CSC 7700: Scientific Computing Module B: Networks and Data Lecture 1: Networks Introduction

Dr. Andrei Hutanu

Administrative

- How are you doing on the assignment
- Any issues logging in to the sites?
- Any difficulties?

Administrative

- Module B: Networks and Data
- Two lectures focused on lower-level networking, then three lectures in November – higher level tools
  - Because of scheduling, but shouldn't be an issue
- https://wiki.cct.lsu.edu/sci-comp/Networks_and_Data
  - Linked from main page
  - Will have all important module information
- Office hours
  - Johnston Hall 350, Wed and Fri 3-4 PM (during lectures and coursework weeks, list on wiki)

Administrative

- Also AIM: handrei1
- Also e-mail: sci-comp-instructors@cct.lsu.edu
- Grading (20% for the module)
  - Assignments/Coursework : half (10% of the final grade)
    - One or two assignments (will decide on the second assignment for the second part in November after some feedback)
    - Exam : other half (10% of the final grade), from all lectures in the module
- Have technical documentation on the wiki
- Also research papers (1-2/lecture)

Why Networks?

- Computer networks are the most recent fundamental human invention (a selection)
  - Fire (250000 years ago) , Agriculture (10000 BC), Wheel (4000 BC), Printing press (1450), Steam Engine (1712), Radio (1896), Penicillin (1928), Computer (1939), Internet (1969), WWW (1991)
  - We're now all connected
  - Still adapting as a society, outsourcing, living in multiple worlds (some virtual), collaboration
  - Fundamental flattener

Not getting much attention

- As opposed to computers
  - Think about how much time you spent studying computers versus studying networks
- History of not going into networking details
  - Networks are just there ..
- Networks are fundamental to modern high-performance scientific computing as well as our personal lives

Example applications

- Social connectivity
  - e-mail, audio, video conferencing, IM, telephony
- News and media
  - newspapers, radio, tv
- Background on size/speed: orders of magnitude
  - Broadband network provider: few Mbps->10-20 Mbps (bits per second, b = bit, B = byte = 8 bits)
  - High-speed mobile: hundreds of kbps -> few Mbps
  - Dial-up: tens of kbps (56kbps)

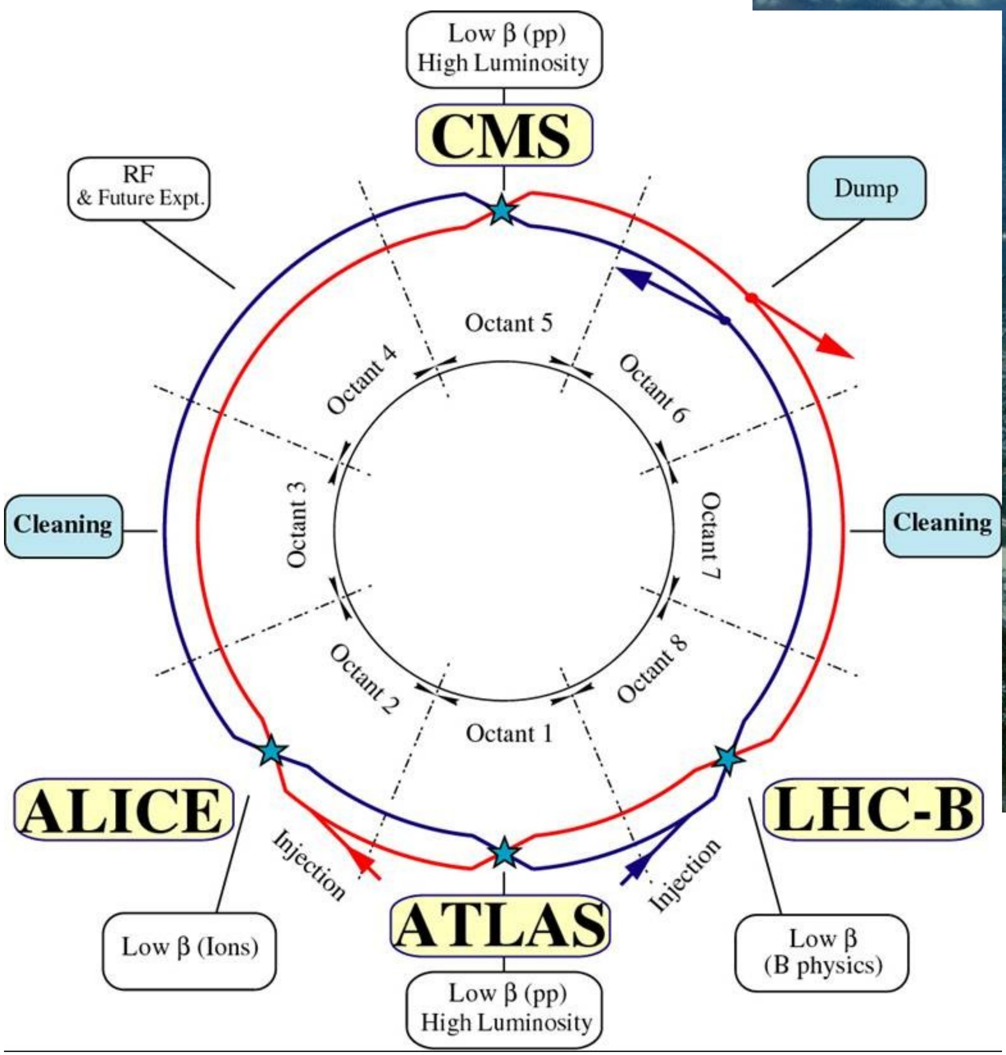
Data size

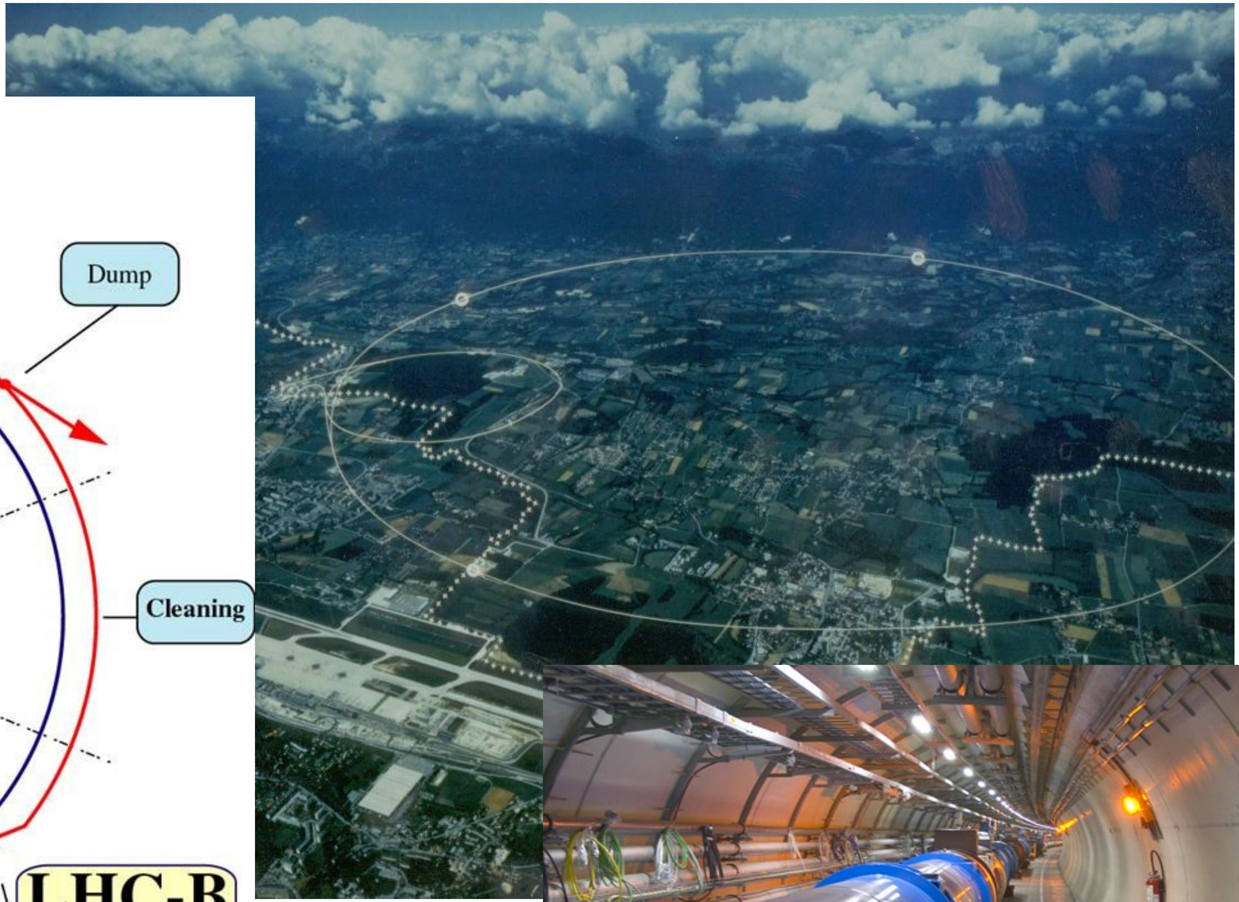
- CD: 7-800 MBytes, DVD: 4.7 GB single layer, 8.5 GB dual layer.
- Blu-ray: 25 GB single layer, 50 GB dual layer
- Bitrate
  - Audio: 8kbps phone (cell), 64 kbps (fixed), 1.4 Mbps CD
  - Digital TV: 2-3 Mbps, HDTV: ~ 15 Mbps
  - DVD: 5-10 Mbps, Blu-ray: up to 40 Mbps

