РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ БИБЛИОТЕКОЙ "LIBFTP"

Олег Орел

28 октября 1993

ВВЕДЕНИЕ

Библиотека "libftp" предназначена для написания прикладных программ в которых существует необходимость передавать файлы по сети TCP/IP. Она является набором функций начиная от примитивов, таких как открытие соединения по протоколу FTP на удаленную машину, до функций высокого уровня, которые передают файлы сами производя соединение/разъединение с сервером. Все функции имеют прототипы в файле **FtpLibrary.h**, который должен быть помещен в каталог стандартных заголовков. Эти прототипы практически полностью описывают назначение функций и их аргументы, но тем не менее необходимо сказать об общей идеологии построения библиотеки и ее компонент.

Вся библиотека, являясь клиентом, использует с противоположной стороны соединения стандартный **FTPD**.

В ОС UNIX существует проблема обработки разного рода ошибок в том числе ошибок ввода/вывода, в данном инструментарии использован единый механизм возврата результата работы любой функции (макрокоманда **EXIT**, определенная в **FtpLibrary.h**) который позволяет предварительно установив свои или стандартные функции обработки ошибок и функции отладки, писать смысловую часть программы, думая только о ее методе работы в идеальных условиях. Так же в функциях передачи данных в обе стороны существует возможность установить максимальное время ожидания потока данных, и в случае когда оно (время) истечет, вызвать определенную процедуру.

При работе с библиотекой, первой всегда вызывается процедура соединения с сервером которая возвращает указатель на вновь созданную структуру данных (типа **FTP**) о соединении с сервером.

1 Структура данных FTP

int **sock**	— дескриптор канала передачи команд на сервер;
FILE ***data**	— описание канала для передачи данных на сервер;
int **errno**	— значение последнего возвращенного библиотекой значения. В случае если оно отрицательно или равно нулю, то произошла ошибка;
char **mode**	— тип передачи данных;
int ch	— вспомогательная переменная используемая для преобразования потока в режиме передачи текстовых файлов;
STATUS (*func)()	— адрес функции, которая вызывается в случае когда от сервера получен ответ об ошибке;
STATUS (*debug)()	— адрес функции, которая вызывается для отладки протокола;
STATUS (*IO)()	— функция вызываемая в случае потери связи с сервером, или по истечению максимального времени на прием/передачу одного символа.

2 Процедуры соединения/разъединения с сервером

STATUS FtpConnect(FTP **, char *hostname [1])

Создает канал соединения с сервером, находящимся на машине hostname, и создает структуру FTP, возвращая на нее указатель. Если предварительно была выполнена процедура **FtplibDebug**(1), то до соединения с сервером включает стандартные подпрограммы обработки ошибок **FtpDebugDebug**, **FtpDebugError**, **FtpDebugIO** (Глава 3, стр. 3).

STATUS FtpUser(FTP *, char *user)

Посылает серверу имя пользователя. Ранее должно было быть произведено соединение

STATUS FtpPassword(FTP *, char *password)

[1] Имя машины может быть как символьное так и цифровое, например **dxcern.cern.ch** или **128.141.201.96**

Посылает серверу пароль. Ранее должна была быть выполнена процедура **FtpUser**

STATUS FtpAccount(FTP *, char *acct)

Посылает серверу имя акаунта. Эта функция сделана для полного соответствия библиотеки протоколу **FTP**, но т.к. мало таких операционных систем в которых необходим этот атрибут пользователя то функция **FtpAccount** в общем то не нужна. Предварительно должна была быть выполнена процедура **FtpUser**.

STATUS FtpLogin(FTP **, char *hostname, char *user, char *password, char *account)

Последовательно выполняет процедуры **FtpConnect**, **FtpUser**, **FtpPassword**, **FtpAccount** (если это необходимо). Если акаунт отсутствует, как обычно и бывает, вместо него надо передавать значение **NULL**

STATUS FtpBye(FTP *)

Завершает сеанс работы с сервером [2]

3 Процедуры отладки программы

Существует возможность предварительно определить три процедуры: [3]

FtpSetDebugHandler(FTP *,function)

Устанавливает процедуру отладки протокола с удаленным сервером. Если ее определить, то она всегда будет вызывается из стандартной функции приема/передачи сообщения с/на сервера. Должна делать возврат, но в принципе имеет полное право прерывать выполнение программы в случае необходимости.

