# ITS 413 Internet Technologies and Applications

*Assignment: Phase 3 Report (Final report)*

By:

Ms.Chittra Roungroongsom ID: 5222780488

Mr.Songklod Amornkul ID: 5222792673

Mr.Chawarat Tantivivathanaphand ID: 5222800633

Date: 28/3/2012

By submitting this report all members of the group listed above agree that each member has contributed approximately equal amounts to designing and performing experiments, as well as to preparing this report. All members agree that this report accurately reflects the experiments conducted by the group members, and is their own work (not works of other groups).

Sirindhorn International Institute of Technology

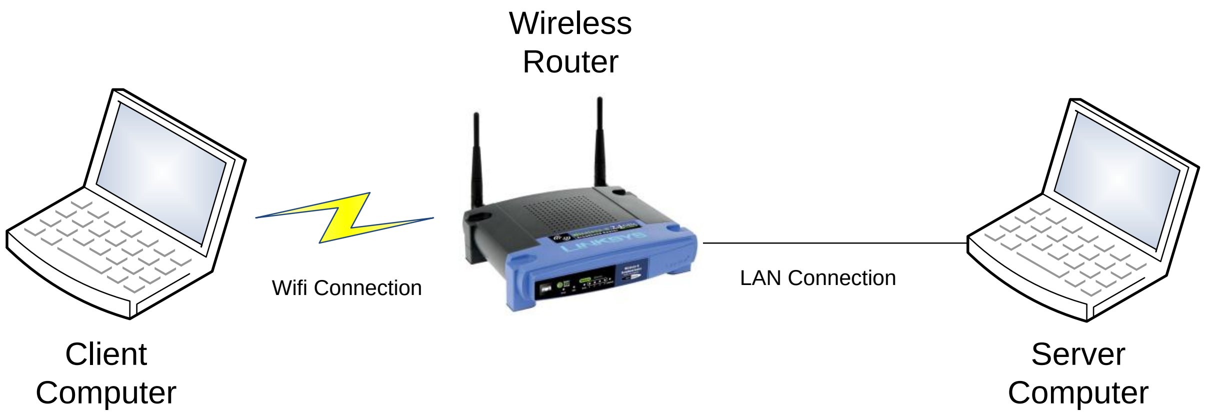
Thammasat University

# Table of Content

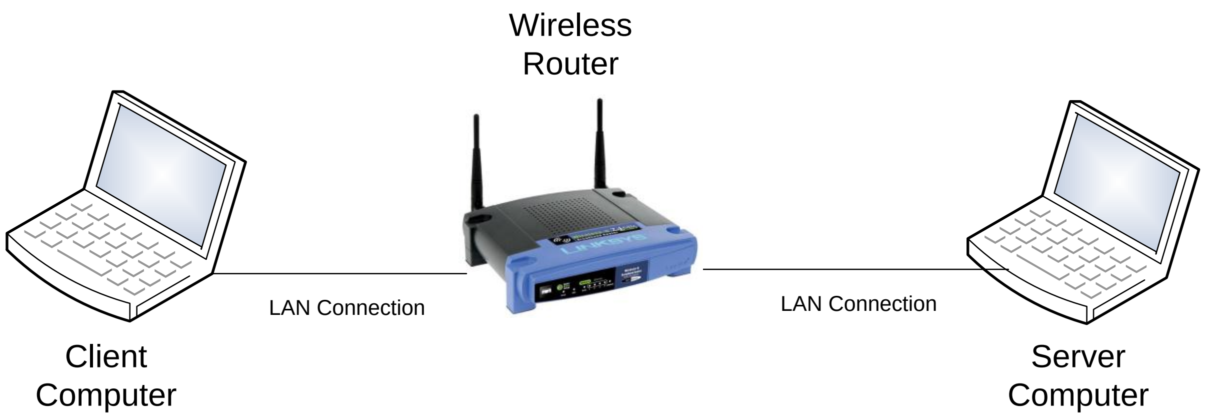
## Aims

To determine how different parameters and scenarios impact on the performance of transport protocols (TCP and UDP) in a wireless LAN and/or Ethernet link.

