Προγραμματισμός Δικτύων (Ε-01)

2η Προγραμματιστική Εργασία

Επέκταση βασικής βιβλιοθήκης δικτυακού προγραμματισμού και χρήση της για την υλοποίηση παράλληλου εξυπηρετητή πρωτοκόλλου μεταφοράς αρχείων

Διδάσκων: Νικόλαος Ντάρμος
Σχολή Πληροφορικής, Πανεπιστήμιο Ιωαννίνων
Ιωάννινα, Δεκέμβριος 2010

1 Γενικά

Η εργασία αυτή έχει ως στόχο την εισαγωγή των φοιτητών σε έννοιες δικτυακού προγραμματισμού και στις επιπλοκές που εισαγάγουν ο ταυτοχρονισμός και τα UNIX σήματα στη σχεδίαση και υλοποίηση δικτυακού λογισμικού.

Βαρύτητα δίνεται και πάλι στη μεταφερσιμότητα του κώδικα και στη δυνατότητα λειτουργίας του σε διαφορετικά περιβάλλοντα (αρχιτεκτονικές, λειτουργικά συστήματα, κτλ.) και με διαφορετικά πρωτόκολλα (TCP, UDP), καθώς και στις διάφορες σχεδιαστικές επιλογές κατά την υλοποίηση της εργασίας με στόχο την καλύτερη απόδοση του συστήματος.

1.1 Παράδοση

Ως καταληκτική ημερομηνία παράδοσης της εργασίας σας ορίζεται η Κυριακή, 6 Φεβρουαρίου 2011 (23:59:59), ενώ η παράδοση θα είναι μέσω e-mail στο `ntarmos@cs.uoi.gr`. Όπως συζητήθηκε και στο πρώτο μάθημα[1], θα πρέπει να παραδώσετε:

1. Τα αρχεία πηγαίου κώδικα.
2. Ένα απλό Makefile.
3. Αναφορά/σχολιασμό (σε μορφή txt/ps/pdf).

Για οποιαδήποτε απορία ή πρόβλημα μπορείτε να επικοινωνείτε μαζί μου στο παραπάνω e-mail ή στο τηλέφωνο 2651008866 ή κατ' ιδίαν τις ώρες γραφείου που έχουμε συζητήσει στο μάθημα. Υπάρχει ακόμα και η δυνατότητα επικοινωνίας μέσω προγραμμάτων IM· ρωτήστε με για λεπτομέρειες.

2 Βασική βιβλιοθήκη

Στην 1η εργασία υλοποιήσατε μία βασική προγραμματιστική διεπαφή (API) συναρτήσεων δικτυακού προγραμματισμού. Οι συναρτήσεις αυτές σκοπό έχουν να «κρύβουν» τις λεπτομέρειες υλοποίησης και τις κλήσεις συστήματος και να προσφέρουν στον χρήστη μία ενιαία πλατφόρμα ανάπτυξης εφαρμογών.

Στα πλαίσια της δεύτερης προγραμματιστικής εργασίας καλείσθε αρχικά να επεκτείνετε την υλοποίησή σας ώστε να υποστηρίζει:

1. χρήση πολλαπλών διευθύνσεων στα άκρα επικοινωνίας και,
2. παράλληλη εκτέλεση των συναρτήσεών σας (π.χ. από πολλαπλά νήματα ή/και πολλαπλές διεργασίες).

Το «ανανεωμένο» API ορίζει τις ίδιες συναρτήσεις για δημιουργία, σύνδεση και καταστροφή άκρων επικοινωνίας, για αποδοχή αιτήσεων σύνδεσης, καθώς και για ανταλλαγή δεδομένων, ενώ για την αποθήκευση και διαχείριση των πληροφοριών των sockets χρησιμοποιεί και πάλι την ίδια δική του δομή δεδομένων. Στο σχήμα 1 φαίνεται συνολικά το header file το οποίο ορίζει το API αυτό. Παρατηρήστε ότι, τουλάχιστον εξωτερικά, δεν υπάρχει κάποια διαφοροποίηση σε σχέση με το API της 1ης εργασίας.

2.1 Υποστήριξη πολλαπλών διευθύνσεων στα άκρα επικοινωνίας

Στο τμήμα αυτό θα πρέπει να αλλάξετε τον κώδικα των συναρτήσεων της βιβλιοθήκης σας έτσι ώστε να υποστηρίζει την ταυτόχρονη χρήση πολλαπλών διευθύνσεων στα άκρα επικοινωνίας.

