Jonathan Metzger CS4513 – Hugh C. Lauer April 4th, 2018

PROJECT 2: DISTRIBUTED SHELL

1. WALK THROUGH

A. Terminal 1: Server (Local and AWS)

i. Port is optional, default is 4513

```
$  ./server -p 1234
'user1' is accessing information with the 'ls' command!
'user1' entered password incorrectly.
```

B. Terminal 2: Client (localhost and 18.222.16.160)

i. Port is optional, default is 4513

ii. Password is optional, default asks you for the password

```
$  ./client -s localhost -p 1234 -u user1 -w pass1 -c "ls"
$  ./client -s 18.222.16.160 -p 1234 -u user1 -w pass1 -c "ls"
Username: user1
Password: pass1

 ** Password is correct. Accessing information. **

$  ./client -s localhost -p 1234 -u user1 -w pass2 -c "ls"
$  ./client -s 18.222.16.160 -p 1234 -u user1 -w pass2 -c "ls"
Username: user1
Password: pass2

** ERROR: Credentials don't match. Exiting... **
```

2. DESIGN

A. Programs / Scripts

i. Common.c

1. *void useage();*
2. *void flagCheck(int argc, char** argv);*
3. *char* concat(char* s1, char* s2);*
4. *Int containToken(char* recevieMessageFromClient, int size);*

ii. ERROR Handlers

1. *ERROR_socket_call();*
2. *ERROR_connect_call();*
3. *ERROR_bind_call();*
4. *ERROR_listen_call();*
5. *ERROR_accept_call();*
6. *ERROR_waitpid_call();*
7. *ERROR_execvp_call();*

iii. Client.c

1. *getHost();*
2. *checkServer();*
3. *sendMessageToServer(char* message, int sock);*
4. *receiveMessageFromServer(int sock);*
5. *checkUsername();*
6. *checkPassword();*
7. *checkCredentials();*
8. *submitInput(int sock);*

iv. Server.c

1. *countInputs(char* command);*
2. *void storeInputs(char** command_array, char* command, int size_array);*
3. *void setServer();*
4. *void openServer();*
5. *int sendMessageToClient(char* msg, int sock);*
6. *char* receiveMessageFromClient(int sock) ;*
7. *void getUsername() ;*
8. *char* validUsername(char* name) ;*
9. *void getPassword() ;*
10. *char* validPassword(char* name) ;*

B. How many runs performed

I performed 20 rounds each to test for Correct and Incorrect password connections. I did for both the local server and AWS. I averaged each test to get an accurate representation of how long it takes for a connection between client and server. I ran these tests on “ls” and “mv” commands to test to see how the server reacted. For the “mv” command I created 20 files ranging from 1MB to 20MB with random data to see how the file size affects connection speed.

C. Recorded Data

I recorded the data for 20 attempts for each test. I graphed the data to represent change between constraints.

