

SW 신기술 기말 고사

힌트

1. ARP 프로토콜이 필요한 이유 및 물리주소 지정 방법
2. IP 패킷 길이 및 단편화 방법
3. ICMP 프로토콜을 사용하는 이유 및 ICMP 오류 메시지 구성
4. UDP 헤더의 구조
5. 소켓의 개념, TCP의 3 단계 핸드셰이크
6. DNS 해석기의 해석 방법(반복적, 귀환적)
7. 인버스 도메인 메시지 표시 방법
8. DNS 메시지에서 DNS 이름 표시 방법
9. HTTP 메시지 헤더의 종류
10. DHCP 클라이언트와 서버 사이에 교환하는 4가지 메시지 종류와 내용과 의미

힌트

1. ARP 프로토콜이 필요한 이유 및 물리주소 지정 방법

- 8장
- 예상 출제 방식 : ARP 프로토콜이 필요한 이유를 묻고, 그림을 주면서 빈칸을 채우는 방식, wireshark 캡처 이미지를 주고서 분석하는 방식

▼ 새로 정리

- ARP 프로토콜이 필요한 이유 (= 동적 매핑이 필요한 이유)

ARP (Address Resolution Protocol)은 IP 주소를 물리적인 MAC 주소로 변환하는 데 사용되는 프로토콜입니다. ARP가 필요한 주요 이유는 ARP는 물리적 네트워크에서 IP 주소를 MAC 주소로 동적으로 매핑하는 기능을 제공합니다. 이를 통해 물리 주소를 몰라도 논리 주소만 가지고서 통신 할 수 있어 효율적 입니다. 컴퓨터가 네트워크에 새로 추가되거나 네트워크 구성이 변경될 때 마다 수동으로 MAC 주소를 업데이트할 필요가 없습니다.

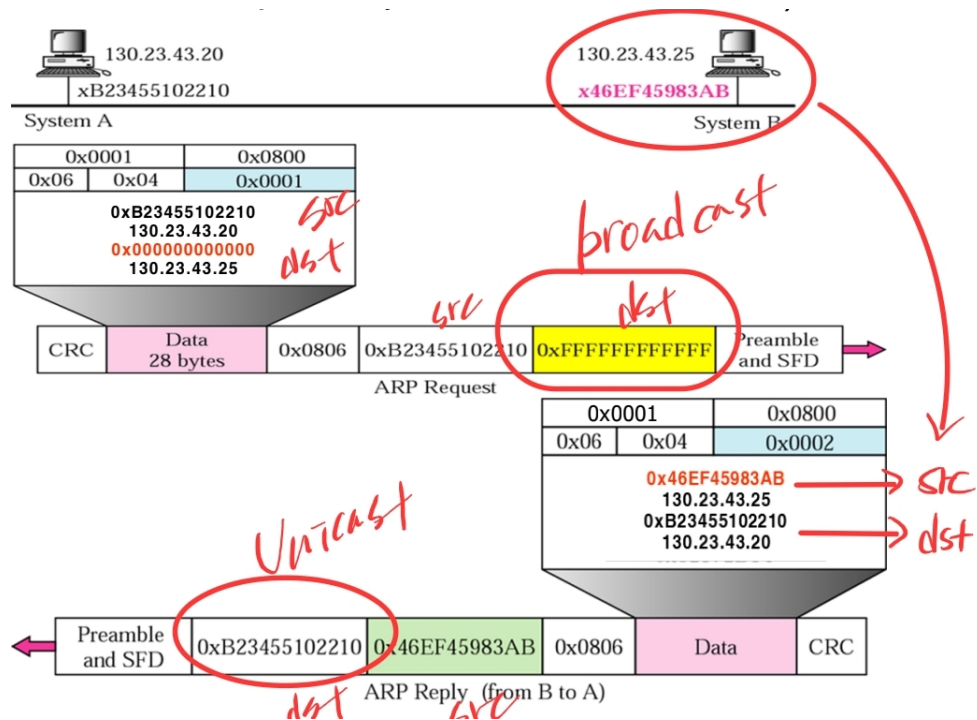
- 물리 주소 지정 방법

1. 먼저 송신 컴퓨터는 수신 컴퓨터의 IP 주소를 알고 있지만 MAC 주소를 모릅니다. 이 경우 송신 컴퓨터는 ARP 요청 패킷을 생성합니다. 이 패킷에는 송신 컴퓨터의 IP 및 MAC 주소, 그리고 수신 컴퓨터의 IP 주소가 포함됩니다. 그런 다음 이 ARP 요청 패킷을 네트워크에 브로드캐스트합니다. (ARP request - Broadcast)

2. 모든 컴퓨터는 이 ARP 요청 패킷을 수신하지만, 요청에 포함된 IP 주소가 자신의 IP 주소와 일치하는 컴퓨터만 응답합니다. 이 컴퓨터는 자신의 MAC 주소를 포함하는 ARP 응답 패킷을 생성하고 이를 송신 컴퓨터로 직접 전송합니다. (ARP reply - unicast)

3. 송신 컴퓨터는 ARP 응답 패킷을 수신하면 수신 컴퓨터의 MAC 주소를 알게 됩니다. 이 정보는 일정 시간 동안 송신 컴퓨터의 ARP 캐시에 저장되어 나중에 같은 수신 컴퓨터로 데이터를 전송할 때 재사용됩니다.

이런 식으로 ARP는 IP 주소를 MAC 주소로 매핑하므로 컴퓨터 간의 효율적인 데이터 교환을 가능하게 합니다.



▼ 컴퓨터네트워크 재할용

ARP :

논리주소를 물리주소로 변환시키는 프로토콜

네트워크 Layer의 프로토콜 단 ip에 포함되지 않는 독립적인 프로토콜

RARP :

물리주소를 논리주소로 변환시키는 프로토콜

네트워크 Layer의 프로토콜 단 ip에 포함되지 않는 독립적인 프로토콜

필요한 이유 :

패킷전송은 물리적인 네트워크를 통해 이루어지므로 수신측의 물리주소를 알아야 한다.

동작방식 :

패킷이 이동하려면 물리주소가 필요하다. 알맞은 물리주소를 찾기 위해서 논리주소를 사용한다.

