Investigating the Multipath Extension of the GRE in UDP Technology

Béla Almási[a], Gábor Lencse[b,*], Szabolcs Szilágyi[a]

[a] *Department of Informatics Systems and Networks, University of Debrecen, Debrecen, Kassai u. 26, Hungary*
[b] *Department of Networked Systems and Services, Budapest University of Technology and Economics, Budapest, Magyar tudósok körútja 2, Hungary*

Abstract

MPT is a novel multipath technology based on the GRE in UDP tunnel specification. In this paper, the conceptual architecture of MPT is disclosed. The designed structure of MPT includes several useful components: the possibility for external software modules to change the run-time parameters (by using the control interface), the choice between flow-based and packet based mappings of tunnel traffic to paths, as well as optional guarantee of in-order packet delivery to the application. The throughput aggregation performance of MPT is investigated in both laboratory and production network environment. The production network measurements proved that MPT is capable for super-aggregation (i.e. the aggregated throughput is higher than the algebraic sum of the throughput of the paths).

Keywords: GRE in UDP, multipath communication, super-aggregation, tunnel

1. Introduction

Multipath solutions have been intensively researched for at least a decade [1, 2, 3]. There are several motivations behind it: many of our common IT devices (e.g. smartphones, tablets, notebooks, etc.) has multiple interfaces (Ethernet, Wi-Fi, 3G, 4G), however, the design of the TCP/IP protocol stack allows only one of them to be used for a given communication session (as a communication session is identified by a five 5-tuple: source IP address, source port number, destination IP address, destination port number and the protocol number denoting TCP or UDP). The aggregation of the transmission capacity of all the interfaces could increase the achieved throughput experienced by the user. Also the automatic changing of the interfaces could improve the quality of experience when one of them becomes overloaded or unable to operate. There were different solutions proposed ranging from Link Layer [3] to Transport Layer [1].

In this paper, we propose MPT, as a Network Layer multipath solution. Its conceptual architecture is unique and distinguishes it from every other existing multipath solutions opening up several areas of applications such as vertical handover without packet loss [4], RTP based video streaming using different network technologies for redundancy purposes [5], efficient throughput aggregation of several paths (tested up to twelve 100Mbps paths in [6]) and also super-aggregation, which means that the aggregated throughput is higher than the sum of the individual throughput of the paths (see later).

*Corresponding author
Email addresses: almasi.bela@inf.unideb.hu (Béla Almási), lencse@hit.bme.hu (Gábor Lencse), szilagyi.szabolcs@inf.unideb.hu (Szabolcs Szilágyi)

This paper is organized as follows. First, the conceptual architecture of MPT is introduced in chapter two. Second, it is examined how MPT relates to other multipath technologies. Then, some high-level design details of MPT are disclosed in chapter four. Finally, both laboratory and production network performance measurements are presented including the demonstration of the super-aggregation capability of MPT.

2. The conceptual architecture of MPT

The MPT architecture targets a Network Layer multipath communication technology. It is based on the "GRE in UDP Encapsulation", (see Fig. 1), which is currently a "Work in progress" state RFC Draft document [7].

The single path GRE in UDP specification distinguishes the IP address used for the identification of the Application (i.e. the Socket ID) and the identification of the physical

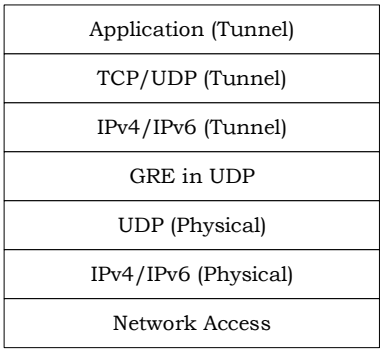

Figure 1: The structure of GRE in UDP

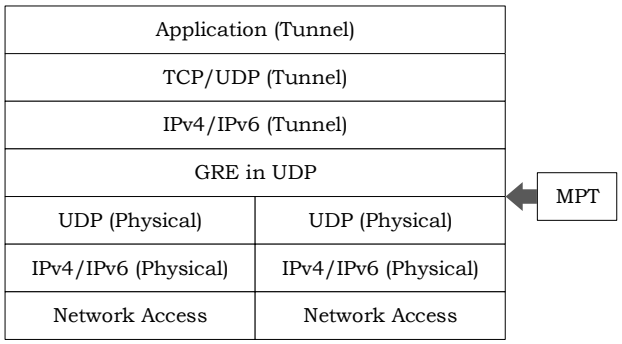

Figure 2: The MPT-GRE conceptual architecture

interface by introducing a tunnel interface: the IP address of the tunnel interface is used by the Application (Tunnel) which is different from the IP address of the physical interface. Even the version number of the Tunnel IP and the Physical Interface IP can be different: IPv4/IPv6 (Tunnel) is independent of the IPv4/IPv6 (Physical). The MPT architecture follows the idea of the GRE in UDP technology but allows the usage of more than one physical interfaces for the tunnel communication (see Fig. 2): below the GRE sublayer, multiple UDP/IP instances can be specified by using multiple interfaces. The usage of multiple physical interfaces enables us to establish a Network Layer multipath communication infrastructure. The mapping (or binding) between the tunnel interface and the physical interfaces (i.e. paths) is controlled by the MPT software component. The purpose of the Network Layer multipath communication is the same as other multipath technologies ([8, 9]): aggregating the capacities of the paths and/or giving redundancy/backup path possibility for the communication. The PDU travelling through the network follows exactly the PDU specification of [7], so using only one path in the MPT architecture will result in the traditional GRE in UDP technology (i.e. the tunnel interface will be mapped statically to a physical interface).

Two important questions of the MPT operation are the "congestion control" on the different paths and the distribution of the packets of the communication among the available paths. The traditional tunneling technology does not require the congestion control below the tunnel interface (as we may assume the presence of congestion control above the tunnel): "a tunnel carrying IP-based traffic should already interact appropriately with other traffic sharing the path, and specific congestion control mechanisms for the tunnel are not necessary" [10]. Using multipath communication below the tunnel interface will a little bit modify this feature: the different paths may have different congestion related properties, so it may produce situations where the congestion cannot be appropriately managed above the tunnel interface, thus the control of congestion below the tunnel interface may be needed. The MPT path selection mechanism can be managed through a control interface (see Fig. 3) using a special control protocol which results in an open framework. In this paper, we do not investigate the questions and solutions for the multipath congestion control problems, but we would like to emphasize that the described architecture is open to work with such kind of solutions: the congestion control mechanism can use the control interface of the MPT software library to influence and modify the properties and parameters of the paths. For example, a congested path can be turned off; or it can be turned on, when becomes uncongested again. Similarly, the traffic distribution between the paths can be modified at any time according to the actual state of the network infrastructure by using the control protocol. Although MPT is not a Software Defined Networking technology, its conceptual working mechanism is quite similar to that. The control interface opens up the possibility of managing the path selection mechanism (i.e. "control plane") of the MPT enabled host by a central software entity, having a much better view on the parameters and state of the network infrastructure.