## Network Diagram




**Wireless Lan Connection**




**Ethernet Link**

## Equipment Specifications

***<u>Laptop #1 (Server Computer)</u>***

**HP Pavilion dv3 2145tx**

Processor: Intel Coe 2 Duo 2.67GHZ
Graphic Card: NVIDIA GEFORCE 512 Mb
Ram: 4.00 GB (2.97 GB Usable)
Network Interface Card: Integrated 10/100Base-T Ethernet LAN (RJ-45 connector)
Wireless Connectivity: 802.11agn
OS: Ubuntu 10.04 (64bit)

***<u>Laptop #2 (Client Computer)</u>***

**Toshiba Portege M800**

Processor: Intel Coe 2 Duo 2.1GHZ
Graphic Card: Onboard (Intel 965 express chipset family)
Ram: 2.00 GB
Network Interface Card: Marvell Yukon 88E8072 (10/100/1000Mbps Gigibit Ethernet)
Wireless Connectivity: 802.11agn
OS: Ubuntu 11.10

***<u>Router</u>***

**Linksys Wireless-G Broadband Router WRT54GL V1.1**

Device type: Wireless router - 4-port switch (integrated)
Data Link Protocol: Ethernet, Fast Ethernet, IEEE 802.11b, IEEE 802.11g
Data Transfer rate: 54 Mbps
Frequency band: 2.4 GHz
OS: OpenWrt (Backfire 10.03.1)

## Experiments and Results

We have conduct 4 experiments which aim to determine how different parameters and scenarios impact on the performance of transport protocols (TCP and UDP) in a wireless LAN and/or Ethernet link**.**

All the experiment conducted in Phase 3 was repeated at least 3 times to find the average value. This was done to ensure that all the data collected is fair and not bias.

Here is the following list of experiments we have conduct

- TCP VS. UDP
- TCP and UDP session
- Impact of packet drops on TCP performance
- Impact of window sizes on TCP performance

Iperf option used in experiment

-c = run iperf as client mode

-s = run iperf in server mode

-u = set client or server as UDP (The default will be TCP if option –u is not use)

-t = set time period for each transmission (unit: second)

-b = specify limit of sending rate (require UDP [-u])

-w = Adjust the TCP protocol’s window size

-P = Enable and set the number of parallel client threads to be executed at a time

## Experiment 1: TCP vs. UDP

### 1.1) Aim:

This experiment is aim to compare the reliability of the TCP and UDP protocol over a wireless connection at increasing period of time.

### 1.2) Connection: Wireless

### 1.3) Parameters:

- ***Bandwidth* (UDP)** = 30 Mbit/s (default value for the experiment 1)
- ***Time* (s)** = 10(s) - 120(s) (increment of 10s per round)

### 1.4) Methodology:

A. We set up the Laptop #1 (IP address 192.168.1.135) as a server (***-s***) with wired-connected to the router in TCP mode.

```
iperf –s
```

B. We set up the Laptop #2 (IP address 192.168.1.197) as a client (***-c***), sending a TCP packet to the server (Laptop #1) via wireless at a time with the period (***-t***) variant from 10, 20 ,30, …, 120 sec

```
iperf –c 192.168.1.135 –t 10
```

C. We set up the Laptop #1 as a server with wired-connected to the router in UDP mode (***-u***).

```
iperf –s –u
```

D. We set up the Laptop #2 as a client, sending a UDP packets (***-u***) to the server (Laptop #1) via wireless at a time with the period (***-t***) variant from 10, 20 ,30, …, 120 sec and the sending rate (***-b***) 30 Mbps

```
iperf –c 192.168.1.135 –u –b 30Mbits –t 10
```

