Testing the Channel Aggregation Capability of the MPT Multipath Communication Library

Gábor Lencse
Department of Telecommunications
Széchenyi István University
Győr, Hungary
lencse@sze.hu

Ákos Kovács
Department of Telecommunications
Széchenyi István University
Győr, Hungary
kovacs.akos@sze.hu

Abstract—**The MPT multipath communication library is a promising solution for several problems including reliable data transmission using TCP, real-time transmission using UDP and also wireless network layer routing problems. MPT can aggregate the capacity of multiple physical channels. In this paper, the channel aggregation capability of the MPT library is tested up to twelve channels with 100Mbps or 1000Mbps speed each. Different scenarios are used: both IPv4 and IPv6 are used as the underlying and also as the encapsulated protocols. For the throughput measurements, two high performance Linux workstations having each three quad-port gigabit Ethernet NICs are used with the `iperf` industrial de facto standard network testing tool. First, the speed of the NICs is limited to 100Mbps by a switch to be able to test the aggregation of high number of channels but then the bare gigabit speed channels are also tested. The aggregate throughput results are presented as graphs in the function of the number of 100Mbps or 1000Mbps channels. The results are interpreted and discussed. Several directions of our future research are outlined and two recommendations are given for the further development of the MPT library.**

multipath communication; performance testing; channel capacity aggregation

I. INTRODUCTION

Many of our ICT devices (e.g. smart phones, tablets, notebooks) have multiple communication interfaces (e.g. Ethernet, WiFi, HSDPA/LTE) but we can use only one of them at a time due to technical reasons: the endpoint of a TCP/IP communication is identified by an IP address plus a port number and the IP addresses are always bound to the network interfaces [1]. The MPT multipath communication library [2] was developed at the *Faculty of Informatics, University of Debrecen*, Debrecen, Hungary. It makes possible to aggregate the transmission capacity of the multiple interfaces of a device. Its performance, especially its channel aggregation capability for two channels was analyzed in [3] and for four channels in [4] using serial links with the speed of a few megabits per second. As the MPT library may be useful for many different purposes including stream transmission [4], cognitive info-communication [5] and wireless network layer roaming problems [6], we decided to do further tests of its channel aggregation capability increasing both the number of physical channels up to 12 and their bandwidth to 100Mbps or even to 1000Mbps.

The remainder of this paper is organized as follows. First, a brief introduction is given to the MPT multipath communication library. Second, our test environment is described. Third, our experiments are presented and our results are interpreted. Finally, our conclusions are given.

II. MPT IN A NUTSHELL

A. The architecture of MPT

The innovation of the MPT multipath communication library can be highlighted by a comparison with the much more well-known Multipath TCP [7].

MPTCP uses multiple TCP sub-flows on the top of potentially disjoint paths, see Fig. 1. This is a good solution for the aggregation of the transmission capacity of the underlying paths. The reliable byte stream transmission offered by TCP is a proper solution for a class of applications such as web browsing, sending or downloading e-mails, etc. However, it is undesirable for another class of applications such as IP telephony, video conference or other real-time communications where some packet loss (with low ratio) can be better tolerated than high delays caused by TCP retransmissions.

The MPT multipath communication library uses UDP/IP protocols on the top of each link layer connection and creates an IP tunnel on top of them. Thus both TCP and UDP can be used over the IP tunnel, see Fig. 2. Therefore retransmissions can be omitted if they are not required. This design makes MPT more general than MPTCP thus permitting MPT more areas of applications.

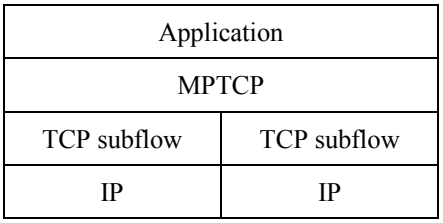

Figure 1. The architecture of the MPTCP protocol stack

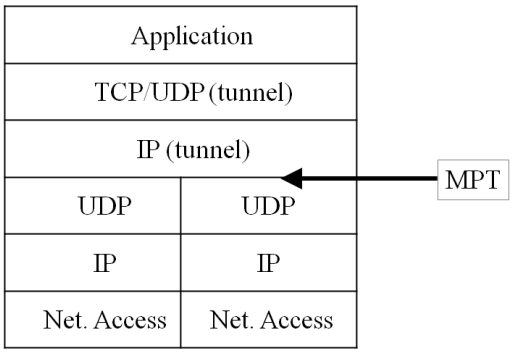

Figure 2. The layered architecture of the MPT software [3]

