Tutorial: Writing A Simple Distributed System with RPC in Linux

Step 0: Set up environment in Linux

- Pyrite cluster support RPC

  Alternatively:
- If you don't have Linux run natively, you can install Linux (such as Ubuntu) on a virtual machine (such as virtualbox).
- Run rpcinfo to check if rpcbind has been installed. If not, install rpcbind (Ubuntu as example) as follows:
  - $sudo apt-get update && apt-get upgrade
  - $sudo apt-get install rpcbind

Next, we learn RPC through Example – a Remote Date Example

Step 1: Develop an IDL file – date.x

```
/*
 * date.x  Specification of the remote date and time server
 */

/*
 * Define two procedures
 *      bin_date_1() returns the binary date and time (no arguments)
 *      str_date_1() takes a binary time and returns a string
 *
 */

program DATE_PROG {
    version DATE_VERS {
        long BIN_DATE(void) = 1;    /* procedure number = 1 */
        string STR_DATE(long) = 2;  /* procedure number = 2 */
    } = 1;                          /* version number = 1 */
} = 0x31234567;                     /* program number = 0x31234567 */
```

Notes:

- Start numbering procedures at 1 (procedure 0 is always the ``null procedure".
- Program number is defined by the user. Use range 0x20000000 to 0x3fffffff.
- Provide a prototype for each function. Sun RPC allows only a single parameter and a single result. Must use a structure for more parameters or return values (see XDRC++ example).
- use *clnt_create()* to get handle to remote procedure.
- do not have to use *rpcgen*. Can handcraft own routines.

Step 2: Compile date.x with rpcgen compiler:

$rpcgen –a –C date.x

This generates the following files:

- Client stub – date_clnt.c
- Server skeleton – date_svc.c
- Sample client program – date_client.c
- Sample server program – date_server.c
- Header file – date.h
- XDR routines used by both the client and the server – no in this example
- Makefile – Makefile.date

When you use pyrite.cs.iastate.edu, make the following change to the Makefile:

```
CFLAGS += -g -I /usr/include/tirpc
LDLIBS += -ltirpc
```

Step 3: Edit the server program (date_server.c) and client program (date_client.c)

```
/* date_server.c */
#include "date.h"
#include <time.h>
long *
bin_date_1_svc(void *argp, struct svc_req *rqstp)
{
	static long  result;
	/*
	 * insert server code here
	 */
	result = time((long *) 0);
	return &result;
}
char **
str_date_1_svc(long *argp, struct svc_req *rqstp)
{
	static char * result;
	/*
	 * insert server code here
	 */
	printf("str_date_1_svc: start!\n");
	result = ctime(argp);
	return &result;
}
```

```
/* date_client.c */

#include "date.h"

void

date_prog_1(char *host)

{

        CLIENT *clnt;

        long  *result_1;

        char *bin_date_1_arg;

        char * *result_2;

        long  str_date_1_arg;

#ifndef DEBUG

        clnt = clnt_create (host, DATE_PROG, DATE_VERS, "udp");

        if (clnt == NULL) {

                clnt_pcreateerror (host);

                exit (1);

        }

#endif  /* DEBUG */

        result_1 = bin_date_1((void*)&bin_date_1_arg, clnt);

        if (result_1 == (long *) NULL) {

                clnt_perror (clnt, "call failed");

        }

        printf("time on host %s = %ld\n", host, *result_1);

        str_date_1_arg = *result_1;

        result_2 = str_date_1(&str_date_1_arg, clnt);

        if (result_2 == (char **) NULL) {

                clnt_perror (clnt, "call failed");

        }

        printf("time on host %s = %s\n", host, *result_2);

#ifndef DEBUG
```

```
	clnt_destroy (clnt);
#endif	 /* DEBUG */
}


int
main (int argc, char *argv[])
{
	char *host;
	if (argc < 2) {
		printf ("usage: %s server_host\n", argv[0]);
		exit (1);
	}
	host = argv[1];
	date_prog_1 (host);
	exit (0);
}
```

Step 4: Compile all the files

```
$make –f Makefile.date
```

Step 5: Run the server and the client

On one terminal:

```
$./date_server
```

One another terminal:

```
$./date_client localhost
```

References:

- http://tharikasblogs.blogspot.com/p/how-to-write-simple-rpc-programme.html
- https://web.cs.wpi.edu/~rek/DCS/D04/SunRPC.html