(2) Third Dup Ack:

Retransmit & lose packet.

Assume (for time being) all later packets made it (continue "from end").

"Fast Recovery".

(decrease congestion window).

(3) SACK.

Retransmit lose packet. (notes?).

Further? (problem!).

Decrease congestion window.

TCP is reliable.

Lost, damaged packets are <u>re-transmitted</u>.

How do we find out (decide, guess) a packet is lost or damaged?

(1) Time-Out.

(2) Third duplicate Acknowledgment.

(3) (later) SACK.

Current situation (more or less. This is implementation dependent):

(1) Time-Out:

Retransmit lost packet,

<u>and</u> all later packets.

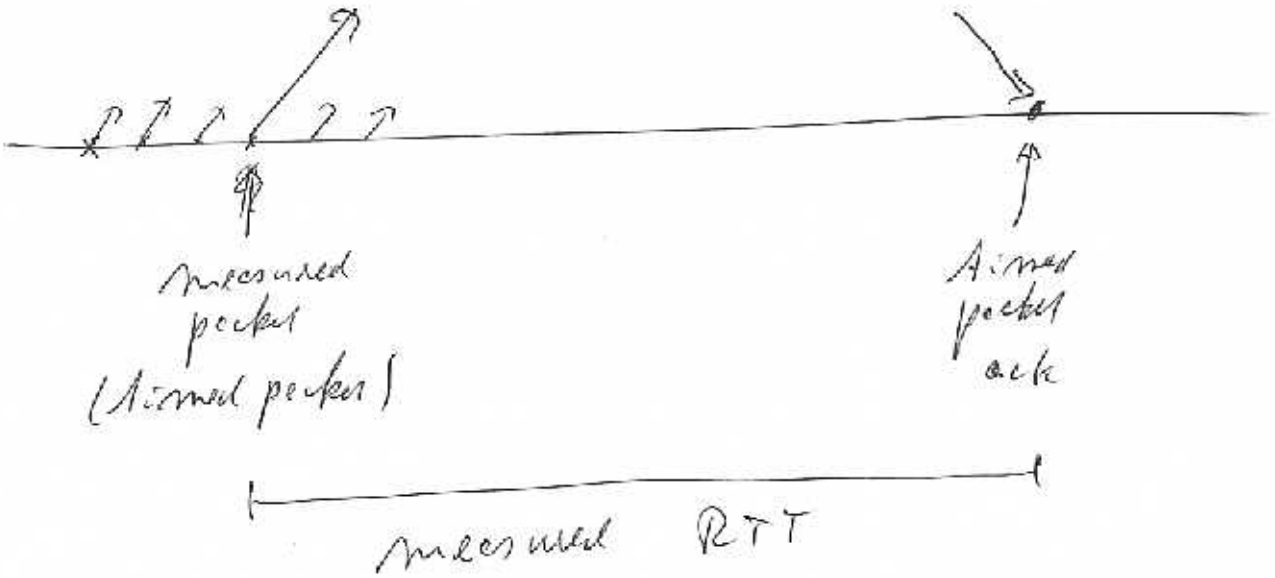
(decrease congestion window)

(Overload Control)

Round Trip Time Estimation.

Certain (not necessarily all) packets are timed: RTT measured.

(1) measure sending time
(2) measure time of ACK.
(3) difference is measured RTT.



$RTT_{meas.}$

In BSD Unix:

Only one outstanding timed packet at a time.

If "timed ack": next data packet becomes timed packet.

RTT_{meas} measured RTT

RTT_{est} estimated RTT.

RTT_{-SD} (estimated) mean and deviation of RTT.

Algorithm:

When "timed ack" returns:

$$Delta = |RTT_{meas} - RTT_{est}|$$

$$RTT_{est} = \frac{7}{8} RTT_{est} + \frac{1}{8} RTT_{meas}$$

$$RTT_{SD} = \frac{3}{4} RTT_{SD} + \frac{1}{4} Delta$$

Re-transmission Timer
(Re-transmission Time-Out)
RTO

Standard:

$RTO = 2 * RTT_{est}$

BSD Unix:

$RTO = RTT_{est} + 4 * RTT_{-SD}$
(new values)

One way or another:
"we have an RTO".

Next: how do we use this?

S/aM Ken 03/30/2004 163

Naive: ~~Back to interesting part~~
RTO, and setting timer.

A "time-out" for every packet.

Heavy on the book-keeping!

//

BSD Unix:

s/ a time-out arrives:

(1) kill RTO timer.

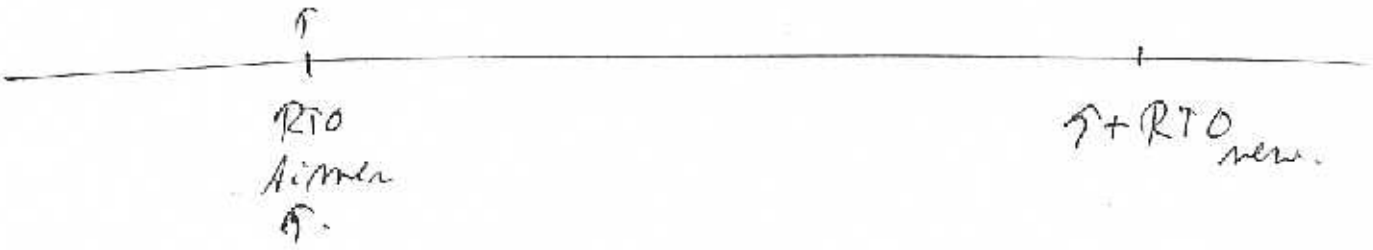
(2) Any more unacked bytes?

NO: No new RTO timer.

yes. Set new RTO timer at
t + RTO

This increases effective RTO interval even more!

If we reach RTO timer:



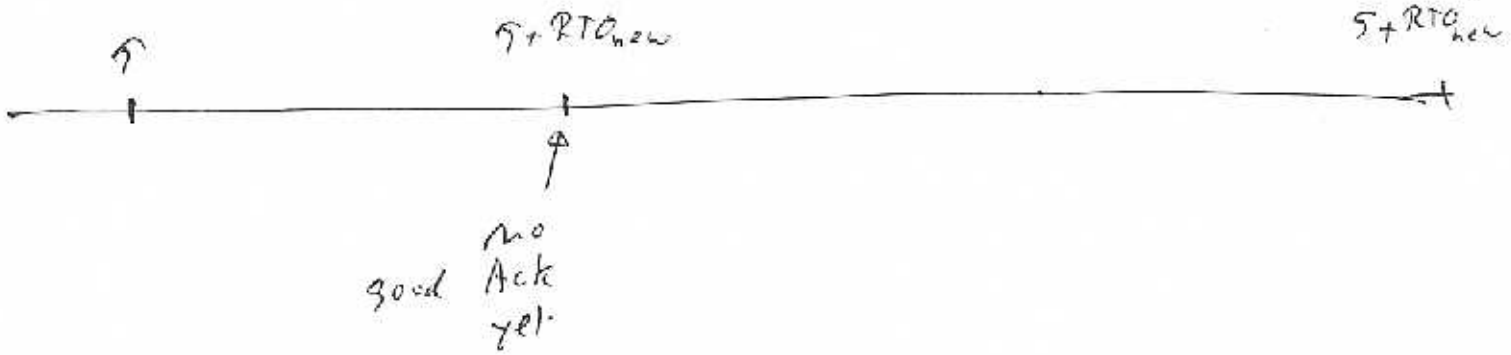
(1) Re-transmit "apparently missing" packet -

~~(Most simple solution~~

(2) Double RTO.

