Comparison of Contemporary Solutions for High Speed Data Transport on WAN 10 Gbit/s Connections

Dmitry Kachan, Eduard Siemens
Department of Electrical, Mechanical and Industrial Engineering
Anhalt University of Applied Sciences
Köthen, Germany
{d.kachan, e.siemens}@emw.hs-anhalt.de

Vyacheslav Shuvalov
Department of Transmission of Discrete Data and Metrology
Siberian State University of Telecommunications and Information Sciences
Novosibirsk, Russia
shvp04@mail.ru

Abstract* – This work compares commercial fast data transport approaches through 10 Gbit/s Wide Area Network (WAN). Common solutions, such as File Transport Protocol (FTP) based on TCP/IP stack, are being increasingly replaced by modern protocols based on more efficient stacks. To assess the capabilities of current applications for fast data transport, the following commercial solutions were investigated: *Velocity* – a data transport application of BitSpeed LLC; *TIXstream* – a data transport application of Tixel GmbH; *FileCatalyst Direct* – a data transport application of Unlimi-Tech Software Inc; *Catapult Server* – a data transport application of XDT PTY LTD; *ExpeDat* – a commercial data transport solution of Data Expedition, Inc. The goal of this work is to test solutions under equal network conditions and thus compare transmission performance of recent proprietary alternatives for FTP/TCP within 10 Gigabit/s networks where there are high latencies and packet loss in WANs. This research focuses on a comparison of approaches using intuitive parameters such as data rate and duration of transmission. The comparison has revealed that of all investigated solutions *TIXstream* achieves maximum link utilization in presence of lightweight impairments. The most stable results were achieved using *FC Direct*. *Velocity* shows the most accurate output on the links without packet losses, and the most accurate output in presence of packet losses is shown by *ExpeDat.

Keywords-high-speed data transport; transport protocol; WAN acceleration, Managed File Transport.

I. INTRODUCTION

The growing demand for the fast exchange of huge amounts of data between distant locations has led to the emergence of many new commercial data transport solutions that promise to transport huge amounts of data many times faster than conventional FTP/TCP solutions. Currently, most common solutions for reliable data transport in IP networks are based on the TCP protocol, which was developed in 1970s. A number of papers describe how TCP, with some tuning, can perform reasonably on Local Area Networks (LAN) with a high available bandwidth [1]. However, it is well known that TCP has a very limited performance when used in long distance networks with a high bandwidth - called "Long Fat Pipe Network (LFN)" [2]. For example, a test with *iperf* using the topology described in Fig 2 on an end-to-end 10 Gbit/s link with a 50 ms round trip time delay (RTT) and in the presence of at least 0.1% packet loss rate shows a data rate of about 40 Mbit/s. Even after increasing socket buffers and windows sizes to 128 MiBytes, the performance of TCP and, accordingly, of most of solutions based on it (SCP, *rsync*, FTP), does not reach more than 60 Mbit/s. Comparable measurements of TCP over 10 Gbit/s were also performed by Wu et al. [1]. In their work, the authors obtained a data rate of less than 1 Gbit/s even in the presence of a loss rate of 0.001% and an RTT of 120 ms. They show a significantly decreasing trend in a data rate with growing packet loss rate. Another example of TCP weaknesses over long distances is described by Armbrust et al. in [3], where the transmission of 10 TBytes of data from Berkeley, California to Seattle, Washington via a common TCP connection takes about 45 days, whereas transmission of 10 TBytes hard drive takes less than one day. A similar solution is described by Armbrust et al. in [4]. Nevertheless, many scenarios of remote collaboration (e.g cloud computing) demand data transport with maximum synchronization times for huge data sets from a few minutes to hours. As a result, many large companies, for which the exchange of huge amounts of data is often critical, avoid using legacy TCP-based transport solutions and either prefer commercial high speed approaches based on both TCP and UDP transport protocols [5] [6] or develop their own solutions based on an open source fast protocol stacks such as UDT [7] and RBUDP [8].

II. RELATED WORK

The main goal of our work is to assess the capabilities of transport solutions in a 10 Gbit/s network. Of interest is the maximal possible end-to-end application data rate on such networks in the presence of impairments such as packet losses and high round-trip times. Currently, there are a few different performance measurements that have been used to assess these impairments in open source and freeware solutions. For example, in [9] Grossman et al. present the performance evaluation of UDT [7] through a 10 Gbit/s network. The article shows how using UDT and in the presence of 116 ms of RTT, this network has a maximum

 ISBN: 978-1-61208-256-1

throughput of 4.5 Gbit/s within a single data stream. Two parallel streams achieve together about 5 Gbit/s and within 8 parallel streams about 6.6 Gbit/s are achieved. Further, a performance result for data transmission using RBUDP was presented at the CineGrid 3rd Annual International Workshop [10]. While the disk access speed limited the data transport speed to 3.5 Gbit/s, on the link from Amsterdam to San Diego only 1.2 Gbit/s was reached. The distance of that path is about 10 000 km through optic fiber, which corresponds to about 100 ms of RTT.

Most other data transport performance results are presented for 1 Gbit/s networks e.g. three rate based transport protocols have been evaluated by Wu et al. in [11]: RBUDP, SABUL and GTP. The overall data rate of applications based on these protocols was compared for all three protocols and for "standard unturned TCP". The experiment was performed on a real network in the presence of 58 ms of RTT and a loss rate of less than 0.1%. The results showed that all solutions utilize the 1 Gbit/s link approximately 90%. These test results show that for open source data transfer solutions, even those using parallel streams, it is quite hard to achieve full, or even close to full, utilization of 10 Gbit/s links.

For commercial closed source solutions, the situation differs significantly. There are several published performance results of commercially available solutions, provided by the manufacturers themselves: *Velocity* [12], *TIXstream* [13], *FC Direct* [14] and *Catapult Server* [15] – all of whom report perfect results. However these results are mainly providing commercial information to attract potential customers and the investigative conditions vary. To overcome this deficit, the main idea behind our work is to place all investigated solutions under equal conditions within the same environment.

