Peer to Peer Implementation of the Classic Five in a Row Game

Principles of Computer Networking
11/28/16

Abdulrahman Abdulbari
absaabdu@iupui.edu
0002864562

Eric Pamperin
epamperi@iupui.edu
0002710390

Ethan Hayn
ehayn@iupui.edu
0002438037

I. Design and Implementation

Our project developed for our course “Principles of Computer Networking” was an implementation of the classic five in a row game, utilizing the traditional ruleset, over a peer-to-peer network. Our development was broken into two separate tasks. Firstly, we focused on designing and creating the base five in a row game, or the application layer. Secondly, we applied a network layer to work in conjunction with the game to support two remote players or clients to play each other over a network. Both the application and network layers were developed and written in Java.

The application layer consists of two main parts. The first is the main game loop that changes turns and allows the players to interact with the game board. The second is a function that is called during the loop after each turn to check to see if the game has been beaten. The game loop starts by checking to see which player’s turn it is to move. After a notification, the indicated player will be prompted to choose a row and column to place his or her piece. After ensuring the slot is empty, the application will mark the two dimensional array game board. These players will keep taking turns until the game has found a winner by utilizing a function at the end of each turn loop. The function that finds a winner searches separately for horizontal, vertical, and diagonal victories. For each condition it will iterate through the gameboard and and count. After iterating through five of the same color, the function returns true indicating a player has won. Otherwise, the count to five will fail and the game will continue.

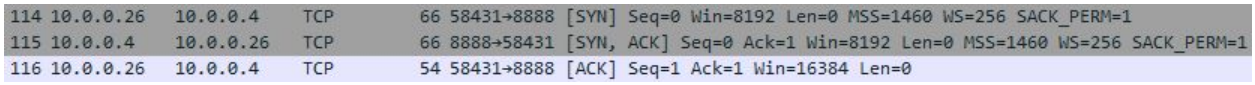
The network layer utilizes the Java’s sockets packages. When the a client decides to host a game, the program starts a socket listener thread. After it is started, the thread will wait for a second player to join. At this point the second client will decline to host on their separate program and the hard coded IP address and port will be used to connect to the the first clients listening thread. After the connections are made, the two clients will be able to interact with the game board during their respective turns. During the game message strings will be sent over the network. These strings will determine how the game is progressed and what each player is able to do during different time intervals.

II. Lessons Learned

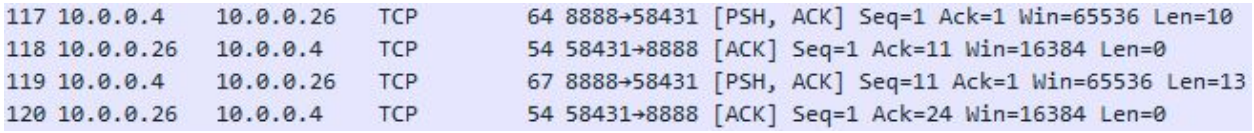
The aspects of utilizing a network to connect clients was a fairly new idea to most people in our group. Utilizing a network without the use of a dedicated server was entirely new to everybody. It took time and research to understand and choose the right approach that best fit the goal in mind. Our criteria simply being the ability to connect two remote players with nothing more than the client machines. After much deliberation, we found the best approach was to use the host designation approach. In doing so, we were able to pass our first hurdle of connecting two clients to each other. Afterwards, we

learned the basics of utilizing messages passed between clients to advance the game. In modern gaming the amount of data passed between clients can be much greater than simple strings and single integers that are used in our implementation. Questions of efficiency and security are clear to be future roads of learning and exploration if any of us utilize these techniques outside of this course.

