Case Study of TCP/IP tunings for High Performance Interconnects

Jenett Tillotson
Senior Systems Engineer
National Center for Atmospheric Research
Email: jtillots@ucar.edu

1. Introduction

TCP/IP remains a central protocol for high performance computers. Even when user codes are utilizing RDMA for communications, TCP/IP is often required for communications with system infrastructure such as parallel file systems or authorization servers. Supercomputers with non-Ethernet interconnects often have an TCP/IP stack configured such as the Internet Protocol over Infiniband network layer. There are very few supercomputers that do not use TCP/IP as a communications layer. While most system professionals understand how to tune the TCP buffer sizes in order to get better performance, there are many other TCP/IP tunings that must be considered in order to get a high performing TCP network.

This lightning talk will present several suggestions for tuning TCP/IP in high speed, low latency interconnects. I will use as an example the TCP/IP tunings for Cheyenne, a 4,032 node HPE/SGI ICE machine with a 100Gb Infiniband interconnect. I will discuss these tunings and present suggestions for how system professionals can implement these tunings at their sites.

2. TCP/IP Tuning Best Practices

There are many good websites that will help a systems professional tune the TCP buffers for a high speed, low latency network. [1] [2] Tuning these buffers will greatly improve communications between nodes in the supercomputer resulting in a higher performing network and a more stable system. Another easy tunable the Address Resolution Protocol (ARP). Linux is tuned by default with a small home or business network in mind. The ARP table size limits and garbage collection intervals are too small for large networks. For many HPC networks, these limits need to be increased to keep the system from experiencing ARP storms due to the ARP garbage collector. Another common mistake sites make is having empty host files on the compute nodes. Unless your host file is changing on a daily basis, it is more efficient to create local host files. This will cause the local compute node to do the majority of the host to IP lookups rather than overwhelming the local Domain Name Service infrastructure. The author personally knows of systems that have improved stability with this simple best practice. Lastly, we have done some exploring with SYN Cookies and Selective Acknowledgements (SACK).

3. Cheyenne TCP Tunings

```
net.core.somaxconn=12000
net.core.netdev_max_backlog=250000
net.core.optmem_max=4194304
net.core.rmem_default=4194304
net.core.wmem_default=4194304
net.core.rmem_max=4194304
net.core.wmem_max=4194304
net.ipv4.neigh.default.gc_thresh1=12216
net.ipv4.neigh.default.gc_thresh2=14216
net.ipv4.neigh.default.gc_thresh3=14964
net.ipv4.neigh.default.gc_interval=200000
net.ipv4.neigh.ib0.gc_stale_time=200000
net.ipv4.neigh.default.ucast_solicit=9
net.ipv4.neigh.default.mcast_solicit=9
net.ipv4.neigh.ib0.mcast_solicit=9
net.ipv4.conf.all.arp_filter=1
net.ipv4.conf.all.arp_ignore=1
net.ipv4.tcp_max_syn_backlog=16384
net.ipv4.tcp_adv_win_scale=1
net.ipv4.tcp_low_latency=1
net.ipv4.tcp_mem=16777216 16777216 16777216
net.ipv4.tcp_rmem=4096 87380 6291456
net.ipv4.tcp_reordering=3
net.ipv4.tcp_timestamps=0
net.ipv4.tcp_window_scaling=0
net.ipv4.tcp_tw_reuse=1
net.ipv4.tcp_syn_retries=12
net.ipv4.tcp_sack=1
kernel.dmesg_restrict=0
```

4. Conclusion

TCP/IP tuning remains a difficult area for large, high performance networks. Case studies such as Cheyenne can be useful to other sites and can help our community build best practices for these specialized networks.

References

[1] *Enabling High Performance Data Transfers: System Specific Notes for System Administrators (and Privileged Users)*; Pittsburgh Supercomputing Center; http://www.psc.edu/tcp-tune.

[2] *Host Tuning*; Energy Sciences Network; https://fasterdata.es.net/host-tuning/background/.