III. BACKGROUND

For IP networks, packet loss behavior depends on many factors such as quality of transmission media, CPU performance of intermediate network devices, presence of cross traffic etc. It is therefore impossible to use one universal value of packet losses for all cases. The best way to assess the approximate values of packet losses is through empirical measurements. In [16] V. Paxson discusses the heterogeneity of packet losses and shows that, even in 1994, the value of packet loss rate in experiments between 35 sites in 9 countries was about 2.7%. He also shows that, within one year, the value of packet losses increased up to 5%. Probably, such packet loss values are not relevant to the current Internet; however the author pointed out that distribution of packet losses is not uniform. Thus, for some connections, ACK packet loss was not observed at all. Nevertheless, relative values of all lost IP packets in both directions for all experiments were approximately equal. Recent views on the packet loss ratio are presented by Wang et al. in [17]. In this research, tests were made across 6 continents between 147 countries, involving about 10 000 different paths. The authors show that across all continents for more than 70% of continental connections, packet loss rate is less than 1%, in Europe and North America this value is even on about 90% of connections. The authors also highlighted that for intercontinental connections, packet loss value in general is lower than for intra-continental – across the entire world, packet loss rate is lower than 1% for about 75% of the connections.

In [18] Settlemyer et al. use a hardware emulator to emulate 10 Gbit/s paths, and they compare throughput measurement results of the emulated paths with real ones. The maximal RTT of a real link used in the research is 171 ms. The authors have shown that differences between profiles of both kinds of paths - emulated and real ones - are not significant, and they conclude that using emulated infrastructure is a much less expensive way to generate robust physical throughput.

IV. TESTBED TOPOLOGY DESCRIPTION

In this work, the following solutions have been investigated: *Velocity*, *TIXstream*, *FileCatalyst Direct*, *Catapult Server*, *ExpeDat*. Manufactures of all these solutions claim that their transport solutions are able to handle data transmission via WAN in the most efficient way.

Since all these solutions are commercial and closed source, it was necessary to get in touch with the support team of each manufacturer for both obtaining trial licenses of their products and consulting them about achieved results. Unfortunately, not all manufactures were interested in such investigations. Thus, for example, it would have been interesting to test Aspera's solution for fast data transport. However we received no answer from this vendor.

To avoid unexpected inaccuracies, the scheme of test topology is kept simple. Fig. 1 presents the typology. The core of the test environment was the WAN emulator *Apposite Netropy 10G* [19], which allows the emulation of WANs under various conditions such as packet delay, packet loss rate and jitter in different variations, with an accuracy of about 20 ns. By comparison, software emulators, such as *NetEm*, provide an accuracy of about tens of milliseconds and this value is greatly dependent on the hardware and operating system [20]. Moreover, software emulators are very limited in their maximum achievable data rates. *Apposite 10G*, for example, enables a transmission through Ethernet traffic with an overall throughput of up to 21 Gbit/s on both copper and fiber optic links.

The testbed topology used here contains two servers, connected via the 10 Gbit/s Ethernet switch Extreme Networks Summit x650 and via the WAN Emulator. The typology was implemented by means of fiber optics with a 10 Gbit/s bandwidth, see Fig. 2.

 ISBN: 978-1-61208-256-1

There is no background traffic on the path since this investigation focuses on the pure applications' performance, not on the fairness aspects of the protocols. The setup corresponds to the case when a L2-VPN is used for big data transmission and another application's traffic is isolated by means of QoS.

Each server is equipped as follows:

- *CPU: Intel Xeon X5690 @3.47GHz;*
- *RAM: 42 GiBytes (speed 3466 MHz);*
- *OS: Linux CentOS 6.3;*
- *NIC: Chelsio Communications Inc T420-CR, 10Gbit/s.*

Operating system socket buffers were extended up to:

- */proc/sys/core/net/wmem_max – 64MiBytes*
- */proc/sys/core/net/rmem_max – 64MiBytes*

The MTU size of all network devices along the path was set to 8900 Bytes.

For sending and receiving data with a rate of 10 Gbit/s, it is necessary to have a storage device that can read on the sender side and write on the receiver side with a sustained rate not less than 1 250 MByte/s (corresponding to 10 Gbit/s). Off-the-shelf hard drives provide read and write rates of up to 100 MByte/s, so in the investigated case, data transfer rate would have been limited by the hard drives. To circumvent this limitation, storage systems such as RAID arrays with write/read rates not less than the expected transport rate must be used.

In the presented experiments, both storage write and read bottlenecks and inefficient file access implementations were avoided by using a RAM-based file system on both servers. In comparison to common hard drives, the read rate of *RAMdisk*, as obtained in several test runs during these investigations, was not less than 4 500 MiBytes/s; the write rate of *RAMdisk* was not less than 3 000 MiBytes/s. Therefore, the servers used for tests were equipped with 42 GiBytes of RAM onboard, but due to the operating system's RAM requirements, it was only possible to use 30 GiBytes of space on *RAMdisk* for test purposes.

Under ideal conditions, a transmission of 30 GiBytes through the network with a bandwidth of 10 Gbit/s without impairments, as explained in (1), should take about 26 seconds.

$$T = \frac{S}{R_i} = \frac{30 \times 1024^3 \times 8 / 10^6}{10 \times 10^3} = 25.76\ s, \qquad (1)$$

where T – time of transmission; S – Size of data, R_i – ideal data rate.

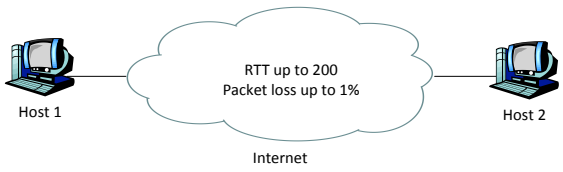

Figure 1. Logical view of topology

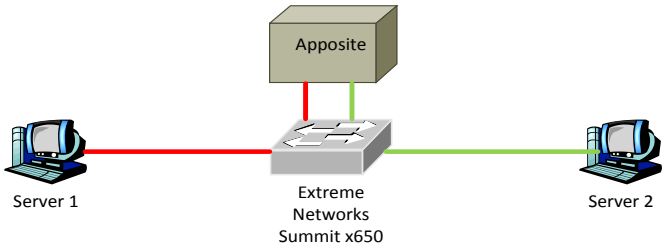

Figure 2. Technical view of topology

However, this calculation disregards L2-L4 headers along with some proprietary protocol headers and the overhead for connection management and retransmissions.

