# Wilmington TCP Tuning

The process for tuning the parameters was similar to the document with some small improvement. Instead of loading each PCAP file and displaying the TCP graph we used an application TCPtrace (<http://www.tcptrace.org/>).

This application converts the PCAP to a text format and after by executing simple script we extract the needed parameters into an excel sheet to display the numbers.

Our analysis looks into:

* Download Time
* Throughput
* Retransmission
* Average Bytes in Flights
* Max byte in flights
* Average RTT
* RTT Min in the connection
* Bytes download
* Client and server window scale

Correlation between excel and the Tuning document:

* Download time used for performance gain.
* Retransmission covers a lot of steps in the Tuning workflow.
* Max byte in flight gets the size of the waves in the graph.
* Average bytes in flight in a stable download it is the number we want to tune such that it will be sufficient to achieve maximum speed. Has a strong correlation to targetQueueSize.
* RTT Min is important in order to understand with average bytes in flight but due to timing issue between LAG server members this is not accurate in the PCAP.

This allows us to be more efficient than opening TCP graph on every download as we now need to look at very few graphs to validate the results.

We noticed some issues with the output from the TCPtrace. In these cases we dropped the specific test results (a single download not the full test):

* With 15K downloads we sometimes saw negative duration for the download approximately 7% of the connections had negative duration.
* Sometimes the bytes in flight was a very large number. For example (18446744073709500000) approximately 5% of the connections. We drop this only for the bytes in flight calculation.

The TCP optimization document refers to 4 parameters to change for the tuning process.

With all TCP OPT scenario that we did, packet loss were minimal while non-optimized downloads did suffer from packet loss. Due to the fact that packet loss was minimal we did not change 2 of the parameters at all (dropReduceWindow and congestionWindowCalcWaitTime) as define in the tuning document.

In the document it is suggested to increase the targetQueueSize by 10K and windowConvergenceSpeed  by 10% in every step.

Due to limitation of time (we could only make changes during MW) we increased targetQueueSize by 20K and windowConvergenceSpeed by 20% each time.

During MW we checked S0,S1,S2,S3,SX scenarios out of:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Scenario | Target Queue Size | Window Convergence Speed | Drop Reduce Window | Congestion Window Calc WaitTime |
| S0 | 40K | 60% | 5 | 400 ms |
| S2 | 80K | 60% | 5 | 400 ms |
| S3 | 80K | 80% | 5 | 400 ms |
| S4 | 70K | 60% | 5 | 400 ms |
| S5 | 70K | 70% | 5 | 400 ms |
| S6 | 90K | 80% | 5 | 400 ms |
| S7 | 120K | 80% | 5 | 400 ms |
| SX | 100K | 80% | 5 | 400 ms |

SX with 100K target queue size didn’t improve the 4G performance beyond S3 in the MW and degraded the 3G performance. So we did not continue with target queue size bigger than 80K until the last days.

Summary of the results 3G and 4G:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 4G | | | | 3G |
|  | NC | MC | FC | Sum | Sum |
| S0 | 28% | 29% | 18% | 24% | 1% |
| S2 | 23% | 41% | 12% | 25% | 8% |
| S3 | 32% | 43% | 31% | 35% | 5% |
| S4 | 41% | 27% | 31% | 32% | 12% |
| S5 | 46% | 40% | 9% | 29% | 2% |
| S6 | 45% | 35% | 27% | 34% | 3% |
| S7 | 44% | 34% | 14% | 29% | NA |

In release 1.6 the same profile and parameters are assigned to 3G and 4G subscribers so we recommend going with targetQueueSize 80K and windowConvergenceSpeed 80% (scenario S3)

We also recommend making the changes in 2 stages

1. targetQueueSize 60K and windowConvergenceSpeed  70%
2. targetQueueSize 80K and windowConvergenceSpeed  80%

An additional recommendation: ConteXtream CMDS logs can provide a large amount of information regarding the TCP connections.

