Internet data transfer record between CERN and California

Sylvain Ravot (Caltech)
Paolo Moroni (CERN)

Summary

- Internet2 Land Speed Record Contest
- New LSR
- DataTAG project and network configuration
- Establishing an LSR: hardware and tuning
- Conclusions
- Acknowledgements

Internet2 LSR Contest (I)

- From http://lsr.internet2.edu:

  "A minimum of 100 megabytes must be transferred 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 or more operational and production-oriented high-performance research and education networks."

  "The contest unit of measurement is […] bit-meters/second."

Internet2 LSR Contest (II)

- *"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 to all U.S. members of the Internet community by their respective vendors or developers prior to or immediately after winning the contest."*
- Award classes: single or multiple TCP streams, on top of IPv4 or IPv6
- Generally available networks and equipment, vs. lab prototypes

Former LSR

- TCP/IPv4 single stream
- By NIKHEF, Caltech and SLAC
- Established on November 19th 2002
- 10978 Km of network: Geneva-Amsterdam-Chicago-Sunnyvale
- 0.93 Gb/sec
- 10136.15 terabit-meters/second

Current LSR

- TCP/IPv4 single stream
- By Caltech, CERN, LANL and SLAC, within the DataTAG project framework
- Established on February 27th-28th 2003 by Sylvain Ravot (Caltech) using IPERF
- 23888 Terabit-meters/second
- 10037 Km of network: Geneva-Chicago-Sunnyvale (shorter distance than the former LSR)
- 2.38 Gb/sec (sustained: a Terabyte of data moved in an hour)

