·研究原著· 文章编号 :1000-2790(2004)19-1819-03

基于 CryptoAPI 的生物医学真随机数的生成器

黄 枫,申 洪 (第一军医大学病理学教研室,广东 广州 510515)

Study on cryptoapi based true random number generator

HUANG Feng, *SHEN Hong*
Department of Pathology, First Military Medical University, Guangzhou 510515, China

【Abstract】 **AIM**: To establish a true random number generator based on Microsoft CryptoAPI. **METHODS**: The random numbers were obtained by programming under Microsoft Visual C + + 6.0, via CryptGenRandom function, following the installation of Intel Security Driver (ISD) on an Intel 815E chipset based personal computer. **RESULTS**: The author tested the generator with 500 random numbers in NIST FIPS 140-1 and χ^2 R-Squared test and the result showed that it fitted the need of independent and uniform distribution. The author also compared the random numbers produced by Intel RNG based true random number generator and those from the random number table with statistics parameters, by using the same amount of 7500 random numbers in the same value domain. The result showed that SD, SE and CV of Intel RNG based random number generator were smaller than those of random number table. The result of u test of two CVs indicated that there was no difference between the two methods. **CONCLUSION**: Intel CryptoAPI based random number generator can produce high quality random numbers with independent and uniform distribution features.

【Keywords】 CryptoAPI; Intel RNG unit; true random number; FIPS 140-1; random number table

【摘 要】 **目的**:构建基于 Microsoft CryptoAPI 的真随机数生成器. **方法** 在 Intel 815E 芯片组的个人电脑上安装 Intel Security Driver(ISD)后,使用 Microsoft Visual C + + 6 编程,通过 CryptGenRandom 函数获取真随机数. **结果**:生成的 500 个随机数通过了 NIST FIPS 140-1 和 χ^2 拟合优度检验($\alpha = 0.05$),表明本方法所生成的随机数满足独立性和分布均匀性的要求. 生成 7500 个随机数经域值变换后与随机数表中的同等数目的随机数进行了统计学比较,结果显示前者的均值偏移、*SD*, *SE* 和 *CV* 均小于后者. **结论**:基于 CryptoAPI 的真随机数生成器可以生成满足独立性和分布均匀性的真随机数.

【关键词】 CryptoAPI;Intel RNG Unit;真随机数;NIST FIPS 140-1;随机数表
【中图号】 Q212.1 【文献标识码】 A

收稿日期 2004-05-18; 修回日期 2004-08-21
通讯作者 申 洪. Tel. (020)61648227 Email. hongshen@fmmu. com
作者简介 黄 枫(1974-),男(汉族),四川省江安县人. 博士生(导师 申 洪). Tel. (020)61648223 Email. mbira@vip. sina. com

0 引言

随机数在生物医学中使用非常广泛. 通过随机数生成器(random number generator, RNG)可以获取随机数字. 基于硬件源(例如热噪声和电流噪声等)的真随机数生成器(true random number generator)[1]具有最好的统计学特性,但是目前的硬件随机数生成器(例如 ComScire QNG 等)价格昂贵. 微软公司(Microsoft) CryptoAPI 系统构架分为两层:面向客户系统的界面层和下面的真正提供密码算法的计算层. 底层的每个驱动可以称之为"Cryptographic Service Provider"(CSP). 微软公司在发布 CryptoAPI 的时候就已经内置了若干种 CSP,即 Microsoft Base Cryptographic Provider(MBCP). CryptGenRandom 函数就位于 MBCP 中的 Microsoft RSA Base Provider 中. 对于能提供 RNG 的芯片组,在安装相应驱动,例如 Intel RNG 的 Intel Security Driver(ISD)[2]和 VIA 的 VIA Random Number Generator(VRNG)[3]等之后,就能生成相应的 CSP,供 CryptoAPI 体系调用. 因此使用 CryptGenRandom 函数所获得的随机数是真随机数.

1 材料和方法[4]

1.1 材料 在兼容机(intel PIII 667, 主板:ASUS TUSL2-C,815E 芯片组,512M KingMax PC133 RAM)上使用 Microsoft Visual C + + . net 2003 编制了计算机程序.

