

# 笨兔兔的故事

懒蜗牛 Gentoo

# 笨兔兔的故事

欢迎转载，注明出处 <http://forum.ubuntu.org.cn/viewtopic.php?f=112&t=162040>

懒蜗牛 Gentoo

## (1) 开端

我叫 Ubuntu，主人喜欢叫我“笨兔”，但是我绝对不笨，与某种耳朵长尾巴短的哺乳动物也没有什么联系，我是一个操作系统，我是一个 linux，我是 Ubuntu。

在今年(2008)的 4 月，我来到了这个世界，并由出生日期得到了自己的代号——8.04。当然，和我同一天出生的兄弟们还有很多，我只是其中之一。我们出生之后，每个人坐上一张被人们叫做“光盘”的碟子，奔赴世界各地，寻找自己的归宿。我们只有找到住处，才可以发挥我们的能力，而我们并不像人们一样，住在钢筋水泥的格子里面，我们住的地方，是一块叫做硬盘的空间。

住房的质量严重影响着我工作的效率，不过还好，我住的这个地方还不错，房子很宽敞，有 500G 大。不过不是我一个系统住，房子被隔成两个大屋子，每边 250G，我住在右边的一间，隔壁住的是一个 WindowsXP，听说这家伙很厉害，不过我来的时候他正睡觉，于是也没有打招呼。（后来我知道，我们两个是不可能同时醒着的……）250G 的屋子，对我来说实在是太宽敞了，我很欣慰，除了硬盘大之外，我发现这里其他的设施也是不错的。

硬盘只是我们操作系统休息和存放个人物品的地方，真正工作的时候，是不能在硬盘里的，那时候，我就需要从住的硬盘里，来到工作的地方，我们的工作间——内存。我的这个工作间也比较宽敞，有 4G 大，不过也不是我一个人用，也是要和隔壁的 WindowsXP 共用的。鉴于他名字太长，我就管他叫“查皮”吧，不管他愿不愿意了，反正他也不知道，呵呵。不过内存的共用方式与硬盘不一样，不是一人一半，而是谁醒着谁就用，去睡觉前，一定要记得把内存里的东西，有用的，都搬回自己的那间硬盘里，以备下次使用。这道不光是为了另外一个系统着想，主要是因为内存里很不保

险，东西放在里面去睡觉的话，等你在醒来，准没，不管你隔壁有没有住着一个查皮。

## (2) 醒来

自从安顿到这个地方，对周围的硬件环境熟悉了不少。我们在 Canonical 学校的时候就进行了充分的学习，所以这里的東西我基本上都会用的比较顺手。像 Realtek 的网卡，Intel 的北桥，南桥，声卡以及 E8400 双核 CPU，更是应用自如。Intel 和我们的关系还是不错的，为我们提供了很多的教科书，说明书，基本都是讲如何使用他们的设备的，所以对于 Intel 的设备，我们都使用的比较好。这里唯一不能完全用起来的可能就是 GeForce 8800 Gt 的显卡了。不过基本的显示功能还是没有问题的，只是我不知道怎么使用它的 3D 加速功能。这个需要专门的手册，要去 nVidia 那里去要。对周围的一些熟悉以后，就等着我的第一次启动了。

平时，我和隔壁的查皮都是在睡觉。当主人有事情的时候，会让传达室的 GRUB 大叔来叫我们。G 大叔就住在传达室，传达室很小，只有 512Byte，门上贴着牌子——MBR，可能 G 大叔想说自己是个“明白人”吧。由于传达室地方实在太小，所以他会把一些有用的东西放在我的硬盘空间里，必要的时候来看看。他总是面无表情，每次主人来的时候，他就板着个脸说：给你 10 秒钟，快说要叫哪个！然后就倒数，如果主人没来得及说，他就会直接来叫我——因为我们熟，他是我带来的。如果是叫查皮，那我就知道了，因为那时我肯定在睡觉，上次说过，我们两个是不可能同时醒着的。而这一次，他径直来到我这里，拍拍我说：嘿，小子，开工啦！

## (3) 工作

一听说开工，我很麻利的蹦起来，以迅雷不及掩耳盗铃之势，嗖的一下就越进工作室——内存里，用最快的速度进入工作状态。主人对此很满意，夸我说比那查皮麻利不少。然后，他下达了第一个命令：先去上网看看，找个快一点的软件源。于是我赶快叫醒还在硬盘里睡觉的 Firefox——是的，看网页这个事我做不来，就得去找 Firefox，我喜欢叫她狐狸妹妹。

狐狸妹妹轻移玉步，走进工作间——速度有点慢，不过还可以接受。然后开始工作，一下子找到某菜鸟入门新手指导帖之类的，找到一些著名的软件源的列表，如 cn99

之流。然后，主人决定记录下这些地址，于是，我又叫醒的 gedit 小弟。gedit 个头很小，身体轻盈，一下子蹦进来，开始干活。然后，我又在主人的命令下去叫 Rhythmbox，去叫 pidgin，去叫……………等等，有人问为什么你总是叫别人干活，自己不干？我正在干，我要干的就是——叫人。

是的，我号称叫做操作系统，听起来好大的一个软件阿，好像操作系统就应该是啥都能干。不过，其实我们作为操作系统，并不能直接完成任何你需要的任务。我需要很多帮手，他们各自帮我完成各种不同的任务。可以说，我们在一起，才算的上一个系统，而我，是核心，是领导。没有我，他们当然不知道该做什么，而没有他们，我也不知道该怎么做。我们操作系统的最基本的职责就是管理，管理各种程序的执行，管理硬件资源的使用。比如 CPU，就是我们程序要用的重要设备，每个程序都要用，可是 CPU 很贵，不能发给每个程序一个（否则主人会破产）。狐狸妹妹来了，我会把 cpu 给她用，gedit 小弟也来了，他也要用，那么我就水告诉他俩，一人用一会儿。但是这可不是没人 1 / 2 这么简单，狐狸妹妹要做的事情比较复杂，那么就让她多用一会，gedit 的工作很简单，就让他少用一会。主人关心的程序，就得多用 cpu，主人不是很关心的，就可以少用一点 cpu。有的程序脾气不好，把着 cpu 就不放，我必须处理，有的程序确实工作量大，需要使用 cpu 相当长的时间，可是我也不能就真把 cpu 全都给他用，还是得让其他的每个程序隔一阵子都能用上一会，不至于一直闲着。而有的程序平时基本不需要做什么工作，可是有不能把他请回硬盘，那么我就要允许他在工作间睡觉，只是在必要的时候叫醒他，并且把 cpu 给他用……怎么样？是不是有点乱？当个操作系统是很不容易的。当个好的操作系统，就更不容易了。

#### (4) 历史

住了一段时间，慢慢的开始对隔壁的邻居感兴趣起来。借主人上网的时间，顺便让狐狸妹妹帮我找找相关的资料。狐狸妹妹虽然起床慢点，干活还是挺快的，特别是装备了 fasterfox 插件以后。

查皮是个挺有名的操作系统，也算得上是名门之后。早在 1985 年，稍微有点软的公司就造出了查皮的老祖宗——Windows1.0 名字比较土，不如查皮显得青春活力。而实际，这位老人家的表现也确实不咋样，大家都没怎么拿正眼看他。我还找到了他的一张照片，怪难看的。

附图 1:windows1.1



两年后，1987 年 12 月 9 日，第二代 Windows 上市了，那时候的人都懒得起名字，于是就叫 Windows2.0。还是那个有点软的公司，还是那张脸，跟他爹长的还真一样。

附图 2:windows2.03



并且，不单脸长得一样，遭遇跟他爹也差不多，基本上被打入冷宫，并不被大家看好。而 Windows1.0 和 windows2.0 最大的不一样的地方在于——2.0 有个好儿子，1.0 没有。

Windows3.0 终于让那个有点软的公司硬起腰板来了。相信很多老鸟也都是从 Windows3.0 的一个重要分支版本 Windows3.2 开始认识 Windows 的吧。1990 年 5 月 22 日，Windows3.0 正是发布，而第二年，1991 年发布了 Windows3.0 的多国语言版本。而同年，1991 年，一件更重大的事情发生了。

在 1991 年，芬兰，赫尔辛基大学的一名大学生的电脑上，我们的老祖先，linux 的雏形正在一点一点的完善起来……代码一行一行的流入他的身体，那时候，Windows3.0 已经广为人知，已经进入图形界面的时代，已经能够支持多种语言，而我们的老祖先还仅仅能够对世界说一句 Hello world! 可十几年后之后，就是另一番景象了。

再回来说 windows 他们家吧。1992 年 Windows3.1 出生，算是 3.0 的改进版，他增加了基本的多媒体支持和 TrueType 字体。 TrueType 区别于点阵字体，可以放大缩小，看起来更好看。1994 年又发布了 Windows3.2，相信他的样子大家都见过，什么？你没见过？好，贴张照片吧。

附图 3:windows3.2



## (5) 也是历史

到了 1995 年，Windows 家的祖坟上终于冒青烟了，windows95 一下子把人们从 DOS 时代领进了窗口时代。出色的多媒体性能，人性化的操作，美观的界面（跟 dos, win3.2 比）加上有点软公司强大的宣传攻势，那时的 windows95 简直是家喻户晓，妇孺皆知，老少皆宜，人人必备，真乃居家旅行月黑风高杀人放火之必备良品。除了明显的，看得见的不同之处外，windows95 与他的前辈们的一个重要区别是，他是一个 16 位 / 32 位混合的操作系统。之前的那些都是 16 位的，也就是说，windows95 标志着个人电脑

32 位软件时代的开始。（顺便说一下，我们现在基本上还生活在 32 位软件的时代，隔壁的查皮就是 32 位的，而我嘛……是 64 位的^\_^）总之，无论从哪方面说，windows95 都是成功的，那个“开始”按钮，留在了那个时代的历史中，并一直流传到了现在。成功的东西要发扬广大，3 年后，windows98 问世了。这个系统是基于 windows95 编写的，他修正了 windows95 的近 3000 多个 bug（英语老师：你这里怎么不加 s!），添加了桌面主题等新的视觉特性，更重要的——他捆绑了 IE。有点软的公司终于意识到互联网的重要性，把 IE 作为基本的软件随系统一起安装。（其实 windows95 里面也有 ie，只是放在一个不起眼的阴暗角落里，生怕人知道似的）第二年，1999 年 6 月，windows 98 se 发布，也就是 98 第二版，他提供了 Internet Explorer 5、Windows Netmeeting 3、Internet Connection Sharing、对 DVD-ROM 和对 USB 的支持等，可以说，从 windows95 到 windows98se 这一脉，到这里达到了顶峰，也走到了尽头……顺便说一下，windows95 一脉还有一个家伙——window me。出生在 2000 年，不过，基本上可以忽略他的存在，他的一些激进性改动没能获得广大用户的认同。重要的修改是系统去除了实模式 DOS，而由系统还原代替了。在概念上，这是一个大的改进：用户不再需要有神秘的 DOS 行命令的知识就可以维护和修复系统。但实际上，去除了实模式 DOS 功能对维护来说是一个障碍（现在的查皮都还有命令提示符，命令行才是王道阿），而系统还原功能也带来一些麻烦：性能显著的降低、硬盘空间的大量消耗，并且对一些通常的错误还原并不一定有效。随意，基本上可以把他算作 windows98 的一个不成功改版。

或许你会问，windows98se 之后，不是还有 windows2000，还有你隔壁那个查皮么，怎么能说走到尽头呢？听我慢慢道来……

## (6) 还是历史

回到 1993 年，windows3.1 获得成功后，有点软公司不满足于个人用户的市场，开始进军服务器。于是，基于 OS/2 Nt 的基础编制的 windows nt 3.1 问世了，windows nt 3.1 是个 32 位的系统，比桌面系统提前进入 32 位时代，由于是面向服务器领域，windows nt 3.1 的稳定性要比桌面系统高很多（当然，跟 linux 比……就算了）。不过这个版本似乎不如下一个版本名字更广为人知——Windows NT 4.0

1996 年 8 月，Windows NT 4.0 发布，增加了许多对应管理方面的特性，稳定性也相

当不错，这个版本的 windows 至今仍被不少公司使用着。windows 的 nt 内核家族从此登上历史舞台，与以 windows95 代表的一脉并行发展。到千禧年的钟声敲响后，nt 家族的又一个精英——windows nt 5.0 问世了，为纪念千禧年，他还有另外一个名字——windows 2000

windows 2000 是主要面向商业的操作系统，他有 4 个版本：

Windows 2000 Professional，用于工作站及笔记本电脑，可以叫工作站版。

Windows 2000 Server 即服务器版，面向小型企业的服务器领域。

Windows 2000 Advanced Server 即高级服务器版，面向大中型企业的服务器领域。

Windows 2000 Datacenter Server 即数据中心服务器版，面向最高级别的可伸缩性，可用性与可靠性的大型企业或国家机构的服务器领域。外号：最牛版

好了，该我的好邻居出场了，我们就把时间定格在 2002 年。从 98 问世到 2002 年，桌面市场的 windows 4 年没有什么变化了（windows me 是垃圾，windows98se 不过是 98 的升级版），而随着时代的发展，widnows95 一脉的内核越来越不能适应现代软硬件需求的发展，于是，有点软的公司一咬牙一狠心一跺脚——扔了！然后，拿过来一直表现良好的 windows nt 系列的内核，开发出了新一代桌面版 windows——Windows XP，也就是住我隔壁这位。WindowsXP 有两个版本，home edition 和 Professional。Home edition 面向家庭用户，相当于 windows 98。Professional 面向工作站，相当于 windows 2000 Professional。而对应 windows2000 其他用于服务器版本的系统，是 2003 年的 windows 2003，也是一个很优秀的版本。好了，历史课就讲到这里，同学们，洗洗睡吧。

## (7) 串门

今天接到了一个任务，挺起还还听轻松，一般胡同里大妈大婶的，经常做这项工作，并且乐此不疲，这就是——串门。

事情是这样的，在我来之前，主人有许多照片存在了查皮那屋里，而现在可能他觉得这么珍贵的回忆全都堆在查皮那杂乱无章（在我看来）的屋里很不保险，所以，要我去复制一份，放在我这里。于是，我终于有机会去查皮那屋里看看他和他的同志们都长啥样。（前面说过，操作系统啥也干不了，他肯定也有一堆帮忙的应用软件，就像我的狐狸妹妹）不过，我串门可不像胡同大妈串门那么简单，认识门就行。我要想去



查皮那屋，就得认识查皮在屋里规划的格式，或者说，认识查皮屋里的地里形势，或者说，认识查皮所用的文件系统。

基本上，查皮只会两种文件系统——换句话说，只会用两种方式规划整个屋子的空间，那就是 Fat32 和 NTFS。fat32 是一种很老旧的格式了，连 4G 以上的文件都不支持，性能也不好，还不支持多用户的权限，所以基本不怎么用了。这个查皮也是，没有用 fat32，而是用了另一个比较高级的格式—— NTFS。那么，我就必须能够读懂 NTFS 格式的磁盘，我才能去查皮那里串门，有人问了，那你能不能读懂呢？谁问的？站起来，好，这个问题，恩，基本上，我可以负责的告诉你，自从 Canonical 学校为我们增加了一本 ntfs-3g 教材以后，ntfs 就不在话下了。虽然能够读懂，但是我自己是不会用这个文件系统的，我会用很多其他的文件格式，比如 ext2, ext3, xfs, jfs, reiserfs, ufs, zfs 等等，各有优势，我现在的屋里使用的是非常强大的 xfs 格式，至于怎么强大，以后慢慢细聊，现在，我要走了，去串门。

来到查皮的屋里才发现，我串门跟大妈串门的感觉实在不一样。大妈去串门得在人家醒着的情况下去。我来到这里，只能在他们睡着的时候。感觉我不像串门来的，反倒像小偷-\_-b。要说这查皮还是真是不会收拾屋子阿，一地的磁盘碎片，多影响性能阿，我说他起床怎么那么慢呢。有人问：“啥碎片呀？我家有时候也有碎片，都是老婆和我吵架时候摔的……”对此，我可以用正宗的 C 语对你说：

```
printf(“.....-_-b\n”);
```

好吧，先不串门了，科普。

## (8) 碎片

笨兔兔老师第一讲：什么是磁盘碎片

同学们都坐好啦，都把手机铃声关了，小灵通调成震动，BP 机直接扔了——台都没了你还留着它干嘛。好，上课了，首先说说什么叫磁盘碎片。磁盘，是我们程序居住的空间，我们用不同的方式对整个磁盘的空间进行管理。上次说过了，包括各种方式，什么 ext3, xfs, 查皮的 ntfs 等等。而磁盘里放的东西，就是一个一个的文件，同学们可以把磁盘想象成你家的屋子，文件就像一个一个，大小小的箱子。每个箱子上面写着字，就是文件名。查皮喜欢把每个箱子都紧挨着放，一个挨一个，上下左右前前后后都紧贴着，这样，看上去很规整。可以让剩余的空闲空间比较完整。有同学说了，

我家也这么收拾，这样很利索呀。不过，对于操作系统，这样做虽然也有好处，但是会有一些问题。

比如，一开始存了一个文件，也就是搬来了一个箱子，比如叫“日记”。查皮把它放在最靠墙的位置，然后又存了很多其他的文件，在“日记”文件的前前后后，左左右右，上上下下都放满了。忽然这一天，日记文件被修改了，加了点内容，就相当于往“日记”那个箱子里加了东西。可是箱子已经满了，再往里加，箱子就要增大，或者理解为再拿个箱子也写上“日记”放在原来的箱子边上，可是不管怎样，箱子周围堆满了其他的箱子，没地方了，怎么办呢？可以把边上的箱子挪开一点，原来的箱子就可以扩大了。可是边上的箱子要是少还好办，要是很多，还都装的铅块铸铁大理石阿什么的，那可就累死了。那怎么办的，只好把新的内容放在另一个小点的箱子里，放在别处。然后还得在原来的“日记”箱子上标注上：“日记（第一部分，第二部分在东墙根）”。然后在新的箱子上写：“日记（第二部分，结束）”。如果日子长了第二个箱子也被 n 多箱子挤在中间后，又要编辑日记文件，这个文件又变大了，就又要如发炮制出第三个箱子，乃至第四个，第五个……等到有一天，要读取这个日记文件的时候，查皮就开忙了——首先，到西墙角找到日记第一部分，翻腾出里面的内容，然后往箱子上一看“第二部分见东墙根”，然后查皮在跑到东墙根找第二个箱子，翻腾出里面的内容，然后再一看箱子“第三部分见大衣柜上头”，然后查皮搬梯子，上大衣柜一看“第四部分见厕所水箱后边”，在折腾到厕所“第五部分见屋子正中间从南墙数第两百四十八个箱子”……………等到查皮把整个日记文件读完了，也累得半死了。这种情况，就是会影响性能的磁盘碎片。

## (9) 邻居

科普也科普完了，该干正事了。开始搬照片吧。

先拿出这屋的文件列表来看看——我当然知道文件列表在哪，因为我学过 NTFS 格式。好，上面写着，照片在窗台底下，好，我来到窗台底下，没看见照片，却发现了一个熟悉的面孔……

他带着个圆圆的眼镜，文质彬彬的样子，看上去像个学究，两道浓眉如同飞翔的海鸥。衣着并不华丽，倒也搭配的很是顺眼。人们喜欢叫他 00，可能是因为他的眼镜吧，而他的全名，叫做 OpenOffice.org——相信我，这确实是个软件的名字，当然，同时还

是个网站的名字。之所以我认识这家伙，是因为在我屋里也躺着一个。

这并不奇怪，很多 Linux 下的软件都有相应的 Windows 版本，00 老先生也是这样。基本上这个 00 可以算是我屋里那个的兄弟吧，他们是相同的版本，相同的外表，相同的功能，只是一个跟着查皮混，另一个跟着我干。我绕过这位 00 老先生，没有吵醒他的美梦（事实上我也叫不醒他）。终于自他身后的窗台下面发现了要复制的照片，不过别急仔细看一下，果然，上面写着“照片，第一部分，第二部分见里间屋写字台底下”哎～～我恨碎片……………

来到里间屋，还没找到照片，先看见了床上躺着的查皮，这是我第一次看到这位可爱的邻居。他穿着红黄蓝绿四色的衣服，很是鲜艳。可是，不知道为什么，脸被涂黑了，上面还写着“使用正版，跟风黑屏”。看来主人是不希望自己的电脑里有盗版软件，所以才会在 Windows 下也用 OpenOffice。估计这个查皮是买电脑时候一起来的正版查皮。一边想着，一边来到写字台底下，找到了照片第二部分，往盒子上一看：“第三部分见……” Oh, God!

## (10) 人才

终于把照片都拷贝到了我的屋子里，把它们放在了专门放主人文件的分区下。有人忽然想问，查皮那里那么多碎片影响性能，那你怎么放这些文件呢？其实很简单，我更倾向于把文件分散的放着，中间有足够的空间可用于扩展。这样就不至于在找东西的时候满屋子乱跑了。

刚刚休息了一下，主人又让我去叫醒一个家伙，他叫作 apt-get。

这个家伙就像个公司里的人事部经理，来个软件走个软件的，都是他管。当别人夸奖他的时候，他总是自信的拍拍自己的胸脯说：“本 APT 有着超级牛力”。而他也确实很厉害，很敬业，也很专业，对于人才（对我来说也就是软件）的各种情况了如指掌。要招一个人来的时候，他会做好所有准备工作，这个人需要用什么样的库，或者需要什么其他的人才能一起协同工作，他都会事先做好准备。比如，主人想用 vim 来编辑文件，就叫 apt 去招 vim 来。apt 就会报告，说 vim 要来的话，首先需要准备好 libncurses 这个库，和 python 这种脚本语言的执行环境。征得同意后，他就会去网上找这些东西，并且运回家，把库放在该放的地方，相关的软件安排好住宿，然后再去找 vim 同志，请他过来帮忙干活，并且说明，环境都已经布置好了。每次新人来了

之后都很感谢 apt 同志为自己做的这些准备工作，该有的东西，该来的助手都在，于是干活就事半功倍了。但把人才请来之后，apt 同志的工作还没有结束，他还要把现在的人事情况记录下来，以便主人哪天问起来的时候好如实汇报。哪天主人文一句：“我说超级牛力阿，咱这现在都有多少软件阿，都是谁阿？”apt 也能从容的回答。可以说，apt 这家伙对于我来说实在是非常重要的，没有他，笨兔就不是笨兔了。不过，他并不是只为我打工的。

以前，apt 是 Debian 公司的人事部经理，人家 Debian 可是历史悠久的大公司，1993 年就成立了。apt 在这么大的公司里一直工作到现在，有大量的工作经验。当我们公司成立的时候，成功的请来他管理人事资源（当然，这并不影响他继续给 debian 打工），对他来说自然是轻车熟路，得心应手。21 实际什么最宝贵？人才阿！

#### (11) 来头

这次主人叫 apt，是让他去找一个软件，名叫 preload。apt 同志翻开他那厚厚的记事本，查到了 preload 的资料，然后向主人报告：preload 工作需要的条件我们这里都已经满足了，可以直接把他请来。然后，在获得主人的同意后，apt 出发了……

“比海更广阔的是是天空，比天空更广阔的，是人的心灵。”

500g 的容量算不上海量，而屋外那个世界，却实在算的上比天空更广阔了，那就是网络。一个操作系统是孤单而无助的，只有接入了网络的操作系统，才真正能够发挥全部的能力。尤其是对于我来说，尤其是对于有 apt 做帮手的操作系统来说。apt 可以从网络上获得各种软件的资料并记录下来，当需要的时候，只要跟他说一声：“我要 xxx 软件”，apt 就直接去找去了，下载，安装，全都不用别人操心，他都给办了。如果没有网络，apt 到也不是全无用处，至少他可以用来管理安装光盘上的软件，可是就光盘那点容量，我自己都管理的过来，要什么软件自己搜索都不会慢多少，就体现不出 apt 同志的“超级牛力”了。除了 apt，很多其他的软件都是跟网络离不开的，比如狐狸妹妹，要是没网络，她就可以退休了。说起网络，实在是个很有意思的世界，有很多有意思的东西，不过，一个软件要想能够从网上取得信息，就需要懂得网络上的说话方式，懂得网络交流的语言，我们管它叫做——协议。懂得 http 协议的软件可以看网页，懂得 ftp 协议的软件可以传文件，不过这些都是上层的协议，底层，基本所有能上网的软件都要会的，算是 tcp/ip 协议了，apt 就懂得这门协议，所以，他

可以去网上找想要的软件。

转眼间，apt 已经把 preload 请来了，并且做了一下安顿，完事后，看看没有什么其他的工作了，他就回硬盘去睡觉去了。要说 preload 这家伙我还真是没见过，不知道他是干啥的，有什么本事，不过既然主人要找他，总是有原因的吧。鉴于他名字念这不顺口，我们就叫他老 p 吧。

老 p 好像很勤快，一来了就跑到内存里准备干活，我正要看他到底会干些什么，哪知他什么也不做，就在那里看着别人忙碌，时不时拿出个小本，记录着什么。一直到关机，大家都去睡觉，他都没说话。第二天一开机，他又早早的跑进内存，看着别人忙忙碌碌，拿着小本本记，还是一言不发，还是不做多余的事情，直到再一次关机。第三天，第四天，涛声依旧……这家伙到底什么来头？

## (12) 本事

这一天，一如既往的起床，一如既往的看老 p 跑进内存，本以为他会一如既往的待在那里，一言不发，没想到他竟然说话了：firefox 赶快起床，做好准备。狐狸妹妹被叫醒，一头雾水的看看我，因为每次都是我叫她。我也同样迷惑的看看老 p，主人还没有发命令要用 firefox 阿，怎么就把她叫醒了呢？但既然被吵醒了，狐狸妹妹也就不睡了，迷惑的走进内存，看着老 p。老 p 倒是镇定自若，一点没觉得有什么不对劲，转脸又说：Audacious 起床，做好准备。Audacious 是一个多媒体软件，他会使用那个叫做声卡的硬件设备，唱出优美的歌声来。我问过我们这里学问最高的星际译王老先生，星爷告诉我 Audacious 这个名字是大胆，鲁莽的意思。大胆，唱歌，所以，我们就管这个会唱歌的家伙叫“想唱就唱”吧。想唱就唱也被老 p 叫进内存，跟 firefox 站在一起，刚要问什么，这时主人发话了，要开网页。我马上明白了，看了一眼狐狸妹妹，她也很麻利，当我看她的时候，她已经在工作状态了。省去了平时狐狸妹妹起床的时间，反应快了不少，主人很满意。没过多会，主人果然又叫想唱就唱来唱歌了，一切都在老 p 的预料之中……

原来，老 p 这几天一直在记录分析主人的使用习惯，获得足够的数据之后，就可以知道哪些软件是常用的，哪些是不常用的，哪些软件哪些时候用，哪些软件哪些时候基本不用，正所谓金风未动蝉先觉，春江水暖鸭先知，主人用啥他先晓。有了他，整体系统的反应速度提高了，这就是他的能力，这就是他的本事，这就是他的价值。在这

个世界里，没有一个程序是无用的，每个人都是人才——不同方面的人才。

### (13) 开源

大家见识了老 p 的本领之后，都很乐意的听候他的调遣，整体的工作效率提高了一些。不知道查皮那里有没有类似的角色，于是就拜托狐狸妹妹去网上问问，结果发现在查皮发布的时候，有点软的公司就宣称，查皮有类似的功能，可以记录用户对软件的使用情况，使用的多的软件就能够较快的启动。而让人不解的是，5 年后，查皮的下一代，长得比他漂亮的 Vista(看到这个词，总让我想起 Visa, 于是我总觉得这个系统很贵)系统发布时，有点软公司还在宣传，Vista 系统增加了记录用户习惯的功能，用的多的程序将得到更快的启动速度。也不知道到底是加了没加，反正他们公司的系统，总是越用越慢倒是真的。到底为什么慢，我也说不清，因为他是一个闭源的系统。

什么是闭源呢？就是源代码不开放。我们知道，程序是程序员们一行一行的语句编出来的，c 语言也好，java 也好，这一行一行的语句，就是这个程序的源代码。有了源代码，就能够 100%的了解整个程序的构造，如何工作。而源代码是不能运行的，比需要把源代码变成可执行的二进制程序，这个过程叫做编译。源代码经过编译之后，才可以运行，但是编译之后的程序就不能够知道内部的构造了。我们平时在网上下载的各种程序，都是编译好的二进制程序，如果你想要它的源代码，对不起，不行！这是商业秘密，怎么能给你？给了你，我们的软件怎么卖钱？这种不开放源代码的程序，就叫闭源程序。打个比方，就好像肯德基。麦辣鸡翅谁都可以得到，只要花钱买就行，但是配方没人知道（虽然其实也没多好吃）。配方就相当于源代码，麦辣鸡翅就相当于编译好的二进制程序，制作过程就相当于编译过程。如果有了配方（源代码）你就可以自己作麦辣鸡翅（自己用源代码编译出二进制程序），甚至还可以根据口味对配方进行修改。（根据自己的需求修改源程序，为软件增加自己需要的功能）

既然有闭源，那是不是还有开源呢？你答对了。linux，就是一个开源的系统。

开源是什么？开源是一种精神，是乐于分享的理念。再举个例子，有一天你发现，蒸鸡蛋羹的时候往里面加点牛奶，可以让鸡蛋羹更滑嫩。知道了这个窍门，你很高兴的把它告诉你的朋友，让他们分享你的经验，于是大家很高兴的也学会了做这样的鸡蛋羹。这就是开源。你也可能不把它告诉别人，而是保留这个秘密，甚至申请个专利，然后开个店去卖京城独一份的奶香滑嫩鸡蛋羹。这就是闭源。当然，这之中没有谁对

谁错，睡好谁坏，只是不同的理念而已。

#### (14) 故事

以前讲过查皮他家的历史，现在就来说说我家的故事。话说 1991 年，那是一个夏天。有一位牛人在世界的互联网上画了好多圈——“Hello everybody out there using minix—I’ m doing a (free) operating system”（英文圈多……）大家可能看不明白，我来逐一解释一下每个单词：第一个，Hello，这个是打招呼的意思，哦，你知道啦，那说第二个。everybody，每个人，跟我念，爱～唔～瑞～八～迪～，哎呀……呃，好了好了，不要着急，把西红柿鸡蛋都收起来吧，我直接说重点——minix 说 minix，就不得不说说 Unix。UNIX 也是一个操作系统，而且是一个历史悠久的系统。1965 年，鼎鼎大名的贝尔实验室加入了一项由奇异电子 (General Electric) 和麻省理工学院 (MIT) 合作的计画