Scientific Computing

- Instruments
  - LIGO (1 TB/day): http://www.ligo-la.caltech.edu/
  - Particle accelerators (LHC, Fermilab): http://lhc.web.cern.ch/lhc/
  - e-Very-Long-Baseline Interferometry (e-VLBI)
  - medical instruments (3D scans)
  - Supercomputer centers
    - Simulations (TB/simulation easy .. PB becoming the norm)
- Need networks to move the data around

LHC (http://lhc.web.cern.ch/lhc/)





© CERN, DOE

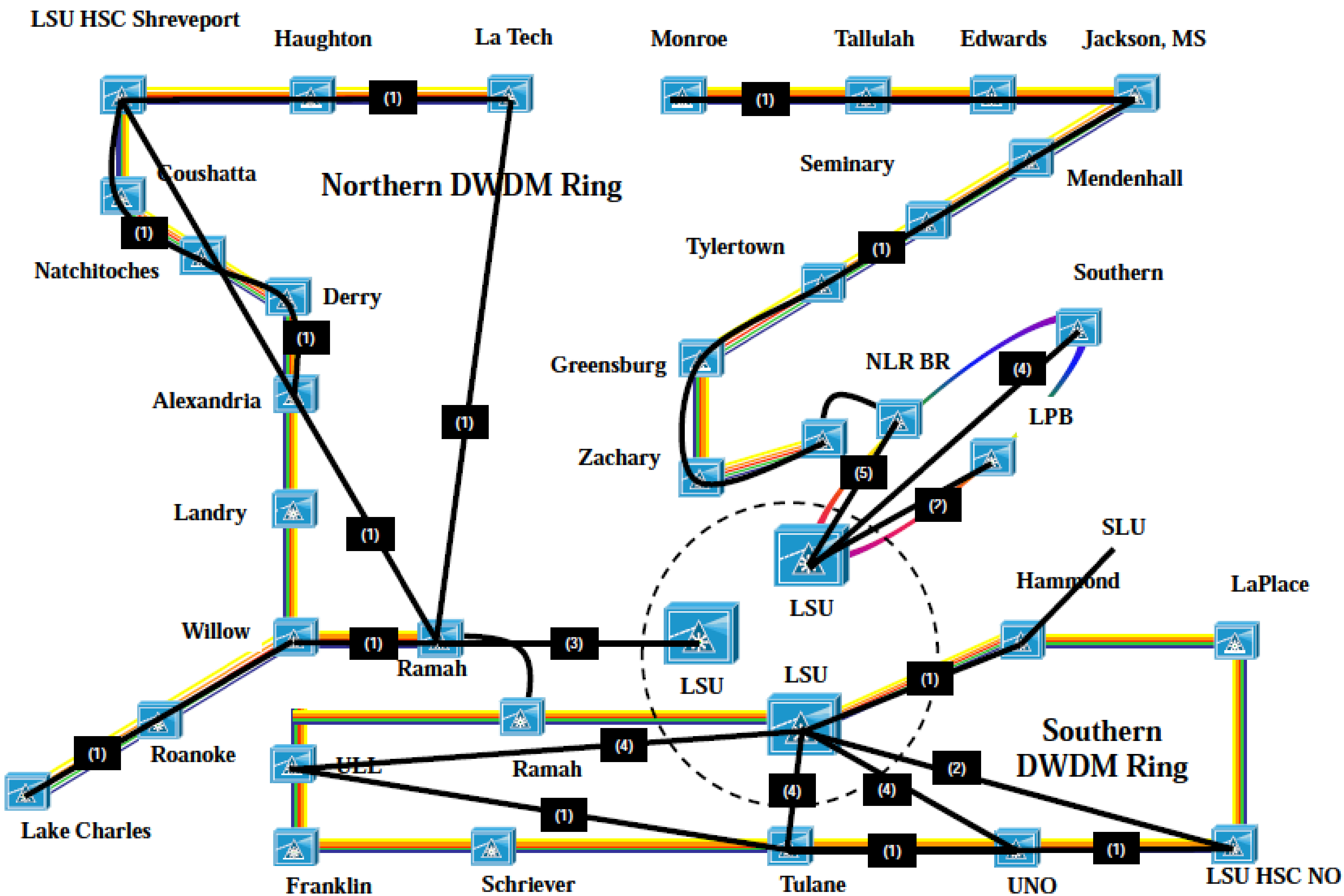
Network usage

- Distributing the estimated 15 PB generated/year
- Tier 1 sites (two in US) .. then beyond
- **Large file transfers**
  - Up to tens of terabytes a day
  - Need 10-30 Gbps network bandwidth
  - 30 Gbps fully utilized means ~ 300 TB/day
  - archival

