2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)

Measuring with JPerf and PsPing: Throughput and Estimated Packet Delivery Delay vs TCP Window Size & Parallel Streams

Nakampe Sydney Sebopetse[1], Chris R. Burger[1], Mofolo Mofolo[1], and Albert A. Lysko[1,2]
[1]NextGen Enterprises and Institutions
Council for Scientific and Industrial Research (CSIR)
Pretoria, South Africa

Albert A. Lysko[1,2]
[2]School of Electrical Electronic and Computer Engineering
North-West University (NWU)
Potchefstroom, South Africa
NSebopetse@csir.co.za

***Abstract*—This work discusses how the TCP window size affects the throughput and its measurements and estimated packet delivery delay on two experimental setups using freely available tools JPerf and PsPing. The first setup is a direct connection between two PCs using gigabit Ethernet. The second setup uses a linked pair of Television White Space (TVWS) devices. The tools JPerf and PsPing are used to perform the throughput and latency measurements, respectively, over a range of different sizes of the TCP window. The results recommend specific ranges of TCP window size to either maximise the throughput or minimise latency. The results also suggest that performing valid measurements of peak throughput require careful consideration in selecting the TCP window size or usage of parallel streams.**

Keywords— TCP, TCP window size, Throughput, Latency, Packet, Delay, Bandwidth, White space devices, WSDs, Television White Spaces, TVWS, Measurements

INTRODUCTION

Today, the transmission control protocol (TCP) version 4 [1] is perhaps the most commonly used network protocol for connecting end user devices over the Internet. The TCP window size is a key parameter of the TCP protocol. It represents how much data a receiving device can receive and store in an intermediate buffer at any given time. It is needed to avoid unnecessary waiting for packet acknowledgement before sending more packets. It thus affects the throughput and latency, especially when both the link capacity and the communication latency are high. The theoretical optimum size of the TCP window (bits), W_{min}, may be expressed as a product of the link's capacity C (in bps) and latency L (in s):

$$W_{\min} = C \times L.$$

The size of this buffer can be made very large to improve the link's achievable throughput. The size is limited by the performance of the computing platform (i.e. memory available and the processing speed).

Substantial work has been done on the effects of the TCP window size on throughput and latency, e.g. [6]6-10]. We focus on the effect of TCP window size when using different types of communication technologies and over a wide range of window sizes, a subject not well covered in the literature.

Measuring a link's peak performance requires setting the TCP window size to an appropriate value or measuring an aggregate of multiple parallel streams. The appropriateness of the window size value is determined by the time it takes for a packet to reach the destination and by the time to receive the acknowledgement of receipt, as indicated above.

This paper analyses experimental results of throughput and latency measurements. The measurements rely on the tools JPerf and PsPing, two of the most commonly used free software tools for network performance evaluations. Two modes, using TCP window size and parallel streams were used.

Section 2 discusses the laboratory measurement setup, including hardware and software components, and measurement methodology. Section 3 presents and analyses the results. The last section concludes the work and lists the next steps.

MEASUREMENT SETUP

The overall setup includes the computers used for performing the measurements and used under the methodology also presented below, as well as software tools.

A. Setup and tools

We used two hardware setups for performing the measurements. One is a reference setup for measuring and testing a direct PC to PC gigabit Ethernet link through a cable. The second setup relies on a radio link, through a pair of Television (TV) White Space Devices (WSDs) [11-13]. Both WSDs are used as sample radio link devices and have the embedded cognitive radio intelligence disabled.

For the measurements, we used JPerf version 2.0.2 [14] and PsPing version 2.1 [15]. Both tools are popular free software.

JPerf [14] is a graphical user interface operating over the tool *iperf*. *iperf* is a link throughput measurement tool. It sends time-stamped packets between PCs and records how quickly a packet moves from the sender to destination and from this estimates the throughput. *iperf* can measure throughput periodically, over a specified time interval (e.g. it can produce averages every second).

PsPing [15] is a free tool used to test latency. *PsPing* was used instead of more traditional *ping* utility as PsPing provides sub-millisecond resolution.

The tools used for measurements are operated in Windows 10 and 7 environments and used default settings for their main functions. *psping* can measure network throughput and bandwidth, but the result depends on the number of iterations and duration of the test.

