UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HA NOI

REMOTE SHELL USING RPC

DISTRIBUTED SYSTEMS

Group 3

Nguyen Hoang Nam	BI10-123
Nguyen Quoc Hung	BI10-072
Nguyen Minh Duc	BI10-031
Nguyen Huy Hung	BI10-071
Hoang Huu Huy	BI10-077

Table of Contents

I. Introduction

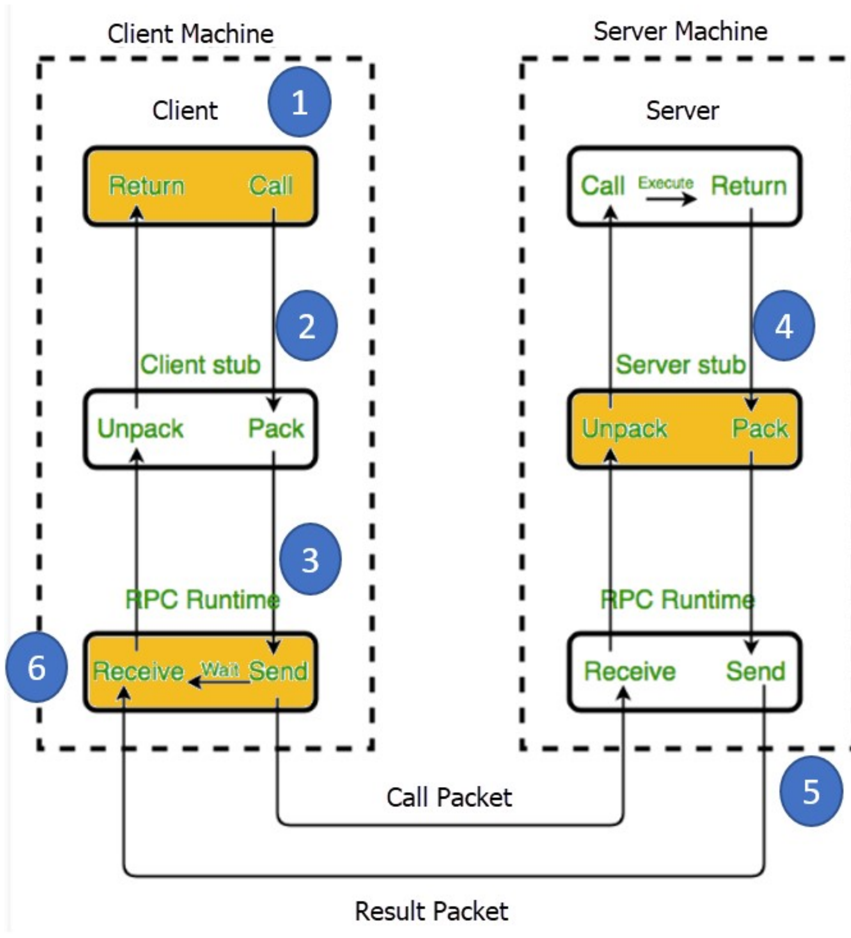
a. Remote Procedure Cell's definition

- Remote Procedure Cell (RPC) is a software communication protocol that allows one program to seek a service from another program on a network without needing to grasp the network's details. RPC is used to invoke other processes on distant systems, just like a local system. A procedure call is also known as a function call or a subroutine call.

b. How RPC works

- The calling environment is stopped when a farther method call is made, the method parameters are sent over the organization to the environment where the method will be conducted, and the method is at that point executed in that environment.
- When the procedure is completed, the comes about are returned to the caller environment, where execution restarts as in case it were a typical method call.
- During an RPC, there are 8 steps :
    1. The client is alluded to as the client stub. The call may be a standard nearby method call, with parameters put into the stack.
    2. The client stub typifies the method contentions in a message and sends it via a framework called. Marshaling is the method of bundling method para.
    3. The message is sent from the client framework to the far-off server machine utilizing the client's neighborhood Oximeters.
    4. Incoming bundles are passed to the server stubby the server OS.

