《使用CryptoPP库的测试程序cryptest.exe》

作者：韩露露、杨波

2019年10月1日

说明

本电子文档为《深入浅出CryptoPP密码学库》的附加电子文档，它最初被存放于GitHub上。任何人都可以复制、传播、使用本示例代码。

⇓

简介

《深入浅出CryptoPP密码学库》内容简介：

本书向读者介绍密码学库CryptoPP（或Crypto++）的使用方法和设计原理。CryptoPP是一个用C++语言编写的、开源的、免费的密码程序库，它最初由Wei Dai开发，现由开源社区维护。CryptoPP库广泛应用于学术界、开源项目、非商业项目以及商业项目，它几乎包括了目前已经公开的所有密码算法，支持当前主流的多种系统平台，并且具有良好的设计结构和较高的执行效率。

全书共15章，主要内容包括随机数发生器、Hash函数、流密码、分组密码、消息认证码、密钥派生和基于口令的密码、公钥加密系统、数字签名、密钥协商等，本书涵盖C++程序设计、设计模式、数论和密码学等知识。

本书最大的特点就是以应用为导向、以解决实际工程问题为目标，理论结合实践，将抽象的密码学变成保障信息安全的实际工具。

本书可以作为密码学、网络安全等专业在校学生的上机实验教材，也可以作为信息安全产品开发者、科研人员、密码算法实现者的参考手册。

⇓

资源

更多示例代码：https://github.com/locomotive-crypto
Crypto++网站：https://www.cryptopp.com/
Crypto++库GitHub地址：https://github.com/weidai11/cryptopp
Crypto++库SourceForge地址：https://sourceforge.net/projects/cryptopp/
Crypto++库Google论坛：
⇒公告通知地址：https://groups.google.com/forum/#!forum/cryptopp-announce
⇒用户群组地址：https://groups.google.com/forum/#!forum/cryptopp-users

目录

1 从源代码中生成cryptest.exe程序

本文档主要介绍从程序库的源代码中生成和使用cryptest.exe程序的方法。

1.1 在Windows系统上生成cryptest.exe程序

在书籍的第2章中详细介绍了通过静态链接库安装CryptoPP库的过程。这个静态链接库是通过cryptlib子项目生成的，如图1所示。类似地，我们可以通过编译和生成cryptest子项目得到cryptest.exe测试程序，如图1所示。

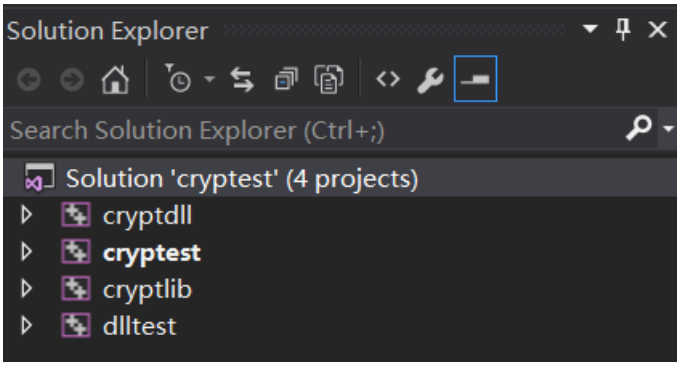

图1 cryptest解决方案下的4个子项目

1.2 在Linux系统上生成cryptest.exe程序

在命令行终端通过make命令编译CryptoPP库源代码时会自动生成cryptest.exe测试程序。当使用make install命令安装程序库时，cryptest.exe测试程序会自动被复制到/usr/local/bin/目录下。也可以使用whereis命令搜索cryptest.exe测试程序所在的位置。

```
$ whereis cryptest.exe
cryptest: /usr/local/bin/cryptest.exe
```

2 使用cryptest.exe程序

当生成并安装cryptest.exe测试程序后，我们就可以使用该程序执行一些简单的测试。若cryptest.exe测试程序所在的目录已被添加至系统环境，则可以在任何目录下运行它。例如，在用户主目录/home/locomotive下以V参数运行cryptest.exe程序：

```
$ cryptest.exe V
7.0.0
```

