TUSHAR DESHMUKH
PE27
1032201698

ASSIGNMENT TITLE: Design a distributed application using RPC

AIM: To demonstrate the use of Remote Procedure Call (RPC) using client server architecture to calculate the square of given number.

OBJECTIVES: To study and implement RPC and client server architecture.

THEORY:
Remote Procedure Call:

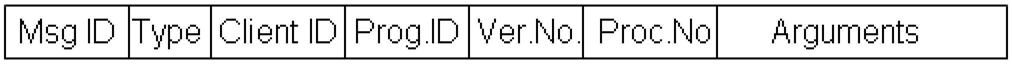
Basic characteristic of RPC is transparency. Transparency is of two types:
a) Syntactic Transparency: In this the RPC & LPC syntax is identical.
b) Semantic Transparency: In this RPC & LPC have same semantics.

Elements of RPC:
The basic elements of RPC code are:
a) Client
b) Client Stub
c) RPC Runtime
d) Server Stub
e) Server

In this type of communication:
a) Client calls a remote procedure by sending packets containing parameters.
b) It is packed and sent to the client.
c) The server receives the packet, unpacks it, computes the result and sends he result back to the client

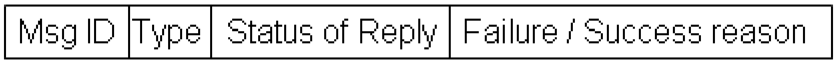
Stub Generation: Generation of strubs for client and server is of two types:
a) Manual: Provide a set of transmission functions from which user constructs his own stubs.
b) Automatic: It uses Interface Definition Language (IDL) i.e. List of procedures selected.

- RPC message format:



Msg ID	Type	Client ID	Prog.ID	Ver.No.	Proc.No	Arguments

- RPC Reply format:



Msg ID	Type	Status of Reply	Failure / Success reason

-Marshalling arguments and results:
a) Taking arguments of client and result.

b) Encoding a message.
c) Decoding the message.

INPUT: Number whose square is to be found.

OUTPUT: Square of number.

PLATFORM: Linux.

CONCLUSION: Thus, RPC has been studied and implemented on Linux platform.

FAQs :
1. What is the difference between RPC and LRPC?
Answer
Difference between RPC and LRPC:
RPC (Remote Procedure Call): A communication protocol that allows a program to execute procedures or functions on a remote server as if they were local. RPC typically involves marshalling parameters, sending requests over a network, executing the procedure on the server, and unmarshalling the results.
LRPC (Local Remote Procedure Call): A variant of RPC used for communication between processes on the same machine. LRPC avoids the overhead of network communication by directly invoking procedures in another process's address space.

2. What does rpcgen do?
Answer
Rpcgen:
Rpcgen is a tool used to generate code for RPC (Remote Procedure Call) implementations. It takes an interface description file as input, typically written in the RPC Language (XDR), and generates client and server stub code in C or other programming languages. rpcgen automates the process of marshalling and unmarshalling parameters, handling network communication, and defining data structures for RPC-based applications.

3. What is meant by packing and unpacking of RPC messages
Answer
Packing and unpacking of RPC Messages:
Packing: Refers to the process of converting data from its native representation in memory to a standardized format suitable for transmission over a network. This involves serializing data structures, such as converting integers to network byte order and encoding complex data types.
Unpacking: The reverse process of packing, which involves decoding received data from the network and reconstructing data structures in memory. Unpacking converts data from the standardized network format back to its native representation. Proper packing and unpacking are essential for ensuring interoperability and data integrity in RPC communication.