ConteXtream CMDS log save the average of maximum speed for all active subscribers. After change in the TCP OPT profile parameters, this number can be compared between similar days of the week with different profile parameters. This comparison will give us the true gains of the change for all subscribers. In particular if we disable TCP Optimization for a few days we can see the performance gain for all subscribers.

Average of maximum speed for all active subscribers calculated by the following criteria:

Active subscriber is subscriber that finishes a TCP connection in the last minute.

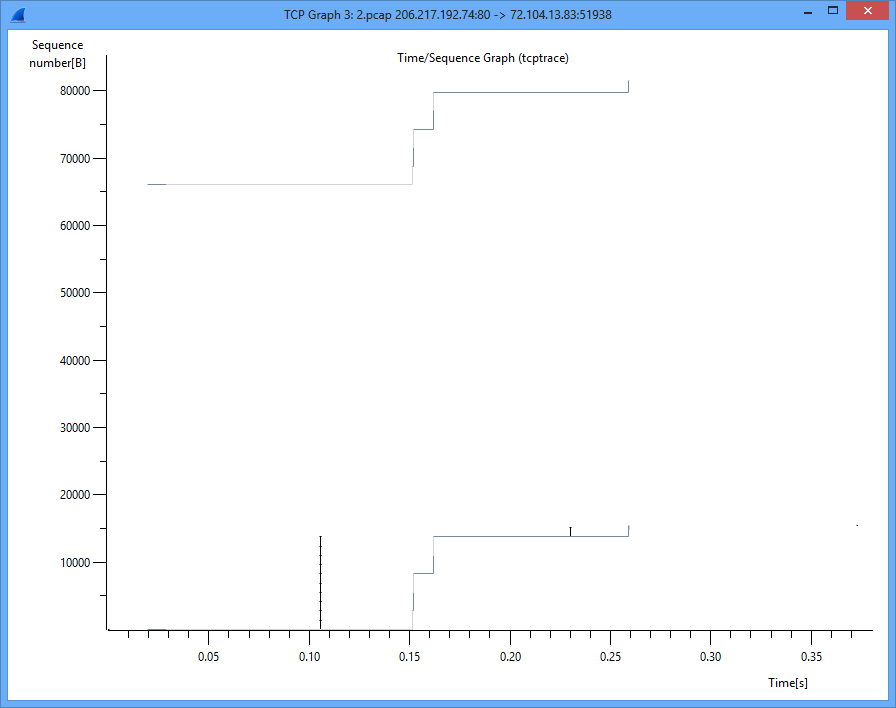
Subscriber maximum speed is the maximum download speed subscriber achieve in the last 10 minutes, calculated as bytes passed in the TCP connection divided by duration of that connection. Maximum speed calculated only on connections that downloaded more than 100KB.

## 15K file size download

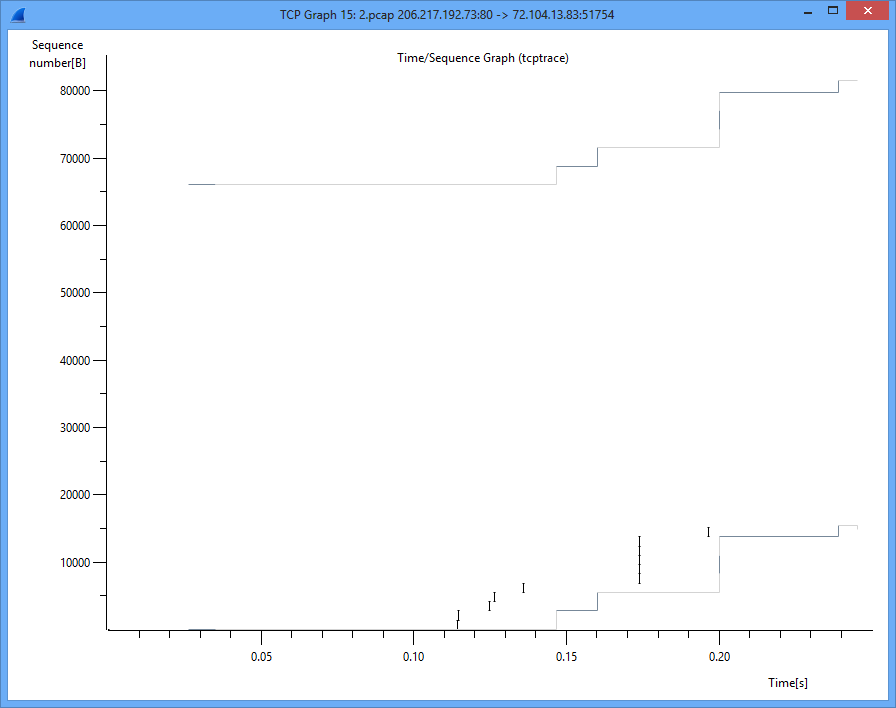
ConteXtream suggests not to include the 15K file size in the tuning calculation since the parameters we change have no influence on 15K download times and download time vary only due to varying conditions.

