EE321 Lab 3

Network Performance Evaluation

## Aim:

The goal of this assignment is to investigate different parameters of network performance using a java media server and client. Using these programs, the goal is to get a measure for throughput, loss rate, packet delay and jitter to see some real-world results for these parameters.

## Procedure:

The first step was to set up the server, client, and packet code. I did this using eclipse by making a new java project and copying and pasting the code supplied. I exported the server code as a jar file to be ran from the command prompt as needed. I then set to measure the parameters withing the client.

I decided to measure throughput by simply incrementing a variable every time a packet is received. Then once a second calculate the average throughput using exponential weighted moving average calculation.

throughput = (1-alpha)\*throughput + alpha\*(successful);

To calculate the loss rate, I recorded the lowest sequence received by the client compared it to the latest sequence number received. Then I saw the difference between the number of packets that should have been received vs how many were received.

loss = (currentSequenceNo-prevSequenceNo) - successful;

lossrate = (1-alpha)\*lossrate + alpha\*(loss);

To get the packet delay I compared the systems current time at receiving the packet to the timestamp within the packet, I then again used an exponential weighted average to calculate the average packet delay.

packetdelay = (1-alpha)\*packetdelay + alpha\*(currentTime - pk.time);

To calculate the jitter, I got the absolute value of the difference between the previous packet delay time to the current one and then again used an exponential weighted average to get a value for the average jitter however this time I used a separate exponential value in order to allow the latest value to have a bigger impact as jitter may change a lot.

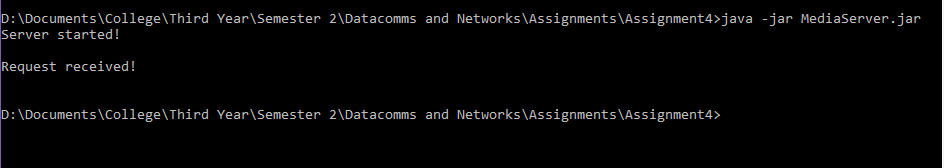
jitter = Math.*abs*(packetdelay-currentTime - pk.time); //initial jitter

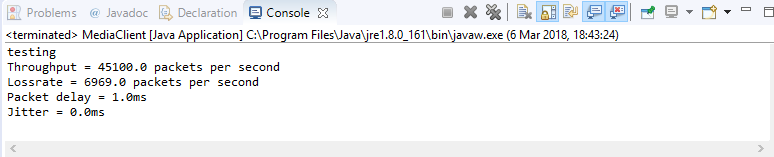
jitter = (1-beta)\*jitter + beta\*(Math.*abs*(packetdelay-currentTime - pk.time));

With all of these properties calculated I can print them out to the client user.

## Results:

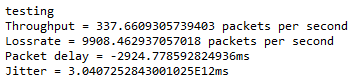
When testing for this assignment I ran into a few issues. The first issue I found that was testing on my own machine the time to transfer all the packets would take less than 2 seconds and therefore would only test the properties once and not give a very accurate estimate.

