准备：

Libssh2.lib （需要手工编译，编译比较麻烦），

编译过程参考：

<http://blog.csdn.net/ly131420/article/details/8888024>

**ActivePerl + OpenSSL + Zlib + Libssh2 + Visual Studio 2008**

SSH2是一套安全通讯协议框架（早期的SSH1由于存在安全漏洞，现在已经不用了），基于SSH2协议的产品目前主要有openssh(<http://www.openssh.org/> )，putty（<http://www.putty.org/> ），SSH Secure Shell Client（从<http://www.moodisk.com/zh_CN/index.html?src=download.php> 可 以下载）等，这些都是开源的，但是这些代码非常难懂而且复杂，一个个函数深层次的调用很快就让人在Ｃ语言代码的海洋中迷失了方向，妄图通过从这些开源软件 中抽取程序代码段来“组装”自己的应用程序是非一般人所能实现的。不过还好网路上出现了一些开源的SSH2开发库，利用这些开发库开发自己的SSH2程序 却要简单得多，由于这些开发库都是开源的，往往是针对linux平台的，而且一般只提供了源代码。在windows上利用这些库还必须要完成：编译有关依 赖库-->编译ssh2库-->集成到开发环境（如Visual Studio）中-->熟悉SSH2库函数用法-->开始编写自己的程序。由于开发基于ssh2协议的例子网上很少，中文资料就更少。本人在 完成这么一个开发环境就断断续续耗费了我一周的时间，现在终于可以开始编写的基于SSH2协议的程序了。我不敢独享，整理出一篇博文，和各位IT同仁分 享。

　　本文的内容安排是：首先介绍如何编译各种依赖库，然后介绍如何把这些依赖库继集成到Visual Studio 2008中，接下来介绍基于SSH2协议的程序的一般框架，最后举一个实际的开发例子。附录A是我自己开发的一个实际例子，附录B列出了全部的 libssh2库函数。

**一·准备一些工具**

　　１、安装Visual Studio 2008开发环境（最好是英文版的，我的是VS2008版本是9.0.30729.1 SP， Windows SDK 6.1）。什么？你不会装！你去google上搜一下“Visual Studio 2008安装过程详解”或者点击参考文章：<http://dev.yesky.com/msdn/329/7823829.shtml>

　　２、安装最新的MSDN文档库（可选，不装也行）。蛮大的，从网上下载要一些时间，或者参考在线文档<http://msdn.microsoft.com/en-us/library/default.aspx>

　　３、安装解压缩工具winRAR。从<http://cncspace.newhua.com/down/files/wrar380sc.exe> 下载并安装。

　　４、安装汇编工具nasm。从<http://www.nasm.us/pub/nasm/releasebuilds/2.06/win32/nasm-2.06-installer.exe> 下载并安装。  
　　５、安装脚本语言ActivePerl。从[http://downloads.activestate.com/ActivePerl/Windows/5.10/ActivePerl-5.10.0.1005-MSWin32-x86-290470.msi](http://downloads.activestate.com/ActivePerl/Windows/5.10/ActivePerl-5.10.0.1005-MSWin32-x86-290470.msi" \t "http://blog.csdn.net/ly131420/article/details/_blank) 下载，然后安装（安装过程中选择默认选项即可）。

**二、编译各种依赖库**

　　LibSSH2库依赖openssl和zlib两个库，所以我们必须先编译zlib和openssl两个库。

　　１、zlib库。网上提供了源码和目标DLL安装包，我们直接下载DLL安装比较快捷，从<http://www.zlib.net/zlib123-dll.zip> 下载，并解压到C:/zlib下（最终存在目录C:/zlib/include即表示正确），把C:/zlib/zlib1.dll拷贝到c:/windows/system32下。

　　２、OpenSSL库。OpenSSL库网上只有源代码，我们首先必须编译。从<http://www.openssl.org/source/openssl-0.9.8k.tar.gz> 下 载源代码包，然后解压到目录C:/openssl-0.9.8k下（最终存在目录C:/openssl-0.9.8k/apps即表示正确）。进入 Visual Studio 2008的命令提示符（开始-->所有程序-->Microsoft Visual Studio 2008-->Visual Studio Tools-->Visual Studio 2008 Command Prompt），依次输入如下命令：

mkdir c:/openssl\_lib

cd C:/openssl-0.9.8k

perl Configure VC-WIN32 --prefix=c:/openssl\_lib

---输出如下的信息：

……

RC4\_CHUNK is undefined

Configured for VC-WIN32.

