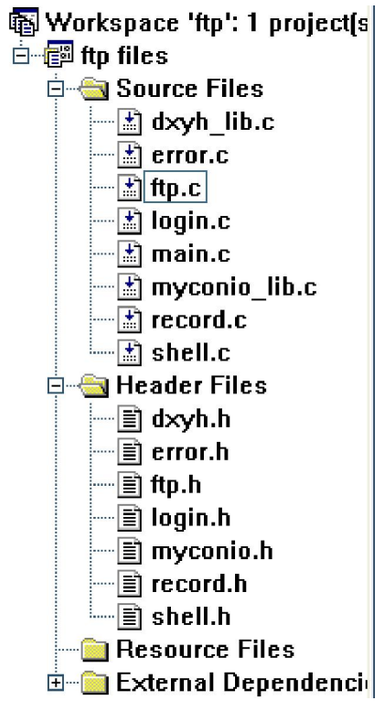
ftp 客户端程序

大家都知道 ftp 是个很实用的工具，传输文件很方便，输入几个命令就可以了。最近闲来无事，将过去写的那个 ftp 服务器翻出来，发现用 Linux/Unix 系统自带的 ftp 客户端可以正常登录，但是当使用到 Windows 下自带的 ftp 客户端程序登录时发现传输不了，连 ls 命令都会失败，所以就打算自己写一个简易的 Windows 下 ftp 客户端程序，方便以后使用。现在有很多人使用虚拟机，当然将虚拟机里面的文件移动到 Windows 下有若干种办法，这个 ftp 客户端配合另外那个 ftp 服务器也可以很方便地实现。

这个客户端程序总体上可分为 3 部分：第一部分是执行具体的 ftp 命令、第二部分是执行具体的 shell 命令，第三部分是支持库（myconio.h）。

所有源文件如下：



dxyh_lib.c —— 常用包裹函数
error.c —— 错误处理
ftp.c —— 具体处理 ftp 客户端命令
login.c —— 处理用户登录
main.c —— 主文件
myconio_lib.c —— 支持库
record.c —— 处理日志
shell.c —— 处理具体的 shell 命令

下面的就是上面源文件的头文件……

具体可支持如下命令（第 1 列所示）：

```
const FTP_CMD ftp_cmds[] = {
    {"system",  ftp_do_syst,    FALSE},
    {"pwd",     ftp_do_pwd,     FALSE},
    {"cd",      ftp_do_cd,      TRUE},
    {"quit",    ftp_do_quit,    FALSE},
    {"bye",     ftp_do_quit,    FALSE},
    {"mkdir",   ftp_do_mkdir,   TRUE},
    {"rmdir",   ftp_do_rmdir,   TRUE},
    {"delete",  ftp_do_dele,    TRUE},
    {"mdelete", ftp_do_mdele,   TRUE},
    {"size",    ftp_do_size,    TRUE},
    {"passive", ftp_do_pasv,    FALSE},
    {"ls",      ftp_do_list,    TRUE},
    {"list",    ftp_do_list,    TRUE},
    {"nlist",   ftp_do_nlst,    TRUE},
    {"type",    ftp_do_type,    TRUE},
    {"binary",  ftp_do_bin,     FALSE},
    {"ascii",   ftp_do_ascii,   FALSE},
    {"put",     ftp_do_put,     TRUE},
    {"mput",    ftp_do_mput,    TRUE},
    {"get",     ftp_do_get,     TRUE},
    {"mget",    ftp_do_mget,    TRUE},
    {"help",    ftp_do_help,    FALSE},
    {"?",       ftp_do_help,    FALSE},
    {NULL, NULL, TRUE}
};
```

