عنوان: **نحوه‌ی استفاده از کتابخانه‌ی OpenSSL در ویندوز**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۵:۲۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: OpenSSL

سؤالی شده
به این مضمون : "یه الگوریتم دارم که بر طبق اون باید اعداد تصادفی خیلی بزرگ تولید کنم، اونها رو جمع و ضرب کنم. اینکه چطوری باید از dll یا lib استفاده کنم رو بلد نیستم. از VS2008 استفاده میکنم..."

سؤال در مورد زبان CPP است. کتابخانه‌ی استاندارد انجام اینگونه عملیات برای زبان‌های C و CPP ، کتابخانه‌ی OpenSSL است. البته شاید الان 100 کتابخانه دیگر را هم لیست کنید، اما کسانی که با مباحث رمزنگاری اطلاعات مدتی کار کرده باشند، بعید است سر و کارشان به این کتابخانه نیفتاده باشد و یک استاندارد در این زمینه به شمار می‌رود؛ همچنین به دلیل سورس باز بودن در اکثر سکوهای کاری موجود نیز قابل استفاده است. بنابراین فراگیری نحوه‌ی کار کردن با آن یک مزیت به شمار می‌رود. قسمتی از این کتابخانه‌ی معظم مرتبط است به کار با اعداد بزرگ. این مورد را هم جهت استفاده در الگوریتم RSA نیاز دارد.
برای استفاده از آن در ویندوز ابتدا باید OpenSSL را کامپایل کنید . کار پر دردسری است. به همین جهت یک سایت فقط به این موضوع اختصاص یافته و هربار آخرین نسخه‌ی OpenSSL را برای ویندوز کامپایل می‌کند و در اختیار علاقمندان قرار می‌دهد : +
در حال حاضر یا باید Win32 OpenSSL v1.0.0a و یا Win64 OpenSSL v1.0.0a را دریافت کنید (برنامه‌ی شما اگر 64 بیتی کامپایل شود، dll های 32 بیتی را نمی‌تواند بارگذاری کند و برعکس).

روش استفاده از کتابخانه‌ی OpenSSL در ویژوال CPP :

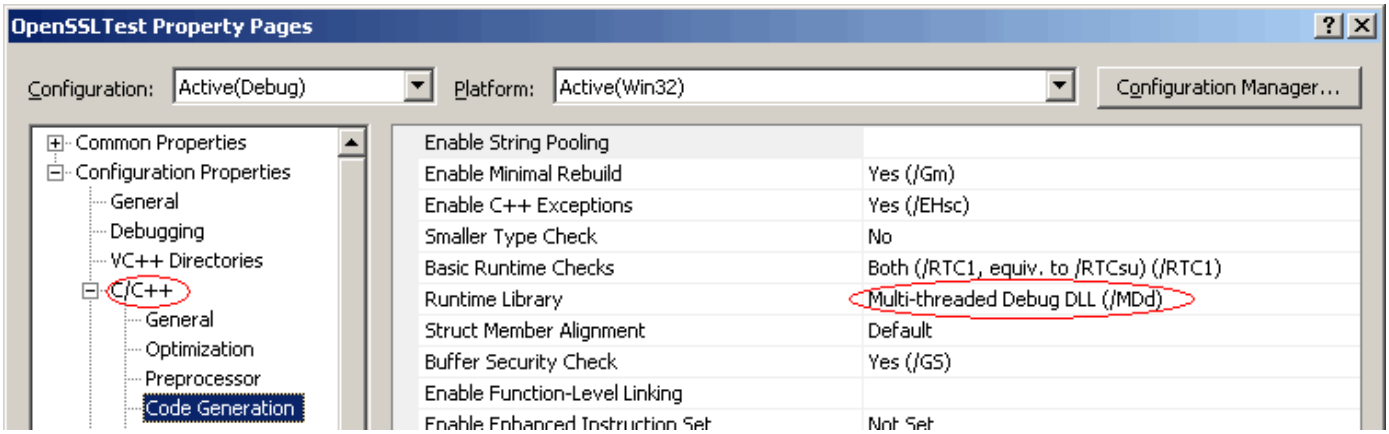
الف) ابتدا فایل‌های کامپایل شده‌ی فوق را دریافت و نصب کنید. اکنون برای مثال یک پوشه‌ی OpenSSL-Win32 در کامپیوتر شما با محتویات این کتابخانه باید ایجاد شده باشد(اگر نسخه‌ی 32 بیتی را دریافت کرده‌اید).
سپس به پوشه‌ی OpenSSL-Win32\lib\VC آن مراجعه کنید. در اینجا فایل‌های کتابخانه‌ای جهت استفاده در ویژوال CPP قرار گرفته‌اند. اگر از محتویات پوشه OpenSSL-Win32\lib\VC\ static استفاده کنید، نیازی به توزیع فایل‌های DLL این کتابخانه نخواهید داشت و اگر از کتابخانه‌های OpenSSL-Win32\lib\VC استفاده کنید، فایل‌های dll را نیز حتما باید به همراه برنامه‌ی خود توزیع نمائید.
سه نوع فایل در آن وجود دارند. ختم شده به MT ، MD و MDd که معانی آن‌ها در مورد چند ریسمانی بودن یا خیر است (برگرفته شده از فایل faq.txt دریافتی):