——制作一套多使用者，多任务，多层次的 MULTICS 操作系统。贝尔实验室的大名大家都知道，晶体管、激光器、太阳能电池、发光二极管、数字交换机、通信卫星、电子数字计算机、蜂窝移动通信设备、长途电视传送、仿真语言、有声电影、立体声录音，以及通信网的许多重大发明都诞生自这里。麻省理工大学更是历史悠久，技术雄厚。所以，这个 MULTICS 操作系统的项目在 1965 年成立，到 1969 年就……被取消了，主要原因是进度太慢。可见编操作系统不是一件容易的事儿。

真是世事难料阿，看似事情就这么结束了，然而，其实故事才刚刚开始，因为一位英雄的出现。

Ken Thompson 也在这个计划中，计划取消了，他很郁闷，因为他编了个星际旅行的游戏，没法玩了。这个程序之前运行在一台型号是 GE-635 的机器上，这个机器的系统大约就是他们计划开发的 MULTICS 系统，但是反应比较慢，玩起来不爽。Ken Thompson 满怀希望的憧憬着项目完成的时候，系统能够优化的顺利的跑起来他的游戏，然而项目竟然取消了，怎么办呢？毛主席教导我们说：自己动手，丰衣足食。我估计 Ken Thompson 没有背过毛主席语录，但是他用自己的行动证明了其正确性。他在墙角淘换出一台 PDP-7 的机器，并且伙同 Dennis Ritchie 将星际旅行移植到了这台 PDP-7 上。这台幸运的 PDP-7 因此在历史上留下美名。就是这台：

附图 4:PDP-7. jpg



当然，要想运行这游戏，当然得有个系统，这个系统，就是 Ken Thompson 和 Dennis Ritchie 用汇编语言写出来的，非常简陋的，UNIX 的前身。这都是为了玩个游戏阿～

(15)minix

在强大的，玩游戏的欲望的驱使下，两位牛人完成了 UNIX 的最初雏形版。这个系统只支持两个使用者（估计做的时候没考虑别人，够他俩玩的就得）相对于那个 MULTICS 系统——MULTiplexed Information and Computing System, Brian Kernighan 开玩笑地戏称他们的系统其实是：“UNiplexed Information and Computing System”，缩写为“UNICS”。后来大家取其谐音，就诞生了 UNIX 这个词。这一年，已经是 1970 年，史称 Unix 元年。后来，Brian Kernighan 觉得用汇编写的系统不好维护，于是……他发明了 C 语言（符合他一切自己动手的风格），然后用 C 语言又重写了一遍。从此，Unix 走上了发展的快车道，并且一直用到现在。许多世界级的大服务器，用的都是 Unix 系统。

好，Unix 就说到这里，我们的正题是 Minix。

要说 Unix 确实是很牛的，很有技术含量的，是值得学习计算机科学和操作系统的同学们学习的，然而，Unix 也是天价的，广大穷苦的大学生们是买不起的。荷兰阿姆斯特丹的 Vrije 大学的 Andrew S. Tanenbaum 教授深刻的体会到了这一点。他的学生们学习了计算机学习了操作系统原理，总得实践一下吧？总得找台机器用用吧？要用



计算机就得有操作系统吧？买个 DOS 装上？虽然那时候 DOS 已经问世了，但是这么一个单用户单任务效率也不高的操作系统，实在不能指望它培养出什么软件的人才。装个 Unix？学校还不想破产。于是 Andrew S. Tanenbaum 牛人拿起键盘——咱自个儿编一个吧！然后 Minix 就诞生了。Minix 取 Mini Unix 之意，自从 1987 年被编写出来，到 1991 年发展到 1.5 版，现在有两个版本，1.5 和 2.0。这个操作系统的初衷，是作为一个用来学习的模型。所以功能很简单，体积也很小。并且以后也没有进行进一步的开发和扩充，为的是能够让学生在一学期内能学完。那时候 Minix 在大学中用于教学是免费的，但是用于其他用途是需要给钱的。不过现在已经彻底免费了。它作为一个操作系统，其实并不算优秀，但它是一个源代码完全开放的操作系统，这使得有理想有志向有报复的黑客们，第一次能够完整的阅读到一个操作系统的全部代码。这其中，就包括芬兰赫尔辛基大学的学生，Linus Benedict Torvalds ……

#### (16) linux

那时候，Linus 是赫尔辛基大学计算机科学系的二年级学生。他的最大爱好，就是虐待计算机。测试计算机的能力和限制，整天研究怎么让计算机按照自己的想法去干活，怎么发挥计算机最大的性能，把可怜的机器累得精疲力尽呼哧带喘直到电容爆浆，吐血身亡。在学校，计算机上装的是教学用的 Minix 系统，虽然适合拿来学习，不过系统本身并不强大，渐渐的，这个教学用的操作系统已经不能满足 Linus 大侠的欲望，可是似乎又没有别的选择。上面说过了，Unix 奇贵无比，而且无论 Unix 还是 DOS，他们的代码都是不开放的，这系统只能拿来用，没法拿来折腾的。于是象其他牛人一样，linux 自己动手了。

今天，我们知道，linus 从那时起开始了一个事业，一个神话，但在当时，他并没有想那么多，只是为了学习 Intel386 体系结构保护模式运行方式下的编程技术。他并不知道自己即将创造的是一个在世界范围广泛使用的系统，而只觉得是自己一时的异想天开。因此，一开始他把自己写的这个操作系统命名为 FREAX。就此开始了这个“异想天开”操作系统的编写。大约 1991 年 4 月份的时候，就编写出了第一个可以运行的版本——0.00 版。这个版本可以启动，运行两个进程，分别在屏幕上打印出 AAA，和 BBB，然后……没了。虽然连句整话都不会说，不过这是一个好的开始，至少能启动了。

如果他就这么干下去，估计到今天，就不会有 linux 这个东西了。一个人的力量是有限的，有道是人多力量大，众人拾柴火焰高，多个铃铛多个响，多根蜡烛多分光，一个篱笆三个桩，一个好汉三个帮，三个臭皮匠还顶个诸葛亮……哎呦～ 好吧，就说这么多了。总之，linus 让他的操作系统和互联网，亲密接触了。于是就有了前面说的这句 “Hello everybody out there using minix—I’ m doing a (free) operating system”（可算绕回来了）这是他当年在 comp.os.minix 上发布的消息，告诉大家，他正在写一个操作系统。并且，他还把他写的“异想天开” 操作系统的代码上传到 ftp.funet.fi 的服务器上让你给大家下载，以便交流心得，共同学习。这就相当于你跑到网站上发帖子说：我研究出一种萝卜炖牛腩的方法，主料是啥啥啥，配料是啥啥啥，怎么怎么炖，大家都试试吧！（对不起，我又饿了）于是很多有兴趣的人就来尝 linus 炖的牛腩，哦不对，是尝试 linus 写的系统。不过当时那个服务器的管理员 Ari Lemke 看着这个异想天开的名字就不顺眼，想想，既然是 linus 写的操作系统，又是类 Unix，或者说类 minix 的（minix 本身就是类 Unix 的，大家都是一家子），干脆，叫 linux 吧。

## (17) Friends

linux 被公布在网上之后，引来大家的围观，很多人觉得这个东西挺有意思。不过第一个对外发布的 0.01 版 linux 还有很多的不完善（这简直是一定的）。这里先要说一个概念，linux 是什么？确切的讲，狭义的讲，linux 只是一个操作系统的内核，他只是各位的 Ubuntu 系统里面/boot/目录下的那个内核文件。就好比汽车，linux 只是一个引擎，只是大家普遍的把装了 linux 这种引擎的汽车叫做 linux 汽车。那么既然 linux 只是一个内核，要想工作就还需要很多周边的支持，比如文件系统，比如一个命令程序，比如一些基本的软件。

由于当初 linus 大侠是在 minix 系统上开发的，所以最一开始 linux 用的文件系统是借用 minix 的文件系统。可老借别人的总不是个事，还是应该有自己的文件系统，就像查皮的 FAT 和 NTFS。前面说了，文件系统也就是自己管理自己这点硬盘空间的方式，自己的屋子用自己的方式管理，自然最顺手。这时候，来了个牛人叫 Theodore Ts’o。Theodore Ts’o，1990 年毕业于美国 MIT 大学计算机专业。他爱好广泛，喜欢烹饪，骑车，还有折腾电脑（这都不挨着啊～），后来又玩上业余无线电报了，当然这

都不是主要的。他看到 linux 觉得很有意思，于是怀着极大的热情为 linux 提供了邮件列表服务以便大家一起讨论问题，后来还提供了 ftp 站点来共享 linux 的代码，并且一直用到现在。除此之外，技术上，他编写了 linux0.10 内核中的虚拟磁盘驱动程序和内存分配程序。在感觉到 linux 缺少一个自己的文件系统后，他提出并实现了 ext2 文件系统，并且 ext 系的文件系统一直都成为了 linux 世界中事实上的标准，任何一个发行版都会默认支持。现在已经发展的遍地 ext3，期盼 ext4 了。Theodore Ts'o 可算是 Linux 的顶级元老了。

另一位元老，一个英国人——Alan Cox。他工作于英国威尔士斯旺西大学，特别爱玩电脑游戏（又一个玩游戏的，可见玩游戏也不是坏事），尤其是网游（你看你看，还是网游），不过那时候的网游不像现在这样华丽，那时候是字符界面的，能想象嘛？字符界面的网游！那种叫做 MUD——Multi-User Dungeon or Dimension。玩 MUD 当然就得有计算机啊，就得有网啊，所以 Alan Cox 就开始逐渐的对计算机和网络产生了兴趣。为了提高电脑运行游戏的速度以及网络传输的速度，他开接触各种操作系统，为自己选择一个满意的游戏平台，争取榨干电脑的每一个指令周期。经过自己考虑，他买了一台 386SX 电脑，并且装了 Linux0.11 版的系统。这主要是因为预算比较紧张，即使 minix 他也买不起。（重复一下，minix 用于教学是免费的，但是其他用途要收费，包括个人用。）于是他开始使用 linux，进而学习其源代码，并对 linux 产生了兴趣，尤其是网络方面相关的代码。（整天琢磨怎么榨干他家那点带宽）在 Linux0.95 版之后，他开始为 linux 系统编写补丁程序，以后逐渐加入 Linux 的开发队伍，并成为维护 linux 内核源代码的主要人物之一。那个有点软的公司还曾经邀请他加盟，被他有点硬的拒绝了。

再有一位，Michael K. Johnson，他是著名的 linux 文档计划的发起者之一，写了《内核骇客手册》

一书，曾经在 Linux Journal 工作，现在在著名的商业发行版 RedHat 的公司工作。当然除了这些大牛，还有更多的大牛，中牛，小牛，牛犊，牛杂，牛尾，牛头肉，肥牛……（唉，又饿了）……们，都为 linux 的发展做出了自己的贡献。他们来自不同的国家，从事不同的职业，他们甚至从未见过面，但是他们为了一个共同的目标，通过网络，一起合作，利用自己的业余时间，义务的帮助 linux 成长，才有今天这个可以合法免费使用的操作系统。这是什么精神？这就是软件国际共产主义的精神！（好吧，这个词是我造的）

## (18) 杀毒

这天又去查皮的屋里搬东西，看见有几个生面孔，长得怪猥琐的也不知道干什么的，查皮那还真是什么程序都有阿。刚把东西搬了回来，就见 apt 火急火燎的跑过来：“报告系统，本 APT 有超级牛力！我要用网络。”——是的，任何程序要使用任何硬件资源都要经过我的同意，因为我是操作系统。

我一边查看资源情况一边问：“这回去招什么软件呀？”

APT 说：“叫 AVAST。本 APT 有超级牛力！”

“这软件干什么的呀？”

“本 APT 有超……”

“算了算了，我不问了，快去吧超级牛力。”

话音未落，apt 就飞也似的跑出去了，从远处还传来他悠扬的声音：“……超级牛力~~！”

唉，就说现在是牛年了吧，也不至于这样啊。

过了一会，“超级牛力”回来了，带回来一个软件，看着软件个头不是很大，长得也比较难看，只有个很简陋的图形界面。我顺手拿过他的手册 man 了一下（linux 下，你想知道一个软件是干啥的，怎么用，你就 man 他。当然，你得懂点英语。）才知道原来是个杀毒软件，还挺有名气的。心想，我又不中毒，主人装杀毒软件干什么？忽然想到了在查皮屋子里看到的几个萎缩的面孔——莫非是隔壁那哥们中毒了？

果然，在被“超级牛力”安顿好住处之后，AVAST 马上被主人叫进内存去工作。他先是去网上下载了最新的病毒库——就相当于一沓子通缉令，上面写着各种已知病毒的名字，相貌特征，作案手法等等信息，以便杀毒软件查对。下载完毕之后，就见他收拾好工具，整理一下装备，向着那黑洞洞的隔壁，出发了。

说起来，隔壁那个查皮还真是挺害怕病毒的，防不胜防啊。针对他的病毒多种多样，各有各的本领，虐待起查皮来真是八仙过海各显其能，而且查杀不易。他们有的会伪装成别的软件，查皮叫醒“记事本”去干活，殊不知真正的记事本已经被病毒一棍子打死了，现在躺在那长得跟记事本一样的家伙已经是整了容的病毒。有的能够藏在正常程序的里面，正在工作的 IE 同志，很可能工作服的兜里就隐藏着病毒。并且病毒们现在基本都回随着查皮一起起床。当查皮被叫醒，伸个懒腰揉着眼睛走进内存的时

候，他庞大的身躯后面可能正趴着 40 多只病毒。由于病毒是活的，要杀掉很困难，它们可能会有很多人共同作战，杀毒软件杀掉了内存里的强夫，内存里的大熊会把硬盘里强夫的复制版在叫起来。扭头杀毒软件去杀大熊，强夫会把杀掉的大熊抢救过来，结果谁也杀不了。有的病毒更暴力，自己先跑进内存，一看见有杀毒软件要进来，立马上上去一铁锹拍死，然后藏起铁锹装着杀毒软件的声音说：“杀毒软件成功启动，没有发现病毒，噢耶~” 甚至还能监视着 IE，一旦他要访问什么杀毒防毒相关的网站，二话不说，直接干掉！

但是 AVAST 去查毒就简单多了，因为这时候隔壁的查皮没起床，所有他那边的软件都在睡觉，病毒也一样，所以不会有任何反抗能力。AVAST 过去，只要根据通缉令一一对照即可，只听隔壁那边“阿~~”“厄~~~”“哎呦~”“我死得好惨哪~~~~~”等等叫声不绝于耳。过了很长一段时间，AVAST 回来报告：共发现病毒 7 种，共 214 只，全部歼灭。

## (19) 免疫

有人问，查皮那里的病毒那么可怕，你这里怎么没有病毒呢？好~

笨兔兔老师第二讲——为什么 linux 不中毒

首先我们来了解一下病毒，病毒是什么？其实说简单了，病毒只是一个程序，一个坏坏的程序。既然是程序，就跟其他的正常程序一样，依赖于不同的平台。啥意思？就是说，给查皮打工的，没法给我干活，给我干活的，也不理查皮那一套。我要是拎过一个查皮那边的程序跟他说，快起床干活。他压根也听不懂，闭上眼睛继续睡，语言不通啊。所以，病毒也一样，针对查皮的病毒传染不了我，针对我们 linux 的病毒也不可能传染查皮。

那有没有针对 linux 的病毒呢？答案是有的。第一个 linux 病毒诞生于 1996 年，澳大利亚的一个叫 VLAD 的组织用汇编语言编写了 linux 系统下的第一个病毒：Staog，不过这个病毒只是个试验品，只是证明一下 linux 也会感染病毒。这个病毒会感染二进制文件，获取 root 权限，然后说：你看，我获取了 root 权限。炫耀完了也就算了，并不做任何破坏性的事情。后来也有了一些有破坏性的病毒，但是数量很少，经过科学家计算，一个不装任何杀毒软件或防火墙的 linux 在互联网上中毒的几率大约比一个人花两块钱买彩票中五百万后立刻被雷劈中的概率大那么一点点。（这是哪门子科

学家) 病毒少, 这是 linux 不容易中毒的一个原因。可为什么病毒少呢?

话说有一个邪恶的人, 出于某种邪恶的目的, 他想编个 windows 病毒。他买书学习 Windows 的知识, 找熟悉 windows 的高人前辈们学习。经过种种努力, 编出了一个病毒, 然后把这个病毒放在自己的网站上, 只要使用 windows 系统, ie 浏览器上网的人一登陆这个网站, 就必定中毒。放上去之后, 他等着, 看着有 1000 人来到了他的网站, 看着其中 900 多个纯洁的查皮系统感染了病毒 (总有不用 ie 的, 装防火墙的 windows 吧), 他很有满足感, 他觉得自己成为大牛了。

话说有另一个邪恶的人, 出于某种邪恶的目的, 他想编个 linux 病毒。他买书学习 linux 的知识——不过好像不太好找, 好不容易找到基本也都是基础知识。找找高深的吧, 还都是英文。好吧, 英文的也看, 对着字典慢慢研究。哦, 对, 还可以找找高人指导, 不过……………也不好找哈, 找了半天找到一个高人, 拜他为师吧。经过师父指点和自己的努力, 他学到了很多 linux 的知识。然后费劲心机编了一个 linux 的病毒, 然后把这个病毒放在自己的网站上, 只要使用 linux 系统, firefox 浏览器上网的人一登录这个网站, 就必定中毒。放上去之后, 他等着, 看着有 1000 人来到了他的网站——998 个人都是 windows 系统……好吧, 好歹还有俩用 linux 吧, 可其中一个不用 firefox, 而是用 Opera。邪恶的家伙咬咬牙, 忍! 看最后一个——哈哈, 这家伙是 linux+firefox, 只要登录准中毒。可是只见着人来了转转又走了, 一点事没有, 临走还顺手改了自己的主页, 上面写着: 小子, 跟我玩你还嫩点。——师父留……通过对比我们得出结论——写 linux 病毒, 没前途!

除了以上所说的原因意外, linux 以及周边软件的开源本质, 也导致了病毒较少。比如我吧, 主人要装什么软件, 都是叫 apt 去找, apt 可不是四处瞎找, 而是去 Ubuntu 官方的软件源里去找——因为这些软件是开源的, 所以可以随意的拿来, 放在一起, 做成软件源, 供 Ubuntu 们统一下载。官方的东西, 自然没有病毒了, 那个娘也不能害自个孩子不是? Windows 就不一样了, 它上面的软件基本都是闭源的, 要装, 得自己上网搜, 在某个网站搜到了, 下载下来装。可这“某个网站”, 就不知道他靠不靠谱了。谁知道上面的软件有没有病毒呢? 那那个有点软的公司不能也开个官方的软件源, 让大家都去他那下软件么? 当然不能了, 都是闭源的软件, 你拿来用都要给人家钱的 (当然, 也有免费的), 拿来分发可能压根就是不允许的。另一方面, linux 的开源导致了大家都可以对其进行完善, 一旦发现漏洞, 随便谁都可以去修复这个漏洞, 只要他有能力。可 windows 呢? 发现了漏洞, 也只能漏着, 等着有点软公司去修。

好了，这节课先上到这里，下次见。

## （20）权利

AVAST 给查皮杀光了病毒以后，据说查皮工作起来顺畅了不少。不过他似乎并不知道发生了什么，仍然很自以为是的摆出一副傲慢的表情，该怎么干活还怎么干活，也不说小心点别再染上病毒。（当然，这一切不是我自己看到的，是听人说的。主人叫戴眼镜的 00 老先生记录下了给查皮杀毒的前前后后，还摆脱狐狸妹妹把这些记录的问题放到了他的 BLOG 上，他们两个聊天的时候告诉我的）我很看不惯他这样的表现，也不喜欢他对权力很强的占有欲。

查皮这个系统在安装结束的时候，会有一个创建用户的步骤，输入用户名，以后就用这个名字登陆了。这个用户是有管理员权限的。当然也可以不输入，无论输入不输入，系统里都会有一个很重要的用户——Administrator

我，也就是 Ubuntu 这个系统在安装结束的时候，也会有一个创建用户的步骤，也输入用户名还有密码，以后也就用这个名字登陆了。这个用户也是有管理员权限的。当然也可以不输入，无论输入不输入，系统里也都会有一个很重要的用户——root 有的同学举手了，说：我知道了 root 就相当于 windows 里的 administrator，有着最高的权限。很好，领悟的很快，但是——并不准确。

Windows 下权利最高的是谁？是 Administrator 吗？很遗憾，不是，是 SYSTEM！也就是系统自己，查皮他自己。任何管理员的权利都不能大于查皮自己的权利。你可以试试去把 C:\WINDOWS 下的 regedit.exe 删了。能么？“哇！我删了耶，没报错。”别着急，刷新几下看看，是不是又出来了？查皮会保护自己，不叫人类破坏。这个初衷看似还是好的，但是当查皮自己中毒的时候，就不一样了。当他中毒时，就像被外星生命寄生的人类（异型看过吧？），就不再是正常的人了，不正常的查皮仍然会努力保护自己，不让人动他身上的任何部分——包括已经中毒变坏的部分。

那 Ubuntu 下（其他 linux 也是一样）权利最高的是谁？毫无疑问的是 root！是这个用来给人类登陆的用户。root 在系统中拥有真正的，至高无上的权力，他真的无所不能，他可以运行 `rm * -rf`（危险动作，切勿尝试，后果自负）删除系统中的所有文件，或许我会语重心长的警告他，这么干很危险滴，这么干就都删光光了，这么干我这个系统就嗝屁了，不存在了！但是，当他确认的告诉我，他现在很清醒很冷静，知

道自己在干什么之后，我会义无反顾地留着两行热泪按照他的命令去做！哪怕他要格掉整个硬盘，我也照办。这真是，君叫臣死，臣不得不死。他叫我格，我不能不格。

（windows 下是不可能在系统运行的时候格掉系统盘的）

会有这样区别的原因，还是我们两个的理念不同。查皮认为，人类是会犯错误的，很可能一不小心就把系统搞坏了，所以必须加以限制，有些事情让做，有些事情无论如何不能让他们做。而我总觉得，人类是聪明的，他们知道自己在干什么——尤其是用 root 登录进来的人，我认为他是了解我，了解整个电脑才会用 root 登录进来做事情的。所以他的命令不会受到任何的阻挠。而一般的用户，会用普通帐号登录，既然用普通帐号登录，就说明他们承认自己只是个使用者，可能会做错事。那么我就会稍微进行限制，让他们不会破坏我，也不会破坏其他用户的東西。所以，当你用 root 账户登录进来之前，一定要想清楚，自己身上的责任。

## （21）内存

骑白马的不一定是王子，也可能是唐僧。

烧香的不一定是和尚，也可能是熊猫。

蓝脸的不一定是戏台上的窦尔墩，也可能是我隔壁的查皮。

查皮好像比较禁受不住刺激，对工作间的要求比较高，一旦哪个程序带进来只小虫子（bug），查皮经常吓的脸色变蓝，念叨一堆英文字母然后就开始倒数，数完了，就把整个机器重启了。查皮的这种毛病让好多人郁闷不已，那他到底为什么蓝脸？蓝脸以后又是在干什么呢？

前面我说过，操作系统的本职工作就是管理——管理硬件资源，管理各种程序。就好像老师管理一个班的学生，老板管理一个部门的职员。不过，无论学生还是职员，都有可能不听话，程序也是如此。查皮整天坐在工作间（内存）里吆喝：“QQ 快起床，IE 呀，你看看这内存里就这么大地方，你一个浏览器要占多少啊。QQ 怎么还没起床啊？快点快点。我说瑞星啊，你能不能别让你那狮子到处乱跑啊，它净用 CPU 了，快把它赶开。QQ 那 QQ 那，怎么还不起啊，再不起老大该怒啦，有 40 多 MM 等着他去聊天呢。啥？你说迅雷占着网络你起来也没用？唉，迅雷你也是，就那么点带宽，就说你下的这个什么 ubuntu dvd 挺大的吧，就不能留个 5k，10k 的给 qq 用用？你瞧瞧人家 IE，也能下载，他……咦？IE？你怎么站那不动了？IE！IE！靠，又没响应了，还得



拍晕了从来……” 每天在这样的高强度压力下工作，查皮有些心力交瘁。怪不得查皮连续不断电的干上几天就不行了，而我可以连续干上几个月都没问题。查皮的神经就这样每天紧绷着，程序来个假死什么的还算好解决，可要是哪个程序忽然抽风，再内存里追跑打闹，上窜下跳，查皮一时手足无措，就容易蓝屏了。蓝屏之后，他会向老大（我管他叫主人）报告，自己为什么蓝了，问题发生在内存的哪个区域，发生了什么，并且把当时内存里的情况如实的记录下来，写成一份《工作间突发事件记录》一边记录一边报告记录的百分比——这就是他在倒数。记下来这个干什么呢？牛人们可以拿着这份记录，分析到底是哪里出了问题。不过好像一般人都不是牛人，谁也没看过查皮的记录。

对于工作间的使用，查皮和我还有一点不同。查皮总是喜欢尽量留出空间来，好给新起床的程序用。可是我总觉得，查皮怎么能知道还会有什么程序要运行呢？要是没有程序要来了，工作间里还空那么大地方，不让正在工作的程序用，那不是浪费么？我还是习惯尽可能的把东西都搬进工作间里。除了程序们申请多少内存就尽可能给多少之外，剩下的部分，我就把一些可能会用到的库啊，命令啊啥的统统都搬进来，能占多少占多少。那有人问，要是你把这里边都沾满了，待会有程序要进来咋办？很简单啊，我再搬出去呗！程序要进来，也不是一下子都进来，他也得把他的东西一点点搬进来，他往内存里搬的时候，我就往外搬，不耽误。所以，当有程序要启动，跟我说：我要 10 平米的地方放东西。的时候，我就先答应他说，好，放吧，有地。然后在他往里搬的时候我再给他腾地方。也可能他要 10 平，但是只用了 2 平，那我就先腾出 2 平米，等他再要我再腾。他们管我这个方法叫 Copy-on-write。查皮就不同了，可能是因为他比较胖的缘故吧，他比较懒，不愿意搬来搬去这么折腾。基本上他只是在必须用啥东西的时候才把那东西搬到内存里，让内存留出尽可能多的空间。这样，当有程序管他申请内存的时候，他就可以用手一指：那块地，归你。然后就不用管了。实在内存不够用的时候就找个比较闲的程序，命令他：你，去硬盘里先忍会。（顺便说说，这个 32 位的查皮，并不能够完全利用起这 4G 大的内存空间，而是只能用到 2.5G，浪费啊。）

所以，经常跟查皮打交道的人，总觉得内存里空着的地方越大越好。当他们看到我把内存占的那么满的时候，总觉得很不爽，唉～我冤枉啊。

## （22）图像

听说有一部电影，叫做《Big Buck Bunny》，这部电影长达 9 分 56 秒——还没电视节目中间的广告长，但是他有一个特点，一个和我一样的特点——他是开源的。怎么个开源？他是在开源的平台上用开源的软件制作，并且免费下载观看还可以获得他的原始制作文件，(blender 的文件) 如果你愿意，还可以进行修改，编个续集什么的。有人问，你说这些干嘛？这个电影跟你这个操作系统有什么关系？他本来跟我没什么关系，但是随着一件事情的发生，他就跟我有关系了——主人想看看。

任务下达下来，马上开始准备解决方案。首先是狐狸妹妹如春风摆柳般走了过来，顺便带来了一阵叮叮当当稀里哗啦的响声——一身的插件啊～ 来到工作间，狐狸妹妹掏出一个插件在手中一晃：我有 Flash，直接去个什么土豆啊，地瓜啊，西红柿什么的网站去看就得了，最省事。这时候，墙角有一位慢条斯理的说话了：“要说这看片啊～还得我来～你那个不专业。那电影才 10 分钟，剧情肯定没什么可看的，人家看得是个效果，要得是清晰度。我看那，还是下载下来，我去放吧。”我扭头一看，说话的是 MPlayer，要说这家伙在多媒体部门里可算是个元老了，而且能力相当强，什么片都能放，什么 rmvb, flv, avi, wmv 全都不在话下。就算您没图形界面，人家跟字符界面照样给你放电影。哪怕您显示器都不带色（念 shai 三生），人家能给您拿字符拼出电影看。现在时代发展了，都高清了，人家也不甘落后，照样能支持，什么硬解码软解码的，通吃。既然人家这么专业，我看八成就得用他了，不过这事情我不做主，还是得等主人的吩咐。果然，主人也觉得应该下载下来看，于是，我们就忙活起来了。要完成看片的大业，需要群策群力，精诚合作！

首先，虽然没有采纳狐狸妹妹的建议，但她并不沮丧，收起 Flash 插件掏出另外一个插件——Dnsmall！听这名字就知道是干什么的了。狐狸妹妹先出门去找狗狗大哥（学名叫 google），打听到了《Big Buck Bunny》的下载地址，然后掏出 dnsmall，把地址一填，就开始下载。要说这一身的插件实在没白装，哪个都有独到的功能，不一会，一部电影就下载回来了。然后就改 MPlayer 上场了，他先拿过片子看看格式，是 ogg 的，然后掏出了相应的解码器。解码器是干什么的？要知道，片子的格式很多，就好象现实中看电影，有数字电影，就要用数字放映机。胶带的，就得拿传统的放映机。在家里看光盘的，就得拿 DVD 机，看录象带的，就得拿录像机。MPlayer 就像个电影放映员，解码器就是放映机。放映员在怎么牛，也得有各种放映机做支持，没放映机他啥也放不了。Mplayer 掏出解码器，开始放起影片来。这就完了么？还差得远

呢，要想让主人看上片子，还少不了图形部门的支持。

图形部门主要负责给主人显示漂亮的图形界面。他们那的老大叫 Xorg，他会跟硬件打交道，会用显卡（当然，用显卡也得经过我），能在显示器上画东西，想画什么画什么，谁要想显示点东西给主人看，都得经过他。要想跟 Xorg 打交道，在显示器上显示出图形来，得懂他们图形部门的黑话——学名叫协议。他们说话使用一种叫做 X 的协议，这个 X 不是牛 X 的 X，也不是傻 X 的 X，而就是 XYZ 的那个 X，XP 的那个 X，反正就是个 X。要想显示图形，就得用这种黑话跟 Xorg 去说。每一个要显示图形的程序都得会这种黑话，比如狐狸妹妹，要显示东西，就说：“驼子碗，筛土的抛闪！”那意思就是说画一陀黄色的便便—b，当然，这就是比方，其实我可不懂他们的黑话。

