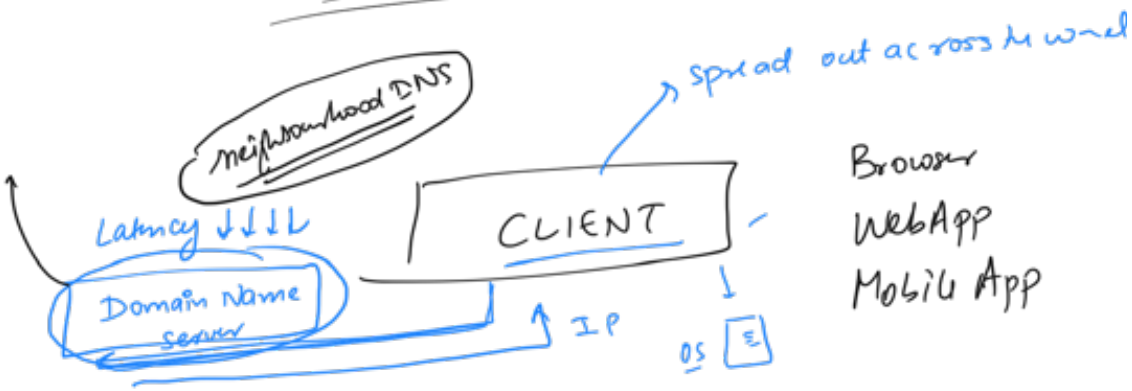
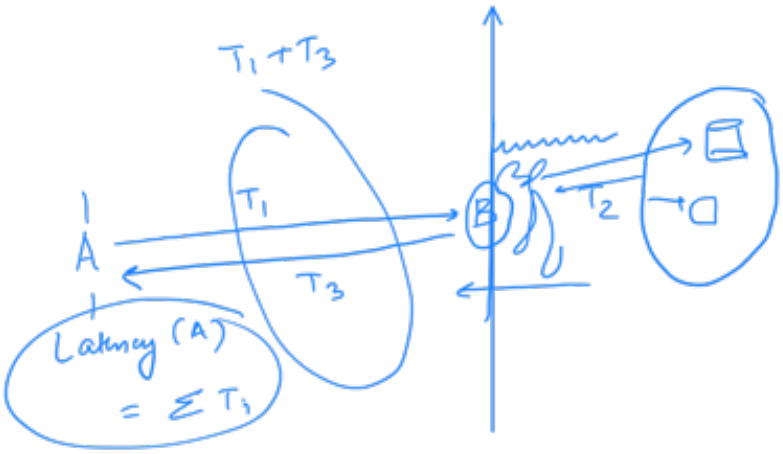
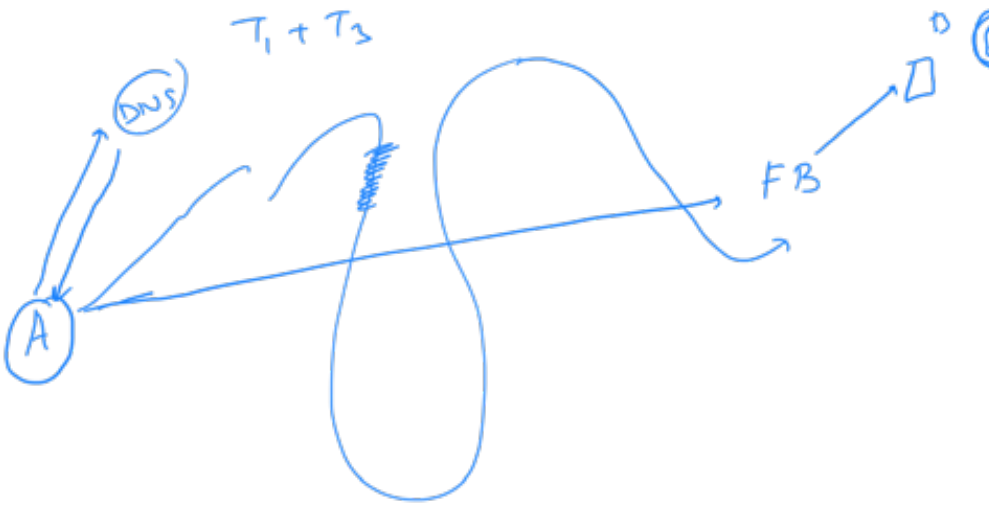
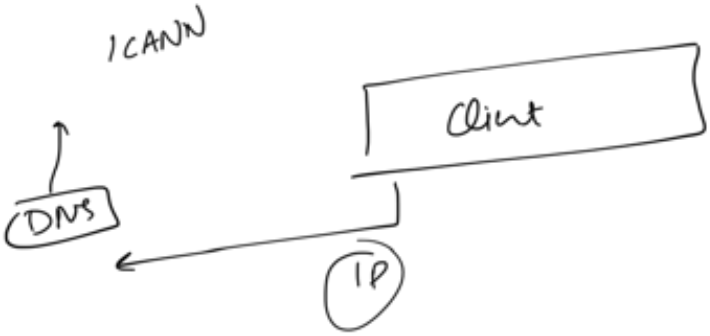
High Level Design

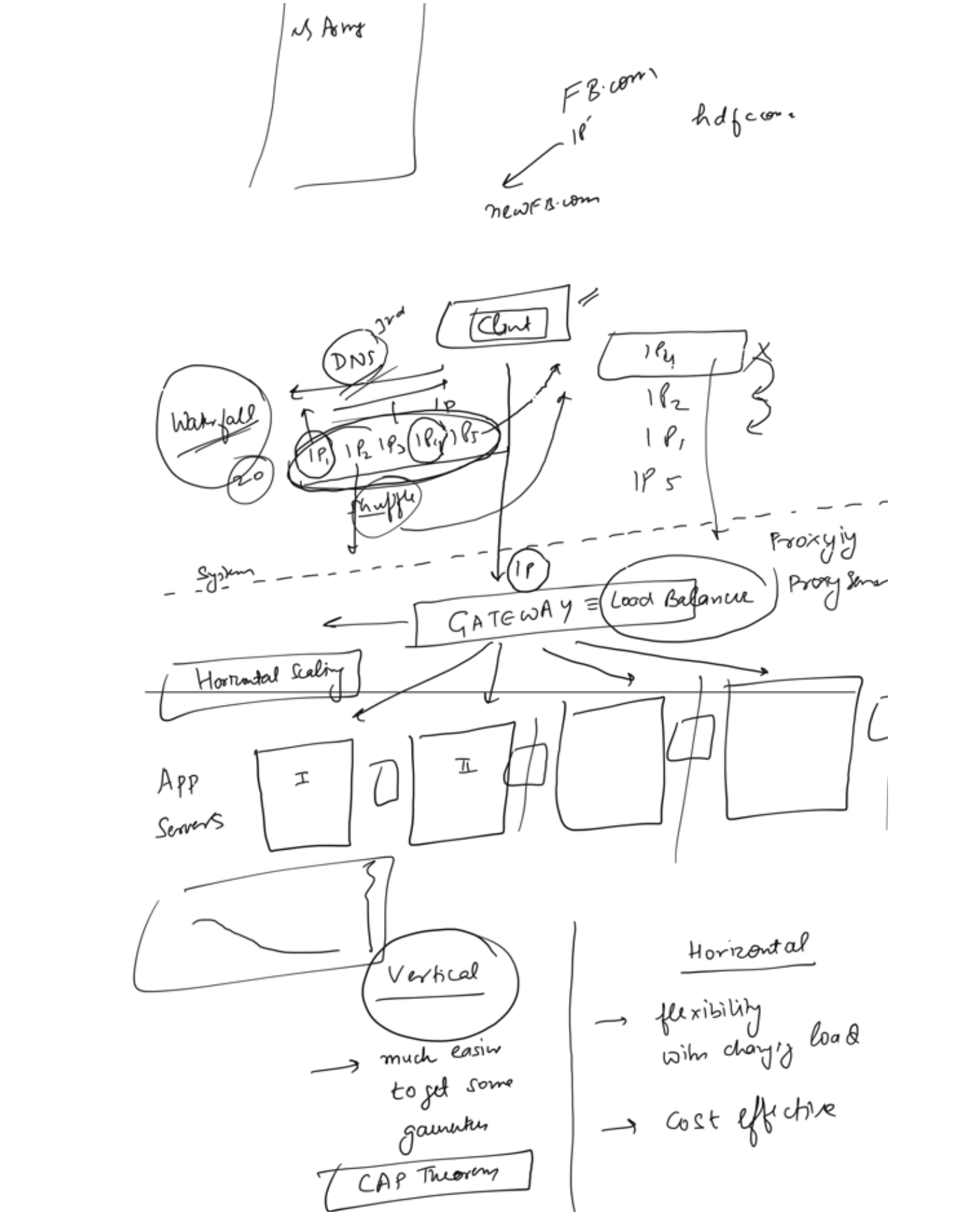


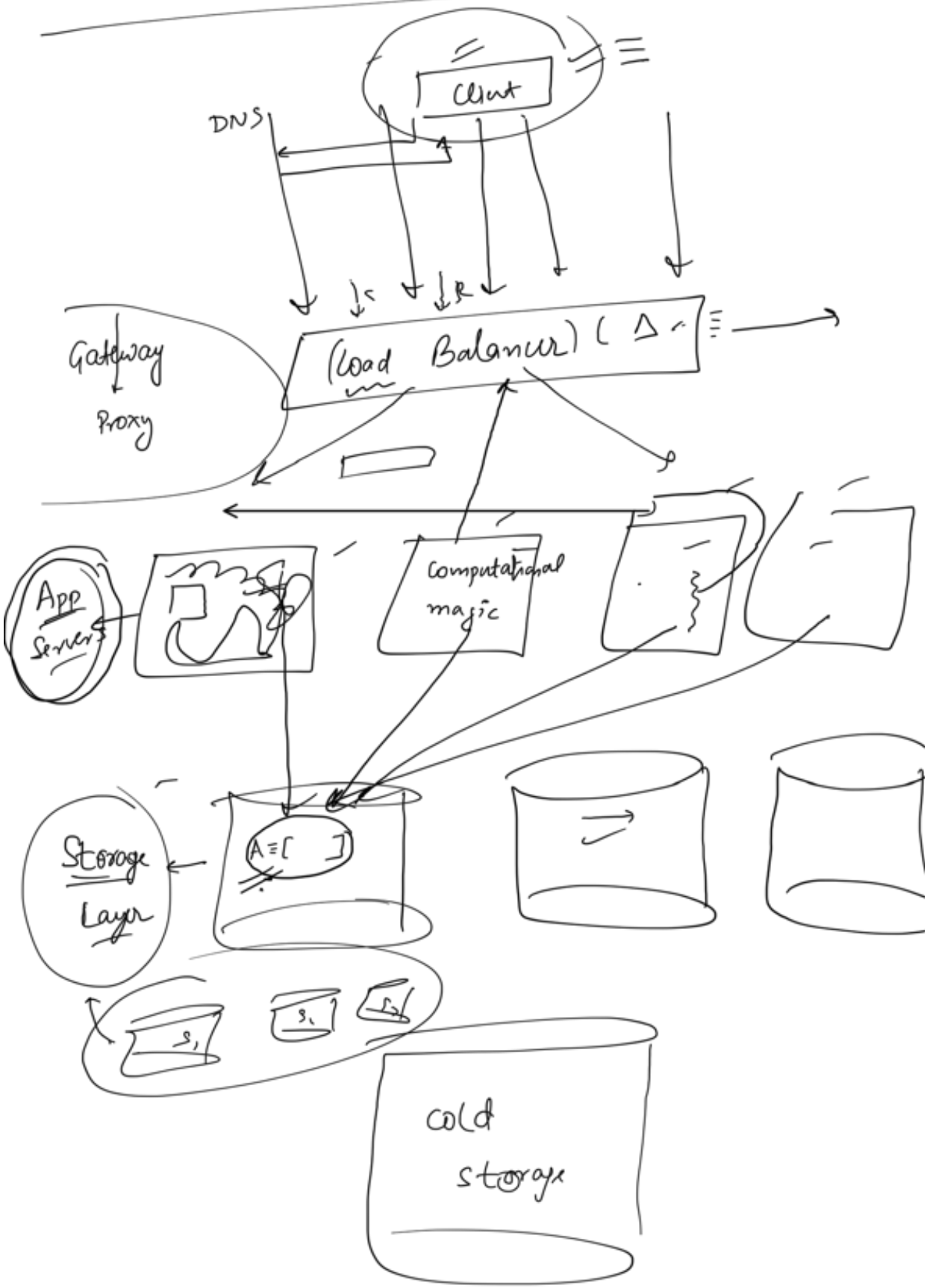


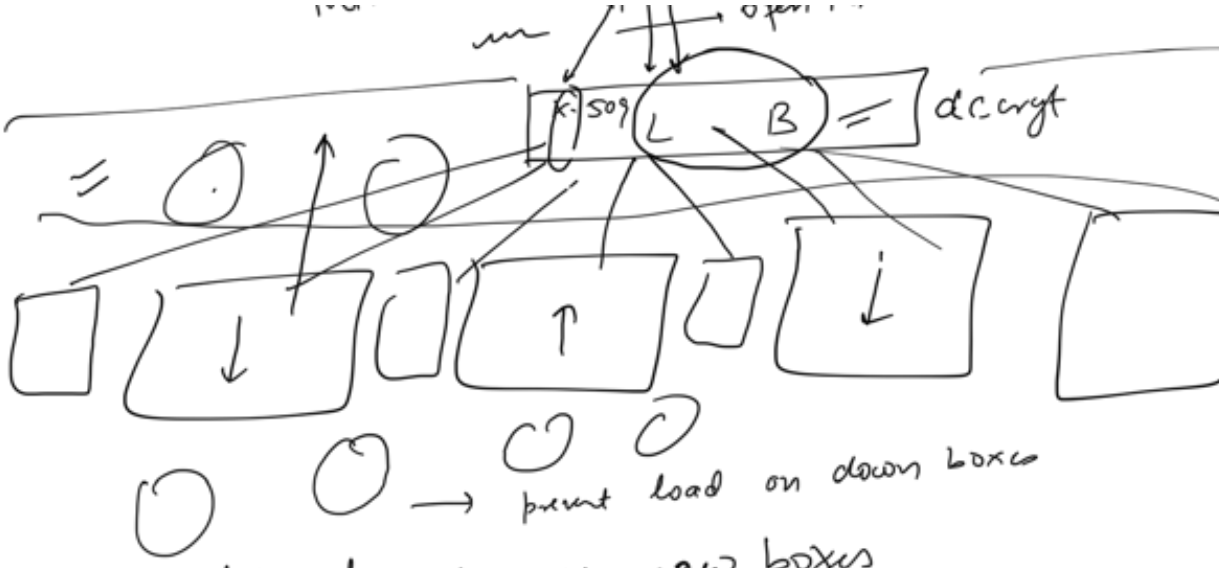




ISPs
Google
FB


Army
FB.com
IP
newFB.com
hdfc.com
Client
DNS
3rd
Waterfall
20
IP1 IP2 IP3 IP4 IP5
Shuffle
IP4
IP2
IP1
IP5
System
IP
GATEWAY = Load Balancer
Proxying
Proxy Server
Horizontal Scaling
App Servers
I
II
Vertical
→ much easier to get some guarantee
CAP Theorem
Horizontal
→ flexibility with changing load
→ cost effective


Client
DNS
Gateway
Proxy
Load Balancer
App
Server
Computational
magic
Storage
Layer
cold
storage