ms/do\_masm

nmake -f ms/nt.mak

---好了，去喝杯咖啡吧，半个小时后应该编译完了。

nmake -f ms/nt.mak test

---如何库编译正确，你应该看到“passwd all tests”字样。

nmake -f ms/nt.mak install

---现在应该在c:/openssl\_lib下安装了openssl库文件和头文件了。

---如果编译出错，那么也可以查看文件C:/openssl-0.9.8k/INSTALL.W32，里面列举了一些错误处理方法。

　　３、LibSSH2库。LibSSH2库网上只有源代码，我们首先必须编译。从<http://nchc.dl.sourceforge.net/sourceforge/libssh2/libssh2-1.1.tar.gz> 下 载源代码包，然后解压到目录C:/libssh2-1.1下（存在目录C:/libssh2-1.1/include表示正确），创建目录“C: /libssh2/lib”和“C:/libssh2/include”，在Visual Studio 2008 IDE中打开C:/libssh2-1.1/win32/libssh2.dsw，编辑文件libssh2.h，把如下的第54行

# define LIBSSH2\_API \_\_declspec(dllexport)

替换成：

# define LIBSSH2\_API// \_\_declspec(dllexport)

在左边的Solution Explorer中右击libssh2\_lib-->Properties-->Configuration Properties:

-->C/C++ --> General -->选择Additional Include Dirextories-->附加

;C:/openssl\_lib/include;C:/zlib/include

         -->Code Generation --> Runtime Library -->选择Multi-thread(/MT)

-->Librarian --> General:

                  Output File --> C:/libssh2/lib/libssh2.lib

                  Additional Dependencies --> libeay32.lib ssleay32.lib zdll.lib

                  Additional Library Directories --> 附加路径C:/openssl\_lib/lib;C:/zlib/lib

最后点击“OK"确定。

现在可以开始编译了，右击libssh2\_lib选择Build。如果编译成功那么在C:/libssh2/lib下存在文件 libssh2.lib了。接下来再把C:/libssh2-1.1/include下的全部文件拷贝到C:/libssh2/include下，把文件 C:/libssh2-1.1/win32/libssh2\_config.h也拷贝到C:/libssh2/include下。好了，那么我们最终编译 的库和头文件布局如下：

C:/libssh2/include下有文件：libssh2.h，libssh2\_config.h，libssh2\_publickey.h，libssh2\_sftp.h；

C:/libssh2/lib下有文件libssh2.lib

--- 如果编译失败，你可以从[这里](http://www.moodisk.com/download_other_c.php) 直接下载我编译好的LibSSH2库。

**三·一个实例**

　　windwos平台上基于SSH2协议的程序框架是：

初始化Winsock动态库（WSAStartup()）

|

V

创建标准的socket套接字并连接（socket(),connect()）

|

V

创建和启用一个SSH2会话（libssh2\_session\_init(),libssh2\_session\_startup()）

|

V

获取指纹数据（libssh2\_hostkey\_hash()）

|

V

认证（libssh2\_userauth\_password()或libssh2\_userauth\_publickey\_fromfile()）

|

V

初始化sftp子系统（libssh2\_sftp\_init()）

|

V

进行各种sftp操作命令（如创建目录、下载等，函数参见本文附录B）

|

V

关闭sftp子系统（libssh2\_sftp\_shutdown()）

|

V

拆除ssh2会话（libssh2\_session\_disconnect()）

|

V

释放ssh2会话结构（libssh2\_session\_free()）

|

V

关闭标准socket套接字（closesocket()）

注意：上面列出的函数的具体用法请参阅网站在线文档：<http://www.libssh2.org/wiki/index.php/Documentation>

　　下面用一个具体的实例进一步说明如何开发一个ｓｓｈ２世纪功能，它实现一个简单的功能：在服务器上创建一个目录，列出另外一个目录下的文件。本实例主要展示SSH2协议的程序框架。