D. System Conditions

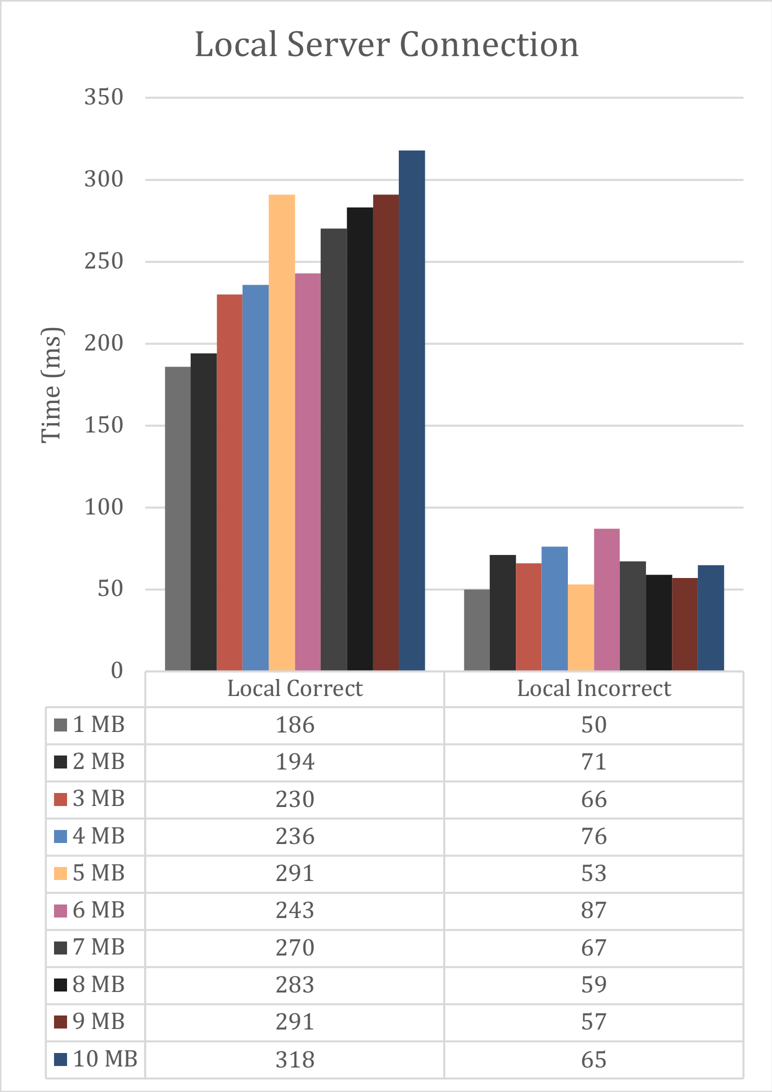
I did each test on the Ubuntu system provided by the professor.

E. Other Information

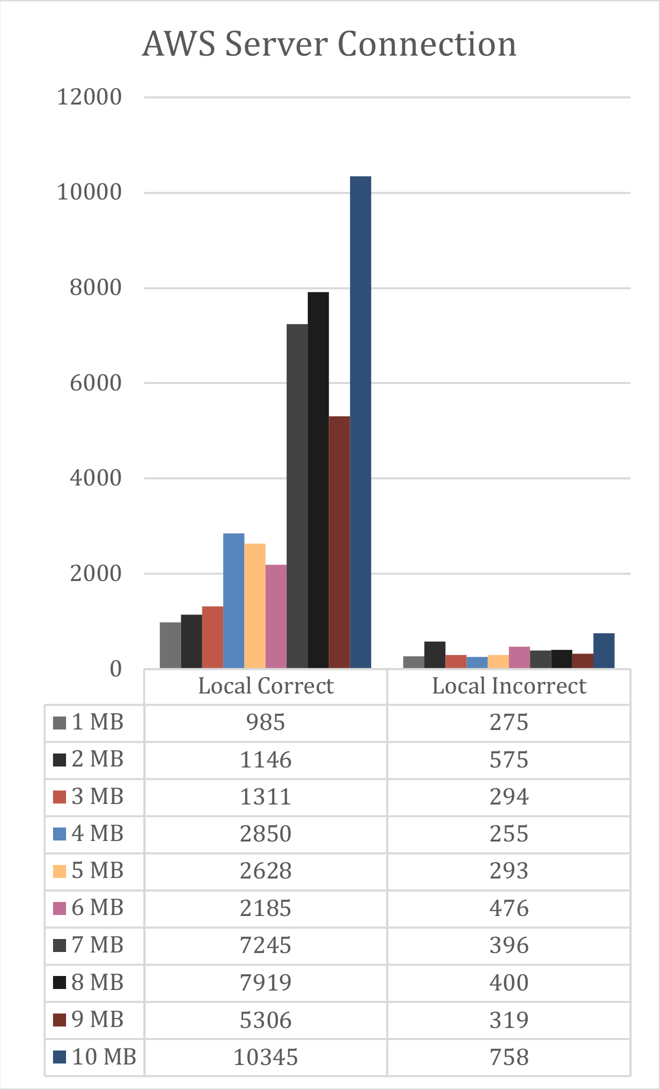
3. RESULTS

A. Tables or Graphs

i. Network per-byte Cost (making file on server)



