Technical Paper

SAS® Data Set Encryption Options

SAS product interaction
with encrypted data storage

Table of Contents

Introduction: What Is Encryption?

Encryption is a formal system that obscures data in such a way that it is readable only by an intended recipient(s) usually though the use of a “shared secret.” For example, the ancient Romans wrapped a strip of cloth around a rod, wrote their message across the cloth, and then unwrapped it. If the cloth were intercepted, it would appear to contain random characters. Upon reaching the intended destination, the cloth would be wrapped around a rod of the same diameter and the message read. The shared secret was the rod’s diameter.

Today, modern computer-based encryption uses mathematical algorithms to produce an encrypted data stream that contains no discernible pattern. The shared secret is usually an input or generated text string called a “key.”

SAS® uses two types of encryption:

- encrypting data during transmission
- encrypting data during storage

This paper discusses encrypting data during storage, referencing a test case using Base SAS® 9.3. It uses a simple SAS program and a large text data file to look at the performance of a baseline case and four different encryption methods. The baseline test uses no encryption. The first two tests examine the **SAS ENCRYPT= Data Set Option** and the **Windows Encrypting File System**, in which the input text file is read unencrypted and the output data file is encrypted. The last two tests examine **Windows BitLocker** and **TrueCrypt**; both the input text file and output data file are encrypted.

Test Configuration

Data

A single test file was used for all tests. The file contained two billion high-quality random integer values ranging from zero to 4,294,967,295 in text format as one value per line. (See Appendix B for a sample.) The total file size was 21.8 GB, which ensures that a considerable number of I/O operations are required to properly illustrate the testing outcomes.

This data provides an equal chance of having any value in the range; the mean is expected to be close to 2,147,483,647.5 (half-way between the minimum and maximum values). The computed mean of 2,147,486,312 is within 0.0001% of the expected norm. The randomness of the values is expected to produce a large standard deviation because the randomness of the values does not yield a normal distribution. The computed standard deviation is 1,239,858,134. See Appendix A for additional test data statistics.

Code

This code was used for all tests. It reads the text data file and calculates the mean using the MEANS procedure:

```
libname testdata 'c:\testdata\';

data testdata.rands;
   infile 'c:\testdata\rdrand.dat';
   input random;
run;

proc means data=testdata.rands;
```

run;

Testing Platforms

Two testing platforms were used:

- Windows 7 Ultimate (Service Pack 1) running inside a virtual machine hosted on a Windows 2012 server. Each instance was given 8 GB of memory and four processors.
- Windows 7 Enterprise (SP1) running on a Dell 7010 (i7-3770 @3.4 GHz) with 16 GB of memory.

Expected Output

The expected output of the PROC MEANS is:

```
              N            Mean         Std Dev         Minimum         Maximum
   2000000000      2147486312      1239858134               0      4294967290
```

No tests deviated from these values. An intermediate data file (“rands”) was also generated requiring 16,351,302,656 bytes (~16 GB) of disk space.

Typical Log Output

The log outputs for the tests were similar. A typical log output looked like this:

```
NOTE: Copyright (c) 2002-2010 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.3 (TS1M2)
      Licensed to SAS Institute Inc., Site 1.
NOTE: This session is executing on the X64_7PRO  platform.



NOTE: SAS initialization used:
      real time           0.12 seconds
      cpu time            0.10 seconds

1          libname testdata 'C:\testdata';
NOTE: Libref TESTDATA was successfully assigned as follows:
      Engine:        V9
      Physical Name: C:\testdata
2
3          data testdata.rands;
4             infile 'C:\testdata\rdrand.dat';

5             input random;
6          run;

NOTE: The infile 'C:\testdata\rdrand.dat' is:
      Filename=C:\testdata\rdrand.dat,
      RECFM=V,LRECL=256,
      File Size (bytes)=23482605474,
      Last Modified=19Feb2013:18:06:06,
      Create Time=10May2013:15:05:52
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
      The minimum record length was 1.
      The maximum record length was 10.
NOTE: The data set TESTDATA.RANDS has 2000000000 observations and 1 variables.
NOTE: DATA statement used (Total process time):
      real time           timevalue
      cpu time            timevalue

7
8          proc means data=testdata.rands;
9          run;

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: The PROCEDURE MEANS printed page 1.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           timevalue
      cpu time            timevalue
NOTE: The SAS System used:
      real time           timevalue
      cpu time            timevalue
```

