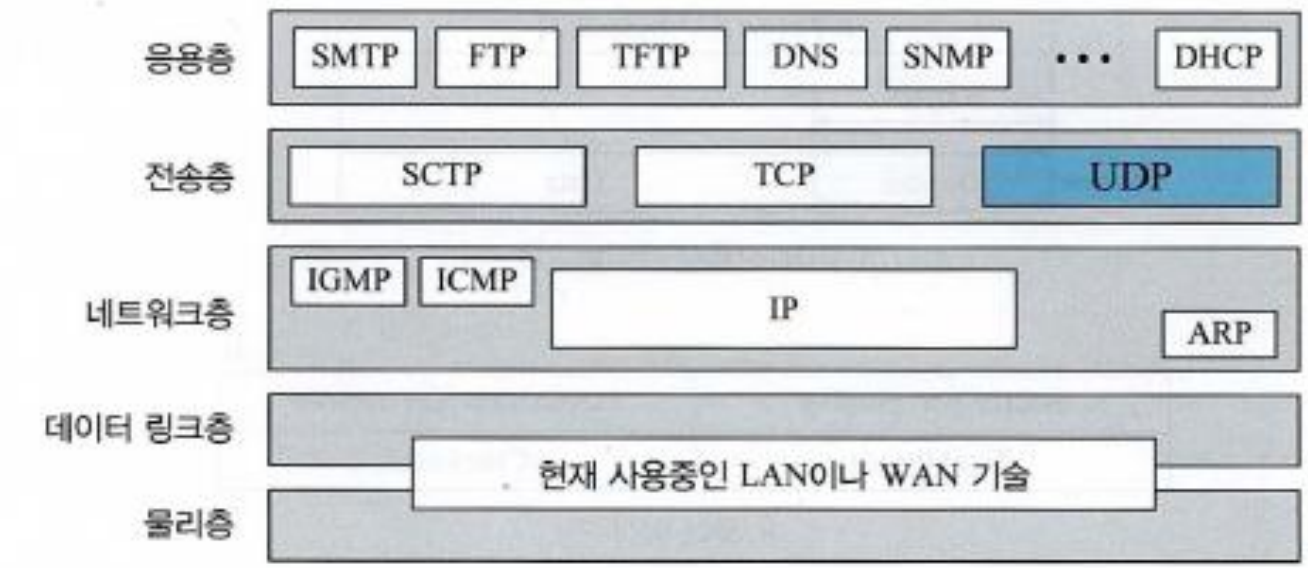
Chapter 11

사용자 데이터그램 프로토콜 (UDP)

1. 개요

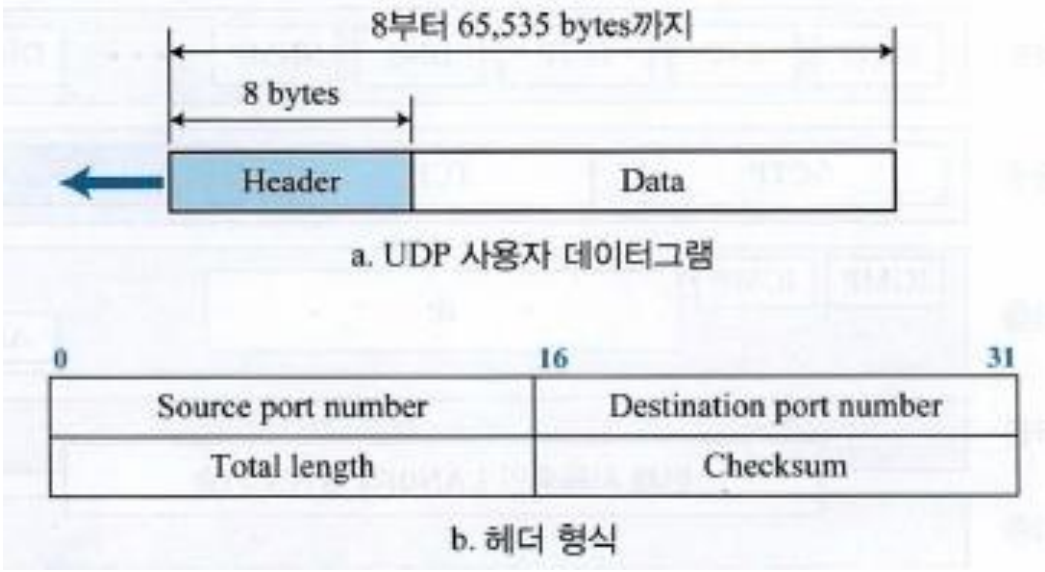
- UDP는 TCP와 같이 응용층과 IP 계층 사이에 위치하며 응용 프로그램과 네트워크 동작 사이의 중개자 역할을 한다.



- 프로세스 대 프로세스 통신을 생성하기 위하여 UDP는 포트 번호를 사용한다.
- UDP는 비연결형, 신뢰성 없는 전송 프로토콜이다.
- 호스트-대-호스트 통신 대신에 프로세스-대-프로세스 통신을 제공하는 것 이외에는 IP 서비스에 추가하는 기능이 아무 것도 없다.

2. 사용자 데이터그램

- 사용자 데이터그램 (user datagram) 이라는 UDP 패킷은 8바이트의 고정 길이 헤더를 가진다.



2. 사용자 데이터그램

- **발신지 포트 번호(source port number)**
  - 이것은 발신지 호스트 상에서 수행되는 프로세스에 의해 사용되는 포트 번호이다.
  - 16비트이므로 포트 번호는 0에서 65,535의 범위 내에 있을 수 있다.
  - 만약 발신지 호스트가 요청을 전송하는 클라이언트라면 대부분의 경우 포트 번호는 프로세스에 의해 요청되고 발신지 호스트의 UDP 소프트웨어가 선택한 임시 포트 번호이다.
  - 만약 발신지 호스트가 응답을 전송하는 서버라면 대부분의 경우 포트 번호는 잘 알려진 포트 번호이다.

- **목적지 포트 번호(destination port number)**
  - 이것은 목적지 호스트 상에서 수행되는 프로세스에 의해 사용되는 포트 번호이다.
  - 이것 역시 16비트이다.
  - 만약 목적지 호스트가 클라이언트의 요청을 받는 서버라면 대부분의 경우 포트 번호는 잘 알려진 포트 번호이다.
  - 만약 목적지 호스트가 서버가 보낸 응답을 받는 클라이언트라면 포트 번호는 임시 포트 번호이다.
  - 이 경우 서버는 수신한 요청 패킷에 있는 임시 포트 번호를 복사하여 사용한다.