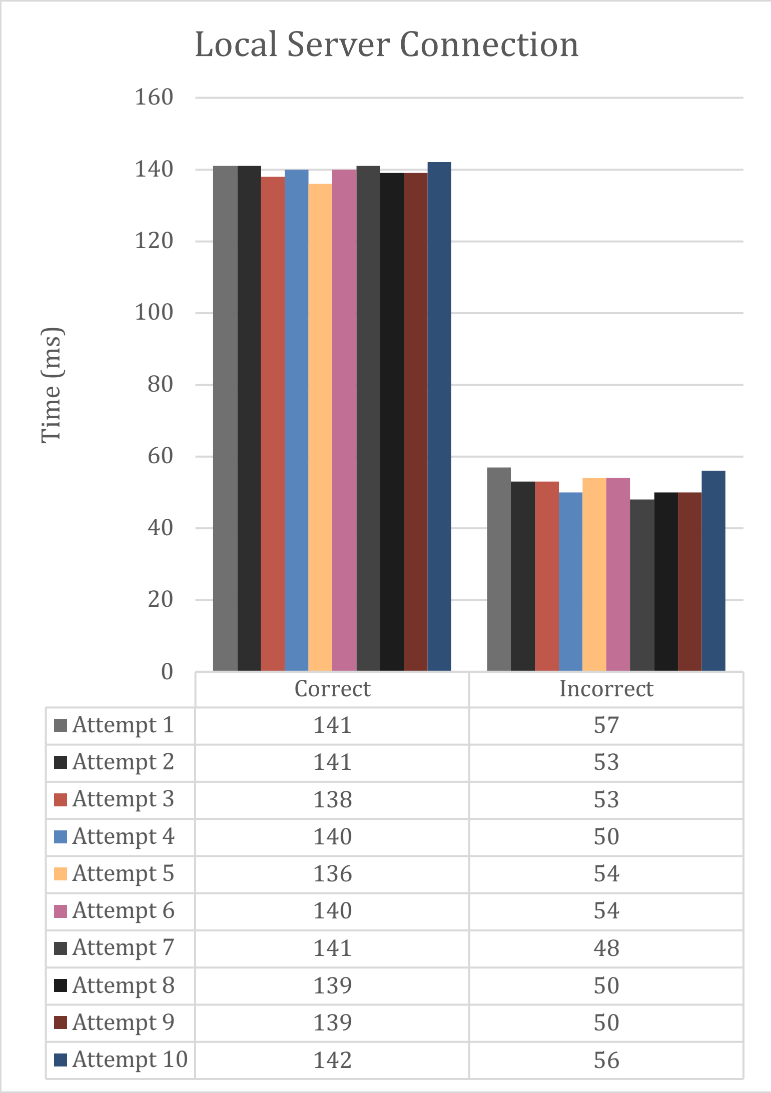
	Local Correct	Local Incorrect
■ 1 MB	186	50
■ 2 MB	194	71
■ 3 MB	230	66
■ 4 MB	236	76
■ 5 MB	291	53
■ 6 MB	243	87
■ 7 MB	270	67
■ 8 MB	283	59
■ 9 MB	291	57
■ 10 MB	318	65