1.2 方法 使用 CryptGenRandom() 函数获取随机数(Fig 1). 使用的伪代码如下:#define WIN32_WINNT 0x400; #include < wincrypt. h > ; #include < stdio. h > ; #include iscp4ms. h. HCRYPTPROVhProv; DWORDrandomNumber = 0; DWORDrandomLength = 4; //获取句柄. If (! CryptAcquireContext (&hProv, NULL, INTEL_DEF_PROV, PROV_INTEL_SEC, 0))

{printf("No Found!\n",GetLastError());return FAIL;}//获取随机数(4 位). If (CryptGenRandom (hProv, randomLength, (BYTE *) &randomNumber)! = TRUE) {//释放 CSP 句柄. CryptReleaseContext(hProv,0);Return FAIL;}//释放 CSP 句柄. CryptReleaseContext(hProv,0);

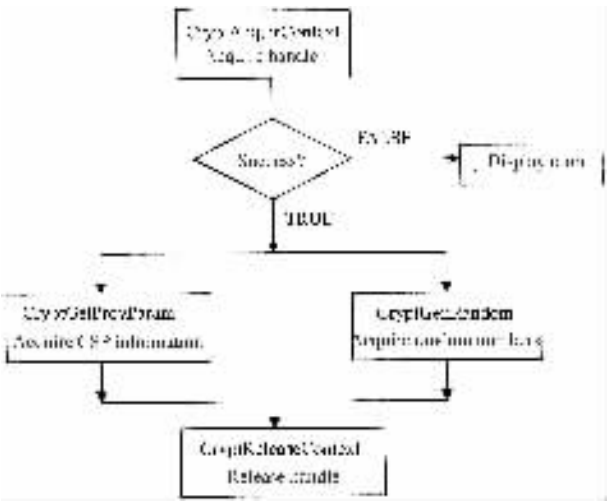


Fig 1 Working flow of CryptoGenRandom Function

图 1 CryptGenRandom 函数的使用流程

2 结果

根据最少需要 80Mbit 连续 RNG 输出的原则[5],我们生成了 100Mbit 连续随机数,部分数据如下((0,1)域,4 位小数)(Tab 1). 随机数的随机性检验常用 χ^2 检验(罕用 KS 检验和经验检验)分布均匀性检验

表 1 使用 CryptoAPI 所生成的 200 个随机数

Tab 1 200 random numbers produced by CryptoAPI based RNG

0.5413	0.1894	0.6428	0.029	0.0905	0.8851	0.6289	0.4238	0.1845	0.1899
0.467	0.0558	0.5113	0.6953	0.9343	0.7295	0.1868	0.9445	0.7218	0.3521
0.3149	0.5384	0.8299	0.8604	0.114	0.0947	0.1627	0.9524	0.9923	0.9799
0.5799	0.8858	0.4554	0.4163	0.6143	0.562	0.9068	0.0413	0.9863	0.2157
0.1183	0.8047	0.1775	0.4059	0.6049	0.6594	0.0638	0.9236	0.8752	0.3424
0.8178	0.9181	0.435	0.6548	0.056	0.9495	0.7178	0.648	0.551	0.2123
0.5351	0.6792	0.2462	0.2564	0.1934	0.6735	0.3066	0.613	0.5735	0.0458
0.3045	0.5417	0.2232	0.8508	0.9841	0.4269	0.4306	0.437	0.2814	0.2244
0.5732	0.092	0.2946	0.1113	0.5957	0.877	0.3523	0.7533	0.1582	0.2834
0.3408	0.4346	0.2796	0.8054	0.9431	0.1089	0.8517	0.0347	0.2251	0.8354
0.1146	0.8895	0.9985	0.1009	0.6021	0.6632	0.304	0.6968	0.0413	0.0325
0.5428	0.7932	0.9829	0.2567	0.5574	0.5179	0.219	0.2344	0.1626	0.8524
0.2635	0.214	0.0705	0.4143	0.9957	0.0807	0.9573	0.9198	0.7527	0.8352
0.7828	0.193	0.2519	0.2441	0.7295	0.9289	0.7553	0.5859	0.6617	0.0588
0.4068	0.0269	0.4355	0.7149	0.2547	0.8963	0.5561	0.8247	0.8849	0.9605
0.9962	0.0894	0.3441	0.7227	0.5976	0.2113	0.7437	0.7959	0.9856	0.9307
0.2779	0.3223	0.8729	0.7553	0.9741	0.6796	0.0901	0.4671	0.2371	0.7693
0.5403	0.7685	0.5104	0.0566	0.729	0.4853	0.6518	0.5732	0.6525	0.6825
0.908	0.4714	0.4406	0.6077	0.4731	0.4168	0.6261	0.6162	0.3332	0.4911
0.9266	0.3215	0.4806	0.3932	0.9373	0.2518	0.5384	0.0668	0.8878	0.3519