B. *The configuration and usage of the MPT library*

MPT is multipath communication library developed under Linux and can be downloaded from [8]. The distribution contains an easy to follow user manual. When setting up MPT, the software must be present at both endpoints. One of them should be configured as server and the other one as client, but the applications see it completely symmetrical. It has simple and straight forward configuration files where the details must be given (e.g. the number of physical connections, the Linux network interface names and IP addresses for each channel, name of the tunnel interface, etc.), see more details later on. When both sides are configured and the MPT software is started on both computers, the applications can use the tunnel interfaces for communication in the ordinary way. It is the task of the MTP library to distribute the user's traffic for all the physical channels to be able to take the advantage of the multiple network interfaces.

III. TEST ENVIRONMENT

A. *Hardware and basic configuration*

Two *DELL Precision Workstation 490* computers were used for our tests. Their basic configuration was:

- DELL 0GU083 motherboard with Intel 5000X chipset
- Two Intel Xeon 5140 2.33GHz dual core processors
- 8x2GB 533MHz DDR2 SDRAM (accessed quad channel)
- Broadcom NetXtreme BCM5752 Gigabit Ethernet controller (PCI Express, integrated)

Three *Intel PT Quad 1000* type *four port* Gigabit Ethernet controllers were added to each computers, thus they both had 3x4+1=13 Gigabit Ethernet ports, from which the integrated one was used for control purposes and the other ones were used for the measurements. The computers were interconnected by a *Cisco Catalyst 2960* switch limiting the transmission speed to 100Mbps and separating the 12 physical connections from each other by VLANs. Different versions of IP (v4 or v6) were used for our experiments. Fig. 3 shows the network that was used in the *IPv4 tunnel over IPv4 connections* tests.

Debian wheezy 7.4 GNU/Linux operating system was installed on both computers. For the first series of experiments, the network interfaces of the computers were configured as shown in Fig. 3.

B. *Configuration of the MPT software*

The version of the MPT library can be identified by the name of the file which contains the date in the YYYY-MM-DD format: `mpt-lib-2014-03-25.tar.gz`. This version of the MPT library contained precompiled 32-bit executables with statically linked libraries thus we did not need to compile it. We set it up following the instructions of the user manual [2]. It was a simple and straight forward task. The contents of the following two configuration files were set as follows. (Their path is relative to the installation directory of MPT.)

The beginning of the `conf/interface.conf` file was:

```
################ Interface Information: ################
12              # The number of the interfaces
65020           # The local cmd port number
1               # Accept remote new connection request
```

