e-VLBI Overview

Chet Ruszczyk
Agenda

• e-VLBI Basics
  – Modes
  – Advantages
  – Expectations

• Network Basics
  – Architecture
  – Transport Protocols
  – Tuning

• Future for e-VLBI
e-VLBI Modes

• 3 Modes for e-VLBI transfers
  – Real-time mode
  – Non-real time mode
  – Psuedo real-time mode

• Real-time mode
  – Data rate generated at the source data acquisition system must be sustained end-to-end on the network to an end system, e.g. the correlator, on the network
    • Data stream must have the same characteristics as if correlating data from disks locally
e-VLBI Modes (cont)

• Non real-time mode
  – Data is buffered at the source and electronically transferred to the destination.
    • Best effort transport

• Pseudo real-time mode
  – Data is recorded at the station and played back from a remote site, via the network, at a later time
    • Data stream must have the same characteristics as if correlating data from disks locally
e-VLBI Advantages

• Rapid processing turnaround
  – Astronomy
    • Ability to study transient phenomena with feedback to steer observations
  – Geodesy
    • Higher-precision measurements for geophysical investigations
    • Better Earth-orientation predictions, particularly UT1, important for military and civilian navigation
e-VLBI advantages (cont)

• Can increase the sensitivity of VLBI observations
  – For most VLBI observations, sensitivity increases as $\sqrt{\text{bandwidth}}$
  – Increasing bandwidth is usually the most cost-effective way to increase sensitivity
  – The growth of network bandwidth data rates far exceeds the recording growth capability data rates
e-VLBI Expectations

• I have a 1 Gbps network connection at the station
  – I want 1Gbps performance or as close to it as possible
  – Will I always be able to achieve it?
  – Yes, no, maybe

• Yes
  – Over short to medium distances
  – Networks are clean
    • No errors between end points
    • Allow aggressive transport protocols
      – Links are under utilized
e-VLBI Expectations (cont)

- No
  - Long haul networks
    - Jumbo frames are not supported
  - Traversing multiple network domains
    - Equipment in path may prohibit performance
    - Errors on one network
  - Are not allowed to use aggressive protocols
    - Shared networks

- Maybe
  - When usage is low, allowed to be more aggressive
    - Between 12AM – 6AM
Network Protocol Stack

Application
Mark5
disk2net

Transport Layer
TCP/UDP

Network Layer
(IP)

Data Link
(Ethernet)

Router

Network Layer
(IP)

Data Link
(Ethernet)

Data Link
(Ethernet)

Transport Layer
TCP/UDP

Application
Mark5
Net2disk

Network Layer
(IP)

Data Link
(Ethernet)
Data Framing

VLBI data

vdif

VLBI data

TCP header

Application data

IP header

TCP Segment

Ethernet header

IP datagram (46 – MTU size) bytes

CRC

Application (Mark5 App)

Transport Layer TCP/UDP

Network Layer (IP)

Data Link (Ethernet)

* MTU (Maximum Transmission Unit) – Normal length 1500, Jumbo frames 9000 bytes
End-to-End Connections

• Circuit Switching
  – Old Phone system
    • Dedicated resources between sender / receiver
    • Whether you send data or not.

• Packet Switching
  – Ability to statistically multiplex multiple data streams on a single channel
  – Goal – efficient utilization of the channel
Quality of Service (QoS)

- Ability to have the packet switched networks
  - Emulate circuit switch services
- Heavily researched but extremely complex to implement and support
- Each network layer can provide some type of QoS
Transport Layer

• Provides end-to-end communication between two or more hosts.
• Isolates application from changes in the underlying hardware
  – IP routing / ATM / SONET
• Provides a number of services to upper layers
  – e.g. Reliable or unreliable delivery of data
Transport Layer

• Protocol
  – How the sender / receiver cooperate to provide that service

• Reliable service
  – TCP (Transmission Control Protocol)
  – Connection based protocol
  – 100 % guaranteed ordered delivery of data
  – Handshaking (acknowledgements)
Transport Layer

• Unreliable service
  – UDP (User Datagram Protocol)
  – Connectionless protocol

• Protocols built on top of UDP
  – Provide reliable transport
  – Congestion control added
    • In Linux user space - outside the kernel
    • In the kernel
  – Overcome short comings of pure TCP protocols
TCP

• Connection based protocol

• Window Based
  – Transmits so much data and wait for acknowledgments that data was received
  – Retransmits if missing data
  – Transmits next window

• Performance is best
  – when the network pipe between sender /receiver is full of data
  – no errors on transmitted packets
TCP

• Originally designed with certain assumptions.
  – When a data segment is lost it is caused by congestion
  – Congestion refers to too many segments competing for network resources
    • Buffer overflows
      – Routers
      – Switches
      – NIC Cards
  – Responsive to congestion notification
    • Reducing sending transmission rate
TCP (cont)

• Performance is dependent on
  – Bandwidth Delay Product (BDP)
    • Transfer rate * Round trip time delay
  – Original TCP does not scale well for
    • High bandwidth networks
    • Long haul networks

• Newer TCP protocols developed
  – Goal: meet the short comings of the original design
  – More aggressive congestion control
TCP Performance

• BDP
  – Need to know the slowest link between sender/receiver
  – Round Trip Time (RTT)
    • Use ping or traceroute

