**- OSI 7 Layer의 각 계층에 대해 설명해주세요.**

- 7계층 (Application Layer, 응용계층) : 최종 목적지로 응용 프로그램과 연관해 서비스를 수행하는 계층입니다. HTTP, FTP, DNS등이 7계층에 속합니다.

- 6계층 (Presentation Layer, 표현계층) : 데이터 압축 혹은 변환이 이뤄지는 계층으로, 데이터의 형식을 정의합니다. JPEG, MPEG등이 6계층에 속합니다.

- 5계층 (Session Layer, 세션계층) : 데이터가 통신하기 위한 논리적 연결을 담당하는 계층으로, 세션을 생성해 통신합니다. API나 Socket이 5계층에 속합니다.

- 4계층 (Transport Layer, 전송계층) : 종단 간 사용자들에게 신뢰성있는 데이터를 전달하는 계층입니다. 각 데이터의 단위는 세그먼트입니다. TCP와 UDP가 4계층에 속합니다.

- 3계층 (Network Layer, 네트워크계층) : IP를 지정하고 라우터로 경로를 선택해 네트워크를 통해 데이터를 전달하는 계층입니다. 각 데이터의 단위는 패킷입니다. Router가 3계층에 속합니다.

- 2계층 (Datalink Layer, 데이터링크계층) : 신뢰성 있는 전송을 위해 에러 검출, 흐름제어를 담당하며 데이터의 물리적인 전송이 일어나는 계층입니다. 데이터의 단위는 프레임입니다. Ethernet이 2계층에 속합니다.

- 1계층 (Physical Layer, 물리계층) : 데이터를 전기 신호로 변경하는 계층으로, 데이터 단위는 비트입니다.

**- 전송계층과 데이터링크 계층 모두 신뢰성 관련 기능을 제공하는데, 어떤 차이가 있나?**

- 전송계층은 논리적으로 연결된, 종단간의 호스트에 신뢰성 관련 기능(흐름제어, 혼잡제어, 오류제어)를 제공

- 데이터링크 계층은 물리적으로 연결된 호스트 사이의 전송으로 직접 묶여있는 호스트-노드 혹은 노드-노드 간의 신뢰성 관련 기능(오류제어, 흐름제어, 회선제어)를 제공

- 전송계층은 논리적 연결 (종단간 호스트) / 데이터링크 계층은 물리적 연결 (인접한 노드 or 호스트)

**- 전송계층이 데이터 분실을 확인하는 방법에 대해 설명해주세요.**

전송계층에선 데이터를 '패킷'단위로 쪼개어 같은 목적지로 전송합니다. 패킷에 각각 번호를 부여해 목적지에서 패킷을 재조립해서 패킷의 분실 여부를 확인

**-전송계층이 제공하는 신뢰성 관련 기능 중, 흐름제어에 대해 설명해주세요.**

- 송신측과 수신측 사이의 데이터 처리 속도를 제어하기 위해서 송신측의 데이터 처리 속도를 조절합니다.

그 결과 수신자의 버퍼 오버플로우를 방지할 수 있습니다.

- 흐름제어 기법으로 슬라이드 윈도우를 사용합니다.

**- 전송계층이 제공하는 신뢰성 관련 기능 중, 오류제어(재전송 프로토콜)에 대해 설명해주세요.**

- 오류제어는, 패킷이 손상되었거나 손실된 경우, 재전송을 통해 오류를 복구할 수 있는 방법으로 재전송 프로토콜이라고도 부릅니다.

- 재전송 프로토콜은 크게 3가지가 있습니다. 재전송은 비효율적이므로 적을수록 좋은 알고리즘입니다.

1) Stop and Wait

ACK를 받고 나서 다음 데이터를 보내는 방식입니다. 흐름제어에서의 슬라이딩윈도우를 사용하지

못하므로, 더욱 더 효율적인 재전송 프로토콜이 필요합니다.

2) Go-Back-N

연속으로 데이터를 보내다가, 오류가 발생한 지점부터 재전송하는 프로토콜입니다. 성공적으로

전송된 데이터까지 재전송하기 때문에 조금 비효율적인 측면이 있습니다.

