mcp: A Reliable Multicast File Transfer Tool

Karl Jeacle and Jon Crowcroft
University of Cambridge Computer Laboratory
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
{firstname.lastname}@cl.cam.ac.uk

Abstract

Much work has been done attempting to scale multicast transmission to hundreds or thousands of receivers, but typical applications requiring bulk data transfer may involve just ten or twenty sites.

We have been investigating how multicast can be exploited in such an environment and have developed a small-scale sender-initiated reliable multicast protocol called TCP-XM. In order to test and evaluate this new protocol we have used it to build a reliable multicast file transfer tool called mcp.

In this paper we describe the mcp transfer tool, review the design of the TCP-XM protocol, and provide experimental results from tests based on both network emulation and live wide area network transfers between sites on the UK eScience network.

1 Introduction

Multi-gigabyte bulk data transfers to small numbers of destinations are becoming increasingly common. Today's Grid environments, for example, see High Energy Physicists distribute such large datasets.

Using multicast for this type of application can provide significant benefits including reduced load on the transmitter, an overall reduction in network traffic, and consequently shorter data transfer times.

Our work in this area has led to the development of the TCP-XM protocol, and subsequently, as a vehicle for its testing and evaluation, the mcp file transfer tool,

2 TCP-XM

TCP-XM [1] is a reliable multicast protocol based on TCP. It combines reliable multicast data transport alongside the existing conventional unicast transport mechanism offered by TCP.

From a user perspective TCP-XM automatically detects underlying network multicast capability and transparently takes advantage of this potential for increased network efficiency. The protocol will transmit data using multicast when possible, but will fall back to unicast operation in all other circumstances.

Our implementation of TCP-XM has been developed as a modification to the lwIP TCP/IP stack. lwIP is a small independent implementation of the TCP/IP protocol stack that has been developed by Adam Dunkels at the Swedish Institute of Computer Science.

We have built this modified stack as a userspace library running over UDP. This library forms the basis for our mcp reliable multicast file transfer tool. It has also been used to build a reliable multicast transport driver for Globus XIO [2].

3 mcp & mcpd

Our multicast file transfer software has been compiled and testing on Linux, FreeBSD and Solaris. It consists of two programs: mcp & mcpd.

mcp is the client transmitter application. It uses TCP for control connections and TCP-XM (over UDP) for data transfers to mcpd receivers running on one or more hosts. Both TCP and TCP-XM connections are opened in parallel to destination hosts. This can be seen in figure 1.

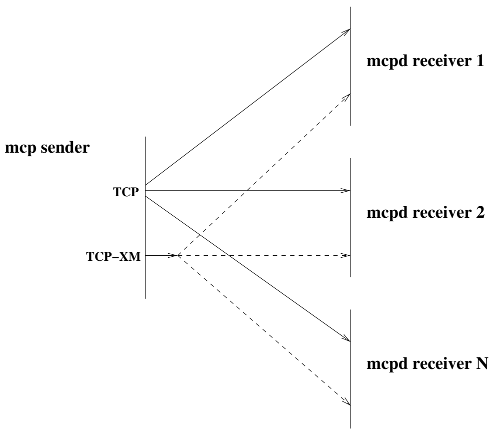

Figure 1: mcp control & data pipes

To transfer a file '`myfile`' to hosts '`host1`', '`host2`', and '`host3`', `mcp` is invoked as follows:
`mcp myfile host1 host2 host3`

The use of multicast transmission is transparent to the user, but a particular multicast group can be specified with the `-g` option. Multicast-only or unicast-only operation with either TCP-XM or TCP can also be forced via a number of other command line options.

On invocation, `mcp` begins by opening TCP connections (on port 22222) to the destination hosts. It sends the name and size of the file to be transmitted, along with the UDP source port it will use for the subsequent TCP-XM data connection, and a suggested UDP destination port for the servers. The servers check for availability of this suggested port and report back to the client with confirmation or an alternative suggestion. `mcp` negotiates with the servers until an agreed destination port has been found.

With UDP ports agreed, `mcp` initiates its TCP-XM engine, and then opens a TCP-XM connection to the destination hosts. The file to be transferred is read from disk and sent on this connection. On completion, the servers send confirmation of the number of bytes received on the TCP control connection. `mcp` then closes both control and data connections.

This sequence of TCP control traffic and TCP-XM data traffic can be seen in figure 2.

