Sean DiGirolamo

Qiang Zeng

CIS 3207

01 April 2018

Networked Spell Checker Documentation

This program starts a server which can be connected to by various clients such as telnet. By default, it has the capacity to service 10 clients at a time, and 10 workers working on spell checking at the time. This can be changed by editing the CONNECTION\_CAPACITY and NUM\_WORKERS macro constants in main.c though.

The spell checker takes words over a network from a client, and then searches for them in a dictionary file, If it is in the dictionary file, it sends “<work> OK” back to the client, and if not, it sends “<word> MISSPELLED” to the client.

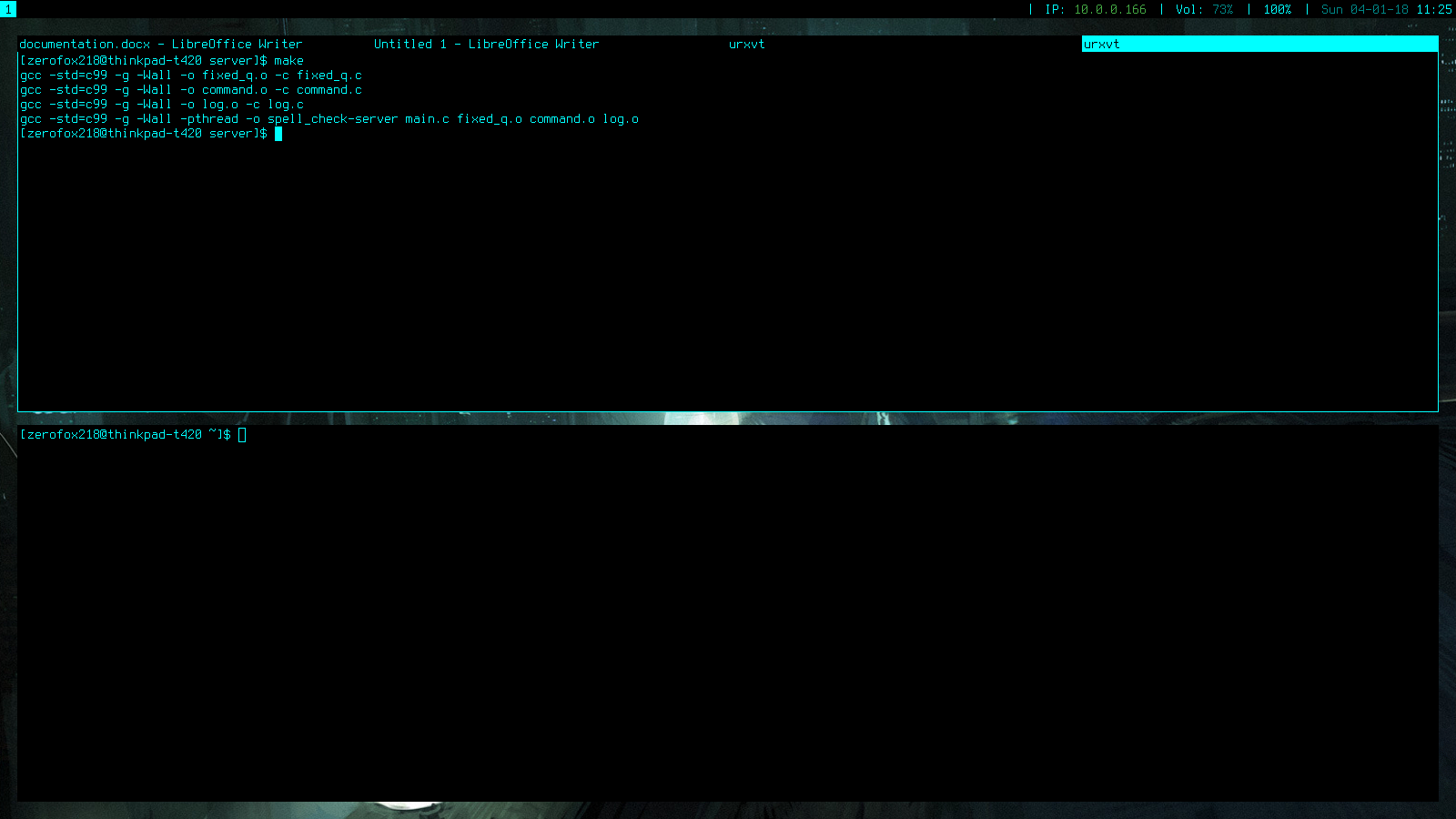
To start the server, simply run the binary file. By default, it will use the dictionary file in /usr/share/dict/words, and the port 9002. If the dictionary is not found, then the program will exit. If the port is not available, the program will exit. You can change the dictionary and port by executing “<binary> [-d dictionary\_file] [-p port\_number]”. Both -p and -d are optional though and if either are missing, the program will use the default value of whichever is missing.