TUSHAR DESHMUKH
PE27
1032201698

CODE:

IDL.X

```
        struct values{
            float num1;
            float num2;
            char operation;
        };

        /*Programme, version and procedure definition*/

        program COMPUTE{
            version COMPUTE_VERS{
                float ADD(values) =1;
                float SUB(values)=2;
                        float MUL(values)=3;
                        float DIV(values)=4;
                float POW(values)=5;
            } =7;

} = 456123789;
```

IDL_client.c

```
#include "IDL.h"
#include <stdio.h>

float compute_7(char *host, float a, float b, char op)
{
    CLIENT *clnt;
    float *result_1;
    values add_7_arg;
    float *result_2;
    values sub_7_arg;
    float *result_3;
    values mul_7_arg;
    float *result_4;
    values div_7_arg;
    float *result_5;
    values pow_7_arg;

    if (op == '+')
    {

        add_7_arg.num1 = a;
        add_7_arg.num2 = b;
```

TUSHAR DESHMUKH
PE27
1032201698

```c
        add_7_arg.operation = op;

        clnt = clnt_create(host, COMPUTE, COMPUTE_VERS, "udp");
        if (clnt == NULL)
        {
            clnt_pcreateerror(host);
            exit(1);
        }

        result_1 = add_7(&add_7_arg, clnt);
        if (result_1 == (float *)NULL)
        {
            clnt_perror(clnt, "call failed");
        }

        clnt_destroy(clnt);

        return (*result_1);
    }

    else if (op == '-')
    {
        sub_7_arg.num1 = a;
        sub_7_arg.num2 = b;
        sub_7_arg.operation = op;

        clnt = clnt_create(host, COMPUTE, COMPUTE_VERS, "udp");
        if (clnt == NULL)
        {
            clnt_pcreateerror(host);
            exit(1);
        }

        result_2 = sub_7(&sub_7_arg, clnt);
        if (result_2 == (float *)NULL)
        {
            clnt_perror(clnt, "call failed");
        }

        clnt_destroy(clnt);

        return (*result_2);
    }

    else if (op == '*')
    {
        mul_7_arg.num1 = a;
        mul_7_arg.num2 = b;
```

TUSHAR DESHMUKH
PE27
1032201698

```
        mul_7_arg.operation = op;

        clnt = clnt_create(host, COMPUTE, COMPUTE_VERS, "udp");
        if (clnt == NULL)
        {
            clnt_pcreateerror(host);
            exit(1);
        }

        result_3 = mul_7(&mul_7_arg, clnt);
        if (result_3 == (float *)NULL)
        {
            clnt_perror(clnt, "call failed");
        }

        clnt_destroy(clnt);

        return (*result_3);
    }

    else if (op == '/')
    {
        if (b == 0)
        {
            printf("You are trying to divide by zero. Please insert a
valid number.\n");
            exit(0);
        }
        else
        {
            div_7_arg.num1 = a;
            div_7_arg.num2 = b;
            div_7_arg.operation = op;

            clnt = clnt_create(host, COMPUTE, COMPUTE_VERS, "udp");
            if (clnt == NULL)
            {
                clnt_pcreateerror(host);
                exit(1);
            }

            result_4 = div_7(&div_7_arg, clnt);
            if (result_4 == (float *)NULL)
            {
                clnt_perror(clnt, "call failed");
            }

            clnt_destroy(clnt);
```

TUSHAR DESHMUKH
PE27
1032201698

```
            return (*result_4);
        }
    }

    else if (op == '^')
    {
        pow_7_arg.num1 = a;
        pow_7_arg.num2 = b;
        pow_7_arg.operation = op;

        clnt = clnt_create(host, COMPUTE, COMPUTE_VERS, "udp");
        if (clnt == NULL)
        {
            clnt_pcreateerror(host);
            exit(1);
        }

        result_5 = pow_7(&pow_7_arg, clnt);
        if (result_5 == (float *)NULL)
        {
            clnt_perror(clnt, "call failed");
        }

        clnt_destroy(clnt);

        return (*result_5);
    }

    else
    {
        printf("Invalid operation. Please insert a valid operation.\n");
        exit(0);
    }
}

int main(int argc, char *argv[])
{
    char *host;

    float number1, number2;
    char oper;
    printf("Enter the 2 numbers followed by the operation to perform:\n");
    scanf("%f", &number1);
    scanf("%f", &number2);
    scanf("%s", &oper);

    host = argv[1];
```