下面准备以程序的执行顺序来介绍，不会涉及过多的代码分析，若有不周，见谅。

我的本地 IP：10.6.173.224，虚拟机：10.6.173.225，使用时你只要保证主机和虚拟机能够 ping 通即可。

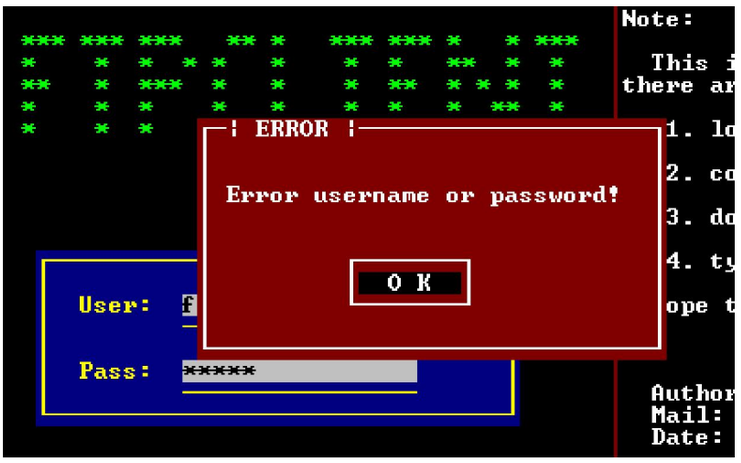
1、开始界面

说明：默认的用户名：ftpclient，密码：123456。

2、输入密码

说明：若使用 input_noechoxy 函数则输入不回显。

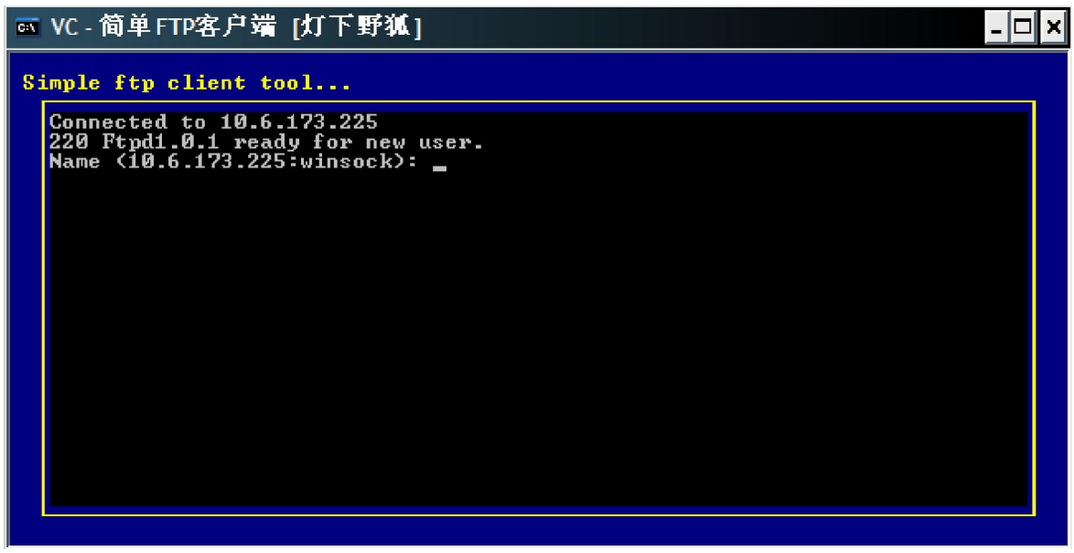
3、输入失败弹出个确认框以示警告



4、下面进入配置界面，若你已正确处理了配置文件，那么一直回车即可。

说明：其中的 tabsize 是设置制表符长度，在执行 shell 命令例如 more、type 等等时有用。

5、按任意键进入到处理 ftp 客户端的部分



说明 若出现此界面，那就说明已成功连接到了 ftp 服务器。输入 ftp 用户名及密码，这里分别为 dengxiayehu 和 123456。

6、成功登录

```
Connected to 10.6.173.225
220 Ftpd1.0.1 ready for new user.
Name (10.6.173.225:winsock): dengxiayehu
331 Please sepcify the password.
Password:
230 Login successful.
215 Linux Type.
ftp>
```

说明：下面就可以开始运行我们说熟悉的 ftp 命令了，下面我仅列举一些，只要达到一个大家在看过后对这个程序的使用有个大致的映像即可，其他的就留给有兴趣研究它的人。

7、命令使用例子，看图就可以了

```
ftp> pwd
257 /
ftp> mkdir try
257 Directory "try" created.
ftp> cd try
250 Directory successfully changed.
ftp> ls
227 Entering Passive Mode (10,6,173,225,131,103).
150 Here comes the directory listing.
226 Directory send OK.
ftp>
```