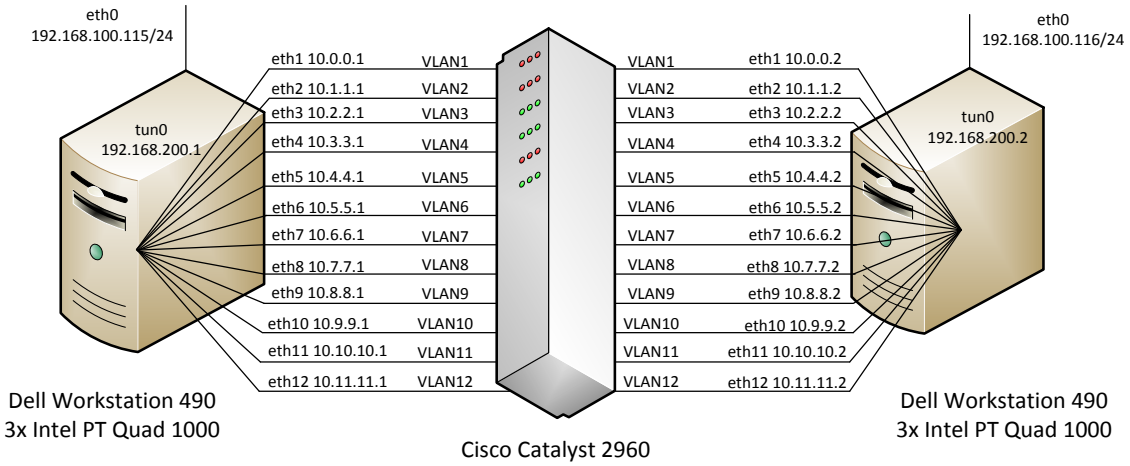

Figure 3. The topology of the test network (IPv4 tunnel over IPv4 connections)

```
################ Tunnel interface ################
tun0  # INT. NAME, the first int. must be the tunnel int.
192.168.200.1/24        # IPv4 address and pref. length
fd00:de:200::1/64       # IPv6 address and pref. length
################ ETH1 interface ################
eth1
10.0.0.1/24
fd00:de:201::1/64
################ ETH2 interface ################
eth2
10.1.1.1/24
fd00:de:202::1/64
```

And it was similar for all the other interfaces, which we do not list due to space limitations.

While the IP setting of the interfaces could be described in a common file for IPv4 and IPv6, the different types of tunnels are to be given in separate connection files.

The IPv4 tunnel over IPv4 paths was defined in the `conf/connections/IPv4overIPv4.conf` file:

```
######## Multipath Connection Information: ########
1               # The number of the connections
################ New Connection ################
TILB            # CONNECTION NAME
3               # SEND(1)/RECEIVE(2) CONNECTION UPDATE
4               # IP VERSION
192.168.200.1   # LOCAL IP
65022           # LOCAL DATA PORT
192.168.200.2   # REMOTE IP
65022           # REMOTE DATA PORT
65020           # REMOTE CMD PORT
12              # NUMBER OF PATHS
0               # NUMBER OF NETWORKS
2               # KEEPALIVE TIME (sec)
5               # DEAD TIMER (sec)
0               # CONNECTION STATUS
0               # AUTH. TYPE
0               # AUTH. KEY
################ Path 0 information: ################
eth1                    # INT. NAME
4                       # IP VERSION
00:15:17:54:d7:30       # LOCAL MAC ADDR
10.0.0.1                # LOCAL IP
00:00:00:00:00:00       # GW MAC ADDR
0.0.0.0                 # GW IP
10.0.0.2                # REMOTE IP
100                     # WEIGHT IN
100                     # WEIGHT OUT
1                       # PATH WINDOW SIZE
0                       # PATH STATUS
################ Path 1 information: ################
eth2                    # INT. NAME
4                       # IP VERSION
00:15:17:54:d7:31       # LOCAL MAC ADDR
10.1.1.1                # LOCAL IP
00:00:00:00:00:00       # GW MAC ADDR
0.0.0.0                 # GW IP
10.1.1.2                # REMOTE IP
100                     # WEIGHT IN
100                     # WEIGHT OUT
1                       # PATH WINDOW SIZE
0                       # PATH STATUS
```

It was also set in the same manner for all the other paths of this connection and for the other connections as well.

Note that the configuration files follow strict format, even the comment only lines must be present. (We experienced fatal errors when the format was not precisely followed.)

IV. EXPERIMENTS AND RESULTS

The channel aggregation capability of the MPT library was measured by using the industrial de facto standard `iperf`. Both IPv4 and IPv6 were used as the IP protocol for the tunnel and also as IP protocol for the underlying channels. It means altogether 2x2=4 series of measurements, were the number of physical channels were increased from 1 to 12. Thus we performed 4x12=48 different tests. The tests were automated by scripts. Due to space limitations, we cannot include the complete measurement scripts, but the key commands only. The ones below belong to the *IPv4 tunnel over IPv4* measurements. The `iperf` command was:

```
iperf -c 192.168.200.1 -t 100 -f M
```

This command performed a 100 seconds long test and printed the throughput in MBytes/s units. This is called the client side in `iperf` terminology. On the other side, the server was started with the following command line:

```
iperf -s
```

The results will follow ordered by the IP versions.

A. Performance of the IPv4 tunnel over IPv4

The results of the `iperf` test are shown in Fig. 4. The throughput aggregation capability of the MPT library proved to be very good, the performance scaled up nearly linearly up to 12 NICs.

B. Performance of the IPv6 tunnel over IPv4

Our next scenario was the IPv6 tunnel over IPv4 paths. Fig. 5 shows the throughput results. The graph is linear up to 12 NICs.

Note that these results are not at all trivial, as before our experiments, MPT has been tested up to 4 physical channels having only a few Mbps speed.

