PROCEEDINGS OF SCIENCE

VLBI_UDP

Simon Casey[1]
The University of Manchester, Oxford Road, Manchester, UK
E-mail: `scasey@jb.man.ac.uk`

Richard Hughes-Jones , Ralph E. Spencer, Matthew Strong
The University of Manchester, Oxford Road, Manchester, UK
E-mail: `R.Hughes-Jones@manchester.ac.uk`
E-mail: `res@jb.man.ac.uk`
E-mail: `mstrong@jb.man.ac.uk`

This paper describes the VLBI_UDP application, which has been designed to transport VLBI data using the UDP protocol. Modifications to provide additional features such as file access and packet dropping are described, and results obtained from tests conducted with the application are presented.

Lighting the Blue Touchpaper for UK e-Science - Closing Conference of ESLEA Project
The George Hotel, Edinburgh, UK
26-28 March, 200

[1] Speaker

PoS(ESLEA)038

1. Introduction

e-VLBI requires vast quantities of data to be sent from several outstations (telescopes) over high-speed computer networks and the Internet to a single correlator. Currently, VLBI data rates of 512 Mbit/s are achievable using the Transmission Control Protocol (TCP) [1]. An alternative to TCP is User Datagram Protocol (UDP), which is what VLBI_UDP relies upon.

2. The case for UDP

Whilst e-VLBI in the EVN can run at 512 Mbit/s with TCP, if longer baselines are used, for example across the Atlantic, TCP may struggle to sustain a constant 512Mbit/s should any packet loss occur. TCP guarantees all data sent will arrive and in the right order, but was designed with congestion control algorithms which reduce the transmission rate by half if any packet loss is detected. There are both software and hardware buffers in the Mark5A systems which can compensate for a reduced transmission rate for short periods of time, but extended slow periods would mean the buffers run empty. The higher the round trip time (RTT), proportional to the physical network distance, the longer it takes TCP to recover back to its previous rate after a packet loss event [2]. UDP, on the other hand, does not guarantee delivery of data, and the transmission rate is governed by the user.

3. VLBI_UDP architecture

VLBI_UDP was originally written as a monolothic application by Richard Hughes-Jones for iGrid 2002 as a simulation of the loads e-VLBI places on the networks. It has since undergone several revisions with extra features being added, and these are detailed below. The current architecture is represented graphically in *Figure 1*. There are 3 components to VLBI_UDP, the sending application, receiving application, and the control application. The send & receive components are run as console applications with no user input. The control is also a console application which drives the send & receive components, and is accessed via a variety of methods. It can either take user input from the console, commands via a webpage through a miniature http interface, or read commands from a file with no user interaction. A single instance of the control application controls multiple send/receive pairs.

3.1 Conversion to pthreads

As a monolithic application, VLBI_UDP was consuming almost all available CPU cycles due to constant polling, checking if there is data to be moved around. Clearly this is not an optimal situation, and so the send and receive programs were both split into 3 threads: control, data input and data output. Splitting the application into threads allows each thread to act with a reasonable amount of independence from the other threads, whilst still allowing communication between them.