1. *Hardware Setup I*

The setup in Fig. 1 shows a gigabit Ethernet link between PC1 and PC2 through an Ethernet cable. The PCs have AMD E2-7110 and Intel Core i7 processors respectively, and 4 GB memory each. This setup serves as a reference setup as it relies on a minimum possible amount of hardware for a link (two gigabit network cards connected by Ethernet cable).

Fig 1: PC1 to PC2 connected via gigabit Ethernet

2. *Hardware Setup II*

Setup II, illustrated in Fig. 2 shows a connection via two WSDs. The radio frequency (RF) ports of the WSDs are connected using a radio cable and an attenuator. A 60 dB attenuator is used to ensure a strong radio link without saturating the receivers. The wireless interface of the WSDs is set to operate with QPSK ½ modulation.

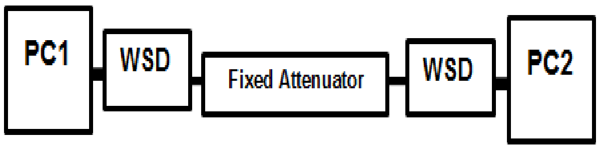

Fig 2: PC1 to PC2 interlinked via WSDs

MEASUREMENT METHOD

Prior to staring the main measurements, the speed achievable with Setups I and II was measured.

The speed achievable with Setup I was measured with the default setting for the TCP window size (56 kB) and using multiple parallel TCP steams. The result is shown in Fig. 3. It is clear that the maximum achievable TCP throughput over the gigabit Ethernet link is around 900 Mbps. Also, the curve shows that for a small TCP window size (56 kB) does not achieve the full speed. Increasing the number of parallel streams from one to about 10 increases the throughput to maximum. This increase is because each stream's throughput is limited by the feedback required by TCP, but the streams are reasonably independent from each other and transfer independent portions of the overall dataset in parallel.

When using between approximately 10 to 20 parallel streams, the throughput stays at its maximum. Assuming that one stream requires 56 kB of memory, all 10 to 20 streams require 560 to 1120 kB of memory. This is slightly more than the optimum size of TCP window required for achieving the best throughput (319 kB), as will be shown later in the paper, in 2.2.3.

Further increasing the number of parallel streams beyond 20 starts to decrease the aggregate throughput. This reduction is likely because the processing overhead (switching between different streams and re-caching different datasets) required by handling multiple parallel streams starts to take a more and more significant portion of the processor's time.

As the radio link between the two WSDs is the main limiting factor for Setup II, its performance was also estimated. The radios' specifications allow for modulation rates up to around 200 Mbps. Fig. 4 lists the WSDs' characteristics. However, the overall speed of the Setup II is hardware-limited to 89 Mbps as a) the WSDs' wired Ethernet network ports can run at a maximum of 100 Mbps and b) the WSDx' radio interface was tested to peak at 89 Mbps under the best conditions.

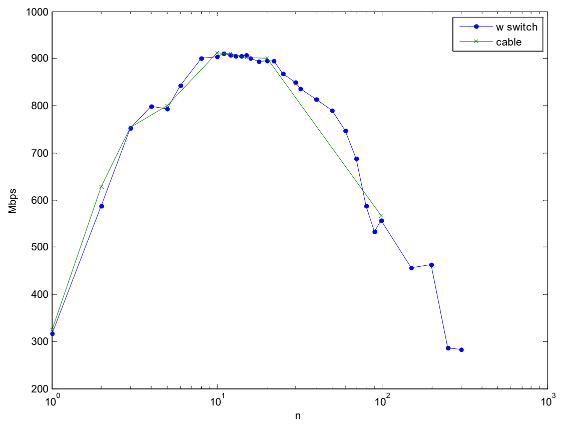

Fig 3: Aggregate throughput (Mbps) measured over the gigabit Ethernet link (Setup I) versus the number of parallel TCP streams. The two curves correspond to using a) a bare cable (green curve with cross as the marker) and b) inserting a common switch in the link (blue curve with dots as markers).

It may also be noted that the TCP throughput in the given WSDs depends on the WSD parameter "distance, km", which expresses the link distance in km. The influence of this parameter is shown in Fig. 5. The likely explanation for the variation is that the embedded protocols use this parameter to decide how long to wait for a response from another node in a half-duplex TVWS network. Increasing this parameter therefore gives more time to the receiver to listen and less time to the transmitter to transmit. This parameter was set to near zero to maximise the throughput.