TUSHAR DESHMUKH
PE27
1032201698

```
    printf("Answer= %f\n", compute_7(host, number1, number2, oper));
    exit(0);
}
```

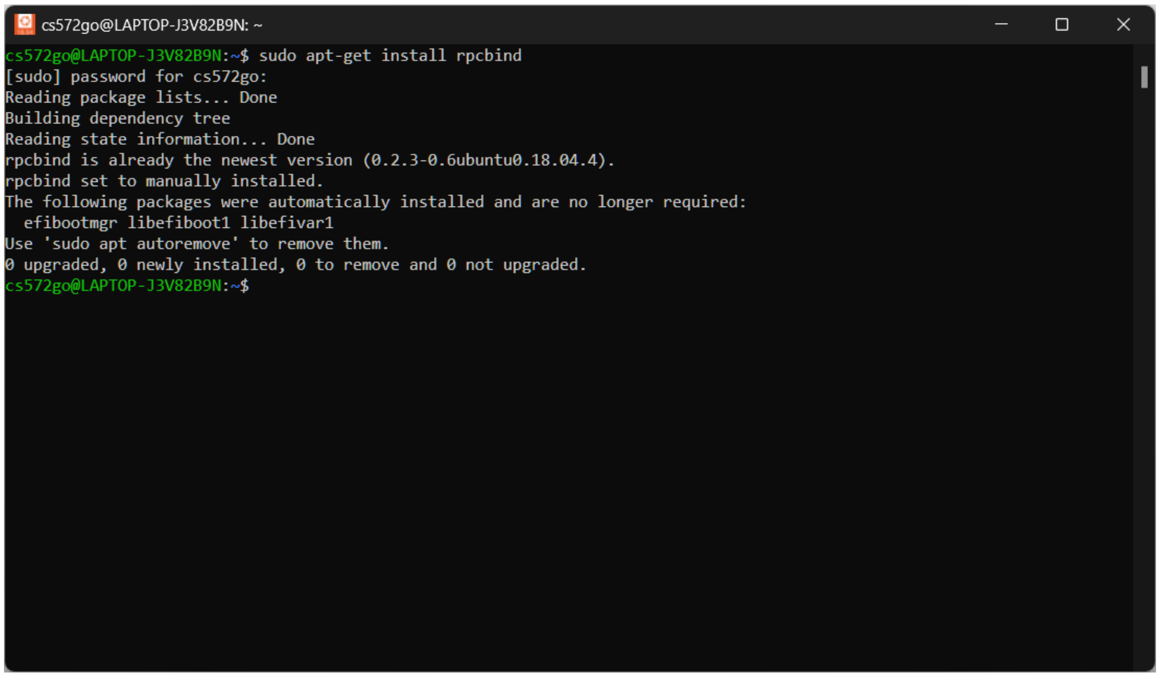
IDL_server.c

```
#include "IDL.h"
#include <stdio.h>

float *
add_7_svc(values *argp, struct svc_req *rqstp)
{
    static float  result;
    result = argp->num1 + argp->num2;
    return &result;
}

float *
sub_7_svc(values *argp, struct svc_req *rqstp)
{
    static float  result;
    result = argp->num1 - argp->num2;
    return &result;
}

float *
mul_7_svc(values *argp, struct svc_req *rqstp)
{
    static float  result;
    result = argp->num1 * argp->num2;
    return &result;
}

float *
div_7_svc(values *argp, struct svc_req *rqstp)
{
    static float  result;
    result = argp->num1 / argp->num2;
    return &result;
}

float *
pow_7_svc(values *argp, struct svc_req *rqstp)
{
    static float  result;

    for (int i=0; i<argp->num2; i++)
    {
```