Baseline

The baseline execution was performed with no encryption. The log output values were as follows:

Virtual Machine

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           16:05.92
      cpu time            12:23.54

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           3:46.20
      cpu time            6:41.29

NOTE: The SAS System used:
      real time           19:52.54
      cpu time            19:05.04
```

7010

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time            9:11.94
      cpu time             5:52.51

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           2:29.86
      cpu time            5:24.62

NOTE: The SAS System used:
      real time           11:41.93
      cpu time            11:17.24
```

The baseline execution of the test shows a real-time value for reading the data file that exceeds the CPU time value by a considerable margin. The bulk of the CPU time represents the time required to convert the text representation into a binary representation. The difference between the real and CPU time represents the actual reading and writing of the file and the associated system overhead; this is a single threaded operation. The reverse occurs for PROC MEANS processing because multiple CPU cores are used causing the system to report almost twice the CPU time as real time. See **Threading in Base SAS** for additional information.

SAS Encrypt Option

The ENCRYPT data set option was added to SAS 6.11 in 1995. The ENCRYPT data set option is based on a proprietary algorithm using a 32-bit fixed encoding. In the nearly two decades since the introduction of ENCRYPT data set option, the dramatic growth in computing speed and capabilities has left this encryption vulnerable to “brute force” attacks that can try every possible key. This type of attack was possible in 1995 also, but the time required to do so was considered “unreasonable.” There is a long history of cipher improvements being overcome by technical advancements, and it is prudent to expect that encryption methods considered secure today might well be vulnerable to such attacks in the future.

However, the SAS ENCRYPT option is still quite valuable. It effectively obscures data such that casual data browsing is thwarted. It’s similar to locking your home or car. It does not prevent people from breaking in or picking a lock, but it serves as a sufficient barrier to prevent curious people from getting in. The ENCRYPT data set option is effective because most people do not have the knowledge or motivation to break into this type of encrypted data.

Setting the ENCRYPT data set option requires that you also set a password on the file, which you can do with other SAS data set options. At minimum, the data set option READ= password is required. You can also use WRITE= and ALTER= data set options to assign additional passwords. For example:

```
libname testdata 'c:\testdata\';
    data testdata.rands (encrypt=yes
                         read=Z6FD7197
                         write=A987C903
                         alter=Go_W2278);
       infile 'c:\testdata\rdrand.dat';
       input random;
run;
proc means data=testdata.rands (read=Z6FD7197);
run;
```

In this example, we request the creation of a data set named testdata.rands. The data set will be encrypted and will have three different associated passwords. The additional passwords allow users to share a password that allows reading but not allow the file to be rewritten or altered. You can also use the PW= data set option to set the read, write, and alter passwords to a single value.

The log files from executing the test program on both the Windows virtual machine and the Windows 7010 machine show the real time and CPU time for both the DATA statement and PROC MEANS:

Virtual Machine

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           16:27.12
      cpu time            12:31.28

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           4:00.51
      cpu time            7:06.67

NOTE: The SAS System used:
      real time           20:28.10
      cpu time            19:38.21
```

7010

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time            9:05.95
      cpu time             6:10.67

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           2:21.83
      cpu time            5:42.17

NOTE: The SAS System used:
      real time           11:27.90
      cpu time            11:52.90
```

The virtual machine shows a real-time difference of ~3% (It took a little bit longer for SAS encryption.) and about the same (2.78%) for the CPU. When using the 7010 machine, the real time difference shows that the SAS encryption run is a bit faster (2%) than the baseline; but the CPU times show the baseline to be 5% faster.

While real time is generally what people are interested in, CPU time is a better indicator of work done. There will be run-to-run differences, especially in the real-time values. These differences are caused by different disk access patterns and different levels of system overhead from one run to the next. The takeaway is that this form of encryption requires very little additional processing time.

