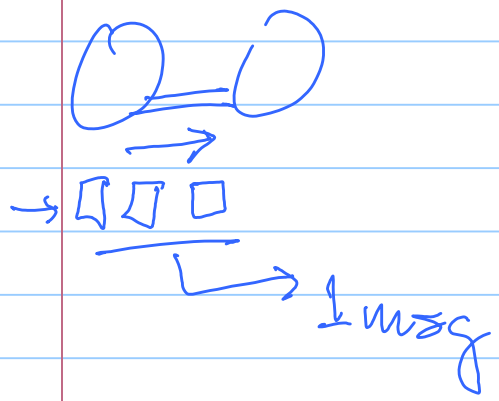
Linux — pipes & msg queues ×



↓ msg

local data passing

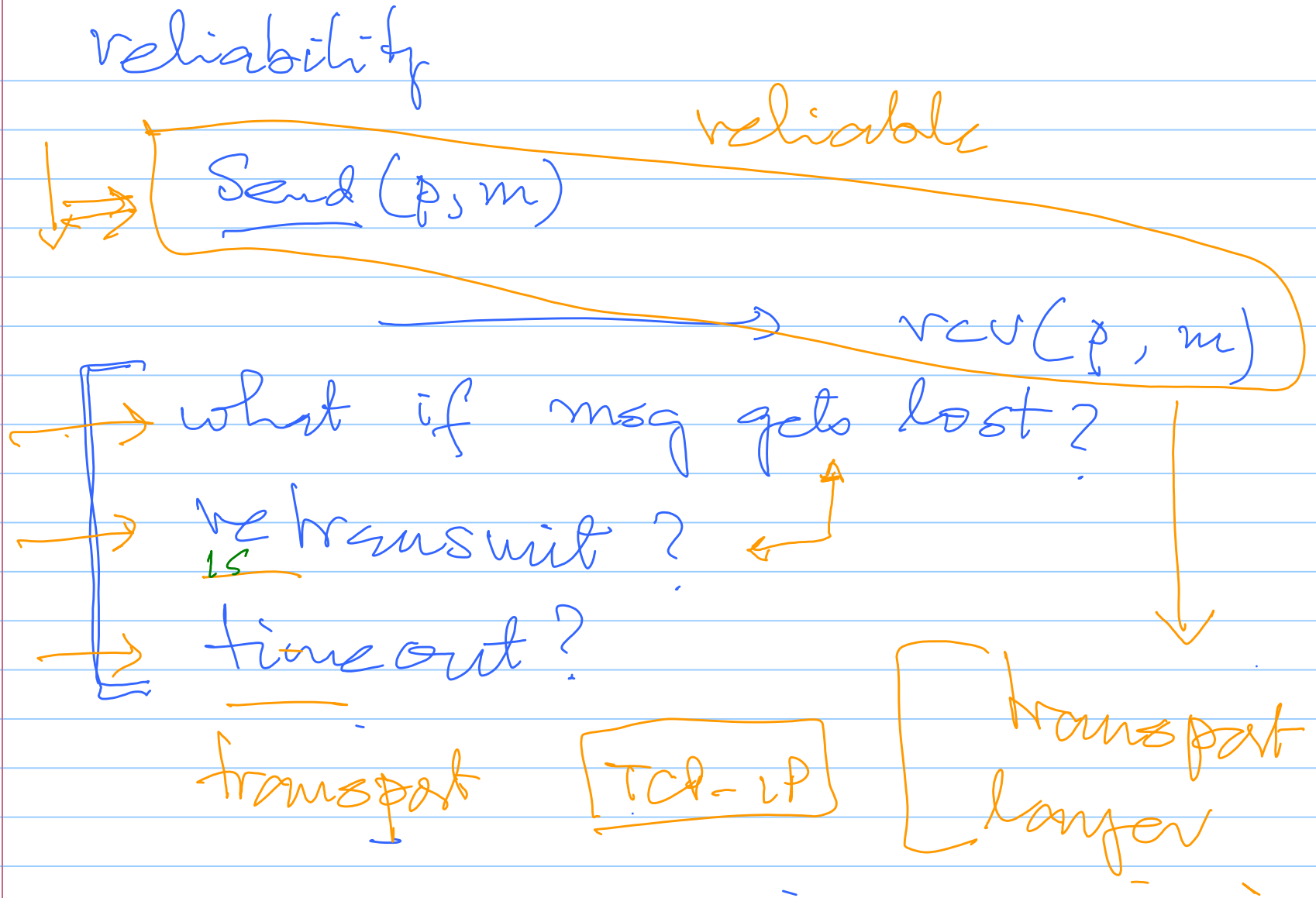
↓

message boundaries → NO

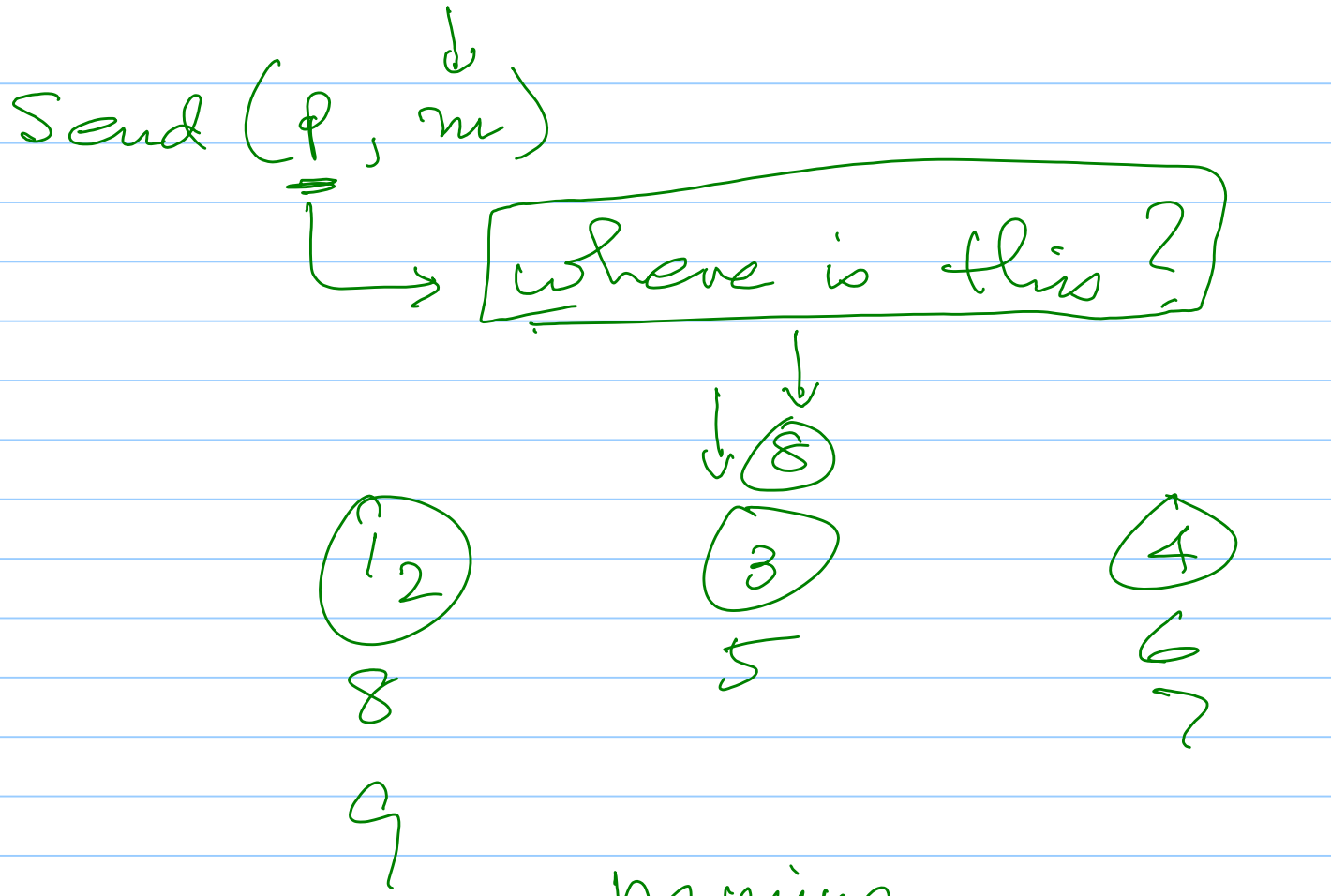
local

yes

↓


reliability
reliable
Send (p, m)
rcv (p, m)
what if msg gets lost?
retransmit ?
is
timeout ?
transport
TCP - IP
transport
layer

