Application Networking Beyond 10 Gbit/s

April 2010

Ronald van der Pol and Freek Dijkstra
SARA Computing and Networking Services,
Science Park 121, 1098 XG Amsterdam, The Netherlands
e-mail: Ronald.vanderPol@sara.nl, Freek.Dijkstra@sara.nl

Paper type
Research paper

Abstract
Wide area networking speeds are being upgraded to 40 and 100 Gbit/s. However, applications face several challenges when trying to take advantage of these higher speeds. SARA has analysed these challenges and performed several measurements with respect to possible bottlenecks. The analysis shows that research is needed with respect to several aspects of network load balancing. SARA's performance and scalability measurements show that a modern off the shelf server is capable of sending 40 Gbit/s of network traffic with multiple parallel flows. The server that was used for the measurements had several limitations. The network interface cards were limited in throughput, but by using several 10 Gigabit Ethernet ports in parallel SARA was able to transmit 39.5 Gbit/s. Disk I/O measurements showed linear speedup with 6 solid state disks up to around 1400 MiByte/s. Based on these results and the architecture analysis we expect that 40 Gbit/s of sustained read speed is probably possible with more disks. This will be done in follow-up measurements with more than 6 disks. This means that a single application is probably capable of sending 40 Gbit/s of traffic on a modern server. Previously, clusters and grid setup were needed to reach these application speeds. This opens new exciting possibilities for demanding applications.

Keywords
Network Load Balancing, High Performance Networking, High Performance Computing, Solid State Disks, 40Gbase, 100GBase

1. Introduction
Several National Research and Education Netorks (NRENs) are starting to deploy 40 Gbit/s and 100 Gbit/s links. On the other hand, many servers are still limited by a 1 Gbit/s interface. And even when a server has a 10 Gbit/s interface, applications can rarely use the full capacity, especially in disk to disk transfers, because of bottlenecks in the server. This is called the end-to-end performance gap. At the same time the user requirements for high performance data transfers are increasing rapidly. The amount of data is exploding and there is a need to transfer these huge data sets from one server to another as fast as possible. The speeds used in streaming are also increasing rapidly. Uncompressed 4K video streaming requires around 6 Gbit/s per stream and remote visualisation of scientific data sets on e.g. tiled panels can easily generate tens of Gbit/s. This mismatch between core network capacity and the capacity available for applications needs to be resolved.

2. High Speed Data Transfers
Typical scientific data sets consist of tens of terabytes of data and the sizes continue to grow. E.g., the datasets used by the physisists involved in the LHC project are in the order of 100 TB. Storing data in places of cheap green sustainable energy is also a hot topic. Some envision a situation where data is continuously moved around to follow the sun or the wind. This results in huge amounts of data that

need to be transferred in a short timeframe from one facility to another. Transferring these amounts of data requires huge network capacity. E.g., sending a 100 TB file on a 10 Gbit/s network takes almost a day. On a 100 Gbit/s network it still takes more than two hours. Even with a terabit per second network it would still take almost 15 minutes. So we are nowhere near a situation where we can transfer our data in seconds from A to B, and in the mean time the data sets keep growing in size. Therefore, we need to increase the single flow network capacity, especially in the access. The bandwidth challenge of SC09 described it like this: *Now that we are in the petaflop era of high performance computing, how does that produce a petabyte data storage challenge, and in turn, a petabit network transport challenge? How fast can you move a scientific petabyte data set over a high performance research network?*

3. High Performance Computing

Petaflop supercomputers are being deployed and within a few years exaflop supercomputers will start to appear. More and more, these supercomputers are interconnected. An example of this is the European DEISA network. Supercomputer applications, like CosmoGrid, are taking advantage of this by running distributed on several supercomputers and synchronising shared data frequently. The faster the exchange of data goes, the faster the overall calculation runs. These environments will require terabit per second network capacities within a few years. The PRACE supercomputer project has defined this research work: *An efficient and scalable communication and I/O infrastructure is crucial for a balanced system reaching Petaflop/s performance. Therefore, special emphasis is given to assess the communication and I/O infrastructure of the prototype systems. For the communication infrastructure, this includes assessment of new generations of high-speed interfaces, switches, network topologies, network transfer protocols and connections to external networks. For the I/O infrastructure, this concerns evaluation of the I/O subsystem, in particular management of data within the system as well as from outside the system.*

4. High End Data Streaming

Streaming to tiled panels for remote scientific visualisation and remote collaboration (e.g. with the SAGE [1] application) generates data flows that can easily exceed 10 Gbit/s. SARA's current tiled panel consists of nine LCD panels of 2560x1600 pixels creating a 35 Megapixel tiled panel. Streaming with 60 frames per second to this panel would generate more than 50 Gbit/s of network traffic. At SC05 SARA streamed almost 20 Gbit/s over two 10G links between Amsterdam and San Diego to a 100 Megapixel tiled panel [2]. Streaming video adds an additional requirement to the network. The jitter must be very low to avoid hickups and artifacts in the video. Streaming at speeds that are close to the backbone capacity puts a lot of strain on the network design and implementation. Therefore, this is also a very useful tool to stress test modern research networks. Monitoring and measuring what is going on in the network is therefore essential.

