Ch.2 정리

1. 네트워크의 기초

→ **네트워크란**

- 노드(서버, 라우터, 스위치 등 네트워크 장치)와 링크(유선 또는 무선)가 서로 연결되어 있으며 리소스를 공유하는 집합

→ **처리량**

- 링크를 통해 전달되는 단위 시간당 데이터양
- 트래픽, 네트워크 장치 간의 대역폭, 네트워크 중간에 발생하는 에러, 장치의 하드웨어 스펙에 영향을 받

→ **지연시간**

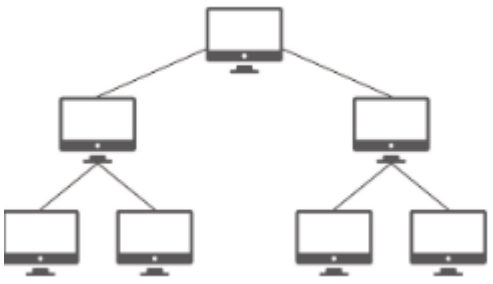
- 요청이 처리되는 시간을 말하며 어떤 메시지가 두 장치 사이를 왕복하는 데 걸린 시간
- 매체 타입(무선, 유선), 패킷 크기, 라우터의 패킷 처리 시간에 영향을 받음

→ **네트워크 토폴리지**

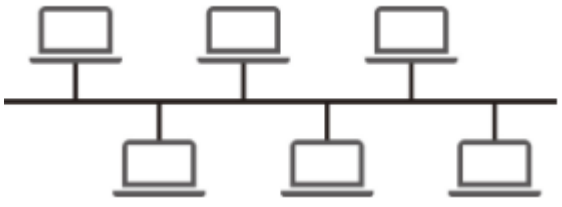
- 노드와 링크가 어떻게 배치되어 있는지에 대한 방식이자 연결 형태

→ **토폴리지 종류**

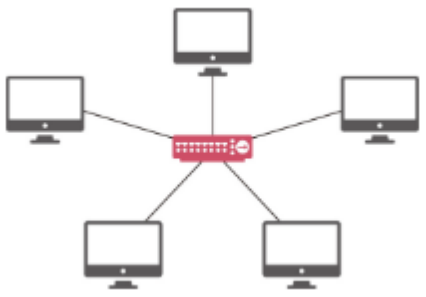
- 트리 토폴로지



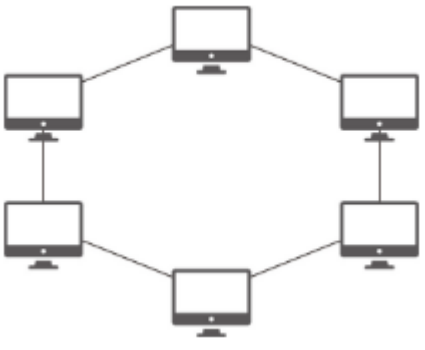
- 버스 토폴로지



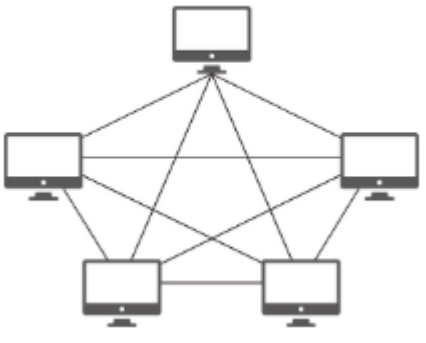
- 스타 토폴리지



- 링형 토폴리지



- 메시 토폴로지(망형 토폴로지)



→ **병목현상**

- 네트워크의 특정 지점에서 데이터 흐름이 제한되어 전체 네트워크 성능이 저하되는 현상
- 주된 원인 : 네트워크 대역, 네트워크 토폴로지, 서버 CPU, 메모리 사용량, 비효율적인 네트워크 구성

→ **네트워크 분류**

- LAN - 근거리 통신망을 의미하며 같은 건물이나 캠퍼스 같은 좁은 공간에서 운영
- MAN - 대도시 지역 네트워크를 나타내며 도시 같은 넓은 지역에서 운영
- WAN - 광역 네트워크를 의미하며 국가 또는 대륙 같은 더 넓은 지역에서 운영

→ **네트워크 성능 분석 명령어**

- ping - 네트워크 상태를 확인하려는 대상 노드를 향해 일정 크기의 패킷을 전송하는 명령어

CMD Test

```
C:\Users\ifork>ping www.google.com -n 12

Ping www.google.com [172.217.25.164] 32바이트 데이터 사용:
172.217.25.164의 응답: 바이트=32 시간=39ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=102ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=39ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=41ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=44ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=37ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=40ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=46ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=41ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=46ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=42ms TTL=115
172.217.25.164의 응답: 바이트=32 시간=47ms TTL=115

172.217.25.164에 대한 Ping 통계:
    패킷: 보냄 = 12, 받음 = 12, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
    최소 = 37ms, 최대 = 102ms, 평균 = 47ms
```

- netstat - 접속되어 있는 서비스들의 네트워크 상태를 표시하는 데 사용되며 네트워크 접속, 라우팅 테이블, 네트워크 프로토콜 등 리스트를 보여줌

CMD Test

```
C:\Users\ifork>netstat

활성 연결

  프로토콜  로컬 주소          외부 주소              상태
  TCP    127.0.0.1:5354         kubernetes:49669       ESTABLISHED
  TCP    127.0.0.1:5354         kubernetes:49671       ESTABLISHED
  TCP    127.0.0.1:49669        kubernetes:5354        ESTABLISHED
  TCP    127.0.0.1:49671        kubernetes:5354        ESTABLISHED
  TCP    127.0.0.1:49672        kubernetes:49673       ESTABLISHED
  TCP    127.0.0.1:49673        kubernetes:49672       ESTABLISHED
  TCP    127.0.0.1:49674        kubernetes:49675       ESTABLISHED
  TCP    127.0.0.1:49675        kubernetes:49674       ESTABLISHED
  TCP    127.0.0.1:50582        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50583        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50585        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50587        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50588        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50589        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50590        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50591        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50592        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50596        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50600        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50602        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50604        kubernetes:49350       TIME_WAIT
  TCP    127.0.0.1:50609        kubernetes:49350       TIME_WAIT
```

