티캐스트 채널에 있던 모든 호스트는 이 정보를 수신해서 MDNS 캐시에 업데이트한다. MDNS는 멀티캐스트 UDP 패킷이며, DNS 패킷을 모델로 한다.

Offset (bytes)	0	1
0	ID = 0x0000	
2	Flags	
4	QDCOUNT	
6	ANCOUNT	
8	NSCOUNT	
10	ARCOUNT	

< 사진 7. MDNS 패킷의 헤더 구조 >

ID: 클라이언트가 보낸 요청과 수신한 응답의 ID가 일치하는지 확인하기 위해 사용된다.

Flags: 패킷의 타입이다. "00 00"은 쿼리, "84 00"은 응답이다.

QDCOUNT: 쿼리 섹션의 항목 수를 지정한다.

ANCOUNT: 응답 섹션의 resource record 수를 지정한다.

NSCOUNT: Authority 섹션의 name server resource record 수를 지정한다.

ARCOUNT: 추가적인 record 섹션의 resource record 수를 지정한다.

## 2) Link-Local Multicast Name Resolution (LLMNR)

LLMNR은 DNS 패킷 형태로 로컬 링크 상에서 호스트의 이름을 조회하는 데 이용되는 프로토콜이다. IPv4 호스트에서는 네트워크상 주변에 있는 단말의 이름을 resolve 하기 위해서 NetBIOS를 이용하여 네임쿼리를 하는 메시지를 브로드캐스팅한다. 이때 노드 상에 요청한 쿼리에 해당하는 호스트가 있다면 네임쿼리에 대해 응답하게 된다. 그런데 NetBIOS는 IPv4 환경에서만 작동한다. 그러므로 양 환경에서 모두 이용될 수 있는 LLMNR 프로토콜이 요구된 것이다.

Bit offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	ID															
16	QR	Opcode				C	TC	T	Z	Z	Z	Z	RCODE			
32	QDCOUNT															
48	ANCOUNT															
64	NSCOUNT															
80	ARCOUNT															

< 사진 8. LLMNR 패킷의 헤더 구조 >

ID: 클라이언트가 보낸 요청과 수신한 응답의 ID가 일치하는지 확인하기 위해 사용된다.

QR: 쿼리인지 응답인지를 식별한다.

OPCODE: 메시지내 쿼리의 종류를 식별한다.

(C - Conflict, TC - Truncation, T - Tentative, Z - Reserved for future used)

RCODE: 응답 코드를 나타낸다.

QDCOUNT: 쿼리 섹션의 항목 수를 지정한다.

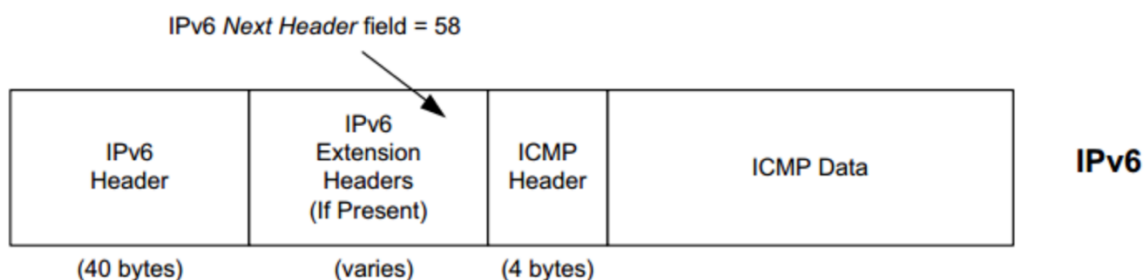
ANCOUNT: 응답 섹션의 resource record 수를 지정한다.

NSCOUNT: Authority 섹션의 name server resource record 수를 지정한다.

ARCOUNT: 추가적인 record 섹션의 resource record 수를 지정한다.