The packet transmission and receive mechanism of the MPT software component works according to the following description (see Fig. 3). The application software sends an IP (i.e. IPv4 or IPv6) packet to the tunnel interface. Then the packet is read by the MPT software. The MPT software looks for the *connection specification*. The *connection* is the basic unit of the MPT multipath communication technology. The concept of *connection* is similar to the one defined in [8]: The connection will determine the parameters and behavior of the multipath communication (e.g. the distribution of the packets between the paths). In other terms, an MPT *connection* is an UDP tunnel between two MPT entities, which can be used to carry user data. We show an example for the deeper understanding of the concept. Let us consider a host with three interfaces: an Ethernet, a Wi-Fi and an LTE. This host may have several MPT connections to a remote host. For example, one of them uses a path through the Ethernet interface and another path through the Wi-Fi interface and the traffic of the connection is distributed in 3:1 ratio between the two paths. Another connection may have paths over all three interfaces and its traffic can be distributed over the three paths in 3:1:1 ratio. We need to introduce another term: an *MPT related communication session* can be e.g. an FTP download, a video stream, a VoIP call, etc., which uses an MPT connection and not directly the physical interfaces. An MPT related communication session can be mapped to a connection according to the host's policy. We may use multiple connections even in the case when using only a single tunnel interface, so multiple mappings can work in parallel. The identification of a connection can be specified by using some selector fields above the tunnel interface (i.e. selector fields can be chosen from the headers appearing above the "GRE in UDP" sublayer). A typical example of a connection specification is the case when we identify the connection by the pair of IP addresses of the peers' tunnel interfaces. In this case only one connec-

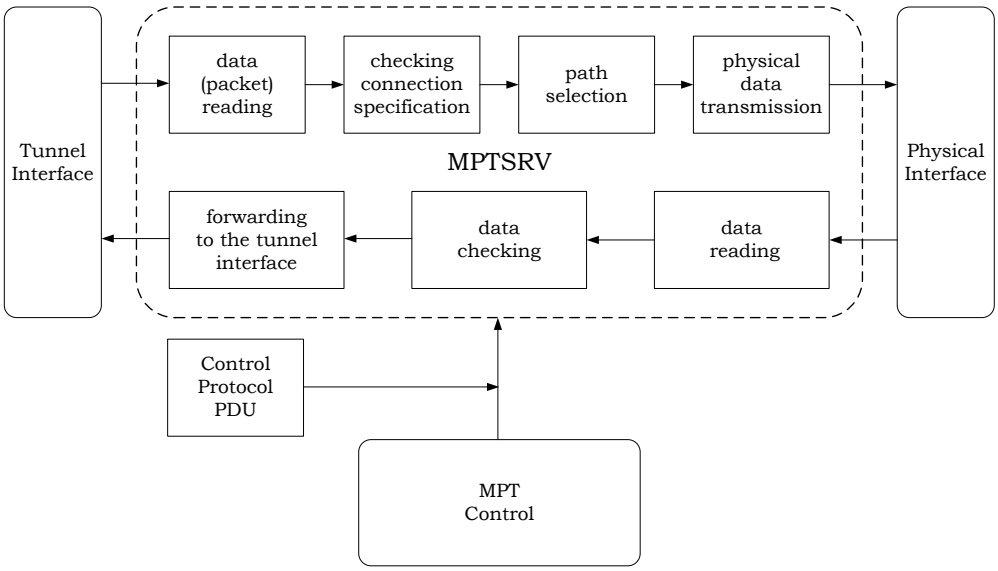

Figure 3: The conceptual MPT working mechanism

tion will be created and all the transmitted packets of the different applications will be mapped to the physical interfaces according to the same connection specification (*per packet based mapping*). Another example of the connection specification can be considered when identifying the connection with the four tuple of IP addresses and port numbers. In this case each communication session will be mapped to the available paths by using different distributions (i.e. performing a *flow based mapping*). In both cases the connection parameters can be defined statically, or can be changed during the operation by using special commands of the control protocol.

A packet (coming from the tunnel interface) will be transmitted to the outgoing physical interface according to the connection specification and state. The outgoing path will be selected according to the actual state of the connection by strictly following the connection description (i.e. the distribution of the packets between the available paths). The path selection is based on the actual values of the connection's counters. If the outgoing path is identified, the packet will be encapsulated into a "GRE in UDP" segment and will be sent out on the physical interface of the chosen path. The fields of the GRE header will be set according to the specification of [7]. The simplest GRE header contains 4 octets: 16 bits of zeros and 16 bits of protocol type identification value (i.e. 0x86DD in the case of using IPv6 on the tunnel interface, or 0x0800 in the case of IPv4).

The packet receiving mechanism is a little bit simpler. When a packet arrives to the physical interface, the destination UDP port number must be the special "GRE-in-UDP" port number value (it is not yet assigned by IANA, as written in [7]). Then the packet will be read by the MPT software. The MPT software runs some checking processes (e.g. connection validity check, GRE sequence number check or GRE *Key* value check, if present). If all the checking mechanisms finish successfully then the packet is transmitted to the Transport and Application Layers through the tunnel interface.

3. Relationship to other multipath solutions

3.1. Multipath TCP

Multipath TCP is defined by an "experimental" state RFC [8] and it is still actively developed by the Multipath TCP Working Group of IETF [11]. MPTCP uses multiple TCP sub-flows on the top of potentially disjoint paths, see Fig. 4. It aims to provide the users both higher throughput and improved resilience to network failures. It was successfully tested in both areas. In an experiment, a single data-stream was transmitted at the rate of 50Gbps over six 10Gbps Ethernet Links using MPTCP [12]. The demonstration of the failover capability of MPTCP was also reported in [13]. Furthermore, MPTCP was also successfully applied for enabling transparent handover between Wi-Fi and 3G (in both directions) [14]. In addition to its practical application, MPTCP is also an important research topic, see e.g. [15].

The most important difference between MPT and MPTCP is that MPT operates in the *Network Layer*, whereas MPTCP is a *Transport Layer* multipath solution. MPTCP is a suitable solution for those applications that use the TCP protocol in the Transport Layer for a reliable byte stream transmission (e.g. it is perfect for downloading web pages, sending and downloading e-mails, etc.). However, there are applications that rather use UDP in the Transport Layer either because they do not need TCP (e.g.