[1] `http://www.cs.uoi.gr/~ntarmos/Courses/NetworkProgramming/Lectures/Lecture01.pdf`

Σχήμα 1: myNetLib.h

```
1  #ifndef __MY_NET_LIB_H__
2  #define __MY_NET_LIB_H__
3
4  typedef enum {
5     TCPEndpoint,      // TCP endpoint.
6     UDPEndpoint,      // UDP endpoint.
7  } EndpointType;
8
9  typedef struct {
10    EndpointType       type;     // Type of the endpoint.
11    int                backlog;  // For TCP/SCTP server sockets.
12    int*               sd;       // Table of socket descriptors.
13    int                sdlen;    // # elements in sd.
14    struct addrinfo*   addr;     // Info for the above descriptors.
15 } EndpointInfo;
16
17 int createServerEndpoint(const char *host, const char *service,
      EndpointInfo *info);
18 int createClientEndpoint(const char *host, const char *service,
      EndpointInfo *info);
19 int closeEndpoint(EndpointInfo *info);
20 int getNextClientFromEndpoint(const EndpointInfo *serverInfo,
      EndpointInfo *clientInfo);
21 int sendDataToEndpoint(const EndpointInfo *info, const void *data, size_t
       datalen);
22 int recvDataFromEndpoint(const EndpointInfo *info, void *data, size_t
      datalen);
23
24 #endif
```

Πιο συγκεκριμένα:

- `createServerEndpoint(...)`: Στην περίπτωση που ο κόμβος στον οποίο θα εκτελεστεί η συνάρτηση αυτή έχει παραπάνω από μία διευθύνσεις ή/και υποστηρίζει και IPv4 και IPv6, η συνάρτηση θα πρέπει να προσπαθεί να δημιουργήσει όλα τα δυνατά άκρα επικοινωνίας.
- `getNextClientFromEndpoint(...)`: Όταν ο εξυπηρετητής έχει περισσότερα από ένα sockets στο ίδιο άκρο επικοινωνίας, η συνάρτηση θα πρέπει να επιλέγει κατάλληλα (π.χ. με χρήση της συνάρτησης `select(2)`).

2.2 Υποστήριξη παράλληλης εκτέλεσης συναρτήσεων

Στο τμήμα αυτό θα πρέπει να ελέγξετε και να τροποποιήσετε κατάλληλα τον κώδικά σας ώστε να επιτρέπει παράλληλη εκτέλεση των διάφορων συναρτήσεών σας από πολλαπλά νήματα ή/και πολλαπλές διεργασίες. Ουσιαστικά αυτό έγκειται στη χρησιμοποίηση συναρτήσεων που είναι ασφαλείς για νήματα (thread-safe) ή/και σήματα (signal-safe) καθώς και στην αποφυγή χρήσης καθολικών και στατικών μεταβλητών. Συμβουλευτείτε τα man pages των συναρτήσεων βιβλιοθηκών που καλείτε στον κώδικά σας και των συναρτήσεων σημάτων και νημάτων.

3 Εφαρμογή μεταφοράς αρχείων

Στο δεύτερο κομμάτι της εργασίας αυτής σας ζητείται να τροποποιήσετε τον απλό πελάτη και τον εξυπηρετητή που υλοποιήσατε στην 1η προγραμματιστική εργασία ώστε αυτοί να εκμεταλλεύονται την πρόσθετη λειτουργικότητα της «ανανεωμένης» βασικής βιβλιοθήκης.

3.1 Πρωτόκολλο μεταφοράς αρχείων

Για καλύτερο έλεγχο, στην εργασία αυτή θα κάνετε κάποιες αλλαγές στο πρωτόκολλο μεταφοράς αρχείων που προδιαγράφηκε στην πρώτη εργασία.

Πιο αναλυτικά:

- Ο πελάτης `upload`, αφού ανοίξει το τοπικό αρχείο και συνδεθεί με τον εξυπηρετητή, θα στέλνει την null-terminated (`'\0'`) συμβολοσειρά «`PUT filename`», το μέγεθος του αρχείου σε bytes ως 64-bit ακέραιο (προσοχή με το endianness!) και κατόπιν τα δεδομένα του αρχείου στον εξυπηρετητή. Στο τέλος της μεταφοράς ο εξυπηρετητής θα πρέπει να επιστρέφει στον πελάτη, πάλι ως 64-bit ακέραιο, τον αριθμό των bytes που έγραψε στο αρχείο ή -1 για λάθος.
- Ο πελάτης `download`, αφού ανοίξει το τοπικό αρχείο και συνδεθεί με τον εξυπηρετητή, θα στέλνει και πάλι μόνο την null-terminated συμβολοσειρά «`GET filename`» στον εξυπηρετητή. Η απάντηση του εξυπηρετητή θα αποτελείται από το μέγεθος του αρχείου σε bytes ως 64-bit ακέραιο (με την «ειδική» τιμή -1 για λάθος στην πλευρά του εξυπηρετητή) και κατόπιν τα δεδομένα του αρχείου.
- Φροντίστε οι πελάτες και ο εξυπηρετητής σας να μην μένουν μπλοκαρισμένοι για πάντα στην περίπτωση που από την εφαρμογή χρησιμοποιείται πρωτόκολλο UDP και κάποια δεδομένα χαθούν. Επίσης προσέξτε ότι για τη σωστή λειτουργία του πελάτη και του εξυπηρετητή, θα πρέπει η μεταφορά των δεδομένων να γίνεται σε δυαδική (binary) μορφή.

