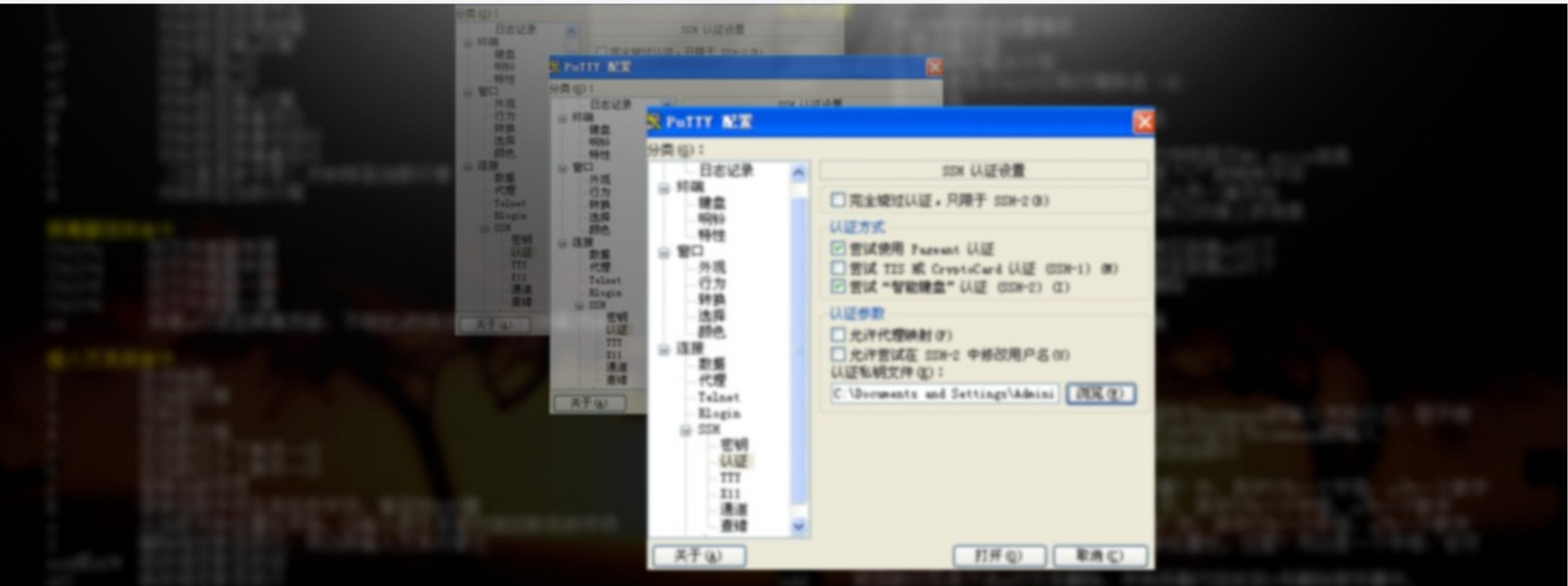
51CTO.com
技术成就梦想

Linux 运维趋势

2010 年 12 月号 总第三期

本期主题：运维与开发 | 关键字：Shell, 脚本, 生产环境, Crontab



内容目录

Linux 运维趋势

杂志策划：51CTO 系统频道

本期主编：杨赛

Logo 制作：高鹏飞

封面制作：徐泽琼

技术顾问：曹江华，黄琨，李洋

特别顾问：王文文，杨文飞

交流圈子：

http://g.51cto.com/linuxops

邮件群组：

groups.google.com/group/linuxops-cn

订阅方式：发送 Email 到

linuxops-cn+subscribe@googlegroups.com

专题页面：

http://os.51cto.com/art/201011/233915.htm

http://down.51cto.com/zt/71

投稿信箱：

yangsai@51cto.com

1999 年在计算机世界报刊看到了一点点关于 Linux 的介绍，我被其中“源代码全部开放”所吸引。

专访资深系统管理员曹江华：从车间走入 Linux 世界

采访、整理/杨赛

人物简介

曹江华，1999 年开始从事构建网络、管理维护、数据库管理工作。1999 年后开始接触 LINUX，将工作中的经验总结后已出版《Linux 服务器安全策略详解》，《Linux 服务器安全策略详解》（第二版），《Red Hat Enterprise Linux 5.0 服务器构建与故障排除》，《Linux 系统最佳实践工具：命令行技术》四本堪称 Linux 系统管理员日常工具书的热销图书，目前关注 Linux 和网络安全管理。