5. The server stub unloads the parameters from the message, a handle known as unmarshalling.
6. When the server handle is total, the server stub marshals the return values into a message…
7. When the server handle is total, the server stub marshals the return values into a message. The message is in this way sent to the transport layer by means of the server stub.
8. The return contentions are unmarshalled by the client stub, and execution returns to the caller.



c. Types of RPC

- There are many RPC models and distributed computing implementations. The Open Software Foundation's Distributed Computing Environment is a common paradigm and implementation.
- RPC configurations are including :
  - The standard mode of operation in which the client initiates a call and does not proceed until the server responds.
  - The client dials a number and proceeds with its own processing. The server does not respond.
  - A feature that allows you to submit several nonblocking client calls in a single batch.
  - The client initiates a nonblocking client/server contact, and the server responds by invoking a client-specific process.
- In the Open Systems Interconnection model of network communication, RPC covers the transport and application layers. RPC simplifies the development of applications that incorporate many programs spread across a network.

d. Pros and cons of RPC

- There are some benefits for people who use RPC:
  - Clients can interface with servers by utilizing high-level dialects to create method calls.
  - It may be utilized in both a conveyed and a local setting. Process-oriented and thread-oriented models are too supported.
  - The client isn't mindful of the basic message-passing mechanism.
  - The code may be revised and redeveloped with the least effort.
  - Provides deliberation, i.e., the client isn't mindful of the message-passing viewpoint of arranged connection.

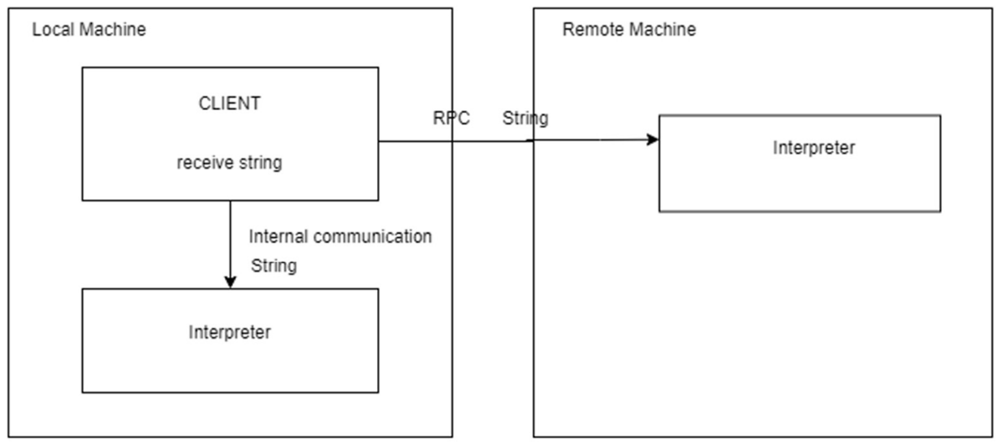
    - To boost execution, a few of the convention levels are skipped.
- However, the RPC also has some disadvantages:
    - The client and server utilize diverse execution settings, and the utilization of assets is moreover more complicated. As a result, RPC frameworks aren't continuously suitable for sending enormous volumes of information.
    - Because it incorporates a communication framework, another computer, and another prepare, RPC is amazingly inclined to failure.
    - RPC does not have an all-inclusive standard; it may be actualized in an assortment of ways.
    - RPC is interaction-based, and as a result, it does not give any equipment plan adaptability.

e. Remote shell's definition

- The remote shell (ssh) is command-line computer software that can run shell commands as another user and on another machine via a network.

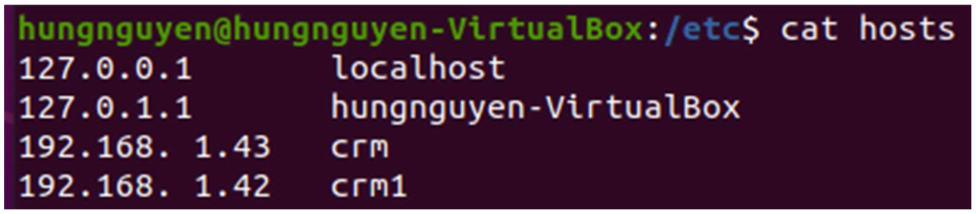
client
commands
Server

II. Analysis and Design



- Client and server connect using the RPC method. So how to create RPC? Our team uses a system remote procedure call called sunRPC. It will provide the most basic of a connection for clients and servers using RPC.
- On the Client side, it will run with 2 components that are identifying the user and determining the incoming command.
- On the server side, it will run and wait for the request from the Client. If it receives a request from the client, it determines the command and returns the data that matches the request from the client.
- About using remote shell, that will receive the shell from the client and transmit it to the server
- How can I create multiple users?