SAS® 9.4

With SAS® 9.4, you can choose to instead specify **ENCRYPT=AES** as a data set option when creating the data and an ENCRYPTKEY='string' value. This uses the AES-256 encryption cipher in the SAS/SECURE product to encrypt the data on disk. The ENCRYPTKEY= must also be specified on attempts to open or update the data to facilitate decryption.

This works with Base SAS and SPD Engine data sets. It also supports SPD Server data sets when Base SAS is installed on the same server as SAS Scalable Performance Data Server 5.1.

For Base SAS data, the entire table and metadata are encrypted. For SPD Engine and SPD Server, the data files and index component files are encrypted, but the metadata file is not (SPD* format stores metadata in a separate physical file from the data partition(s) that hold the rows of data).

None of this affects how the data is stored in memory or passed over a network connection; this is done only between the storage data access layer and the file system.

Since SAS 9.3 was used in testing for this paper, this feature was not included.

Windows EFS (Encrypting File System)

The EFS is available on all versions of windows supporting NTFS version 3.0 and later. This uses much stronger encryption algorithms and much longer keys (more than 255 bits) than the classic SAS ENCRYPT option. This makes the actual encrypted data much more difficult to decrypt without the proper keys. This encryption is tied to the user account so that the user doesn't have to keep track of long keys.   But it provides a significant weakness:

> "… the cryptography keys for EFS are in practice protected by the user account password, and are therefore susceptible to most password attacks. In other words, encryption of files is only as strong as the password to unlock the decryption key." (**http://en.wikipedia.org/wiki/Encrypting_File_System**).

The files are better encrypted but not necessarily more secure. (Refer to the full text of Wikipedia article.) This system is relatively easy to use and generates (and stores) its own keys. You can encrypt at the file, directory, or drive levels. For this test, the LIBNAME statement in the baseline code was changed to point to an encrypted directory. But the text data is read from an unencrypted source. The testdata.rands data file was written to the encrypted directory and then read from to complete the PROC MEANS processing. The log time values were as follows:

Virtual Machine

```
NOTE: 2000000000 records were read from the infile 'e:\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           16:10.01
      cpu time            12:08.70

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           6:03.25
      cpu time            6:34.14

NOTE: The SAS System used:
      real time           22:13.79
      cpu time            18:43.14
```

7010

```
NOTE: 2000000000 records were read from the infile 'e:\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time            9:43.84
      cpu time             6:04.21

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           2:24.22
      cpu time            5:18.99

NOTE: The SAS System used:
      real time           12:08.17
      cpu time            11:23.36
```

The virtual machine shows a real-time difference of ~12% (took 11.84% longer for EFS). The CPU times were inverted; it took 1.84% more time for the baseline run. Since the encryption is handled by the operating system (That is, it does not count toward SAS CPU usage.) you would theoretically not see any difference between the baseline CPU and EFS CPU. However, variations in memory handling and task scheduling do introduce small time variances. The 7010 machine shows EFS requiring 3.74% more real time and 0.87% more CPU time. The CPU time variance can be ignored because encryption is occurring at the operating system level. The real-time difference shows the time the OS spend encrypting the file.

Something interesting to note is the additional time used executing the PROC MEANS step in the VM case. The decryption of the testdata.rands data file (as it is being read for the PROC MEANS processing) requires considerable CPU resources, and this leaves fewer CPU resources for the computations required by PROC MEANS. This led to a 60% increase in real time for PROC MEANS processing and an 11% overall increase in real time for the run. This is less noticeable in the 7010 run since that machine has double the number of cores available (and twice the memory).

TrueCrypt

TrueCrypt is a free open-source disk encryption tool which supports drive (volume) level encryption. It provides a variety of encryption algorithms and uses 256-bit keys. TrueCrypt installs as a separate tool that can be used to create and mount encoded volumes. Disks can be mounted on demand or auto-mounted. You can also set it to request a password each time or to use a file that can be stored on a removable device.