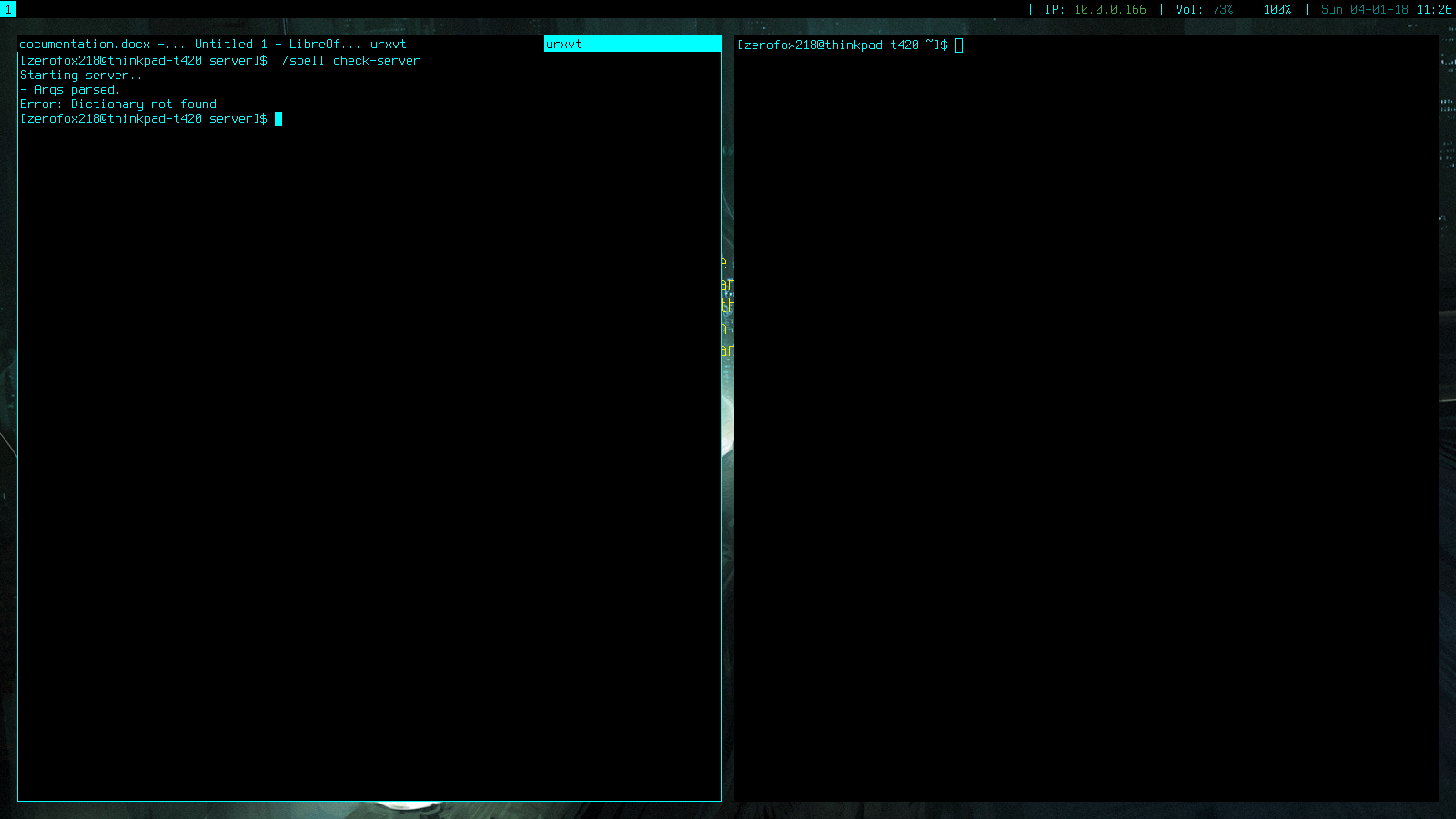
Once executed, the program will load the dictionary, set up a listening port on the specified port, and create some necessary threads. A number of worker threads will be created to work on client jobs when they arrive. In addition, a logger thread will be created that will log any server activity in log.txt, in the same directory that the server program was run from. After this, the server will then begin listening for clients. When it recieves one, it will list it as a job for workers to spell check and send their results to the client.