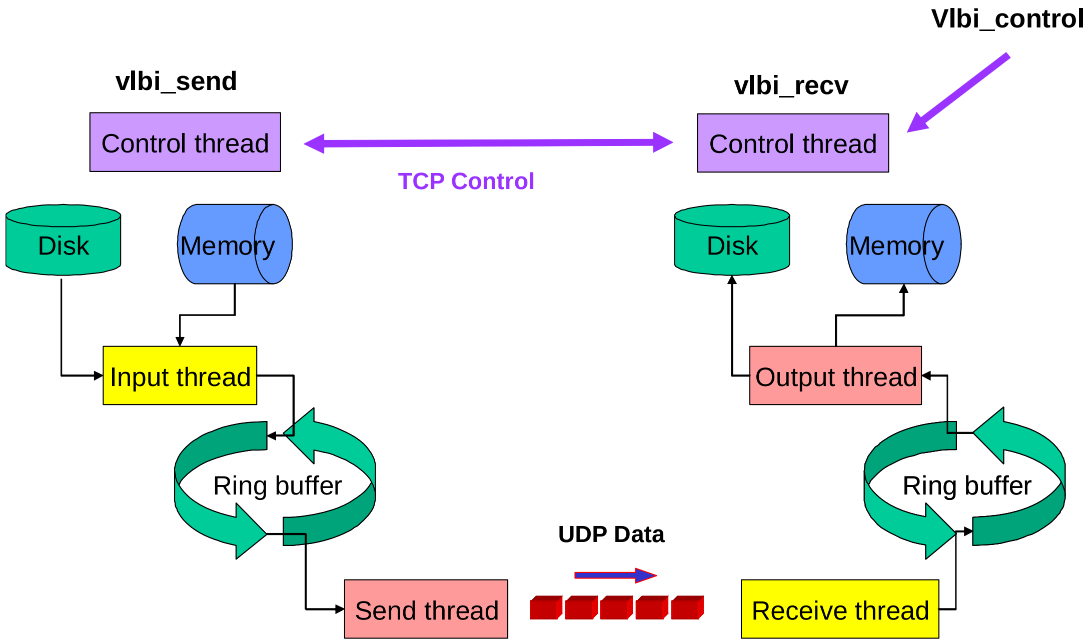

Figure 1: Flow diagram representing VLBI_UDP architecture

3.2 Ringbuffer

A ringbuffer was present in the original iGrid2002 application, but was incomplete so far as it didn't handle out of order packets correctly – not a problem for the demo but needed correcting for use with real data. As each UDP packet is received, it is written directly to the next usable location in the ring buffer. Each packet has a sequence number which allows missing and out of order packets to be caught. If there were one or more packets missing immediately previous to the received packet, then it would be in an incorrect position. A function RingMove() is called, which moves the last packet forward the required number of positions within the ring buffer such that it is then correctly placed. The next available location is now set to after the new location of the last packet.

Should the 'missing' packet(s) subsequently arrive, out of order, then they are first written to the next available location as before. The sequence number is checked, and RingMove() is called with a negative offset to place the packet back where it should have been. In this case, the next available location doesn't change and so the next packet will be written to where the last packet originally arrived. This process is illustrated in *Figure 2*.

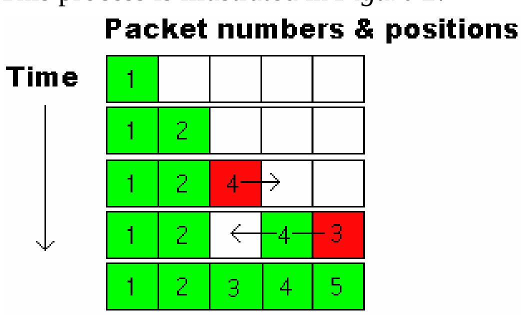

Figure 2: Simulation of packets being placed into ring buffer

3.3 File access

To allow for testing with real data, and as a precursor to interfacing with actual VLBI hardware, file access was implemented. Linux large file support is used, a necessity when dealing with VLBI data sets which are almost exclusively >2GB.

3.4 Packet dropping [3]

A packet dropping function has been added, which, when combined with the file access mode, facilitates the creation of data sets with missing packets under controlled conditions. This function is implemented only in the sender module. The send thread receives a pointer to a packet of data from the ring buffer, passes this to the dropping function as a parameter, along with a choice of dropping algorithm. The return value is a pointer, which will be the same as that passed to the function if the packet was not dropped, else will be a pointer to a randomly initialised portion of memory. Currently there are two algorithms available. The first drops single packets at a steady rate with no randomisation, the second can drop a bunch of between 1 and 10 consecutive packets, the value chosen randomly. To maintain a fractional loss rate f in the 2nd case, after a bunch of n packets are dropped, the subsequent $n(1/f - 1)$ packets are not dropped.

4. Results from tests with VLBI_UDP

VLBI_UDP has been used both as a demonstration tool at events such as iGrid2002, and more recently at the Geant2 network launch, as well as a tool to probe network conditions over extended periods of time. *Figure 3* demonstrates a 24 hour flow, transmitting data from a PC based in Jodrell Bank over a dedicated gigabit fibre connection into a PC based in Manchester University. Each point represents the average received bandwidth in a 30 second period, and it can be seen that rate stability is mostly maintained to 1 part in 1000

