PartB - Wireshark

Sravya Beesabathuni

March 5, 2018

1 High level view of the analysis_pcap_tcp for congestion control:

- Files : analysis_pcap_tcp.java and TCPDataPacket.java
- Separate function call congestionControl() is written for the implementation of congestion control. Uncomment this in the code for it to run.
- For each flow, we set the initial cwind value to icwind = 1460 (1 MSS) and the ssthreshold to be largest window size which I have calculated by looping through all the packets, to get the max:*ssthreshold*
- We perform congestion control over all the sent packets to get the following information : how many retransmissions due to timeout and how many retransimissions due to triple duplicate ack.
- We take the timeout threshold to be 2*RTT which we have calculated earlier.
- We loop through all the sent packets and compare with all similar packets (similar packets have same sequence number) sent before the current packet, if the time stamp difference between them is greater than 2*RTT , then it is considered to be retrasnmission due to timeout and if for the current sent packet, we have three acks(information retrieved from receiveMap which stores the number of packets received for each source port), it is considered to be retrasnmission due to triple ack. Used *TCP Tahoe* for congestion control.
  In both the cases, we perform the following:

```
currentssthreshold = cwind/2;
cwind = icwind;
```

  Else we check for congestion avoidance :

```
if (cwind >= currentssthreshold) // congestion avoidance
cwind = cwind + 1460/ cwind;
```

```
else {
cwind += 1460; // cwind = cwind + 1MSS
}
cWindList.add(cwind);
```

We add all the congestion windows to cWindList, to print the first 10 windows in the output.

- timeoutRetransimission and tripleAckRetransimission variables give us the count due to each retransmission in each flow

2 Congestion Control code :

```
public static void congestionControl() {
    System.out.println("======PART B==============");
    List keys = new ArrayList(sourceMap.keySet());
    for (int i=0;i<sourceMap.size();i++) {
      long currentssthreshold = ssthreshold; // largest window size
      long icwind = 1460;
      long cwind = icwind;
      List<Long> cWindList = new ArrayList<Long>();
      List<TCPDataPacket> currentSentPackets = sourceMap.get((Integer)keys.get(i));
      Double timeOutLimit = 2 * rttList.get(i);
      int timeoutRetransimission = 0;
      int tripleAckRetransimission = 0;
      for (int j=0;j<currentSentPackets.size();j++) {
        TCPDataPacket currPacket = currentSentPackets.get(j);
        for (int k=0; k<j;k++) {
          TCPDataPacket prevPacket = currentSentPackets.get(k);
          if (currPacket.getSeqNumber() == prevPacket.getSeqNumber()) {
            if (currPacket.getTimeStamp() - prevPacket.getTimeStamp() > timeOutLimit) {
              timeoutRetransimission++;
              currentssthreshold = cwind/2;
              cwind = icwind;
            } else {//check for triple ack
                if (receiveSeqMap.containsKey(currPacket.getSeqNumber())
                    && receiveSeqMap.get(currPacket.getSeqNumber()).size() >=3) {
                  tripleAckRetransimission++;
                  currentssthreshold = cwind/2;
                  cwind = icwind;
                }
            }
          } else { // TCP Tahoe
            if (cwind >= currentssthreshold) // congestion avoidance
              cwind = cwind + 1460/ cwind;
            else {
```

```
            cwind += 1460; // cwind = cwind + 1MSS
          }
          cWindList.add(cwind);
        }

      }
    }
    System.out.println("=======TCP Flow for "+(Integer)keys.get(i)+"====");
    System.out.println("timeoutRetransimission ===="+timeoutRetransimission);
    System.out.println("tripleAckRetransimission ===="+tripleAckRetransimission);
    int count = 0;
    for (int m=0;m<cWindList.size();m++) {
      if (count <10) {
        System.out.println("Cwind size "+(m+1)+": "+cWindList.get(m));
        count++;
      }

    }
  }
}
```

3 Answers : Window sizes, Timeout Retransimissions and Triple Ack Retransimissions

```
  libpcap version 1.7.4
======PART B===============
=======TCP Flow for 43502====
timeoutRetransimission ====0
tripleAckRetransimission ====0
Cwind size 1: 2920
Cwind size 2: 4380
Cwind size 3: 5840
Cwind size 4: 7300
Cwind size 5: 8760
Cwind size 6: 10220
Cwind size 7: 11680
Cwind size 8: 13140
Cwind size 9: 14600
Cwind size 10: 16060
=======TCP Flow for 43500====
timeoutRetransimission ====90
tripleAckRetransimission ====4
Cwind size 1: 2920
```

```
Cwind size 2: 4380
Cwind size 3: 5840
Cwind size 4: 7300
Cwind size 5: 8760
Cwind size 6: 10220
Cwind size 7: 11680
Cwind size 8: 13140
Cwind size 9: 14600
Cwind size 10: 16060
=======TCP Flow for 43498====
timeoutRetransimission ====1
tripleAckRetransimission ====2
Cwind size 1: 2920
Cwind size 2: 4380
Cwind size 3: 5840
Cwind size 4: 7300
Cwind size 5: 8760
Cwind size 6: 10220
Cwind size 7: 11680
Cwind size 8: 13140
Cwind size 9: 14600
Cwind size 10: 16060
```