**How to build OpenSSL from source**

Updated steps 29/6/2016:  
  
Building 32-bit OpenSSL-1.0.2h on Windows 8.1 (see also INSTALL.W32)

o Source downloaded from https://www.openssl.org/source/openssl-1.0.2h.tar.gz

o Launch Windows SDK 7.1 commnd prompt (Windows SDK installed from GRMSDKX\_EN\_DVD.iso).

o setenv /x86

o perl Configure VC-WIN32 no-asm --prefix=c:/some/openssl/dest/dir

o ms\do\_ms

o nmake -f ms\ntdll.mak

o Optional:

nmake -f ms\ntdll.mak test

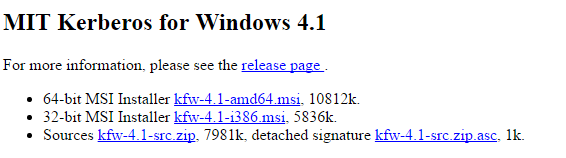
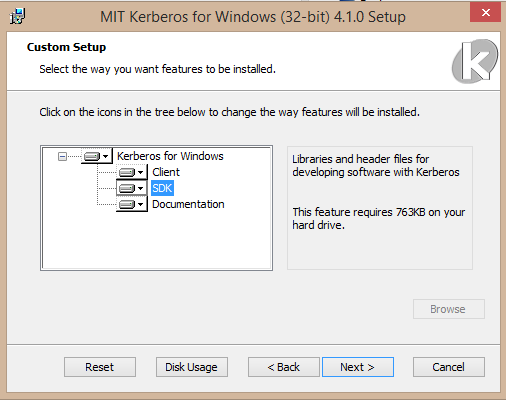
(All the tests passed for me!)

o nmake -f ms\ntdll.mak install

(Looks possibly helpful: <http://p-nand-q.com/programming/windows/building_openssl_with_visual_studio_2013.html>)

---  
  
**Building OpenSSL with Kerberos support  
  
What is Kerberos?**From <https://web.mit.edu/kerberos/#what_is>:  
*“Kerberos is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol is available from the*[*Massachusetts Institute of Technology*](http://web.mit.edu/)*. Kerberos was created by MIT. The Kerberos protocol uses strong cryptography so that a client can prove its identity to a server (and vice versa) across an insecure network connection. After a client and server has used Kerberos to prove their identity, they can also encrypt all of their communications to assure privacy and data integrity as they go about their business.”*

**How does Kerberos work?**From <http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.6/doc/install.html>:  
*“Kerberos V5 is based on the Kerberos authentication system developed at MIT. Under Kerberos, a client (generally either a user or a service) sends a request for a ticket to the Key Distribution Center (KDC). The KDC creates a ticket-granting ticket (TGT) for the client, encrypts it using the client's password as the key, and sends the encrypted TGT back to the client. The client then attempts to decrypt the TGT, using its password. If the client successfully decrypts the TGT (i.e., if the client gave the correct password), it keeps the decrypted TGT, which indicates proof of the client's identity.The TGT, which expires at a specified time, permits the client to obtain additional tickets, which give permission for specific services. The requesting and granting of these additional tickets is user-transparent.”*

**Kerberos Installation on Windows**Page<http://web.mit.edu/kerberos/dist/>includes link “**Kerberos for Windows Release 4.1 - current release**” on page<http://web.mit.edu/kerberos/dist/#kfw-4.1>**:** **Installing kfw-4.1-i386.msi**Note: I selected the “Custom” installation option, which enabled me to select SDK:  


The files are installed here…  
**Header files:**C:\Program Files (x86)\MIT\Kerberos\include  
**Lib files:**C:\Program Files (x86)\MIT\Kerberos\lib\i386  
**Note:** I had real problems because of the spaces in these paths, so I created a copy here:  
C:\MIT\Kerberos\include  
  
**Building OpenSSL-1\_0\_2h with Kerberos support**o Launch VC++ shortcut from desktop (“C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\vcvarsall.bat"" x86”).  
operl Configure VC-WIN32 no-asm --with-krb5-dir=C:/MIT/Kerberos --with-krb5-flavor=MIT --prefix=C:/OpenSSL-Build  
(see here for options: https://github.com/ChatSecure/OpenSSL/blob/master/Configure)   
o ms\do\_ms

o nmake -f ms\ntdll.mak  
o nmake -f ms\ntdll.mak install   
This will populate directory C:/OpenSSL-Build.