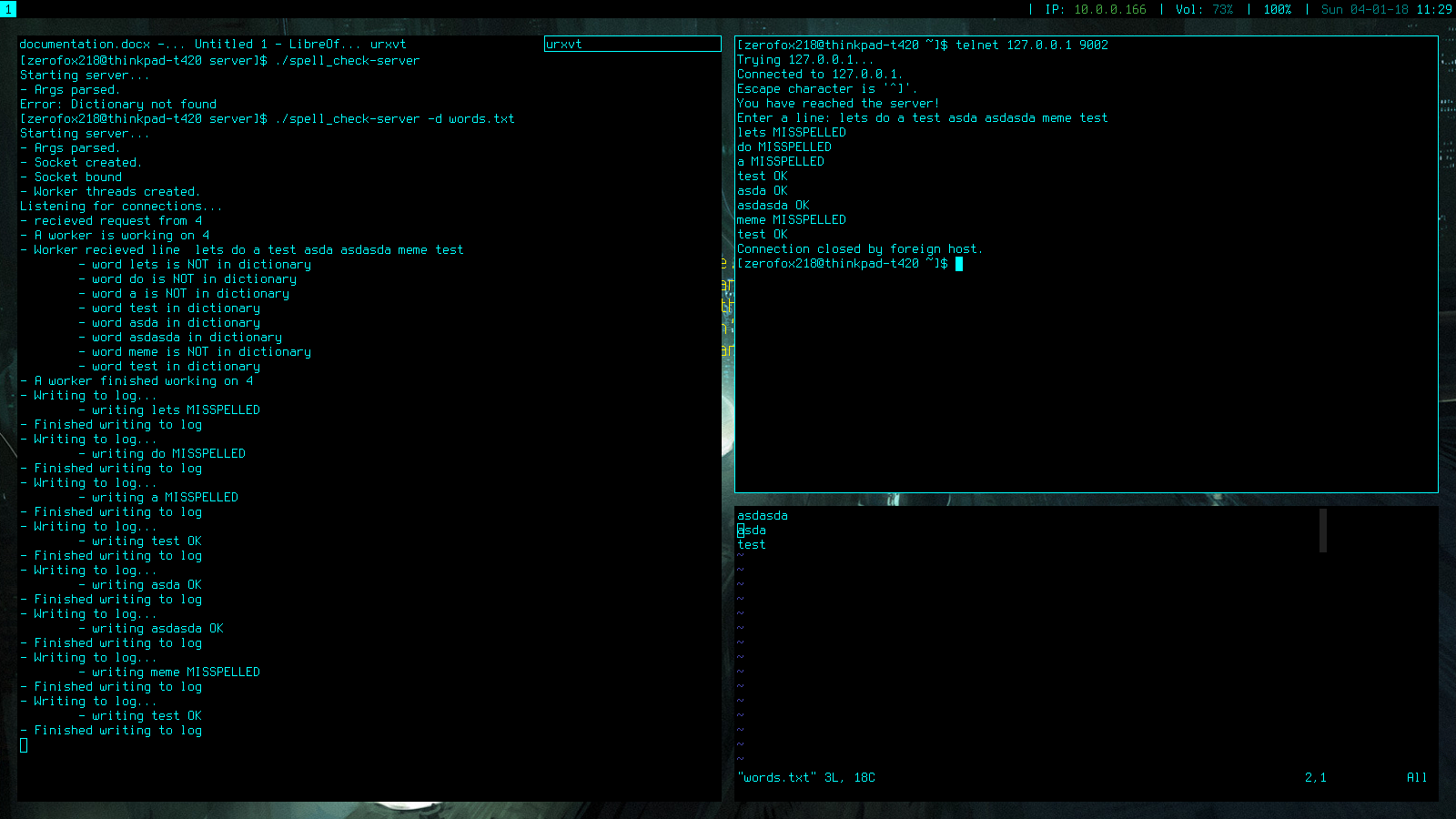
There are many programs capable of connecting to the server. I recommend telnet though. To connect with telnet, simply run “telnet <ip/domain> <port>”. Once connected, the server will prompt the client to enter a word. Once a word is entered, the server will send a message to the client indicating whether or not each word was spelled properly