호스트1 → **호스트2**로 통신을 시도 상황이라면... ARP 프로토콜 사용

1. 호스트1 브로드캐스트 “(호스트2번의 ip주소)의 물리 주소를 아는 사람?”
2. 라우터조차 모른다? 그러면 → 라우터의 물리 주소를 반환 유니캐스트(“그거 일단 나주셈”) → 패킷이 라우터로 이동
3. 라우터”(호스트2번의 ip주소)로 보내주세요" → 인터넷이랑 연결된 라우터 물리주소로 패킷 전달 → 패킷이 인터넷으로 전달됨
4. 인터넷에서는 물리 주소 없이 오직 논리 주소만으로 찾아감 → 드디어 호스트2의 물리 주소를 아는 라우터를 찾음 → 물리 주소를 해당 라우터의 물리 주소로 설정하고 패킷을 전달 → 라우터가 호스트2의 물리 주소로 바뀌어서 호스트2에게 전달

자신의 논리 주소를 모른다면 ... RARP 사용

1. broadcast “나 (호스트1의 물리 주소)인데 논리 주소 좀 알려줄 사람?”
2. unicast “(호스트1의 물리 주소)아 너 (호스트1의 논리 주소)다.”

ARP request는 broadcast

ARP reply는 unicast

2. IP 패킷 길이 및 단편화 방법

- 9장
- 예상 출제 방식 : IP 패킷 구조도에서 빈 칸을 채우는 방식, 구조도를 보고서 질문에 대답하기.

▼ 새로 정리

- IPv4
: IP 패킷의 최대 길이는 65,535 바이트입니다.
: 그러나 이론적인 최대 패킷 크기와는 달리, 실제 네트워크 환경에서는 패킷 크기가 일반적으로 MTU (Maximum Transmission Unit)라고 하는 장치별 제한에 의해 제한됩니다. 이 값은 일반적으로 Ethernet 네트워크에서는 1500 바이트입니다.
: IP 헤더의 크기는 20 ~ 60 바이트이고, 나머지는 데이터 부분에 할당됩니다.

VER : IP 프로토콜 버전
HLEN : 헤더 길이 (4비트, 4바이트를 1로 축약)
Service Type : TOS
Total length : 16비트, 20~65536(= 2의 16승) bytes

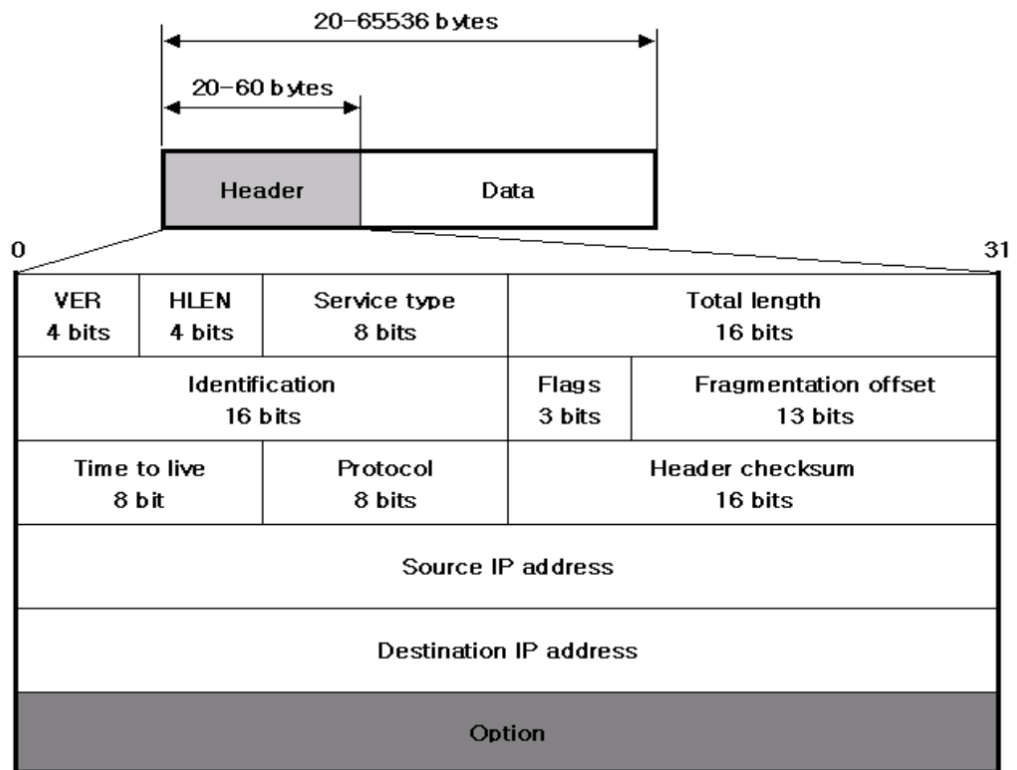
▼ 컴퓨터네트워크 재활용

IP 패킷의 단편화할때 ip head의 값을 계산하는 방법이 있다.

ip head는 바뀌지 않지만 다른 부분은 바뀐다. (more bit, fragement offset?)

바뀌는 것과 바뀌지 않는 것.

IP 패킷의 구조



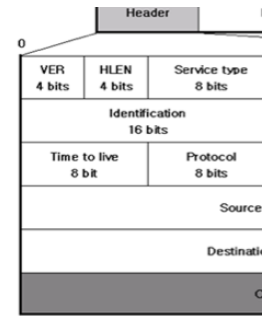
IP 패킷 단편화란?

패킷을 전송하고자 하는데 패킷의 크기가 MTU(Maximum Transmission Unit)를 초과하면 한번에 전송할 수 없다.

패킷을 MTU 이하의 조각으로 분할하는 것을 단편화(Fragmentation), 분할된 조각을 단편(Fragment)이라고 한다.

