第16卷 第11期
2006年11月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.11
Nov. 2006

Linux内核随机数产生器的设计与实现

李正平,徐 超,陈军宁,谭守标
(安徽大学 电子科学与技术学院,安徽 合肥 230039)

摘 要:随机数产生器在科学研究的许多领域具有重要意义。Linux内核利用系统环境噪声的随机特性,实现了一个高强度的随机数产生器。以Linux 2.6.10内核源代码为基础,分析了其内核随机数产生器的设计。介绍了随机数产生器的基本原理和设计思想,并对其具体的实现细节,如输入输出接口作了详细的阐述,最后讨论了系统启动导致随机数质量下降的问题以及Linux内核随机数产生器的解决方法。

关键词:随机数产生器;Linux内核;熵
中图分类号:TP316;TP301 **文献标识码**:A **文章编号**:1673-629X(2006)0087-02

Design and Implementation of Linux Kernel Random Number Generator

LI Zheng-ping, XU Chao, CHEN Jun-ning, TAN Shou-biao
(School of Electronic Science & Technology, Anhui University, Hefei 230039, China)

Abstract: Random number generator plays an important role in many research fields. By collecting noise in the environment of machine, Linux kernel implements a strong random number generator. Based on the kernel source code ver 2.6.10, this paper analyzes the design and implementation of kernel random number generator. First, an introduction to basic principle and design idea of the generator is presented, and then gives a detailed analysis about the implementation, including the exported input and output interface. Finally, a discussion about the effect of system startup and the adopted solution of Linux kernel random number generator is given.

Key words: random number generator; Linux kernel; entropy

0 引 言

随机数在许多领域都有重要应用,如Monte Carlo模拟、密码学和网络安全。随机数的质量直接关系到网络安全系统的可靠性和安全性,关系到Monte Carlo模拟结果的可信度[1]。自从计算机诞生起,寻求用计算机产生高质量的随机数序列的研究就一直是个长期受到关注的课题。Linux内核从1.3.30版本开始实现了一个高强度的随机数发生器,文中根据Linux 2.6.10内核的源代码[2,3],详细分析该随机数产生器的设计与实现。

1 基本原理

Linux内核采用熵来描述数据的随机性。熵(entropy)[4]是描述系统混乱无序程度的物理量,一个系统的熵越大则说明该系统的有序性越差,即不确定性越大。在信息学中,熵被用来表征一个符号或系统的不确定性,熵越大表明系统所含有用信息量越少,不确定度越大。计算机本身是可预测的系统,因此,用计算机算法不可能产生真正的随机数。但是机器的环境中充满了各种各样的噪声,如硬件设备发生中断的时间,用户点击鼠标的时间间隔等是完全随机的,事先无法预测[3,5]。Linux内核实现的随机数产生器正是利用系统中的这些随机噪声来产生高质量随机数序列。

内核维护了一个熵池(entropy pool),用来收集来自设备驱动程序和其它来源的环境噪声。理论上,熵池中的数据是完全随机的,可以产生真随机数序列。为跟踪熵池中数据的随机性,内核在将数据加入池的时候将估算数据的随机性,这个过程称作熵估算。熵估算值描述池中包含的随机数位数,其值越大表示池中数据的随机性越好。而当从熵池中提取随机数的时候,内核会同时降低熵估算值[6]。

2 设计与实现

Linux内核随机数产生器在/drivers/char/random.c[2]中作为字符设备实现。其模块初始化函数rand_initialize()调用create_entropy_store()分别创建名为random_state的缺省熵池,一个名为sec_random_state和一个名为urandom_state的熵池。熵池用struct entropy_store[2]来表示,定义为:

```
struct entropy_store {
  struct poolinfo poolinfo;
  _u32 * pool;
  const char * name;
```

收稿日期:2006-02-24
作者简介:李正平(1979-),男,安徽宣城人,副教授,研究方向为操作系统原理、核技术应用、嵌入式系统。

```
spinlock_t lock_cacheline_aligned_in_smp;
unsigned add_ptr;
int entropy_count;
int input_rotate;
};
```

