네트워크 개론 Part 18 : TCP Transmission Control Protocol의 이해 6

조인준
KBS 미디어기술연구소
차장

지금까지 TCP와 관련하여 TCP 패킷의 헤더 구조, 핸드쉐이킹, 데이터 전송 방식에 대해 다루며 Go-Back-N ARQ와 Selective Repeat ARQ까지 설명하였습니다. 이 과정에서 개념적 모델을 좀 더 구체적으로 설명하기 위해 실제 헤더 및 페이로드 데이터 등과 연관지어 설명하다 보니, 추상적 모델에 대한 구체적 예를 드는 과정에서 조사한 자료의 오류나 자료 간 내용을 종합하는 과정에서 발생한 실수로 잘못된 내용도 전달하게 되어 지난 편에서 앞선 내용에 대한 수정을 올리기도 했습니다.
이번 편에서도 앞선 내용에서 오류를 발견하여 사과의 말씀과 함께 수정 내용을 안내드립니다.

'TCP(Transmission Control Protocol)의 이해 3, 4, 5'에서 슬라이딩 윈도우의 크기를 TCP 헤더의 윈도우 사이즈(Window Size) 필드를 통해서 정의하고 각 패킷은 TCP 헤더의 시퀀스 번호(Sequence Number)로 구분된다는 언급이 있었습니다. 이에 대해 해당 내용을 '슬라이딩 윈도우의 크기는 TCP 헤더의 윈도우 사이즈(Window Size)와 연관되고 각 패킷의 구분에 TCP 헤더의 시퀀스 번호(Sequence Number)가 이용된다.' 정도로 표현을 수정합니다.

'TCP(Transmission Control Protocol)의 이해 3, 4, 5'에서 슬라이딩 윈도우 관련 내용은 개념적 설명 자료들을 참고하여 작성되어 사용하는 기술과 상황에 맞게 재해석되어 구현된다는 점을 숙지하지 못하여 발생한 실수입니다. TCP 헤더의 윈도우 사이즈는 전송 가능한 데이터의 바이트 수를 표현하는 값이기에 패킷의 수는 이 바이트 수에 영향을 받으며, 시퀀스 번호도 실제 구현에서 전송된 바이트 수와 관계되기에 패킷을 직접적으로 구분한다기보다 구분을 위해 이용되는 값입니다.

표 1. 윈도우 사이즈 관련 수정 내용

TCP는 UDP와 비교하며 안정적인 전송을 보장하는 프로토콜이라 알려져 있습니다. TCP에는 다음과 같은 세 가지 안정적 전송을 보장하는 기술이 있으며 이번 편의 주제가 바로 이 기술들입니다.

① 흐름제어 (Flow Control)　② 오류제어 (Error Control)　③ 혼잡제어 (Congestion Control)

이전 내용을 통해 소개된 Go-Back-N ARQ, Selective Repeat ARQ가 흐름제어 및 오류제어와 관련이 있어서 지금 소개드리

는 내용이 익숙하게 느껴지실 것 같습니다. 그러면 우선 흐름제어의 개념을 설명하고 이를 통해 Go-Back-N ARQ, Selective Repeat ARQ가 어떻게 이와 연결되는지 알아보도록 하겠습니다.

휴대폰과 사무용 PC, 사물인터넷 기기 등 인터넷에는 수많은 데이터가 흐르고 있고 이 데이터들은 많은 경우 TCP/IP 프로토콜을 이용하여 전송되고 있습니다. 이렇게 복잡한 네트워크에서 데이터를 보내는 디바이스와 받는 디바이스는 서로 다른 하드웨어와 소프트웨어로 이루어져 있을 가능성이 높고, 이로 인해 처리속도도 서로 차이가 납니다. 데이터를 보내는 디바이스와 받는 디바이스가 서로 처리속도가 다를 경우에 어떤 현상이 일어날 수 있는지 다음 두 가지로 나누어 살펴보겠습니다.

① 데이터 수신 디바이스 처리속도 ≥ 데이터 송신 디바이스 처리속도
② 데이터 수신 디바이스 처리속도 < 데이터 송신 디바이스 처리속도

우선 데이터 수신 디바이스의 처리속도가 데이터 송신 디바이스의 처리속도보다 빠르다면 데이터 통신이 순조롭게 잘 이루어질 것입니다. 데이터를 보내는 속도보다 처리하는 속도가 더 빠르니 수신 디바이스의 버퍼에 데이터가 쌓일 틈이 없기 때문입니다. 하지만 이 반대의 경우 즉, 데이터 수신 디바이스의 처리속도가 데이터 송신 디바이스의 처리속도보다 느리다면 어떻게 될까요?