→ prevent load on down boxes

intranet → rgistr new boxes

→ balance out

— X.509 —

Layer 4 ≡ Transpat Layr

addition security

~~security~~ Source IP
Source Port
Dest IP
Dest Port

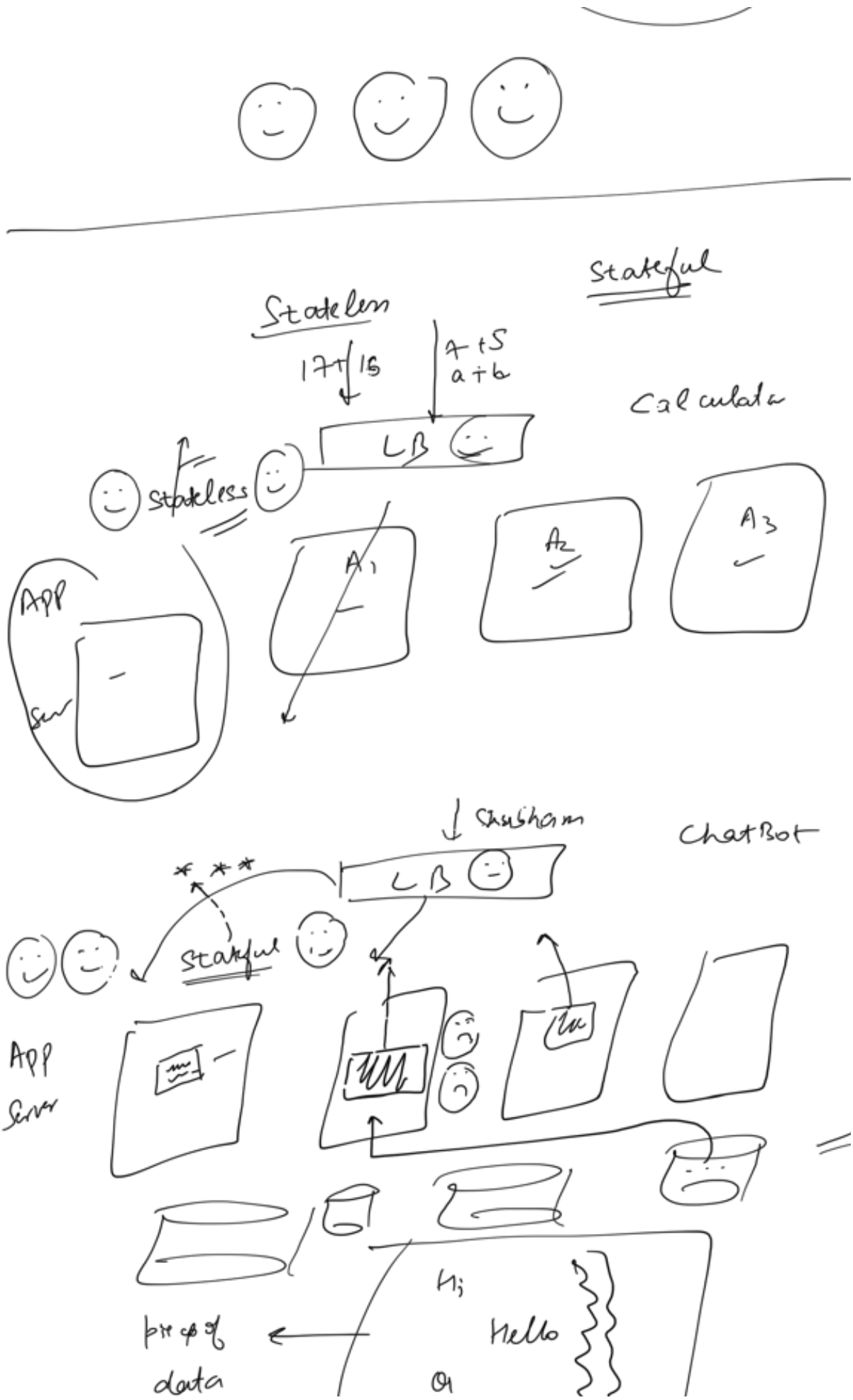
fast
Simple

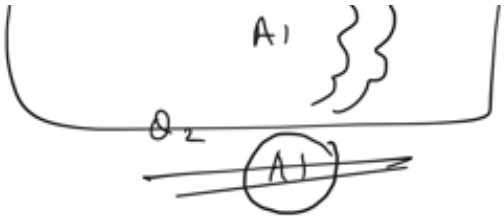
Layr 7 LB ≡ Aplicatn

user id ≡ 707
uid = 107

much bttr flexibilty

Slowerd

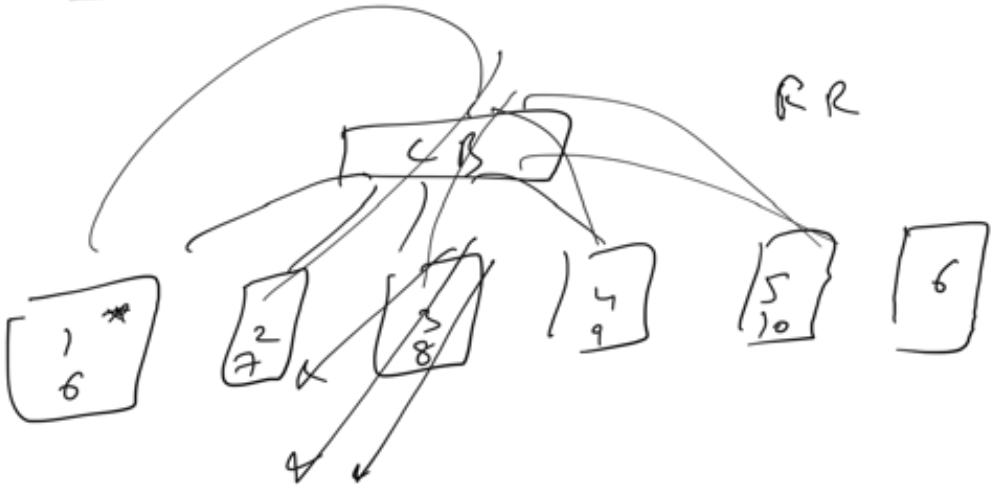

Stateless
Stateful
17+15
4+5
a+b
Calculator
LB
stateless
App
Server
A1
A2
A3
ChatBot
LB
stateful
App
Server
Hi
Hello
Bye
data

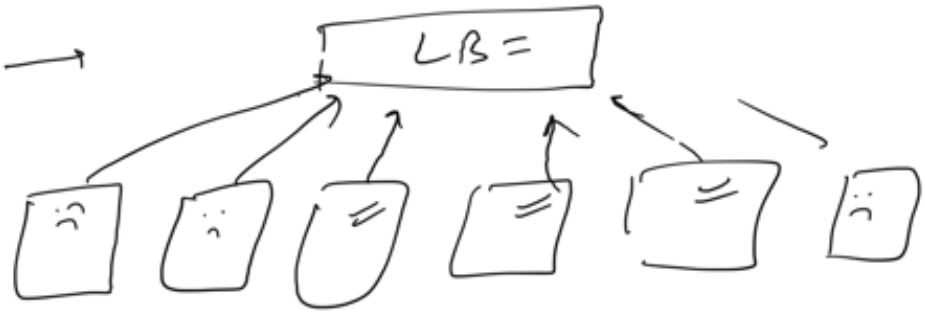


Load Balancing

Stateless

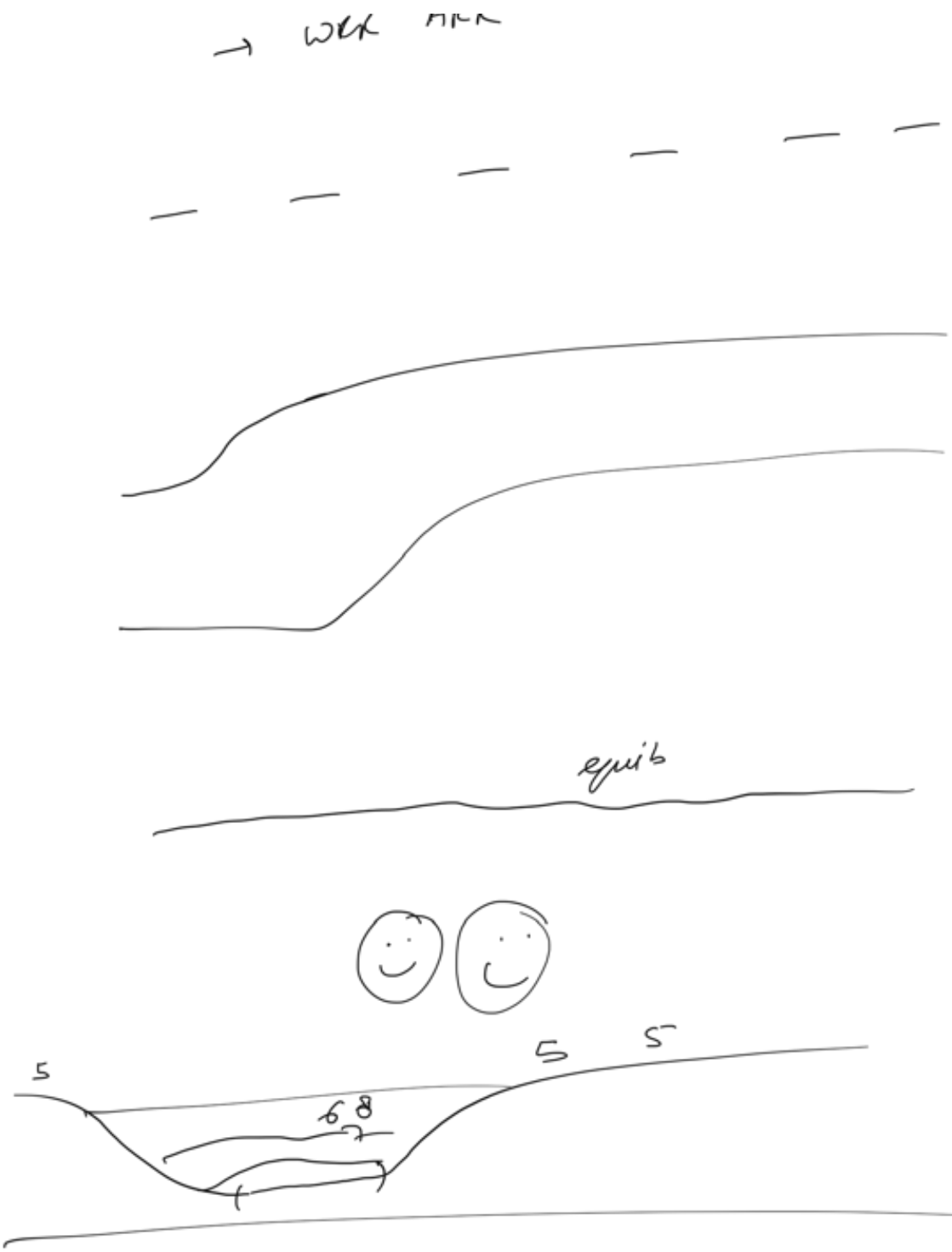
→ Round Robin

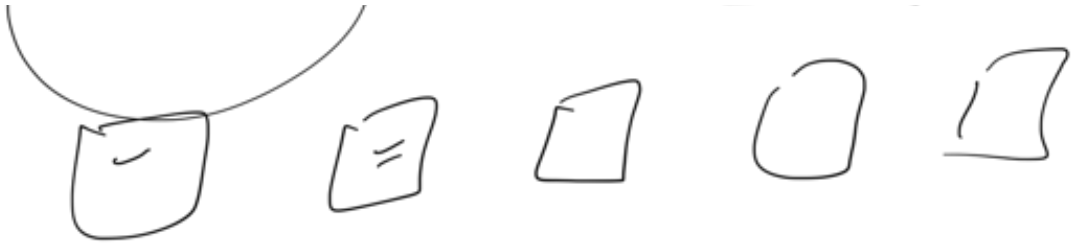




WRR

→ (Least Connection first)


equib
5
5
5
6 8
Load Balancing in Statfull
2 B
u_id b_id
LB
Mapping



Sol (1) Maintain Map BAD

map-size = order of key-size

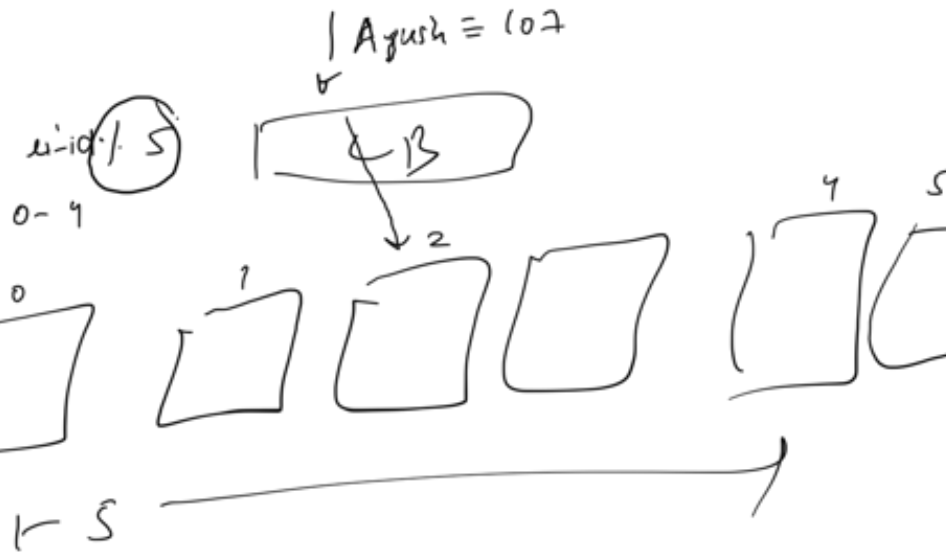
Load Balancing func will slow down

Sol (2) Hash

BAD!!!

10 boxes ≡ 10 App Server

uid %. 10 = 0-9

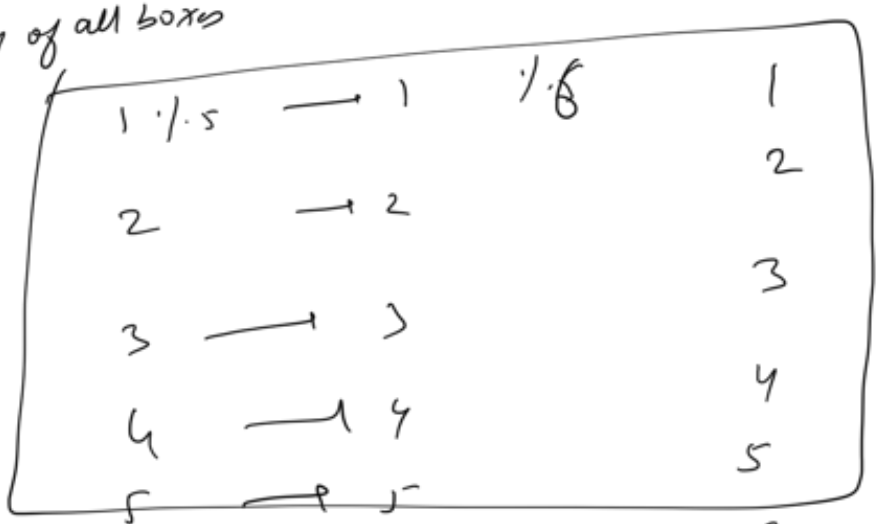


→ gaurav of stickiness using hashing

→ LB :)

Downside

On server inc/dec you will have %6 to move state of all boxes internally



in LB

③ Ranges

1 - 1000

1001 - 5000
5001 - 10000

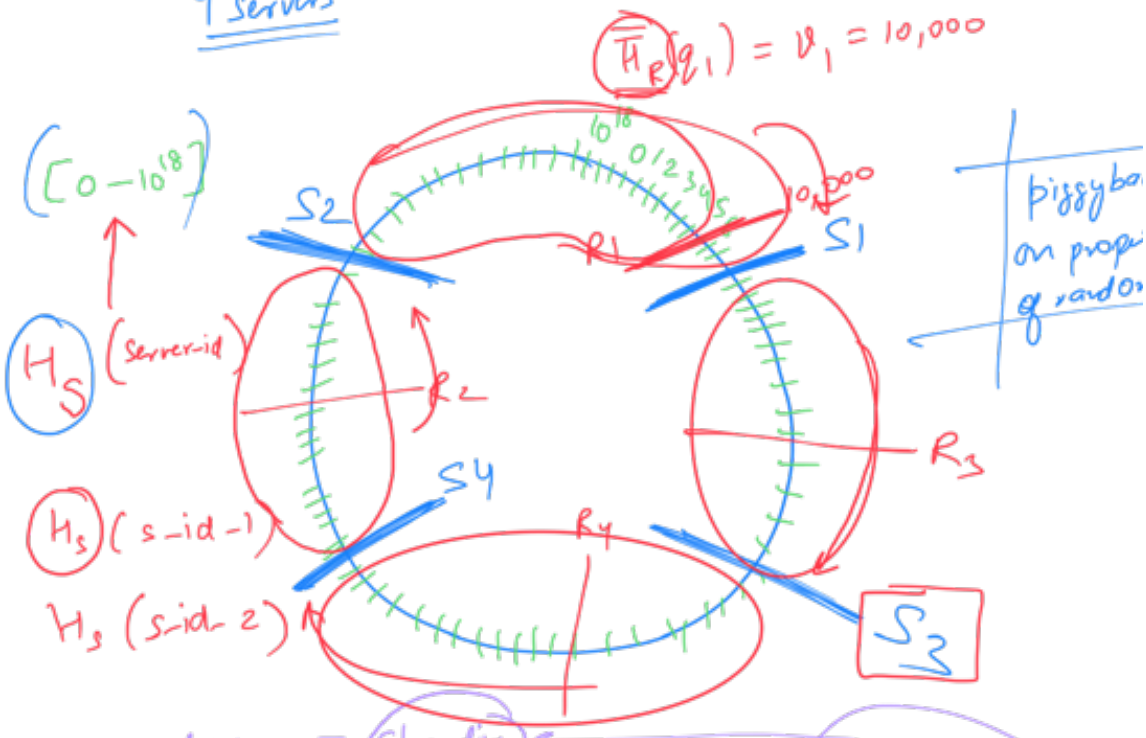
Bottleneck: Add / Removal of servers is inefficient