　　1、在Visual Studio 2008开发环境中新建一个工程： File-->New-->Project...-->展开Visual C++ --> 选择Win32 Console Application，Name处输入testsftp，Location处出入C:/projects，Solution Name处输入testftp，勾选Create directory for solution，最后点击OK进入下一步再点击Next。这一页只点选Console application，其他项都不要勾选或点选。最后点击Finish按钮完成新工程的创建。

　　2、在左边的工程浏览器窗口中展开Source Files，双击“testsftp.cpp”文件打开，里面只有寥寥数行：

// newsftp.cpp : Defines the entry point for the console application.  
//

#include "stdafx.h"

int \_tmain(int argc, \_TCHAR\* argv[])  
{  
 return 0;  
}

　　3、在屏幕左边的Solution Explorer中右击工程testsftp-->Properties-->Configuration Properties:

-->C/C++ --> General -->选择Additional Include Dirextories-->加C:/libssh2/include

         --> Code Generation -->Runtime Library -->选择Multi-thread(/MT)

-->Linker

     --> General -->选择Additional Library Directories --> 加C:/libssh2/lib

     --> Input -->选择Additional Dependencies --> 加libssh2.lib

     --> Command Line -->在Additional options中加 Ws2\_32.lib

最后点击“OK"确定。

　　4、开始编译。按F7或点击菜单Build-->Build Solution。

**附录A：我的testsftp.cpp文件内容**

// testsftp.cpp : Defines the entry point for the console application.  
//

#include "stdafx.h"

#include <libssh2\_config.h>  
#include <io.h>  
#include <libssh2.h>  
#include <libssh2\_sftp.h>

#ifdef HAVE\_WINSOCK2\_H  
# include <winsock2.h>  
#endif  
#ifdef HAVE\_NETINET\_IN\_H  
# include <netinet/in.h>  
#endif  
#ifdef HAVE\_SYS\_SOCKET\_H  
# include <sys/socket.h>  
#endif  
# ifdef HAVE\_UNISTD\_H  
#include <unistd.h>  
#endif  
#ifdef HAVE\_ARPA\_INET\_H  
# include <arpa/inet.h>  
#endif  
#ifdef HAVE\_SYS\_TIME\_H  
# include <sys/time.h>  
#endif

#include <sys/types.h>  
#include <fcntl.h>  
#include <errno.h>  
#include <stdio.h>  
#include <ctype.h>

int \_tmain(int argc, char\* argv[])  
{  
 int sock, i, auth\_pw = 1;  
 struct sockaddr\_in sin;  
 const char \*fingerprint;  
 LIBSSH2\_SESSION \*session;  
 int rc;  
 LIBSSH2\_SFTP \*sftp\_session;  
 LIBSSH2\_SFTP\_HANDLE \*sftp\_handle;

#ifdef WIN32  
 WSADATA wsadata;  
 WSAStartup(MAKEWORD(2,0), &wsadata);  
#endif

 /\*  
 \* The application code is responsible for creating the socket  
 \* and establishing the connection  
 \*/  
 if((sock = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP))==INVALID\_SOCKET){  
  fprintf(stderr,"failed to create a socket!/n");  
  return -1;  
 }  
 sin.sin\_family = AF\_INET;  
 sin.sin\_port = htons(22);  
 if((sin.sin\_addr.s\_addr = inet\_addr("192.168.104.105"))==INADDR\_NONE){  
  fprintf(stderr,"The address is invalid!/n");  
  return -1;  
 }  
 if(connect(sock, (struct sockaddr\*)(&sin),sizeof(struct sockaddr\_in))!= 0){  
  fprintf(stderr, "failed to connect!/n");  
  return -1;  
 }

 /\* Create a session instance\*/  
 if(!(session = libssh2\_session\_init())){  
  fprintf(stderr,"Init SSH session failed!/n");  
  goto CLOSESOCKET;  
 }

 /\* ... start it up. This will trade welcome banners, exchange keys,  
 \* and setup crypto, compression, and MAC layers  
 \*/  
 if((rc = libssh2\_session\_startup(session, sock))){  
  fprintf(stderr, "Failure establishing SSH session: %d/n", rc);  
  libssh2\_session\_free(session);  
  goto CLOSESOCKET;  
 }