3) Selective Repeat (SR)

오류가 발생한 데이터만 재전송하는 방식입니다. 다만, 수신 측 버퍼의 데이터가 순차적이지

않다는 것이 단점입니다.

정렬의 과정이 추가로 필요하고, 별도의 버퍼가 필요합니다. 따라서 Go-Back-N 재전송 프로토콜과

비교하여 상황에 맞게 사용하는 것이 좋습니다.

**- 전송계층이 제공하는 신뢰성 관련 기능 중, 혼잡제어에 대해 설명해주세요.**

네트워크의 혼잡을 피하기 위해 송신측에서 보내는 데이터의 전송 속도를 제어하는 것입니다.

여기서 혼잡은, 네트워크 내에 패킷의 수가 과도하게 증가하는 현상을 의미합니다.

이러한 혼잡 현상을 방지하고 제거하기 위한 기능이 혼잡제어 입니다.

1) AIMD 방식 (Additive Increase / Multiplicative Decrease)

처음 패킷 하나를 보내 문제가 없다면 Window Size를 1씩 증가시키는 방식

문제가 발생하면 Window Size를 절반으로 줄인다.

초기에 높은 대역폭을 사용하지 못하여 오랜 시간이 걸리고 네트워크가 혼잡해지는

상황을 미리 감지하지 못한다.

2) Slow Start 방식

처음 패킷을 하나씩 보내는 것은 같지만 매 전송마다 2배씩 증가하여 데이터의 크기가

지수함수적으로 증가한다.

전송되는 데이터의 크기가 임계값에 도달하면 혼잡 회피 단계로 넘어간다.

혼잡 현상이 발생하면 Window size를 1로 줄인다.

혼잡 현상이 발생했던 Window Size 절반 까지는 지수함수 꼴로 증가하고 이후부터는

1씩 증가한다.

3) Fast Recovery 방식

혼잡 시 1로 줄이지 않고 절반으로 줄이고 선형 증가시키는 방식

혼잡 상황을 한 번 겪은 이후로는 AIMD 방식으로 동작합니다.

4) 혼잡 회피 방식

임계값을 넘어가면 1씩만 선형적으로 증가시키는 방식

**- DNS가 무엇인지 설명해주세요.**

Domain Name Server의 약어로, IP주소를 문자로 표현한 주소로 바꾸는 시스템 혹은 서버를 의미

**- www.google.com 도메인을 브라우저에 입력했을 때 일어나는 일을 순차적으로 설명해주세요.**

1) www.google.com을 브라우저 주소창에 입력합니다.

2) 브라우저는 캐싱된 DNS 기록(브라우저 캐시, OS 캐시, 라우터 캐시, ISP 캐시)을 통해 해당 도메인 주소에 대응되는 IP주소가 있는지 확인합니다.

3) 요청한 URL이 캐시에 없는 경우, ISP의 DNS 서버가 www.google.com을 호스팅하고 있는 서버의 IP주소를 찾기 위해 DNS Query를 보내 찾습니다.

4) 브라우저는 www.google.com에 해당하는 IP 주소를 획득해 해당 서버와 TCP 연결을 시도합니다.

(3-way handshake)

5) TCP 연결이 완료되면, 브라우저가 웹 서버에 HTTP Request를 보냅니다.

6) 서버는 해당 Request를 처리하고 Response를 생성해 브라우저에게 보냅니다.

7) 브라우저는 해당 Response를 읽어들여 사용자에게 보여줍니다.

**- TCP Connection 시 3-way handshaking을 한다고 했는데, 이에 대해 설명해?**

TCP는 신뢰성있는 전송을 위해 1:1로 서로 잘 연결되었음을 확인하는 과정을 거칩니다. 이 과정이 바로 3-way handshaking이라고 부릅니다.

1) Client가 Server에게 접속 요청 SYN 패킷을 보냅니다.

2) 서버는 SYN 패킷을 받고, 해당 요청을 수락하는 ACK와 Client와 연결하겠다는 SYN 패킷을 동시에 보냅니다.

3) Client는 서버에게 확인 응답으로 ACK 패킷을 보냅니다.

