ΑΣΚΗΣΗ ΕΡΓΑΣΤΗΡΙΑΚΟΥ ΜΑΘΗΜΑΤΟΣ #3
ONC (UNIX) Remote Procedure Calls

Στόχος: (α) Να φτιάξουμε (και να ορίσουμε ως τέτοια σε έναν υπολογιστή του δικτύου) μια 'ρουτίνα εξυπηρέτησης' η οποία δεχόμενη ως παραμέτρους δύο ακεραίους, να επιστρέφει το άθροισμά τους. (β) Στη συνέχεια να χρησιμοποιήσουμε τη ρουτίνα αυτή (για μία πρόσθεση) στο κυρίως πρόγραμμά μας που τρέχουμε σε έναν άλλον υπολογιστή του δικτύου.

Α. /* Αρχικές-προπαρασκευαστικές ενέργειες */

Δημιουργήστε αρχικά το απαιτούμενο περιγραφής IDL αρχείο **add.x:**

```
struct intpair {
      int a;
      int b;
};

program ADD_PROG {
            version ADD_VERS {
                  int ADD(intpair) = 1;
      } = 1;
} = 0x23451111;
```

καλούμε στη συνέχεια το πρόγραμμα rpcgen για τη μετάφραση του ανωτέρω αρχείου περιγραφής και την εξαγωγή των απαιτούμενων βασικών αρχείων:

```
rpcgen -C add.x
```

θα δημιουργηθούν αυτόματα για εμάς τα αρχεία add.h, add_svc.c, add_clnt.c add_xdr.c, περισσότερες λεπτομέρειες για τα οποία μπορείτε να βρείτε στο αντίστοιχο πιο πλήρες φυλλάδιο που είναι αναρτημένο στο eclass για το εν λόγω αντικείμενο. Αν το καλέσουμε (το rpcgen) ειδικότερα με την παράμετρο -a θα κάνει επιπλέον για εμάς τα εξής: (α) θα μας δώσει έτοιμα templates κώδικα για τα ανωτέρω αρχεία, καθώς και (β) και ένα έτοιμο makefile ('Makefile.add) για τη μετάφραση. Εκτελέστε πιο συγκεκριμένα τα ακόλουθα:

```
rpcgen -a -C add.x          /* ξανά ... */

make -f Makefile.add        /* για compilation  */

mv Makefile.add Makefile    /* renaming ώστε από δω και στο εξής όποτε χρειάζεται να
                               ξανακάνουμε compilation να πληκτρολογούμε μόνο 'make'... */
```

Η ρουτίνα εξυπηρέτησης (server function) περιέχεται στο αρχείο add_server.c και είναι μία ρουτίνα η οποία προς το παρόν δεν κάνει τίποτα, παρά μόνο έχει τα ακόλουθα μέσα σε σχόλια (προκειμένου να συμπληρώσουμε εμείς αυτό που θέλουμε να κάνει):

```
/*
* insert server code here
*/
```

Αρχικά, αντικαταστήστε το παραπάνω κομμάτι κώδικα με την ακόλουθη εντολή:

```
printf("add function called\n");
```

Πριν κάνετε make για compilation, ανοίξτε το αρχείο Makefile και κάντε τις ακόλουθες αλλαγές (με σκοπό να λειτουργεί ο server πάντα στο foreground και να υπάρχει πλήρης συμβατότητα του παραγόμενου κώδικα με ANSI C, αντίστοιχα):
(α) βρείτε τη γραμμή ορισμού των `CFLAGS:` -> `CFLAGS += -g`
και αλλάξτε τη σε -> `CFLAGS += -g -DRPC_SVC_FG`
(β) βρείτε τη γραμμή ορισμού των RPCGENFLAGS: -> και αλλάξτε τη σε: -> `RPCGENFLAGS = -C`

Β. **/* Αρχικό-δοκιμαστικό τρέξιμο */**

Τώρα, εκτελέστε -> `make`

Ανοίξτε τώρα ένα ακόμα παράθυρο (εκτός από αυτό στο οποίο δουλεύετε).