4 Consistent Hashing

LB for Stateful Applications

const addition/removal of servers

4 servers

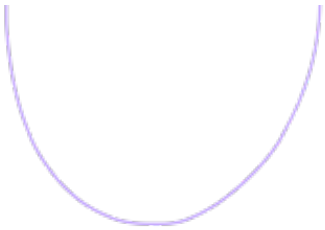


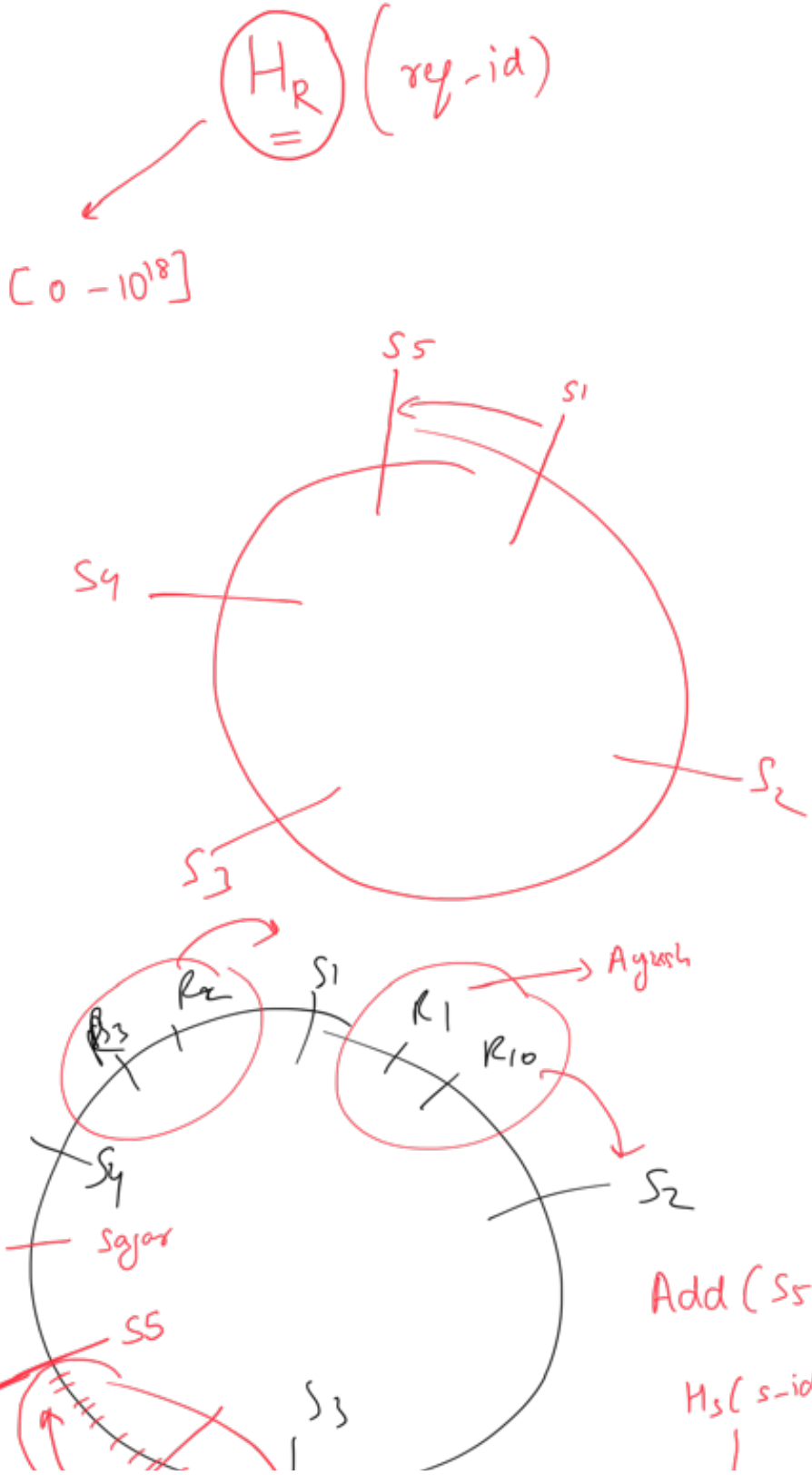
distribute ≡ Sharding key

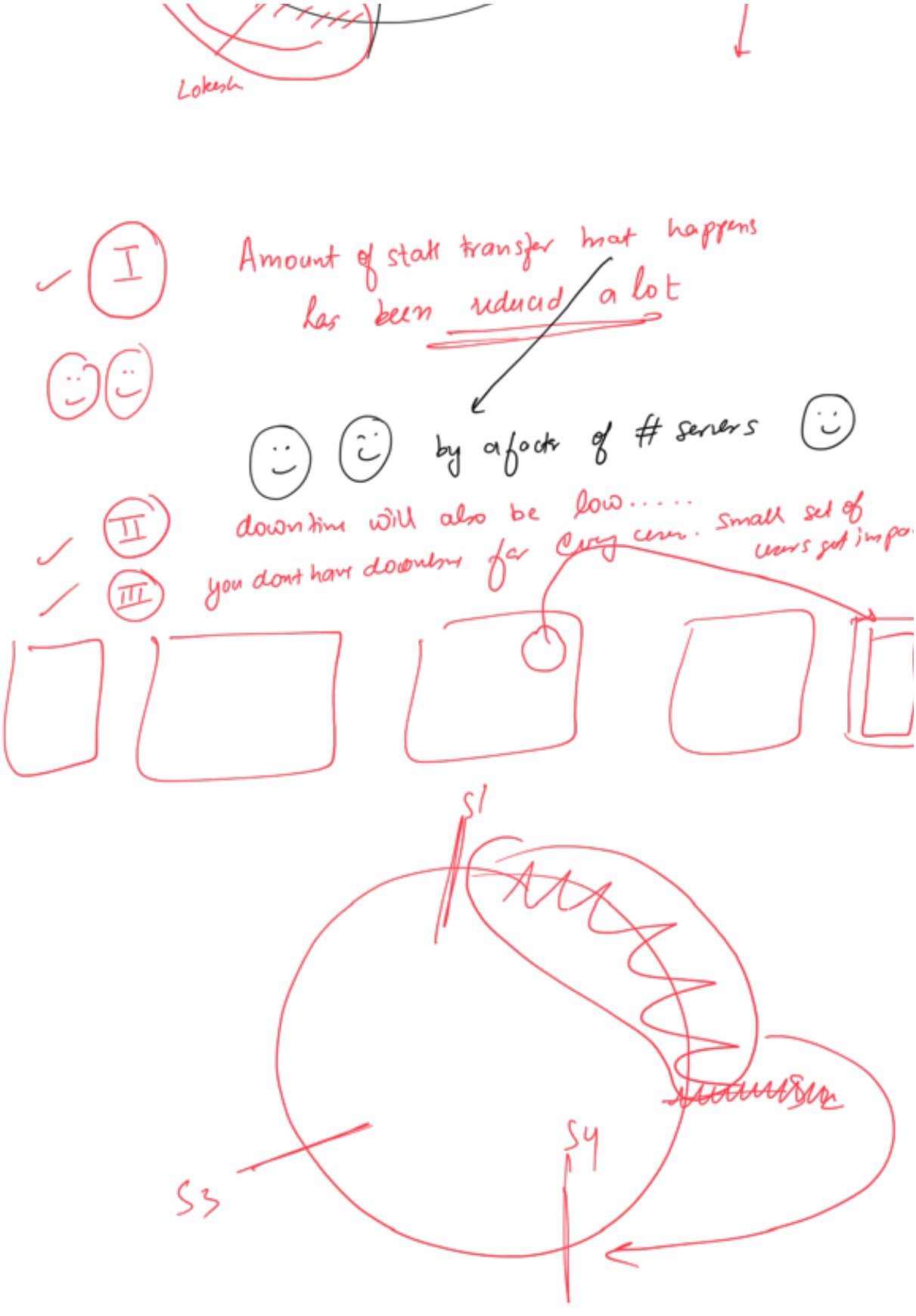
req_id ≡ user id

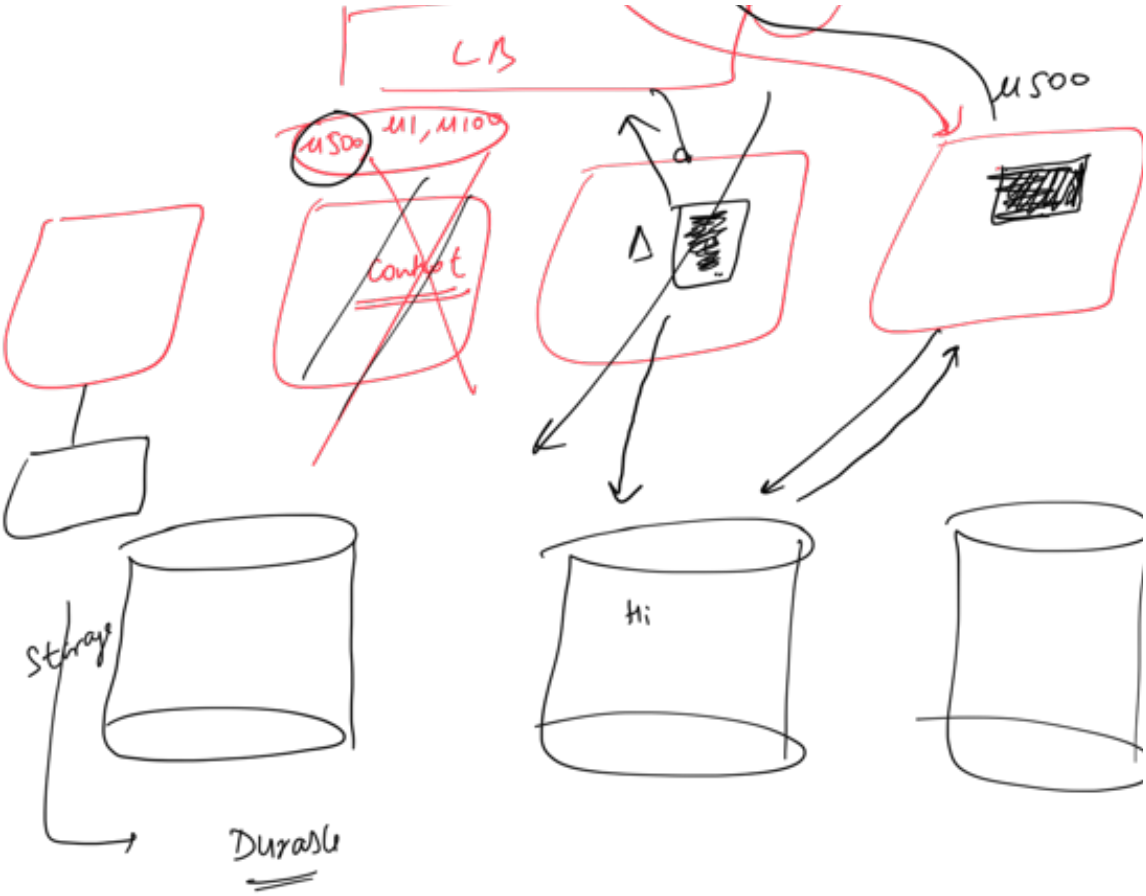
req_id ≡ location id

* $0 - 10^{18}$ almost




H_R (req-id)
[0 - 10^{18}]
S5
S1
S4
S2
S3
R2
S1
R3
R1
Ayush
R10
S4
S2
Sgar
S5
S3
Add (S5)
H_S (s-id, S)


Lokesh
I
Amount of state transfer that happens has been reduced a lot
by a factor of # servers
II
III
downtime will also be low.....
you dont have downtime for every user. small set of users get impa
S1
S3
S4
HE(4500)