2. 사용자 데이터그램

- 길이 (Length)
  - 이 16 비트 필드는 헤더와 데이터를 합한 사용자 데이터그램의 전체 길이를 나타낸다.
  - 16비트는 0에서 65,535 사이의 전체 길이를 지정할 수 있다.
  - UDP 사용자 데이터그램의 길이 필드는 실제로 필요 없다.
  - 사용자 데이터그램은 IP 데이터그램에 의해 캡슐화되기 때문이다.
  - IP 데이터그램에는 전체 길이를 정의하는 필드와 헤더의 길이를 정의하는 필드가 있다.

- 검사합(checksum)
  - 이 필드는 헤더와 데이터를 모두 포함한 사용자 데이터그램 전체에 대해 오류를 탐지하기 위해서 사용된다.

2. 사용자 데이터그램

- **예제. 다음은 16진수 형식으로 UDP 헤더를 덤프한 것이다.**

CB84000D001C001C

  - 발신지 포트 번호는 얼마인가?
  - 목적지 포트 번호는 얼마인가?
  - 사용자 데이터그램의 총 길이는 얼마인가?
  - 데이터의 길이는 얼마인가?
  - 데이터의 전송 방향이 클라이언트에서 서버 쪽으로 가는 것인가 아니면 그 반대 방향인가?
  - 클라이언트 프로세스는 무엇인가?

3. UDP 서비스

프로세스-대-프로세스 통신

- UDP(User Datagram Protocol)는 전송층의 비연결형 프로토콜이다.
- UDP는 포트 번호를 이용하여 PC의 프로세스 간을 연결한다.
- 이것을 [프로세스-대-프로세스 통신]이라고 한다.
- 포트 번호는 2바이트 주소로 전송층의 식별자를 나타낸다.
- 그와 동시에 포트 번호는 PC의 프로세스를 식별하는 번호이기도 하다.
- UDP 포트 번호가 결정되면 서버 측의 애플리케이션도 정해진다.
- 또한 서버 측에서 동작하는 애플리케이션은 [서비스] 또는 [데몬] 등이라고 불린다.
- 포트 번호에서 애플리케이션을 결정하기 위해 인터넷에서 번호를 미리 정해 둘 필요가 있다.
- 그래서 0~1023 번 포트에 대해서는 [잘 알려진(Well-Known) 포트], 1024~49151번의 포트는 [등록된(Registered) 포트]로 미리 애플리케이션에 할당되어 있다.
- 또 49152~65535번 포트를 [동적(Dynamic) 포트]로 통신에 필요할 때마다 할당해 사용하고 있다.
- UDP는 IP 주소와 포트 번호로 구성된 소켓을 이용하여 프로세스 대 프로세스 통신을 제공한다.

3. UDP 서비스

프로세스-대-프로세스 통신

UDP를 이용하는 잘 알려진 포트

포트	프로토콜	설명
7	Echo	메시지 전송장치에게 수신한 메시지를 다시 전송
9	Discard	접속 테스팅을 위한 Null 서비스
11	Users	Active users
13	Daytime	요청하는 호스트에게 날짜와 시간 정보를 보낸다.
17	Quote	오늘의 한마디를 연결된 호스트에게 보낸다.
19	Chargen	자막 생성 서비스 (Character Generation service); 끊임없이 문자 스트림을 보낸다
53	Domain	Domain Name Service (DNS)
67	Bootps	부트스트랩 프로토콜 (Bootstrap Protocol: BOOTP) 서비스; 동적 호스트 설정 프로토콜 (Dynamic Host Configuration Protocol: DHCP) 서비스에 의해서도 사용된다.

3. UDP 서비스

프로세스-대-프로세스 통신

UDP를 이용하는 잘 알려진 포트

포트	프로토콜	설명
68	Bootpc	부트스트랩 (BOOTP) 클라이언트; 동적 호스트 제어 프로토콜 (DHCP) 클라이언트에 의해서도 사용된다.
69	TFTP	Trivial File Transfer Protocol (TFTP)
111	RPC	NFS (네트워크 파일 시스템)에 의해 원격 명령 실행에 사용되는 RPC (Remote Procedure Call)
123	NTP	네트워크 시간 프로토콜 (NTP)
161	SNMP	단순 네트워크 관리 프로토콜 (SNMP)
162	SNMP	SNMP 트랩

3. UDP 서비스

비연결형 서비스

- UDP는 TCP 프로토콜의 경우에 볼 수 있는 연결 설정 (connection establishment)과 연결 종료(connection termination)의 과정이 없다.
- 이것은 UDP에 의해 전송되는 각각의 사용자 데이터그램이 다른 경로를 통하여 전달될 수 있다는 것을 의미한다.
- 비연결형 서비스 결과 중 하나는 UDP를 사용하는 프로세스가 UDP에 하나의 긴 데이터 스트림을 보내고 UDP가 이 스트림을 서로 연관된 사용자 데이터그램으로 나누기를 기대할 수 없다는 것이다.
- 대신에 하나의 시용자 데이터그램에 들어갈 수 있도록 각 요청의 크기가 충분히 작아야 한다.
- 65,507바이트(65,535 - UDP 헤더 8바이트 - IP 헤더 20바이트)보다 작은 메시지를 보내는 프로세스만이 UDP를 사용할 수 있다.

3. UDP 서비스

흐름 제어

- UDP는 흐름 제어 기능이 없으며 따라서 창(윈도우) 메커니즘도 없다.
- 수신 측에서는 들어오는 메시지로 인해 오버플로우가 발생할 수 있다.

