Written by : Barak Weichselbaum, all rights reserved 2000-2009

Site: http://www.komodia.com/
Email : barak@komodia.com

Komodia's TCP/IP library V5.1 encryption usage
By Barak Weichselbaum

Introduction

This manual covers how to use the new encryption classes of the TCP/IP library V5.1 (http://www.komodia.com/index.php?page=tcpip51.htm)

SSL Class – CTCPSocketAsyncSSL

This class uses CTCPSocketAsync class in conjunction with the OpenSSL library.
With the library there's a compiled version of OpenSSL and the needed .h files, if you want to compile a newer version of OpenSSL, you can read my blog post about the process of compiling OpenSSL (http://barakw.blogspot.com/2008/11/ssl-over-tcpip.html#links)

When using CTCPSocketAsyncSSL it's important to have OpenSSL .DLL files in the .DLL load path, the .DLL are supplied inside the OpenSSL\Bin directory.

The sample that is included in the library called "TestSSL", opens a server and a client SSL and sends data back and forth.

Working sequence

First thing to do in the application after calling the library socket initialization is to initialize OpenSSL by calling:

```
CTestSocket::InitializeSSL();
```

Server socket

Server socket must have a pair of keys, a public key, and a private key, to create a self signed key you can visit http://panoptic.com/wiki/aolserver/How_to_generate_self-signed_SSL_certificates for more information on the matter.

To load the keys (which is a must, a server socket can't work without them) you need to call the method LoadCertificatesForServer, specifying the path to both keys.

From here programming the socket continues like a normal CTCPSocketAsync.

Client socket

You may want to verify that when connecting to a SSL server that the certificate is valid, therefore there's a need to call LoadCertificateStore to load a certificate store with all the root certificate information, to get the file you can visit http://curl.haxx.se/docs/caextract.html for more information, this step is optional, without this file, all certificates will be valid.

Overridables

There are new functions to override that can give more information during the SSL session (this is optional):

OnSSLError – Will be called in case on an SSL error, return value of false will terminate the session.
OnSSLEvent – SSL specific event such as: Bad certificate (only for client, and only if the store was loaded), and SSL handshake complete

Cleanup

Before cleaning up the sockets, a cleanup routine must be called to free OpenSSL library resources:

```
CTCPSocketAsyncSSL::UninitializeSSL();
```

Disabling SSL

Before calling Create method it's possible to turn off SSL, I found this option usefull method in several applications. To disable SSL call the method DisableSSL

BlowFish class – CTCPSocketAsyncBlowFish

This class uses the BlowFish algorithm (http://en.wikipedia.org/wiki/Blowfish_(cipher)) and the CTCPSocketAsyncMsg class which is a messaged base socket class that is based upon CTCPSocketAsync.

Working sequence

The only thing to do here is set the shared key, which is done by calling SetKey.
An optional security measure would be to change the initial two random seeds of the sequence in the code, because this is shared with all the library users, it can be found on line 49 and 50 of file TCPSocketAsyncBlowFish.cpp, numbers must match for both chains.

Sending and receiving data

Unlike CTCPSocketAsync, sending and receiving data is done using two methods ReceiveMsg and SendMsg, which are methods to send data organized in messages which means that if you sent 1000 bytes, you will receive the same 1000 bytes in one message, normal TCP can break the message down since TCP is a stream based protocol.