### 1.5) Experiment Results:

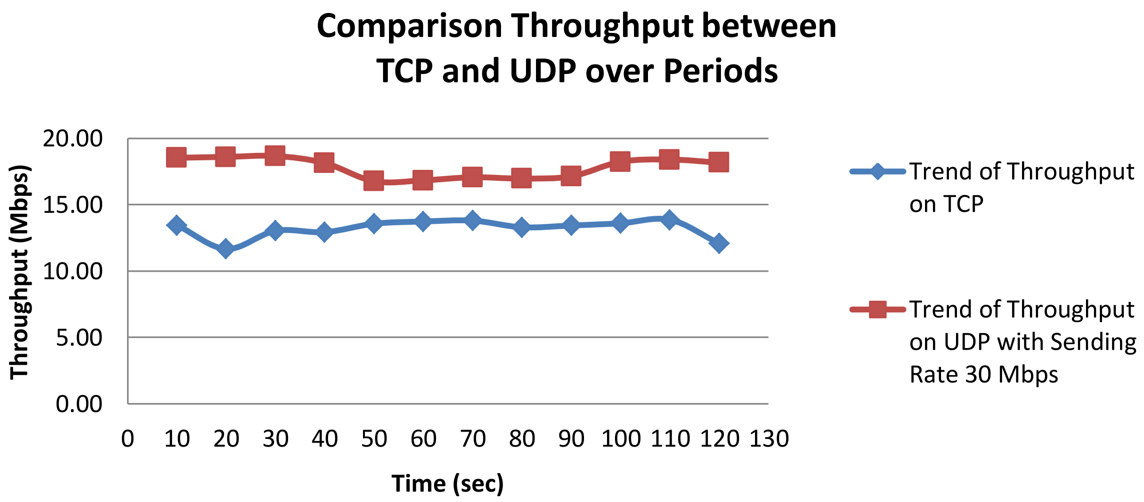



***Figure 1.1***

### 1.6) Explanation:

According to the graph in Figure 1.1, it is showed that the throughput on both protocols is quite stable as the time period for each transmission is increase. But it is clear to say that the throughput of TCP is always lower than UDP (within this scope of experiment).

It is because the TCP is known as connection-oriented protocol, a connection can be made from client to server need to be established and send data along that path. In additions, TCP offers the guaranteed delivery i.e. flow control and error control to make the transmission more reliable.

In contrast, the UDP is a simpler message-based connectionless protocol that just sends the data at sending rate requested by the user. It ignores the connection establishment and the guaranteed delivery. Therefore, the UDP requires less time to send data than the TCP and lead the higher throughput than TCP also. So, the trade-off between how fast you transmit and the reliable in transmission you will get in sending some data.

### 1.7) Conclusion:

Using the different protocol will determine the factor that impacts the performance of data transmission. If user sends data using TCP, user will get the throughput lower than of UDP. It is because the TCP is a connection-oriented protocol and requires more time to send while the UDP is connectionless protocol and requires less time and better in term of performance.

## Experiment 2: TCP and UDP Sessions

### 2.1) Aim:

The aim of this experiment is to see how different protocol (UDP, TCP) shared the limited resource among each other when single thread or multiple threads of both UDP and TCP protocol were executed at the same time.

### 2.2) Connection: wired

### 2.3) Parameters:

- ***Sending Rate*** – define for UDP = 5 Mbit/s (Default value for experiment 2)
- ***Time*** – 60s (Default value for experiment 2)
- ***Parallel threads in session***
  - 1 TCP session and 1 UDP session
  - 1 TCP session and 2 UDP session
  - 2 TCP session and 1 UDP session
  - 2 TCP session and 2 UDP session

### 2.4) Methodology:

A. We set up the Laptop #1 (IP address 192.168.1.135) as a server (***-s***) with wired-connected to the router in TCP mode.

```
iperf –s
```

