01 SATURDAY OCTOBER

September'16

M	T	W	T	F	S	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

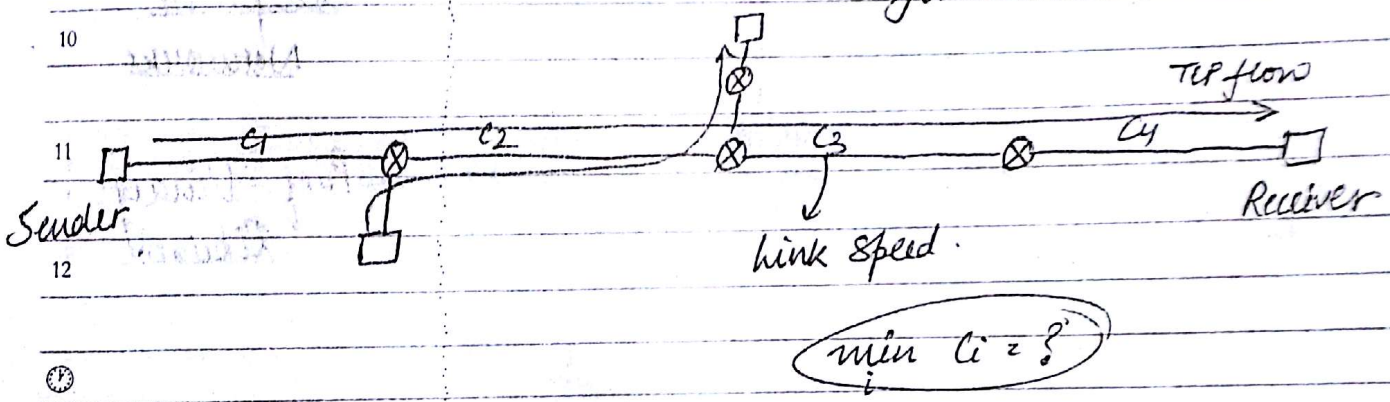
October'16

M	T	W	T	F	S	S
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

TCP. Congestion Control

09/11/16

Window, W = min (Cong. window (cw), Advrtd. Window)
↳ Congest" Control. ↳ Flow Control



min C_i = ?

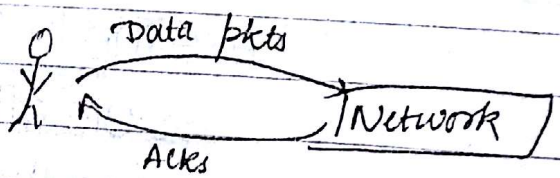
* Want ?

1) Use available bandwidth efficiently
2) Must be "fair" to other TCP flows
3) "Learn" Correct Data Rate.
   ↳ Congest" window set by Congest" Control Algorithm
4) Avoid Congestion in Network.
   ↳ We infer data rate by sending pkts. and receiving acks.

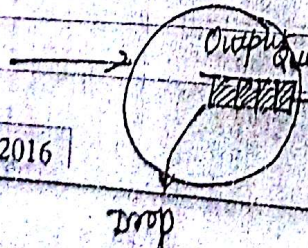
* Ideas :

02 Sunday

Observe data pkts.



Infers data rate from RTT.



$\frac{CW}{RTT} > C$ { Bottleneck Speed }

2016

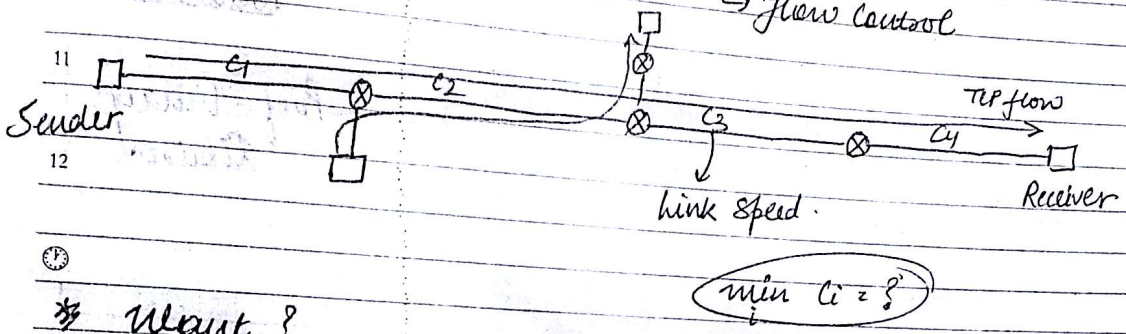
MESSAGES

01 SATURDAY
OCTOBER

TCP Congestion Control

09/11/16

Window, W = min (Cong. Window (CW), ← Congest^n Control.
Advert. Window)
↳ Flow Control



min C_i = ?

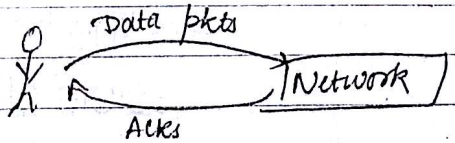
Want ?

1) Use available bandwidth efficiently
2) Must be "fair" to other TCP flows
3) "Learn" Correct Data Rate.
   ↳ Congest^n Window set by Congest^n Control Algorithm
4) Avoid Congestion in Network.
   ↳ We tune data rate by sending pkts. and receiving acks.

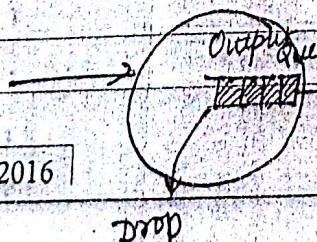
Ideas :

02 Sunday

Observe data pkts.