需要注意的是，由于某些测试参数（选项）需要使用程序库源代码包中TestData文件夹和TestVectors文件夹下的测试数据和测试用例，所以运行测试程序前需要将这两个文件夹拷贝至“当前”目录。否则，运行cryptest.exe程序时会出错。例如，以tv参数运行cryptest.exe程序时错误信息提示如下：

```
$ cryptest.exe tv
Using seed: 1569671499
Exception caught: Can not open file TestVectors/all.txt for reading
```

下面以Linux系统为例，演示cryptest.exe程序的使用方法。

2.1 为cryptest.exe程序配置运行环境

“配置”该测试程序运行环境的方法非常简单，即将TestData文件夹和TestVectors文件夹拷贝至当前目录。例如，若将在用户主目录下的crypto目录下运行测试程序，则先执行如下的目录创建和数据拷贝命令：

```
$ mkdir crypto
$ cd ./crypto
$ cp  -r ../cryptopp/TestData ./
$ cp  -r ../cryptopp/TestVectors ./
```

2.2 查看cryptest.exe测试程序的可用参数

以空参数或“h”参数运行cryptest.exe程序即可得到测试程序的“帮助信息”。

```
$ cryptest.exe h
Test Driver for Crypto++(R) Library, a C++ Class Library of Cryptographic Schemes

- To generate an RSA key
cryptest g

- To encrypt and decrypt a string using RSA
cryptest r

- To sign a file using RSA
cryptest rs privatekeyfile messagefile signaturefile

- To verify a signature of a file using RSA
cryptest rv publickeyfile messagefile signaturefile

- To digest a file using several hash functions in parallel
cryptest m file

- To encrypt and decrypt a string using DES-EDE in CBC mode
cryptest t

- To encrypt or decrypt a file
cryptest e|d input output

- To secret share a file (shares will be named file.000, file.001, etc)
cryptest ss threshold number-of-shares file

- To reconstruct a secret-shared file
cryptest sr file share1 share2 [....]
(number of shares given must be equal to threshold)

- To information disperse a file (shares will be named file.000, file.001, etc)
```

```
cryptest id threshold number-of-shares file

- To reconstruct an information-dispersed file
cryptest ir file share1 share2 [....]
(number of shares given must be equal to threshold)

- To gzip a file
cryptest z compression-level input output

- To gunzip a file
cryptest u input output

- To encrypt a file with AES in CTR mode
cryptest ae input output

- To base64 encode a file
cryptest e64 input output

- To base64 decode a file
cryptest d64 input output

- To hex encode a file
cryptest e16 input output

- To hex decode a file
cryptest d16 input output

- To forward a TCP connection
cryptest ft source-port destination-host destination-port

- To run the FIPS 140-2 sample application
cryptest fips

- To generate 100000 random files using FIPS Approved X.917 RNG
cryptest fips-rand

- To run Maurer's randomness test on a file
cryptest mt input

- To run a test script (available in TestVectors subdirectory)
cryptest tv filename

- To run validation tests
cryptest v

- To display version number
```

```
        cryptest V

- To run benchmarks
cryptest b [time allocated for each benchmark in seconds] [frequency of CPU in
gigahertz]
```

2.3 运行cryptest.exe程序

下面演示cryptest.exe测试程序各个参数的使用方法。

2.3.1 生成RSA算法密钥—cryptest g

cryptest g命令可以产生RSA公私钥对，使用方法如下：

```
$ cryptest.exe g
Key length in bits: 1024
Save private key to file: prikey
Save public key to file: pubkey
Random Seed:12192912301023123
```

上述命令以12192912301023123为种子，生成长度为1024比特的随机RSA公私钥对，公钥保存在pubkey文件中，私钥保存在prikey文件中。pubkey文件和prikey文件位于当前目录下。

```
$ ls
prikey  pubkey  TestData  TestVectors
```

2.3.2 用RSA算法加解密字符串—cryptest r

cryptest r命令可以用cryptest g命令产生的RSA公私钥实现字符串加解密。下面演示加解密字符串“I likeCryptography.”的方法。

```
$ cryptest.exe r
Private key file: prikey
Public key file: pubkey
Random Seed: 123456
Message: I like Cryptography.
Ciphertext: 188097CD98E6C18F1F578F7A801345A82131B073063664537947BCB93DE36C1B013
8E093D8737B6BDD8E0B7D8A56EBC00B8C26EF81033666AD7CE5BD6FBDF13A1544AC3620665BBA24
61E0A0F7FA68BF09C0D724F39D3476E68B9A7DF99047123E68A74C71695804A0A1EBEDD151094CD
10CEC1B7685446295BB75F0C33779E1
Decrypted: I like Cryptography.
```