오류 제어

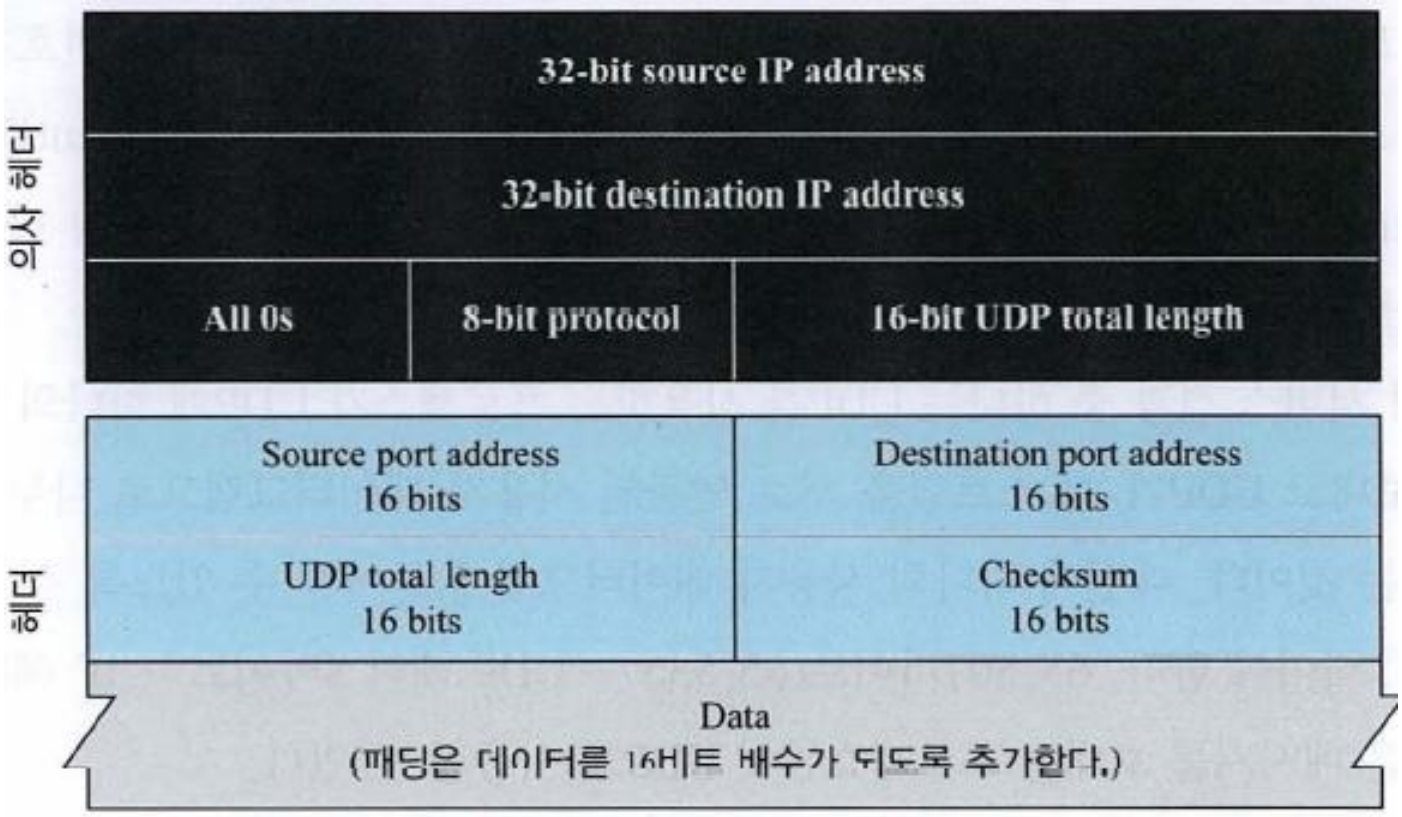
- UDP에는 검사합을 제외한 오류 제어 메커니즘이 없기 때문에 메시지가 손실되거나 중복되었는지 송신자가 알 수 없다.
- 수신자가 검사합을 사용하여 오류를 탐지하면 사용자 데이터그램을 그냥 폐기한다.

3. UDP 서비스

오류 제어

검사합

- UDP 검사합 계산은 IP의 경우와는 다르게 의사 헤더(pseudo header), UDP 헤더, 응용 계층에서 온 데이터를 포함한다.

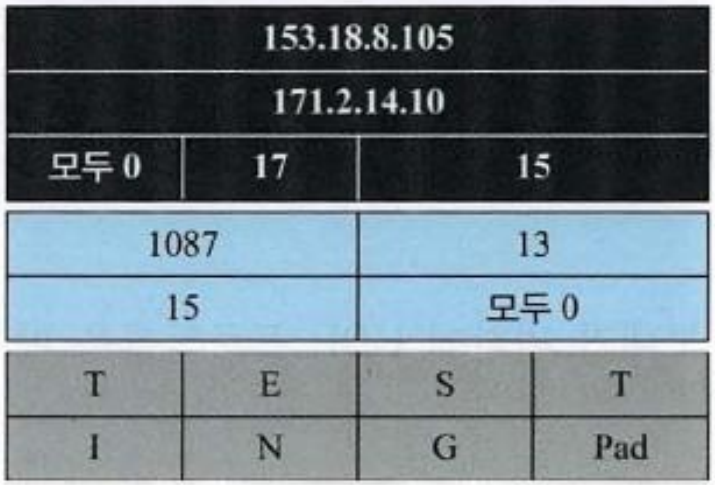


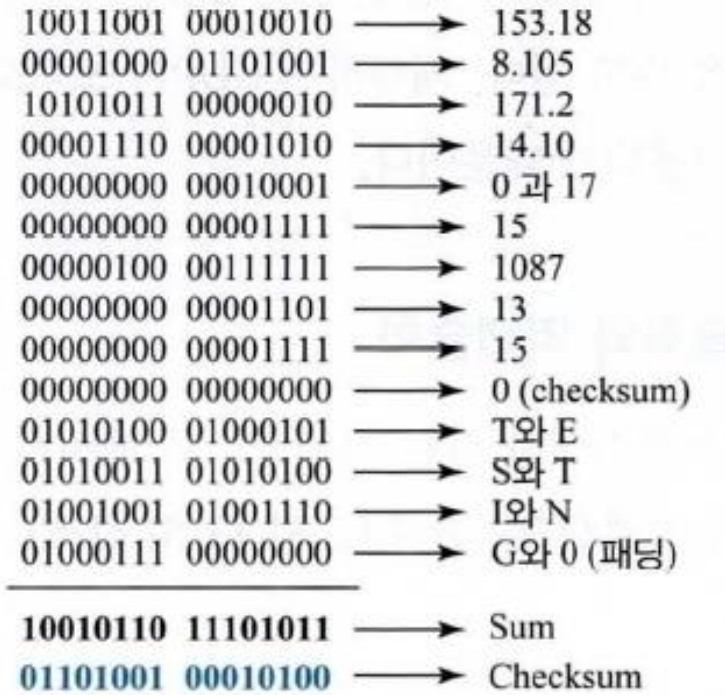
3. UDP 서비스

오류 제어

검사합

- UDP 패킷의 송신자는 검사합을 계산하지 않을 수도 있다.
- 이 경우에 검사합 필드는 0으로 채워진 후에 UDP 패킷이 전송된다.





3. UDP 서비스

오류 제어

검사합

- 다음의 가상 상황에서 검사합으로 어떤 값이 전송되는가?

① 송신자는 검사합을 포함하지 않기로 결정한다.

② 송신자는 검사합을 포함하기로 결정했지만, 검사합의 값이 모두 1 이다.

③ 송신자는 검사합을 포함하기로 결정했지만, 검사합의 값이 모두 0 이다.