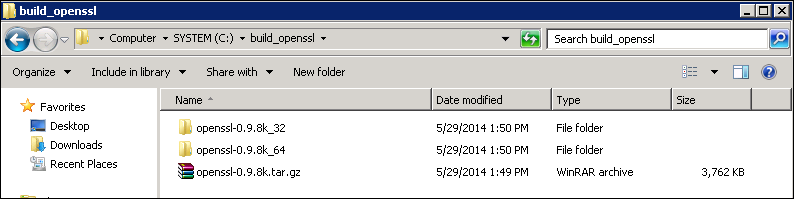
(All command-line options: https://github.com/ChatSecure/OpenSSL/blob/master/Configure).  
  
These are the ciphers that I should now re-test with, to see if the OpenSSL client no longer complains:

|  |
| --- |
| KRB5-RC4-SHA(TLS\_KRB5\_WITH\_RC4\_128\_SHA) |
| KRB5-RC4-MD5(TLS\_KRB5\_WITH\_RC4\_128\_MD5) |

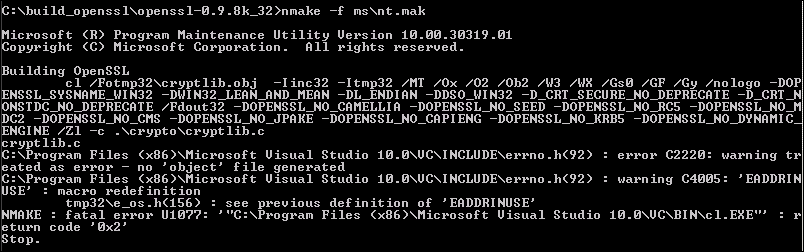
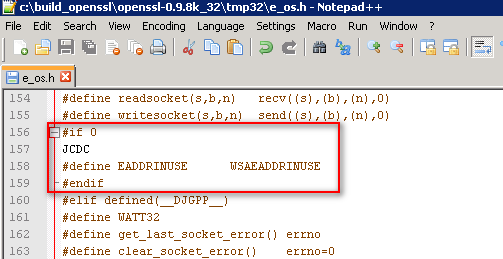
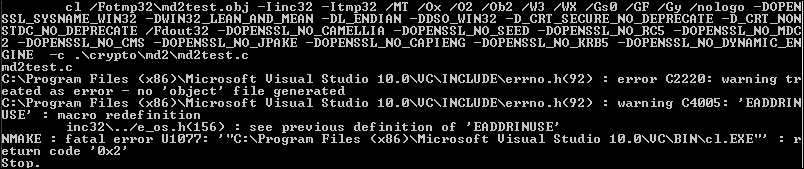
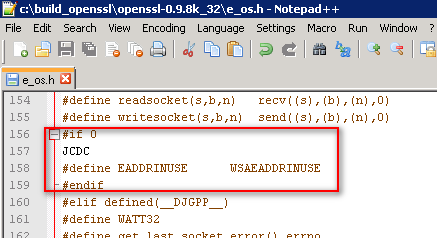
|  |
| --- |
| KRB5-DES-CBC3-SHA(TLS\_KRB5\_WITH\_3DES\_EDE\_CBC\_SHA) |
| KRB5-DES-CBC3-MD5(TLS\_KRB5\_WITH\_3DES\_EDE\_CBC\_MD5) |

*JeremyC 26/12/2016.*  
---  
  
Original steps (before I knew about Windows SDK compiler)s:  
  
1. Download the free “Visual C++ 2010 Express” from <http://www.visualstudio.com/en-us/downloads/download-visual-studio-vs#DownloadFamilies_4>

2. Download OpenSSL sources from <http://www.openssl.org/source/>. I will be building <http://www.openssl.org/source/openssl-0.9.8k.tar.gz>

3. Extract the source twice. One for a 32-bit build, and the other for a 64-bit build:  


4. Build the 32-bit static libraries:  
o Open the Visual C++ command prompt.  
o cd to the 32-bit folder.  
o Run the following:  
  
perl Configure VC-WIN32 --prefix=C:\\Build-OpenSSL-VC-32  
ms\do\_ms  
nmake -f ms\nt.mak   
nmake -f ms\nt.mak install  
  
**(Note the double back-slash “\\”. This is the location where the built files will be installed by the last step).**

Doing this, I got the following error:  
  
To fix this, I added the following and this progressed the build a little further:  
  
I could not find an answer for this on the Internet, but the cause is suspected to be due to the fact that the age of these sources is much greater than the version of Visual C++ being used to compile them.  
  
A lot later, I then got the EADDRINUSE error again:  
  
To fix this, I again made the same fix as before:  
  
  
And that’s it. The build then completed successfully. Total output directory size approx 2.5MB.  
  
References:  
http://developer.covenanteyes.com/building-openssl-for-visual-studio/  
  
JeremyC 29/5/2014  
**END**