2.3.3 用RSA算法对文件签名—cryptest rs

cryptest rs privatekeyfile messagefile signaturefile命令可以用cryptest g命令产生的RSA私钥实现对文件的签名。它的具体使用方法如下：

```
cryptest.exe rs prikey pubkey sigtxt
```

上面的命令用私钥prikey对公钥文件pubkey（也可以是其他待签名的文件）进行签名，签名的结果存储于文件sigtxt。文件sigtxt的内容如下：

```
$ cat sigtxt
50E6C6F5B8612A100BF73FC4C129FC7D21E64B1072A6EEC15844649C9A238D1EA94C7B026E03E87
4C20D1382B1310970A79EAC4EFFA5305898A0ED9836081015B090E38F4CFEDAB123EC4275B623AC
7A3CB3902BCFA675104E8D7E8125377CB9DC153489D43B6E41B9B2A1B590FDCD03D1888AD22E7E3
9C00A32C11E201D3FB5
```

2.3.4 用RSA算法验证文件的签名—cryptest rv

cryptest rv publickeyfile messagefile signaturefile可以用cryptest g命令产生的RSA公钥实现对文件的验签。它的具体使用方法如下：

```
$ cryptest.exe rv pubkey pubkey sigtxt
valid signature
```

上面的命令实现用公钥文件pubkey验证上一步对pubkey文件的签名sigtxt的有效性，验证的结果显示“签名有效”。

2.3.5 用一些Hash函数并行计算文件摘要—cryptest m file

cryptest m file命令用一些Hash函数并行计算文件的消息摘要。它的具体使用方法如下：

```
cryptest.exe m sigtxt
SHA-1: a55d1cc45d0a68a13d2391d2a994d464ac637127
RIPEMD-160: d7318c0b0f592357e3ecd97c1303894b0a426d52
Tiger: ad7291ffe9e23d6e20ac9d3a74fac0484b805c58e96e3cfc
SHA-256: b52ab3199349281ed1ad24a7a55e5dad24288f4bbdab618b110e8bff5d897762
SHA-512: a5a8708c4f7ebd0a8832cdf759c0e436a95e8cf3067b46f04fb62ada3973448ee45e1d
6cf2bc8758467127d7096055924ffcb2d96bef968bb1589ab5ff7214f6
Whirlpool: 511e1e07ef5f2a1546f86bc8b6910eaede53aeed1c8d696fbf94e83032ac18d332da
a1542b7613f4317e55e9a72376dc0756bcd8b2042bd2b157b603b9ae5cb3
```

上面的命令用Hash函数SHA-1、RIPEMD-160、Tiger、SHA-256、SHA-512、Whirlpool分别计算文件sigtxt的摘要。

2.3.6 以CBC模式运行DES-EDE算法并加解密字符串—cryptest t

cryptest t命令以CBC模式运行DES-EDE算法并加解密字符串。它的具体使用方法如下：

```
$ cryptest.exe t
Passphrase: 123456
Plaintext: I like Cryptography.
Ciphertext: 8D9A579E3F80DC92A0FD92703039CCFD66A4D02172205AAA66D5D6CEB6DF5298A2B
C53DF290292E749C19E03E336B6DC0507D3125ED8A9D8791EB1C577FC81943336ED063919A71F7A
055226368BE14A1DE245C56FD41867
Decrypted: I like Cryptography.
```