So, under real conditions, for some packet overhead and retransmission handling, each solution needs a certain amount of time for connection initialization and the releasing of the network path. Besides this, in high-performance implementations, initialization of the protocol stacks and the internal buffers often takes a significant amount of time, which is also investigated during this research.

V. EXPERIMENTAL RESULTS

The experiment on each data transport solution under consideration consists of 25 consecutive tests. Each test comprises the transfer of a 30 GiBytes file from one server to another through the network emulator. The RTT latency range is varied from 0 to 200 ms in steps of 50 ms and the packet loss rate takes the values 0; 0.1; 0.3; 0,5 and 1 %. Since one km of fiber optics delays the signal by about 5 µs, the maximum RTT in this test corresponds to 20 000 km of fiber channel in both the forward and the backward directions. The RTT is configured symmetrically across the forward and backward paths of the emulator; thus 200 ms of RTT would delay data by 100 ms and acknowledgments or other control information in the backward direction by another 100 ms. The packet losses are randomly injected according to a normal distribution, whereby the set loss ratio is applied to both forward and backward direction. Such packet loss behavior is easier to reproduce, and it is more complicated for protocols to handle than typical packet loss behavior on the Internet [16]. An attempt was made to configure each solution so that the maximum possible data rate and the minimum possible overall transmission time were achieved. The tuning of the operating system and the configuring of parameters are described below for each solution. All the tests were repeated 4 times to avoid inaccuracies, and the best result of each series is presented on the plots.

The results of each test contain two parameters: data rate and transfer duration. The first parameter is average data rate i.e. the average speed of data transportation shown by the application during the experiment. The second parameter is independent of the solution output and represents the time interval. This time interval was collected

by means of the operating system and shows the period of time between the launching of the send command and the time of completion of this command. This time interval contains not only the time of actual data transmission but also the time for service and retransmission overhead.

A. *Velocity*

This solution was developed in the USA. It is a TCP-based file transfer application, and, according to the vendor's web site, it allows the available path capacity to be fully utilized. *Velocity ASC* is also available with on-the-fly data encryption of up to 24 Gbit/s and AES encryption of up to 1 600 Mbit/s. The supported platforms are *Windows*, *Mac OSX*, *Linux* and *Solaris*. According to the user manual, this solution automatically adapts its parameters to network conditions and chooses the optimal parameters for data transmission. Fig. 3 shows the behavior of the transport rate. The results of tests in the presence of delays of more than 0.1 % are not shown since the data rate here was lower than 100 Mbit/s.

Increasing latencies do not significantly affect *Velocity*'s data rate behavior, slowing it down to only 8 Gbit/s. The solution performs reasonably in the presence of small packet loss rates without any emulated delay (back-to-back RTT latency in the testbed is about 0.15 ms). Thus it achieves a data rate of 9 Gbit/s in the presence of 0.1 % of packet loss, and this value decreases down to 500 Mbit/s with a packet loss of 1%. However, this configuration does not correspond to the situation in Wide Area Networks. In the presence of 0.1 % packet loss and at least 50 ms RTT, the data rate is reduced to 250 Mbit/s.

By default *Velocity* uses multi-streaming TCP. It opens 7 TCP sockets on every single test on each side. When the number of streams is manually set to 1, the data rate in presence of 200 ms RTT without packet loss is about 2.2 Gbit/s. The transfer durations of the solution are shown in Fig. 4. The numbers on the plot are obtained for two cases: without latency and with a latency of 200 ms. A result worth noting was obtained at a loss rate of 0.1% and an RTT of 0 ms. Under these conditions, the data rate in the presence of losses is, with 800 Mbits/s, less than the value without loss rate. However, the transfer duration in the latter case is higher by only 0.1 ms. This behavior was observed in several experiments.

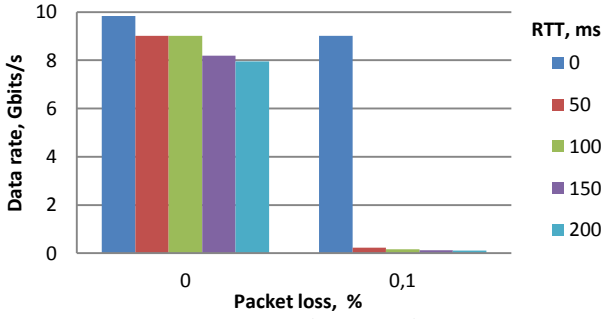

Figure 3. Behavior of data rate of *Velocity*

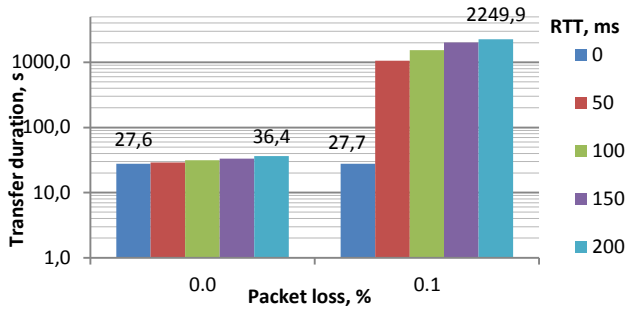

Figure 4. Data transfer duration of Velocity

B. *TIXstream*

This transfer engine has been developed by Tixel GmbH, Germany, which spun off from Technicolor Corporate Research in 2010. The core of *TIXstream* is Tixel's proprietary Reliable WAN Transfer Protocol (RWTP) [21], which provides high-performance data transmission between two hosts in the network using only one UDP-socket on each host. The application works under Linux OS.

TIXstream 3.0 (the latest version) provides up to 20 Gbit/s end-to-end performance. It has a platform-independent web-based user interface for the management of data transmission between remote SAN- and NAS systems. *TIXstream* also provides on-the-fly AES-256 encryption of data without any effect on data rate [22]. *TIXstream* has a peer-to-peer architecture and uses one TCP socket for control communication and one UDP socket for data transmission connection on both sides.

Parameters of application:

- *RWTP Buffer size – 4362076160 Bytes (4 GiBytes)*
- *MSS = 8800 Bytes*
- *Receiver buffer size (on both sides) = 1073741824 Bytes (1GiByte)*
- *Sender buffer size (on both sides) = 1073741824 Bytes (1GiByte)*

The behavior of *TIXstream*'s data rate as a function of network impairments is shown in Fig. 5.