随着 IT 技术在上世纪 90 年代开始在中国普及，这个产业也簇生了国内第一批系统管理员、网络管理员的诞生。相对于现在无所不在的以 Windows 为基础的企业网络和奔跑在无数 Linux 服务器上的网站，90 年代的 IT 基础架构还处于非常原始的状态。互联网进入国内差不多是 1995 年左右，而 1999 年正是 IT 行业飞速发展的一个转折点。那个时候的技术人们，无论是开发者还是运维们，虽然可以获取的资讯和技术知识相对贫乏，但是那时的人们都充满着对技术的热情，投入到新技术的学习和使用当中。

国内著名的 Linux 专家曹江华老师，就是 1999 年投入到运维领域的。51CTO 系统频道近日对曹江华老师进行了邮件专访，请曹江华老师谈了谈自己的职业发展经历。一方面让大家了解一下那个时候的系统运维的工作状况，另一方面也给现在的系统运维们分享一些学习经验（这些经验之谈可能不仅仅影响到你在 Linux 方面的学习）。

51CTO：首先简单的介绍一下您自己吧。能否简单的介绍一下您在运维领域的经历？比如什么时候进入这行，现在在哪里工作等等。

曹江华：我毕业于工科大学机电一体化专业，当时从事的是 CAD 设计。后来开始从事小型数据库的应用。1999 年开始从事构建网络、管理维护、数据库管理工作，1999 年后开始接触 LINUX 成为 Linux 系统管理员。目前关注 Linux 和网络安全管理。现在是 linuxpilot（现在是唯一的中文 Linux 专业纸媒体了）的专栏作家和自由撰稿人。

51CTO：您最初接触系统管理员这方面的工作是怎样的情况？看您的一些介绍，好像您一开始是从事 CAD 设计的吧。能跟我们描述一下当时您是如何进入运维这个领域的么？您当时的工作环境是怎样的？如何学习一项新的技能呢？

曹江华：我进入计算机系统管理员非常偶然。大学学的机电一体化专业计算机课程只有两门（for77 计算机语言和 CAD 绘图）。工作以后到了北汽集团发动机工厂生产线，一干就是三年。第一年实习。后边两年是当班车间主任。这三年的工作几乎和计算机没有任何关系。三年后工厂发动机工厂生产线停产了，我进入北汽的计算机中心。其实我当时也没什么关系，那时国有企业效益太差了，分配来的计算机专业的大学生全走了，实在没有人可用了；因为

我工厂生产线三年，一直是劳模，领导说你去干干吧，我就去了。刚开始是CAD绘图，这个大学学过，当时几乎都忘了用三个月时间重新学习。1997年开始使用dos环境的DbaseIII数据库，后来是Foxbase数据库，现在可能计算机专业同学可能都没人知道了。

1999年在计算机世界报刊看到了一点点关于Linux的介绍，我被其中“源代码全部开放”所吸引。我有点不敢相信，因为当时正在学习编程，正苦于为找不到系统代码而发愁。从那以后，每一份杂志我都要浏览一下有没有Linux方面的消息。我在没有电脑、连Win32操作都不熟练的的情况下，买了一本《Linux从入门到精通》，然后安装《微电脑世界》随书赠送的蓝点Linux（51CTO编辑注：蓝点/BluePoint是那个时候红极一时的中国Linux发行版，只用了半年就在纳斯达克上市，市值4亿美元。不过后来由于种种原因而以出售作为终结，所以现在的Linux用户们可能没听说过这个发行版）。我硬是把这本书从头到尾看了一遍，终于搞明白了什么是分区、什么是mount。因为我没有学习过Windows，所以和其他用户相比，对命令行并不排斥。对于我来说感觉Windows系统和Unix、Linux是一样的，没有什么不同。