Tune data rate from RTT.

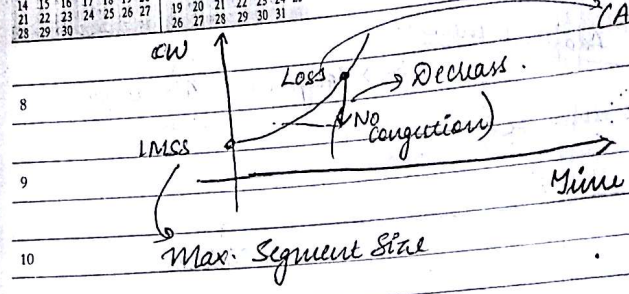


Packets/sec

$\frac{CW}{RTT} > C$ { Bottleneck Speed }

MESSAGES

2016

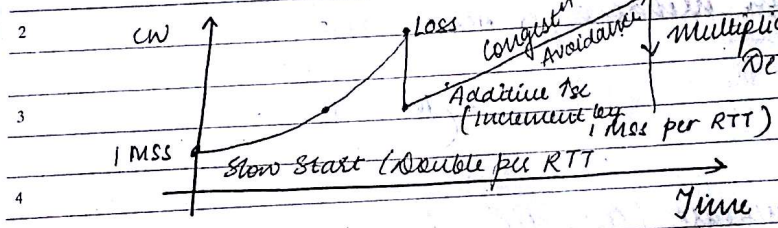


RTT = 120 ms

MSS = 12000 bits

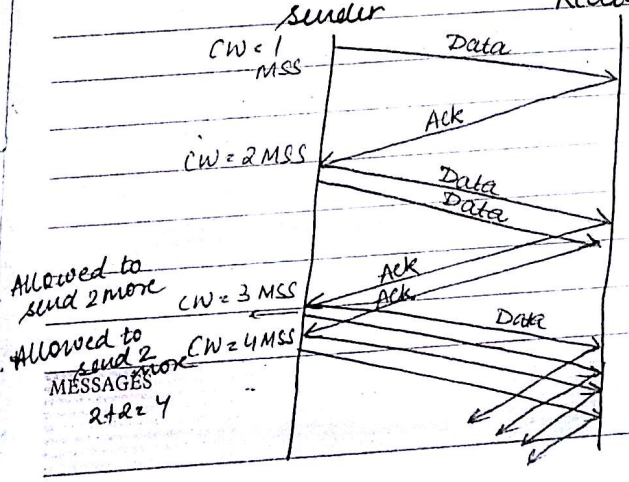
$\frac{MSS}{RTT} = \frac{12 \times 10^3}{120 \times 10^{-3}} = 10^5$ bits/sec.

= 0.1 Mbps



Slow Start

CW = 1 MSS initialised



On receiving every Ack, at Sender

CW = CW + 1 MSS

Provided

CW < SS-Threshold.

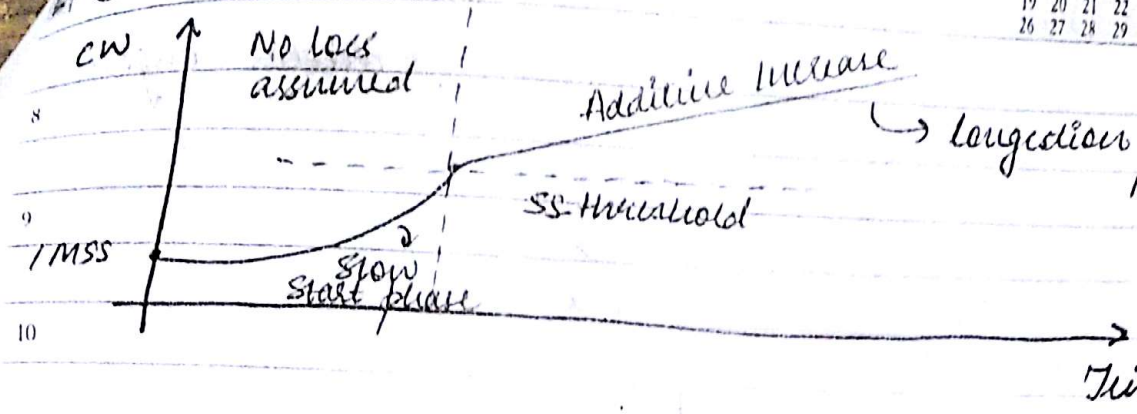
↓

Slow Start Threshold

2016

Scanned by CamScanner

04 TUESDAY
OCTOBER



Additive Increase : Want to increase by 1MSS per RTT

ACKs per RTT $\approx \frac{CW}{1MSS}$ (# of pkts per RTT).

Increase in Window per Ack $= \frac{1MSS}{(CW/1MSS)}$

$= \frac{MSS \times MSS}{CW}$

Additive Increase (Congestion Avoidance Phase)