There is no visibly decreasing effect on data rate behavior till 100 ms RTT and till 0.1 % of packet loss. The

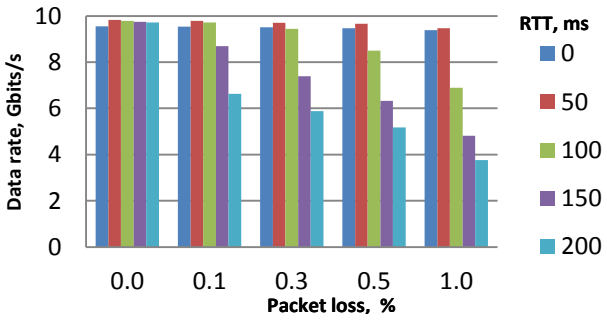

Figure 1. Behavior of data rate of *TIXstream*

solution achieves not less than 9.7 Gbit/s (97% of capacity) with these impairments. With higher delays in the presence of heavy packet losses, the figure shows decreasing data rates down to 3 750 Mbit/s, as on a path with 200 ms of RTT and 1% of packet losses. However, with modest impairments that correspond to fairly normal WAN links, for example RTT 150 ms and packet loss rate 0.1 %, *TIXstream* achieves a data rate of about 8 700 Mbit/s; an 87 % utilization of a 10 Gbit/s link. It is worth noting that in the presence of 50 ms of latency, *TIXstream* performs better than without any latency for all values of loss rate. This behavior was found in several experiments.

Fig. 6 shows that the transfer duration quite accurately corresponds to the behavior of the data rates. However, the theoretically minimum time of transmission calculated in Section IV, with a data rate of 8700 Mbit/s, is 29.62 s versus the 37.25 s that was measured in the experiment. These 7.63 seconds were spent on connection initialization, and the establishing and releasing of the control channel. Since no packet loss shall occur on this link, time for packet retransmission shall be neglected.

C. FileCatalyst Direct

FileCalatyst Direct was developed by Unlimi-Tech Software Inc., a Canadian based company. Like *TIXstream*, it transmits data via UDP and implements packet loss management, rate and congestion control in the user layer. The application obeys a client-server architecture and the solution operates under *Windows*, *Mac OSX*, *Linux* and *Solaris* operating systems. The data sheet on the vendor's website shows that this solution provides data rates of up to 10 Gbit/s [23] and that there is an option to use AES encryption for secure transmission. *FC Direct* provides both, graphical and command line user interfaces for server and client applications.

Parameters of application:

- *Start rate 9000000 (9Gbit/s)*
- *Application unit size 8800 Bytes*
- *Buffers* = 3840000000 *Bytes (3,58 GiBytes)*
- *Number of send sockets 10*
- *Number of receiver sockets 4*

As shown on Fig. 7, *FC Direct* achieves 90 to 94 % link utilization, even under high network impairments. Data rate behavior is fairly immune to growing latency and packet loss ratio. The data rate of *FC Direct* shows values between 9 Gbit/s and 9.4 Gbit/s for all the tests. During the tests, the Linux system monitor reveals that each data transmission opens 10 UDP sockets on the sender side and 4 UDP sockets on the receiver side and one TCP socket on each side. In this mode, maximal data rates can be achieved. Data packets from ten sender sockets are not uniformly distributed over all four receiver sockets, but according to a special proprietary distribution rule. The vendor does not call it multi-streaming but "more intelligent resource management". However, with this behavior, significant firewall transversal issues are to be expected.

The distribution of session time durations showed on Fig. 8 has slightly monotonically increasing behavior with rising latencies.

D. Catapult Server

The *Catapult Server* is TCP-based and was developed by XDT PTY LTD, Australia. This solution follows a client-server architecture and, according to the vendor's web site, provides up to 8 Gbit/s on the 10 Gbit/s link. The solution functions under the *Windows*, *MAC OSX* and *Linux* operating systems. The vendor positions the solution as a high data rate transmitting tool for networks with a high level of latency but without any packet losses. To prepare the operating system for high speed transmissions, the vendor suggests using a shell script to change network parameters. By default, this script extends the TCP buffers to 32 Mbytes. However, for our tests, the 64 Mbytes setting was chosen since better performance had been reached with

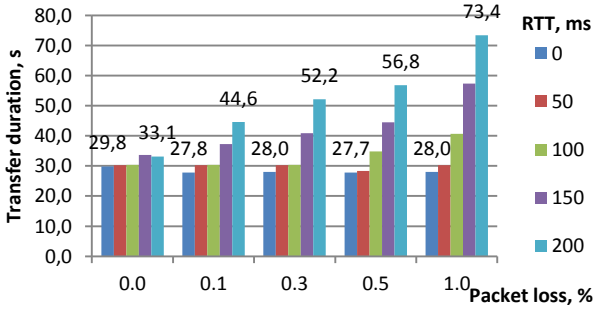

Figure 2. Data transfer duration of *TIXstream*

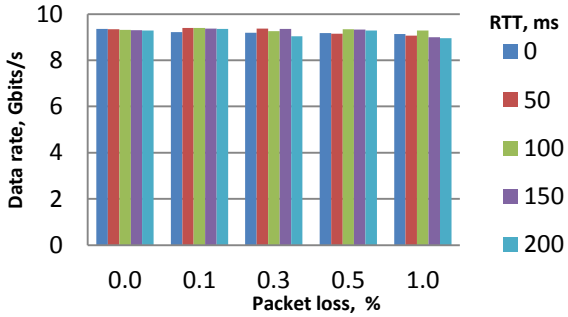

Figure 3. Behavior of send rate of FC Direct

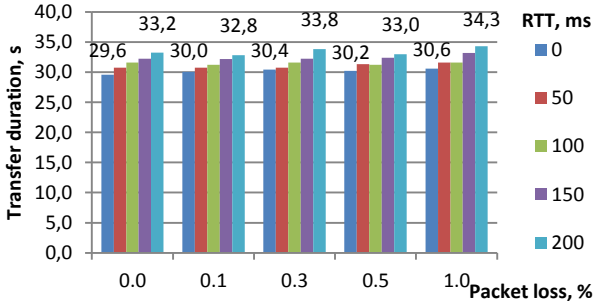

Figure 4. Data transfer duration of FC Direct

this setting. The script changes are:

- *tcp_congestion_control=htcp*
- *net.ipv4.tcp_rmem=4096 87380 67108864*
- *net.ipv4.tcp_wmem=4096 65536 67108864*
- *net.ipv4.tcp_no_metrics_save=1*
- *net.core.netdev_max_backlog=250000*
- *net.core.rmem_max=67108864*
- *net.core.wmem_max=67108864*

To improve the behavior of this solution in the presence of packet losses, the manufacturer's support team also suggested applying the following configurations:

- *net.ipv4.tcp_timestamps=1*
- *net.ipv4.tcp_sack=1*

Note that the command line client of XDT - *sling shot copy*, which was used for the tests, shows the data rate as GB/s, probably it means GiByte/s. Furthermore the value 1.1 GB/s immediately follows the value 1.0 GB/s, without any intermediate values. However the solution shows a transfer duration with an accuracy of up to milliseconds. Therefore, the data rate was calculated as

$$R_{XDT} = \frac{S}{T_o}, \qquad (2)$$

whereby R_{XDT}– is the data rate of XDT, which is used for result presentation; S – data size (30 GiByte), T_o – transfer duration from the output of client application.

Fig. 9 represents the data rate of *Catapult Server* dependent on network impairments. The presence of packet losses on the link makes the transmission ineffective, so the data rate is reduced to less than 100 Mbit/s. However, in the presence of 150 ms RTT without packet loss, transmission is about 8300 Mbit/s

Fig. 10 shows the transfer durations for Catapult technology.

E. ExpeDat

ExpeDat is a UDP-based data transport solution developed by Data Expedition Inc., USA. The core of this application comprises the Multipurpose Transaction Protocol (MTP) [24], developed by the founder of Data Expedition. *ExpeDat* supports *Windows*, *Mac OSX*, *Linux* / *Solaris*, *NetBSD* / *FreeBSD*, *AIX* and *HP-UX* platforms. According to the company's web site, *ExpeDat* allows

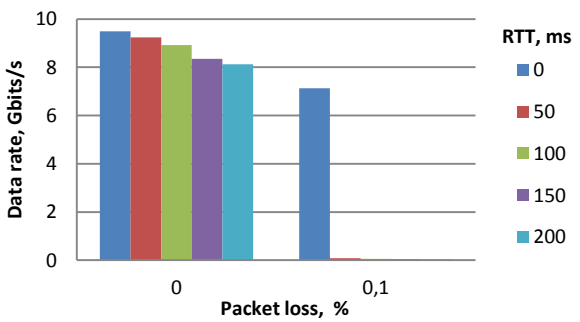

Figure 5. Behavior of data rate of XDT Catapult

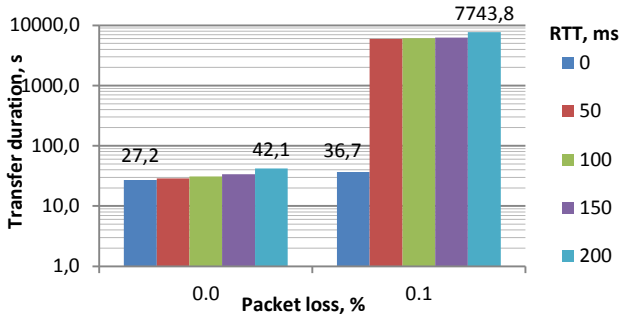

Figure 6. Data transfer duration of XDT Catapult

transmission of data with 100 % utilization of allocated bandwidth and in the presence of on-the-fly AES encryption [25]. It implements the transport protocol logics on a UDP channel, and uses a single UDP socket on each side of the connection for both data transmission and control information.

Though the product web site [25] claims that the solution has "zero-config installation", the significant increase of data rate, namely from 2 Gbit/s up to 9 Gbit/s, even without impairments (RTT=0 ms, packet loss = 0%), was obtained only after application of configuration changes as follows:

- *MSS – 8192 Bytes*
- *Environment variable MTP_NOCHECKSUM=1*

With high values of packet loss on the channel, the higher results were achieved using the following option on the command line:

- *-N 25*

The use of this option shows that heavy packet loss rate is introduced in a channel.

In Fig. 11, the data rate values of *ExpeDat* tests are presented. The plot shows that network latencies lead to a much higher reduction of the transmission rate than packet loss.

The distribution of transfer times is presented in Fig. 12.

VI. COMPARISON OF THE SOLUTIONS

Since not all of the investigated solutions perform well in the presence of heavy packet losses, the comparison of data rates was split into two stages. The first stage is

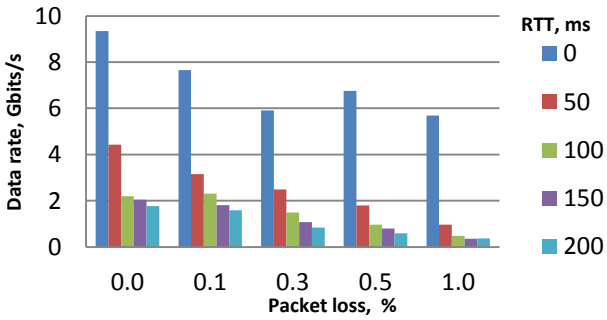

Figure 7. Behavior of data rate of *ExpeDat*

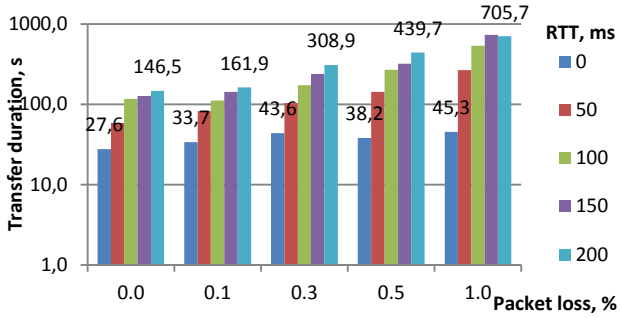

Figure 8. Data transfer duration of *ExpeDat*

dedicated to a comparison of all presented solutions on the networks with different packet latencies and without any packet loss. In the second stage, the solutions are compared under harder conditions for terrestrial networks - with packet loss of 1% and the whole range of investigated RTT. Only solutions that show a data rate higher than 1% from maximal channel capacity (100 Mbit/s) have been considered in the second stage.