其中 name 表示熵池的名称，entropy_count 为当前熵池的熵估计值，lock 自旋锁用来同步对熵池的访问；pool 指向一个内存缓冲区，该缓冲区保存熵，每次有新的随机数加入熵池时，内核会将它们存入 pool 缓冲区，其位置由 add_ptr 确定；poolinfo 包含熵池的有关信息，包括熵池的大小，即 pool 缓冲区的大小；此外，poolinfo 结构还包含 5 个 tap 与 input_rotate 一起实现对池中数据的混合(mixing)，从而保证池中数据更难被攻击者破解。

内核实现了一系列接口函数，用于获取系统环境的噪声数据，并加入熵池，分别是：

```
void add_interrupt_randomness(int irq);
void add_keyboard_randomness(unsigned char scancode);
void add_mouse_randomness(_u32 mouse_data);
void add_disk_randomness(struct gendisk * disk);
```

其中 add_interrupt_randomness()函数利用设备两次中断的间隔时间作为噪声源将随机数据加入熵池。要使设备的中断作为系统噪声，必须用 SA_SAMPLE_RANDOM 标志注册其中断服务程序[7]。这样，每当设备发生中断时，中断系统会自动调用 add_interrupt_randomness()将熵加入熵池。Add_keyboard_randomness()将按键的扫描码和两次按键之间的时间间隔作为噪声源；而 add_mouse_randomness()则利用鼠标位置和连续两次鼠标中断时间间隔填充熵池；最后 add_disk_randomness()函数则以连续两次磁盘操作之间的间隔产生随机数。

上面的函数最终都是通过调用 add_timer_randomness()函数将熵加入熵池的。Add_timer_randomness()首先估算所加数据的熵，再调用 batch_entropy_store()函数将数据加入熵池。

为了避免由于中断的延迟过长从而导致系统性能的下降，batch_entropy_store()并不直接将熵加入熵池，而是将其加入队列中。当队列长度达到一定长度后，由 keventd 内核线程通过调用 batch_entropy_process()函数将队列中的熵加入池中。

Batch_entropy_process()函数枚举队列中的每个熵，对每个熵调用 add_entropy_words()函数将其加入熵池，但它并不更新熵池的熵估算值。为此，batch_entropy_process()对每个熵调用完 add_entropy_words()后，立刻调用 credit_entropy_store()函数更新熵估算值。

相对于输入接口，Linux 内核同样实现了一系列输出接口，用来向用户空间或内核其他模块输出随机数序列。

其中 void get_random_bytes(void * buf, int nbytes)函数用于向内核其他模块输出随机数。它从熵池中返回 nbytes 个字节的随机数序列存入 buf 中。该函数优先从 urandom_state 池中返回随机数，如果不存在 urandom_state 熵池则从 sec_random_state 池返回数据，只有在前两者都不存在的时候才从缺省池 random_state 返回随机数。get_random_bytes()总是返回 nbytes 个字节的随机数序列，即使熵池的熵估算值为 0。

此外，内核提供了两个字符设备：/dev/random 和/dev/urandom，其 read 函数(random_read()和 urandom_read())用于向用户模式程序输出随机数序列。上层应用程序可以通过 read 系统调用，从它们获取随机数序列。相对于/dev/urandom 接口，/dev/random 输出的随机数序列质量更高，适合于高强度的加密算法。/dev/urandom 总是返回所请求的随机数序列，无论熵池的熵估算值是否为零；而/dev/random 则只返回熵估算值所允许的最长的随机数序列，当熵估算值为零时，请求将被阻塞，直到熵估算值增大到一定域值。

上述输出接口最终均是通过调用 extract_entropy()函数输出随机数序列。Extract_entropy()函数使用 SHA 或 MD5 算法散列(Hash)熵池，将散列后的结果作为随机数序列输出给用户使用，这样避免了直接访问熵池中的内容。由于从 SHA 或 MD5 算法散列的结果反推原始数据的可能性几乎为零，所以这种设计极大地提高了安全性，攻击者无法直接访问熵池，也无法根据过去的随机数序列预测将来的序列。

