CSCI 53700-FALL 2016

Report for Assignment 3

-simple rpcgen examples

Feng Li

11/21/16

1. Problem Description

In this lab, the task is to develop a simple distributed computing environment consisting of multiple clients and a server. Rpcgen utility is used to implement the following functions:

1. print the server side date and time
2. Merge two lists
3. Reverse the client input
4. return a list of current directory
5. Add two integer matrix

2. System Design

System Overview

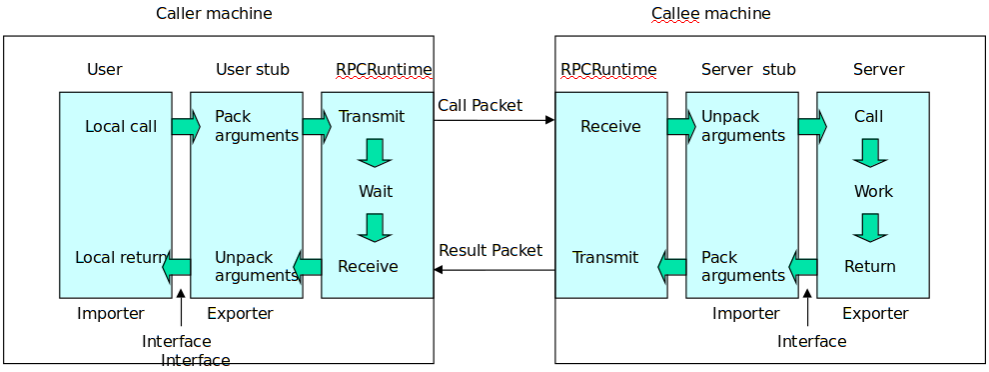

Figure 1 Remote Procedure Call

As shown in figure 1, RPC is based on stubs, which makes remote calls as simple as local calls. Remote calls involves 5 components:

1. user
2. user stub
3. RPC communication package(rpc runtime)
4. server stub
5. server

When the caller wants make a remote call, it will interact with the client side stub and the call will be invoked in the server(callee) side. The RPC runtime is responsible for re-transmission, acknowledgments, packets routing and encryption.

3. Implementation

Rpcgen utility requires a C-stype input file, in which different datatypes can be defined and remote functions are declared. Each remote program can have different versions and each version will include multiple functions with corresponding function index.

```
/* The directory program definition */

program CLOCKPROG
{
    version MYVERS
    {
        string GETTIME()=1;
        intlist MERGE(coupled_int_list l) = 2;
        string REVERSE(string s) = 3;
        string READDIR(string s) = 4;
        matrix ADDMATRIX(coupled_matrix m) = 5;
    } = 1;
} = 0x31111111;
~
```

Using ‘rpcgen -C *.c -a’ command, the rpcgen utility will generate the server and client side C stubs, Makefile and application skeletons in both server and client side .

In the thread-unsafed configuration(without ‘-M’ option) of rpcgen, the returned value of rpcgen is stored in static variables; In the thread-safe configuration, a pointer to the results need to be passed in to the rpcgen-generated code instead.

4. Results

As shown in the sample outputs(‘examples.txt’) from the source code package, client can invoke remote procedure using rpcgen stubs. Following are some screenshots of different functions

data and time

this will invoke the time() function in server side and return a string which stores the current time information in the server.

```
lifen@in-csci-rrpc03:~/Workspace/Feng_DS3$./my_rpc_client 10.234.136.56
1. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
1

user input 1
time in server: Mon Nov 21 11:32:50 2016
```

Merge two lists

Lists are marshaled in client side and then sent to the server. Once client gets results from RPC call, it will un-marshal the result list and display in local terminal.

```
1. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
2

user input 2
list 1:
                5 4 3 2 1
list 2:
                7 6 5 4 2
after remote merge
                5 4 3 2 1 7 6 5 4 2
```

ReverseEcho

Similar to merge, ‘string’ datetype is used.

```
1. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
3

user input 3
please type a sentence
hello world how are you
string read: hello world how are you

after reverse: uoy era woh dlrow olleh
```

List files

In the server side, functions from “dirent.h” are called to get all the file names in current server directory.

```
1. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
4

user input 4
all the files under current directory:
        the files in directory ".":
.       ..      Makefile.my_rpc my_rpc_server.c my_rpc_svc.o    multithread    m
y_rpc_clnt.c    my_rpc_svc.c    Makefile        .my_rpc_client.c.swp    my_rpc_x
dr.o    my_rpc.h        my_rpc.x        README  .git    my_rpc_client.c my_rpc_c
lient.o my_rpc_xdr.c    my_rpc_client   my_rpc_server   archives        my_rpc_s
erver.o my_rpc_clnt.o
1. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
```

Add matrix:

Matrix is defined in two parts:

1. dimensions d1 and d2

2. a 1-d array to store all the elements.

The result can be shown in the following screenshot:

```
l. datatime; 2. Merge 3. ReverseEcho; 4.ListFile 5.Add Matrix q: quit
5

user input 5
matrix a:
        1       2       3
        4       5       6
matrix b:
        1       2       3
        4       5       6
array is mashalled
result:
        2       4       6
        8       10      12
```

Quit and unexpected input

When the client receives a ‘q’ from user, the program will exit. If user types unexpected characters, the system will ignore it and ask user to input again.

5. Conclusion

Rpcgen is a C/C++ implementation of RPC principal. By using a C-style input file, this utility can automatically generate C files for stubs in server and client sides, along with other configurations files for XDR and package making. It significantly reduces the difficulty when developing RPC programs.

XDR datetypes can be defined in the input file of rpcgen, in which user can declare different datatypes from simplest integer to more complicated structures. Since rpcgen will take care of all communication between stubs, the

remaining work for user is only to fill in the remote function implementation and client side invoking routines.

Rpcgen can be also configured with thread-save option ‘-M’ which will will make a multi-threading server which can accepts requests from different clients in the same time.

Pros and cons:

In the current implementation, there are still some limitations. For example, in the ‘list merge’ and ‘matrix add’ function, the user doesn’t provide the real ‘input’. If user want to change different input instances, he has to modify the source code and re-compile the project.

Multithreading is not completed yet. So the test results can only show the communication between server and one client.

References:

1.http://www.cisco.com/c/en/us/td/docs/ios/sw_upgrades/interlink/r2_0/rpc_pr/rprpcgen.html

2. https://docs.oracle.com/cd/E19683-01/816-1435/rpcgenpguide-21470/index.html

3.Figure 1 is from lecture slides of CSCI 53700 in IUPUI.