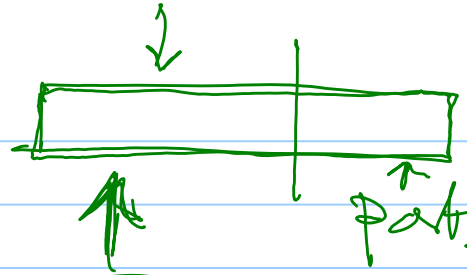
How to implement port mapping



naming

<name → location>

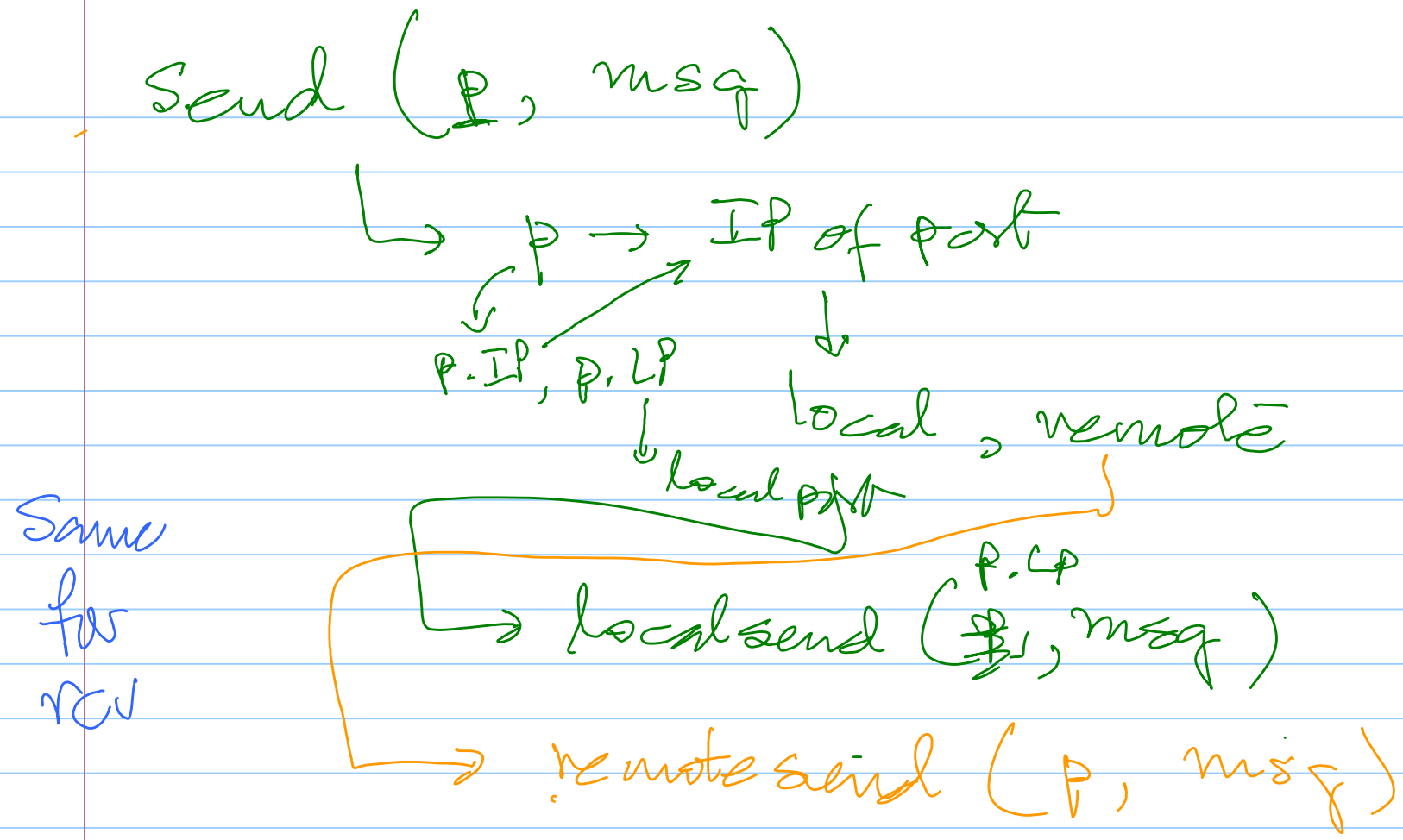
port → 2 parts →



IP address
(location)

∴ location transparency is no longer available . . .

local to the IP addr


send (p, msg)
p → IP of port
p.IP, p.LP
local, remote
local port
p.LP
localsend (p, msg)
remotesend (p, msg)
Same for rcv

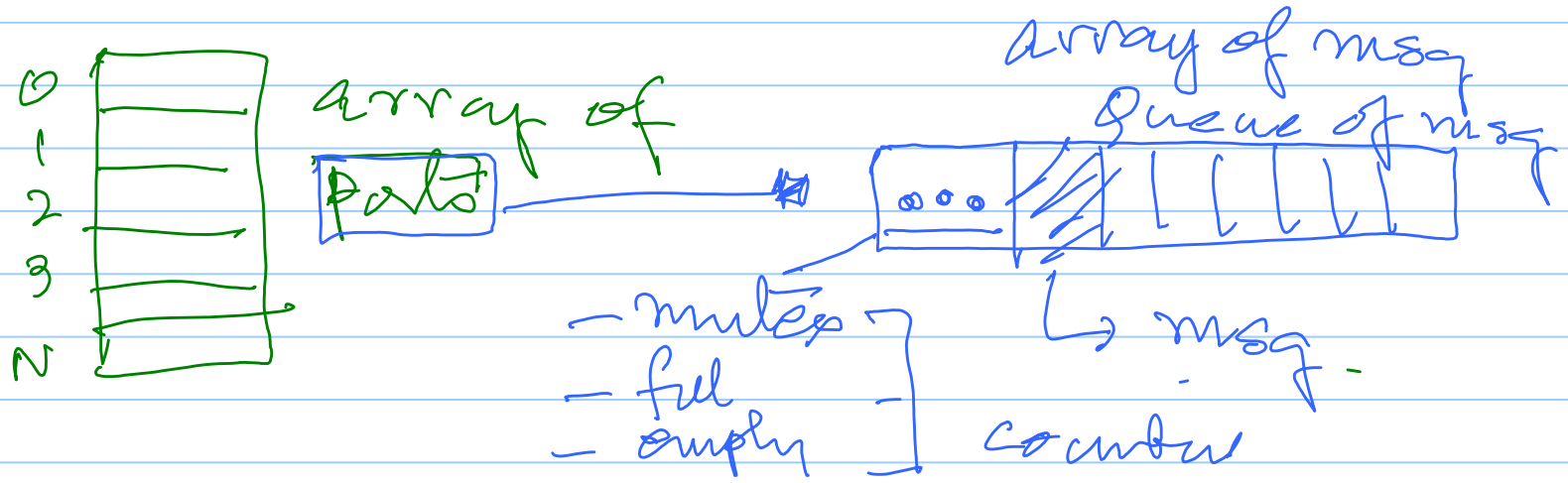
local send (p, msg)

