UDP-like TCP

Mark Kettenis

October 11, 2006

1 eVLBI data transfer requirements

The Mark5A data recorder used by most radio telescopes used for eVLBI in Europe and North America comes with a standard application for doing data transfers over IP networks. The Mark5A application software can use either UDP or TCP for its data transfers, but earlier tests have shown that UDP transfers only works for completely lossless networks. Unfortunately TCP transfers don't work completely satisfactory either, even after tuning some of the Linux kernel parameters to allow for the larger TCP window sizes required for high-bandwidth transfers. This puts a rather severe limitation on the data rates we can use for eVLBI, especially to telescopes to which we don't have a dedicated lightpath, even though enough bandwidth should be available.

There are known limitations in the standard TCP protocol that make it difficult to utilize the full bandwidth of "fat pipes" especially on links with large round trip times. These limitations can probably be circumvented by using UDP-based data transfers, but getting those to work would require significant modifications to the Mark5A application software. Since the Mark5A software is not structured very well, this is something we would rather avoid doing.

There are TCP variants that are designed to work better on high-bandwidth connections. Using one of these variants might make it possible to make better use of the available bandwidth without change the applications software. Unfortunately the Linux kernel used on the Mark5A data recorder does not implement these TCP variants. It also does not provide an easy way to add such a variant. But newer Linux kernels do implement some of the more generally usable variants. They also provide the infrastructure to easily add new TCP implementation as kernel modules. So a small project was started to see whether we could replace the Linux kernel on the Mark5A data recorder and get better data rates using other protocols.

2 Circuit TCP

Since bandwidth tests conducted by others showed that the standard TCP variants included in Linux kernels did not improve bandwidth usage very much, the decision was made to concentrate on testing a somewhat non-standard one: Circuit TCP (C-TCP), which is designed for dedicated end-to-end circuits[1]. The key feature of C-TCP is to maintain a fixed transmit rate. It does this by eliminating TCP slow start completely, and clamping the TCP congestion window size at the appropriate buffer size needed to fill the connection. This effectively throws all attempts to be fair towards other TCP streams on the same connection out of the window. It essentially turns TCP into a connection-based UDP with retransmissions.

C-TCP is designed for dedicated end-to-end circuits. On these links fairness is not an issue. The same holds for the traffic to telescopes that are connected to JIVE through a so-called lightpath. But even for telescopes connected over GEANT it might be an option if enough bandwidth is available and packet loss is mostly occuring at the end hosts.

There is a C-TCP implementation for the Linux kernel, but unfortunately only for a specific kernel version. Since we could not get it to work with any kernel version compatible with the Mark5A software, we decided to write our own implementation. This allowed us to make some simplifications. We ended up with a very small kernel module that used a C-TCP-like protocol for connections to the Mark5A data port, and used TCP-BIC for all other connections.

3 Bandwidth tests

To be able to test a newer Linux a kernel one of the Mark5A's at JIVE was installed with Ubuntu 5.10 (which uses Linux 2.6.17). Attempts to install the current stable versions of the Mark5A application software on that distribution failed. It turned out that this version does not support newer Linux kernels. So we installed a development version of the Mark5A software that did support Linux 2.6.17. This Linux version uses TCP-BIC as its default TCP implementation, which is supposed to be better at utilizing high-bandwidth links than standard TCP.

The performance of this system was tested between two Mark5A's at JIVE, but it was soon realised that the dedicated character of the link and the short round trip time between the systems masked all data transfer problems experienced in real eVLBI operations. So a copy of the software installation was shipped to Medecina, in order to do some meaningful testing.

[1]Mudambi, Zheng, Veeraraghavan

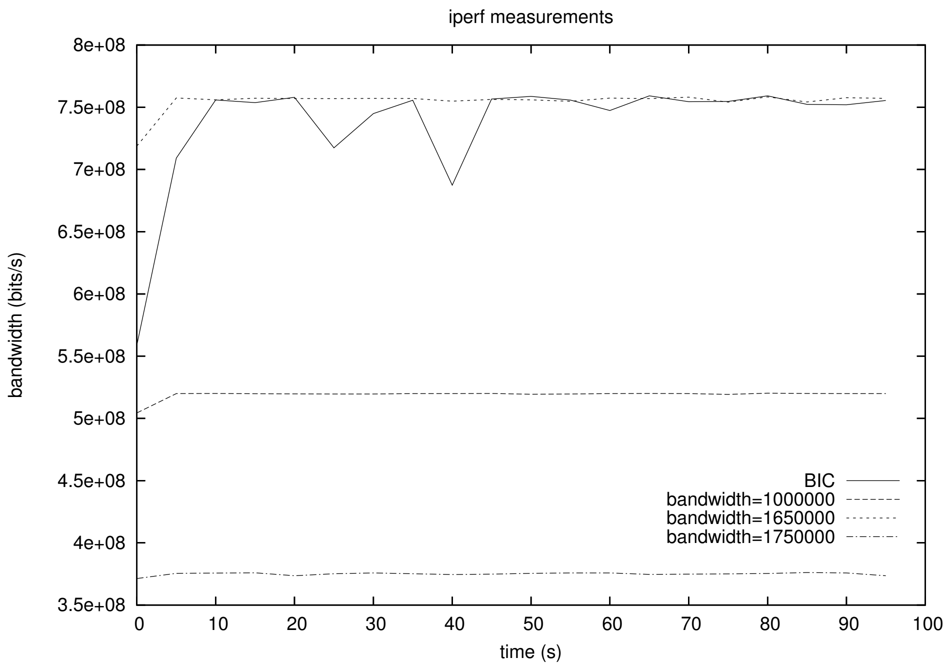

Figure 1: Available bandwidth measured by iperf on the Medecina-JIVE link.

After its arrival, iperf tests were conducted between Medecina and JIVE. Some interesting results were obtained. As can be seen from figure 1 the maximum transfer rate obtained with our protocol module was slightly higher than TCP-BIC. But more importantly, the fluctuations in the transfer rate are much smaller. It also became clear that clamping the congestion window size at the right size is fairly critical. Choosing it too low will needlessly limit the throughput. Choosing it too high will cause too much congestion and will cause the throughput to collapse completely due to retransmissions.

4 eVLBI data transfers

Attempts to transfer VLBI data between Medecina and JIVE failed. There seems to be a bug in the development version of the Mark5A software application that causes the Mark5A to transmit at half the requested rate. This means that the data that is sent to the correlator is corrupted. It is impossible to establish whether the network protocol changes make an actual difference.

5 Conclusions

The UDP-like TCP variant developed shows some promises. It results in flows that seem to be more stable. Whether using unfair TCP variant on the public research network is acceptable remains to be seen.

Unfortunately the new protocol could not be tested with the Mark5A application software. Doing further research on TCP variants is pointless until we have Mark5A application software and hardware drivers that work with newer Linux kernels.