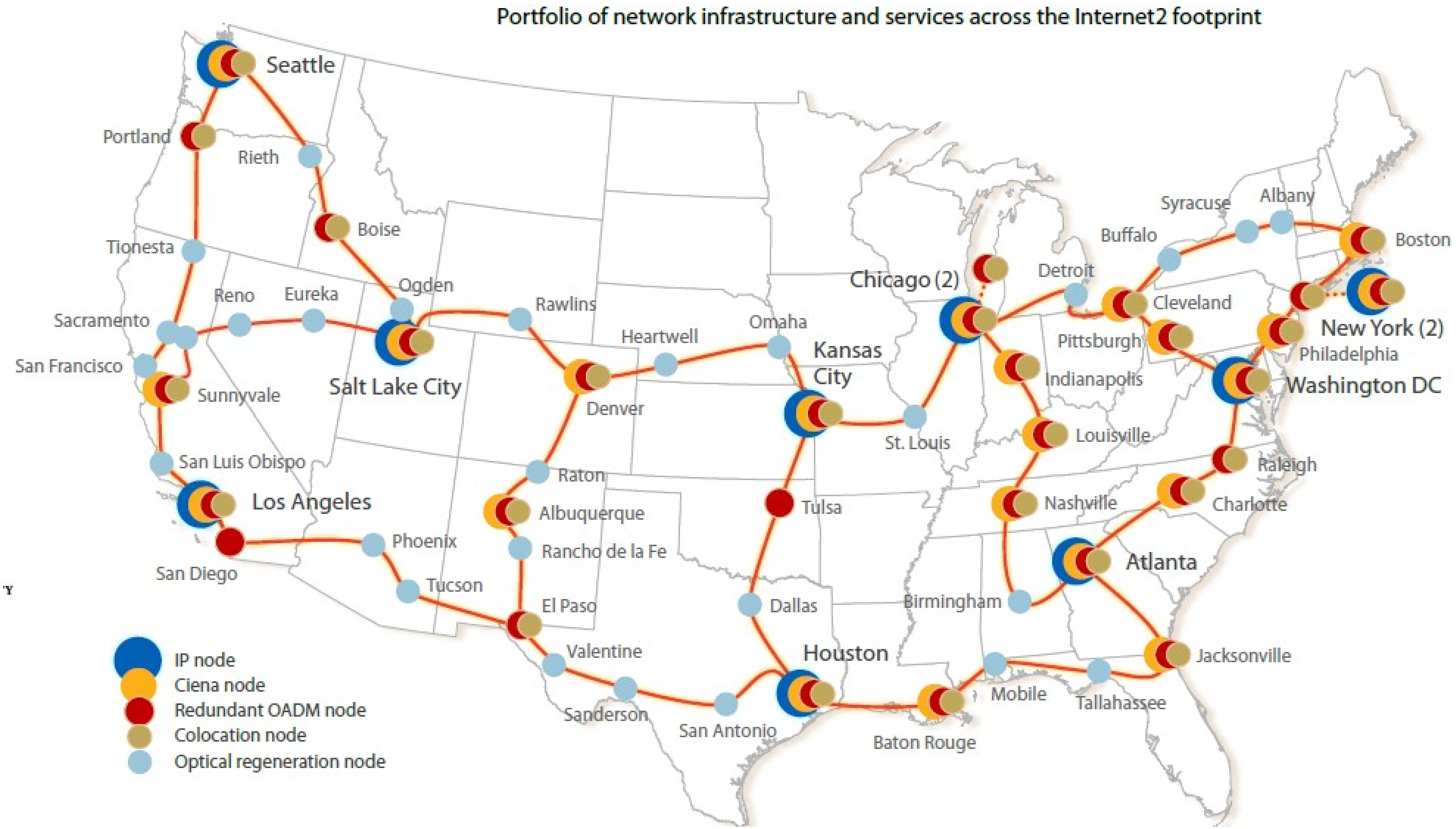
Research/High-speed networks

- LSU campus: 1Gbps (some 100 Mbps)
- LONI – Louisiana Optical Network Initiative: 10Gbps
- National and international networks: multiple 10Gbps
- Soon to be introduced
  - 40Gbps and 100Gbps
  - Standards ratified (June 2010)

LONI


LSU HSC Shreveport
Haughton
La Tech
Monroe
Tallulah
Edwards
Jackson, MS
Coushatta
Northern DWDM Ring
Seminary
Mendenhall
Natchitoches
Derry
Tylertown
Southern
Alexandria
Greensburg
NLR BR
LPB
Zachary
Landry
SLU
Hammond
LaPlace
LSU
Willow
Ramah
LSU
LSU
Roanoke
ULL
Ramah
Southern
DWDM Ring
Lake Charles
Franklin
Schriever
Tulane
UNO
LSU HSC NO
(1)
(3)
(4)
(5)
(2)