이처럼, 서버와 클라이언트 모두가 SYN와 ACK를 서로 주고 받는 과정을 완료하면 비로소 연결이 되었다고 말할 수 있습니다.

**- 그렇다면 TCP Connection을 해제하는 경우는 어떻나요?**

3-way handshaking이 아닌, 4-way handshacking을 진행합니다.

1) Client가 Server에게 FIN 신호를 보냅니다.

2) Server는 Client가 보내온 FIN 신호에 응답하는 ACK 신호를 보냅니다.

3) Server는 Client에게 FIN 신호를 보냅니다.

4) Client는 Server가 보내온 FIN 신호에 응답하는 ACK 신호를 보냅니다.

**- 연결해제 시 4-way handshacking을 사용하는 이유는 무엇인가요?**

Client는 Server에게 데이터 요청을 모두 보냈다하더라도, Server는 Client에게 보내야 할 데이터가 남아있을 수 있기 때문에 FIN과 ACK를 한 번에 보내지 않고, 서버 측에서 보낼 데이터 마저 다 보냈을 경우 FIN 메시지를 보낸다.

**- 포트가 무엇이고, 소켓이 무엇인지 설명해주세요.**

- 포트는 네트워크를 통해 데이터가 주고받는 프로세스를 식별하기 위해서 호스트 내부적으로 프로세스가

할당받는 고유한 값입니다.

- 하나의 IP 주소 내에 개별적으로 부여된 통신 프로세스입니다.

- 소켓은 네트워크 상에서 동작하는 프로그램 간 통신의 종착점으로, 두 시스템 사이의 네트워크 연결을

나타내는 객체입니다.

- 소켓을 열기 위해서는 IP, 포트번호, 프로토콜이 필요하며, 송수신자 모두 소켓을 열어야 합니다. 하나의

프로세스가 같은 포트를 가지고 여러개의 소켓을 열 수 있습니다.

**- TCP와 UDP의 차이에 대해 설명해주세요.**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**- HTTP의 특징에 대해 설명해주세요.**

- 비연결지향 (connectionless) : client가 server에 request를 보내고, 서버가 해당 요청에 맞는 response를

보내면 바로 연결을 끊습니다.

- 상태 비저장 (stateless) : 연결을 끊는 순간 client와 server의 통신은 끝나며, 상태정보를 유지하지 않는

특징이 있습니다.

- 하지만, 가끔씩 이 '상태'에 대한 저장이 필요한 경우가 있어서 쿠키와 세션이 등장하게 됩니다.

**- HTTP와 HTTPS의 차이점에 대해 설명해주세요.**

1) HTTP

- world wide web에서 정보를 주고받을 수 있는 프로토콜

- client-server간의 request-response 프로토콜

- 주로 HTML 문서를 주고 받는데 사용함

2) HTTPS

- HTTP의 보안상 문제를 해결하기 위해 나온 프로토콜

- HTTP는 TEXT로 자원을 주고 받기 때문에 네트워크 중간을 가로챈다면, 내용이 유출되는 사고가 생김

- 따라서 SSL, TLS를 이용해 암호화를 거쳐 주고 받음 (응용-전송 계층 사이에 위치)

- HTTP Request와 Response 데이터는 네트워크로 보내지기 전, 전송-응용 계층 사이에서 암호화 됨.

- Body 부분만 암호화, Header는 암호화 X

- Client가 Server로 접속을 요청하면, Server는 공개키를 Client에게 전달. Client는 자신의 대칭키를 받은

공개키로 암호화해서 Server에 전달. Server는 개인키로 복호화해서 Client의 대칭키를 획득. 얻어낸

대칭키를 사용해 암호문을 주고 받음

- 암호화 비용이 발생하고, 암호화 과정에서 서버에 부하가 발생하며, 연결이 끊기면 재인증 시간이 소요됨.

**- HTTP Method에는 어떤 것들이 있는지 설명해주세요.**

텍스트, 스크린샷, 영수증, 폰트이(가) 표시된 사진

자동 생성된 설명

**- 멱등성에 대해 설명하고, HTTP 메소드에서는 어떤 메소드가 멱등성 메소드인지 설명해주세요.**

- 특정 메소드를 여러번 요청했는데, 매번 요청 결과가 같은 경우 해당 메소드를 멱등성 메소드라고 부릅니다.