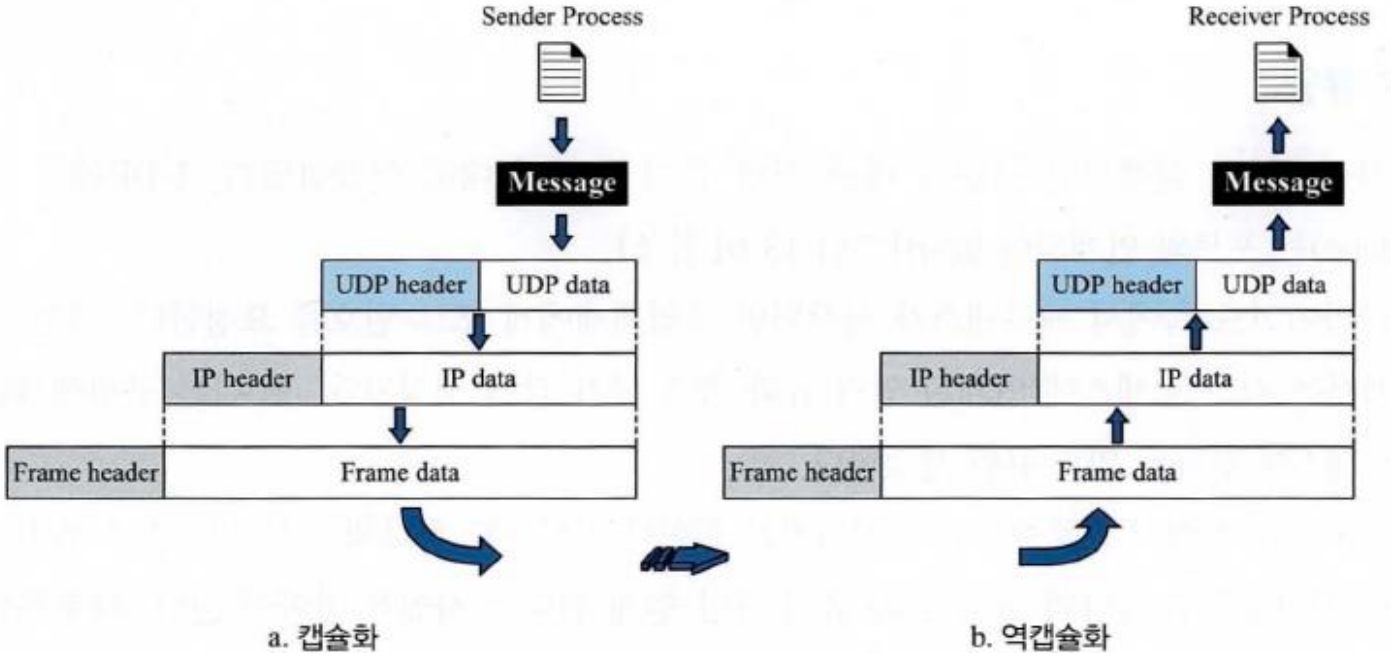
3. UDP 서비스

혼잡 제어

- UDP는 비연결형 프로토콜이기 때문에 혼잡 제어를 제공하지 않는다.
- UDP에서는 패킷은 작고 산발적으로 전송된다고 가정한다.

캡슐화와 역캡슐화

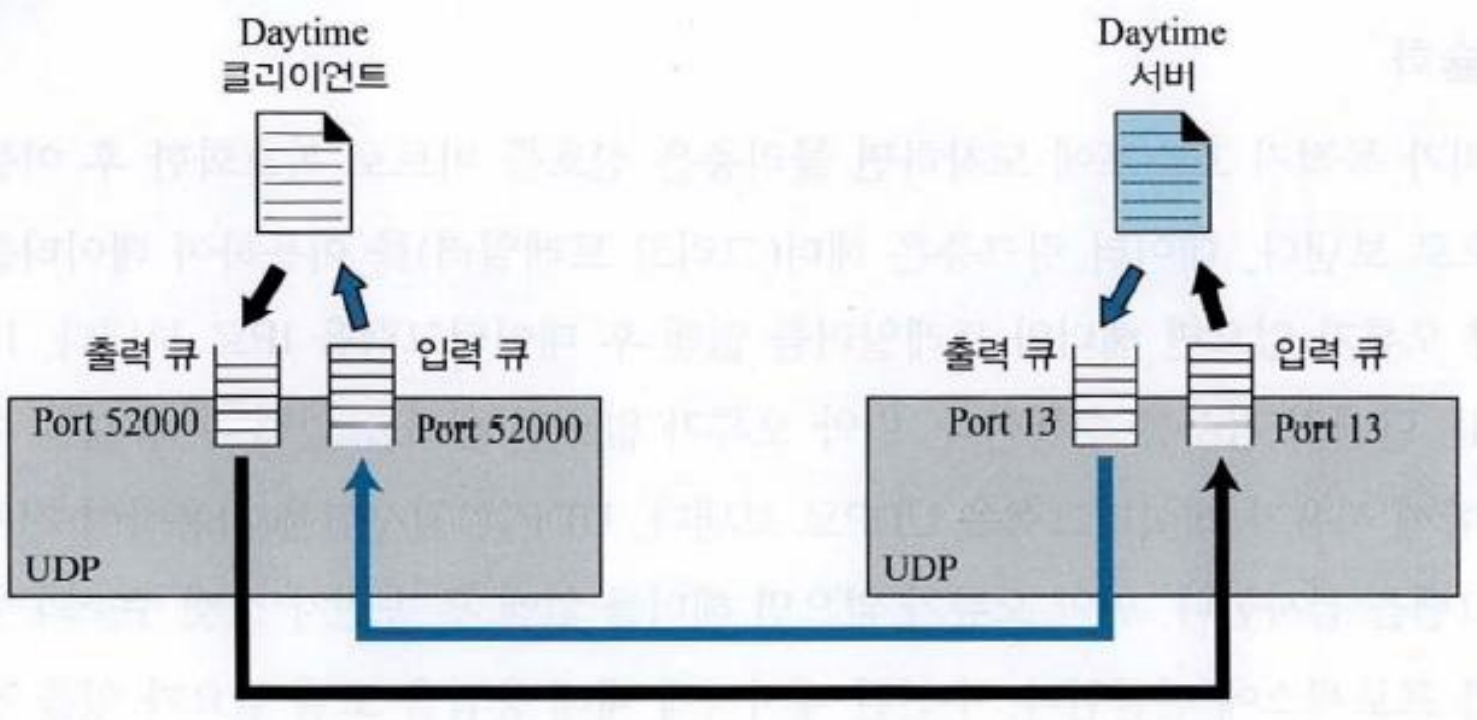
- 한 프로세스에서 다른 프로세스로 메시지를 보내기 위하여 UDP 프로토콜은 메시지를 캡슐화하고 역캡슐화한다.



a. 캡슐화 b. 역캡슐화

3. UDP 서비스

- 큐잉
  - UDP에서 큐(queue)는 포트와 연계되어 있다.