Internet2

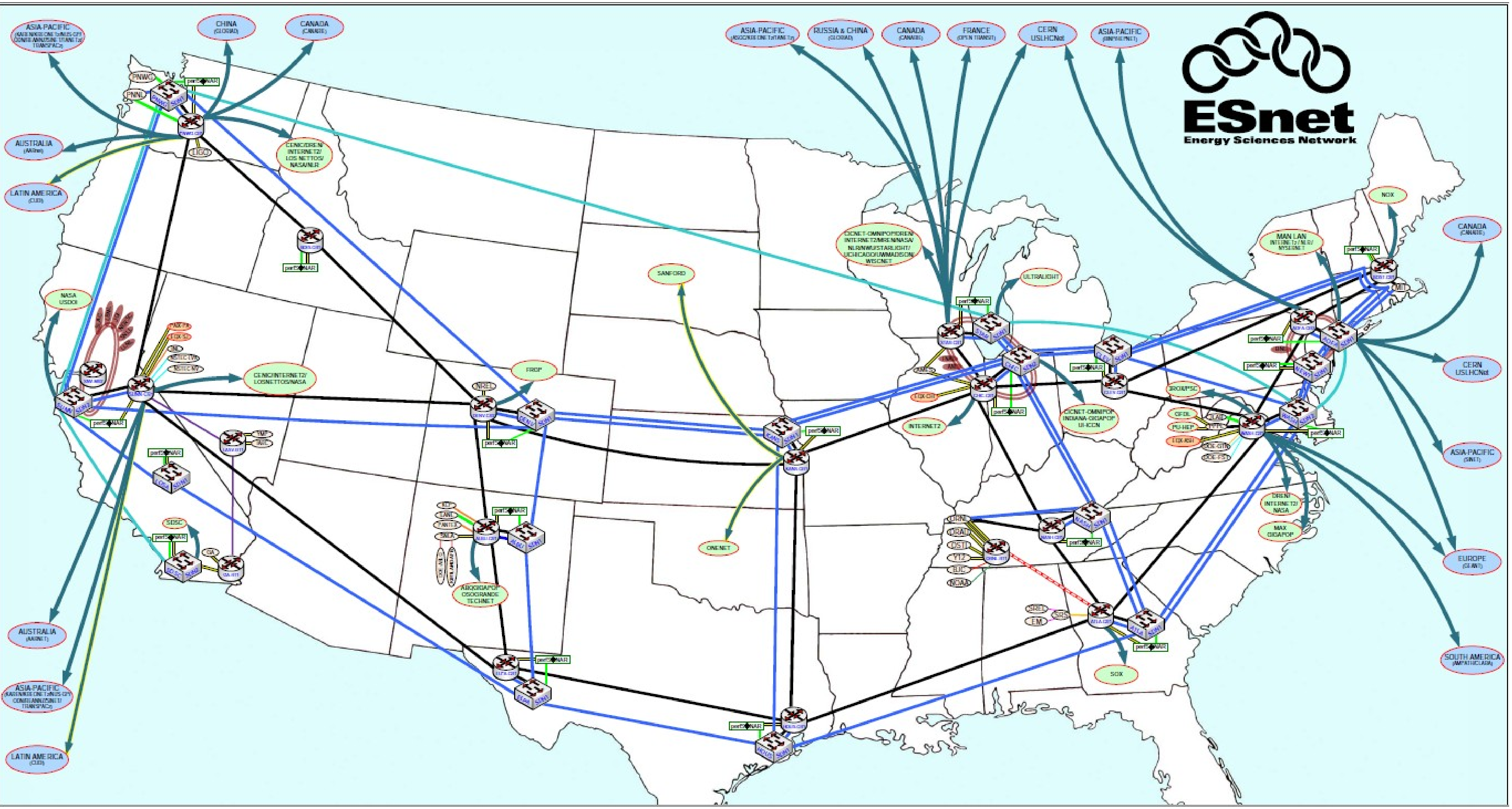


© Internet2

National LambdaRail

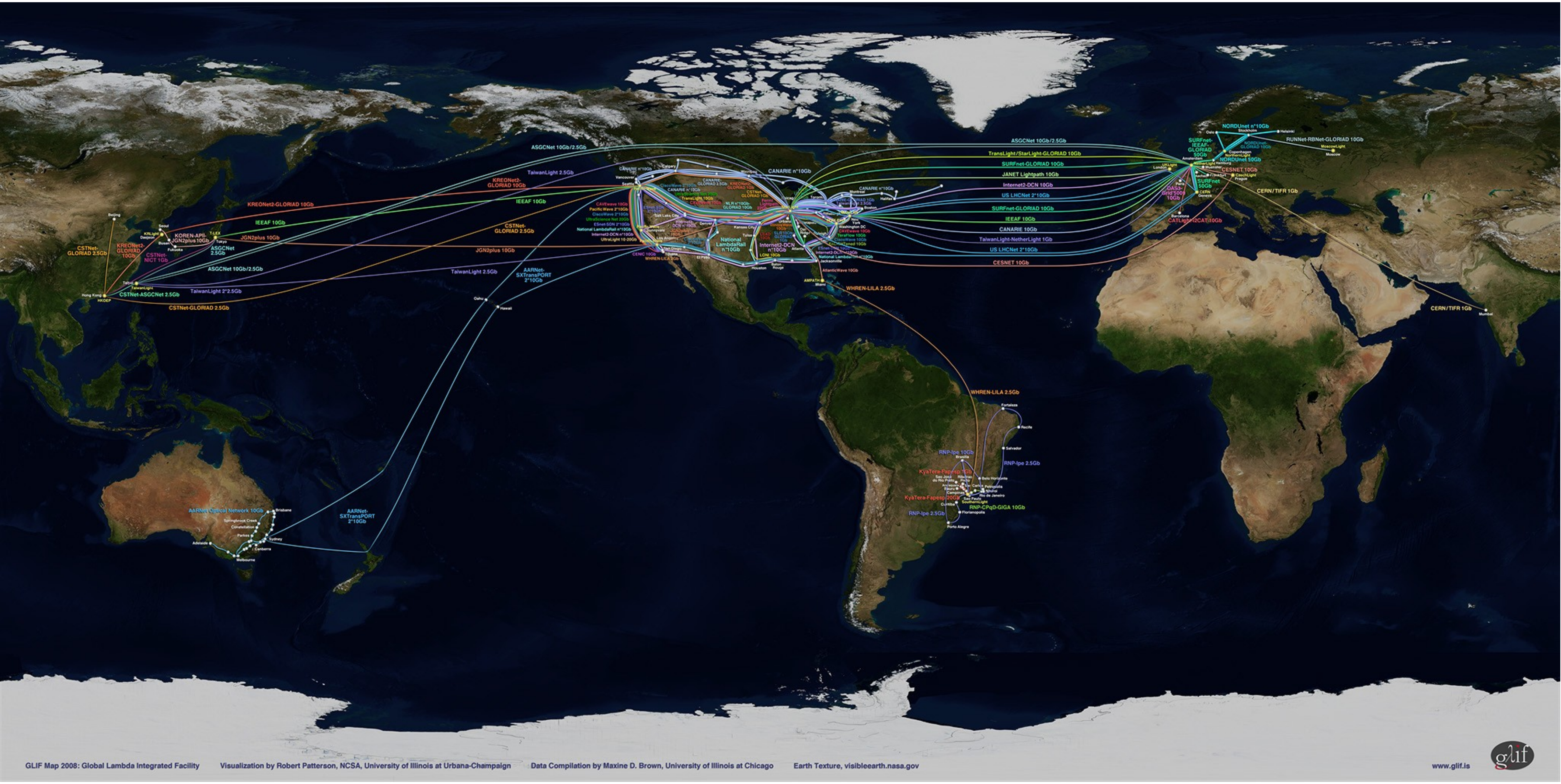
© National LambdaRail, Inc

ESnet



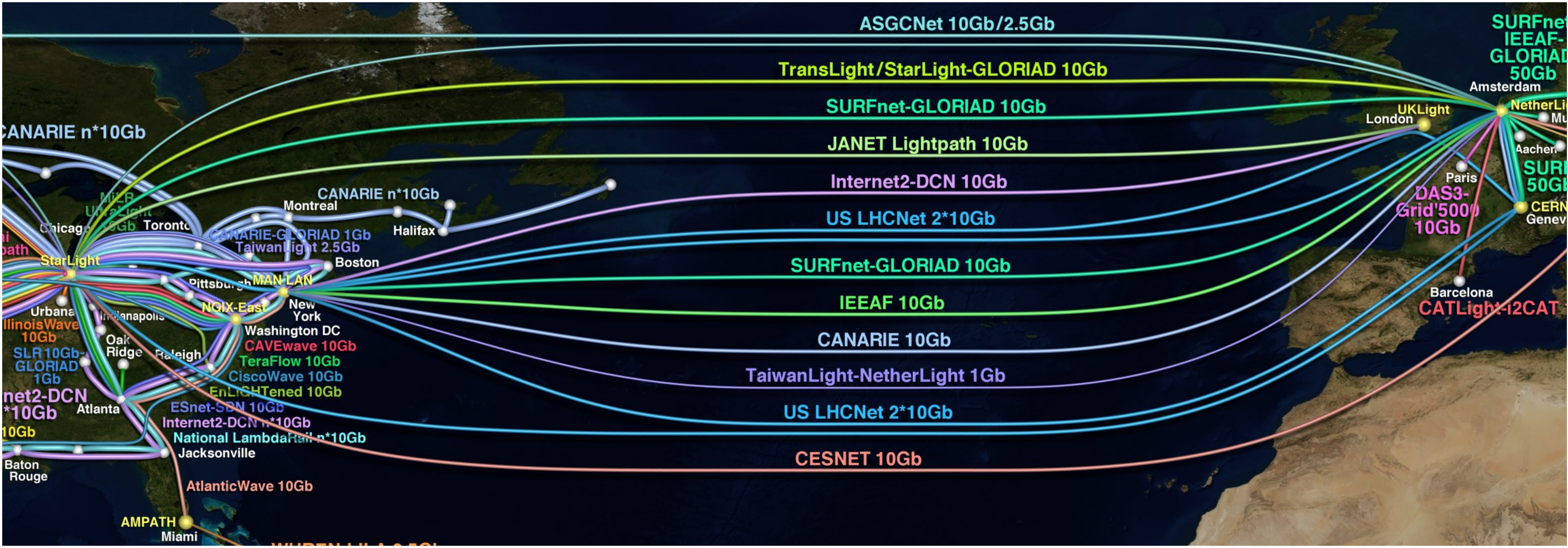
• © ESnet, DOE, LBNL

GLIF



GLIF Map 2008: Global Lambda Integrated Facility
Visualization by Robert Patterson, NCSA, University of Illinois at Urbana-Champaign
Data Compilation by Maxine D. Brown, University of Illinois at Chicago
Earth Texture, visibleearth.nasa.gov
www.glif.is

http://ww.glif.is

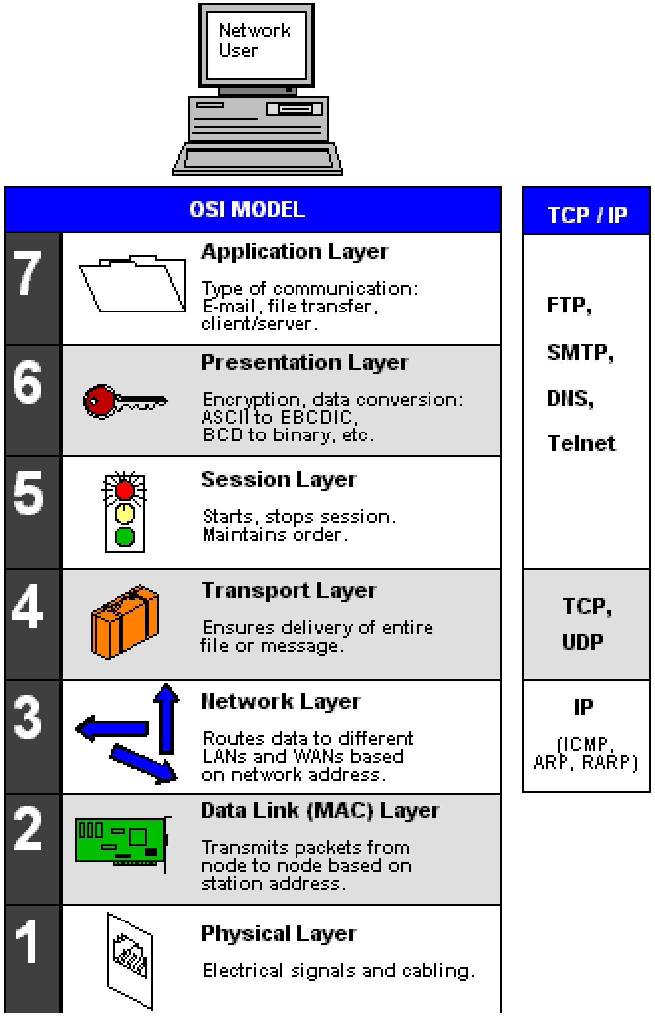


- The Global Lambda Integrated Facility (GLIF) Map 2008 visualization was created by Robert Patterson of the Advanced Visualization Laboratory (AVL) at the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign (UIUC), using an Earth image provided by NASA. Data was compiled by Maxine D. Brown of the Electronic Visualization Laboratory (EVL) at the University of Illinois at Chicago (UIC). Funding was provided by GLIF and US National Science Foundation grants # SCI-04-38712 to NCSA/UIUC and # OCI-0441094 to EVL/UIC. For more information on GLIF, see http://www.glif.is/.