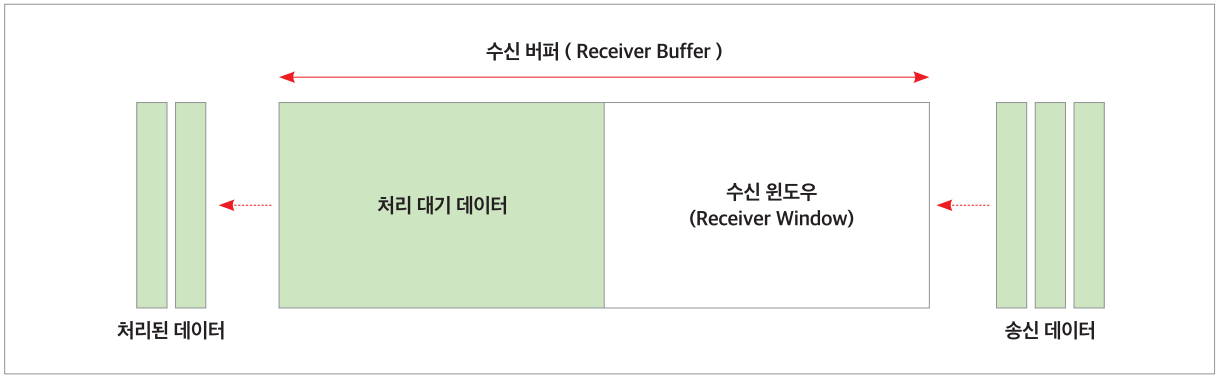

그림 1. 수신 버퍼와 수신 윈도우

[그림 1]을 통해서 송신 디바이스보다 느린 처리속도를 가진 수신 디바이스에서 일어나는 현상을 짚어보겠습니다. [그림 1]과 같이 수신 디바이스는 송신된 데이터를 우선 수신 버퍼에 받아 놓습니다. 이때 수신 디바이스가 초당 처리할 수 있는 패킷의 수는 2개이며, 송신 디바이스는 초당 3개의 패킷을 보낸다고 가정하겠습니다. 편의를 위해 패킷의 데이터양은 모두 같다고 하겠습니다. 초당 세 개의 패킷이 버퍼에 들어오고, 이 중 두 개가 빠져나가니 1초마다 처리 대기 데이터에 해당하는 패킷 수는 한 개씩 증가합니다. 만약 수신 버퍼에 100개의 패킷을 저장할 수 있다면 100초 후면 수신 버퍼는 가득 차고, 이후에 들어오는 패킷은 수신 버퍼에 저장되지 못하고 버려지게 됩니다. 이런 수신 버퍼의 부족으로 인한 통신실패를 방지하기 위한 것이 흐름제어입니다.

• 흐름제어

Go-Back-N ARQ, Selective Repeat ARQ에서 설명된 슬라이딩 윈도우와 연계하여 흐름제어가 어떤 방식으로 동작하는지 [그림 2]를 통해 단순한 예를 들어보겠습니다. 우선 앞서 말씀드린 바와 같이 수신 디바이스의 버퍼가 넘치지 않도록 송신 디바이스의 데이터 전송을 조절하는 것이 흐름제어입니다. 송신 디바이스가 수신 디바이스의 버퍼 잔량([그림 1] 수신 윈도우)을 알 수 있는 방법은 수신 디바이스에서 버퍼 잔량을 지속적으로 피드백해주는 것뿐입니다. 이 피드백은 송신 디바이스가 보낸 패

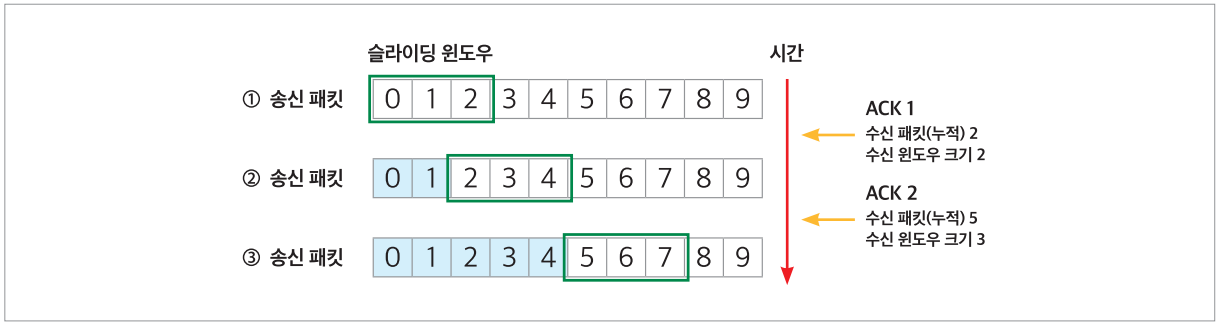

그림 2. 수신 윈도우와 슬라이딩 윈도우에 의한 흐름제어

킷들에 대해 수신 디바이스가 잘 받았다는 신호로 보내는 ACK 메시지의 TCP 헤더 내부 Window Size 필드를 통해 가능합니다. 즉 16비트로 된 Window Size를 통해 현재 수신 버퍼에서 수용 가능한 데이터의 바이트 수를 알려주는 것입니다.