51CTO：回顾您在系统运维领域这么多年的经历，您觉得哪段时间是您成长最快的？那段时间有没有什么记忆深刻的人或事，或者什么特别的经历或机遇？

曹江华：在系统运维领域多年，感觉还是在北汽集团计算机中心的五年印象最深刻。当时企业内已经建立了网络环境，可是那时国有企业效益太差了根本没有条件升级计算机。当时win 95已经问世了，可是我们的计算机配置大多是386-486，只有几台586还是领导办公打字用。我在386上运行windows几乎跑不起来，运行linux倒是没有什么影响。我就是这样开始了Linux之路。

51CTO：您对Linux和Solaris系统写过不少文章，也出版过好几本书籍了。最初开始写这方面的文章，是有什么契机么？

曹江华：我开始写作是看到2000年《开放系统世界》创刊号，我把自己写的两篇Linux配置总结的稿件发送给他们，马上他们的编辑联系我文章也发表在第四期上。当时Linux用户很少，文章奇缺。现在看起来自己写的两篇稿件是ABC之类扫盲文章。从此之后开始了和《开放系统世界》长达七年的合作，我和国晓平编辑合作一共写了文章147篇共80多万字。

后来又开始和网管员世界、IT168、IBM开发者频道、计算机世界、中国计算机报、51CTO、微电脑世界、天极网、IT专家网等十多家媒体合作，加起来也有200多篇文章100万字。

51CTO：撰写过这么多文章和书籍，您觉得对您自己收获最大的是什么？

曹江华：写过这么多文章和书籍，对自己收获最大是理论知识的学习。我的专业不是学计算机，刚开始就是使用Linux、Solaris等系统，并没有理论基础。写过文章和书籍过程中看了许多工具书，等于自学一遍计算机理论。当然，我所涉及的主要是类Unix操作系统。

51CTO：您也是经历过很多技术变迁的老系统管理员了，从Netware，IBM和HP的Unix，Sun的Solaris，到Windows NT和Linux，您都或多或少接触或了解过。您自己是如何看待并适应这种技术变迁的？有没有什么建议提给现在一些年轻的系统管理员们？

曹江华：其实我学习使用的主要是类Unix操作系统。要说建议，这里有一篇我发表在2005年9月的《网管员世界》的一篇文章，感觉可以当作给现在一些年轻的系统管理员们经验。

（接下篇）

推荐文章：

从网管员来说，命令行实际上就是规则，它总是有效的，同时也是灵活的。

系统学习 Linux 的 11 点建议

文/曹江华

一、从基础开始

常常有些朋友在 Linux 论坛问一些问题，不过，其中大多数的问题都是很基础的。例如为什么我使用一个命令的时候，系统告诉我找不到该目录，我要如何限制使用者的权限等问题，这些问题其实都不是很难的，只要了解了 Linux 的基础之后，应该就可以很轻易的解决掉这方面的问题。而有些朋友们常常一接触 Linux 就是希望构架网站，根本没有想到要先了解一下 Linux 的基础。这是相当困难的。

二、Linux 命令是必须学习的

虽然 Linux 桌面应用发展很快，但是命令在 Linux 中依然有很强的生命力。Linux 是一个命令行组成的操作系统，精髓在命令行，无论图形界面发展到什么水平这个原理是不会变的，Linux 命令有许多强大的功能：从简单的磁盘操作、文件存取、到进行复杂的多媒体图象和流媒体文件的制作。

不同版本的 Linux 命令数量不一样，这里笔者把它们中比较重要的和使用频率最多的命令，按照它们在系统中的作用分成几个部分介绍给大家，通过这些基础命令的学习我们可以进一步理解 Linux 系统：

