Computer Network (컴퓨터 네트워크)

사전 발표 준비 및 전체적인 이론 학습

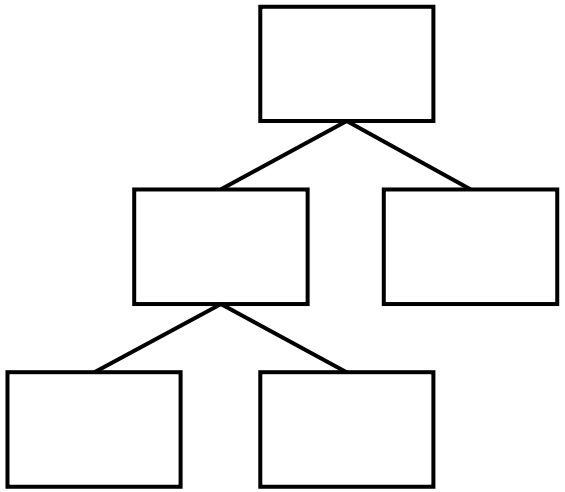
컴퓨터 네트워크

컴퓨터 네트워크

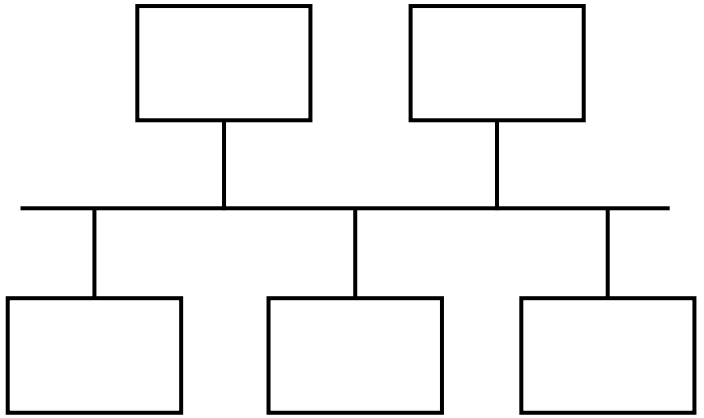
: Node(장치) 사이를 연결하는 Link(연결)의 조합 → 리소스 공유

Network topology : Node와 Link의 조합(연결) 형태

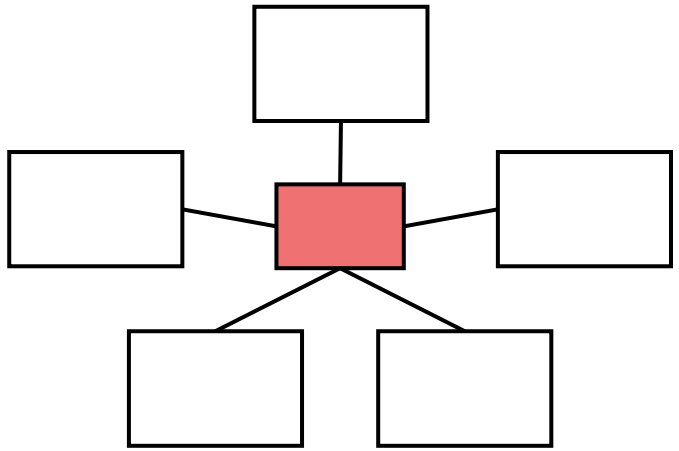
트리 (Tree)



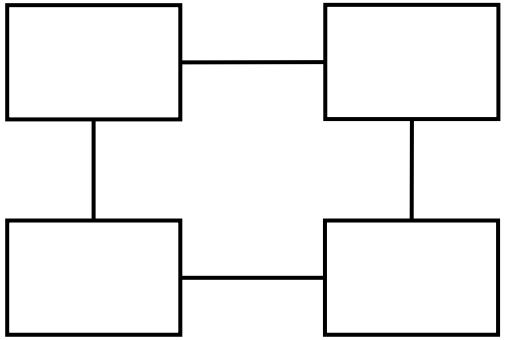
버스 (Bus)



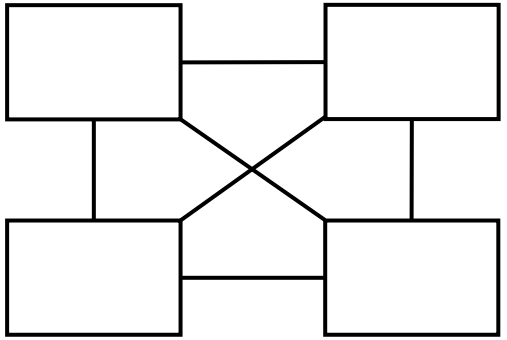
스타 (Star)