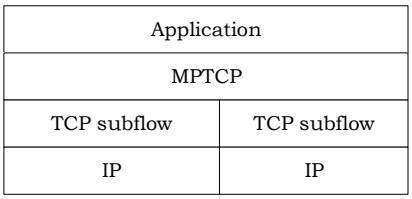

Figure 4: The architecture of the MPTCP protocol stack

short DNS queries and answers) or because the retransmission mechanism of TCP would be expressly harmful for them (e.g. real-time applications such as IP telephony or videoconferencing, which can better tolerate a low rate packet loss than the long delays and high jitter caused by TCP retransmissions). MPT can be used with both classes of applications (preferring TCP or UDP) therefore, it can be considered as another type of solution (targeting other areas of application) than MPTCP.

3.2. Mobile IPv6 and its extensions

Mobile IPv6 (originally defined in [16], and its current definition can be found in [17]) made it possible for a node to remain reachable while wandering around the Internet. This is achieved by the dynamic update of the binding between its *home address* (assigned to the node from its home network) and its *care-of address* (assigned to the node from the visited network). Even though a node may listen to multiple care-of addresses (during the link change it may receive packets from the previous link while it is already available through the new one), the node has to register its primary care-of address with its *home agent* (a router in its home network) thus mobile IPv6 is not a real multipath solution.

Mobile IPv6 has several extensions. E.g. the *Network mobility (NEMO) basic support protocol* [18] enables networks (connected by mobile routers) to move over the Internet. The *Flow binding* extension [19] made it possible to assign different *flows* (identified by selector fields) to different care-of addresses. This is a real Network Layer multipath solution and can be considered as the special case of MPT, as MPT can be also configured to use *flow based mapping*. By using per flow path selection, one may achieve for example that the 3G interface of a mobile device is used for VoIP communication while (at the same time) the Wi-Fi interface is used for file downloading. However, a single application cannot utilize the transmission capacity of both interfaces because it cannot provide two types of packet selectors for its packets. The aggregation of the transmission capacity of multiple interfaces is a simple task for MPT with *per packet based mapping*.

Proxy mobile IPv6 [20] can be considered an extension (or a variation) of Mobile IPv6. In this solution, the *proxy mobility agent* performs the mobility related signaling on behalf of the mobile node, thus the protocol stack of the mobile node does not have to be able for it. A *multi homed mobile node* may connect to the same proxy mobile IPv6 domain through more than one interface and use these interfaces simultaneously. An update to proxy mobile IPv6 [21] specifies protocol extensions to PMIPv6 to distribute specific traffic flows on different physical interfaces. (Thus it is the flow based mapping special case of MPT again.)

3.3. Other similar solutions

3.3.1. Multiple interfaces and provisioning domains

The MIF WG of IETF focuses on the situation when a host has multiple interfaces and they are connected to different provisioning domains [9]. However, the situation is considered rather a "problem" to solve (e.g. how to *choose* between the interfaces) then a "possibility" to exploit. They do not mention something like the distribution of the traffic of the applications over the multiple interfaces, therefore we do not consider it a multipath solution.

3.3.2. Openflow link-layer multipath switching

OLiMPS [22] is a novel solution, which uses the logic of the Link Layer, that is, it calculates routes as if the nodes were connected with LANs. It uses the OpenFlow [23] technology and it was designed for the Large Hadron Collider Open Network Environment (LHCONE). OLiMPS is a multipath solution and it enables *per flow* load distribution. Ref. [22] states that multipath can be achieved in several ways, and its authors used MPTCP from among them in their latest publications [24] and [25].

3.3.3. The TRILL protocol of routing bridges

Routing bridges aimed to combine the advantages of routers and bridges [26]. They use TRILL (Transparent Interconnection of Lots of Links) protocol, which supports multipath forwarding. However, it is "designed to be a Local Area Network protocol and not designed with the goal of scaling beyond the size of existing bridged LANs" [26].

4. MPT implementation - proof of the concept

In this chapter, we describe the most important design features of our implementation of the discussed multipath GRE in UDP concept. This implementation must be considered only as a prototype, which was created to "prove the concept". The software development was performed by the Multipath Communication Research Group in the Faculty of Informatics at the University of Debrecen. The currently available MPT-GRE implementation [27] is a continuation of the earlier started MPT multipath library development. The research studies on the original MPT library started in 2013 (see [28], [29]). The conceptual structure of the MPT library in 2013 was quite similar to the current architecture of the MPT-GRE system. The most important difference is the lack of the GRE in UDP sublayer in the old version (as it was developed as a standard IP in UDP tunneling technology). We included

the GRE in UDP sublayer into the model, thus the GRE in UDP header was included into the new PDU (according to [7]). This modification makes it possible to build a new service feature into the MPT environment: the *Sequence Number* field of the GRE header can be used to guarantee the in order delivery of the packets to the receiver's tunnel interface. This feature can be very important, if the used paths have quite different parameters (especially delay values). Out of order packet arrival will certainly occur if the delay of the paths are different (even in the case, when the difference is not too big), and this may cause serious TCP performance problems. By using the GRE Sequence Numbers we have the possibility of reordering the incoming packets at the receiver side, according to their sending-order.

4.1. Design of the Control Interface

We have implemented the most important control interface functions into the MPT-GRE library. These commands can be used to create an MPT *connection* (that is a tunnel) between the MPT entities of two hosts, to add or delete *paths* (that is underlying physical connections) to or from a connection, etc.

We note that the same control interface is used for the local administration of the MPT entity (accessing a UDP port through the loopback interface) and for the communication of the MPT client with the MPT server (through UDP communication).

In what follows, we give a short overview of the currently available control interface commands:

`mpt path up/down` - this command can be used to "turn on" or "off" a specified path. If the path status is changed to "down", then it is not used by the connection, (i.e. no data is sent through that path by the MPT software).

`mpt interface up/down` - all the paths, that are based on the given local physical interface are "turned on" or "off" by starting the `mpt path up/down` command for each considered path.

`mpt connection create` - This command can be used to establish a "Multipath GRE in UDP Service Provider". The MPT server of the Service Provider can be started without active connections, and the clients of the Service Provider are able to establish the multipath GRE in UDP tunnel connection to the Provider's MPT server dynamically, "as needed". The whole configuration is sent from the client to the server and the server creates a new connection using the received parameters.