A comparison between the distribution of transfer duration without packet losses and the ideal value shows which solution spends more time on service needs such as the initialization and releasing of a channel.

A further comparison shows the discrepancy between the actual time of transmission and the calculated time from the output of all solutions. This analysis is also split into two stages as described above and shows how the values from the output of the particular solution correspond to reality.

Fig 13 shows a consolidated diagram of transmission data rates of all tested solutions in the presence of increasing latency and without any packet loss on the path. The first set of bars shows how fast a large set of data can be transmitted in a back-to-back connection. For this case, the highest result was achieved by *Velocity*. However all solutions showed results of not less than 9.3 Gbit/s. With increased latencies, *Velocity* performs worse, and of all the remaining cases, *TIXstream* shows the best performance with up to 9.8 Gbit/s (98% channel utilization) without any significant decrease at higher RTTs. *FC Direct* also shows very stable

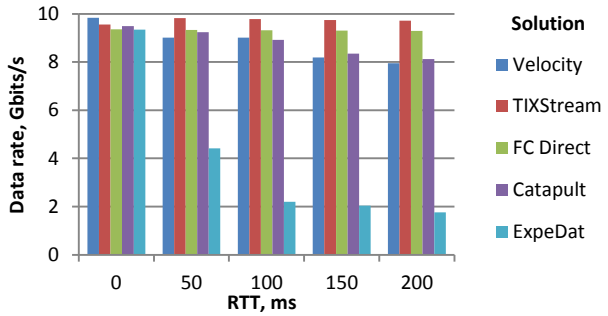

Figure 9. Comparison of data rates of investigated solutions; packet loss = 0

results. For all presented impairments, its data rate lies between 9.2 and 9.3 Gbit/s. All other solutions show decreasing data rates on increasing round-trip-times.

TCP-based solutions obviously cannot cope efficiently with the presence of packet losses on the path. Although for all solutions except *Velocity*, the support teams of the respective manufacturers were involved, those solutions did not provide adequate results in the presence of packet loss. Fig. 14 represents the behavior of solutions of stage two in the presence of 1% of packet loss.

With an RTT of 0 and 50 ms, *TIXstream* shows the best results. However, starting at 100 ms, throughput is decreased whereas *FC Direct* shows nearly constant data rates. The *ExpeDat* data rate abruptly decreases down to 5.7 Gbit/s on zero-delay links, and with increasing latencies *ExpeDat*'s results are lower than 1 Gbit/s.

As pointed out in Section IV, the theoretical minimum transfer duration for the transport of 30 GiBytes of data via a 10 Gbit/s WAN is 25.76 seconds. Fig. 15 compares, for each solution, the ideal value with the lowest transfer durations obtained during the experiments.

The minimum transfer duration was achieved by XDT *Catapult server*. The time is only 1.4 seconds longer than the theoretical minimum. This means that XDT *Catapult* initializes the software stack along with protocol buffers, and establishes and closes the connection within less than 1.4 seconds. The time overhead of *Velocity*, *TIXstream* and *ExpeDat* is slightly higher but still less than 2 seconds. The worst result was obtained when using *FC Direct*: it needed about 3.8 s. for ramping-up and finishing the application.

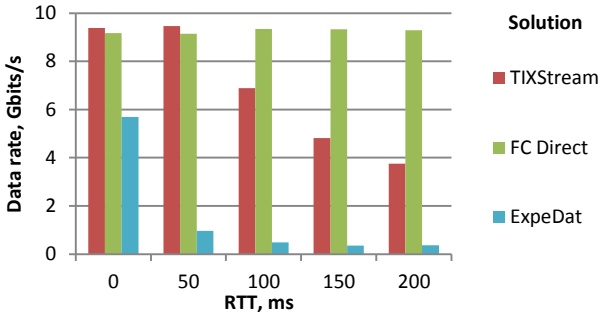

Figure 10. Comparison of data rate of tested solutions; packet loss = 1%

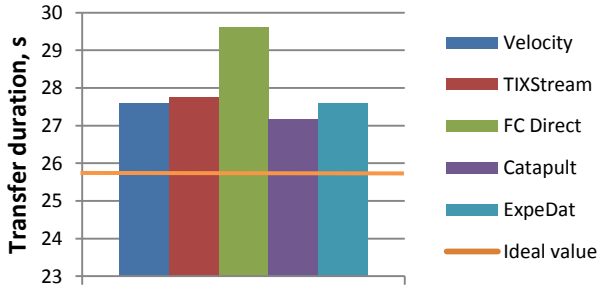

Figure 11. Comparison of transfer durations with theoretical minimum

Also of interest is the accuracy of the performance outputs of the transport solutions. During the experiment, the actual data rate was obtained from the output of the running application, and program run time was also measured by means of the operating system. Transfer durations with a transmission of 30 GiByte with a data rate from output are calculated as

$$T_t = \frac{S, [Bits]}{R_o, [\frac{Bits}{s}]} \quad (3)$$

where T_t – is the calculated transfer duration; R_o– data rate from output; S – data size (30 GiByte).

The differences between calculated transfer durations and real program run time for the tests performed without packet loses are presented in Fig. 16. The discrepancy of the values generally has an increasing trend at higher latencies. *Velocity* showed the lowest value of discrepancy along the tests without packet loss. The second TCP-based solution – XDT *Catapult* - shows good results on RTTs below 200 ms. However, with 200 ms RTT, this solution shows the worst of all results. *ExpeDat* almost always has the lowest discrepancy values for all cases. In the presence of RTTs of 100 ms and 150 ms without loss rate, the solution showed negative result, meaning that the actual transfer duration was lower than the calculated one. It is evident that the output of some solutions show the average achieved data rate including service processes such as connecting, initializing and releasing the link, and some of the solutions show the average data rate of the transmission process only.