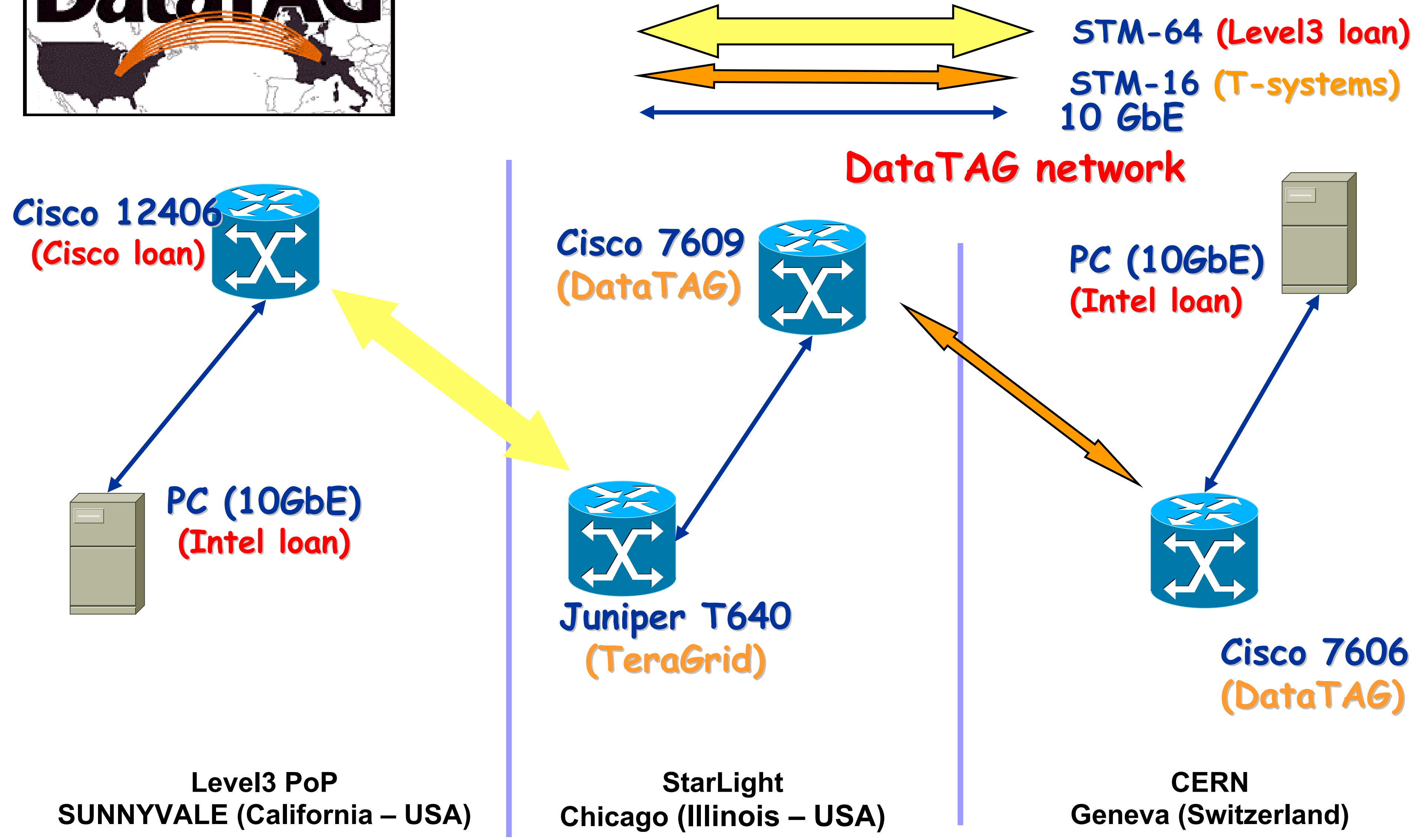
DataTAG project

- **Full project title: *"Research and technological development for a transatlantic GRID"***
- **IST project (EU funded), supported by the NSF and the DoE (Caltech): *http://www.datatag.org***
- **Partners: PPARC (UK), INRIA (FR), University of Amsterdam (NL), INFN (IT) and CERN (CH)**
- **Researchers also from Caltech, Los Alamos, SLAC and Canada**
- **Test-bed kernel: transatlantic STM-16 between Geneva (CERN) and Chicago (StarLight), with interconnected workstations at each side.**
- **Test-bed extensions provided by GEANT, SURFnet, VTHD and other partners, in Europe and North America**

DataTAG as test-bed for LSR

- Research on TCP as part of the DataTAG programme
- The Geneva-Chicago link was the main environment for the LSR
- Network extension made available: Chicago-Sunnyvale STM-64
- Router at Sunnyvale
- 10 GbE interfaces on DataTAG PCs

LSR network configuration



Establishing an LSR: hardware (I)

- No LSR without good hardware
- A lot of bandwidth: minimum 2.5 Gb/sec on the whole path (thanks to Level3 for the STM-64 on loan between Chicago and Sunnyvale)
- Powerful routers (Cisco 7600 and GSR, Juniper T640)
- Powerful Linux PCs on both sides
- Intel 10 GbE interfaces

Establishing an LSR: hardware (II)

- Linux PC at CERN:
  - Dual Intel® Xeon™ processors, 2.40GHz with 512K L2 cache
  - SuperMicro P4DP8-G2 Motherboard
  - Intel E700 chipset
  - 2 GB RAM,PC2100 ECC Reg. DDR
  - Hard drive: 1 x 140 GB - Maxtor ATA-133
  - On board Intel 82546EB dual port Gigabit Ethernet controller
  - 4U Rack-mounted server
- Linux PC at Sunnyvale:
  - Dual Intel® Xeon™ processors , 2.40GHz with 512K L2 cache
  - SuperMicro P4DPE-G2 Motherboard
  - 2 GB RAM, PC2100 ECC Reg. DDR
  - 2* 3ware 7500-8 RAID controllers
  - 16 Western Digital IDE disk drives for RAID and 1 for system
  - 2 Intel 82550 fast Ethernet
  - 2*SysKonnect Gigabit Ethernet card SK-9843 SK-NET GE SX
  - 4U Rack-mounted server
  - 480W to run 600W to spin up

Establishing an LSR: hardware (III)

- Intel 10 GbE interfaces: Intel Pro/10 GbE-LR
- Not yet commercially available when the LSR was set, but announced as commercially available shortly afterwards