링 (Ring)



메시 (Mesh)



Network topology 가 중요한 이유 : 안정성

- Single Point of Failure (SPOF) : 단일 장애점
- BottleNeck : 병목현상

Tree	; 계층형 토폴로지
Bus	: 중앙 통신 회선 하나에 여러 노드가 연결 → ( ) 문제
Star	: 중앙 장치에 모두 연결
Ring	: 각 장치가 양 옆 장치와 연결
Mesh	; 망형 토폴로지 : 그물망처럼 연결

Network 분류

양경식

LAN [Local Area Network]

: 근거리 통신망

- 전송 속도 빠름, 혼잡도 낮음

MAN [Metropolitan Area Network]

: 대도시 지역 네트워크

- 전송속도 평균, 혼잡도 평균

WAN [Wide Area Network]

: 광역 네트워크

- 전송속도 느림, 혼잡도 높음

[chatgpt, DALL-E]

> ping : 네트워크 상태를 확인하려는 노드로 패킷 전송

```
C:\Users\Gaeng>ping www.google.com

Ping www.google.com [172.217.161.196] 32바이트 데이터 사용:
172.217.161.196의 응답: 바이트=32 시간=37ms TTL=113
172.217.161.196의 응답: 바이트=32 시간=37ms TTL=113
172.217.161.196의 응답: 바이트=32 시간=38ms TTL=113
172.217.161.196의 응답: 바이트=32 시간=37ms TTL=113

172.217.161.196에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
    최소 = 37ms, 최대 = 38ms, 평균 = 37ms
```

> netstat : 현재 접속되는 서비스의 네트워크 상태 표시

```
C:\Users\Gaeng>netstat

활성 연결

  프로토콜  로컬 주소          외부 주소              상태
  TCP    127.0.0.1:9010         kubernetes:54895       ESTABLISHED
  TCP    127.0.0.1:9100         kubernetes:54896       ESTABLISHED
  TCP    127.0.0.1:51311        kubernetes:54856       ESTABLISHED
  TCP    127.0.0.1:54856        kubernetes:51311       ESTABLISHED
  TCP    127.0.0.1:54895        kubernetes:9010        ESTABLISHED
  TCP    127.0.0.1:54896        kubernetes:9100        ESTABLISHED
  TCP    127.0.0.1:54974        kubernetes:54977       ESTABLISHED
  TCP    127.0.0.1:54974        kubernetes:54978       ESTABLISHED
  TCP    127.0.0.1:54974        kubernetes:55052       ESTABLISHED
  TCP    127.0.0.1:54977        kubernetes:54974       ESTABLISHED
  TCP    127.0.0.1:54978        kubernetes:54974       ESTABLISHED
  TCP    127.0.0.1:55052        kubernetes:54974       ESTABLISHED
  TCP    172.30.1.77:49585      134:https              ESTABLISHED
```

> nslookup : DNS 확인

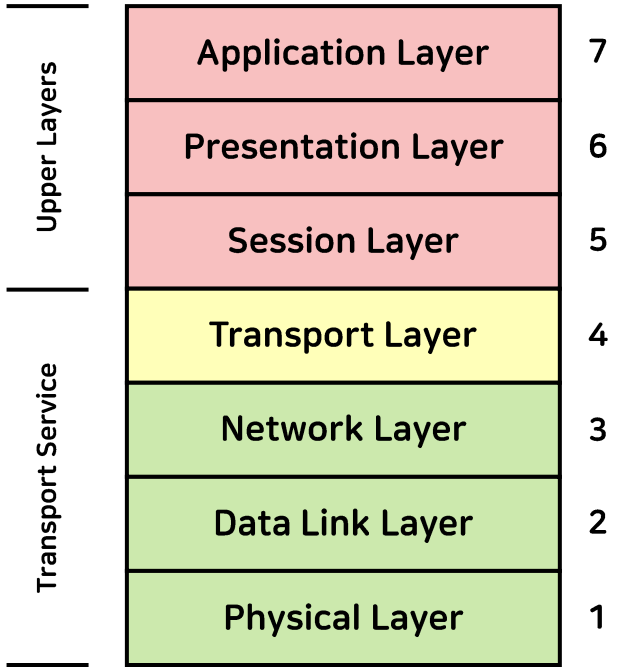
```
C:\Users\Gaeng>nslookup
기본 서버:  kns.kornet.net
Address:  168.126.63.1

> google.com
서버:    kns.kornet.net
Address:  168.126.63.1

권한 없는 응답:
이름:    google.com
Addresses:  2404:6800:400a:80e::200e
          172.217.25.174
```

