CS244B Sprint 2020 Project
Joseph Allen (SUNetID: joeallen)

Debugging Logs in a Distributed System

Logsync (https://github.com/joecroninallen/logsync)

My final project was to make a utility called “logsync” for viewing multiple timestamped log files in sync. The idea was inspired by my interest in learning more about the Tendermint algorithm, and that led to my wanting to see how each of the nodes interact. Naturally, I wanted to be able to see what each node was doing roughly around the same time. My plan was to investigate Tendermint in more detail and experiment changing parameters and analyze the consequences. In the end, my interest in developing a log viewing utility that I could continue to develop and maintain after the class became the primary focus, and the Tendermint logs served mainly as a test case for my log utility tool.

For a little background, in my days as a backend server programmer, I would run into bugs and would want to look at what the server nodes were doing at the same time the client crashed or some error was being reported. Maybe I would have a tcp dump and a log file and I would want to see roughly around the same time what was happening in each node.

I have often found myself using mssh to run a grep command looking for things in the logs of different servers. I recently worked at a big software company that for some political reason got rid of the 3rd party logging tool that we had in production. I would have loved to have a utility like logsync when investigating various bugs. At my current job, we run everything in a docker container so it's particularly easy for me to use the tool as is because I hard coded the default timestamp format that Docker uses. Eventually I will make the timestamp format configurable so it can support log files with different types of timestamps. For example, maybe one log file just has a unix time in millisecond, while another one as YYYY-MM-DD-hh-mm-ss-millsecond type form.

The basic idea for the logsync utility is that you feed it multiple log files. It could even be a server log file and a tcp dump. The idea is that you give it some log files where most of the lines are timestamped. You can then enter a timestamp to jump to, or you can enter a number to indicate the number of steps to jump, and every time you hit enter, it jumps that many steps. By step, it means looking at the next timestamp across all the log files being viewed and advancing the next one by one log line. If you type in a negative number, then you step backwards.

I made logsync as a terminal based program using ncurses. This is nice because it can be portable and easy to run from a distributed server. If that server is set up with a NFS that has access to different server log files, it doesn't even matter if the log files live on that machine. I tested logsync on Debian (buster) and my macbook, and it magically worked on both.

My code has been documented for the purposes of the CS244B teaching staff to get a gist of what I did. Please refer to the github repository here:
https://github.com/joecroninallen/logsync

The README explains how to build it and run it against sample log files, but I will write out the steps to make things easy:

1) git clone git@github.com:joecroninallen/logsync.git
2) cd logsync
3) make build
4) ./logsync test_data/logs/*

This loads the 4 sample files and starts them all at the beginning, so what you see are the first log lines of each of the Tendermint nodes. At this point, there is an edit box

below, and here are the commands that you can type, as explained in the README:

1) "head" jumps all files to the beginning
2) "tail" jumps all files to the end
3) Any positive number jumps that many steps, where each step chooses the next log line based on time stamp and advancing that file foward one.
4) Any negative number goes back that many steps.
5) Also it is possible to search based on a timestamp like "2020-05-25|08:47:33.663" to jump to the closest log line for all the files

As I continue to develop the tool, I will later add the ability to grep the files and then jump to that location in one of the files, and have all the other files go to that corresponding timestamp. I will also allow you to select one of the panes and zoom into that log file and zoom out. Also, I want it to work with log-rotated files that all live in one directory. You might also notice that I don't copy and paste yet with this tool, so that is up next. Most importantly, I will make it easier to enter a search time by making the UI more sophisticated. Because each log file might have different ways of showing a timestamp, that is another thing to configure as I mentioned. Eventually the goal will be to make this production quality and put it out there for people to use and enjoy. I personally will take advantage of it on day 1 at my current job.