第9章　备　　份

本章将讨论为什么要备份，怎样备份，何时备份以及如何通过备份恢复数据。

9.1　备份的重要性

谈到备份的重要性，相信大家都有过痛失数据的惨痛经历。而这种丢失数据的心情，比失恋还要难受，甚至让人心灰意冷。因为你的数据是有价值的。一旦丢失，重新建立它，得花你许多时间和精力，有的甚至还要花不少钱；更糟的是，有时甚至不可能重建它。数据包含着你的喜与乐，包含了你的工作成果，所以你应该保护它，采取相应的行动，以避免失去它。解决之道就是备份。

数据可能丢失的原因基本上可归结于四个：硬件故障、软件错误、人为因素和自然灾害（第五个原因就是“其他原因”）。虽然现在硬件越来越可靠，但仍然可能存在例外的情况。保存数据的硬件部分中，最关键的部分是硬盘，除了会发出微小的噪声外，它几乎是最理想的存储设备。现在的软件就不那么可靠了，当然，坚如磐石，固若金汤的软件产品也不是没有，只不过少罢了。人才是最不可靠的；他们不是操作失误，就是有目的地、或者恶作剧地破坏你的数据（比如，前不久，管理员就对我的主页开了个玩笑，让它消失了一晚上）。自然灾害只能怪你运气不好。总的说来，谁也不能绝对保证你的数据不遭破坏或损毁。

所谓备份，就是保护数据的一种手段。通过保留若干个备份的方式，能够在数据受损的情况下，尽快地恢复它（用不着伤心，用不着花太多的时间、精力和钱，只须从备份恢复丢失的数据即可）。

适当地备份是非常重要的。和现实生活中没有永恒的事物一样，备份迟早会无效。适当备份过程中最重要的是保证备份能够发挥作用；你肯定不希望看到自己的备份不能用吧（不要笑，有些人就是这样的）。有时候是屋漏偏遭连夜雨，就在你制作备份的时候，系统却崩溃了；如果你只有一个备份媒体的话，它也有可能遭到损毁，让你气得鼻子冒烟。或者说，在你试图恢复数据时，发现自己居然忘了对一些重要数据进行备份，比如一个有 15 000用户的站点上的用户数据库。最为理想的是，你的所有备份都能用。

9.2　选择备份媒体

关于备份，最重要的是如何选择备份媒体。在选择备份媒体时，需要考虑这几个因素：价格、可靠性、速度、可得性和可用性。

价格是比较重要的因素，因为你可能有许多数据需要多次备份。选择便宜点的存储媒体是必须的。

可靠性相当重要，因为一个被破坏了的备份无异于没有备份，想象吧，面对丢失了的数据，你满心以为用备份来恢复，结果备份也是被破坏了的，这种情形真让人欲哭无泪。在理想情况下，备份媒体能够保存数据的时间长达数年。对备份媒体而言，你使用媒体的方式也会对其可靠性产生一定的影响。一般来说，硬盘是非常可靠的，但如果备份媒体和你正在备

份的磁盘处于同一台机器上，备份媒体就不那么可靠了。

通常情况下，如果备份时不需要交互操作的话，速度就不是十分重要。只要它不需要人监管，花上两小时也无所谓。但另一方面，如果因为计算机空闲，备份不能完成的话，就应该考虑备份速度了。

显然，可得性是必需考虑的因素，总不可能在没有备份媒体的情况下，使用它吧。比如说，你没有光盘刻录机，怎么能将自己的数据备份在光盘上呢？（将来逐渐不需要拥有备份媒体，而是可从别的计算机上获得备份数据）。如果不考虑存储媒体的可得性，就不可能在灾难之后恢复数据。

在考虑备份频率时，可用性最为重要。制作备份的方式越简单越好。备份媒体绝对不应该不好用。

备份媒体一般采用软盘和磁带。软盘非常便宜、相当可靠、速度不是非常快、非常容易获得，但不适合量大的数据。磁带则有贵的、有便宜的，它相当可靠、速度相当快、非常容易获得（和磁带的容量大小有关），相当好用。

另外还有一些备份媒体。它们通常不太容易获得，但它们在其他方面的表现却非常出色。比如，光盘既有软盘的优点（随机访问，很快就能恢复单独的一个文件），又有磁带的优点（保存大量的数据）。