B. We set up the Laptop #1 as a server with wired-connected to the router in UDP mode (***-u***).

```
iperf –s –u
```

A. We set up the Laptop #2 (IP address 192.168.1.197) as a client (***-c***), sending a TCP packets to the server (Laptop #1) via wired connection for (***-t***) 60s, and the number of multiple parallel threads are (-P) 2 per sessions

B.

```
iperf –c 192.168.1.135 –t 60 –P 2
```

C. We set up the Laptop #2 as a client, sending a UDP packets (***-u***) to the server (Laptop #1) via wired connection for (***-t***) 60s. The fixed sending rate (***-b***) is 5 Mbit/s, and the number of multiple parallel threads are (-P) 2 per sessions

```
iperf –c 192.168.1.135 –u –b 5M –t 60 –P 2
```

D. Prompt the commands in turn

### 2.5) Experiment Results:

1 UDP session and 1 TCP session
(Sum throughput of the network = 94.66)

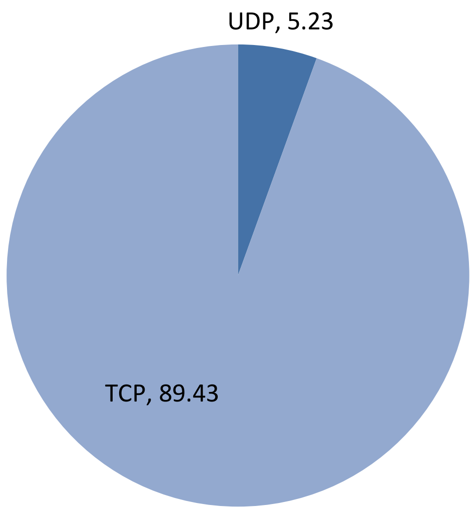



***Figure 2.1***

**2 UDP sessions and 1 TCP session**
(Sum throughput of the network = 94.87)

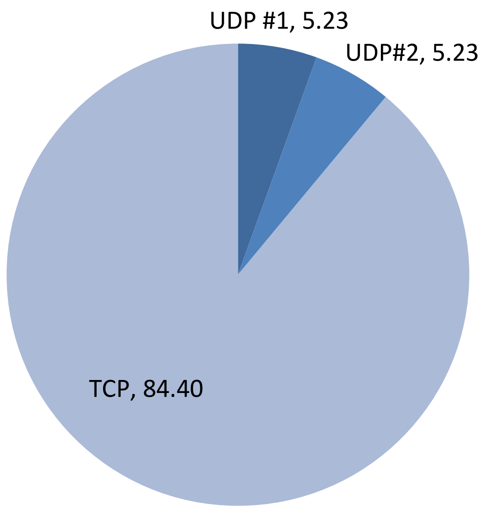



***Figure 2.2***

**1 UDP session and 2 TCP sessions**
(Sum throughput of the network = 95.34)

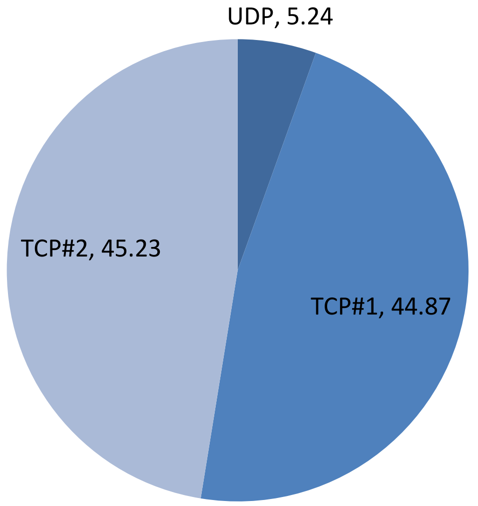



***Figure 2.3***

**2 UDP sessions and 2 TCP sessions**
(Sum throughput of the network = 96.13)

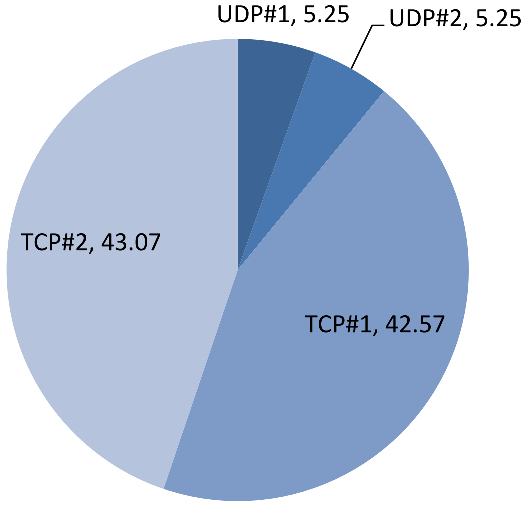



***Figure 2.4***

### 2.6) Explanation:

According to result display in all of the pie chart, the overall network throughput stayed about 95 Mbit/s at bottleneck. However when there was one or more UDP sessions present in the network consisting of varied number of TCP sessions, the TCP sessions will share the sending rate equally among each other, this is call "TCP fairness" Which is part of TCP approach. The more number of UDP in sessions, the more it will impact TCP performance cause the decreasing of throughput in each TCP session while the throughput of UDP is still the same (5 Mbit/s).