→ we are not losing the ultimate source of truth

(A) cold start ≡ request's latency inc for some time

Hi

Hello

θ1

A1

θ2

A2

Online Learning Algo's model ≡

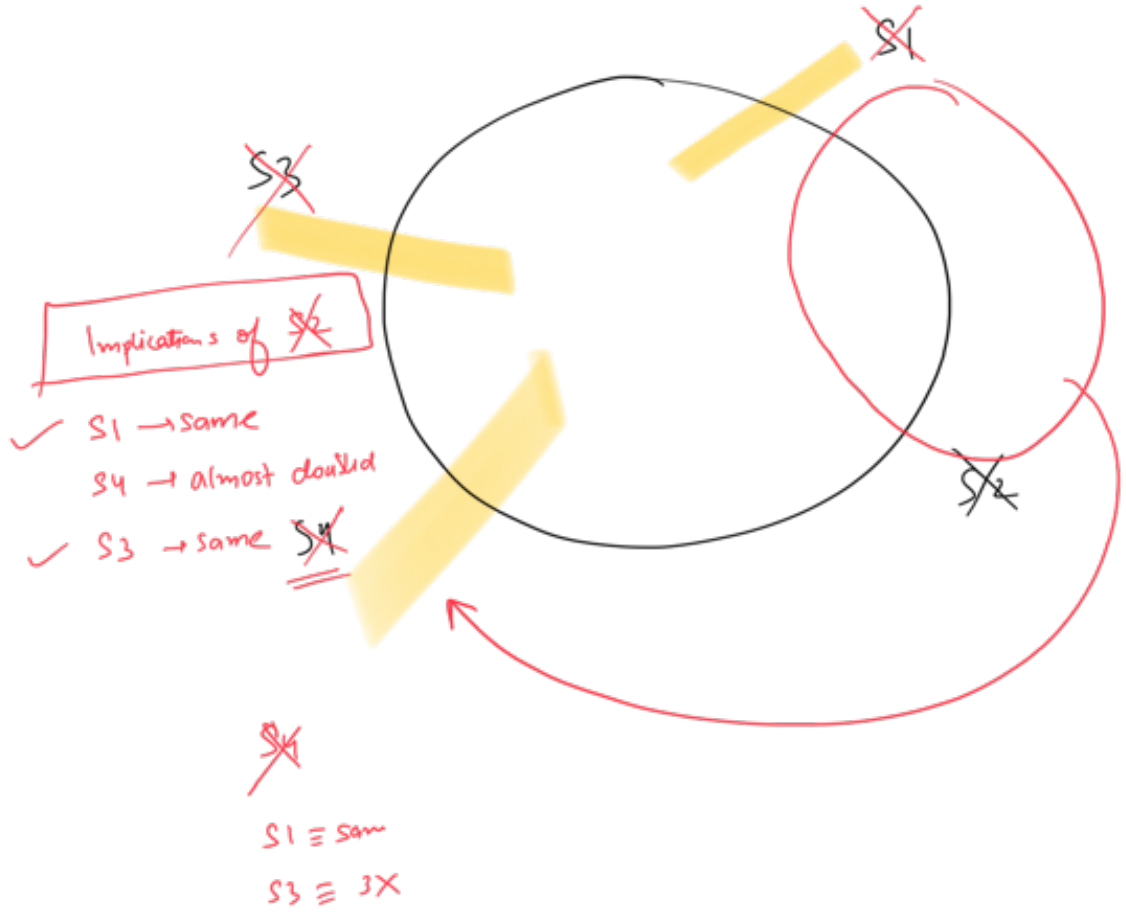
(B) pre-emptive copying

(I) ← storage layer → new App Server

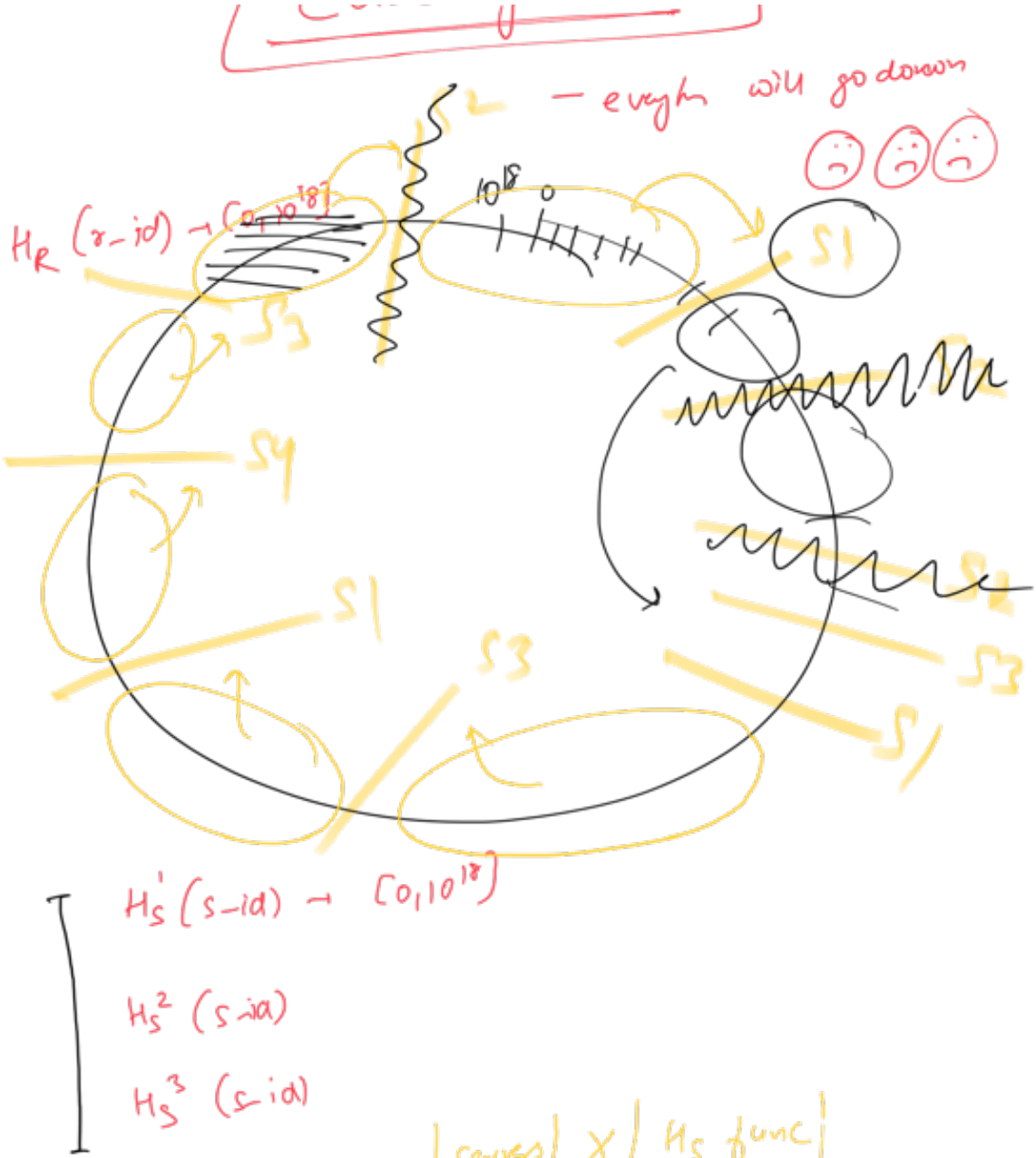
(II) ← Mslave

Solved problem

'impact of server addition or removal is minimized'



" Cascadiy Failures "

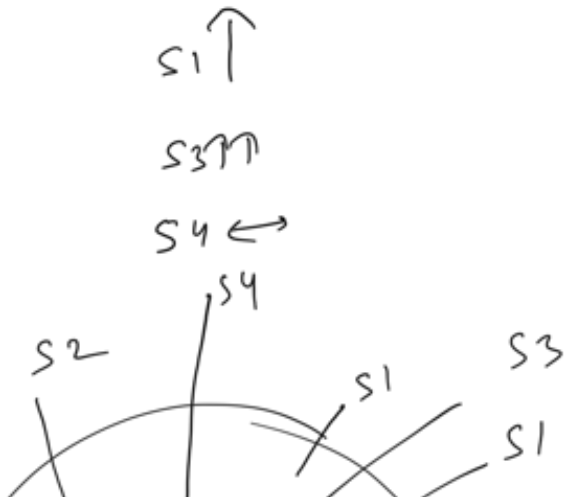


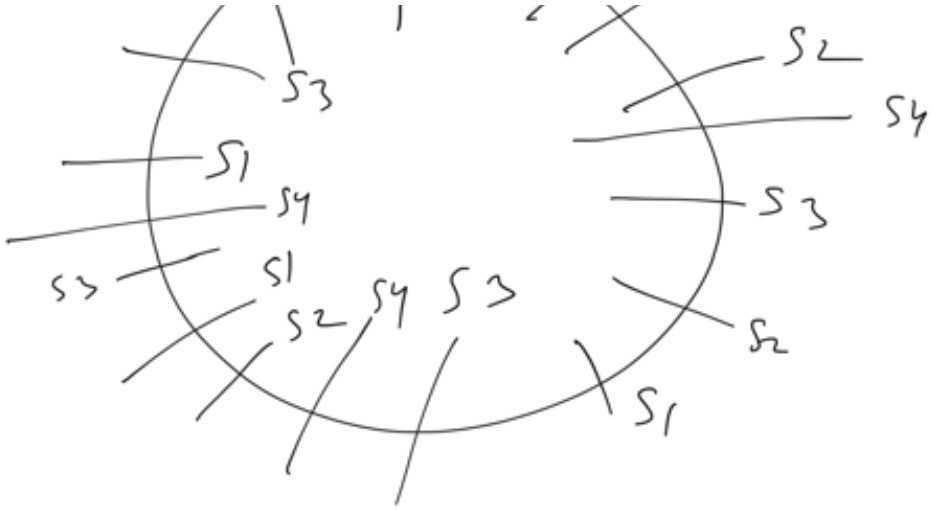
H_S^1 (s-id) → $[0, 10^{18})$

H_S^2 (s-id)

H_S^3 (s-id)

|servers| X |H_S func|

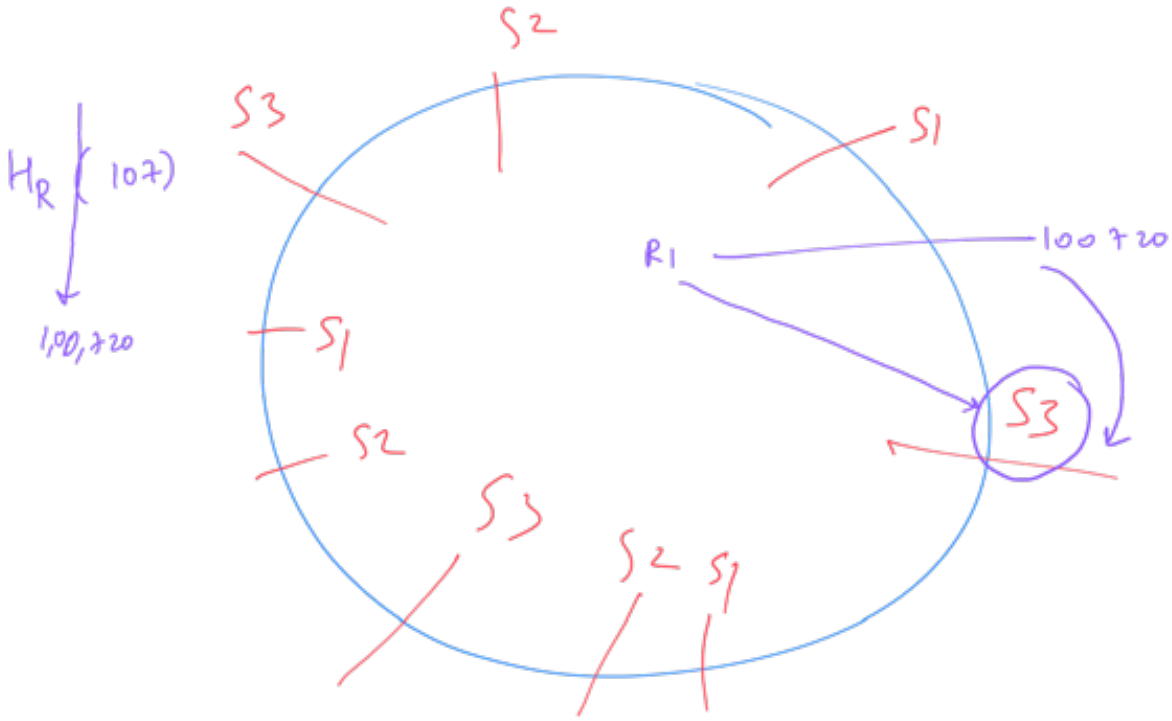




solved

Cascading Failures

(Naa, the load will get divided almost equally.

