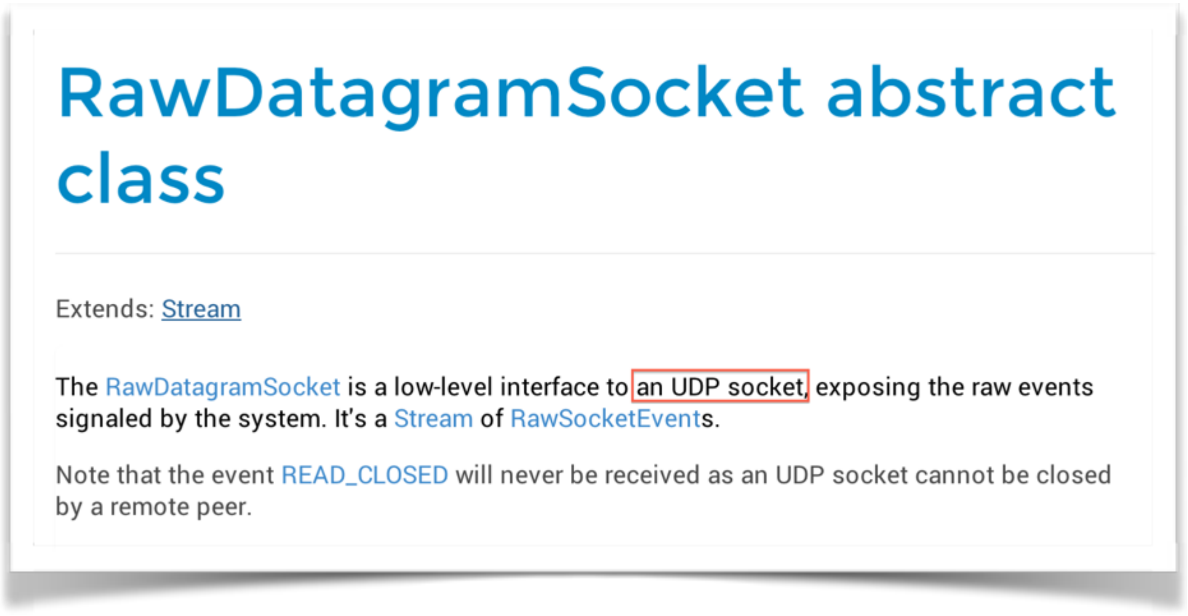
Computer Network hw2 Report
b01902102 資工三 葉子瑄

1. I am not using C, but i do use UDP socket



RawDatagramSocket abstract class

Extends: Stream

The RawDatagramSocket is a low-level interface to an UDP socket, exposing the raw events signaled by the system. It's a Stream of RawSocketEvents.

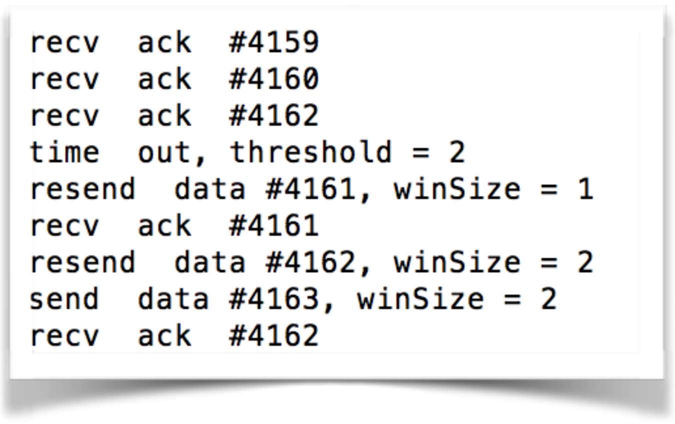
Note that the event READ_CLOSED will never be received as an UDP socket cannot be closed by a remote peer.

2.Execution instruction
a.cd b01902102_hw2/
b "pub get"
c. cd bin/
d. "dart b01902102_hw1_send.dart", "dart b01902102_hw1_recv.dart", "dart b01902102_hw1_agent.dart", run three programs on each terminal.It is fine to ran them in any order.

3.What you do, and how you do it
packet:
a. List of integer (each int is 7-16 bit)
b. [type, position 5, 6 of sn(_ _XXXX), position 3. 4 of sn, position 1, 2 of sn, targetIp's 1st number, targetIp's 2nd number, target Ip's 3rd number, target Ip's 4th number, target port/100, target port%100, self Ip's 1st number, self Ip's 2nd number, self Ip's 3rd number, self Ip's 4th number, self port/100, self port%100, data(length:100).....]

send:
a. bind at 127.0.0.1
b. read the file, determine how many part the file should be cut(N). new a list of size N, and filed with false, to check whether the whole file is sent, and to compute the which part of file should be send in next window.
c. start listener, when it receive ack:
    1. check whether the sn of ack is in current window.
    2. if not, ignore it. (to avoid write to unexpected memory area)
    3. Otherwise, cancel timer, flag sent for current window(to check whether the window is done), and all file(to check whether the whole file is sent, and to compute the which part of file should be send in next window).
    4. to check whether the sent packets in current window receive ack. if so, adjust the convection control, move the point to point to next package should send. and call _sendWindow().
d. call _sendWindow()
e. in _sendWindow():

1. new timer list and flag list(to check which received ack)
2. a for loop to send packet in current window
3. check whether sn is bigger the last packet, if so new Timer that do nothing when wake, and flag received ack(to avoid write to unexpected memory area)
4. Otherwise, send packet(use _current and _last to know whether is retransmission). And then, add new to list of timers, when send each package
5. when timer wake(time out), cancel other timer, adjust convection control, call _sendWindow().



