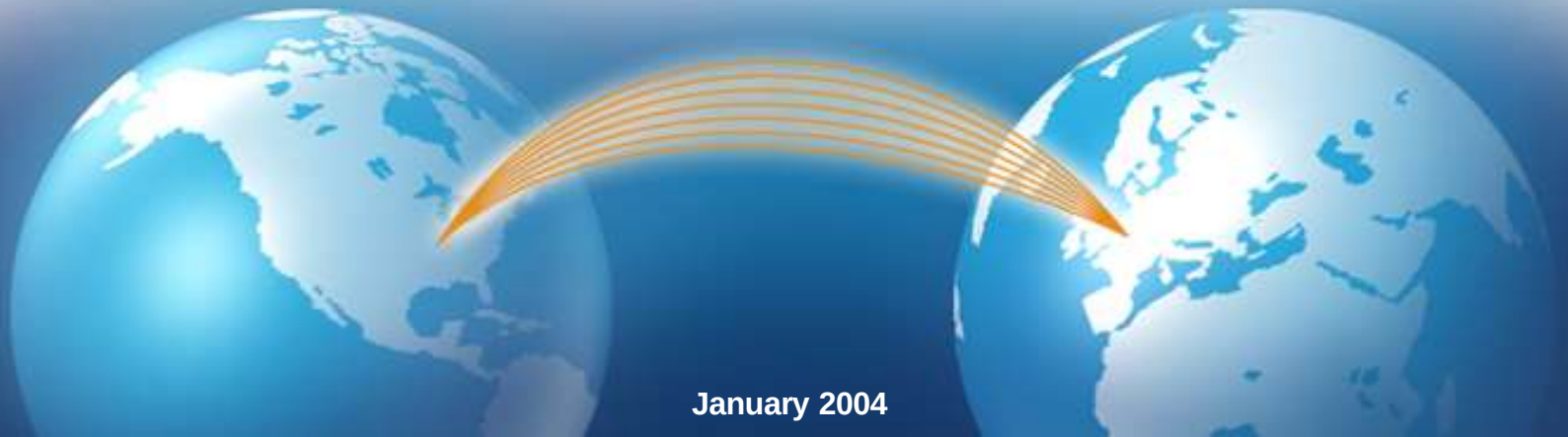
IPv6 Land Speed Record

Edoardo Martelli, CERN
DataTAG



January 2004

The DataTAG project

The DataTAG project has created a large-scale intercontinental Grid testbed involving the European DataGrid project, several national projects in Europe, and related Grid projects in the USA.

The project is exploring some forefront research topics like the design and implementation of advanced network services for guaranteed traffic delivery, transport protocol optimisation, efficiency and reliability of network resource utilization, user-perceived application performance, middleware interoperability in multi domain scenarios, etc.

The DataTAG is a project co-funded by the European Union, the U.S. Department of Energy, and the National Science Foundation.The partners are CERN (project leader), INFN, INRIA, PPARC and UvA

Member Organizations

CERN

INFN
ISTITUTO NAZIONALE DI FISICA NUCLEARE

INRIA

PPARC

UNIVERSITEIT VAN AMSTERDAM

http://www.datatag.org/


DataTAG

Partners


CALIFORNIA INSTITUTE OF TECHNOLOGY
1891


NSF


DEPARTMENT OF ENERGY
UNITED STATES OF AMERICA


DANTE


SURFnet

Internet 2 Land Speed Record competition

A primary goal of Internet2 is to deploy and demonstrate advanced networking capabilities that will make their way into the global commodity Internet. The Internet2 Land Speed Record (I2-LSR) competition for the highest-bandwidth, TCP based, end-to-end networks data transfer is an open and ongoing contest. The aim of this competition is to encourage people to deploy new applications and protocols able to take advantage of the very high speed networks available today.

http://lsr.internet2.edu/

LSR contest rules

- [...]
- A data transfer must run for a minimum of 10 continuous minutes over a minimum terrestrial distance of 100 kilometers with a minimum of two router hops in each direction between the source node and the destination node across one of more operational and production-oriented high-performance research and educations networks.
- All data must be transferred end-to-end between a single pair of IP addresses using TCP/IP protocol code implementations of RFC 793 and RFC 791.
- Instances of all hardware units and software modules used to transfer contest data on the source node, the destination node, the links, and the routers must be offered for commercial sale or as open source software.