显示当前目录为根目录、创建了一个文件夹、进入文件夹、列出文件夹内容（此时为空）。

下面是对应于上面命令的服务器的响应：

```
(ftpd.c:512:19098) send resp: 257 /
(ftpd.c:575:19098) Got cmd = MKD try
(ftpd.c:537:19098) received a valid cmd MKD try
(ftpd.c:512:19098) send resp: 257 Directory "try" created.
(ftpd.c:575:19098) Got cmd = CWD try
(ftpd.c:537:19098) received a valid cmd CWD try
(ftpd.c:672:19098) dest dir-path is: try
(ftpd.c:512:19098) send resp: 250 Directory successfully changed.
(ftpd.c:575:19098) Got cmd = PASV
(ftpd.c:537:19098) received a valid cmd PASV
(ftpd.c:1000:19098) local bind: 10.6.173.225: 33639
(ftpd.c:512:19098) send resp: 227 Entering Passive Mode (10,6,173,225,131,103).
(ftpd.c:575:19098) Got cmd = LIST
(ftpd.c:537:19098) received a valid cmd LIST
(ftpd.c:512:19098) send resp: 150 Here comes the directory listing.
(ftpd.c:512:19098) send resp: 226 Directory send OK.
```

8 开始传输文件

注意：要用 ascii 模式传输文本文件，用 binary 模式传输非文本文件，否则文件可能会传输不正常。在具体输入命令时，你可以只要输入命令的前面几个字母就可以了，例如 asc、bin 等等。

在具体介绍之前，先来看看我们这个 shell 可以干些什么：

```
const SHELL_CMD shell_cmd_table[] = {
    { "clear",  shell_clear },
    { "cls",    shell_clear },
    { "exit",   shell_exit },
    { "list",   shell_list },
    { "ls",     shell_list },
    { "dir",    shell_list },
    { "cd",     shell_cd },
    { "pwd",    shell_pwd },
    { "rename", shell_rename },
    { "move",   shell_mv },
    { "mv",     shell_mv },
    { "copy",   shell_cp },
    { "cp",     shell_cp },
    { "mkdir",  shell_mkdir },
    { "rmdir",  shell_rmdir },
    { "delete", shell_delete },
    { "cat",    shell_type },
    { "type",   shell_type },
    { "wc",     shell_wc },
    { "echo.",  shell_echo },
    { "more",   shell_more },
    { "edit",   shell_edit },
    { NULL, NULL }
};
```

说明：所有 shell 命令都以‘！’字符打头，单独按下‘！’的话就进入到循环 shell 模式，提示符变为‘#’。

```
ftp> !_
```

```
[F:\code\C++_COde\socket\ftp]# ls
-rw-     257    Dec 07 17:20    shell.h
-rw-     872    Dec 08 18:12    ftp.h
-rw-    6341    Dec 08 19:52    dxyh_lib.c
-rw-   28145    Dec 08 19:16    shell.c
-rw-    5093    Dec 06 08:36    ftp.dsp
-rw-    9062    Dec 09 01:24    login.c
-rw-     413    Dec 09 00:36    main.c
-rw-   10424    Dec 09 00:10    myconio.h
-rw-    1366    Dec 08 19:52    dxyh.h
-rw-     531    Dec 02 09:04    ftp.dsw
-rw-    2838    Dec 08 18:48    record.c
-rw-     384    Dec 09 00:42    login.h
-rw-   68797    Dec 09 00:30    myconio_lib.c
-rw-   23479    Dec 09 01:22    ftp.c
-rw-    2913    Dec 08 18:22    error.c
-rw-     806    Dec 08 13:54    record.h
-rw-     473    Dec 09 00:14    ftp-configure.ini
-rw-     806    Dec 03 13:34    error.h
[F:\code\C++_COde\socket\ftp]# _
```

说明：此时的 ls 显示的是 win 下的文件内容。

看下面具体的 shell 命令操作示例：

```
[F:\code\C++_COde\socket\ftp]# mkdir F:\test
[F:\code\C++_COde\socket\ftp]# cp shell.c F:\test
[F:\code\C++_COde\socket\ftp]# cd F:\test
(current dir) F:\test
[F:\test]# exit
```

说明：在 F 盘下创建了一个目录，将当前文件夹中的 shell.c 拷贝到该目录，然后进入它，最后输入 exit 命令返回到 ftp 客户端模式。

```
ftp> ascii
200 Switching to ASCII mode.
ftp> pwd
257 /try
ftp> ls
227 Entering Passive Mode (10,6,173,225,131,105).
150 Here comes the directory listing.
226 Directory send OK.
ftp> !pwd
F:\test
ftp> put
(local-file) shell.c
(remote-file) shell.c
local: shell.c remote: shell.c
227 Entering Passive Mode (10,6,173,225,131,106).
150 Ready to receive file.
In ascii mode...
226 File received OK.
28145 bytes transferred in 0.0177 secs(1.6e+003 kbytes/s)
ftp> _
```