9.3 选择备份工具

用于制作备份的工具有许多。过去用于备份的 Unix工具是tar、cpio和dump。另外，还有许多第三方工具包（即有自由软件，又有商业软件）。备份媒体的选择直接对备份工具的选择产生影响。

从备份角度来说，tar和cpio极为相似。两者都能把数据保存在磁带上，并从磁带恢复数据。两者均适用于任何一种媒体，因为内核设备负责管理低级控制设备和越来越像用户级程序设备。tar和cpio的有些Unix版本可能不能备份非普通文件（比如符号链接、设备文件、带有超长路径名的文件等），但它们的Linux版本则可以轻松胜任所有文件的备份。

dump不同于前两者的地方在于：它直接读取文件系统，而不是通过文件系统读取文件，它也是专门针对备份而设计的；tar和cpio其实是用于对文件进行归档的，但同样可用于备份。

直接读取文件系统有诸多好处。比如说，可在不影响文件时间戳的情况下，备份文件；如果利用tar和cpio，你就必须先以只读方式装入文件系统。此外，如果所有文件都需要备份的话，直接读取文件系统也更为有效，因为这样可减少磁头的“运动量”。直接读取文件系统的缺点在于备份程序只能专用于一种文件系统类型； Linux dump只能理解ext2类型的文件系统。

dump还支持备份级别（稍后着重讨论）；对 tar和cpio来说，必须借助于其他工具，才能实施备份级别。

至于第三方备份工具，不在本书讨论之列。 Linux Software Map列出了许多用于备份的自由软件。

9.4 简单备份

简单备份方案指一次性备份所有的文件，然后再备份上次备份之后所做的修改。这种方

案中，第一次备份称为“完全备份”（full backup），后一次备份称为“新增备份”（incremental backups）。完全备份通常比新增备份费力得多，因为需要写入磁带的数据比后者多得多，而且完全备份需要的磁盘（或软盘）可能不止一盘。但和通过完全备份恢复数据相比，从新增备份中恢复数据所花的时间和精力是前者的数倍。当然，数据的恢复也是可以优化的，这样一来，就可以一直备份自上次完全备份以来的所有数据；这时虽然有些费神，但至少除了恢复一个完全备份和一个新增备份以外，无须再恢复其他的东西了。

如果你打算每天都进行备份，而且手中有 6盘磁带，就可用磁带 1来备份第一次完全备份（比如说，从星期五开始），磁带2和5用于新增备份（星期一到星期四）。然后，用磁带 6开始第二次完全备份（第二个星期五），并再次用磁带2和5进行新增备份。如果你不打算在执行新的完全备份之前，改写磁带 1，就不会有异常情况发生。在用磁带 6进行第二次完全备份之后，要把磁带1保存在某个地方，以便在其他备份磁带受损的情况下，仍然还有起死回生的最后一线希望。在需要进行下一次完全备份时，可采用磁带 1，让磁带6保持原样。

如果你手中的磁带不止6盘，可用多盘磁带来保存完全备份。每次执行完全备份时，都采用最早的那盘。这样，就能够有数周以前的完全备份，从而方便你找出原来的、现已删除了的文件或某文件以前的版本。

9.4.1 如何利用tar进行备份

利用tar，可轻松地进行完全备份：

```
# tar -create -file /dev/ftape /usr/src
tar: Removing leading / from absolute path names in the archive
#
```

上面的例子中，采用了tar的GNU版本及其长选项名。过去的tar版本只能识别单一字符选项。对于一盘磁带或软盘装不下、而且带有超长路径名的备份文件， GNU版本也是能够处理的；但并非所有的版本都能如此（Linux只用GNU tar）。

如果一盘磁带装不下你的备份，你就需要用多卷（-M）选项：

```
# tar -cMf /dev/fd0H1440 /usr/src
tar: Removing leading / from adsolute path names in the archive
Prepare volume \#2 for /dev/fd0H1440 and hit return:
#
```

注意，在开始备份之前，应该格式化软盘，否则就用另一个窗口或虚拟终端，在 tar要求新软盘的时候，开始备份。

做完备份之后，应该利用-compare(d)选项来检查备份是否正确：

```
# tar -compare -verbose -f /dev/ftape
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
....
#
```