Terminology/Network Layers

From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.

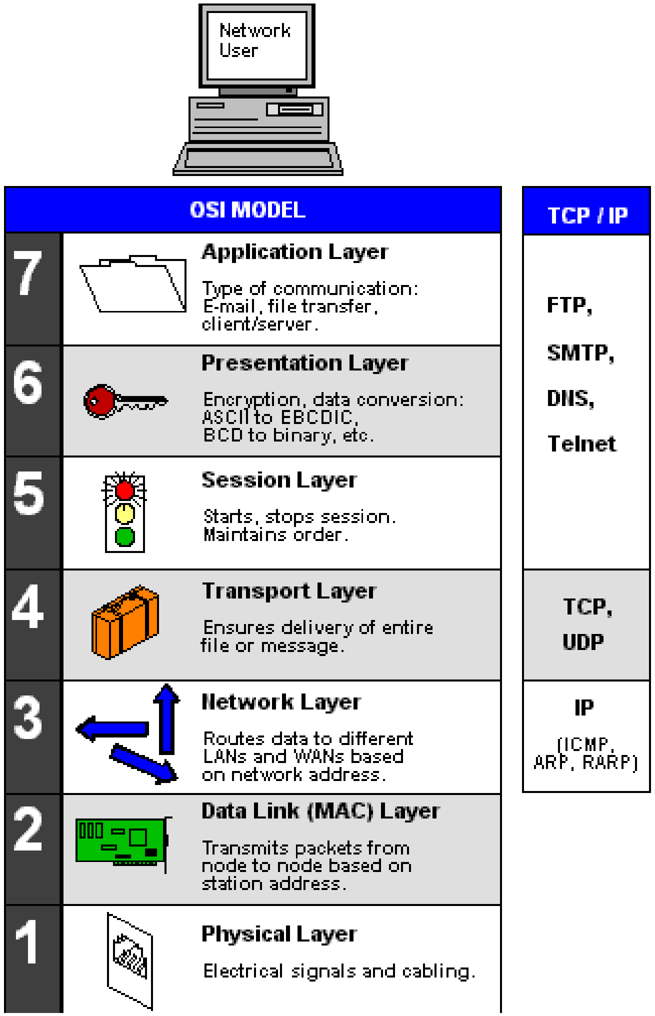
- Network = multiple connecting elements/nodes
- Link – point to point connection between two nodes
- Path – multiple links & nodes
- Layered design
- Each layer builds on functionality of layers below
  - Also hides the layers below



Network Layers
Physical

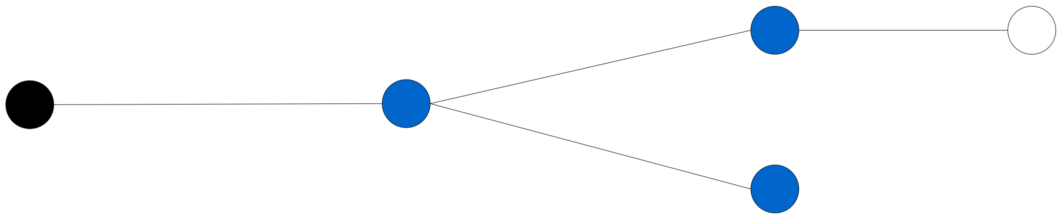
From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.

- Protocol = sequence of messages, responses. Convention, rules, format for transmitting data
- Layer 1 – physical, point-to-point. Transmitting data bits

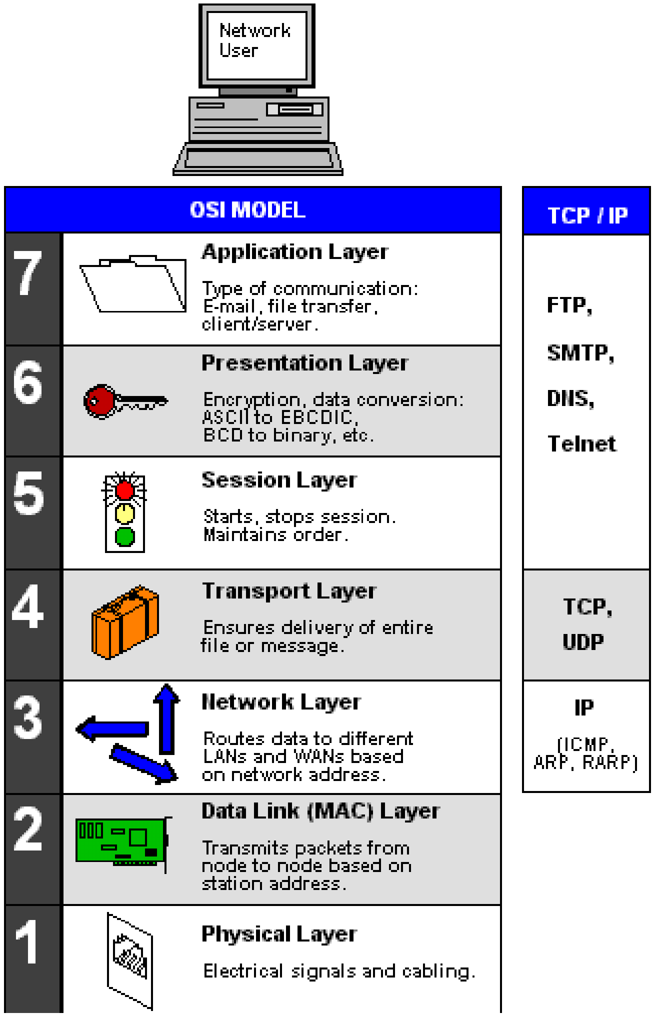


Network Layers Switching

- Layer 2 (switched network) – data link layer, adds addressing for single domain or LAN. Ethernet, DSL .. Flat addressing. Single path.
- MAC/HW address

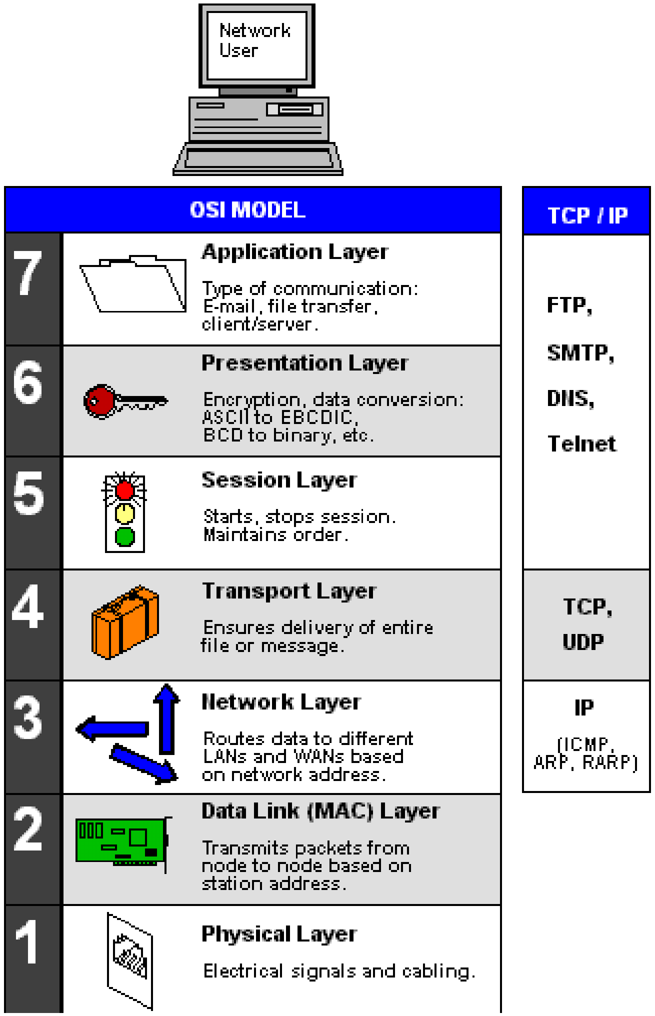


From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.

