CS241 #37 – RPC, Protocol Buffers, anon mmap

1> RPC Concepts & Definitions

What is RPC? Privilege separation?

What is stub code? What is marshalling?

What is server stub code? What is unmarshalling?

2> Implementing RPC in C. Simple stub code example:

3> How do you marshal an int? float? struct? Linked list? Graph?

4> What is IDL (Interface Design Language)?

5> Complexity and latency of RPC vs local calls?

6> Working with structured data

Transferring large amounts of structured data:

JSON vs xml vs Google Protocol Buffers

7> Challenge: What argument(s) to this program will cause it to print "Admin/Debug rights"?

#define N (20)

int admin, debug;

int histogram[N];

static int hash(char\* str) {

int c, h = 0; // sdbm hash

while (c = \*str++)

h = c + (h << 6) + (h << 16) - h;

return h;

}

int main(int argc, char\*\*argv){

while(argc>1) {

char\*word= argv[ --argc];

int h = hash(word);

histogram[ (h<0?-h:h) % N ] ++;

}

if(admin || debug) puts("Admin/Debug rights");

return;

}

**> cp gotcha**

What do these two lines do?

cp ../\*.c .

cp ../\*.c

**> Case study: anonymous mapping**

void quit(char\*mesg) {

fprintf(stderr,"%s\n",mesg);

exit(1);

}

void child(char\* shared) {

for(int i = 0; i < 100;i++) {

// write into shared

sprintf(shared,"! The value of i is %d\n",i);

sleep(1);

}

}

void parent(char\*shared) {

while(1) {

if(\*shared) {

puts(shared);

\*shared = 0;

}

sleep(1);

}

}

int main() {

// Use MAP\_ANON instead of MAP\_FILE

size\_t size = 4096;

char \*shared\_mem = mmap(NULL, size,

PROT\_READ | PROT\_WRITE,

MAP\_ANON | MAP\_SHARED, 0, 0);

if(shared\_mem == (char\*)-1) quit("mmap");

pid\_t pid = fork();

if(pid ==0) {

child(shared\_mem);

} else {

parent(shared\_mem);

}

exit(1);

}