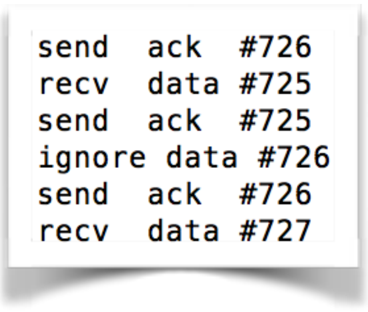
```
recv   ack   #4159
recv   ack   #4160
recv   ack   #4162
time   out, threshold = 2
resend  data #4161, winSize = 1
recv   ack   #4161
resend  data #4162, winSize = 2
send   data #4163, winSize = 2
recv   ack   #4162
```

receiver:

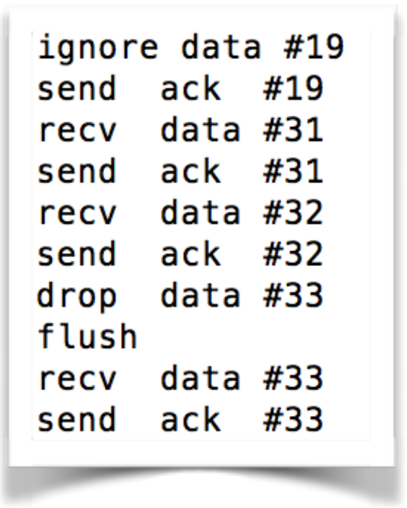
a. if the file with the filename, delete file, and then create file with the filename.

b. start listener:

1. if recv FIN, cancel the listen, close program
2. else if recv ACK, ignore it.
3. else(recv PSH), push to buffer(call buffer.push(data, sn)).
4. if already write to file or in buffer, return 0. else if buffer is full or should not accept in current buffer(e.g. recv 34, when buffer only accept 1-32) return -1, else, push to buffer, buffer._filled++(how may package in buffer), index[sn%size] = true and then return 1.
5. if return value is 1 || 0, send ack. else (-1), drop the packet, and judge whether the buffer is full. if full, flush(call buffer.popAll() and write to file). Otherwise, do nothing.
6. when call buffer.popAll(), buffer._filled = 0, buffer._round = 0(to present 1-32, 33-64 or…. should accept), all in _indexes = false(present which index has store packet);

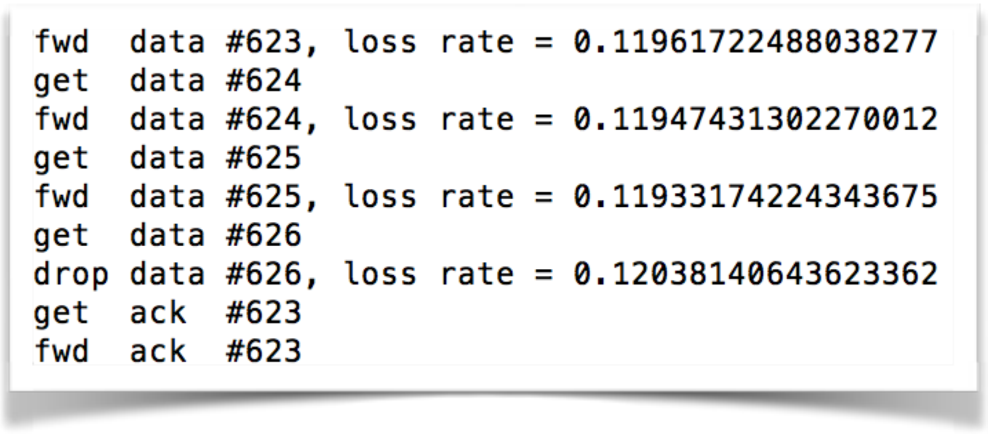


```
send   ack   #726
recv   data #725
send   ack   #725
ignore data #726
send   ack   #726
recv   data #727
```



```
ignore data #19
send   ack   #19
recv   data #31
send   ack   #31
recv   data #32
send   ack   #32
drop   data #33
flush
recv   data #33
send   ack   #33
```

agent:
a. initial with recvNum = 0, dropNum = 0,
b. parse received packet,
c. if is type of PSH, recvNum++, than random an double, if the double small then 0.1, drop the packet. otherwise forward it, dropNum++.
d. if type != PSH, forward it.



```
fwd  data #623, loss rate = 0.11961722488038277
get  data #624
fwd  data #624, loss rate = 0.11947431302270012
get  data #625
fwd  data #625, loss rate = 0.11933174224343675
get  data #626
drop data #626, loss rate = 0.12038140643623362
get  ack  #623
fwd  ack  #623
```

4.Challenging issues and solutions

a. how to determine which packet should sent in next window when sb time out.
   -> at first after recv a ack, I will set _lastestSn = sn when last = an-1, but it would have problem when recv 5 3 4. Finally, I use a list of size N to flag which recv ack, and go through it to find the _lastestSn.

b. call sendWindow in for loop? but I don't know how to call asynchronism function in in for loop, when I don't know how many loop.
   -> call send window when timeout or recv ack.

c. sequence Number and port overflow.
   -> parse sequence number to three part i.e. (sn/10000).floor(), ((sn%10000)/100).floor, and sn %100
   -> parse port to two part. i.e (port/100).floor(), port % 100