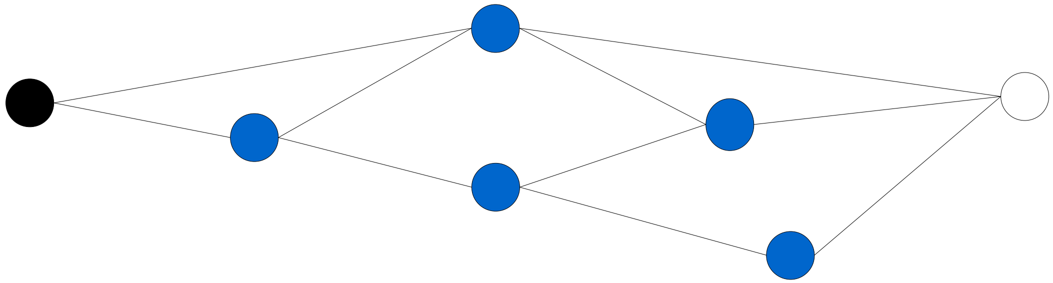


Network Layers
Routing

From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.



- Layer 3 (routed network) – network layer, transmits data from a source to a destination via multiple networks. IP. Routable addresses. Multiple paths. Decision-making. Algorithms.
- IP addresses



Network Layers Transport

- "Layer 4"
- Differentiate between applications on the same machine
- Ports (another layer of addressing)!
- Provide transport services
  - For example, transport reliability, ordering.

Transport Protocols

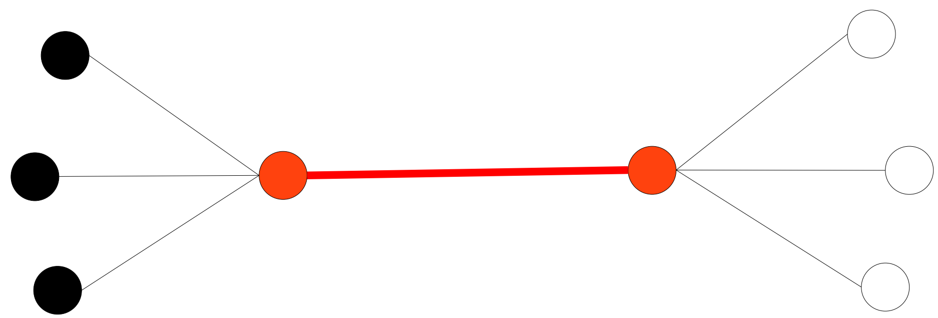
- Connection-oriented or connectionless
  - Establish a logical connection between sender and receiver or not
- Reliable or unreliable
  - Data might get lost
  - Lower layers do not guarantee reliability
  - Need retransmission to provide reliability (-)
- Ordering
  - Will all the bytes arrive in the same order?
  - Need buffering to implement (-)

Transport Protocols

- Other features
- Byte-oriented vs. packet or message-oriented
  - Deal with data as a sequence of bytes or as packets
- Flow control
  - To not overwhelm the receiver

Congestion Control

- Multiple transfers going over the same link or through the same device



- In packet switched networks (vast majority, circuit switched networks presented later) every packet is equal
  - Creates congestion

Congestion Control

- Unmanaged congestion leads to network overload and crash
  - Quantity of useful data through the network will keep decreasing as senders keep retransmitting lost data
- All streams going over the congested link should equally reduce their transmission rate
- Using some type of feed-back that tells the sender there is a congestion in the network
  - Lack of acknowledgments
  - Transmission delay
  - Using a search method to find the network capacity

Congestion Response

- Generally, reducing the data transmission size, or rate
- Congestion control implementation
  - Using a congestion window = the amount of data that is “in transmission”
    - Reduced when congestion is detected
    - Increased when data is successfully transmitted
  - Using a transmission rate that is modified in response to congestion/successful data transmission
- Sometimes, when there is a guarantee of available bandwidth (and only then), congestion control is not needed

Transport Protocols: UDP

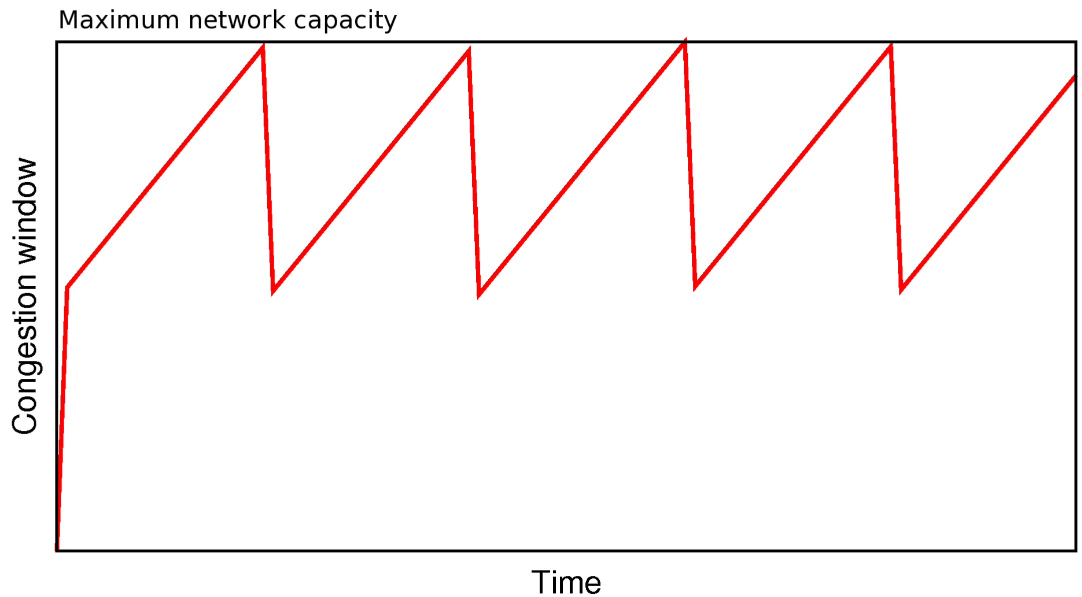
- Standard (IETF RFC)
- Unreliable
- Connectionless
- Unordered
- No congestion control
- Packet-oriented
- Basically IP + ports, offering basic means to transmit data between applications

Transport Protocols: TCP

- Standard
- Reliable
- Ordered
- Byte-oriented (provides byte stream semantics)
- Connection-oriented
- Congestion control
  - With changes/improvements in implementation over time

TCP Congestion Control

- Using a congestion window
- If a packet is lost, window reduced in half
- If data is successfully transmitted, at most one segment is added to the window each RTT (round-trip-time)



Ideal TCP congestion control on a link with no other traffic

Low TCP performance

TCP Throughput (Mbps)	RTTs Between Losses	W
1	5.5	8.3
10	55.5	83.3
100	555.5	833.3
1000	5555.5	8333.3
10000	55555.5	83333.3

- What is needed to achieve a particular transfer rate with TCP (from ftp://ftp.rfc-editor.org/in-notes/rfc3649.txt - wiki papers)
- 55555 RTT * 0.1s (100 ms) RTT = 1 ½ hours

Transport Protocols: Others

- SCTP: standard, congestion control, message oriented, ordering is optional
- Many protocols designed in response to poor TCP performance over wide-area high-capacity networks
- TCP variants: Scalable TCP, High-speed TCP, TCP Vegas, Fast TCP, Compound TCP, BI-TCP, CUBIC, TCP Westwood (+)
- Reliable protocols implemented using UDP: LambdaStream, UDT, RBUDP
- Network elements involved in transport protocol: XCP
- Group communication: Group Transport Protocol

Network Characteristics

- Names: easy to remember
  - Resolved to addresses using DNS service (can use “host <name>” utility)
- Round-trip-time
  - Can measure with “ping <destination IP/name>” utility
- Network route
  - Can find out with “traceroute <destination IP/name>” utility

Transport Performance

- Iperf
- http://sourceforge.net/projects/iperf/
- Download and scp to machines, or find a direct link and download using wget (wget <URL>) or some other tool
- Decompress archive: tar xzvf iperf-2.0.5.tar.gz
  cd iperf-2.0.5
  ./configure –prefix=$HOME/iperf_install
- make; make install => The executable will be in your home directory (cd command gets you there), under iperf_install/bin
- netperf: http://www.netperf.org/netperf/ (alternative)

using iperf

- Client: sends the data; server: receives the data
- Can measure both TCP and UDP performance
- Server should be started first (with iperf -s), because client connects to the server
- On the client, make sure you use the right name/address for the server (iperf -c <server name or address>)
- Common error: connect failed: Connection refused
  - Means the server is not running or client cannot reach server for some reason

