Homework 4- RPC Implementation of Password Cracker

Johnu George **Sood Nitesh**

RPC Definition File

```
program LSP_PROGRAM {
    version LSP_VERS {
        int sendfn(LSPMessage1) = 1;//RPC function for sending packets from client to server
        void callbackfn(int)  = 3; //RPC function called by client used for registering callback function using the transient number
    } = 1;
```

RPC Model

In our model, client communicates to the server via normal rpc calls while server communicates to various clients using call back functions registered during the time of creation of clients.

Design Enhancements and RPC modifications

The changes which are added are described below. Rests of the functionalities remain same. Protobuf and socket code is removed completely.

Client Code

1. When lsp_client_create is called,

   a) Function Clnt_create is called , instead of creating sockets.

   b) New thread, RPC Thread is created .It gets an available transient number and registers a callback function using svc_register . This transient number is used as program identifier for RPC callback from the server.

   c) Connection Request is sent from client once callback function is registered.

2. RPC Thread

   a) RPC thread registers call back function for receiving packets from server.It calls svc_run and goes into infinite loop ,waiting for the packets.

b) When callback function of the client is called by the server, packets are queued in client's rpcInbox queue.

3. Write Thread/Epoch Thread

a) Packets are de-queued from outbox queue and packets are sent out using RPC function 'sendfn'.

b) network_send_message is modified to call RPC function 'sendfn' for sending the LSP packet to the server

4. Read Thread

a) network read message is modified to read the packets and de-queue from the Client's rpcInbox queue.

Server Code

1. RPC Thread
   a) svc_udpcreate is called and rpc functions are registered using svc_register

   b) RPC thread calls svc_run and goes into infinite loop, waiting for the packets.

   c) When receive function of the server is called by the client, packets are queued in server's rpcInbox queue.

2. Write Thread/Epoch Thread

a) Packets are de-queued from outbox queue and packets are sent out to respective clients using callback functions registered by them.

b) network_send_message is modified to call the callback function for sending the LSP packet to the client.

3. Read Thread

a) network_read_message is modified to read the packets and de-queue from the server's rpcInbox queue.

Additional Datastructures

Client

Queue<LSPMessage*> rpcInbox; // To store the received packets from the server

CLIENT *clnt_handle; // client handle obtained by clnt_create

Server

Queue<rpcInbox_struct> rpcInbox; // to store received packets from the client which contains the message, address and port of the client

std::map<rpcInbox_struct,int,conn_comp> client_map; // mapping from address and port of client ,to program identifier

Team Members and Contribution

Johnu George - 50 %

Sood Nitesh - 50%