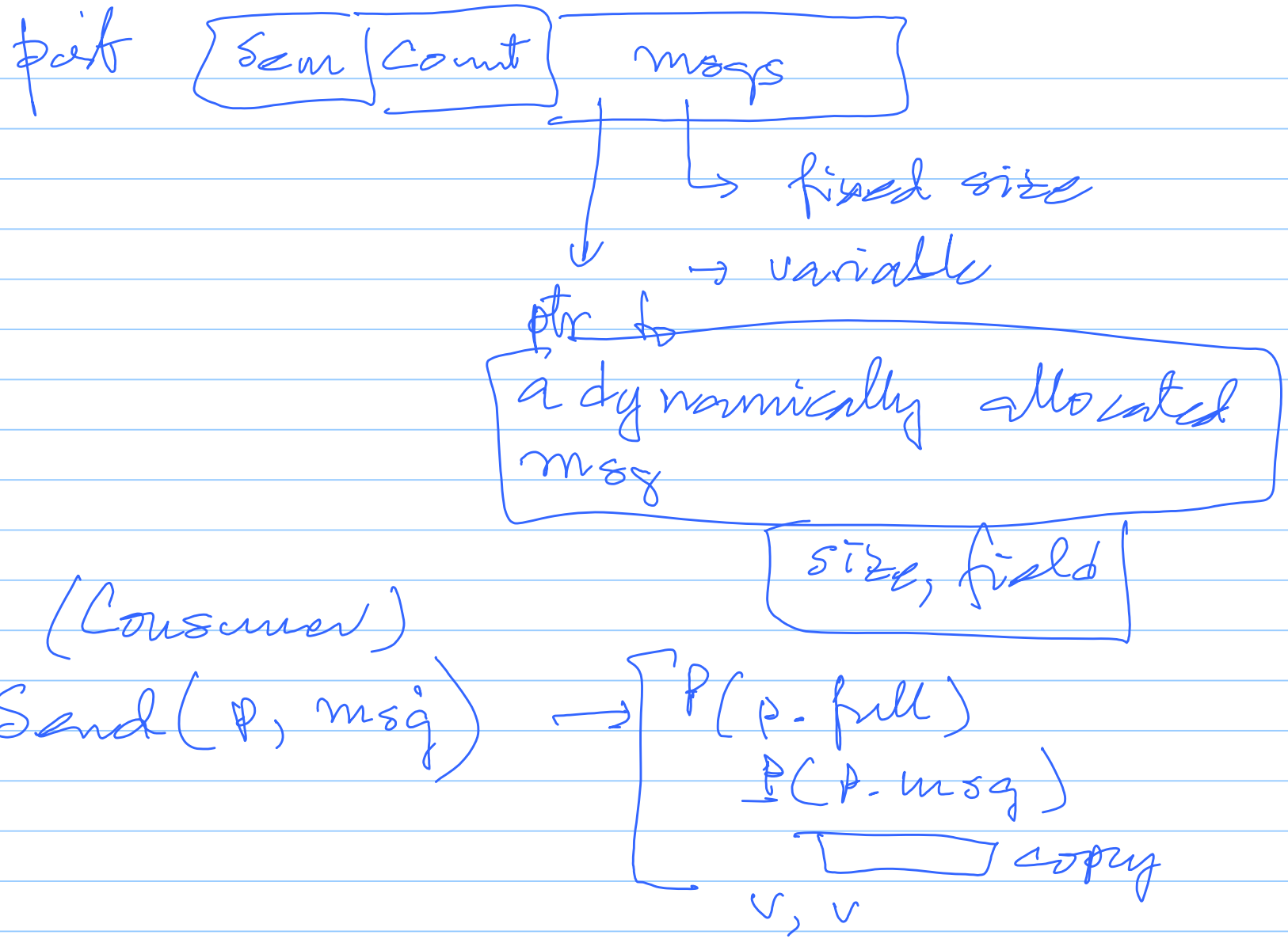
↳ local port

↳ number

port implementation

→ shared memory → kernel




pair
Sem
Count
msgs
→ fixed size
→ variable
ptr
a dynamically allocated msg
size, field
(Consumer)
Send(p, msg)
P(p.full)
P(p.msg)
copy
V, V

local send → prod

" rcv → cons

port → bounded buffer

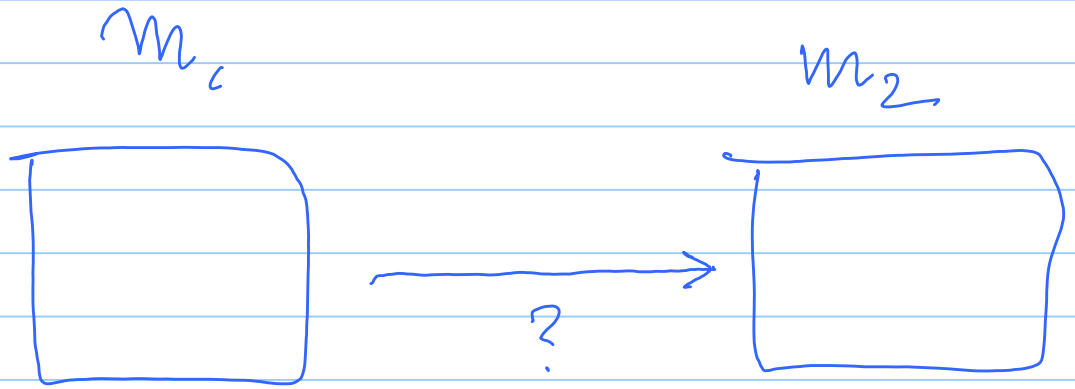
↳ sender do not block unless port full

∴ rcvr gets msg or blocks till msg available

multiple machines

remote send (p, msg)

[X] remote rcv (p, msg)



M_1 send to M_2

⇒ network send

→ network_send(IP, msg)

M_2 recvs

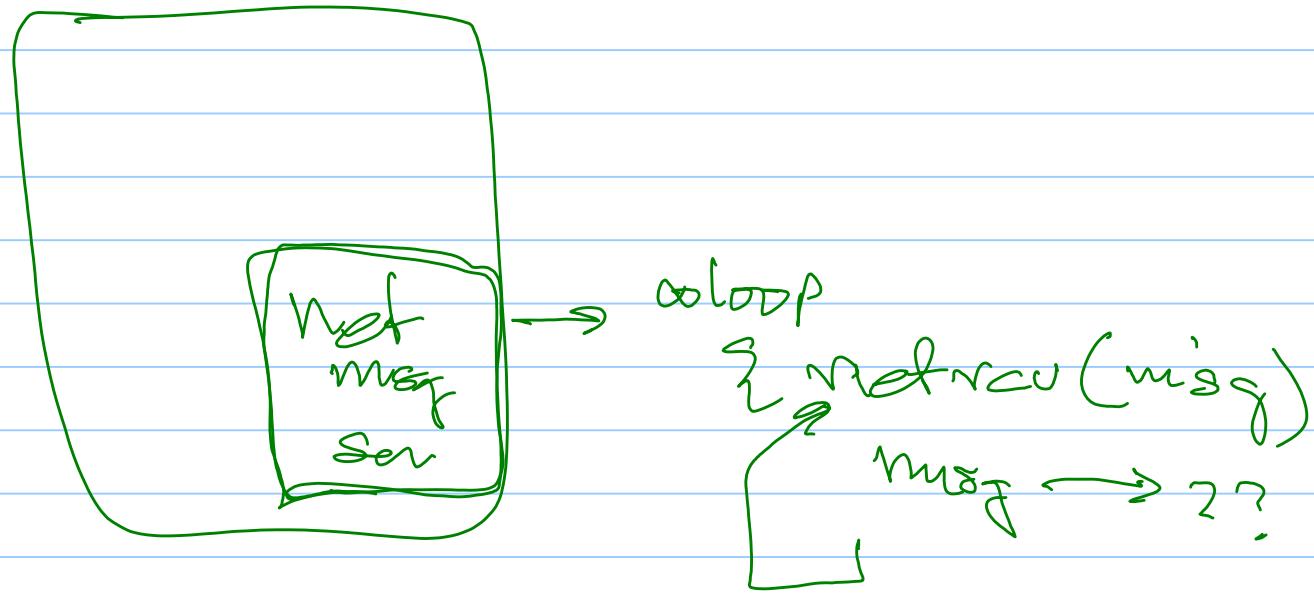
⇒ return (msg)