Στο πρώτο παράθυρο τρέξτε το πρόγραμμα του server:

```
./add_server
```

Στο άλλο παράθυρο τρέξτε το πρόγραμμα του client:

```
./add_client localhost
```

στο παράθυρο του server θα δείτε το ακόλουθο αποτέλεσμα:

```
add function called
```

Γ. **/* Ολοκλήρωση της επιθυμητής λειτουργίας (result = a + b) */**

Ανοίξτε τώρα το αρχείο προγράμματος του client: add_client.c. Προσέξτε ότι στη συνάρτηση add_prog_1 ορίζεται μία μεταβλητή add_1_arg (η οποία είναι τύπου intpair, όπως αυτός ορίστηκε στην αρχή στο αρχείο add.x). Αυτή περνιέται σαν παράμετρος στην κλήση της απομεμακρυσμένης διαδικασίας λίγες γραμμές μετά:

```
result_1 = add_1(&add_1_arg, clnt);
```

Πριν γίνει η ανωτέρω κλήση, αρχικοποιήστε την add_1_arg ώστε να περιέχει τους αριθμούς 123, 22.

```
add_1_arg.a = 123;
add_1_arg.b = 22;
```

Ανοίξτε τώρα πάλι το αρχείο προγράμματος του server: add_server.c. Παρατηρήστε ότι η πρώτη παράμετρος της συνάρτησης add_1_svc είναι η εισερχόμενη παράμετρος τύπου intpair. Προσθέστε τώρα μετά από την εκτύπωση του μηνύματος "add function called\n", άλλη μία εντολή εκτύπωσης η οποία να τυπώνει τις τιμές των εισερχόμενων παραμέτρων, καθώς επίσης και τις απαιτούμενες εντολές για να γίνεται η απαιτούμενη πράξη (result = a + b):

```
printf("add function called\n");
printf("parameters: %d, %d\n", argp->a, argp->b);
result = argp->a + argp->b; printf("returning: %d\n", result);
return &result;
```

Θα χρειαστεί τέλος να προσθέσετε επίσης στο πρόγραμμα του client (ανοίξτε πάλι το αρχείο add_client.c) μια εντολή για να τυπώνεται το αποτέλεσμα που επιστρέφεται σε αυτήν:

```
if (result_1 == (int *) NULL) {
clnt_perror (clnt, "call failed"); }
else {
printf("result = %d\n", *result_1); }
```

Τώρα, μεταφράστε πάλι και εκτελέστε πάλι server και client όπως παραπάνω στο ΒΗΜΑ Β (σιγουρευτείτε πρώτα ότι έχετε σκοτώσει τον προηγούμενο server).

Στο παράθυρο του server θα δείτε τώρα ως αποτέλεσμα το:

```
parameters: 123, 22
returning: 145
```

Στο παράθυρο του client θα δείτε δε ως αποτέλεσμα το:

```
result = 145
```

Δ. **/* Εκτέλεση από άλλον υπολογιστή του δικτύου*/**

Μεταφέρατε το εκτελέσιμο add_client σε έναν άλλον υπολογιστή του δικτύου (με sftp). Συνδεθείτε στον υπολογιστή αυτό του δικτύου (με ssh) και τρέξτε:

```
./add_client mpix      /* mpix: ο δικός σας υπολογιστής όπου έχετε σηκώσει το server */
```

Ε. **/* Διάβασμα των εισόδων (a, b) παραμετρικά από τη γραμμή εντολών */**

Ανοίξτε πάλι το αρχείο προγράμματος του client: add_client.c. και…

1. Αντικαταστείστε τη main() με τον ακόλουθο κώδικα

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

        if (argc != 4) {
                printf ("usage: %s server_host num1 num2\n", argv[0]);
                exit(1);
        }
        host = argv[1];
        if ((a = atoi(argv[2])) == 0 && *argv[2] != '0') {
                fprintf(stderr, "invalid value: %s\n", argv[2]);
                exit(1);
        }
        if ((b = atoi(argv[3])) == 0 && *argv[3] != '0') {
                fprintf(stderr, "invalid value: %s\n", argv[3]);
                exit(1);
        }
        add_prog_1(host, a, b);
}
```

2. Αλλάξτε την add_prog_1 ώστε να δέχεται δύο επιπλέον παραμέτρους (a, b):

```
void
add_prog_1(char *host, int a, int b)
```

3. Διαμορφώστε κατάλληλα την αρχικοποίηση των παραμέτρων a, b πριν σταλούν:

```
add_1_arg.a = a;
add_1_arg.b = b;
result_1 = add_1(&add_1_arg, clnt);
```

4.

Μεταφράστε πάλι και εκτελέστε πάλι server και client (σιγουρευτείτε πρώτα ότι έχετε σκοτώσει τον προηγούμενο server), δίνοντας από τη γραμμή εντολών και τους αριθμούς που θέλετε να προσθέσετε:

```
./add_client mpix 25 12        /* mpix: ο δικός σας υπολογιστής - server */
```