> tracert : 목적지 노드까지의 네트워크 경로 확인 (Linux : traceroute)

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

최대 30홉 이상의
www.google.com [172.217.161.196](으)로 가는 경로 추적:

  1    <1 ms    <1 ms    <1 ms  172.30.1.254
  2     *        *        *     요청 시간이 만료되었습니다.
  3     1 ms     1 ms     1 ms  125.141.249.34
  4     *        *        *     요청 시간이 만료되었습니다.
  5     7 ms     7 ms     7 ms  112.174.49.121
  6     6 ms     7 ms     7 ms  112.174.84.122
  7    31 ms    30 ms    31 ms  72.14.242.8
  8    32 ms    32 ms    32 ms  108.170.248.249
  9    32 ms    32 ms    32 ms  108.170.252.16
 10    33 ms    33 ms    32 ms  209.85.241.107
 11    38 ms    38 ms    38 ms  142.250.58.92
 12    39 ms    39 ms    39 ms  192.178.108.233
 13    38 ms    39 ms    39 ms  108.170.235.5
 14    38 ms    37 ms    37 ms  kix07s03-in-f4.1e100.net [172.217.161.196]

추적을 완료했습니다.
```

OSI 7 Layer

김혜란



Application Layer
: 프로토콜에 따른 메시지 포맷을 정하고 Human - Machine Interface 역할을 함

Presentation layer
: 메세지를 1과 0으로 코딩, encryption, compression

Session Layer
: 자격 증명 및 허가 처리, session restoration

Transport Layer
: 에러 검출 및 복구 및 흐름 제어를 end-to-end로 진행함

Network Layer
: 라우터(Router)를 통해 경로를 선택하고 주소를 정하고(IP) 경로(Route)에 따라 패킷을 전달

Data Link Layer
: 에러 검출 및 복구 및 흐름 제어를 Physical Layer 위에서 진행함
: Point to Point
: 스위치