✓ 플래그 (Flag)

- 세 개의 비트로 단편화 여부 표시
- 미사용(예약) 비트, Don't fragment, More fragment bit
- 첫번째 비트 : 예약 (reserved)
- 두번째 비트 : 단편화 금지 (Do not Fragment): D=1
 - . 만약 해당 서브넷이 단편화를 필요로 하고 D=1이면 패킷 폐기
 - . ICMP (Internet Control Message Protocol)로 오류정보를 송신자에 전달
- 세번째 비트 : 추가 단편화 비트 (More Fragment)
 - . M=1 추가적인 fragment 들이 있음, M=0 마지막 패킷



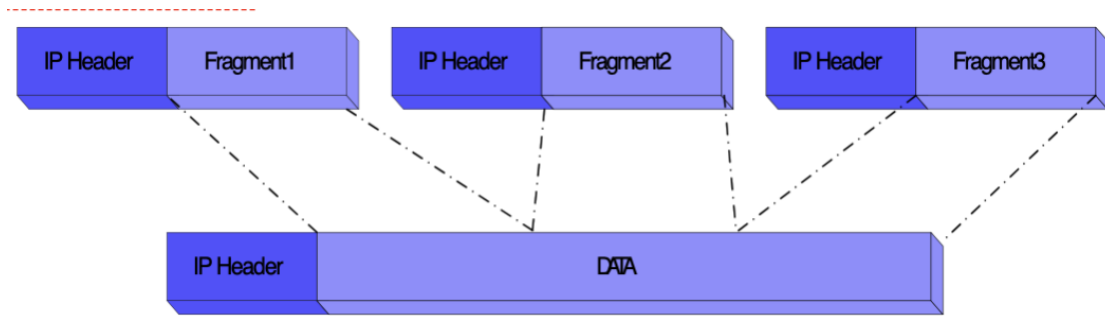
Flags 3bit

- 첫번째 비트 : 미사용 비트
- 두번째 비트 : 단편화 금지 → 단편화가 필요하면 단편화를 수행할지 선택 → D = 1 이면 단편화 금지
- 세번째 비트 : 추가 단편화 비트 → 해당 단편이 마지막 단편인지 다른 단편이 있는지 표시 → M = 1 아직 더 많은 단편이 있음, M = 0 이면 마지막 패킷임.

Fragmentation offset

단편화된 조각들을 하나의 데이터그램으로 합칠 때, 상대적인 위치를 표시

해당 단편이 원래는 몇번째에 위치했는지 표시 (0부터 시작)(8바이트씩 묶음)(헤더는 제외)



단편화 예시

편집 원본 편집

- 4000바이트의 패킷을 전송하려고 하는데 MTU가 1500이다.
- 패킷은 아래와 같이 분할된다.

순서	페이로드	헤더	More Flag	offset
1	1480	20	1	0
2	1480	20	1	185
3	1020	20	0	370

• 해설

- 패킷이 4000바이트라면 페이로드 3980바이트, 헤더 20바이트이다.
- 각 단편 모두 헤더를 가져야 하므로 최대 1480바이트로 분할될 수 있다.
- 즉 1480, 1480, 1020 으로 분할된다.
- 첫번째, 두번째 단편은 뒤에 이어질 단편이 있으므로 more flag가 1이다.
- offset은 8바이트 단위로 표시된다. $1480 / 8 = 185$, $2960 / 8 = 370$

$$2960 = 1480 + 1480$$

$$4000 = 20 + 1480 + 1480 + 1020$$

페이로드 : 순수한 적재량, 순수한 데이터의 량

https://itwiki.kr/w/IP_단편화

3. ICMP 프로토콜을 사용하는 이유 및 ICMP 오류 메시지 구성

- 10장
- 예상 출제 방식 : ICMP의 필요성 & ICMP를 IP 데이터 그램에 캡슐화하는 방법

▼ 새로 추가

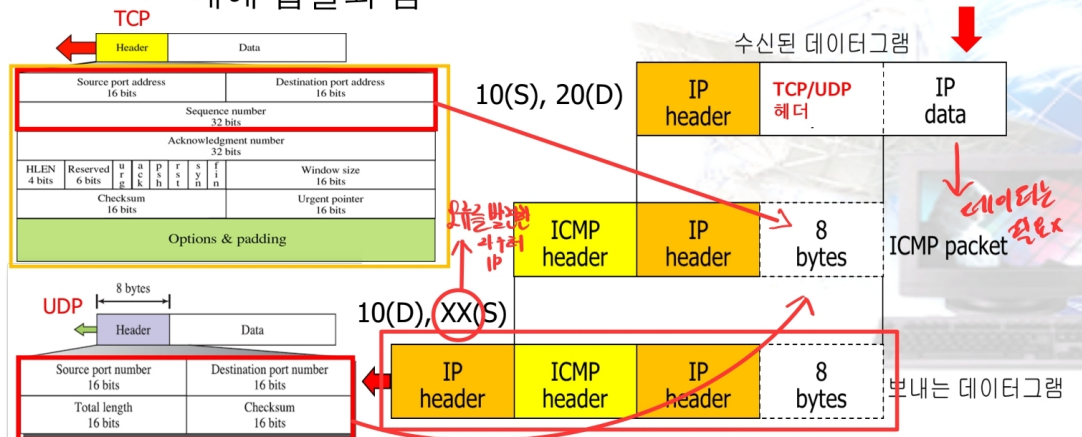
- 사용 이유 : 오류에 관한 보고 기능과 네트워크 상태 진단 기능을 통해 IP를 보조하는 기능을 수행
 1. 오류 보고
 2. 네트워크 진단 및 트러블슈팅
 3. 네트워크 운영 정보 제공
- 예외
 - ICMP 오류 메시지를 전달하는 데이터 그램에 대해서 ICMP 오류 메시지를 생성하지 않음.
 - 첫번째 단편에 대해서만 ICMP 오류 메시지 생성

10.3 오류 보고



■ 오류 메시지를 위한 데이터 필드의 내용

- ICMP는 오류 패킷을 생성하고 이것은 IP 데이터그램 내에 캡슐화 됨



▼ 컴퓨터네트워크 재활용

• 필요성 :

- IP는 비연결 지향형 프로토콜로, 패킷이 확실히 전송된다는 보장이 없다
- IP 패킷이 전송되는 목적지에 전달되지 못할 수 있고, 전달되더라도 원하는 서비스 포트가 존재하지 않는 경우도 발생한다.
- IP 혼자서는 패킷 전달이 성공했는지 실패했는지 알 수 없다. 그래서 송신측의 상태를 알려주는 ICMP가 필요하다.