`mpt keepalive` - The keepalive message is used to check the availability of the paths regularly. Each path has got two timing configuration parameters: the KEEPALIVE_TIME parameter sets the frequency of the keepalive messages to send. The DEAD_TIME parameter is the amount of time that makes sure of receiving keepalive message from the peer of the path. If no keepalive message arrives in a path for DEAD_TIME seconds then the path is considered as "dead", and the `mpt path down` command is started for the given path to disable the usage of the path immediately. The keepalive mechanism can be turned off by using zero timer values in the configuration. The purpose of the keepalive mechanism can be illustrated by the following example: If an unexpected routing problem occurs, we do not have to wait for the convergence of the routing mechanism (which can take a long time is some special cases) to continue the communication session: the multipath communication library will disable the unavailable path immediately when the routing breakdown is sensed. Thus, the communication will continue to work through the remaining paths of the connection. Similarly, if a keepalive message arrives on a path, which status is "down", the `path up` command is started to enable the usage of the path in the communication. It means, if the routing mechanism reached the converged state, and the previously unavailable path starts to work again, the path will be automatically "turned on" when the reception of the keepalive message proves its usability.

`mpt reload` - this command can be used to reload the configuration parameters.

The control interface communication is based on UDP. In our current solution it uses a different port number than the one reserved for the GRE in UDP data transmission. The reason of this decision is that this solution ensured the separation of the data communication and the control communication. A control communication may take a long time (e.g. interface turning up and getting IP address for the interface may take 10-15 seconds in some situation). This long operation of the control communication could also block the data communication if we used the same port number. As for security, IPsec policy can be applied easily for the control interface communication, but - as the authentication seems to be the most important - local authentication solution can be applied, too. The control message communication follows a 4-way-handshake to perform the necessary information exchange between the peers for the requested functionality. The only exception is the case of sending keepalive messages: it contains only one single way transmission (repeated regularly, as specified by the KEEPALIVE_TIME parameter).

We note that the control interface was designed to be expandable with new commands, thus new control paradigms can be integrated into MPT. The current commands can be found in the user manual of MPT available with the MPT-GRE library [27].

4.2. Mapping of the tunnel traffic to paths

At the sender side, we have to map the tunnel traffic to the paths of the connection. In the current implementation, we use the IP address pairs of the connected tunnel interfaces as the ID of the connection. This solution will perform a *per-packet* based mapping. If we would like to implement a *flow based* solution we would have to include the port number of the application into the connection ID.

The *per-packet based mapping* raises further questions concerning the distribution of the packets among the paths. In the current MPT software, we use a configuration parameter WEIGHT_OUT for each path, which will determine the distribution of the packets. For example, if we have three paths (path_1, path_2 and path_3) with weights of them set to 12, 15 and 10 then the sending cycle consist of 37 packets (it is the sum of the weights, in the case of normalized weights, which means that the greatest common divisor of the weights is 1). In each sending cycle, 12 packets will be transmitted through the first path, 15 packets will be sent through the second and 10 packets will be transmitted through the third path. In order to approximate the weights at each packet sending, the sequence of the path indices in a sending cycle is the following vector: (2, 1, 3, 2, 1, 2, 3, 1, 2, 3, 1, 2, 2, 3, 1, 2, 1, 3, 2, 1, 2, 3, 1, 2, 3, 2, 1, 2, 3, 1, 2, 3, 1, 2, 1, 2, 3).

The sending vector is calculated by the following algorithm:

Algorithm 1 to calculate_sending_vector:

Require: $W[i](1 \leq i \leq N)$, the vector of the weights of the paths.
(Note: We have N paths with indices $(1, \ldots, N)$

Ensure: $O[j](1 \leq j \leq M)$, the sending vector containing the indices of the paths; where M is the length of the sending cycle.

$lcm :=$ Least Common Multiple for $(W[1], \ldots, W[N])$
$M := 0$
$s[i] := 0$, for all i $(1 \leq i \leq N)$
(Note: $s[i]$ will store the sum of the increments for path i, where the increment is $lcm/W[i]$)
while true do
  $z := min(s[1] + lcm/W[1], \ldots, s[N] + lcm/W[N])$
  $k :=$ The smallest index i, for which $(z = s[i] + lcm/W[i])$ is true
  $M := M + 1$
  $s[k] := z$
  $O[M] := k$
  if $(s[i] = z)$ is true for all i $(1 \leq i \leq N)$ **then**
    return
  end if
end while

The current MPT implementation uses a configuration file based static weighting of the paths, which can be used efficiently, if the throughputs of the paths do not change during the communication: we can set the weights of the paths according to their previously measured throughputs. The control interface opens the way to change the weight of a path in runtime-mode, thus by recalculating the sending vector, the distribution of the packets among the paths will be changed immediately.

We have determined the WEIGHT_OUT values for our tests as follows: the throughput values of the different paths were measured by using iperf3 and the throughput values expressed in Mbps were used as the WEIGHT_OUT values. Their setting may be automated by using a short script that measures the values and writes them into the configuration file. It could also be implemented in MPT as a command, but currently it is out of scope of MPT.

In a more general case, when the parties of a given communication, which we call now sender and receiver, measured their upload and download speed of their network links independently from each other, the weights can be calculated as:

$$WO_{ij} = min(SU_i, RD_j) \quad (1)$$

where: SU_i and RD_j denote the upload speed of the i-th link of the sender and the download speed of the j-th link of the receiver, respectively, and WO_{ij} is the WEIGHT_OUT value of the sender for the path defined by these links.

The weights are scalars, their ratio is what counts. They are to be determined using the same method for all links. If expressing them in Mbps does not give a fine enough resolution then 100kbps or even smaller units should be used.

4.3. Handling packet reordering

The MPT environment offers the usage of a special buffer-array to receive and temporarily store the incoming packets. If we require in order transmission through the tunnel then the packets will be reordered according to their GRE *Sequence Number* before writing them to the tunnel interface. As packets may be lost during the transmission, we may not wait for a sequence number value infinitely. We use a parameter named MAX_BUFFDELAY_MSEC which limits the maximum waiting time (in milliseconds) for a missing sequence number value. We have to be careful when setting this parameter. If the value is too small, then we may incorrectly consider a sequence number as lost, and if it arrives later, we have to drop it to keep on the order-right delivery. If the value is too large, then the packet loss will be determined too late, and thus the communication performance may decrease. Our experience shows that a feasible choice could be: a few times the RTT (Round-Trip Time) of the slowest path.

4.4. User space implementation

Our MPT implementation works fully in user space including both data and control plane. Thus it is kernel version agnostic and it can be compiled under different CPU architectures. Besides x86, it was successfully tested on ARM (both on Raspberry Pi and Android).