$RTO_{new} = 2 * RTO_{old}$

Next:

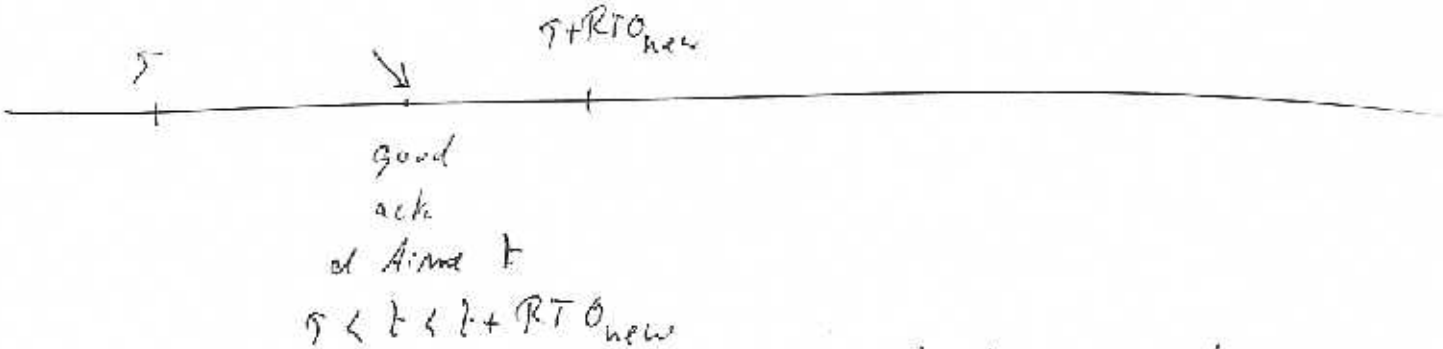


$RTO'_{new} = ?? * RTO_{new}$. etc. keep doubling

(up to ~ 60 sec.)

---

or : hopefully :



Compute $RTO^* = ??$ Back to original $f(\text{new } RTT_{est})$ etc.

More general:

$$RTT_{est,new} = \alpha * RTT_{est.old} + (1-\alpha) RTT_{meas.}$$

BSD Unix:

$$\alpha = \frac{7}{8}.$$

If we time more packets per RTT:
make α "somewhat" larger.
($(1-\alpha)$ "somewhat" smaller).

"Exponential Smoothing"

Time series $(X_t)_t$ (say stock market)

$$\hat{X}_t = (1-\alpha) X_t + \alpha \hat{X}_{t-1} = (1-\alpha) X_t + \alpha(1-\alpha) X_{t-1} + \alpha^2 \hat{X}_{t-2} = \cdots$$

($\hat{X}_{t-1}$: previous estimated mean)

$$= (1-\alpha) \sum_{k=0}^{\infty} \alpha^k X_{t-k}.$$

exponential smoothing determines "length of memory".
α close to zero: short memory.
α close to one: long mem.

Mathematically: $|\alpha| < 1$

Common Sense: $0 < \alpha < 1$

Karn's Algorithm

Simplistic:

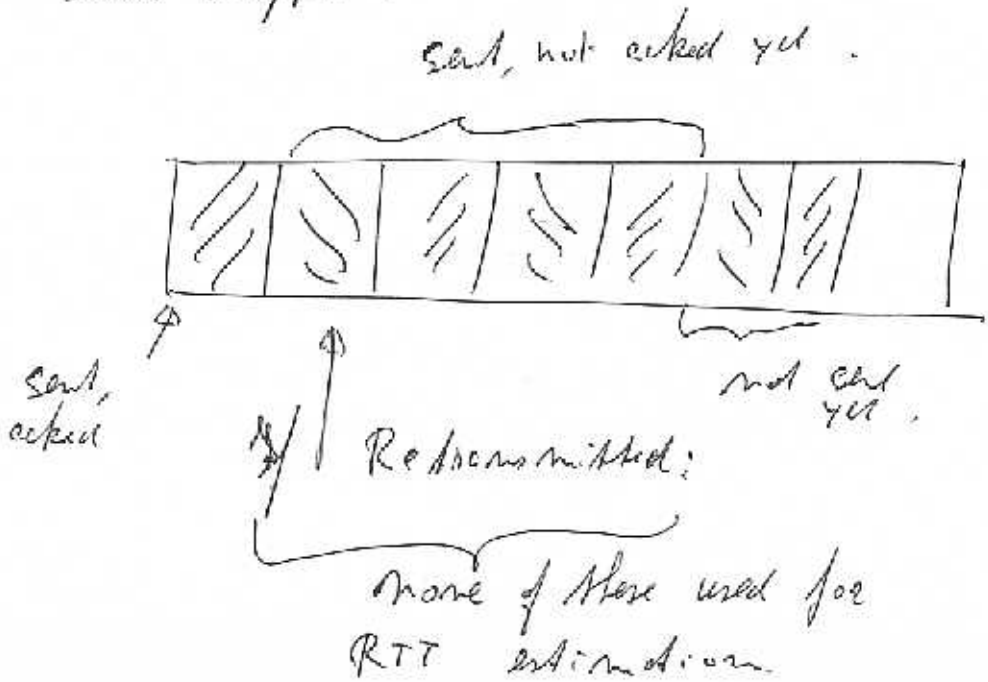
Stop estimating RTT during recovery.

(Do not used data sent twice for or more for RTT est updates)

Complication: protect against sudden increase in B actual RTT.

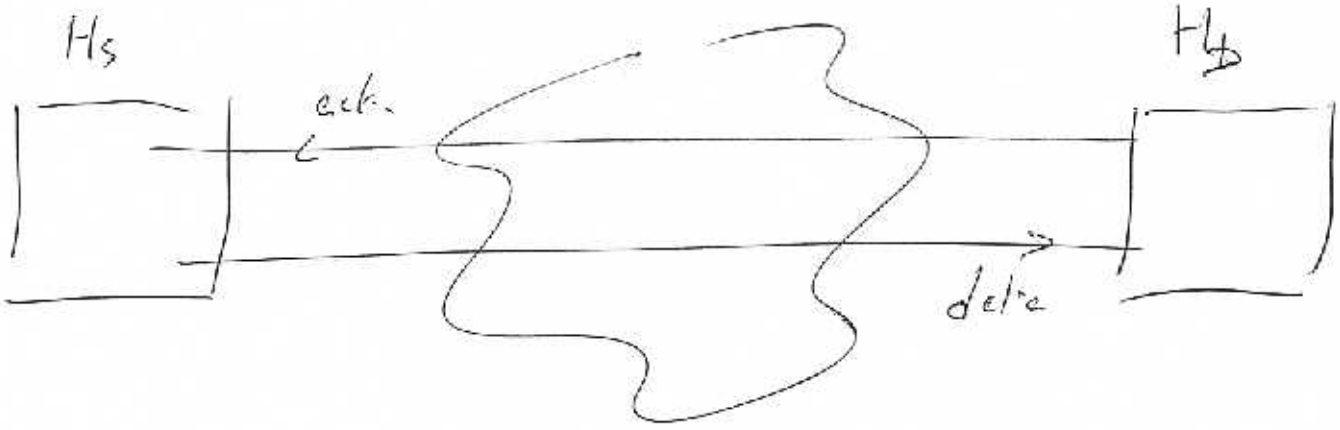
Copper is noisy - nearly.

