Design:

We initialize server programs on each of our vms, which establish sockets and wait for connection. Upon execution of the client (grepping) program, we specify the arguments to pass to grep. We then connect sockets to all available vms, using hard-coded hostname strings for vm01-07. If any of these connections fails, we record the failure internally, as well as printing a relevant message to cerr, and continue with the execution of the code. Our program then passes these arguments via the established sockets to the instances of the server function, which then run exec(grep -Hn $provided\_arguments). We apply the -Hn arguments to give the indication of which logfile yields the printed grep results. The results of the exec call are returned via the same socket back to the client, which has created threads to receive the servers’ responses. Once all of the threads (and responses) have completed, using pthread\_join, the returned strings are then run through std::count to determine the number of lines in each response. In our checked-in code in git, we print the received strings to terminal in our server code, but while running time tests, we have disabled this feature so as to not let terminal IO dominate the time taken to transfer the data via the sockets. We then print the numbers of lines received from each logfile to the terminal, followed by a summed total of the number of matched lines among all active server programs. If we failed to connect a server previously, or if one went down while we were awaiting a response, the failed status of the specific server will be put out to terminal at this point as well.

Testing:

We have run several tests manually, testing our program against a local version of grep, scanning the same log files to ensure we get the same results. For simplicity’s sake, we have used grep’s -c capability to establish our count correctness. We have tested with only one running server, and with all seven running. We have run tests where grep returns no results whatsoever, tests where one server returns several results, and others return zero, and tests where every server returns several results (grepping [-]). After running several of these tests, we established our connectivity protocol as explained above, to prevent any of the mistakes we have noticed while testing.

Timing:

Time to run dgrep on “htm” with 4 60MB log files on 4 other systems: 0.614 seconds real

Time to run local grep: 0.992 seconds real

Time to run dgrep on “aiff” with 4 60MB log files on 4 other systems: 0.047 seconds real

Time to run local grep: 1.006 seconds real

Time to run dgrep on “[^$&=]” with 4 60MB log files on 4 other systems: 2.906 seconds real

Time to run local grep: 0.830 seconds real