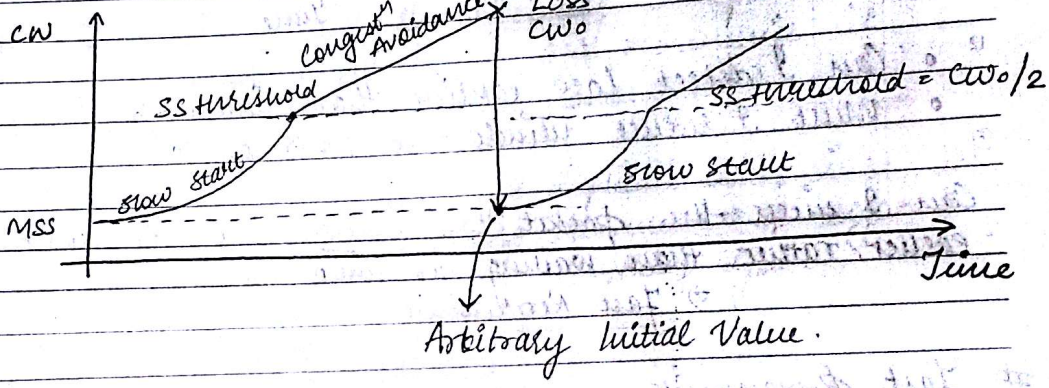
On receiving ACK,

$$CW = CW + \frac{(MSS)^2}{CW}$$

TCP :

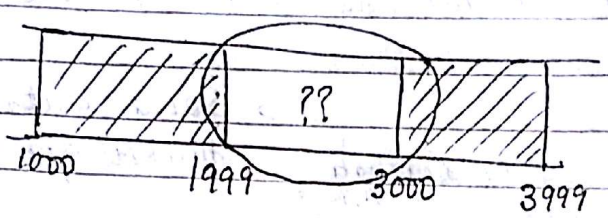
- TAHOE
- RENO
- VEGAS

- TCP Tahoe



$$CW = \alpha . CW$$

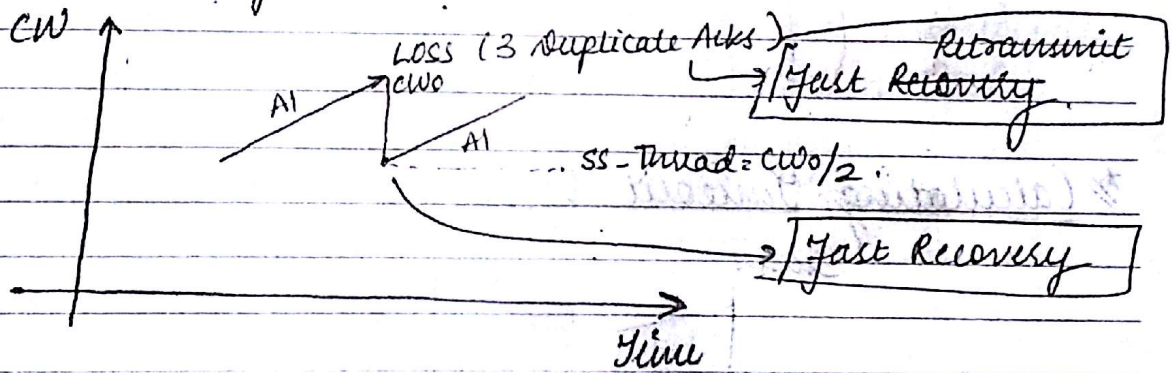
→ There might be many more losses after this, but we are sure about Seq. 2000 getting lost.



Note: Fast Retransmit: On receiving 3 duplicate acks, assume packet loss and retransmit it. (Cancel Timer).

We donot do this after 1/2 duplicate acks, since it is possible that pkt reordering occurred. ⇒ If slight delay, no sense in resending everything again.

Fast Recovery



CW = $\frac{CWo}{2}$ After 3 Duplicate Ack loss

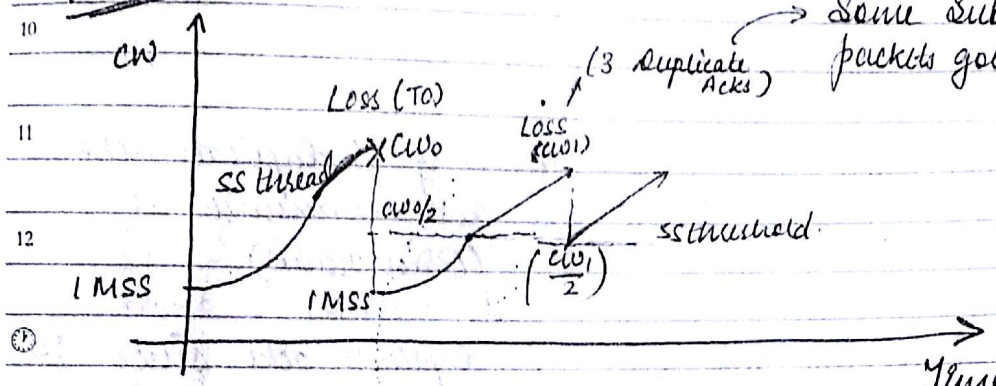
08 SATURDAY
OCTOBER

September'16 | October'16

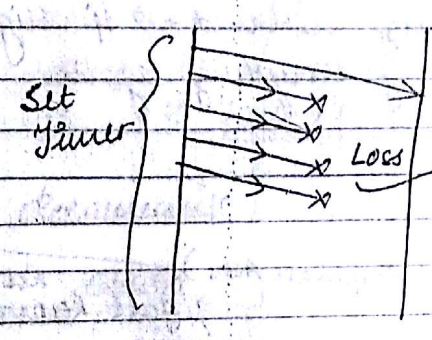
TCP Reno :

TCP Tahoe + TCP Retransmit + TCP Recovery.