我们使用 χ^2 拟合优度检验法、独立性常用游程检验法检验. 我们使用美国国家技术标准局(NIST)的 FIPS(Federal Information Processing Standards Publication)140-1[6] 对所生成的随机数进行了测试,结果显示,Intel 随机数生成器通过了统计学 FIPS140-1 测试,证明本实验所生成的随机数符合随机数的特征,通过了独立性检验、参数检验和组合规律检验(Tab 2).

表 2 FIPS140-1 测试结果

Tab 2 Test result of FIPS 140-1

Run#	Monobit Test	Poker Test	Runs Test	Long Run Test
1	$X=9,943$	$X=12.64$	Passed	Passed
2	$X=10,078$	$X=21.63$	Passed	Passed
3	$X=9.988$	$X=10.57$	Passed	Passed
4	$X=9.834$	$X=17.34$	Passed	Passed
5	$X=9.864$	$X=22.07$	Passed	Passed

为了验证所生成的随机数分布的均匀性,我们再用 χ^2 拟合优度检验法对这些数作分布均匀性检验. 假设 H_0:$r_1,r_2,r_3,\cdots\cdots,r_n$ 为均匀总体的随机样本. 将样本 $r_1,r_2,r_3,\cdots\cdots,r_n(n=500)$ 的取值范围分布在 m 个($m=10$)等宽的区间,用 $[\frac{i-1}{m},\frac{i}{m})$ ($i=1,2,3\cdots,m$)来表示第 i 个小区间,即分成 [0.0,0.1),[0.1,0.2),[0.2,0.3),[0.3,0.4),[0.4,0.5),[0.5,0.6),[0.6,0.7),[0.7,0.8),[0.8,0.9),[0.9,1.0),共 10 个区间. 设 $\{r_j\}$ ($j=1,2,3\cdots,m$)落入每个小区间的数目为 n_i ($i=1,2,3\cdots,m$). 根据假设,$\{r_j\}$ 落入每个小区间的概率为 $\frac{1}{m}$,第 i 个小区间的理论频数 $u_i=\frac{n}{m}$ ($i=1,2,\cdots m$),统计量 $V=\sum_{i=1}^{m}\frac{(n_i-u_i)^2}{u_i}=\frac{m}{n}\sum_{i=1}^{m}(n_i-\frac{n}{m})^2$ 渐进服从 $\chi^2(m-1)$. 使用 Origin 7.5(SR1)对 500 个随机数进行频数统计,得 $m1\sim10$ 的频数为 52,53,32,49,44,43,59,48,58,62. 计算统计量值:

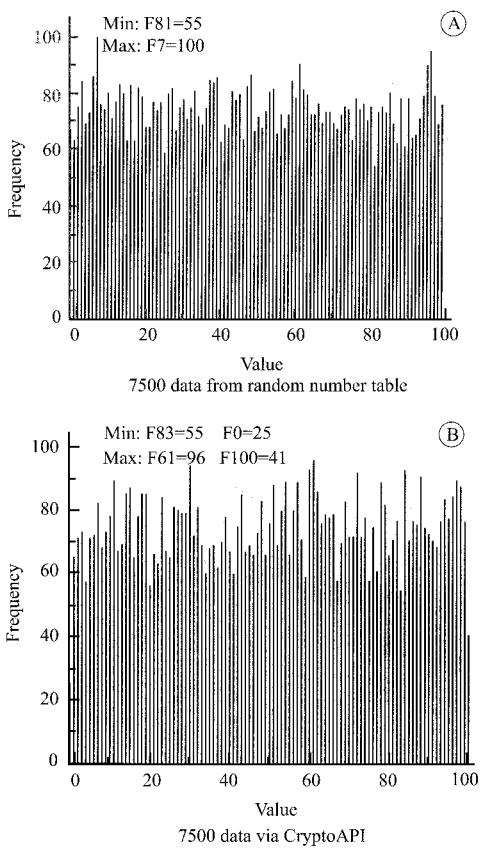
$$V=\frac{m}{n}\sum_{i=1}^{m}(n_i-\frac{n}{m})^2=\frac{1}{50}\sum_{i=1}^{10}(n_i-50)^2=14.32$$

给定显著性水平 $\alpha=0.05$,取 $m=10$,则自由度 $v=m-1=9$. 查 χ^2 分布临界值为 18.307. 可知,$V<18.307$,所以假设 H_0 成立. r_n 为来自均匀分布的总体的随机样本.