FtpSetErrorHandler(FTP *,function)

Определят функцию обработки ошибок. После ее определения в случае возвращения сервером неудовлетворительного ответа будет вызываться указанная функция. При этом знак у кода ошибки меняется на '-', и т.о. результат становится меньше нуля.

FtpSetIOHandler(FTP *,function)

[2] Как видно из описания процедур соединения/разъединения из одной программы можно одновременно соединятся с несколькими серверами

[3] Если в любую из функций, описанных ниже, вместо параметра function передать значение **NULL**, то это будет означать отключение отладки. При отключенной отладке результат работы можно определить или же по возвращаемому функцией значению (Если она типа **STATUS**) или по переменной errno в структуре **FTP**

Определение функции обработки ошибок ввода/вывода. При передаче данных или команд на сервер, может возникнуть ситуация когда связь с сервером будет потеряна (сюда входят практически все сбои сети и сбои при работе сервера на удаленной машине) при этом будет вызвана указанная функция. Она вызывается так же по истечению максимального времени при ожидании очередного символа с сервера во время передачи данных. (**timeout**)

FtpDebug(FTP *)
Подключение стандартных функций отладки протокола таких как

FtpDebugError	— печатает строку возвращенную сервером и прерывает программу;
FtpDebugDebug	— печатает строку возвращенную сервером;
FtpDebugIO	— печатает строку **strerror(errno)** и прерывает программу.

Во все процедуры передаются три аргумента:
1. Структура **FTP**;
2. Значение возвращенное функцией, если оно меньше единицы то произошла ошибка;
3. Символьное сообщение описываемое ошибку (char *).

FtplibDebug(1 or 0)
Включает/выключает автоматическое включение всех видов отладки при выполнении функции **FtpConnect(FtpLogin)**

4 Процедуры передачи данных с сервера

STATUS FtpRetrTimeout(FTP *, char *command, char *inp, char *out [4], long time)
Посылает команду **command** на сервер, причем если в команде встретится подстрока %s то на ее место будет подставлена строка **inp**. Создает канал для передачи данных, и то что будет передано сервером в этот канал будет скопирует в локальный файл **out**. Если в течении времени **time**, который измеряется в секундах, с сервера не придет не одного символа, то функция возвратит статус который будет означать ошибку ввода/вывода. В случае когда **timeout**=0, максимальное время на уровне библиотеки равно бесконечности, но в этом случае статус об ошибке ввода/вывода происходит по истечению timeout'а в ядре TCP/IP. Таким образом, если **timeout** в

[4] Если имя локального файла **out** совпадет со строками ***STDIN***, ***STDOUT***, ***STDERR(** то вместо открытия нового файла произойдет дублирование потока соответственно с каналами **stdin**, **stdout**, **stderr**

параметре time больше чем timeout в ядре TCP/IP, он никогда не прервет передачу данных. [5]

FtpRetr(FTP *, char *command, char *inp, char *out)
Вызывает тоже действие что и FtpRetrTimeout, но с выключенным timeout'ом.

FtpGetTimeout(FTP *, char *inp, char *out, long time)
Передает с сервера файл **inp** в локальный файл **out**, при этом устанавливается **timeout=time**.

FtpGet(FTP *, char *in, char *out)
Вызывает функцию **FtpGetTimeout** с выключенным максимальным временем ожидания данных

FtpDirectory(FTP *, char *pat[6], char *out)
Передает содержимое директории, описанного параметром **pat**, с сервера в файл **out**.

FtpDir(FTP *, char *out)
Передает содержимое текущей директории с сервера в файл **out**.

5 Процедуры передачи данных на сервер

FtpStorTimeout(FTP *, char *command, char *inp, char *out, long time)
Передает содержимое локального файла **inp** на сервер, предварительно послав ему команду, составленную из **command** и **out**. Параметр **time**, задает максимальное время на отправку одного символа.

FtpStor(FTP *, char *command, char *inp, char* out)
Вызывает запуск предыдущей процедуры с параметром **time=0**.