- nslookup - DNS에 관련된 내용을 확인하기 위해 쓰는 명령어

CMD Test

```
C:\Users\ifork>nslookup
기본 서버:  bns1.hananet.net
Address:  210.220.163.82

> google.com
서버:    bns1.hananet.net
Address:  210.220.163.82

권한 없는 응답:
이름:    google.com
Addresses:  2404:6800:400a:813::200e
          142.250.76.142
```

- tracert - 목적지 노드까지 네트워크 경로를 확인할 때 사용하는 명령어

CMD Test

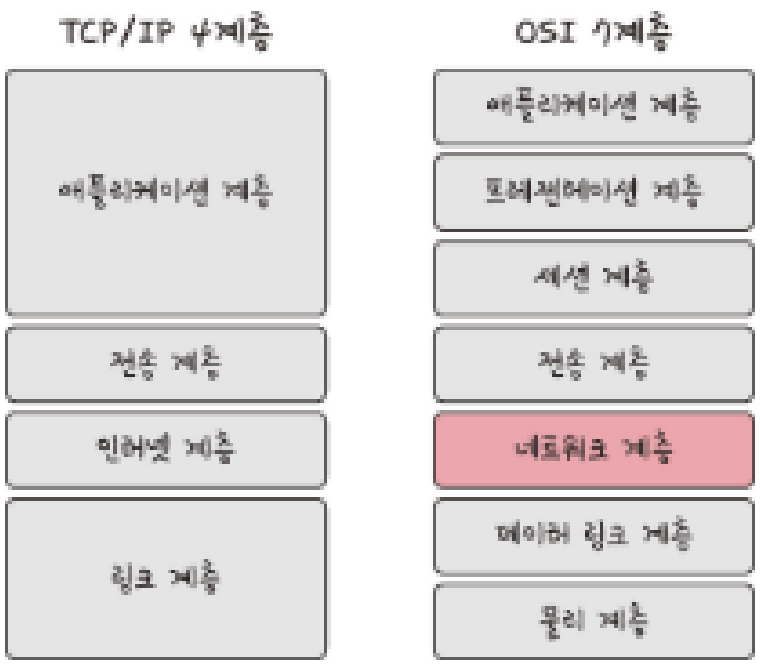
```
C:\Users\ifork>tracert www.google.com

최대 30홉 이상의
www.google.com [142.250.206.196](으)로 가는 경로 추적:

  1     2 ms     2 ms     2 ms  192.168.0.1
  2    21 ms     5 ms    15 ms  175.119.193.1
  3     8 ms     2 ms     4 ms  100.71.37.129
  4    52 ms     7 ms     5 ms  100.71.38.49
  5     3 ms     3 ms    33 ms  10.44.248.28
  6     3 ms     3 ms    49 ms  10.222.18.140
  7    48 ms    28 ms    57 ms  10.222.22.111
  8     *        *        *     요청 시간이 만료되었습니다.
  9    33 ms    34 ms    34 ms  142.251.52.31
 10   169 ms    59 ms    36 ms  108.170.242.102
 11     *       67 ms     *     209.85.246.83
 12    76 ms    51 ms    40 ms  142.250.229.250
 13    38 ms    56 ms    45 ms  192.178.108.227
```

→ **네트워크 프로토콜 표준화**

- 다른 장치들끼리 데이터를 주고받기 위해 설정된 공통된 인터페이스
- IEEE 또는 IETF라는 표준화 단체가 정함

2. TCP/IP 4계층 모델

→ **계층 구조**



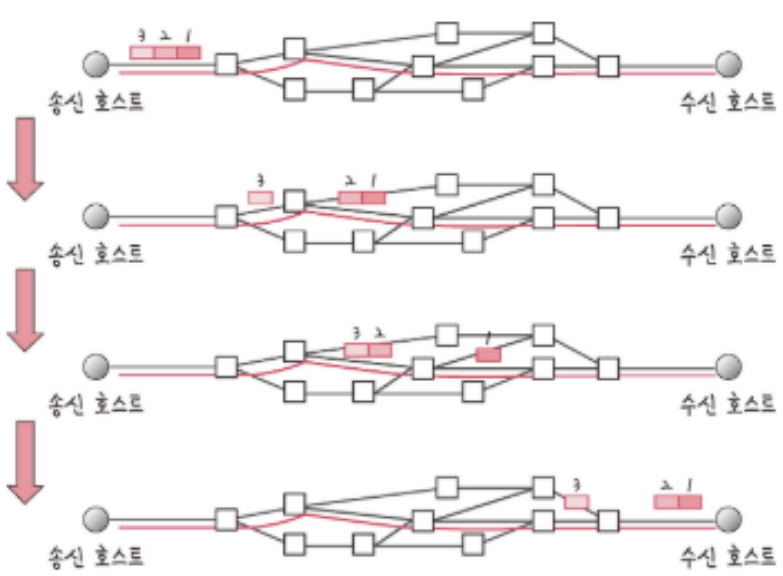
→ **애플리케이션 계층**

- 응용 프로그램이 사용되는 프로토콜 계층이며 웹 서비스, 이메일 등 서비스를 실질적으로 사람들에게 제공하는 층
- FTP(장치와 장치 간의 파일을 전송하는 데 사용되는 표준 통신 프로토콜)
- HTTP(World Wide Web을 위한 데이터 통신의 기초이자 웹 사이트를 이용하는 데 쓰는 프로토콜)
- SSH(보안되지 않은 네트워크에서 네트워크 서비스를 안전하게 운영하기 위한 암호화 네트워크 프로토콜)
- SMTP(전자 메일 전송을 위한 인터넷 표준 통신 프로토콜)
- DNS(도메인 이름과 IP 주소를 매핑해주는 서버)

