CIS 553: Networked Systems February 26, 2020

Problem Set 1

Instructor: Prof. Vincent Liu *Department of CIS, University of Pennsylvania*

Turn in your solutions in on **March 2, 2020** by 10:00 p.m. via Canvas. Grading will be lenient, as long as you give an honest effort to answer every question. **Please work alone** and do not discuss answers with other students until after the deadline. Note that you should NOT assume this is what the exam will look like. For one thing, there are topics here that will not be on the exam, and topics on the exam that are not included here. Instead, treat this as an opportunity to review some of the material and debug any basic misunderstandings about the basic ideas covered so far.

1. True or False

   (a) Protocols govern the **syntax** and **semantics** of communication.

   T

   (b) Bit stuffing is a valid way to frame messages.

   T

   (c) Circuit switching typically involves three distinct phases of communication.

   T

   (d) Security was one of the original design goals of the Internet.

   F

   (e) The use of carrier sense limits the maximum size of the network and minimum size of frames.

   F

   (f) Spanning trees are used to decide which ports to disable in an Ethernet network.

   T

   (g) Iterative lookups are generally disabled in DNS.

   F

   (h) MAC addresses are guaranteed to be globally unique.

   T

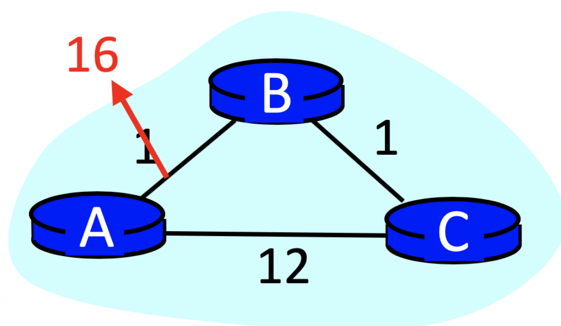
2. What are the layers in the Internet protocol stack? What are the principal responsibilities of each of these layers?

   Briefly:
   Application – The user–level application protocols.
   Transport – Break up packets into process–to–process channels.
   Network – Transmission of packets across connected Layer–2 networks.
   Data link / Physical – Transmission of \textbf{frames} through a physical medium.

3. Provide a concrete example of 'soft state' in the Internet and briefly describe an instance of its use. What is the benefit of using soft state in your example?

   Many options. ARP, learning switch forwarding table, DNS cache, spanning tree, etc.
   Soft state allows for eventually recovery of network state that is naturally regenerated on failure or network change.

4. The following diagram represents three routers using distance vector without any defense against the 'count-to-infinity problem' (e.g., split horizon, poison reverse, small infinity etc). Links are all 10 Mbps. Assume that the network has already been running for some time and that routing has converged. Then, at time 0, the A-B link goes from cost 1 to cost 16, and both A and B detect the change immediately.



   (a) What are the routing tables before the cost change?

A:

dest	next	cost
B	B	1
C	B	2

B:

dest	next	cost
A	A	1
C	C	1

C:

dest	next	cost
A	B	2
B	B	1

   (b) How many rounds of messages must occur before the network converges?

   11 (+1 also okay)

   (c) Assuming that processing and queuing delay are zero, and that routing update packets are all 1000 bits, at what time do each of the routers reach their final routing table state? (three numbers)

   Assume propagation delay is 1ms per link (not originally specified).
   1000 bits / 10Mbps + 1ms = 1.1ms
   11∗1.1ms = 12.1ms

5. This question deals with Classful IPv4 addressing (pre-1994) and CIDR (post-1994).

   (a) What technique is necessary for routing with CIDR addresses, but is NOT necessary for Classful IPv4 addressing?

   LPM

(b) In class, we mentioned that Classful addressing can be implemented in P4, and sketched a solution. Write an IPv4 **header** definition and a set of **tables** to parse class A, B, and C addresses. You can use as many tables as you need. Each should include the relevant match **keys**, but can just list `aiForward` as the only possible action. You can assume whatever you need from the other portions of the switch (control plane, parser, Ethernet processing, etc.).

As a hint, here's the beginning of the IPv4 header:

```
header ipv4_t {
    bit<4>    version;
    bit<4>    ihl;
    bit<8>    diffserv;
    bit<16>   totalLen;
    bit<16>   identification;
    bit<3>    flags;
    bit<13>   fragOffset;
    bit<8>    ttl;
    bit<8>    protocol;
    bit<16>   hdrChecksum;
    ...
}
```

```
header ipv4_t {
...
bit<32> srcAddr;
bit<1> classA;
bit<1> classB;
bit<1> classC;
bit<5> restOfA;
bit<8> restOfB;
bit<8> restOfC;
}

table tiClassA {
        keys = {
                // taken together these are the /8
                classA : exact; // 0
                classB : exact;
                classC : exact;
                restOfA: exact;
        }
        actions = {aiForward;}
}

table tiClassB {
        keys = {
                // taken together these are the /16
                classA : exact; // 0
                classB : exact; // 1
```

```
                classC : exact;
                restOfA: exact;
                restOfB: exact;
        }
        actions = {aiForward;}
}

table tiClassA {
        keys = {
                // taken together these are the /24
                classA : exact; // 0
                classB : exact; // 0
                classC : exact; // 1
                restOfA: exact;
                restOfB: exact;
                restOfC: exact;
        }
        actions = {aiForward;}
}
```

6. Is the count-to-infinity problem an issue in BGP? Why or why not?

   Not an issue because the path vector will, through its loop prevention, prevent a node from ever trying to use a stale route that passes through itself.

7. Describe why an application developer might choose to run an application over UDP rather than TCP.

   Many possible answers:
   1. UDP does not require state. It can be used to implement truly stateless services (DNS, for instance, mostly uses UDP in practice).
   2. Lower startup latency. TCP requires a handshake that delays the first packet we send.
   3. We don't want TCP's features. TCP will wait for lost packets to be retransmitted. If we care about latency more than reliability, we might want to just forget about lost packets.