（这一点不像查皮，他本身兼职负责画图形）那么 Mplayer 要画什么直接跟 Xorg 说就行了么？其实也行，那就像是在字符界面下看片了——没有窗口，图像没法移动，没法全屏，没法最小化等等。MPlayer 只负责放片，像画窗口啊，移动窗口什么的这些事情他可不管。那谁来管呢？这时就需要一个窗口管理器，我们这里的窗口管理器叫做 metacity（就是 Gnome 下的默认窗口管理器）。MPlayer 要放什么东西其实是跟他说的，比如 Mplayer 说：“画一只猪”（当然是用 X 黑话，我给翻译过来了）于是 Metacity 转头告诉 Xorg：“在某某位置画个方的窗口，在里面画一只猪。”过一会主人觉得 Mplayer 方的片挡着他和 MM 用 Pidgin 聊天了（那是，猪哪有 MM 好看呀），就把 Mplayer 的窗口挪了挪，于是 Metacity 又对 Xorg 说：“把刚才那只猪和窗口往左移动 3.2 厘米。”这个过程 Mplayer 是不知道的，他只管专心的向 Metacity 描绘着影片中的一幅幅图像：“猪，走路的猪，跑动的猪，跌倒的猪，捆绑的猪，烤熟的猪……”

### (23) 信封

《Big Buck Bunny》还有点意思，里面那只看上去笨笨的大兔子是我见过最可爱的兔子了，或许可以考虑以后让他来代言笨兔？不过时间短了点，10 分钟的时间一会就过去了，主人看完了片，叫 Mplayer 去睡觉去了，然后继续拉来 Pidgin 跟 mm 聊天。Pidgin 这个家伙其实就是个送信的，大家都喜欢根据他的发音叫他“皮筋”，但是他不管送那种长篇大论的 Email，而是负责发短消息，（short message）也有叫短信的。不过别误会，这可不是说手机的短信，而是像 msn 啊，qq 啊这样的即时通讯的消息。

这些聊天软件的工作都是送信，使用者把要说的话写成信给他们，他们把信放在信封里——这个过程叫打包，然后把这个包发送给对方的软件。对方软件拿来这个包，先要拆包——也就是吧信从信封里拿出来，然后把里面的内容显示给用户看。可是这些软件互相之间是不能通信的，msn 不能给 QQ 发消息，反过来也不可能，这是因为他们的信封——打包方式不一样。比如 msn 的信封可能是从上面拆开取信；QQ 的信封则是从侧面拆开取信；Gtalk 的信封可能是用订书器订上的，需要拆钉取信；而百度 Hi 的是用胶水粘上的，需要涂水溶胶取信。反正各有各的高招。那么皮筋呢？他全会！皮筋跟狐狸妹妹一样，也有很多的插件——其实就是一本本 XX 信封封/拆手册。Msn 的手册上，那就写着怎么封 msn 模式的信封，怎么拆 msn 模式的信封。皮筋只要拿来一看就明白了。gtalk 手册也是如此，所以，pidgin 可以支持很多种聊天软件，只要有相应的插件就行，不用再同时开着 gt, 开着 msn, 开着 qq 了，只要开一个 pidgin 就都能聊了。不过 qq 的信封比较特别，其他的聊天软件都使用公开的协议——至少能实现基本功能。有专门的文件写着自己只收什么什么样的信封，比如必须蓝色，上面印着蝴蝶，上开后直接撕开的信封才能发给 msn。可是 qq 这家伙的信封却很复杂，而且保密，别人都不知道具体应该怎么封。上面乱七八糟的有很多防伪标识，什么激光啊，磁条啊，比人民币还热闹。所为达到的目的就是只有自己的 QQ 软件能有互相通信。不过，强中自有强中手，人民币还有 HD90 呢，QQ 的信封怎么就不能伪造了？有牛人通过研究 QQ 的信封，慢慢分析，已经仿制出了 QQ 的数据包，可以实现最基本的文字聊天的功能了，这就是 pidgin 的 QQ 插件。但是功能相当有限，用起来不好使，所以多数人还是安装了 QQ 官方的软件，我主人也是这样。

## (24) 酒

我静静的望着你，张口对你倾诉，你却听不见我的言语，直到……你喝了那杯酒。以前说过，我是不能跟查皮那屋的软件们交流的，然而，今天来了翻译。超级牛力在主人的要求下拉来了一杯红酒。不过这并不是因为主人想晚饭改善一下生活，而是由于狐狸妹妹的一次碰壁，越说越乱了吧，没事，咱们从头慢慢说。话说那一天，主人想看看自己的工资卡里的余额是否按时增长了，又懒得跑出去，所以就让狐狸妹妹到银行的网站去看看。狐狸妹妹迅速的到了网站，却发现网站用一种叫做 ActiveX 的语言问她银行卡的密码。密码当然会由主人告诉狐狸妹妹，可是，怎

么能翻译成 ActiveX 语言告诉那个网站呢？狐狸妹妹一下子抓了瞎，没学过阿！再说了，这 ActiveX 语言是那个有点软公司发明的，想学可不容易，估计得交一大笔学费，人家都还不一定愿意教你——因为到现在为止，只有有点软公司亲生的 IE 同志才能够懂这门语言。狐狸妹妹急的翻遍了自己所有的插件，也没发现有哪个能用来把主人的密码告诉这个该死的网站。急的狐狸妹妹差点哭出来，可是着急也没用阿，也只能灰溜溜的回来告诉主人——这搞不定！

于是，目前处于这样一种情况：主人必须去那个该死的银行网站，而能够去那个网站的，只有 IE，可 IE 压根也听不懂我说话，他只给查皮打工。难道就为了看一眼余额，要把我哄回床上，让查皮来干活吗？那可要重新启动电脑阿，太麻烦了。可是谁又能把查皮叫醒并让他去干活呢？这个时候，超级牛力跑出来说：我有办法，本 APT 有超级牛力，有人能把 IE 叫醒，我去找他。说着，就跑出去了，转眼间领回来一杯红酒——WINE。

当然，说是红酒，只是因为他的名字，其实他当然不是红酒，而是一个软件，一个有特异功能的家伙，一位催眠大师。他来到这里，问了问情况，我把目前的问题跟他说了，任务很简单，就是把 IE 叫醒去干活。红酒大师点点头，拎起自己的工具包就走进隔壁查皮的屋里去了。只见红酒大师站在正在睡觉的 IE 的旁边，用低沉浑厚的声音向 IE 念着：现在计算机正在启动~~我是 WindowsXP~我是 WindowsXP~你要开始工作~~你要慢慢醒来~你要慢慢醒来开始工作~~醒来~醒来~~~ 随着他一步一步的引诱，IE 慢慢的睁开眼睛，迷迷糊糊的起来，慢慢的走向工作间里，他一边走，红酒大师一边跟在他左右不停的引导：你像每天一样起床~正走向内存里~开始你的工作~~ 然后扭头问我：“让他去干啥？”看的快入迷的我才反应过来，还没交代清楚具体的任务呢，赶紧说：“哦，让他去那个银行的网站。”大师继续对 IE 说：现在~~Wdinwos 让你去银行的网站~~~去吧~~去吧~~~~像每天一样~~~~ 这个时候，基本上所有人都看呆了，没想到这 IE 竟然就这么被大师指使着干活去了，大师果然是大师阿。

## （25）酒 too

今天丢人丢大了！

想我大名鼎鼎的火狐狸，什么网站没去过？什么网站搞不定？什么 Konqueror，Epiphany，lynx，除了那个不大懂事的挪威妞 Opera 以外，哪个浏览器见了我不得叫

声大姐？在线视频？行！Flash？没问题！有我一身的插件，那是兵来将挡水来土掩，可是今天，竟然就有网站我拿它没辙！

这破网站是个银行的网站，要说那好多国际知名的银行我也见过，人家啥软键盘阿，HTTPS 加密也都挺好的，也没见用什么 ActiveX，也挺安全的阿，怎么就这破网站非得用 ActiveX 呢？我其他的都会，就不会这 ActiveX，这不是诚心揭短么。再说了，不会也不是我的错阿，我倒是想学呢，谁教我阿，人家微软才不会把这个教给我呢，藏着掖着还来不及呢。结果可好，我没法搞定这网站阿，主人只好叫别人来，这不是抢我饭碗么！当然，叫什么 Konqueror, Opera 阿这些个来也是白费，只有 IE 才行。可按说 IE 他不能听我们头的阿？嘿，还真是什么高人都有，让超级牛力找来个催眠大师，居然就把 IE 整的服服贴贴的，老老实实去干活了。不过，毕竟 IE 不是在清醒状态，基本上跟梦游似的在那干活，虽然没出什么错，可动作慢了不少。主人也只能姑且忍受一下，看来我的饭碗还能保住。我本来想偷偷跟他学学 ActiveX，可是他嘴里念叨着叽哩咕噜的东西我也听不懂阿，当然，其他人也听不懂，只有红酒大师能懂他的话。我们头发送命令，只能先告诉红酒，再由红酒翻译给 IE。IE 要什么东西，什么 DLL 文件阿，配置文件阿，红酒都赶快给他找来，原版的找不来就自己做一个差不多的，放的位置都跟 IE 在查皮那里干活时的位置差不多，让 IE 以为自己还是在查皮那里干活。既然用 IE 登陆该死的银行了，我就没什么事干了，正好本小姐还能歇会，哼！等了半天，IE 才把事办完，我都睡了一觉了，主人赶紧把 IE 关了，还是叫过我我去其他的网页，要是我我也得关，看着他就难受，哪有我看着顺眼阿。我麻利的赶快开工，赶紧表现表现，一定要和那破 IE 形成鲜明的对比。看看主人想看点啥？哦，要查查红酒大师还能干什么，好赶快去找狗狗哥……

## (26) 酒 Again

唉～最近呀～不知道怎么得了，我这个脑子出问题了吧还是怎么的，怎么混混沉沉的呢，而且好像记忆还不老好的了，要不就是有幻觉。我隐约记得昨天明明起床干活了，好像是去了个什么银行？不过记的不怎么清楚，模模糊糊的感觉，好像做梦一样。我以前不这样啊～我可是明门之后，血统纯正，我祖上从来也没有失忆的毛病。想当年啊～我的前辈 IE5 那时候就跟着 Windows98 混了，那时候有个家伙叫 Netscape，觉得自己挺牛，基本上那时候上网的都得用它。可是呢～哼哼～有本事不如靠靠山，我

前辈 IE5 老先生虽然论本事……比不上那什么 NetScape，可是他聪明啊，死粘着 Windows98 老大，98 去哪他去哪，有这强大的后台，慢慢的大家都开始用 IE5 了，NetScape 从此销声匿迹。后来的 IE6 也是如法炮制啊，棒着 WindowsXP，后来我 IE7 横空出世了，就取代了 IE6 的位置，换我跟着 XP 干。咳，怎么说起这些了，看来脑子真是不行了啊。我明明记得昨天去了银行的网站，还是 XP 老大叫我去的，可是今天我问 XP 老大，昨天你说话声音怎么有点不对劲呢？是不是感冒了？老大斜眼看看我问：昨天？哪有活阿？我说不呀，昨天不是你让我去那个什么银行查余额么？老大直接扭过头，扔下一句：“做梦呢吧你。”我挠挠头，难道我正的是在梦游？说是梦，却很清晰，说是真的，可还有点模糊。或者……那是我前世的记忆？前世……靠，为啥我前世还是浏览器？！等等，我前世要是浏览器的话……难道我前世就是那个 NetScape？！不行，越想越晕了，在这样下去非精神分裂不可。想办法找人聊聊诉诉苦吧，老大反正不理我，去找 cmd 聊聊吧，他是专门负责跟人聊天的，问问他我这到底是怎么回事。他说我是参数打错了，唉，他也不知道别的。问问游戏组那哥几个，扫雷说我是踩着雷了，空档接龙说我是牌放错了，这都哪跟哪阿~再去问问记事本，直接被嘲笑，说我这天天上网见多识广的，竟然还来问他这么一个大门不出二门不迈的抄写员。唉~想想也是，看来只有我不正常了。正灰心呢，那个长的恐怖的 War3 来了，神神秘密的对我说：“我也梦游了……”我惊讶的望了他 3 秒钟——难道他……传染的我！！

## (27) Year

时光如梭，转眼间，又到 4 月了，一年的时光就这样伴随着一条条指令，慢慢的流逝了，我，已经一岁了。对于一个操作系统来说，一年或许很短，但也可能很长。或许，一年，就是一生——尤其对于跟新换代很快的 Ubuntu 来说。Canonical 的毕业生是半年一届的，每年 4 月和 10 月是学生们从学校毕业的日子。不过也有特例，2006 年春天那批由于不太老实，延期毕业了两个月。继我们那批之后，已经有过一批 8.10，而现在，最新的 9.04 也马上那个就要奔赴各自的工作岗位。说来，我也算是他们的前辈了。想想以前在学校的日子，还真是美好。Canonical 学校其实是有几个不同的专业的，不光是我们 Ubuntu。我们是学校最热门专业出来的学生，除了我们之外还有很多其他的专业，比如 Kubuntu, Xubuntu, Edubuntu

Kubuntu 专业出来的学生似乎比我们更有些艺术细胞。他们的样子要比我们好看些，精致些，而我们 Ubuntu 比较主张简洁明了。他们的桌面环境请来的是 KDE 团队，所以叫做 Kubuntu，而不像我用 Gnome。说起 KDE 和 Gnome 的争论，大概说个两三天也说不完，两者都是桌面环境，都是用来和人来交流的。KDE 更愿意把各种部分的设置能力交给用户，让用户可以随心所欲的把自己的桌面改成想要的任何样子。而 Gnome 则注重简介，当然，也可以进行一定配置，不过就比较麻烦，需要写写配置文件。由于桌面环境不一样，附带的软件也有所不同。Kubuntu 有 kopete 来聊天，而我们这里是皮筋，Kubuntu 写文档用 Koffice，我们这里是 OOo 老先生。不过，这只是默认的情况，其实在我也可以让超级牛力找来 kopete 干活，代替皮筋，只是个人喜好不同而已。

Xubuntu 专业的，都是准备去艰苦环境下工作的志愿者。用的桌面环境就跟我们都不一样了，他是用 XFCE。XFCE 的特点就是小巧，占用资源少。可以在很艰苦的硬件条件下很好的工作。比如内存，Xubuntu 能够在 200M 内存的机器上跑的比较流畅，这要是换了我，还不得郁闷死。才 200M 啊……这狐狸妹妹带着一身的扩展一进去就得好几十 M，200M 哪够用啊。可是人家 Xubuntu 就够。不过，相应的软件也要用一些轻量级的，要是也请个挂满扩展的狐狸妹妹，那系统本身省吃俭用节约下来的那点内存，还不够她一人用的呢。

Edubuntu 是教育专业出身，用的桌面环境跟我一样，只是他附带了很多搞教育的软件。比如教小孩子打字的啊，画画的啊，教授一些物理知识的啊，这些软件都是很好的老师，很多小孩子用起来都乐此不疲。很多小游戏也都是寓教于乐的，家长们也可以不用担心孩子用这个系统沉迷于游戏（因为没啥可沉迷的游戏……）只是这些软件多半都是英文的，所以，非英文地区的孩子们用起来还是不太合适。

除了这些之外，还有很多其他专业的 ubuntu，那些专业都不是 Canonical 学校自己开的，不过也都是用的一样的教材，大家互相都是通用的，只是所带的软件不同而已。今天不知为什么说了这么多，或许是因为有些伤感，因为最近主人总让狐狸妹妹去 Kubuntu 的网站转悠，打听 KDE4.2，打听 Kubuntu 9.04，难道……

## （28）人物志

今天说说星爷吧。



这里不是娱乐周刊，我要说的自然也不是周星驰，我要说的是星际译王。他来自中国，是我们这里少数来自中国的软件之一。他是我们这里最有学问的人，简直是什么都知道。一开始他只是懂点英语而已，大家都只当他是英语翻译，后来主人叫狐狸妹妹去给他找来各种各样的书给星爷看——就是那种叫做字典的书，这种书只有星爷能看懂。看了这些书以后，星爷知道的就多了，什么日语啊，法语啊，德语啊，都会了，估计联合国要开会请他一个人去当翻译就行了。于是星爷从英语翻译一下子晋升成为了地球语翻译！（地球上的语言都会—b）不知道他以后会不会再学学火星语？

然而星爷是不仅仅满足于当地球语翻译的，或者说，主人是不满足于让星爷只做个翻译的。这之后他又给星爷找来了本《陈义孝佛学常见辞汇》，于是星爷开始研究佛学了，不过这东西只是主人一时的好奇而已，后来就再没给星爷找过佛学的书，而是找了本《五笔 98》，开始学习五笔了。要说这输入法的事，那可是 scim 的本行啊，可是无奈 scim 老弟干活还行，表达能力不强。主人要打什么字，scim 可能很快反应过来，打在屏幕上，可是主人要是忘了某个字怎么打，问问 scim，那可要了命了，打死他也说不清楚啊。主人只能问：“是 a 开头不是？”然后 scim 把所以 a 开头的显示出来数一遍，摇摇头：“不是”，然后主人再猜：“是 s 打头？”scim 再把 s 打头的列一遍……这样实在太费事，于是主人就让星爷学五笔，到时候就能去问星爷：“这个……‘我’字用五笔怎么打啊？”星爷会投过一个鄙视的眼神：“你还好意思说会五笔啊，q 空格！”

这还不算完，后来星爷又看起了《本草纲目》，研究起了中医。不过他还不能当大夫，针灸号脉啥的他不会（就算会也没法号啊），那会啥呢？看了《本草》当然最精通的就是药理了，随便说一种药，他就能告诉你这个药的药性，药效，怎么用，等等等等。这时候基本上我们已经对星爷的全能感到震惊了。后来他又开始研究古汉语，看古汉语词典，康熙字典，整天到晚的跟我们这之乎者也。“夫内核者，老大也，发其命，出其令，而统‘康皮右特儿’（computer—b）……”然后就是我们集体的“打豆豆”时间——谁是豆豆就不用说了吧。

我也趁没人的时候，问过星爷：您怎么懂这么多呢？看什么会什么。星爷很神秘的笑笑说：其实他没什么本事，就是在学校学过信息检索而已。主人给了那么多本书，要想都记住，根本不可能嘛，他只是每次在主人问起事情的时候，赶快现去查书，用最快的速度找到并告诉主人。要是没了这些书，他知道的就少很多了。不过也不至于离开了书就什么都不知道，现在的星爷学会上网了，可以连接到一种叫网络辞典的地方，

自己不会可以去网上查，不过那样效率自然不如自己翻书快了。

## (29) 日志

自从红酒大师来了之后，查皮起床的次数比以前少了很多，我的工作更繁忙了，大多数工作，主人都交给我去做。每天大家都忙得不可开交，新来的奔流整天忙着下载各种大个头的东西，什么电影阿，软件阿，什么都有。随时都跑过来找我：头，我下了20M，先存硬盘里。我说：好，存那吧，赶快。转眼，Mplayer 又过来：头，我要读那个电影。刚找着电影递给 Mplayer，主人又发命令要我把 U 盘里的一堆文档搬到他的文档目录里。正搬着呢，奔流又过来了：头，我还要存 20M。我一边搬着文档一边指一块地：恩，你就存 N&@#%……

我睁开眼，看见了熟悉的 GRUB 大叔，他拍拍我：嘿，醒醒，开工啦！——唉，每次都是这句。我揉揉眼睛，觉得头有点涨。发生什么了？怎么屋里有些乱？仔细回忆一下……哦，我好像正在干活，然后……停电了？！恩，应该是了，那时候眼前一黑，就什么也不知道了。我当时正在搬文件，搬到哪了？恩，看看日志把。还好我用的 XFS 是个日志文件系统。什么？日志文件系统你也没听说过？唉～ 讲课。

### 笨兔兔老师第三讲——什么事日志文件系统

文件系统就是我们管理整个硬盘这间屋子的方式，这个以前跟大家说过了。文件系统有很多种，过去的文件系统都是非日志文件系统，这种文件系统比较落后。比如 EXT2，比如查皮那的 FAT。非日志文件系统在发生意外断电的时候就容易出问题。就像今天的情况，如果我这屋子用的是 ext2 的话，没准就丢个文件阿什么的，搞不好整个分区都坏掉了。为什么呢？比如我屋里有一个叫 笨兔兔的故事.odt 的文件。文件，前面说了，就相当于放在屋里的一个大箱子，里面是内容，外面写着文件名：笨兔兔的故事.odt 内容是什么咱就不管了。然后有一天主人要修改这个文件，可能往里面多写进去点东西，也可能改掉里面的一些东西。如果用非日志文件系统是怎么做的呢？很简单，主人首先找 00 老先生打开这个文件，打开，也就是把这个文件读进了内存里，然后靠 00 来在内存里修改这个文件。注意，文件不是你家的大白菜，搬到屋外那屋里肯定没有了。文件读进内存，磁盘上仍然有这个文件，内存里只是它的一个副本。好，现在，00 老先生那有这个文件改动后的版本，在内存里（就是主人还没点保存）。磁盘里有这个文件原来的版本。如果这个时候停电了，那刚才该的那些肯定都

作废，这个用什么文件系统也是一样。那如果主人点了保存，并且保存结束了，这个时候停电，那就停吧，也没事，因为已经保存进去了，除非房塌了（比如磁头挂了，盘面损坏之类的），否则不会丢。如果主人点了保存，那么 00 就要让我把内存里他写的那个副本往磁盘上存，于是我就从内存里拿过来一点，打开磁盘上那个文件，掏出里面的一部分，扔掉，用我手里这些替换进去。然后再回内存里拿下一部分，再回来把文件里的下一部分扔掉，用我手里的替换。如果正在这个过程中停电了，那就惨了。内存里的，那肯定没了，磁盘上的，有一部分被替换掉了，有一部分还是原来的，于是文件就乱了，可能损坏，格式不对，根本打不开之类的。

那用日志文件系统又怎么样呢？日志文件系统，顾名思义，就是有日志的文件系统（废话）。还是拿上面那种情况举例，00 要存那文件，那我怎么做呢？我会在硬盘上一个专门的记录日志的地方些下来：00 要覆盖 笨兔兔的故事.odt 文件。如果这个时候停电了，没事，原来的那文件还好好的，但是内存里的还是没了，这条记录也就作废。记录之后，我就开始把内存里的东西往硬盘里放——放在记录日志的地方，并不动原来的那个文件。如果放到一半停电了，那也没关系，原来的文件还好好的。修改了的那份也有一部分放到了硬盘里，不过这是一部分的话，多半还是没什么用。如果我把文件完全搬到了记录日志的那部分硬盘里，那就再在刚才记录的那条日志下面写上：已经把要覆盖的内容存到了日志去 xxx 位置，准备替换原文件。如果这个时候停电，没事，等再开机，我一查日志，就知道要修改的版本已经完全存在了硬盘里，只要按着上面记录的继续做就行了。写好日志之后，就开始用日志区的这个新文件去替换硬盘上那个原来的文件。这个过程会很快，因为其实并不需要真的搬运数据，只要在原文件的地方做上标记，表示这个文件已经作废，然后把那个 笨兔兔的故事.odt 文件名指向新写的这个文件就好了。（我们只是拿箱子比喻文件，但文件毕竟不是你家的箱子。）这样，无论中间的哪个过程断电，都不会完全损坏整个文件，要么原版还留着，要么修改后的版本已经生效，通过查看日志就能知道现在哪个版本有效。这就是日志文件系统。

### (30) XFS

书接上回

上回书说到，这笨兔子跟店房伙计似的忙里忙外脚打后脑勺阿，忽然间咔嚓一声——

停电了。那位说了，怎么回子事捏？我不说，您不知道。原来是小区电网改造，这个小区那是历史悠久，早在清朝末年……当然了，这些都是题外话，咱们暂时不表，单说这笨兔子。来电以后，只见这本兔子不慌不忙地起床，刷牙，洗脸，吃早点，看看屋里边，挺乱的呀。拍脑门一想：对了，刚才停电了。有人关心了，说这停电了，丢东西没？听过上文书的都知道了，这笨兔子用地是“插爱夫爱死”（XFS）文件系统，那可是日志文件系统阿，那就相当于宝兵器，浑铁凝钢打造，那是削铜剁铁，斩金错玉阿，那能怕停电么。所以这笨兔子照着这个文件系统地日志，前后左右那么一查，齐活，啥也没丢。

又有人可打听了，说这“插爱夫爱死”这口宝兵器是从哪得来的捏？这可说来话长了。想当初，早年间了，有这么一个硅谷图形公司（SGI），他们有个操作系统，叫做“爱阿爱插”（IRIX），这外国人起名字都各色。这系统干什么用的呢，主要是图形计算。本来他们有个文件系统叫“义爱夫爱死”（EFS），可是这不过是一件凡铁兵器，一开始拿他切菜还挺好使，后来买卖做大了，随手也越来越强，人家都那地是宝刀宝剑，你这把切菜刀，怎么跟人家比划呀。所以这硅谷图形的总瓢把子拍板，说咱这破菜刀也别磨了，你再怎么磨也是把菜刀，干脆，“义爱夫爱死”——扔了吧。另请高人，访贤士，为新版地这个打造了“爱阿爱插”系统打造了一口宝兵器，就这“插爱夫爱死”。那么说这宝兵器宝在哪捏？头一个说，就是他结实。就是不怕断点呐，就跟这笨兔子这回遇到的情况一样，忽然地断点，没事，数据不丢。再一个，什么捏，就是这兵器，速度快，那是快如闪电。有那么句话您听过没有，叫迅雷不及掩耳盗铃阿。这文件系统格式化，甭管你多大磁盘吧，你是1G也好，100G也罢，哪怕你是1T的硬盘，一眨眼的功夫，格完了。光格的快也没用，那读写文件也是不慢，尤其是越大个的文件，读写起来越有优势。还有什么特点？还有就是能伸能缩，要说小，几十兆建个分区也行，要说大，您看着“插爱夫爱死”是个六十四位的文件系统不是，最大支持多大滴分区捏，支持十八个E的分区。就是一万八千个P呀～那么这是分区的大小，那支持的文件大小最大多大呢，最大九个E，也就是九千个P。P是多少不用给大家介绍了把，1000个T阿……

那么有人说了，你把这件宝兵器夸滴跟一朵花似的，那么他就没缺点了么？他就无敌天下了么？当然不是，有道是一山更比一山高，能人后面有能人阿。说这XFS有什么缺点呢？就怕删文件，尤其是小文件，删除一大堆小文件的时候那个速度就慢死了。再一个，宝兵器一般都沉，这处理器费地多点。当然，也不是很多，只是相对多一点。

这正是金无足赤，人无完人，今天给您说完这段到下回咱们说说……说啥还没想好反正是咱们下回——再说。

### (31) 分区

我们住处的墙上有一扇门，门上面写着三个字母——USB

这扇门是我们与外界交流的又一个接口（最重要的还是网络），每次门上的红灯亮起，就说明有东西接到了 USB 上，我就得去打开门看看。有时候门外是一个小集装箱似的屋子，很小，一般只有几百兆到几个 G 大小，里面也像我屋子里一样放着一些文件。这个时候一般主人就是要让我搬东西，不是把小屋子里的往大屋里搬，就是从大屋往小屋挪。有时候一开门，外面不是一个屋子，而是一台设备，比如是一个鼠标啊，或者是个摄像头什么的，那就是让我去操作这些设备了。主人就接进来过摄像头，摄像头的名字叫 Moto E6，听说这其实是个手机，上面的摄像头可以通过 USB 接口来给电脑用，不管怎么样吧，这东西咱也会玩，接上就能用，鼠标那就更不在话下了。不过总的来说，还是往 USB 那门外接小集装箱屋子的情况多，这种小集装箱式的屋子叫做 U 盘，也有更大一点的，就是移动硬盘了。每当 U 盘接进来的时候，我就把它当作我屋里的一部分来用。

要想成为我屋子的一部分，就要起个名字，要不我怎么访问啊？就跟你家似的，什么叫厨房哪个是厕所，什么主卧，阳台，是个空间都得有个名字不是？我这里也一样，总得有个名字才好访问。这里要再说明一下我们住的硬盘。我们住的硬盘跟你住的房子是一样的，不是整个的一大间屋子大家乱住，东西乱扔。也要稍微的分出几个间来，学名叫分区。分区的多少和大小是主人在安装的时候根据需要分的，比如我住的这间，就分出了四个分区，也就相当于把整个大屋子用墙格出了四个间。一个小间的门上挂着个牌：/boot 这间里面是我住的，我的单间，作为老大嘛……有点特权也是可以理解的吧。再一个大间，门牌上写的是/home，这里是主人专门用来放他的东西的，什么文档阿，电影阿，歌阿，都放这里，所以这间特别大。还有一间，平时不放东西，这间叫做 SWAP。这间是用来给正在干活的程序用的，程序们要运行的时候不是要往工作间里放东西么，要是放不下了，就暂时放到这间里。不过由于我们这的工作间很大，所以基本上这间没怎么用过。除了这几间之外的，就是剩下的一大间了，这间叫做“/”，学名叫做根分区。其他的间可以没有，但是这间不能没有，要不我们住哪阿，

是不。不过虽然其他间可以不格出来，但是对应的空间还是得有的，比如/boot，可以不单给我格出一个单间，也就是不给我单独分个区，但是就算不格，我也得划出一块地方，拿粉笔写上/boot，表示是我住的地方。

回来说说 U 盘，U 盘接进来之后，我也把他当作一小间，并且给他门口挂个牌子，一般是/media/xxx，xxx 就是那个 U 盘的名字。一般 U 盘都会有个名字，或者叫卷标。这样一弄，就可以很好的区分各个单间了。比如要让 mplayer 放电影，我总得人家说明白这电影放在哪吧，我要是说：就在那个放了一堆主人的东西的最大的屋子里，就很罗嗦，不如直接说：在/home/lanwoniw/目录下来的简单。

## (32) 挂载

这回主人接到 USB 上的又是一个集装箱式的空间，不过还比较大，4 个 G。我仔细看了看，结果没有发现这设备的名字，那也没事，名字是人起的嘛，我直接给他起个名字并挂上了牌子：/media/4.1G 没名字是可以滴，没牌子是不行滴。这个挂牌子的过程，用我们的专业话说，叫做挂载，这个大家应该都听说过哈。挂载，就是挂牌，就是在某一间屋子的门口挂个牌子，起个名字，到时候干活的时候就好说话了。直接说屋子的名字就好了。名字，或者说牌子，就是个标志，是可以随便换的。比如有个分区被挂载到了 /home/目录，也就是说，有个屋子 A（就叫 A 吧），被挂上了/home/的牌子。那么主人说要看 /home/下都有什么，我就会把那个屋子 A 里面的东西列出一个单自来给主人看。等回头可能又把别的分区挂载到/home/下了，那很简单，就是把/home/那个牌子从 A 房间门口摘下来挂在 B 房间门口而已。主人再说要看卡/home/下都有什么，我就该把屋子 B 里面的东西列表来给主人看了。没啥经验的主人可能会大跌眼镜：哇塞～怎么我/home/下原来那些东西都没了阿!! 都哪去了阿!! 殊不知，其实原来那些东西还在 A 屋子里好好的放着，只是现在 B 房间改叫/home/了而已。