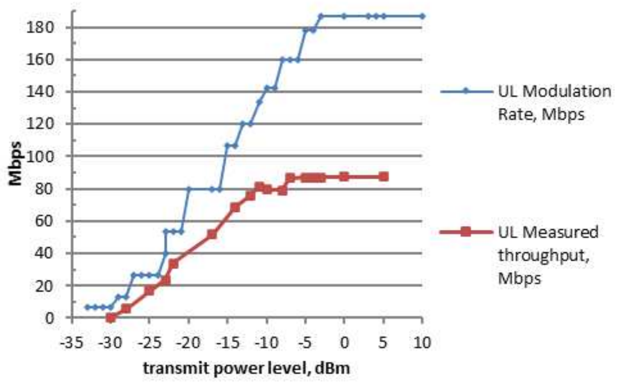

Fig 4: Measured modulation rate and throughput versus power level for an isolated WSD link [13].

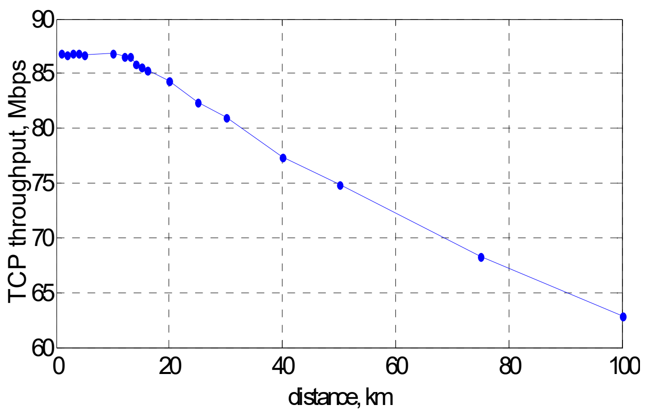

Fig 5: TCP throughput vs. WSD parameter “distance, km” [13].

For the main measurements, the following steps were taken:

A. *Communication between the two PCs is achieved*

- The PCs are set to the same subnet.
- Link latency and round trip delay are measured using PsPing, with default parameters.

B. *For Setup II (with WSDs), communication between the WSDs is established*

- The WSDs are rebooted and set to the default setting.
- Step A is performed on the PCs controlling the WSDs.

C. The expected optimal TCP window size is estimated with the measured latency:

$$S_O = W_B \times t_R,$$

with S_O being the optimal window size in bits, W_B the bandwidth in bits per second (bps) and t_R the return delay in s.

Since we are working with window sizes in bytes, we should scale the parameter accordingly:

$$S_O = W_B \times t_R \div 8 \text{ bits/byte},$$

with SO being the window size in bytes, W_B the bandwidth in bps and t_R the return delay in s.

In anticipation of the next section covering measured results, we calculate the optimal packet size for Setups I and II respectively:

Setup I: S_O = (1 Gbps)×(2.55 ms) ÷ 8 bits/byte= 319 kB
Setup II: (100 Mbps)×(2.45 ms) ÷ 8 bits/byte= 31 kB

D. *Measuring latency and throughput for different TCP window sizes (changing the TCP window size in JPerf, results in a change in the throughput recorded):*

- The throughput is measured using JPerf for various values of the TCP window size.

  Both directions (PC1 to PC2, and PC2 to PC1) are tested separately.

The TCP window size and measured throughput values are used to calculate the equivalent latency, i.e. an estimated packet transmission delay. The need to receive an acknowledgement for every TCP packet must be taken into account.

RESULTS AND DISCUSSION

This section presents and elaborates on the measured and calculated results.

A. Measured throughput versus TCP window size

Fig. 6 presents the throughput measured for various configurations of the two setups.

The throughput curve starts with a minimum, as the acknowledgment of a correct receipt of a transmitted packet is required before a new packet may be transmitted and the packet may need to be broken down into yet smaller packets manageable with the given buffer size. The initial points are at around 50 kbps which may be compared to the maximum capacities of the Setup I and I links being 1 Gbps and 100 Mbps, respectively. This is likely due to the TCP window size being less than or around the size of a single packet.

As the TCP window size starts to accommodate more than one packet, the throughput increases dramatically. The curves continue to grow steadily, until the TCP window size nears the optimal TCP window size (estimated in 2.2.3 as 319 kB and 31 kB for Setups I and II, respectively).