• 기능 : 오류에 관한 보고 기능과 네트워크 상태 진단 기능을 통해 IP를 보조하는 기능을 수행

• ICMP 메시지 종류

오류 보고 메시지 : 라우터나 호스트에서 IP 패킷을 전달할 때 문제가 발생하였을 경우 사용

네트워크 진단 기능 : 라우터나 네트워크에 대한 정보를 얻고자 할 때 사용

Category	Type	Message
Error-reporting messages	3	Destination unreachable 목적지에 도달할 수 없음
	4	Source quench 라우터 버퍼 부족
	11	Time exceeded 응답 시간 초과
	12	Parameter problem
	5	Redirection
Query Messages	8 or 0	Echo request or reply
	13 or 14	Timestamp request or reply
	17 or 18	Address mask request or reply
	10 or 9	Router solicitation or redirection

출처 : <https://lopicit.tistory.com/285>

4. UDP 헤더의 구조

- 11장
- 예상 출제 방식 : UDP의 전체 혹은 데이터의 최대 크기를 구하기.

▼ 새로 추가

UDP 헤더는 매우 간단하며, 이는 UDP가 신뢰성보다는 효율성을 중요시하는 프로토콜.

이로 인해 UDP는 스트리밍 미디어, VoIP, DNS 등과 같이 데이터의 신속한 전달이 중요하고, 일부 데이터 패킷의 손실이 허용되는 애플리케이션에 주로 사용됩니다.

UDP (User Datagram Protocol)는 8 바이트의 고정된 크기의 헤더를 가진다.

Src port 16 bits	Dst port 16 bits
Total length 16 bits	Checksum 16bits

Total length : 헤더와 데이터를 합한 전체 길이. 최대 2의 16승 바이트인 65535 바이트에서 IP 헤더의 최소 크기인 20바이트를 뺀 65515 바이트다.

UDP 데이터의 최대 크기는 $65535 - 20 - 8 = 65507$ 바이트다.

▼ GPT

UDP(User Datagram Protocol) 헤더의 구조는 상대적으로 간단하며, 아래와 같은 4개의 필드로 구성되어 있습니다:

1. ****Source Port (16 bits)****: 이 필드는 패킷을 전송하는 프로세스의 포트 번호를 나타냅니다. 이는 응답을 올바른 프로세스로 라우팅하는 데 사용됩니다.
2. ****Destination Port (16 bits)****: 이 필드는 패킷의 목적지 프로세스의 포트 번호를 나타냅니다. 이는 패킷을 올바른 프로세스로 라우팅하는 데 사용됩니다.
3. ****Length (16 bits)****: 이 필드는 전체 UDP 데이터그램의 길이를 바이트 단위로 나타냅니다. 이 길이는 UDP 헤더와 페이로드를 모두 포함합니다.
4. ****Checksum (16 bits)****: 이 필드는 데이터그램의 오류 검사를 위해 사용됩니다. 이 값은 송신측에서 데이터그램을 전송하기 전에 계산되며, 수신측에서는 이 값을 사용하여 데이터그램이 전송 도중에 오류가 발생했는지 확인합니다.

UDP 헤더는 매우 간단하며, 이는 UDP가 신뢰성보다는 효율성을 중요시하는 프로토콜이기 때문입니다. 이로 인해 UDP는 스트리밍 미디어, VoIP, DNS 등과 같이 데이터의 신속한 전달이 중요하고, 일부 데이터 패킷의 손실이 허용되는 애플리케이션에 주로 사용됩니다.

5. 소켓의 개념, TCP의 3 단계 핸드셰이크

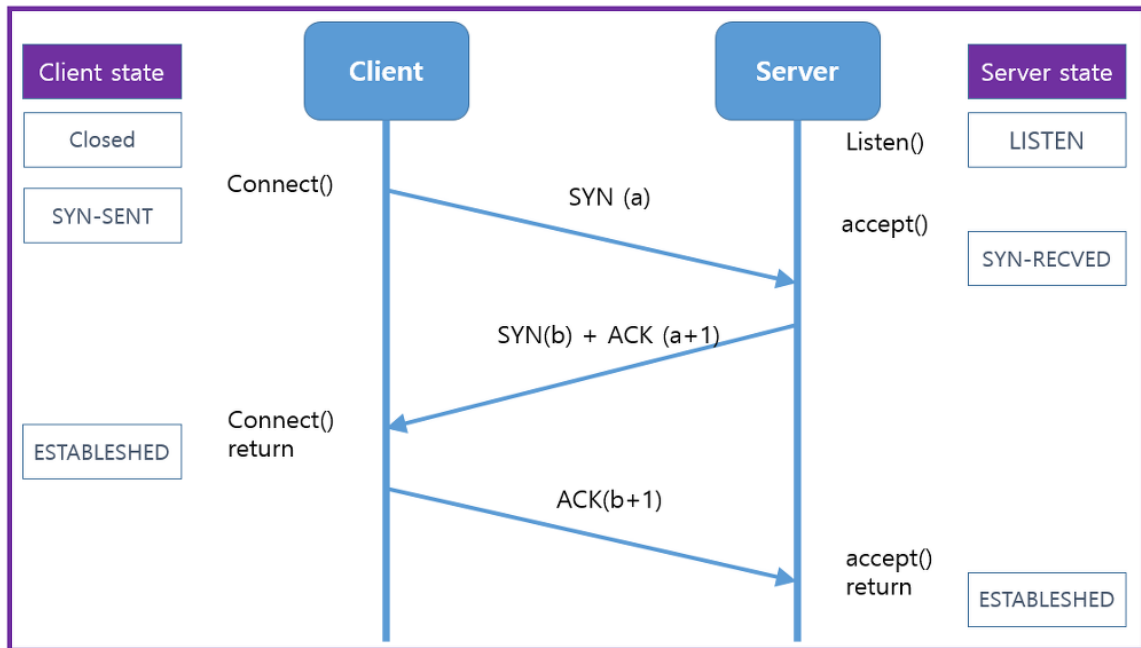
- 12장
- 예상 출제 방식 : 비교적 나올 가능성이 크지 않음. TCP의 3 단계 핸드셰이크 과정 혹은 이유를 서술.