- GET, PUT, DELETE를 멱등성 메소드에 속하고 POST는 멱등성 메소드가 아닙니다.

- (POST 요청을 반복하면 같은 내용이더라도 다른 데이터가 계속 추가된다.)

**- HTTP의 Keep-alive 헤더에 대해 설명해주세요.**

HTTP는 매번 연결을 끊고 새로 생성합니다. 특정 시간까지는 접근이 없더라도 연결상태를 유지하도록 하는 헤더입니다.

HTTP 1.1부터 지원합니다.

**- CORS가 무엇인지 설명해주세요.**

교차 출처 리소스 공유(Cross-Origin Resource Sharing, CORS)는 추가 HTTP 헤더를 사용해 한 출처에서 실행

중인 웹 어플리케이션이 다른 출처의 선택한 자원에 접근할 수 있는 권한을 부여하도록 브라우저에 알려주는

체제입니다.

Web Application은 리소스가 자신의 출처(도메인, 프로토콜, 포트)와 다를 때 교차 출처 HTTP 요청을 보냅니다.

**- REST API와 RESTful 하다는 것이 무엇인지 설명해주세요.**

REST는 HTTP URI를 통해 자원을 명시하고 HTTP Method를 통해 자원을 처리하도록 설계된 아키텍처입니다.

이러한 REST를 기반으로 만든 API를 REST API라고 부릅니다.

RESTful 하다는 것은, REST 원리를 잘 따르는 시스템을 RESTful 하다고 표현합니다.

REST 원리에는 다양한 조건들이 포함되어있습니다.

URI 작성 조건

HTTP Method로 행위 표현

self-descriptive

HATEOAS 등등.

HTTP의 자원을 제대로 활용하고 있지 않다면 REST하지 않다고 할 수 있습니다.

**- 쿠키와 세션이 왜 필요한지, 어떻게 다른지 설명해주세요.**

HTTP 프로토콜은 비연결형, 무상태의 특성을 가지기 때문에, server는 client가 누구인지 매번 확인이 필요합니다.

이러한 특성을 보완하기 위해서 쿠키와 세션이 탄생했습니다.

1) 쿠키 : 클라이언트 로컬(하드디스크)에 저장되는 키와 값이 들어있는 작은 데이터 파일

2) 세션 : 일정 시간 동안 같은 브라우저로부터 들어오는 일련의 요구를 하나의 상태로 보고 그 상태를

유지하는 것.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

세션은 서버의 자원을 사용하기 때문에 사용자가 많을 수록 소모되는 자원이 많기 때문에 상황을 잘 고려해서 사용해야 한다.

**- Session은 서버에 저장되고, Cookie는 클라이언트에 저장되는데, 쿠키가 안되는 상황에서 세션은 사용할 수 있을까요?**

쿠키를 사용할 수 없는 환경에서는 URL 뒤쪽에 Parameter 값으로 전달하는 경우가 있습니다.

**- JWT Token이 무엇인지 설명해주세요.**

JWT는 JSON 포맷을 이용하는 Claim 기반의 웹 토큰이며, 토큰 자체를 정보로 사용하는 Self-Contained 방식으로 정보를 안전하게 전달합니다.

JWT는 헤더(Header).내용(Payload).서명(Signature)로 구성되며 각 파트를 점(.)으로 구분합니다.

- 헤더(Header) : 토큰의 타입과 해시 암호화 알고리즘(방식지정)으로 이루어져 있다.

- 내용(Payload) : 토큰에 사용자가 담고자 하는 정보를 담는다. 내용에는 Claim이 담겨있고, JSON(Key/Value)형태의 한 쌍으로 이루어져 있다.

- 서명(Signature) : 토큰을 인코딩하거나 유효성 검증할 때 사용하는 고유한 암호화 코드이다.

헤더와 내용의 값을 인코딩한다.

**- URI와 URL이 무엇이 다른지 설명해주세요.**

URI

URI는 특정 리소스를 식별하는 통합 자원 식별자(Uniform Resource Identifier)를 의미합니다.