 /\* At this point we havn't yet authenticated.  The first thing to do  
 \* is check the hostkey's fingerprint against our known hosts Your app  
 \* may have it hard coded, may go to a file, may present it to the  
 \* user, that's your call  
 \*/  
 if((fingerprint = libssh2\_hostkey\_hash(session, LIBSSH2\_HOSTKEY\_HASH\_MD5))){  
  printf("Fingerprint: ");  
  for(i = 0; i < 16; i++) {  
   printf("%02X ", (unsigned char)fingerprint[i]);  
  }  
  printf("/n");  
 }

 if (auth\_pw) {  
  /\* We could authenticate via password \*/  
  if ((libssh2\_userauth\_password(session, "sftpuser", "abc123"))) {  
   printf("Authentication by password failed./n");  
   goto SHUTDOWN;  
  }  
 } else {  
  /\* Or by public key \*/  
  if (libssh2\_userauth\_publickey\_fromfile(session, "sftpuser","/home/username/.ssh/id\_rsa.pub","/home/username/.ssh/id\_rsa","abc123")) {  
    printf("/tAuthentication by public key failed/n");  
    goto SHUTDOWN;  
  }  
 }

 fprintf(stderr, "libssh2\_sftp\_init()!/n");  
 if(!(sftp\_session = libssh2\_sftp\_init(session))){  
  fprintf(stderr, "Unable to init SFTP session/n");  
  goto SHUTDOWN;  
 }

 /\* Since we have not set non-blocking, tell libssh2 we are blocking \*/  
 libssh2\_session\_set\_blocking(session, 1);

 fprintf(stderr, "libssh2\_sftp\_opendir()!/n");

 //建目录  
 if(libssh2\_sftp\_mkdir(sftp\_session, "sftpdir/cba",  
                            LIBSSH2\_SFTP\_S\_IRWXU|  
                            LIBSSH2\_SFTP\_S\_IRGRP|LIBSSH2\_SFTP\_S\_IXGRP|  
       LIBSSH2\_SFTP\_S\_IROTH|LIBSSH2\_SFTP\_S\_IXOTH)==-1)  
  fprintf(stderr,"Create dir failed!/n");

 //浏览一个目录中的文件  
 if(!(sftp\_handle = libssh2\_sftp\_opendir(sftp\_session, "sftpdir"))){  
  fprintf(stderr, "Unable to open dir with SFTP/n");  
  goto SHUTDOWN;  
 }

 fprintf(stderr, "libssh2\_sftp\_opendir() is done, now receive listing!/n");  
 do {  
  char mem[512];  
  char longentry[512];  
  LIBSSH2\_SFTP\_ATTRIBUTES attrs;

  /\* loop until we fail \*/  
  rc = libssh2\_sftp\_readdir\_ex(sftp\_handle, mem, sizeof(mem), longentry, sizeof(longentry), &attrs);  
  if(rc > 0) {  
   /\* rc is the length of the file name in the mem  
   buffer \*/

   if (longentry[0] != '/0') {  
    printf("%s/n", longentry);  
   } else {  
    if(attrs.flags & LIBSSH2\_SFTP\_ATTR\_PERMISSIONS) {  
     /\* this should check what permissions it  
     is and print the output accordingly \*/  
     printf("--fix----- ");  
    }  
    else {  
     printf("---------- ");  
    }

    if(attrs.flags & LIBSSH2\_SFTP\_ATTR\_UIDGID) {  
     printf("%4ld %4ld ", attrs.uid, attrs.gid);  
    }  
    else {  
     printf("   -    - ");  
    }

    if(attrs.flags & LIBSSH2\_SFTP\_ATTR\_SIZE) {  
     /\* attrs.filesize is an uint64\_t according to  
     the docs but there is no really good and  
     portable 64bit type for C before C99, and  
     correspondingly there was no good printf()  
     option for it... \*/

     printf("%8lld ", attrs.filesize);  
    }

    printf("%s/n", mem);  
   }  
  }  
  else  
   break;

 } while (1);

 libssh2\_sftp\_closedir(sftp\_handle);  
 libssh2\_sftp\_shutdown(sftp\_session);

