# Selective Repeat이 다음과 같이 실행 되었을 때, ack2를 sender가 수신 후 어떻게 동작 할 지 그림 및 글로 설명하시오.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Sender Window Base : 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Sender Window Base : 6

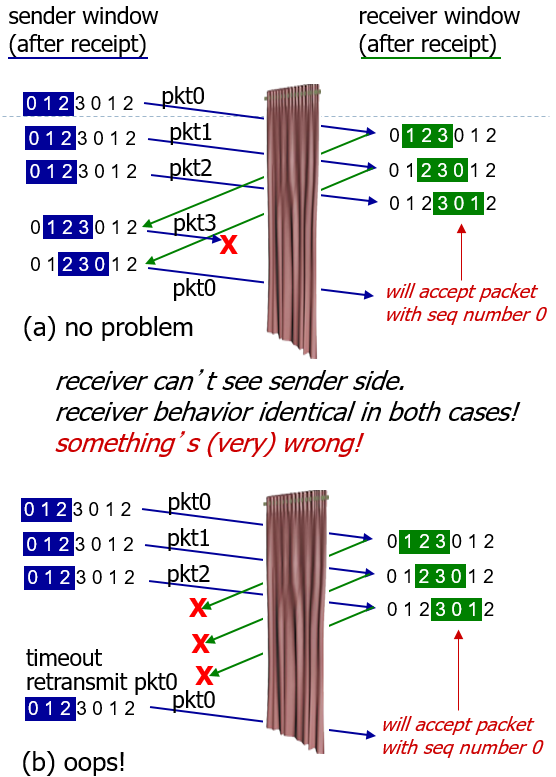
Sender은 Ack2를 수신 후 window의 ack3, 4, 5이 Acked인 것을 확인 하고 base를 6으로 이동한다.

# 스트리밍, 인터넷 전화 같은 어플리케이션에서 TCP가 아닌 UDP를 사용하는 이유

UDP는 “Best Effort”만을 제공함으로 TCP에 비해 오버헤드가 적고 핸드 셰이킹을 하지 않기 때문에 빠른 송수신이 가능하다. 스트리밍이나 인터넷 전화 같은 경우에는 데이터의 손실이 나더라도 손실을 복구하는 것 보다 다음 데이터를 받는 것이 중요하기 때문에 UDP도 사용한다.

TCP의 경우 신뢰성을 보장하기 때문에 이미 지나간 데이터 손실에 대한 처리가 필요하기 때문에 실시간에는 UDP가 유리하다고 볼 수 있다. 또한 UDP에 비해 오버헤드가 크고 핸드 셰이킹도 필요하기 때문에 전송 속도가 느린 점도 있기 때문에 스트리밍, 인터넷 전화같은 어플리케이션에서는 UDP가 더 유리하다.

# Selective repeat dilemma를 해결할 수 있는 방안을 제시하고 해결방안이 아래의 selective repeat dilemma 상황에서 어떻게 동작해 문제를 해결하는지 설명하시오.



해결법 : seq의 수를 window size의 두배 크기이상 가지게 선언한다. 이렇게 정하게 되면 window size를 n이라고 두고 최악의 경우인 n개의 time out이 난다면 2\*n까지 닿게 되므로 Selective repeat dilemma 문제가 발생하지 않게 된다.

Pk0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

Pk1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

Pk2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

Pk0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 |