TCP Reno

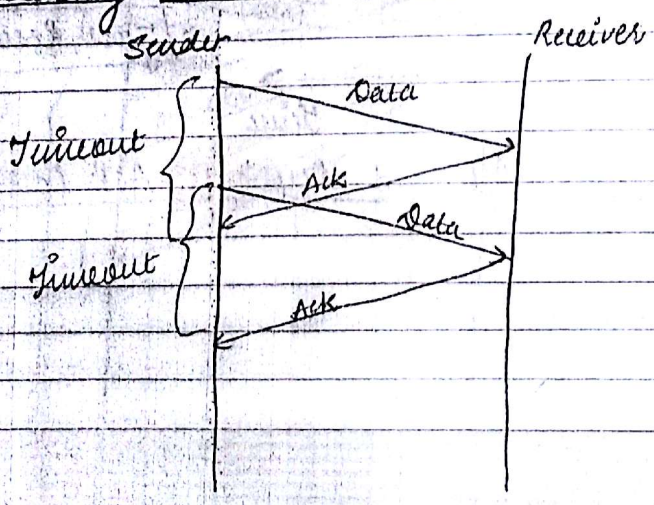


→ Some Subsequent packets got through.



→ Many Losses so no duplicate Acks (Bad situation).

✱ Calculating Timeout Value



09 Sunday

MESSAGES

2016

November'16 | December

RT-

2 ✱ Sampl

ti

N

✱ TCP

1k
2k
3k

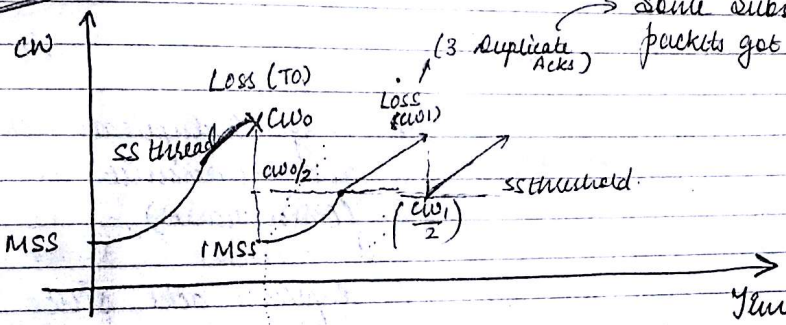
MESSAGES

08 SATURDAY
OCTOBER

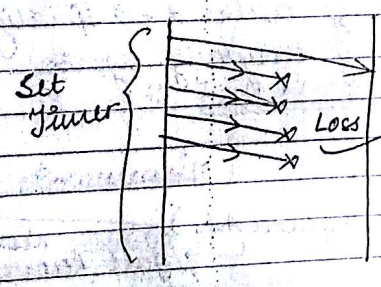
TCP Reno :

TCP Tahoe + TCP Retransmit + TCP Recovery.

TCP Reno

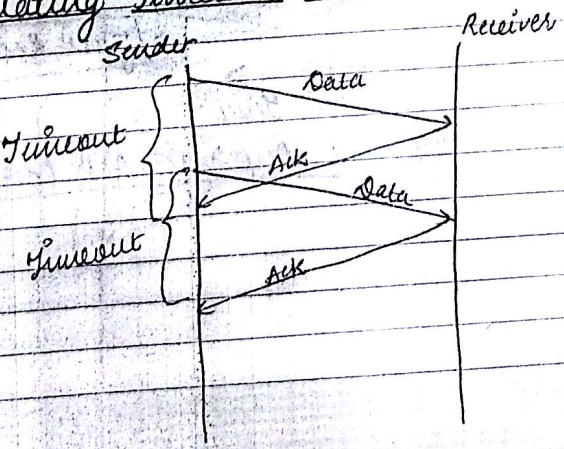


(3 Duplicate Acks) → Some Subsequent packets got through.



→ Many losses so no duplicate Acks (Bad situation).

Calculating Timeout Value



09 Sunday

MESSAGES

2016

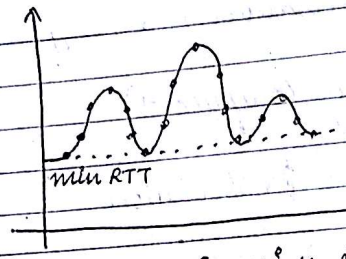
RT

Sampl

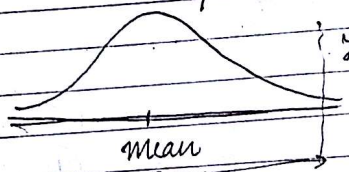
TCP

1t
2t
3t

MESSAGES



Need to set the timeout & Wait, full, etc.



* ~~Samples of RTT $x_1, x_2, x_3, \ldots x_N$~~

$$\bar{x} = \frac{1}{N} \sum x_i$$

$$\text{STD. DEV} = \sqrt{\frac{1}{N} \sum_i (\bar{x} - x_i)^2}$$

Computationally Involved. Difficult to squareroot everytime.

Instead Use.

$$\text{Mean Deviation} = \frac{1}{N} \sum_{i=1}^{N} |x_i - \bar{x}|$$

* ~~TCP Algorithms for Timeout:~~

1. Have estimated (EST) RTT.
2. Sample RTT → Just calculated RTT.
3. Difference = Sample RTT - Est RTT $|x_i - \bar{x}|$

MESSAGES

2016

11 TUESDAY OCTOBER

September'16 / October'16

M	T	W	T	F	S	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

M	T	W	T	F	S	S
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

November'16 / Dec

M	T	W	T	F	S	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

M
5
12
19
26

July

S	M
	dama
12	4
19	11
26	18
	25

4) Deviation = $(1-\alpha)$ Deviation + α |Difference|

5) Mean Deviatn (Easy computatn)

5) Timeout = $\mu \times$ Est.RTT + $\phi \times$ Deviation.