说明：以 ascii 模式上传了一个文件 shell.c，我们接下来到虚拟机下去看看是否传输成功。

（服务器的输出）

```
(ftpd.c:575:19098) Got cmd = PASV
(ftpd.c:537:19098) received a valid cmd PASV
(ftpd.c:1000:19098) local bind: 10.6.173.225: 33642
(ftpd.c:512:19098) send resp: 227 Entering Passive Mode (10,6,173,225,131,106).
(ftpd.c:575:19098) Got cmd = STOR shell.c
(ftpd.c:537:19098) received a valid cmd STOR shell.c
(ftpd.c:512:19098) send resp: 150 Ready to receive file.
##########################
Bytes received: 28145
28145 bytes received in 0.0706 secs(3.9e+02 Kbytes/s)
(ftpd.c:1470:19098) STOR (shell.c) successfully
(ftpd.c:512:19098) send resp: 226 File received OK.
```

进入到/try 目录，用 more 命令查看没有问题。

```
[root@localhost try]# cd /try
[root@localhost try]# ll
total 32
-rw-r--r--  1 root root 27035 Nov  4 23:51 shell.c
[root@localhost try]# more shell.c
#include <io.h>
#include <fcntl.h>
#include <direct.h>
#include <sys/stat.h>
#include <time.h>
#include "myconio.h"
#include "dxyh.h"
#include "shell.h"
#include "error.h"

static bool     do_shell_flg    = TRUE;
static bool echo_on_flg         = TRUE;
static char cur_directory[MAX_PATH];
static Status shell_once(const char *arg);
static Status shell(void);
static Status shell_clear(const char *arg);
static Status shell_exit(const char *arg);
static Status shell_list(const char *arg);
static Status show_single_item(const char *filename);
static Status show_directory(const char *dir);
static Status shell_cd(const char *arg);
static Status shell_pwd(const char *arg);
--More--(2%)
```

下面再把它拉出来，这次是使用 bin 模式（对于文本文件来说，虽然传输方式不对，但能做个反面教材）。

```
ftp> ls
227 Entering Passive Mode (10,6,173,225,131,107).
150 Here comes the directory listing.
-rw-r--r-- 1 root root    27035 Nov 04 23:55 shell.c
226 Directory send OK.
ftp> size shell.c
213 Size of "shell.c" is 27035.
ftp> bin
200 Switching to Binary mode.
ftp> get
(remote-file) shell.c
(local-file) back.c
remote-file: shell.c local-file: back.c
227 Entering Passive Mode (10,6,173,225,131,108).
150 File status OK, about to transfer.
226 Require file transferd OK.
27035 bytes received in 0.0131 secs(2e+003 kbytes/s)
ftp> _
```

在 win 下用 notepad 打开看看，发现如下情况：

```
#include <io.h>■#include <fc
echo(const char *arg);■stati
, &st)) {■
cFileName);■            shou
ff_set, "    %-12s\r\n", fil
incorrect.\r\n");■
```

黑块就是这么产生的。

下面再个压缩包，放进去再拉出来：

说明：大概 30 几兆。

```
ftp> type
Using binary mode to transfer file.
ftp> pwd
257 /try
ftp> put
(local-file) arm-elf.tar.gz
(remote-file) arm-elf.tar.gz
local: arm-elf.tar.gz remote: arm-elf.tar.gz
227 Entering Passive Mode (10,6,173,225,131,109).
150 Ready to receive file.
226 File received OK.
36813092 bytes transferred in 10.8 secs(3.3e+003 kbytes/s)
ftp>
```

到虚拟机下去解压：

```
[root@localhost try]# tar xzvf arm-elf.tar.gz .
./usr/local/arm-elf/
./usr/local/arm-elf/bin/
./usr/local/arm-elf/bin/nm
./usr/local/arm-elf/bin/strip
./usr/local/arm-elf/bin/ar
./usr/local/arm-elf/bin/ranlib
./usr/local/arm-elf/bin/as
./usr/local/arm-elf/bin/ld
./usr/local/arm-elf/bin/elf2flt
./usr/local/arm-elf/bin/gcc
./usr/local/arm-elf/bin/ld.real
./usr/local/arm-elf/lib/
./usr/local/arm-elf/lib/ldscripts/
./usr/local/arm-elf/lib/ldscripts/armelf.x
./usr/local/arm-elf/lib/ldscripts/armelf.xbn
./usr/local/arm-elf/lib/ldscripts/armelf.xn
./usr/local/arm-elf/lib/ldscripts/armelf.xr
```