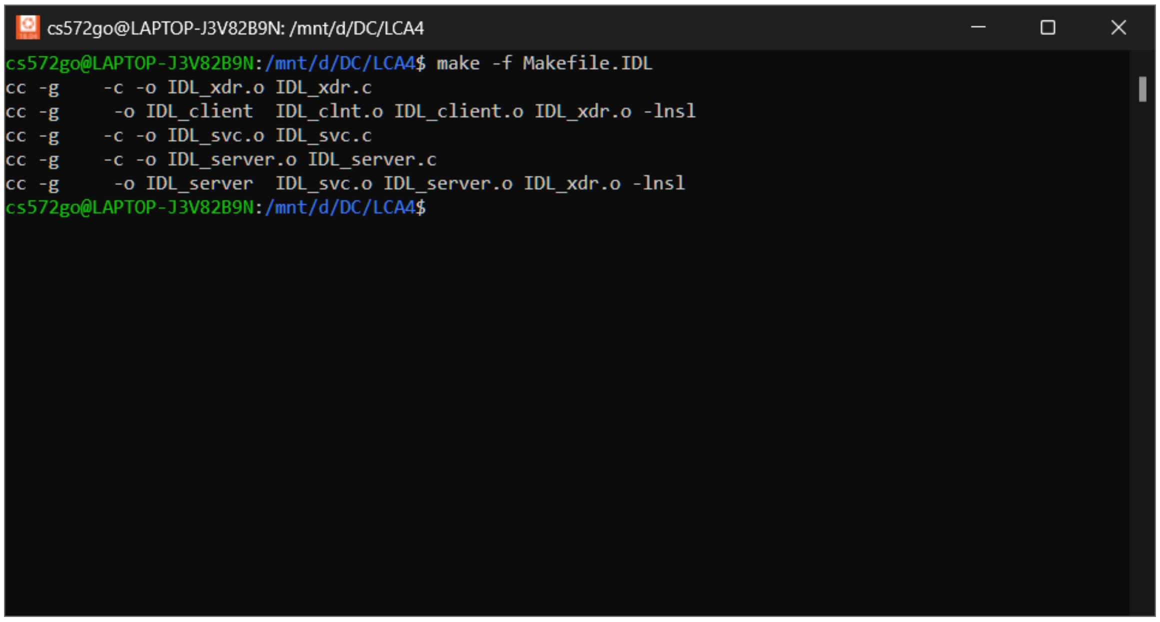
TUSHAR DESHMUKH
PE27
1032201698

```
        result = result * argp->num1;
    }
    return &result;
}
```

OUTPUT:

