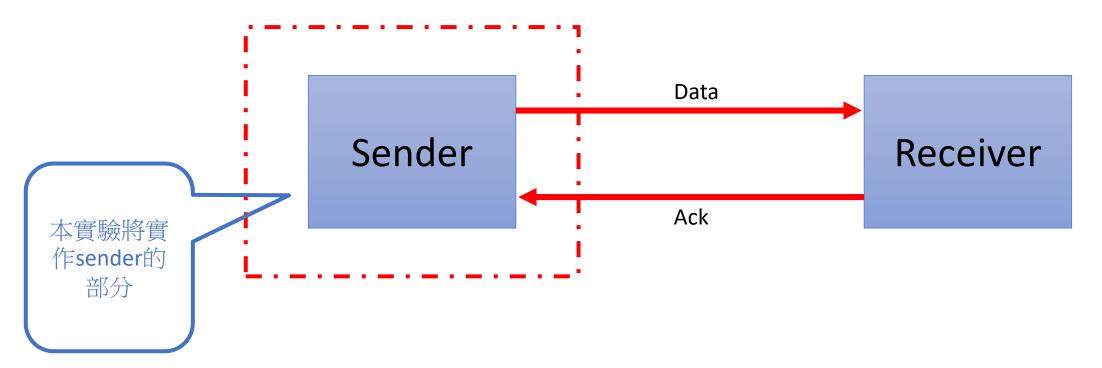
# Lab 2: Go-Back-N RDT

HackMD: https://hackmd.io/@KentShen/H1pxm9LVc

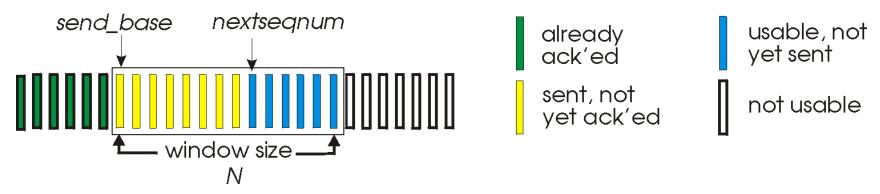
# 實驗場景

- 由sender端傳data到receiver,接著receiver會回應ACKs
- Receiver端運行go-back-n,並會檢查是否收到所有data
- Python 3.8.10



#### Go-Back-N: sender

- k-bit seq # in pkt header
- "window" of up to N, consecutive unack' ed pkts allowed



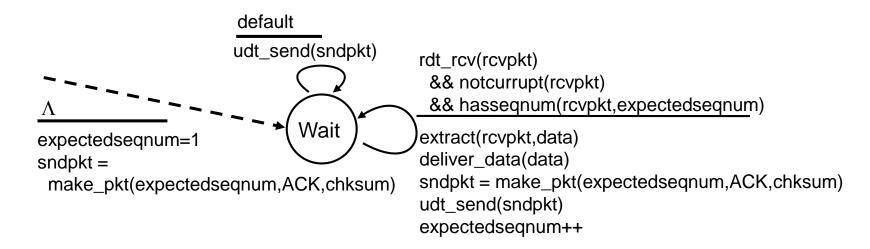
- ACK(n): ACKs all pkts up to, including seq # n -"cumulative ACK"
  - may receive duplicate ACKs (see receiver)
- timer for oldest in-flight pkt
- timeout(n): retransmit packet n and all higher seq # pkts in window

Transport Layer 3-3

#### GBN: sender extended FSM

```
rdt_send(data)
                       if (nextseqnum < base+N) {</pre>
                          sndpkt[nextseqnum] = make_pkt(nextseqnum,data,chksum)
                          udt_send(sndpkt[nextseqnum])
                          if (base == nextseqnum)
                            start_timer
                          nextseqnum++
                       else
   Λ
                        refuse_data(data)
   base=1
  nextsegnum=1
                                           timeout
                                           start timer
                             Wait
                                          udt_send(sndpkt[base])
                                          udt_send(sndpkt[base+1])
rdt_rcv(rcvpkt)
 && corrupt(rcvpkt)
                                          udt_send(sndpkt[nextseqnum-1])
                         rdt_rcv(rcvpkt) &&
                           notcorrupt(rcvpkt)
                         base = getacknum(rcvpkt)+1
                         If (base == nextseqnum)
                            stop_timer
                          else
                           start_timerort Layer
```

#### GBN: receiver extended FSM



# ACK-only: always send ACK for correctly-received pkt with highest *in-order* seq #

- may generate duplicate ACKs
- need only remember **expectedseqnum**
- out-of-order pkt:
  - discard (don't buffer): no receiver buffering!
  - re-ACK pkt with highest in-order seq #

Transport Layer 3-5

### Sender端設定

```
send_base = 0 → send_base指標位置
next_seq_num = 0 → next_seq_num位置
cwnd_size = 3 → window的大小為3
num_pkt = 10 → 總共要送出10個封包
```

### Sender端需完成功能

- Send\_base及next\_seq\_num要在正確的位置
- Window內的data需接連送出
- 設定timer的時間為5秒
- Timeout時需要重傳window內所有的data
- Sender結束前需送出10筆資料

# 成果繳交

• 繳交檔案:sender.py

• 繳交時間:May 5, 2022