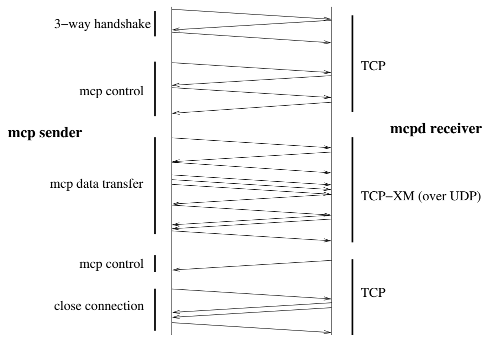

Figure 2: mcp control & data flow

`mcpd` is the corresponding receiver daemon for `mcp`. It listens on TCP port 22222 for incoming control connections (configurable with `-p` flag), and its children subsequently accept incoming TCP-XM over UDP data connections.

4 Performance

We have tested how `mcp` performs on both local and wide area networks, and in network emulation tests where network conditions are varied. Department LANs were used for local testing, while sites around the UK eScience network were used for WAN tests. Bandwidth restrictions, increased latency, and packet loss were all induced for the emulated network tests to assess the impact of adverse conditions on performance.

4.1 Local area tests

This section describes the results of TCP-XM running on n distinct hosts on a local area network.

Figure 3 shows a comparison of the TCP and TCP-XM data transfer rate to n hosts on a local departmental LAN. As would be expected, TCP's throughput declines as the host count increases. TCP-XM peaks at a much lower rate, but then consistently maintains this rate despite the introduction of more destination hosts.

Figure 4 shows the number of bytes being sent on the wire for the same transfer. Because TCP-XM is multicasting, it naturally scales. TCP is sending more and more data

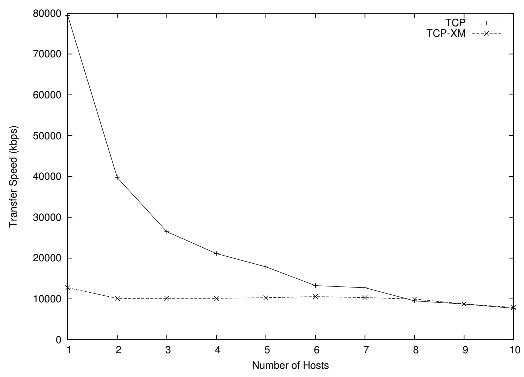

Figure 3: LAN speed

as the host count increases, so performance inevitably suffers. Note that the TCP-XM data is split in two: unicast bytes and multicast bytes. The unicast bytes are barely visible at the bottom of the graph. These account for connection setup, close, and retransmissions. The majority of the TCP-XM data transfer is composed of multicast bytes.

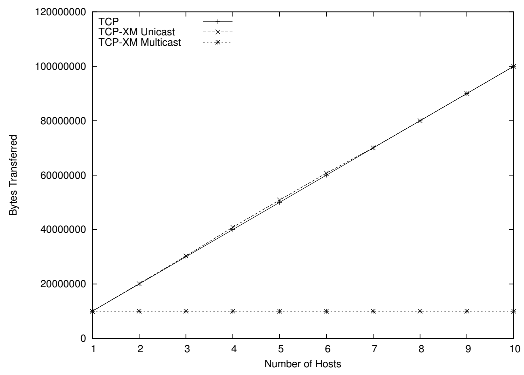

Figure 4: LAN efficiency

4.2 Wide area tests

For wide area testing, shell accounts on machines at eScience Centres around the UK were obtained. These machines and locations were primarily chosen as they are representative of the target audience for our work i.e. physicists requiring bulk data transfer to a relatively small number of regional sites.

The UK eScience Centres are connected via the JANET academic network. Figure 5 illustrates the geographical connections.

Many of these sites have functional multicast

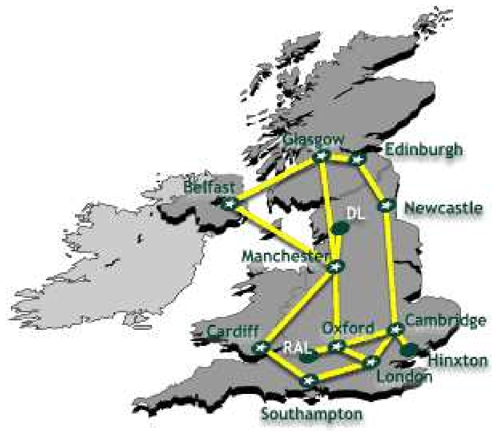

Figure 5: The UK eScience network

connectivity. This is, in large part, due to the frequent use of the AccessGrid video conferencing system.

The specifications, network connectivity and operating systems used by the hosts varies widely from site to site. Some hosts are high speed machines connected close to the WAN backbone. Others are smaller and older departmental machines with poorer connectivity.