Ports

- Other error:
  - When running the server, you may get this error:
    - bind failed: address already in use
    - That means somebody else is running a server on the same port ..
    - To avoid such errors we'll assign ports for each of you to use (starting with 5100 – next slide)
  - Specify port with -p <your port number> on both client and server
  - Server run with $HOME/iperf_install/bin/iperf -s -p <port> -w 30M -i 1
  - Client run with $HOME/iperf_install/bin/iperf -c <server name> -p <port> -w 30M -i 1

Port assignment

- Bohara, Bidur 5101
  Chiu, Chui-Hui 5102
- Cui, Cheng 5103
  Guidry, Richard D Jr 5104
- Jaladi, Madhava Kumar 5105
  Khurana, Sandeep 5106
- Kogler, Daniel Scott 5107
  LIU, Ke 5108
- Mantha, Pradeep Kumar 5109
  Nagabandi, Karthik Kumar 5110
- Pathak, Mukta Suhas 5111
  Rastegar Tohid, Mohammad 5112

Port assignment

- Saripalli, Sai 5113
  Shakya, Shobhit Sandesh 5114
- Tyson, Brandy Michelle 5115
  Vellore Singaravelu, Rajesh K 5116
- Xue, Lin 5117
  Yin, Dengpan 5118
- Zebrowski, Ashley Nicole 5119
  Zhang, Qian 5120
- Zhu, Ling 5121
- Any other ports that are free .. just pick a random one

Machines

- Abe (at NCSA): abe-ipib-gw01.ncsa.uiuc.edu or abe-ipib-gw02.ncsa.uiuc.edu. - use SSH and NCSA password
- QueenBee (at LSU): gridftp-qb.loni-lsu.teragrid.org (same as qb1.loni.org) – GSISSH fine
- Lonestar (at TACC): 129.114.50.240 or 129.114.50.241 .. Unfortunately they have the same, ambiguous host name – GSISSH fine
- Alternative to Lonestar, use Ranger (also at TACC): login1.ranger.tacc.teragrid.org (or any of following - login2, login3, login4) – GSISSH fine
- Steele (at Purdue): tg-steele.purdue.teragrid.org – GSISSH fine
  - tg-data-01.rcac.purdue.edu or tg-data-02.rcac.purdue.edu

Example (server)

- ssh <username>@abe-ipib-gw01.ncsa.uiuc.edu (username for Abe/NCSA and Abe test machine)

```
[hutanu@abe-ipib-gw01 ~]$ $HOME/iperf_install/bin/iperf -s -p 5123 -w 30M -i 1
------------------------------------------------------------
Server listening on TCP port 5123
TCP window size: 32.0 MByte (WARNING: requested 30.0 MByte)
------------------------------------------------------------
[  4] local 141.142.31.151 port 5123 connected with 129.114.50.240 port 44265
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec    167 MBytes  1.40 Gbits/sec
[  4]  1.0- 2.0 sec    548 MBytes  4.60 Gbits/sec
[  4]  2.0- 3.0 sec    764 MBytes  6.41 Gbits/sec
[  4]  3.0- 4.0 sec    783 MBytes  6.57 Gbits/sec
[  4]  4.0- 5.0 sec    783 MBytes  6.57 Gbits/sec
[  4]  5.0- 6.0 sec    783 MBytes  6.57 Gbits/sec
[  4]  6.0- 7.0 sec    783 MBytes  6.57 Gbits/sec
[  4]  7.0- 8.0 sec    780 MBytes  6.54 Gbits/sec
[  4]  8.0- 9.0 sec    783 MBytes  6.57 Gbits/sec
[  4]  9.0-10.0 sec    777 MBytes  6.52 Gbits/sec
[  4]  0.0-10.0 sec  6.79 GBytes  5.82 Gbits/sec
[hutanu@abe-ipib-gw01 ~]$ $HOME/iperf_install/bin/iperf -s -p 5123 -w 30M -i 1
```

Example (client)

- ssh <tgXXXXXX>@129.114.50.240 (username for Lonestar/TACC and Lonestar address)

```
bash-3.00$ $HOME/iperf_install/bin/iperf -c abe-ipib-gw01.ncsa.uiuc.edu -p 5123
-w 30M -i 1
------------------------------------------------------------
Client connecting to abe-ipib-gw01.ncsa.uiuc.edu, TCP port 5123
TCP window size: 60.0 MByte (WARNING: requested 30.0 MByte)
------------------------------------------------------------
[  3] local 129.114.50.240 port 44265 connected with 141.142.31.151 port 5123
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   202 MBytes  1.69 Gbits/sec
[  3]  1.0- 2.0 sec   536 MBytes  4.50 Gbits/sec
[  3]  2.0- 3.0 sec   742 MBytes  6.22 Gbits/sec
[  3]  3.0- 4.0 sec   783 MBytes  6.57 Gbits/sec
[  3]  4.0- 5.0 sec   783 MBytes  6.57 Gbits/sec
[  3]  5.0- 6.0 sec   783 MBytes  6.57 Gbits/sec
[  3]  6.0- 7.0 sec   783 MBytes  6.57 Gbits/sec
[  3]  7.0- 8.0 sec   780 MBytes  6.54 Gbits/sec
[  3]  8.0- 9.0 sec   783 MBytes  6.57 Gbits/sec
[  3]  0.0-10.0 sec  6.79 GBytes  5.83 Gbits/sec
bash-3.00$ 
```

UDP transmission

- By default iperf uses TCP
- -u (on both client and server) switches to UDP
- UDP does not have automatic tuning of transfer speed, so have to specify on the sender side
- -b <speed> tells the client how fast to transmit the data
  - Could be too fast, and the server will report packet loss
  - Could also be to low, the client will not use all the available bandwidth
  - Use binary search to find the top UDP speed (and under 0.1% loss report)!

Example (UDP server)

- ssh <username>@qb1.loni.org

```
[ahutanu@qb1 ~]$ ./iperf_install/bin/iperf -s -w 30M -i 1 -u -p 5123
------------------------------------------------------------
Server listening on UDP port 5123
Receiving 1470 byte datagrams
UDP buffer size: 32.0 MByte (WARNING: requested 30.0 MByte)
------------------------------------------------------------
[  3] local 208.100.92.21 port 5123 connected with 129.114.50.164 port 55728
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec  45.1 MBytes   379 Mbits/sec   0.004 ms    0/32187 (0%)
[  3]  1.0- 2.0 sec  46.7 MBytes   392 Mbits/sec   0.003 ms    0/33312 (0%)
[  3]  2.0- 3.0 sec  47.6 MBytes   400 Mbits/sec   0.002 ms    0/33975 (0%)
[  3]  3.0- 4.0 sec  46.2 MBytes   388 Mbits/sec   0.004 ms    0/32984 (0%)
[  3]  4.0- 5.0 sec  45.1 MBytes   378 Mbits/sec   0.003 ms    0/32154 (0%)
[  3]  5.0- 6.0 sec  45.4 MBytes   381 Mbits/sec   0.035 ms    0/32395 (0%)
[  3]  6.0- 7.0 sec  45.4 MBytes   381 Mbits/sec   0.004 ms    0/32381 (0%)
[  3]  7.0- 8.0 sec  47.8 MBytes   401 Mbits/sec   0.003 ms    0/34087 (0%)
[  3]  8.0- 9.0 sec  47.9 MBytes   402 Mbits/sec   0.002 ms    0/34148 (0%)
[  3]  9.0-10.0 sec  46.0 MBytes   386 Mbits/sec   0.005 ms    0/32782 (0%)
[  3]  0.0-10.0 sec   464 MBytes   389 Mbits/sec   0.616 ms    0/330922 (0%)
[  3]  0.0-10.0 sec  1 datagrams received out-of-order
```

Example (UDP client)