Send Buffer:

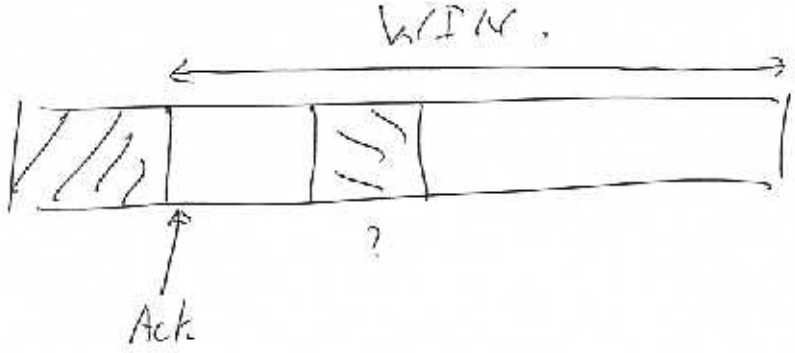


Plus Bells and Whistles!

Effective window.

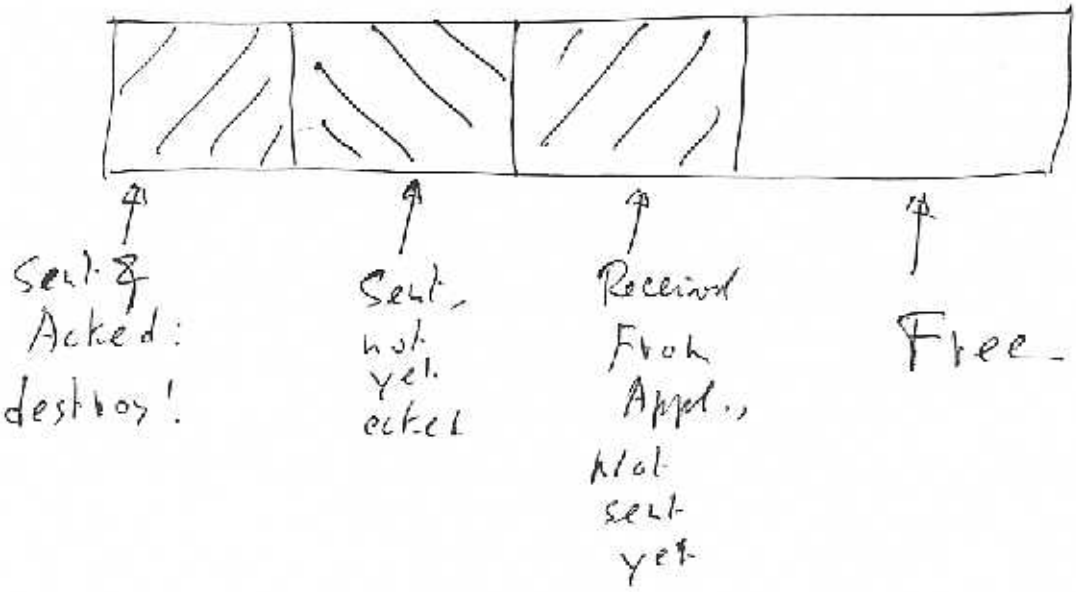


Dem : Receive Buffer



WIN = Advertised window.

Source Buffer (Send Buffer)



Amount of stuff sent, not yet acked : $\leq$ Send Buff.

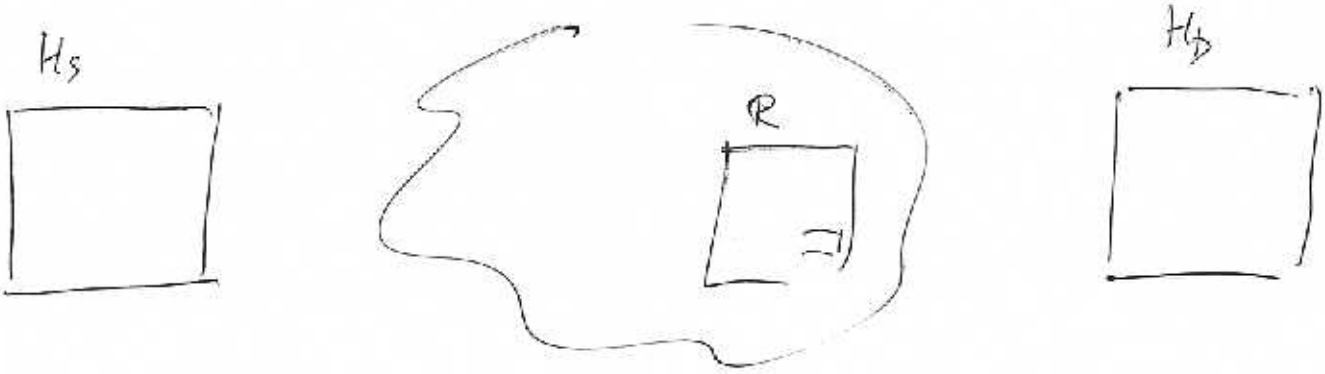
Combine:

Amount of stuff sent, not yet acked

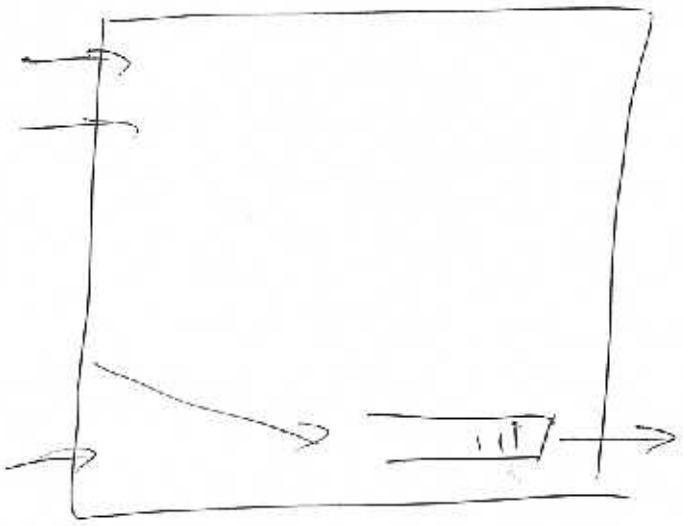
$\leq$ min (Advertised WIN, Send Buff).

($\leq$ min (~~Ad~~ Send Buff, Rcve Buff, 2^k * bytes).

Algo : Congestion Window.



Router :



Too many packets to some output buffer in short time : Congestion !

Buffer overflow !

Flow flows must decrease !

$$cwnd(n) = \begin{cases} cwnd(n) + \dfrac{1}{cwnd(n)} & \text{if Ok.} \\ \text{must be expressed in MSS.} & \\ \dfrac{cwnd(n)}{2} & \text{if not Ok.} \end{cases}$$

BSD Unix: cwnd is expressed in Bytes, not MSS.

$$\frac{cwnd}{MSS} = \frac{cwnd}{MSS} + \frac{1}{\left(\frac{cwnd}{MSS}\right)} \quad * \cancel{MSS}$$

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

Funny Story: See Stevens II p 999

(Since changed!)

