PROTOCOL DESIGN PAPER

KUNAL VOHRA
CS-544 Computer Networks

Application Layer Game Protocol Design

Service description : -

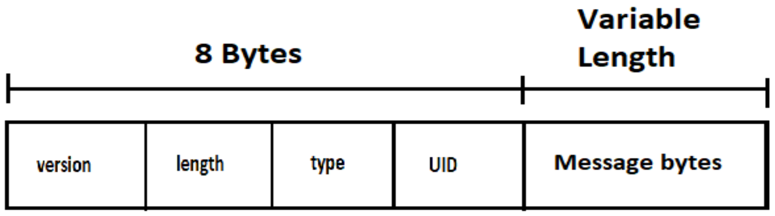
This paper presents an application layer protocol which will take service from TCP in the transport layer. The protocol will be based on Client – Server type of architecture . The main purpose of the server would provide game playing (single player as well as multi player) to a client logged in to the server. Since TCP is being used at the transport layer, the server should be able to host games which require a less frequent frame updates like tic tac toe.

Any user port should work fine on the server side. The same port number will be used on the client side alternating with the server just for implementation of the protocol on the same computer. Transfer of commands will be bidirectional and full duplex.

In a game playing type Application layer protocol, since the game is not hosted locally, there are a number of updates exchanged between a player and a server. Consider the example of a tic tac toe game board where 2 players connect to a server to play. The server will first load the game board for both players. When player 1 makes a move, the application layer protocol on client's will send the information of their latest move to the server. This information is translated on the server side and then an update in the game state is issued at the game server. Thus, every move means an exchange and updating of states on both client and player side.

Message definition : -

Messages will be transmitted in bytes . A fixed length header specifying message length, a message type, version number and a user ID (one provided by server to the client) will be a part of the message packet along with the binary data. The PDU of the message is as below : -



Messages will be of two kinds

1) Messages between the client and the server

These are the transaction messages which occur between the Client and server from the time of client's connection request till the time of disconnection.

2) Update messages transferred between the client and the server
These are the update data messages which the player and server exchange while the game is being played.

The following is an example of how messages will be exchanged by the system

(client) Connection request (xx.xx.xx.xx.1234)

(server) Enter User : (client) bob | (server) Server Busy

(server) Password : (client) birdman | (server) Invalid User. Retry

(server) Waiting for response... | (server) Invalid Password. Retry

(server) Success... | (server) Failed to connect to server

(client) launch game

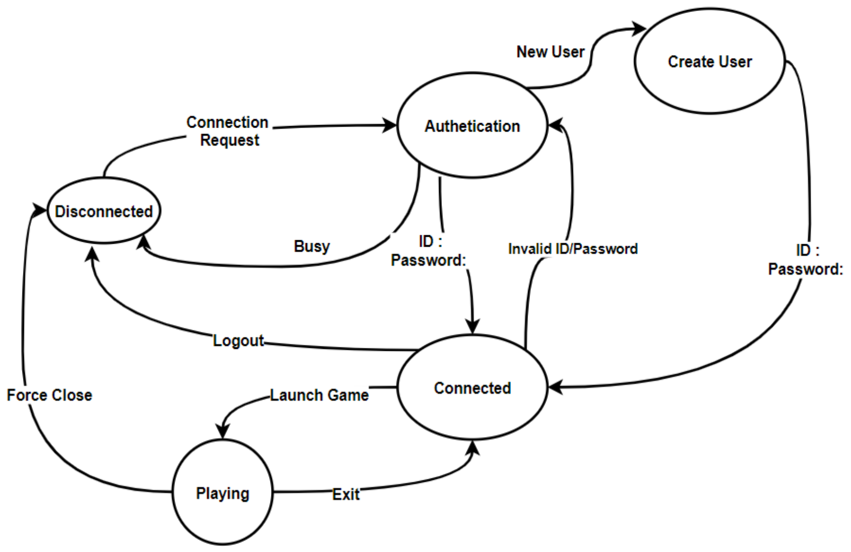
(client) logout

(server) Disconnected

apart from the above-mentioned messages, the server and client will exchange game states in the form as messages as well.

Use of TCP ensures that the packets are delivered in order and serialization of packets.

DFA : -



Encryption : -

Encryption is important to ensure that the account credentials are not sniffed by thieves. The server must thus implement a key based on client data which will be shared between both the client and the server. But this may not prevent direct sniffers or someone who intercepts a stream directly and mirror the clients key. Instead an asymmetric system can be used here . The server has a private key and a public key. Which is well known to all its clients. The client randomly generates a per-session symmetric encryption key, encrypts with the server's public key, and sends that over which ensures anonymity over the server.

An RSA key can also be implemented in the system. However, this implementation is a much basic implementation and requires very minimal security for now. So, Encryption will be considered as one of the future scopes for this project.

Security implication:-

Security is an important consideration for any system over the internet. However, it is one of the most important requirements of a game protocol as cheating or hacking may spoil the end user experience. A server must be in command in all times in our system. If the server starts listening to a client on a gaming protocol, it will end up being very vulnerable to cheaters. Since this is the first version of the protocol, a major focus is on definition of connections, states and messages. For now our system remains heavily vulnerable to being hacked or cheaters, but there is only so much a player can cheat in tic tac toe (for example).

Extensibility : -

The application protocol provides extensibility since it accommodates various fields which support extensibility in the protocol structure. New kinds of messages can be defined in the system which shall work well due to the PDU also having variable length field for a message. The version field also provides the protocol to have updated versions to the same protocol.

As per current design I fell the following features might be a good addition to the protocol in the future :

Clients being able to play as guests on the server.

User and Server ports be defined as per use of the respective computers.

A custom designed transport layer protocol based over UDP which can help lower the bandwidth requirement of the protocol.

A loading state which can show if there is some sort of delay due to packet retransmission.

A further analysis of this protocol may show results with bad performance in case of lossy networks which can be implemented either over UDP in the transport layer or a combination of UDP and TCP. by defining the transport layer with limited retransmission ability. In fact, this can achieve the desired (efficient network) effect in multiple ways for example by defining a reliable packet type as well as an unreliable packet type or by simply providing a modified ACK system with a NACK for missing packets. These are in theory, powerful mechanisms and they will be tested as a future scope of this project.