- ssh <tgXXXXXX>@login4.ranger.tacc.teragrid.org

File Edit View Terminal Help

```
login4% $HOME/iperf_install/bin/iperf -c qb1.loni.org -w 30M -i 1 -u -b 1500M -p
 5123
------------------------------------------------------------
Client connecting to qb1.loni.org, UDP port 5123
Sending 1470 byte datagrams
UDP buffer size: 60.0 MByte (WARNING: requested 30.0 MByte)
------------------------------------------------------------
[  3] local 129.114.50.164 port 55728 connected with 208.100.92.21 port 5123
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  45.1 MBytes   379 Mbits/sec
[  3]  1.0- 2.0 sec  47.2 MBytes   396 Mbits/sec
[  3]  2.0- 3.0 sec  47.1 MBytes   395 Mbits/sec
[  3]  3.0- 4.0 sec  46.2 MBytes   388 Mbits/sec
[  3]  4.0- 5.0 sec  45.1 MBytes   378 Mbits/sec
[  3]  5.0- 6.0 sec  46.1 MBytes   386 Mbits/sec
[  3]  6.0- 7.0 sec  44.7 MBytes   375 Mbits/sec
[  3]  7.0- 8.0 sec  47.9 MBytes   402 Mbits/sec
[  3]  8.0- 9.0 sec  47.7 MBytes   400 Mbits/sec
[  3]  9.0-10.0 sec  46.7 MBytes   392 Mbits/sec
[  3]  0.0-10.0 sec   464 MBytes   389 Mbits/sec
[  3] Sent 330923 datagrams
[  3] Server Report:
[  3]  0.0-10.0 sec   464 MBytes   389 Mbits/sec   0.615 ms    0/330922 (0%)
[  3]  0.0-10.0 sec  1 datagrams received out-of-order
login4%
```

Tuning parameters (Window size)

- Window (or buffer size): -w parameter (most important on the receiver size for UDP, and on sender size for TCP). Decreases the chance that data gets dropped by the receiver (UDP), or not enough data is sent by the sender to fill the pipe (TCP).
  - In theory you need a buffer that will hold enough data to fill the network pipe (capacity of network pipe = latency * line rate; latency = RTT/2, RTT can be measured with ping). Use 10Gbps for line rate.
  - In practice, can use -w 30M (30 Mbytes)
  - Can check the effect of varying the window size

Tuning parameters (packet size for UDP)

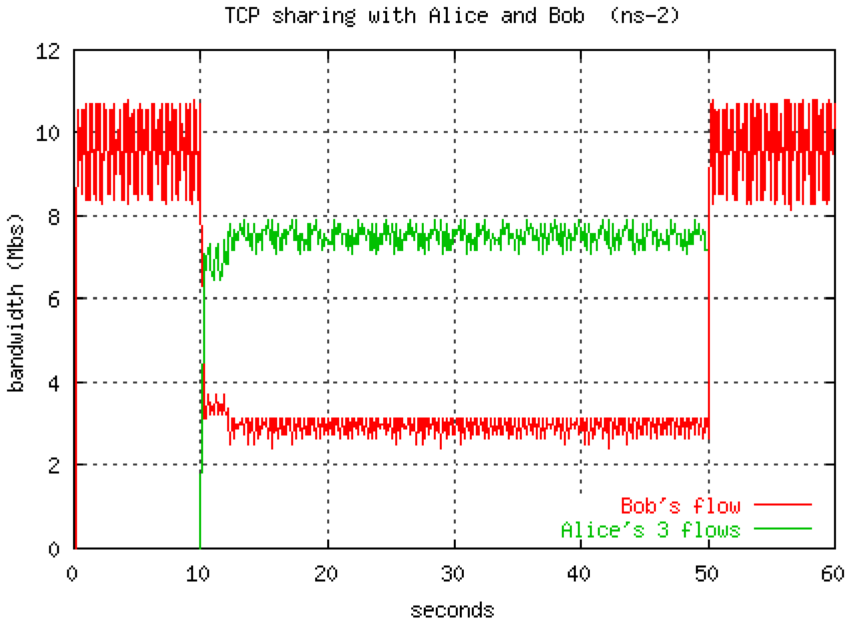
- Packet size: -l <size in bytes>. Have to set on both client and server. The larger the packet size, the smaller the number of packets to be transmitted.
  - Higher throughput (CPU not busy with lots of packets)
  - Higher chance of packet loss. Also can add fragmentation if network link does not support large packet size (ping -s <packet size> destination verifies if the specific packet size is supported by the network)
  - Default packet size: 1470 bytes
  - Maximum possible packet size: 9000 bytes
  - QueenBee (LSU) only supports default packet size! But all others support 9000 byte packet sizes (jumbo frames)
  - Recommended: 8000 bytes (-l 8000)

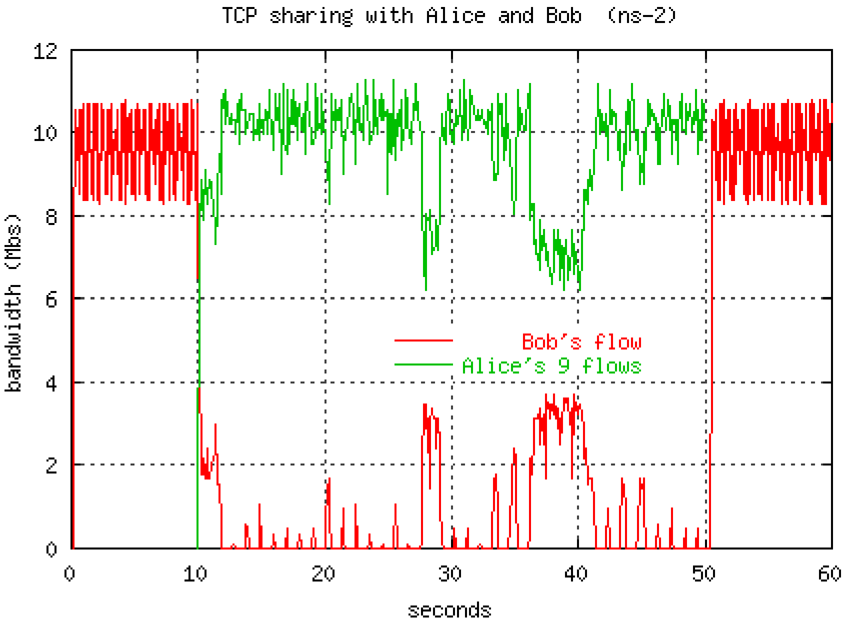
Tuning (parallel streams for TCP)

- Sometimes using parallel streams helps performance, because when a packet is dropped/lost, only a single stream (from the bundle) reduces the speed
  - In a way, this is “cheating” TCP's congestion control mechanism (in personal computing this is easily exploited by some tools such as P2P file transfers .. one reason why ISP's are looking at alternatives - later)
- use -P <number of streams> parameter on client/sender
- Report will be given for each stream, with a total under: SUM

Note on parallel streams

- Should not be used on congested networks
  - http://www.csm.ornl.gov/~dunigan/netperf/parallel.html
  - Unfair, possibly useless, dangerous when CC needed





Important parameters

- -t <seconds> : how many seconds should the transfer run for (default 10s) – influences TCP performance
- -i <interval>: how often should the performance output be printed (default only at the end of the transfer). Use -i 1 to get printouts each second .. see how the performance evolves
- Note: Test both directions (switch client and server)!
- Useful tool: ifconfig (gives IP addresses of machines, maximum MTU/packet size) – sometimes not in path, but under /sbin/ifconfig

Assignment (part 1)

- Run iperf between TeraGrid sites and write a report on the transport performance
- Need at least a minimum report on TCP and UDP speed using at least three sites (NCSA, TACC plus one of LSU/Purdue)
- Optimize, check the effect of changing the window size, changing the packet size (UDP), changing the number of parallel streams (TCP), changing the duration of the transfer (TCP), use all four sites ..
- Graphs would be nice
- Teams (up to two). Optional but recommended. Three if justified (for example because #students not even ..)

UDT library

- Feel free to use in your analysis for bonus points (not required though)
- http://sourceforge.net/projects/udt/
- http://udt.sourceforge.net/udt4/index.htm
- High-performance data transmission protocol implemented using UDP, has library implementation so can be used without administrative privileges
- Has appserver/appclient test examples (that can be edited for example to change the packet size, or MTU – need to change both client and server)