当系统启动的时候，由于启动过程是个确定的可预测的过程，这种情况下，熵池的熵值将非常小，导致产生的随机数序列质量下降，从而给攻击者破解的可能。为了克服系统启动过程的可预测性带来的影响，Linux 操作系统在系统关机的时候保存当前熵池的内容，当系统下次启动的时候恢复上次关机时熵池的数据，这样就有效增加了熵池的熵估算值，避免了随机数序列质量的下降。

3 小 结

随机数产生器在科学研究的许多领域具有重要意义。通过收集系统环境中的噪声，如设备中断的时间间隔、用户的按键操作及随机的硬盘读写操作，Linux 内核实现了一个强大的随机数产生器，用来向内核其他部分(如网络模块)和用户模式应用程序提供高质量的随机数序列，满足加解密算法、网络安全、Monte Carlo 模拟等方面需求。

参考文献：

[1] Householder A S. Monte Carlo Method[M]. New York: American mathematical society, 2000.

[2] Linux Kernel Source Code ver 2.6.10[EB/OL]. 2005. http://www.kernel.org.

[3] 倪继利. Linux 内核分析及编程[M]. 北京：电子工业出版社，2005.

[4] Cover T, Thomas J. 信息论基础[M]. 北京：清华大学出版

(下转第 91 页)

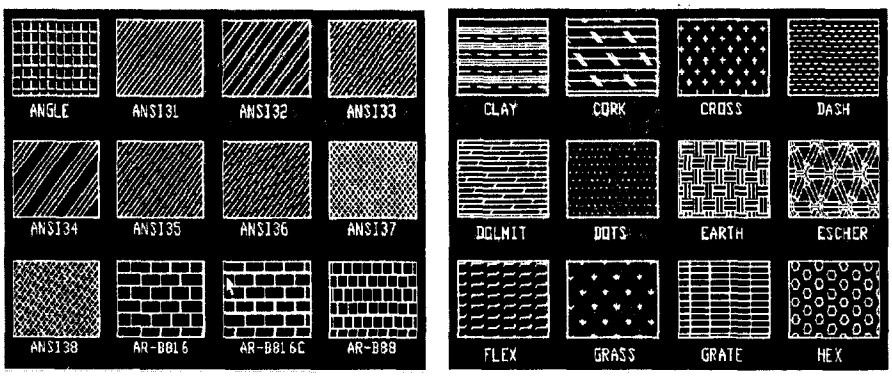

图 1　博士 CAD 系统的部分剖面线填充图案

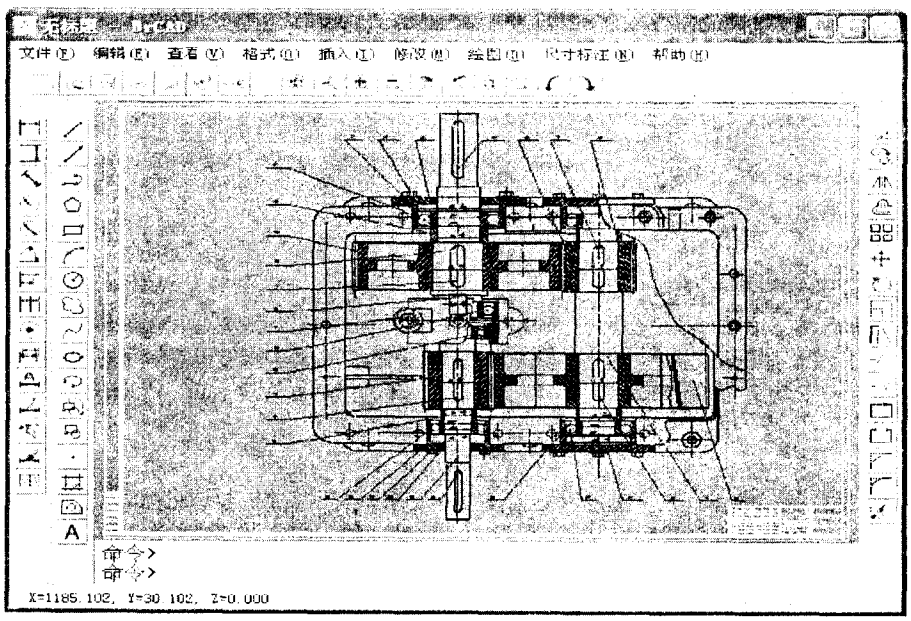

图 2　博士 CAD 系统生成的工程图纸(其中有大量的剖面线)