3.2 Υποστήριξη ταυτοχρονισμού για TCP εξυπηρετητές

Στο τμήμα αυτό της εργασίας, καλείσθε να προσθέσετε στον εξυπηρετητή σας τη δυνατότητα ταυτόχρονης εξυπηρέτησης πολλαπλών πελατών πρωτοκόλλου TCP. Θα πρέπει να προτείνετε έναν σχεδιασμό τον οποίο εσείς θεωρείτε κατάλληλο για το έργο αυτό και στη συνέχεια να τον υλοποιήσετε. Στόχος είναι ένας εξυπηρετητής ο οποίος θα μπορεί να υποστηρίξει παράλληλα 10.000 πελάτες[2]. Στην αναφορά σας θα πρέπει να εξηγήσετε την αρχιτεκτονική στην οποία καταλήξατε καθώς και τους λόγους που σας οδήγησαν σε αυτή την απόφαση. Για την επιλογή σας θα πρέπει να λάβετε υπόψιν σας (και να εξηγήσετε στην αναφορά σας), μεταξύ άλλων:

- Την ορθολογιστική λειτουργία του εξυπηρετητή σας.
- Τις απαιτήσεις που θα έχει η σχεδίασή σας από το λειτουργικό σύστημα και το υλικό του κόμβου στον οποίο θα εκτελείται.
- Την πολυπλοκότητα υλοποίησης και αποσφαλμάτωσης/συντήρησης της σχεδίασής σας.

Για την υλοποίηση του εξυπηρετητή σας θα πρέπει και πάλι να χρησιμοποιήσετε την βιβλιοθήκη που υλοποιήσατε στο πρώτο μέρος της εργασίας σας. Είστε ελεύθεροι να παρέμβετε

[2] `http://www.kegel.com/c10k.html`

στον κώδικά της αν θεωρείτε ότι απαιτείται, καθώς και να προσθέσετε κώδικα ο οποίος εξαρτάται από το λειτουργικό σύστημα στο οποίο κάνετε την ανάπτυξη (αν για παράδειγμα θέλετε να χρησιμοποιήσετε κλήσεις συστήματος που είναι διαθέσιμες μόνο σε Linux, BSD ή κάποιο άλλο λειτουργικό), αλλά θα πρέπει να το αναφέρετε και να εξηγήσετε την επιλογή σας στο παραδοτέο. Σε κάθε περίπτωση, όμως, δεν πρέπει να αλλάξετε την προγραμματιστική διεπαφή της βιβλιοθήκης· με άλλα λόγια, θα πρέπει ακόμα να μπορείτε – υποθετικά – να αντικαταστήσετε τα αρχεία της υλοποίησης της βιβλιοθήκης σας με αυτά κάποιου συναδέλφου σας και ο κώδικας να συνεχίσει να λειτουργεί.

3.3 Υποστήριξη ταυτοχρονισμού για UDP εξυπηρετητές

Στο τελευταίο τμήμα της εργασίας αυτής καλείσθε να προσθέσετε στον εξυπηρετητή σας υποστήριξη και για ταυτόχρονη εξυπηρέτηση πολλαπλών πελατών πρωτοκόλλου UDP. Οι απαιτήσεις είναι οι ίδιες με αυτές του πρώτου τμήματος. Προσέξτε ότι για την ορθή υποστήριξη του UDP με πολλαπλούς παράλληλους πελάτες ενδεχομένως να χρειασθεί επανασχεδιασμός των συναρτήσεων της βασικής σας βιβλιοθήκης ή του εξυπηρετητή σας. Το τμήμα αυτό είναι προαιρετικό και θα μετρήσει προσθετικά στη βαθμολογία σας, εφόσον υλοποιηθεί σωστά.