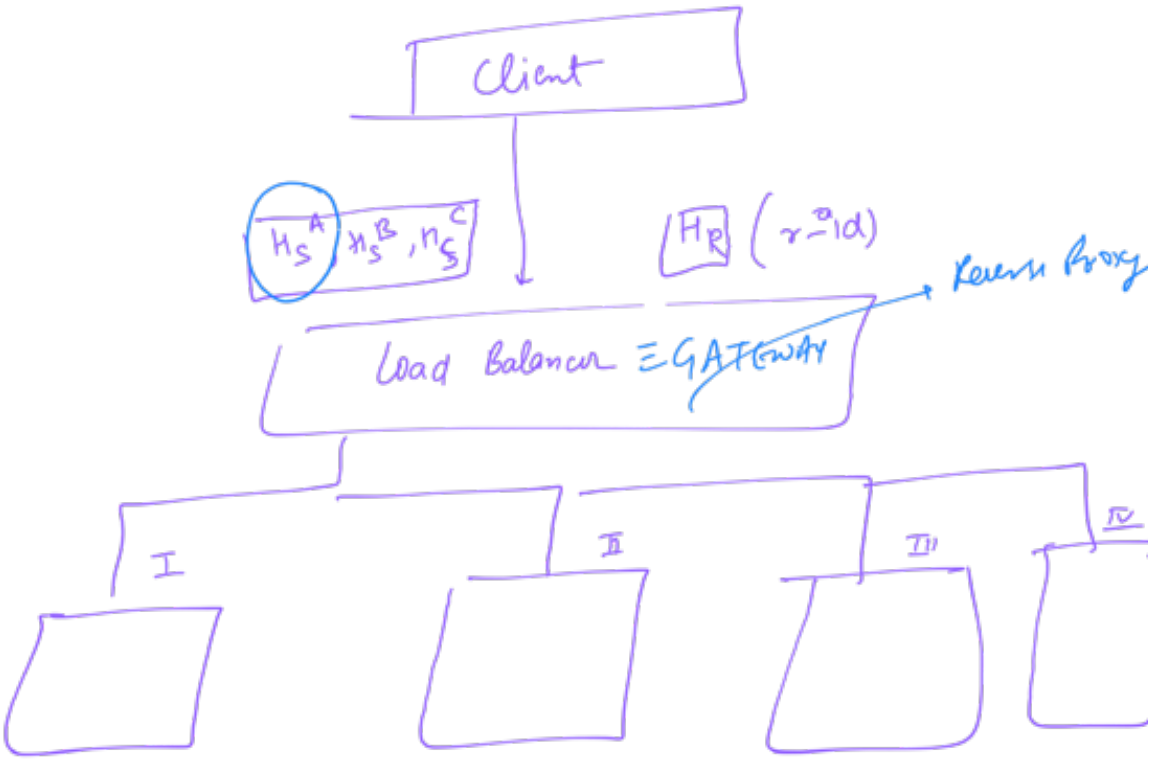


$\equiv \sqrt{|H_S| \uparrow\uparrow} \equiv$ Randomness $\uparrow\uparrow$

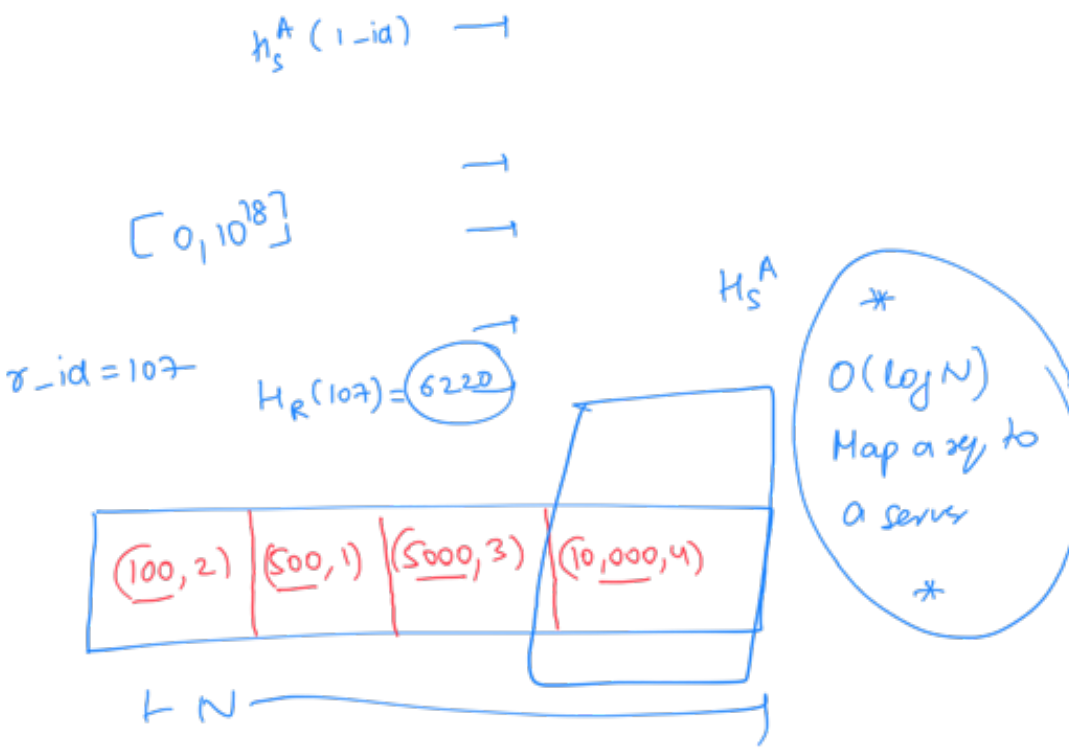
Equitable Distribution TT

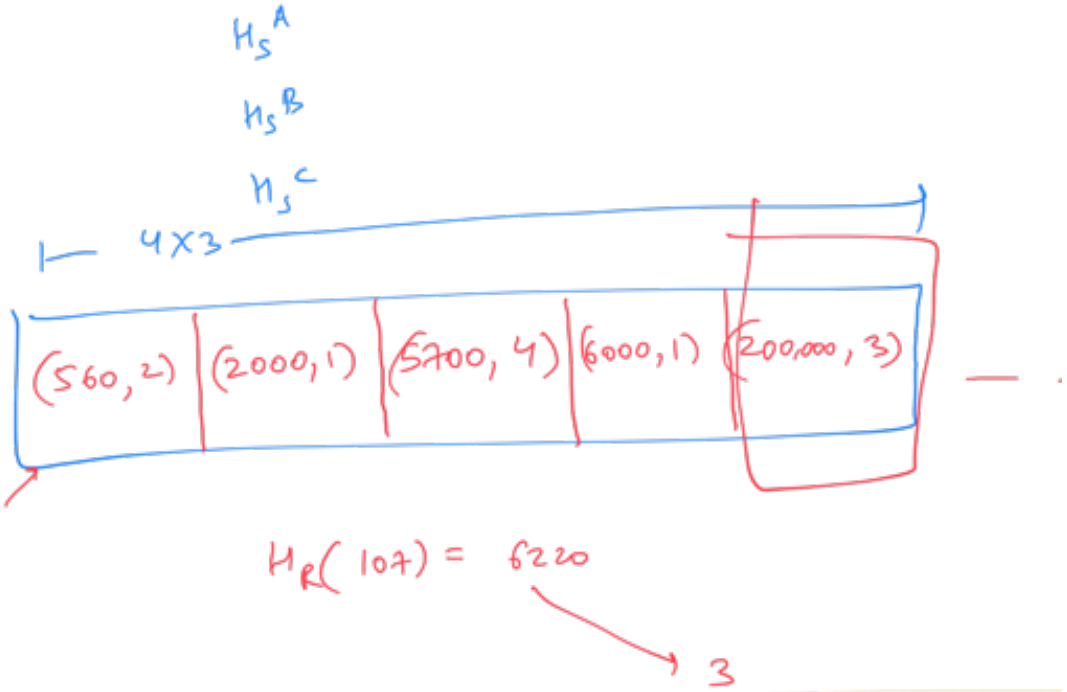
(Hs) TTT ≡ Time for Resolution / TTT

ASG



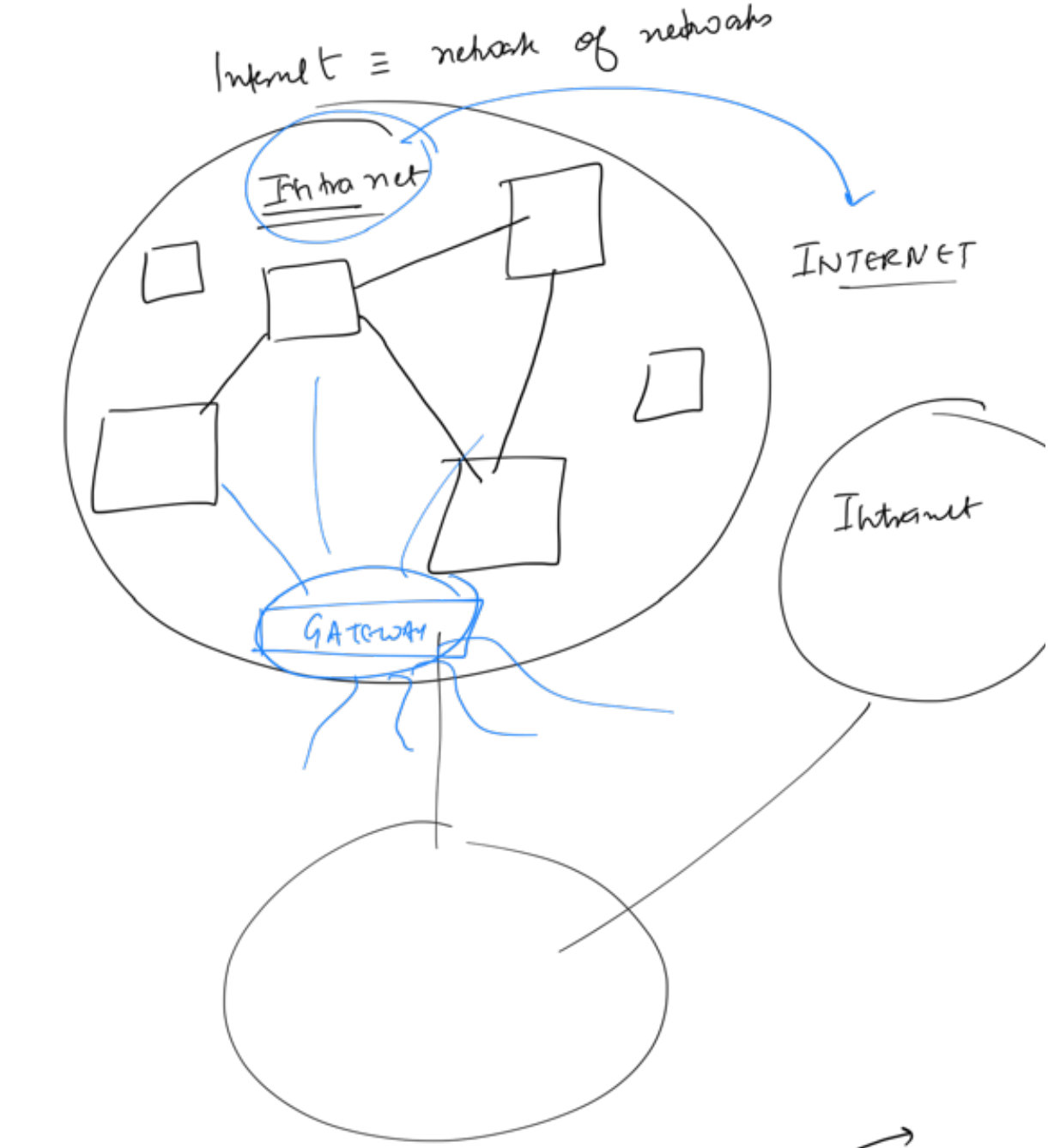
10 servers

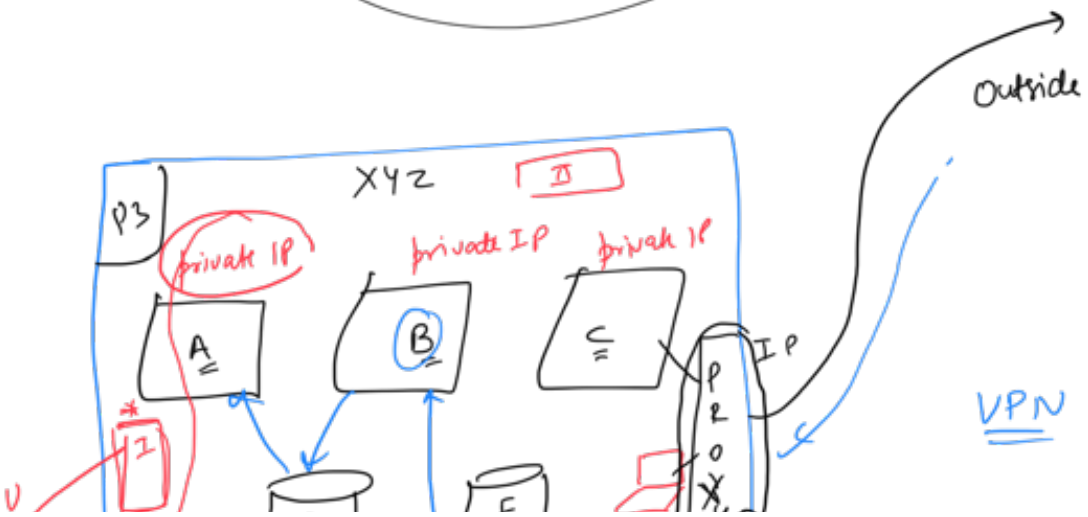


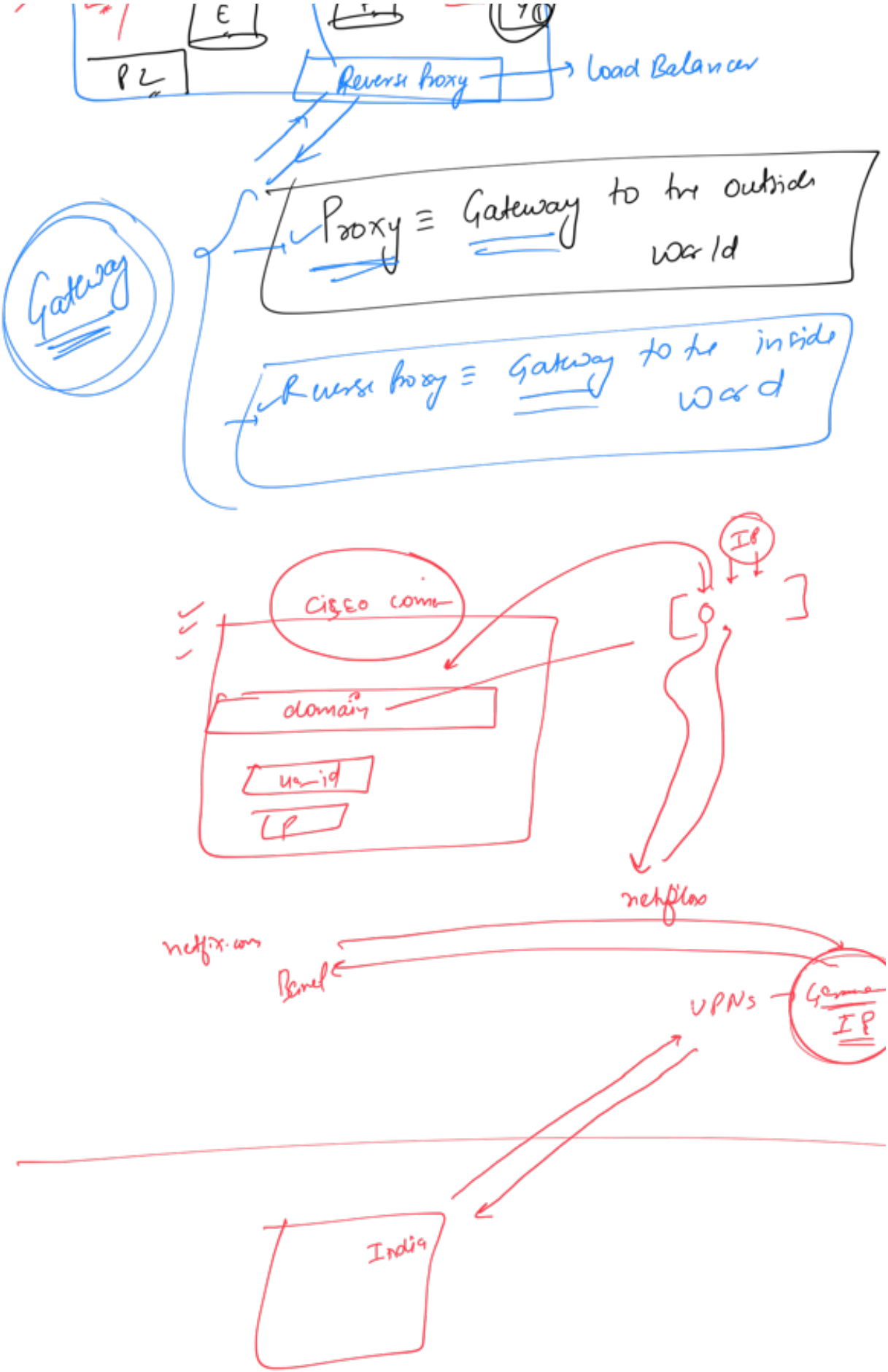


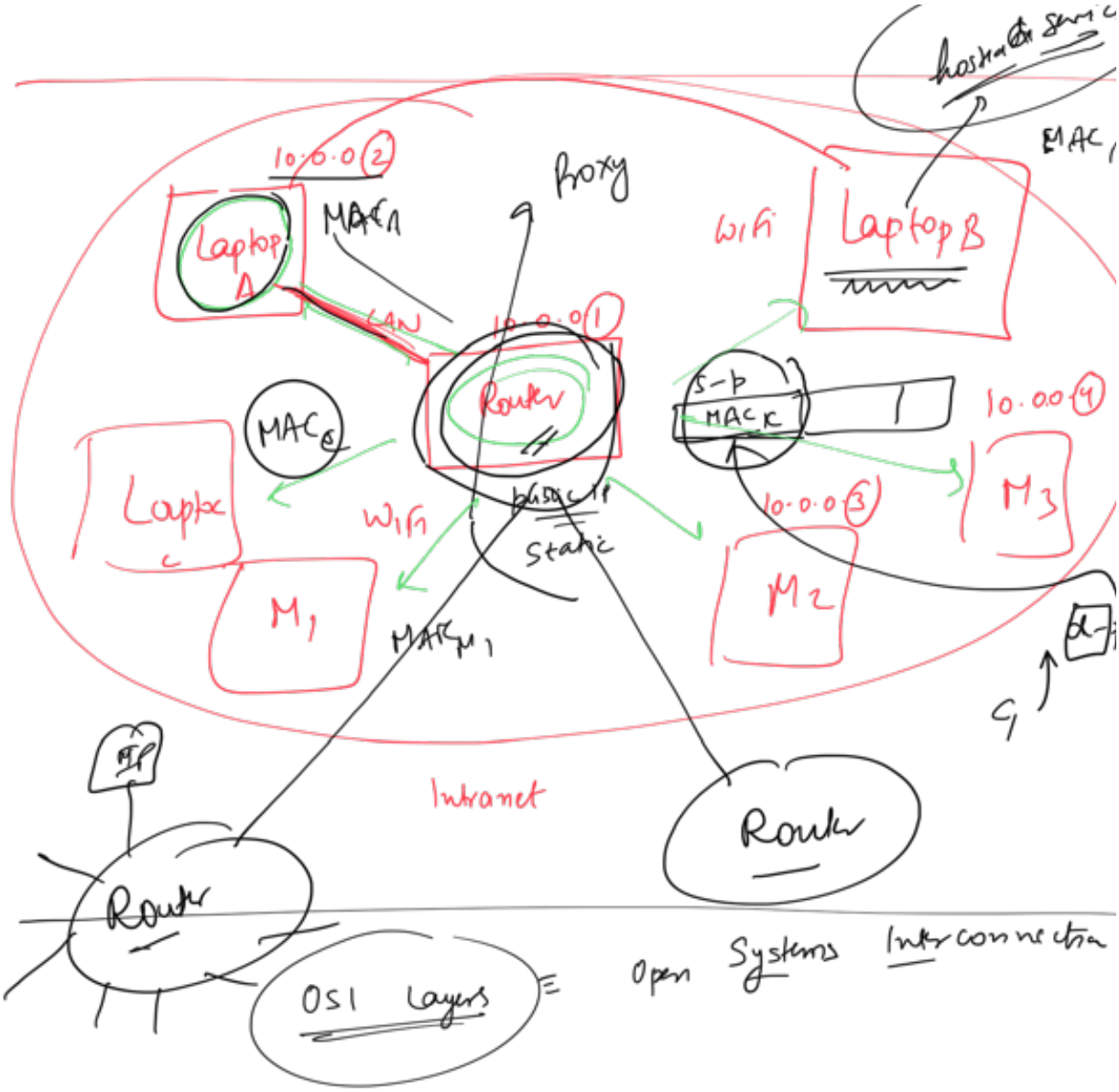
High Level Design

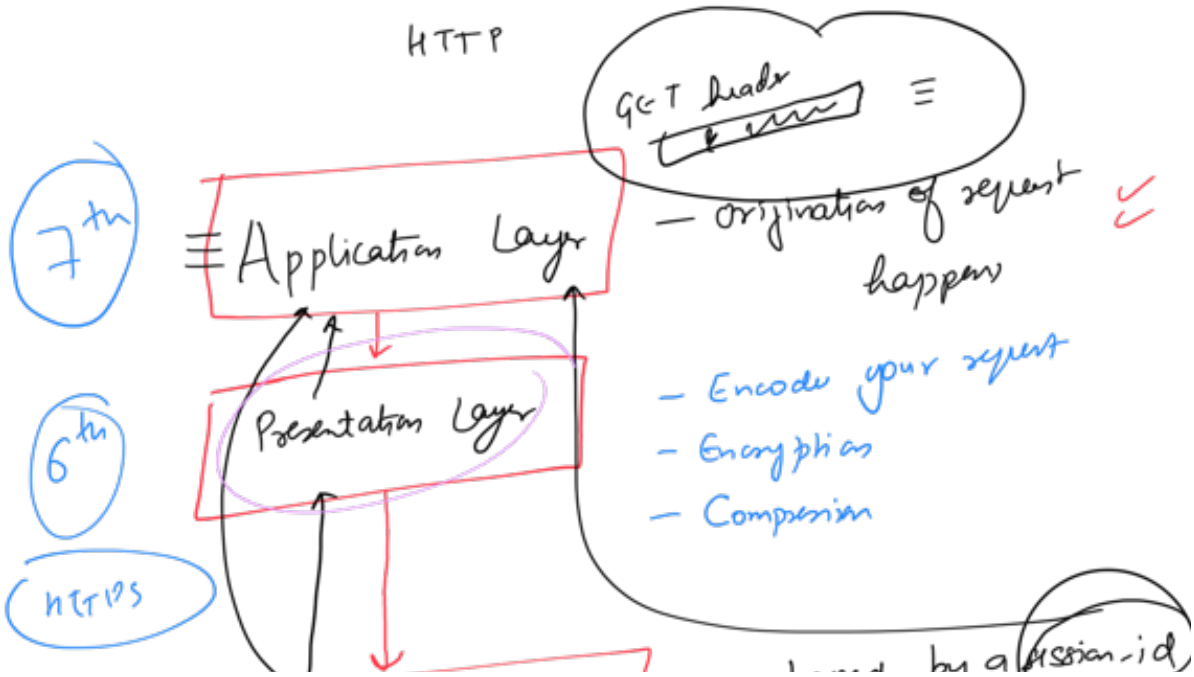
OSI Layers

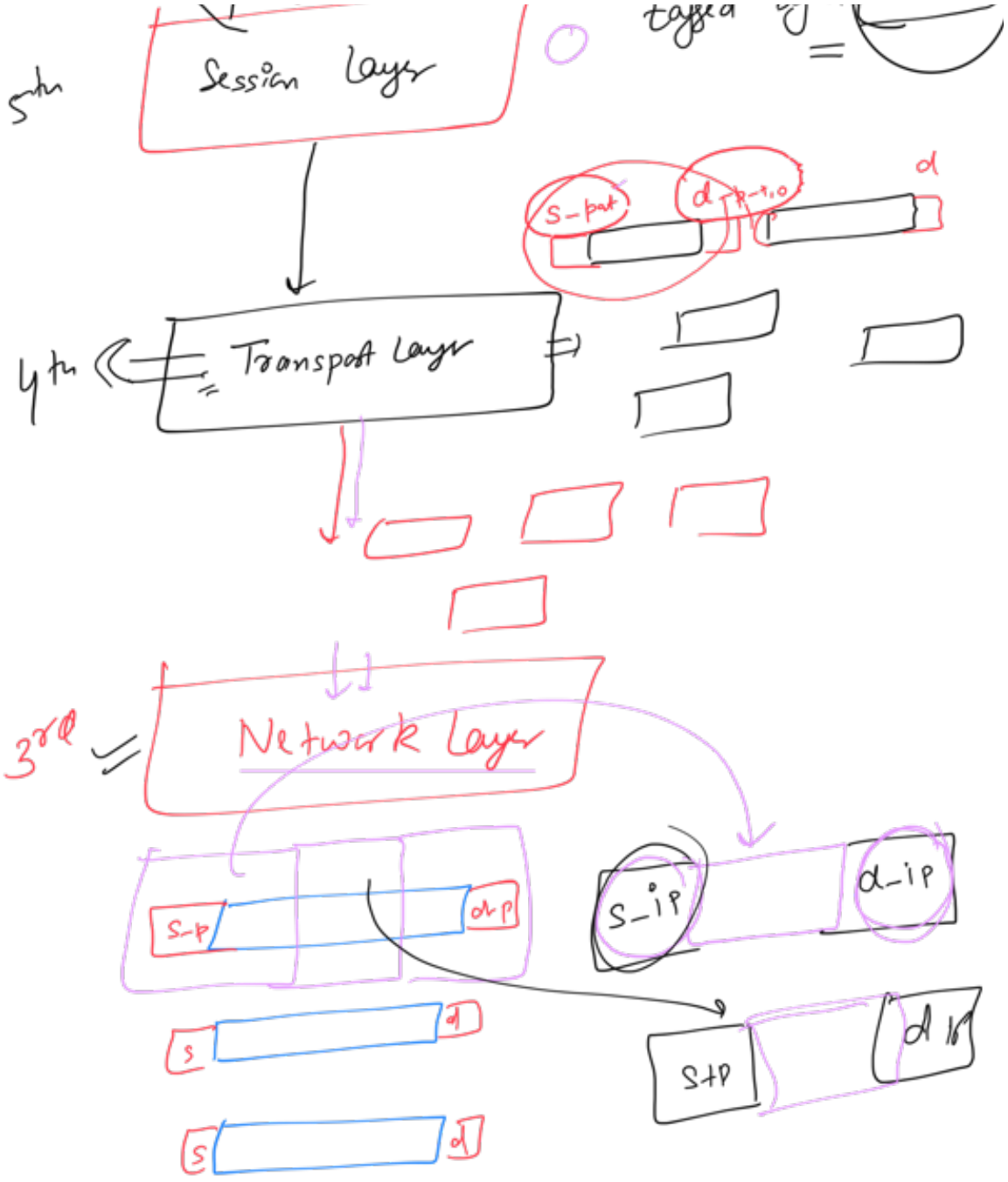

Internet ≡ network of networks
Intranet
INTERNET
Intranet
GATEWAY

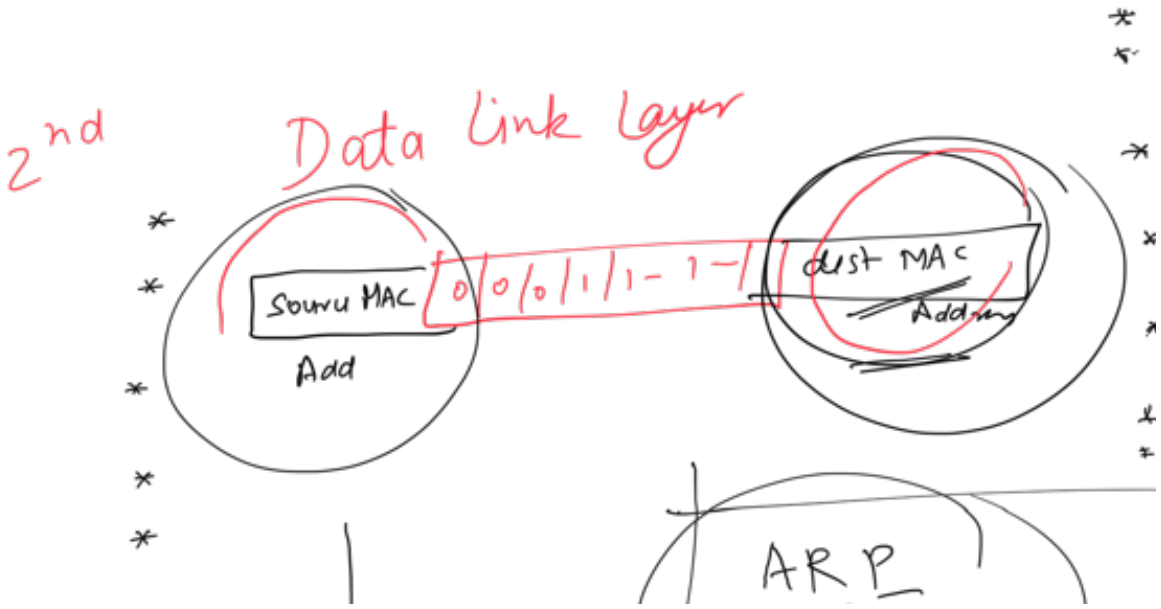

Outside
XYZ
II
P3
private IP
private IP
private IP
A
B
C
IP
proxy
VPN

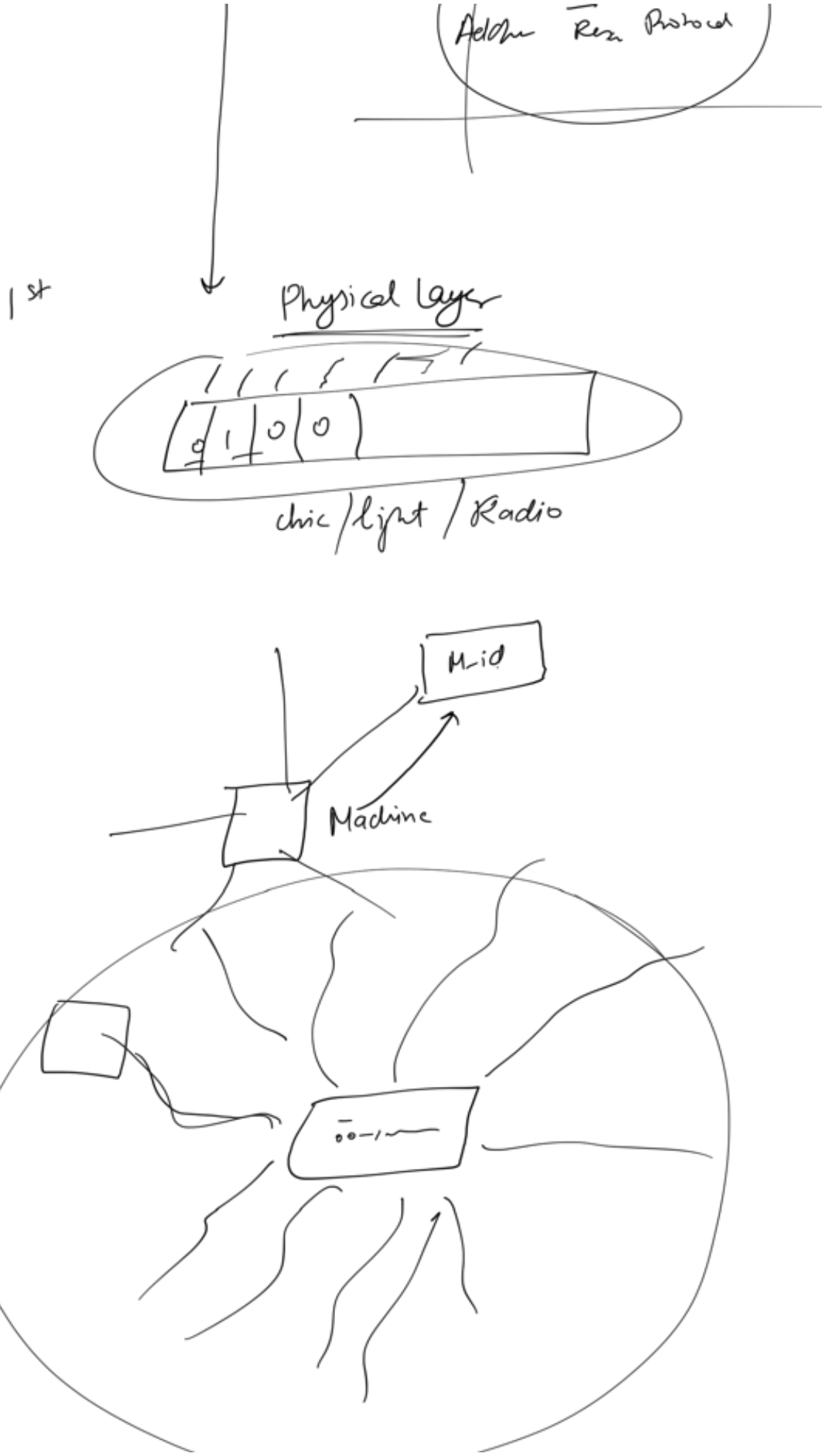

Reverse Proxy
Load Balancer
Gateway
Proxy ≡ Gateway to the outside world
Reverse Proxy ≡ Gateway to the inside world
Cisco conn
domain
user-id
IP
IP
netflix
netflix.com
