CSE5232 - Network Programming

Milestone 2
Report

Xiang Li

Objective

Program an iterative server that stores locally in a sqlite database a set of peers and gossip text messages that were already seen by this machine. The server recognizes commands both over TCP and UDP. There are 3 types of accepted messages:

To store a gossip message:
GOSSIP:mBHL7IKilvdcOFKR03ASvBNX//ypQkTRUvilYmB1/OY=:2017-01-09-16-18-20-001Z:Tom eats Jerry%
The server will check if it already new it and simply discard it if it was known.

To store a peer message:
PEER:John:PORT=2356:IP=163.118.239.68%
The server will store the name and address of this peer if it is not yet known, or to update its address.

To query for the list of peers:
PEERS?
The expected answer has the format:
PEERS|2|John:PORT=2356:IP=163.118.239.68|Mary:PORT=2355:IP=163.118.237.60|%"

Technology

Java: For TCP/UDP server and client interaction
Sqlite: Database
Make File: For compilation of project
Bash Script: For running the TCP/UDP server and client

Architecture

1. Project Directory Structure

src - contains .java files
lib - contains the library which are needed for project
bin - contains compiled java files. It is empty at first.
Script files - for executing TCP/UDP Server and Client Java files
makefile - file used to compile the project
xiangli_m2.db - Database file

2. Java files Details

Main.java - starts server

TcpClient.java - starts TCP client
TcpServer.java - listens to client and starts TCP server thread
TcpServerThread.java - communicates with the client

UdpClient.java - starts UDP client
UdpServer.java - starts UDP server thread
UdcpServerThread.java - communicates with the client

Database.java - all database related methods
TableValue.java - defines form for tables
GetOpt.java - processes GetOpt input

3. Database Structure

Database Name - xiangli_m2.db
Table Name - gossip
Columns - id Primary Key
encoded
datetime
message

Table Name – peers
Columns - id Primary Key
peername
port
ip

The database has already been stored a gossip message and two peer messages. If you want to test a new database, you should delete **xiangli_m2.db** first.

User Manual

1. Unzip the “xiangli_m2.tar.bz2”.

2. Compile:

```
cd <Project Path>
make
```

Or

```
chmod +x compile.sh
./compile.sh
```

```
Last login: Fri Mar 17 23:48:41 on ttys000
➜  /Users/lixiang > Downloads/xiangli_m2
➜  /Users/lixiang/Downloads/xiangli_m2 > make
javac -classpath src -g -d bin src/Main.java
javac -classpath src -g -d bin src/TcpClient.java
javac -classpath src -g -d bin src/UdpClient.java
➜  /Users/lixiang/Downloads/xiangli_m2 >
```

3. Execute:

Start a server:

```
chmod +x runServer.sh
./runServer.sh < Protocol Type> <Port Number>
```

Example for start a TCP server with port 2356:

```
./runServer.sh t 2356
```

```
➜  /Users/lixiang/Downloads/xiangli_m2 > chmod +x runServer.sh

➜  /Users/lixiang/Downloads/xiangli_m2 > ./runServer.sh t 2356
*** Started TCP Server, waiting for Client ***
```

Then, open a new window of terminal. Start a TCP client:

```
cd <Project Path>
chmod +x runTcpClient.sh
./runTcpClient.sh <Port Number>
```

We should use the same port:

```
./runTcpClient.sh 2356
```

```
Last login: Sat Mar 18 03:49:44 on ttys001
➜  /Users/lixiang > Downloads/xiangli_m2
➜  /Users/lixiang/Downloads/xiangli_m2 > chmod +x runTcpClient.sh

➜  /Users/lixiang/Downloads/xiangli_m2 > ./runTcpClient.sh 2356

This is TCP Client
Connection succeeded! Enter 'peers?' for query, 'q' for quit
```