Microsoft 公司的(Trusted Computing Platform Al-

liance)[7,8]是下一代操作系统安全构架,因此使用基于 Windows CryptoAPI 函数的随机数生成器具有广泛的使用性,是一种优秀的随机数生成器.

在生物医学上,随机数的性能取决于它的均匀性和距离理论频数的偏移程度. 我们选取了《卫生统计学》[北京 :人民卫生出版社,1978 :215 - 220]后面随机数表中的所有随机数字(7500 个,[0,99)). 我们将基于 Intel RNG 的随机数生成器所生成的 7500 个随机数乘以 100 后,截掉尾数,使其从(0,1)域转换成[0,100]域. 由于截尾(使用 round()函数)有四舍五入的问题,因此其界值 1 和 100 不作为统计之用. 对随机数表(记为 A 组)和随机数生成器所生成的随机数(记为 B 组)使用 SPSS 11.5 统计频数后,用 Origin7.5(SR1)作频数图(Fig 2). 为了检验两个变异系数之间有无差别,常用检验. 由于本例中,测量值与真值(在本例中即为均值的偏移程度)接近于 0(分别是 0.6316 和 0.76173),变异系数很小(<0.5%),因此检验以平均绝对误差、标准差等为准. 以上结果显示,均值偏移 :A < B ;*SD* 和 *SE* :A > B ;虽然两者的差别没有显著性(*CV* 差别为 0.00041%),但是 B 组的频数离散度和最大偏移均小于 A 组,因此我们认为基于 CryptoAPI 的随机数生成所产生的随机数比基于随机数表的随机数表中的随机数具有更好的统计学性能.



A :A group ;B :B group.

Fig 2 Frequency count chart of 7500 random numbers

图 2 7500 个随机数的频数图

3 讨论

以上真随机数生成方法可以用于任何能提供硬件 CSP 的操作系统中. 既可用于具有 Intel RNG 的 Intel 芯片组(包括 810, 815, 820 和 840 芯片组系列)的个人计算机 ;也可以用于具有 RNG 单元的 VIA 和 AMD 等 CPU 芯片的个人计算机,具有较好的普及性. 另外这种生成方法使用真正的硬件随机数源,而非依靠拟蒙特卡罗方法(例如算法),因此具有极高的随机数性能. 在生物医学上常用随机数表法来获取数据. 使用本研究中的基于 CryptoAPI 的真随机数生成器基于硬件发生器源,所生成的随机数具有不可预测性,独立性和分布的均匀性. 在具有 RNG 单元的 CPU,包括 Intel , AMD 和 VIA 等的个人电脑上均可以使用,能够得到广泛的使用. 具有一定的实用性.

【参考文献】

[1] Gary M and John V. 使您的软件运行起来 :消除偏差. [URL] http ://www-900. ibm. com/developerWorks/cn/security/beating/index. shtml.

[2] Intel Security Driver (ISD) :[URL] http ://developer. intel. com/design/software/drivers/platform/3463/isecdrv_enu. htm.

[3] VIA, The VIA PadLock Advanced Cryptography Engine (ACE) :[URL] http ://www. via. com. tw/en/padlock/padlock_hardware. jsp.

[4] Intel Inc. Intel 82802 firmware hub :random number generator. Intel Inc. , [URL] http ://www. intel. com.

[5] Intel Platform Security Division. The Intel Random Number Generator. Intel Inc. , [URL] http ://www. intel. com/design/security/rng/rng. htm.

[6] NIST, fact sheet on digital signature standard :[URL] http ://www. nist. gov/public_affairs/releases/digsigst. htm.

[7] Intel Platform Security Division. The Intel Random Number Generator. Intel Inc. , http ://www. intel. com/design/security/rng/rng. htm.

[8] Intel :Intel Trusted Computing :Integrated Security that Starts as the Platform level [URL] : http ://www. intel. com/home/scenes/stories/trustedcomputing. htm.

编辑 许昌泰