Reref
VPNs
IP
India

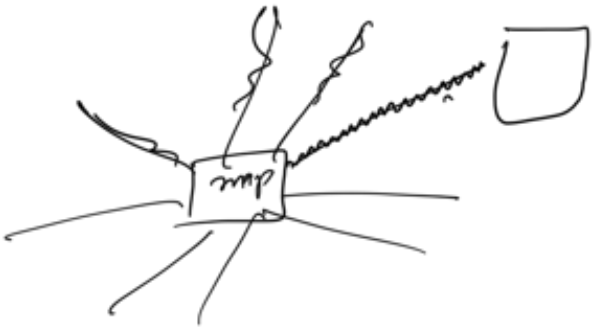

10.0.0.2
Laptop A
MAC A
LAN
Proxy
10.0.0.1
Router
WiFi
Laptop B
MAC B
Laptop C
MAC C
WiFi
Static
10.0.0.4
10.0.0.3
M1
M2
M3
MAC M1
Intranet
Router
Router
OSI Layers
Open Systems Interconnection


HTTP
GET /index
7th
Application Layer
originates of request happens
6th
Presentation Layer
Encode your request
Encryption
Compression
HTTPS


5th
Session Layer
S_port
d_ip-1.0
d
4th
Transport Layer
3rd
Network Layer
S-p
d-p
S_ip
d_ip
S
d
S
d
S+p
d


2nd
Data Link Layer
Sourv MAC
Add
0 0 0 1 1 1 - 1 - 1
dest MAC
ARP


1st
Physical layer
1 0 0
chic / light / Radio
M-id
Machine



7 Application

6 Presentation

5 Session

Client

4 Transport

3 Network

2 Data Link

1 Physical Layer

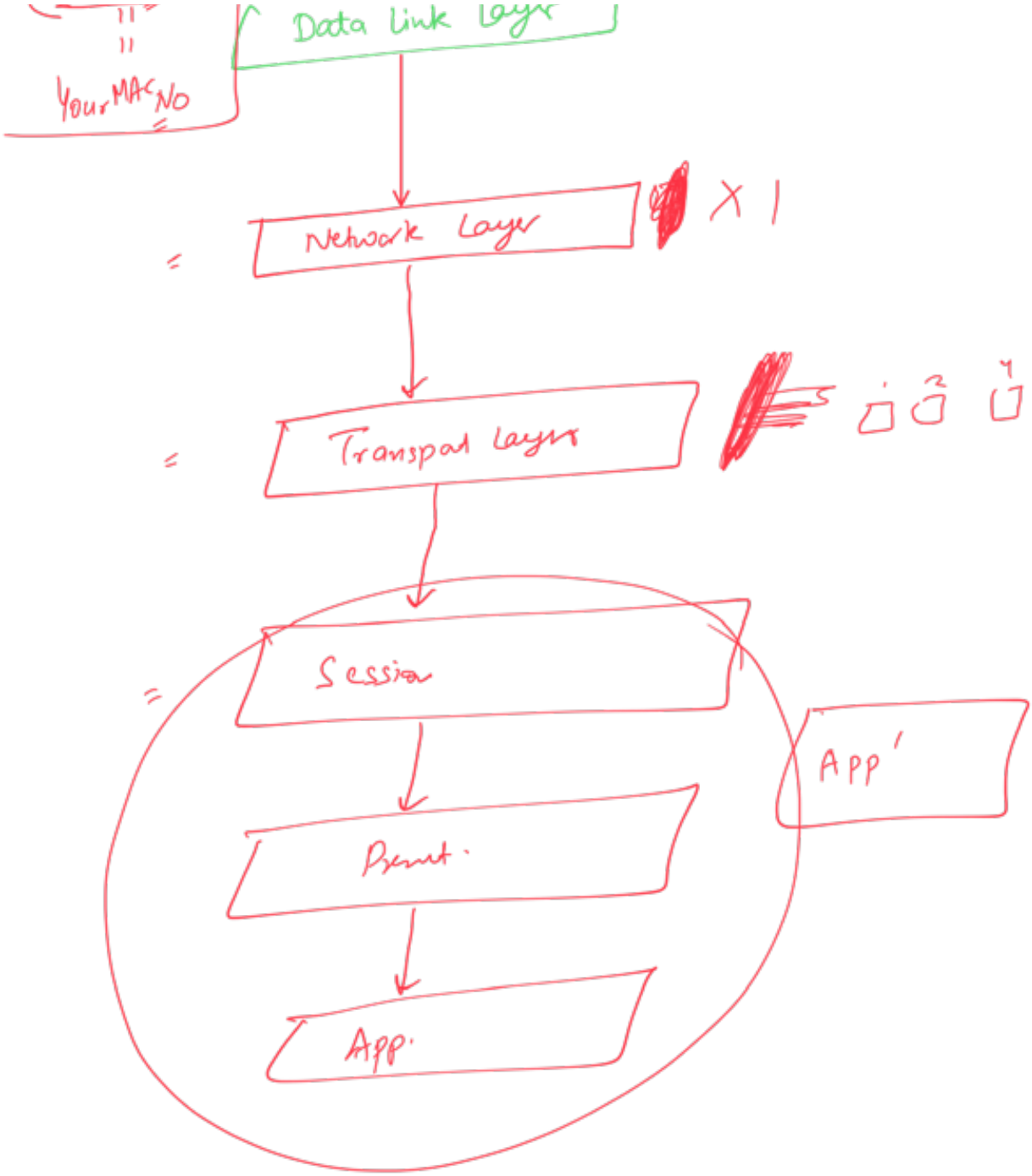
Physical Layer

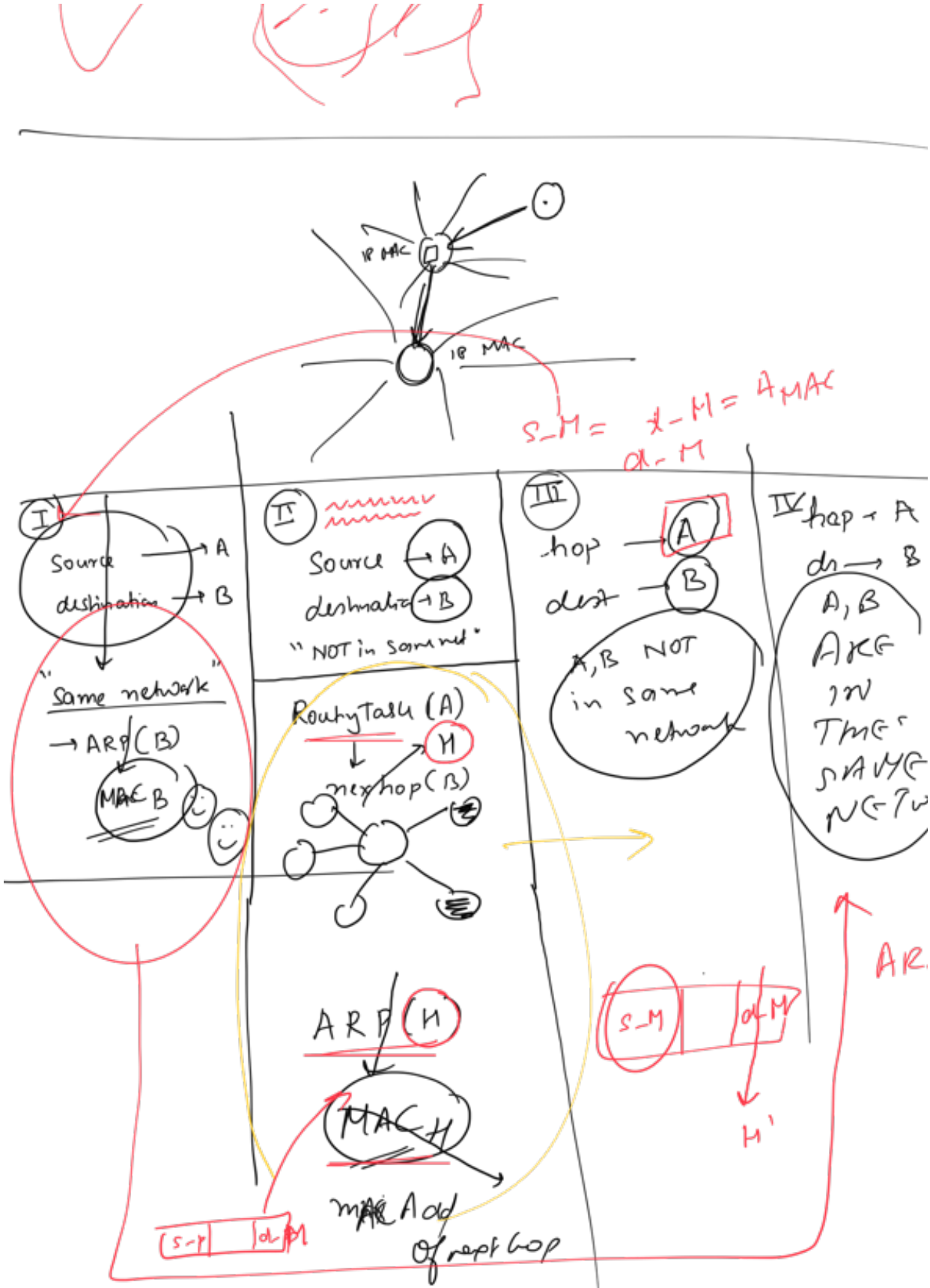
devices

X N

Dest MAC No

X N


Data link layer
Your MAC No
Network Layer
X 1
Transpal layer
Sessio
Presnt.
App.
App'