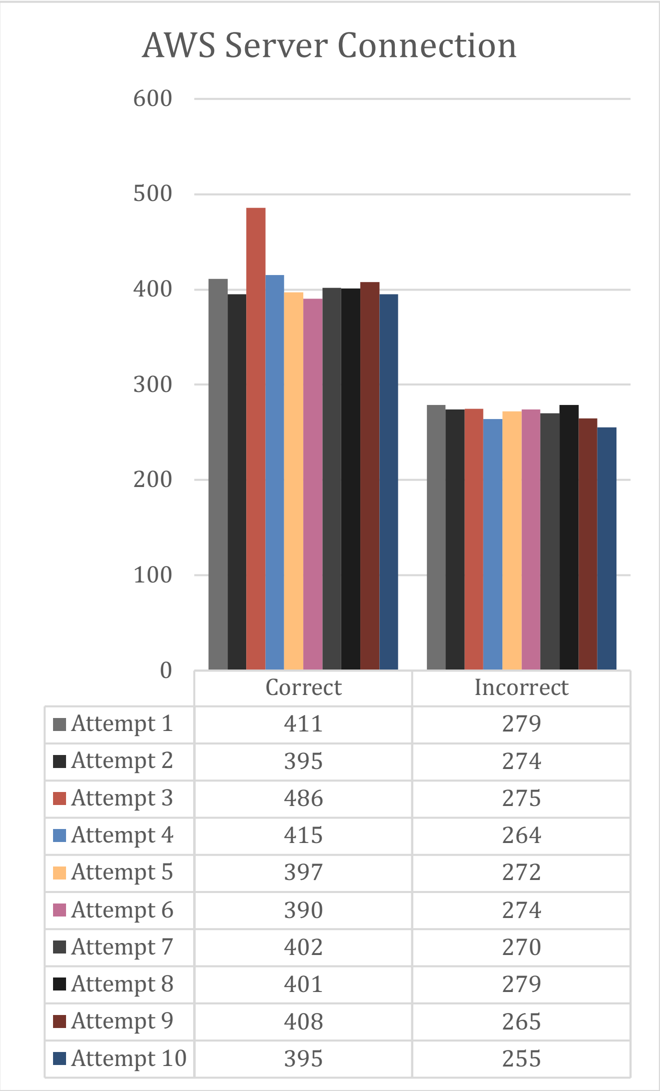
	Local Correct	Local Incorrect
■ 1 MB	985	275
■ 2 MB	1146	575
■ 3 MB	1311	294
■ 4 MB	2850	255
■ 5 MB	2628	293
■ 6 MB	2185	476
■ 7 MB	7245	396
■ 8 MB	7919	400
■ 9 MB	5306	319
■ 10 MB	10345	758

Time (ms)	Local Correct	Local Incorrect	AWS Correct	AWS Incorrect
Total	2542	651	41920	4041
Average	254.2	65.1	4192	404.1
STD	43.685	11.130	3307.399	159.968

ii. Local vs. Remote Costs (Connect to Server



	Correct	Incorrect
■ Attempt 1	141	57
■ Attempt 2	141	53
🟥 Attempt 3	138	53
🟦 Attempt 4	140	50
🟧 Attempt 5	136	54
🟪 Attempt 6	140	54
■ Attempt 7	141	48
■ Attempt 8	139	50
🟫 Attempt 9	139	50
🟦 Attempt 10	142	56



	Correct	Incorrect
■ Attempt 1	411	279
■ Attempt 2	395	274
🟥 Attempt 3	486	275
🟦 Attempt 4	415	264
🟧 Attempt 5	397	272
🟪 Attempt 6	390	274
■ Attempt 7	402	270
■ Attempt 8	401	279
🟫 Attempt 9	408	265
🟦 Attempt 10	395	255

Time (ms)	Local Correct	Local Incorrect	AWS Correct	AWS Incorrect
Total	**1397.7**	**525**	**4100**	**2707**
Average	**139.8**	**52.5**	**410**	**270.7**
STD	**1.767**	**2.915**	**27.829**	**7.484**

B. Statistical Analysis (Mean and STD)

i. “head -c $((i*1000000)) /dev/urandom > test_documents/test$i.txt” command

I have calculated that it takes on average 254 milliseconds to make files on the local computer server and 4192 milliseconds to make files on the AWS server.

Time (ms)	Local Correct	Local Incorrect	AWS Correct	AWS Incorrect
Total	2542	651	41920	4041
Average	254.2	65.1	4192	404.1
STD	43.685	11.130	3307.399	159.968