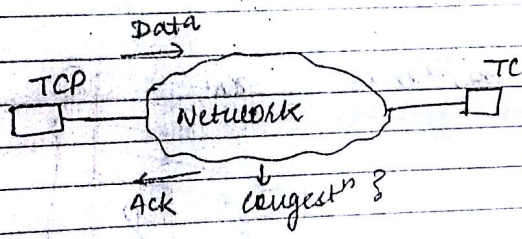
Usually, $\mu = 1$

$\phi = 4$.

TCP Vegas:

15/11/2016

Tahoe Reno

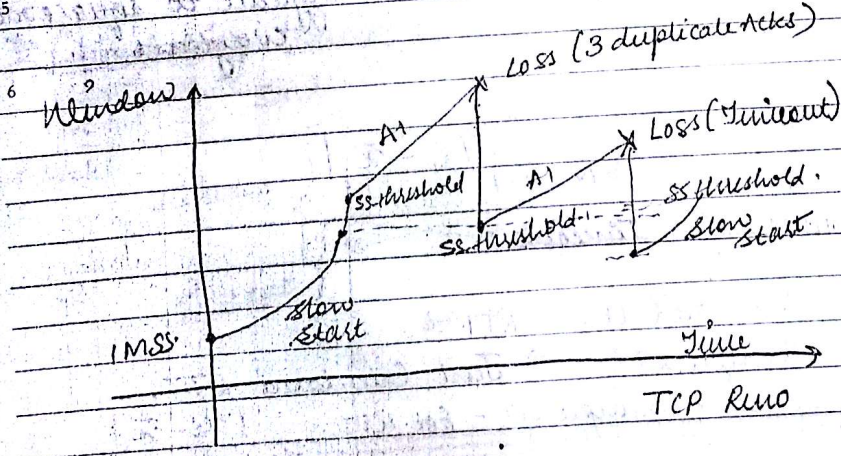


XXXXX

Fill Queue

Pkt Drop

Timeout — 3 Dup ackd.



TCP Reno

MESSAGES

2016

Tahoe Reno

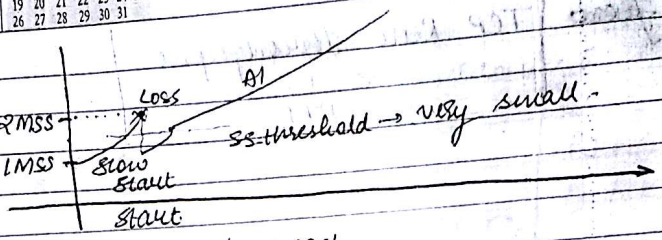
Failure

Q.1. A

Q.2. E

Window

MESSAGES

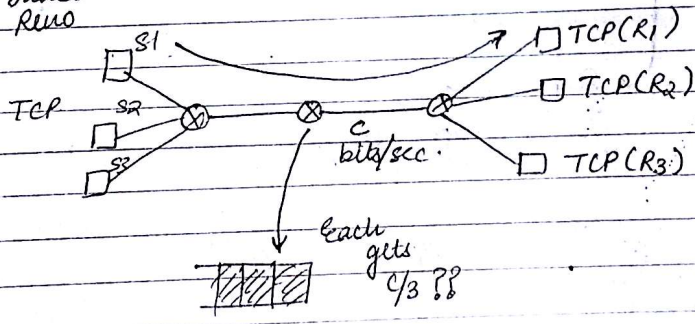


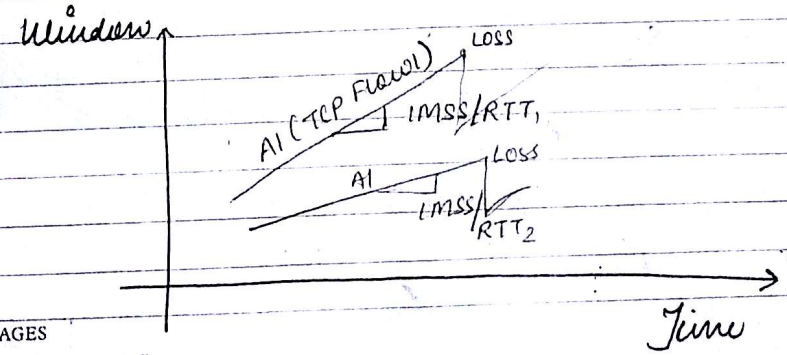
Experience loss early (Unlucky)

Fairness:

Q.1. Are TCP flows of same type (Tahoe, Reno, etc.) fair to each other (Do they get same throughput)?

Q.2. Are TCP flows of different types fair to each other?





MESSAGES

MESSAGES

2016

2016

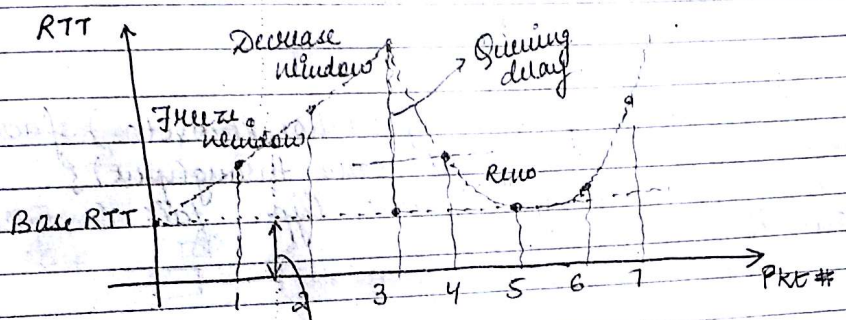
Queue: Drop if 3 dup ackt.