▼ 새로 추가

- TCP는 연속성보다 신뢰성이 필요한 통신에 사용한다.

소켓(socket)은 네트워크에서 통신을 할 수 있는 인터페이스를 나타냅니다. 소켓은 보통 프로세스 또는 애플리케이션 간의 네트워크 통신을 가능하게 하는 엔드포인트(endpoint)로 볼 수 있습니다. 소켓은 IP 주소와 포트 번호의 조합으로 표현되며, 이를 통해 네트워크 상의 특정 위치를 식별할 수 있습니다.

- 3-Way handshake는 TCP의 연결을 초기화 할 때 사용한다면, 4-Way handshake는 세션을 종료하기 위해 수행되는 절차입니다.



TCP의 3단계 핸드셰이크(3-way handshake)는 TCP 연결을 초기화하는 과정입니다. 이 핸드셰이크는 신뢰할 수 있는 통신을 보장하며, 두 시스템이 서로 데이터를 수신하고 전송할 준비가 되었음을 확인합니다. 이 3단계 핸드셰이크가 완료되면, 클라이언트와 서버 사이에 TCP 연결이 확립되고, 양방향 데이터 전송이 가능해집니다. 이 과정은 신뢰성 있는 데이터 전송을 보장하며, 패킷 손실이나 순서 재정렬 등의 문제를 방지합니다.

▼ 컴퓨터네트워크 재할용

유닉스의 소켓은 IP와 PORT를 합친 개념이다.

TCP 특징

- 연결형 서비스로 가상 회선 방식을 제공한다.
- 3-way handshaking 과정을 통해 연결을 설정하고 4-way handshaking을 통해 해제한다.
- 흐름 제어 및 혼잡 제어.
- 높은 신뢰성을 보장한다.
- UDP보다 속도가 느리다.
- 전이중(Full-Duplex), 점대점(Point to Point) 방식.

가상 회선 방식을 제공한다는 것은 발신지와 수신지를 연결하여 패킷을 전송하기 위한 논리적 경로를 배정한다는 말입니다. TCP는 연결형 서비스로 신뢰성을 보장합니다. 그래서 3-way handshaking 과정도 사용하고, 데이터의 흐름제어나 혼잡 제어와 같은 기능도 합니다. 하지만 이러한 기능때문에 UDP보다 속도가 느리게 됩니다. (이러한 기능들이 CPU를 사용하기 때문입니다.) **연속성보다 신뢰성이 중요할때 사용하는 프로토콜입니다**

▼ GPT

소켓(socket)은 네트워크에서 통신을 할 수 있는 인터페이스를 나타냅니다. 소켓은 보통 프로세스 또는 어플리케이션 간의 네트워크 통신을 가능하게 하는 엔드포인트(endpoint)로 볼 수 있습니다. 소켓은 IP 주소와 포트 번호의 조합으로 표현되며, 이를 통해 네트워크 상의 특정 위치를 식별할 수 있습니다.

소켓은 크게 두 가지 타입이 있습니다:

1. ****스트림 소켓 (Stream Sockets)****: 이는 TCP(Transmission Control Protocol) 위에서 동작하며, 신뢰성 있는 연결 지향적 서비스를 제공합니다. 스트림 소켓은 데이터를 순서대로 보내고, 패킷의 손실, 중복, 또는 재정렬 없이 데이터를 안정적으로 전송하는 데 사용됩니다.
2. ****데이터그램 소켓 (Datagram Sockets)****: 이는 UDP(User Datagram Protocol) 위에서 동작하며, 연결을 설정하지 않고 데이터를 전송합니다. 데이터그램 소켓은 연결 설정 없이 빠르게 데이터를 전송하는 것이 필요한 경우에 사용되며, 데이터의 순서나 도착을 보장하지 않습니다.

소켓 프로그래밍은 네트워크 프로그래밍의 핵심 요소 중 하나로, 애플리케이션 간의 데이터 전송을 가능하게 합니다. 예를 들어, 웹 브라우저와 웹 서버 사이의 통신은 HTTP 프로토콜을 사용하는데, 이 HTTP 프로토콜은 TCP 소켓 위에서 동작합니다.

TCP의 3단계 핸드셰이크(3-way handshake)는 TCP 연결을 초기화하는 과정입니다. 이 핸드셰이크는 신뢰할 수 있는 통신을 보장하며, 두 시스템이 서로 데이터를 수신하고 전송할 준비가 되었음을 확인합니다. 다음은 3단계 핸드셰이크의 과정입니다:

1. ****SYN (Synchronize)****: 클라이언트가 서버에 연결 요청을 보내기 시작합니다. 이 패킷은 초기 순서 번호를 포함하며, 이 순서 번호는 클라이언트가 생성하는 임의의 값입니다. 이 패킷은 SYN 플래그를 설정하여, 이 패킷이 연결 요청임을 나타냅니다.
2. ****SYN-ACK (Synchronize-Acknowledgement)****: 서버는 클라이언트의 요청을 수락하고, 클라이언트에게 응답을 보냅니다. 이 응답은 서버의 초기 순서 번호를 포함하며, SYN과 ACK 플래그가 설정됩니다. 이것은 클라이언트의 SYN 요청을 수락하고, 클라이언트에게 자신의 초기 순서 번호를 보내는 것을 나타냅니다.
3. ****ACK (Acknowledgement)****: 마지막으로, 클라이언트는 서버에게 응답을 보내, 서버의 SYN-ACK 패킷을 수신하였음을 확인합니다. 이 ACK 패킷은 클라이언트의 초기 순서 번호에 1을 더한 값을 포함하며, 서버의 초기 순서 번호에 1을 더한 값을 포함합니다. 이 패킷은 ACK 플래그가 설정됩니다.

이 3단계 핸드셰이크가 완료되면, 클라이언트와 서버 사이에 TCP 연결이 확립되고, 양방향 데이터 전송이 가능해집니다. 이 과정은 신뢰성 있는 데이터 전송을 보장하며, 패킷 손실이나 순서 재정렬 등의 문제를 방지합니다.