3. UDP 서비스

큐잉

- UDP에서 큐(queue)는 포트와 연계되어 있다.
- 클라이언트 측에서 프로세스가 시작되면 운영체제에게 포트 번호를 요청한다.
- 어떤 구현에서는 각 프로세스에 연계된 입력 큐와 출력 큐가 같이 생성된다.
- 프로세스가 수행되는 동안 큐는 동작을 한다.
- 프로세스가 종료되면 큐는 제거된다.
- UDP는 메시지를 하나씩 가져와 UDP 헤더를 추가한 후 IP에 전달한다.
- 출력 큐에는 오버플로우가 발생할 수 있다.
- 만약 이러한 일이 발생하면 운영체제는 클라이언트 프로세스에게 메시지를 더 보내지 말고 기다리라고 요청한다.
- 메시지가 클라이언트에 도착하면 UDP는 사용자 데이터그램의 목적지 포트 변호에 지정된 포트 번호에 대한 입력 큐가 생성되어 있는지 확인한다.
- 만약 그러한 큐가 있다면 UDP는 수신된 사용자 데이터그램을 큐의 끝에 추가한다.
- 만약 그러한 큐가 없다면, UDP는 사용자 데이터그램을 폐기하고 ICMP 프로토콜에게 “port unreachable" 메시지를 서버로 보낼 것을 요청한다.

3. UDP 서비스

큐잉

- 서버 측에서는 큐를 생성하는 메커니즘이 다르다.
- 가장 간단한 경우 서버는 수행되기 시작할 때 잘 알려진 포트를 시용하여 입력 큐와 출력 큐를 요청한다.
- 서버가 수행되고 있는 동안에는 큐는 열려 있다.
- 서버에 메시지가 도착하면 UDP는 사용자 데이터그램의 목적지 포트 변호 필드에 지정된 포트 번호를 위한 입력 큐가 생성되어 있는지 확인한다.
- 만약 그러한 큐가 있으면 UDP는 수신된 사용자 데이터그램을 큐의 끝에 추가한다.
- 만약 그러한 큐가 없다면 UDP는 사용자 데이터그램을 폐기한 후 ICMP 프로토콜에게 “port unreachable" 메시지를 클라이언트로 보낼 것을 요청한다.
- 입력 큐에는 오버플로우가 발생하면 UDP는 사용자 데이터그램을 폐기한 후 클라이언트에게 “port unreachable" 메시지를 보내도록 ICMP에게 요청한다.
- 출력 큐에는 오버플로우가 발생하면 서버에서 메시지를 더 이상 보내지 말고 기다릴 것을 요청한다.

3. UDP 서비스

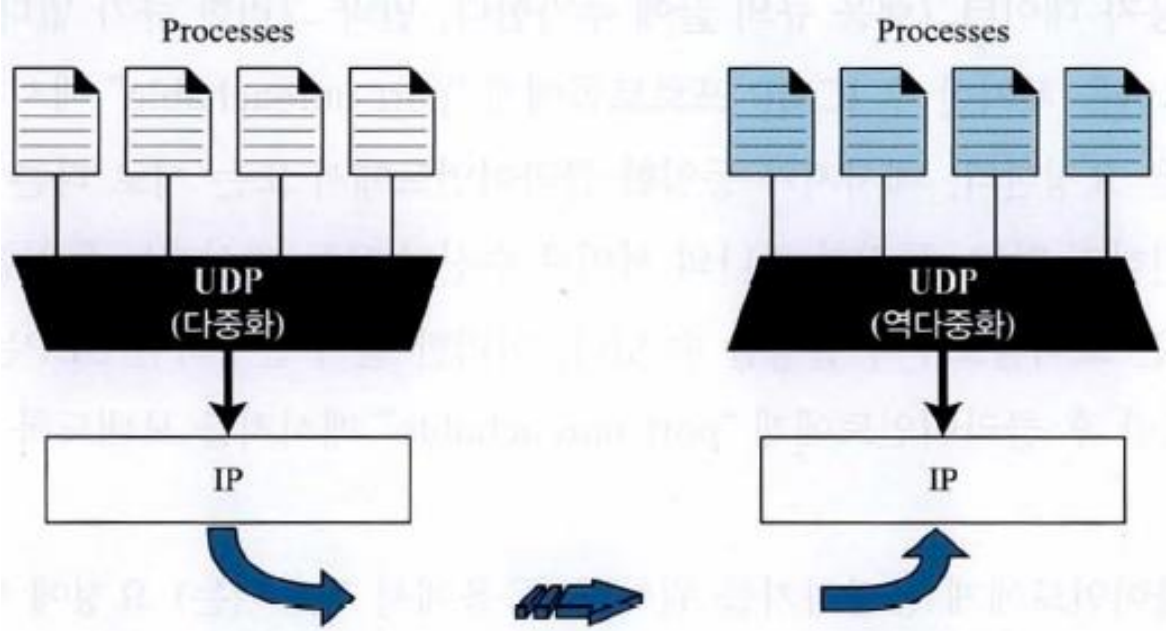
다중화와 역다중화

다중화

- UDP는 프로세스마다 할당된 포트 번호에 의해서 구분되는 서로 다른 프로세스로부터 메시지를 수신한다.
- UDP는 헤더를 추가한 후 IP로 사용자 데이터그램을 보낸다.

역다중화

- UDP는 오류를 점검하고 헤더를 없앤 후 UDP는 포트 번호에 따라 각 메시지를 적절한 프로세스로 보낸다.



4. UDP 응용

UDP의 특징

오류 제어 미제공

- UDP는 오류 제어를 제공하지 않으며, 따라서 신뢰성 없는 서비스를 제공한다.
- 응용층으로 전달되는 메시지들은 동일하지 않은 지연이 생기게 된다.
- 어떤 응용들은 원래부터 동일하지 않은 지연을 인식하지 못하는 경우도 있지만, 어떤 응용들에게는 일정하지 않은 지연은 매우 심각하다.

혼잡 제어 미제공

- UDP는 혼잡 제어를 제공하지 않는다.
- 그렇지만 UDP는 오류가 발생할 수 있는 네트워크에서 추가적인 트래픽을 생성하지는 않는다.
- TCP는 패킷을 여러 번 재전송하여 혼잡을 유발하거나 또는 혼잡 상태를 더 악화시킨다.
- 따라서 UDP에서 오류 제어를 제공하지 않는 것은 혼잡이 큰 문제인 경우에는 장점이다.