5. Technologies to Increase Access Network I/O Capacity

Disk transfers have increased by using disk striping and disk clusters. Similarly, most computers have several compute cores to increase the total compute power. The next logical step is to investigate if multiple network interface cards and multiple flows can be used efficiently to increase the total amount of network I/O of a server. In order to achieve terabits per second we need to have many high capacity parallel flows between the same two endpoints in the network. This is a huge paradigm shift and work needs to be done in several areas.

6. Load Sharing with Link Aggregation

Whenever a higher capacity network technology is introduced (like the current introduction of 40G and 100G), it becomes available for core backbone links first. Next, it becomes available on edge routers and switches. Finally, interface cards for servers become available. During these transitions, link aggregation can be used to combine older generation network interfaces in order to be able to use the increased backbone capacity. SARA is investigating how to maximise the end-to-end throughput of a single application, e.g. a file transfer. This means we are dealing with a single flow (or a few flows, as will be discussed in the next section). In order to use link aggregation to increase the network capacity for a single flow, several additional problems arise. Most load sharing algorithms are based on a hash of the header that includes fields like the source and destination Ethernet and/or IP addresses, source and destination port, etc. That does not work for a single flow, because all packets would be mapped to the same link. Round-robin load balancing could be used to distribute single flow traffic evenly over multiple links. However, this introduces possible difficulties with respect to packet re-ordering and packet loss. When packets are sent over multiple links, they need to be reassembled at the receiver end in the correct order. This requires buffering at the receiving end, especially when many links are aggregated. When the RTT is large, a lot of buffer capacity is needed.

Several optimisations need to be investigated. Each network stream probably needs to be handled by its own compute core. Modern network interface cards also have various optimisation features, like checksum and segmentation offload, that need to be used efficiently. How to use these features in a multi-link setup where packets of the same flow are distributed over several interfaces needs to be investigated. Link aggregation will usually be done locally between a server and a switch, where the switch already has a higher capacity uplink. However, sometimes higher capacity WAN links are not available and several lower capacity WAN links need to be used in parallel. In this case link aggregation can be done end-to-end between the servers at either end. When using link aggregation over large distances, packet loss and packet re-ordering will be larger than will be the case for a link aggregation setup locally in a LAN. The effect on throughput needs to be investigated.

7. Load sharing over multiple IP streams

Link aggregation works at the data link layer. Aggregation can also be done at the transport layer. The IETF has recently started two working groups that are relevant to this. The Multiple Interfaces (MIF) working group will document current practises and identify issues and gaps with respect to the servers that have multiple interfaces and connections to the network [3]. The Multipath TCP (MPTCP) working group will work on *mechanisms to enable transport protocols to use multiple paths simultaneously and to distribute the load among the subflows of each path based on congestion* [4]. This transport layer aggregation also works when there is not a layer 2 connection between the end sites. It is therefore more versatile than layer 2 link aggregation. It will also work when the various end-to-end links do not have the same capacity, because it will distribute the packets based on congestion and not just in a round-robin manner. Of course, transport layer aggregation can also be used on layer 2 links and lightpaths. The throughput of link aggregation and transport layer aggregation in various setups needs to be compared and investigated.

8. Typical System Architecture

Appendix A shows the specifications of the server that was used by SARA for the tests described in this document. It is a Dell PowerEdge T610 [5]. This is an off the shelf server with Intel Xeon 5550 processor and Intel 5520 chipset [6]. Figure 1 shows the block diagram of the architecture of this

combination. The Xeon 5550 has four cores supporting 8 threads and runs at 2.66 GHz. It has three memory channels to DDR3 memory running at 1333 MHz. This has a peak transfer rate of 10667 MByte/s for each channel.