Similar to Fig. 16 but with a packet loss of 1%, Fig. 17 shows the differences of calculated transfer durations and real program run time. A comparison of these two figures shows that FC Direct has almost the same discrepancies for all RTT cases except for the 200 ms and 150 cases, where the discrepancy is higher in the presence of packet loss than without it. The results of *TIXstream* have a decreasing trend and *ExpeDat* shows again the lowest values – the actual times of transmission are almost equal to the calculated ones.

VII. CONCLUSION

This work compares the state of the art of commercial solutions for reliable fast data transport via 10 Gbit/s WAN IP networks in the presence of high delays and varying packet loss rates. The main problem of such research is that the vendor companies usually hide the technology used for the accelerated data transport. The protocol used in *ExpeDat* solution – MTP - is covered by some US patents. However this does not mean that *ExpeDat* does not use any algorithms besides the ones described in those patents. The only independent method to assess these commercial solutions is to externally observe the solutions during tests under well-defined conditions.

All investigated solutions position themselves as reliable high-speed transfer applications designed to provide alternatives to FTP/TCP and overcoming the pure TCP

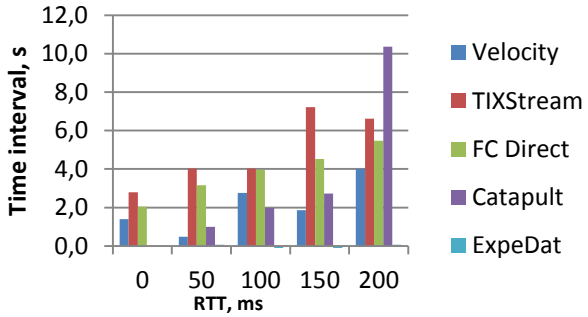

Figure 12. Difference between calculated and actual transfer durations; Packet loss=0

performance on 10 Gbit/s-WAN connections by orders of magnitude. Two of them, *Velocity* and XDT *Catapult*, exploit the TCP stack of the *Linux* OS and the rest - *FC Direct*, *TIXstream* and *ExpeDat* use UDP sockets and implement the protocol logics in the user-level.

The results obtained show that solutions based on TCP inherit its native problems on 10 Gbit/s links – a significant decrease of data rate down to 1% of the link capacity in the presence of packet loss on the path. The commercial solutions achieve a higher speed by increasing TCP window size or by establishing multiple parallel TCP streams. However, the experiments show that this solution only works on links without any packet loss. However, even the known STCP [26] on WAN networks with a low loss rate show a reasonable result of about 5 Gbit/s [1]. Although in that paper, the authors tested pure protocol performance, their results show that it is possible to achieve good results by only tuning the TCP on such networks.

UDP-based solutions show a good utilization of a 10 Gbit/s path even under bad network conditions such as a loss rate of 1% in the presence of RTTs of up to 200 ms. The best link utilization at the highest impairment value was achieved by FileCatalyst *Direct* – the values were never lower than 93% for all performed tests. For the loss ratio up to 0.3% and RTT up to 100 ms, *TIXstream* shows a better utilization of about 97 %.

Transmission duration measurements were primarily intended to prove that the solutions show accurate data

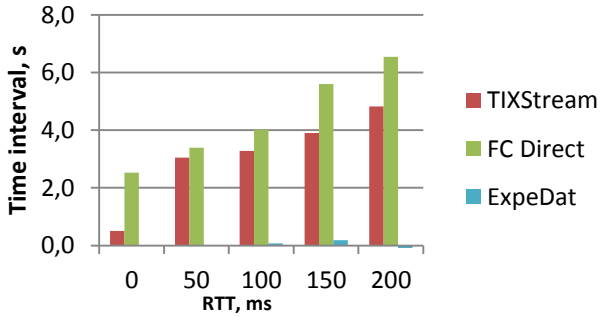

Figure 13. Difference between calculated and actual transfer durations; Packet loss =1

transport numbers in their outputs. The comparison showed that the lowest transfer duration of each solution is fairly close to the ideal one and that the discrepancy of the obtained output values are close to reality for all solutions.

Each solution uses some time for the allocation of system resources and the initialization of network resources,. This time cannot be neglected, at least not in sessions with up to 30 GiBytes data transport. The comparison presented in Fig. 15 attempts to assess this service time. Probably, the time overhead is also due to the solutions not fully utilizing the bandwidth. It was found that the data rate of *FC Direct* is not the lowest one, but the transfer duration is higher than for all other solutions under test conditions. This result is possibly due to known java performance bottlenecks, because *FC Direct* is the only solution written purely in java.

When the solutions work under very light network impairment conditions (for example back-to-back) and the data rate achieves maximal value, the CPU usage is fairly high. For example, the maximum achieved data rate of *ExpeDat* seems to be due to CPU limitations. The system monitor showed 99% CPU usage in the *ExpeDat* process, and it also showed that one core of twelve is used in 99%. Other solutions, e.g. *TIXstream*, showed a CPU usage of about 150%, the usage of two used cores was about 70 % and 80 % respectively on the sender side, and on the receiver side 3 cores were used with a usage of 40%, 90%, 30%. This solution distributes the performance among several cores to maximally use the available bandwidth when possible.

One more significant point of resources management is the socket use. As shown in Section V.C, *FC Direct* uses different numbers of sockets on the sender and receiver sides. This use causes no problems for corporate LANs or simple back-to-back connections. However, for data transport using more complex structures, like real Internet connections, this use could cause problems on such devices such as firewalls. Such problems are well known even in simple multi streaming cases. This is an even worse situation in which each sender socket is sending data to different destination ports, so at least M x N port pairs must be tunneled in the firewall. It is very likely that intrusion detection systems can consider such behavior as violent traffic.

VIII. FUTURE WORK

The analyzed solutions were tested on their abilities in the presence of high values of latency and packet losses. However, delay jitter is also a common network impairment and measurements using different values and different jitter patterns would also be of interest.

The present research shows the behavior of the solutions in an empty path such as VPN. Further investigations could be made into the behavior of the solutions in the presence of back-ground traffic.

The testbed topology was simplified to get a first representation of the presented solutions. An extension of the experimental topology makes sense for in-depth investigations.

During the experiments, only the performance of data transfer for commercial applications was investigated. To get a deeper understanding of only the telecommunication part, it would be of interest to make tests with the technology cores (e.g. protocol stacks without any wraps as e.g. file system).