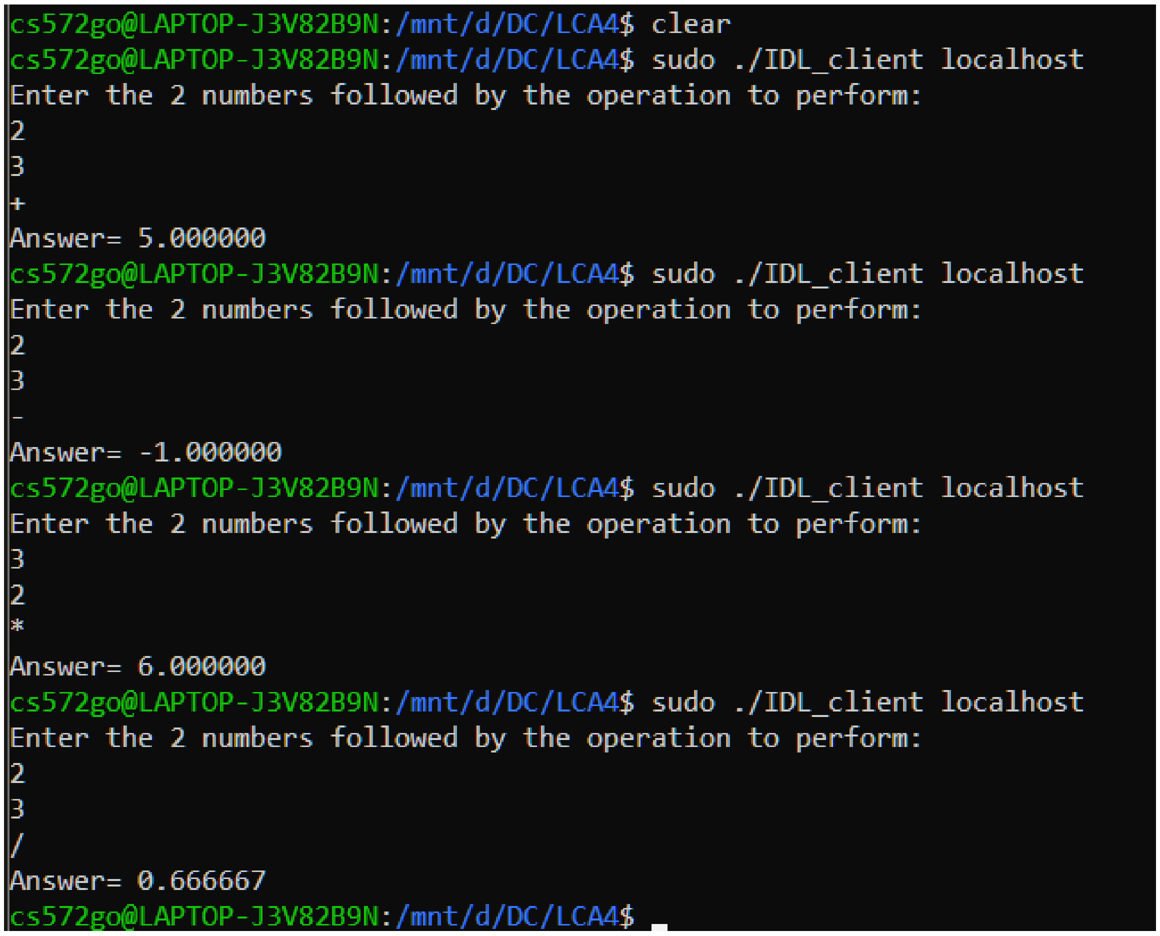


```
cs572go@LAPTOP-J3V82B9N:~$ sudo apt-get install rpcbind
[sudo] password for cs572go:
Reading package lists... Done
Building dependency tree
Reading state information... Done
rpcbind is already the newest version (0.2.3-0.6ubuntu0.18.04.4).
rpcbind set to manually installed.
The following packages were automatically installed and are no longer required:
  efibootmgr libefiboot1 libefivar1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
cs572go@LAPTOP-J3V82B9N:~$
```



```
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ make -f Makefile.IDL
cc -g    -c -o IDL_xdr.o IDL_xdr.c
cc -g     -o IDL_client  IDL_clnt.o IDL_client.o IDL_xdr.o -lnsl
cc -g    -c -o IDL_svc.o IDL_svc.c
cc -g    -c -o IDL_server.o IDL_server.c
cc -g     -o IDL_server  IDL_svc.o IDL_server.o IDL_xdr.o -lnsl
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$
```

TUSHAR DESHMUKH
PE27
1032201698



```
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ clear
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ sudo ./IDL_client localhost
Enter the 2 numbers followed by the operation to perform:
2
3
+
Answer= 5.000000
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ sudo ./IDL_client localhost
Enter the 2 numbers followed by the operation to perform:
2
3
-
Answer= -1.000000
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ sudo ./IDL_client localhost
Enter the 2 numbers followed by the operation to perform:
3
2
*
Answer= 6.000000
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$ sudo ./IDL_client localhost
Enter the 2 numbers followed by the operation to perform:
2
3
/
Answer= 0.666667
cs572go@LAPTOP-J3V82B9N:/mnt/d/DC/LCA4$
```