• Example
  – Round trip time 170 msecs
    • (NyAlesund – Haystack)
  – Assume you have a 1 Gbps link
    \[ BDP = (1\text{Gbits/sec}) \times (1\text{byte}/8\text{bits}) \times (170\text{msec}) \]
    \[ BDP = 21.25 \text{ MBytes} \]
    • In order to fill the network pipe require 21.25 MB of data on the line

• Default TCP buffer size is 64KBytes
  – Adjust the kernel buffer size allocation
  – Application sets the window size based on BDP
UDP

• Characteristics
  – Connection-less protocols
    • No connection needs to be established to start
    • Start transmitting data
  – Transmit data at whatever rate you wish
  – No feedback
    • Congestion in network
    • Flow control
      – Receiver telling sender to slow down rate
  – No guaranteed delivery
    • Order not guaranteed
UDP (cont)

- Overcome TCP’s shortcomings
  - approach a more efficient approach to utilizing the channel

- Class of protocols created using UDP as the base protocol but add:
  - Guaranteed delivery of data
    - selective acknowledgments
  - Congestion control
    - How to react to errors in transmission
  - Order guaranteed
  - Flow control
    - Slow the rate down, speed up
UDP (cont)

• NOTE:
  – Some domains do not allow UDP traffic
    • Denial of Service (DoS) attack
    • Lack of congestion control
  – Network protocol stack is optimized for TCP not UDP
    • TCP used in 99.99 % of applications
Newer Transport Protocols

• “TCP Friendly”
  – Flows behave under congestion
  – Responsive to congestion notification
  – In steady-state does not use more bandwidth than a conformant TCP protocol
    • Drop rate
    • Round trip time (RTT)
    • Maximum transmission unit (MTU) size

• Not all protocols are TCP friendly
  – Usually ones designed for
    • Long haul
    • high bandwidth

• UDP and TCP can fall into this category

6th IVS TOW May 2011
Tsunami UDP

- Developed by Indiana University
- Advantages
  - No slow start up for transmission
    - Maximum rate at the start
  - Specify speedup / slow down recovery when errors are seen
  - Data transmission
    - default priority for data integrity
  - Data rate priority
    - disables retransmissions
    - maximizes bandwidth
- Adopted by astronomy community for ftp application -
Tsunami UDP

• Disadvantages
  – Assumes you have
    • a priori knowledge about the quality of your network connection
    • For a long haul network, do you truly know the utilization of all links connecting you end-to-end
      – over many domains?
  – If configured incorrectly
    • Poor performance for user
    • Poor performance for other users sharing the same network link
Tuning

• How to tune a Linux distribution
  – Earlier kernels required manual configuration
  – Based upon the BDP
  – Tune both the sender and receiver

• Linux 2.6.18 and greater have full auto-tuning
  – 4MB maximum buffer sizes
  – Manual tuning is not generally recommended
  – Sender tuning has been enabled for years
Tuning (cont)

• Autotuning adjusts the receive buffer size
  – Autotunes for each connection
  – `cat /proc/sys/net/ipv4/tcp_moderate_rcvbuf`
  • 1

• Per connection memory space defaults in
  – `/proc/sys/net/ipv4/tcp_rmem` - TCP receive buffer
  – `/proc/sys/net/ipv4/tcp_wmem` - TCP send buffers
Tuning (cont)

- tcp_rmem and tcp_wmem
  - Three element array
    - minimum, default, maximum buffer size
  - Balances memory usage
    - Not just TCP window size
    - Actual memory usage
      - socket data structures
  - Sets bounds on auto-tuning
  - Must tune both end systems for optimal performance
Tuning (cont)

- **Maximum > BDP**
  - Dependent on the amount of memory you have in your system
    - e.g. 2MB of memory, the maximum value will be 2MB
      - Can never achieve more than what is physically available.

- **Middle value = Initial buffer size**
  - Set for typical usage
    - small flows
  - If set to large = maximum value
    - with autotuning will waste memory
    - Can hurt performance
  - typical values
    - tcp_rmem = 87380
    - tcp_wmem = 16384

6th IVS TOW May 2011
Tuning (cont)

• rmem_max, wmem_max
  – Maximum buffer size an application can request
    • setsockopt()
      – SO_SNDBUF
      – SO_RCVBUF

• You do not need to adjust these unless your are planning to use some form of application tuning.

• NOTE: Manually adjusting socket buffer sizes with setsockopt() disables autotuning. Application that are optimized for other operating systems may implicitly defeat Linux autotuning.
Tuning (cont)

• Data Link Layer Configuration
  – MTU – Maximum Transmission Unit
    • Normal user 1500 bytes is sufficient
      – Accounts for 95% of the users of the Internet
    • Present maximum is 9000 bytes
      – Not all equipment on the WAN can support this
      – Tests to determine if the path supports jumbo frames
      – Included in the 2.6.18 kernel, needs to be enabled
  • Use “ifconfig” command
    – determine the interfaces settings
    – to change MTU size if possible
  – Autotuning available
    • /proc/sys/net/ipv4/tcp_mtu_probing (disabled by default)
New Services

- **Internet2 / Geant**
  - Now offer the ability to dynamically request
    - dedicated bandwidth over the long haul network
  - Advantages
    - Use of aggressive transport protocols
    - Maximize end-to-end capacity
    - Errors are not congestion over the long haul network
    - Perfect for real-time and time critical transfers
  - Disadvantages
    - $ required for service
Questions / Comments?

Thank you.