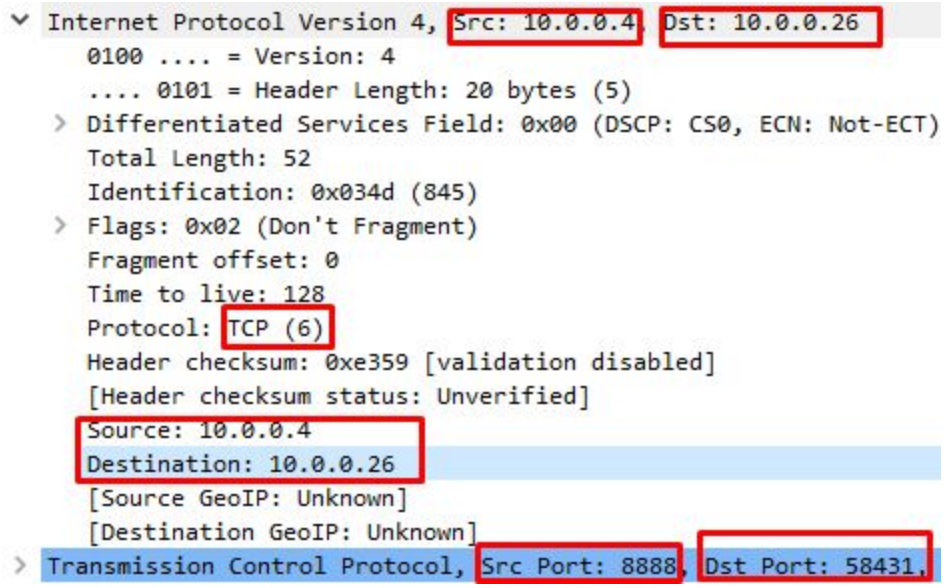
III. Wireshark Analysis



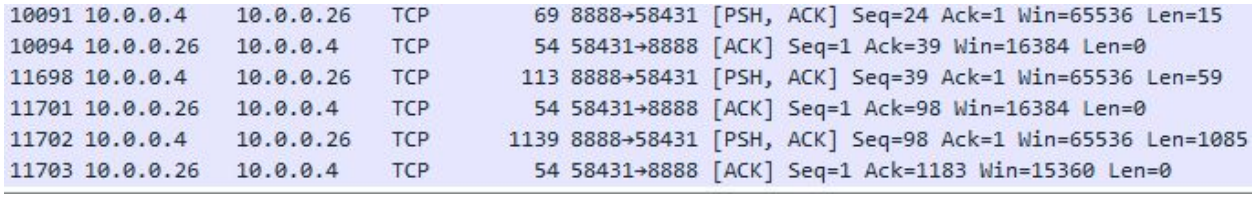
The program executes. In this situation, Player 1 has an IP address of 10.0.0.4 and Player 2 has an IP address of 10.0.0.26. Player 1 enters 1 to host the game and Player 2 enters 0 to join Player 1's game. Once Player 2 enters 0, a packet is send to initiate a connection with Player 1; SYN(synchronize). Player 1's game will send an acknowledgement back to Player 2's game to confirm that he is allowed to join the game. The TCP window size is 8192.



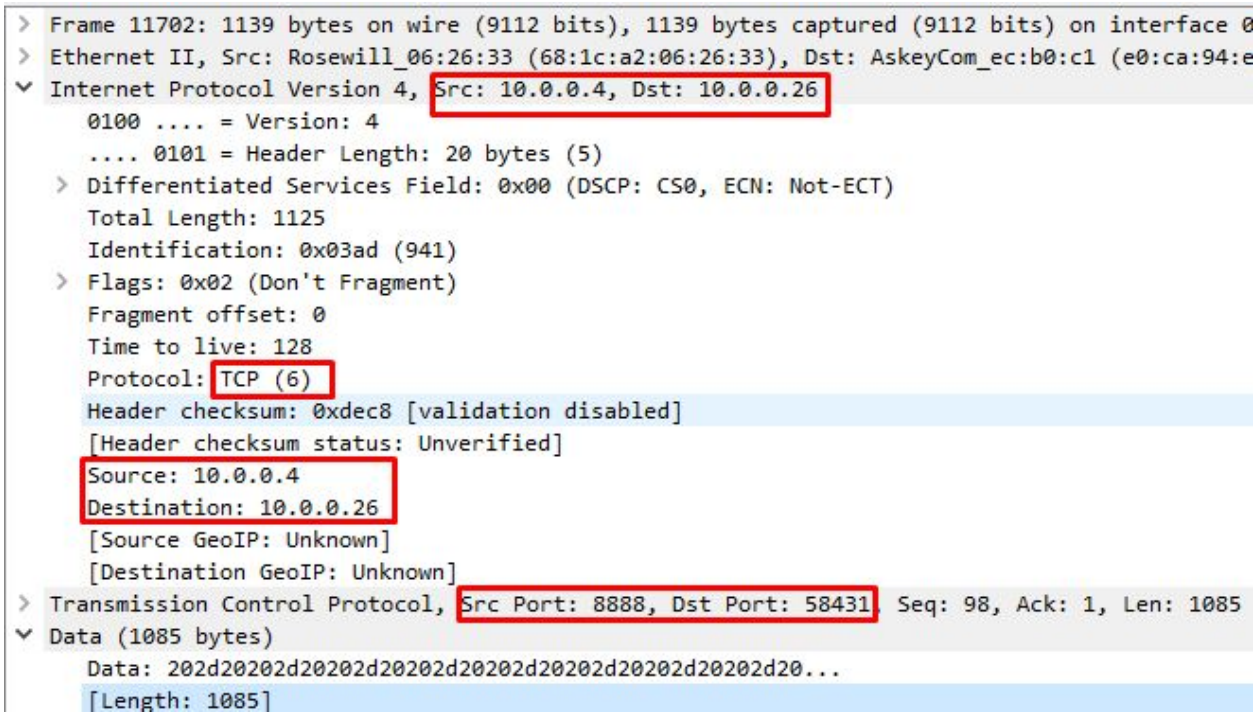
This deals with sending the messages to Player 2 on what Player 1 is currently doing. Packet #117 sends "Player 1" to Player 2 to indicate that it is currently Player 1's turn and the message is eventually acknowledged. After that, packet #119 sends "What Row?" and "What Column?" which is acknowledged by Player 2.