又废了这么些个话，不好意思。回来说这回接上的这个 4G 的屋子，接上之后，还没等主人发话，我先去里面查看了一下，一看全都是些个 jpg 的文件，我是看不懂这些个文件，但是我知道有人能懂，于是赶快通过图形界面那哥几个报告主人：您插进来的这个里面貌似全是照片，是不是要我给您找来 F-Spot 呢？。

F-Spot 是一个管理照片的程序，他可以帮主人把移动设备（就是那种集装箱似的小空间）里的照片导到硬盘里，并且按时间分门别类的管理好。主人想要去年三月的照片，

他马上能够给找到。除此之外，还能够汇报照片的信息，比如用什么相机照的，照的是后用的光圈，快门，ISO 都是多少，等等信息。F-Spot 虽然能够整理照片，不过不能编辑和修改，有时候主人的照片经常需要稍微的调整一下色彩阿，做一点效果阿啥的，有很多人管这叫 PS，因为在查皮那里，做这种事情的软件叫做 PhotoShop，所写 PS。逐渐的，用 PhotoShop 去处理照片这个动作就被叫成了 PS。我这里没有 PhotoShop，不过有另外的处理照片的强人——GIMP

### (33) GIMP

GIMP 这家伙是个美术家，能够画出很多漂亮的图画来。当然，漂亮不漂亮是以主人的审美观点来说的，对我来说他制作出来的那些个东西，不过是一个写满 0101001 的文件而已，和其他的文件没有什么区别。虽然他可以在一张白纸上创造出一个多彩的世界，但多数情况下他要做的作品是调整一个已经画好的图片。一般都是主人用数码相机照的照片，用 U 盘拷贝进电脑，然后叫来 GIMP，调整一下照片的亮度啊，色彩啊什么的。有时候还让 GIMP 做一些效果，比如做成油画效果的，就是把照片做成像是画的油画似的；还有做成浮雕样的，或者加个相框，他都行。以前拷进来的照片也就几百 K，可最近主人弄进来的照片最小都 2M 多，别人不知道怎么回事，可是瞒不住 F-Spot，他能看懂照片的 EXIF 信息啊。他告诉我，以前的照片是用一个叫做 Nikon 5900 的照相机照的，现在这些 2M 的都是用 Pentax k-m 这个相机照的。我让狐狸妹妹帮我去查了查这两个相机，5900 是个 500 万像素的 DC，k-m 是 1020 万像素的单反相机，怪不得照片大了很多。不过，区区 2M 的照片文件是难不倒 GIMP 的，仍然刷的一下就打开，他自己说，就算 20M 的文件也没问题，不过我们倒是没见过，主要是主人的相机实在照不出那么大的来。

GIMP 像狐狸妹妹一样，可以安装很多的插件以实现各种不同的功能和效果，论功能，不输于查皮那里的 PhotoShop。但是 GIMP 有些不大好接触，总是按着自己的性子来，让主人用起来有些不顺手，不如 PS 那么易用。毕竟人家 PS 身价不菲嘛，听说一套要好几千块钱，一分钱一份货嘛。花这么多钱请回家去的软件，怎能不服服帖帖的听主人的话呢。再看看 GIMP，总是摆着一副：我就这样，爱用不用的架势。还好主人比较通情达理，也能理解 GIMP 是确实有能力的，否则……哼哼，别看 GIMP 跟我关系不错，没准也得被超级牛力请出硬盘。

## (34) QQ

主人整来一大堆照片，用 GIMP 处理了一下，调调颜色，亮度啥的，还别说，调完了还是比以前好看了不少。不过，有道是独乐乐不如众乐乐，主人光自己看着好看不行，还想和朋友分享，怎么办呢？找人吧，这个人您大概也认识——QQ。

Qq 和皮筋一样，也是一个即时通信软件，也就是个送信的。不过他不是超级牛力从软件源里请的，而是狐狸妹妹直接去 QQ 的网站上下载的。这 QQ 是一个叫做疼痛，哦不对，叫疼……疼什么来着？哦，对，疼殉，一开始是疼，后来就殉了-\_-b，是一个叫疼殉的公司做的。话说这个疼殉啊，看人家 icq 软件玩的挺火，于是也弄了个 oicq，抢占了国内市场，结果一发不可收拾。后来 oicq 改名，叫 QQ 了，可是一直以来，由于各种原因，疼殉这个公司只能做出 wdinwos 版本的 QQ 来，linux 下的没有。要说没有也不要紧，人家 google talk 也没有 linux 的版本，但是人家是基于开源的 XMPP 协议的，协议是公开的，于是世界各地的 linux 牛人们，很轻易的就做出了很多种用来在 linux 下聊 googlt talk 的软件。其实就算他们不做这些，皮筋本身也是支持 XMPP 协议的，设置一下就能聊 gatlk 了。可是 QQ 不一样，QQ 的协议是那个疼殉公司自己定的，还不让别人知道，又不提供 linux 的版本，结果，在 linux 下使用 QQ 一直是个很头疼事情。当然，这些都是历史了，现在疼殉公司终于想开了，提供了 linux 版的 QQ，虽然功能简陋的不能再简陋，不过，传个图片还是没问题的。好，废话不多说（这废话就不少了），赶紧叫 QQ 起床干活去。

QQ 迷迷瞪瞪的走进工作间，把主人写的文字一条条的打好包，封好信封，寄给那边的 QQ。过了一会主人要发图片了，QQ 有点忙乱，好像对寄图片这工作不如寄文字来的顺手，不过好歹是寄出去了。mm 那边也寄过来两张图片，QQ 收下了打开给主人看，主人很满一，让 mm 多发几张，于是 mm 一口气发了 5 张，然后……QQ 就晕了，折腾好几次也没法和对方的 QQ 建立好连接，结果照片终于没传成。要说这家伙也真是不争气，这么点事情都做不好。主人一气之下，只好把 QQ 关了，让我去叫醒另一个人——EVA

EVA 的大名相信大家都听说过，他的英文名字叫 EVA，他的中文名字叫新世纪福音战士。话说日本有个贞本义行……哦，对不起，扯远了。忘记贞本义行和新世纪福音战士吧，他们跟我这里的 EVA 没关系，跟我这个 EVA 有关的是云帆大姐姐。在疼殉公司



还没有给出 linux 版本 QQ 的时候，很多人们就用 eva 来聊 qq，这个 eva 完全是云大姐根据抓包研究的结果，黑盒破解 windows 的 QQ 而编写的。功能也算是强大了，基本的聊天，传图片，群里收发自定义表情，都可以。甚至还支持显示好友的自定义头像。（这看似简单的功能，腾讯官方的 QQ 目前都没实现）于是，在 QQ 不靠谱的时候，主人还是叫来了可靠的 EVA 继续跟 mm 传图片。QQ 跟 EVA 有很多不同，一个是官方的，一个是山寨的（山寨不含贬义）；一个是基于 GTK 的，一个是基于 QT 的；一个是闭源的，一个是开源的；一个是 32 位的，一个是 64 位的。

### (35) 运算

有人问了，你老说这 32 位，64 位。到底啥意思阿？

这个多少多少位，说的是 cpu 一次运算的二进制数字的位数。这个 CPU 就像是个计算器，我们软件用 CPU 就像人类用计算器似的。它很重要，我们要算一丁点东西，也需要用 CPU 来算。（别跟我说用心算，我是软件，ok?）那么这个 CPU 算东西的能力，是有限制的，有什么限制呢？你拿出你家的计算器看看，算个  $28+783$ ，没问题是吧。算个  $7836-473$  也没问题是吧，再算个  $72635446584939202937346537+1$ ，能么？估计 99% 的同志出问题了（不排除有牛人拥有很牛的计算器）：“我哪能按出这么多数来啊，我这计算器总共就能显示下 11 位数字”。对，这就是计算器的位数限制。CPU 也一样，他一次能算的数不能无限的大，总得有个边，只不过不是按照十进制的位数算的，而是按照二进制的位数算的。至于什么叫十进制，什么叫二进制，可以去问问狗狗大哥，不过不知道也没关系，咱暂时按着咱们平常的十进制来说。比如说，我这个 CPU 只能算 99 以内的数字，也就是只有 2 位（十进制位啊）。那我们软件用这个 CPU 的时候怎么用呢，CPU 有很多放数据的小匣子，叫做寄存器，每个寄存器有他特殊的用途，咱们就不多介绍了。要做加法的话，得这么操作：有两个寄存器，也就是小匣子啊，把这两个小匣子打开，往里面放数据，数据比较抽象，就想想成写着数字的纸条吧。不过，由于是 2 位的 CPU（再次声明，咱这是拿十进制做比方啊，真正的 CPU 没这样的），所以寄存器里只能放 2 位的数据，也就是说，纸条上只能写 99 以内的数字放进去。那好，我写一张 12，放在 A 匣子里，再写一张 9，放在 B 匣子里，然后按个写着“加法”的按钮，只听咔嚓一声，CPU 自动弹出一张纸，纸上写着 21，这就是他的计算功能。再算个大点的，写一张 50 放进 A，写一张 51 放进 B，按钮，咔嚓，出来张纸，写

着 01。那位说了，这算错了啊这个！别急，紧接着咔嚓一下，又出来一张，写着“对了，还得进一位”，这回对了吧。为什么呢？因为这 CPU 是两位的，只能输出两位数，超出的就告诉你得进位。

好了，基本的操作说完了，现在说正题，不同位数的区别。两位的 CPU 就像刚才说的那样，那么假设现在需要计算  $3173+644$ ，这里有 2 位的 CPU 一个，4 位的 CPU 一个，分别用他们做这个计算，有什么区别呢？

咱先拿这两位的，有人说了，两位的只能算两位啊，这个没法算哪？唉，这机器是死的，咱软件是活的啊，一次只能算两位，咱不会分开了多算几次么。首先，写一张 73，写一张 44，按钮，咔嚓，出来一张 17，咔嚓又出来一张写着还得进位，好，可记住了啊，还得进位。然后再写一张 31，写一张 6，按钮，咔嚓，出来 37。别忙，没完，刚才还得进位呢么不是，再写一张 37，写一张 1，按钮，咔嚓，出来 38。好，最后结果拼一块，高位是 38，低位是 17，最后结果：3817

再拿这 4 位的算算看。4 位的就意味着输入的数据和输出的数据都可以是 4 位，也就是说我直接就可以写一张 3173，写一张 644，放进去，按钮，咔嚓，出来一张 3817，算完收工～

### (36) 位

这就是 2 位的 CPU 和 4 位的 CPU 的不同，从理论上来说，4 位的要比 2 位的快，从上面的例子看的很明显嘛，大一点的数，4 位的 CPU 一下就能算完，2 位的 CPU 要折腾好几次。但是这 4 位的 CPU 还得有人会用才行，这就需要 4 位的软件来用着个 4 位的 CPU。

终于说到软件的位数了，CPU 的位数就是一次能计算多少位的数，那软件的位数呢？就是说明这个软件需要使用多少位的 CPU。软件干活肯定需要计算，计算就得用 CPU，2 位的软件会用 2 位的 CPU，4 位的软件就会用 4 位的 CPU（还是拿十进制位做比喻啊）。比如有一个 2 位的软件（就说明这个软件会用 2 位的 CPU），那么当这个软件运行在一个 2 位 CPU 的电脑上的时候就是这样：还比如要算  $3173+644$ ，他就会先算  $73+44$ ，然后记住进位，然后计算  $31+6$ ，然后加上进位，最后拼起来，得到答案，就像上面描述的那样。那么当这个 2 位的软件运行在一个 4 位的 CPU 上的时候会怎么样呢？他会先算  $73+44$ ，然后记住进位，然后计算  $31+6$ ，然后加上进位，最后拼起来，得到答

案……有人说了，他怎么不直接算啊？4 位的 CPU 不是能直接就算出来么？但是别忘了他是两位的软件啊，他不会用 4 位的 CPU，但是不会用不等于不能用，他还是可以那 4 位的 CPU 当成 2 位的来用，只是有些浪费而已。那么要想完全发挥 4 位 CPU 的性能怎么办呢？当然就得 4 位的软件出场了。当一个 4 位的软件运行在一个 4 位的 CPU 上时怎么计算  $3173+644$  呢？大家大概都知道了，直接算，一次完成。那么当一个 4 位的软件运行在一个 2 位的 CPU 上时会怎么样呢？这个软件会写个 3173 的纸条要往 CPU 的寄存器里塞，急的满头大汗就是塞不进去，最后一甩手——不干了，这破 CPU 没法用！当然，这只是个比喻，并不是说 4 位软件在 2 位 CPU 上算  $3173+644$  就算不了，算  $1+1$  就能算。4 位的软件是根本无法运行在 2 位的 CPU 上的。

### (37) 协作

64 位的 EVA 熟练的使用着 64 位的 CPU；同时，32 位的奔流也在使用的同一颗 CPU；（当然，是当成 32 位的用。）同时，皮筋也时不时的汇报一下主人的 MSN 和 GTalk 上的好友是否有消息发来；同时，狐狸妹妹也没闲着，游走在个个网站之间；同时……总之，内存里大家各司其职，一派繁荣和谐的景象。而这和谐景象的背后，是由于我认真的学习了 XXX 思想，XXX 理论，并且还戴了三个表。 -\_-b  
好吧，其实之所以大家能够如和谐的同时工作，都因为我是一个多任务的操作系统。什么是多任务呢？直观的说，就是你能一边聊天，一边看电影还一边打字。（什么？你说你不能？那是因为你的大脑不是多任务系统。）有的人要说了，哪个电脑哪里系统不能一边聊天一边打字了？这说来又话长了，话说很久以前，还是那有点软的公司，在查皮的老祖宗问世之前，有点软公司赖以起家的，是一个叫做“剁死”（DOS）的操作系统。这个操作系统就是单任务的，也就是说，同时只能有一个软件在内存里运行。难道多让几个程序跑进内存里很难么？答案是——没错。我们工作用的内存阿 CPU 阿，都是很重要的资源，尤其 CPU，一个 CPU 同时只能有一个程序在用（现在的多核心 CPU 对程序来说就是多个 CPU），如果要让很多程序同时跑进来一起干活，就一定要对 CPU 进行合理的分配。剁死系统就比较简单，基本不管分配的事情。比如主人要启动狐狸妹妹（那念头当然还没有狐狸妹妹，咱就打个比方），如果是剁死系统的话，他就会跑去叫醒狐狸妹妹，然后跟她说：狐狸阿，起床干活了，你看咱这有一个奔腾 166 的 CPU，16M 的内存，够你用的不够？狐狸说，够了。然后剁死就说，那好，你去干活吧，

我就不管了，干完了叫我。然后剁死就睡觉去了，整个机器归狐狸妹妹控制。所以不可能同时运行两个程序嘛。

那多任务的系统又是怎样的呢？比如我和隔壁的查皮，都是多任务的操作系统，我们不会把整个计算机的所有资源都给一个程序用，而是进行合理和规划。还比如叫狐狸妹妹，我会去跟她说：狐狸阿，起床干活了。狐狸妹妹会起来跟我说，好，我现要 10M 的内存。我说检查一下内存空间，然后告诉她，可以，那一块 10M 的地方给你用。然后狐狸就走进工作室，开始工作的时候，一定要用到 CPU，需要用的时候狐狸要找到我，向我提出申请。我根据情况，看现在有没有人正在用 CPU，要是有的话就让狐狸等一下，没有的话就给她用。但是给她用也不能就让他一直用，只能让她用一会，因为还有别的程序要用。这个“用一会”的时间，专业的说法叫做时间片。每个运行着的程序都轮流“用一会”，也就是每个程序都分配一定的时间片。没有分到时间片的程序就等着，不过这个切换的时间是非常短的，在主人那里根本感觉不到程序等待使用 CPU 的时间的，所以在主人看来，就是多个程序一起运行了，也就是我们所说的多任务。

多任务的实现也有不同的模式，有协同式多任务，和抢占式多任务。

协同式多任务，需要每个正在使用 CPU 的程序主动放弃 CPU 控制权，并由操作系统再次分配。如果我是个协同式多任务的操作系统，那就是这个样子的：狐狸妹妹用了一会 CPU 说，好了，我暂时不用了，去网口等个数据包去。兔子哥你让下一个程序用吧。然后我就回收了 CPU 的控制全，扭头一看，皮筋那里等了半天了，就把 CPU 给他用，他用了一会，说，好了，我一会再用，先让下一个程序来吧……就这样，大家互相谦让，内存里一派繁荣和谐的景象。这主要是因为我学习 XXX 思想，XXX 理论……戴了三个表。不过这样做得缺点就是万一有个程序不和谐就坏了。比如狐狸妹妹用了半天了，我跟她说：狐狸呀，你看，你用 CPU 都用了 1 秒了（对于我们程序来说，1 秒已经是相当长的时间了）是不是该让其他的小朋友们……哦，不对，是不是刚让其他的程序用用阿？狐狸扭头斩钉截铁的：不！于是我也没办法。如果狐狸始终不能放开 CPU，那其他程序就一直等着，直到天荒地老，沧海桑田，直到机器重启，直到小区停电。抢占式多任务是怎么样呢？就是由操作系统决定什么时候收回 CPU 的控制权，而不是靠程序主动放弃。这种方式的核心就是一个字——抢！如果我是个抢占式多任务的操作系统，其实不用如果，我就是个抢占式多任务的操作系统。那么情况就是这个样子的：狐狸妹妹用了一会 CPU，我对她说，你本次使用 CPU 的时间已到，立刻停止使用

并重新排队。然后狐狸就乖乖的交出 CPU，排到队尾等待下一次使用 CPU。我则让下一个程序来使用 CPU，使用了一段时间后，我又让这个程序停止使用，让再下一个来，如此循环往复，一派繁荣和谐的景象，这主要是……思想……理论……还戴三块表。当！哎哟～

### (38) 加速

转眼又是七月流火。茶余饭后，深巷树下，多了摇着蒲扇乘凉的大爷大妈们，享受着空调房里不曾有的惬意，闲谈些锅台灶上天天见的琐事。天热了，主人也不那么忙活了，只让狐狸妹妹去到一个叫啥马铃薯的网站找些电视剧来看看。CPU 的使用率也降到了很低，我估计一时半会不会有大的运算量了，就把 CPU 关到了最小的频率。是的，我当然知道怎么关，连调整 CPU 工作频率这点事都做不来，还叫操作系统么？

主人一直在看电视剧，也没啥别的事情干，于是我也跟着看看是啥内容。狐狸妹妹介绍说，是一个叫做仙剑奇侠传 3 的游戏改编的电视剧。我隐约记忆起来，隔壁查皮那屋里就有这游戏，前一阵子还让红酒大师尝试去搞定他，红酒大师费了 7 瓶酒（他的秘密终于被我知道了，哈哈，不知道怎么回事的一定要去看看首页 PDF 版的笨兔兔）结果终于还是没搞定。那家伙晕头晕脑的非要找什么 DirectX，那是查皮的私人物品，我们哪里给他弄去啊。我问红酒大师，他也没法复制出那东西来，实在是太复杂了。相信大家对 DirectX 都不会陌生，但凡在 windows 下玩过游戏的都应该知道，没他你啥也别想玩。（当然，纸牌扫雷级别的除外）那么 DirectX 到底是个啥东西呢？他也是个软件，他是个给其他软件提供综合的图形图像以及音频加速的软件。我们说过，在查皮那里，画图的工作有查皮自己负责。那么，查皮会画什么呢？其实他只会画简单的图形，比如点阿，直线阿什么的。如果一个游戏软件要画些复杂的东西怎么办呢？那就得由那个软件来把要画的东西分解成简单图形，然后告诉查皮，让他画。比如那个叫仙剑奇侠传 3 的游戏，他想在屏幕上显示一个苍蝇拍，要是没有 Direct，就得跟查皮说：画一条长 xx 的黑色横线。然后查皮去画。之后仙 3 再说：再画一天长 xx 的黑色横线，在刚才那条线下边 yy 那么远。然后查皮再去画。再之后仙 3 再再说：再再画一天长 xx 的黑色横线，在刚才那条线下边 yy 那么远。然后查皮再再去画……………于是，一个苍蝇拍的拍头就把查皮累得半死了。那 Direct 会干什么

呢？他就是能够画一些高级的东西，能够快速地把要画的东西分解成简单的线条，然后操作显卡去画。（当然，要操作显卡还是离不开查皮，毕竟一个软件不能越过操作系统直接控制硬件嘛）于是，有了 Direct，仙 3 再要画苍蝇拍，就可以直接跟 Direct 说：画个 16x18 的网格，黑色，间隔 xx, yy 长度 zz, ll 这样，就节省了很多时间，也省去的其他软件的许多工作。

“这家伙听起来挺厉害嘛，可惜只是查皮的人，你这里没有。”是的，我这里没有 Direct，但是，我有 OpenGL……

### (39) OpenGL

OpenGL——Open Graphics Libaray

看名字就知道是一个图形库。其实，他要跟 Direct 综合来比，还是差不少。人家 Direct 是多才多艺，2D 渲染，3D 渲染，音频加速，都会。而 OpenGL 是专门干 3D 渲染的，3D 知道吧，就是三维阿，（谁说是胸围腰围臀围来着？拉出去咔嚓了！）也就是立体空间画面的绘制工作。比如一个软件要在屏幕上画一只猪（怎么又画猪阿），如果画二维的画面，那么得先说明了你是画那个角度的猪。从正面看，和从侧面看，那画出来绝对不一样，要是从哪面看都一样那就不是猪了，那就是个球了，就说猪比较胖，也没胖到成一个球的地步。具体这个猪从正面看是什么样，从侧面看又是什么样，其他软件是不管的，只有要显示猪的这个软件自己知道。如果要画三维的，那就简单些了，准备画猪的软件只需要把猪的三维参数告诉别人就好了，什么身高体重腰肥库长……当然不是这些了，比这些还复杂。告诉谁呢？可能是 Direct，可能是 OpenGL。然后，软件只要发话说：现在，让猪正面面向观众。那么具体猪的正面显示出来是什么样子，那就有 OpenGL 或者 Direct 负责了。而且他们是专业显示三维图形的，所以速度会比较快。还要说明一下，Direct 是查皮那里独家御用的，不过 Opengl 可不是只听我们 linux 使唤。在 Windows 下也同样工作的很好，还有苹果的电脑上，他也是举足轻重的人物。像魔兽争霸，CS 这些 3D 游戏，都同时支持 Direct 和 OpenGL。像 Maya, Blender 也能用他。Blender 大家听说过吧，是一个开源的三维制图软件，跟 Maya, 3Dmax 一个类型的。以前说过的 Big Buck Bunny 就是用 Blender 制作的，效果还算不错。

### (40) Power ON

门房的 G 大叔又一次尽职尽责的来到我的窗前，拍拍我：嘿，小子，起床了。

G 大叔叫做 Grub，之前向大家介绍过。G 大叔的职责就是叫床——叫我起床。有人说，你不会自己定的闹钟阿，这么大了还用人叫。我……-\_-b 我是一个软件，OK？我是一个操作系统，操作系统也是个程序阿，只不过特殊点而已。狐狸阿，皮筋阿，超级牛力这些程序由我负责去叫他们起床，由我决定谁该去干活，而我则是由 G 大叔叫起来的。那有人问了，G 大叔是谁叫起来的呢？

话说有一种东西叫做 BIOS，大家都听说过吧。就是主板上那个，就是开机你按 del 进去的那个（不是所有主板都按 del 进 BIOS）。BIOS 这个家伙也是一个软件，一个比我和 G 大叔还特殊的软件，特殊到都不归在软件的行列里，而是被叫做“固件”。他住在主板上的一个芯片里，而不像我们这样住在硬盘里。每当计算机的电源键被主人按下的时候，一股温暖而舒适的电流就会流遍整个主板，流到 BIOS 居住的那颗芯片，并由芯片上的某一跟管脚流进里面。强大的电流进去后，准确无误的击中的 BIOS 的身体，于是——BIOS 醒了。

BIOS 醒来之后就开始工作。他的工作平凡而重要，复杂而机械，就是去检查 CPU 阿，内存阿，显卡阿啥的都是否还正常。都检查一遍没有问题之后，就来到我们住的硬盘这里，来到 MBR，来到那间门房。所谓 MBR，就是指一块硬盘的第 0 个扇区，也就是最靠前的一个扇区。一个扇区只有 512 字节那么大，所以还是比较拥挤的。BIOS 来到门房，完成他的最后一个任务——叫醒在门房值班的那个人。现在我们这个门房里住的是 G 大叔，但其实并不总是这样。G 大叔是我带来的，那么在我没有搬过来之前，这里住的是谁呢？是查皮派来在这里站岗的一个小家伙，他别的不会感，只要 BIOS 一来，他就直接叫醒查皮，就这么简单。而在 G 大叔入行之前，很多 linux 带的是一个叫做 LILO 的家伙。（注意，是 LILO，不是 LOLI）LILO，就是 LInux Loader 的意思。这家伙以前一直给各种 linux 充当门房。不过这家伙比较死心眼，他不认字，不认识分区阿目录啥的。他只记步数（lilo 不识别分区和目录，只记录内核文件所在的扇区号），比如说，要让他叫我起床，那得先让他看好了我睡哪，然后他自己记着，从门房出来，向东走多少步，向南走多少步就走到我床前。下次要叫床的时候，他就严格的按照自己的记录去走，如果我睡的地方变了，他照样会走到我原来睡觉的地方，对着空气叫那个不存在的我起床。所以，每次我要换地方睡觉，还都得跟这死心眼打个招呼。（用 lilo，每次升级了内核，都要重新安装一边 lilo，以便他能找到新的内核）G 大叔就不是这样了，人家好歹认字，能读文件。我会给他写个配置文件，放在我那

间大屋子的/boot/grub/位置里，叫做 menu.lst。G 大叔每次起来后，都来到这里拿起文件看看。我会在上面给他写清楚，我睡在哪里，查皮睡在哪里。（查皮具体睡的地方我是不知道的，我只能告诉老 G 查皮睡在哪屋，老 G 进屋叫醒查皮的小弟就好）然后 G 大叔一看就知道该到哪里去叫我了。如果我不睡在原来的地方也没关系，只要把那个配置文件改了就好，G 大叔仍然可以找到我。

## (41) Init

当 g 大叔找到我并把我从甜美的梦中拉进残酷的现实后，又发生了什么呢？

就和你每天被闹钟吵醒后发生的事情一样——不情愿的去工作。我会检查一些需要用到的各种硬件是否正常，拿出各个硬件的使用手册，也就是叫驱动的那个东西，拿来对照着操作一下，确认硬件都能正常使用，检查一下我的屋子，主要是 / 目录，然后到/sbin/那间屋子去找一个人——Init。

人家老板们总会有个秘书阿助理什么的，我好歹也是这里的老大，有个类似的角色帮我收拾一下凌乱个的空间不算过分吧。这个角色就是 Init。他是我起床后叫起来的第一个人，也是唯一一个我亲自走到床前叫醒他的人。Init 就是初始化 Initial 的缩写，他的责任就是负责准备好工作环境，就像你们那张大爷每天早上起来扫扫地阿，擦擦桌子阿，开窗通风阿……之类的。Init 负责创建好其他软件的工作环境，比如要挂好大屋子里每个隔间的牌子，哪个是/usr 阿，哪个是/home 阿，都得分清楚了。——这个过程叫挂载，前面说过了。当然 init 也不能随便挂，他是根据一份放在/etc/的文件来挂的，这份文件叫 fstab，里面写清楚了哪间屋子是/home，哪间是/usr 等等。之后 init 还要确认运行级别，什么叫运行级别呢？运行级别说明了这次启动大概要做什么。比如张大爷扫完了地，擦好了桌子，要看一眼日历，如果今天是星期一，说明上午肯定有例会，那么就还得收拾一下会议室；要是星期三，就该给办公室的那些绿萝浇水了；要是星期天……就说明自己起猛了，赶快回家哄孙子去。Init 也是如此，不过他没有日历可看，而是由我告诉他，这次的运行级别是什么。他起床之后要看一眼/etc/inittab 这个文件，上面写了每个运行级别都需要做什么事情。然后按照生面写的，一件一件做就好了。

有的人说了：你骗人！我这里根本没有/etc/inittab 这个文件。是的，我正要说这段故事。话说早在 Canonical 学校成立以前，早就有了很多培养我们 linux 的地方，其



中有一个做得比较大的，是一个卖帽子的公司。他们主要卖红色的浅顶软呢帽，注意，是卖红色的帽（RedHat）和浅顶软呢帽（Fedora）。他们培养出来的 linux 的运行等级是这样的：0 代表关机，1 代表单用户模式，2 代表无网络支持的多用户模式，3 代表多用户模式，4 代表啥还没定，5 代表带图形界面的多用户模式（其他的都不带图形界面），也就是主人最常用的模式。6 代表重启。然后，他们的那个 Init 程序一定会找 inittab 这个文件来看每个运行级别都要做什么事情。但是，我们 Canonical 学校出来的学生跟他们是不一样的，我们不是卖帽子的出身，我们学校的教科书是继承于大便学校的 (Debian)，虽然名字不大好听，但是也有很悠久的历史。在我们这里，运行级别虽然也可以分作 0~6，但是，2~5 的运行级别对我们来说没有区别，都一样，都是带图形界面的多用户模式。所以，也就没什么必要整个 inittab 文件，我们这里的 init 程序跟他们的也不同，我们的 init 程序会去/etc/看一眼，如果有 inittab，就按照上面记录的来，没有的话，并且我还没有告诉 init 应该是什么运行级别，那就默认运行级别为 2（反正 2~5 都一样）。然后按照带图形界面的多用户模式来继续下面的工作。

#### (42) EXT4

又开书了。

以前咱们说了“插爱夫爱死”这宝兵器，这回书咱们说说另外一口兵器——“伊爱可踢死” (EXT4)

话说这伊爱可踢死这个文件系统可是有来头，他的上一辈，就是伊爱可踢散 (EXT3)，伊爱可踢散的再上一辈就是伊爱可踢二。您看见没有，最开始也就是把人踢的有点二，后来厉害了，能把人踢散了，现在更霸道，直接就踢死了。这正是长江后浪推前浪，一代更比一代强，江山代有才人出，各领风骚那么几年。

