

Práctica 2

Programación Distribuida y Tiempo Real

Lucas Di Cunzolo
Santiago Tettamanti

Abstract—RPC - Introducción.

Index Terms—Programación Distribuida y Tiempo Real, c, \LaTeX , rpc



1 PUNTO 1

2 PUNTO 2

2.1 A

El flag -N intenta de generar código en un estándar más nuevo que el ANSI (flag -C). Al intentar compilar el código con ese flag, utilizando el código original, falla. Esto se debe a que `rpcgen`, genera un `.h` las funciones con el tipo SIN puntero, a diferencia del flag -C, que los genera como punteros a operand.

2.2 B

El flag -M sirve para generar código seguro para la concurrencia, utilizando un parámetro extra al servicio, de tipo `int*`.

El flag -A, es flag por default, que dependiendo el sistema en el que se compila, va a ser (o no), seguro para la concurrencia multihilo.

A

3 PUNTO 3

`rpcgen` utiliza la estructura definida en el archivo `.x` para generar estructuras C en ambos puntos (cliente y servidor), con las cuales va a trabajar casteando.

El servicio recibe punteros a estas estructuras, las cuales va a trabajar y retornar nuevamente casteando a (`caddr_t`), el cual es equivalente a un `void*` Esto nos permite trabajar con cualquier tipo de C, siempre volviendo a castear a la estructura definida a partir del `.x`

4 PUNTO 4

Implementación.

APPENDIX

```

gcc -c -Wall -DRPC_SVC_FG simp_clnt.c
gcc -c -Wall -DRPC_SVC_FG simp_xdr.c
simp_xdr.c: In function 'xdr_operands':
simp_xdr.c:12:20: warning: unused variable 'buf' [-Wunused-variable]
    register int32_t *buf;
                    ^
gcc -o client simpclient.o simp_clnt.o simp_xdr.o -lnsl
gcc -c -Wall -DRPC_SVC_FG simpservice.c
gcc -c -Wall -DRPC_SVC_FG simp_svc.c
gcc -o server simpservice.o simp_svc.o simp_xdr.o -lrpcsvc -lnsl
simp_svc.o: In function 'simp_prog_1':
simp_svc.c:(.text+0x153): undefined reference to 'simp_prog_1_freeresult'
collect2: error: ld returned 1 exit status
Makefile:15: recipe for target 'server' failed
make: *** [server] Error 1

```

Option	Flag	Comments
-a	-a	Use 32-bit integers
-b	-b	Use 64-bit integers
-c	-c	Use 16-bit integers
-d	-d	Use 8-bit integers
-e	-e	Use 4-bit integers
-f	-f	Use 2-bit integers
-g	-g	Use 1-bit integers
-h	-h	Use 0-bit integers
-i	-i	Use 1-bit integers
-j	-j	Use 2-bit integers
-k	-k	Use 4-bit integers
-l	-l	Use 8-bit integers
-m	-m	Use 16-bit integers
-n	-n	Use 32-bit integers
-o	-o	Use 64-bit integers
-p	-p	Use 128-bit integers
-q	-q	Use 256-bit integers
-r	-r	Use 512-bit integers
-s	-s	Use 1024-bit integers
-t	-t	Use 2048-bit integers
-u	-u	Use 4096-bit integers
-v	-v	Use 8192-bit integers
-w	-w	Use 16384-bit integers
-x	-x	Use 32768-bit integers
-y	-y	Use 65536-bit integers
-z	-z	Use 131072-bit integers
-A	-A	Use 262144-bit integers
-B	-B	Use 524288-bit integers
-C	-C	Use 1048576-bit integers
-D	-D	Use 2097152-bit integers
-E	-E	Use 4194304-bit integers
-F	-F	Use 8388608-bit integers
-G	-G	Use 16777216-bit integers
-H	-H	Use 33554432-bit integers
-I	-I	Use 67108864-bit integers
-J	-J	Use 134217728-bit integers
-K	-K	Use 268435456-bit integers
-L	-L	Use 536870912-bit integers
-M	-M	Use 1073741824-bit integers
-N	-N	Use 2147483648-bit integers
-O	-O	Use 4294967296-bit integers
-P	-P	Use 8589934592-bit integers
-Q	-Q	Use 17179869184-bit integers
-R	-R	Use 34359738368-bit integers
-S	-S	Use 68719476736-bit integers
-T	-T	Use 137438953472-bit integers
-U	-U	Use 274877906944-bit integers
-V	-V	Use 549755813888-bit integers
-W	-W	Use 1099511627776-bit integers
-X	-X	Use 2199023255552-bit integers
-Y	-Y	Use 4398046511104-bit integers
-Z	-Z	Use 8796093022208-bit integers
-a	-a	Use 32-bit integers
-b	-b	Use 64-bit integers
-c	-c	Use 128-bit integers
-d	-d	Use 256-bit integers
-e	-e	Use 512-bit integers
-f	-f	Use 1024-bit integers
-g	-g	Use 2048-bit integers
-h	-h	Use 4096-bit integers
-i	-i	Use 8192-bit integers
-j	-j	Use 16384-bit integers
-k	-k	Use 32768-bit integers
-l	-l	Use 65536-bit integers
-m	-m	Use 131072-bit integers
-n	-n	Use 262144-bit integers
-o	-o	Use 524288-bit integers
-p	-p	Use 1048576-bit integers
-q	-q	Use 2097152-bit integers
-r	-r	Use 4194304-bit integers
-s	-s	Use 8388608-bit integers
-t	-t	Use 16777216-bit integers
-u	-u	Use 33554432-bit integers
-v	-v	Use 67108864-bit integers
-w	-w	Use 134217728-bit integers
-x	-x	Use 268435456-bit integers
-y	-y	Use 536870912-bit integers
-z	-z	Use 1073741824-bit integers
-A	-A	Use 2147483648-bit integers
-B	-B	Use 4294967296-bit integers
-C	-C	Use 8589934592-bit integers
-D	-D	Use 17179869184-bit integers
-E	-E	Use 34359738368-bit integers
-F	-F	Use 68719476736-bit integers
-G	-G	Use 137438953472-bit integers
-H	-H	Use 274877906944-bit integers
-I	-I	Use 549755813888-bit integers
-J	-J	Use 1099511627776-bit integers
-K	-K	Use 2199023255552-bit integers
-L	-L	Use 4398046511104-bit integers
-M	-M	Use 8796093022208-bit integers
-N	-N	Use 17592186044416-bit integers
-O	-O	Use 35184372088832-bit integers
-P	-P	Use 70368744177664-bit integers
-Q	-Q	Use 140737488355328-bit integers
-R	-R	Use 281474976710656-bit integers
-S	-S	Use 562949953421312-bit integers
-T	-T	Use 1125899906842624-bit integers
-U	-U	Use 2251799813685248-bit integers
-V	-V	Use 4503599627370496-bit integers
-W	-W	Use 9007199254740992-bit integers
-X	-X	Use 18014398509481984-bit integers
-Y	-Y	Use 36028797018963968-bit integers
-Z	-Z	Use 72057594037927936-bit integers

Fig. 1. Servidor