FtpPutTimeout(FTP *, char *in, char *out, long time)
Передает локальный файл **in** на сервер в файл с именем **out**, при этом **timeout=time**

FtpPut(FTP *, char *in, char *out)
Вызывает процедуру **FtpPutTimeout** с параметром **time=0**

[5] **Timeout** в ядрах разных TCP/IP разный
[6] Это первый аргумент для команды ls(dir)

6 Процедуры чтения/записи в файл на сервере

Для того, чтобы производить ввод/вывод из/в файлы которые находятся на сервере, причем не копируя их предварительно в локальный файл, а работая непосредственно с оригиналом, существует возможность открыть файл на сервере на чтение/запись/дозапись и затем с помощью или же обычных процедур ввода/вывода или же при помощи процедур **FtpRead** и **FtpWrite**, которые в отличии от первых преобразуют текстовые файлы, производить необходимые операции. [7]

FtpData(FTP *, char *command, char *param, char *mode)
Создает канал для передачи данных предварительно послав серверу команду которая составляется из параметров **command** и **param**. Параметр **mode** указывает может быть или "r" или "w"

FtpOpenRead(FTP *,char *filename)
Открывает для чтения файл с именем **filename** на сервере

FtpOpenWrite(FTP *,char *filename)
Открывает для записи файл с именем **filename** на сервере

FtpOpenAppend(FTP *,char *filename)
Открывает для дозаписи файл с именем **filename** на сервере

FtpOpenDir(FTP *, char *files)
Создает канал для чтения удаленного листинга директории, параметр files передается команде **ls** на сервере в качестве 1-го параметра

int FtpRead(FTP *)
Читает символ из потока данных, если была установлена текстовая мода передачи [8], преобразует переходы на новую строку. При обнаружении конца потока возвращает **EOF**

FtpGetString(FTP *, char *str)
Чтение одной строки из потока данных при помощи функции **FtpRead**.

FtpWrite(FTP *, char c)
Пишет символ в поток данных, если была установлена текстовая мода передачи, преобразует переходы на новую строку. При обнаружении ошибки ввода/вывода возвращает **EOF**

FtpClose(FTP *)
Закрывает ранее открытый поток данных.

[7] Естественно, такие функции как **seek**, **ioctl**, ... для этих файлов недопустимы.
[8] Установлена по умолчанию.

7 Команды для сервера

FtpCommand(FTP *, char *command, char *param, int ok1, ok2, ok3, ..., okN, EOF)
Посылает команду, составленную из параметров **command** и **param**, и считывает ответ сервера, если код ответа не совпадает не с одним значением **ok**, то знак кода ответа меняется на '-'. В случае если установлен handler обработки ошибок вызывает его.

FtpType(FTP *,char *mode)
Устанавливает моду передачи файлов, mode может быть "A", "I", "S",....

FtpBinary(FTP *)
Устанавливает двоичную моду передачи файлов

FtpAscii(FTP *)
Устанавливает текстовую моду передачи файлов

FtpMkdir(FTP *,char *dirname)
Создает директорию на сервере

FtpChdir(FTP *,char *dirname)
Меняет активную директорию на сервере

FtpRm(FTP *,char *filename)
Удаляет файл на сервере

char *FtpPwd(FTP *)
Возвращает активную директорию на сервере

FtpMove(FTP *,char *oldfilename, char *newfilename)
Переименовывает на сервере файл **oldfilename** в файл **newfilename**

FtpGetFile(FTP *,char *filename)
Дает команду серверу начала передачи файла от сервера к клиенту, отличие от **FtpOpenRead** в том что, **FtpGetFile** не создает канал для передачи данных

FtpPutFile(FTP *,char *filename)
Дает команду серверу начала передачи фала от клиента к серверу, отличие от **FtpOpenWrite** в том что **FtpPutFile** не создает канал для передачи данных

FtpPort(FTP *, int a, int b, int c, int d, int e, int f)
Команда серверу создать канал для передачи данных. Причем **a.b.c.d** это IP адрес клиента а **e*256+f** номер порта.

8 Подпрограммы передачи сообщений в/из сервера

FtpSendMessage(FTP *, char *message)
Посылает сообщение серверу

int FtpGetMessage(FTP *)
Принимает сообщение от сервера и возвращает его код

FtpMessage(int Number)
Возвращает по коду сообщения его содержимое

9 Функции полного сеанса работы

FILE *FtpFullOpen(char *filename,char *mode)
Разбирает строку filename которая должна быть или же типа **host/user/password:filename** или же типа **filename**, и в зависимости от этого открывается файл или же на сервере **host** или же локальный файл. Параметр mode должен содержать один или два символа. Первый задает тип открытия файла "r","w" или "a". Второй символ может содержать символ "b" что будет задавать двоичную моду передачи

FtpFullClose(FILE *f)
Закрытие файла

Алфавитный указатель

Содержание