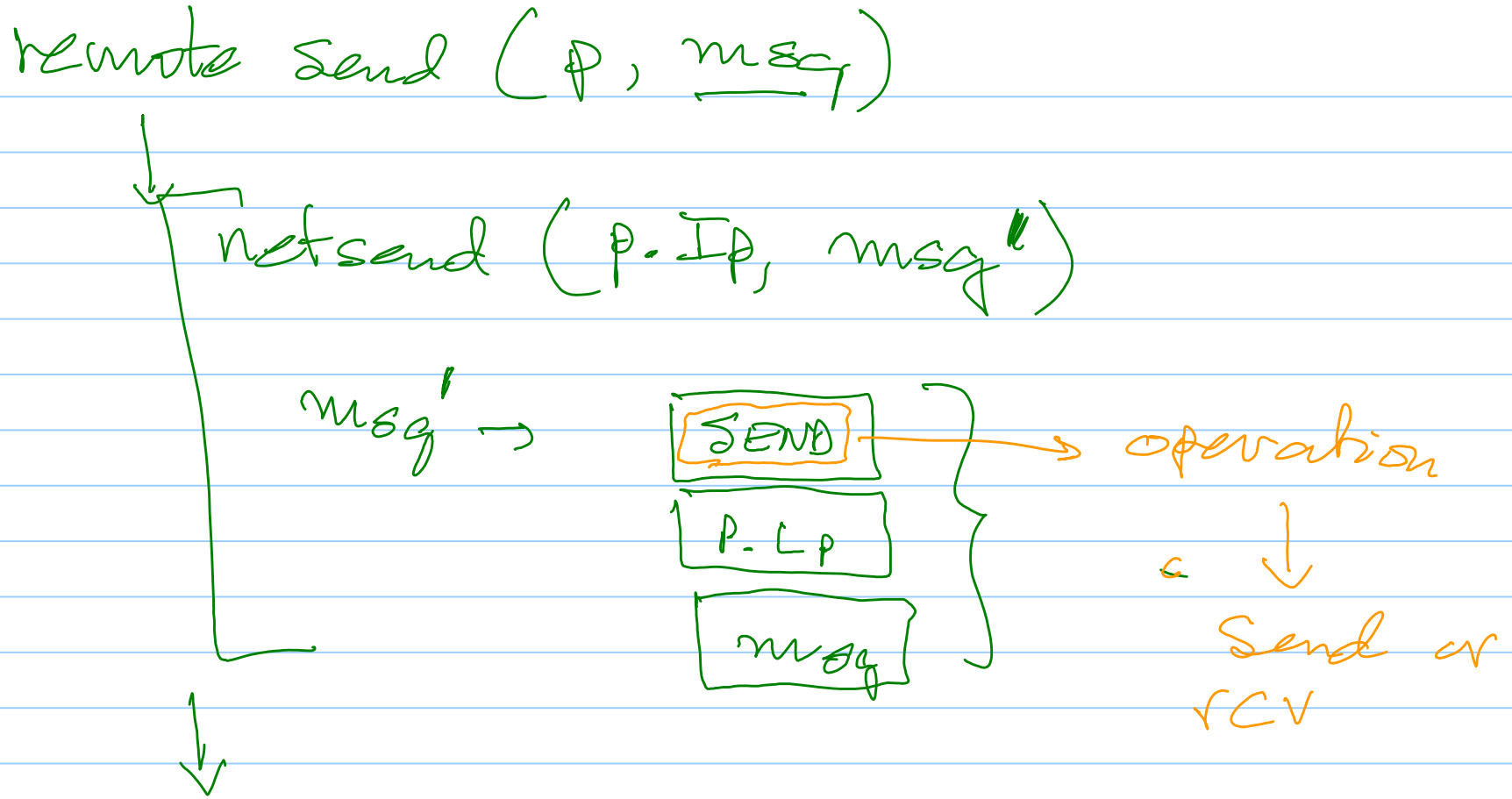
msg sent to recvs

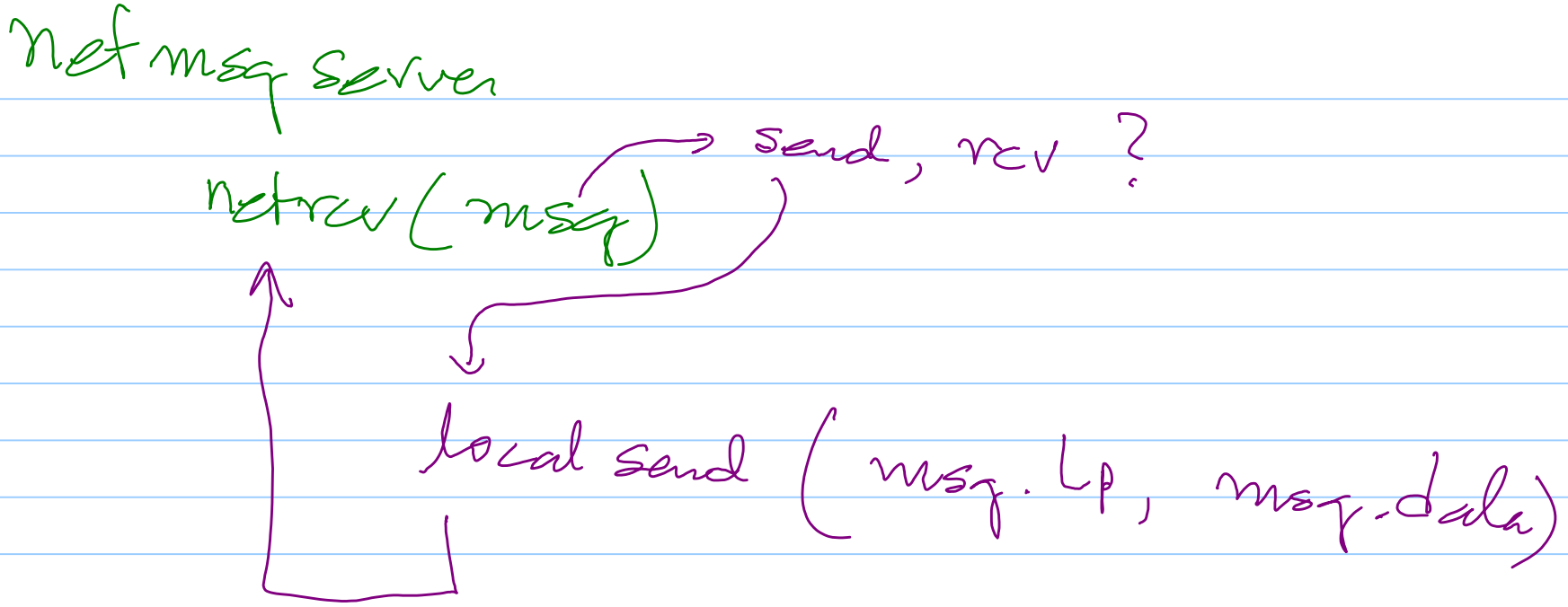
IP addr

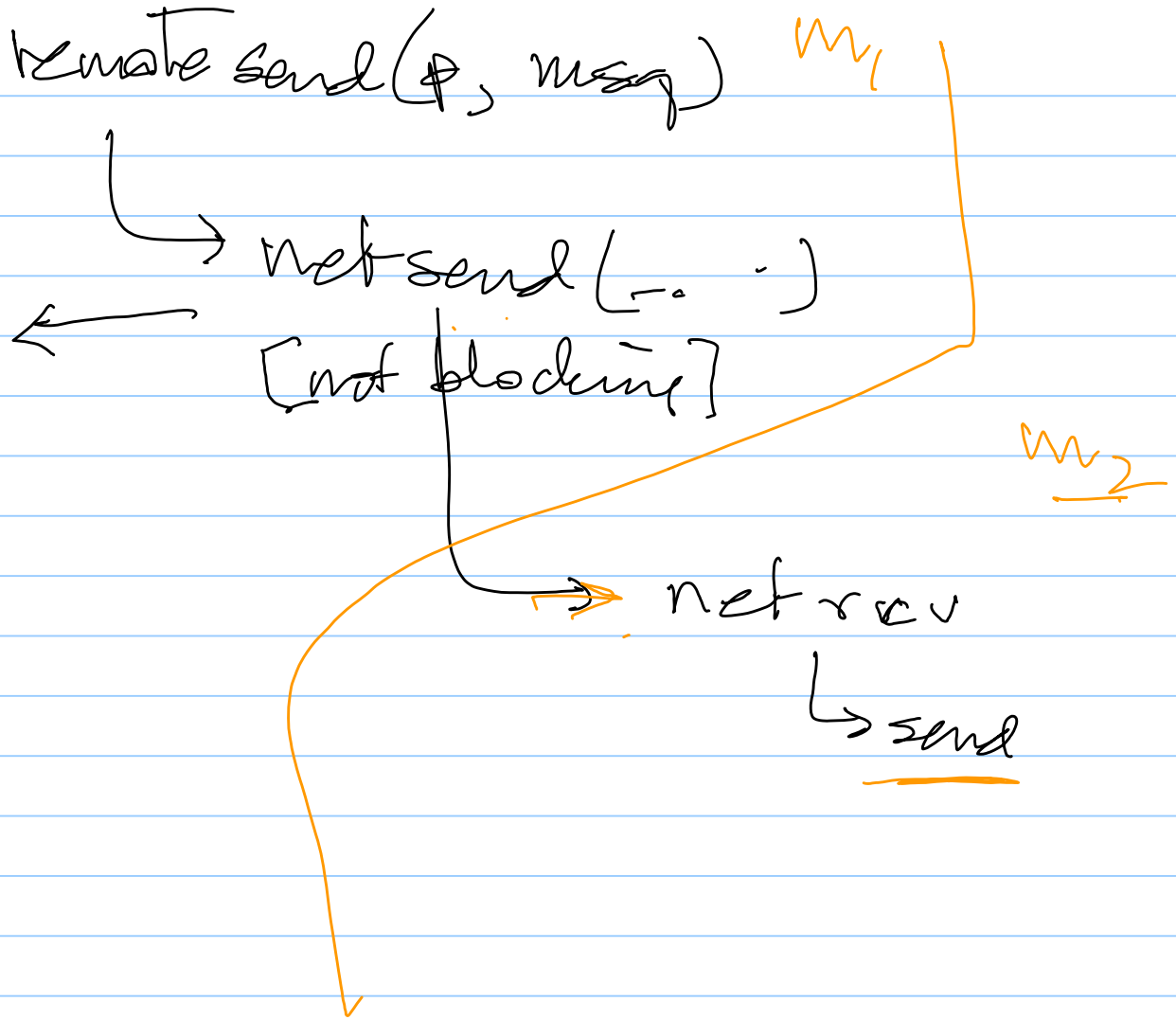
Network message server

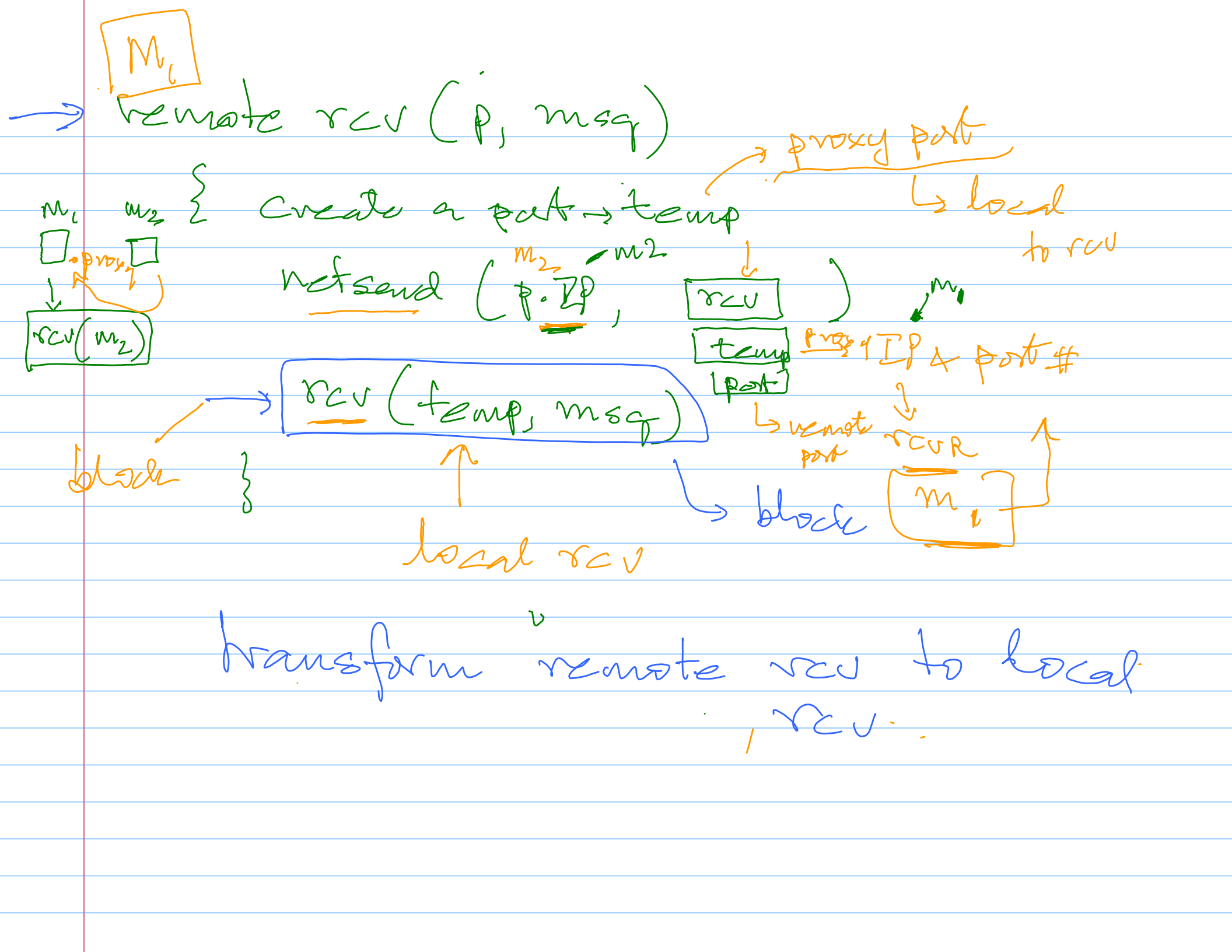