Scanned by CamScanner

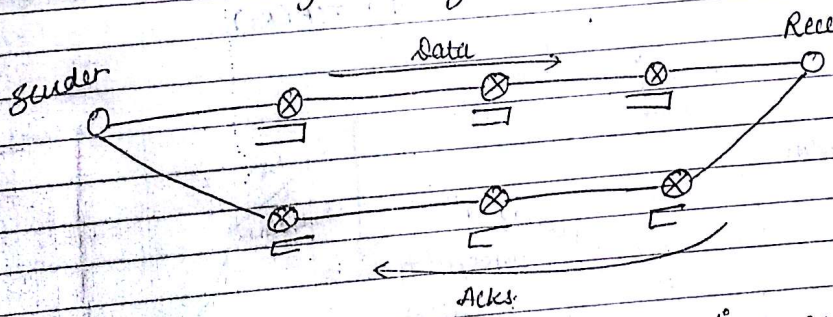
13 THURSDAY
OCTOBER

TCP Reno Throughput $\propto \frac{1}{RTT}$

Vegas



Drop delay,
Forwarding delay.



We are more concerned about congestion on data path than Acks path, we can measure one-way delay.

Clock

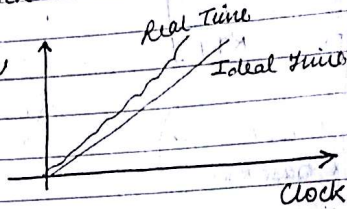
Vegas

In cong avoidance phase, (CW > ss

MESSAGES

2016

① Data Rate higher than acks.
② Clock at receives not synchronised ⇒ using one-way delay is not used?



⇓
Packet Time Stamp & Ack Time Stamp unrelated.

Vegas:

① Base RTT : Minimum RTT observed so far.
RTT : Current measured RTT.

$$\text{Expected Rate} = \frac{\text{Congest}^n \text{ window}}{\text{Base RTT}}$$

↳ Data rate I would get if no congestion (no queuing delay).

$$\text{Actual Rate} = \frac{\text{Congest}^n \text{ window}}{\text{RTT}}$$

Actual Rate < Expected Rate

Diff = Expected Rate − Actual Rate

In congest^n avoidance phase, (CW > SS_Thresh)

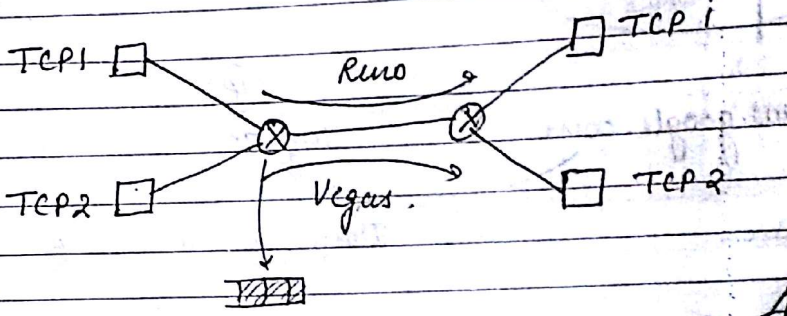
- Diff < α → Additive increase like Reno
- Diff > β → Additive decrease of congest^n window
- α < Diff < β → Don't change congest^n window

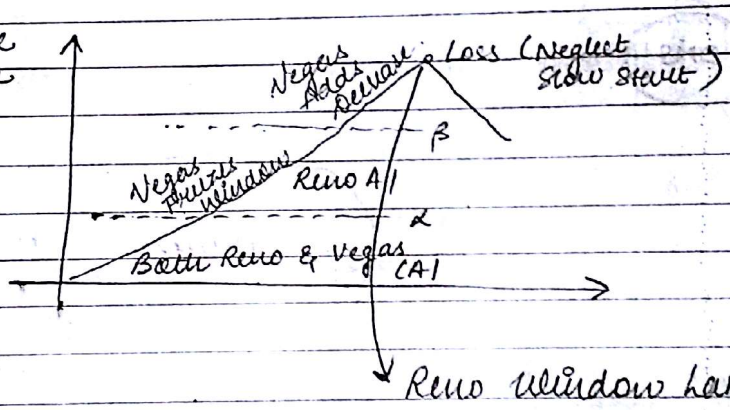
receives
0

data
ay

Vegas couldn't succeed due to poor performance.
No ~~value~~ good values of α, β were found.

Vegas v/s Reno





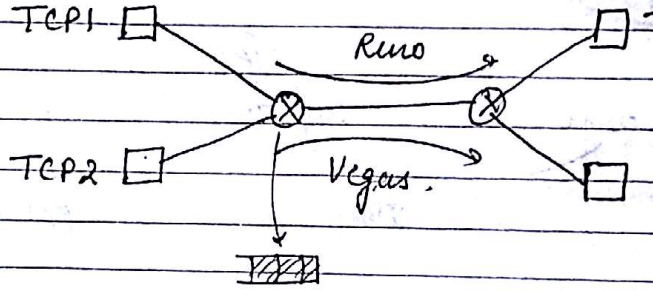
Assume "Congestion Avoidance Phase" only.
(Neglect Slow Start)

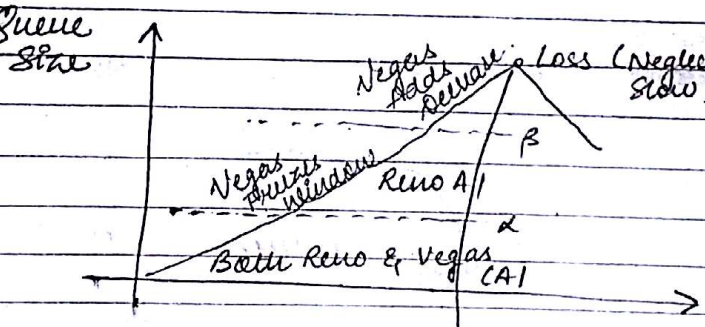
- TCP Vegas is like a gentleman. Decreases the window size as soon as the queuing delay ↑ses. while Reno keeps on.