5. Initial testing of our MPT implementation

In this chapter, we describe the results of five measurements. The first two of them investigate the throughput aggregation capability of the MPT software. In the

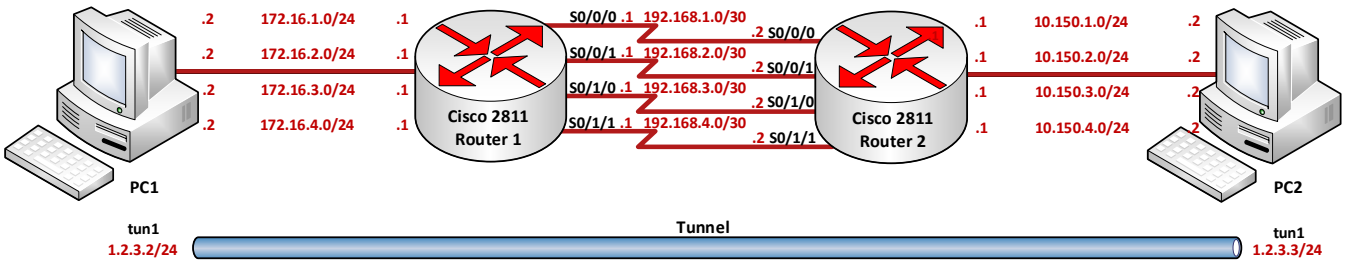

Figure 5: Four wired paths laboratory measurement environment

first case, we used a test laboratory environment containing four wired paths. As only the measurement traffic was present in the laboratory, the delay was almost constant and homogeneous in the system. In the second case, we used a real-life business network environment (going through the Internet Cloud of the University of Debrecen) to aggregate the throughput of a wired (Ethernet) and a wireless (Wi-Fi) link. The third measurement investigates the usefulness of the GRE Sequence Number, which facilitates in-order delivery of the received packets to the tunnel interface. The last two measurements were performed by some of our colleagues and we only cite their results concerning the comparison of the path aggregation capacity of MPT with that of MPTCP and the testing of the switchover speed of MPT.

5.1. Test laboratory environment with 4 wired paths

We built a test-laboratory environment in order to analyze the throughput aggregation efficiency of the GRE in UDP based multipath communication library. It contains four partially disjoint paths (see Fig. 5). The measurement method is the same as it was in [29]: one file (20MB) was transferred between the hosts, using the built in ftp software of the Ubuntu Linux operating system. But in this case we used the new, GRE based MPT software library instead of the old, UDP based tunnel solution that was used in [29]. The PCs (Intel Core i5, 2500MHz CPU, 8GB RAM, 1TB HDD) are connected to the routers with 100Mbps wired connections. Their two Fast Ethernet links are common in the four paths. The bottleneck point of the communication was set to the four different serial links between the routers: the speed of these links were set to 1Mbps or 2Mbps, so ensuring that the shared Fast Ethernet links will not limit or influence the throughput aggregation measurement. Taking into consideration the symmetry of the system, we can distinguish five different measurement cases, depending on the number of the included 2Mbps serial links: (1-1-1-1), (2-1-1-1), (2-2-1-1), (2-2-2-1), (2-2-2-2). We have performed 10 measurements for each case (i.e. 50 measurements in all), to see the stability of the measurements and focusing to the variances of the measured values. In the MPT configuration, we

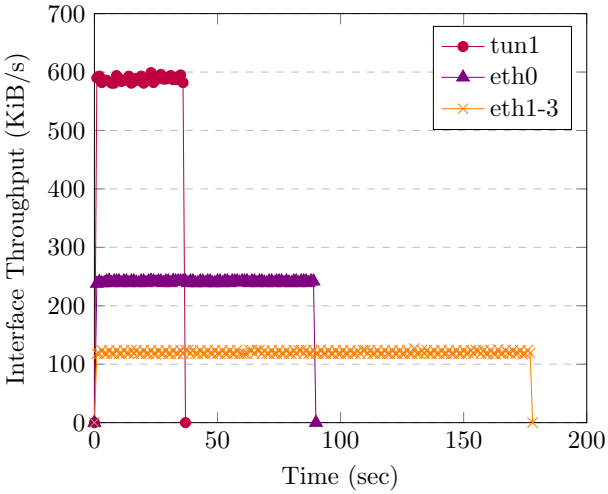

Figure 6: 20MB file transfer with clock rates 2M/1M/1M/1M

used a per packet based distribution between the paths, setting the weights of the paths according to the speed of the serial link in the path. The measurements showed very good stability and very small variances (less than 5%): the differences of the measured values in the same case study were almost not noticeable. As it could be expected, the measurements results are the same as they were in [29]. The efficiency of the throughput aggregation of the GRE in UDP multipath communication environment is better than 95 percent in each case. The detailed results for the cases of (2-1-1-1) and (2-2-2-1) can be seen in Fig. 6 and Fig. 7.

5.2. Super aggregation of two paths in a production network environment

In this evaluation, we performed measurements using the wired and the wireless network of the University of Debrecen. The structure of the environment can be seen in Fig. 8. The "Server" node was a fast PC (Intel Core i7, 3GHz, 8GB RAM, 500GB HDD, two pieces of 1000Mbps network interface cards for network connection). The "Client" node was a Lenovo T420 notebook (Intel Core i5, 2.5GHz, 8GB RAM, 500GB SSD, one 1000Mbps RJ-45

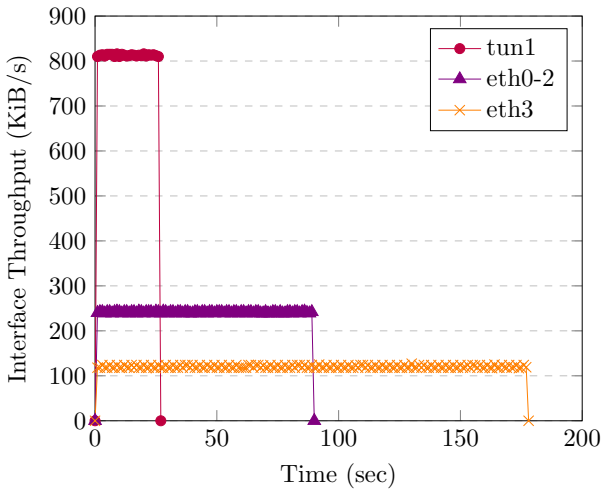

Figure 7: 20MB file transfer with clock rates 2M/2M/2M/1M

interface and one 802.11n wireless interface). The Ubuntu Linux 14.04.3 LTS was installed on both computers, using the Westwood TCP congestion control implementation. The client was connected to the wired network infrastructure by a 100Mbps Ethernet link.