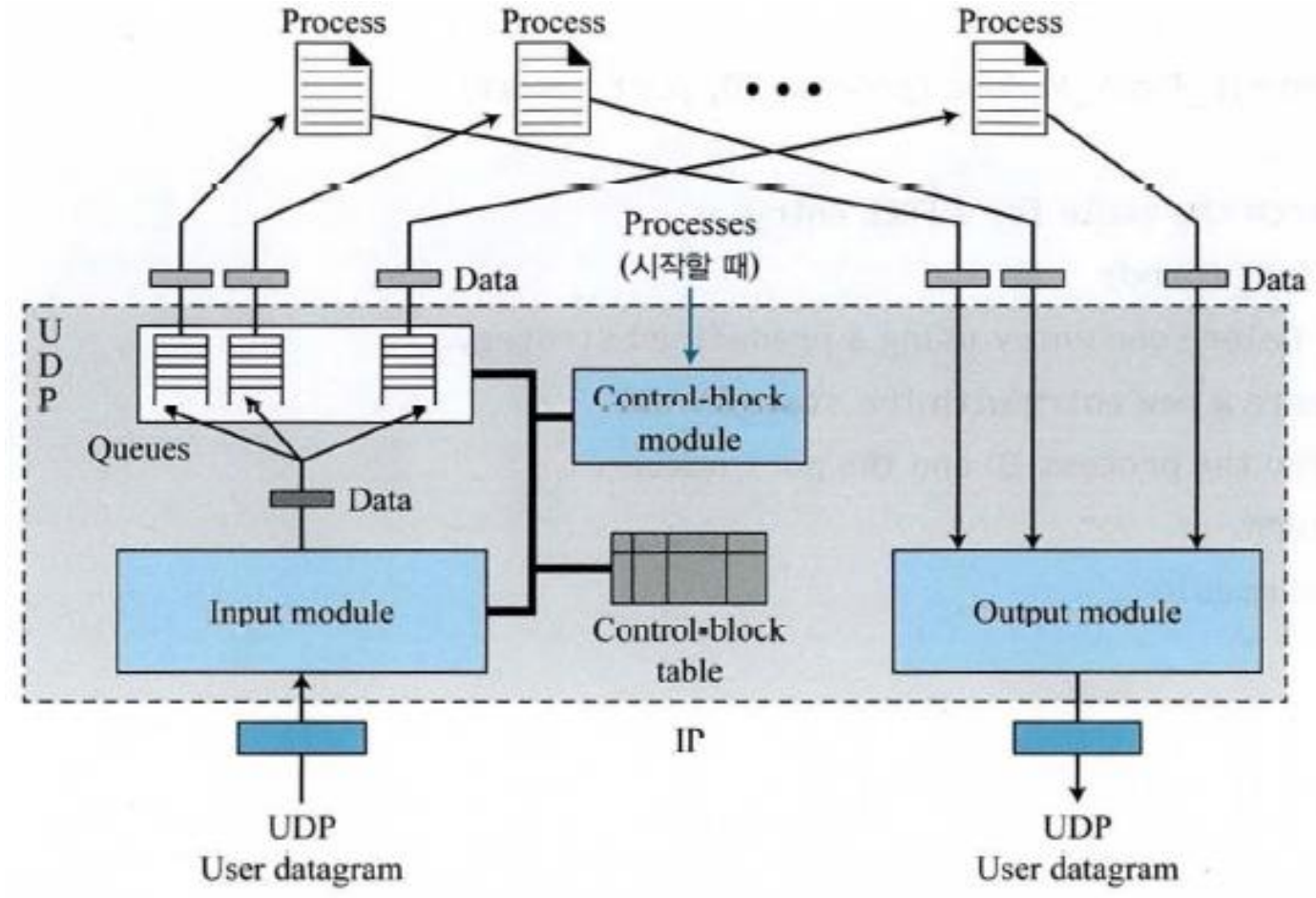
4. UDP 응용

UDP의 대표적인 응용

- UDP는 단순한 요청-응답 통신을 필요로 하고 흐름 제어와 오류 제어에는 관심이 없는 프로세스에 적절하다.
- UDP는 내부에 흐름 제어와 오류 제어 메커니즘을 가지고 있는 프로세스에 적합하다.
- UDP는 멀티캐스팅에 적합한 전송 프로토콜이다.
- UDP는 SNMP(Simple Network Management Protocol)와 같은 관리 프로세스에 사용된다.
- UDP는 라우팅 정보 프로토콜(RIP : Routing Information Protocol)과 같은 경로 갱신 프로토콜에 사용된다.
- UDP는 일반적으로 수신된 메시지의 단편들 간의 지연이 일정한 실시간 응용에서 사용된다.

5. UDP 패키지

- UDP 패키지는 제어 블록 테이블, 입력 큐, 제어 블록 모듈, 입력 모듈, 출력 모듈 이라는 다섯 개의 구성 요소를 갖는다.



5. UDP 패키지

- 제어 블록 테이블
  - UDP는 열린(open) 포트를 추적하기 위한 제어-블록 테이블(control block table)을 가지고 있다.
- 입력 큐
  - UDP는 각 프로세스 당 하나씩 배정되는 입력 큐(input queues)의 집합을 사용한다.
- 제어 블록 모듈
  - 제어 블록 모듈은 제어 블록 테이블의 관리를 담당한다.
  - 이 모듈은 프로세스가 시작될 때 운영체제로부터 포트 번호를 요청한다.

5. UDP 패키지

제어 블록 모듈

```
1  UDP_Control_Block_Module (process ID, port number)
2  {
3      Search the table for a FREE entry.
4      if (not found)
5          Delete one entry using a predefined strategy.
6      Create a new entry with the state IN-USE
7      Enter the process ID and the port number.
8      Return.
9  }
```

5. UDP 패키지

입력 모듈

```
1  UDP_INPUT_Module (user _datagram)
2  {
3      Look for the entry in the control_block table
4      if (found) {
5          Check to see if a queue is allocated
6          If (queue is not allocated)
7              allocate a queue
8          else
9              enqueue the data
10     }
11     else{
12         Ask ICMP to send an "unreachable port" message
13         Discard the user datagram
14     }
15
16     Return
17 }
```

5. UDP 패키지

출력 모듈

1	UDP_OUTPUT_MODULE (Data)
2	{
3	Create a user datagram
4	Send the user datagram
5	Return.
6	}

5. UDP 패키지

예제

- 예제 시작시 제어 블록 테이블

State	Process ID	Port Number	Queue Number
IN-USE	2,345	52,010	34
IN-USE	3,422	52,011	
FREE			
IN-USE	4,652	52,012	38
FREE			

- 처음 동작은 목적지 포트 번호가 52,012인 사용자 데이터그램이 도착하였다.
- 입력 모듈은 이 포트 번호를 탐색하여 찾아낸다.
- 이 포트에는 큐 번호(Queue Number) 38이 배당되어 있으므로 포트가 전에 이미 사용되었음을 알 수 있다.
- 입력 모듈은 데이터를 38번 큐에 보낸다.
- 제어 블록 테이블은 변하지 않는다.