安装和登录命令：

```
login, shutdown, halt, reboot, mount,
 umount, chsh
```

文件处理命令：

```
file, mkdir, grep, dd, find, mv , ls ,
 diff, cat, ln
```

系统管理相关命令：

```
 df, top, free, quota , at, lp, adduser,
 groupadd kill, crontab, tar, unzip,
 gunzip , last
```

网络操作命令：

```
ifconfig, ip , ping , netstat , telnet,
 ftp, route, rlogin rcp , finger , mail ,
 nslookup
```

系统安全相关命令：

```
passwd , su, umask , chgrp, chmod, chown,
 chattr, sudo, pswho
```

三、选择一本好的工具书

工具书对于学习者而言是相当重要的。一本错误观念的工具书却会让新手整个误入歧途。目前国内关于 Linux 的书籍有很多不过精品的不多，笔者强烈建议阅读影印本的“O'Reilly 原版 Linux 图书”，而且出版社还提供了一个非常好的路线图：

http://www.oreilly.com.cn/guide/guide_linux.php

四、选择一个适合你的 Linux 发行版本

目前全球有超过 1 百多个 Linux 发行版本，在国内也能找到十几个常见版本。如何选择请根据你的需求和能力，Redhat Linux 和 Debian Linux 是网络管理员的理想选择。

五、养成在命令行下工作的习惯

一定要养成在命令行下工作的习惯，要知道 X－window 只是运行在命令行模式下的一个应用程序。在命令行下学习虽然一开始进度较慢但是熟悉后，您未来的学习之路将是以指数增加的方式增长的。从网管员来说，命令行实际上就是规则，它总是有效的，同时也是灵活的即使是通过一条缓慢的调制解调器线路，它也能操纵几千公里以外地远程系统。

六、选择一个适合你的 Linux 社区

随着 Linux 应用的扩展，出现了不少 Linux 社区。其中有一些非常优秀的社区：

www.linuxforum.net、http://www.chinaunix.net/，但是这几个论坛往往是 Linux 高手的舞台，如果在探讨高级技巧的论坛张贴非常初级的问题经常会没有结果。

七、勤于实践

要增加自己 Linux 的技能，只有通过实践来实现了。所以，赶快找一部计算机，赶快安装一个 Linux 发行版本，然后进入精彩的 Linux 世界。对 Linux 命令熟悉后，可以开始搭建一个小的 Linux 网络，这是最好的实践方法。Linux 是网络的代名词，Linux 网络服务功能非常强大，不论是邮件服务器、Web 服务器、DNS 服务器等都非常完善。当然你不需搭建所有服务，可以慢慢来。

八、如何得到联机帮助

主流 Linux 发行版都自带非常详细的文档（包括手册页和 FAQ），从系统安装到系统安全，针对不同层次的人的详尽文档，仔细阅读文档后 40%问题都可在此解决。

查阅经典工具书和 Howto，特别是 Howto 是全球数以万计的 Linux、Unix 的经验总结非常有参考价值通常 40%的问题同样可以解决。

如果上面的措施没有解决问题，此时你就需要 Linux 社区的帮助了。Linux 的使用者一般都是专业人士，他们有着很好的电脑背景且愿意协助他人，Linux 高手更具有鼓励新手的文化精神。如何在 Linux 社区获得帮助，需要说明的是你要有周全的思考，准备好你的问题，不要草率的发问，否则只会得到到草率的回答或者根本得 不到任何答案。最好先搜寻一下论坛是否有您需要的文章。这样可以获得事半功倍的效果。

九．用 Unix 思维学习 Linux

Linux 是参照 Unix 思想设计的，理解掌握 Linux 必须按照 Unix 思维来进行。思想性的转变比暂时性的技术提高更有用，因为他能帮助你加快学习速度。

十．学习专业英文

如果你想深入学习 Linux，看不懂因为文档实在是太难了。写的最好的，最全面的文档都是英语写的，最先发布的技术信息也都是用英语写的。安装一个新的软件时先看 README，再看 INSTALL，然后看 FAQ，最后才动手安装，这样遇到问题就知道为什么。