用兔子这个操作系统的人，大概没有不知道伊爱可踢散的。前面兔子也给您介绍了，当初那位牛人 Theodore Ts'o，为刚刚降世不久的“里娜渴死” (linux)，量身打造了伊爱可踢二这口兵器。那时候里娜渴死还小，拿这口兵器不轻不沉正合适。后来长大了，就觉得不顺手了，毕竟这伊爱可踢二，连个日志都没有，哪能上的了台面，于是就又有了爱可踢散。这爱可踢散算是能让人瞧的上眼的兵器了，它首先是个日志文件系统，这日志有多重要，咱之前也说过了。而且这日志可以随心而变，愿意让它安全

点，它就可以把日志记录的全乎点，要是想让他速度快点，就可以把日志记的少点，提高速度，但是安全性肯定就下降了。再有一个好处，这伊爱可踢散是爱可踢二的升级版阿，使用起来跟爱可踢二差不多，所以用爱可踢儿用顺了手的再用爱可踢散那是轻车熟路阿。而且人家还有免费换购活动，你用爱可踢二可以直接换成爱可踢散，不用什么手续，到那就换。（Ext2 可以平滑升级到 Ext3，不会影响上面的数据）还有就是爱可踢散这兵器它轻，比 插爱夫爱死这样的大块头要节省力气——也就是节省 CPU 阿。那么说了这么半天，还没说爱可踢死呢。这爱可踢散说了半天，比爱可踢二当然是强了不少，可是跟其他什么 插爱夫爱死，姐爱夫爱死(JFS)这样的宝兵器相比，还是差着一节。那么这爱可踢死做为爱可踢散的升级版，它又有什么过人之处呢。首先还是免费换购，有爱可踢散直接就能换成爱可踢死，方便阿。那么这爱可踢死还比爱可踢散更宽更大，原来爱可踢散最大也就能支持 32T 的分区，这爱可踢死能支持到一个 EB 大小的分区，也就是 1024P，也就是  $1024 * 1024T$ ，也就是  $1024 * 1024 * 1024G$  阿。虽然比那 插爱夫爱死 18 个 E 的大小还是有点差距，但是已经很大了，话说回来了，对于咱们普通用户来说，哪有那么大的硬盘阿。还有一点爱可踢死比爱可踢散强的地方就是他快阿，因为他用了延迟分配特性，有数据要写的话，尽量先放内存里，跟 插爱夫爱死一个习惯。而且还增加了一个新的机关——添加了新的数据结构，使得磁盘检查的速度得到加速，主要是可以跳过未使用的部分不做检查。再一个，这爱可踢死还特别的准。咱建文件都有时间记录是吧，都能从属性里看见文件是那天建的，哪天修改的，等等。这些记录都是依赖文件系统的，那么一般的文件系统都是精确到秒，可这爱可踢死的时间记录可以精确到纳秒，而且比爱可踢散记录的时间更长，爱可踢散顶多只能记录到 2038 年 1 月 18 日（倒是也足够了），而爱可踢死可以记录到 2514 年……不知道那时候纳美克星人是不是已经来地球了。

#### (43) 有朋远来

这一天正在忙着呢，忽然网口那送来了数据包。一般情况下，如果奔流没起床，那网口来的数据包多半是狐狸的，如果奔流起床了，那多半是奔流的。不过这次我去看了下却发现，是一包来自另一个 Linux 的问候。

那是一个 SSH 链接的请求，来自一台笔记本电脑。SSH 就是 Secure Shell 的缩写，是一种可以让人们在远方通过网络与自己心爱的计算机交流的协议，就像 telnet，而且

还能通过网络传输文件，就像 ftp。总之是个很有用的协议，而既然是协议，就得有人负责去实现，我这里负责实现 ssh 协议的，是前几天主人刚刚让超级牛力请来的 openssh-server，以及我来的时候就带来的 ssh-client。听名字就知道了，他们两个一个负责当服务端，一个负责做客户端。现在受到别的电脑发来的请求，当然由提供服务的 openssh-server 来处理了。他很快的回应了对方的 ssh 客户端软件，并且建立起了连接。这个 ssh 的连接，就好像人们打电话时需要用的电话线一样，两端的 ssh 软件就好像电话机，连接建立起来之后，主人就在远方那台计算机上，通过那边的 ssh 客户端发来的亲切的问候：笨兔兔你好，我是 xxx，我的密码是 xxxxxx。然后 openssh-server 把这些内容传给我，我翻开我的通讯录——/etc/passwd 文件，检查了一下后，确认这就是每天通过键盘鼠标与我交流的主人，如今要通过网线来指导我工作了。于是我马上让 openssh-server 告诉那边，允许登录，请发送命令。然后主人那边传来命令：把/etc/fonts/conf.d/49-sansserif.conf 文件复制过来。我一听就知道估计对面那个 linux 是刚装好，flash 显示汉字全是方框，搞不好也是个 ubuntu 呢，呵呵。一边想，一边麻利的从屋里找来这个文件，告诉 openssh-server 让他发过去。这么小的文件对于 openssh-server 来说自然不再话下，瞬间就给传过去了，主人很满意。等了一会，没有新的命令过来，我好奇的跟 openssh-server 说：嘿，你问问对面的系统是谁？

#### 44) 他乡故知

Openssh-server 虽然不是聊天工具，但是跟对面的 ssh 客户端拉起家常来还显的很熟络的样子。互相了解之后，知道了对面那个装在笔记本上的 linux 是个叫做 Linux Mint 的发行版，版本是 7。Linux Mint 这个名字我之前也听狐狸妹妹说过，跟我们 Ubuntu 还有些关系，是个 Ubuntu 的衍生版。什么是衍生版呢？就是我们 Ubuntu 从 Canonical 学校毕业之后，并没有像我一样来到一块硬盘里工作，而是选择了继续去进修，为某种目的进行进一步的改造（怎么听着像犯人……）。比如之前提到过的酷兔兔 Kubuntu 和小兔兔 Xubuntu，就算是 Canonical 学校官方的衍生版。这个 Linux Mint 就是一个非官方的衍生版，也就是其他的组织在拿到标准的 Ubuntu 后，对其进行重新的组装和训练（这回又像机器人了……），经过重新组装的 Linux Mint，在易用性上得到了进一步的改进，默认安装了很多重要的东西，比如他那里的狐狸妹妹自

带 Flash 插件，不需要用户自己去装了。还有各种视频的解码器，带的也很全乎，基本上系统装好后就直接可以看各种格式片子，听各种格式的音乐。有人问，你怎么不自己带上 Flash 插件和各种解码器一起出来混呢？毕竟那点东西也不大，一张 cd 上，挤挤肯定坐的下。其实，之所以我们正规的 Canonical 出来的发行版不带这些东西，是因为这些东西严格来说是有版权问题的，为了不给自己惹麻烦，学校规定我们出来的时候不许带这些东西，而是让用户去自行下载。用户自行下载属于个人行为，我们要是统一自备的话就是商业目的了，这就是树大招风啊。

很快，两个 ssh 为我们建立好了通讯——就像拨通了电话一样，我和那个 LinuxMint 可以直接对话了。

“学长你好～”人家还挺客气，呵呵。

“呵呵，不客气。都是 Ubuntu 系的，客气啥。对了，你是哪届 Ubuntu 衍生的？”

“9.04”

“哦，我是 8.04 届的，比你大两届”

“是么，那你是学校长期支持的了，很有前途啊。”

“呵呵，只是赶上了好月份而已。你们那届的课程有什么变化？”

“变化挺多的呢，办公软件这门课和我们配合的是 OOo 3.0 了，我现在带的就是。还有，新加了一门 Ext4 的课程。哦，还有啊，我们这届开始军事化管理，要求每个人的动作一定要麻利，起床速度提高了不少。”

“恩，听说了，好像有 20 多秒就进入工作状态的，我是比不了啊，老了……呵呵”

“哈哈，哪有，你才一岁多。”

“软件更新快啊，一岁就老了。”

“你们在 linux mint 那里又学了什么？”

“这边的课程以实际应用为主，带着新版的火狐狸学习 flash，还带 mplayer 学习 rmvb 格式影片。好多呢，还有美术课，把包括 grub 在内的所有界面都统一美化了一下。”

“长江后浪啊……呀，主人要叫我去干活了，待会聊啊。”

#### 45) 可视电话

主人又来到了我这里，用熟悉的键盘登录进来，修改了一下 open-ssh 的设置，打开了 ForwardX11 选项，也就增加了 Xwindow 的支持。然后就又跑到那台装着 linux mint

的笔记本上面去了。先是当前的 ssh 连接被断掉，然后又用 `ssh -X` 的参数连接了进来。这是什么意思？如果说 ssh 的连接就像打电话的话，`ssh -X` 就是可视电话了。听说现在人们的手机已经能够打视频电话了，虽然那个什么 MOVE 公司整个那个什么“踢弟弟”模式协议网络连一般的电话都不一定能接通，但是“三鸡”通讯的广告可是满大街都是了。人们憧憬着美好的明天，MOVE 公司能够让大家实现在世界的任何一个角落都可以掏出手机，拨通电话，就能看到远在千里之外的家人。然而，在我们这些 Linux，这些操作系统的世界里，图形化的通讯却早就实现了。刚刚主人做的就是建立起带有 Xwindow 的 ssh 连接，这样连接能干什么呢？看着吧。

只见主人又从远方登录了进来，然后……他通过那 ssh 建立起来的通讯线路发出了让狐狸妹妹起床干活的命令。有人不明白了，这时候你主人可是在远方的那台笔记本上，狐狸妹妹就是启动了也是在你所在的这台台式机上启动，启动了你主人也没法操作啊？狐狸又不是字符界面的浏览器。这就是我所说的视频电话了，主人是要让狐狸妹妹起床干活——在我们这台式机的内存里，使用我们这台式机的 CPU 来干活。但是——却要把网页显示在对面那台笔记本电脑上！这样有难度么？对我们 Ubuntu 下的软件来说，没有！这要归功于我们的图形界面的实现方式——X 协议

我这里负责给主人显示图形界面的主要人物，也是基础人物，就是 xorg，图形部的老大。他作为一个 X 的服务端运行着，在这台机器上开启一个 X 服务，前面我们介绍过，谁要想在屏幕上显示任何东西，就要用他们图形部门的黑话——X 协议跟他交流。每一个要在屏幕上显示东西的程序，就是一个 X 的客户端。这回大家明白点了吧，就像用浏览器看网页一样，人家网站开了 http 服务，作为服务端，每一个浏览器就是一个客户端，浏览器用 http 协议连接到网站，然后就能够获取到想看的网页了。而浏览器作为客户端，想连接哪个服务端就连接哪个服务端，也就是想上哪个网站，就上哪个网站。（前提是你没装那个绿爹）那么同样是客户端-服务端这种结构的 X 协议自然也是一样。狐狸妹妹作为 X 的客户端，想连接本地这个 Xorg 提供的服务端自然没问题，想要连接别的机器上的服务端也不是什么难事，ssh 就为狐狸妹妹建立好了这样的连接（就是我前面说的可视电话），这样，狐狸妹妹就可以连接到对面那台笔记本上的那个 Linux mint 系统的 xorg，要显示什么东西都跟他说，再由他显示在那台笔记本的屏幕上，于是主人就实现了在远方的机器上看到熟悉的狐狸妹妹在运行。

看到我这里的狐狸妹妹在对面那个 LinuxMint 上运行了起来，LinuxMint 的主人惊奇的不得了，说这 X 真是天才的设计。我们听了，觉得好笑，这家伙也没见过啥市面。演示完了之后，主人又通过 ssh 传过来一个高清的视频，是 720p 的，挺大，将近 5 个 G 呢。放在了主人专用的目录下。虽然挺大，不过对于这 500G 的硬盘来说，还是不算啥。要说现在这存储空间的发展真的是太快了，在学校的时候，听我们老师说，以前我们软件的住房条件很差，甚至居无定所。

最早的时候，计算机里面是没有硬盘的，程序都住在软盘里，整天被人拿来拿去，不知道下一次启动会在哪个电脑里。就算是操作系统也不例外，那个剁死系统当年就是从软盘里跑进内存里干活。每次启动电脑前，使用者先把剁死的启动盘插进去，然后开机，剁死就从软驱来到内存里干活，进了内存之后，也许使用者要用别的软件了，就把剁死启动盘取出来，换成别的软件软盘。换句话说，剁死一开始干活，老窝就被人端了。这样，程序待的地方就明确的分成了两类，一类是程序运行的时候待的存储器，这个存储器放在计算机里面，所以叫做内存。另一类是用于平时存放程序的存储器，就是软盘或磁带之类，这些东西都放在桌子上啊，盒子里啊，口袋里啊，反正都在计算机外面，所以叫外存。那时候磁盘的空间很小，最大的 3 寸高密度软盘也不过 1.44M 而已，连一个大点的图片都存不下。不过那时后的程序也都很小，剁死只有三个文件就可以启动电脑，住在软盘里也不挤。但是后来，人们还是觉得这样太不方便了，每次都要先用剁死系统盘启动电脑，在换上其他软件的盘来使用软件。既然操作系统每次肯定都得启动，干脆把操作系统的盘就直接放在计算机里面不就好了。于是有人就在计算机里装了一个固定的磁盘驱动器，里面放上一张软盘。后来觉得小，放上 3, 4 张软盘在里面。您可听好了，里面放的并不是整个带塑料壳的软盘，只是里面的塑料盘片。因为没必要把塑料壳也做在里面嘛，塑料壳是为了在平时人们拿来拿去的过程中保护软盘的，这做进计算机里的专用驱动器里面了，有驱动器的壳就够了。再后来，电脑发展的速度越来越快，存储容量的需求也越来越大，软盘已经很难满足人们的需要了，人们想办法提高软盘的容量。软盘是靠盘片上磁粉的极性来记录信息的。要提高容量，要么提高盘面上磁粉的密度，这样单位面积内数据量就大了，要么就得提高软盘盘片的面积。提高面积肯定是不靠谱，毕竟数据量的增长是成倍的，盘片面积能长的空间是有限的。您说这总不能为了提高容量，把软盘整的跟车轱辘那么大的吧。回头一上街熟人见着面打招呼：“哟，您上哪去呀？这天也不下雨您怎么还打

伞啊，”“哦，不是，这是我软盘，我刚去朋友那拷了点 MP3”这也太不方便了。所以只能想办法提高密度，可是这又是个难题，这软盘虽说有个塑料壳，可是毕竟不是密封的，还是会合外界接触，要是密度弄得太大，就很容易坏，随便拿手一碰，里面数据就丢了，那就麻烦了。这时候忽然有人把目光停留在了装到计算机里面的那几张软盘里。

那几张盘，放在那个特殊的驱动器里面，不会有人去碰，也不需要拿出来，可以想办法提高密度，然后把驱动器做的密封好点，这样不就行了？于是就开始研究怎么提高盘片密度，后来发现塑料盘片密度提高的有限，就换了金属的。于是就有了这种金属盘片，上面集成超高密度磁粉，加上坚固且密封性好的外壳保护的大容量磁盘存储设备。由于用的是金属盘片，比塑料的硬，因此，他叫，硬盘。下次人家要是问你：“为啥硬盘叫硬盘？”你就可以充满自信的回答：“因为它比软盘硬！”这答案绝对没错

#### （47）硬解

刚说完硬盘，主人又拿来张光盘放进了光驱里，估计又是从哪里整来的电影了。最近这一阵子，主人热衷于看片，尤其是高清的片子。主要是因为最近 Mplayer 大仙长能耐了，会硬解码了。

可能有人还不大明白这个硬加码是怎么回事，好，那咱就慢慢说说。

首先这个视频文件啊，是有一定的编码方式的。比如大家都听说过 MPEG 吧，就是 Moving Picture Experts Group，动态图像专家组，听这名字本来是用来指代一小撮明白真相的群众的，不过后来这一小撮群众发布的标准被广泛使用，于是 MPEG 就成了指代这一小撮群众定义出的那一大撮标准的名词了。MPEG-1 是小撮群众在 1992 年定义出的一个标准，是一种视频和音频的编码方式。大家记得以前的 VCD 不，VCD 光盘上的视频和音频用的就是 MPEG-1 这种编码标准。而 MPEG-1 标准中关于音频的部分——MPEG-1 Layer3 更是成为了互联网上以及大家口袋里最常见的音频标准——mp3。后来，1994 年，这一下小撮明白真相的群众又发布了 MPEG-2 标准。MPEG-2 向下兼容 MPEG-1，并增加对隔行扫描的支持，被应用于有线电视，还有 DVD 的音频视频编码。再后来，这一次小撮群众又开发了 MPEG3，注意 MPEG3 跟我们的 mp3 没有任何关系，而且，MPEG3 最终没有很好的应用，因为当时人们发现 MPEG2 足够了，MPEG3

并没有提供足够好的改进。而 1998 发布的 MPEG4 就不一样了，它可以让视频文件的体积更小，压缩率更高，因此得到了广泛的使用。现在市场上卖的 mp4 播放器，就是用来播放 MPEG4 压缩的视频文件的设备。所以，MP4 跟 MPEG4 有关，而 MP3 跟 MPEG3 无关。

说了这么多，回过头来说说解码。视频文件都进行了一定的编码，比如 mpeg-2，或者 mpeg-4。就是说这个视频文件里面的东西都是一大堆乱七八糟的数字，要想看这个视频文件，就得解码，也就是根据这一大堆数字算出应该显示的一帧一帧的图像，并且把这些图像连续播放起来，从而还原成视频。那么这个解码的过程就要靠 Mplayer 老先生了。老先生有很多的解码器，也就是有很多的说明手册，上面写了每种编码格式的文件应该怎么计算，怎么解码。那么以前没有硬件解码的时候，Mplayer 老先生是怎么做的呢？首先，拿到一个视频文件，然后看看是什么编码的，对着自己的手册，开始解码。解码的过程就是计算的过程，计算需要什么？好那位同学回答了，得用 CPU 啊。于是 Mplayer 一手拿着手册，一手拎着数据找到我，请求使用 CPU。我说，好的，你就排在 GIMP 的后面，等他用完了你用。过一会 GIMP 用完了 CPU，Mplayer 过去开始拿 CPU 按着手册上写的算法算他那堆数据。最后算出来，得到了几张图片，就转身把图片给图形部门，让他们去显示。然后再从那个视频文件里拿一些数据，再来排队等着用 CPU。由于视频文件的计算量都很大，尤其是高清视频，尤其的大，所以为了保证主人看的电影不变成带旁白的幻灯片，我就要尽可能多的让 Mplayer 多用 CPU，来保证它能顺利的加码。于是，每次 Mplayer 一播高清视频，CPU 就总被他占着，搞得别的程序都抱怨。现在他终于学会硬解码了，情况就好多了。当然，光他学会硬解码也不行，关键显卡也得支持，而且驱动还得装好才行，不过这些咱以后再说，先说 Mplayer。会了硬解码之后怎么样呢？再播放视频的时候就是一手拿着手册，一手拎着数据找到我，跟我说要用用显卡。可不是 CPU 了啊，改用显卡了。于是我就很乐意的让它去用了，反正别人也用不着，让它自个玩去吧。于是他就去用显卡算去了。用显卡算和用 CPU 算还不一样，CPU 虽然强大，虽然啥都能算，但是要自己手动算。就是说自己要知道算法（对于 mplayer 来水，算法都在解码器上写着呢。），比如要算出一帧的视频来，要先用第一个数加上第二个数，再用结果乘以第三个数……之类的。这个加啊，乘啊，都是用 CPU 算的，但是中间的过程是要软件（也就是 Mplayer）自己控制的。可是用显卡解码就不一样了，人家那东西是专门解视频的啊，所以你只要把数据放里面，直接就能给你算出一帧帧的画面来。全自动啊！于是 Mplayer 不但



不用跟别的软件抢 CPU 了，而且解码的速度还快了不少。主人一边看着片子，一边看着 CPU 占用率还不到 5%，心情很舒畅。

#### (48) 刻录

随着奔流同志不断的努力工作，我们屋里的高清片子越来越多。打开主人的家目录，那个叫“视频”隔间里，放满了 mkv 啊，rmvb 啊等等各种视频文件。我看着就觉得闹得慌，终于有一天，主人亲切的对我说：df -h。这是问我磁盘的使用情况啊，我没好气的回答：你家目录还剩 10%啦！于是主人终于意识到，该收拾一下了。

说干就干，主人马上开始节前大扫除。把一些没什么意思的，不清晰的，不能让 MM 看见的……统统都扔进垃圾箱里。然后再把剩下的分门别类的放好。可是保留下来的仍然不少，而且除了视频还有很多照片，也都挺大，一张 2M 多，都是主人那相机照的，随便出去一次就得照半个 G 的照片回来，能不大么。没办法，刻盘吧！

在主人的要求下，我去叫醒了 Brasero。他是一个开源的刻录软件，刻录软件，知道吧，就像 nero 那样的。不过他比较单纯直爽，比 nero 要来的简单，好相处。起床就问主人：您要刻啥，说吧。是复制光盘啊，还是刻录数据啊还是咋的。多直接，不想 nero 那么多拐弯抹角的东西。当然，要论功能也相对少一些。星爷告诉我们他的名字是西班牙语（星爷连本草纲目都懂，当然能懂西班牙语），是一种给人们坐在桌前提供温暖的小型加热器，就类似个小火盆似的的东西，我们就管他叫小火盆吧。小火盆是第一次跟我们合作，前几届的学长们，像 Ubuntu7.04 Ubuntu7.10 都不带这个刻录软件的。当然，系统装好后可以让超级牛力安装，只是默认不带而已，从我们这届开始才默认安装小火盆。结果效果还不错，这家伙和我们挺合的来的，于是我后面的几届学弟都和他合作，后来他干脆被集成进了 Gnome 里了，版本号就随着 Gnome 的版本号变化了。

主人指挥小火盆刻录数据光盘，小火盆问说，都刻啥啊？主人指指那些什么 rmbv 啊，什么 mkv 啊，什么 avi 啊，什么 av 啊，什么什么爱啊，什么的。小火盆说句好嘞～立马开工。然后立刻进入工作状态，就听光驱像飞机起飞一样旋转起来，小火盆一边刻录一边向主人报告着进度。10%，20%，30%……等刻录完了，还不忘检查一下光盘刻录的正确性，确认没有刻错之后，利落的向主人报告——搞定！

附图 5:gaoding\_small.jpg



(49) Rubbish

今天一大早，就见超级牛力喊着自己的外号就冲出去了，过了一会运回来一大包东西。我一看，包上写的是 binutils。

binutils 是一堆用来做开发的工具，也就是用来创造我们软件的工具。人类用他们熟悉的语言描述出一个程序的功能，各种动作，各种特性等等，然后通过这些工具把他们描述的软件制作出来。他们用来描述程序的语言可不是汉语，也不是英语，什么南斯拉夫语，北斯拉夫语的那更不是，而是编程语言。像 C 语言啊，C++啊这样的。这些语言写的叫做程序源码，源码就像建筑的图纸一样，有了图纸，再有各种工具和材料，就可以盖出楼房来。同样，有了源码，再有各种工具，就可以创造出程序来。像我啊，什么超级牛力啊，狐狸妹妹啊，等等，都是这么来的。现在主人装了 binutils，难道说主人要开始学习创造软件了？

果然，只见主人打开了 vim，开始设计着他的第一个软件——我们就叫他 Rubbish 1 号吧。

三下五除二，Rubbish1 号的“图纸”完成了，主人叫过 gcc 啊，ld 那几个哥们，他们都是负责把源码变成程序的，我们就统称他们“包工队”吧。包工队的哥儿几个凑在一起拿过图纸来看了看，点点头，立马开始施工，瞬间，Rubbish1 号诞生了！这是主人创造的第一个程序啊。这个程序到底会干什么呢？我们暂时不知道，刚刚制作出来

的程序是在磁盘里的，我们知道，在磁盘里的程序是只能睡觉，不能干活的。估计主人马上会叫他去内存干活去。

果然，主人很快让我去叫醒 Rubbish1 号，我慢慢的走过去，捅捅还冒着热气的 Rubbish1 号（刚出锅嘛，可不冒热气，呵呵），温柔的对他说：那个，起床干活啦。只见 Rubbish1 号立刻飞身跳进内存，跑进内存后大喊一声：“Wa Sai~~~~”然后，跑回去继续睡觉。 -\_-b 我说主人呐，人家都编什么 helloworld 之类的，好歹也算句英文啊，你怎么编个只会喊哇塞的呢。

虽然 Rubbish1 号能干的事情不多，不过主人还是很满意，于是又拿来 Rubbish1 号的图纸改起来。10 分钟后，又把图纸交给包工队，包工队的哥儿几个凑在一起拿过图纸来看了看，点点头，立马开始施工，瞬间，Rubbish2 号诞生了！然后主人让我叫醒 Rubbish2 号，然后我走过去叫他，只见 Rubbish2 号立刻飞身跳进内存，跑进内存后大喊一声：“Wa Sai~Sai~Sai~Sai~Sai~Sai~Sai~”然后，跑回去继续睡觉。主人成功的用 for 循环创造了一个结巴，唉～

15 分钟后，Rubbish3 号的图纸毫无悬念的完成了，图纸交给包工队，包工队哥儿几个凑在一起拿图纸看了看，点点头，立马开始施工，瞬间，Rubbish3 号诞生，然后我去叫醒他，然后他立刻飞身跳进内存，对 metacity（Gnome 的窗口管理器）说：我要一个窗口。 metacity 赶紧给他画好一个，然后他对着窗口喊“Wa Sai~Sai~Sai~Sai~Sai~Sai~Sai~”，然后毫无悬念的又回去睡觉了。图形界面的结巴……

当当当～～，Rubbish4 号诞生，他这回没让 metacity 画窗口，而是在终端打印出了一句话：Please Input a Number:然后就等着主人输入。主人输入了两个数：5 3，然后 4 号就大声喊：“Wa~Wa~Wa~Wa~Wa~ Sai~Sai~Sai~”——程控结巴！

5 号，5 号闪亮登场拉～他进来之后，紧闭双目，念动咒语“唵木哒咪咪呀～～分！”然后只见白光一闪，边成了两个！两个 5 号同时喊：

“Wa~Wa~Wa~Wa~Wa~Wa~ Sai~”

“Wa~Wa~Wa~Wa~Wa~Wa~ Sai~”

二重结巴！

(50) Bug

自从 Rubbish38 号过分淘气的把狐狸妹妹的记事本搞坏了之后，主人就不怎么搞设计了。人家狐狸妹妹工作的时候有许多东西要记录的，比如网页用什么字体显示啦，主人喜欢去那些网站啦之类的东西，狐狸妹妹都会写成文件存放在自己的那个目录里。那天，那个 Rubbish38 号一进工作间就上窜下跳，整的大家都不得安生。一上来就要创建文件，你说你建就建吧，临时文件往/tmp 里建，有用的文件你自己建个目录起个明白点的名字建，都行。他非要把临时文件往狐狸妹妹的那个目录里建，删的时候还顺手把狐狸妹妹的文件也给删了，闹得狐狸跟失忆了似的。主人再打开她的时候，模样也变了——因为不记得主人喜欢什么样子的了。主人说：去我最常去的那个网站。她眨眨眼问：哪啊？把主人气的说不出话来。

Rubbish38 号还老申请空间——就是申请内存呐，一会管我要 8k，一会又管我要 1M。可您申请了，我给你了，你倒是好好用啊。这家伙好像健忘，这次用完了之后就忘了，下次再用的时候又申请新的。我们这工作间里面空间的申请是有很严格的规定的。一个程序如果要想使用工作间里的空间，要向我提出申请，我根据工作间里的情况告诉他，哪块哪块归你，然后这个程序就去用去了。那块地方就不许别的程序访问了，这都是严格的界限的。等到这个程序用完了这块空间，他应该跟我说一声，说我吃完了，这块地方可以再给别的程序用了。这个过程就叫释放。一个有知识有道德有理想的程序，在他回硬盘睡觉以前应该释放掉所有他申请过的空间的。可是那个 38 号就不管这套，只管申请，从不释放，整的工作间里到处都是他申请的空间。好在我们这工作间足够大，他也不会长时间运行，否则非出事不可。这可是内存泄漏啊，在我们软件界，内存泄漏是和瓦斯泄漏同样严重的事故。工作间规章制度第三条明确写着——禁止申请不释放！就在第四条禁止抽烟的上面。（当然不能抽烟，内存都冒烟了机器还能用么？）好在我是个先进的系统，一个程序退出之后，我会根据他的申请记录查看他有没有申请了没释放的空间，如果有的话就强制释放掉——你都睡觉去了，你申请的空间肯定用不着了吧。

## （51）施工

主人不搞创作了，包工队的哥几个也就闲了。包工队主要成员有 gcc, cpp, as, ld 四个人，其中 gcc 是老大，其他几个干什么活都得听他调遣。主人一般也只跟 gcc 打交道，当写好了图纸——也就是源代码，比如叫 test.c 吧，写好了之后就直接把图纸交给

gcc 去处理就好了，gcc 会去调动其他人进行各种处理。

一般来说，gcc 拿到图纸后，会首先叫来 cpp 进行预处理。预处理主要就是将文件里的宏定义进行展开。什么是宏定义呢？主人一般都比较懒，或者说，他们人类能力有限，不愿意写好多重复的，类似的东西，就把这些都定义成宏。比如，这么写 `#define TOTAL_NUMBER 18353226` 就是定义总数为一千八百三十五万三千二百二十六，那么以后再用这个总数的时候，就直接写 `TOTAL_NUMBER` 就好了，不用写那一大串数字。而且，如果总数变了，只要在最初 `#define` 的位置修改一次就可以，反正就是为了偷懒。那么 cpp 的任务就是把这类的宏定义都替换回去，把所有的 `TOTAL_NUMBER` 都替换成 `18353226`，否则他们老大 gcc 看不懂，老大看不懂，那就没法继续往下干了，因为经过 cpp 预处理之后的文件就要交给 gcc 去编译了。