2.3.7 加密或解密文件—cryptest e|d

cryptest e|d input output命令可以实现文件的加解密。它的具体使用方法如下：

（1）加密（e）文件：

```
$ cryptest.exe e sigtxt ciphertxt
Passphrase: 12345678
```

上面的命令利用口令12345678对文件sigtxt进行加密，加密的结果存储于文件ciphertxt。由于文件ciphertxt中的内容为不可读辨识的信息，所以使用cat等命令查看其内容时会显示“乱码”。此时，可以使用od命令以十六进制形式“阅读”它的内容。

```
$ od -x ciphertxt
0000000 7bf4 2fbe 54a6 2c91 4883 ca38 882a 88cf
0000020 ee0e c0ad e342 3ad6 1b85 47b8 3bb3 fd00
0000040 ec3e 7d8b 838e 0216 826f af03 81b9 9ebb
0000060 575f 36bf 8ba1 154b 0414 0e97 0d27 3035
0000100 7549 6ae4 6733 4914 4e58 d382 32e8 794f
0000120 6b1d 795e 82ff 8e85 0f57 ab2f 447f 7b84
0000140 de34 97bf c691 a049 4b84 41dd 7e5f a1b4
0000160 2799 5d14 9c7d 4551 49b1 e86e fab7 d6a4
0000200 4395 ac2a deb5 e1a4 7b18 0bc0 cbb5 ebb7
0000220 cd22 69a8 1357 461f 06f4 e259 0162 6e3f
0000240 f5ac 03b4 c11e ed40 06c2 d32e 944c a2e6
0000260 3696 fb70 c776 e39b 60e0 4d5c dd16 98f6
0000300 086b c47b 1c79 c908 f2fd f575 a597 9d74
0000320 3ade f559 e4a2 3613 60bf 8390 2a08 c87f
0000340 d673 9ac6 4e34 c4eb c43c 1d66 9446 17d4
0000360 0652 8efe 46b8 d34c b51e 8a42 1557 9134
0000400 cbbe 4e89 bc97 bb66 a2f4 09e4 a586 7035
0000420 829c c6d4 7a54 5fbe 206d cbd5 0b9e 9719
0000440 7a6f 8f90 409b 863b da2f 3c9a 1f37 066c
0000460 cd25 bcd5 1afb 9e19 d111 8229 9e76 b60b
0000500 a0aa f546 ae83 aace
0000510
```

（2）加密（d）文件：

```
$ cryptest.exe d ciphertxt recovertxt
Passphrase: 12345678
```

上面的命令利用口令12345678对文件ciphertxt进行解密，解密的结果存储于文件recovertxt。文件recovertxt中的内容与文件sigtxt中的内容完全一致。

使用cat命令查看文件recovertxt中的内容。

```
$ cat recovertxt
50E6C6F5B8612A100BF73FC4C129FC7D21E64B1072A6EEC15844649C9A238D1EA94C7B026E03E87
4C20D1382B1310970A79EAC4EFFA5305898A0ED9836081015B090E38F4CFEDAB123EC4275B623AC
7A3CB3902BCFA675104E8D7E8125377CB9DC153489D43B6E41B9B2A1B590FDCD03D1888AD22E7E3
9C00A32C11E201D3FB5
```

注意：解密时使用的口令应与加密时使用的口令完全一致。

2.3.8 对文件执行秘密分割—cryptest ss

cryptest ss threshold number-of-shares file命令可以实现文件（或秘密信息）的分割。它的具体使用方法如下：

```
$ cryptest.exe ss 3 5 recovertxt
Random Seed: 1234567890
```

上面的命令以1234567890为随机数发生器的种子，对文件recovertxt进行秘密分割。文件被分割为5个份额，陷门的个数被设置为3。新产生的这5个文件分别是recovertxt.000、recovertxt.001、recovertxt.002、recovertxt.003、recovertxt.004。

2.3.9 根据秘密份额重构文件—cryptest sr

cryptest sr file share1 share2 [....]命令与cryptest ss threshold number-of-shares file命令相对应，可以从分割的文件份额中恢复出原始文件。它的具体使用方法如下：

```
$ cryptest.exe sr  srtxt recovertxt.000 recovertxt.002 recovertxt.004
```

上面的命令从3个份额recovertxt.000、recovertxt.002、recovertxt.004中恢复出原始文件并将恢复出的信息存储于srtxt文件中。

```
$ cat srtxt
50E6C6F5B8612A100BF73FC4C129FC7D21E64B1072A6EEC15844649C9A238D1EA94C7B026E03E87
4C20D1382B1310970A79EAC4EFFA5305898A0ED9836081015B090E38F4CFEDAB123EC4275B623AC
7A3CB3902BCFA675104E8D7E8125377CB9DC153489D43B6E41B9B2A1B590FDCD03D1888AD22E7E3
9C00A32C11E201D3FB5
```

2.3.10 对文件执行信息分割—cryptest id

cryptest id threshold number-of-shares file命令与cryptest ss threshold number-of-shares file命令功能类似，关于它的使用方法详见2.3.8部分相关内容。