TVP/IPv6 single stream LSR

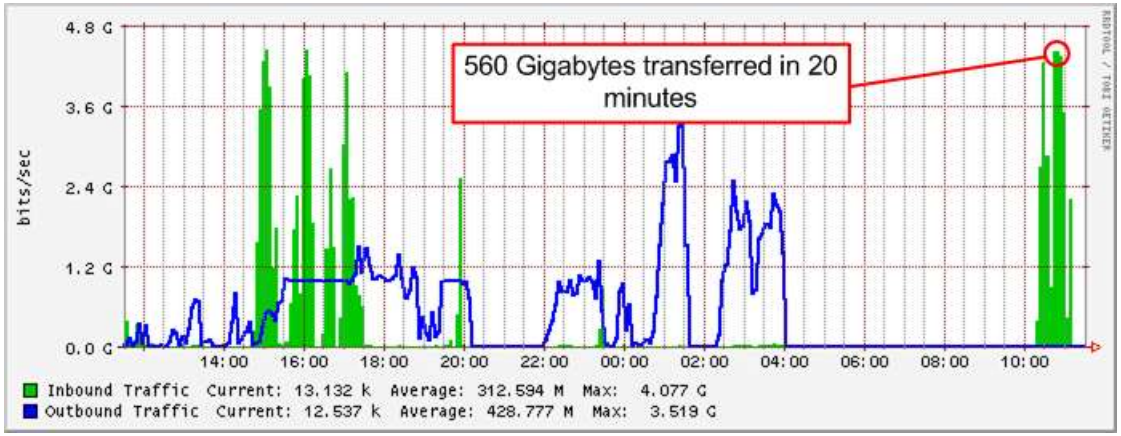
On November 18, 2003, Caltech and CERN transferred 560 Gigabytes of data in 20 minutes between Geneva and the Caltech stand at SuperComputing 2003 in Phoenix, through the LHCnet/DataTag, Abilene and SciNet backbones, using a single TCP/IPv6 stream.