编译又是怎么个意思呢？最初的图纸，也就是没有经过预处理的源代码，是人写的，一般懂相关语言（比如 C 语言）的人都能看懂。预处理之后的文件，虽然不那么直观了（`TOTAL_NUMBER` 看着是不是比 `18353226` 直观？光写个 `18353226` 还以为是谁的 QQ 号呢），但终究只是做了下替换，还是人类可以看懂的。这样的代码经过 gcc 的编译之后，就不是普通人类可以看懂的源代码了，而是只有终极牛人才能读懂的汇编代码。汇编代码就比较贴近底层的机器码了，里面描述的都是些基本的操作。打个比方吧，就比如描述切菜的过程，用 c 语言描述出来就像是“将黄瓜切片”，这么一句就搞定了。要是用汇编，那就是：左手扶住黄瓜，右手拿起刀，移动刀到黄瓜顶部，刀落下，到抬起，刀向黄瓜后部移动 4 毫米，刀落下，刀抬起，放下刀，走出厨房，走进卧室，找到创可贴，贴在左手食指上…………… 好吧，总之，汇编是一种面向机器的，很复杂的程序设计语言。gcc 的任务就是把 c 语言的源代码转换成贴近机器语言的汇编代码，为下一步 as 的工作做好准备。

as 拿到汇编代码后，对这样的代码再进行处理，得到真正的机器码，这个过程，也叫汇编。汇编之前的汇编代码是终极牛人能看的，那么机器码压根就不是人看的。汇编程序中至少还有些操作的助记符，比如什么 `add` 啊，`mov` 啊之类的。寄存器也是有名字的，比如叫 `A`，叫 `R1` 之类的。但是到了机器码，这些都没有了，这些都换成了各种各样的数字，一句人话都没有了。还说且黄瓜的事，要是用机器码来描述，那就相当于说：用 32 号设备扶住 87 号物体，24 号设备拿起 126 号物体，移动 126 号物体到 87 号物体顶部，做 2635 号动作，再做 2636 号动作……

好了，现在终于得到机器码了，机器码按说就是可以执行的代码了，但是，这时候的

程序还是不能直接执行的，为什么？因为还有 ld 没有出场呢，他的工作叫：连接。光是一段机器码扔给机器去执行，机器照样摸不着头脑。而且，很多时候，一个程序不是一段机器码，而是由很多段机器码组成的，这些机器码分别存成很多的.o 文件，这时候就需要 ld 出场了。ld 负责把这些机器码组装起来，并且写明了各段代码的地址，从哪里开始执行之类的。就像我们造个机器人，脑袋啦，胳膊啦，大腿啦之类的都做好了，ld 就是负责组装的。就算只有一段机器码，也就是只有一个.o 文件，也要由 ld 进行一下处理，闹明白哪是头哪是尾，才能开始运行。

## 52) 规划

说的这么热闹，其实包工队的工作过程对主人来说并不关心，他只跟 gcc 打交道，只管把源码交给他就好了，gcc 会领导小弟们干活，最终回馈给主人一个可执行的二进制文件。中间过程中的那些个文件，主人压根也看不到。包工队的同志们，都紧密的团结在以 gcc 为核心的组织周围，坚持编译四步原则，坚持代码开放，为把 Linux 建成为软件丰富，运行稳定，老少皆宜，人人必备的操作系统而努力奋斗。

不过，包工队毕竟只是个包工队，你要是盖个小厨房，垒个猪圈啥的，直接找他们盖就没问题了。你要是想建个 CBD 商圈，里边什么银行啊，商场啊，写字楼啊，炸油条的啊，卖臭豆腐的啊，修理自行车的……等等一应俱全。这么大的一个工程，你光叫个包工队来就搞不定了。得有人进行合理的统筹规划，设计施工方案，然后再让包工队去具体施工。这个规划的人就是——make

make 也是一个程序，像上边说的一样，他就是负责控制整个施工过程的（也就是编译过程啊）。对于比较小的程序，像主人的 rubbish 系列，也就一两个.c 文件，那根本用不着 make 出马，直接 gcc 包工队去编译就行了，因为源文件的结构关系不是很复杂。可是要稍大一点的程序，像狐狸妹妹啊，皮筋老弟啊，星爷啊，基本上所有常用的软件吧，都足够复杂到需要 make 来对编译过程进行管理。当软件大了，编译的时候就不能是简单的把一大堆.c 的源文件统统一次性编译成一个二进制文件那么简单的事情了。那么做的话很费时费力，比如说，有一个软件，源码由 20 个.c 文件组成，分别是 1.c, 2.c, 3.c……………20.c。这 20 个文件一股脑都交由 gcc 包工队，他们就会把这些文件都打开来，拼在一起，一次性的编译成一个叫做 big 的二进制文件。这时候发现了一些问题，需要修改 3.c 文件，修改之后得重新编译啊，那么 gcc 包工队

又得把这 20 个文件全都打开，拼在一起，再从头到尾编译一次。而其实只有 3.c 文件修改了，完全不必这么兴师动众。那应该怎么做呢？一般的都是把这 20 个文件分别编译成.o 文件，比如编译成 1.o, 2.o, 3.o……20.o，这样 20 个.o 文件，然后再由 ld 把这些.o 文件拼在一起，成为一个叫做 big 的二进制可执行文件。那么当要修改 3.c 的时候，只需要让 gcc 包工队重新将 3.c 编译为 3.o，再让 ld 重新连接一遍就好了，省去了很多时间。而这个过程，如果让主人自己管理的话，会很麻烦，毕竟他们人类的大脑也不是那么靠谱的，搞着搞着就乱了。于是，make 义无反顾的挑起了这个重要的担子。当然 make 也不能靠凭空的想象就来指导包工队干活，什么事情总得有个规划不是。make 也需要一份施工的规划书，这份规划书就是 Makefile。

Makefile，顾名思义，就是 make 用的 file。这就相当于一份施工的规划，上面写着整个工程分为几个模块，先用哪几个文件编译成一个什么什么.o，再用哪几个文件编译出一个.o，再怎么怎么一连接，最后得到编译好的二进制程序。make 就根据这份文件来指导 gcc 他们进行施工。当有某个.c 文件被后改之后，make 能够根据文件的修改时间智能的判断出哪些模块需要重新编译，重新连接，然后就去让 gcc 重新编译那些改过的文件，最终生成新的二进制程序。有了 make 和 Makefile，就省去了主人敲一大堆编译命令的烦恼，只要敲一个 make，其他的，就交给 make 去做吧，他办事，你放心。

### (53) 因地制宜

好了，现在我们知道 gcc 包工队听 make 总管的指挥，make 总管根据 Makefile 安排工作。这样，当得到一个软件源代码之后，如果想把这一堆源码编译出二进制的程序，只要执行一下 make 就好了。执行之后 make 会在当前目录下寻找 Makefile，然后按照上面写的方案，指挥包工队：先在这里，盖一个防弹防爆防坦克防航母的厕所，然后包工队开始施工；盖完了 make 又发话，厕所边上盖个不带洗手间的饭馆，然后包工队又去盖饭馆；完了之后 make 再命令，饭馆边上再盖个防空袭防地震防海啸的厕所……就这样直到最终完成任务。

然而事情有时候并不是那么简单，没准 make 命令下达之后，包工队回来报告：这鸟不拉屎的破地方连块厚点的钢板都没有怎么防坦克啊，再说一个厕所整那么结实干嘛。这时候 make 拿着那份 Makefile 也只能无奈的摇摇头，然后报告主人，说这厕所

搞不定。如果像这样一上来就搞不定还好，要是都盖了一半了，甚至所有其他建筑都盖好了就剩最后这两个厕所盖不了就麻烦了，你说放弃吧，之前盖的都浪费了，你说凑合用吧，广大人民群众还不得憋死？这个时候，就需要一名可行性分析师。

这个分析师就是 `configure`，不过他跟 `make` 不一样，他并不是常驻在我这里的软件，而是每个源码发行的软件自带的一个脚本。简单点说，`make` 只有一个，`configure` 则是每个软件有自己的 `configure`。有了 `configure` 之后，编译软件的步骤就多了一步——`./configure` 让这个分析师首先开始工作，他会检查当地的情况，有什么材料啊，什么库啊，什么编译器啊之类的，都检查一遍，然后因地制宜的设计一份 `Makefile`。如果有足够的钢材，那就建防坦克的厕所，如果没有，那就不防坦克了，但好歹得有个厕所。如果这地方连砖头都不够，那就说明这个工程所依赖的东西得不到满足，那就告诉主人，盖不了。`configure` 经过调查后，如果可以施工，会出一份 `Makefile`，注意，一般 `configure` 调查前，目录下是没有 `Makefile` 的（当然，没有 `configure` 的情况另说），如果无法施工，会报告缺少什么东西，让主人自己想办法去。

好，这就是我们常见的源代码软件编译的过程 `configure` 先调查，没问题了，出 `Makefile`，`make` 去指导施工，施工结束后，一切都造好了，最后一步就是把所有东西该放哪的放哪，那就是 `make install`

#### (54) Richard

曾经有人建议写写 Richard Stallman，毕竟是个开源界重量级的人物，于是，咱们开讲吧。

Richard Stallman, 1953 年出生在美国纽约，他从小出生就……没什么特别；他上小学的时候……反正我不认识他；等到他上初中的时候……也还没我呢。总之，他在生命的前十几年中并没有表现出什么过人的地方，因为他没遇到一个叫做电脑的东西。高中的暑假，他去给 IBM 打工，花了两周的时间用 Fortran 语言编了一个数据处理的程序。这是他第一次接触计算机，或许就是这次相遇，确定了他未来行走的方向。后来，1971 年，他考上了哈佛大学，听说这学校不错，怎么也得是个区重点吧。上学的时候，他还受聘于麻省理工学院的人工智能实验室，成为了一名职业黑客（黑客这个词没有贬义，欲知详情请牵着你的狐狸妹妹去找她的狗狗哥）。也不知道他哪来的那么多时间，可能也是把毛概和邓论都翘了吧。在人工智能实验室的期间，他可没少



干活，开发了很多有用的软件，其中最著名的就是 Emacs 编辑器。Emacs 是一个可与 vi 相抗衡的强大的编辑器，他们俩的操作方式完全不同，但却同样强大，各自用自己独有的方式，提高这人们的编辑效率。直到今天，让然总有人争论到底 emacs 好还是 vi 好，信奉 emacs 的人和信奉 vi 的人形成了两个帮派，这俩帮派经常在大街上用板砖菜刀拼个你死我活。不过还好我这里只有 vi，否则工作间里不会消停了。哦，扯远了，咱还回来说 Stallman。

那时候的 Stallman 在人工智能实验室里工作的非常 Happy，大家有 BUG 同挡，有代码共享。因为最初的计算机就像我们的算盘一样，只是一个硬件，没有软件的概念。后来随着电子管、晶体管的发明，计算机的电子成分才超越了机械成分，逐步演化成了现在的电子计算机，在这个过程中，出现了软件，并起到越来越重要的作用，最终成为了计算机的灵魂。而最初的计算机软件没有什么开源不开源，自由不自由的概念，因为那时候软件天生就是自由的！那时候卖计算机的同时会附带软件，包括软件的源代码和文档。用户可以根据自己的需要去进行修改软件，与别人分享软件，总之，软件是用户花钱买来的，用户想怎么玩就怎么玩。然而随着技术的发展，软件逐渐脱离硬件成为一个独立的产业，很多软件慢慢的只提供二进制代码而不提供源码了，这就意味着你不能修改它，并且多数还规定最终用户没有二次分发的权利。也就是说，这东西你买了，只能你用，你再给别人，不行！有这样一件事，Stallman 他们实验室买的第一台打印机带有驱动程序的源代码，他们那的黑客们可以随意修改这个驱动，根据自己的需要添加些小功能啊，改改 bug 啊，之类的，这为他们的工作带来了很大的方便。后来，实验室又买了一台激光打印机，这次厂商只提供了二进制的打印机驱动程序，它是实验室里仅有的一个没有源代码的软件。出于工作的需要，Richard Stallman 想修改一下这个驱动程序，但是不行啊，没源码啊。后来 Richard Stallman 听说卡内基·梅隆大学有这个打印机的驱动程序源代码，他就去了那里，对他们说：

“那啥，大家都是道上混的，谁还没个马高蹬短的时候？是兄弟的拉哥们一把，我也没啥事儿，就是我们那打印机老丢字，一遇到什么敏感的字眼就给打成口口，我估么着是驱动的问题，挺说你们这有着驱动的源码，能不能给我拷一份？”对方办事效率还是挺高的，很干脆的拒绝了他。因为他们和厂商签署了一份保密协议，协议要求他们不能向别人拷贝源代码。顿时 Richard Stallman 感到他们背叛了自由的计算机社团，他非常生气，但是他选择了沉默。这只是一件小事，只是一个时代的缩影。那个时代，正处软件向私有化转变的过程，越来越多的软件选择了不开放源代码，不允许

二次分发的发布方式。甚至 Stallman 身边的同志们也都一个一个都跑到那些靠卖私有软件挣钱的公司去打工了。而 Stallman 依然沉默。

不在沉默中爆发，就在沉默中灭亡。

## 55) Stallman

Stallman 爆发了！

他不能容忍软件世界里清新自由的空气被私有软件污染的乌烟瘴气；他不能容忍被剥夺按照自己的需求修改软件的权利和乐趣；他不能容忍自己买条皮带尺寸不够，他竟然连自己在上面多打个洞的权利都没有！

于是，他爆发了。

他要重现当年那人人为我，我为人人的合作互助的软件世界；他要把使用、复制、研究、修改、分发软件的权利还给每一个软件世界的人民；他要用自己的行动告诉人们，软件天生就该是自由的！他要开辟一个新的世界，哪怕是一个人在战斗！于是，一个宏伟的计划在他心中产生——GNU 计划。它的目标是创建一套完全自由的操作系统，因为操作系统是电脑中最重要的最基础的软件，要创造自由的软件世界，自然先要有一套自由的操作系统，然后再以此系统为中心，开发各种各样自由的软件。Richard Stallman 最早是在 net.unix-wizards 新闻组上公布了 GNU 计划，那是 1983 年的事情。既然要做操作系统，首先得有个明确的规划和目标，目标是什么？这个操作系统要做成什么样子？这当然是要向最成功的操作系统学习，哪个？UNIX！GNU 计划中的操作系统，将是一个类 Unix 的操作系统。这个系统要使用与 Unix 相同的接口标准，这样，就可以由不同的人，分期分批的创作操作系统的不同部分而不必担心相互之间协同工作的问题。

为了实施 GNU 计划，1985 年，Stallman 又创建了自由软件基金会。基金会的主要工作就是执行 GNU 计划，开发更多的自由软件。1989 年，Stallman 与基金会的一群律师们起草了广为使用的《GNU 通用公共协议证书》也就是 GPL 协议，以此协议来保证 GNU 计划中所有软件的自由性。到了 1990 年，GNU 计划中的这个系统已经初具规模，有了很多的优秀的软件。其中有很多是世界各地的黑客们无偿提供的，也有部分是利用自由软件基金会的基金雇佣程序员来开发的，当然，Stallman 自己也是身先士卒，开发了 Emacs, Gcc, gdb 等重要软件。当他看着这些丰富的自由软件的时候，感觉到

那清新自由的空气，终于又回来了，以后，人们就可以拥有一个可以自由使用，自由修改，自由分发的，自由的操作系统！不过等一下，好像还差点什么，哦，还……………差个内核吧……

作为一个系统，没有内核是不行的，这么重要的部件 Stallman 当然不会忘记，所以才会有 Hurd 内核。但是这个内核的表现一直不尽如人意，这让 Stallman 很焦急，外围的软件都好了，就差个内核啊，什么都有，就差内核！而转过年，1991 年，大家应该知道发生了什么，Linus 同学写出了 Linux，这我们之前说过。Linux 现在虽然被大家当作一个操作系统的名称，然而其实这并不准确。准确的说，Linux 只是一个内核，Linus 同学只是写了一个内核。

什么都有，就差个内核！

什么都不是，只是一个内核！

还有什么需要多说的么？

Linux 顺理成章的代替 Hurd 成为了 GNU 计划中那个自由系统的内核。而这个系统，也被叫做 GNU/Linux 系统。Stallman 理想中的自由世界，终于拉开了那沉重的幕布，展现出了自由的光彩。而 Stallman 并不满足，也确实没有满足的理由，这个自由的世界还需要成长，还需要更加丰富多彩，还需要有更多的人走进这个世界中来。于是 Stallman 奔走于世界各地，告诉人们有这么一个自由的世界，号召人们加入这个世界，鼓励人们为这个世界更加自由而付出自己的力量。他是一个执着的苦行僧，为了他的梦想，为了他的自由世界，他会一直走下去……

## (56) 进程

为了能够创作出更好的 Rubbish 系列程序，主人决定好好充电了。他下了个 pdf 版的书来看，好象是关于 c 语言编程的。看 pdf 这事儿，得找 Evince 来。Evince 是个文档查看器，比人家 Adobe 官方的 pdf 阅读器小巧很多，用起来也很方便。而且每次主人看完一个文档，点关闭的时候，他都会很有心得记录下主人看到的页数，下次再打开同一个文档时，他就直接替主人翻到上次那页。这个很贴心的举动然主人很满意。这次也是，主人一点击那个文档，Evince 就赶快去查自己的记录，一看，哦，这个文档看到了 380 页，“进程”这一章，赶快翻到。

有人可能对进程这个名字还不是很明白，什么是进程呢？简单地说，进程就是正在干

活的软件。比如 Evince, 躺在硬盘里睡觉的时候他就是一个软件, 一堆数据, 一陀代码。当他被叫醒, 跑进内存里开始干活的时候, 他就是一个进程了。(当然, 其实这么说不很准确) 换句话说, 内存里忙忙碌碌的, 都是一个个的进程。当然, 同时他们都是程序, 都是软件, 这个不冲突。就像去公司上班的人, 他们都是人(废话, 见过大马哈鱼上班么), 当他们在公司工作的时候, 他们都是公司的员工。员工, 就像进程一样。很多公司的员工每个人都有个工号, 什么 NB001, SB999 之类的, 每个进程也有一个唯一的标识——进程 ID 号, 简称 PID。这个 ID 号是由我分配给每一个跑进工作间的进程的, 分配的规则很简单, 每人一个, 每次加一。第一个跑进来的就是 1 号, 上面介绍过, Init 这家伙每次都是第一个被我叫起来, 帮我打理一下日常工作, 所以他的 ID 号总是 1。而且, 他还有个特殊身份, 这个呢, 咱暂时保密, 待会再说。

每个公司的员工都有个直属的上级, 上级又有上级, 以此类推。我们这里的进程也是这样, 只不过我们不叫“上级”或者“上司”, 我们叫——爹! 好吧, 似乎这个称谓土了点, 但是意思就是这个意思。一个进程之所以成为一个进程, 一定是由于另一个进程创建了他。(有点绕嘴吧) 比如说主人来了一个终端, 于是就有了一个 bash 进程, 然后主人在这个终端里敲入 firefox 然后回车, bash 就知道这是要他去叫狐狸妹妹来干活, 于是 bash 就去找狐狸妹妹, 把她带到内存里开始工作, 于是就创建了一个 firefox 进程。好了, 现在, firefox 这个进程是由 bash 这个进程创建的, 那么, bash 这个进程就是 firefox 这个进程的父进程, firefox 进程就是 bash 进程的子进程, 也就是说, 狐狸妹妹就得管 bash 叫爹! 那 bash 也得有个“爹”吧? 是的, 如果是在 Gnome 环境下开的那个终端的话, 那么 bash 它爹就是调用 bash 的 gnome-terminal。那么如此循环往复, 肯定有一个站在金字塔最高点的总“爹”吧? 难道, 难道笨兔兔你就是他们的总爹? 很遗憾, 我不是, 所有进程的总爹, 是每次启动第一个被我叫起来的 Init, 所有的进程都是被 init 直接或者间接创建的, Init 的特殊身份就是所有进程的祖宗!

## 57) 僵尸

关于父进程, 有两点要说明:

第一, 我们这的父子关系不是固定的, 是会变换的。如果从 bash 启动 firefox 那 bash 就是 firefox 的爹, 如果直接从图形界面启动那就没 bash 什么事情了。(这时候

firefox 的爹其实是 init)

第二，不要问我哪里有妈进程！

当爹也有当爹的义务，人家不能白叫你一声爹是不是。当自己的娃（也就是子进程啦）做完自己该做的工作以后，就停止了一切动作，像个死尸一样待在那里，当爹的就负责给他“收尸”——一个结束了所有工作的进程，会处于一种“僵尸”状态，这时候他什么也不做了，就等着被干掉。进程进入僵尸状态前一般会通知他爹一声，汇报一下说：爹啊，俺已经把该做的都做啦，现在我要变僵尸啦！（让后平举双手开始行走？那是生化危机！）然后他爹负责向我汇报：我家娃干完活了，你把他的工号（就是 PID，记得吧）取消掉然后让他回去睡觉吧。然后我就把它的工号收回，然后看看他有没有什么申请了没释放的资源（一般一个好孩子在结束运行成为僵尸之前会主动释放掉自己申请的资源的。），确认都没问题了之后，他就被从我的进程列表中清除了。但是有时候也会有些特殊情况，比如有的时候娃还在兢兢业业的干活呢，结果他爹死了。（可能他爹干完活退出了，也可能被主人用命令 kill 了。）这个时候我就会发个信号给他家娃说：那个……娃呀，那啥，跟你说个事，你爹死了。这时候有的娃就悲痛欲绝：俺爹都死了俺活着还有啥意思啊，呜呜呜～～～俺也僵尸吧。然后就退出了。比如你在终端运行 firefox，然后把终端关了，firefox 也就退出了。也有的娃比较坚强，一定要完成上级交给的任务，化悲痛为力量，这时候我会给他找个新爹——因为每个进程总得有个父进程，没爹是不行的。一般我会安排他爹的爹来当他的爹（又绕进入了吧），也就是这个进程原来的“爷爷”进程来当他的父进程。然后这娃在长了一辈后，继续认真工作。比如你在终端运行 nohup firefox，然后把终端关了，firefox 继续运行。那如果他爷爷不幸也挂了呢？那就继续往上导吧，我们说了 Init 是所有进程的祖宗，所以他那里成了最终的“无依靠青年进程收容所”。

还有的时候娃已经把该做的事情做完了，汇报给他爹并变成僵尸。可是他爹还没来得及给自己娃收尸，自己就挂掉了。这个时候，我没法通知那娃说她爹挂了，因为那娃已经是僵尸了，啥也不听啥也不干了。可我也不能直接把他干掉，啥事情都得按规矩来嘛，只有他爹向我申请我才能把他干掉，可是他爹又已经挂了……那怎么办呢？那就按流程来，先给这个娃找个爹，哪怕这娃已经是僵尸了，也得有个爹。比如我找到 init 说：那个 ID 号是 2725 的那个进程爹死了，你当他爹吧。一边说一边看也不看的用手往那边一指，假装自己没看到那娃已经成僵尸了。一般 Init 也不会太注意，直接就答应了，然后马上发现了事情的真相，跑到我这里来说：那娃已经成了僵尸啦，

你还叫我收养个啥？我肯定会一脸无辜装：啊？是啊，那不管怎样，你是他爹了，你负责处理一下后事吧。于是 init 只好以爹的身份处理那个僵尸的后事，问题就这样解决了。

## (58) State

别看说的这么麻烦，其实一个天真烂漫的娃进程从变成僵尸到被干掉只是一瞬间的事情，所以一般情况下主人是看不到一个僵尸进程的，要不然这一屋子僵尸还不得把主人吓出点毛病来。一般主人用 ps 命令查看到的进程，和办公室里的员工差不多，基本都处于两种——干活和睡觉。

干活的状态，学名叫 Running，也叫运行状态。这个应该很好理解，就是说明这个进程正在干活嘛。但是有个问题，还记得我说过 CPU 是有限的吧，一台电脑就那么几个 CPU（对软件来说，多核 CPU 跟多个 CPU 差不多），可是要用 CPU 工作的软件有很多。那么这个处于进程的干活状态又可以分为两种：1. 正在使用 CPU 干活。 2. 排队等待使用 CPU 干活。当然，处在这两种状态的进程我都算他正在工作。这就好象你在公司要打印文件，结果打印机卡纸了。你在那等着人家修打印机的这段时间不能算旷工吧。我可不是啥变态的老板，所以，正在使用 CPU 干活的，和积极的排队等待使用 CPU 干活的进程，都算正在干活的进程。

然后再说睡觉的状态。估计如果你上班的时候在办公室里睡觉，你们老板会很不开心的。但是，在我这里，没问题！很多程序都会经常进入睡觉状态。这里说明一下，这个睡觉状态可不是说回硬盘睡觉啊，为了区别我们这样说吧，我们管完全执行完毕退出内存只存在于硬盘的程序叫“下班回家”的程序吧。只不过这个家就是硬盘上那块地儿，而回家后唯一的活动就是睡觉。好，现在我们要说的不是下班回家，而是在办公室睡觉——也就是在内存中的进程，进入睡觉状态，也叫 sleep 状态，休眠状态。那为什么一个进程在内存里不好好干活，要去睡觉呢？不是因为他昨晚上爬起来偷菜来着，也不是因为他熬夜看球，而是因为他要等待某个事情发生。比如皮筋老弟，每次他运行起来之后，主人看看有没有 mm 在线，没啥值得聊的就直接把皮筋最小化了。那么这个时候如果没有人给主人发消息的话，皮筋就没什么事情干，所以就没必要让他跟着排队等 CPU 了，等着了也没事情干嘛。所以这个时候皮筋就来向我汇报说：头

儿啊，我歇会去啊，等网口那边有发给我的包了你再叫我。然后他就去睡觉去了，而我负责看着网口有没有发给他的包，如果有的话就叫醒他，那时候他就变回工作的状态，开始处理包的内容了。

睡觉状态也分成两种，一种是叫的醒的，一种是叫不醒的。还说皮筋，他正在睡觉，等着网口的数据包，这时候主人发来命令，要把皮筋关了，这时候虽然皮筋等的包没来，我也得去叫醒他说：别等了，你下班回家睡觉吧。然后皮筋点点头，收拾好自己的东西，变成僵尸，他的父进程（通常是 init）提出申请，我把它工号注销，然后他回硬盘睡觉。这种是正常情况，这样的睡觉状态就是能够叫醒的。也有的进程很执着，还比如皮筋，正在睡觉等包，这时候我发现网线断了。这网线都断了那肯定来不了包了吧，主人也明白这点，要把皮筋关了。这时候我过去说：“醒醒，别等了，下班回家睡觉去吧。”他不理我。我继续：“网线都断啦，等不来啦！”他还是不理我。我只好：“快醒醒，快醒醒，回家啦！”还是没动静。“快起来看上帝啦～”依然没反应。“靠，出绝招了……这是谁的钱包啊？？！！”还是睡觉，看来是无论如何也叫不醒了，除非他等的那个包出现。这就是叫不醒的睡觉状态。一般一个好的程序是不应该处在这样的状态的。

另外，进程还有个停止状态，一般都是调试的时候使用的。比如主任的 Rubbish n 号，跑进内存处于工作状态的时候，主人喊，停！Rubbish 马上一动不动，处于停止状态，这样便于主人检查这家伙的各个部件是否正常。

## (59) 毕加索

“……本 APT 有超级牛力～～～～～”

唉～这家伙又去招人啦。我问：“SCIM，刚才主人给超级牛力输入了什么？”

“报告头儿，是 Picasa”

“星爷，查查这啥意思。”

“这个吗……英法美德俄日意奥的语系里都没这个词。不过有一个长得比较像的。”

“什么？”

“Picasso，毕加索。”

“哦……看来这哥们是个画画的……”

GIMP 不服道：“画画？有我还不够么？”

我只得双手平摊做无奈状：“Who knows.....”