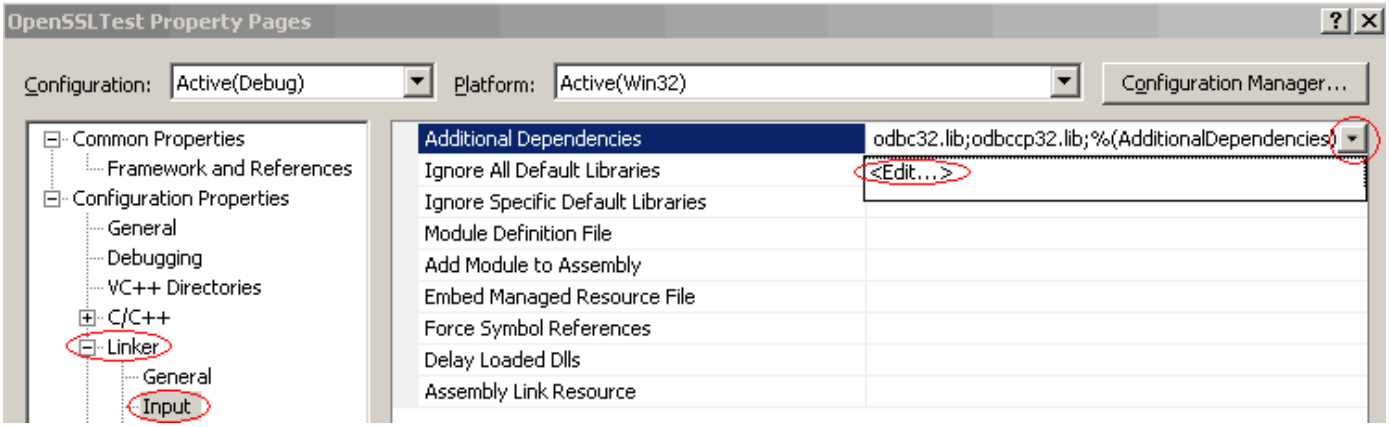
```
Single Threaded           /ML        -  MS VC++ often defaults to  this for the release  version of a
new project.
   Debug Single Threaded     /MLd  -  MS VC++ often defaults to  this for the debug version of a new
project.
   Multithreaded             /MT
   Debug Multithreaded       /MTd
   Multithreaded DLL         /MD        -  OpenSSL defaults to this.
   Debug Multithreaded DLL   /MDd
```