For this test, the LIBNAME statement in the baseline code was changed to point to an encrypted drive. Note that both the input source file and output file are stored on the encrypted volume. (This means the input data requires decryption before it is usable by SAS.) TrueCrypt 7.1a was used for these tests. The log time values were as follows:

Virtual Machine

```
NOTE: 2000000000 records were read from the infile 'e:\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           20:33.75
      cpu time            12:20.31

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           3:57.25
      cpu time            6:28.42

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-2414
NOTE: The SAS System used:
      real time           24:31.51
      cpu time            18:49.03
```

7010

```
NOTE: 2000000000 records were read from the infile 'e:\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time            9:44.39
      cpu time             6:14.74

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time            3:08.93
      cpu time             5:26.94

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-2414
NOTE: The SAS System used:
      real time           12:53.40
      cpu time            11:41.81
```

In the virtual machine, TrueCrypt shows a 23.39% increase in real time and a 1.34% decrease in CPU time. Again since the operating system is accounting for the encryption time, the SAS reported CPU time can be ignored. This tool fares much better on the 7010 hardware (more cores, more memory) showing only a 10.18% increase in real time.

Windows BitLocker

BitLocker is a full disk encryption product that is bundled with desktop versions of Enterprise and Ultimate on Windows7, and Enterprise and Pro versions on Windows 8. Additionally, BitLocker is bundled with Windows Server 2008, Windows Server 2008 R2, and Windows Server 2012.

By default BitLocker uses a 128-bit key, but it can be configured to use a 256-bit key. BitLocker can be configured to use the Trusted Platform Module (TPM) hardware which works with the OS to secure the encryption key.

It can also be configured to use a key stored on a removable device (such as a USB thumb drive). Because this is a full-disk encryption tool, both the input source and output are encrypted, so we can expect times to be greater. No changes to the baseline SAS code were needed for this method. The log time values were as follows:

Virtual Machine

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           3:16:54.35
      cpu time            12:04.93

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           2:51:19.48
      cpu time            6:39.20

NOTE: The SAS System used:
      real time           6:08:15.31
      cpu time            18:44.34
```

7010

```
NOTE: 2000000000 records were read from the infile 'c:\testdata\rdrand.dat'.
NOTE: DATA statement used (Total process time):
      real time           11:47.82
      cpu time            06:03.27

NOTE: There were 2000000000 observations read from the data set TESTDATA.RANDS.
NOTE: PROCEDURE MEANS used (Total process time):
      real time           2:23.75
      cpu time            5:31.06