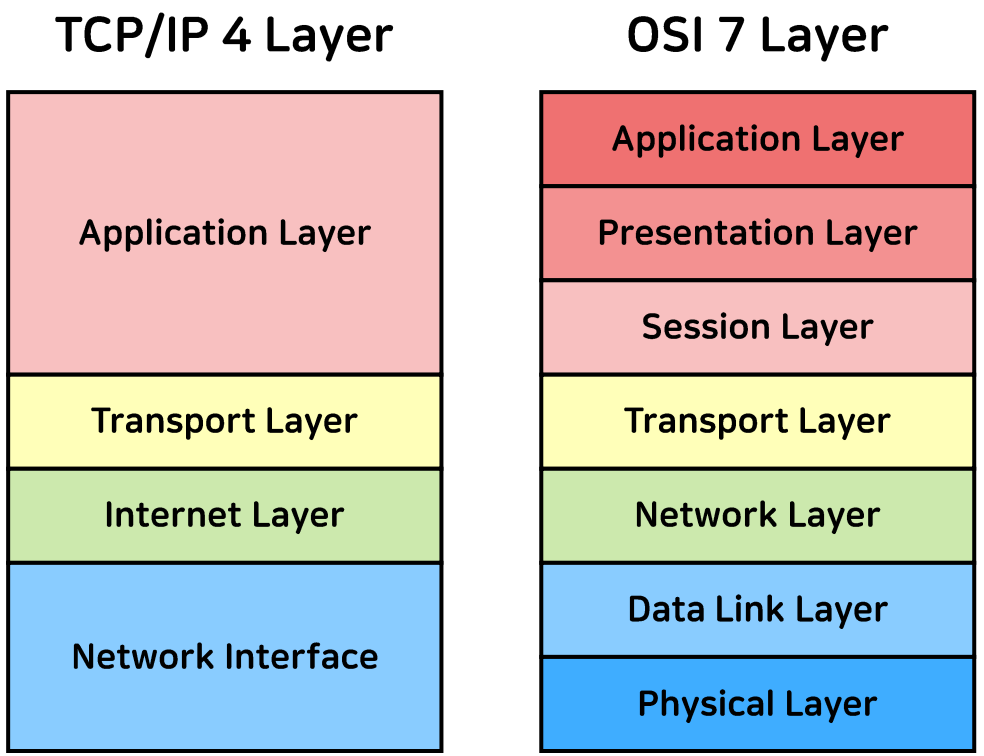
Physical Layer
: 물리적인 전송 매체를 통해 비트 단위의 데이터(0,1)를 전송
: 통신 케이블, 리피터, 허브

Application Layer
: OSI 7계층의 세션 계층(5), 표현 계층(6), 응용 계층(7)에 해당
: HTTP, FTP, SMTP, DNS 등의 다양한 프로토콜 사용

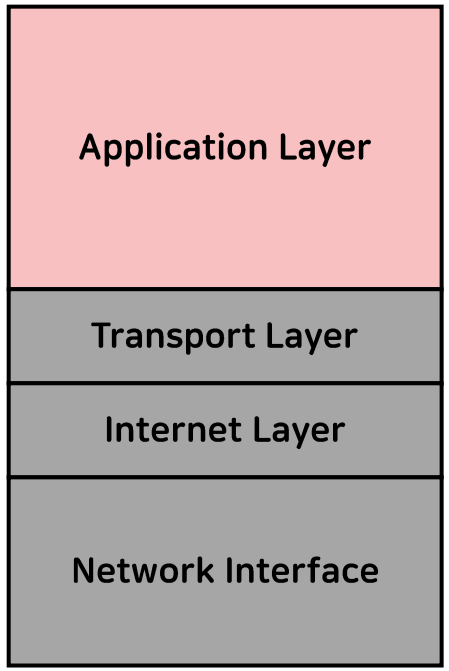
Transport Layer
: OSI 7계층의 전송 계층(4)에 해당
: Port를 열어서 프로그램 간 데이터 전송 가능하게 함
: TCP는 신뢰성 있는 데이터 전송을, UDP는 빠른 전송을 제공

Internet Layer
: OSI 7계층의 네트워크 계층(3)에 해당
: 어드레싱(addressing), 패키징(packaging), 라우팅(routing) 기능을 제공
: IP 프로토콜

Network Access Layer
: OSI 7계층의 물리계층(1)과 데이터 링크 계층(2)에 해당
: TCP/IP 패킷을 네트워크 매체로 전달하는 것과 네트워크 매체에서 TCP/IP 패킷을 받아들이는 과정을 담당
: 이더넷 프로토콜



Protocol (in Application Layer)



TCP/IP 4 Layer

FTP (File Transport Protocol) : 파일 전송

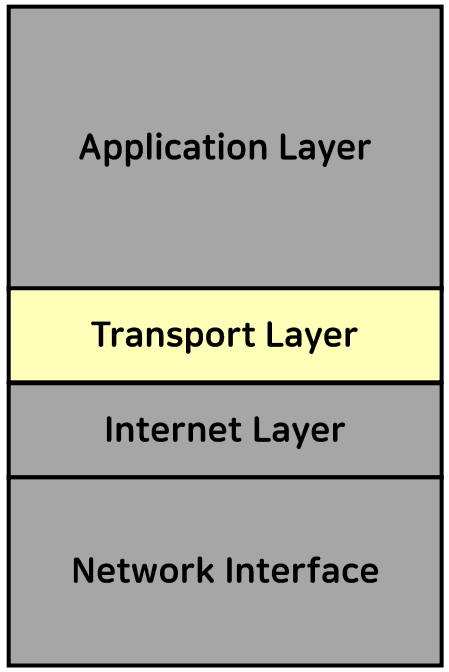
SSH (Secure SHell) : 보안되지 않은 네트워크에 접속하는 암호화 네트워크

HTTP (HyperText Transfer Protocol) : World Wide Web. 웹 사이트

SMTP (Simple Mail Transfer Protocol) : 전자 메일 전송

DNS (Domain Name Service) : 도메인과 IP 주소 매핑

Protocol (in Transport Layer)

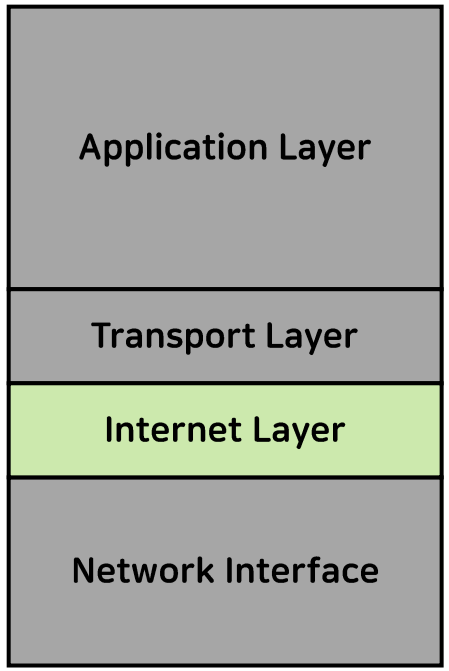


TCP/IP 4 Layer

TCP (Transmission Control Protocol)
: 순서 보장, 속도 느림, 응답 패킷 전송
ex) HTTP 통신, Email

UDP (User Datagram Protocol)
: 순서 미보장, 속도 빠름, 응답 패킷 미전송
ex) 음성 통화

Protocol (in Application Layer)



TCP/IP 4 Layer

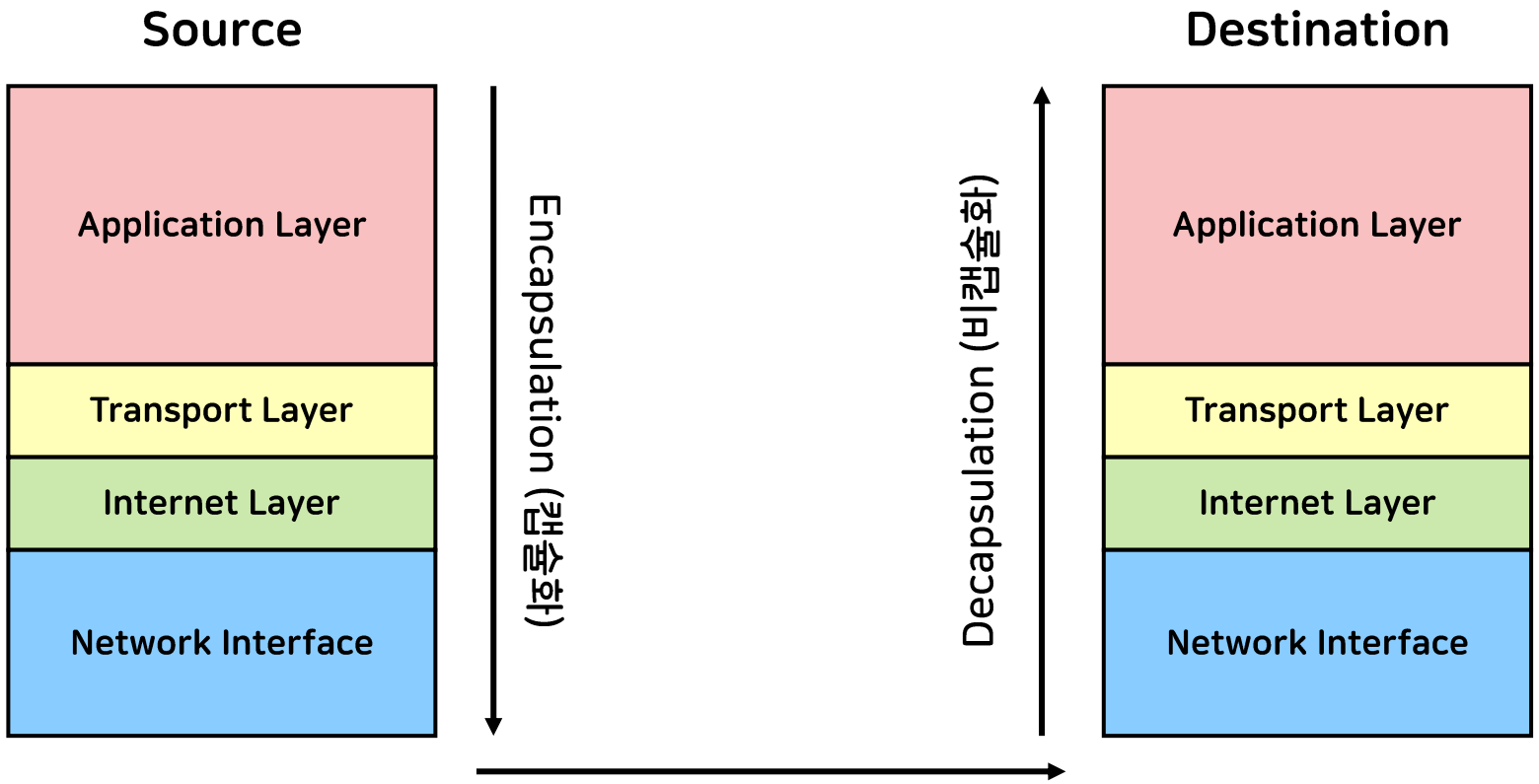
IP (Internet Protocol) : 목적지까지 경로 탐색

