TCP/IP 흐름제어 & 혼잡제어

흐름제어

송신측과 수신측의 데이터 처리 속도 차이를 해결하기 위한 기법

수신측이 송신측보다 빠르면 문제가 없으나, 그 반대의 경우 문제가 생김

(수신측에서 제한된 저장용량을 초과한 이후에 도착하는 데이터는 손실이 될 수 있으며, 손실이 된다면 불필요한 응답과 데이터 전송이 양측에 발생함)

⇒ 이러한 문제가 발생하지 않기 위해, 송신 측의 데이터 전송량을 수신측에 따라 조절해야 함.

이를 위해 Stop and Wait 방식과 Sliding Window 방식이 존재함

Stop and Wait

매번 전송한 패킷에 대해 확인 응답을 받아야만 그 다음 패킷을 전송하는 방법

Sliding Window

수신측에서 설정한 윈도우 크기만큼 송신측에서 확인응답 없이 세그먼트를 전송할 수 있게 하여, 데이터 흐름을 동적으로 조절하는 제어 기법

윈도우에 포함되는 모든 패킷을 전송하고, 그 패킷들의 전달이 확인되는대로 이 윈도우를 옆으로 옮김으로써, 그 다음 패킷을 전송함

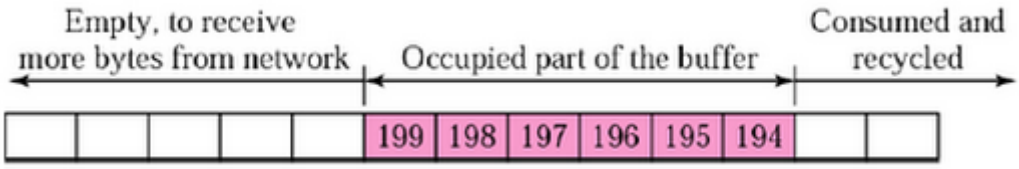
전송은 되었으나, ACK 를 받지 못한 바이트의 숫자를 파악하기 위해 사용되는 프로토콜

> (LastByteSent) - (LastByteAcked) <= (ReceivecWindowAdvertised) (마지막에 보내진 바이트) - (마지막에 확인된 바이트) <= (남아있는 공간)

동작 방식 : 윈도우에 포함되는 모든 패킷을 전송하고, 패킷들의 전달이 확인되는대로 윈도우를 옆으로 옮김으로써 다음 패킷들을 전송함

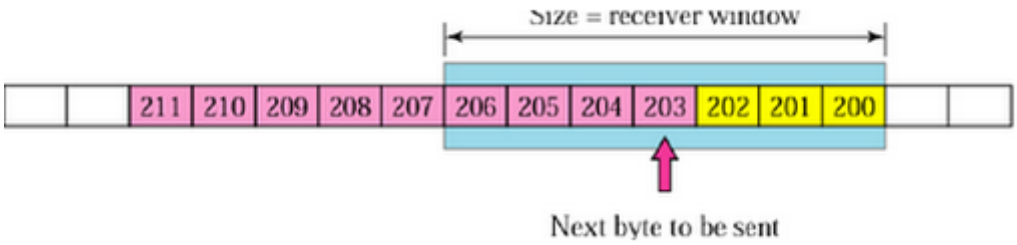
윈도우 : TCP/IP를 사용하는 모든 호스트들은 송신과 수신을 위해 2개의 윈도우를 가지고 있음. 호스트들은 실제 데이터를 보내기 전 3 way handshaking 을 통해 수신 호스트의 receive window size 만큼 자신의 send window size 를 맞춰야함

- **수신 윈도우**



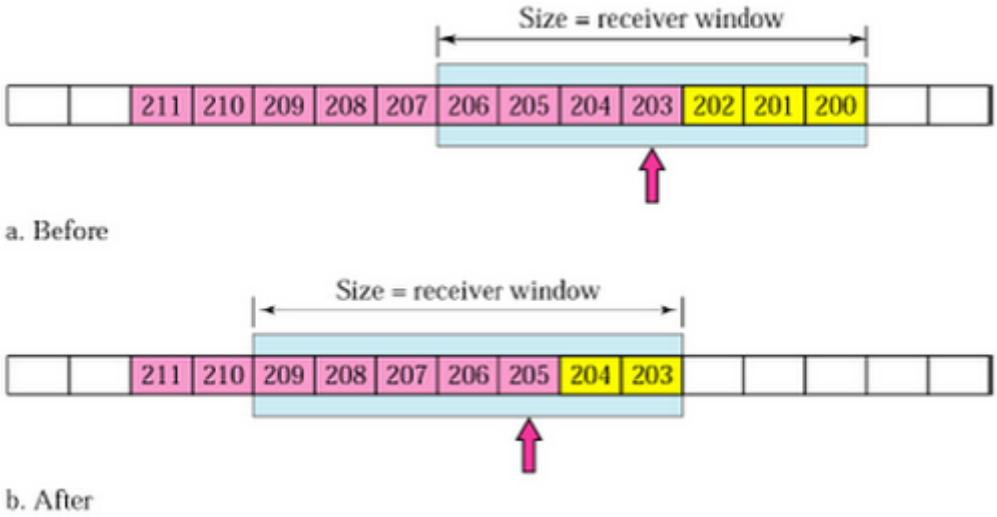
1. 수신 프로세스가 처리할 바이트는 194바이트부터
2. 수신 윈도우는 200의 수신을 기다림