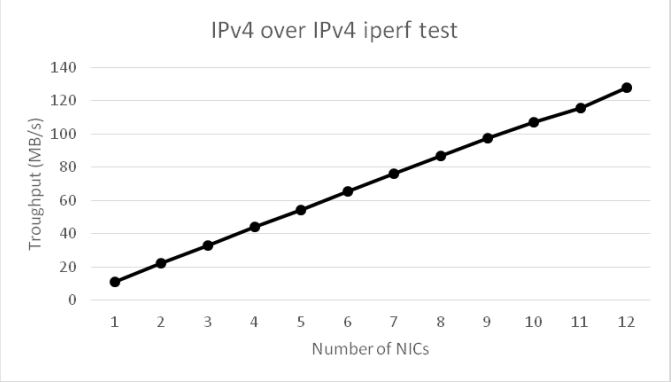

Figure 4. The throughput results of the iperf test of an IPv4 tunnel over IPv4

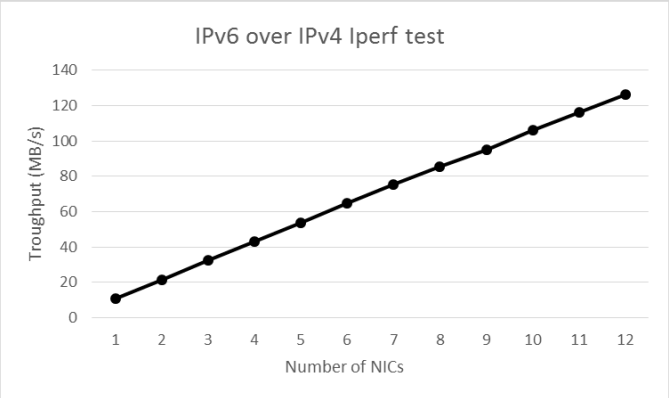

Figure 5. The throughput results of the iperf test of an IPv6 tunnel over IPv4

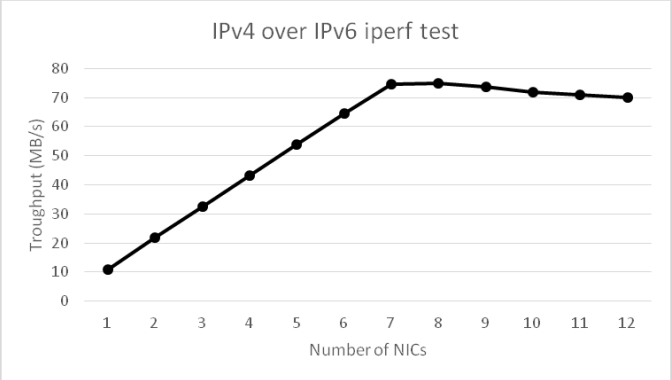

Figure 6. The throughput results of the iperf test of an IPv4 tunnel over IPv6

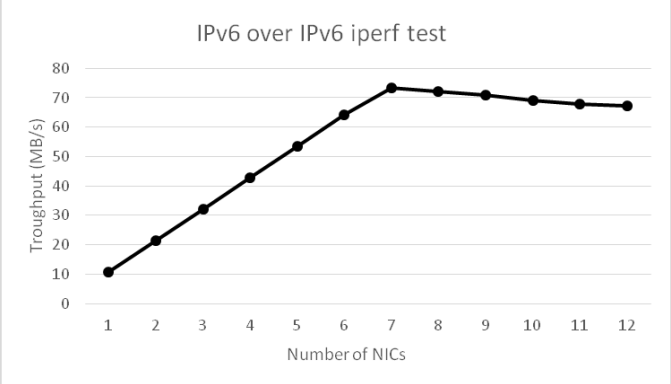

Figure 7. The throughput results of the iperf test of an IPv6 tunnel over IPv6

C. *Performance of the IPv4 tunnel over IPv6*

The throughput on Fig. 6 reached its maximum value at 7 NICs, it could not increase for 8 NICs and it showed somewhat degradation for higher number of NICs.

D. *Performance of the IPv6 tunnel over IPv6*

Our next scenario was the IPv6 tunnel over IPv6 paths. Fig. 7 shows the throughput results. Here, the throughput has a definitive maximum at 7 NICs and shows a bit more degradation than in the previous case.