ICMP (Internet Control Message Protocol)

ARP (Address Resolution Protocol) : 목적지 ip 주소로 MAC주소 검색

RARP (Reverse ARP) : MAC 주소를 IP주소로 변환

Data Transport



Protocol Data Unit

Message

Segment(TCP), Datagram(UDP)

Packet

Frame (DataLinke Layer)
Bit (Physical Layer)

IP address

IP 주소는 네트워크 통신에 있어서, 각각의 통신기기에 할당된 식별 번호이다.
IP 주소는 통신기기마다 고유하게 할당되어 있는 것이 아닌, 인터넷 서비스 공급자가 제공받는 것이다.

Subnet Mask

아이피 주소에서 네트워크 주소와 호스트 주소를 나눠주는 역할을 한다.
호스트 간의 네트워크 통신은 같은 네트워크 주소/네트워크 대역에서만 이루어진다.
라우팅을 이용해 다른 네트워크 대역의 호스트와 연결 가능하다.
따라서 Ip 주소가 같더라도 subnet mask가 다르면 다른 주소가 된다.

IPv4 vs IPv6

IPv4는 32bit의 주소 체계를 사용한다. 4개의 10진수로 구성된 숫자로 표현되며 각 숫자는 .으로 구분된다.
ex) 192.168.0.1
패킷 교환 방식을 사용하여 데이터를 전송한다.
IPv4는 한정된 주소 공간을 가져, 주소 고갈 문제가 발생해 IPv6가 등장했다.

IPv6는 128bit의 주소 체계를 사용한다. 8개의 16진수 블록으로 구성되며 각 블록은 :으로 구분된다.
ex) 2001:0db8:85a3:0000:0000:8a2e:0370:7334
주소를 자동으로 구성해주며 DHCP 서버 없이도 네트워크에 연결될 수 있다.

MAC address

MAC 주소는 네트워크 인터페이스 카드나 네트워크 장비에 할당된 고유 식별자로, 물리적 장치의 주소이다.
IP 주소와 달리 네트워크에서 장치 간의 직접적인 통신을 담당한다.
48bit 주소로, 12자리의 16진수로 표시된다.
Ex) 00:14:22:01:23:45
MAC 주소는 네트워크에서 장비를 식별하고, 스위치나 라우터 같은 장비가 데이터를 정확하게 확인할 때 사용된다.
IP주소는 네트워크 계층에서 사용되지만, 실제 데이터 전송은 MAC 기반으로 이루어진다.

	IP address	MAC address
주소 형식	숫자 기반	16진수 기반
기능	네트워크에서 장치 식별 및 데이터 전달 경로 지정	네트워크 인터페이스 장치의 고유 식별자
계층	네트워크 계층	데이터 링크 계층
변경 가능 여부	변경 가능	변경 불가