IP MAC
IP MAC
S_M = A_MAC
d_M
I
Source → A
destination → B
"Same network"
→ ARP(B)
MAC B
II
Source → A
destination → B
"NOT in same net"
RoutyTalk (A)
H
nexthop(B)
ARP(H)
MAC H
MAC Add
of nexthop
S_M
d_M
H'
III
hop → A
dest → B
A, B NOT in same network
IV
hop → A
dr → B
A, B
AKG
IN
TME
SAVE
NETW
AR

CAP THEOREM

only guarantee $\frac{2}{3}$ properties simultaneously

C ≡ Consistency

A ≡ Availability

P ≡ Partition Tolerance

(I) Consistency → Immediate Consistency

Any read that you do, must return the value after the latest write

(II) Availability

II Availability

Every and any request that goes to a non-dead server, must be responded to

---

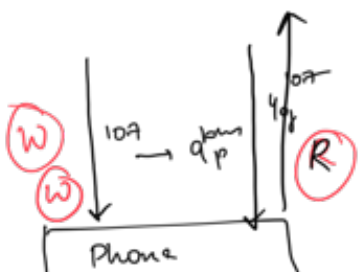
III Partition Tolerance

The system as a whole is allowed to have partitions

You have accepted the fact that network partitions can happen at any time

and the system as a whole will NOT stop functioning

Event Reminder Service

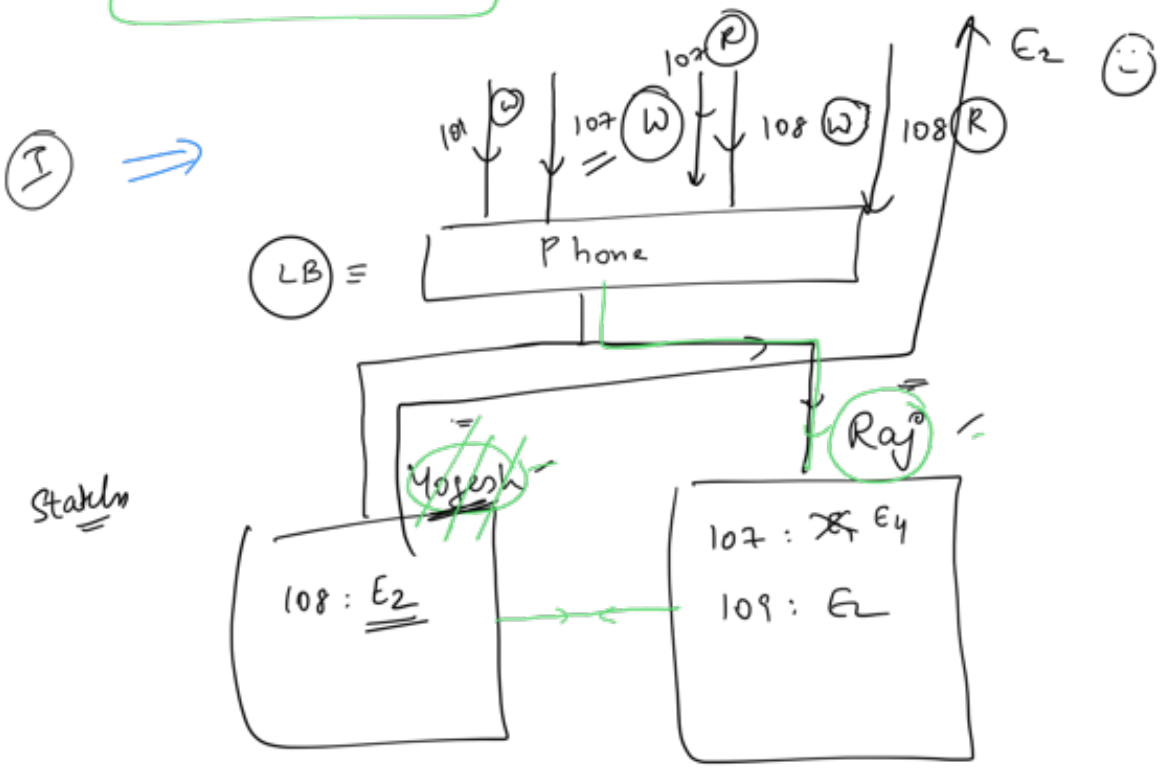


DOCS NOT Require Partition Tolerance

107 - Tom

Consistent

Availability

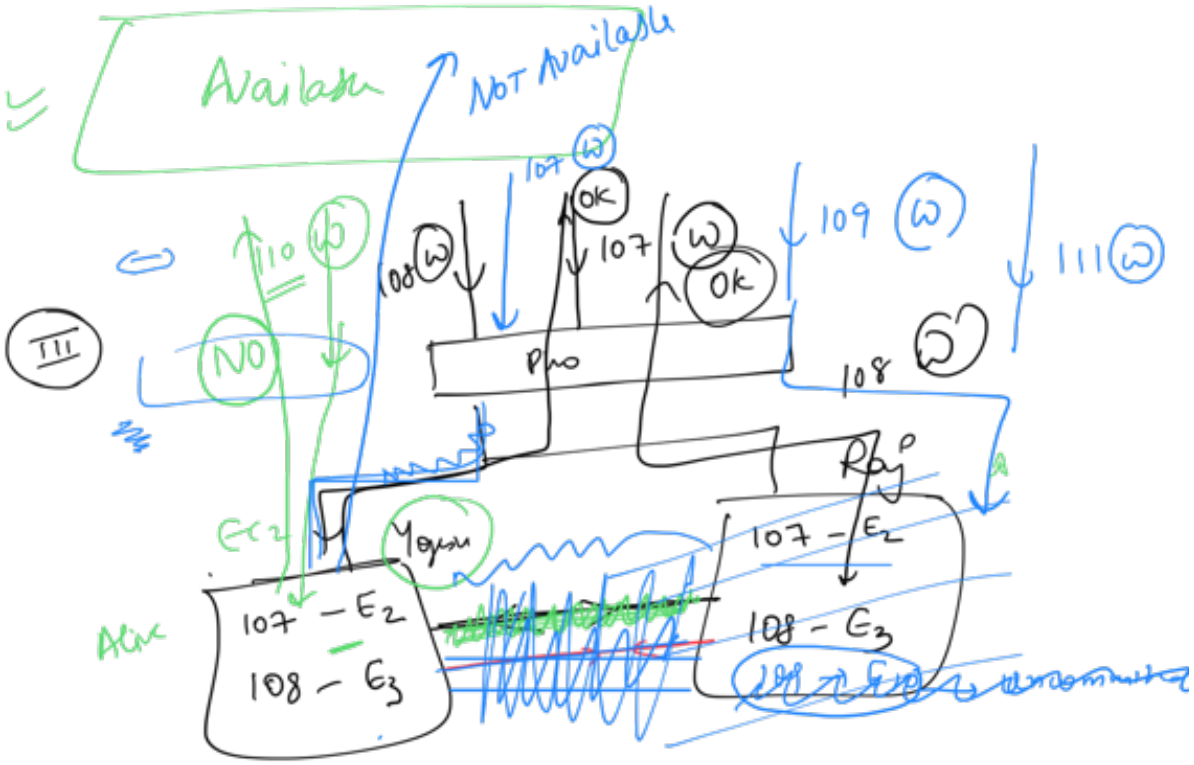