十一．最后是 Linux 学习的路线图

掌握至少 50 个以上的常用命令。

熟悉 Gnome/KDE 等 X-windows 桌面环境操作 。

掌握.tgz、.rpm 等软件包的常用安装方法

学习添加外设，安装设备驱动程序（比如网卡）

熟悉 Grub/Lilo 引导器及简单的修复操作 。

熟悉 Linux 文件系统 和目录结构。

掌握 vi,gcc,gdb 等常用编辑器，编译器，调试器 。

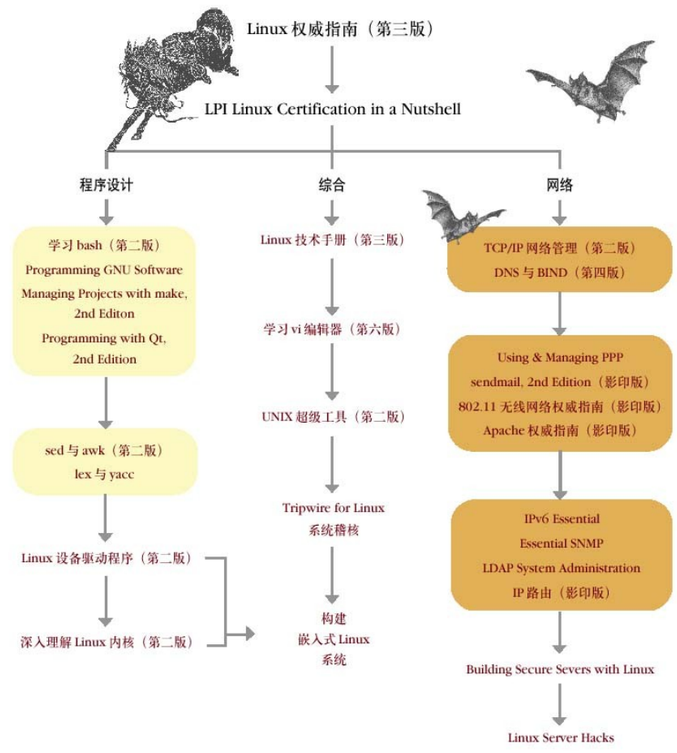
理解 shell 别名、管道、I/O 重定向、输入和输出以及 shell 脚本编程。

学习 Linux 环境下的组网。

访谈原文：

http://os.51cto.com/art/201012/236245.htm

《系统学习 Linux 的 11 点建议》一文最初刊于 2005 年 9 月的《网管员世界》。



Linux 学习图书路线图

首先必须明确，理论上检测另外一台机器是否宕机是无法做到的。

如何检测一台机器是否宕机？

文/日照

编者按：本文来自淘宝核心系统运维团队博客，文中描述的宕机检测方式适用于大规模集群环境，但原理仍是基于双节点有需求的朋友们可以借鉴。

检测一台机器是否宕机的应用场景如下：

1，工作机器宕机，总控节点需要能够检测到并且将原有服务迁移到集群中的其它节点。

2，总控节点宕机，总控节点的备份节点（一般称为 Slave）需要能够检测到并替换成主节点继续对外服务。

检测一台机器是否宕机必须是可靠的。在大规模集群中，机器可能出现各种异常，比如停电，磁盘故障，过于繁忙导致假死等。对于机器假死，如果总控节点认为机器宕机并将服务迁移到其它节点，假死的机器又认为自己还可以提供服务，则会出现多个节点服务同一份数据而导致数据不一致的情况。

首先必须明确，理论上检测另外一台机器是否宕机是无法做到的，有兴趣的同学可以参考 Fischer 的论文。可以简单理解如下：A 机器往 B 机器发送心跳包，如果 B 机器不发送响应，A 无法确定 B 机器是宕机了还是过于繁忙，由于 A 和 B 两台机器的时钟可能不同步，B 机器也无法确定多久没有收到 A 机器的心跳包可以认为必须停止服务。因此，A 机器没有办法确定 B 机器已经宕机或者采取措施强制 B 机器停止服务。