2.3.11 根据信息份额重构文件—cryptest ir

cryptest ir file share1 share2 [....]命令与cryptest sr file share1 share2 [....]命令功能类似，关于它的使用方法详见2.3.9部分相关内容。

2.3.12 压缩文件—cryptest z

cryptest z compression-level input output命令可以实现文件压缩。它的具体使用方法如下：

```
$ cryptest.exe z 2 srtxt zoutput.txt
```

上面的命令以2级压缩级别将srtxt文件进行压缩，压缩后的内容存储于文件zoutput.txt中。

2.3.13 解压文件—cryptest u

cryptest u input output命令与cryptest z compression-level input output命令相对应，它可以实现文件的解压缩。它的具体使用方法如下：

```
$ cryptest.exe u zoutput.txt uoutput.txt
```

上面的命令将压缩文件zoutput.txt中的内容进行解压缩，并将解压后的内容存储于文件uoutput.txt中。

```
$ cat uoutput.txt
50E6C6F5B8612A100BF73FC4C129FC7D21E64B1072A6EEC15844649C9A238D1EA94C7B026E03E87
4C20D1382B1310970A79EAC4EFFA5305898A0ED9836081015B090E38F4CFEDAB123EC4275B623AC
7A3CB3902BCFA675104E8D7E8125377CB9DC153489D43B6E41B9B2A1B590FDCD03D1888AD22E7E3
9C00A32C11E201D3FB5
```

2.3.14 以CTR模式运行AES算法加密文件—cryptest ae

cryptest ae input output命令以CTR模式运行AES算法对文件加密。它的具体使用方法如下：

```
$ cryptest.exe ae sigtxt ae_sigtxt
Exception caught: AES/CTR: 0 is not a valid key length
```

运行cryptest ae命令时出现异常——0不是有效的秘钥长度。

2.3.15 以Base64编码文件—cryptest e64

cryptest e64 input output命令以Base64编码规则对文件进行编码。它的具体使用方法如下：

```
$ cryptest.exe e64 sigtxt e64_sigtxt
```

上面的命令将文件sigtxt中的内容以Base64编码规则进行编码，并将编码后的内容存储于文件e64_sigtxt中。

用cat命令查看文件e64_sigtxt中的内容。

```
$ cat e64_sigtxt
NTBFNkM2RjVCODYxMkExMDBCRjczRkM0QzEyOUZDN0QyMUU2NEIxMDcyQTZFRUMxNTg0NDY0
OUM5QTIzOEQxRUE5NEM3QjAyNkUwM0U4NzRDMjBEMTM4MkIxMzEwOTcwQTc5RUFDNEVGRkE1
MzA1ODk4QTBFRDk4MzYwODEwMTVCMDkwRTM4RjRDRkVEQUIxMjNFQzQyNzVCNjIzQUM3QTND
QjM5MDJCQ0ZBNjc1MTA0RThEN0U4MTI1Mzc3Q0I5REMxNTM0ODlENDNCNkU0MUI5QjJBMUI1
OTBGRENEMDNEMTg4OEFEMjJFN0UzOUMwMEEzMkMxMUUyMDFEM0ZCNQ==
```

2.3.16 以Base64解码文件—cryptest d64

cryptest d64 input output命令与cryptest e64 input output命令相对应，它能够对Base64编码的文件进行解码。它的具体使用方法如下：

```
$ cryptest.exe d64 e64_sigtxt d64_sigtxt
```

上面的命令将文件e64_sigtxt中的内容以Base64编码规则进行解码，并将解码后的内容存储于文件d64_sigtxt中。

用cat命令查看文件d64_sigtxt中内容。

```
$ cat d64_sigtxt
50E6C6F5B8612A100BF73FC4C129FC7D21E64B1072A6EEC15844649C9A238D1EA94C7B026E03E87
4C20D1382B1310970A79EAC4EFFA5305898A0ED9836081015B090E38F4CFEDAB123EC4275B623AC
7A3CB3902BCFA675104E8D7E8125377CB9DC153489D43B6E41B9B2A1B590FDCD03D1888AD22E7E3
9C00A32C11E201D3FB5
```

2.3.17 以Base16编码文件—cryptest e16

cryptest e16 input output命令与cryptest e64 input output命令的使用方法类似。前者以Base16编码规则对文件进行编码，而后者以Base64编码规则对文件进行编码。关于它的使用方法可以参考2.3.15部分相关内容。