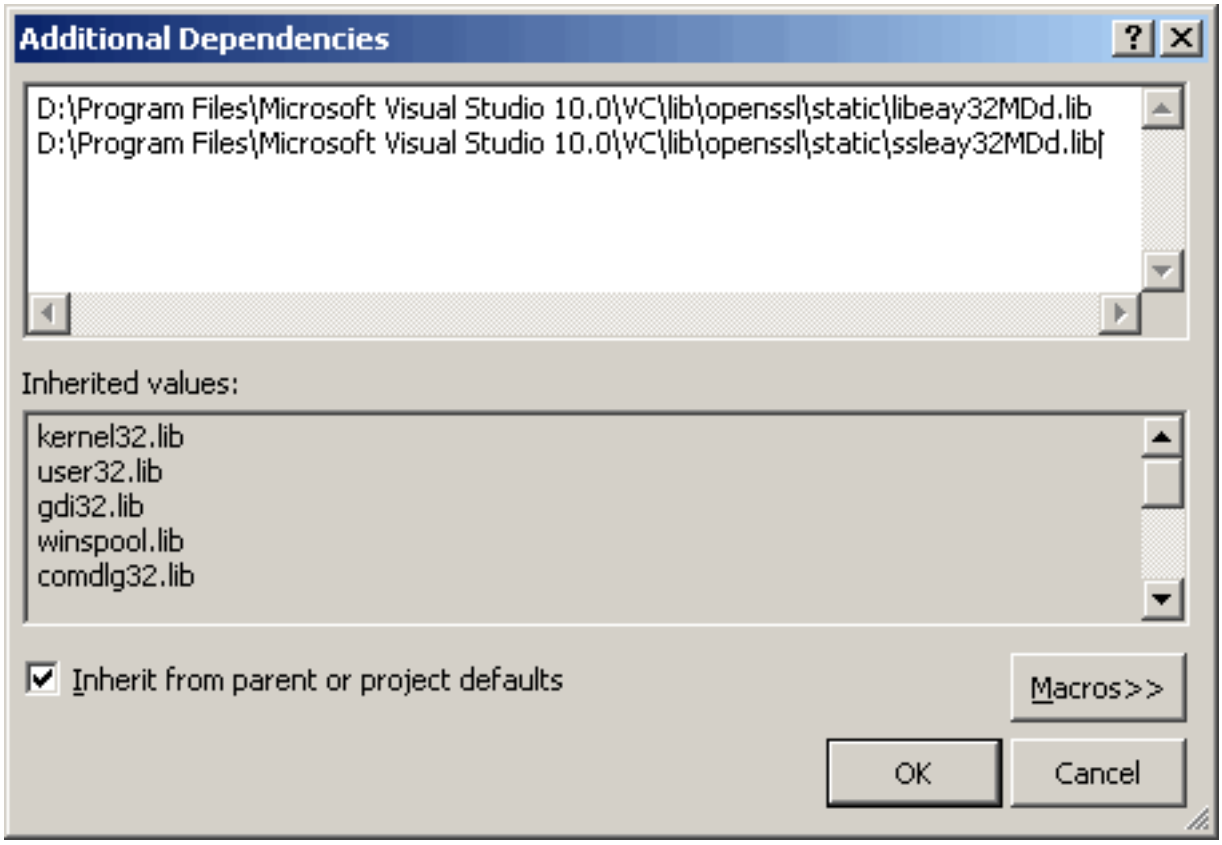
ب) جهت سهولت کار، پوشه‌ی OpenSSL قرار گرفته در مسیر OpenSSL-Win32\include را در آدرس زیر کپی نمائید:
C:\Program Files\Microsoft Visual Studio 10.0\VC\include
به این صورت حین استفاده از این کتابخانه نیازی به مشخص سازی محل قرارگیری فایل‌های include نخواهد بود.

ج) اکنون یک پروژه‌ی جدید Visual C++\Win32\Win32 console application را در VS.NET آغاز کنید؛ برای مثال به نام OpenSSLTest .

د) سپس به منوی پروژه، گزینه‌ی خواص پروژه مراجعه کرده و مطابق تصاویر زیر، این فایل‌های کتابخانه‌ای را معرفی کنید (انتخاب MD یا MT یا MDd بر اساس runtime library انتخاب شده است که در تصاویر مشخص گردیده):


OpenSSLTest Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Configuration Properties
General
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generation
Enable String Pooling
Enable Minimal Rebuild
Yes (/Gm)
Enable C++ Exceptions
Yes (/EHsc)
Smaller Type Check
No
Basic Runtime Checks
Both (/RTC1, equiv. to /RTCsu) (/RTC1)
Runtime Library
Multi-threaded Debug DLL (/MDd)
Struct Member Alignment
Default
Buffer Security Check
Yes (/GS)
Enable Function-Level Linking
Enable Enhanced Instruction Set
Not Set


OpenSSLTest Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Common Properties
Framework and References
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
Additional Dependencies
odbc32.lib;odbccp32.lib;%(AdditionalDependencies)
Ignore All Default Libraries
<Edit...>
Ignore Specific Default Libraries
Module Definition File
Add Module to Assembly
Embed Managed Resource File
Force Symbol References
Delay Loaded Dlls
Assembly Link Resource