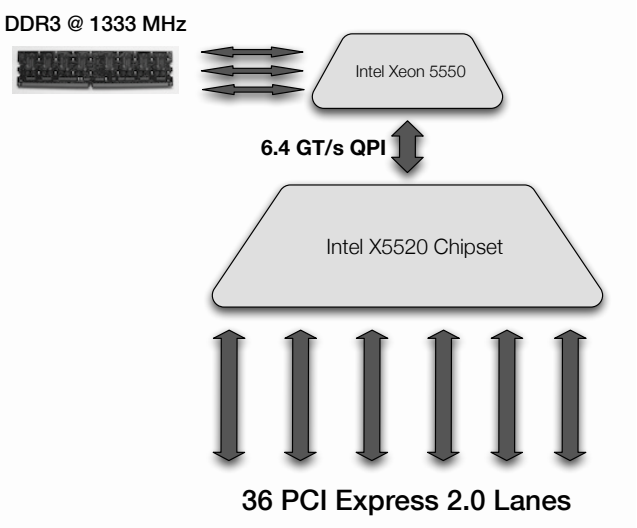

Figure 1 Typical architecture of an Intel Xeon 5550 with 5520 chipset

The I/O controller is the Intel 5520 I/O Hub, which is connected to the CPU with a 6.4 GT/s Quick Path Interconnect (QPI). The Intel 5520 supports 36 PCI Express 2.0 lanes of 500 MByte/s each, for a peak transfer rate of 18 GByte/s. The Dell T610 server used by SARA has two PCI Express 2.0 x8 slots, three PCI Express 2.0 x4 slots and a PCI Express 1.0 x4 slot for the PERC RAID controller.

9. Filesystem and Disk I/O

In order to determine the scalability of multiple solid state disks several tests were done. These measurements are based on previous work described in this report [7]. SARA reached the highest throughput with the BTRFS filesystem on Linux and ZFS filesystem on FreeBSD.

9.1 Solid State Disks

Solid state disks (SSDs) use non-volatile NAND (flash) memory instead of the rotating magnetic platters of a mechanical hard disk drive (HDD). The major advantages of solid state disks compared to regular hard disks are low energy consumption, high reliability (due to the lack of moving parts), and near-zero access times. For our purpose, a streaming server, we are only interested in the sustained bulk read times from disk. Despite that file systems have gone to extensive length to tune the sequential read performance for mechanical hard disk drives, earlier reports show that solid state disks still outperform mechanical hard disk drives [8,9]. There are currently two types of SSDs available on the market, single-level cell (SLC) and multi-level cell (MLC) SSDs. SLC SSDs achieve higher write performance, while MLC SSDs have lower cost and more capacity [8]. The file read performance is roughly the same for MLC and SLC SSDs [9]. For this reason, we conducted our experiments with MLC SSDs, using Intel's X25-M SSDs. The Intel specification for the X25-M SSDs claims a maximum throughput of 250 MByte/s sustained read speed. External parties have achieved up to 154 MByte/s in practice [9].

9.2 Experimental Setup

Read performance was measured for up to six solid state or hard disk drives in various configurations.

All experiments were performed using the sequential read speed in the IOzone benchmark. While sequential read has no meaning for solid state disks, this
allowed a good comparison to mechanical hard disk drives. The Phoronix test suite was used to automate the tests. Each test was repeated three times, while verifying that the CPU usage stayed well below 100% (making sure that the CPU was not the bottleneck). The file size of each test was 8 or 32 GByte, well above the 6 GByte internal memory of the machine.

9.3 Results

We got good sustained read performance for a single disk, easily exceeding 200 MByte/s and sometimes even over 250 MByte/s. Figure 2 shows the results of solid state disks versus hard disk drives. There was a levelling off at around 813 MiByte/s, or 6.82 Gbit/s.

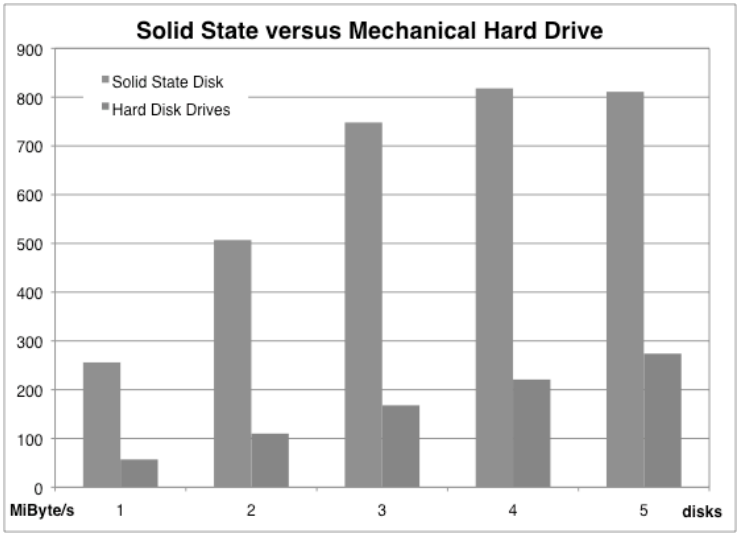

Figure 2 Disk array read performance for up to five solid state disks

This is close to the theoretical maximum of the PCI Express 1.0 x4 bus, so we repeated the measurement with a PCI Express 2.0 x8 slot. Because the cable between the PERC RAID controller card and the SAS distribution frame with 8 SAS connectors was too short, we had to partly disassemble the server to do the tests (see figure 3).