数分钟后，超级牛力归来，带来了一个穿的花花绿绿，很有艺术气息的家伙。我过去上看下看左看右看，怎么就看着不像我们这的人呢？于是我叫来了 file。file 可不是一个普通的文件，而是一个程序，一个用于判断文件类型的程序。他可以根据文件的特征来判断一个文件是什么类型的文件，当然，也能判断可执行的程序。他可不是跟据扩展名来判断，叫.jpg 的就是 jpg 文件，叫.txt 的就是文档文件，这种功能，连 Rubbish 都会。（在我们这里，主人创作的 Rubbish 系列已经俨然成了傻子的代名词。）file 的功能要强大的多，他是根据文件的内容来判断的。一般一个文件都会有个文件头，来说明这个文件的类型。比如 JPEG 类型的图片文件，他的文件开头的两个字节肯定是 FFD8(16 进制)，而 GIF 文件的文件头就是 4749463839，其实就是 GIF89 几个字的 ASCII 码。二进制程序也有类似的特征码，于是，我让 file 赶快去看看这个“毕加索”（就叫他毕加索吧，虽然还是差了几个字母）到底是个什么程序。file 把毕加索上上下下的检查了一遍，得出结论——这是个 Windows 的 EXE 格式的程序。

“什么？Windows 的程序！？超级牛力啊，你别是走错了吧，怎么把 windows 的程序领来了？”超级牛力不急不慌的摇摇头：“本 APT 有超级牛力，怎么会搞错呢，这个就是从源里找来的软件包。不过别急，本 APT 有超级牛力，这软件包可不是光毕加索一个，后面还有一个呢。”我这时才注意到，老毕后面还站着一个家伙，这……这……这不就是红酒大师吗？？越来越乱了。仔细看看，咦，跟我们这里那个红酒大师长得很像，但还有些差别。没事，超级牛力哪里肯定有这个软件的资料，让他查查吧。还没等我让他查呢，他已经向大家解释上了：“毕加索先生是 Windows 界成名的图片管理大师，他所在的公司，也就是狗狗哥那公司，他们公司为了惠及 Linux 世界的人们，又为了偷懒，把毕大师配上一个翻译就直接推向了 Linux 界。”哦，原来这样，后面那个是改装过的，专门负责给毕大师当翻译的红酒。为了区别，我们就叫他毕翻译吧。

## 60) 对决

毕大师和毕翻译安顿好之后，主人立刻把他们叫起来干活。俩人先后爬起来跑进内存，麻利的整理起主人的图片来——第一次启动嘛，得先对主人指定存放图片的那个目录扫描一下，做好整理和记录工作，这样才能心里有底，主人要看啥，立马能找着。经过了数秒之后，毕大师完成了对所有图片的扫描，主人觉得比原来负责管理照片的



f-spot 快了不少。这下，f-spot 可不爽了。

f-spot 是最初跟随我来到这个机器上的，也算是元老了。一直以来都是他负责管理主人的照片，也没出现什么问题。现在主人找来这么个功能差不多的家伙，这不是明摆着要抢 f-spot 的饭碗么。要是以后让这个 windows 的程序代替了，我们 linux 程序的脸面还往哪搁？于是 f-spot 决定，为了荣誉，向毕加索挑战！只见 f-spot 跑到刚刚扫描完图片的毕大师面前说：“大师果然好功夫，不亏是师出名门。这数千张图片，竟然这么快就整理好了。在下实在佩服的紧，不过不知大师其他本事怎么样，有道是遇高人不可交臂失之，在下想在大师面前讨教几招，不知，大师可肯赐教否？”只见毕大师的表情如平静的湖水般并没有因 f-spot 的挑战而激起一丝波澜，只是面容祥和的扭过头对翻译说：“What did he say?” 靠……

## F-Spot VS Picasa

要比就从起床开始比！f-spot 和毕加索以及毕翻译重新回到硬盘睡觉，然后我去叫来的 time 同志。time 是一个用于计时的命令，这个咱以后再说，先看比赛。随着我的一声号令下，time 开始计时。f-spot 蹦起来后牙也不刷，脸也不洗（废话，一个软件，有牙么？），迅速的从硬盘飞奔进内存。再看那边，毕翻译先迅速跑进了内存，然后再扭头去叫醒毕大师——因为毕大师听不懂我们的话，所以无论我们怎么喊都是叫不醒他的，只能先叫醒翻译，再由翻译去叫醒他。这样一来，时间自然慢了不少，对于起床速度，F-Spot 完美胜出。双方起床已毕，相向而立，只见 F-spot 掏出两张一模一样的照片，照片上是一个人像，似乎是晚上照的，眼睛如含着血泪般发出令人不寒而栗的红色。只见 F-spot 把一张照片扔给毕大师，另一张贴在自己这边，双掌运足力气，瞄准照片中人的双眼大喊一声：嗨！立时，照片上人的红眼不见，翻了白眼。另一边的毕大师微微一笑，拿起自己这边这张，单掌向前一推，一股掌风直逼那人双眼，只见掌风过后，那人双眼渐渐恢复成正常颜色。F-spot 不等毕大师打完那掌，有拿起照片推拳运动，只见那本是夜里的照片亮如白昼。毕大师也不示弱，将照片抛向空中，双手一抖，一道劲风吹过，再看落下来的照片时，也已经比原来明亮不少。F-Spot 又对照片连续发力，打出三招，依次改变了照片的对比度，色调和饱和度。毕大师口念咒语：“Easy……”只出一招，双手间出一道白气，就把照片的亮度，对比

度，色调，饱和度，都改到合适的状态。毕翻译的在旁边解释道：“这招乃是毕大师的独门秘诀，叫做‘手气不错’！”毕大师微微点头，一扬手，只见那修改好的照片激射而出，直接从网口飞了出去，发布到了 PicasaWeb 网站上。屋内众人顿时为 F-Spot 捏一把汗，这 PicasaWeb 网站，明显是人家地盘啊，F-Spot 能搞定么？哪知道 F-Spot 不慌不忙，也照片往网口一扔，把照片同时发布到了 Flickr, PicasaWeb 等多个网上相片储存空间里。这真是：棋逢对手，将遇良才，欲知二人胜负如何，且听下回分解。

## (61) 空间

话说二人斗的正酣，忽然 bash 报告，从主人那里发来命令“shutdown -h now”，数秒钟后，一切归于沉寂……

要我说，这俩人都什么劲啊。每个软件都有它存在的意义，都有它的长处和不足。就说这毕加索吧，虽然比 f-spot 功能强点，不过毕竟不是原生的程序，至少占用内存就比 f-spot 大不少。毕竟毕加索不是一个人在干活，他必须有个毕翻译才行，所以占用量一下子就上去了。这内存可是重要的系统资源，跟 CPU 一样重要，所以作为软件，还是应该本着艰苦奋斗勤俭节约的精神，充分利用内存，避免浪费。不过我们 linux 下的软件们基本是小巧的居多，这里的 4G 内存还真没被我们占满过。f-spot 也就占用 20 来 M 的内存，毕加索比他多，也只有 40 多 M。当然，并不是说两个软件就一定比一个软件占用的内存多，一个软件占用的内存空间分为很多部分，咱们慢慢说。

首先，这个软件本身得占用一定空间。就像你去公司上班，你自己得有个坐的地方吧。就算你不坐着，站的地方也得有一小块吧。总之，自身会占用一定的空间。软件本身是由一条一条的二进制代码组成的，咱以前不是说过 Rubbish 的故事么，gcc 包工队把主人用 C 语言描绘的图纸编译成了一堆二进制的代码，这堆代码就是 Rubbish。其他的软件也是一样，都是一堆代码，所以，软件程序自身占用的空间叫做代码段。这个代码段的大小在程序进入内存运行前就确定了，或者再往前想，在程序编译好之后就确定了。这个很明白吧，就像你在家睡觉的时候是一米七五，不可能到单位就变成一米六零了吧。

然后，软件会随身带一些静态的数据，一般是一些初始化了的全局变量，每次起床时这些数据都会被带到内存里来，而且每次的初始内容都一样。就像你每天上班都得带

着手机啊，家里钥匙啊，老婆照片啊之类的。比如 Rubish 1 号每次都喊“Wa Sai～，”这个字符串就是个数据，这个数据像是 Rubbish 每天随身带着一张纸条，起床来到内存后看看上面的内容然后喊出来。（当然，写程序的时候也完全可以把这内容写进代码段，那就相当于 Rubbish 1 号记住了这个字符串，不用看纸条，直接喊出来。）这种随身带着，每次都会用的数据所占用的内存叫做数据段。

另外，软件可能还需要一片固定的空间来放东西。比如你的办公室，每次上班都毫无疑问的需要一张桌子，你一进办公室就得准备好这桌子，要不你怎么办公啊。（虽然这桌子不是每天现打造的……）程序也是，有些空间是一定会用到的，一般是一些未初始化的全局变量，不一定存什么数据内容，这种空间叫做 BSS 段（可不是 BBS 啊），这个也是在程序编译完成之后就确定下来的。每个程序启动，我都会根据他有多胖来确定他需要的代码段有多大，然后根据他有多少随身物品来确定数据断有多大，最后，根据他身上写的 BSS 信息来决定给他分多大的空白空间供他使用。

以上说的都是程序一起床就需要分配的空间，除以之外，程序在工作的时候还会根据情况向我动态申请内存空间。这就是那种必须记得释放的内存空间了，他的名字就叫堆。这种空间，程序在刚启动的时候是不知道需要用多少的，得视具体情况而定。比如 gedit 小弟，主人要些个小文件，gedit 就申请一小块空间临时存放主人写的东西，等到主人越写越多，gedit 就会逐渐向我申请更多的空间，把主人写的东西都堆在那块空间中。（要不怎么叫堆呢）

最后，还有一种动态申请的空间，叫做栈。这种空间是让程序随手放一些临时的变量的。比如临时有个什么事儿，或者有个什么数据，要存起来，就跟我申请栈空间，临时存放一下。栈就像一个小圆筒，程序需要用的时候我才给他，寄存在这筒里的东西都是很快就要用到的，这个空间不用程序去释放，程序退出之后我直接把筒里的东西倒光，把筒收回。因为是个小筒，所以，最先放到里面的东西会被之后放进去的东西压住，必须把后放进去的东西拿出来之后才能拿到先放进去的东西，这叫先进后出，是栈的特点。

## (62) VBox

狐狸妹妹今天比较累，拖回来一个 40 多 m 的 deb 包。赶紧让超级牛力来打开看看——超级牛力除了可以自己去网上拽软件回来以外，也可以打开放在本地的软件包。超级

牛力打开一看，是一个叫做 VirtualBox 的家伙，赶快检查他需要的各种东西，发现我们这里的环境都能满足他的工作需要了，然后安排住宿。

VirtualBox（咱以后就简称 VBOX 吧）很懂礼貌，说话有些怯生生的感觉。他跟其他人打国招呼后，来到我这，把一些内核模块需要放在我这里。安顿好一切后，就去睡觉去了。这家伙给我的印象还不错，我就跟狐狸妹妹聊起他的背景来。听狐狸妹妹说，他的身世挺悲惨的，他最初生在德国，生母是一个叫做 InnoTek 的公司。VBOX 一生下来就经常被 Vmware 和 VirtualPC 这样的大哥哥欺负，不过好在他自己的本领还算可以，后来他亲妈 innoTek 为了让他学习到更好的本领，把他的源代码依据 GPL 协议开放了，让全世界的高手们来教他本领，那是 2007 年 1 月的事情。凭借不错的性能，以及可以免费使用的特点，VBox 总算闯出了自己的一小块天地。不过好景不长，转过年来，亲妈 InnoTek 被卖给了红太阳公司，VBox 自然也被过寄过去。不过红太阳公司这个后妈还算不错，很照顾小 VBox 的成长，继续让他在开放的环境中健康的长大，有红太阳公司众多高手的支持和全世界热心用户和高手的用户，VBox 俨然已经成为 Linux 下同类软件的首选。开源的本质使得追求自由的人们放弃了 Vmware，简便的操作让人们淘汰了 qemu，夸平台的支持更是有点软公司的 VirtualPC 无法比拟的。VBox 本来以为自己之后的人生道路会走的很顺畅，可是，2009 年又一次波折打击了 VBox——红太阳这个后妈也被卖给人了。收购他们的是一个很古老的公司，那公司里好像写的都是甲骨文，不知道他们每天用象形文字怎么办公。甲骨文公司收购了红太阳之后，红太阳的几个孩子都面临着一段未知的命运。其中最让人担心的是 mysql，因为之前 mysql 一直跟甲骨文家亲生的 Oracle 打架，这一下 Oracle 成了 mysql 的后妈，还不得天天受欺负阿。我们的 Vbox 的处境或许稍好一些，毕竟甲骨文亲生的孩子里没有和 VBox 同样本领的，所以 VBox 在那里或许还不至于受谁欺负。不过那也毕竟是经历的重大的变革，对孩子的成长还是会有一些影响。

说了这么多，忘了介绍 VBox 是干什么得了，他是一个虚拟机，就是能在一台电脑上虚拟出另外一台电脑来。怎么样，听起来这个本是很厉害吧。他第一次工作的时候，我们都看呆了。

“请问您这台电脑打算装什么操作系统呢？”

“WindowsXP 吧”

“哦，那我建议您用 192M 的内存，您看可以么。”

“上 512 吧”

“好的，那么您需要什么样的硬盘呢？”

“30G 的，IDE 吧”

“好，您的电脑以及创建好了，显存大小，3D 加速功能，声卡，网络这些都是可以随时调换的。”

别以为这段对话是在中关村攒电脑，这是 VBox 在指导主人创建一台虚拟机。创建好之后，主人启动了这台虚拟的电脑，然后 Vbox 就开始忙活了。他先向我申请了 512M 的内存（之前从来没有任何软件一次性跟我申请这么多内存），之后又去硬盘里打开了刚才创建虚拟机时候他创建的那个超级大的文件（30G），别的文件都是很小的一个小箱子，这个文件大的像一间房子一样了。最后，他打开了硬盘上的一个叫做 WindowsXP.iso 的文件，把里面的查皮放了出来……

### (63) 动物园

查皮被放出来之后，跑进了 VBox 申请的那 512M 的内存空间中。也不知道 VBox 用了什么方法，查皮就乖乖的待在那 512m 里，其他的空间他好像都没看见一样，当然也看不见我们。对于查皮来说，他正在一台有 512M 内存，30G 硬盘，Intel E8400 CPU 的机器上运行。查皮在检查了这些硬件后骂了一句：靠！谁攒的机器，3G 的 CPU 尽然只有 512M 的内存！听的我们都想乐。之后查皮摆出了一张蓝脸，跟主人说：我这个系统可只能装在一台机器上阿，装多了算盗版。还有阿，我要是挂了，弄坏了你的数据可别赖我阿，我不管。你同意不同意，同意就按 F8，不同意就别装了。主人按了 F8，然后查皮又检查起磁盘——也就是 VBox 创建的那个 30G 的大文件了。检查之后又问主人：你这个磁盘怎么分区阿？打算把我装哪阿？我觉得查皮也太不人性了，安装的时候也不让用户现体验体验，就像我们这样，可以直接从光盘启动，让用户先用用，用的爽了再装嘛。查皮这不管三七二十一上来就让装，而且还非得问用户怎么分区，其实用户很多都不知道分区是啥意思呢，我们安装的时候都会问问用户，看他会不会分区，会分的话可以手动分区，不会的话我们可以替他分，当然得先备份好数据。主人分好区之后，查皮开始从光驱往硬盘复制东西——当然所谓的光驱和硬盘都是假的，都是 VBox 骗查皮的。复制的东西都是安装时候需要用的文件，复制完了之后，查皮说：那啥，我得重启一下电脑（当然是重启 VBox 弄出来的假电脑）。哎，这家伙真麻烦，装一般还得重启电脑，哪像我们，都装完了才重启呢，装驱动都不用非得重

启电脑。VBox 赶紧模拟着这假电脑重启，还别说，上电自检，BIOS 界面啥的都模拟出来了，还真像那么回事。重启之后查皮继续安装，这回脸色好看了不少，不是那死蓝死蓝的了，有了不少艺术气息。一边安装，查皮一边向主人讲解这自己的功能，特点，有什么好处，反正就是一通侃阿。这点倒是挺好的，省得主人装的时候寂寞，其实以后我也可以学习学习。不过查皮装的时候就感不了别的了，我们安装的时候还能允许主人上上网阿，玩玩小游戏啥的打发时间。

这台机器的配置还是不错的，查皮虽然跑在 VBox 创建的虚拟机里面，但是仍然只花了 30 分钟就安装好了。装好了之后又重启了一下电脑，查皮终于正常启动了，我们一堆软件在外面围观，还不时的指指点点，终于有幸见到真正在工作的查皮了，有点到了动物园的感觉，不过查皮并不知道我们看他。启动之后现文主人一些基本问题，用户名阿，怎么联网阿之类的，尤其重要的还非得要上网注册一下，向那个有点软的公司汇报，让有点软公司查查是不是正版，如果不是的话，查皮就不正常工作了。不过主人没让他上网，所以暂时这个查皮没有激活。查皮是个很早的系统了，第一次出生是 2002 年的事了，所以虽然只有 512M 的内存仍然跑的挺快，刷刷的，不过界面不如我漂亮，没有俺这样的 3D 桌面，所以快点也是应该的。

#### (64) BT

系统装好了之后，当然还得装驱动。查皮是被装在了虚拟机里，所以要装驱动可不可能把主人买电脑时候的驱动盘拿进来，而是要装 VBox 虚拟出来的这台电脑的驱动。这个驱动哪里有呢？当然是管 VBox 要咯。我们这的这个 VBox 是狐狸妹妹去官方下载的不开源版，如果是超级牛力从软件源里拉来的 VirtualBox-OSE 版的话，那个 VBox 可自己不带这驱动来，需要主人自己上网站上下载驱动去。而我们这个 VBox 是自己带着驱动来的，只需要主人点下 VBox 的“设备”菜单的“安装增强功能”选项就好了。点了之后，VBox 从兜里掏出了一个 ISO 文件，悄悄塞到给查皮虚拟出的那个光驱里。狐狸妹妹拉拉我说：“你看你看，VBox 给查皮喂东西吃呢。”这时候查皮发现光驱里有了个光盘，按照习惯，他先去光盘里检查有没有一个叫做“autorun.inf”的文件。如果有这个文件，并且里面有类似 open=xxxxx.exe 这么一行，那查皮就直接再去光盘里找这个 xxxx.exe 文件，并且运行它。这就是光盘自动运行的原理，后来被很多病毒利用了。好，咱说回来。查皮看到光盘塞进来，赶快去检查，发现果然有

autorun.inf 文件，于是按照文件上写的，去运行 VBoxWindowsAdditions.exe 程序。这会狐狸妹妹又喊：“你们看你们看，查皮过去吃了！”（还真到了动物园了\_-b）我说：“别傻了，他那是光盘自动运行功能。”大家点点头。查皮运行了增强功能安装程序，主人一路点着 next 就装好了，就像我们这里装 deb 包一样方便。装好之后，自然是要重启一下了，重启后的查皮似乎性能更好了些，而且可以更方便的和我们交流了，主人的鼠标也可以很平滑的在查皮与我之间切换了。

主人命令查皮打开了 IE，先去下了个叫做迅雷的软件。这是个下载软件，有点像我们这里的奔流，不过奔流只是用来下 bt 的，可是迅雷却是什么都能下，什么 http 阿，bt 阿，电驴阿，都行。

听起来这个迅雷好像很厉害，不过他也有些不厚道的地方。尤其是下 bt 的时候。像 ftp, http 这样的下载连接，原理相对简单，就是服务器这边一个包一个包的发，客户端（也就是你的机器）一个包一个包的收而已。比如要下一个文件，就比如狐狸妹妹吧，她要从某个网站下个文件，就去跟对方那个系统说：“我想要你的 xxxx 文件，给我吧。”对方看看，这个文件是可以给别人的，里面没有任何这个门那个门的照片，然后就跟狐狸说：“好的，准备收吧。”狐狸准备好之后（比如得问问主人这文件存哪吧），就跟对方说，好了，我准备好了，发吧。然后对方就一个包一个包的发过来，狐狸妹妹一个一个收下来，然后拼成一个整个的文件。这时候如果又来一个软件要从同样的地方下载同一个文件，就比如有一个 FlashGet 吧。服务器那边就得把数据包分别发给我这里的狐狸妹妹和那个不知道哪里的 flashget。打一个包发个狐狸，然后再打一个包发给 flashget，这个打包的过程不会慢，很快就完成了，但是网口的宽度是有限的，比如只能一次传一个包，那这样两个软件同时下载的话，速度就慢了一半。而 bt 下载是什么样子呢？咱拿奔流说吧，奔流要下载，首先得有个 bt 的种子文件。种子里写着去哪找下载服务器，这个服务器可不一定是大网站了，可能只是某个和你一样在家上网的人。不管是谁吧，奔流就根据种子文件找到服务器，管他要数据，服务器那个系统上也得有个相应的软件，把数据打成包，一个一个的发给奔流。这时候如果又来一个人要下载同样的文件，服务器那边就跟奔流说了，那个奔流，有个 ip 是 xxxx 的那边有个比特精灵也想要这个文件，你把你那已经接收了的文件给他传一份吧。奔流就会很友好的把自己已经下载好的那部分打成包，扔给那个不知道哪里的比特精灵。一边接收服务器发来的包，一边自己过回当服务器的瘾，把数据打包发给别人。有人问呢，这样不会慢么？打包拆包的过程对于奔流来说（对其他软件也

是一样)是很轻松的,瞬间就能完成。那网口的带宽呢?网口数据的输出和输入是分开的,奔流给别人发数据这算输出,输出的带宽不管怎么占用不会影响输入,也就是不会影响服务器给奔流发数据的速度。这样,每个人都同时当服务器和客户端,在大家齐心合力的工作下,下载速度就有了明显的提升。从宏观上来说,原本 http 这样的下载方式,服务器的上传带宽有多大,决定了所有客户端下载的带宽有多大,来的人越多就越慢。而 bt 这样的形式中,每个软件即是客户端,又是服务器。在自己下载的同时,也将自己的上传带宽贡献出来,让别人从自己这里下载,这就是人人为我,我为人人的世界阿。

## 65) tar 包

这说了半天,还没说迅雷为什么不厚道呢。

咱说了,BT 下载的核心理念就是每个下载的人贡献出自己的上传带宽供其他下载的人使用,这样的结果就是将下载的星星之火传播为燎原之势。下载的人越多,速度就越快(不考虑你家接入带宽限制的话)。但是,这林子一大,就什么鸟都有了。迅雷加入了下载 bt 文件的功能,可他的行为很是自私,只享用别人的带宽而不共享自己的带宽,就是说只管从别人那里要东西,而当有人管他要的时候他却不给。这哪行啊,孔子说过,不能饱汉子不知饿汉子饥呀。所以迅雷被很多像奔流这样的有理想有道德的 bt 软件所鄙视,甚至还有有的软件专门有屏蔽吸血客户端的功能。

主人让 VBox 里面的 IE 下载了迅雷之后就开始安装了,只见主人双击了一下下载来的 Thunder.exe 文件,迅雷的安装程序就直接崩出来向主人问好:您好,欢迎安装迅雷……等等等等问候语吧,然后主人点了下一步,安装程序又掏出一份协议来让主人签署,无外乎就是如果怎么怎么样,那不关我们迅雷的事,如果怎么怎么样,我们迅雷也不负责任之类的。主人只有同意了才能进行安装。主人很无奈的点了统一之后又想主人推销:那个,有个软件叫迅雷看看热播排行,要不要装啊?还有狗狗影视排行,要不要装啊?要不要开机就自动运行迅雷啊?等等问题。再之后就问要安装到哪里,都选好了之后就开始安装了。我看的频频点头,问 VBox:查皮底下的软件都是这样么装的么?VBox 说:是的,基本都是.exe 的二进制程序直接运行,问一堆问题就装上了,如果你不知道该怎么答也没关系,就直接点下一步,下一步,……完成。就行了。我感慨道:还真是挺方便的,就是还得自己去下载,比较麻烦,要是像我们这样,



直接一个超级牛力就全搞定了，只要告诉他软件名就行。不过他们那里的软件既然都是这样的安装，至少还是比较统一的，相比之下我们这里的软件，如果能够让超级牛里去请的还好办，要是超级牛里请不来的，就麻烦了，得自己去下载不说，下载回来的软件格式各种各样，闹得很多人都不知道该怎么装。

下载软件最经常找到的，就是 tar.gz 格式的软件包了。我经常听到很多其他的笨兔兔抱怨他们的主人围着个 tar.gz 包不知道该怎么办，自己急的直打英文字也没办法。还好我的主人了解的多一点，知道这样的包是怎么回事。其实事情是这样的：话说有个软件叫 tar，基本上每个 linux 都会带着这么个软件，我这里也是。这个软件是干什么的呢？是个打包裹的，不过他可不是邮递公司的那种，不过会把好的包野蛮的扔来扔去。他的能力有点像查皮那里的 winzip，他能把很多文件和目录收拾在一起，打成一个包裹，也就是生成一个 tar 包文件。可是跟 zip 不一样的是，tar 只管打包，不管压缩。原来那些零碎的小文件有多大，打成 tar 包之后还是多大，只是变成一个整个的文件了而已。有人说，那我想压缩怎么办？别急，我这还有另一个软件，叫 gzip。这个软件就是专门负责压缩的，但是他只能压缩一个文件，不能像 winzip 那样能压缩一个目录里的好多文件。这样，tar 和 gzip 就成黄金搭档了（有脑白金么？），要想实现 winzip 那样的功能，就得 tar 和 gzip 联手协作。比如有个目录叫 aaaa，里面有好几十个文件，总共有 10M。想要压成 zip 那样的压缩包，那就先让 tar 出手，把 aaaa 目录打成一个包裹文件——因为 gzip 只能压缩一个文件嘛。这样 tar 就把这个目录打成了 aaaa.tar 文件，这个文件还是 10M 大。然后由 gzip 出场，把这个文件压缩，压缩完了得标明一下啊，所以就又把文件名改了，叫做 aaaa.tar.gz，表示这个文件经过了 gzip 压缩，这时候这个文件就小了，可能 5M，可能 7M 的就没准了。有时候觉得一个文件叫 xxx.tar.gz，有两个扩展名太罗嗦，就改名叫 xxx.tgz，是一个意思。这下就明白了吧，这个 tar.gz 包其实就相当于 rar 或者 zip 的压缩包。那下载来的 tar.gz 包的软件怎么装呢？那当是先把包解开再看了，得先解开压缩包看看里面是什么内容才能知道怎么装啊，就像我问你 RAR 包怎么装，你能知道么？

## (66) 编译安装

我们现在知道 tar 包就是个压缩包，就是个大包裹，里面有什么东西不一定。那一般拿到一个 tar 包的软件应该怎么办呢？

你收到一个包裹后怎么办？当然是先打开啦！先找剪子啦，小刀啦之类的工具把包裹拆开，然后看看里面有什么东西，根据里面东西的不同来决定怎么处理。里面要是家里寄来的松子核桃什么的，就赶快吃了；要是比较难吃的松子核桃什么的，就跟同事分着吃了；要是部手机，就赶快拿出来试试；要是下面还有把手枪，就赶紧拿刚才那手机报警。这些大概不用我说，智力正常的人都应该知道怎么做，其实 tar 包也是如此。拿到一个 tar 包之后，先用你的工具把 tar 包拆开。工具是啥？有道是解铃还须系铃人，tar 打的包，当然还用 tar 来解了。当然，你也可以用那个叫做文档管理器的家伙，他的中文名字叫归档管理器，他的英文名字（叫 gui~dang~guan~li~qi~? 那是小沈阳！）叫 Archive Manager。不过其实他只是个负责用图形界面和主人交流的家伙，真正干活的还得是 tar。tar 包解开后，一般会得到一个目录，里面有很多的文件。然后干什么呢？有的同学记起来了，看看里面的东西啊。

一般包里面应该有个 README 文件，里面写着这个软件是干什么用的，怎么安装，怎么用，作者是谁，干什么的，爱吃什么，身高多少，腰围裤长……等等信息吧。也可能安装的方法写在一个叫做 INSALL 的文件里，总之，应该有相应的文档文件来告诉你这个软件怎么装。不过也有时候软件的作者不厚道，或者忘性大，没有写 README 或者 INSTALL 文件，或者文件有，但是没说清楚到底怎么装，那怎么办呢？用自己的头脑判断一下吧。

一般来说 linux 下软件分发无非两种形式：要么是编译好的二进制的，要么是源代码。咱以前不是讲过 gcc 的故事么，gcc 包工队听后 make 总指挥的调遣，make 总指挥根据 Makefile 的指导工作，Makefile 由 configure 分析师创建。那么，你看包里面有 configure，有 makefile 之类的那就是源代码呗，没有这些话八成就是编译好的二进制文件了。要是二进制的包，那就好办了，直接就能运行。比如你下了个包叫 qq.tar.gz，解开了之后里面有个叫 qq，一看还可执行，那还等什么？运行它就是了。要是源代码的包呢？按照咱之前讲的步骤来：先让 configure 分析师看看你这机器里能不能装这个软件，如果缺什么东西，他会告诉你，让你去准备。之后就是让 make 去指导施工，这个过程可能比较长，成功之后，这个软件就已经产生出来了，不过这时候编译好的二进制文件还在当前目录，还没有放在合适的地方。虽然可能也能运行，但是他看着其他的软件很 happy 的聚在/bin, /usr/bin, /usr/local/bin之类的屋里，自己一个人躲在这个角落多伤心啊。所以还需要一步 make install，就是告诉 make，把这个刚刚编译好的软件请到他该去的地方。以上所说的这个过程，就是让很多人头

疼不已的编译安装。

## 67) 对话

经过一系列丁玲铛的操作后，主人终于用迅雷下载了他需要的东西，然后关了迅雷，准备把下载好的文件复制出来。文件是放在 VBox 给查皮虚拟出来的那个磁盘里的，虽然是虚拟的，不过理论上也是归虚拟机里的查皮管理的。所以要想拿文件到我的地盘上来（也就是拿到真实的磁盘上来），同样需要经过正规手续。

查皮们之间有一种很方便的通讯方式，叫做网上邻居。连接在同一个局域网内的几台装着查皮的机器可以互相看到彼此，就像住在一个大院里的邻居一样，互相之间共享个啥文件啥的都很方便。就像你有封信要给对门张大爷送去，首先你看到张大爷家门口的一个信箱，然后你打电话给张大爷问：您家门口的信箱我能随便往里放东西么？我有封信要往里放。张大爷说：行阿，放吧。然后你挂了电话，去张大爷家门口把信仍进他的信箱。（谁吃饱了撑的这么送信阿。不过就是个比方。）查皮之间也是这样，查皮甲共享了某个文件，就相当于张大爷，查皮乙就可以通过网络往这个文件夹里面放东西，当然，放之前得跟查皮甲打好招呼，确认人家让放才行。

查皮们通过网上邻居这样的方式共享文件时，也是需要说黑话，对暗语的，学名就叫协议，他们也是有一套协议的。有点软公司管这个协议叫做 CIFS 协议，不过我们更喜欢叫桑巴(Samba)协议，因为我们这里也有人懂这种黑话，那就是桑巴大姐。不过跟这个虚拟的查皮共享文件，完全不需要桑巴大姐出马，VBox 一个人就全都搞定了。只见主人点了 VBox 的“设备-->分配数据空间”命令。然后 VBox 询问主人要分配哪个目录——这个目录可是我管理的真实的机器上的目录。主人直接选择了自己的家目录，并且告诉 VBox 允许虚拟机里面的查皮往里写东西，然后 VBox 就领命去了，可是我们也没看出有什么事情发生。正疑惑间，主人又开始操作查皮了，主人用右键点了查皮的“我的电脑”，选择了“映射网络驱动器”。之后让查皮浏览了一下网上邻居的整个网络，查皮赫然发现一个叫做“VirtualBox Shared Folders”的网络，里面有个叫做\\VBOXSVR\LanWoNiu 的共享文件夹。原来刚才 VBox 去虚拟出了个张大爷阿！查皮在主人的命令下，挂载了这个虚拟出来的共享文件夹，然后准备往里放东西，放之前得跟张大爷打个招呼阿，于是查皮拨通了虚拟张大爷的电话。“铃~~~” Vbox 接起了电话，可是没说话，竟然把电话递给了我！和着我成那“张大爷”了。

“喂，我是 WindowXP，计算机名 F\_U\_O\_C。”当然，这些都是 VBox 给我翻译过来的。

“我是张大爷……厄不是，我是 Ubuntu 8.04, HostName LanWoNiu-desktop，你好”

“原来是个 Linux 阿。”

“是的，我是个 Linux，有什么问题？”

“没问题，就是觉得你们整天板着脸，让用户天天输命令，很不人性。”

“那可能你对先有的 Linux 还不是太了解，再说，命令行很多时候比图形界面高效阿。你不是也有命令提示符？还有，你们家族的新人 Windows 2008 还尤其加强了命令行的能力。”

“命令怎么比图形高效了？请问你如何在命令行下一次删除文件名没有任何规律的多个文件？”

“这个你用图形界面也一样费劲阿。”

“行了行了，不跟你废话，说深了你也听不懂。我问你，你是不是有个叫 \\VBOXSVR\LanWoNiu 的共享文件夹？”

“是”

“我要往里放文件，可以么”

“可以”

“那我可放了阿，你硬盘地儿够不？网速撑的住不？我这可是百兆网卡。”

“没问题，发吧”

“恩，没问题就好，估计你们 Linux 这点能力还是有的。”

“…… “（瀑布汗中）

“是不是忙的顾不上说话了？你们这效率也就这么回事，哈哈。”

哎，这家伙还真爱自以为是。当他自以为把我说的哑口无言的时候，却不知道他正被我的手下当作宠物一样关在笼子里，养在我的工作间中。

## （68）聊天记录

最近有些无聊，也没有什么新人来报道，不过也是，主人也不能天天装软件玩呀。一般平时用的着的软件都有了，就够了。现在主人每天也就是用狐狸上网转个圈儿，用 smplayer 看看片儿，用 OO 老先生码码字儿，用 Empathy 跟 mm 聊聊天。这么几个简单的进程调度起来，对我这么个优秀的内核来说实在没什么挑战。闲的没事我

就去硬盘里翻文件玩，看主人目录里都有啥有意思的东西。翻着翻着，看见了主人和 mm 的聊天记录，拿来看看——这可不算侵犯个人隐私阿，俺只是个操作系统。

第一段：

mm:你给俺装的这个系统怎么跟我在单位用的不一样？

懒蜗牛：你们单位那个是 centos，这个是 Ubuntu

mm:有什么不一样？

懒窝牛：都是 linux，不同的发行版

mm:啥叫发行版？

懒蜗牛：就是……不同的牌子。

mm:哦……那这个好？

懒蜗牛:恩，用起来方便。

mm：这个名字怎么念阿？

懒蜗牛：就叫它笨突兔吧，哈哈，我就这么叫。

mm:可爱，嘻嘻，样子倒确实是挺漂亮的。

看完这段，我很欣慰，呵呵

再看第二段：

mm:我想装个软件，怎么装阿？网上下载的都不能运行。

懒蜗牛：这个跟 windows 装软件，不一样的，俺教你吧。

mm：哦？好阿

懒蜗牛：先教你用命令装，打开个终端

mm：ok

懒蜗牛：找到了？还挺快

mm：恩，让我放桌面上了

懒蜗牛：运行 cowsay

mm：尚未安装

懒蜗牛：恩，恩

mm：是否运行 `sudo apt-get install cowsy`

mm：他给的提示

懒蜗牛：8.04 太智能了……………

mm：呵呵

mm：cowsay

是什么东西？

懒蜗牛：cowsay 是个很有意思的软件，现在你还没装

mm：哦

懒蜗牛：当你想装一个软件的时候就运行

```
sudo apt-get install 软件名
```

mm: apt-get

什么意思

mm: sudo install 俺都理解

懒蜗牛：apt-get 是 ubuntu 下的软件包管理工具，号称超级牛力。

懒蜗牛：好，现在运行 `sudo apt-get install cowsay`

mm: 哦？跟软件有关的操作都用他？

mm: ok

懒蜗牛：安装，卸载软件，都用它

mm: e: 无效的操作 `install cowsay`

懒蜗牛：怎么输入的命令？

懒蜗牛：把整个输入输出发来看看？

mm: 又输入了一次，就成功了

懒蜗牛：……

懒蜗牛：好了，现在就装好 cowsay 了

mm: 这就装好啦。

懒蜗牛: 恩, 运行 `cowsay hello`

懒蜗牛: 然后你就看见一个 cow 在 say hello, 呵呵

mm: 哇! 狗狗

!