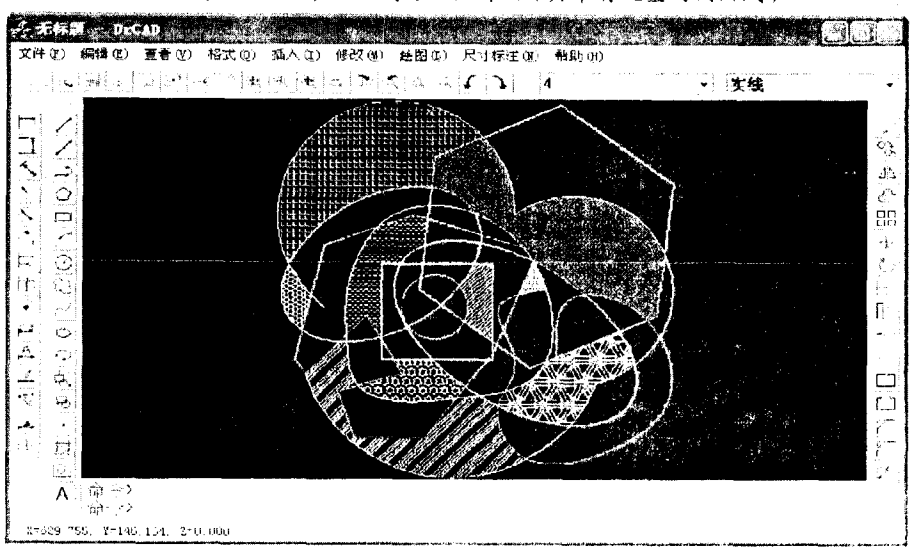

图 3　博士 CAD 系统在一个任意复杂的拓扑区域内生成的各种不同类型的剖面线

45, .530330086,0, 0,.75

* ANSI35, ANSI Firc brick, Refractory material

45, 0,0, 0,.25

45, .176776695,0, 0,.25, .3125, -.0625,0, -.0625

图 1～图 3 是博士 CAD 系统的部分样图。

4　结　论

文中所论述的 FDHP 算法无需对边界进行复杂而又不稳定的搜索,通过种子点对在其虚拟像空间中进行区域填充以快速得到其所具的最小拓扑区域,从而确定起边界,在其区域内进行剖面线绘制。该算法新颖、高效、快速并具有非常好的鲁棒性等优点。

该方法已在博士 CAD 系统中予以实现,从实际应用的情况来看,效果十分良好,是一种优秀的剖面线填充算法,可广泛应用于任何 CAD 及相关的图形系统中。

参考文献:

[1] 彭群生,鲍虎军,金小刚. 计算机真实感图形的算法基础[M]. 北京:科学出版社,1999.

[2] 孙家广,杨长贵. 计算机图形学[M]. 北京:清华大学出版社,1997.

[3] 唐荣锡. CAD/CAM 技术[M]. 北京:北京航空航天大学出版社,1994.

[4] 王志强,肖立瑾. 一种高效可靠的剖面线参数化绘制技术[J]. 机械科学与技术,1998,17(4):673-675.

[5] 陈正鸣,吴玉光. 剖面线的快速绘制技术[J]. 计算机工程与设计,1999,20(5):47-51.

(上接第 88 页)

社,2003.

[5] Bovet D P, Cesati M. Understanding the Linux Kernel[M]. 2nd ed. 北京:中国电力出版社,2004.

[6] Love R. Linux Kernel Development[M]. 2nd ed. 北京:机械工业出版社,2005.

[7] Corbet J, Rubini A. Linux Device Driver[M]. 3rd ed. 南京:东南大学出版社,2005.