Effective window size:

receiveWind $2^h * 65535$

min (cwnd , send_window, advertised window).

(make sure to get units right).

If the WIN_{eff} is "constant"

and the source has plenty of stuff to

send:

$$thput = \frac{WIN_{eff}}{RTT}.$$

---

1 TByte ea in 4 hours.

10^{12} Bytes in $4 * 60 * 60$ sec:

69,444,444.44 Bytes/sec. = 555,555,555.55 b.b/sec

in 50 msec = $\frac{1}{20}$ sec.

3,472,222.22 bytes in 50 msec.

Receive Buffer and Send Buffer must be at

at least 3.5 MBytes!

What kind of window scaling do we need?

$$\frac{3.5 \text{ MBytes}}{65535 \text{ Bytes}} \sim 52.98$$

$$32 < 53 < 64$$

We need $\boxed{k = 6}$.

Choose $k = 6$.

$$2^6 = 64$$

$64 * 65535 = 4,194,240$ Bytes.

Choose $k = 6$,

Receive Buffer,

Send Buffer

Both of them 4.2 MBytes.

Also modify TCP behavior!

Slow here 04/02/04.

Silly window syndrome.

(e.g. Telnet).

" Many Small packets.

Can be due to slow source appl (Telnet)

Slow dest. Appl (~~od~~ adv. window)

Solution:

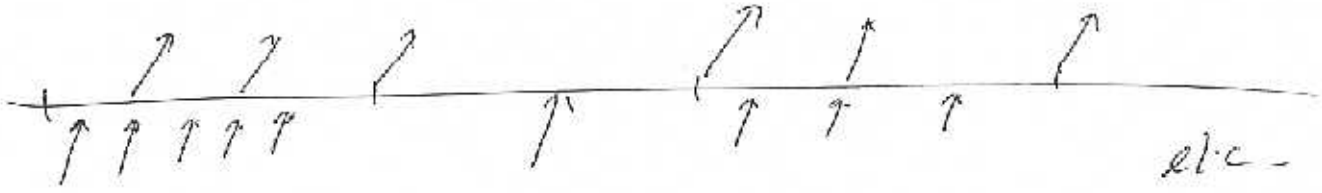
Two seemingly unrelated solutions.

(1) At Source Side:
Nagle's algorithm.

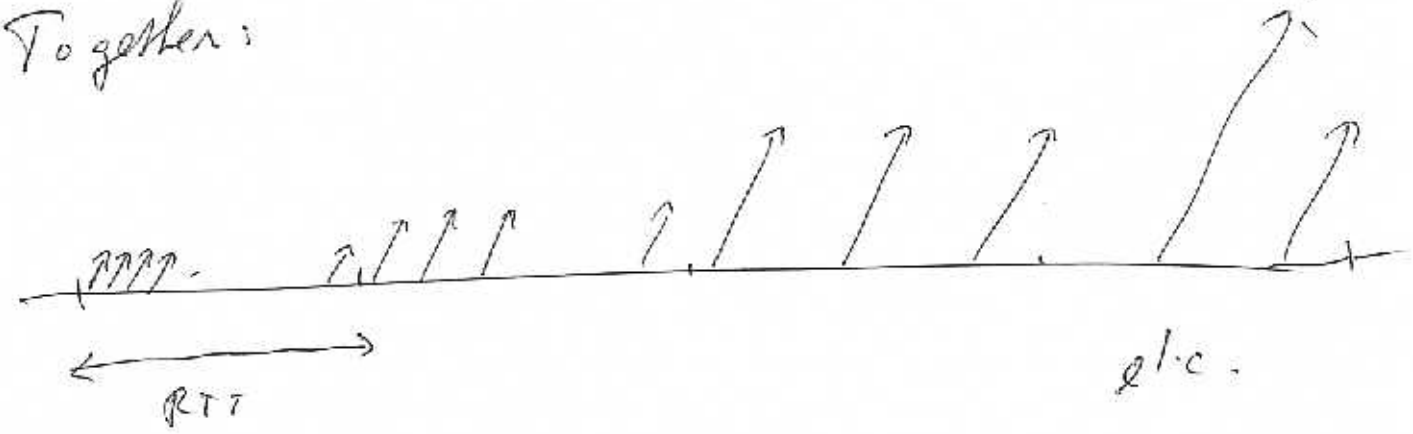
Start: first send.
Later: only if (1) whole MSS,
or (2) Good Ack.

---

(2) Dest Side. Delayed Acknowledgments.



Together:



Soon:

either 1 data packet per clock tick on other side,

or whole MSS.

Unless push !

Routing.

Comer p 253 etc.

Routers exchange information about the "state of the network", so each can construct a forwarding table.

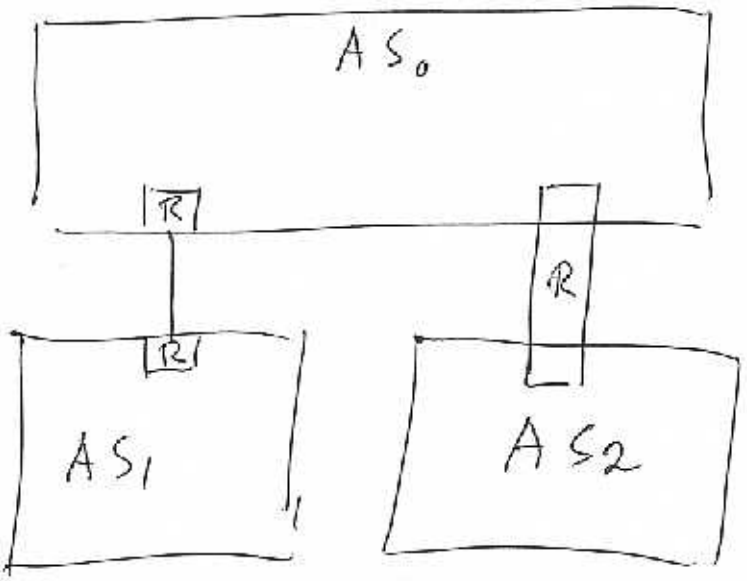
Question: do routers trust each other?

Autonomous Systems ASs

"Administratively Autonomous".

(Autonomous Domains
Administrative Domains).

Within AS: Routers "trust" each other.



Routing "between" ~~Es~~ ASs:

Exterior Gateway Protocol. EGP.

Routing "inside" ASs:

Interior Gateway Protocol. IGP.

Examples:

EGP: only one: BGP. Ch. 15
Border Gateway Protocol.
(IETF).

IGP:

RIP Routing Information Protocol.
V1 and V2. Ch 16

OSPF Open (Shortest Path First) Ch. 16

IS_IS Intermediate System –
Intermediate System.

Hello.

RIP: IETF

IS-IS: originated in ISO
(International Standards Org).