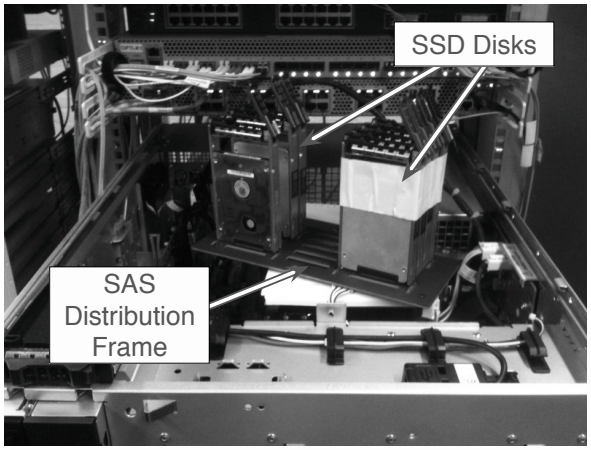

Figure 3 Experimental setup with the RAID controller in a PCI Express 2.0 slot

The result of these tests was a perfect linear speedup up to around 1400 MiByte/s, as shown in figure 4.

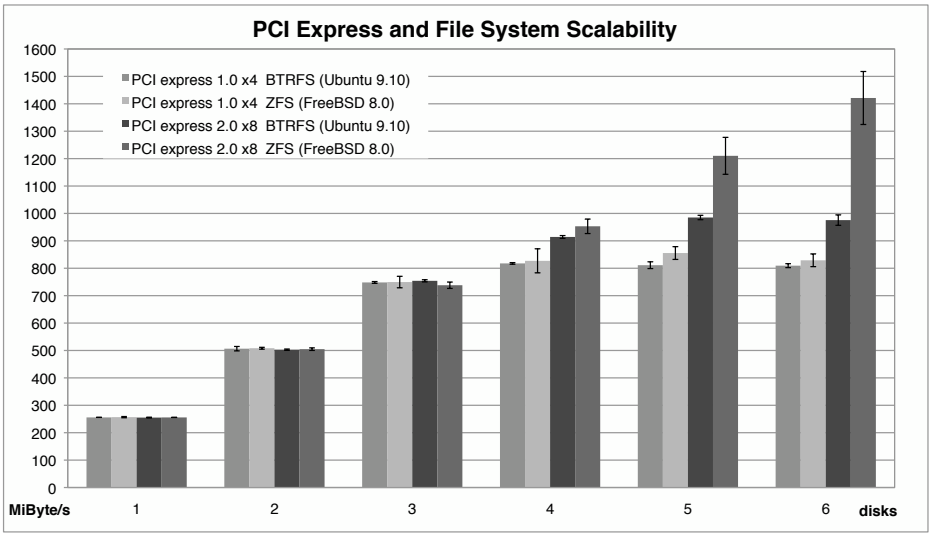

Figure 4 Measurements for PCI Express 1.0 and 2.0

9.4 Filesystems

Both operating system and file system are involved in the disk I/O operations, and are thus in part responsible for the disk I/O performance. All current I/O queues and file systems are designed with the limitations and particularities of mechanical hard disk drives in mind. For example, lots of effort has gone into tuning the physical location of blocks on a disk (to enhance sequential read performance) and in the queuing order for disk operations (minimising the movement of arms and platters). Neither of these enhancements is of any meaning to solid state disks. Instead, solid state disks require enhancements geared towards wear levelling and reducing the number of times a page has to be updated. The minimum write size on solid state disks is a page, which has a size of 4 kB. However, the minimum delete size is a block, which has a size of 512 kB or 128 pages. This means that in order to update a single page, the other 127 pages in the block also need to be rewritten to disk. The most significant enhancement for solid state disks is the introduction of the TRIM function in the DATA SET MANAGEMENT command of the ATA specification. The Operating System can use the TRIM function to signal to the SSD firmware which data blocks have been deleted by the file system. The helps the SSD firmware to manage the data blocks on the disk more efficiently. The performed tests examined the following file systems:

- Ext4
- Btrfs
- Nilfs2
- XFS
- ZFS

File systems specifically designed for flash memory, such as JFFS2, UBIFS or LogFS, can not be used with SATA or ATA-based solid state disks; they can only be used on raw flash memory. All file systems were tested under Linux (Ubuntu 8.10, Linux 2.6.31), with the exception of ZFS, which was run under FreeBSD 8.01. The filesystems with the highest throughput were BTRFS on Linux and ZFS on FreeBSD. The results are show in figure 5.

