Network

1. 네트워크란?

1-1. 프로토콜

송신자와 수신자간의 통신을 위해서는 서로 약속된 규칙을 따라야한다. 그것을 바로 프로토콜이라고 한다.

1-2. 계층모델

- OSI 7계층

- 응용계층 : 사용자 응용프로그램
- 표현계층 : 데이터 압축, 암호화
- 세션계층 : 논리적 연결 설정
- 전송계층 : TCP/UDP 연결 설정 (세그먼트/데이터그램)
- 네트워크계층 : IP주소 할당, 라우팅 (패킷)
- 데이터링크계층 : Mac주소 할당, 인터페이스 일치 (프레임)
- 물리계층 : 물리적 데이터 전송

- TCP/IP계층

OSI 7계층에서 전송계층까지는 동일하지만 그 위로는 사용자 응용프로그램이 구현한다.

1-3. LAN, MAN, WAN

네트워크 규모별로 구분한다.

- LAN : 학교, 기업단지같은 소규모 지역
- MAN : 도시 지역
- WAN : 전역

1-4. 네트워크 구성 단위

- 게이트웨이

- 리피터 : 신호 증폭
- 브리지 : 2계층 구조
- 라우터 : 라우팅 역할, 3계층 구조, 라우팅 테이블 보관

1-5. 제어 관리

- 흐름제어

송신측과 수신측의 속도 조절

수신측의 버퍼가 꽉차고 송신측의 전송속도가 더 높으면 수신측이 데이터를 받을수가 없다. 이럴때 양쪽간의 속도를 조절해주는 것이 흐름제어이다.

- stop and waiting
- sliding window

- 혼잡제어

네트워크상의 라우터에 데이터가 몰릴 경우 라우터의 데이터 처리 과부하가 생긴다. 이로인해 생길 데이터 손실을 위해 송신측의 전송 속도를 낮추는 것이다.

- AIMD
- Slow Start
- Fast Recovery (빠른 회복)
- Fast retransmission (빠른 재전송)

- 오류제어

신뢰있는 통신을 목적으로 한다.

- stop and wait ARQ
- go back N ARQ
- selective reject ARQ

2. 데이터링크 계층

2-1. LLC / MAC

- LLC 층

  LAN의 종류와 상관없이 공통의 프로토콜 사용한다.

- MAC 층

  물리적 인터페이스 통일

2-2. 오류제어/흐름제어

- stop and wait

  프레임을 하나 보낼때마다 잘받았다라는 ACK을 받아야 다음 프레임을 전송한다.

  이는 신뢰성을 제공해주지만 전송 효율은 떨어지는 단점이 있다.

- sliding window

  프레임을 연속적으로 보내는 방식이다. 한꺼번에 다 보내는 것이아니라 winodw size를 정하여 보내는 방식이다.

  window 고정 size가 3이라면 3개의 프레임을 연속으로 보내면서 window size를 1씩줄이고 수신측의 ACK을 기다린다. 수신측의 ACK이 올때마다 window size를 1개씩 증가시킨다.

- go back N

  슬라이딩 윈도우 기법처럼 운용을하다가 오류가 생겼을 경우 수신측은 받지 못한 프레임(오류가 난 프레임)을 재요청한다. NAK으로 재요청을 하면 송신측은 재요청받은 프레임부터 다시 재전송을한다. 이는 제대로 전송된 뒤 프레임도 같이 재전송을 해야하는 문제가 있다.

- 선택적 재전송

  go back N의 제대로 전송된 프레임의 재전송을 막기위해 문제가 생긴(오류가 생긴) 프레임만 재요청을 하고 해당 프레임만 재전송하는 방식이다. 이는 프레임 도착 순서가 달라질 수 있는 문제점이 있다.

3. 네트워크 계층

3-1. IP주소 할당

- IPv4
- IPv6

3-2. 라우팅

3-3. 혼잡제어

- AIMD (Additive Increase / Multicative Decrease)

  합증가 곱감소 방식이라고 불린다. 윈도우의 크기를 1씩 증가시키면서 보내다가 혼잡 발생 예측시 (응답이 안오거나, 데이터 유실 상황) 윈도우 사이즈를 1/2를 줄인다.

- Slow Start

  AIMD는 윈도우를 선형적으로 증가시키기 때문에 네트워크에 호스트가 많지 않아도 느린속도로 증가되는 문제가 있다.