처음 TCP 핸드쉐이킹 시에 ACK 메시지를 통해서 수신 윈도우의 크기(설명의 편의를 위해 헤더에 실리는 바이트 수가 아닌 [그림 2]의 패킷 수로 하겠습니다. 패킷의 크기는 모두 같다고 가정합니다)를 3이라고 송신 디바이스에 알려줍니다. 수신 디바이스의 버퍼가 3개 패킷을 받을 수 있으므로 송신 디바이스는 ACK 없이 우선 세 개의 패킷을 보냅니다. 이것이 [그림 2]의 ①의 상황입니다. ①과 같이 송신된 데이터를 받은 수신 디바이스는 이 중 2개를 버퍼에서 꺼내어 처리하고 잘 받았다는 ACK 메시지(ACK 1)를 보냅니다. 이때 ACK 메시지의 TCP 헤더에 포함된 확인응답 번호(Acknowledgement Number : 수신 처리된 데이터의 누적 바이트량을 알 수 있음) 및 윈도우 사이즈(Window Size)를 통해 송신 디바이스는 패킷 0, 1의 정상 처리 및 수신 버퍼의 잔량을 알 수 있습니다. 수신 디바이스의 버퍼에 2개 패킷의 여유가 있음을 안 송신 디바이스는 ②와 같이 슬라이딩 윈도우를 이동하여 2개 패킷(3번, 4번)을 추가로 전송하고 ACK 메시지를 기다립니다(처리 확인된 패킷은 푸른색으로 표시). 잠시 후 수신 디바이스가 보낸 두 번째 ACK 메시지(ACK 2)의 확인응답 번호를 통해 현재 5개까지의 패킷(송신패킷 0, 1, 2, 3, 4)이 정상처리 되었고 수신 버퍼는 3개 패킷의 여유가 있다는 것을 확인한 후 ③과 같이 슬라이딩 윈도우를 오른쪽으로 이동하여 3개 패킷(패킷 5, 6 ,7)을 보냅니다. 이렇듯 지속적으로 수신 디바이스 버퍼의 잔량을 확인하며 버퍼가 넘치지 않도록 데이터를 송신하는 것이 TCP의 흐름제어입니다.

• 오류제어

지금까지 간략하게 흐름제어에 대해 알아봤습니다. 그렇다면 오류제어는 어떻게 이루어질까요? 사실 흐름제어에 관한 설명에서 슬라이딩 윈도우를 보시며 지난 연재를 통해 설명된 Go-Back-N ARQ, Selective Repeat ARQ에서 이미 흐름제어에 관한 개념이 소개되었다는 인상을 받으셨을 것 같습니다. 맞습니다. Go-Back-N ARQ, Selective Repeat ARQ에 대한 설명에 이미 흐름제어에 관한 내용이 담겨 있었고, 오류제어에 관한 내용도 함께 담겨 있었습니다. TCP의 오류제어는 제대로 송신되지 않거나 받은 후 이상이 발견된 패킷들에 대해 재송신을 요구하는 ARQ(Automatic Repeat reQuest)에서 이미 그 내용이 소개되었으며 추가로 소개해 드릴 내용은 없습니다. 용두사미처럼 거창하게 시작해서 너무 싱겁게 끝나가는 느낌인가요? 하지만 끝날 때까지 끝난 것이 아니라는 말처럼 아직 혼잡제어가 남아 있습니다. 혼잡제어는 흐름제어와 오류제어와 같이 기존의 연재를 통해서 소개된 개념이 아닌 새로 소개되는 개념입니다. 이에 관해서는 다음 편에서 설명을 이어가겠습니다.

P.S.

C군이 여러분께 전하는 내용 중 전문적 성격이 짙은 것은 엄밀한 언어를 사용하여 설명하는 데는 한계가 있습니다. 본 내용은 설명하는 대상에 대한 전체적 맥락의 이해에만 이용하시고, 그 이상은 권위 있는 전문 자료를 참고하시기 바랍니다.