---

[1] ZFS is not supported in the Linux kernel due to a licensing incompatibility

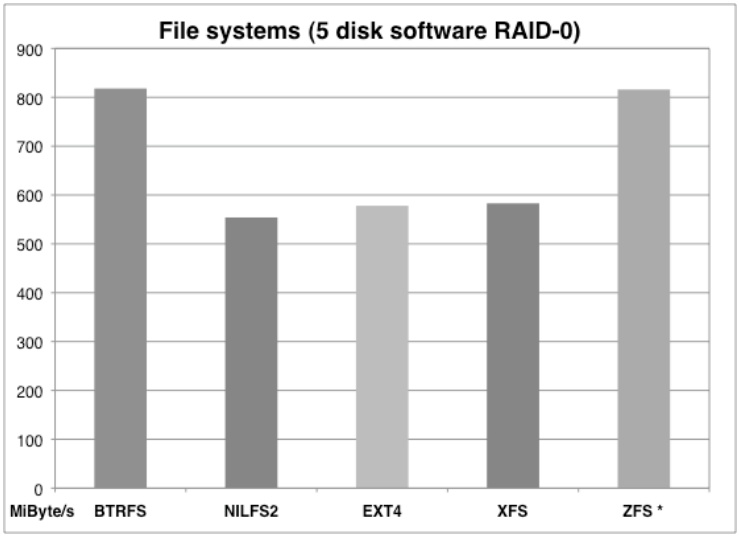

Figure 5 Read performance with different filesystems

10. PCI Express Data Rates

Figure 1 shows that all I/O connections to and from the CPU pass through a 6.4 GT/s Quick Path Interconnect (QPI). The Intel 5520 chipset handles the I/O and supports 36 PCI Express 2.0 lanes. Figure 7 shows how these lanes can be grouped to form the various x2, x4, x8 and x16 slots (see also figure 2-2 of [6]).

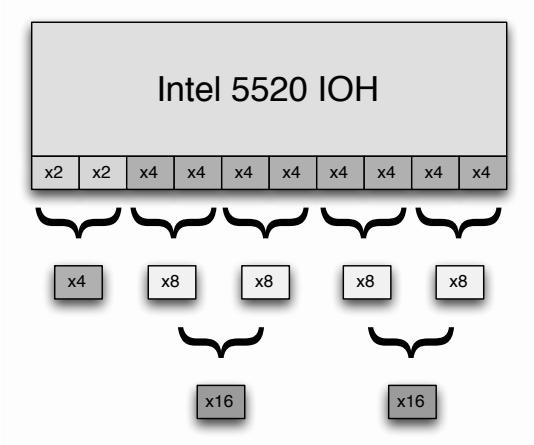

Figure 6 Intel 5520 chipset PCI Express lane combinations

In the T610 server 32 of these 36 lanes are configured as two x8 PCI Express 2.0 slots for full-height, full length cards, three x4 PCI Express 2.0 slots for full-height, half-length cards and one x4 PCI Express 2.0 slot for the PERC controller. PCI Express 1.0 has a clock of 2.5 GHz and operates at 2.5 GT/s. For data encoding the 10b/8b algorithm is used, which results in an effective data rate of 250 MB/s. For PCI Express 2.0 the clock rate was doubled to 5.0 GHz, resulting in 5.0 GT/s and 500 MB/s effective data rate. This gives the following effective data rates for the various PCI Express slots, as shown in table 1.

	X4	X8
PCI Express 1.0	8 Gbit/s	16 Gbit/s
PCI Express 2.0	16 Gbit/s	32 Gbit/s

Table 1 Effective data rates of PCI Express slots

The two x8 PCI Express 2.0 slots have an effective data rate of 32 Gbit/s and the three x4 PCI Express 2.0 slots have an effective data rate of 16 Gbit/s. This is only true when PCI Express 2.0 cards are used. When using PCI Express 1.0 cards the effective data rate is cut by half.