Table 1 shows the average round-trip times and transfer rates seen from Cambridge to other sites around the network. The round-trip times vary in range from approximately 5 to 21 milliseconds. The transfer rates attainable on single TCP connections varied from as little as 1.5 Mb/s to over 50 Mb/s.

While this mixture may not be conducive to optimal headline results, it allows a truly representative set of protocol performance results for a live wide area network.

Site	RTT (ms)	B/W (Mb/s)
Belfast	18.6	16.0
Cardiff	13.5	22.4
Daresbury	21.3	28.1
Glasgow	16.2	33.9
Imperial	17.1	78.0
Manchester	9.9	34.5
Newcastle	11.8	1.7
Oxford	7.0	5.7
Southampton	8.8	39.3
UCL	4.9	42.1

Table 1: WAN RTTs & bandwidth

Figure 6 shows how TCP and TCP-XM compare when a transfer to n hosts takes place using a wide area network. As in the local area, TCP can outperform TCP-XM, but less so than might be expected. The inherent bottlenecks present across the WAN, and the varied performance specification of receivers, prevent TCP from achieving the same strong results that are possible on a LAN. For the particular order and set of hosts used in this experiment, TCP-XM can match or outperform TCP for five or more hosts.

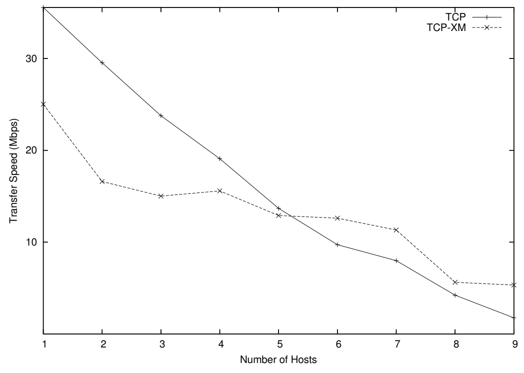

Figure 6: WAN speed

Figure 7 once again shows TCP's inefficiencies as the host count increases. More interestingly, we can see more clearly how TCP-XM is combining unicast and multicast. Unlike the LAN test above, not all destinations are multicast capable, so TCP-XM cannot quickly switch to multicast after connection setup. The number of bytes unicast by TCP-XM is therefore much more significant. There is an obvious step up in unicast bytes sent each time TCP-XM encounters a destination host without multicast.

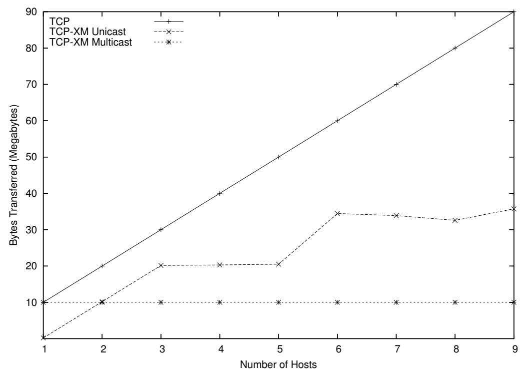

Figure 7: WAN efficiency

We should mention that lwIP currently lacks features such as TCP window scaling. Coupled with some optimisation in our implementation, we would expect considerable improvements in performance to be possible. And, of course, while kernel-based TCP will always have an advantage over a userspace implementation of TCP-XM encapsulated in UDP, the same would not be true of a kernel-based TCP-XM implementation.

In addition, the key benefit from TCP-XM and its multicast capability is not its raw data transfer rate, but its ability to reliably transfer large amounts of data in a far more efficient manner. In an appropriate application domain, end-users may find that the benefits offered by this improved efficiency compensate for any performance penalties incurred by running in userspace.

5 Summary

This paper describes the design and implementation of the `mcp` reliable multicast file transfer tool. It overviews TCP-XM protocol operation and provides detailed experimental results on both a local area network and on the UK eScience wide area network.

Our results show that `mcp`'s use of TCP-XM mean that where multicast is available, and the destination host count increases, large reductions in network traffic lead to a significant increase in network efficiency. The impact of this reduction in traffic can be fully appreciated when network conditions deteriorate. With less traffic to send, TCP-XM can easily outperform TCP when bottlenecks or packet loss are present.

References

[1] Karl Jeacle and Jon Crowcroft. Reliable high-speed Grid data delivery using IP multicast. In *Proceedings of All Hands Meeting 2003*, Nottingham, UK, September 2003.

[2] Karl Jeacle and Jon Crowcroft. Extending Globus to support Multicast. In *Proceedings of All Hands Meeting 2004*, Nottingham, UK, September 2004.