용도	장치간 데이터 전송	장치 간 직접적인 통신

ARP

Address Resolution protocol

IP 주소를 MAC 주소로 변환해주기 위해 사용되는 동적 매핑 프로토콜
특정 IP 주소에 대해 broadcast를 통해 ARP request 메시지를 만들어 수신자에게 MAC 주소를 요청한다.
요청한 MAC 주소 정보를 Unicast를 통하여 ip와 mac 주소 정보를 응답 메시지로 받는다.

RARP

Reverse Address Resolution protocol

MAC 주소를 IP 주소로 변환해주기 위해 사용되는 동적 매핑 프로토콜
Broadcast 방식으로 MAC를 담고 있는 RARP request 메시지로 수신자에게 IP주소 요청
요청한 IP주소 정보를 Unicast를 통하여 MAC와 IP 주소 정보를 메시지로 받는다.

Broadcast
로컬 네트워크에 연결되어 있는 모든 시스템에게 프레임을 보내는 방식

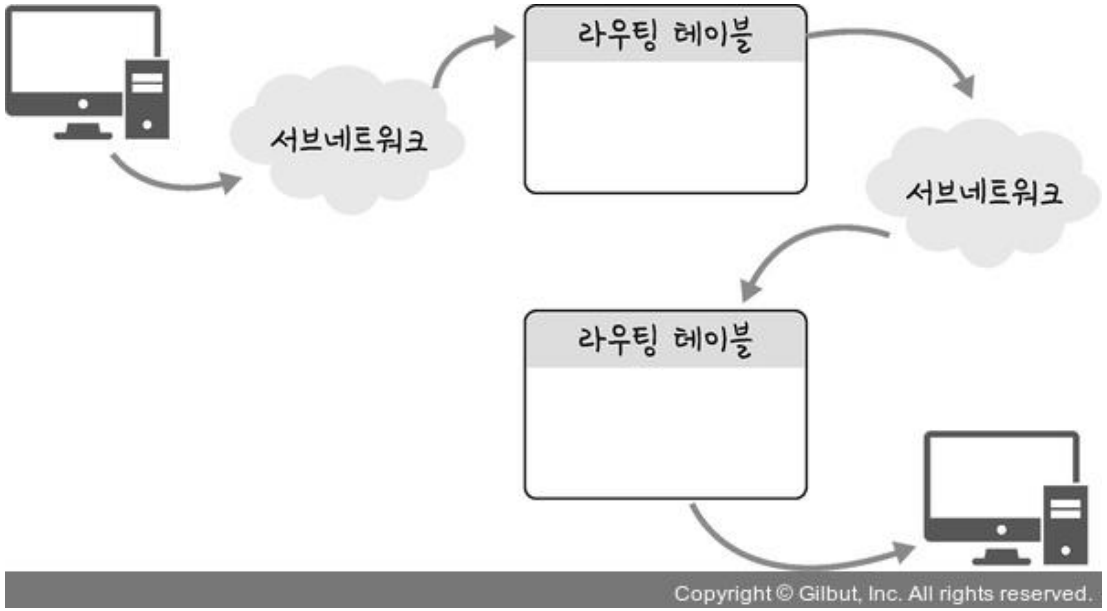
Unicast
정보를 전송하기 위한 프레임에 자신의 MAC 주소와 목적지의 MAC 주소를 첨부하여 전송하는 방식

Hop by hop

IP 주소를 통해 통신하는 과정을 hop by hop이라고 한다.

통신망에서 각 패킷이 여러 개의 라우터를 건너가며 통신을 한다.
각각의 라우터에 있는 라우팅 테이블의 IP를 기반으로 패킷을 전달하고 다시 전달한다.

통신 장치에 있는 라우팅 테이블의 IP를 통해 시작 주소부터 시작하여 다음 IP로 계속해서 이동하는 라우팅 과정을 거쳐 패킷이 최종 목적지까지 도달하도록 하는 통신이다.



: Hyper-Text Transfer Protocol

프로토콜 : 컴퓨터 또는 전자 기기 간의 원활한 통신을 위해 지키기로 약속한 규약.
프로토콜에는 신호 처리법, 오류 처리, 암호, 인증, 주소 등을 포함한다.

[출처: namu.wiki/w/프로토콜]

OSI Layer 중, Application Layer에서 사용하는 프로토콜
서버와 클라이언트(웹 브라우저)간의 통신을 위해 사용

- 요청-응답 사이클을 기반으로 함
  - 클라이언트가 HTTP Request Message를 서버로 전송
  - 서버가 요청을 처리하고 응답 메시지를 생성
  - 서버가 Respond Message를 클라이언트로 전송
  - 클라이언트가 응답을 처리하여 화면에 표시

- HTTP 메서드
  - GET : 서버에서 리소스를 가져옴
  - POST : 서버에 데이터를 전송해 새로운 리소스를 생성하거나, 기존 리소스를 수정
  - PUT : 기존 리소스를 수정
  - DELETE : 리소스를 삭제