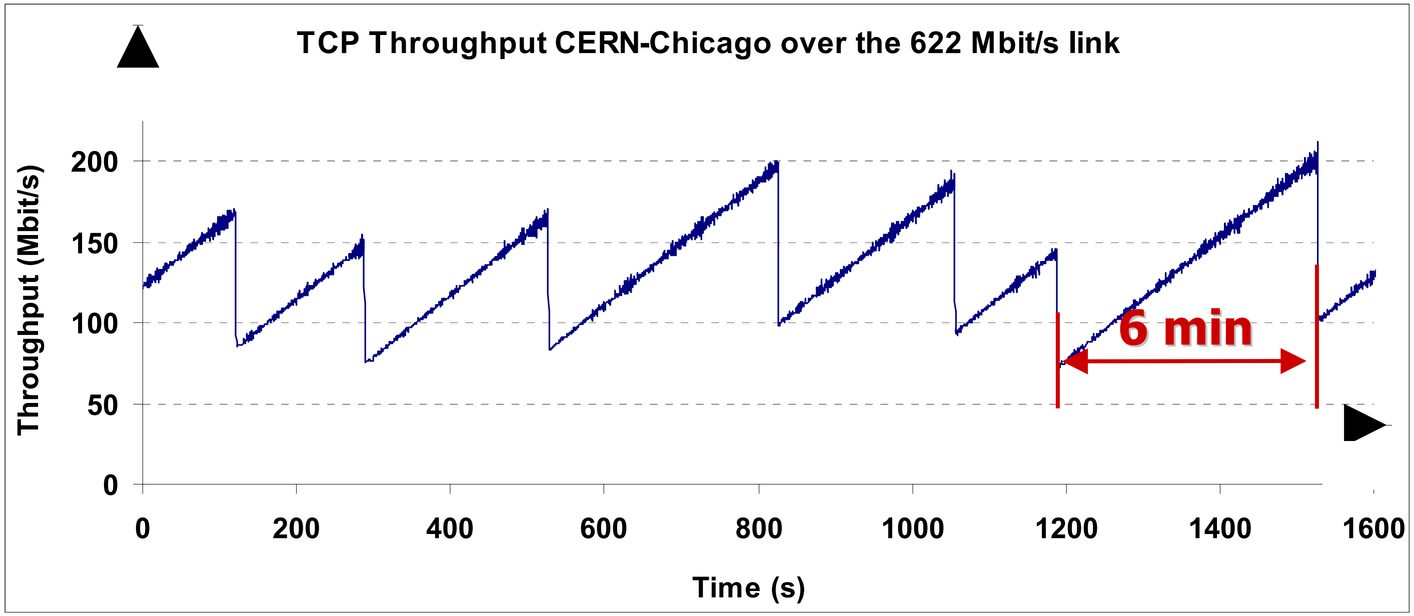
Establishing an LSR: standard tuning

- MTU set to 9000 bytes
- TCP window size increased from the Linux default of 64K: essential over long distance
- But standard Linux kernel (2.4.20)
- Standard tuning is not enough for LSR: why?

TCP WAN problems

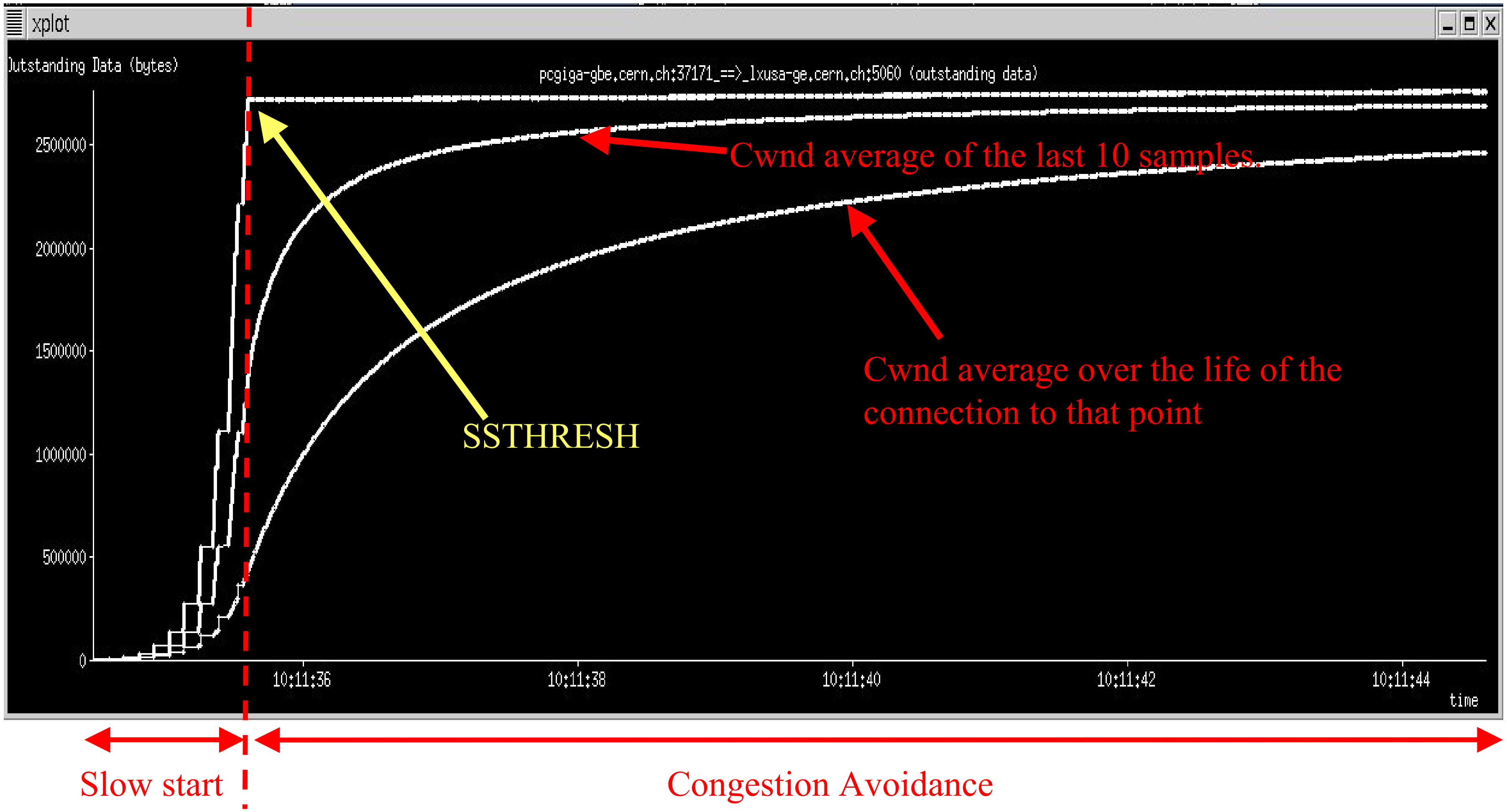
- Responsiveness to packet losses is proportional to the square of the RTT: R=C*(RTT**2)/2*MSS (where C is the link capacity and MSS is the max segment size). This makes it very difficult to take advantage of full capacity over long-distance WAN: not a real problem for standard traffic on a shared link, but a serious penalty for LSR
- Slow start mode is “too” slow using default parameters: they are good for standard traffic, but not for LSR