net msg server




remote send (p, msg)
netsend (p.Ip, msg')
msg' →
SEND
P.Lp
msg
operation
Send or
rcv

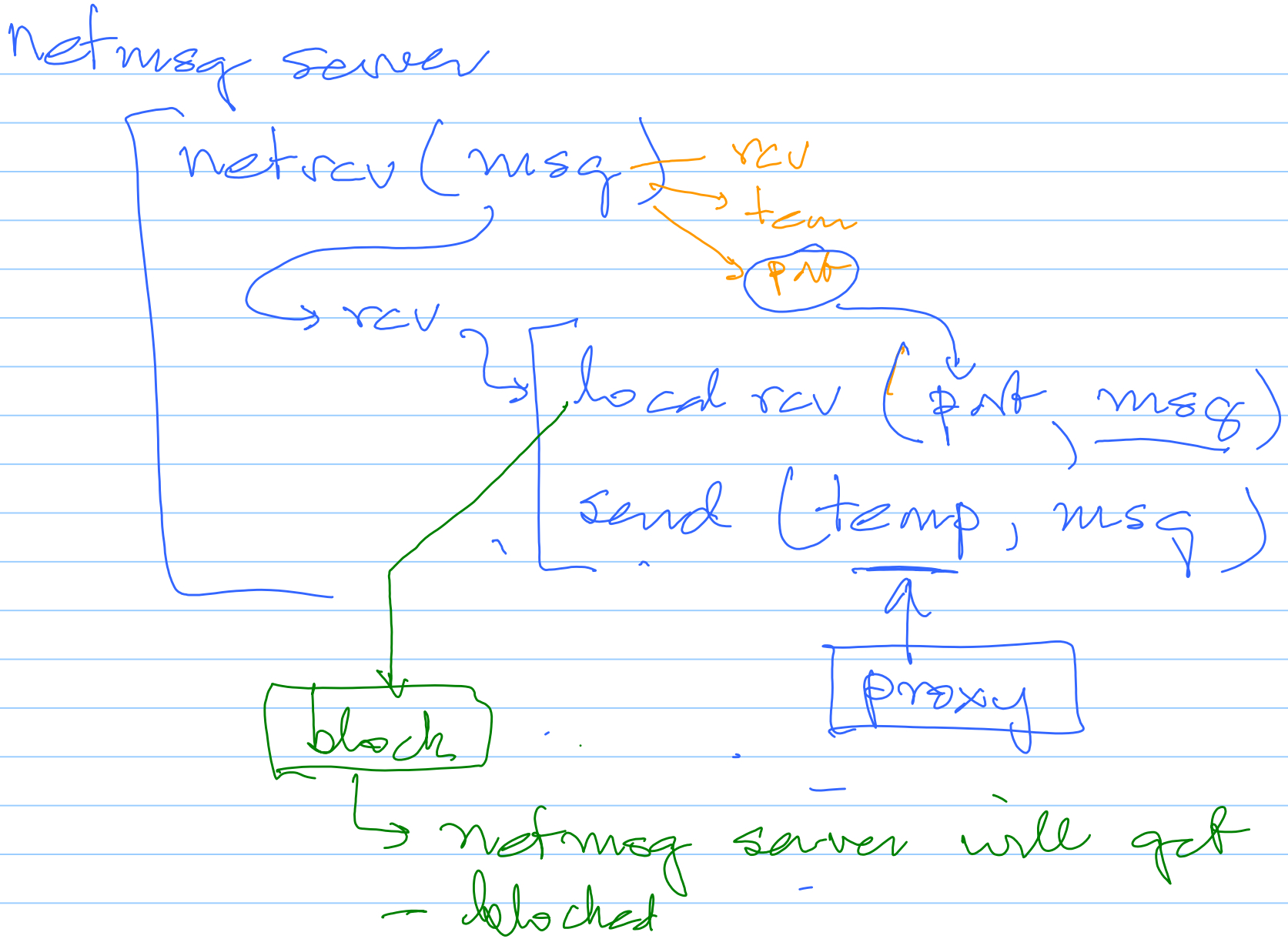

netmsg server
netrcv(msg)
send, rcv ?
local send (msg.lp, msg.data)