11. Network I/O Tests

A series of tests using multiple 10 Gbit/s network cards was performed in order to investigate possible bottlenecks when using multiple NICs simultaniously. For these tests SARA had access to seven 10GE NICs of three different types [11]:

- one 10GE SR NIC (Dell E15729)
- four 10GE UTP NICs (Intel E10G41AT2)
- two dual port 10GE SFP+ NICs (Intel E10G42AFDA)

All seven cards clock at 2.5 GHz2. This means that we can reach 10 Gbit/s only in the two PCI Express x8 slots, which have a maximum data rate of 16 Gbit/s. The three PCI Express x4 slots are limited to a maximum data rate of 8 Gbit/s when clocking at 2.5 GHz. The following tests were done with two 10GE UTP NICs in the two PCI Express x8 slots (slot 2 and slot 3) and a 10GE UTP NIC in a PCI Express x4 slot (slot 1). The tests were done with Ubuntu 9.10 with a Linux 2.6.31 kernel. We used the following kernel tuning parameters in /etc/sysctl.conf:

```
net.core.rmem_max = 102400000
net.core.wmem_max = 102400000
net.core.rmem_default = 524287
net.core.wmem_default = 524287
net.core.optmem_max = 524287
net.core.netdev_max_backlog = 300000
```

All tests were done using UDP traffic and an MTU of 9000 bytes. Three iperf streams were used, one for each of the three interfaces. The following arguments were used:

```
sender$ iperf -u -i 1 -t 300 -c 10.20.20.133 -l 63K -w 200K -b 10000M
receiver$ iperf -u -s -B 10.20.20.133 -l 63K -w 200K
```

The results are presented in table [2].

	Slot 1	Slot 2	Slot 3	total
No CPU Pinning	6.74 Gbit/s	9.24 Gbit/s	9.23 Gbit/s	25.21 Gbit/s
With CPU Pinning	6.91 Gbit/s	9.93 Gbit/s	9.93 Gbit/s	26.77 Gbit/s

Table 2 UDP Throughput

2 despite the branding as a PCI Express 2.0 card

This shows that SARA was able to reach full 10G throughput on both PCI Express x8 slots and slightly less than 7 Gbit/s on the PCI Express x4 slot. This slot has a theoretical maximum data rate of 8 Gbit/s. The effect on CPU usage was also investigated. The server has a single Xeon 5550 processor, which has 4 cores (and 8 threads). When the iperf processes were started without pinning them to a particular core, the CPU usage was as shown in table [3].

ID	User	System	Nice	Idle	IOwait	Hardware IRQ	Software interrupt	Steal time
CPU 0	2.0	60.8	0.0	12.0	0.0	0.7	24.6	0.0
CPU 1	2.0	4.6	0.0	91.4	0.0	0.0	2.0	0.0
CPU 2	0.0	3.9	0.0	96.1	0.0	0.0	0.0	0.0
CPU 3	0.0	2.6	0.0	97.4	0.0	0.0	0.0	0.0
CPU 4	0.0	3.1	0.0	96.2	0.0	0.0	0.6	0.0
CPU 5	0.7	0.3	0.0	99.0	0.0	0.0	0.0	0.0
CPU 6	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
CPU 7	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0

Table 3 CPU usage without pinning

When using CPU pinning with *taskset*, this resulted in a CPU usage as shown in table [4].

ID	User	System	Nice	Idle	IOwait	Hardware IRQ	Software interrupt	Steal time
CPU 0	0.3	0.0	0.0	99.0	0.0	0.0	0.7	0.0
CPU 1	0.6	34.6	0.0	9.8	0.0	0.0	5.1	0.0
CPU 2	2.9	31.0	0.0	59.6	0.0	0.0	6.6	0.0
CPU 3	0.0	19.2	0.0	76.3	0.0	0.0	4.5	0.0
CPU 4	2.0	0.0	0.0	98.0	0.0	0.0	0.0	0.0
CPU 5	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
CPU 6	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0
CPU 7	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0

Table 4 Pinning each iperf to a specific core

The idea of pinning is to force the distribution of the processes over the cores in the most efficient way with respect to CPU load and possible I/O bottlenecks [12].

Linux assigns CPU 0 and CPU 4 to the two threads of core 0, CPU 1 and CPU 5 to core 1, CPU 2 and CPU 6 to core 2 and CPU 3 and CPU 7 to core 3. The following commands were used to pin each of the three iperf processes to a particular core:

```
$ taskset 0x00000002 iperf ....
$ taskset 0x00000004 iperf ....
$ taskset 0x00000008 iperf ....
```

This pins the first iperf to CPU 1 (core 1), the second iperf to CPU 2 (core 2) and the third iperf to CPU 3 (core 3).
We also did a test with six 10GE ports. Figure 7 shows the results when using 6 UDP streams without

pinning the iperf processes to a core.