For the aggregation performance test, we used the "super-aggregation" idea, introduced by Cheng-Lin Tsao *et al.* in [30]: the download throughput of a wireless multipath environment can be greater than the algebraic sum of the paths' download throughput if the TCP ACK messages are drawn off from the Wi-Fi communication. The MPT environment can be configured asymmetrically, so we could simply create a configuration, which uses both paths for data downloading and uses only the Ethernet wired path to send the ACK messages to the server, thus producing a lossless, reliable ACK transmit. Removing the ACK messages from the half-duplex Wi-Fi path significantly decreases the number of Wi-Fi media access requests and this spared capacity will be used in the download direction.

We note that the choice of the Westwood TCP congestion control implementation was important because, unlike Reno, Westwood can differentiate between random packet loss and packet loss due to congestion therefore it performs significantly better over wireless links [31].

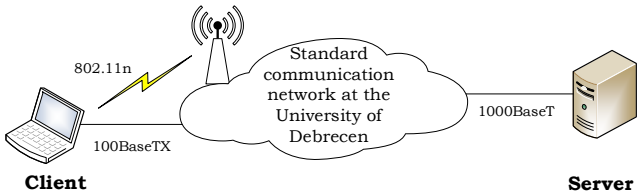

Figure 8: Super-aggregation measurement environment

As the measurements were performed in the business network at the University of Debrecen, connecting the client and the server to different networks (i.e. the measurement traffic competed with the normal users' traffic), we used the iperf3 tool in multithread mode (with 35 threads) to decrease the effect of the other users' traffic on the measurement results. Each measurement round contained three download throughput measurements: first, we tested the throughput from the server to the client by using only the 100Mbps Ethernet wired link of the client. The second measurement used only the Wi-Fi connection of the client, and the third test was used by the MPT environment, combining the two paths. Each test ran for 20 seconds, and 15 seconds break was inserted after each measurement. The configuration files of the measurement environment and the measurement scripts can be found at `http://irh.inf.unideb.hu/user/almasi/mpt/superaggregation`.

The WEIGHT_OUT value of the Wi-Fi path was determined using the before mentioned asymmetric configuration: download traffic was sent through Wi-Fi and the TCP ACK messages were offloaded (they were sent through Ethernet).

We note that the used 2.4 GHz frequency band is rather loaded at the campus, as it is also used by the building security system and there is a sensor network nearby. The measurements were taken at night to decrease variable Wi-Fi traffic coming from human users. In order to check if the results would be similar or not, the measurements were repeated three consecutive nights (October 2, 3 and 4, 2015) and 100 measurement rounds were executed at each night.

The results are shown in Table 1.

Besides the average throughput, we also gave the standard deviation to show how steady or altering the throughput was. The efficiency was calculated as follows:

$$E = \frac{T_{MPT}}{T_{Ethernet} + T_{Wi-Fi}}.100\% \qquad (2)$$

where E, $T_{Ethernet}$, T_{Wi-Fi} and T_{MPT} denote the efficiency, the Ethernet throughput, the Wi-Fi throughput and the MPT throughput, respectively.

The efficiency was between 125% and 130% all three nights. In order to show the reason of the visibly higher standard deviation of the Wi-Fi throughput (and somewhat smaller efficiency) at the second night, all the measured throughput values were plotted in Fig. 9. There it can be clearly seen that the Wi-Fi throughput was significantly higher at the beginning of the second night, but it returned to its "usual" value after that. (The higher Wi-Fi throughput at the beginning made also the average Wi-Fi throughput higher, which resulted finally in somewhat smaller efficiency.) We believe that this is a feature of the real life system and it would not be fair to exclude those measurements. The point is still clear: the super aggregation was successfully demonstrated using the MPT technology.

Table 1: Super-aggregation measurement results (three nights, 100 rounds/night)

	Measured Transfer Speed (Mbps)					
	First Night		Second Night		Third Night	
	Average	Std. Dev.	Average	Std. Dev.	Average	Std. Dev.
Ethernet	94.19	0.33	94.19	0.33	94.20	0.14
Wi-Fi	51.65	2.39	55.28	7.97	52.02	4.59
MPT	189.33	9.20	188.33	9.49	189.56	8.80
Efficiency	129.82 %		125.99 %		129.64 %	

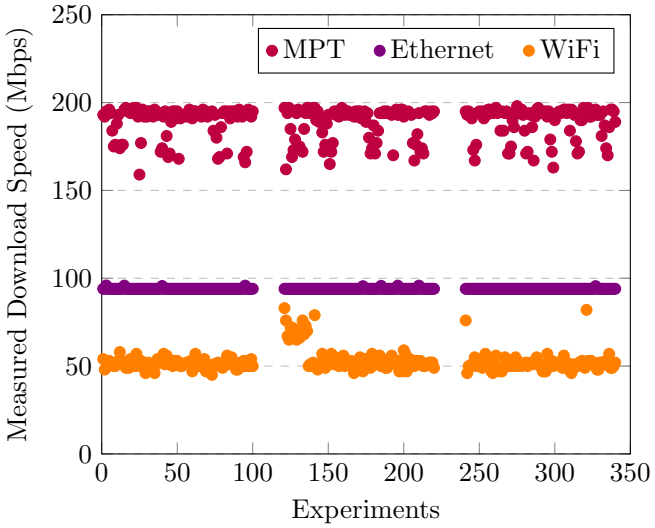

Figure 9: Super-aggregation measurement results (three nights, 100 rounds/night)

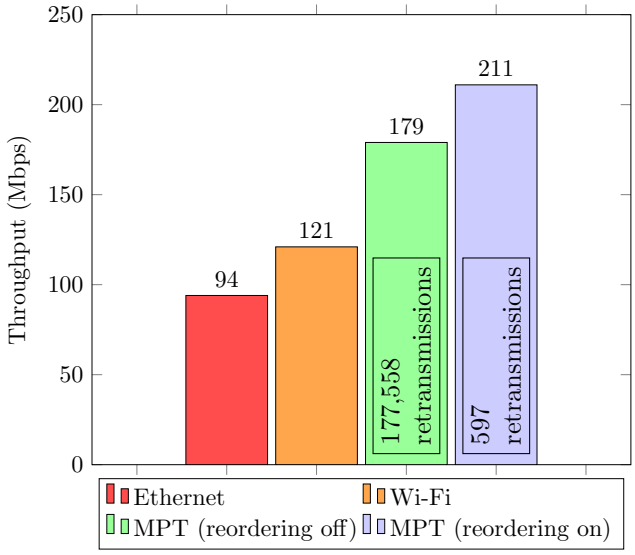

Figure 10: The benefit of packet reordering

5.3. The benefit of GRE packet numbering

We tested the benefit of packet reordering made possible by the optional GRE Sequence Number field. For our measurements, an Ethernet + Wi-Fi throughput aggregation setup was used. On the server side, the outbound packet ratio between the paths was set to 94:121 (Ethernet:Wi-Fi), while it was set to 10000:1 (Ethernet:Wi-Fi) on the client side, thus off-loading the Wi-Fi connection from the client side. With the reordering mechanism disabled, the connection took up to tens of seconds to stabilize, and even then we observed several significant drops in the throughput also resulting in the average throughput measured on the tunnel being lower than the sum of the averages measured on each of the separate paths. The TCP retransmission count was 177,558 packets during a 5 minutes long iperf3 measurement. After enabling the reordering mechanism (even with maximum 5 milliseconds of buffered time) the connection became noticeably more stable, the throughput of the tunnel stabilized (showing the expected aggregation) and TCP retransmission count also dramatically dropped to 597 packets for a 5 minutes long iperf3 measurement. Please refer to Fig. 10 for comparison. These results justify the use of the GRE in UDP technology in MPT.