E. *Evaluation of the results*

Our results suggest that only the version of the underlying IP protocol makes a significant difference in the channel capacity aggregation performance of the MPT library and the version of the encapsulated IP has only a minor influence on it. When the underlying protocol was IPv4, the throughput was linear up to 12 NICs, and thus we could not reach the limits of MPT (see Fig. 4 and Fig. 5). When the underlying protocol was IPv6, we reached the performance limit of the system at 7 NICs. The further increase of the number of NICs could not result in the increase of the throughput, rather some degradation of the throughput can be observed.

At this point, we may only state that this is the performance of our system composed of the above described hardware and software. But we are interested in the limits of the MPT library and not that of the hardware used for testing. We have checked and logged the CPU utilization of the MPT software during the measurements. We did so on both the client and on the server during all the 4 series of measurements thus we got 2x4=8 graphs. The CPU usage of the MPT client and of the MPT server was practically the same. The version of the upper IP protocol made no significant difference. Therefore we include only two typical ones of them. Fig. 8 shows the CPU utilization of the MPT client during the IPv4 over IPv4 measurements. The gaps with 0% CPU usage can be well observed between the measurements thus the 12 measurements can be easily identified. Even though CPU utilization shows some fluctuations, its near linear growth can be observed. It reached 160-180% at 12 NICs. Note that the CPU utilization of the `iperf` program was always under 50% thus there was free CPU capacity available from the 400% of the four CPU cores.

Fig. 9 shows the CPU utilization of the MPT client computer in the IPv6 over IPv6 measurements. The CPU utilization reached 160-180% at 7 NICs and it did not grow any more. There is a visible correspondence between the CPU utilization and the throughput, see Fig. 7.

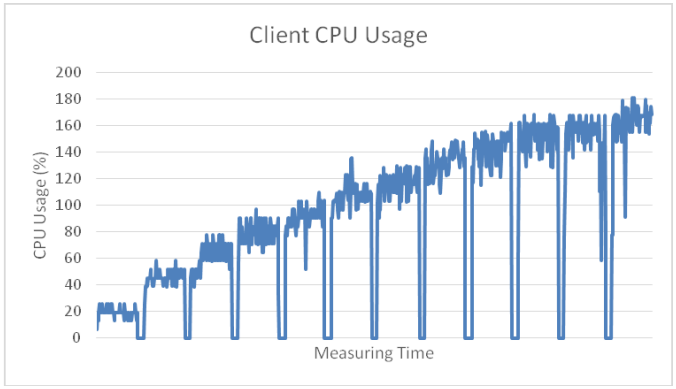

Figure 8. MPT CPU utilization (out of 400%), IPv4 tunnel over IPv4

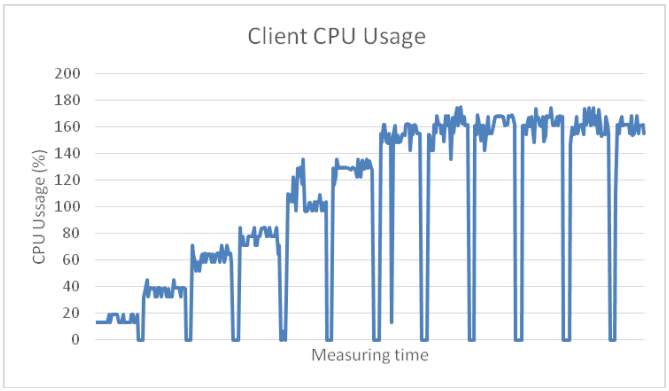

Figure 9. MPT CPU utilization (out of 400%), IPv6 tunnel over IPv6

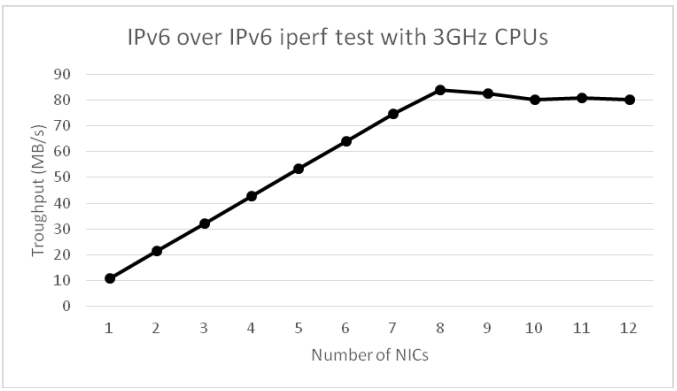

Figure 10. The throughput results of the iperf test of an IPv6 tunnel over IPv6 using 3GHz CPUs