웹 기술에서 사용하는 논리적 또는 물리적 리소스를 식별하는 고유한 문자열 시퀀스입니다.

URL

URL은 흔히 웹 주소라고도 하며, 컴퓨터 네트워크 상에서 리소스가 어디 있는지 알려주기 위한 규약입니다.

URI의 서브셋입니다.

**- GET과 POST가 다른 점에 대해 설명해주세요.**

- GET은 데이터를 조회하기 위해 사용되는 방식으로 데이터를 헤더에 추가하여 전송하는 방식입니다.

URL에 데이터가 노출되므로 보안적으로 중요한 데이터를 포함해서는 안됩니다.

- POST는 데이터를 추가 또는 수정하기 위해 사용되는 방식으로 데이터를 바디에 추가하여 전송하는

방식입니다.

완전히 안전하다는 것은 아니지만 URL에 데이터가 노출되지 않아 GET 보다는 안전합니다.

- 또한, GET은 캐싱이 가능하며, 멱등성 메소드입니다.

- 반면 POST는 캐싱이 불가능하며, 멱등성 메소드가 아닙니다.

**- Connection Timeout과 Read Timeout의 차이에 대해 설명해주세요.**

- 서버 자체에 클라이언트가 어떤 사유로 접근을 실패했을 시 적용되는 것이 Connection Timeout입니다.

- 즉, 접근을 시도하는 시간 제한이 Connection Timeout 되는 것을 말합니다.

- 클라이언트가 서버에 접속을 성공 했으나 서버가 로직을 수행하는 시간이 너무 길어 제대로 응답을

못 준 상태에서 클라이언트가 연결을 해제하는 것이 Read Timeout입니다.

- 이 경우는 클라이언트는 해당 상황을 오류로 인지하고, 서버는 계속 로직을 수행하고 있어 성공으로

인지해 양 사이드간 싱크가 맞지 않아 문제가 발생할 확률이 높습니다.

**- 공인 IP와 사설 IP의 차이에 대해 설명할 수 있나요?**

- 공인 IP는 ISP(인터넷 서비스 공급자)가 제공하는 IP 주소이며, 외부에 공개되어 있는 IP주소 입니다.

- 사설 IP는 일반 가정이나 회사 내 등에 할당된 네트워크 IP 주소이며, IPv4의 주소부족으로 인해

서브넷팅된 IP이기 때문에 라우터(공유기)에 의해 로컬 네트워크상의 PC나 장치에 할당됩니다.

- 사설 IP 주소만으로는 인터넷에 직접 연결할 수 없고, 라우터를 통해 1개의 공인 IP를 할당하고,

라우터에 연결된 개인 PC는 사설 IP를 각각 할당 받아 인터넷에 접속 할 수 있습니다.

**- UDP가 TCP에 비해 갖는 장점에 대해 설명해보세요.**

- 최소전송률만 보장되면 되고, 지연률에 민감한 경우엔 UDP가 TCP보다 낫습니다.

- 연결 설정을 위한 지연이 없습니다. (3-way handshacking)

- 연결 상태를 저장하지 않기 때문에 더 많은 클라이언트를 수용할 수 있습니다. (TCP는 연결을 유지하므로

송수신 버퍼, 각 파라미터를 갖고 있어야 함.)

- TCP는 세그먼트마다 20바이트의 오버헤드를 갖습니다. UDP는 8바이트의 오버헤드를 갖습니다.

- 어플리케이션 레벨에서 신뢰성을 보장하면 UDP를 사용하더라도 신뢰성 있는 데이터 전송을 구현할 수

있습니다.

**- UDP에 Checksum 필드가 있는 이유는 무엇일까요?**

- 모든 데이터링크 계층이 오류검사를 하는 것은 아니기 떄문입니다. 종단간의 데이터 전송 서비스가

오류검사를 제공한다면 데이터링크 계층에서 오류검사를 제공하지 않는 경우 UDP가 제공해야 하기

때문에 Checksum 필드가 존재합니다.

- 그러나, UDP는 신뢰성 있는 프로토콜이 아닙니다. 체크섬을 통해 오류를 발견하더라도 오류를 회복하기

위해서 어떤 일도 하지 않습니다.