The questions of system resource consumptions were addressed very briefly here. It would also be interesting to research this topic more extensively.

IX. REFERENCES

[1] Y. Wu, S. Kumar, and S.-J. Park. "Measurement and performance issues of transport protocols over 10 Gbps high-speed optical networks". Computer Networks, vol 54. 2010, pp. 475-488. doi:10.1016/j.comnet.2009.09.017

[2] H. Kamezawa, M. Nakamura and M. Nakamura. "Inter-Layer Coordination for Parallel TCP Streams on Long Fat Pipe Networks". Proc. of the 2004 ACM/IEEE conference on Supercomputing. Pittsburg, PA, USA. 2004, pp. 24 -34.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. "Above the Clouds: A Berkeley View of Cloud Computing". 2009. pp. 19-25. Tec. Rep. No. UDB/EECS-2009-28.2009.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. "A view of Cloud Computing". Communications of the ACM. ACM New York, NY, USA, April 2010, Vol. 53, Issue 4, pp. 50-58. doi:10.1145/1721654.1721672

[5] S. Höhlig. "Optimierter Dateitranfer über 100 Gigabit/s". 100 Gigabit/s Workshop of the DFN, Mannheim. Sept. 2011.

[6] Aspera. Aspera. "custumer Deluxe Digital Studios". [retrieved: 11, 2012] http://asperasoft.com/customers/customer/view/Customer/show/deluxe-digital-studios/.

[7] Y. Gu and R. L. Grossman. "UDP-based datatransfer for high-speed wide area networks". Computer Networks. Austin, Texas, USA. May 2007, Vol. 51, issue 7, pp. 1465-1480. doi:10.1016/j.comnet.2006.11.009,

[8] E. He, J. Leigh, O. Yu, and T. A. DeFanti. "Reliable Blast UDP : Predictable High Performance Bulk Data Transfer". Proc. of IEEE Cluster Computing. Chicago, USA. Sept. 2002, pp. 317-324.

[9] R. L. Grossman, Y. Gu, X. Hong, A. Antony, J. Blom, F. Dijkstra, and C. de Laat. "Teraflows over Gigabit WANs with UDT". Journal of Future Computer Systems. Volume 21, 2005, pp. 501-513. doi:10.1016/j.future.2004.10.007

[10] L. Herr and M. Kresek. "Building a New User Community for Very High Quality Media Applications On Very High Speed Networks". CineGrid. [retrieved: 02, 2013] http://czechlight.cesnet.cz/documents/publications/network-architecture/2008/krsek-cinegrid.pdf.

[11] X. Wu and A. A. Chien. "Evaluation of rate-based transport protocols for lambda-grids". High performance Distributed Computing, 2004. Proc. of 13th IEEE International Symposium on High Performance Distributed Computing.Honolulu, Hawaii, USA. 2004, pp. 87-96.

[12] Bitspeed LLC. "From Here to There - Much Faster". Whitepaper. [retrieved: 10, 2012.] http://www.bitspeed.com/wp-content/uploads/2011/10/BitSpeed-White-Paper-From-Here-to-There-Much-Faster.pdf.

[13] Tixel GmbH. Tixstream: Overview. [retrieved: 10, 2012] http://www.tixeltec.com/ps_tixstream_en.html.

 ISBN: 978-1-61208-256-1

[14] File Catalyst. "Accelerating File Transfers". Whitepaper. [retrieved: 10, 2012] http://www.filecatalyst.com/collateral/Accelerating_File_Transfers.pdf.

[15] XDT PTY LTD. "High-Speed WAN and LAN data transfers". XDT. [retrieved: 10, 2012.] http://www.xdt.com.au/Products/CatapultServer/Features.

[16] V. Paxson. "End-to-End Internet Packet Dynamics". Networking, IEEE/ACM Transactions. 1999, Vol. 7, Issue 3, pp. 277-292. doi: 10.1109/90.779192

[17] Y. A. Wang, C. Huang, J. Li, and K. W. Ross. "Queen: Estimating Packet Loss Rate between Arbitrary Internet Hosts". Proc. of the 10th International Conference on Passive and Active Network Measurement. Seoul, Korea. 2009, pp. 57-66.

[18] B. W. Settlemyer, N. S. V. Rao, S. W. Poole, S. W. Hodson, S. E. Hick, and P. M. Newman. "Experimental analysis of 10Gbps transfers over physical and emulated dedicated connections". Proc. of Computing, Networking and Communications (ICNC). Maui, Hawaii, USA. 2012, pp. 845-850.

[19] Apposite Technologies. "Apposite". [retrieved: 10, 2012] http://www.apposite-tech.com/index.html.

[20] A. Jurgelionis, J.-P. Laulajainen, M. I. Hirvonen, and A. I. Wang. "An Empirical Study of NetEm Network Emulation Functionalities". 20. ICCCN. Maui, Hawaii, USA, 2011, ISBN 978-1-4577-0637-0, pp. 1-6. doi: 10.1109/ICCCN.2011.6005933

[21] Tixel GmbH. White Papers and Reports. Tixel. [retrieved: 11, 2012] http://www.tixeltec.com/papers_en.html.

[22] Tixel GmbH. Tixel news. tixel.com. [retrieved: 10, 2012] http://www.tixeltec.com/news_en.html.

[23] FileCatalyst. FileCatalyst. "Direct". [retrieved: 10, 2012] http://www.filecatalyst.com/collateral/FileCatalyst_Direct.pdf.

[24] Data Expedition, Inc. Data Expedition. "Difference". [retrieved: 10, 2012] http://www.dataexpedition.com/downloads/DEI-WP.pdf.

[25] Data Expedition, Inc. Overview. Data Expedition, Inc. [retrieved: 10, 2012.] http://www.dataexpedition.com/expedat/.

[26] R. Stewart, Q. Xie, Motorola, K. Morneault, C. Sharp, Cisco, H. Schwarzbauer, Siemens, T. Taylor, Nortel Networks, I. Rythina, Ericsson, M. Kalla, Telcordia, L. Zhang, UCLA, V. Paxson and ACIRI. "Stream Control Transmission Protocol". IETF, RFC 2960. [retrieved: 01, 2013] http://www.ietf.org/rfc/rfc2960.txt.

 ISBN: 978-1-61208-256-1