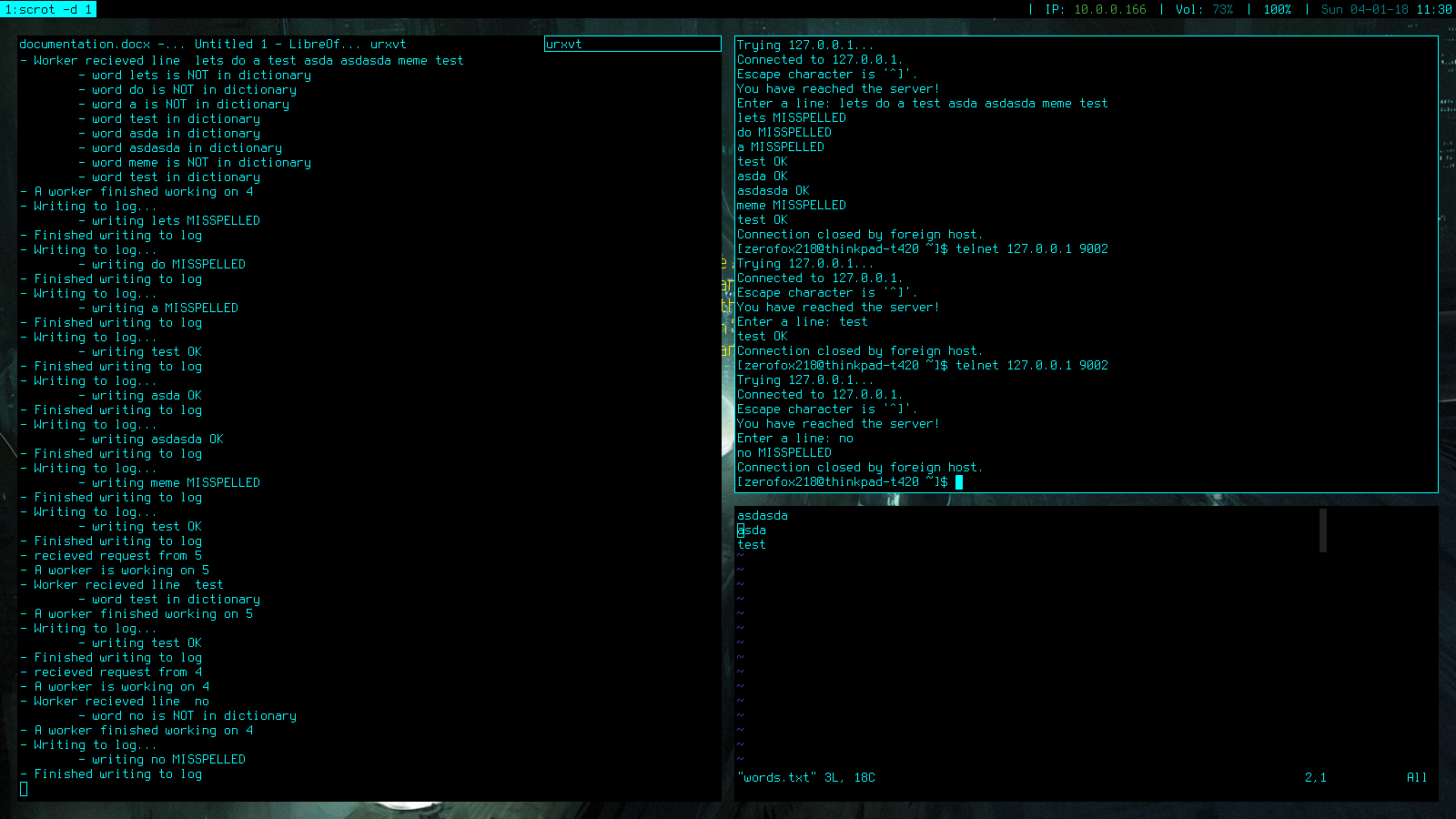
I tested this program by running it in a variety of different ways. One way was by running it with a missing dictionary, to which to program printed an error and exited. After that, I tried running it normally, and then sending a multiword string to it. The server properly spell checked each individual word and printed it to the client. Next, I tried doing so with a single word. The single word was properly spell checked and sent to the client. After that, I attempted running multiple client connections at the same time. The program properly was able to service each client.