F. *Measurements with faster CPUs*

The Intel Xeon 5140 2.33GHz dual core processors of the test computers were replaced by Intel Xeon 5160 3GHz dual core processors. The throughput of the IPv6 tunnel over IPv6 paths scenario was measured and Fig. 10 shows the results. The faster CPUs made it possible to fully utilize the capacity of 8 NICs and the degradation started from 9 NICs. This is an important result because it convinced us that the aggregation capability of MPT does not have a built in limit, rather it depends on the performance of the CPUs.

It is another issue that MPT was written as a serial program and thus it is not able to fully utilize the available processing power of the multiple CPU cores. As for the current trend of the evolution of the CPUs, it would be desirable to improve MPT in this field.

G. *Measurements with Gigabit Ethernet*

We could not reach the throughput capacity limit of the system in the two tests where the underlying protocol was IPv4. As our Dell computers had only 3 PCI Express slots, we could not insert more NICs. Therefore we removed the Cisco switch and interconnected the 12 Ethernet ports of the two computers directly, thus they were enabled to operate in gigabit mode. (The original 2.33GHz CPUs were kept.)

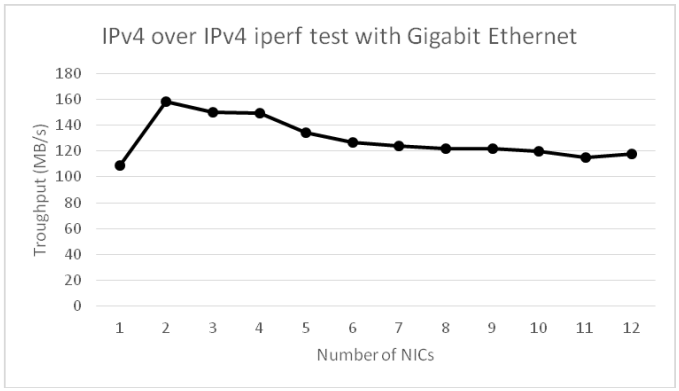

Figure 11. The throughput results of the iperf test of an IPv4 tunnel over IPv4 using Gigabit Ethernet

The results of the IPv4 over IPv4 tests are shown in Fig. 11. The throughput reached 160Mbytes/s at two NICs and it degraded for higher number of NICs, but it remained still higher than the throughput of a single NIC. This is in a correspondence with the values of the CPU utilization in Fig. 12.

The results of the IPv6 over IPv4 tests are shown in Fig. 13. The throughput reached its maximum value at two NICs again, (it is now less than 160Mbytes/s) and it degraded for higher number of NICs, but it remained still higher than the throughput of a single NIC. The CPU utilization graph is not included because it is undistinguishable from the one shown in Fig. 12.

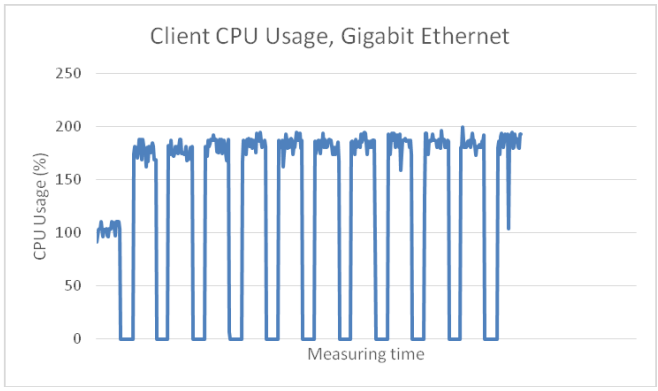

Figure 12. MPT CPU utilization (out of 400%), IPv4 tunnel over IPv4, Gigabit

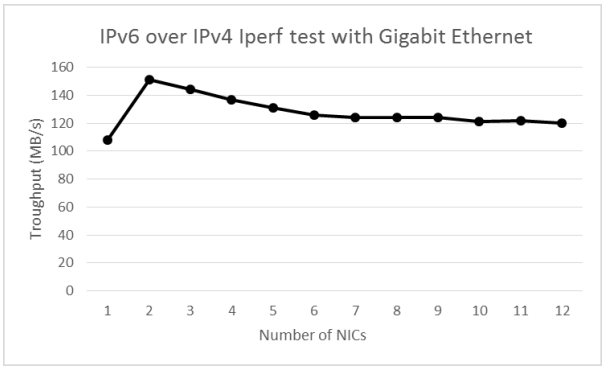

Figure 13. The throughput results of the iperf test of an IPv6 tunnel over IPv4 using Gigabit Ethernet