5. UDP 패키지

예제

- 몇 초 후 프로세스가 시작된다.
- 이 프로세스는 운영체제에 포트 번호를 요청하고 52,014를 할당받는다.
- 프로세스는 자신의 ID(4,978)와 포트 번호를 제어 블록 모듈에 보내어 테이블 내에 엔트리를 생성한다.
- 모듈은 첫 번째의 FREE 엔트리를 취하여 수신된 정보를 입력한다.
- 이 목적지를 향한 사용자 데이터그램이 도착하지 않았으므로 이 시점에서 모듈은 큐를 배당하지는 않는다.
- 이후의 제어 블록 테이블

State	Process ID	Port Number	Queue Number
IN-USE	2,345	52,010	34
IN-USE	3,422	52,011	
IN-USE	4,978	52,014	
IN-USE	4,652	52,012	38
FREE			

5. UDP 패키지

예제

- 52,011 포트에 사용자 데이터그램이 도착한다.
- 입력 모듈은 테이블을 검사하여 이 목적지를 위한 큐가 아직 배당되지 않았음을 알게 된다.
- 이것은 이 패킷이 이 목적지로 온 첫 번째 사용자 데이터그램이기 때문이다.
- 모듈은 43 번 큐를 생성한다.
- 이후의 제어 블록 테이블

State	Process ID	Port Number	Queue Number
IN-USE	2,345	52,010	34
IN-USE	3,422	52,011	43
IN-USE	4,978	52,014	
IN-USE	4,652	52,012	38
FREE			

5. UDP 패키지

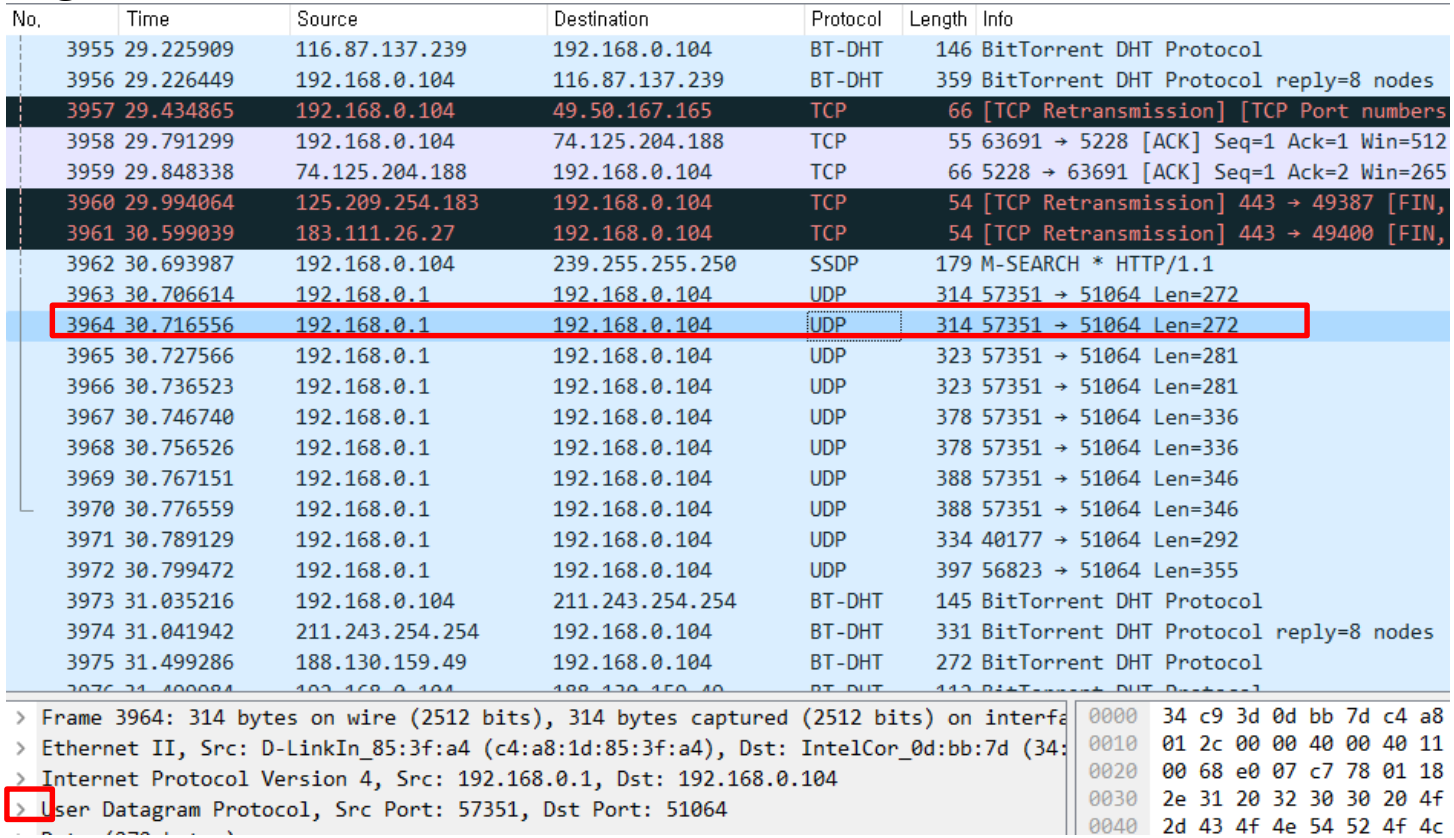
예제

- 몇 초 후 52,222 포트에 사용자 데이터그램이 도착한다.
- 입력 모듈이 테이블을 검사한 결과 이 목적지를 위한 엔트리를 발견하지 못한다.
- 사용자 데이터그램은 폐기되고 발신지에 “unreachable port" 메시지를 보낼 것을 ICMP에 요청한다.

State	Process ID	Port Number	Queue Number
IN-USE	2,345	52,010	34
IN-USE	3,422	52,011	43
IN-USE	4,978	52,014	
IN-USE	4,652	52,012	38
FREE			

6. UDP 덤프 분석

- UDP는 TCP와 쌍을 이루는 전송층 프로토콜이다.
- 와이어샤크를 실행한 다음 패킷을 캡처한 후에 UDP 프로토콜을 선택해보자.
- 패킷 목록 정보에서 3964번 프레임을 선택한 다음 패킷 상세 정보의 [User Datagram Protocol] 앞에 있는 [>]를 클릭해 본다.