Distance: 11539 km

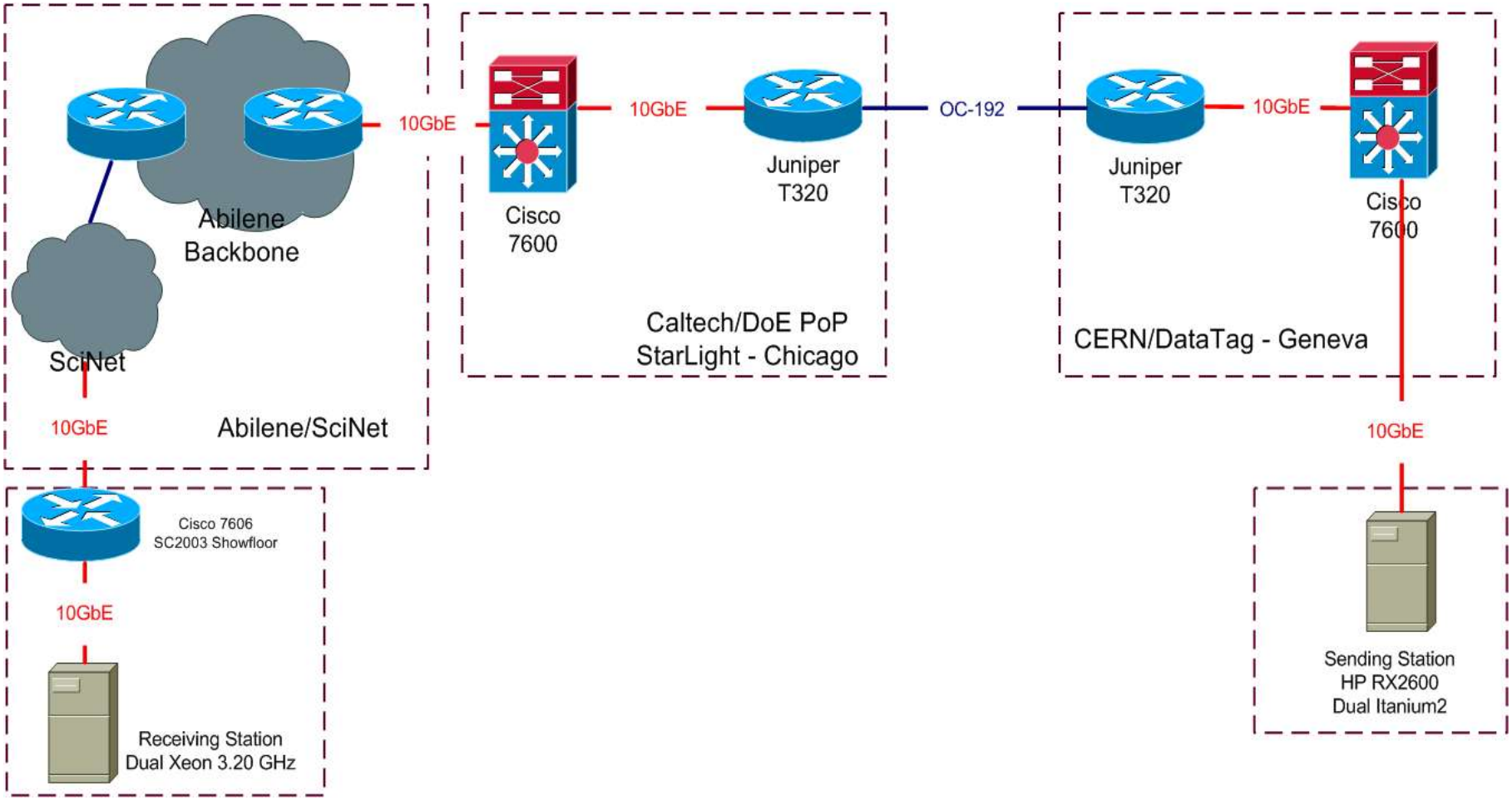
Data transferred: 560 Gbytes

Average speed (over 20 minutes): 4.00 Gbps

Record submitted: 46,156,000,000,000,000 meters-bits/sec



Network map


Abilene
Backbone
SciNet
10GbE
Abilene/SciNet
10GbE
Cisco
7600
10GbE
Juniper
T320
Caltech/DoE PoP
StarLight - Chicago
OC-192
Juniper
T320
10GbE
Cisco
7600
CERN/DataTag - Geneva
10GbE
Sending Station
HP RX2600
Dual Itanium2
Cisco 7606
SC2003 Showfloor
10GbE
Receiving Station
Dual Xeon 3.20 GHz

Traceroute

```
[root@oplapro28]# traceroute6 2001:468:1f07:1323::5
traceroute to 2001:468:1f07:1323::5 (2001:468:1f07:1323::5) from 2001:1458:e000:100::10, 30
hops max, 24 byte packets
 1  2001:1458:e000:100::1           (2001:1458:e000:100::1) 0.688 ms   (Juniper T320 Geneva)
 2  2001:1458:e000:1::1             (2001:1458:e000:1::1)   127.237 ms (Juniper T320 Chicago)
 3  2001:1458:e000:1:1::1           (2001:1458:e000:1:1::1) 127.383 ms (Abilene Core Router)
 4  iplsng-chinng.abilene.ucaid.edu (2001:468:ff:f12::2)    *          (Abilene Core Router)
 5  iplsng-kscyng.abilene.ucaid.edu (2001:468:ff:1213::2)   140.653 ms (Abilene Core Router)
 6  snvang-kscyng.abilene.ucaid.edu (2001:468:ff:1317::2)   185.082 ms (Abilene Core Router)
 7  losang-snvang.abilene.ucaid.edu (2001:468:ff:1417::1)   *          (Abilene Core Router)
 8  2001:468:ff:14c3::2             (2001:468:ff:14c3::2)   192.436 ms (SciNet Router)
 9  2001:468:1f07:1323::5           (2001:468:1f07:1323::5) 192.784 ms (End host in Phoenix)
```