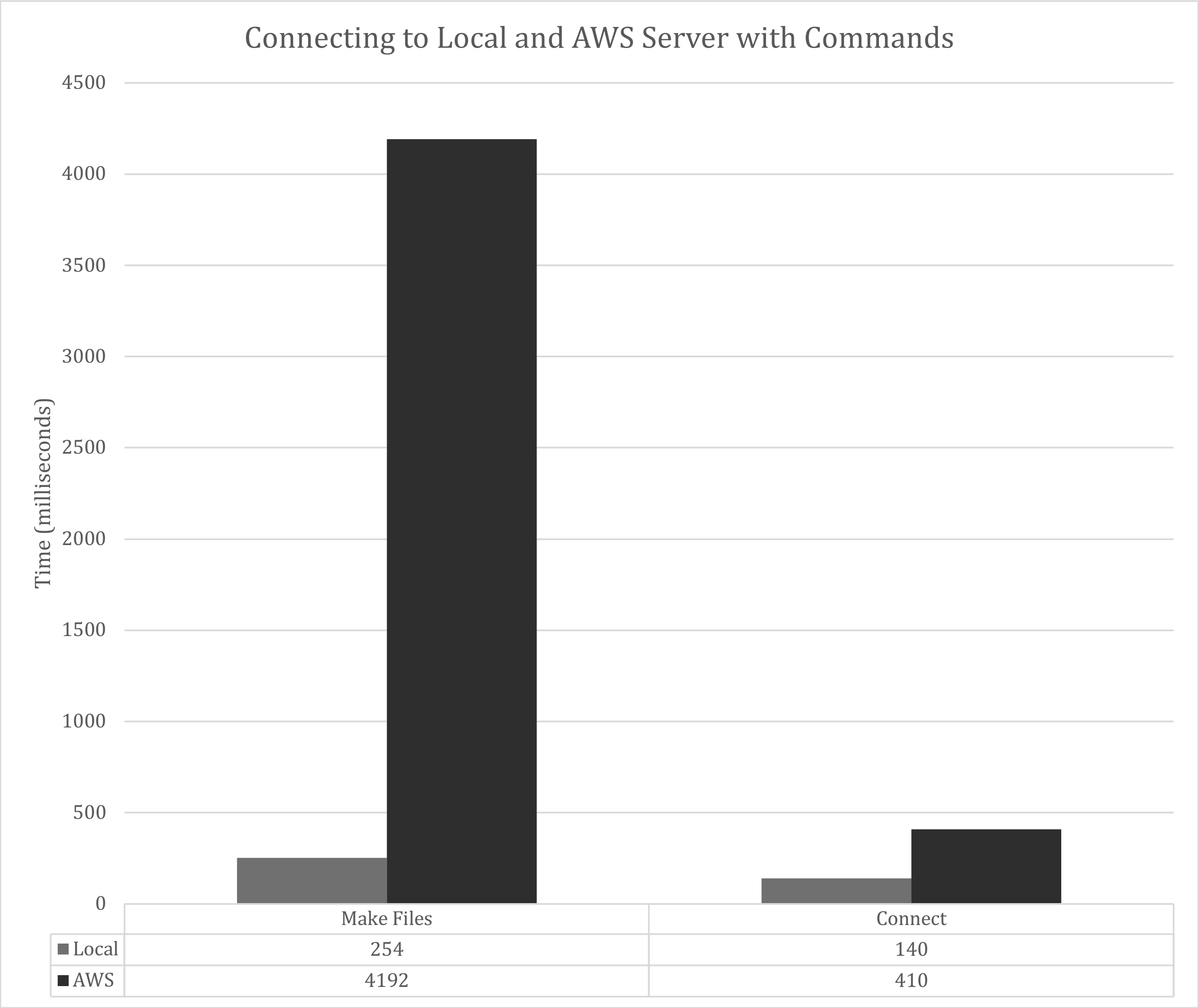
ii. “ls” command

I have calculated that it takes on average 140 milliseconds to connect to the local computer server and 410 milliseconds to connect to the AWS server.

Time (ms)	Local Correct	Local Incorrect	AWS Correct	AWS Incorrect
Total	1397.7	525	4100	2707
Average	139.8	52.5	410	270.7
STD	1.767	2.915	27.829	7.484

4. ANALYSIS

A. Interpret Results



B. Results Meaning

From the results, I got the average time it takes to connect and make files on the local and remote server (AWS).

C. Subjective Analysis

I can conclude that it takes longer to connect to the AWS server than the Local server. The "Making files" command takes longer than the "ls" command: about twice as much for local and about 10 times longer than connect on the AWS.