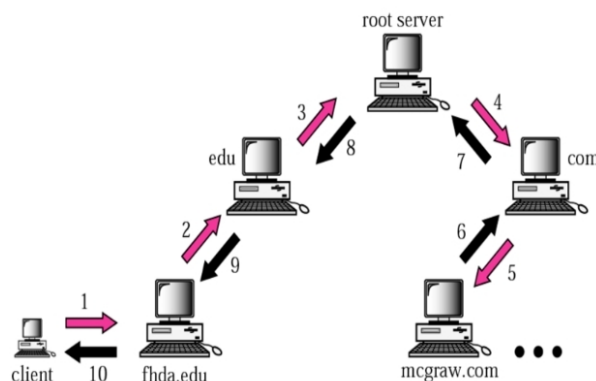
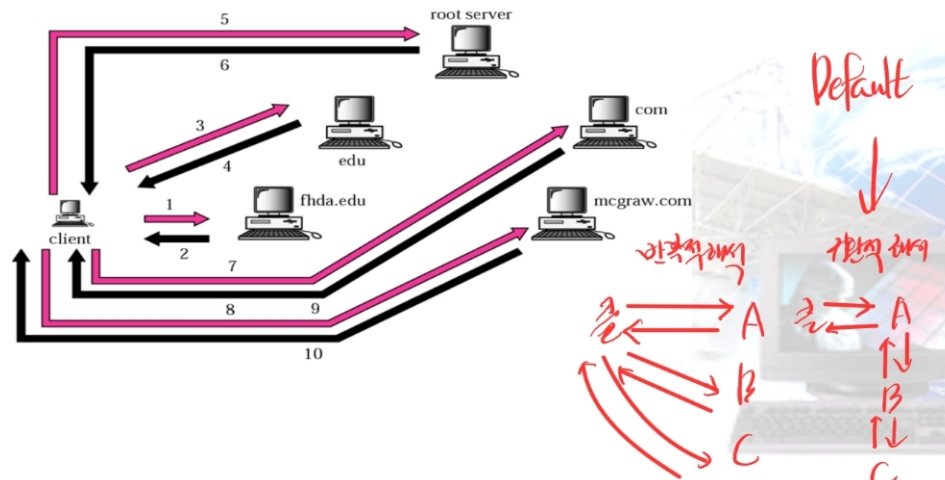
6. DNS 해석기의 해석 방법(반복적, 귀환적)

- 13장
- 예상 출제 방식 : 출제하기엔 너무 쉽다. DNS 메시지에 반복적, 귀환적 요청을 구분짓는 flag에 대한 설명.

▼ 새로 추가

도메인 네임 시스템은 호스트의 도메인 이름을 호스트의 네트워크 주소로 바꾸거나 그 반대의 변환을 수행할 수 있도록 하기 위해 개발되었다.

DNS(Domain Name System) 해석기(resolver)는 도메인 이름을 IP 주소로 변환하는 역할을 합니다. 이 변환 과정은 주로 반복적(iterative) 또는 귀환적(recursive) 두 가지 방법으로 수행됩니다.



- DNS 메시지에서의 반복적, 귀환적 응답에 해당하는 내용.
-? : 8번째 비트!!
- RD (recursion desired) : 1은 귀환적 응답을 요청을 의미. 0은 반복적 응답. 질의 메시지에서 사용.

▼ GPT

DNS(Domain Name System) 해석기(resolver)는 도메인 이름을 IP 주소로 변환하는 역할을 합니다. 이 변환 과정은 주로 반복적(iterative) 또는 귀환적(recursive) 두 가지 방법으로 수행됩니다.

1. ****반복적 해석 (Iterative Resolution)****: DNS 클라이언트는 DNS 서버에 질의를 보내고, DNS 서버는 가장 적합한 다음 DNS 서버의 주소를 응답으로 제공합니다. 이 과정을 클라이언트가 원하는 정보를 얻을 때까지 반복합니다. 반복적 해석에서 클라이언트는 모든 요청을 직접 관리하며, 각 단계에서 어떤 서버에게 질의할지 결정합니다.

2. ****귀환적 해석 (Recursive Resolution)****: DNS 클라이언트는 DNS 서버에 질의를 보내고, 이 서버는 요청된 정보를 제공할 때까지 다른 DNS 서버에게 질의를 계속합니다. 서버가 응답을 받으면 이 정보를 원래의 클라이언트에게 전달합니다. 귀환적 해석에서 DNS 클라이언트는 단일 질의를 보내고, 응답을 기다립니다. 모든 작업은 DNS 서버에 의해 수행되며, 클라이언트는 결과만 받게 됩니다.

이 두 방법은 각각의 장단점이 있습니다. 반복적 해석은 클라이언트가 직접 제어하기 때문에 유연성이 높지만, 클라이언트가 모든 작업을 수행해야 하므로 리소스를 더 많이 사용할 수 있습니다. 귀환적 해석은 서버가 모든 작업을 수행하므로 클라이언트의 부담을 줄일 수 있지만, 서버에 더 많은 부하가 가해질 수 있습니다.

7. 인버스 도메인 메시지 표시 방법

- 13장
- 예상 출제 방식 : 인버스 도메인 질의에는 IP 주소가 반대로 되어야 하고, 마지막에 `in-addr.arpa`를 붙인다.

▼ 새로 추가

인버스 도메인(inverse domain)은 IP 주소를 도메인 이름으로 변환하는 데 사용됩니다. 인버스 도메인 질의에는 IP 주소가 반대로 되어야 하고, 마지막에 `in-addr.arpa`를 붙인다.

1. IP 주소를 역순으로 나열합니다. 예를 들어, IP 주소가 "192.0.2.1"인 경우, 이를 "1.2.0.192"로 나열합니다.
2. ".in-addr.arpa"를 역순으로 나열한 IP 주소의 끝에 추가합니다. 위의 예에서는 "1.2.0.192.in-addr.arpa"가 됩니다.

▼ GPT

인버스 도메인(inverse domain)은 IP 주소를 도메인 이름으로 변환하는 데 사용됩니다.

