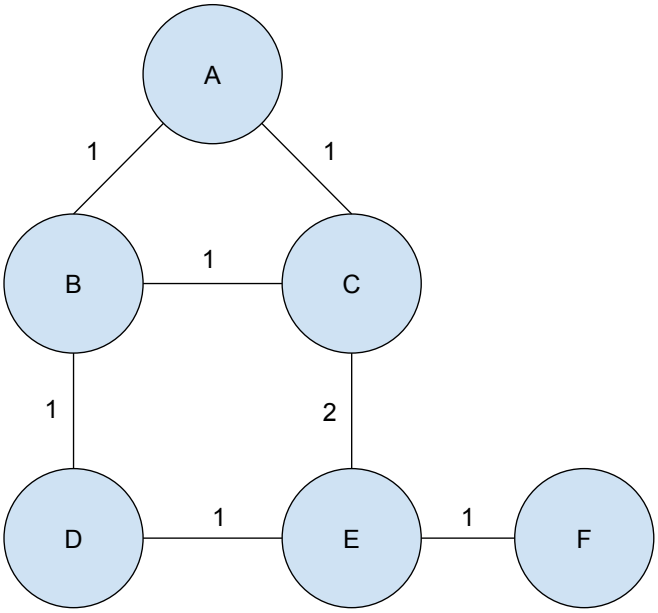
CS 168 Introduction to the Internet
Fall 2017 Scott Shenker

Discussion 5: Reliable Transport and IP

1 Review: Poisonous Routes

Consider the network topology below:



For this problem, assume that for initial convergence, full updates are sent, and afterwards partial updates are used to recover from failures.

(1) After routing converges, what does A's vector table look like?

	A	B	C	D	E	F
A	0	1	1	2	3	4
B	1	0	1	1	2	3
C	1	1	0	2	2	3

(2) Suppose the links BC, BD and DE go down. What will A's table look like after routing converges?

	A	B	C	D	E	F
A	0	1	1	∞	3	4
B	1	0	∞	∞	∞	∞
C	1	∞	0	∞	2	3

(3) Was poison reverse, route poisoning, or both used in the last step? Why?

Solution: This was route poisoning and poison reverse. None of the routers have a path to D anymore, so they poisoned the route. B's path to everywhere else goes through A, and C's route to B goes through A. Both used poison reverse to lie to A in order to avoid loops.

(4) Suppose the BD and DE links comes back up, routing converges, but then the CE link fails. What does A's table look like after routing converges again?

	A	B	C	D	E	F
A	0	1	1	2	3	4
B	1	0	∞	1	2	3
C	1	∞	0	∞	∞	∞

(5) Was poison reverse, route poisoning, or both used in the last step? Why?

Solution: This was poison reverse. Both B and C were now using paths through A, and so had to lie to it in order to avoid loops.

2 Reliable Transport

Bob thinks that using ACKs on every packet is very wasteful, and wants to design a transport protocol that is reliable but uses very few ACKs. He comes up with a scheme that he believes provides reliability but only sends at most $log(n)$ ACKs, where n is the number of packets. The protocol is as follows:

- Let P_i be the i^{th} packet. When the receiver sees *any* of the packets in the set

$$\{P_{2^i} | 1 \leq i \leq \lfloor log_2 n \rfloor\} \cup \{P_n\}$$

  it sends back an ACK for that packet.

- The sender will send packets $(P_{2^{i-1}}, P_{2^i}]$ in order (i starts at 0), and wait for the last packet to be ACKed. If the sender does not hear back after some time, it sends this window of packets again until the last packet is ACKed. Then, i is incremented and sends the next window of packets. One can think of this as a window that starts at packet 1, with size 1. Whenever the last packet in a window is ACKed, the window size is doubled and the sender moves to the next set of packets. This process repeats until P_n is ACKed.

(1) Is this transport protocol reliable? Why or why not?

Solution: No, it is not reliable. If only the packets that are designated to be ACKed make it through, the receiver will ACK them and the sender will happily accept that all the data was received, even though only $log(n)$ packets actually made it through.

(2) If you said the protocol was not reliable, what is a modification that you could make in order to fix that? Try and make the smallest change possible.

Solution: Simply change the ACKs from individual ACKs to cumulative ACKs that are only sent if the target packet and all previous packets have been received.

(3) Given that the protocol is reliable (or it wasn't and you apply your fix from the last part), does it actually save any bandwidth? Why or why not?

Solution: Not at all. In fact, this protocol uses much more bandwidth. As the window size increases (exponentially!), the odds that a packet is dropped is larger and larger, since there are more packets (more chances for drop) and the large number of packets sent at once will almost guarantee to congest the network if a large file is being sent. This means that entire windows will be resent constantly, wasting lots of bandwidth.

As an example, consider the transfer of a 1 GB file. The last 500 MB of the file will be sent as a single window. Assume about 1 KB of data is sent per packet. Then the last window is 500,000 packets wide! If even one of these packets are lost, then all 500,000 will be resent until every packet has been received.

3 IP Headers

IPv4 and IPv6 have very different headers. In particular, IPv6 dropped many of the fields that IPv4 headers had. Why did IPv6 drop the following fields from it's header?

Checksum:

Solution: Checking packet integrity is the end hosts' problem. Corrupt packets will either be mis-delivered and ignored or be retransmitted anyway (they won't be ACKed).

Header Length:

Solution: The IPv6 packet header is constant length, there is no need to include it. The "Next Header" field tells you about any additional information needed in order to consider options and such things.

Fragmentation:

Solution: Again, this is a problem of the end hosts. It is easy to discover the MTU of a path the packets will take, and fragmenting in the network is very expensive and poses security risks.

IPv6 also hosts much larger address fields (128 bits). Is that enough bits, or might some future IPvN need to expand these fields further?

Solution: 128 bits is plenty for the potential future of the human race. $2^{128} \approx 10^{38}$ addresses.

It has been (over)estimated that there will be 50 billion internet connected devices by the year 2020. Lets continue that trend to the year 2100 with an estimate of 50 trillion internet connected devices on Earth. Lets also suppose that humanity has conquered many solar systems, and controls 1000 planets, each with as many devices as Earth.

This means there would be about $1000 \cdot 50 \cdot 10^{12} = 5 \cdot 10^{16}$ internet connected devices on the inter(planetary)net.

$100\% \cdot \frac{5 \cdot 10^{16}}{10^{38}} \approx 5 \cdot 10^{-20}\%$ address space use. IPv6 gives us more than enough addresses.