- 무상태성 (Stateless)
  - 기본적으로 HTTP의 각 요청은 독립적으로 처리되며, 이전 요청과 연결되지 않음
  - 필요에 따라 쿠키나 세션 기술을 사용해 상태를 유지

: Representational State Transfer API

- 두 컴퓨터 시스템이 인터넷을 통해 정보를 안전하게 교환하기 위해 사용하는 인터페이스 [출처: AWS]
  - Resource : RESTful API에서 모든 것은 리소스로 취급되고,
    리소스는 고유한 URI(Uniform Resource Identifier로 식별
  - CRUD : 데이터베이스에서 데이터를 처리하는 기본적인 네 가지 작업
    RESTful API는 HTTP methods를 이용하여 자원을 CRUD 방식으로 관리

    CRUD 작업은 각 HTTP Methods와 직접적으로 매핑
    - Create : 새로운 데이터를 생성 POST
    - Read : 기존 데이터를 조회 GET
    - Update : 기존 데이터를 수정 PUT(전체)/FATCH(부분)
    - Delete : 리소스를 삭제 DELETE
  - 무상태성 (Stateless), 리소스 표현(by JSON, XML, …), Status Code(200, 404, …)

- 장점
  - 확장성 : 클라이언트, 서버를 독립적으로 개발 가능
  - 유연성 : JSON, XML등의 표준 형식은 다양한 플랫폼과 언어에서 사용 가능

- 단점
  - 무상태성으로 인한 부담 : 각 요청이 독립적이므로 매번 필요한 모든 정보를 표현해야 함
  - 복잡한 트랜잭션 처리가 필요한 경우 RESTful API만으로는 구현하기 어려울 수 있음

HTTP 1.0 (1996)

- 단일 요청-응답 : 클라이언트가 서버에 요청을 보내고, 응답을 받으면 TCP 연결이 종료됨
- 헤더 : 요청과 응답에 HTTP 헤더를 추가하여 메타데이터를 전송할 수 있게 함
- Context-Type : HTML 외에도 이미지, CSS 등 다양한 파일 형식 전송 지원
- 캐싱 : 상태 코드(200, 400, …)와, 캐시를 통해 브라우저가 로컬 데이터를 효율적으로 사용하여 트래픽 줄임

- 하지만, HTTP 1.0이 매번 TCP 연결을 설정/해제하는 것이 비효율적이었음

HTTP 1.1 (1997)

- Persistent Connection : 하나의 TCP연결을 여러 요청에 재사용할 수 있게 함
- Pipelining : 클라이언트가 응답을 마냥 기다리지 않고, 여러 요청을 동시에 보낼 수 있게 함
- Host Header : 동일한 IP 주소에서 여러 도메인을 호스팅할 수 있도록 함