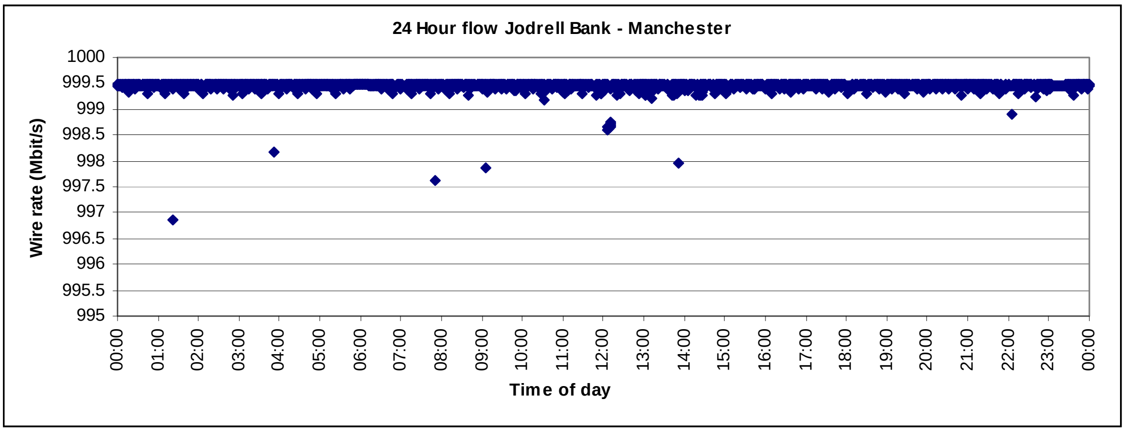

Figure 3: 24 Hour flow from Jodrell Bank Observatory to Manchester University

Figure 4 shows 3 simultaneous transfers into JIVE, one from Manchester travelling over a UKLight dedicated gigabit lightpath, another from Manchester but crossing the conventional packet switched Internet, and the third from Bologna again over the conventional packet switched Internet. The lightpath performed as expected, with the transmit rate purposely capped at 800 Mbit/s and showing almost no packet loss. The second flow was capped at 600 Mbit/s, since this was travelling via the Manchester University campus access link and so would have swamped regular campus Internet traffic. Packet loss is present here due to contention, most likely over the campus access links, and can be seen to decrease through the test period, representing a decrease in campus traffic. The third plot was limited at 400 Mbit/s as the sending PC was underpowered and this was the maximum rate it could transmit at.

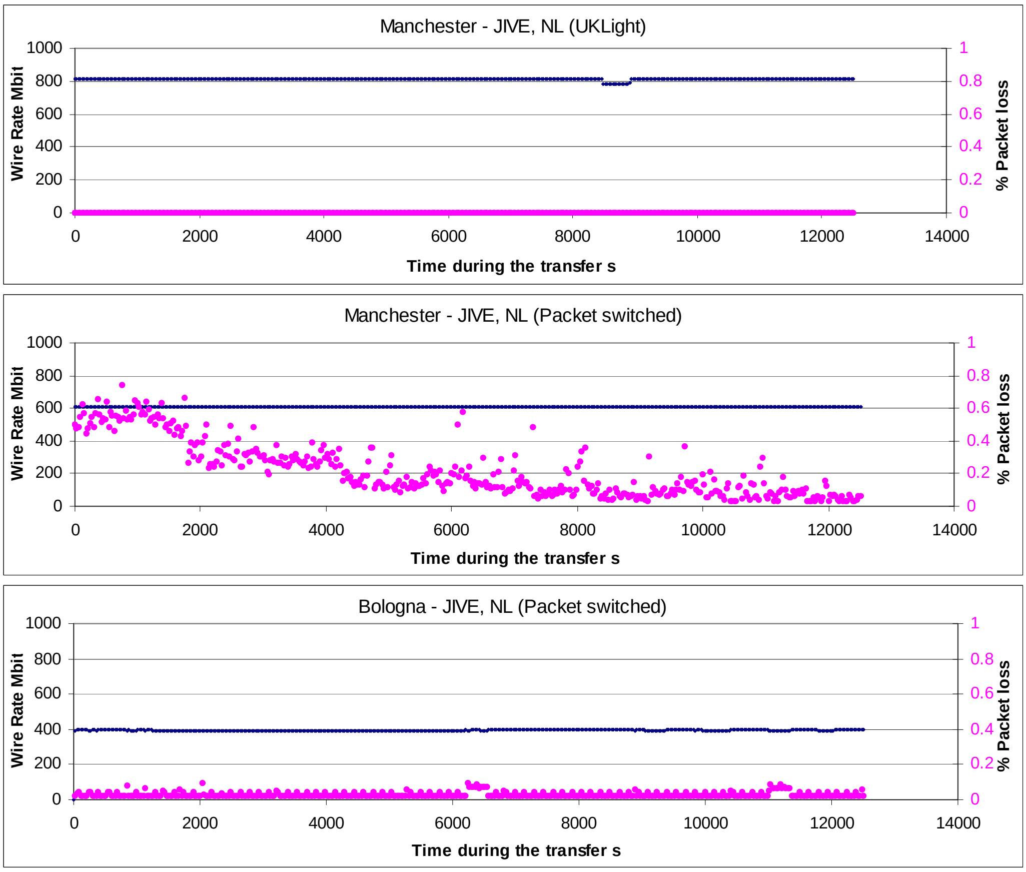

Figure 4: Multiple streams into JIVE

5. Conclusion

As transport protocols for VLBI, both TCP and UDP have their advantages and disadvantages. Currently TCP is a suitable transport protocol, but with the demand for higher data rates and longer baselines, it may be that TCP is unable to keep up and so this paper shows that UDP can provide a suitable alternative.

References

[1] M. Strong et al., *Investigating the e-VLBI Mark 5 end systems in order to optimise data transfer rates as part of the ESLEA Project.* In proceedings of Lighting the Blue Touchpaper for UKe-Science – Closing Conference of ESLEA Project, POS(ESLEA)024, 2007.

[2] R. Hughes-Jones et al., *Bringing High-Performance Networking to HEP users* in proceedings of CHEP04 Interlaken, 27 Sep - 1Oct 2004

[3] S. Casey et al., *Investigating the effects of missing data upon VLBI correlation using the VLBI_UDP application.* In proceedings of Lighting the Blue Touchpaper for UKe-Science – Closing Conference of ESLEA Project, POS(ESLEA)025, 2007.