OSPF: IETF, grew out of IS-IS

BGP: IETF

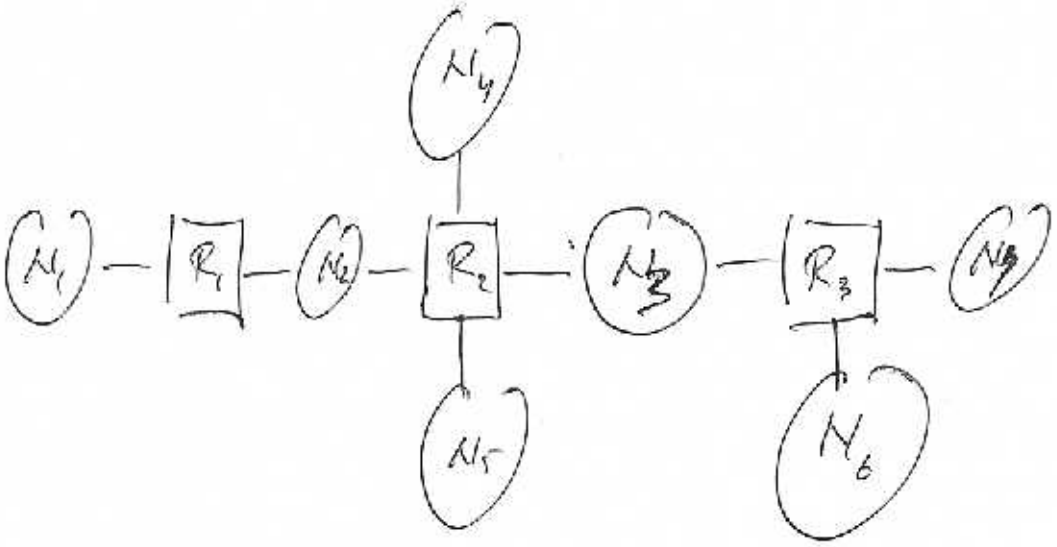
BGP allows "policy routing".
(e.g. traffic of customer A is not allowed on the network of long haul company Z).

Distance Vector Protocols

Basic idea: every router has a routing table and sends <u>advertisements</u> to its neighbors:

"I know how to reach network N, and this is how good the route is. If you send me stuff for N, I will forward it."

Sign here 04/06/2004



Suppose R_1, R_2, R_3 boot at the same time. (Unrealistic)!

R_1 R_2 R_3

R_1

net'n	dist	next
N_1	0	??
N_2	0	??

R_2

N_2	0	??
N_3	0	??
N_4	0	???
N_5	0	??

R_3

N_3	0	?1
N_6	0	
N_7	0	

RIP V1 advertises only network & distance (hopcount).
RIP V2 also advertises "next hop" and mask. (CIDR)

advertises:

N_1	0
N_2	0

N_2	0
N_3	0
N_4	0
N_5	0

Let's assume R_1 advertises to R_2.

R_2

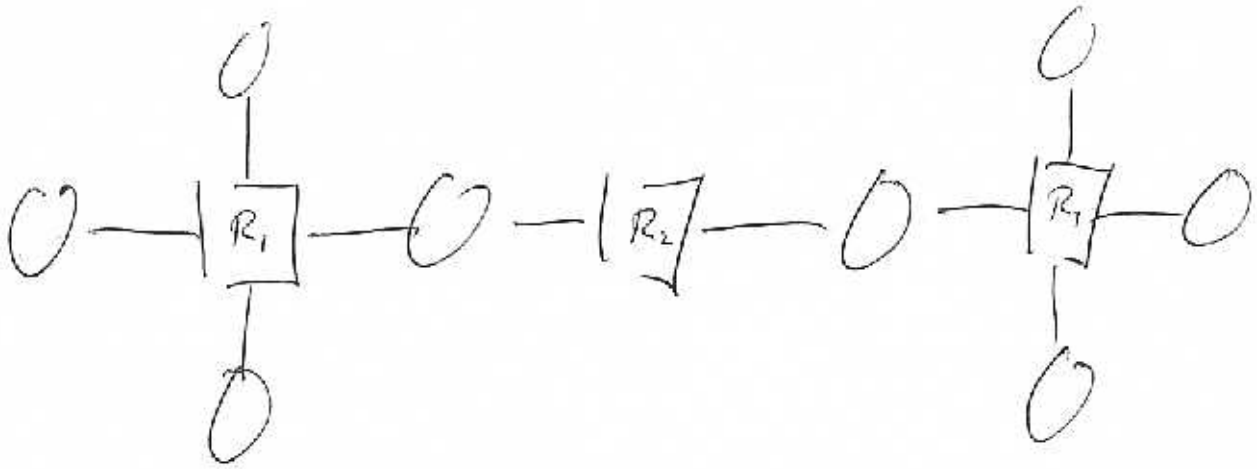
N_1	1	R_1
N_2	0	
N_3	0	
N_4	0	
N_5	0	

Let's assume next R_2 advertises to R_3

N_1	1
N_2	0
N_3	0
N_4	0
N_5	0

R_3:

N_1	2	R_2
N_2	1	R_2
N_3	0	
N_4	1	R_2
N_5	1	R_2
N_6	0	
N_7	0	



Router R_x currently has hop count $d_{x,N}$ to network N.

Gets advertisement from router R_y: $d_{y,N}$.

(1) If $d_{x,N} > d_{y,N} + 1$:
change route $d_{x,N} = d_{y,N} + 1$, next hop y.

(2) If $d_{x,N} < d_{y,N} + 1$ and next hop is y:
$d_{x,N} = d_{y,N} + 1$, next hop is y.

(3) Otherwise: no change.

Problem with RIP:

" slow convergence

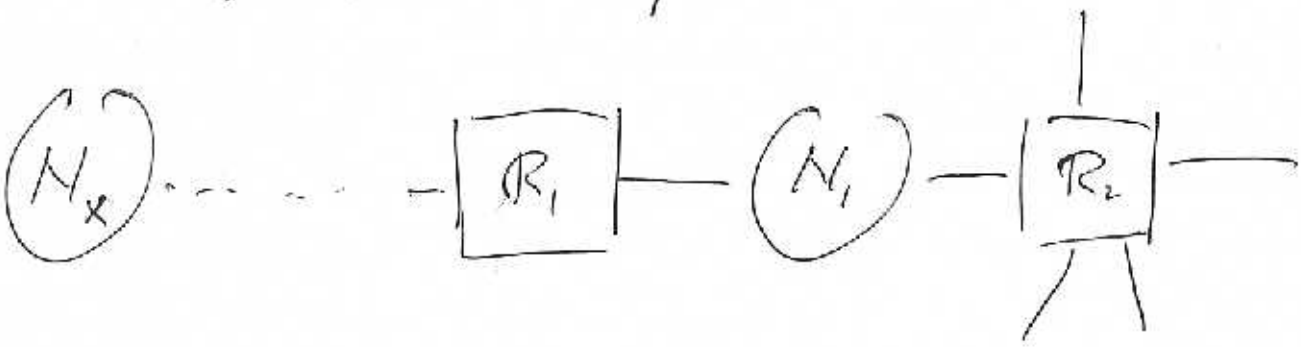
advertise once every 15 sec.

distance D: takes D * 15 sec

Problem with RIP:

"Walk to infinity."

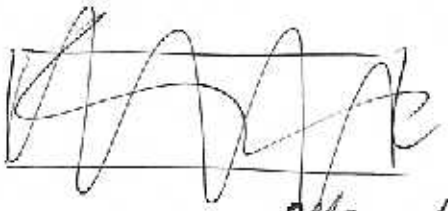
"Count to infinity".



d_{R_1, N_x} $\qquad$ $d_{R_2, N_x} = d_{R_1, N_x} + 1$.