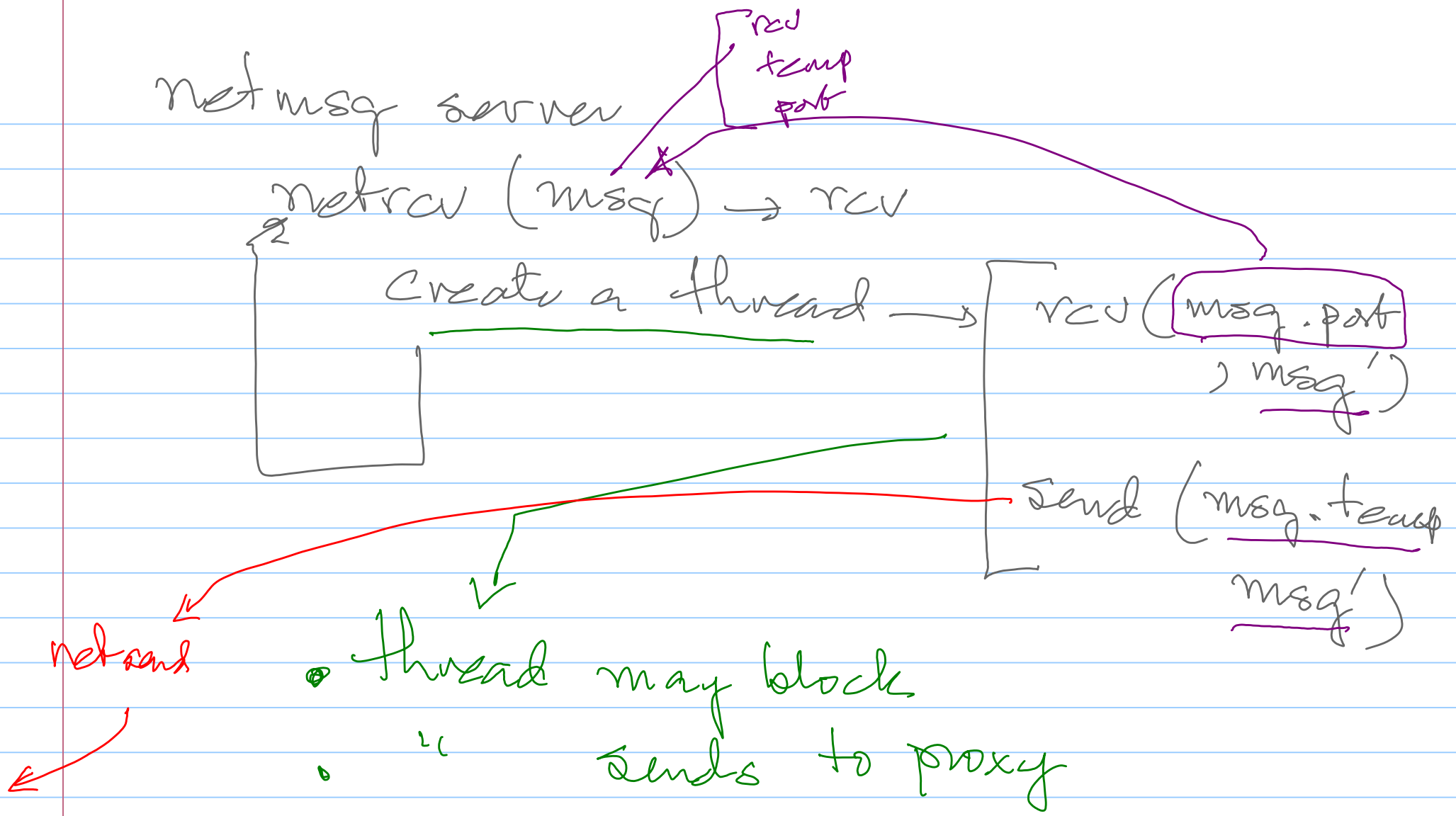

remote send(p, msg)
net send(...)
[not blocking]
net rcv
send
M1
M2


M_1
remote rcv (p, msg)
{ create a port → temp
proxy port
local to rcv
M_1 M_2
proxy
rcv(M_2)
netsend (p.IP, M_2, M_2)
rcv
temp
port
M_1
proxy IP & port #
remote port
rcv (temp, msg)
block
}
local rcv
block
RCVR
M_1
transform remote rcv to local rcv.

Send does not block

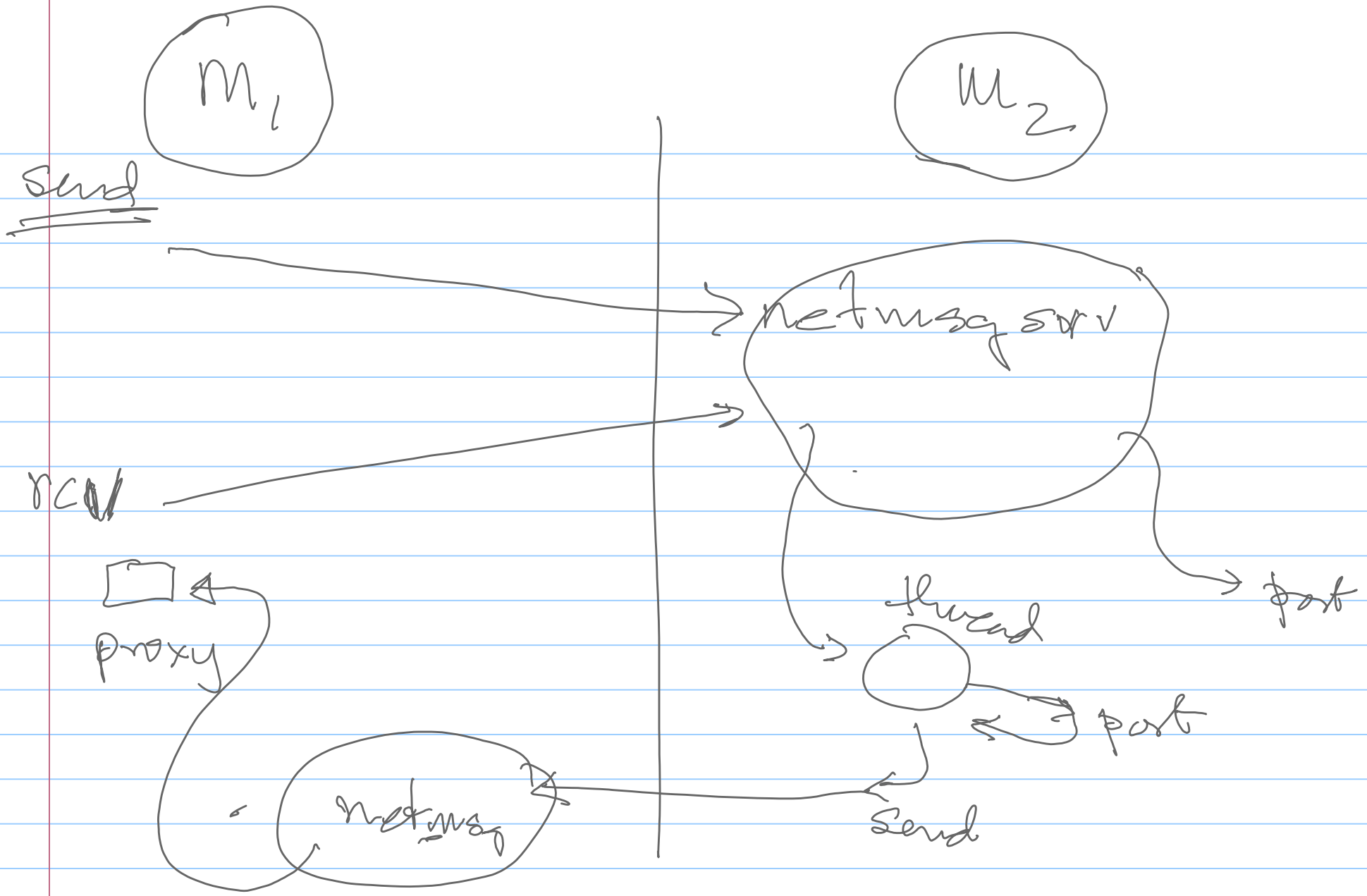
rcv blocks


netmsg server
netrcv (msg)
rcv
temp
port
rcv
local rcv (port, msg)
send (temp, msg)
proxy
block
netmsg server will get
- blocked


