Subject: SFTP or full SSH2 support for U++?
Posted by Oblivion on Sun, 10 Jan 2016 11:58:48 GMT
View Forum Message <> Reply to Message

Hello,

Some time ago when I uploaded my FTPS implementation to the bazaar section, Daniel (Unodgs) asked me If I have any plans to implement an SFTP class.
And I'd said yes.

Now I am here to deliver my promise. :)

But before I upload the package to the bazaar section, I have to make a choice and would like to hear your opinion on the subject.

First of all, I decided not to implement the SSH2 protocol from the scratch. I could've, but it would be a) tedious and b) wasting my time.
So I decided searching for alternatives.
The best alternative that I could have come up with was wrapping an existing, multiplatform (Linux and Windows, at least) library.
I found out that libssh2 was the way to go. It was plain C, easy to wrap up and had both blocking and non-blocking modes of operation available.

I finally implemented an SFTP class for U++ using libssh2, which has the below features:

1) Allows blocking and non-blocking operations. Namely, it can work in both synchronous and asynchronous modes.
I use HttpRequest and NetworkProxy class' asynchronous design here too, since that design proved very effective.

NetworkProxy class.

3) Takes advantage of U++ streams in file upload and download operations, and uses gates for progress tracking when needed.

4) Supports both MD5 and SHA 1 methods.

5) Has SFtp::DirEntry class for easy parsing of directory entries.

In short, currently we have a sync/async SFTP wrapper that can authenticate using "public key" method and/or Username/password combination, and perform Open/Close/Read/Write/List/Rename/Delete operations on remote file system entries. Also its API is very similar to my FTPS class, and its design derives from NetworkProxy and HttpRequest classes and follows the U++ coding style.

Now, the class is abstracted nicely so that I can either let it be a standaloune, robust SFtp class, or take it one step further and wrap up the whole functionality of libssh2.
Latter means adding SSHSession, SSHChannel, Scp classes to the package too.
What is your opinion?

(For example I can also easily add the SCP protocol to the package. SCP is known to perform better in some type file read/wtire operations.)

Also, since the libssh2 uses its own allocators (alloc, realloc, dealloc) but also gives the user the choice to implement his/her own, I'll definietly need help here.
I'm not experienced with U++ allocators.

Here is an actual, barebone SFtp example, demonstrating directory listing and file download, using a public SFtp test server:

```
#include <Core/Core.h>
#include <SFTP/SFTP.h>

using namespace Upp;

static const char* CRLF = "\r\n";

// This callback allows getting directory entries one by one.
// It is optional.
bool ListDirectory(SFtp::DirEntry& e)
{


 String ls = e.GetEntry();
 if(!ls.IsEmpty())
  // When available, traditiona UNIX style directory listing can be used.
  Cout() << e.GetEntry() << CRLF;
 else {
  Cout()  << e.GetName()          << CRLF
    << e.GetUserID()  << CRLF
    << e.GetGroupID()           << CRLF
    << e.GetSize()    << CRLF
```

```
    << e.GetLastAccessed()        << CRLF
    << e.GetLastModified()        << CRLF
    << "Permissions:"  << CRLF
    << "R: " << e.IsReadable()  << ", "
    << "W: " << e.IsWritable()  << ", "
    << "X: " << e.IsExecutable()    << CRLF;
 }
 return false;
}

CONSOLE_APP_MAIN
{
 // This example demonstrates directory listing (ls) and file download operations,
 // using a public SFTP test server.
 // Note that this example uses blocking mode. It is synchronous.
 // There is also a non-blocking, asynchhronous mode.

 FileOut wallpaper("/home/genericuser/Wallpapers/SFTP-download-test-wallpaper.jpg");
 SFtp::DirList dirlist; // SFtp::DirList is a vector containing SFtp:DirEnty elements.

 // Enable logging.
 SFtp::Trace();

 SFtp sftp;
 if(sftp.User("demo-user", "demo-user").Connect("demo.wftpserver.com", 2222)) {
  // Get server banner.
  Cout() << sftp.GetBanner() << CRLF;
  if(sftp.OpenDir("/download") && sftp.ListDir(dirlist, callback(ListDirectory))) {
   if(sftp.Open("/download/F11_wallpaper_06_1600x1200.jpg", SFtp::READ)) {
    if(sftp.Get(wallpaper)) {
            Cout() << "File successfully downloaded.\r\n";
                        return;
                    }

   }
            }
 }
 Cout() << "Failed.\r\n" << "Reason: " << sftp.GetErrorDesc() << CRLF;

 // Instead we could have used the SFtpGet() convenience function.
 // int rc = SFtpGet(wallpaper, "demo-user", "demo-user", "demo-wftpserver.com", 2222,
"/download/F11_wallpaper_06_1600x1200.jpg");

}
```

If Sftp is sufficient, I'll upload it to the bazaar next weekend. If not, I'l still add it to the bazaar next

weekend but implement the missing classes incrementally over time (next class to implemnt will be SCP). :)

Regards,

Oblivion