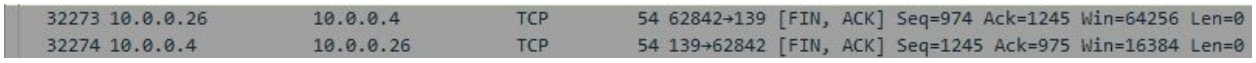
This is the packet that contains a simple “Player *” string . The data originates from the HOST (10.0.0.4) and goes to Player 2 (10.0.0.26). Using the TCP protocol with a source port of 8888 and destination port of 58431. The TCP window size is 256.



In this screenshot, packet # 11702 is carrying 1139 bytes of data. This data mainly consists of the data needed to map out the game board with the most recent moves by Player 1 and 2. You can find the mapped out data in the game screenshots below.



This is the packet that contains the player position data. The data originates from Player 1 (10.0.0.4) and goes to Player 2 (10.0.0.26). Using the TCP protocol with a source port of 8888 and destination port of 58431. The data being transmitted is a total of 1085 bytes.



As seen above, the game at this point has ended because Player 1 has managed to get 5 in a row. FIN indicates that the program wants to close the connection. Therefore, the program is instructed to close the TCP connection because there is no more data to send.

```
Host Game?
1
Server: Player 1
Server: What row?
0
Client: 0
Server: What column?
0
Client: 0
Server: Row: 0 Column: 0
Server:  B - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server: Player 2
Server: What row?
```

This is how the game screen looks during Player 1’s turn. The player is asked for the row and column which is inserted into a 2D array. The array is printed out into several lines.

```
Server:  B B B B B - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - W - - - - - - - -
Server:  - - - - - - - - - - - W - - - - - - -
Server:  - - - - - - - - - - - - W - - - - - -
Server:  - - - - - - - - - - - - - - W - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server:  - - - - - - - - - - - - - - - - - - -
Server: The Winner is Player 1(Black)!- Horizontal
```

This is how the game looks once a player wins the game. In this case, Player 1 has won the game for getting 5 black’s in a row horizontally.

IV. Team Contributions

Abdulrahman Abdulbari - 2nd iteration of application layer, wireshark analysis

Eric Pamperin- Project Manager, 1st iteration of application layer, created presentation

Ethan Hayn - Network layer implementation, technical writer

References

Hashimi, Sayed Ibrahim. “Peer-to-Peer Applications Made Easy.” *JavaWorld*. N.p., 9 May 2005. Web. 27 Nov. 2016.

“Introduction to the Peer-to-Peer Sockets Project - O’Reilly Media.” N.p., n.d. Web. 27 Nov. 2016.

“Lesson: All About Sockets (The Java™ Tutorials > Custom Networking).” N.p., n.d. Web. 27 Nov. 2016.