All in all, the program seems to be doing everything it is supposed to do. Screenshots can be found below.

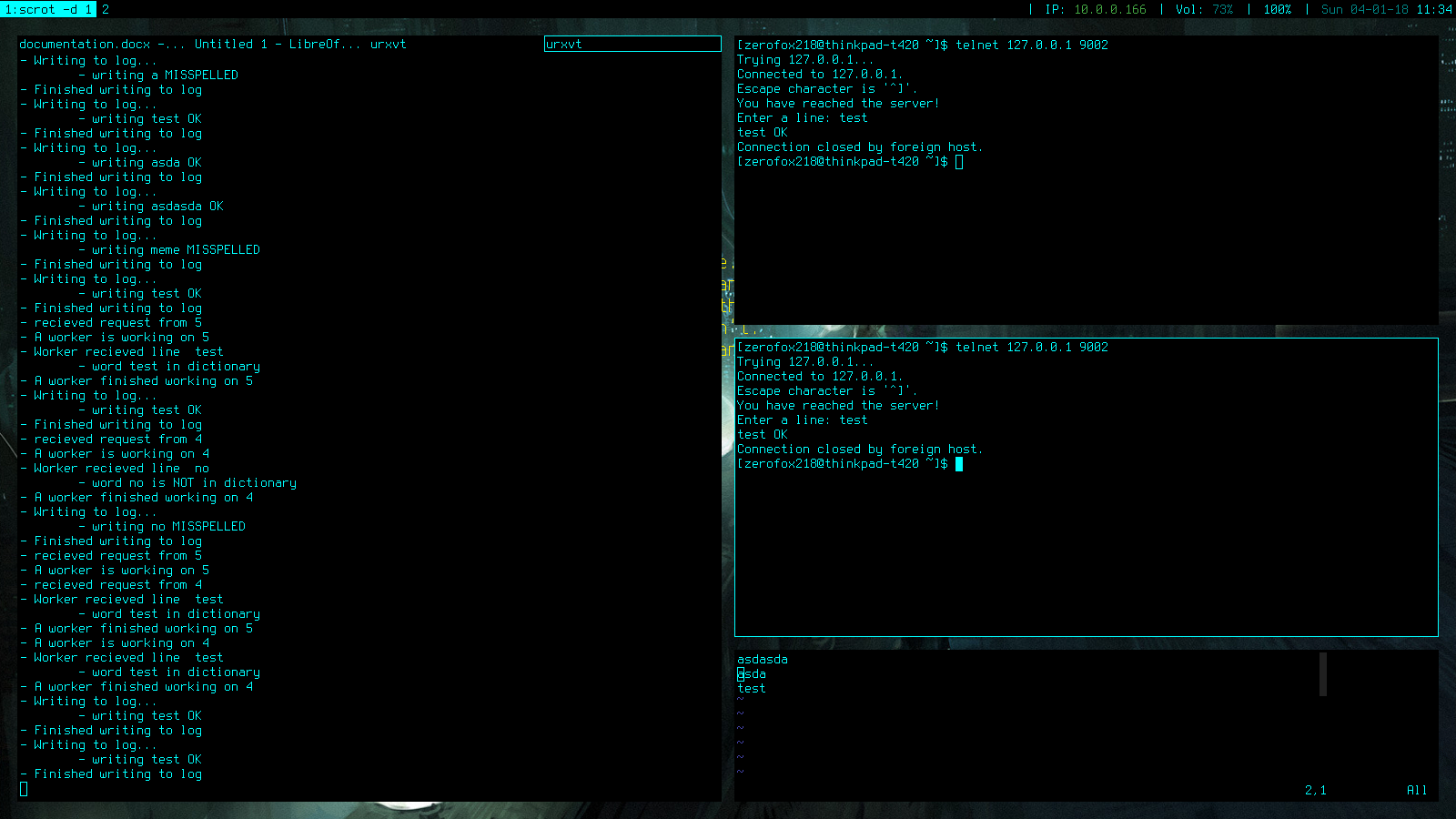










Sss

ss

ss