当然，工程实践中，由于机器之间会进行时钟同步，我们总是假设 A 和 B 两台机器的本地时钟相差不大，比如相差不超过 0.5 秒。这样，我们可以通过 Lease 机制进行宕机检测。Lease 机制就是带有超时时间的一种授权。假设总控节点需要检测工作节点是否宕机，总控节点可以给工作节点发放 Lease 授权，工作节点持有有效期内的 Lease 才允许提供服务，否则主动下线停止服务。工作节点的 Lease 快要到期的时候向总控节点重新申请 Lease（一般称为 renewLease），总控节点定时检测所有工作机的 Lease 授权是否合法，如果发现某台工作机 Lease 失效，可以将工作机上的服务迁移到集群中的其它机器，这时因为工作机发现自己 Lease 失效会主动停止服务。当然，这里需要注意，由于总控节点和工作机的时钟可能不一致且有网络延迟，总控节点上的 Lease 超时时间要长，也就是说，如果工作节点的 Lease 超时时间是 12 秒，总控节点可能需要 13 秒后才能确认工作节点已经停止了服务，从而避免数据不一致问题。

同构节点之间的选主也有一个宕机检测问题。比如总控节点宕机，备份节点需要能够检测并升级为主节点继续对外服务。Mysql 数据库经常采用 Heartbeat + DRBD + Mysql 的高可用性方案，据说能够达到 3 个 9 的高可用性，主节点和备节点维持 Heartbeat，当提供服务的主节点出现故障时，备节点的 Heartbeat 检测到主节点没有心跳（例如，Ping 不通主节点），备节点自动接管虚拟 IP，升级为主节点提供 Mysql 读写服务。

由于 Heartbeat 检测机器主节点宕机不可靠，这个方案存在众所周知的脑裂问题，即集群中可能同时存在多个主节点同时提供服务。解决这个问题本质上还是需要引入仲裁节点，比如 Heartbeat + DRBD 方案中引入 Fence 节点使出现问题的节点从集群中脱离，或者引入分布式锁服务，比如 Chubby 的开源实现 Zookeeper 服务。分布式锁服务实现主节点选举大致如下：主节点和备节点到 Chubby 中抢锁，抢到锁的节点在锁的有效期(Lease 期)

内提供服务，当主节点锁的 Lease 快要到期时，主节点申请延长锁的超时时间，正常情况下分布式锁服务总是优先满足主节点的请求，当主节点出现故障时，备节点能够抢到锁切换为主节点提供服务。

最后还有一个问题，假设总控节点通过 Lease 机制检测工作节点是否宕机，这种方案是可靠的，不过当总控节点宕机时，如果不采取任何措施，集群中的所有工作节点都将因为无法重新申请 Lease 而停止服务，这就是带有总控节点的设计固有的脆弱性，某个设计或者编码的错误都有可能造成严重的影响。解决这个问题一般会有一个叫做 Grace Period 的机制，工作节点 Lease 超时时将停止服务，但是工作节点并不一开始就重启或者下线，而是处于一种危险状态(称为 Jeopardy)，这种状态持续一个 Grace Period，比如 45 秒。如果在 Grace Period 内总控节点重启，工作节点和总控节点重新联系上从而可以切换为正常状态继续提供服务。

如果需要较好地理解宕机及选举相关的问题，可以阅读并思考 Paxos 相关的论文，比如 Paxos made simple, The Part-time Parliament, Paxos made live, Paxos made practical, Chubby 等。

原文：

http://rdc.taobao.com/blog/cs/?p=708

小阅读：谁在编写 Linux 内核？

根据 2010 年“谁在写 Linux”报告显示，提交给 2.6.35 内核的代码量比去年发布的 2.6.30 内核的代码量少 18%，过去一年来提交的内核代码减少的原因很多，包括阶段代码新的提交流程。