  Slow Start는 지수적으로 윈도우 사이즈를 증가시키다가 혼잡상태가 발생하면 윈도우 크기를 1로 줄여 선형적으로 증가시킨다.

- Slow Start Threshold

  Threshold(임계점)까지는 Slow Start를 하다가 이 임계점을 지나면 윈도우 사이즈를 선형 증가 시킨다.

  <타임아웃, ACK 중복3번일 경우>

  - TCP Tahoe

  Threshold를 절반으로 줄이고 윈도우 사이즈를 1로 줄인 다음 Slow Start를 한다. 다시 Threshold에 도달하면 선형증가를 한다.

  - TCP Reno

  Threshold를 절반으로 줄이고 윈도우 사이즈를 Threshold 부터 선형 증가를 한다.

4. 전송 계층

4-1. TCP / UDP

- TCP

  송신자와 수신자가 서로 신뢰있는 통신

SOCK_STREAM

- UDP

  수신 신뢰와 관계없이 송신자는 데이터를 연속적으로 보낸다.

  SOCK_DGRAM

4-2. TCP 3 way handshake / 4 way handshake

- 연결 요청

  1. 클라이언트->서버 : 연결 요청 SYN(x)
  2. 서버->클라이언트 : 응답 ACK(y, x+1)
  3. 클라이언트->서버 : 응답 ACK(y+1)

- 연결 종료

  1. 클라이언트->서버 : FIN
  2. 서버->클라이언트 : ACK

     서버가 남은 데이터 전송
  3. 서버->클라이언트 : FIN

     클라이언트 : Time wait (서버가 보낸 데이터 마저 받기 위해서)
  4. 클라이언트 : ACK

4-3. Well Known Port Number

- FTP (DATA) : 20
- FTP (Control) : 21
- Telnet : 23
- SMTP : 25
- DNS : 53
- HTTP : 80

4-4. Socket 통신

1. 서버, 클라이언트 : socket(domain, type, protocol) type 인자에서 TCP/UDP를 정한다.
2. 서버 : bind() - socket()으로 만들어진 파일 디스크립터를 주소와 바인딩시킨다.
3. 서버 : listen() - 클라이언트 요청 대기
4. 클라이언트 : connect() - 서버에 연결 요청
5. 서버 : accpet() - 새로운 소켓 생성하여 클라이언트와 연결
6. 서버, 클라이언트 : send()/recv() - 데이터 전송
7. 서버, 클라이언트 : close() - 연결 종료

- UDP 연결일때는 listen() 생략, sendto()/recvfrom()을 대신 사용

5. 웹서비스

웹 서버의 TCP 포트 번호 : 80

사용자가 URL을 입력하는 것은 해당 웹서버의 welcome.html 파일을 요청하는 것이다.

1. 클라이언트가 URL을 입력한다.
2. URL을 DNS 서버에 전송하여 해당 웹서버의 IP주소를 얻는다.
3. IP주소와 80 포트번호로 웹서버와 TCP연결을 한다.
4. 클라이언트는 웹서버에 welcome.html 파일을 요청한다 (GET)
5. 웹서버는 클라이언트에게 welcome.html 파일을 전송한다.
6. 서버와 클라이언트는 TCP연결을 종료한다.

5-1. HTTP / HTTPS

HTTP는 클라이언트와 서버의 웹통신을 위한 프로토콜이다.

HTTPS는 HTTP의 보안문제를 해결해주는 프로토콜이다. 보안문제 해결로는 공개키 암호화 방식을 사용한다.

1. 서버 기업은 공개키와 암호키를 만든다.
2. 신뢰할 수 있는 CA(공개키 저장 관리 기업)에 서버 기업은 공개키를 주면서 계약을 맺는다.
3. CA는 서버 기업의 이름과 공개키 그리고 인증서를 만든다.
4. CA는 만든 인증서를 CA의 개인키로 암호화하여 서버 기업에 준다.
5. 클라이언트가 서버 기업에 요청을 하면 서버 기업은 CA기업이 만들어준 인증서를 전송한다.
6. 클라이언트는 신뢰할수있는 CA의 공개키를 이미 알고 있다.
7. CA 공개키로 서버 기업이 준 인증서를 해독한다.
8. 클라이언트는 서버 기업의 공개키를 얻을 수 있다.
9. 클라이언트와 서버는 신뢰 통신을 할 수 있다.

> https://jeong-pro.tistory.com/89