이를 위해 DNS에는 PTR(Pointer) 레코드라는 특별한 타입의 레코드가 있습니다. PTR 레코드는 주로 역방향 DNS 조회(reverse DNS lookup)에 사용되며, 주로 메일 서버와 같이 IP 주소가 신뢰할 수 있는 호스트 이름과 연결되어 있는지 확인하는 데 사용됩니다.

인버스 도메인 메시지는 아래와 같은 방법으로 표시됩니다:

1. IP 주소를 역순으로 나열합니다. 예를 들어, IP 주소가 "192.0.2.1"인 경우, 이를 "1.2.0.192"로 나열합니다.

2. ".in-addr.arpa"를 역순으로 나열한 IP 주소의 끝에 추가합니다. 위의 예에서는 "1.2.0.192.in-addr.arpa"가 됩니다.

따라서, PTR 레코드는 다음과 같은 형식으로 나타낼 수 있습니다:

```
1.2.0.192.in-addr.arpa. IN PTR example.com.
```

이는 "192.0.2.1" IP 주소가 "example.com" 도메인 이름에 매핑되어 있다는 것을 나타냅니다.

IPv6에서는 비슷한 방식이 적용되지만, ".ip6.arpa" 도메인이 사용됩니다. 또한, 각각의 16진수 값이 개별적으로 역순으로 나열됩니다. 예를 들어, "2001:0db8:85a3:0000:0000:8a2e:0370:7334" IPv6 주소의 PTR 레코드는 "4.3.3.7.0.3.e.2.a.8.0.0.0.0.0.0.3.a.5.8.8.b.d.0.1.0.0.2.ip6.arpa"가 됩니다.

8. DNS 메시지에서 DNS 이름 표시 방법

- 13장
- 예상 출제 방식 : 이미지를 제시하고 해석하기. 하는 이유 설명?

▼ 새로 추가

- 압축

: DNS에서는 하나의 메시지 안에 도메인 이름이 반복되면 반복되는 이름을 오프셋 포인터로 대체한다.

: 이러한 방식을 사용하면 DNS 메시지에서 도메인 이름을 효율적으로 표현하고 메시지 크기를 최소화할 수 있다.

▼ GPT

DNS 메시지에서 이름은 라벨(label)이라는 개념을 사용하여 표현됩니다. 각 라벨은 도메인 이름의 구성요소를 나타냅니다. 예를 들어, "www.example.com"이라는 도메인 이름은 "www", "example", "com" 세 개의 라벨로 구성되어 있습니다.

DNS 메시지에서 이름을 표현하는 데는 두 가지 방법이 있습니다: 직접 참조와 포인터를 사용한 참조입니다.

1. **직접 참조**: 각 라벨의 길이를 바이트로 나타낸 후 해당 라벨이 따릅니다. 각 라벨이 이런 식으로 표시되고, 끝은 길이가 0인 라벨로 표시합니다. 예를 들어, "www.example.com"은 다음과 같이 표현됩니다: 3www7example3com0.

2. **포인터를 사용한 참조** : DNS 메시지 내에서 이전에 사용된 도메인 이름을 재사용할 때 포인터를 사용하여 참조합니다. 이는 메시지를 축소하고 중복을 줄이는 데 도움이 됩니다. 포인터는 2바이트 필드로, 처음 두 비트는 1로 설정되고 나머지 14비트는 메시지 시작 부분으로부터의 오프셋을 나타냅니다.

이러한 방식을 사용하면 DNS 메시지에서 도메인 이름을 효율적으로 표현하고 메시지 크기를 최소화할 수 있습니다.

9. HTTP 메시지 헤더의 종류

- 14장
- 예상 출제 방식 : 모르겠음. 굳이 출제한다면 이미지를 주고 요청인지 응답인지 선택.

▼ 새로 추가

- HTTP

WWW에서 데이터에 접근하기 위해서 사용. TCP사용하지만 연결 X. 클라이언트, 서버 간 양방향 전송.

- 메서드

요청 메시지	설명
GET	클라이언트가 서버로부터 문서를 읽어 올 때 사용.
HEAD	클라이언트가 문서 자체가 아닌 문서의 어떤 정보를 원할 때.
POST	클라이언트가 서버에게 어떤 정보를 제공할 때.
PUT	클라이언트가 저장할 새 문서 혹은 교체할 문서를 서버에 제공.
PATCH	PUT과 비슷하나 기존 파일에서 수정되어야 할 내용만 포함.
COPY	파일을 다른 위치로 복사.
MOVE	파일을 다른 위치로 이동.
DELETE	서버에서 문서를 제거하기 위해 사용.
LINK	문서에 링크를 생성하거나, 문서를 다른 위치로 링크하기 위해서 사용.
UNLINK	LINK 메소드에 의해 생성된 링크를 삭제하기 위해 사용.
OPTION	클라이언트가 서버에게 사용 가능한 옵션을 질의하기 위해 사용.

응답 메시지	설명
100번대	정보에 관한 것.
200번대	성공적인 요청을 표시.
300번대	클라이언트를 다른 URL로 지정.
400번대	클라이언트 사이트에서 오류 발생 표시.
500번대	서버 사이트에서 오류 발생 표시.

- 헤더

클라이언트와 서버 사이에 추가적인 정보 교환.

헤더	요청	응답	설명
일반 헤더 (General)	O	O	메시지에 대한 일반적인 정보 제공. connection, date, MIME-version, encoding, via
요청 헤더 (Request)	O	X	클라이언트의 구성과 클라이언트가 선호하는 문서 형식을 명기.
응답 헤더 (Response)	X	O	서버의 구성과 요청에 대한 특별한 정보 명기
엔티티 헤더 (Entity)	O	O	주로 응답 메시지에 존재. 문서의 본문에 대한 정보 제공. PUT 이나 POST 메소드 같이 본문을 포함하는 요청 메시지도 사용.