netmsg server
netrcv (msg) → rcv
rcv
temp
port
Create a thread
rcv (msg.port, msg')
send (msg.temp, msg')
netsend
thread may block
sends to proxy

note

→ send blocking → can be fixed using threads


m_1
m_2
send
netmsg srv
rcv
proxy
thread
port
port
netmsg
send

TCP - IP

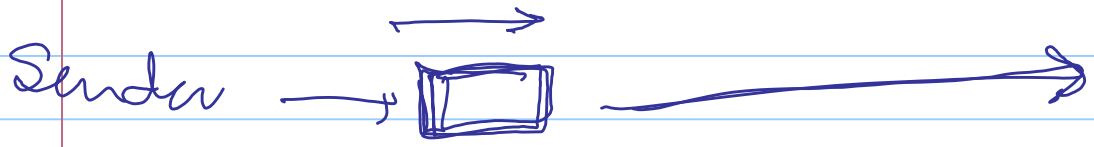
↳ network transport

↳ [packet loss
retransmit
timeout]

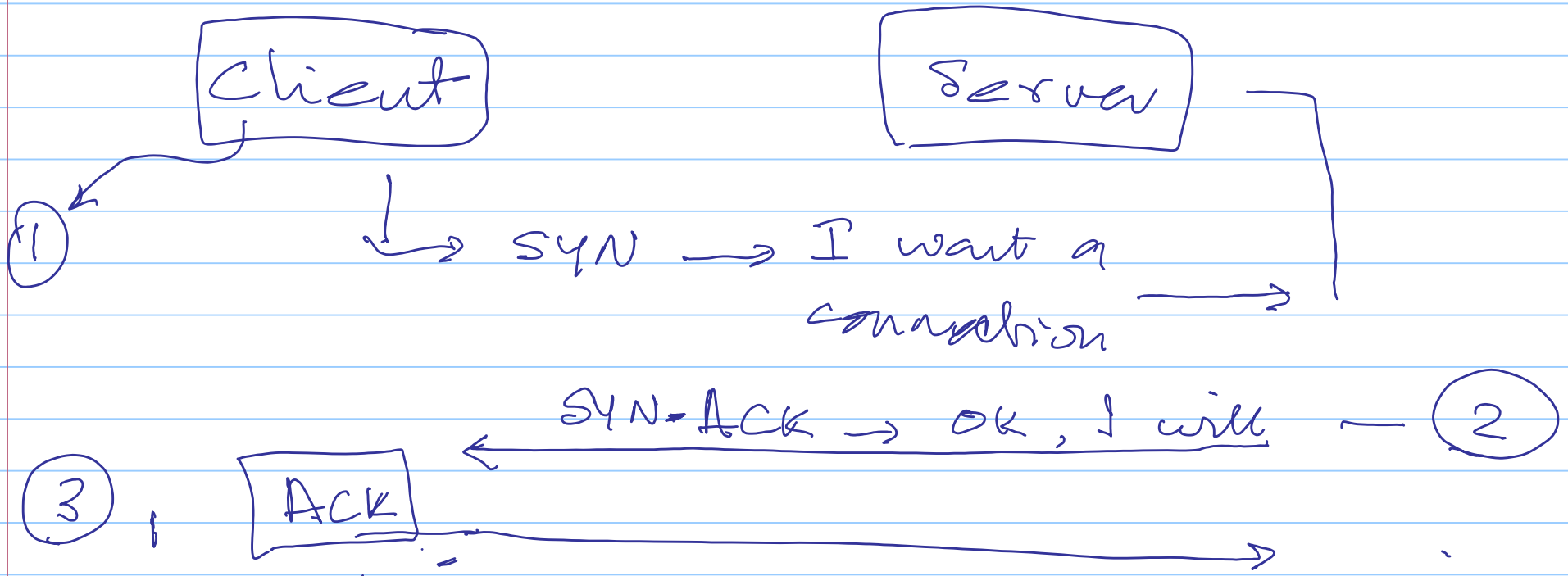