MESSAGES

Vegas couldn't succeed due to poor performance.
No ~~value~~ good values of α, β were found.

Vegas V/s Reno





Assume "Congestion Avoidance Phase" only.
(Neglect Slow Start)

Reno Window larger.

- TCP Vegas is like a gentleman. Decreases the window size as soon as the queuing delay rises. while Reno keeps on.

MESSAGES

2016

TUESDAY
18 OCTOBER

16/11/2016

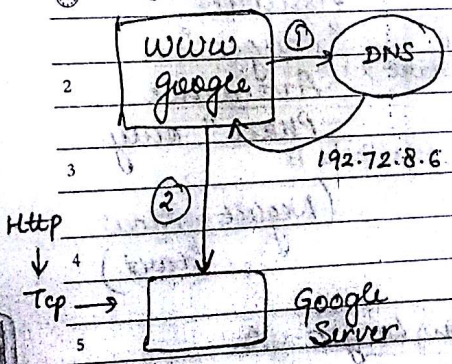
User

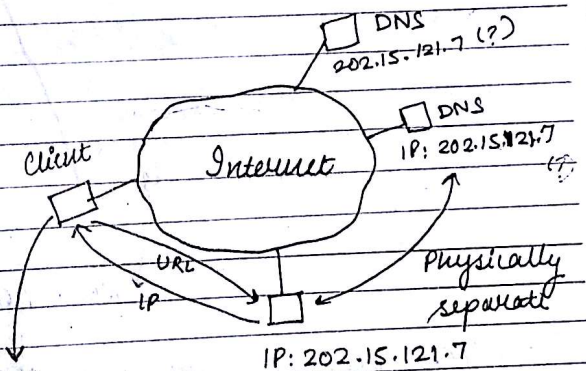
Application → Port
Transport → Protocol Number
Network → IP address
DLL → MAC address
Physical

URL: www.google.com → IP: 192.72.8.6

DNS : Domain Name System

Browser





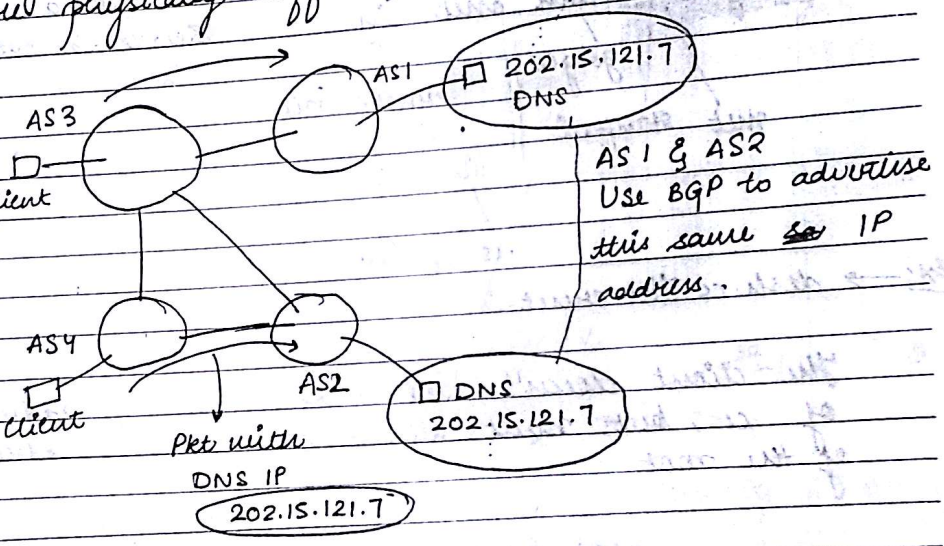
Cache
How long to cache?

Single Server
① Too much traffic
② Table size too large
③ Single point of failure
④ Easy target for attack
⑤ Making changes/updates cumbersome

MESSAGES

2016

They all use the same logical address but physically different



AS1 & AS2 Use BGP to advertise this same IP address.

Clients send DNS packets which reach any one of physical servers (anycast).

→ This solves ①, ②, ③ & ④ issues. But table size is still to large. What to do? As each one of them has to keep information of all others.

⇒ Need a hierarchical structure for this.

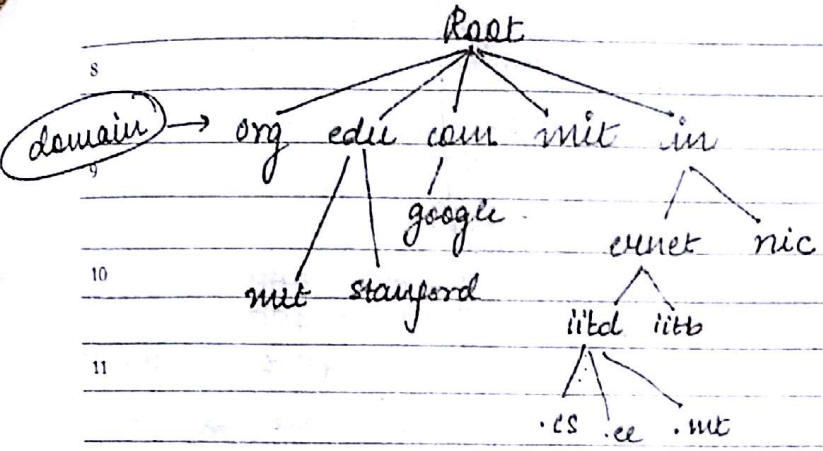
DNS Hierarchy

Root

Have DNS Server for each domain.

