CMSC 417 Midterm #1 (Fall 2001) - Solution

1.) (10 points) What does the term protocol-layering mean? Why is it a good idea?

Layering is dividing functionality into well-defined interfaces at functional boundaries. It allows different implementations of each layer to be swapped out without requiring changes to other layers. Layering is good because it allow graceful evolution of different layers. It also allows multiple (incompatible layers) to co-exist (i.e. both Ethernet and Token Ring running TCP/IP at the higher levels) and communicate.

2.) (15 points) Explain what is meant by the count-to-infinity problem in distance vector routing. How does BGP, which uses distance-vector, avoid this problem?

Slow to hear about failed links since information propagated is only the number of hops. A failed link may still be in the routing table due to cycles in the possible routes (i.e. A->B->A->C).

BGP avoids this problem by maintaing an explicit path, not just a hop count. This allows routes with cycles to be eliminated easily.

3.) (15 Points) An OC-3 (155 Mbps) link connects Washington DC to Los Angeles (10 mili-seconds apart at the speed of light). If you wanted one connection to be able to fully use the link to send packets from DC to LA, what packet size (in bytes), is needed if a sliding window protocol with a window size of 10 packets is used?

The Round-trip time is 20msec = 1/50 sec.
In an RTT, the link can send 1/50 * 155,000,000 ≅ 3Mega-bits

Packets must therefore be about 3Mb/10 ≅ 300Kbits / 8 bits/byte ≅ 40KB

4.) (20 points) If I wish to send the message 0110 1111_{LSB} using a Hamming code (even parity in check bits), what would the transmitted message be? Please show your work in calculating the message.

Bits	1	2	3	4	5	6	7	8	9	10	11	12
Data			1		1	1	1		0	1	1	0
PB_1	0		1		1		1		0		1	
PB_2		1	1			1	1			1	1	
PB_4				1	1	1	1					0
PB_8								0	0	1	1	0
MSG	0	1	1	1	1	1	1	0	0	1	1	0

Some people used odd parity, this resulted in a few points being deducted.

Some people used a different MSB, this was given full credit if done correctly

5.) (25 points) The following C program has four threads: reader1, reader2, writer1 and writer2. that share the integer variable, dataCount. Threads write1 and write2 execute writeFunc() and the reader threads execute readFunc(). readFunc() should repeatedly do the following: block until dataCount is greater than 0, then execute readIt, and decrement dataCount by 1. writeFunc() should repeatedly do the following: call writeit, and then increment dataCount by 1. To ensure correct operation, any number of readIt calls may be active at once, but when a call to writeit is running, no calls to readit or other calls to writeit may be active. Your solution should not allow starvation of readers/writers, and should let new readers in while readIt is active. Complete the code bellow to achieve these requirements.

```
int dataCount = 0;
      pthread_muext mutex;
      pthread_cond_t cond;




readFunc(void *arg) {
 while (1) {
      pthread_lock(&mutex);
      while (dataCount <= 0 || wCount >= 1)
            pthread_cond_wait(&cond, &mutex);
      dataCount--;
      rCount++
      pthread_unlock(&mutex);
      readIt();
      pthread_lock(&mutex);
      rCount--;
      pthread_unlock(&mutex);
      pthread_signal(&cond);




 }
}
writeFunc(void *arg) {
 while (1) {

      pthread_lock(&mutex);
      while (rCount >= 1 || wCount >= 1)
            pthread_cond_wait(&cond, &mutex);
      wCount++
      pthread_unlock(&mutex);
      writeIt();
      pthread_lock(&mutex);
      wCount--;
      pthread_unlock(&mutex);
      pthread_signal(&cond);




  }
}

main() {
 pthread_t read1, read2, write1, write2;
 pthread_init(&read1, NULL);  pthread_init(&read2, NULL);
 pthread_init(&write1, NULL); pthread_init(&write2, NULL);


pthread_cond_init(&cond, NULL);
pthread_mutex_init(&mutex, NULL);

 pthread_create(&read1, NULL, eat,        );
 pthread_create(&read2, NULL, eat,        );
 pthread_create(&write1, NULL, grow,      );
 pthread_create(&write2, NULL, grow,      );
}
```

6.) (15 points)

a) (10 points) Explain what it means to tunnel one protocol through another? How is this different than layering?

Encapsulating one packet type into another to allow moving data through a network that can't handle the tunneled traffic (i.e. Ethernet tunneled through IP). Layering adds new functionality as you work up the stack, but tunneling could be involve sending a lower level protocol through a higher level one.

b) (5 points) Why might you wish to tunnel IP through IP?

Security (ipSEC)

Mobility (tunnel packets for the mobile host through the Internet to the foreign agent).