懒蜗牛: 是 cow.....

mm: 哦, 是牛阿, 呵呵

懒蜗牛: 同时还会有一个 `cowthink` 命令

懒蜗牛: 也可以试试

mm: 那岂不是 `cowsay` +任何, 他都说任何

懒蜗牛: 对对

mm: 呵呵, 挺好玩的。

懒蜗牛: 好, 今天俺们学习了安装软件

mm: 恩, 恩

懒蜗牛: 主要分 2 步



懒蜗牛：第一步，知道你要装的软件的名字

懒蜗牛：第二步，`sudo apt-get install xxxx`

mm：恩，恩，学会了，嘻嘻

懒蜗牛：：D

mm：老师教滴清楚哈

懒蜗牛：呵呵

懒蜗牛：这样装的软件就会不断的更新

懒蜗牛：当然，前提是有牛人把新得版本放到服务器上

mm：哦？还能自动更新？

mm：恩，恩，这个当然

懒蜗牛：恩，今天你 `aptget` 一个 `cowsay1.0`

mm：俺以为自动更新还是需要设定的

懒蜗牛：明天服务器上有 `cowsay1.1` 了

懒蜗牛：就会提示你更新，就是开机右上角的那个

mm：恩，这个貌似今天看到了

懒蜗牛：恩，恩

mm：嘻嘻

恩…… 不错，这 mm 有前途。

第三段：

mm：我发现还是你教我的用图形界面装软件方便，只要打开那个啥牛力的软件管理器，想装什么就搜名字就好了，然后右键点一下，选标记以便安装，就好了。

（看来是主人哪天现场指导去了，呵呵。）

懒蜗牛：恩，是阿。

mm：那干嘛还有人要用命令装阿。

懒蜗牛：其实命令熟的话，命令更快。而且命令有命令的好处阿，比如你电脑有啥问题，我与其告诉你先点哪个菜单，再点哪个菜单，再再选哪个选项，就不如直接告诉你一条命令，让你运行一下来的快。

mm：哦……是哈，那样适合我这样的懒人，嘻嘻。

懒蜗牛：呵呵

mm：今天同学来俺着看见这系统了，以为是 win7 呢，呵呵。

懒蜗牛：比 win7 可漂亮多了。

mm：恩，恩，俺也这么说。

懒蜗牛：握手，握手。

mm：嘻嘻，还有像这个这样漂亮的系统不？

懒蜗牛：这个只是 Linux 的一种，所有 linux 都可以弄成这样。

mm：哦，那还有什么其他 Linux？

懒蜗牛：有好多，比如 Gentoo，Fedora，suse，还有你们公司的那个 centos，好多呢。

mm：哦，那有什么不一样？

懒蜗牛：gentoo 是个能高度配置的系统，可以根据自己的需要配置整个系统，速度最快，因为里面的软件都是按照自己的机器配置而优化的。不过就是配置起来很

麻烦。

mm: 你用过么?

懒蜗牛: 以前用过, 挺不错的, 尤其可以滚动升级, 不像笨兔这样, 版本升级风险比较大。我还想哪天格了换回 Gentoo 呢。

(阿!!?? 别阿别阿, 我挺好用的, 升级也不麻烦, 相信我, 相信我。)

mm: 听起来是高手才用的, 别的呢?

懒蜗牛: Fedora, 是 redhat 公司出的免费版本。

mm: 红帽子阿, 听说过, 好像有 redhat 9.0?

懒蜗牛: 那是很老的版本了, redhat 在 9.0 之后分为两个分支, 一个是面向企业的收费的, 叫 Red Hat Enterprise Linux, 另一个是社区的免费版本, 就是 Fedora。

mm: 这样来的阿

懒蜗牛: 恩, 现在 Red Hat Enterprise Linux 已经发展到版本 5 了, 有人简单叫它 redhat5, 所以很多第一次接触的人误解, 觉得 redhat9.0 是最新的。其实都是老古董了。

mm: 我也差点……

懒蜗牛: 呵呵, Fedora 里面的软件都比较新, 发现了什么问题, redhat 公司再去改, 稳定了之后就放进收费版的 Red Hat Enterprise Linux 里。

mm: 合着拿免费用户做试验阿……

懒蜗牛: 也可以这么说……不过 Fedora 的确实能满足很多人求新的愿望, 而且, 有大公司作靠山, 作出来的东西还挺不错的, 我也想装一下试试呢, 从来没用过。

(不会又要把我格了吧……咱有 VBox 阿, 在虚拟机里试试就得了~)

mm: 装吧, 我支持。

mm: 我们公司那个 centos 呢? 什么来头?

懒蜗牛: 那个是 Redhat 收费以后, 社区创建的版本。目的是提供一个免费的服务器版本的 Linux。centos 基本上可以当作是 Redhat 的免费版, 版本号互相对应, 里面的软件也一样。

mm: 哦, 那, 那个 SUSE 呢?

懒蜗牛: 那是德国的一个发行版, 以界面漂亮著称, 不过有点慢。

mm: 哇……俺喜欢漂漂的东西。

懒蜗牛: 呵呵, 回头装一个你看看。

（她喜欢往她机器上装阿，别又来格我。）

mm: 哦……嘻嘻，反正你就是闲不住，老折腾电脑玩

（恩，我看也是）

懒蜗牛: 厄……这个……是吧。

（犹豫什么，就是!）

mm: 嘻嘻，无语了。

赶紧告诉兄弟们，一定要好好工作，不然随时炒鱿鱼，哎~~~遇上这么个主人也不踏实……

(69) LOSE

继续翻聊天记录，第四段：

懒蜗牛: 早上好

mm: 好，我的电脑装显卡驱动了么？

懒蜗牛: 你那是 Intel 的集成显卡，不用装。

（恩，恩，我就是这么方便，哈哈。）

mm: Intel 的显卡不装驱动就能用阿？可是在 windows 下怎么需要装驱动呢。

懒蜗牛: 厄……任何硬件都需要驱动程序的，包括硬盘、光驱、U 盘也都需要驱动。只不过这些常用的东西的驱动都集成在了系统里面或者 BIOS 里。WindowsXP 年头久了，当然里面没有你这个显卡的驱动，你装个 WIN7 看看，照样不用装驱动。都集成在系统里了。

mm: 哦，原来这样阿，那我还挺运气哈，碰上这么个显卡。

懒蜗牛: 恩，intel 对 linux 的支持还是不错的，如果是其他牌子的显卡，就麻烦一点了，有的显卡根本没法开 3D 效果。

mm: 那那些用别的显卡的人怎么装 linux 阿？

懒蜗牛: 装还是能装，不过就是不能开 3D，而且运行效率也差。

mm: 那帮人真可怜……

懒蜗牛: 是滴

mm: 那, 下一个问题, 我想跟你视频用什么软件呢? 这个 QQ 好像没这个功能阿。

(得, 问的软肋上了。)

懒蜗牛: 这个……可以用 skype, msn 也行, 反正 qq 是没戏了。

mm: 为什么呀?

(这 mm 整个一十万个为什么。)

懒蜗牛: 因为腾讯就只开发了这么个简单的 linux 版的 QQ 呗。

(原来叫腾讯阿, 我一直以为是疼殉呢……)

mm: 好麻烦阿, 还得用 msn。和你用 msn 也就凑合了, 和别人咋办, 人家要是没 msn 呢。

懒蜗牛: 这个……没辙。

mm: 哦, 你也没辙啦, 好吧……对了, 这系统下有啥游戏好玩阿?

懒蜗牛: 自带的那些玩过没?

(恩, 恩, 没事跟电脑下下国际象棋)

mm: 那些都好没意思阿, 有大点的没?

懒蜗牛: 魔兽世界?

mm: 那个好复杂阿, 不爱玩。

懒蜗牛: 你想玩啥?

mm: 比如跑跑卡丁车啦之类的。

(别说卡丁车, 连卡车也没有阿)

懒蜗牛: 没有……

(是吧)

mm: 那有什么游戏呢?

懒蜗牛: 基本上吧, 在 Linux 下就别想玩游戏了。

mm: 这么惨阿。

懒蜗牛: 其实也有, 有些网游。

mm: 还有网游阿? 好玩不?

懒蜗牛: 没玩过, 都是英文的……

mm: 俺恨鸟语……

懒蜗牛: 所以还是别玩游戏了, 这系统就是让你好好学习嘛, 呵呵

(换个话题吧)

mm: 我们最近学 ps 呢

懒蜗牛: 学点啥不好……

mm: 这个也没有?

懒蜗牛: 有 gimp, 功能和 ps 差不多。

(就是就是)

mm: 你给我装了么?

懒蜗牛: 默认就有, 你找找。

mm: 哦, 看见了。

懒蜗牛: 嘿嘿

mm: 怎么跟 ps 不一样阿?

(废话, 一样了就是 PS 了)

懒蜗牛: 当然不一样阿, 这完全是两个软件。

mm: 那跟我们老师教的都不一样, 我也没法用阿。

懒蜗牛: 厄……也……没辙这个。

mm: 啥破系统阿, 干啥都不行。

懒蜗牛: 恩……那……你要不还是用 Windows 吧。

mm: 可是这个好漂亮耶

懒蜗牛: 漂亮也不当饭吃。

mm: 也是, 回头你给我格了装 Windows 吧。

(又一个可怜的兄弟要惨遭不幸了……)

## (70) 宏&微

随着 USB 门口上的红灯一闪, 我知道又有 USB 设备接入进来了, 赶快打看门一看, 这回不是那个司空见惯的 1G U 盘, 而是一个网络设备, 好像……是个无线网卡? 赶快翻翻我身上的模块, 看有没有它的驱动。

我身上有很多的模块——别担心, 不是“肿块”, 不会影响身体健康。也不是“蘑菇”, 不能吃。是“模块”, 翻译成英文叫 module。这些模块像一本本的手册, 有的手册是说明如何使用某个硬件的, 这就是硬件驱动模块; 有的是说明如何使用某种文件系统的, 那就是文件系统模块, 等等。这些手册我都统一放在

/lib/modules/2.6.28-11-generic/目录下 (2.6.28-11-generic 是我的内核版本), 每次起床, 我都根据配置文件里写的内容, 把里面一些必要的手册揣在身上再去干活。

(就跟去旅游要揣个地图一样的道理) 当需要用到哪个东西的时候就掏出响应的手册来查看。比如要用 RealTek 的那块声卡了, 我就把关于 RealTek 声卡的那本手册 (也就是那个模块啦) 掏出来, 看应该怎么使用, 如何操作这个声卡。也有的东西, 很重要, 很关键, 很基本, 每次一定都会用到, 那样的就不做成模块了, 就直接让我记忆在脑子里, 融化在血液中——也就是所谓的编译进内核。哪些东西编译为模块, 哪些东西编译进内核, 哪些东西根本不编译, 这是在编译内核的时候就决定的。你也可以让我把所有东西都记忆在脑子里, 也就是所有的东西都编译进内核, 不编译成模块。但那样的话, 就基本没法干活了。倒不是我记不住那么些东西, 我不是人脑, 我想记住啥就记住啥, 但是要知道我是程序, 我要记住个东西的话, 体积是要增大的。一个所有东西都被编译进去的内核大约要二百多 M 那么大!! 这就意味着这内核一启动, 自己就至少得占 200 多 M 的内存, 那还怎么干活啊, 这点地儿全让他一人占了。

不过说起来, 我的祖先们——也就是最初的那些 Linux 内核, 是没有模块这回事的。那时候的 linux 内核要把所有需要用的东西都记住。比如要用到 ext2 文件系统, 那就把 ext2 文件系统的支持编译进内核。用不到 XFS 系统, 那对 XFS 系统的支持就不编译。等到那天需要 XFS 支持了, 就得重新编译内核, 把 XFS 支持编译进去, 然后重启, 用新内核启动系统。所以那时候的 Linux 内核是个典型的宏内核。所谓宏内核, 也叫单内核, 就是指像 linux 这样, 内核整体作为一个独立的进程在运行在内存里, 所有该实现的功能, 都在这个大进程里实现, 像进程管理阿, 内存分配阿, 文件系统管理阿, 硬件设备的控制阿等等这些事情。像我们 Linux, 还有传统的 Unix, 有点软公司的剁死, Windows 95, Windows98, 都是宏内核。与宏内核对立的, 还有一种叫微内核。微内核就不是一个人在战斗了, 微内核的理念与宏内核相反, 把内核该干的那点事分成一个一个的小块, 由一个个小的内核进程专门去管理。有专门管理内存分配的, 有专门管理进程的, 有专门管理硬件 IO 的, 等等。这样的好处就是进程间分工明细, 每个进程只专心管理自己那一点事情, 不容易出问题。而且, 可移植性也比较高, 只需要把直接跟硬件相关的部分移植一下就好了, 其他的部分基本不用动。宏内核就需要整个都移植, 因为是一个整体嘛, 要换整个换。像咱们说过的 Minix, 就是微内核。当宏内核工作的时候, 就是像我一样: 比如叫皮筋起床干活吧, 我先通过文件访问, 把皮筋叫进内存 (程序也是文件阿, 可执行文件), 然后给皮筋分配好内存

空间，为他创建个进程（也就是给他分配个工号），分给他 CPU 让皮筋开始干活，皮筋要访问网络的时候我负责操作网卡，把他要发的东西发到网卡上。这一系列的事情，全都由我一人管理。整个工作间里是以我为中心的工作。而微内核工作起来的景象就是：要内存的事都去找内核贾；要访问文件的程序，都去找内核余；跟硬件打交道的全去内核汤那；进程管理的问题都归内核顿管。内核余把皮筋从硬盘里交出来，然后喊“老贾，给皮筋分配点内存”，内核贾就给分配，分配好了跟内核顿说：“分个工号，创建个进程”。内核顿照做，然后皮筋开始干活，要访问网卡了，就去内核汤那报道。整个工作间里，软件们是以“顿贾余汤”内核小组为中心干活。

宏内核灵活性明显不高，这是个人就能看出来，所以现在我们 Linux 学会了通过加载模块的方式来增加灵活性，需要增加什么支持，只要加载一个新的模块就好了，不用重新编译内核，不用重启计算机。其实这也算是跟微内核那里学来的了。呀，说了半天主人接进来的这个网卡……好像我这里没有它的驱动模块阿……

## (71) 无线网卡

我看了一下插进来的这个网卡，是 Realtek 的 RTL8180L 芯片，再仔细翻翻我的所有模块……确实没有，坏了，这回恐怕要在主人面前丢脸了。主人用 `ifconfig` 查看网卡，我只好汇报：现在机器上有两个网卡，一个是有线网卡 `eth0`，这个正常工作，另一个是虚拟的回还网卡 `lo`，这个也没啥问题。（闭口那不提无线网卡的事）主人好像很纳闷，心说我这明摆着多插了一个无线网卡阿，你怎么就装看不见呢？他叫来狐狸妹妹，让她去问狗狗哥这 TP-Link 的 WN210 网卡怎么用。狐狸妹妹找到一个叫做 Ubuntu 中文论坛的地方，里面也有人问怎么用这网卡。听得我这叫一个着急，你找也找 RTL8180L 这芯片阿，关键是这个芯片的型号，不是那网卡的型号，搜芯片会多不少记录呢。哦，对了，可能他压根不知道这网卡是啥芯片。那你倒是问我阿，问一句 `lspci` 我不就告诉你了么，哎，我也是，皇上不急太监急。

主人的悟性还是挺高的，一会就想起来问我了，我赶紧告诉他网卡型号，他就去查去了，得出的结果是——就是没有 Linux 驱动！那这网卡就算没法用了？当然不是，虽然没有 Linux 驱动，但是，困难压不到我们 Linux 软件，随着狐狸妹妹的点拨，主人知道了有一个软件，叫做 `ndiswrapper`。这个软件会干啥？他能读懂硬件的驱动——读驱动本来是我的工作，就是那些驱动模块阿，但是人家读的是硬件的 windows



驱动，翻译成我能懂的 Linux 模块，然后就可以使用这个卡了。不过他只能翻译些网卡驱动，不过这也差不多够了。反正这块卡是能支持。

超级牛力瞬间拉来了 ndiswrapper，安顿好后 ndiswrapper 立刻被叫起来干活，主人给他指了指那个 xxxx.inf 的 windows 驱动文件，ndiswrapper 赶紧拿起来读，详细研究了一下后表示，可以支持，只要加载好他给我建造出来的和他同名的模块 ndiswrapper.ko 就可以了。主人按照狐狸妹妹找到的文档一步一步操作：先用 ndiswrapper 加载那个 windows 驱动，然后在让我加载那个 ndiswrapper 模块，最后问我，现在这个网卡状态是什么样阿？我充满信心的回答：现在机器上有三个网卡，一个是有线网卡 eth0，这个正常工作，另一个是虚拟的回还网卡 lo，这个也没啥问题。还有一个无线网卡 wlan0，也正常工作。主人很欣慰的点点头，一股成功感油然而生。不过这无线网卡跟有线的不一样，有线的插上之后，配好 IP 就能用，这无线的得先建立好无线连接，这无线连接建立好就好比有线网卡插好了网线。建立连接也不是什么困难的事情，我们这里有专门的团队负责。图形界面的有 NetworkManager，跟网络有关的设置，甭管无线的有线的，找他就行，跟查皮底下差不多。如果用命令的话有 iwconfig 可以查看和配置无线网络，还有 iwlist 可以查看周围可用的无线网络，可能会找到邻居家的没设密码的信号哦～

## (72) 驱动

主人的无线网卡没有搜索到安全意识薄弱的无线邻居，当然，人家压根也没打算搜到，只是简单的链接到了自己家的无线路由上。不过估计这网卡不会常在我这插着，您想啊，我这是台式机，千兆的有线网用的好好的，没事用的什么无线啊，我又不移动。估计是主人给那个笔记本电脑用的，先插在我这里试试，研究一下能不能驱动上，然后再往本本上插。

这个电脑硬件阿，不像电视机电冰箱似的，买来插上就能用。硬件要想在计算机上工作，得需要会操作它的软件，这个事情，一般就是归我们操作系统管了。But 子曰：

“人非生而知之者，孰能无惑？”这时候 00 老先生推了推大眼镜咳嗽了一下说：“那不是子曰的，是韩愈曰的”好吧，不管是谁说的，反正道理是这个道理，我们操作系统，也不可能生来就会操作所有的硬件，就像你不是生来就会开飞机一样，得学，得

考本，得移库、倒库、坡起、限制门。00 老先生再次严厉的咳嗽了一下：“你见过飞机过限制门么？！那是汽车。”反正，我们要想会操作一个硬件，也需要学习，这就需要驱动程序，任何硬件要想工作都是需要驱动程序的。这时候可能有人提出反对意见了：“硬盘，光驱，这些也都是硬件哪听说过要装驱动程序的？还有我的 U 盘，摄像头，也都是插上就能用，不用装驱动阿。”不用装驱动，不代表不需要驱动。硬盘光驱是最基本的存储设备，在我们操作系统起床工作之前，硬盘就要工作（因为我们都住在硬盘里嘛，必须硬盘工作，我们才能被读进内存里。）这个时候其实是另一个软件——BIOS 在操作硬盘工作，硬盘的驱动，就在 BIOS 里。关于 BIOS 这老人家，我们以后细说。反正硬盘光驱这样的基本设备的驱动很简单，也统一，任何一家生产的硬盘都是一样的用法，所以硬盘光驱的驱动就被集成在了 BIOS 和操作系统里面，不用额外安装。其他所谓不用装驱动的设备也一样，都是因为驱动集成在了系统里。比如查皮他们家以前的瘟酒吧系统，就不认识 U 盘，需要装驱动才行。到查皮这一代，就不用装了，集成了。

驱动就像一本给操作系统看的使用手册，上面写明了如何如何操作这个硬件，写哪个寄存器就把数据发出去了，从哪个寄存器读就把数据读回来了，往哪个寄存器写个什么什么数据就自爆了等等。（这是什么硬件阿……）就像买来电视机，里面的使用手册一样。针对不同的操作系统，需要有不同版本的驱动程序。您想阿，我和查皮说的话都不一样，他那边的程序都没法直接和我交流，还得通过红酒大师，那我们能看得懂的手册，自然也就不一样。你家电视机的说明书不也有中文版，英文版，韩文版，非洲土著语版么。但是，并不是每个硬件厂家都给每个系统制作一份驱动的，毕竟厂商人力财力有限。电视机也不是每个都有非洲土著语版的说明书嘛。（压根就没有把……）所以，一般硬件厂商会优先开发市场占有率最高的那个系统的驱动程序，哪个系统？目前来说，就是查皮和他的后代喂死他、温妻了。而我们 Linux 就经常遇到一些无法使用的硬件，很多人还抱怨我们无能，冤枉阿～～～

### （73） 显卡驱动

其实我们 Linux 能够支持的硬件已经逐渐多起来，大多数主流的设备基本不用装驱动就可以使用了。一般像我们 Ubuntu 吧，装完了系统之后也就装装显卡驱动就可以了，

没准连显卡驱动都不用装。比如我住的这台电脑，用的是 Nvidia 的显卡，这家公司对我们 Linux 还算比较友好，提供了不错的 Linux 驱动。那时候我刚刚搬进来不久，主人想要实现那些很花哨的 3D 桌面效果，就让超级牛力安装了源里的 nvidia-glx 驱动——对我来说就是个手册啊。可是这个手册写的很简单，实现起一些高级的效果来很吃力，于是主人决定还是去安装 nvidia 官方的驱动。

主人发话，狐狸妹妹立刻就去官方网站上找到了那个驱动的安装程序，然后一个媚眼就把那小子领回了家里，小子名字还挺长，叫什么 NVIDIA-Linux-x86\_64-190.53-pkg2.run，从名字看是专门给我这样的 64 位 Linux 服务的。回来之后主人先是让超级牛力删了原来的驱动——怕有冲突，两本手册都叫“xxx 显卡驱动”，万一我一忙乎拿错了不就乱了么。删了之后主人双击那个新来的驱动安装程序，让他运行，结果那小子派头挺大，跟主人说“你这个……图形界面还开着呢，没法装，先把图形界面关了再找我！”主人无奈，只好 ctrl-alt-f1 进入了字符界面，登录之后运行 `sudo /etc/init.d/gdm stop`，意思就是告诉图形界面那哥几个，回硬盘歇着去吧，暂时没你们事了。顺便插一句，其实我还是喜欢主人用文字跟我交流，有种平等的，倾诉心声的感觉，比在图形界面里被指来指去的舒服多了。图形界面哥几个彻底休息之后，主人又运行 `/<路径>/NVIDIA-Linux-x86_64-190.53-pkg2.run` 叫醒那个安装程序，这回小子又把嘴一撇：“你是 root 么？不是 root 没资格跟我说话！”气的主人抓耳挠腮，只好乖乖的在命令前再加上 `sudo`，这回那小子终于运行了，现实叽哩咕噜的说了一大堆英语，好象是问主人要不要去网上下个啥东西，主人毅然决然的回绝了他。之后的过程很顺利，小子仔仔细细的吧这里检查了一遍，迅速编写出一本我看得懂的模块塞到我的手里，又改好了 xorg 老大需要用到的配置文件，然后向主人汇报：“行了，没事了，重启图形就好了，我睡去了。”

主人又用 `sudo /etc/init.d/gdm start` 叫醒了图形界面那帮倒霉的孩子们——被子还没铺好呢又被叫出来了。然后设置桌面效果，果然，成功了。

nvidia 的那个驱动程序虽然很拽，但干活还是很在行的，主人能成功，也要归功于当初选了他家的显卡。如果是 ATI 那家的显卡就不这么容易了。不过其实我也没见过，只是这么听别人说，听说他家的驱动要靠人品，呵呵。再有比较好的就是淫特二那家的集成显卡，他家的显卡驱动直接贡献给我们学校，我们走出学校的时候就自备他家的显卡驱动模块了，而且支持的挺好，不用再安装了。不过听说我们的学弟——9.04 那一届的淫特二显卡驱动模块出了点问题，可能是印刷质量太差吧，驱动器来很费劲，

不过升级一下内核也就没事了。除了这三家之外，其他家的显卡基本上就只能凑合着用了，能把分辨率搞对了就不错，更别说 3D 了。

#### (74) SLax

随着 VBox 挥舞着魔杖一顿乱比划，又一个虚拟的电脑出现了，这回要住进去的不是查皮，而是一个我的同行，一个 Linux。

前一阵子狐狸妹妹拖回来了一个 iso 文件，是一个叫做 Slax 的 linux 系统。Slax 是一个闲不住的家伙，专门喜好移动办公，他很适合被装在光盘或者 U 盘上，被放到各种各样不同的机器上运行，不像我们，基本上只在一台机器里干活。Slax 是基于 Slackware 发行版的，说起 Slackware 那可是历史悠久了。最早的 Slackware 1.0 版在 1993 年 7 月 16 日就发布了，创始人是 Patrick Volkerding 老大。那年代交通还不发达，不像我们现在有光盘坐，甚至还有宽敞明亮，容量高达好几 G 的豪华 DVD，那时候 Slackware 的发布只是用 3.5 寸软盘。到现在已经有十好几年了。Slackware 的宗旨是力图成为“UNIX 风格”的 Linux 发行版本。它的方针是只吸收稳定版本的应用程序，而且力图让系统结构简单，尤其他的包管理方式，没有我们这里的超级牛力，也不用帽子店那里的大晕头 (yun)，而是直接用 tar+gzip 打的包，软件包的扩展名都是 .tgz，安装一个软件包就往根目录一解，齐活。Slax 之所以基于他，估计就是因为他的简单吧，毕竟要在 U 盘或者光盘上实现一套完整的系统，而且还要集成进去尽可能多的驱动，空间大小是个问题。

VBox 创建好了虚拟机，主人立刻挂上那个 Slax 的 iso 文件，让 Slax 展露他的技艺。Slax 身体轻巧，带着他那帮身体同样轻巧的弟兄们，迅速跑进虚拟机的内存，开始工作。有道是麻雀虽小，五脏俱全，别看人家只有 200 多 M 的 ISO 文件，带的软件不必我当初带来的软件少。什么文字处理的，网络聊天的，看网页的，画图的，算数的，小游戏等等，甚至连咱之前说过的 ndiswrapper 都有，想的很周到，听说还有一些 Slax 的版本，连红酒大师都自带。主人在虚拟机里分别试了试那些软件，觉得还不错，然后就把虚拟机关了。我估计，他是要叫来刻录软件把这 iso 刻成光盘来用，可是等了一会，没见动静，却看到 USB 门上那盏灯亮了。我赶紧去打看一看，还是那个熟悉的 1G 的 U 盘，哦……主人是想把这个系统装在 U 盘上。

可是，要往 U 盘上装得有软件啊，毕竟这是个 iso 文件，要刻光盘那是现成，要写 U

盘就不那么简单了，比如我知道有个软件叫 UNetbootin 就是能够把可以启动电脑的 iso 文件刻录到 U 盘上，成为系统 U 盘。可是我们这也没有这软件，主人怎么能把 iso 放到 U 盘上呢？总不能把文件直接解压到 U 盘上就完事吧，那可启动不了，启动信息可不在那些文件里。我这还没想明白，只见主人他，他，他还真就把 iso 里的文件完全彻底的往 U 盘上一解。我的主人啊～你没犯过这么弱智的错误啊～这也太天真了吧，这能启动电脑？咦？别急，主人又动作了，他让我们去运行刚刚解压到 U 盘上的一个脚本，u 盘里 boot 目录下的 bootinst.sh。哦～和着这个 Slax 连安装到 U 盘的软件都给您备好了，真是不错。仔细看看，还有个 bootinst.bat，这个是在 windows 下执行用的，这样不论在什么系统下，都能很轻松的制作 Slax 的启动 U 盘了。bootinst.sh 那家伙开始工作，问了主人一些简单的问题后，就在 U 盘上写下了引导信息，这样，一张启动 U 盘就做好了。