According to Figure 2.4, each UDP session gained the throughput about 5 Mbit/s. So the remaining throughput that available in network for TCP sessions is 95 Mbit/s – 2*5 Mbit/s = 85 Mbit/s. It is according to the TCP fairness, if TCP is fair with "N" connections using "R" Mbit/s, each TCP connection should achieve "R" / "N" Mbit/s. As you can see that the throughput was equally shared among two TCP sessions. It showed that two UDP sessions made each TCP session gain less throughput compare to Figure 2.3 that has only one UDP session in the network.

To conclude, when there are other non-TCP sessions such as UDP, running together with TCP sessions in the same network, TCP sessions will receive unfair treatment compared to UDP. Since TCP fairness does not apply to UDP, it will gain the exact throughput that they requested by user and let TCP sessions share the rest of the available resource and while the TCP fairness will still apply among all TCP threads. And due to this TCP does not gain the fair treatment when compare to UDP. From this experiment, it can say that more the more number of TCP session, less resource is available for TCP, thus reduce the possible throughput it can send.

### 2.7) Conclusion:

Execute multiple sessions in UDP impacts on performance of TCP by decreasing the overall throughput of TCP because all sessions running need to share bandwidth among the network. TCP session may receive unfair treatment compared to UDP sessions in the whole network.

## Experiment 3: Impact of Packet Drops on TCP performance

### 3.1) Aim:

This experiment demonstrates how packet drop occur in transmission will impact on throughput in a wireless LAN or not and how it impacts.

### 3.2) Connection: wireless

### 3.3) Parameters:

- ***Time*** – static period at 60s (default value for experiment 3)
- ***Loss*** –0% to 25 % (increment of 5% per round)

### 3.4) Methodology:

A. We want to drop data packets transfer from Laptop #2 to Laptop #1 at certain percent (example: 5%). Since laptop #1 transmits on wireless interface wlan0 to randomly drop packets sent by computer A run the command:

```
tc qdisc add dev wlan0 root netem loss 5%
```

B. To change wireless lan port 0 to drop packets randomly (in percentage)

```
tc qdisc change dev wlan0 root netem loss 5%
```