→ **전송 계층**

- 송신자와 수신자를 연결하는 통신 서비스를 제공
- 연결 지향 데이터 스트림 지원, 신뢰성, 흐름 제어를 제공
- 애플리케이션과 인터넷 계층 사이의 데이터가 전달될 때 중계 역할
- TCP - 패킷 사이의 순서를 보장하고 연결지향 프로토콜을 사용해서 연결을 하여 신뢰성을 구축해서 수신 여부를 확인하며 '가상회선 패킷 교환 방식'을 사용
- UDP - 순서를 보장하지 않고 수신 여부를 확인하지 않으며 단순히 데이터만 주는 '데이터그램 패킷 교환 방식'을 사용

→ **가상회선 패킷 교환 방식**

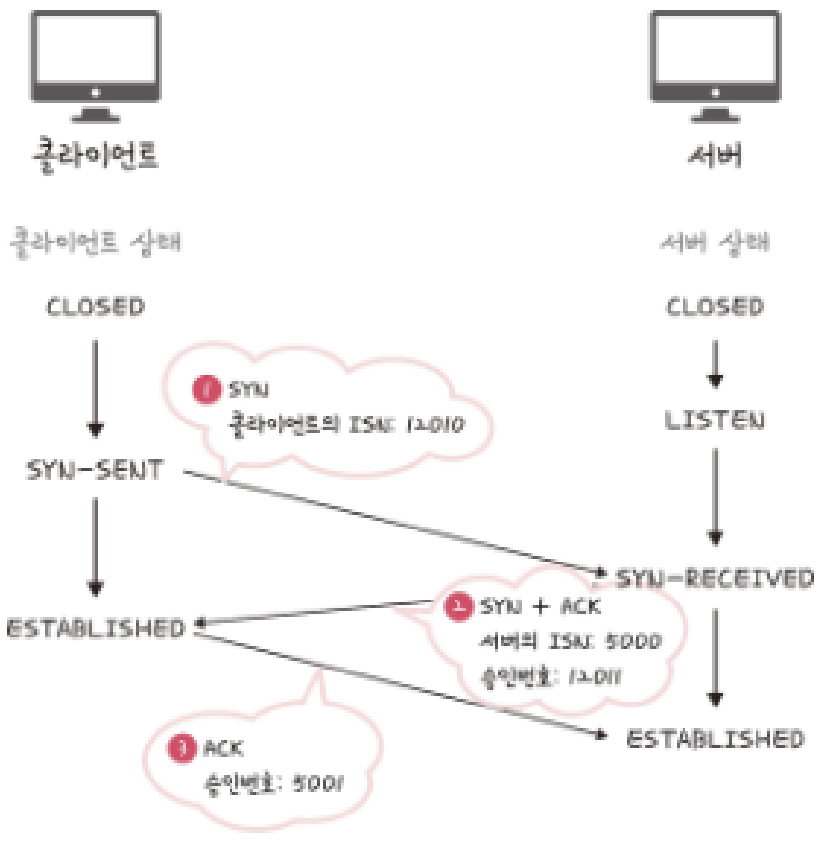


v순서대로 3,2,1도착

→ **데이터그램 패킷 교환 방식**

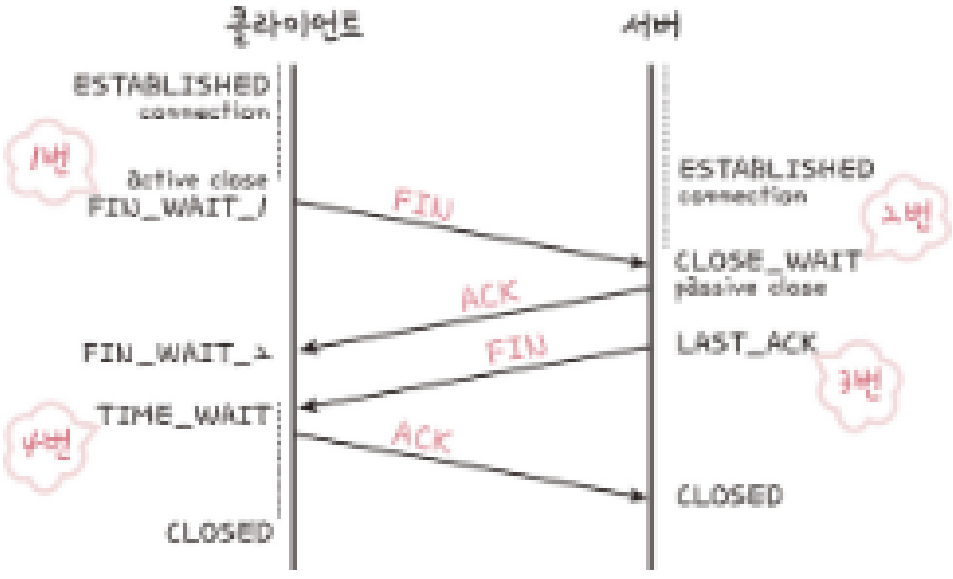
v순서상관없이 2,3,1도착

→ **3-웨이 핸드셰이크**



- SYN - SYNchronization의 약자, 연결 요청 플래그
- ACK - ACKnowledgement의 약자, 응답 플래그
- ISN - Initial Sequence Numbers의 약어, 초기 네트워크 연결을 할 때 할당된 32비트 고유 시퀀스 번호