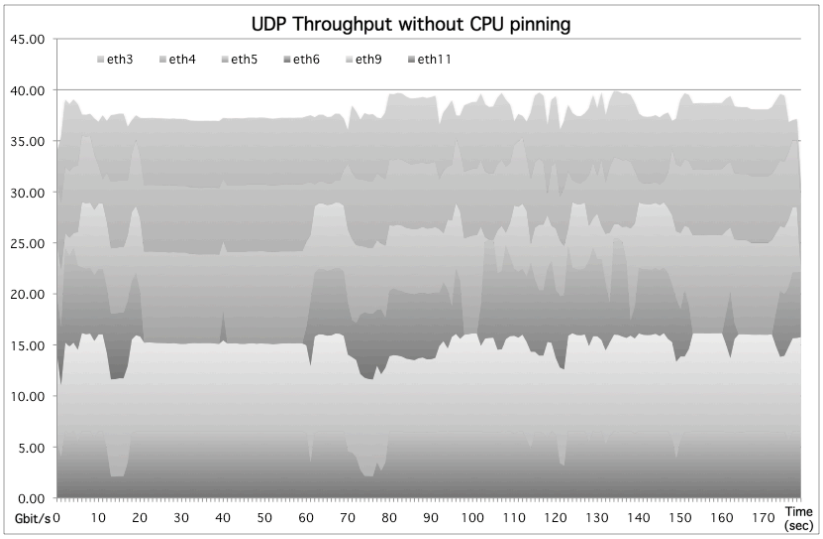

Figure 7 Six UDP streams without CPU pinning

We repeated the tests with pinning the iperf processes to a core. Figure 8 shows these results.

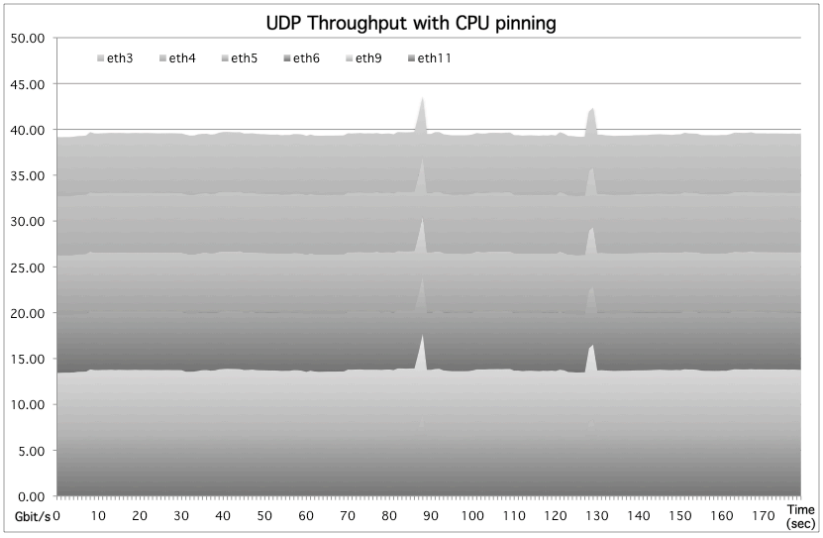

Figure 8 Six UDP streams with CPU pinning

12. Conclusions

The purpose of the work described in this document was to investigate bottlenecks in an off the shelf server when trying to transfer data from disk to network with higher than 10 Gbit/s speeds. For this work SARA used a single processor Intel Xeon~5550 based server
with Intel 5520 chipset and various 10GE NICs. For the disk I/O tests six Intel X25-M G2 Solid State Disks were used. SARA was able to reach a network I/O throughput of 26.77 Gbit/s with three 10GE NICs and a disk I/O throughput of 8.22 Gbit/s with 6 SSD disks.

Based on these results, SARA expects to be able to reach 40 Gbit/s of network I/O with two dual port PCI Express 2.0 x8 NICs. As expected, getting the data from disk with this speed is probably going to be more of a challenge. This probably requires 5 to 6 PCI Express 2.0 x8 RAID controller cards and 40 to 50 SSD disks. Finding a motherboard with 7 to 8 PCI Express 2.0 x8 slots may be difficult. The PCI Express I/O turned out the major cause of lower than expected thoughputs. Although the various

10GE NICs were advertised as PCI Express 2.0 cards, it turned out they were all using 2.5 GHz clocking, which resulted in PCI Express 1.0 data rates. Therefore, it was important to insert these cards in one of the two x8 slots in order to reach 10 Gbit/s throughput. The x4 slots are limited to 8 Gbit/s data rate when using 2.5 GHz clocking. With the SSD disk I/O tests we also experienced the effect of PCI Express slots with limited data rates. With the PERC RAID controller in a x4 slot the throughput leveled off at around 6.8 Gbit/s. With the PERC RAID controller in a x4 slot the leveling off accured at around 8.2 Gbit/s. At this moment, we are not sure why there is still leveling off in a x8 slot. The data rate of a PCI Express 2.0 x8 slot is 16 Gbit/s. In the tests that SARA has done so far, there was no evidence that the processor or memory was a bottleneck, although pinning processes to particular cores might be needed to take full advantage of the multi-core architecture.