6. UDP 덤프 분석

- [User Datagram Protocol] 앞의 [>]를 클릭하면 UDP 헤더가 전개된다.

```
> Frame 3964: 314 bytes on wire (2512 bits), 314 bytes captured (2512 bits) on interfa
> Ethernet II, Src: D-LinkIn_85:3f:a4 (c4:a8:1d:85:3f:a4), Dst: IntelCor_0d:bb:7d (34
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.104
v User Datagram Protocol, Src Port: 57351, Dst Port: 51064
  ① Source Port: 57351
  ② Destination Port: 51064
  ③ Length: 280
    Checksum: 0xc0c7 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 88]
  > [Timestamps]
  ④ UDP payload (272 bytes)
> Data (272 bytes)
```

6. UDP 덤프 분석

- UDP는 DNS나 DHCP에서 이용될 뿐만 아니라 IP 전화나 비디오 회의, 네트워크 게임과 같은 멀티미디어 통신에 아주 적합한 프로토콜이다.
- UDP는 비연결형이므로 연결 확인 등이 없고 헤더도 단순해서 고속으로 처리할 수 있다.
- 멀티미디어 통신에서는 일부 정보가 누락되어도 통신 그 자체에는 그다지 영향이 없기 때문에 재전송이나 순서 제어 등은 필요 없기 때문에 이와 같은 이유 때문에 UDP를 이용한다.

6. UDP 덤프 분석

- 다음 그림은 Skype라는 IP 전화 소프트웨어로 통신할 경우의 패킷 캡처 예이다.

No.	Time	Source	Destination	Protocol	Length	Info
198	61.869551	192.168.1.1	192.168.1.2	DNS	133	Standard query response 0x3137 PTR 140.195.216.80.in-addr.arpa PTR c80…
200	62.546310	192.168.1.2	192.168.1.1	DNS	99	Standard query 0x3138 A client-86-31-70-81.midd.adsl.virgin.net
201	62.546352	192.168.1.2	192.168.1.1	DNS	107	Standard query 0x3139 A host86-130-63-111.range86-130.btcentralplus.com
202	62.546377	192.168.1.2	192.168.1.1	DNS	103	Standard query 0x313a A AOrleans-257-1-93-25.w86-220.abo.wanadoo.fr
203	62.546401	192.168.1.2	192.168.1.1	DNS	92	Standard query 0x313b A c80-216-195-140.cm-upc.chello.se
204	62.549479	192.168.1.1	192.168.1.2	DNS	115	Standard query response 0x3138 A client-86-31-70-81.midd.adsl.virgin.n…
205	62.551917	192.168.1.1	192.168.1.2	DNS	123	Standard query response 0x3139 A host86-130-63-111.range86-130.btcentr…
206	62.554242	192.168.1.1	192.168.1.2	DNS	119	Standard query response 0x313a A AOrleans-257-1-93-25.w86-220.abo.wana…
207	62.556683	192.168.1.1	192.168.1.2	DNS	108	Standard query response 0x313b A c80-216-195-140.cm-upc.chello.se A 80…
214	66.122955	192.168.1.2	165.124.253.241	UDP	60	35990 → 15294 Len=18
215	66.249381	165.124.253.241	192.168.1.2	UDP	60	15294 → 35990 Len=11
216	66.249507	192.168.1.2	165.124.253.241	UDP	65	35990 → 15294 Len=23
217	66.357031	192.168.1.2	192.168.1.1	DNS	85	Standard query 0x313c PTR 124.6.190.65.in-addr.arpa
218	66.357080	192.168.1.2	192.168.1.1	DNS	88	Standard query 0x313d PTR 241.253.124.165.in-addr.arpa
219	66.370892	165.124.253.241	192.168.1.2	UDP	93	15294 → 35990 Len=51
220	66.371270	192.168.1.2	81.236.228.111	UDP	60	35990 → 42764 Len=18
221	66.431975	81.236.228.111	192.168.1.2	UDP	93	42764 → 35990 Len=51
222	66.432373	192.168.1.2	86.142.117.124	UDP	60	35990 → 36474 Len=18
223	66.468533	192.168.1.1	192.168.1.2	DNS	135	Standard query response 0x313c PTR 124.6.190.65.in-addr.arpa PTR cpe-0…
224	66.712287	192.168.1.1	192.168.1.2	DNS	140	Standard query response 0x313d PTR 241.253.124.165.in-addr.arpa PTR en…
225	67.092614	192.168.1.2	192.168.1.1	DNS	96	Standard query 0x313e A cpe-065-190-006-124.triad.res.rr.com

```
> Frame 221: 93 bytes on wire (744 bits), 93 bytes captured (744 bits)
> Ethernet II, Src: AskeyCom_19:27:15 (00:16:e3:19:27:15), Dst: 3Com_96:7b:da (00:04:7
> Internet Protocol Version 4, Src: 81.236.228.111, Dst: 192.168.1.2
> User Datagram Protocol, Src Port: 42764, Dst Port: 35990
v Data (51 bytes)
    Data: e23a025f5ec745914e1ceb1117e878e03c920351a275564f559024d926539fc058f4e1aa…
    [Length: 51]
```

```
0000  00 04 76 96 7b da 00 16  e3 19 27 15 08 00 45 00   ··v·{···  ··'···E·
0010  00 4f 45 37 00 00 67 11  16 61 51 ec e4 6f c0 a8   ·OE7··g·  ·aQ··o··
0020  01 02 a7 0c 8c 96 00 3b  ff fa e2 3a 02 5f 5e c7   ·······;  ···:·_^·
0030  45 91 4e 1c eb 11 17 e8  78 e0 3c 92 03 51 a2 75   E·N·····  x·<··Q·u
0040  56 4f 55 90 24 d9 26 53  9f c0 58 f4 e1 aa 65 30   VOU·$·&S  ··X···e0
0050  7c 8c 77 a4 35 93 b8 46  8b 45 15 a4 e5           |·w·5··F  ·E···
```

6. UDP 덤프 분석

- 와이어샤크는 Skype 프로토콜을 디코드(해석)할 수 없기 때문에 패킷 상세 정보에서는 [Data(75bytes)]와 같이 그대로 표시된다.
- 데이터를 세분화하여 중단되지 않도록 보내는 것을 [스트리밍 (streaming)]이라고 하는데 이 스트리밍 기술에 의해 지연이 큰 인터넷에서도 Skype로 쾌적하게 전화할 수 있게 되어 있다.
- 또한 IP 전화 등에서 UDP에 RTP(Real-time Transport Protocol)가 표준적으로 이용되어 미디어 데이터가 전송된다.

Thank You