→ **4-웨이 핸드셰이크**



→ **인터넷 계층**

- 장치로부터 받은 네트워크 패킷을 IP 주소로 지정된 목적지로 전송하기 위해 사용되는 계층
- IP - 인터넷 프로토콜로, 데이터 패킷을 목적지 주소로 전달하는 역할
- ARP - IP 주소를 물리적 네트워크 주소(MAC 주소)로 변환하는 프로토콜
- ICMP - 네트워크 상태를 진단하고 오류 메시지를 전달하는 프로토콜
- IGMP - IP 네트워크에서 멀티캐스트 그룹 멤버십을 관리하기 위해 사용되는 프로토콜

→ **링크 계층**

- 전선, 광섬유, 무선 등으로 실질적으로 데이터를 전달
- 장치 간에 신호를 주고받는 '규칙'을 정하는 계층

→ **전이중화 통신**

- 양쪽 장치가 동시에 송수신할 수 있는 방식

→ **반이중화 통신**

- 양쪽 장치는 서로 통신할 수 있지만, 동시에는 통신할 수 없으며 한 번에 한 방향만 통신할 수 있는 방식

→ **CSMA/CD**

- 데이터를 '보낸 이후' 충돌이 발생한다면 일정 시간 이후 재전송하는 방식

→ **CSMA/CA**

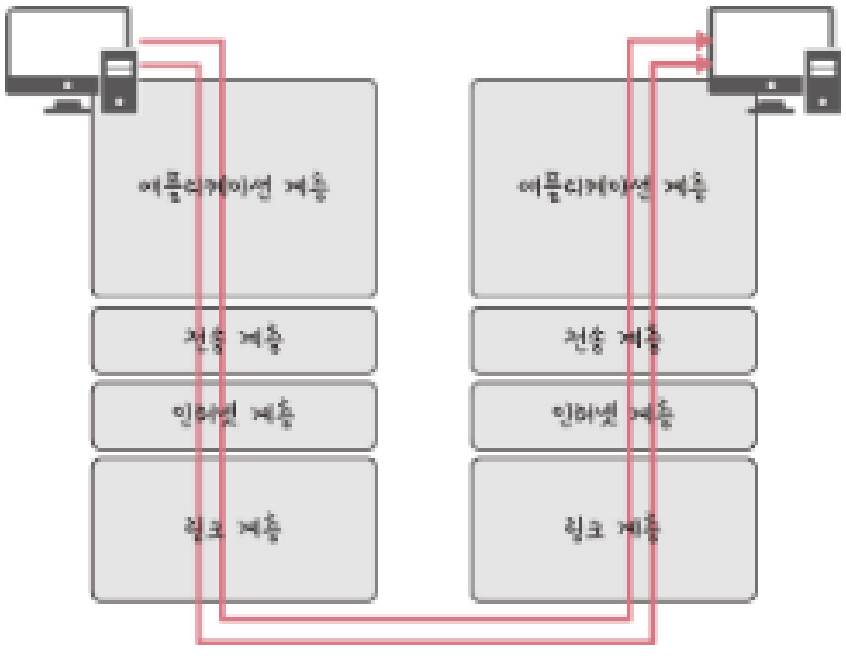
- 장치에서 데이터를 보내기 전에 캐리어 감지 등으로 사전에 가능한 한 충돌을 방지하는 방식

→ **와이파이**

- 전자기기들이 무선 LAN 신호에 연결할 수 있게 하는 기술

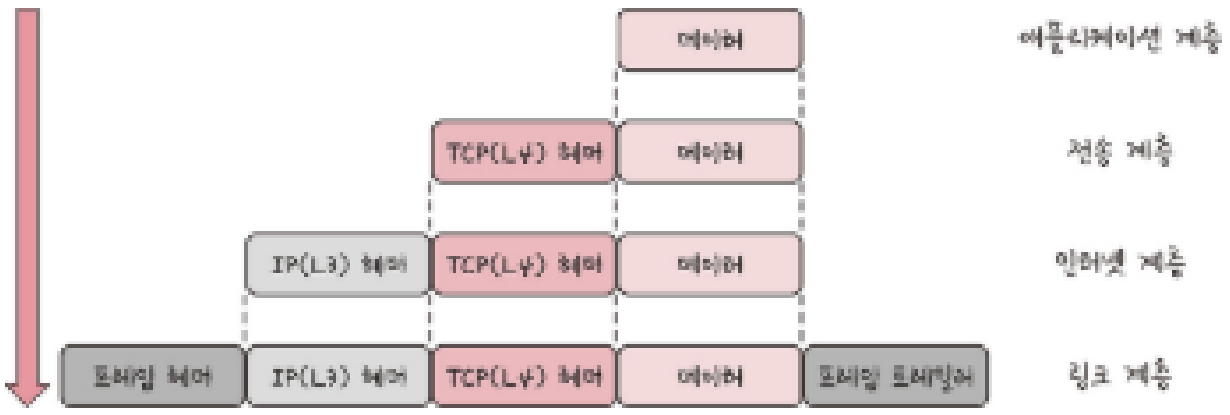
→ **이더넷 프레임**

→ **계층 간 데이터 송수신 과정**



→ **캡슐화 과정**

- 해당 계층의 헤더를 삽입하는 과정



3. 네트워크 기기

→ **라우터**

- 여러 개의 네트워크를 연결, 분할, 구분시켜주는 역할

→ **L7 스위치**

- 로드밸런서라고도 하며, 서버의 부하를 분산하는 기기
- 클라이언트로부터 오는 요청들을 뒤쪽의 여러 서버로 나누는 역할을 하며 시스템이 처리할 수 있는 트래픽 증가가 목적

