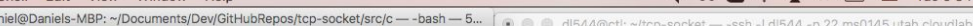


# Project #2

# Daniel Ernesto Lopez Barron


ID: 16215333

This shows the different hosts that I used to test the Project 2. I used the `hostname` command to output the domains used to test the project. The right window runs the client application in my local machine, whereas the left window shows the server application on a remote session.



The screenshot displays two terminal windows side-by-side. The left window is a local terminal on a Mac, showing the command `hostname` and the output `daniels-mbp.kc.umkc.edu`. The right window is a remote terminal session connected to a cloud instance, showing the command `hostname` and the output `ctl1.nidan.nosql-json-pg0.utah.cloudlab.us`.

In this image we can see the content of the local file `hello.txt`, which we will send to the server by using the tag `:file` in the command line and then indicating the file path. Furthermore, the next image illustrates how the file is received in the server side, by reading the `:file` tag, the server knows that a file is being transmitted. Once the transmission is over, both client and server stop the communication and close the connection.



The screenshot displays two terminal windows side-by-side. The left window, titled 'Terminal', shows a user named 'daniel' at a prompt 'daniel@daniels-mbp: ~ - -bash-- 57x34'. The user has opened a file named 'hello.txt' and its contents are displayed: 'Hello world, My name is Daniel and I want to try this example, and see if it is possible to send the text file to the server with out any trouble. Wish me luck! Daniel'. The right window, titled 'd1544@ctl: ~ - -ssh -l d1544 -p 22 ms0145.utah.cloudlab.us - 57x34', shows a user named 'd1544' connecting to a remote server. The user runs the command 'python server.py ctl.nidan.nosql-json-pg0.utah.cloudlab.us 9999', which sends the content of 'hello.txt' to the server. The output of the script is shown as 'client> hello', 'client> my', 'client> name', 'client> is', and 'client> Daniel'.

The following image shows how to communicate with the server, where the server will echo the message coming from the client, but as uppercase text. Also, if the client chooses to stop the communication, it can send the flag `$die`, and both server and client will stop.

```
[daniel@ctl:~]$ ./client ctl.nidan.nosql-json-pg0.utah.cloudlab.us 9999
Please enter msg: Hello_again
server> HELLO_AGAIN
Please enter msg: $die
Closing the socket
daniel@ctl:~$
```

```
d1544@ctl:~/tcp-socket$ python server.py ctl.nidan.nosql-
json-pg0.utah.cloudlab.us 9999
client> Hello_again
client> Client requested closed connection
server> Server closed
d1544@ctl:~/tcp-socket$
```

```
Terminal Shell Edit View Window Help
c — daniel@Daniels-MBP: ~/Documents/Dev/GitHubRepos/tcp-socket/src/c — -bash — 5...
daniel@Daniels-MBP: ~/Doc... daniel@Daniels-MBP: ~/Doc... daniel@daniels-mbp: ~ — -... +
daniel@:c$ hostname
daniels-mbp.kc.umkc.edu
daniel@:c$ ./client ctl.nidan.nosql-json-pg0.utah.cloudlab.us 9999
Please enter msg: hello
server> HELLO
Please enter msg: my
server> MY
Please enter msg: name
server> NAME
Please enter msg: is
server> IS
Please enter msg: Daniel
server> DANIEL
Please enter msg: :file
Indicate file's path: /Users/daniel/hello.txt
INFO> File stored as: /Users/daniel/hello.txt
daniel@:c$ |

client> hello
client> my
client> name
client> is
client> Daniel
:file
Hello world,
My name is Daniel and
I want to try this example, and see if
it is possible to send the text file to
the server with out any trouble.

Wish me luck!

Daniel

:eof
# READING A FILE
Hello world,
My name is Daniel and
I want to try this example, and see if
it is possible to send the text file to
the server with out any trouble.

Wish me luck!

Daniel

>> A Friendly message from the server

client> Client requested closed connection
server> Server closed
dl544@ctl:~/tcp-socket — ssh -l dl544 -p 22 ms0145.utah.cloudlab.us — 57x34
dl544@ctl:~/tcp-socket — ssh -l dl544 -p 22 ms0145.utah.cloudlab.us
```

The following image shows the content of both server and local client. In the server side (right part of the image) a temporary file called tempFile.txt gets created with the content of the client side plus a message appended by the server application. Similarly, the local client's file hello.txt (left part of the image) gets overwritten with the content from the server

```
Terminal Shell Edit View Window Help
c — daniel@Daniels-MBP: ~/Documents/Dev/GitHubRepos/tcp-socket/src/c — -bash — 5...
daniel@Daniels-MBP: ~/Doc... daniel@Daniels-MBP: ~/Doc... daniel@daniels-mbp: ~ — -... +
daniel@:c$ cat /Users/daniel/hello.txt
Hello world,

My name is Daniel and
I want to try this example, and see if
it is possible to send the text file to
the server with out any trouble.

Wish me luck!

Daniel

>> A Friendly message from the server
daniel@:c$ |

dl544@ctl:~/tcp-socket$ ls
client  README.md  src
Makefile  server.py  tempFile.txt
dl544@ctl:~/tcp-socket$ cat tempFile.txt
Hello world,
My name is Daniel and
I want to try this example, and see if
it is possible to send the text file to
the server with out any trouble.

Wish me luck!

Daniel

>> A Friendly message from the server
dl544@ctl:~/tcp-socket$
```