NOTE: The SAS System used:
      real time           14:11.70
      cpu time            11:34.53
```

Even though Microsoft doesn't support BitLocker running on a virtual machine, the numbers are included for completeness. Clearly, more than six hours of real time and only 18 minutes of CPU time indicates some sort of problem (perhaps the very reason Microsoft doesn't support BitLocker on VMs). The 7010 values are much more in line with expectations, showing a 21.34% increase in real time. Again in this case, CPU time can be ignored since SAS isn't doing the encryption work.

Performance Summary

The values in the tables below compare the results of the various tests. They are not intended to endorse one encryption solution over another. They provide a general indication of the performance costs across a few of the different solutions available.

Virtual Machine			
Method	**Time**	**Time Difference**	**Change from Baseline**
Baseline	19:52.5		
SAS Encrypt option	20:28.1	00:35.6	2.98%
EFS	22:13.8	02:21.2	11.84%
TrueCrypt	24:31.5	04:39.0	23.39%
BitLocker	6:08:15	5:48:22	1752.77%

Dell 7010 (i7-3770@3.4 GHz)			
Method	**Time**	**Time Difference**	**Change from Baseline**
Baseline	11:41.9		
SAS Encrypt option	11:27.9	00:14.0	2.00%
EFS	12:08.2	00:26.2	3.74%
TrueCrypt	12:53.4	01:11.5	10.18%
BitLocker	14:11.7	02:29.8	21.34%

Considerations

- You should keep your security keys safe. Back up and secure your keys because losing your key (password) means losing access to your file.
- Recovering your data after a drive failure can be complicated by encrypted data.

- Your organization might have specific policies that require or disallow some or all forms of encryption.

- Your country might have specific laws regarding the use of encrypted data or the transportation of encrypted files across borders.

Encryption Is Not Security

Encryption can provide a level of access control in appropriate situations. However, encryption is not an inclusive security solution. If you have extremely sensitive data, you need a comprehensive set of security policies designed to balance between your access and restrictions. You should consider restricting physical access to the machine containing your data, limited remote access to the data, frequent backups on to a media which can be secured in a similar manner to your hardware. We recommend consulting with data security professionals and have them design an appropriate system for your needs. Also, ensure that frequent audits are conducted to verify that protocols are being followed.

Appendix A: Additional Test Data Statistics

The following tables show additional statistics related to the tests:

The SAS System

The MEANS Procedure

Analysis Variable : random

N	Mean	Std Dev	Minimum	Maximum
2000000000	2147486312	1239858134	0	4294967290

The SAS System

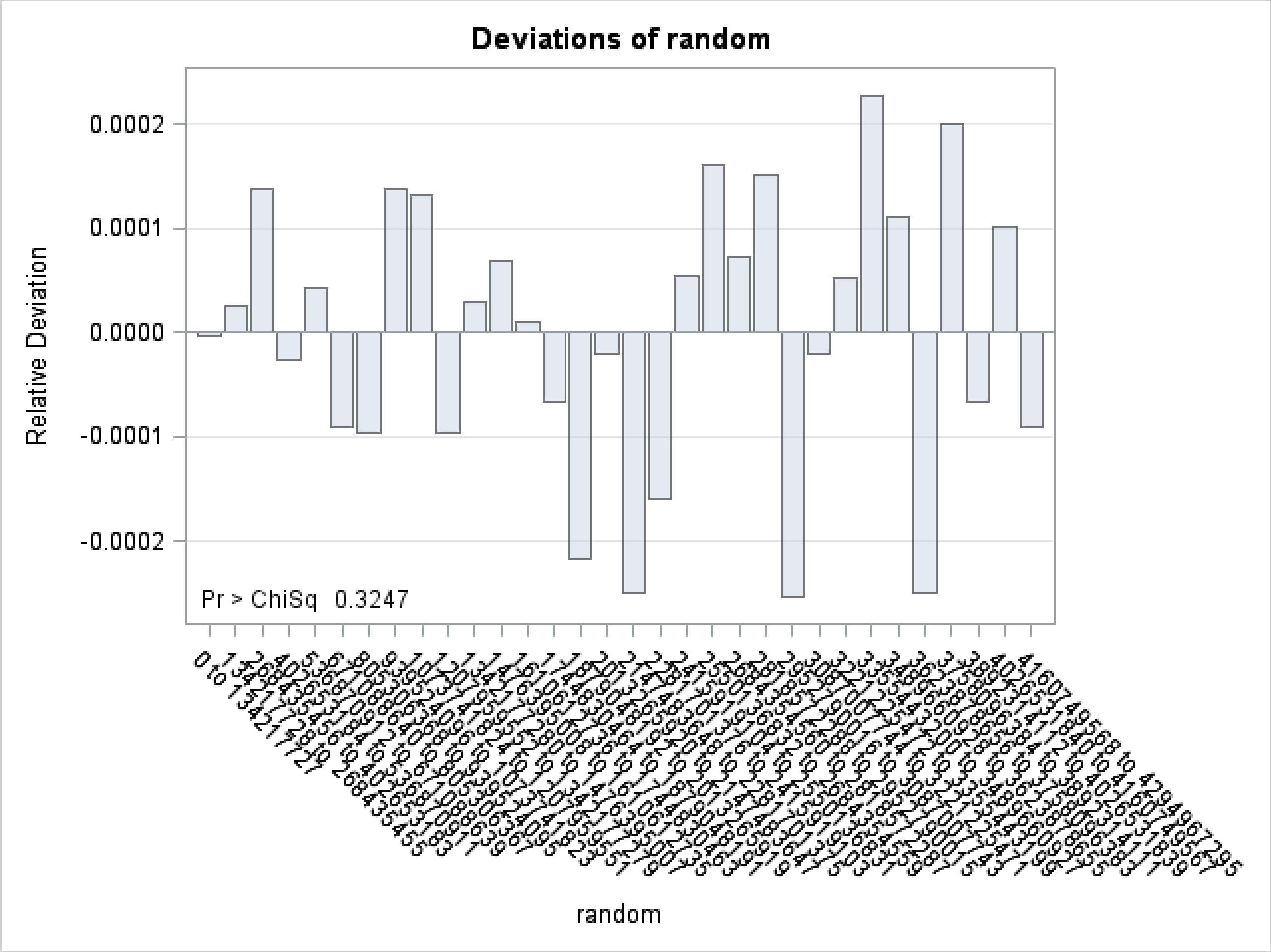
The FREQ Procedure

random	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0 to 134217727	62499789	3.12	62499789	3.12
134217728 to 268435455	62501516	3.13	1.25E8	6.25
268435456 to 402653183	62508591	3.13	1.8751E8	9.38
402653184 to 536870911	62498388	3.12	2.5001E8	12.50
536870912 to 671088639	62502598	3.13	3.1251E8	15.63
671088640 to 805306367	62494302	3.12	3.7501E8	18.75
805306368 to 939524095	62493958	3.12	4.375E8	21.87
939524096 to 1073741823	62508565	3.13	5.0001E8	25.00
1073741824 to 1207959551	62508191	3.13	5.6252E8	28.13
1207959552 to 1342177279	62493919	3.12	6.2501E8	31.25
1342177280 to 1476395007	62501836	3.13	6.8751E8	34.38
1476395008 to 1610612735	62504312	3.13	7.5002E8	37.50
1610612736 to 1744830463	62500603	3.13	8.1252E8	40.63
1744830464 to 1879048191	62495836	3.12	8.7501E8	43.75
1879048192 to 2013265919	62486426	3.12	9.375E8	46.87
2013265920 to 2147483647	62498656	3.12	1E9	50.00

random	Frequency	Percent	Cumulative Frequency	Cumulative Percent
2147483648 to 2281701375	62484372	3.12	1.0625E9	53.12
2281701376 to 2415919103	62490055	3.12	1.125E9	56.25
2415919104 to 2550136831	62503375	3.13	1.1875E9	59.37
2550136832 to 2684354559	62510056	3.13	1.25E9	62.50
2684354560 to 2818572287	62504535	3.13	1.3125E9	65.62
2818572288 to 2952790015	62509392	3.13	1.375E9	68.75
2952790016 to 3087007743	62484133	3.12	1.4375E9	71.87
3087007744 to 3221225471	62498712	3.12	1.5E9	75.00
3221225472 to 3355443199	62503231	3.13	1.5625E9	78.12
3355443200 to 3489660927	62514209	3.13	1.625E9	81.25
3489660928 to 3623878655	62506968	3.13	1.6875E9	84.38
3623878656 to 3758096383	62484474	3.12	1.75E9	87.50
3758096384 to 3892314111	62512568	3.13	1.8125E9	90.63
3892314112 to 4026531839	62495827	3.12	1.875E9	93.75
4026531840 to 4160749567	62506328	3.13	1.9375E9	96.88
4160749568 to 4294967295	62494279	3.12	2E9	100.00

Chi-Square Test
for Equal Proportions

Chi-Square	34.0090
DF	31
Pr > ChiSq	0.3247



Sample Size = 2000000000

Appendix B: Sample of Input Text

The following is a sampling of the input text strings we used in the tests:

3765942325
1347220684
2358711933
2171620557
3071066184
237891836
3600398618
1888748332
4138421474
73125830
3685910070
147201046
2078966022
3286146719
2995197989
986991375
2809955479
1904094526
1124967452
253850862
3701293869
1600685964
637458566
1198030078
3396783799
1245678434
1237318208
4185480118
732763701
1634411298
764402738
3094541832
2613339292
1294169298
3487549375