→ **L3 스위치**

- 라우터의 하드웨어 기반의 라우팅을 담당하는 장치

→ **L2 스위치**

- 장치들의 MAC 주소를 MAC 주소 테이블을 통해 관리하며, 연결된 장치로부터 패킷이 왔을 때 패킷 전송

→ **브리지**

- 두 개의 근거리 통신망(LAN)을 상호 접속할 수 있도록 하는 통신망 연결 장치
- 통신망 범위를 확장하고 서로 다른 LAN 등으로 이루어진 '하나의' 통신망을 구축할 때 쓰임

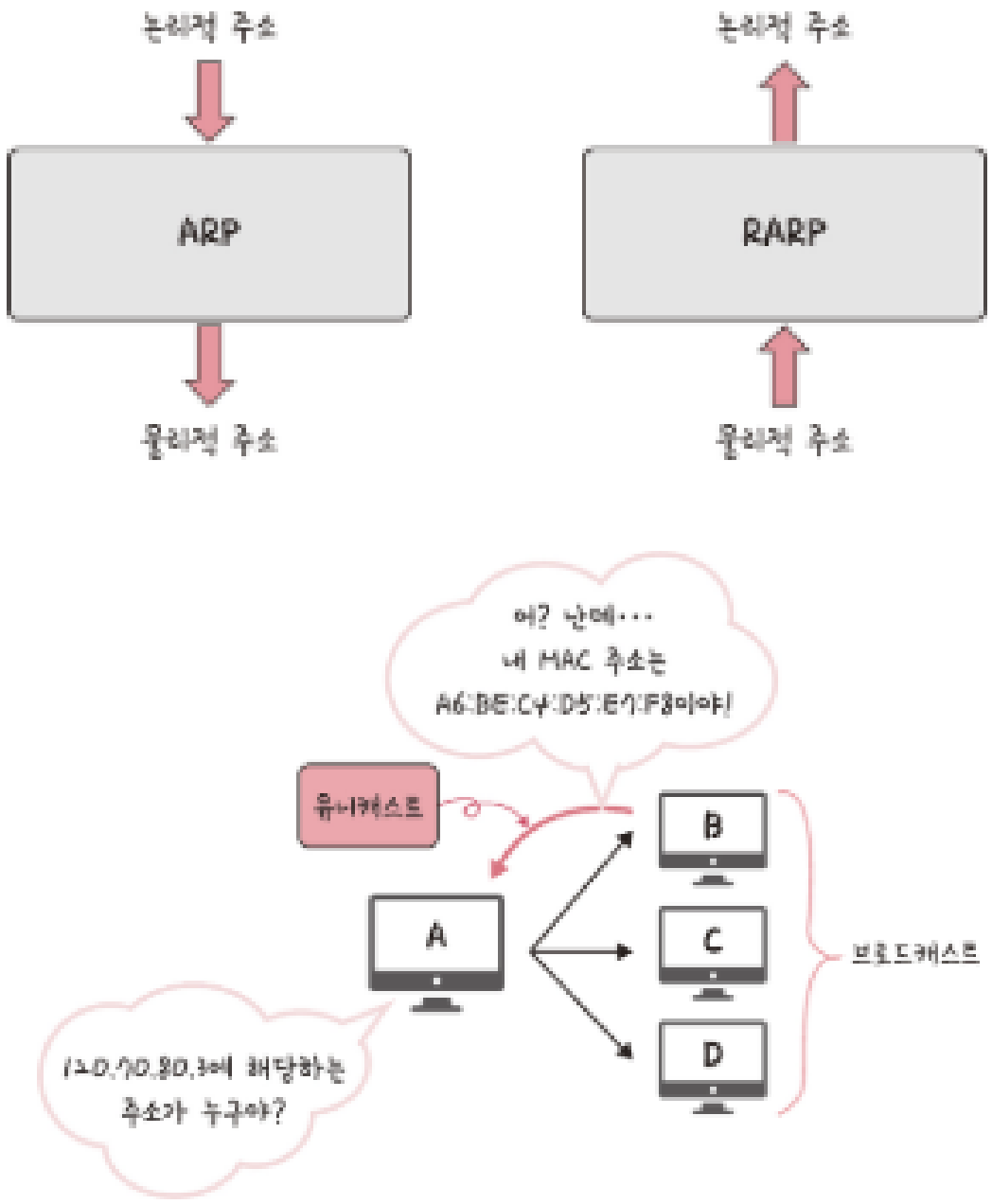
→ **리피터**

- 들어오는 약해진 신호 정도를 증폭하여 다른 쪽으로 전달하는 장치

4. IP 주소

→ **ARP**

- IP 주소로부터 MAC 주소를 구하는 IP와 MAC 주소의 다리 역할을 하는 프로토콜



→ **홉바이홉 통신**

- IP 주소를 통해 통신하는 과정

→ **라우팅 테이블**

- 송신지에서 수신지까지 도달하기 위해 사용되며 라우터에 들어가 있는 목적지 정보들과 그 목적지로 가기 위한 방법이 들어 있는 리스트

→ **게이트웨이**

- 서로 다른 통신망, 프로토콜을 사용하는 네트워크 간의 통신을 가능하게 하는 관문 역할하는 소프트웨어 또는 컴퓨터

→ **IP 주소 체계**

- IPv4 - 32비트를 8비트 단위로 점을 찍어 표기
- IPv6 - 64비트를 16비트 단위로 점을 찍어 표기

→ **클래스 기반 할당 방식**

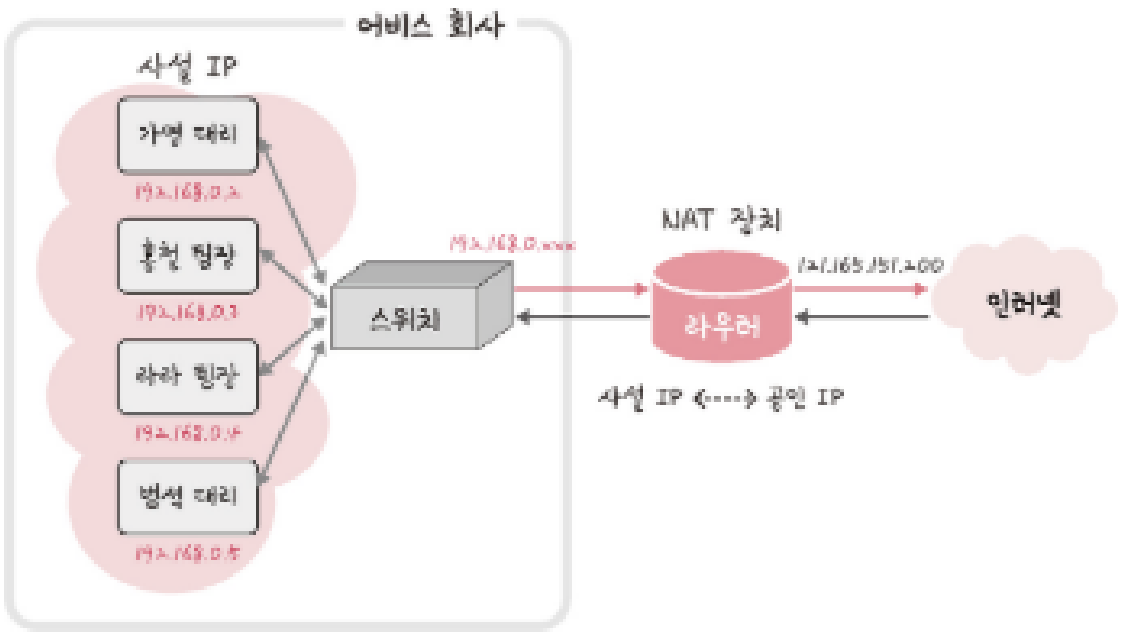
- 클래스 A·B·C는 일대일 통신으로 사용되고 클래스 D는 멀티캐스트 통신, 클래스 E는 앞으로 사용할 예비용으로 쓰는 방식

→ **DHCP**

- IP 주소 및 기타 통신 매개변수를 자동으로 할당하기 위한 네트워크 관리 프로토콜