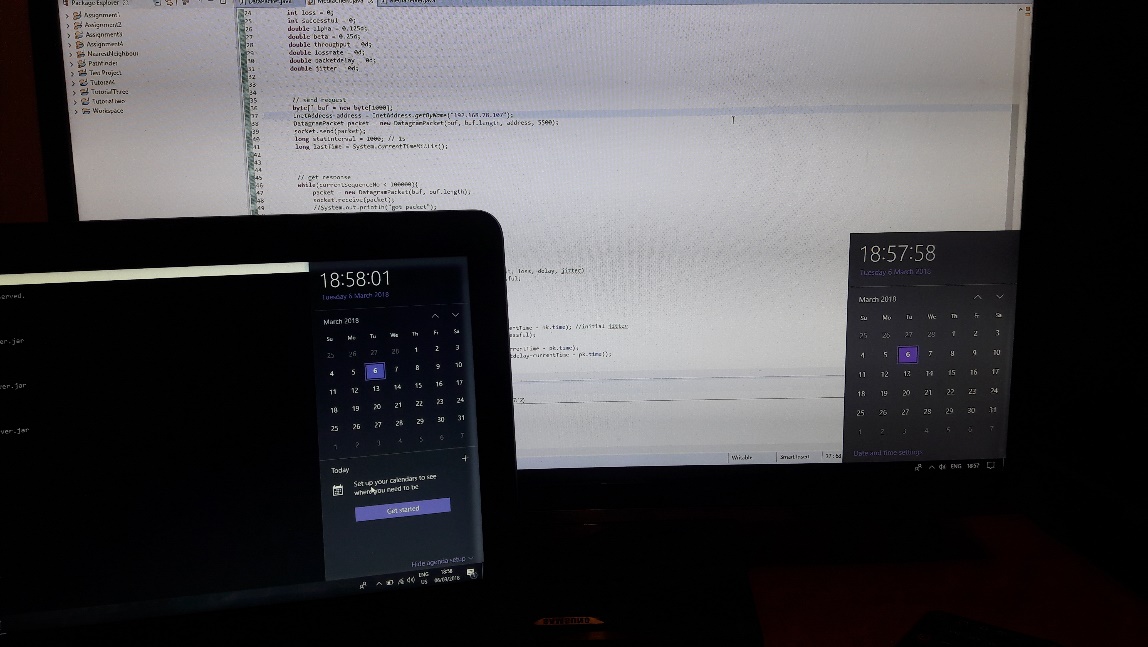




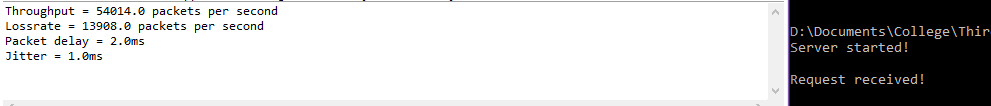
We can see here that only one test was carried out, also my jitter is equal to 0 as it has no previous packet delay data to compare to.

So, with this in mind I decided to test this using two separate systems connected to the same network and using local ip to connect. However, I then got a negative packet delay and noticed the two systems had slightly different system times so testing this way would not be possible.





I also noticed the loss rate was very high when sending data to the separate machine. So I went back to testing on a single machine. To get the jitter I therefore had to record the previouspacketdelay to test for jitter on the first test. These were my final results.



## Source code:

**import** java.io.\*;

**import** java.net.\*;

**import** java.util.\*;

**public** **class** MediaClient {

**public** **static** DataPacket convert(**byte**[] buf) **throws** Exception{

ByteArrayInputStream byteStream = **new** ByteArrayInputStream(buf);

ObjectInputStream is = **new** ObjectInputStream(**new**

BufferedInputStream(byteStream));

DataPacket pk = (DataPacket)is.readObject();

is.close();

**return** pk;

}

**public** **static** **void** main(String[] args) **throws** Exception {

DatagramSocket socket = **new** DatagramSocket();

**int** currentSequenceNo = 0;

**int** prevSequenceNo = 0;

**int** tests = 0;

**int** loss = 0;

**int** successful = 0;

**double** alpha = 0.125d;

**double** beta = 0.25d;

**double** throughput = 0d;

**double** lossrate = 0d;

**double** packetdelay = 0d;

**double** prevpacketdelay = 0d;

**double** jitter = 0d;

// send request

**byte**[] buf = **new** **byte**[1000];

InetAddress address = InetAddress.*getByName*("localhost");

DatagramPacket packet = **new** DatagramPacket(buf, buf.length, address, 5500);

socket.send(packet);

**long** statInterval = 1000; // 1s

**long** lastTime = System.*currentTimeMillis*();

// get response

**while**(currentSequenceNo < 100000){

packet = **new** DatagramPacket(buf, buf.length);

socket.receive(packet);

//System.out.println("got packet");

successful++; //i added

DataPacket pk = *convert*(packet.getData());

**long** currentTime = System.*currentTimeMillis*();

**if**(pk.seq < prevSequenceNo) prevSequenceNo=pk.seq;

currentSequenceNo = pk.seq;

**if**(currentTime > lastTime + statInterval)

{

//Compute and display network parameters (throughput, loss, delay, jitter)

loss = (currentSequenceNo-prevSequenceNo) - successful;

**if**(tests == 0) {

throughput = successful;

lossrate = loss;

packetdelay = (currentTime - pk.time);

jitter = Math.*abs*(packetdelay-prevpacketdelay);

}

**else** {

throughput = (1-alpha)\*throughput + alpha\*(successful);

lossrate = (1-alpha)\*lossrate + alpha\*(loss);

packetdelay = (1-alpha)\*packetdelay + alpha\*(currentTime - pk.time);

jitter = (1-beta)\*jitter + beta\*(Math.*abs*(packetdelay-prevpacketdelay));

}

lastTime = currentTime;

prevSequenceNo = currentSequenceNo;

System.***out***.println("Throughput = "+throughput+" packets per second");

System.***out***.println("Lossrate = "+lossrate+" packets per second");

System.***out***.println("Packet delay = "+packetdelay+"ms");

System.***out***.println("Jitter = "+jitter+"ms");

successful = 0;

tests++;

}

**else** prevpacketdelay = (currentTime-pk.time);

}

socket.close();

}

}