13. Acknowledgements

This work was partly funded by the GigaPort3 project. Pieter de Boer helped with the architecture analysis. Igor Idziejczak and Mark Meijerink helped with network tests.
Daan Muller and Sebastian Carlier performed the original SSD tests described in section 9.

Appendix A

Specifications of the Dell T610 server that was used for the solid state disk throughput tests.

- Intel Xeon x5550 2.66GHZ
- 6 GB (3 x 2 GB) DDR3 @ 1333 MHz
- 2x 2.5" 7200 RPM 250 GB SATA Hard Disks
- 6x 2.5" 160 GB Intel X25-M G2 Solid State Disks
- 1x PERC 6/i SAS RAID Controller Card 256MB PCIe
- 1x 10GE SR NIC
- 4x 10GE UTP NICs
- 2x dual port 10GE SFP+ NICs

References

[1] B. Jeong, L. Renambot, R. Jagodic, R. Singh, J. Aguilera, A. Johnson, J. Leigh:
"High-Performance Dynamic Graphics Streaming for Scalable Adaptive Graphics Environment",
Proceedings of SC06, Tampa, FL, 2006.
http://www.evl.uic.edu/files/pdf/SAGE-SC06-final.pdf
[2] B. Stolk, P. Wielinga: "Building a 100 Mpixel graphics device for the OptIPuter",
Future Generation Computer Systems 22 (8), 972-975, 2006
[3] Multiple Interfaces (mif) working group
http://www.ietf.org/dyn/wg/charter/mif-charter.html
[4] Multipath TCP (mptcp) working group
http://www.ietf.org/dyn/wg/charter/mptcp-charter.html
[5] Dell PowerEdge T610 Technical Guidebook
https://noc.sara.nl/wiki/images/6/62/Server-poweredge-t610-tech-guidebook.pdf
[6] Intel 5520 Chipset and Intel 5500 Chipset Datasheet
http://www.intel.com/Assets/PDF/datasheet/321328.pdf
[7] S. Carlier, D. Muller: "SSD Performance",
UvA SNE Master project 2010
http://staff.science.uva.nl/~delaat/sne-2009-2010/p30/report.pdf

[8] T. Yoshii, C. Black, and S. Chahal: “Solid-State Drives in the Enterprise: A Proof of Concept”, Intel white paper
http://download.intel.com/it/pdf/Solid_state_drives_in_Enterprise.pdf
[9] G. Gasior: “Intel's X25-E Extreme solid-state drive”
http://techreport.com/articles.x/15931
[10] Intel X-25M SSD disks
http://www.intel.com/design/flash/nand/mainstream/index.htm
[11] Intel Ethernet Server and Desktop PCI Express Adapters
http://download.intel.com/network/connectivity/products/prodbrf/252454.pdf
[12] V. Vishwanath, J. Leigh, T. Shimizu, S. Nam, L. Renambot, H. Takahashi, M. Takizawa, O. Kamatani: “The Rails Toolkit (RTK) - Towards End-System Topology-Aware High End Computing”, Proceedings of the 4th IEEE International Conference on e-Science,
12/07/2008 - 12/12/2008
http://www.evl.uic.edu/files/pdf/RTK_e-Science08_Vishwanath.pdf

Biographies

Ronald van der Pol is a network researcher at SARA. He has been working in the field of Education and Research Networks for more than twenty years. His former employers include the VU University in Amsterdam, SURFnet and NLnet Labs. At SURFnet he specialized on IPv6 and is co-author of several RFCs. In recent years his focus is on optical networks, especially hybrid network monitoring, end-to-end performance and network access for demanding applications. He holds masters degrees in both Physics and Computer Science.

Freek Dijkstra received his M.Sc. in applied physics from the Utrecht University in 2002, and received his Ph.D. in Computer Science at the University of Amsterdam in June 2009 on the topic of *Path finding in multi-layer transport networks*. Dijkstra is currently
working as network researcher at SARA, were he develops standards and software for use in innovative networks. His interests are topology descriptions and monitoring of multi-layer and multi-domain networks. He is co-chairing the Network Markup Language working
group in the Open Grid Forum (OGF).