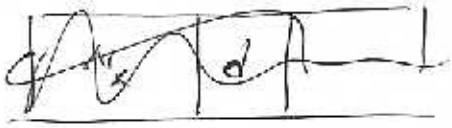
Now the connection between R_1 and N_x breaks.

And, unfortunately, R_2 advertises to R_1 right after.



R_1 gets from R_2

N_x	"infinity"	none



R_x	d_{R_2, N_x}

R_1

N_x	$d_{R_2,N_x}+1$	R_2

advertises to R_2 :

R_2 :

N_x	$d_{R_2,N_x}+2$	R_1

etc.

"walk to infinity".

"Solution": Make infinity small.

infinity = 16.

N	16	?

"I know about N, but you can not get there from here".

Solutions:

Split Horizon.

If in R_X:

N	d	R_Y

then R_X does not advertise network N to router R_Y

Poison Reverse. (Poisoned Reverse?).

If in R_X:

N	d	R_Y

then R_X advertises to R_Y:

N	16

(16 = inf)

Triggered updates:

Immediately send advertisements when receiving "Bad news".

Hold-down: ~~The opposite of~~ After "bad news", do not use "good news" for a while.

RIP uses UDP.

port. 520.

Is OK: neighbors only..

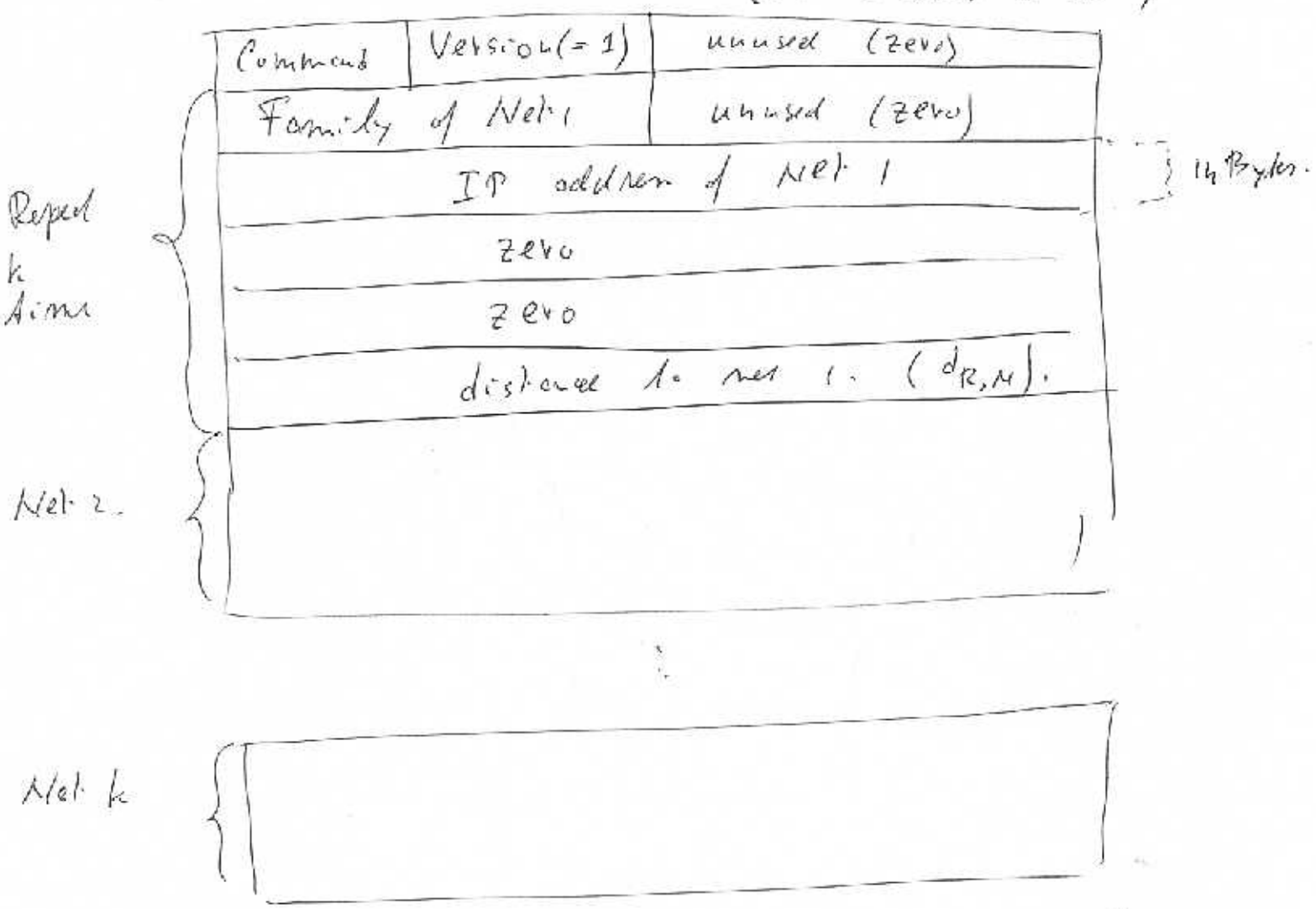
SAM Lee 04/13/2004

~~105~~ 186

RIP 1 format.

Comer p. 301

one every 30 sec.
(or one every 15 sec.?)



Commands: See Comer p 301 { 1 Request, 2 Response, ... }

Version: 1 for RIP 1. (!)

Family: 16 bits = 2 for TCP/IP

Address: 14 Bytes (!).

Route Tag

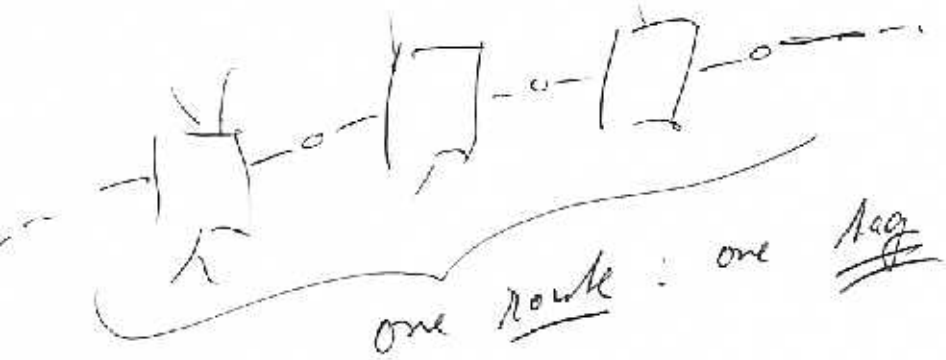
Used to identify "originator" of route -
E.g. "Autonomous System Number" from
BGP. (~~to~~ both are 16 bits).
Also: Used to identify "Route"

RIP: no message length (# routes).

RIP uses UDP port 520.

UDP message length!

UDP: Ok. neighbors only.



one route: one tag

Hello Protocol:

There are two of them!

(1) Hello protocol and seventies.
"whole protocol".

(2) Hell. Protocol, part of OSPF.
"know thy neighbor".

Hello Protocol.