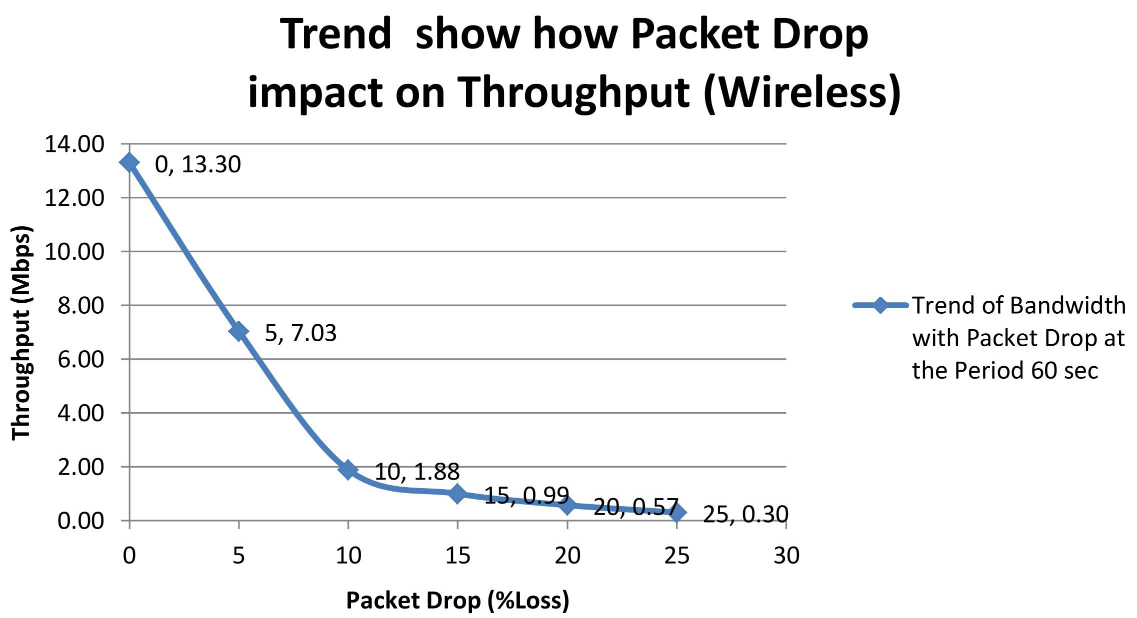
C. To set up TCP server

```
iperf –s
```

D. We set up the Laptop #2 as a client, sending a UDP packets (***-u***) to the server (Laptop #1) via wireless at a time with the period (***-t***) 60 sec.

```
iperf –c 192.168.1.135 –t 60
```

### 3.5) Experiment Results:




### 3.6) Explanation:

According to graph, the throughput of the network reached its maximum value of 13.30 Mbps when there were no packets dropped (loss = 0%). Which when we start to apply the packets drop in the network, the throughput exponentially decreased as the packets drop rate increased.

The dropping of packet can occur from several reasons. In the case that the packets never reach to the receiver, it may cause from the congestion when sending rate is greater than output capacity at router and the packet is refused from the router and loss. For TCP that provide reliability in transmission, when the sender will time out for waiting the ACK, the sender will try to retransmit the packets which require more time to send those packets and eventually lead the throughput. Although, in the case that the packets reach to the receiver but the buffer at the receiver is full or ACK to reply back is lost, they results in the same manner as well.

### 3.7) Conclusion:

When packets drop occur in the network it impacts the performance, in which result in exponential decrease of throughput as the rate of packet drop increase. The cause may come from several reasons, such as, time out at sender before ACK is coming, and packets were being at router or receiver, packet lost during transmission.

## Experiment 4: Impact of Window Sizes on TCP Performance

### 4.1) Aim:

The aim of this experiment is to see how window sizes impact the throughput of TCP protocol in a wired connection as well as how it impacts.