Geneva - Chicago (LHCnet/Datatag):	7067 km
Chicago - Indianapolis (Abilene):	263 km
Indianapolis - Kansas City (Abilene)	727 km
Kansas City - Sunnyvale (Abilene):	2403 km
Sunnyvale - Los Angeles (Abilene):	489 km
Los Angeles - Phoenix (SciNet):	590 km
Total:	**11539 km**

Equipment - End Systems

At CERN: HP RX2600 server
- Dual Intel Itanium2 1.5 Ghz CPU
- 4GB RAM
- Intel PRO/10GbE LR network adapter
- Linux kernel 2.6.0-test5

At Caltech: PC server
- Dual Intel Xeon 3.20 Ghz CPU
- Supermicro X5DPE motherboard (E7501 chipset)
- 2 GB RAM
- Intel PRO/10GbE LR network adapter
- Linux kernel 2.4.22

Network equipment

CERN:

Juniper T320 Router

Chicago Starlight:

Juniper T320 Router

Cisco 7609 Switch

Abilene Backbone:

Juniper T640 Routers

SciNet Backbone:

Juniper T320 Router

SC2003:

Cisco 7609 Switch

IPv6 vs IPv4

In the same period and with almost the same setup (the distance was shorter, but the endstations were the same) also the TCP/IPv4 LSR was set. But IPv4 was better:

IPv4

Distance: 10949 KM
Data transferred: 2.3 Terabytes
Average speed (over 60 minutes): **5.64 Gbps**
Record submitted: 61,752,360,000,000,000 meters-bits/sec

IPv6

Distance: 11539 km
Data transferred: 560 Gbytes
Average speed (over 20 minutes): **4.00 Gbps**
Record submitted: 46,156,000,000,000,000 meters-bits/sec

Sender CPUs at 100% in both cases. Linux TCP/IPv4 stack more efficient.

Background

Responsiveness to packet losses is proportional to the square of the RTT (Round Trip Time): R=C*(RTT**2)/2*MSS (where C is the link capacity and MSS is the max segment size). This makes very difficult to take advantage of full capacity over long-distance WAN: this is not a real problem for standard traffic on a shared link, but a serious penalty for long distance transfers of large amount of data.

The TCP stack was designed a long time ago and for much slower networks: from the above formula, if C is very small, the responsiveness is kept low enough for any terrestrial RTT: modern, fast WAN links are "bad" for TCP performance, since TCP tries to increase its window size until something breaks (packet loss, congestion); then restarts from a half of the previous value until it breaks again. This gradual approximation process takes very long over long distance and degrades performance.

Knowing a priori the available bandwidth,TCP is prevented from trying larger windows by restricting the amount of buffers it may use: without buffers, it won't try to use larger windows and packet losses due to congestion of the link can be avoided.

The product C*RTT yields the optimal TCP window size for a link of capacity C, so it's sufficient to allocate just enough buffers to let TCP reach the maximum performance from the existing bandwidth, without attempting to go beyond.