V. DIRECTIONS OF FURTHER RESEARCH AND RECOMMENDATIONS FOR FURTHER DEVELOPMENT OF MPT

Instead of the precompiled 32-bit version of the MPT library, we are intending to compile and test a 64-bit version. This may include the IPv6 performance of the MPT library by the more efficient handling of the 128 bits long IPv6 addresses.

Even though `iperf` is considered to be industrial standard, we plan to test the performance of the MPT library with some other tools, such as downloading files over http or ftp protocols using `wget`.

We also plan to compare the performance and throughput aggregation capability of the MPT library with that of the standard MPTCP.

As unlike MPTCP, MPT uses UDP, therefore it is also worth testing MPT with real time applications.

As for the further development of MPT, we have two recommendations:

- Using keyword parsing in the configuration files (instead of the current strict syntax) would make the MPT library more user-friendly.
- Enabling MPT to fully utilize the computing power of multiple CPU cores would improve its overall performance when using it for the aggregation of several high speed channels in multi-core environments.

VI. CONCLUSIONS

We have tested the throughput aggregation capability of the MPT multipath communication library up to twelve 100Mbps link layer network connections with all the possible combinations of IPv4 and IPv6 as the underlying or the top protocols.

When the underlying protocol was IPv4, the throughput scaled up linearly up to 12 NICs regardless of the version of the encapsulated IP (IPv4 or IPv6). It exceeded 120Mbytes/s for 12 NICs.

When the underlying protocol was IPv6, the throughput scaled up linearly up to 7 NICs regardless of the version of the encapsulated IP, but there the throughput reached its performance plateau (with a value higher than 70Mbytes/s) and it showed somewhat degradation for higher number of NICs.

We have shown that the above performance limit depends on the computing power of the CPU and it is not a fixed built in feature of the MPT library.

With the help of 12 Gigabit Ethernet connections, we have also shown that the behavior of the system is similar also in the case when IPv4 is applied as the underlying protocol: the system reached its performance plateau at two NICs (its value was about 160Mbytes/s) and then the throughput showed somewhat degradation for higher number of NICs.

We conclude that the MPT multipath communication library is a good tool for the aggregation of the capacity of several channels.

We have given the directions of our future research and two recommendations for the further development of the MPT library.

REFERENCES

[1] B. Almási, A. Harman, “An overview of the multipath communication technologies”, Proceedings of the Conference on Advances in Wireless Sensor Networks 2013 (AWSN 2013), Debrecen University Press, Debrecen, Hungary, ISBN: 978-963-318-356-4, 2013, pp. 7-11.

[2] B. Almási, “MPT Library User Guide”, can be downloaed from: http://irh.inf.unideb.hu/user/almasi/mpt/

[3] B. Almási, Sz. Szilágyi, "Throughput Performance Analysis of the Multipath Communication Library MPT", Proceedings of the 36th International Conference on Telecommunications and Signal Processing (TSP 2013), (Rome, Italy, July 2-4, 2013), pp 86-90.

[4] B. Almási, Sz. Szilágyi, “Multipath ftp and stream transmission analysis using the MPT software environment”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 11, (November 2013) pp. 4267-4272.

[5] B. Almási, “Multipath Communication – a new basis for the Future Internet Cognitive Infocommunication”, Proceedings of the CogInfoCom 2013 Conference, (Budapest, Hungary, December 2-5, 2013), pp. 201-204.

[6] B. Almási, “A simple solution for wireless network layer roaming problems”, Carpathian Journal of Electronic and Computer Engineering Vol. 5, No. 1, (2012) pp. 5-8.

[7] A. Ford, C. Raiciu, M. Handley and O. Bonaventure, “TCP Extensions for Multipath Operation with Multiple Addresses” IETF, January 2013, ISSN: 2070-1721, RFC 6824.

[8] MPT source code, http://irh.inf.unideb.hu/user/almasi/mpt/