### 4.2) Connection: wired

### 4.3) Parameters:

- ***Time*** – static period at 60s (default value for this experiment)
- ***Window Size*** – Keep doubling with 2 Kbits from 2 to 512 Kbits

### 4.4) Methodology:

A. To set up TCP server with doubling 2, 4, 8, 16, …, 512 Kbits

```
iperf –s –w 2K
```

B. We set up the Laptop #2 as a client, sending a UDP package (***-u***) to the server (Laptop #1) via wireless at a time with the period (***-t***) 60 sec.

```
iperf –c 192.168.1.135 –t 60
```

### 4.5) Experiment Results:

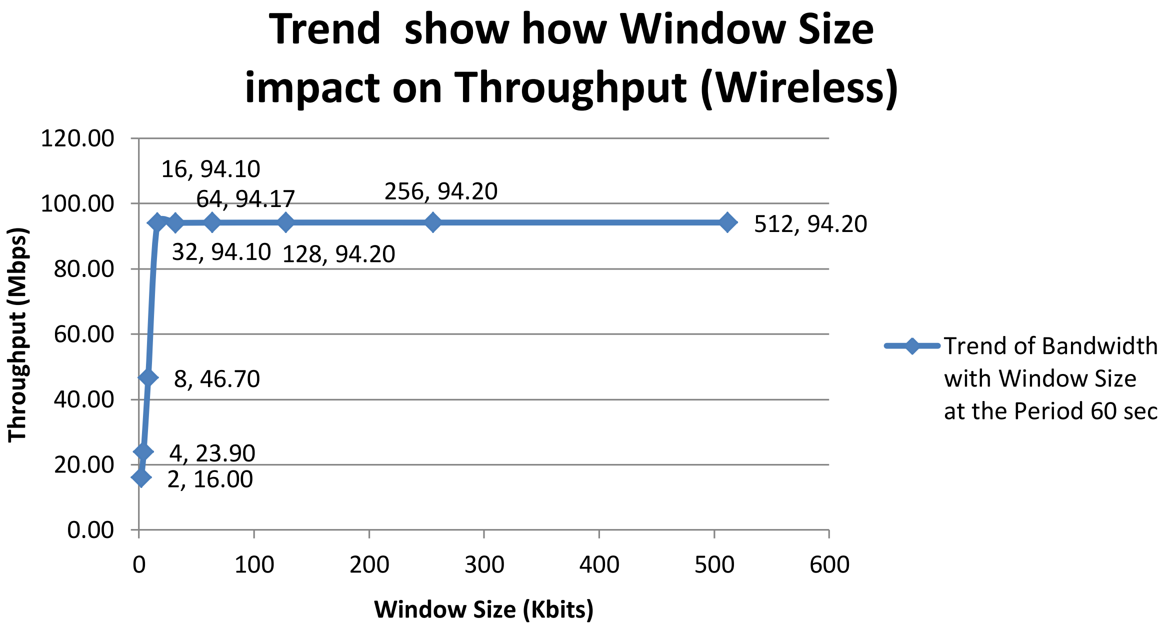



***Figure 4***

### 4.6) Explanation:

According to the graph, it shows that the throughput of the network increase as the window size increases until it reaches the maximum sending rates possible within the network. So, we can say that the window sizes of the sender impact on TCP performance result in increasing the throughput when the window size increase until reach the maximum value.

For the first scenario of the plot, the throughput is low. This is because the time to transmit the window is less than the time to receive the first ACK. So it needs to wait long round trip time (RTT) before getting ACK and result in low throughput. But in the transmission with a larger window size, the RTT to transmit the packets were larger than the time to receive the first ACK. Therefore as soon as the client has finished transmitting the window of packets, the client has already received an ACK and can immediately send the next packet. However, the sender doesn't wait and spends all the time transmitting and producing the maximum throughput.

The result refers to the Bandwidth Delay Product (BDP). If the advertised window (awnd - receive buffer) is equal to or greater than the BDP, then the client can achieve maximum throughput. If the advertised window is less than the BDP, then the throughput will be awnd/RTT

### 4.7) Conclusion:

The window sizes of the sender impact on TCP performance result in increasing the throughput when the window size increases until reach the maximum value of throughput. It is because the window sizes is large enough for one round trip time that when sending data, the sender do not need to wait for ACK and can send data more immediately. So, it reaches the maximum throughput.