After that point, the achievable throughput remains approximately steady until the TCP window size leads to the need to work with extremely large buffers. It is assumed that the differences in the behaviour of the curves in that region of the plot are due to the differences in the performance of the PCs.

From the above, it is clear that the following rules are important in deciding on the TCP window size:

- TCP window size below a value sufficient to accommodate several packets leads to very slow performance (orders of magnitude below the link capacity);
- To maximise the throughput, increasing the TCP window size to around the optimum value maximises the throughput achievable with TCP v4.
- Increasing the TCP window size above the optimum value does not offer noticeable improvements.

Furthermore, one may compare the shapes of the curves in Fig. 3 and 6 and observe that the use of the TCP window offers a much more efficient way to gain and maintain high throughput. At the same time, if one's target is in reducing the packet delivery delay, the use of parallel TCP streams may be the only way to keep the delay low while trying to improve the efficiency of using the available link capacity.

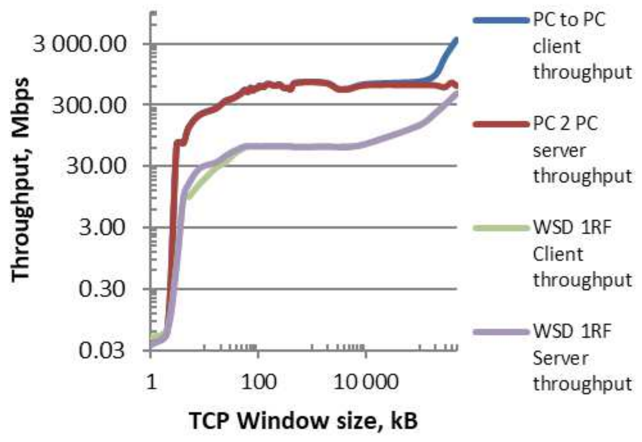

Fig 6: Throughput (Mbps) versus TCP window size (kB). The legend notations are as follows: "PC to PC" refers to Setup I. "WSD" refers to Setup II

B. *Estimated packet delivery delay*

Fig. 7 shows the estimated packet delivery delay (excluding the latency) for the two setups, as a function of the TCP window size.

The equivalent latency of both setups starts at a relatively high value. This excessive delay may be explained by having to wait for receipt confirmation for many small packets. These small packets are also likely to lead to a quick overflow of the buffer, dropped packets and the need to re-transmit packets. This inefficiency is the dominant factor restricting throughput, regardless of the capacity offered by the link.

After the TCP window size allows more than one packet to be transmitted within the window, the packet delivery delay quickly drops to around its minimum. Here it is possible to see the drastic difference between the gigabit Ethernet and much slower radio links. The time to deliver one packet includes the time to transmit this packet (inversely proportional to the capacity of the link) and the latency. For sending one packet, the latency is approximately the same. The observed difference in the delays is roughly inversely proportional to the capacity of the links. This effect of processing the multiple packets with a feedback loop may be contrasted to a negligible difference in the ping-measured single-packet latency between Setups I and II.

As the TCP window size continues to increase, the plot shows that the equivalent packet delivery delay rises approximately monotonically. This rise may be explained by needing to wait for more packets confirmed as received in the receiver's buffer, before releasing these packets for further processing. Presumably, this effect could be reduced by modifying the buffer management software to allow packets to be released for processing in the receiver before the acknowledgement has been sent.

After the TCP window size reaches about 10 MB, and especially after 100 MB mark, the curves start to behave differently, which is likely due to the different processing capabilities of the PCs used in the testing.

From the above, one may conclude that

- The TCP window size must accommodate more than one packet to avoid unnecessary large protocol delays;
- Voice over IP, gaming, and other applications requiring low latency may benefit from having the smallest viable TCP window size, as it minimises the delay in delivering the packets (at the expense of substantially lowering the throughput achievable with TCP v4);
- Increasing the TCP window size past one packet size increases the packet delivery rate approximately proportionally.

CONCLUSIONS AND NEXT STEPS

The paper presents the calculated packet delivery delay and throughput experimentally measured with JPerf on a link clear from interference.

The measurements followed a careful characterisation of the test setup. The measurements indicate that the TCP window needs to be more than one packet large in order to ensure a reasonable throughput but should be around an optimum value, i.e. the product of link capacity and link latency, to maximise the throughput. The packet delivery delay important for VoIP and gaming applications is the lowest for the smallest viable TCP window size able to accommodate just a few packets. This would, however, be at the expense of sacrificing the achievable TCP throughput and possible heavy underutilisation of the available channel bandwidth.

