Report

b04902027 資工三 陳昇

1. Execution

$ python2 sender.py

$ python2 agent.py

$ python2 receiver.py

2. Program structure

(1) sender

winSize = 1

threshold = 16

base = 1

last = base + winSize

next = 1

send(fin)

recv(finack)

yes

base > lines

no

if(next=base)

set timer

send(data[next])

next ++

yes

next < last

no

recv(ack)

base = ack\_num + 1

finack?

end

yes

no

timeout?

yes

no

next = base

threshold = max(winSize/2 , 1)

winSize = 1

last = base + winSize

base = last?

no

yes

winSize \*= 2

or winSize ++

(2) agent

fwd(ack)

ack

loss?

data

recv

no

no

finack?

fwd(data)

end

yes

(3) receiver

fin

recv

send(finack)

end

exp = 1

data

drop(data)

send(dup\_ack)

no

data\_i = exp?

yes

drop(data)

send(dup\_ack)

flush

yes

buff full?

no

buff(data)

send(ack)

exp++

3. Difficulties and Solutions

(1)

When implementing timeout, the process will be blocked at "recv" due to the fact that the socket is in blocking mode in default. It disables us to see whether it reaches the timeout or not. The solution is to change the socket into non-blocking mode, which helps us to get the right timing.

(2)

When implementing flush, the best way to make sure the tidiness of code is to packet the procedure into a function. By doing so, we have to be aware of every variable used in the function is set to be global.