2.3.18 以Base16解码文件—cryptest d16

cryptest d64 input output命令与cryptest d64 input output命令的使用方法类似。前者以Base16编码规则对文件进行解码，而后者以Base64编码规则对文件进行解码。关于它的使用方法可以参考2.3.16部分相关内容。

2.3.19 发送一个TCP连接—cryptest ft

cryptest ft source-port destination-host destination-port命令可以向目标主机发送一个TCP连接。

它的具体使用方法暂时未知。

2.3.20 运行FIPS 140-2示例程序—cryptest fips

cryptest fips命令可以运行FIPS 140-2示例程序。它的具体使用方法如下：

```
$ cryptest.exe fips
FIPS 140-2 compliance was turned off at compile time.
```

注意：若要执行此命令，则在程序编译阶段应该开启FIPS 140-2规范选项。

2.3.21 用X.917随机数发生器产生随机文件—cryptest fips-rand

cryptest fips-rand命令使用FIPS批准的X.917随机数发生器产生100000个随机文件。它的具体使用方法如下：

```
$ cryptest.exe fips-rand
```

使用ls命令查看在当前目录下生成的随机文件。

```
$ ls
// ...
// ...
24997.rnd  3999.rnd   55000.rnd  70003.rnd  85007.rnd  recovertxt.003
24998.rnd  399.rnd    55001.rnd  70004.rnd  85008.rnd  recovertxt.004
24999.rnd  39.rnd     55002.rnd  70005.rnd  85009.rnd  sigtxt
2499.rnd   3.rnd      55003.rnd  70006.rnd  8500.rnd   srtxt
249.rnd    40000.rnd  55004.rnd  70007.rnd  85010.rnd  TestData
24.rnd     40001.rnd  55005.rnd  70008.rnd  85011.rnd  TestVectors
25000.rnd  40002.rnd  55006.rnd  70009.rnd  85012.rnd  uoutput.txt
25001.rnd  40003.rnd  55007.rnd  7000.rnd   85013.rnd  zoutput.txt
25002.rnd  40004.rnd  55008.rnd  70010.rnd  85014.rnd
```

注意：由于产生的随机文件较多，所以该命令需要耗费一定的时间。

2.3.22 用Maurer随机性测试检测文件随机性—cryptest mt

cryptest mt input命令使用Maurer随机性测试检测文件的随机性。它的具体使用方法如下：

```
$ cryptest.exe mt 84308.rnd
Maurer Test Value: 0.999533
```

上面的命令使用Maurer随机性测试方法测试文件84308.rnd的随机性。

2.3.23 运行测试脚本—cryptest tv

cryptest tv filename命令使用TestVectors子目录下的测试脚本对程序进行测试。当不指定filename参数时，它会执行所有的测试。它的具体使用方法如下：

```
$ cryptest.exe tv
// ...
// ...
Testing KDF algorithm HKDF(SHA-1).
....
Testing KDF algorithm HKDF(SHA-256).
...
Testing KDF algorithm HKDF(SHA-512).
.....
Testing KDF algorithm HKDF(Whirlpool).
.....
Tests complete. Total tests = 7829. Failed tests = 0.
```

2.3.24 运行有效性测试—cryptest v

cryptest v命令可以验证程序的有效性。它的具体使用方法如下：

```
$ cryptest.exe v
// ...
// ...
Testing NaCl library functions...

passed    crypto_box, crypto_box_beforenm, crypto_box_afternm
passed    crypto_box_open, crypto_box_open_afternm
passed    crypto_box_keypair pairwise consistency
passed    crypto_sign, crypto_sign_open, crypto_sign_keypair
passed    crypto_sign_keypair pairwise consistency

All tests passed!

Seed used was 1569813202
Test started at Mon Sep 30 11:13:22 2019
Test ended at Mon Sep 30 11:13:26 2019
```