- Because using it on a computer, it is possible to create a different user with a different IP address or virtual user.



```
hungnguyen@hungnguyen-VirtualBox:/etc$ cat hosts
127.0.0.1       localhost
127.0.1.1       hungnguyen-VirtualBox
192.168. 1.43   crm
192.168. 1.42   crm1
```

III. <u>Implementation</u>

- In Client-side:

```
char *host;

if (argc < 3) {
        printf ("usage: %s server_host\n", argv[0]);
        exit (1);
}
host = argv[1];
comp_prog_1 (host, argv[2]);
```

- We have 3 arguments for connecting server, host, or user and command request.
- This will detect the user and command user requests.
- On the Server side:

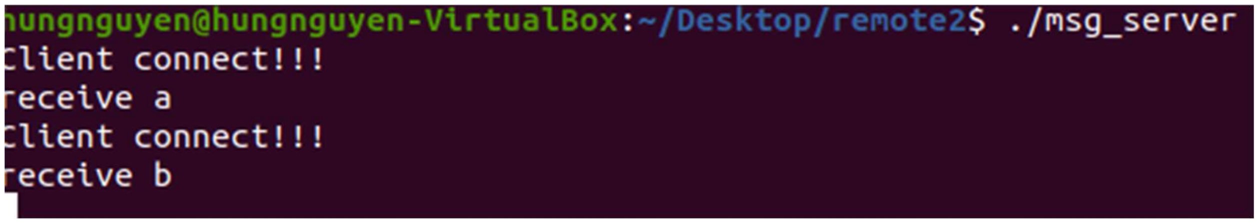
```
static int  result;

/*
 * insert server code here
 */
printf("Client connect!!!\n");
printf("receive %s\n", argp->method);
result = *argp->method;
return &result;
```

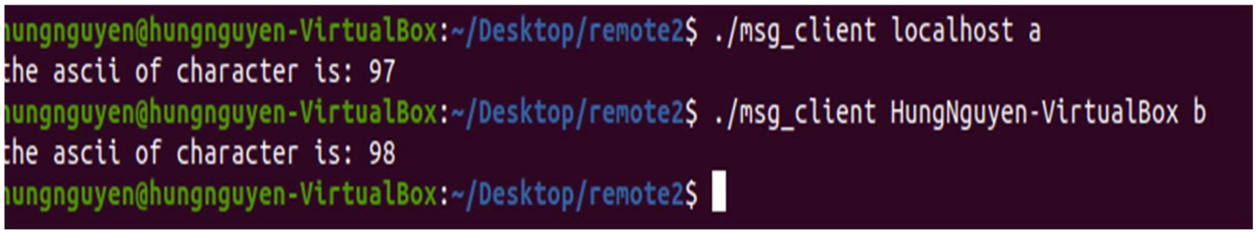
- We define the server stub for command from the client.
- About the system, We use the sun RPC port mapper protocol, which is TCP/UDP port-based.

IV. Results

- Here are some images for our results:



```
ungnguyen@hungnguyen-VirtualBox:~/Desktop/remote2$ ./msg_server
lient connect!!!
eceive a
lient connect!!!
eceive b
```



```
ungnguyen@hungnguyen-VirtualBox:~/Desktop/remote2$ ./msg_client localhost a
he ascii of character is: 97
ungnguyen@hungnguyen-VirtualBox:~/Desktop/remote2$ ./msg_client HungNguyen-VirtualBox b
he ascii of character is: 98
ungnguyen@hungnguyen-VirtualBox:~/Desktop/remote2$
```

V. Conclusion and Future work

- At the end of our project, we find out that:
    - RPC is a good way for connect between client and server.
    - For multiple users the way access is by check the host ip address.
- For the future work, we need find a way to apply remote shell into our project. From that, we can get commands from clients across a computer network so that our project can be for multiple users.

VI. References

https://www.techtarget.com/searchapparchitecture/definition/Remote-Procedure-Call-RPC

https://users.cs.cf.ac.uk/Dave.Marshall/C/node33.html

https://en.wikipedia.org/wiki/Remote_procedure_call

https://en.wikipedia.org/wiki/Remote_Shell