WK-42 DAY(293)

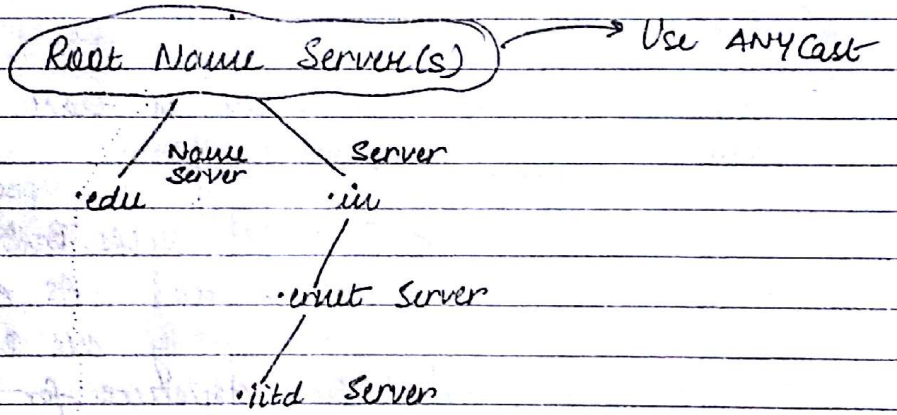
20 THURSDAY OCTOBER



We have a DNS Server for each domain.

Ex:- dosh.cs.iitd.ernet.in

The client doesn't need to know the ip address of .cs, but must atleast know the ip address of the root.



13 logical Servers:

A. root-server.net

B. root-server.net

!

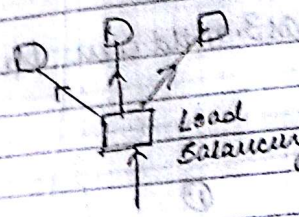
!

M. root-server.net → One IP address, many ~~IP addresses~~ physical machines distributed in internet

FRIDAY 21 OCTOBER

DNS

UDP — Specs: Max. size of pkt – 512 bytes



DNS → UDP
- Less overheads (No SYN/FIN)
- Faster

*) DNS server stores resource records.

< Name, Value, Type, Class, TTL >
- Name: Domain Name
- Value: Info. we are looking for
- Type: How to interpret value?
- Class: IN (Internet)
- TTL: Validity in secs. of soln now. $2^{32}-1$

Different for that of packet TTL (IP TTL) which dec by 1 after every hop.

Type:
'A' : IP address
'NS' : Name of host running DNS server in domain corresponding to "Name" field.
'CNAME' : Canonical Name (Alias)
'MX' : Name of host running email server for domain specified in "Name" field.

Root has : < edu, a3.nstld.com, NS >
DNS server for Edu.
< a3.nstld.com, 128.112.93.1, A >

DNS cache

Address

Address

ANYCast

many
cal
ated in
rnet

MESSAGES

MESSAGES

2016

NOVEMBER DECEMBER

22 SATURDAY
OCTOBER

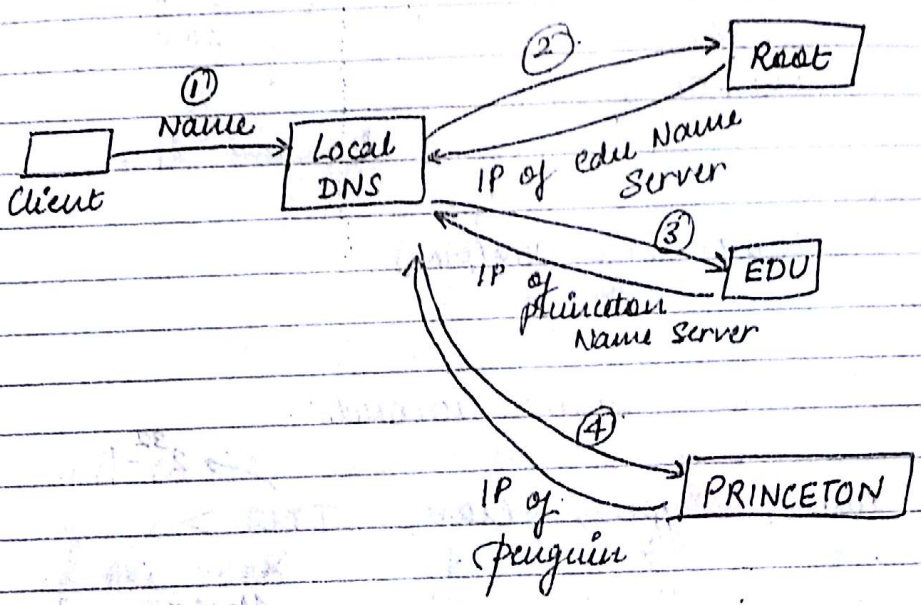
September'16

M	T	W	T	F	S	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

October'16

M	T	W	T	F	S	S
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

as3.nstld.com has <princeton.edu, dns.princeton.edu, NS>
<dns.princeton.edu, 112.191.7.16, A>



eg: penguin.cs.princeton.edu.

23 Sunday

MESSAGES

2016