▼ GPT

HTTP (Hypertext Transfer Protocol) 메시지는 클라이언트(보통 웹 브라우저)와 서버 간의 요청 및 응답을 나타내며, 이러한 메시지는 여러 가지 헤더를 포함할 수 있습니다. 이 헤더들은 메시지의 여러 측면을 설명하고 제어하는 데 사용되며, 주요 헤더 유형에는 다음이 포함됩니다:

1. ****일반 헤더(General Headers)****: 클라이언트와 서버 모두에서 사용되며, 메시지 전반에 걸쳐 적용되는 헤더입니다. 예를 들어 `Date`, `Cache-Control`, `Connection` 등이 있습니다.
2. ****요청 헤더(Request Headers)****: 클라이언트에서 서버로의 요청에 특화된 헤더로, 요청 메시지의 세부 정보를 제공합니다. 예를 들어 `User-Agent`, `Accept`, `Accept-Language`, `Host` 등이 있습니다.
3. ****응답 헤더(Response Headers)****: 서버에서 클라이언트로의 응답에 특화된 헤더로, 응답 메시지의 세부 정보를 제공합니다. 예를 들어 `Server`, `WWW-Authenticate`, `Set-Cookie`, `Location` 등이 있습니다.

4. ****항목 헤더(Entity Headers)****: HTTP 메시지의 본문(또는 "엔터티")에 적용되는 헤더로, 본문의 콘텐츠와 관련된 정보를 제공합니다. 예를 들어 `Content-Type`, `Content-Length`, `Content-Encoding`, `Last-Modified` 등이 있습니다.

HTTP/1.1에서 이러한 헤더들이 모두 정의되어 있지만, HTTP/2와 같은 더 최신 버전에서는 약간 다른 형태로 표현될 수도 있습니다.

10. DHCP 클라이언트와 서버 사이에 교환하는 4가지 메시지 종류와 내용과 의미

- 16장
- 예상 출제 방식 : 나올 가능성이 높음. 4가지의 종류와 내용을 서술하는 방식.

▼ 새로 추가

DHCP

1. IP부족 해결
2. 보안 : 호스트의 PC에 정보를 저장하지 않음.
 - Relay agent
 - UDP 사용. 서버는 67번, 클라이언트는 68번 well-known 포트 사용.

DHCP 종류	방향	설명
Discover (DHCPDISCOVER)	클 → 서	요청
Offer (DHCPOFFER)	클 ← 서	제안
Request (DHCPREQUEST)	클 → 서	수락
Acknowledgment (DHCPACK)	클 ← 서	할당

DHCP(Dynamic Host Configuration Protocol)는 IP 주소를 동적으로 할당받기 위한 프로토콜로, 클라이언트와 서버 사이에서 네 가지 메시지 유형이 주고받아집니다. 이들 메시지는 DHCP "사용 절차" 또는 "사용 순서"를 구성하며, 일반적으로 "DORA" 프로세스라고도 합니다.

1. ****DHCPDISCOVER****: 클라이언트가 네트워크상의 DHCP 서버에 IP 주소를 요청하는 브로드캐스트 메시지입니다. 클라이언트가 부팅될 때 전송되며, 이 메시지에는 클라이언트의 MAC 주소가 포함됩니다.

2. ****DHCP OFFER****: DHCP 서버가 DHCPDISCOVER 메시지에 응답하여 사용 가능한 IP 주소를 제안하는 메시지입니다. 이 메시지에는 제안된 IP 주소, 네트워크 마스크, DHCP 서버의 IP 주소, 레이스(Lease) 시간 (클라이언트가 IP 주소를 보유할 수 있는 시간) 등의 정보가 포함됩니다.

3. ****DHCP REQUEST****: 클라이언트가 DHCP OFFER 메시지에서 제안된 IP 주소를 수락하고 이를 서버에 알리는 메시지입니다. 이 메시지는 서버에 수락된 IP 주소를 알리며, 동시에 네트워크에 다른 DHCP 서버가 있는 경우 해당 서버에 이 정보를 알립니다.

4. ****DHCP ACK****: DHCP 서버가 DHCP REQUEST 메시지에 응답하여 IP 주소 할당을 확인하는 메시지입니다. 이 메시지는 클라이언트에게 IP 주소가 성공적으로 할당되었음을 알리며, 이에 대한 모든 필요한 설정 정보 (예: DNS 서버 주소, 게이트웨이 주소 등)를 제공합니다.

이러한 메시지 교환을 통해 클라이언트는 네트워크에 동적으로 추가될 수 있으며, 서버는 IP 주소를 효율적으로 관리하고 재활용할 수 있습니다.

▼ GPT

DHCP(Dynamic Host Configuration Protocol)는 IP 주소를 동적으로 할당받기 위한 프로토콜로, 클라이언트와 서버 사이에서 네 가지 메시지 유형이 주고받아집니다. 이들 메시지는 DHCP "사용 절차" 또는 "사용 순서"를 구성하며, 일반적으로 "DORA" 프로세스라고도 합니다.

1. ****DHCPDISCOVER****: 클라이언트가 네트워크상의 DHCP 서버에 IP 주소를 요청하는 브로드캐스트 메시지입니다. 클라이언트가 부팅될 때 전송되며, 이 메시지에는 클라이언트의 MAC 주소가 포함됩니다.

2. ****DHCP OFFER****: DHCP 서버가 DHCPDISCOVER 메시지에 응답하여 사용 가능한 IP 주소를 제안하는 메시지입니다. 이 메시지에는 제안된 IP 주소, 네트워크 마스크, DHCP 서버의 IP 주소, 레이스(Lease) 시간 (클라이언트가 IP 주소를 보유할 수 있는 시간) 등의 정보가 포함됩니다.

3. ****DHCP REQUEST****: 클라이언트가 DHCP OFFER 메시지에서 제안된 IP 주소를 수락하고 이를 서버에 알리는 메시지입니다. 이 메시지는 서버에 수락된 IP 주소를 알리며, 동시에 네트워크에 다른 DHCP 서버가 있는 경우 해당 서버에 이 정보를 알립니다.

4. ****DHCP ACK****: DHCP 서버가 DHCP REQUEST 메시지에 응답하여 IP 주소 할당을 확인하는 메시지입니다. 이 메시지는 클라이언트에게 IP 주소가 성공적으로 할당되었음을 알리며, 이에 대한 모든 필요한 설정 정보 (예: DNS 서버 주소, 게이트웨이 주소 등)를 제공합니다.

이러한 메시지 교환을 통해 클라이언트는 네트워크에 동적으로 추가될 수 있으며, 서버는 IP 주소를 효율적으로 관리하고 재활용할 수 있습니다.