Enter in client window:
GOSSIP:mBHL7IKilvdcOFKR03ASvBNX//ypQkTRUvilYmB1/OY=:2017-01-09-16-18-20-001Z:Tom eats Jerry%

```
Last login: Sat Mar 18 03:49:44 on ttys001
➜  /Users/lixiang > Downloads/xiangli_m2
➜  /Users/lixiang/Downloads/xiangli_m2 > chmod +x runTcpClient.sh

➜  /Users/lixiang/Downloads/xiangli_m2 > ./runTcpClient.sh 2356

This is TCP Client
Connection succeeded! Enter 'peers?' for query, 'q' for quit
GOSSIP:mBHL7IKilvdcOFKR03ASvBNX//ypQkTRUvilYmB1/OY=:2017-01-09-16-18-20-001Z:Tom
 eats Jerry%
Store now
```

The server will receive this message and store it to database.

```
➜  /Users/lixiang/Downloads/xiangli_m2 > ./runServer.sh t 2356
*** Started TCP Server, waiting for Client ***
Started a new TCP server thread!
Creating new database
Opened database successfully
Received: GOSSIP:mBHL7IKilvdcOFKR03ASvBNX//ypQkTRUvilYmB1/OY=:2017-01-09-16-18-2
0-001Z:Tom eats Jerry%
```

Enter in client window:
PEER:John:PORT=2356:IP=163.118.239.68%
and
PEER:Mary:PORT=2355:IP=163.118.237.60%

```
PEER:John:PORT=2356:IP=163.118.239.68%
Store now
PEER:Mary:PORT=2355:IP=163.118.237.60%
Store now
```

Enter in client window:
PEERS?
We will get the result for querying for stored peers in database

```
PEERS?
PEERS|2|John|PORT=2356|IP=163.118.239.68|Mary|PORT=2355|IP=163.118.237.60|%
```

Enter **q** to quit for client:

```
q
Disconnected
➜  /Users/lixiang/Downloads/xiangli_m2 >
```

Example for start a UDP server with port 2356:

```
./runServer.sh u 2356
```

```
➜  /Users/lixiang/Downloads/xiangli_m2 > ./runServer.sh u 2356
*** Started UDP Server, waiting for Client ***
Started a new UDP server thread!
This database already exists
Opened database successfully
```

Then, open a new window of terminal. Start a UCP client:

```
cd <Project Path>
chmod +x runUdpClient.sh
./runUdpClient.sh <Port Number>
```

We should use the same port:

```
./runUdpClient.sh 2356
```

```
➜  /Users/lixiang/Downloads/xiangli_m2 > chmod +x runUdpClient.sh

➜  /Users/lixiang/Downloads/xiangli_m2 > ./runUdpClient.sh 2356
This is UDP Client. Enter 'peers?' for query, 'q' for quit
```

The following steps is same with the TCP part.

In my codes, I used SQL command `INSERT OR IGNORE INTO` to store message for gossip, and set the **message** column (i.e. **Tom eats Jerry)** as Unique, so the database will store message if it is new, or discard it by default.

I used SQL command `INSERT OR REPLACE INTO` to store message for peers, and set the **peername** column (i.e. **John)** as Unique, so the database will update message if its information of other columns is new, or discard it by default.

4. Remove class files and the database file:

```
cd <Project Path>
make clean
```

In the execution phase, if there is an exception:
`java.net.BindException: Address already in use (Bind failed)`

You can use this command to list the process (pid) running on port.
`lsof -i:<port>`
Then kill the process with:
`kill <pid>`

I tried to run my program in my account on code01.fit.edu, but it always display the following information, so I run my program locally.

```
xiang2015@code01 xiangli_m2 $ ./runTcpClient.sh 2356
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Cannot create GC thread. Out of system resources.
# An error report file with more information is saved as:
# /udrive/student/xiang2015/xiangli_m2/hs_err_pid59476.log
xiang2015@code01 xiangli_m2 $ _
```

Conclusion

Through this project, I learned how to implement communication between client and server bash on TCP and UDP, and interact with the Sqlite database. This project successfully responds to multiple clients on TCP and UDP, storing the data for messages which meet the required format in a local database file.

References

Projects2015/P2/anita_devi-rini_simon. (n.d.). Retrieved from

http://cs.fit.edu/~msilaghi/17/2017sprNPG/Projects2015/P2/anita_devi-rini_simon/

Projects2015/P2/biro_maxim/maxim_biro_andrew_binns. (n.d.). Retrieved from

http://cs.fit.edu/~msilaghi/17/2017sprNPG/Projects2015/P2/biro_maxim/maxim_biro_andrew_binns/