5.4. Comparison with MPTCP

Ákos Kovács made an initial comparison of MPT and MPTCP at the Széchenyi István University. He compared their path aggregation capabilities in laboratory environment up to twelve 100Mbps Ethernet links. He found that MPT showed somewhat better performance than MPTCP up to 8 links and MPT significantly outperformed MPTCP for higher number of links, because the aggregation capability of MPTCP is limited up to 8 paths [32].

5.5. Testing switchover speed

There is an ongoing research at the University of Debrecen where the experienced throughput of a multipath mobile device is measured. The preliminary results showed that when link failures were caused then the connection recovered really quickly (in 1-2 seconds) and transmission continued on the other path [33].

6. Future work

We view the architecture of MPT as an open protocol which is to be completed with some further parts. In the same way as TCP may work with different congestion control algorithms, MPT may work with different methods

for detecting congestion (or capacity change) on the individual paths and switching the paths down/up or changing their weights. Such algorithms can be easily suited into MPT: the control interface of our current implementation makes it possible to switch paths up/down or change their weights (using the `mpt path up|down` command or changing the WEIGHT_OUT values in the configuration file and using the `mpt reload` command). Thus these algorithms can be developed "outside" of MPT, and may be integrated into it later on after being tested and found to be appropriate. We plan to develop such algorithms and analyze their performances in both closed and open network environments. As Marius Georgescu pointed it out: whereas the isolated network environments (laboratory testbeds) are suitable for a fine-grain performance analysis, the open environments (production networks) are necessary for a better analysis of operational characteristics [34]. Thus we plan to use both of them.

Another important direction of our future research is to find an algorithm for selecting the value of the MAX_BUFFDELAY_MSEC, even for changing it adaptively. We plan to build a testbed and/or a simulation environment to investigate how the different parameters (e.g. the RTT and the transmission speed of the different paths) influence its "optimal" value, and also to assess how the deviations from the "optimal" value influence the performance and quality of the communication.

We also plan to write an Internet Draft (future RFC) about MPT.

7. Conclusion

We have given a survey of multipath solutions, and showed that MPT significantly differs from all of them by extending the GRE in UDP specification to be capable for using multiple paths. We have disclosed both the conceptual architecture and the design features of MPT.

The control interface of MPT opens up the possibility to change the run-time parameters of the system thus makes possible the prompt reaction to environmental changes (e.g. by switching off/on paths in case of network breakdown/repair).

We have shown that using a special reorganization of the traffic in a wired and wireless multipath environment (this can be easily specified in the MPT configuration) can produce a super-aggregation of the throughput.

We conclude that the multipath GRE in UDP architecture of MPT has the potential to become a widely usable Network Layer multipath solution and therefore it deserves the attention of the networking research community.

Acknowledgements

Szabolcs Szilágyi's work was supported by the ÚNKP-16-4-I New National Excellence Program of the Ministry of Human Capacities.

References

[1] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, D. F. Towsley, Multi-path TCP: A joint congestion control and routing scheme to exploit path diversity in the internet, IEEE/ACM Transactions on Networking 14 (6) (2006) 1260–1271. doi:10.1109/TNET.2006.886738.

[2] M. Tekaya, N. Tabbane, S. Tabbane, Multipath routing mechanism with load balancing in ad hoc network, in: Proc Int. Conf. Computer Engineering and Systems (ICCES), Cairo, Egypt, 2010, pp. 67–72. doi:10.1109/ICCES.2010.5674892.

[3] Y. Yu, S. Fang, K. M. M. Aung, C. H. Foh, H. Li, Y. Zhu, A layer2 multipath solution and its performance evaluation for data center Ethernets, Int. Journal of Commun. Systems 27 (11) (2014) 2555–2576. doi:10.1002/dac.2488.

[4] B. Almási, A solution for changing the communication interfaces between WiFi and 3G without packet loss, in: Proc. 37th Int. Conf. on Telecommunications and Signal Processing (TSP 2014), Berlin, Germany, 2014, pp. 73–77. doi:10.1109/TSP.2015.7296420.

[5] B. Almási, M. Kósa, F. Fejes, R. Katona, L. Pusok, MPT: A solution for eliminating the effect of network breakdowns in case of HD video stream transmission, in: Proc. 6th IEEE Conf. on Cognitive Infocommunications (CogInfoCom 2015), Gyor, Hungary, 2015, pp. 121–126. doi:10.1109/CogInfoCom.2015.7390576.

[6] G. Lencse, Á. Kovács, Advanced measurements of the aggregation capability of the MPT multipath communication library, Int. Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems 4 (2) (2015) 41–48. doi:10.11601/ijates.v4i2.112.

[7] E. Crabbe, L. Yong, X. Xu, T. Herbert, GRE-in-UDP encapsulation, IETF Draft.
URL https://tools.ietf.org/html/draft-ietf-tsvwg-gre-in-udp-encap-19

[8] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, TCP extensions for multipath operation with multiple addresses, IETF RFC 6824 (2013).

[9] M. Blanchet, P. Seite, Multiple interfaces and provisioning domains problem statement, IETF RFC 6418 (2011).

[10] L. Eggert, G. Fairhurst, Unicast UDP usage guidelines for application designers, IETF RFC 5405 (2008).

[11] Multipath TCP working group documents.
URL http://datatracker.ietf.org/wg/mptcp/documents/

[12] C. Paasch, G. Detal, S. Barré, F. Duchene, O. Bonaventure, The fastest TCP connection with multipath TCP.
URL http://multipath-tcp.org/pmwiki.php?n=Main.50Gbps

[13] S. Barré, O. Bonaventure, C. Raiciu, M. Handley, Experimenting with multipath TCP, in: Proc. ACM SIGCOMM 2010 conference, New Delhi, India, 2010, pp. 443–444. doi:10.1145/1851182.1851254.