Example: recovering from a packet loss



- TCP reactivity
  - Time to increase the throughput by 120 Mbit/s is larger than 6 min for a connection between Chicago and CERN.
- Packet losses is a disaster for the overall throughput

Example: slow start vs. congestion avoidance



Establishing an LSR: what goes wrong

- Behaviour of the TCP stack: if C is very small, it keeps the responsiveness low enough for any terrestrial RTT. Therefore modern, fast WAN links are “bad” for TCP performance
- 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

Establishing an LSR: further tuning

- Knowing a priori the available bandwidth, prevent TCP 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 can be avoided
- The product C*RTT yields the optimal TCP window size for a link of capacity C
- So, allocate just enough buffers to let TCP squeeze the maximum performance from the existing bandwidth and nothing else

Further tuning: Linux implementation

- Tuning TCP buffers (numbers for STM-16):
  - echo “4096 87380 128388607” > /proc/sys/net/ipv4/tcp_rmem
  - echo “4096 65530 128388607” > /proc/sys/net/ipv4/tcp_wmem
  - echo 128388607 > /proc/sys/net/core/wmem_max
  - echo 128388607 > /proc/sys/net/core/rmem_max
- Tuning the network device buffers:
  - /sbin/ifconfig eth1 txqueuelen 10000
  - /sbin/ifconfig eth1 mtu 9000
- Both on sender and receiver

Even further tuning: Linux implementation

- TCP slow start mode vs. congestion avoidance mode is another performance penalty in the sender for the LSR
- On Linux (sender side only):
  - sysctl -w net.ipv4.route.flush=1
- This prevents TCP from using any previously cached window value, i.e. speeds up slow start mode and gets to congestion avoidance mode at exponential speed (otherwise the growth of the congestion window would start at half of some previously cached value)

IPERF parameters

- On the sender:
  - iperf -c 192.91.239.213 -i 5 -P 3 -w 40M -t 180
- On the receiver:
  - iperf-1.6.5 -s -w 128M
- IPERF is available at *http://dast.nlanr.net*

Conclusions: how useful in practice?

- The LSR result cannot be immediately translated into practical general-purpose recommendations: it relies on
  - some *a priori* knowledge (the physical link speed)
  - dedicated bandwidth
  - *ad hoc* TCP tuning: good for LSR, not for general-purpose traffic
- Nevertheless, work is ongoing for a more modern TCP stack: the new LSR demonstrates that fast WAN TCP is possible in practice, by tweaking TCP a bit

Conclusions: sustained throughput

- The LSR definition has only very limited provisioning for requiring sustained throughput (100 Megabytes are not much)
- However the achieved LSR shows that high <u>sustained</u> throughput is in principle possible with TCP over long distance
- Former results could sustain the throughput only for 40-60 seconds, before some TCP feedback mechanism kicked in

Other remarks

- The bottleneck for things like the LSR is now in the end hosts: no non-trivial tuning was needed on the network where the LSR was established
- Incidentally, although single-stream, the new LSR was also good enough to establish the new LSR for multiple IPv4 streams
- No TCP packet was lost during the LSR trial window
- Details of the new record are not published on http://lsr.internet2.edu yet

Acknowledgements: people

- Sylvain Ravot (working for Caltech at CERN)
- Wu-Chun Feng (LANL)
- Les Cottrell (SLAC)

DataTAG

Acknowledgements: industrial partners

Level(3) COMMUNICATIONS

CISCO SYSTEMS

intel.

Deutsche Telekom

Los Alamos
NATIONAL LABORATORY

DataTAG

Stanford
Linear
Accelerator
Center

Acknowledgements: organisations

Information Society
Technologies

CERN

CALIFORNIA INSTITUTE OF TECHNOLOGY
1891

STARLIGHT℠

TERAGRID

NSF

DEPARTMENT OF ENERGY
UNITED STATES OF AMERICA