Dr. M. Grabert
Johannes Mayer
Abteilung Angewandte Informationsverarbeitung

19. Juni 2001
Blatt 7

Systemnahe Software (SS 2001)

Abgabetermin: 25. Juni 2001

Beispiellösung

11 Wie komm' ich bloß an meine Mails? (10 Punkte)

Wollten Sie nicht immer schon mal eine kleine Übersicht, was sich denn so alles in Ihrer Mailbox tummelt. Am liebsten wäre es Ihnen, wenn sie nur auf der Kommandozeile ein Kommando eintippen müssten und schwups schon haben Sie Titel, Absender und Datum der letzten sagen wir 5 Mails. Na gut, können wir machen!
Schreiben Sie also ein C-Programm welches (ähnlich zu `popen()`) in einem Sohn-Prozess entweder `telnet` oder `nc` ausführt. (Als Kommandozeilenargumente müssen der Server und der Port an `telnet` oder `nc` übergeben werden!) Zu diesem Prozess brauchen wir aber (im Gegensatz zu `popen()`) zwei Verbindungen – eine zum Lesen und eine zum Schreiben. Danach kommuniziert der Vater-Prozess via POP3-Protokoll mit dem Server (über die beiden Pipe-Verbindungen). Zunächst loggt er sich durch die Kommandos `USER` und `PASS` ein. Danach fordert er mittels `LIST` eine Liste der Mail-Nummern an. (Diese ist so sortiert, dass die neueste Mail am Ende steht.) Und merkt sich die Nummern der letzten 5 Zeilen in dieser Liste (oder entsprechend weniger, falls die Liste nicht so lang ist). Daraufhin fordert er von jeder dieser max. 5 Mails mittels `top` den Header an. Von den Zeilen, die er dabei bekommt, gibt er aber nur diejenigen aus, welche mit `From:`, `Date:` oder `Subject:` beginnen. Zum Schluss beendet er mit dem Kommando `quit` die Verbindung zum POP3-Server. Den Namen des Servers, den Benutzernamen und das Passwort erhält das C-Programm via Kommandozeile.
Im folgenden ist ein Beispiel für eine POP3-Session, bei der der Header der neuesten Mail angefordert wird, angegeben.

```
turing$ nc turing 110
+OK <23742.992617730@turing.mathematik.uni-ulm.de>
user jmayer
+OK
pass XXXXXXXX
+OK
list
+OK
```

```
1 2548
2 3150
.
top 2 0
+OK
Return-Path: <weber@informatik.uni-ulm.de>
Delivered-To: jmayer@mathematik.uni-ulm.de
...
Message-ID: <3B23552B.99F5C879@informatik.uni-ulm.de>
Date: Sun, 10 Jun 2001 13:08:27 +0200
From: Michael Weber <weber@informatik.uni-ulm.de>
Organization: University of Ulm
...
To: all@informatik.uni-ulm.de
Subject: Diplomfeier
Content-Type: text/plain; charset=iso-8859-1
Content-Transfer-Encoding: 8bit


.
quit
+OK
turing$
```

Ihr Programm sollte sich dann in etwa so verhalten:

```
thales$ pop3 turing jmayer XXXXXXXX
Date: Fri, 15 Jun 2001 15:07:45 +0200
From: Birgit Lange <birgit.lange@bibliothek.uni-ulm.de>
Subject: Fwd: Testzugang zu EBSCO Business Source Premier
----------------------------------------
Date: Sun, 10 Jun 2001 13:08:27 +0200
From: Michael Weber <weber@informatik.uni-ulm.de>
Subject: Diplomfeier
thales$
```

P.S.: Richten Sie Ihren POP3-Mail-Account auf unseren Rechnern ein, falls sie das nicht bereits getan haben. (Hilfe hierzu gibt es unter *http://www.mathematik.uni-ulm.de/admin/qmail/pop.html.*) Arbeiten Sie auf Ihrem eigenen POP3-Account!