不检查备份意味着你可能在已经丢失原始数据之后，才发现自己的备份有误。

新增备份利用-newer(-N)选项，通过tar来执行的：

```
# tar -create -newer '8 Sep 1995' -file /dev/ftape /usr/src -verbose
tar: Removing leading / from absolute path names in the archive
usr/src/
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/modules/
usr/src/linux-1.2.10-includes/include/asm-generic/
usr/src/linux-1.2.10-includes/include/asm-i386/
usr/src/linux-1.2.10-includes/include/asm-mips/
usr/src/linux-1.2.10-includes/include/asm-alpha/
usr/src/linux-1.2.10-includes/include/asm-m68k/
usr/src/linux-1.2.10-includes/include/asm-sparc/
usr/src/patch-1.2.11.gz
#
```

不幸的是，tar注意不到文件的inode信息是何时更改的，比如，文件的访问许可部分或文件名是何时更改的。出现类似情况时，解决办法是用 find，并把当前文件系统状态和以前备份过的文件列表进行对比。Linux ftp站点上可找到所需的脚本和程序。

9.4.2 如何利用tar恢复文件

tar抽取文件所用的选项是-extract(-X)：

```
# tar -extract -same-permissions -verbose -file /dev/fd0H1440
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/kernel.h
...
#
```

还可通过在命令行指定的方式，抽取特定的文件或目录（其中包含所有文件和子目录）：

```
# tar xpvf /dev/fd0H1440 usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
#
```

如果你只想看备份卷上有些什么文件的话，可用 -list(-t)选项：

```
# tar -list -file /dev/fd0H1440
usr/src/
usr/src/linux
usr/src/linux-1.2.10-includes/
usr/src/linux-1.2.10-includes/include/
usr/src/linux-1.2.10-includes/include/linux/
usr/src/linux-1.2.10-includes/include/linux/hdreg.h
usr/src/linux-1.2.10-includes/include/linux/kernel.h
...
#
```

注意，tar始终依序读取备份卷，所以对于大型卷来说，读取速度是非常慢的。但是，在

使用磁带机或其他媒体时，也不可能采用随机访问数据库的方式。

tar不能正确处理已删除的文件。如果你需要从一个完全和新增备份恢复一个文件系统，而且你已经在两次备份之间删掉了一个文件，那么在完成恢复之后，这个文件又出现了。如果这个文件中是一些不再可得的敏感数据的话，问题就严重了。

9.5 多级备份

前一小节中解释的简单备份通常适用于个人或小型网站。对于机构或大型网站来说，则需要采用多级备份。

简单备份共有两个备份级别：完全备份和新增备份。笼统地讲，还可分为更多的备份级别。完全备份是1级，新增备份的级别分别是1、2、3等。每个新增级别上，都可备份自同级或上一级的上次备份以来的所有变动。

采用多级备份的目的是能够更便宜地保持较长的备份历史。前一小节的示例中，备份历史又回到了前一次完全备份。这是可以通过多盘磁带进行扩展的，但一周用一盘新磁带，可能代价太高了。拥有较长的备份历史是很有用的，因为被删除或损毁的文件通常很长时间都不能被察觉。即使某文件的版本不是很新，但总比根本没有强。

利用多级备份备份历史能够在少花钱的情况下，得以轻松扩展。举个例子来说，如果我们有10盘磁带，就可以把磁带1和2用于月备份（每月的第一个星期五），磁带3到6用于周备份（每月除开第一个星期五的星期五；注意，有时一个月可能有 5个星期五，所有可能还需要4盘磁带），磁带7到10用于日备份（星期一到星期四）。只用4盘磁带，我们已能够把备份历史从两周（所有的日备份磁带用完之后）扩展到了两个月。我们不能恢复每个文件在两个月内的每个版本，但我们能够选择最完整的版本进行恢复，这是假不了的。

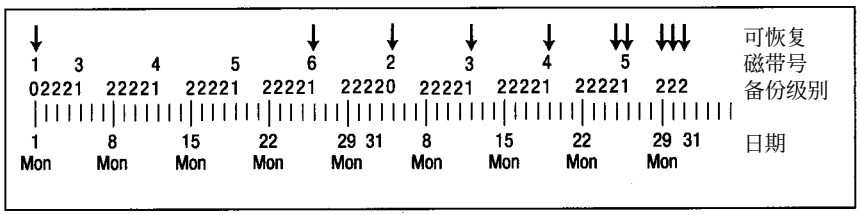

图9-1 多级备份日程表示例

图9-1展示了每天所用的备份级别，以及月末时，可从哪些备份中恢复。

备份级别还可用于使文件系统的恢复时间缩短至最少。如果你有许多新增备份，这些备份的级别号是单向递增的，那么，在你需要重建整个文件系统时，就需要一一恢复所有的新增备份。如果你采用的备份级别不是单向递增的（即是无序的），就可以减少备份数。