Stable

Partition Tolerance

NOT CONSISTENT

2/3



CONSISTENT

PARTION TOLERANT

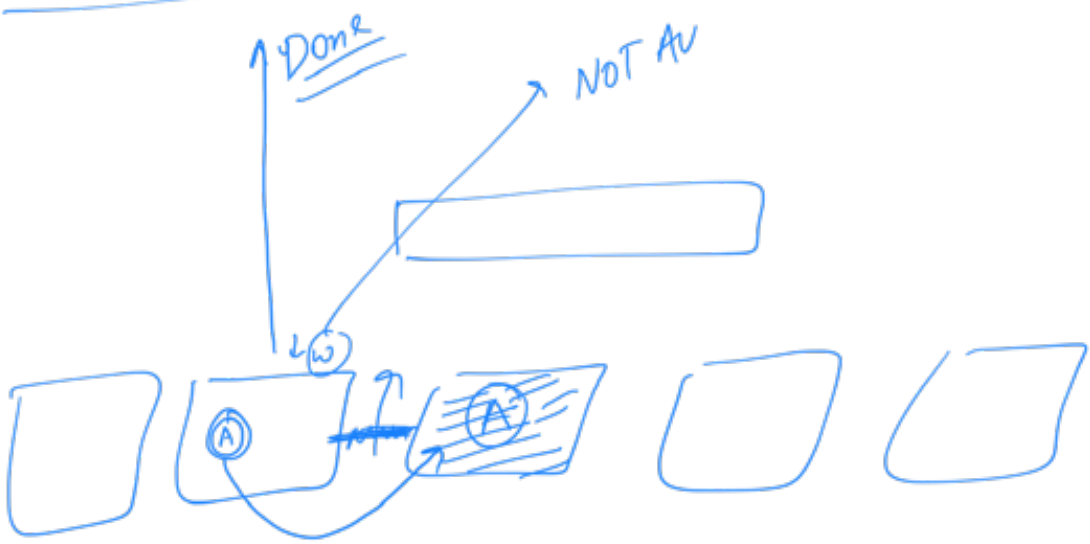
NOT AVAILABLE

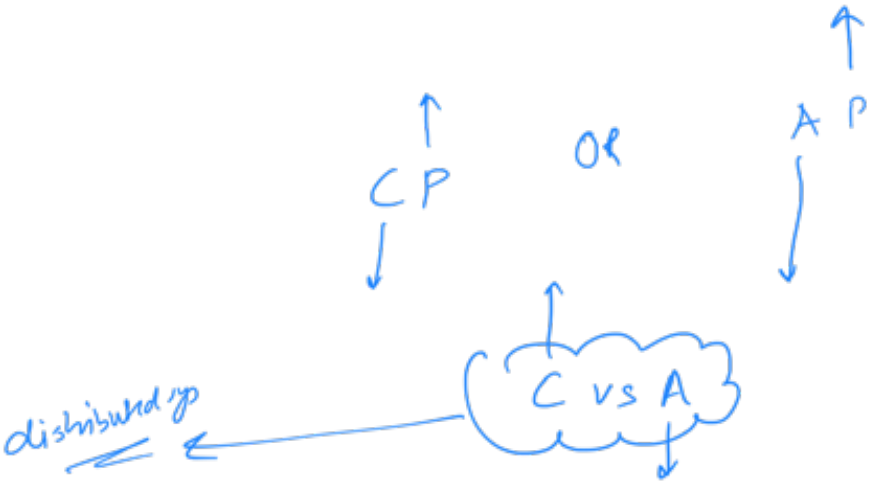
→ CP OR AP

CA — if you are only

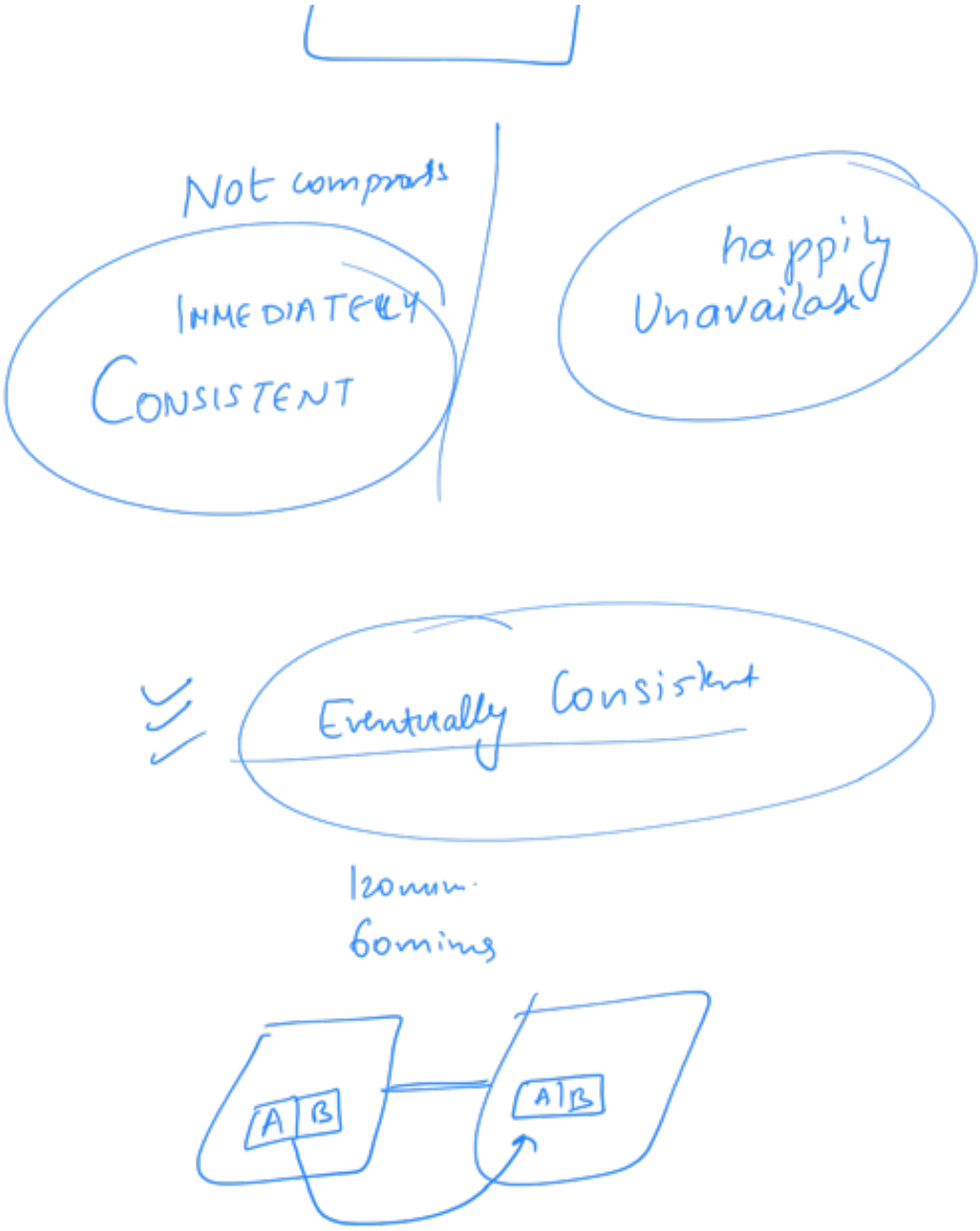

Vertical scaling

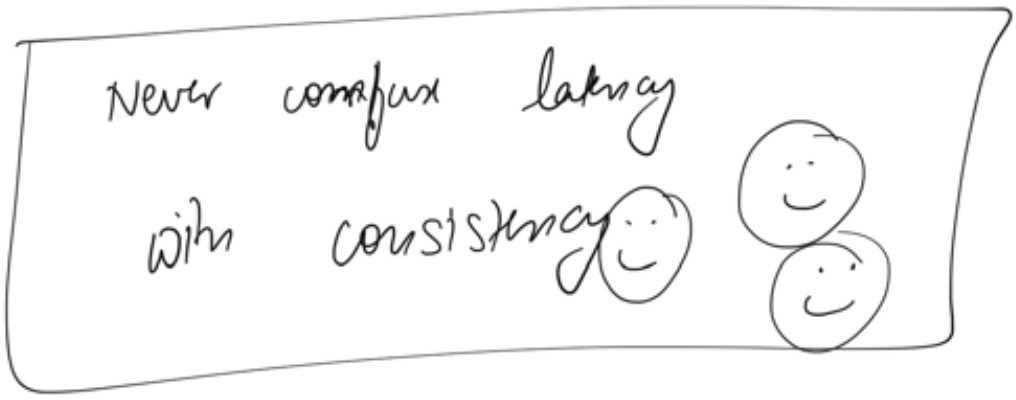

L1
L2

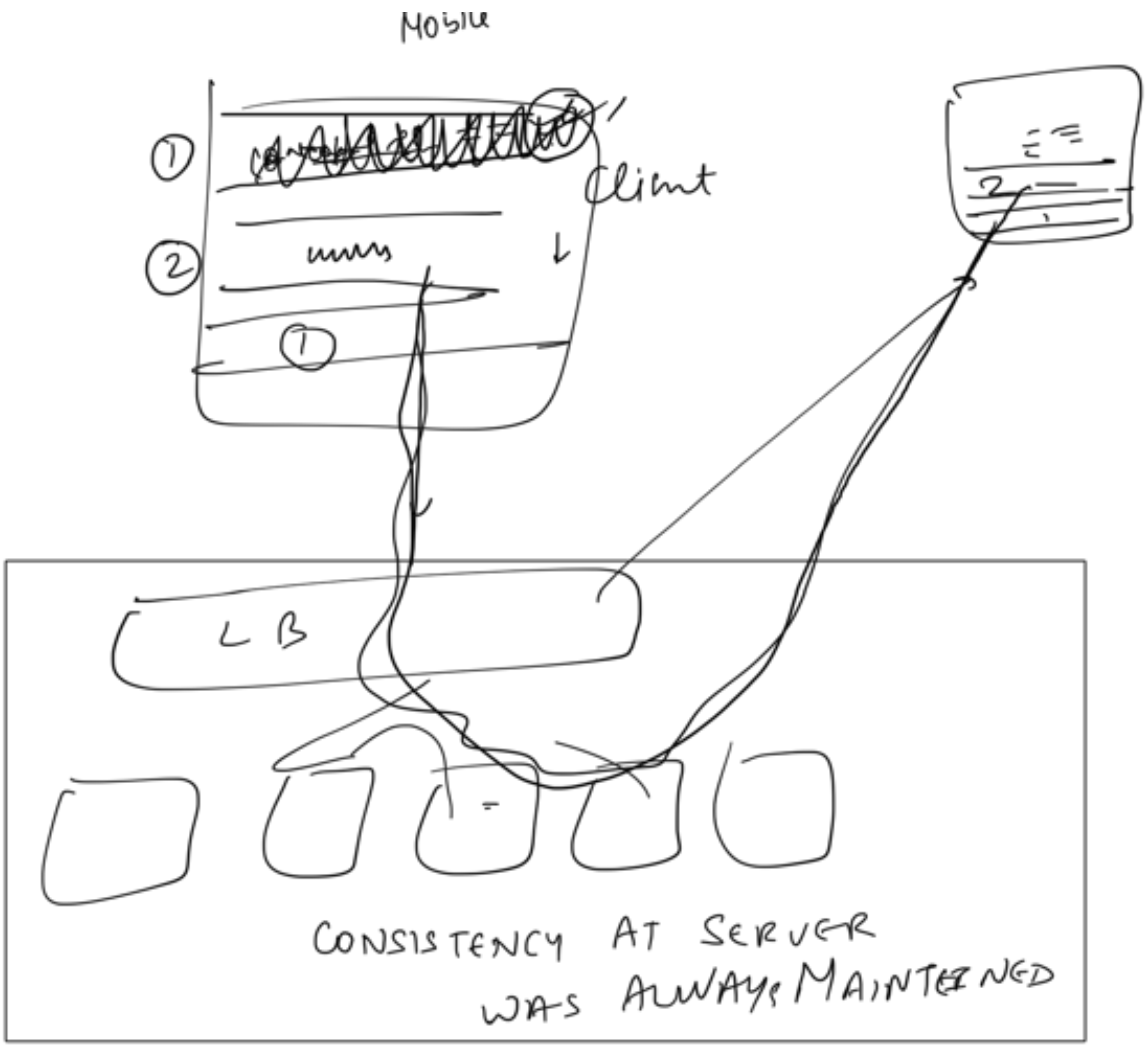

Done
NOT AV
W
A
A

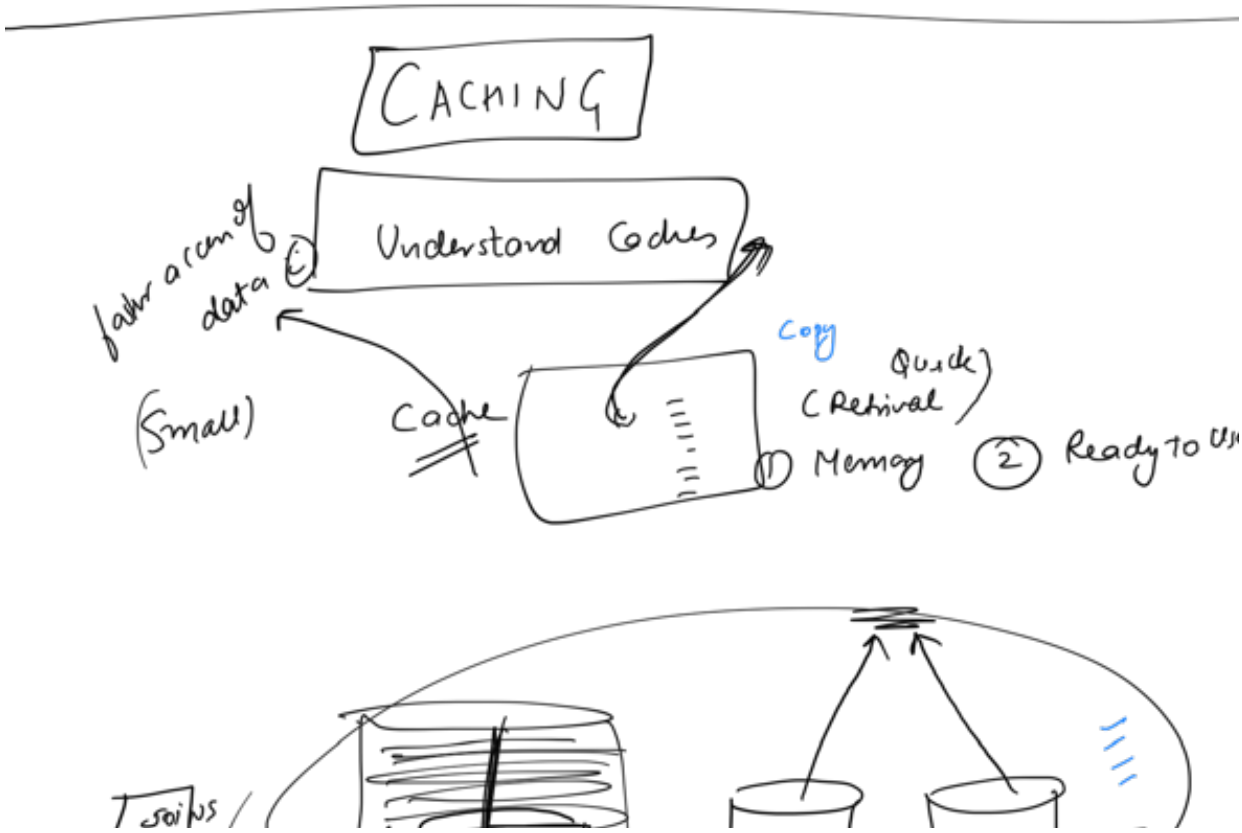

CP
OR
AP
C vs A
distributed sys

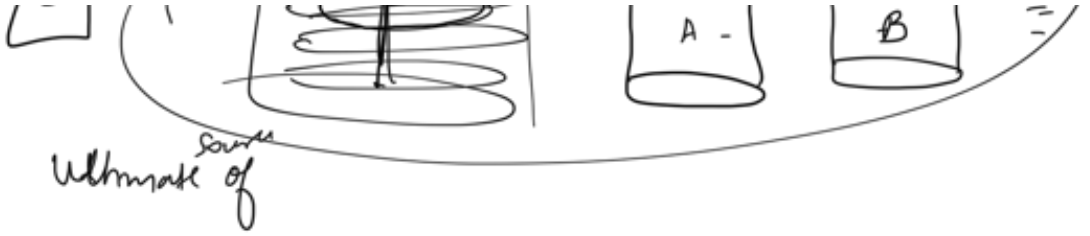

CA
not scaled

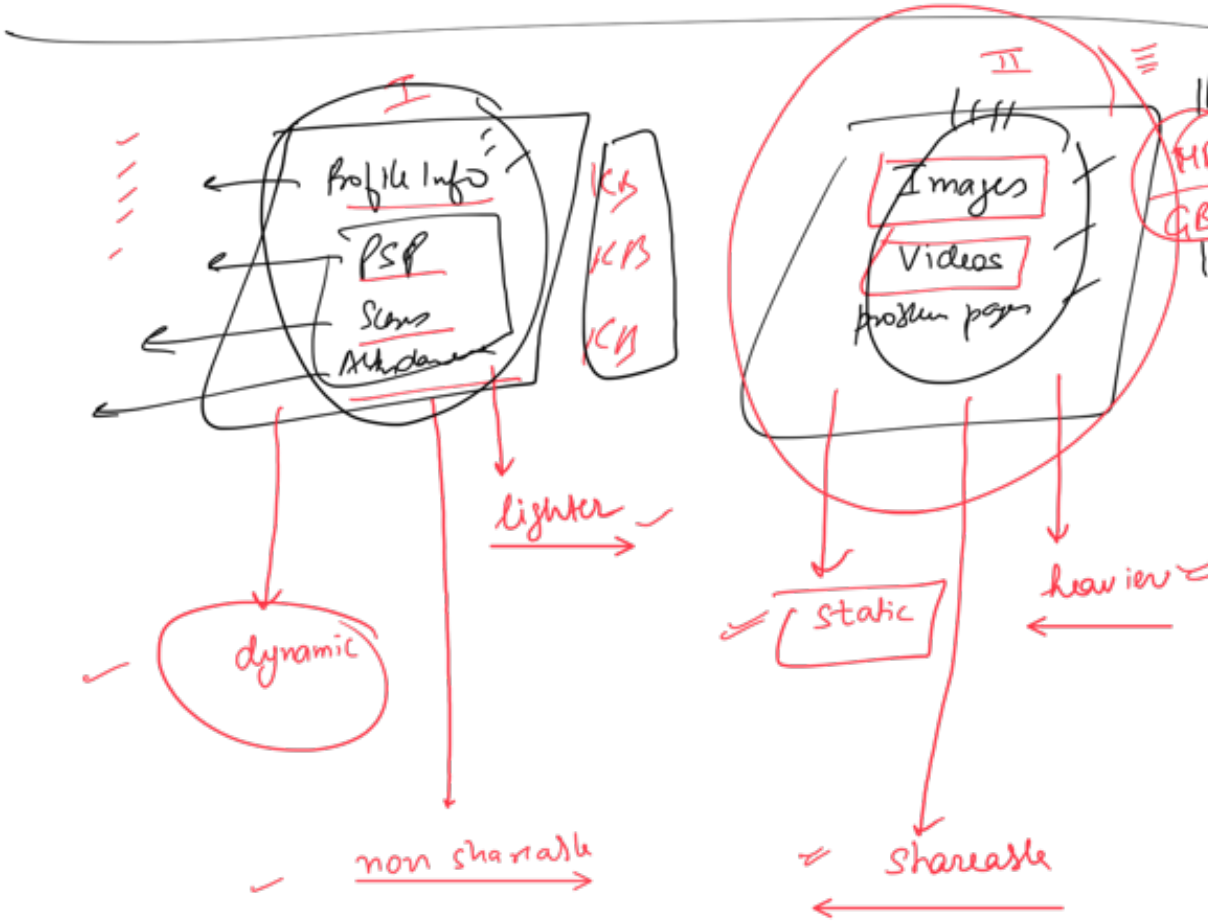

Not compats
IMMEDIATELY
CONSISTENT
happily
Unavailab
Eventually Consistent
120mm.
Coming
A B
A B

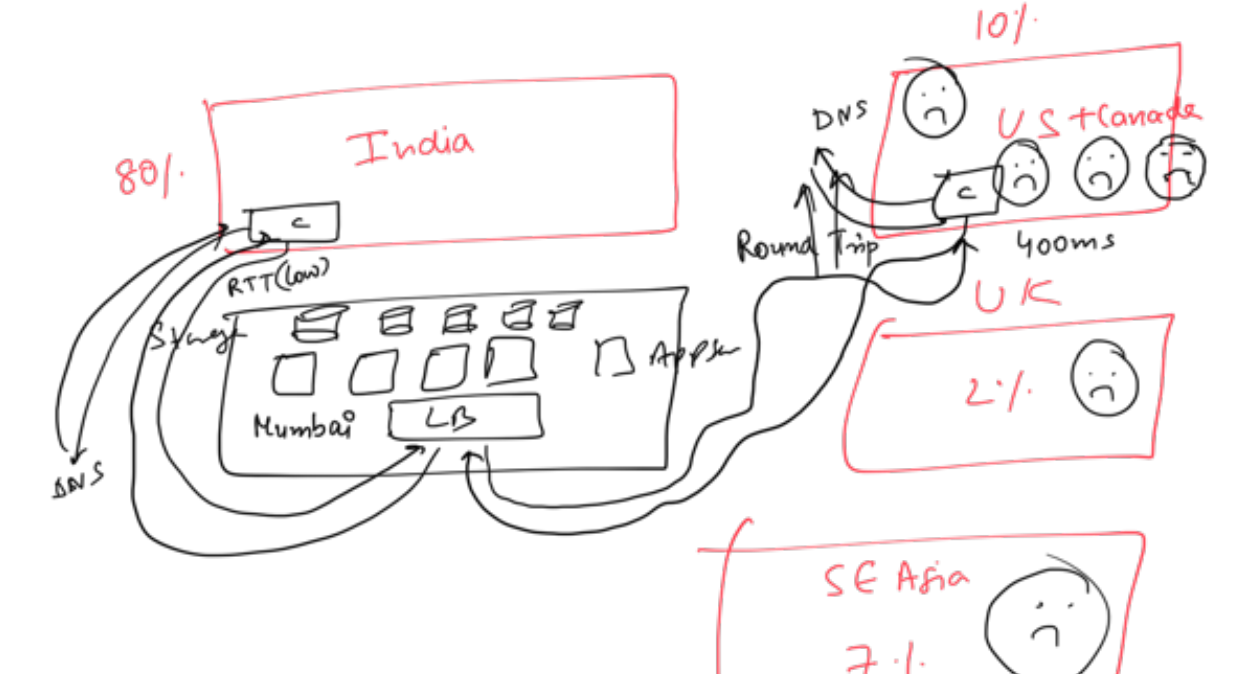

Never confuse latency
with consistency


Mobile
Client
LB
CONSISTENCY AT SERVER
WAS ALWAYS MAINTAINED


CACHING
Understand Codes
Cache
(Small)
Memory
Ready to


A -
B
Ultimate Source of


I
Profile Info
PSP
II
Images
Videos
lighter
dynamic
static
heavier
non shareable
Shareable


10%
DNS
US + Canada
400ms
Round Trip
UK
2%
80%
India
RTT(low)
Mumbai
LB
DNS
SE Asia
7%