- 네트워크 장치의 IP 주소를 수동으로 설정할 필요 없이 인터넷에 접속할 때마다 자동으로 IP 주소를 할당

→ **NAT**

- 패킷이 라우팅 장치를 통해 전송되는 동안 패킷의 IP 주소 정보를 수정하여 IP 주소를 다른 주소로 매핑하는 방법



v홍철 팀장과 가영 대리는 하나의 IP

- 단점 : NAT는 여러 명이 동시에 인터넷을 접속하게 되므로 실제로 접속하는 호스트 숫자에 따라서 접속 속도가 느려질 수 있음

5. HTTP

→ **HTTP**

- 전송 계층 위에 있는 애플리케이션 계층
- 웹 서비스 통신에 사용

→ **RTT**

- 패킷이 목적지에 도달하고 나서 다시 출발지로 돌아오기까지 걸리는 시간이며 패킷 왕복 시간

→ **HTTP/1.0**

- 기본적으로 한 연결당 하나의 요청을 처리
- 서버로부터 파일을 가져올 때마다 TCP의 3-웨이 핸드셰이크를 계속해서 열어야 하기 때문에 RTT가 증가하는 단점
- 해결 방법 : 이미지 스플리팅, 코드 압축, 이미지 Base64 인코딩

→ **HTTP/1.1**

- 매번 TCP 연결을 하는 것이 아니라 한 번 TCP 초기화를 한 이후에 keep-alive라는 옵션으로 여러 개의 파일을 송수신할 수 있음
- 단점 : 문서 안에 포함된 다수의 리소스(이미지, css 파일, script 파일)를 처리하려면 요청할 리소스 개수에 비례해서 대기 시간이 길어짐 (HOL Blocking) + 무거운 헤더 구조

→ **HTTP/2**

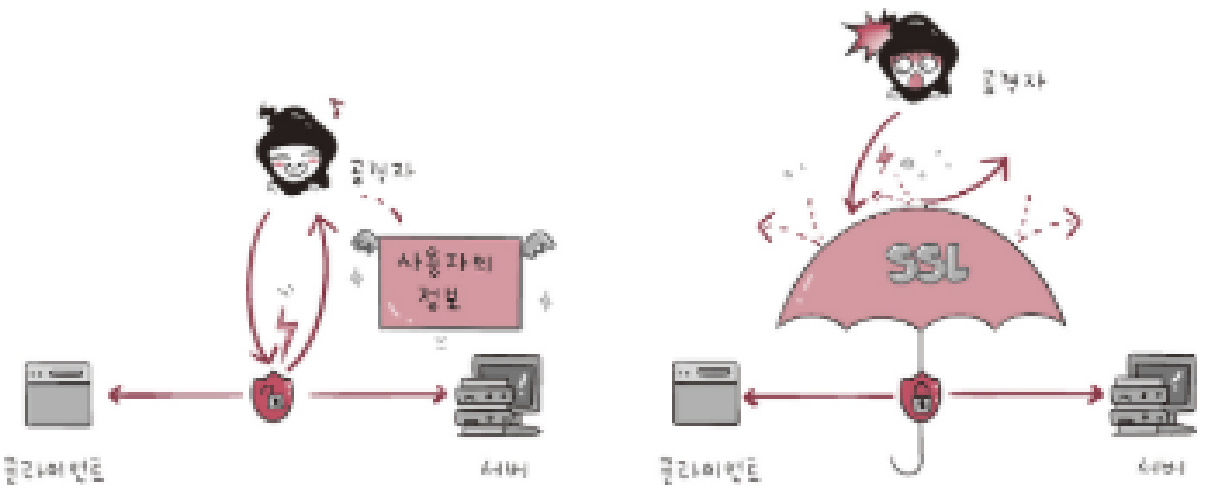
- HTTP/1.x보다 지연 시간을 줄이고 응답 시간을 더 빠르게 할 수 있으며 멀티플렉싱, 헤더 압축, 서버 푸시, 요청의 우선순위 처리를 지원하는 프로토콜
- 1.1의 단점 해결을 위한 병렬적인 스트림(stream) + 헤더 압축(허프만 코딩)