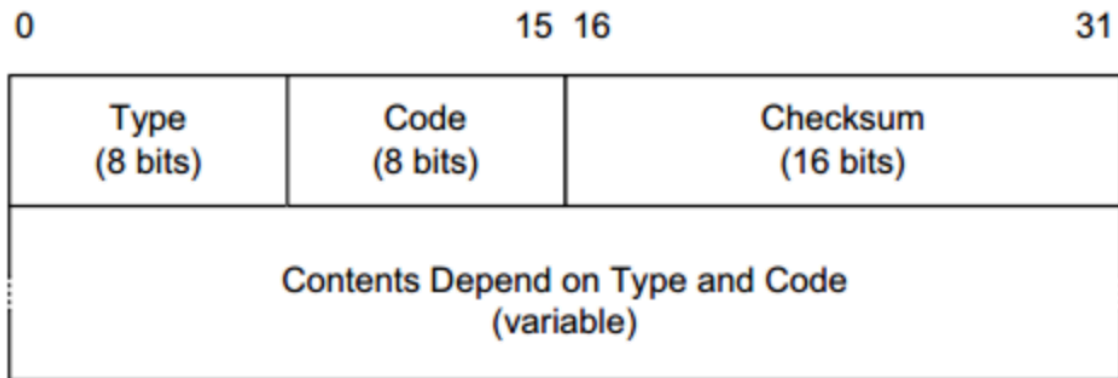
### 3) Internet Control Message Protocol version 6 (ICMPv6)

ICMP는 기존 IPv4 네트워크에서 오류 메시지 전달 등의 주요 제어 메시지에 사용되는 프로토콜이다. ICMPv6는 ICMP의 IPv6 버전이라고 할 수 있다. 하지만 단순히 ICMP의 기능들을 IPv6 버전으로 바꾼 것이 아닌 ICMP에는 없었던 많은 기능이 새로 추가되었다. ICMPv6는 IPv4에서의 ICMP 기능과 Address Resolution Protocol(ARP), Internet Group Management Protocol(IGMP) 기능을 통합한 제어 프로토콜이다. ICMPv6 패킷은 IPv6의 기본 헤더 뒤에 붙어 따라온다.



< 사진 9. IPv6의 기본 헤더 뒤에 붙는 ICMP 헤더 >





< 사진 10. ICMP의 헤더 구조 >

Type: 패킷의 ICMP 유형을 포함한다.

Code: 모든 ICMP 유형에는 다른 코드도 포함될 수 있다.

Checksum: ICMP 헤더의 유효성을 확인한다.

#### 4. 새로 알게 된 지식

짧은 시간을 캡처했지만 IPv6가 현재 대략 어느 정도의 비율로 네트워크 레이어를 차지하고 있는지 알 수 있었다. 과제를 하면서 검색을 통해 알아본 IPv4에서 IPv6로의 전환하는 이슈를 해결하는 여러 방법도 알아보았다. 수업시간에 IPv6를 이용하는 프로토콜에 대해서는 다루지 않았기 때문에 IPv6의 패킷 분석은 모든 게 새로웠다. 그중에서도 이미 배웠던 DNS와 유사한 방식으로 작동하는 MDNS 프로토콜이 흥미로웠다. MDNS는 전통적인 DNS를 확장하여 로컬 영역 네트워크에서 서비스를 검색하기 위해 작동하는 프로토콜이다. MDNS는 클라이언트-서버 모델에서 작동하고 클라이언트는 각 인스턴스에 등록된 일치하는 record를 멀티캐스팅하여 서비스 인스턴스를 나타내는 record에 대한 멀티캐스트 DNS 쿼리를 보내고 서버는 이에 대해 응답한다. 이러한 record에는 요청된 서비스의 인스턴스에 대한 주소 지정 정보 및 메타 데이터가 포함된다. Record는 Time-To-Live(TTL)와 함께 전송되고, 이전 멀티 캐스트에서 받은 record를 캐시 하여 향후 쿼리에 대해 로컬로 응답할 수 있지만, TTL이 만료되기 전에 캐시된 record를 주기적으로 갱신해야 한다.

#### 5. 관찰 후 소감

과제를 시작하면서 과제 1, 2와 같은 방법으로 기숙사에서 캡처를 시도했지만, DHCP와 다양한 IPv6를 사용하는 패킷이 나오지 않았다. 왜 나오지 않는지에 대해 생각해보았고 필자는 개인 네트워크를 사용하는 것 보다 열린 네트워크를 사용하는 게 캡처에 좋을 것이라고 생각하여 자리를 옮겼다. 전반적으로 과제 1, 2보다 어려운 주제였던 것 같다. 우선 주소의 표기가 지금까지 봐왔던 것과 달라서 익숙하지가 않았다. 그리고 IPv6는 현재 크게 사용되고 있지 않기 때문에 인터넷에서 정보를 찾는 것도 전보다 어려웠다. 하지만 수업시간에는 다루지 않았던 다양한 IPv6 기반 프로토콜을 공부하고 분석하면서 IPv6라는 새로운 네트워크 레이어 프로토콜에 익숙해지고 부족한 부분을 채울 수 있었다.