[14] C. Paasch, G. Detal, F. Duchene, C. Raiciu, O. Bonaventure, Exploring mobile/WiFi handover with multipath TCP, in: Proc. 2012 ACM SIGCOMM workshop on Cellular networks: operations, challenges, and future design, Helsinki, Finland, 2012, pp. 31–36. doi:10.1145/2342468.2342476.

[15] R. Khalili, N. Gast, M. Popovic, J.-Y. L. Boudec, MPTCP is not Pareto-optimal: Performance issues and a possible solution, IEEE/ACM Trans. on Networking 21 (5) (2013) 1651–1665. doi:10.1109/TNET.2013.2274462.

[16] D. Johnson, C. Perkins, J. Arkko, Mobility support in IPv6, IETF RFC 3775 (2004).

[17] C. P. (Ed), D. Johnson, J. Arkko, Mobility support in IPv6, IETF RFC 6275 (2011).

[18] V. Devarapalli, R. Wakikawa, A. Petrescu, P. Thubert, Network mobility (NEMO) basic support protocol, IETF RFC 3963 (2005).

[19] G. Tsirtsis, H. Soliman, N. Montavont, G. Giaretta, K. Kuladinithi, Flow bindings in mobile IPv6 and network mobility (NEMO) basic support, IETF RFC 6089 (2011).

[20] S. G. (Ed), K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, Proxy mobile IPv6, IETF RFC 5213 (2008).
[21] C. B. (Ed.), Proxy mobile IPv6 extensions to support flow mobility, IETF RFC 7864 (2016).
[22] H. Newman, A. Barczyk, M. Bredel, OLiMPS: OpenFlow link-layer multipath switching, in: ASCR Next-Generation Networks for Science (NGNS) Principal Investigators' (PI) Meeting, Rockville, MD, 2014.
URL http://www.orau.gov/ngnspi2014/presentations/newman_h.pdf
[23] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, OpenFlow: Enabling innovation in campus networks, ACM SIGCOMM Computer Communication Review 38 (2) (2008) 69–74. doi:10.1145/1355734.1355746.
[24] R. van der Pol, M. Bredel, A. Barczyk, N. van Adrichem, F. Kuipers, B. Overeinder, Experiences with MPTCP in an intercontinental multipathed OpenFlow network, in: TERENA Networking Conference (TNC 2013), 2013.
URL https://tnc2013.terena.org/getfile/356
[25] M. Bredel, Z. Bozakov, A. Barczyk, H. Newman, Flow-based load balancing in multipathed layer-2 networks using OpenFlow and Multipath-TCP, in: Proc. ACM Third Workshop on Hot Topics in Software Defined Networking (HotSDN 2014), 2014, pp. 213–214. doi:10.1145/2620728.2620770.
[26] R. Perlman, D. E. 3rd, D. Dutt, S. Gai, A. Ghanwani, Routing bridges (RBridges): Base protocol specification, IETF RFC 6325 (2011).
[27] B. Almási et al, MPT MultiPaTh GRE in UDP communication library.
URL http://irh.inf.unideb.hu/user/szilagyi/mpt/
[28] B. Almási, S. Szilágyi, Throughput performance analysis of the multipath communication library MPT, in: Proc. 36th Int. Conf. on Telecommunications and Signal Processing (TSP 2013), Rome, Italy, 2013, pp. 86–90. doi:10.1109/TSP.2013.6613897.
[29] B. Almási, S. Szilágyi, Multipath FTP and stream transmission analysis using the MPT software environment, International Journal of Advanced Research in Computer and Communication Engineering 2 (11) (2013) 4267–4272.
[30] C.-L. Tsao, S. Sanadhya, R. Sivakumar, A super-aggregation strategy for multi-homed mobile hosts with heterogeneous wireless interfaces, Springer Wireless Networks 21 (2) (2015) 639–658. doi:10.1007/s11276-014-0794-y.
[31] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, R. Wang, TCP westwood: Bandwidth estimation for enhanced transport over wireless links, in: Proc. 7th annual international conference on Mobile computing and networking (MobiCom 2001), Rome, Italy, 2001, pp. 287–297. doi:10.1145/381677.381704.
[32] A. Kovács, Comparing the aggregation capability of the MPT communications library and multipath TCP, in: Proc. 7th IEEE Conf. on Cognitive Infocommunications (CogInfoCom 2016), Wroclaw, Poland, 2016, pp. 157–162. doi:10.1109/CogInfoCom.2016.7804542.
[33] F. Fejes, R. Katona, L. Pusok, Multipath strategies and solutions in multihomed mobile environments, in: Proc. 7th IEEE Conf. on Cognitive Infocommunications (CogInfoCom 2016), Wroclaw, Poland, 2016, pp. 79–84. doi:10.1109/CogInfoCom.2016.7804529.
[34] M. Georgescu, IPv6NET: A collection of methodologies for the evaluation of IPv6 transition technologies, Ph.D. thesis, Graduate School of Information Science, Nara Institute of Science and Technology, Japan (2016).
URL https://library.naist.jp/mylimedio/search/av1.do?target=local&bibid=80430&lang=en

About authors

Béla Almási received MSc and PhD in mathematics and computer science from University of Debrecen, Hungary in 1989 and 1996, respectively.

He works as an associate professor at the Department of Informatics Systems and Networks, Faculty of Informatics at the University of Debrecen, Hungary. His research interests include the establishment and performance evaluation of stochastic queueing models for communication systems and investigating the multipath communication technologies. He was the head of the Multipath Communication Research Group at the Faculty of Informatics, University of Debrecen.

Gábor Lencse received MSc and PhD in computer science from the Budapest University of Technology and Economics, Budapest, Hungary in 1994 and 2001, respectively.

He works full time for the Department of Telecommunications, Széchenyi István University, Győr, Hungary since 1997. Now, he is an associate professor. He is also a part time senior research fellow at the Department of Networked Systems and Services, Budapest University of Technology and Economics since 2005. His research interests include the performance analysis of communication systems, parallel discrete event simulation methodology and IPv6 transition methods.

Szabolcs Szilágyi received his MSc in electrical engineering and computer science at the University of Oradea, Romania in 2009, and his PhD at the University of Debrecen, Hungary in 2015.

He has been working for the Department of Informatics Systems and Networks, Faculty of Informatics, University of Debrecen since 2013. Now, he is a senior lecturer. The area of his research includes performance analysis of communication networks and investigating the multipath communication technologies. He is the new head of the Multipath Communication Research Group at the Faculty of Informatics, University of Debrecen.