→ **HTTPS**

- 애플리케이션 계층과 전송 계층 사이에 신뢰 계층인 SSL/TLS 계층을 넣은 신뢰할 수 있는 HTTP 요청 → 통신 암호

→ **SSL/TLS**

- 전송 계층에서 보안을 제공하는 프로토콜
- 클라이언트와 서버가 통신할 때 SSL/TLS를 통해 제3자가 메시지를 도청하거나 변조하지 못하도록 함
- 보안 세션을 기반으로 데이터를 암호화하며 보안 세션이 만들어질 때 인증 메커니즘, 키 교환 암호화 알고리즘, 해싱 알고리즘이 사용



v인터셉터 방지

→ **보안 세션**

- 보안이 시작되고 끝나는 동안 유지되는 세션
- SSL/TLS는 핸드셰이크를 통해 보안 세션을 생성하고 이를 기반으로 상태 정보 등을 공유

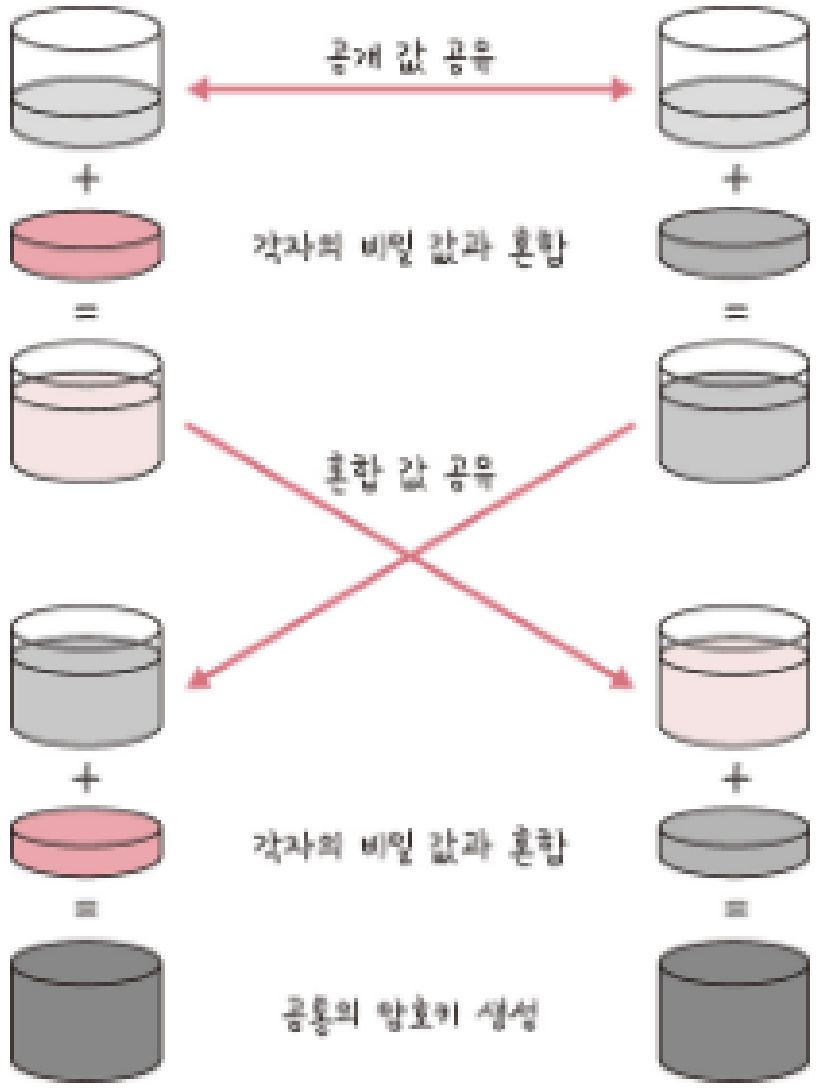
→ **인증 메커니즘**

- CA(Certificate Authorities)에서 발급한 인증서를 기반으로 이루어짐
- CA에서 발급한 인증서는 안전한 연결을 시작하는 데 있어 필요한 '공개키'를 클라이언트에 제공하고 사용자가 접속한 '서버가 신뢰'할 수 있는 서버임을 보장함
- 신뢰성이 엄격하게 공인된 기업들만 참여할 수 있으며, 대표적인 기업으로는 Comodo, GoDaddy, GlobalSign, 아마존 등이 있음

→ **암호화 알고리즘**

- 키 교환 암호화 알고리즘으로는 대수곡선 기반의 ECDHE(Elliptic Curve Diffie-Hellman Ephermeral) 또는 모듈식 기반의 DHE(Diffie-Hellman Ephermeral)를 사용합니다. 둘 다 디피-헬만(Diffie-Hellman) 방식을 근간으로 만들어졌습니다.