ه) اکنون یک مثال ساده در مورد ضرب دو عدد بزرگ به صورت زیر می‌تواند باشد:

```
#include "stdafx.h"
#include <openssl/bn.h>
#include <string.h>


void RotateBytes(unsigned char *in, int n)
{
 unsigned char *e=in+n-1;
 do {
  unsigned char temp=*in;
  *in++=*e;
  *e-- =temp;
 } while(in<e);
}

int _tmain(int argc, _TCHAR* argv[])
{
 //دو عدد بزرگ جهت آزمایش
 unsigned char testP[] =
{0xD1,0x31,0x85,0x4D,0x00,0xD6,0x31,0x97,0x3A,0xFC,0xD2,0x27,0x02,0xEF,0xC2,0xA7};
   unsigned char testA[] =
{0xC7,0x1B,0x25,0x72,0x03,0xCB,0x72,0x03,0xCF,0x23,0x27,0x2D,0x00,0xD6,0x31,0x98};

 //تبدیل آرایه‌های فوق به فرمت اعداد بزرگ
 BIGNUM *a = BN_new();
 //it should be in "big-endian" form
 RotateBytes(testA, 16);
 BN_bin2bn(testA, 16, a);

 BIGNUM *p = BN_new();
 //it should be in "big-endian" form
 RotateBytes(testP, 16);
 BN_bin2bn(testP, 16, p);


 //ضرب این دو عدد در هم
 BIGNUM *result = BN_new();
```

```
 BN_CTX *ctx = BN_CTX_new();

 BN_mul(result, a, p, ctx);

 //نمایش نتیجه
 //حاصل از چند بایت تشکیل شده؟
 int num = BN_num_bytes(result);
 if(num>0)
 {
  unsigned char *tmpdata;
  if((tmpdata=(unsigned char *)malloc(num)))
   memset(tmpdata, 0, num);

  //تبدیل عدد با فرمت اعداد بزرگ به آرایه‌ای از بایت‌ها
  BN_bn2bin(result, tmpdata);
  RotateBytes(tmpdata, num);

  for(int i=0; i<num; i++)
  {
   if(i%16==0) printf("\n");
   printf("%02X ",tmpdata[i]);
  }

  if(tmpdata) free(tmpdata);
 }


 //آزاد سازی منابع
 BN_free(a);
 BN_free(p);
 BN_CTX_free(ctx);

 return 0;
}
```

در مورد شرح توابع کتابخانه OpenSSL به اینجا مراجع کنید : +

علت استفاده از تابع RotateBytes ، تغییر endian ورودی است.

نظرات خوانندگان

نویسنده: SpEEdY
تاریخ: ۱۳۸۹/۰۶/۱۰ ۰۰:۴۲:۱۱

خیلی ممنون.

نویسنده: علی اقدم
تاریخ: ۱۳۸۹/۰۶/۱۱ ۱۱:۰۶:۴۱

سلام
آقای نصیری یه سوال دارم خارج از این مبحث
تو انتشار بعضی کامپوننت ها و نرم افزار ها اعلام میشه که نسخه RTW است ،منظور از RTW چیه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۱۱ ۲۰:۰۴:۱۲

سلام،
منظور Release to Web است. یک RTM هم داریم که Release to Manufacturing است. اطلاعات بیشتر:
http://blogs.msdn.com/b/canthe/archive/2005/06/24/432468.aspx

نویسنده: Ali Safaie
تاریخ: ۱۳۸۹/۰۶/۱۲ ۰۹:۱۲:۱۵

در کتاب های طراحی الگوریتم ، برای جمع و ضرب اعداد بسیار بزرگ روش بسیار کارآمدی معرفی شده است که می توان از آن هم استفاده نمود.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۱۲ ۱۰:۵۷:۱۹

بله؛ البته اگر علاقمند هستید که چرخ را مجددا اختراع کنید همیشه وقت هست.
شبیه به پروژه OpenSSL را شروع کنید.
بعد باید بهینه سازی شود.
هزاران تست برای آن نوشته شود تا صحت عملکرد آن اثبات شود.
مشکلات امنیتی آن برطرف شود. (مثل همین کتابخانه OpenSSL که خیلی از به روز رسانی‌های آن فقط امنیتی بوده)
نگهداری شود و خیلی مسایل دیگر.
بعد تازه می‌شود OpenSSL .

نویسنده: ali
تاریخ: ۱۳۸۹/۰۶/۱۸ ۰۲:۲۶:۳۶

سلام ضمن تشکر از مقالات خیلی خوب شما که واقعا ارزنده هستند و از اینکه برای یاد دادن به دیگران وقت می گذارید.
با نظرتون در رابطه با ابدا مجدد چرخ تقریبا موافقم اما یه برنامه نویس باید تکنیک های متفاوتی را دست داشته باشه تا بتونه هر زمان بنا به شرایط از تکنیک مطلوبتر استفاده کنه
خیلی خوبه که از پروژه های معروف و تست شده استفاده کنیم چون خیالمون تقریبا راحتتره اما نه همیشه
من 2 پیاده سازی از الگوریتم ضرب اعداد بزرگ دارم که اتفاقا کار ساده ایی هست برای تکمیل شدن این مقاله اون ها رو برای شما در اسرع وقت upload میکنم