发现解压成功。

好了，到此这个 ftp 客户端介绍得就差不多了，其他命令自己去试试。

下面再来看看一些 shell 命令的使用情况：

rename 命令

```
[F:\test]# ls
-rw-   28145    Dec 09 20:58    shell.c
-rw-36813092    Jan 31 19:10    arm-elf.tar.gz
-rw-   28145    Dec 09 21:30    shell_bak.c
[F:\test]# rename shell_bak.c bak_shell.c
[F:\test]# ls
-rw-   28145    Dec 09 20:58    shell.c
-rw-36813092    Jan 31 19:10    arm-elf.tar.gz
-rw-   28145    Dec 09 21:30    bak_shell.c
[F:\test]#
```

```
[F:\test]# echo Hello ftpclient > 1 >> 2 > 3
[F:\test]# ls
-rw-   28145    Dec 09 20:58    shell.c
-rw-36813092    Jan 31 19:10    arm-elf.tar.gz
-rw-   28145    Dec 09 21:30    bak_shell.c
-rw-      16    Dec 09 21:38    1
-rw-      16    Dec 09 21:38    2
-rw-      16    Dec 09 21:38    3
[F:\test]# type 1 2 3
1th file: 1
Hello ftpclient
2th file: 2
Hello ftpclient
3th file: 3
Hello ftpclient
[F:\test]#
```

上面显示的是 echo 命令的用法，当然还支持 ehco.、echo on、echo off 等。

wc 命令：

```
[F:\test]# ls
-rw-   28145     Dec 09 20:58     shell.c
-rw-36813092     Jan 31 19:10     arm-elf.tar.gz
-rw-   28145     Dec 09 21:30     bak_shell.c
-rw-      16     Dec 09 21:38     1
-rw-      16     Dec 09 21:38     2
-rw-      16     Dec 09 21:38     3
[F:\test]# wc shell.c
 1111   3598   27035  shell.c
[F:\test]# wc shell.c bak_shell.c 1 2 3
 1111   3598   27035  shell.c
 1111   3598   27035  bak_shell.c
    1      2      16  1
    1      2      16  2
    1      2      16  3
 2225   7202   54118  total
[F:\test]#
```

最后介绍一下 more 命令，至于编辑器命令，这个比较耗时，而且比较麻烦，暂时没有写，这里先借用一下著名的编辑器 edit（使用库函数 system 即可）。

```
Simple ftp client tool...
more file: shell.c
#include <io.h>
#include <fcntl.h>
#include <direct.h>
#include <sys/stat.h>
#include <time.h>
#include "myconio.h"
#include "dxyh.h"
#include "shell.h"
#include "error.h"

static bool   do_shell_flg   = TRUE;
static bool echo_on_flg      = TRUE;
static char cur_directory[MAX_PATH];
static Status shell_once(const char *arg);
static Status shell(void);
static Status shell_clear(const char *arg);
static Status shell_exit(const char *arg);
static Status shell_list(const char *arg);
--More--( 1%)
```

按下空格或回车往下输出，esc 退出 more 命令：

```
Simple ftp client tool...
    usr_cprintf(buff);
    return SHELL_OK;
}

static Status shell_pwd(const char *arg)
{
    char path[MAX_PATH];

    if (0 == GetCurrentDirectory(MAX_PATH, path)) {
        win_err_sys("GetCurrentDirectory error");
        return SHELL_ERR;
    }
    usr_cprintf("%s\r\n", path);
    return SHELL_OK;
}

static Status shell_cd(const char *arg)
{
    if (NULL == arg)
--More--(16%)
```

到此这个程序相信大家已会用了，至于那个支持库 myconio.h 的说明文档我已上传到网上过，大家有空找找，不过这里这个库是最新的，也是最为健壮，也就是 bug 最少的一版，呵呵。

至于具体的程序的分析，就留给需要的人他们自己了。

灯 下 野 狐
２０１０年 12 月
Mail:dengxiayehu@yeah.net