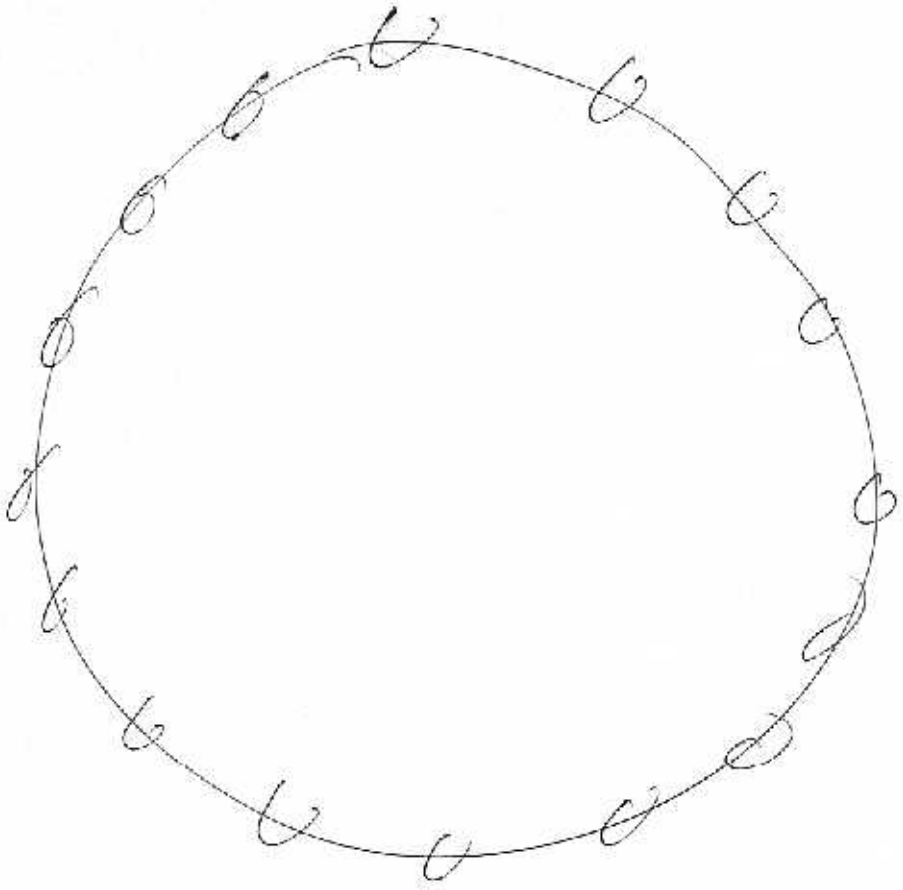
Based on Delay Estimation.

Instead of ~~a hop~~ hop count :

estimated delay.

Find the "minimal delay" route.

Caused Oscillation.



Oscillatory Behavior

Can be countermanded by changing the delay estimate "very slowly".

Still, a problem with "all or nothing" routing.

OSPF.

OSPF.

(1) This is a link state protocol.

Every router sends advertisements describing the local network.

Includes Type of Service routing.

Delay

~~Reliability~~ Throughput

Reliability.

Each router develops a map of the whole internetwork (whole area) and in that area finds the "best route" to all networks.

(2) These ports had better be consistent.

Cost of a link.

Try to find the route that minimizes the cost.

$$\text{minimize} \sum_{l \in Path} c_l$$

Just like in RIP:
a mathematical optimization.

How are the costs used?

To distribute load

Comer p 309

ToS Routing
Load Balancing: Equal Cost Multipath

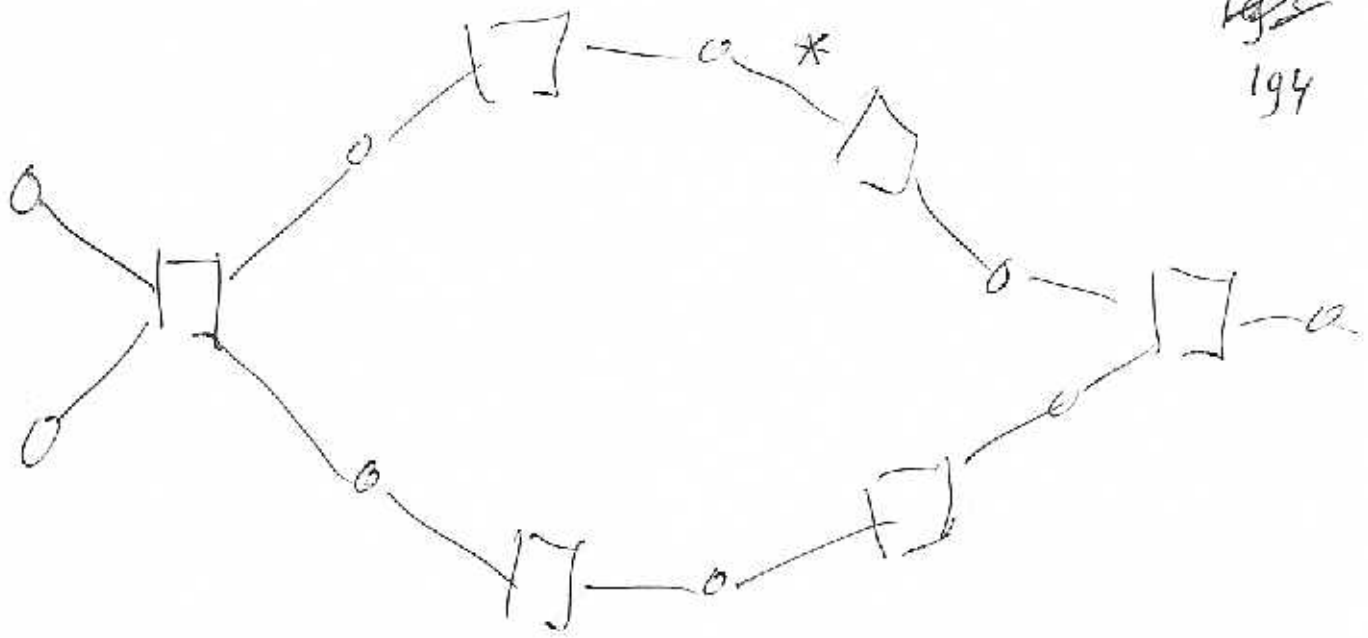
Areas
Authentication!
All kinds of routers.
→ Link State
Virtual network ← Transit Network
External link, External link

B

OSPF les an options
"Base" "Equal Cost Multi-path".

OSPF : RFC 2328

OSPF equal cost multi-path
RFC 2178. (?)



H. (Actually, many of these links probably are point to point)

* is "too busy".

increase its cost, until people start re-routing.

What happens?

"Bang".

Routers

people

costs are integer

OSPF: RFC 2328.

OSPF, Multipath 2178
Eqld Cos

(Check?)
(Wrong order).

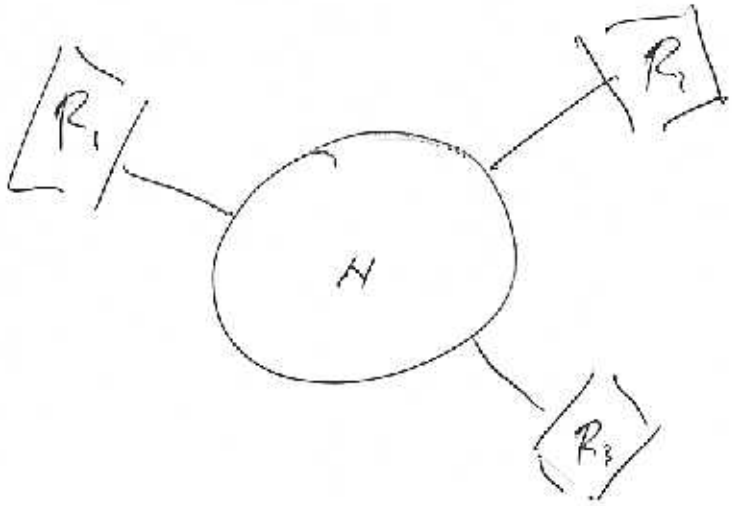
RFC numbers need checking!