2.3.25 显示版本号—cryptest V

cryptest V命令可以显示cryptest.exe程序的版本号。它的具体使用方法如下：

```
$ cryptest.exe V
7.0.0
```

注意：“V”为大写。

2.3.26 运行速度基准测试—cryptest b

cryptest b [time allocated for each benchmark in seconds] [frequency of CPU ingigahertz]命令会对算法的执行速度进行测试。它的具体使用方法如下：

```
$ cryptest.exe b
// ...
// ...
<TR><TD>TEA/CTR (128-bit key)<TD>78<TD>0.525
<TR><TD>XTEA/CTR (128-bit key)<TD>64<TD>0.527
<TR><TD>SKIPJACK/CTR (80-bit key)<TD>43<TD>1.878
<TR><TD>SEED/CTR (1/2 K table)<TD>68<TD>0.532
<TR><TD>SM4/CTR (128-bit key)<TD>96<TD>0.632
<TR><TD>Kalyna-128(128)/CTR (128-bit key)<TD>149<TD>0.558
<TR><TD>Kalyna-128(256)/CTR (256-bit key)<TD>114<TD>0.601
<TR><TD>Kalyna-256(256)/CTR (256-bit key)<TD>150<TD>0.729
<TR><TD>Kalyna-256(512)/CTR (512-bit key)<TD>121<TD>0.820
<TR><TD>Kalyna-512(512)/CTR (512-bit key)<TD>153<TD>1.038
```

```
<TR><TD>SIMON-64(96)/CTR (96-bit key)<TD>507<TD>0.483
<TR><TD>SIMON-64(128)/CTR (128-bit key)<TD>496<TD>0.497
<TR><TD>SIMON-128(128)/CTR (128-bit key)<TD>376<TD>0.508
<TR><TD>SIMON-128(192)/CTR (192-bit key)<TD>364<TD>0.514
// ...
// ...
<TR><TD>ECMQVC over GF(2^n) 233 Key-Pair Generation with precomputation<TD>0.97
<TR><TD>ECMQVC over GF(2^n) 233 Key Agreement<TD>4.10
</TABLE>
<P>Throughput Geometric Average: 1004.361791
<P>Test started at Mon Sep 30 11:26:27 2019
<BR>Test ended at Mon Sep 30 11:31:49 2019
</BODY>
</HTML>
```

上面的基准测试输出与前面的测试结果不同，它的结果显得有些杂乱、难懂。事实上，cryptest b命令的输出内容是一个HTML格式的文档。利用输出重定向操作，可将它的测试结果输出至test.html文件中。

```
$ cryptest.exe b > test.html
```

等待基准测试执行完毕后，在当前目录下会看到生成的test.html文件。使用浏览器打开该文件，即可看到速度基准测试有意义、可读的输出结果，如图2所示。

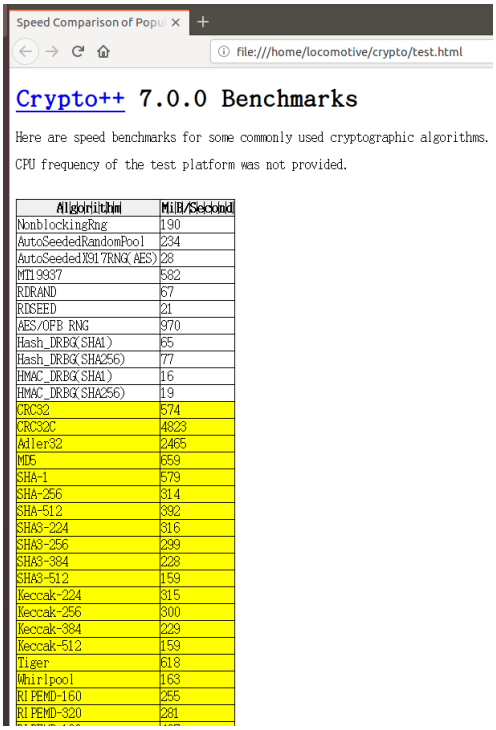

图2　基准测试生成的test.html文件中的内容

从下面的链接可获得这个test.html文件以及它对应的PDF文件：
https://github.com/locomotive-crypto/crypto_book_1st/tree/master/others/using%20cryptest.exe

3 声明

Cryptography

⇓

⇓

⇓

该文档为《深入浅出CryptoPP密码学库》的附加电子文档——书籍正文和随书电子文档的补充。关于CryptoPP中密码学原语的具体使用方法，详见书籍相应章节内容。

由于作者水平有限，错误之处在所难免。欢迎通过如下方式反馈相关问题：

⇒ email:locomotive_crypto@163.com

⇓

⇓

⇓

《深入浅出CryptoPP密码学库》