为了最小化恢复数据所需的磁带数，可为每个新增磁带采用一个较小的备份级别。但是，执行备份的时间将会增加（每个备份都要复制自前一次完全备份以来的所有更改过的数据。因此，dump手册页建议采用另一种较好的方法，如表 9-1所示（有效的备份级别）。利用其中的备份级别：3、2、5、4、7、6、9、8、9，以此类推。这样一来，就可将备份和恢复所需要的时间缩至最短，只须你执行两天一次的备份工作。恢复所用的磁带数取决于完全备份之间的时间，但它少于简单备份所需的磁带数。

表9-1 利用多级备份的有效备份方案

磁带编号	备份级别	备份（日期）	恢复磁带
1	0	n/a	1
2	3	1	1, 2
3	2	2	1, 3
4	5	1	1, 2, 4
5	4	2	1, 2, 5
6	7	1	1, 2, 5, 6
7	6	2	1, 2, 5, 7
8	9	1	1, 2, 5, 7, 8
9	8	2	1, 3, 5, 7, 9
10	9	1	1, 2, 5, 7, 9, 10
11	9	1	1, 2, 5, 7, 9, 10, 11
...	9	1	1, 2, 5, 7, 9, 10, 11

新的解决之道可能会减少你的工作负担，但意味着你还需要多了解其他的一些东西。究竟采用哪个方案，必须自己拿主意。

dump本身支持备份级别。而对tar和cpio来说，必须通过外壳脚本来实现。

9.6 要备份什么

任何情况下，人们都希望尽可能地保留备份。例外之一：可以轻松重新安装的软件（得了解一下“轻松”一词的含义；有人把利用一打软盘来安装软件也称为轻松），即使备份配置文件非常重要，只需重新配置即可。另一个例外是 /proc文件系统；由于只有它才包含了始终由内核自动生成的数据，要对它进行备份绝不值得尝试。特别是 /proc/kcore文件，最没有必要备份，因为对你而言，它只是当前物理内存的一个镜像而已；它的数量相当大。

灰色区内包含的是新闻假脱机、日志文件和 /var内的其他一些东西。你必须自己决定哪些是最重要的。

很显然，需要备份的东西是用户文件（ /home）和系统配置文件（ /etc，但可能还有散布在文件系统各处的其他文件）。

9.7 压缩备份

备份会占用大量的空间，可能会花很多钱。为了减少所需空间，节省开支，所以要对备份进行压缩。常见的方式有许多。有的程序内置压缩支持；比如 GNU tar的-gzip（-z）选项，它在把备份写入备份媒体之前，通过 gzip压缩程序，把整个备份压缩在一起。

不幸的是，压缩备份通常会带来许多不便。由于压缩的本性，如果其中有丁点错误，其他的压缩数据都将不能用了。有的备份程序有内置的纠错功能，但面对大量的错误，也是无法解决的。这意味着如果备份是像 GNU tar那样压缩的，也就是说，把整个备份压缩成一个整体，其中有一个错误的话，将导致其他所有备份数据的丢失。由于备份必须可靠，因此，这种压缩方式不见得是上上之策。

另一种方法是对每个文件进行单独压缩。这意味着其中一个文件丢失，其他的文件却不会受损。丢失的文件总之已经受损，所以只能说，这种方法比没有使用压缩好一点。 afio程序

（cpio的变体）也用于此。

压缩会花些时间，这样一来，会令备份程序不能以很快的速度，把数据写入磁带驱动器（如果磁带驱动器不能连续收到数据，它就必须停下来等待数据的写入；这样一来，备份程序的速度甚而更慢，停停走走的状态对磁带和驱动器来说，都不是件好事）。但这种情况可以通过缓冲备份输出的方式得以避免，但同样收效甚微。不过，这种情况一般发生在较慢的机器上。