End systems settings

For Linux 2.4.22:

set mmrbc to 4k reads, modify only Intel 10GbE device IDs

setpci -d 8086:1048 e6.b=2e

set the MTU (max transmission unit) - it requires your switch and clients to change too!

set the txqueuelen

ifconfig eth2 mtu 9000 txqueuelen 50000 up

call the sysctl utility to modify /proc/sys entries

net.ipv4.tcp_timestamps = 0 # turns TCP timestamp support off, default 1, reduces CPU use

net.ipv4.tcp_sack = 0 # turn SACK support off, default on

on systems with a VERY fast bus -> memory interface this is the big gainer

net.ipv4.tcp_rmem = 10000000 10000000 10000000 # sets min/default/max TCP read buffer, default 4096 87380 174760

net.ipv4.tcp_wmem = 10000000 10000000 10000000 # sets min/pressure/max TCP write buffer, default 4096 16384 131072

net.ipv4.tcp_mem = 10000000 10000000 10000000 # sets min/pressure/max TCP buffer space, default 31744 32256 32768

net.core.rmem_max = 524287 # maximum receive socket buffer size, default 131071

net.core.wmem_max = 524287 # maximum send socket buffer size, default 131071

net.core.rmem_default = 524287 # default receive socket buffer size, default 65535

net.core.wmem_default = 524287 # default send socket buffer size, default 65535

net.core.optmem_max = 524287 # maximum amount of option memory buffers, default 10240

net.core.netdev_max_backlog = 300000 # number of unprocessed input packets before kernel starts dropping them, default 300

Before all new connections: sysctl -w net.ipv4.route.flush=1

iperf

Iperf (http://dast.nlanr.net/Projects/Iperf/) version 1.7.0 was used to generate traffic and measure performance.

Sender:

```
[root@oplapro28 iperf-1.7.0]# ./iperf -B  2001:1458:e000:100::10 -c 2001:468:1f07:1323::5 -i10
-w250m   -t 1200 -l9000
------------------------------------------------------------
Client connecting to 2001:468:1f07:1323::5, TCP port 5001
Binding to local address 2001:1458:e000:100::10
TCP window size:  500 MByte (WARNING: requested  250 MByte)
------------------------------------------------------------
[  3] local 2001:1458:e000:100::10 port 32785 connected with 2001:468:1f07:1323::5 port 5001
....
[  3]  0.0-1201.0 sec   560 GBytes  4.00 Gbits/sec
```

Receiver:

```
[root@b7 iperf-1.7.0]# ./iperf -V -s -B 2001:468:1f07:1323::5 -i10 -t1200 -w250m -l9000
------------------------------------------------------------

Server listening on TCP port 5001
Binding to local address 2001:468:1f07:1323::5
TCP window size:  500 MByte (WARNING: requested  250 MByte)
------------------------------------------------------------
[  4] local 2001:468:1f07:1323::5 port 5001 connected with 2001:1458:e000:100::10 port 32785
....
[  4]  0.0-1201.2 sec   560 GBytes  4.00 Gbits/sec
```

Conclusions

Special tuning of end systems is needed (althought not in the TCP stack)

No serious performance penalty for IPv6 with respect to IPv4, but still some differences in the end system implementations

The bootleneck is now in the end systems and not in the network

Acknowledgements

California Institute of Technology (Caltech):

Harvey Newman <newman@hep.caltech.edu>
Steven Low <slow@caltech.edu>
Julian Bunn <julian@cacr.caltech.edu>
Suresh Singh <suresh@cacr.caltech.edu>
Yang Xia <yxia@hep.caltech.edu>
Sylvain Ravot <ravot@caltech.edu>
Dan Nae <Dan.Nae@cern.ch>

CERN:

Olivier Martin <olivier.martin@cern.ch>
Paolo Moroni <paolo.moroni@cern.ch>
Edoardo Martelli <edoardo.martelli@cern.ch>
Daniel Davids <daniel.davids@cern.ch>
Sverre Jarp (CERN OpenLab) <sverre.jarp@cern.ch>
Andreas Hirstius (CERN OpenLab) <andreas.hirstius@cern.ch>

Partners: Caltech, CERN

Sponsors: Intel, Cisco Systems, HP, T-Systems/DeutscheTelecom, DoE, NSF, European Union

Acknowledgements: CERN OpenLab, DataTag, StarLight