→ **디피-헬만 키 교환 암호화 알고리즘**



v처음에 공개 값을 공유하고 각자의 비밀 값과 혼합한 후 혼합 값을 공유 → 각자의 비밀 값과 또 혼합 → 공통의 암호키가 생성

→ **해싱 알고리즘**

- 해싱 알고리즘은 데이터를 추정하기 힘든 더 작고, 섞여 있는 조각으로 만드는 알고리즘
- SSL/TLS는 해싱 알고리즘으로 SHA-256 알고리즘과 SHA-384 알고리즘을 사용

→ **SHA-256 알고리즘**

- 해시 함수의 결괏값이 256비트인 알고리즘
- 해싱을 해야 할 메시지에 1을 추가하는 등 전처리를 하고 전처리된 메시지를 기반으로 해시를 반환
- 해시 - 다양한 길이를 가진 데이터를 고정된 길이를 가진 데이터로 매핑(mapping)한 값
- 해싱 - 데이터를 해시로 바꿔주는 일이며 해시 함수가 이를 담당
- 해시 함수 - 임의의 데이터를 입력으로 받아 일정한 길이의 데이터로 바꿔주는 함수

→ **SEO**

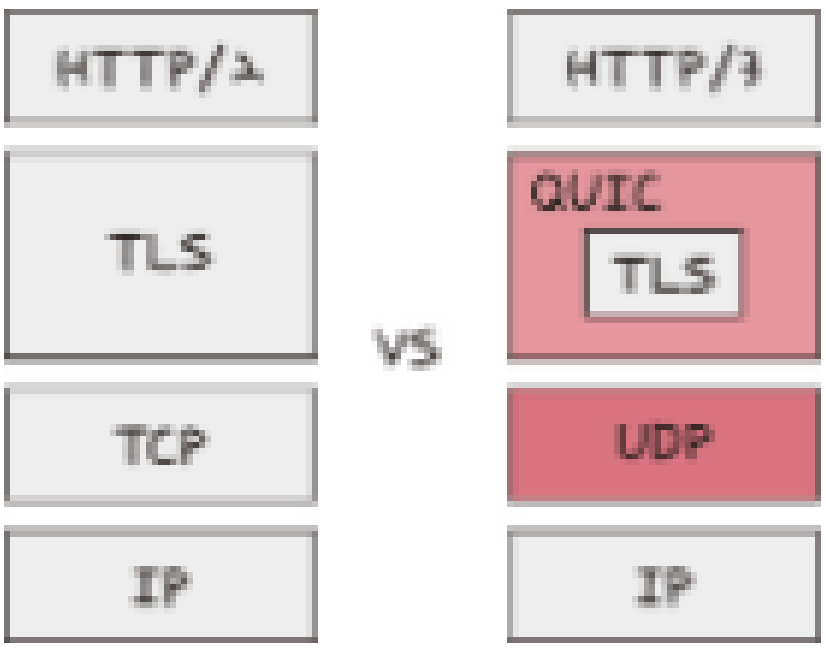
- 검색엔진 최적화
- 사용자들이 구글, 네이버 같은 검색엔진으로 웹 사이트를 검색했을 때 그 결과를 페이지 상단에 노출시켜 많은 사람이 볼 수 있도록 최적화하는 방법
- SEO 개선의 다양한 방법 : 캐노니컬 설정, 메타 설정, 페이지 속도 개선, 사이트맵 관리

→ **HTTPS 구축 방법**

1. 직접 CA에서 구매한 인증키를 기반으로 HTTPS 서비스를 구축
2. 서버 앞단의 HTTPS를 제공하는 로드밸런서를 둠
3. 서버 앞단에 HTTPS를 제공하는 CDN을 둬서 구축

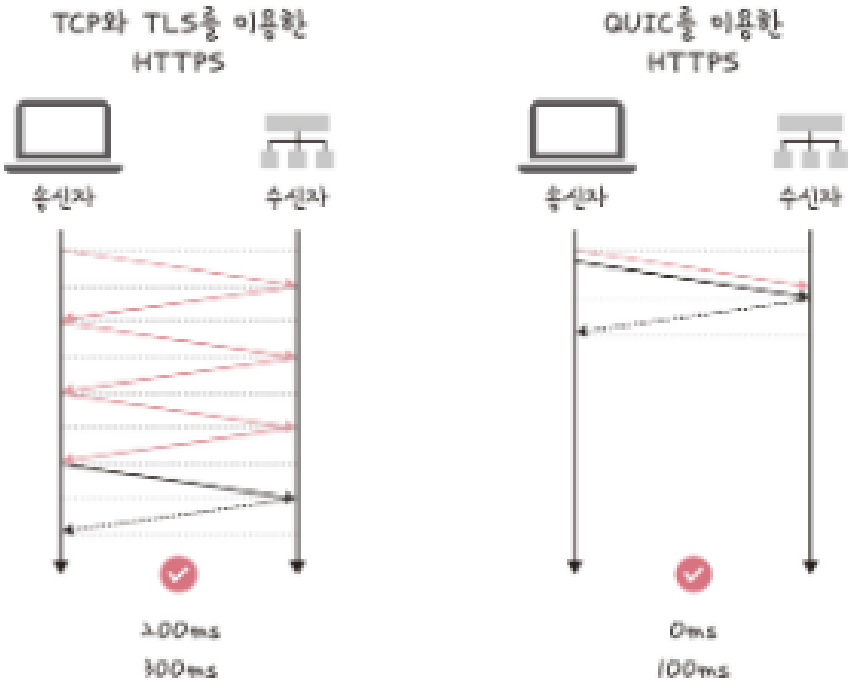
→ **HTTP/3**

- TCP 위에서 돌아가는 HTTP/2와는 달리 HTTP/3은 QUIC이라는 계층 위에서 돌아가며, TCP 기반이 아닌 UDP 기반으로 돌아감
- 멀티플렉싱을 가지고 있으며 초기 연결 설정 시 지연 시간 감소



→ **HTTP/2와 HTTP/3의 RTT비교**

- 통신을 시작할 때 번거로운 3-웨이 핸드셰이크 과정을 거치지 않아도 됨


TCP와 TLS를 이용한
HTTPS
송신자
수신자
300ms
300ms
QUIC를 이용한
HTTPS
송신자
수신자
0ms
100ms