→ routing →

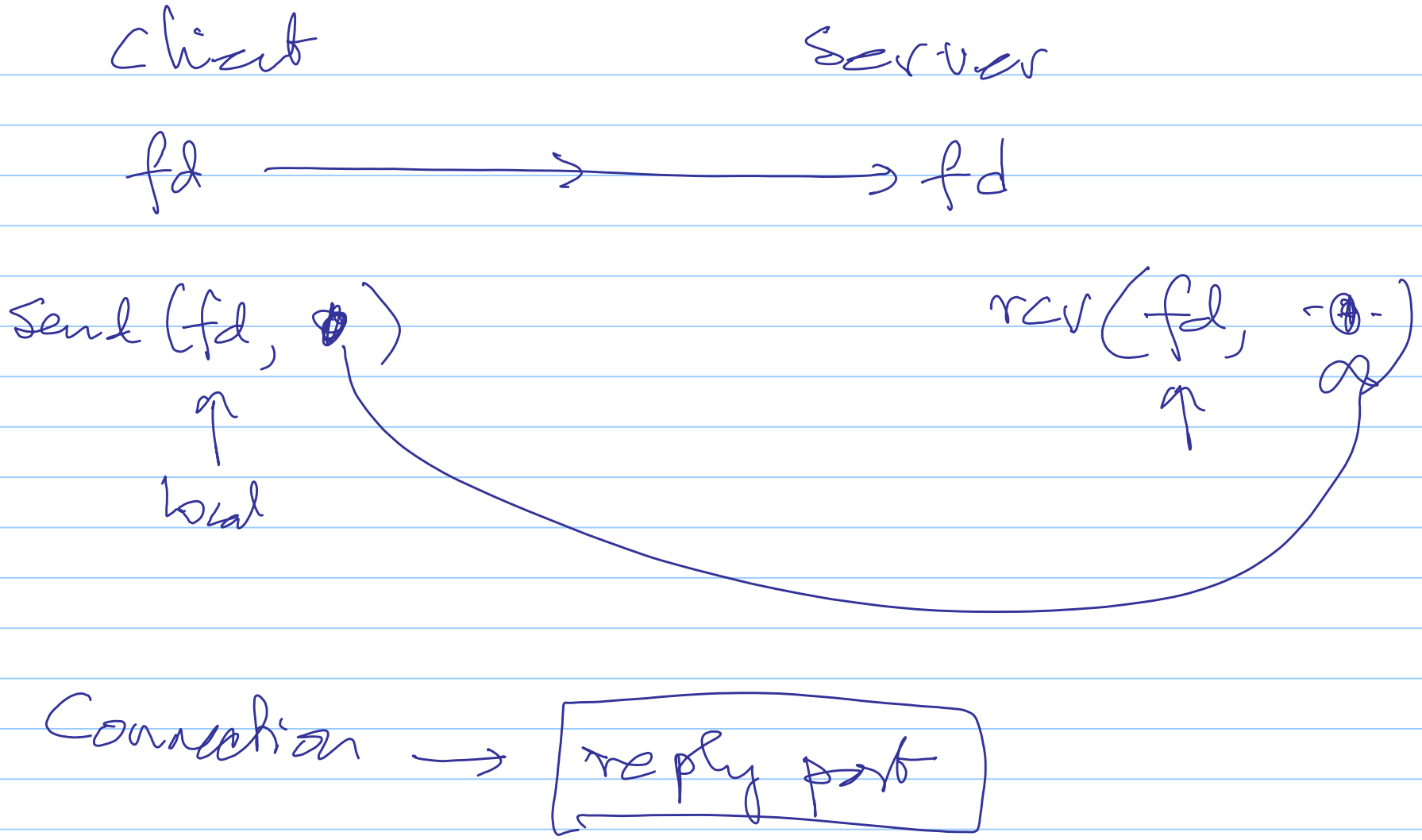
→ flow control → connection

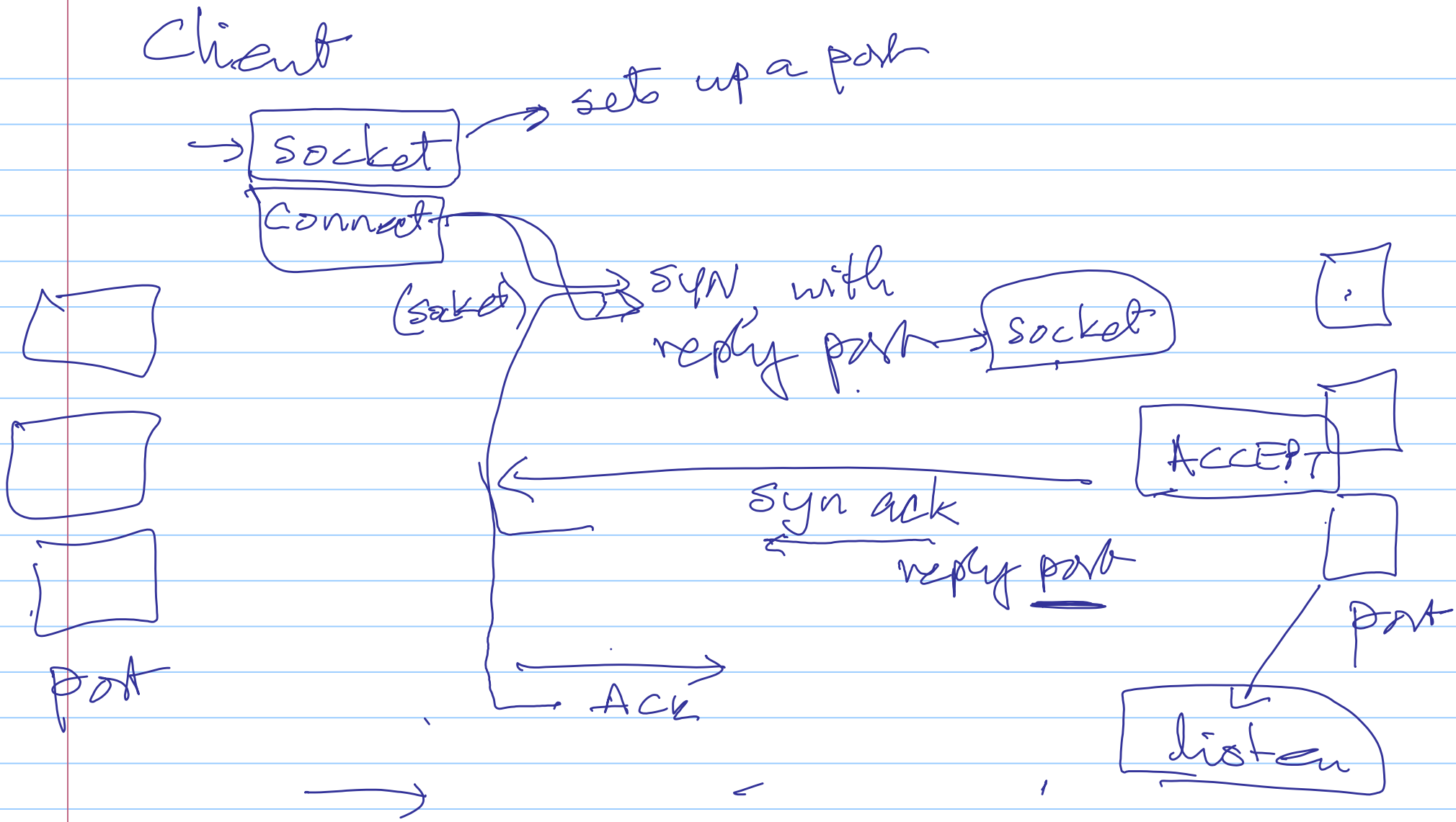
TCP IP

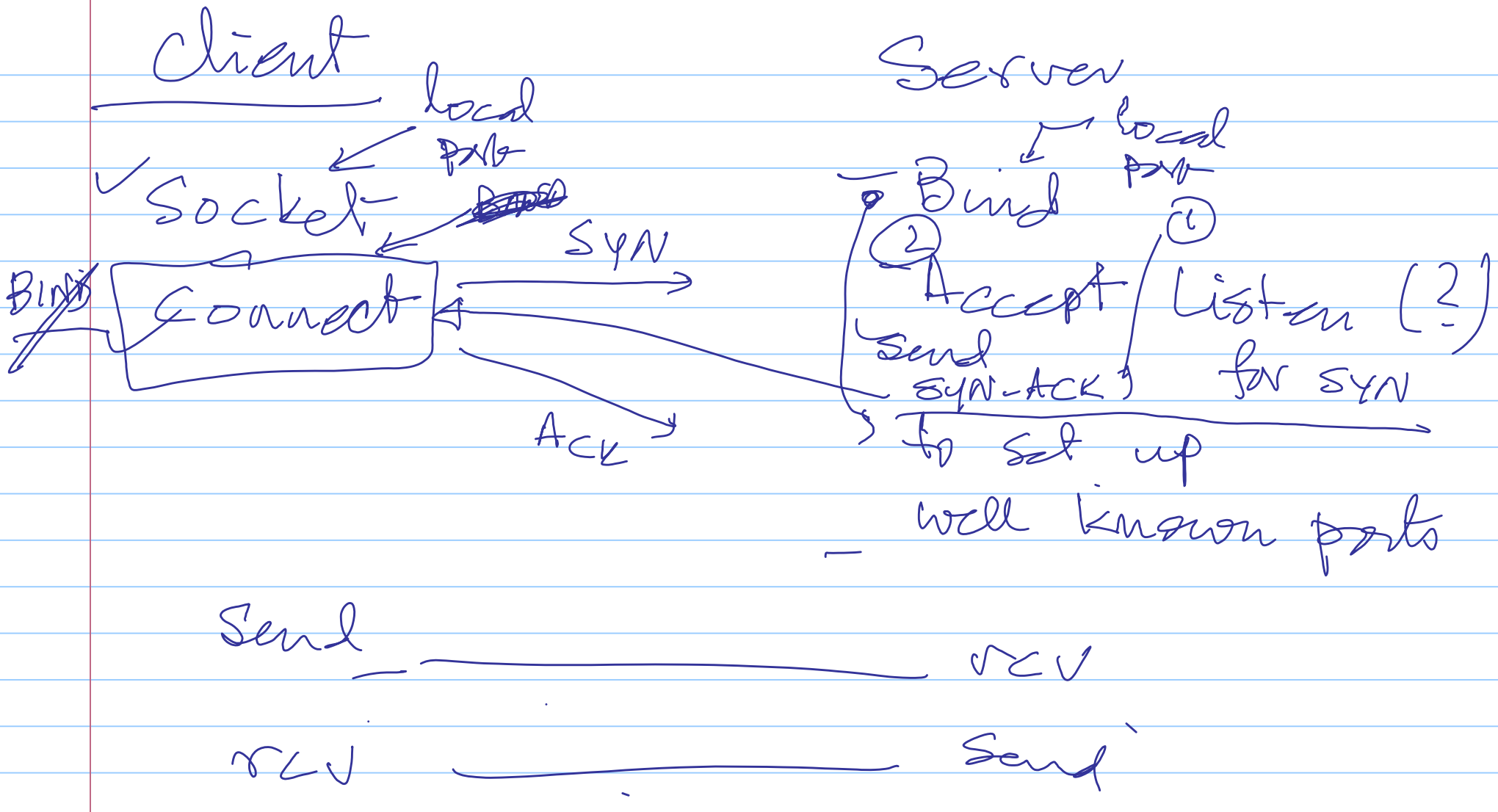


Connection




Client
Server
fd
fd
send(fd, @)
rcv(fd, -@-)
local
Connection
reply port


Client
socket
sets up a port
Connect
(socket)
SYN with
reply port
socket
ACCEPT
syn ack
reply port
port
port
ACK
listen



Send ——— rcv

rcv ——— Send