Network Performance Troubleshooting
March 2008

Lab 1

Sequence 1

1. download `server.c` from the location specified by the instructor
2. review the code with the instructor and build the programme using

   ```
   $ cc -g -Wall -o server server.c
   ```

3. run the server

   ```
   $ ./server
   ```

4. use another terminal to inspect the state of the server socket, run

   ```
   $ netstat -tan
   ```

5. use yet another terminal to connect to the server

   ```
   $ telnet <IP address of server> 3490
   ```

6. inspect the state of the server and client sockets again as you did in 4.
7. use `lsof` to inspect the server resources

   ```
   $ lsof | grep server
   ```

8. kill the server and immediately check the status of the socket, note the new status
9. quickly try to restart the server programme. Does it start? Why?

Sequence 2

1. stop the server and wait until the socket is destroyed
2. run tcpdump to start capture:

   ```
   # tcpdump -s0 -w lab1.pcap host <IP addr of server> and port
   3490
   ```

3. inspect the result of the capture with tcpdump

   ```
   $ tcpdump -vv -r lab1.pcap
   ```

4. inspect the result of the capture with wireshark

Sequence 3

1. inspect kernel statistics with netstat:

   ```
   $ netstat -s
   ```

2. inspect driver statistics using ethtool

   ```
   # ethtool -S eth0
   ```

3. inspect the txqueuelen using ifconfig

   ```
   # ifconfig
   ```

4. inspect the status of the hardware assisting mechanisms

   ```
   # ethtool -k eth0
   ```

Lab 2

Sequence 1

1. In coordination with the instructor, locate a remote server on the network
2. On this server, create a random file on a tmpfs filesystem to avoid disk IO interference with the measurements

   ```
   # mkdir /mnt/fs
   # mount -t tmpfs -osize=XXM tmpfs /mnt/fs
   # dd if=/dev/zero of=/mnt/fs/dummy
   ```

3. Disable the TCP window scaling and download the file with scp:

   ```
   # echo 0 > /proc/sys/net/ipv4/tcp_window_scaling
   # scp server:/mnt/fs/dummy /dev/null
   ```

4. Re-enable TCP window scaling and download the file again
5. Use ping to get an estimation of the round-trip time (RTT)
6. Calculate the optimal TCP window size and adjust the buffer sizes accordingly:

   ```
   # echo '39321600 39321600 39321600' >
   /proc/sys/net/ipv4/tcp_rmem
   # echo 39321600 > /proc/sys/net/core/rmem_max
   ```

7. Redo the file download test and note the download speed

Sequence 2

1. In coordination with the instructor select a source package like samba, the kernel or text-utils
2. Export an NFS file system from a server
3. On separate terminals run:

   ```
   # watch 'cat /proc/net/rpc/nfsd | grep th'
   # watch 'cat /proc/net/snmp'
   # watch nfsstat
   ```

4. Mount the exported NFS file system as a version 2 client and build the source while timing the Make and watching the statistics. Take a snapshot of NFS statistics before and after the run
5. Delete the files and unmount the NFS file system to drop all caches
6. Mount the exported NFS file system as a version 3 client with 32 KB read/write sizes and build the source while timing the Make and watching the statistics. Take a snapshot of NFS statistics before and after the run

7. Install the RHEL 5.2 candidate kernel supplied by the instructor and redo step 5. and 6.