- Head-of-Line Blocking 문제
  - Pipelining은 요청 순서대로 응답해야 하므로, 앞선 요청이 지연되면 뒤따르는 요 도 지연됨

HTTP 2 (2015)

- 구글의 SPDY 프로토콜을 기반으로 개발

- Multiplexing : 하나의 TCP 연결에서 여러 요청과 응답을 동시에 처리할 수 있게 함
- 헤더 압축(HPACK) : 중복된 HTTP 헤더 정보를 압축하여 네트워크 대역폭을 최적화
- Server Push : 클라이언트의 요청이 없더라도 서버가 미리 리소스를 전송할 수 있어 페이지 로딩 속도 개선
- 우선순위 처리 : 중요한 리소스를 먼저 전송하도록 우선순위를 설정할 수 있음

- Application 계층에서는 HOL Blocking 문제를 해결했지만,
  TCP 계층에서 패킷이 손실되면 전체 스트림이 지연되는 문제 여전

HTTP 3 (2022)

- QUIC 프로토콜(UDP기반)을 사용해서 TCP계층의 HOL Blocking 문제 해결
- 0-RTT 연결 설정 : 이전 연결 정보를 캐시하여, 재연결시 핸드셰이크 없이 빠르게 연결할 수 있음
  - 모바일 환경에서 네트워크가 자주 변경되어도 기존 연결을 유지할 수 있게 됨
- TLS(Transport Layer Security)를 기본으로 적용하여 보안성을 강화

- 한계점
  - 방화벽이나 보안 장비가 UDP 트래픽을 허용하지 않는 경우가 있음
  - QUIC 프로토콜은 복잡성이 높아서 개발자나 관리자의 학습 장벽이 높음
  - UDP 기반이어서 혼잡 제어 및 패킷 손실 복구 알고리즘을 개발자가 직접 구현해야 함

Task

T/F - 2문제

서술형 - 1문제

Database

데이터베이스

데이터베이스 용어 설명	DBMS, 엔티티/릴레이션, 애트리뷰트/도메인, 필드/레코드, 인덱스
데이터베이스 용어 설명	키(기본키, …), 관계(1대1, 1대N, …), 정규화(어떤 느낌인지만)
데이터베이스 용어 설명	RDBMS, NoSQL 설명

Curriculum

1주차 : Orientation	(9/4)
2주차 : Data Structure	(9/11)
3주차 : Algorithm 1	(9/25)
4주차 : Algorithm 2	(10/2)
5주차 : Computer Architecture	(10/16)
6주차 : Operating System	(11/6)
7주차 : Computer Network	(11/13)
8주차 : DataBase	(11/20)
9주차 : Design Pattern & Security	(11/27)
10주차 : Real Problems	(12/4)