The tests also suggest that measurements of peak TCP throughput could be achieved by optimising TCP window size in a more efficient manner, e.g. with less memory and computational resources, as compared to using parallel streams. The use of TCP window offers a much more efficient way to gain and maintain high throughput. At the same time, if one's target is in reducing the packet delivery delay, the use of parallel TCP streams may be the only way to keep the delay low while trying to improve the efficiency of using the available link capacity.

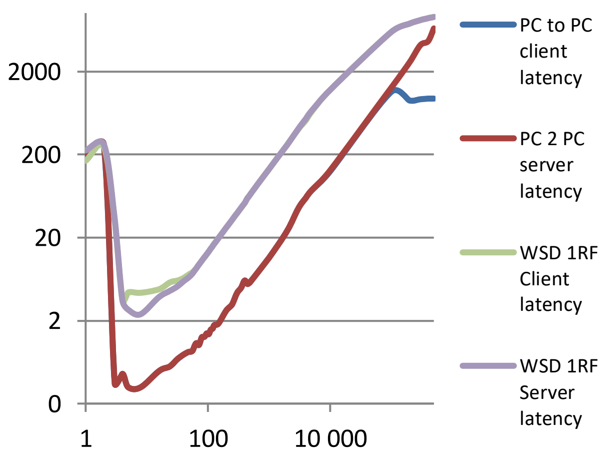

Fig 7: Equivalent packet delivery delay (ms) versus TCP window size (kB)

References

[1] W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols. Addison-Wesley Pub. Co., 1994.

[2] W. R. Stevens; G. R. Wright, TCP/IP Illustrated, Volume 2: The Implementation, 1994.

[3] W. R. Stevens, TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols, 1996.

[4] RFC 675, Specification of Internet Transmission Control Protocol.

[5] Guru99.com. 2021. TCP vs UDP: What's the Difference?. [online] Available at: https://www.guru99.com/tcp-vs-udp-understanding-the-difference.html. Last accessed on 12 February 2021.

[6] D. Madhuri and P. C. Reddy, "Performance comparison of TCP, UDP and SCTP in a wired network," 2016 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, 2016, pp. 1-6, doi: 10.1109/CESYS.2016.7889934.

[7] Shriram A. et al. Comparison of Public End-to-End Bandwidth Estimation Tools on High-Speed Links. In: Dovrolis C. (eds) Passive and Active Network Measurement. PAM 2005. Lecture Notes in Computer Science, vol 3431. Springer, Berlin, Heidelberg. 2005. https://doi.org/10.1007/978-3-540-31966-5_24.

[8] JDSU Application Note "TCP Wirespeed: Testing TCP Throughput to 10 G," 2010. Available online at https://www.viavisolutions.com/de-de/literature/tcp-wirespeed-testing-tcp-throughput-10g-application-notes-en.pdf . Last accessed on 2019-11-15.

[9] Spirent TCP Network Latency and Throughput Or 'Why your customer doesn't receive the Throughput they paid for', 2016. Available online at https://www.spirent.com/Assets/WP/WP_TCP-Network-Latency-and-Throughput . Last accessed on 2018-06-22.

[10] E. Pelletta and H. Velayos, Performance measurements of the saturation throughput in IEEE 802.11 access points, Third International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'05), April 2005.

[11] H. Mauwa, et al., "Systematic analysis of geo-location and spectrum sensing as access methods to TV white space," ITU Kaleidoscope: ICTs for a Sustainable World (ITU WT), 2016, pp. 1-8.

[12] M.T. Masonta, D.L. Johnson, and M. Mzyecel, "The White Space Opportunity in Southern Africa: Measurements with Meraka Cognitive Radio Platform," in R. Popescu-Zeletin et al. (Eds.) AFRICOMM 2011, LNICST 92, pp. 64–73, 2012.

[13] A. A. Lysko, "Lessons learned from TVWS trials, tests and further research," Invited talk for WAPAZOLA 2019, Pretoria, South Africa, 17 September 2019.

[14] Rarst.net . Available online at https://www.rarst.net/software/JPerf/ , 2010. Last accessed at 2018-06-22.

[15] PStools, 2017, Available online at https://ss64.com/nt/psping.html. Last accessed on 2020-12-30.