SHUTDOWN:

 libssh2\_session\_disconnect(session, "Normal Shutdown, Thank you for playing");  
 libssh2\_session\_free(session);

CLOSESOCKET:  
#ifdef WIN32  
 Sleep(1000);  
 closesocket(sock);  
#else  
 sleep(1);  
 close(sock);  
#endif  
 printf("all done/n");  
 return 0;  
}

**附录B：libssh2.lib中函数列表**

**1. Session API:**   
libssh2\_session\_init()   
libssh2\_session\_abstract()   
libssh2\_session\_callback\_set()   
libssh2\_banner\_set()   
libssh2\_session\_startup()   
libssh2\_session\_disconnect()   
libssh2\_session\_free()   
libssh2\_hostkey\_hash()   
libssh2\_session\_method\_pref()   
libssh2\_session\_methods()   
libssh2\_session\_last\_error()   
libssh2\_session\_flag()   
**2. Userauth API:**libssh2\_userauth\_list()   
libssh2\_userauth\_authenticated()   
libssh2\_userauth\_password()   
libssh2\_userauth\_publickey\_fromfile()   
libssh2\_userauth\_hostbased\_fromfile()   
libssh2\_userauth\_keyboard\_interactive()  
**3. Channel API:**Channel Creation and Setup   
libssh2\_channel\_open\_session()   
libssh2\_channel\_direct\_tcpip()   
libssh2\_channel\_forward\_listen()   
libssh2\_channel\_forward\_cancel()   
libssh2\_channel\_forward\_accept()   
libssh2\_channel\_setenv()   
libssh2\_channel\_request\_pty()   
libssh2\_channel\_process\_startup()   
libssh2\_channel\_x11\_req()   
libssh2\_scp\_recv()   
libssh2\_scp\_send()   
Channel Shutdown and Destruction   
libssh2\_channel\_send\_eof()   
libssh2\_channel\_eof()   
libssh2\_channel\_close()   
libssh2\_channel\_wait\_closed()   
libssh2\_channel\_get\_exit\_status()   
libssh2\_channel\_free()   
Input/Output   
libssh2\_channel\_set\_blocking()   
libssh2\_channel\_read()   
libssh2\_channel\_write()   
libssh2\_channel\_handle\_extended\_data()   
libssh2\_channel\_flush()   
libssh2\_poll\_channel\_read()   
libssh2\_poll()   
Windowing   
libssh2\_channel\_window\_read()   
libssh2\_channel\_receive\_window\_adjust()   
libssh2\_channel\_window\_write()   
**4. SFTP Subsystem:**   
Protocol startup and shutdown   
libssh2\_sftp\_init()   
libssh2\_sftp\_shutdown()   
libssh2\_sftp\_last\_error()   
File/Directory Access   
libssh2\_sftp\_open()   
libssh2\_sftp\_opendir()   
libssh2\_sftp\_read()   
libssh2\_sftp\_readdir()   
libssh2\_sftp\_write()   
libssh2\_sftp\_close()   
libssh2\_sftp\_closedir()   
libssh2\_sftp\_seek()   
libssh2\_sftp\_rewind()   
libssh2\_sftp\_tell()   
libssh2\_sftp\_fstat()   
libssh2\_sftp\_fsetstat()   
File/Directory Manipulation   
libssh2\_sftp\_rename()   
libssh2\_sftp\_unlink()   
libssh2\_sftp\_mkdir()   
libssh2\_sftp\_rmdir()   
libssh2\_sftp\_stat()   
libssh2\_sftp\_lstat()   
libssh2\_sftp\_setstat()   
libssh2\_sftp\_symlink()   
libssh2\_sftp\_readlink()   
libssh2\_sftp\_realpath()   
**5. Publickey Subsystem:**libssh2\_publickey\_init()   
libssh2\_publickey\_shutdown()   
libssh2\_publickey\_add()   
libssh2\_publickey\_remove()   
libssh2\_publickey\_list\_fetch()   
libssh2\_publickey\_list\_free()

参考文献：

1. Using libcurl with SSH support in Visual Studio 2008.pdf（点击[这里](http://www.moodisk.com/download_other_c.php) 下载）

2. <http://www.libssh2.org/wiki/index.php/Documentation>