OSPF.

Types of "links"

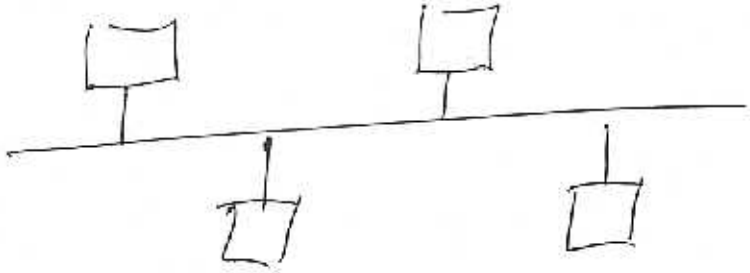
(1) Point to Point.

(2)

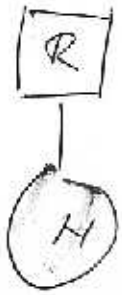


Transient link
Transit network.

(3) or:



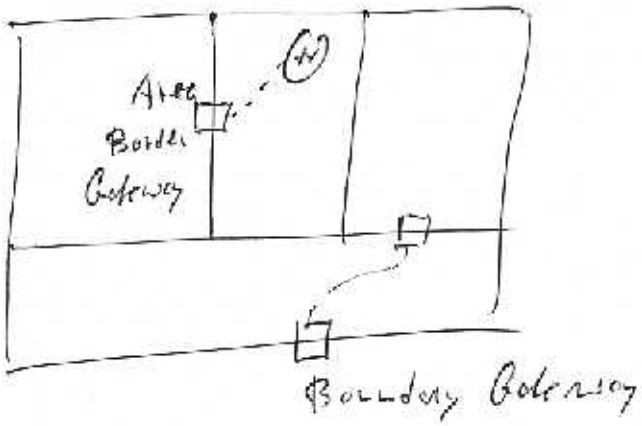
(3) Stub



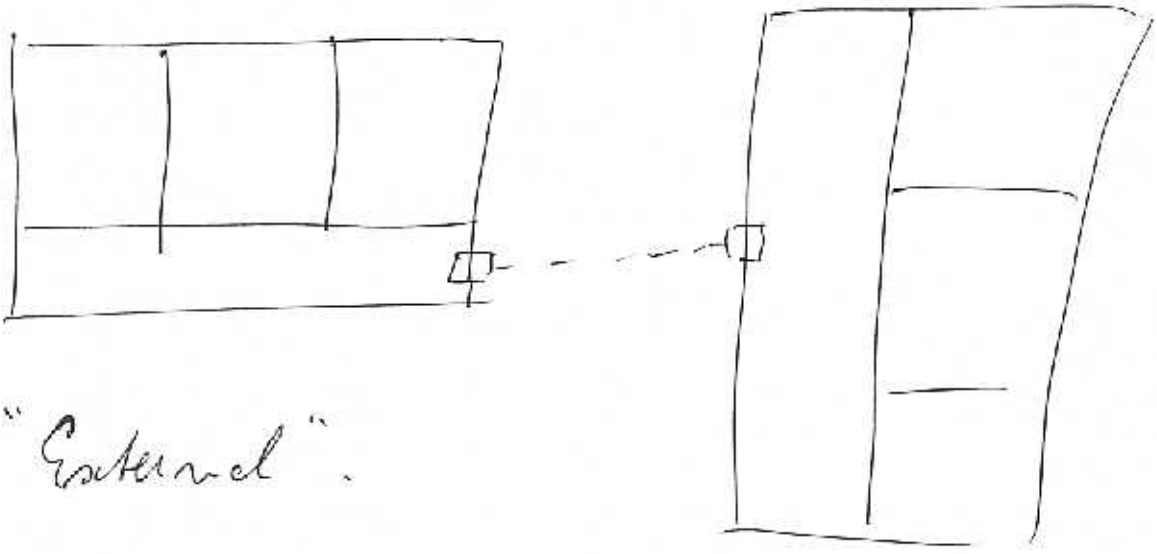
one router
on network.

PtP Transient Stub: are real.
(physical).

There are more that are "virtual".

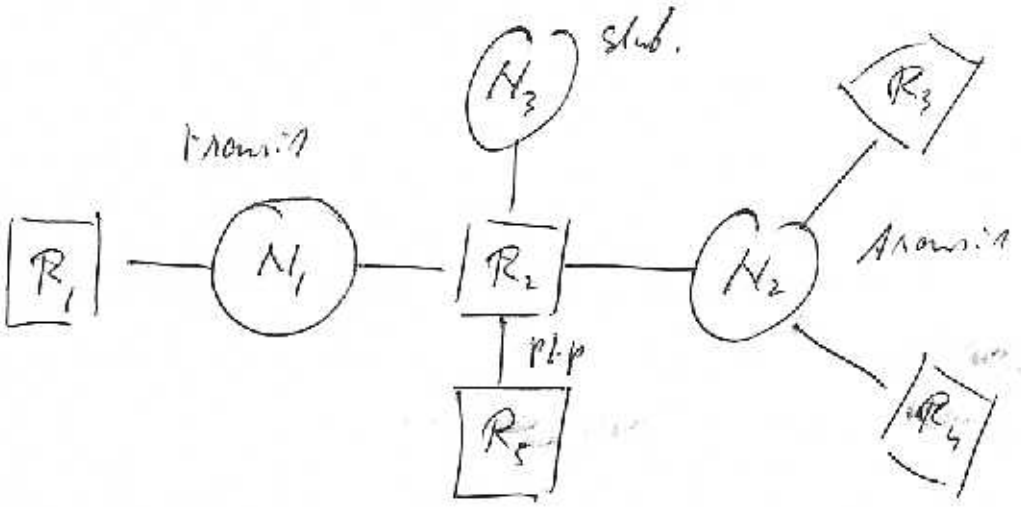


Summary Link



"External".

Such links are advertised
"kind of like a stub".



RFC 2328

From.

To	R_1	R_2	R_3	R_4	R_5	N_1	N_2	N_3
R_1						0		
R_2					3	0	0	!
R_3							0	
R_4							0	
R_5		2						
N_1	3	2						
N_2		3	4	11				
N_3		7						

Blank is "inf".
Not zero.

Each column is an advertisement.

R_1 & R_2 advertises

R_5	2
N_1	2
N_2	3
N_3	7

Who advertises for N_2 ?

One from R_2, R_3, R_4.

The "designated router" for N_2.

For example R_2. in that case:

R_2 advertises

R_5	2
N_1	2
N_2	3
N_3	7

for itself

and it advertises

R_2	0
R_3	0
R_4	0

for N_2.

~~Recursively~~:

Every Router gets picture of whole network.

(Hopefully: Same!)

Peculiarity:

Distance from Node to (next) router is always zero!

Distances asymmetrical.

Who advertises for N_3?

Nobody.

Routing with partial information