Far server 15K

In the far server there is higher RTT in the WAN then in the RAN. The server initial window size is very big 13K. 10 packets are sent as a single burst and after the first ACK the remaining 2 packets are sent to the client. This is not common on regular internet servers. Since the server is far it takes a long time until the remaining packets arrive to the client. This gives the TCP OPT the performance gain.

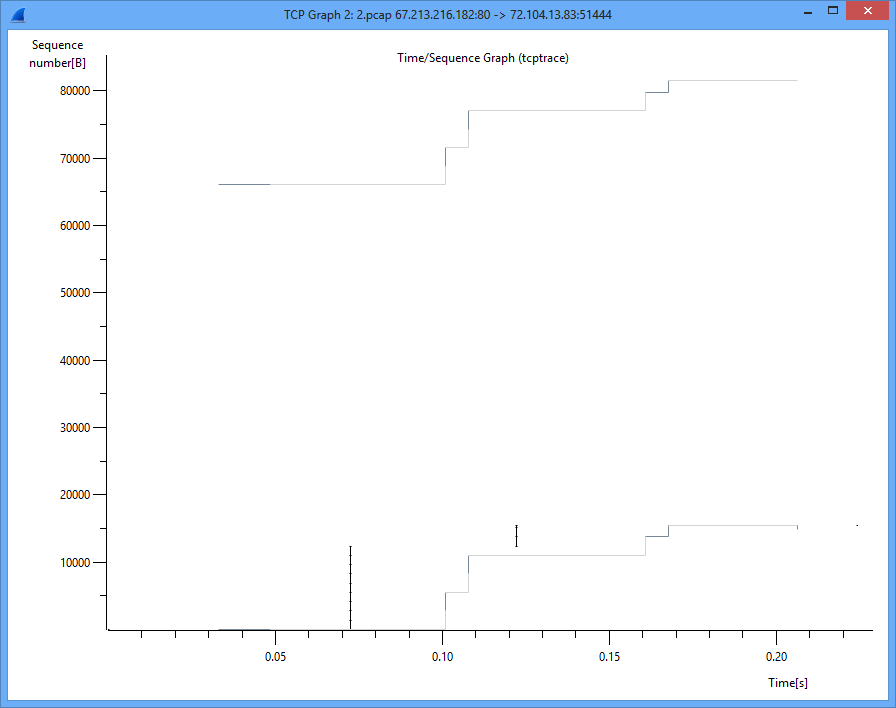


The TCP OPT sends smaller burst while packets arrive but when the packet arrives it immediately ACKs the packet to the server such that the server can send the remaining 3 packets sooner which allow the TCP OPT to send faster the remaining packets.



Near server 15K

There is no performance gain on the local server (Washington) since the server RTT is much less than the RAN RTT. The server initial window size is very big 12K. 9 packets are sent as a single burst and after the first ACK the remaining 3 packets are sent to the client. This is not common on regular internet servers.



The TCP OPT sends the first 8K. 6 packets wait for ACK and then sends the remaining 6 packets. The initial window define as 10000 but if the client has more than 5 connections it will be reduced as it happens in most the small file tests. In 1.6.1 there is a feature to increase the initial window size without increasing the minimum window size.