- **송신 윈도우**



1. 수신 윈도우보다 작거나 같은 크기로 송신 위도우를 지정하게 되면 흐름제어 가능

- **송신 윈도우의 이동**



1. Before상태에서 203~204를 전송하면 수신측에서 확인응답으로 203을 보냄
2. 송신측은 이를 확인응답을 받아 After사애와 같이 윈도우를 203~209범위로 이동시킴
3. 그 후 205~209가 전송 가능한 상태가 됨

혼잡제어

손신측의 데이터 전달과 네트워크의 데이터 처리 속도 차이를 해결하기 위한 기법

한 라우터에 데이터가 몰리는 경우 등의 네트워크 혼잡을 피하기 위해 송신측에서 보내는 데이터의 전송속도를 강제로 줄이는 작업

네트워크 내에 패킷의 수가 과도하게 증가하는 현상을 방지하는 현상도 혼잡제어라고 함

송신측과 수신측의 전송속도를 다루는 흐름제어와 달리, 혼잡 제어는 호스트와 라우터를 포함한 넓은 관점에서의 전송 문제를 다룸

혼잡제어 해결방법

1. **AIMD (Additive Increase / Multicative Decrease) 합 증가 / 곱 감소**

   패킷을 하나씩 보내고 문제없이 도착하면 윈도우 크기 (단위 시간 내에 보내는 패킷의 수)를 1씩 증가시켜가며 전송하는 방법

만약 패킷 전송을 실패하거나, 일정시간을 넘으면 패킷을 보내는 속도를 절반으로 줄이게됨

매우 공평한 방식으로, 여러 호스트가 한 네트워크를 공유하고 있으면 나중에 진입하는 쪽이 불리하지만, 시간이 지날수록 평행상태로 수렴하는 특징이 있음

문제점으로는, 초기 네트워크의 높은 대역폭을 사용하지 못해 오랜 시간이 걸리고, 혼잡해지는 상황을 미리 감지하지 못해 네트워크 혼잡을 경험한 후 대역폭을 조정하는 방식

2. **Slow Start (느린 시작)**

   합 증가 / 곱 감소 방식과 마찬가지로, 패킷을 하나씩 보내는 것부터 시작하고, 패킷이 문제 없이 도착하면 각각의 ACK 패킷마다 윈도우 크기를 1씩 늘려줌

   위와 같은 방법으로 한 주기가 지나면, 윈도우 크기는 2배가 되며, 이러한 전송속도는 지수 함수의 꼴로 증가함

   만약 혼잡제어 현상이 발생하면, 윈도우 크기를 1로 떨어뜨림

   초기에 네트워크 수용량을 예상할 수 있는 정보가 없으나, 혼잡제어 발생 시 네트워크 수용량을 어느정도 예상할 수 있으므로, 윈도우 크기의 절반까지는 이전과 같이 지수함수 꼴로 크기를 증가시키고, 이후부터는 완만하게 1씩 증가시킴

3. **Fast Retransmit (빠른 재전송)**

   패킷을 받는 쪽에서 먼저 도착해야 할 패킷이 도착하지 않고, 다음 패킷이 도착한 경우에도 ACK 패킷을 보내게 됨

   순서대로 잘 도착한 마지막 패킷의 ACK 패킷보다 다음 패킷이 먼저 도착하여 다음 패킷의 순번을 ACK 패킷을 먼저 실어서 보내게 된다면, 중간에 패킷 하나가 손실되게 되는데 이를 해결하는 방법

   빠른 재전송은 중복된 순번의 패킷을 3개 받으면 재전송을 하게 됨. 이러한 현상은 약간의 혼잡한 상황이 발생한 것이므로, 혼잡을 감지하고 윈도우 크기를 줄이게 됨

4. **Fast Recovery (빠른 회복)**

   혼잡한 상태가 되면, 윈도우 크기를 1로 줄이지 않고 반으로 줄이고, 선형 증가시키는 방법

   혼잡 상황을 한 번 겪고나서부터는 순수한 합 증가 / 곱 감소 방식으로 동작하게됨