Lösung:

```
/*
 * AI4 SS01 Blatt 7 (c) mg,jm sai uni ulm 2001
 * Via POP3-Protokoll die Header der letzten ... Mails
 * holen und ausgeben - unter Verwendung von telnet oder nc.
 */
```

```
#include <stdio.h>
#include <unistd.h>
#include <signal.h>
#include <wait.h>
#include <fcntl.h>
#include <string.h>

#define PROGRAM  "nc"       /* nc oder telnet */
#define PORT     "110"      /* Portnummer (als String!) */
#define LAST     5          /* von den letzten ... Mail */

/*
 * Kommando cmd (evtl. mit Argumenten arg1 und arg2) nach fpwrite
 * schreiben und auf eine Antwort (eine Zeile warten).
 * Rueckgabewert: true, falls Antwort mit "+OK" beginnt, und sonst false
 */
int putCommand(FILE *fpread, FILE *fpwrite, char *cmd, char *arg1, char *arg2) {
   char line[512], *s = line;

   if (arg1 == NULL)                          // Kommando senden
      fprintf(fpwrite, "%s\n", cmd);
   else if (arg2 == NULL)
      fprintf(fpwrite, "%s %s\n", cmd, arg1);
   else
      fprintf(fpwrite, "%s %s %s\n", cmd, arg1, arg2);

   fflush(fpwrite);                           // Sende-Puffer leeren

   fgets(line, sizeof(line), fpread);         // Antwort holen
   if (!strncmp("+OK", line, 3)) return 1;    // Antwort = +OK ?

   s += 4;                                    // "-ERR" ueberspringen
   while (*s == ' ') s++;                     // Leerzeichen danach auslassen

   fprintf(stderr, "Server Error: %s", s);    // Fehlermeldung ausgeben

   return 0;
}

/*
 * Gibt die Headerzeilen der Mail Nummer num, die mit From:, Subject: oder
 * Date: beginnen, auf die Standardausgabe aus.
 * Rueckgabewert: true, falls das POP3-Kommando top Erfolg hat, und sonst false
 */
int getMessage(FILE *fpread, FILE *fpwrite, int num) {
   char line[512];
```

```
   sprintf(line, "%d", num);         // Kommando top senden
   if (!putCommand(fpread, fpwrite, "top", line, "0")) return 0;

      /*
       * Zeile fuer Zeile lesen, bis eine Zeile kommt, deren
       * erstes Zeichen "." ist (und deren 2. Zeichen nicht "." ist).
       * Nur die relevanten From:-, Subject:- und Date:-Zeilen
       * ausgeben.
       */
   while (1) {
      fgets(line, sizeof(line), fpread);
      if (line[0] == '.' && line[1] != '.') break;
      if (!strncmp("From:", line, 5) ||
          !strncmp("Subject:", line, 8) ||
          !strncmp("Date:", line, 5))
         printf("%s", line);
   }

   return 1;
}

void pop3client(FILE *fpread, FILE *fpwrite, char *user, char *passwd) {
   char line[512];
   int n, num[LAST], i, j, k;

      /*
       * Am Anfang alle Zeilen weglesen, bis eine Zeile kommt,
       * die mit "+OK" beginnt.
       */
   while (fgets(line, sizeof(line), fpread) && strncmp("+OK", line, 3));

      /*
       * Authentisieren und Liste der Mail-Nummern anfordern
       */
   if (!putCommand(fpread, fpwrite, "user", user, NULL)) return;
   if (!putCommand(fpread, fpwrite, "pass", passwd, NULL)) return;
   if (!putCommand(fpread, fpwrite, "list", NULL, NULL)) return;

      /*
       * Nummern der Mails einlesen (wird durch eine Zeile, die
       * mit "." und nicht ".." beginnt beendet).
       * Nur die letzten LAST Nummern speichern.
       */
   n = 0;
   while (1) {
```

```
      fgets(line, sizeof(line), fpread);
      if (line[0] == '.' && line[1] != '.') break;
      sscanf(line, " %d ", &num[n++ % LAST]);
   }

      /*
       * LAST neueste Mails ausgeben (falls es so viele gibt).
       * Nach Datum sortiert (automatisch) mit der neuesten
       * Mail zuletzt.
       */
   k = n > LAST ? LAST : n;   // Anzahl auszugebender Header
   i = n - k;                 // neueste Mail zuletzt
   for (j = 1; j <= k; j++) {
      if (j > 1) puts("----------------------------------------");
      if (!getMessage(fpread, fpwrite, num[i++ % LAST]))
         return;
   }

      /*
       * Sitzung mit dem POP3-Server beenden.
       */
   if (!putCommand(fpread, fpwrite, "quit", NULL, NULL)) return;
}

int main(int argc, char **argv) {
   int pid;
   int cspfd[2], scpfd[2];
   FILE *fpread, *fpwrite;

   if (argc != 4) {
      fprintf(stderr, "Usage: %s <pop3 server> <user name> <password>\n", argv[0]);
      exit(1);
   }

   if (pipe(cspfd) < 0          // Client ----> Server (Pipe)
       || pipe(scpfd) < 0)      // Server ----> Client (Pipe)
      perror("pipe()"),
      exit(1);

   switch (pid = fork()) {
      case -1:                  // Fehler beim forken ;-(
         perror("fork()");
         exit(1);
      case 0:                   // Sohn = Server
            /*
             * Umlenkungen:
```

```
         * - Standardeingabe liest aus Client-Server-Pipe
         * - Standardausgabe schreib in Server-Client-Pipe
         * - Standarderror landet im Muell
         */
      close(0); dup(cspfd[0]);
      close(1); dup(scpfd[1]);
      close(2); open("/dev/null", O_WRONLY);

      close(cspfd[0]); close(cspfd[1]);   // alle Pipe-Enden zumachen
      close(scpfd[0]); close(scpfd[1]);   // ... es gibt Duplikate!

        /*
         * argv[1] = Name des Servers
         */
      execlp(PROGRAM, PROGRAM, argv[1], PORT, NULL);

      perror("execlp()");
      exit(1);
   default:                  // Vater = Client
      fpread = fdopen(scpfd[0], "r");
      fpwrite = fdopen(cspfd[1], "w");

      close(cspfd[0]); // scpfd[0] nicht schliessen!! Warum?
      close(scpfd[1]); // cspfd[1] nicht schliessen!! Warum?

        /*
         * argv[2] = Benutzername, argv[3] = Passwort
         */
      pop3client(fpread, fpwrite, argv[2], argv[3]);

      fclose(fpread);
      fclose(fpwrite);

      sleep(1);
      if (waitpid(pid, NULL, WNOHANG) < 0) {
         fprintf(stderr, "Error: server did not terminate\n");
         kill(pid, SIGKILL);   // nc/telnet killen
         waitpid(pid, NULL, 0);
      }
  }

  return 0;
}
```