报告解释了从 2.6.28 内核开始的代码阶段树。最初的状态树启动了一个进程，将大量树外的代码合并到主 Linux 内核中。根据今年的报告显示，从 2.6.31 开发周期可以看出，因积压下来的大量代码需要处理，这一进程就放慢了。报告还指出，新的驱动程序经由阶段树源源不断进入主 Linux 内核。

虽然新代码的提交步伐和去年相比不在同一水平，但整体来看进展还是不错。报告指出，自 2.6.30 内核开始，每天有 9058 行代码添加到 Linux 内核，包括周末和节假日。

过去五年半以来，从 2005 年的 2.6.11 内核到 2010 年的 2.6.35 内核，平均每一小时就会有 4.02 个补丁应用到内核树上。

内核发布后，就进入维护期，根据需要更新补丁。有些版本的内核更新补丁会多一点，如 2.6.32 内核是过去五年来更新补丁最多的一个版本（报告指出 2.6.32 内核总共有 1793 个修复补丁）。McPherson 指出，Linux 2.6.32 内核经过长时间的补丁修复，现在正稳定更新中。

正如最近五个内核版本提交的代码数量下降一样，为 Linux 内核贡献代码的企业数量也在下降。2.6.30 内核已知的代码贡献企业是 245 家，但到 2.6.35 版本时，贡献代码的企业数量就下降到 184 了。

一些知名的 Linux 内核代码贡献企业也发生了细微的变化。

◆贡献最多的仍然要数 Red Hat，2.6.30 内核有 12%的代码变化都是 Red Hat 贡献的

◆英特尔次之，贡献了 7.8%

◆Novell 贡献了 5%

◆IBM 贡献了 4.8%

◆从 2.6.30 内核开始，诺基亚贡献了 2.3%的代码，德克萨斯仪器提交了 1.5%的代码，无线厂商 Atheros 添加了 1.4%的 Linux 内核代码。

◆相反，Ubuntu Linux 背后的 Canonical 在最近五个 Linux 内核版本中所做的贡献却很少。

原文：

http://os.51cto.com/art/201012/236379.htm

推荐阅读：

Linux 内核入门，包教会

http://os.51cto.com/art/201007/213583.htm

在 2010 年 11 月-12 月之间，发生了下面这些事……

八卦，趣闻与数字 2010.11 - 2010.12

收集整理/51CTO 系统频道

【Solaris】根据 Oracle Solaris Express 11 的许可协议，这个发行版不可以用于生产环境，只能用于开发和演示。

http://os.51cto.com/art/201011/233519.htm

【超级计算机】Linux 依然是超级计算机的绝对主力，对比 6 月份数据发现 Linux 总体上升 5 台，牢牢占有 82%的份额，其中 CentOS 上升 1 台，SLES 9 下降 1 台，SLES 10 下降 2 台，SUSE/openSUSE/SLES 系还在下降。

http://os.51cto.com/art/201011/233506.htm

【Ubuntu】Ubuntu 11.04 Alpha 1 中我们就见到了 Unity 界面首次在桌面版中的应用，你可以使用 Ubuntu Launcher 来打开 Pin 在该栏中的应用程序，并且可以在当前运行的应用程序间切换。

http://os.51cto.com/art/201012/236734.htm

【OpenSSH】开源也是要钱的。没有钱，项目的开发成本、服务器的费用等等都无法支撑。

http://os.51cto.com/art/201011/233550.htm

【LibreOffice】现在看来，要 OOo 与 LO 复合已无希望；在 LO 前进道路上充满着很多挑战，前景并不乐观。

http://os.51cto.com/art/201011/234193.htm

【Novell】2010 年 11 月 22 日，IDG 传出消息，Novell 已经在周一同意了 Attachmate 公司以 22 亿美元出价的收购，整个收购过程将在未来数月内完成。

http://os.51cto.com/art/201011/234526.htm

【Linux 内核】由于一个仅有 200 余行代码的补丁，未来的 Linux Kernel 2.6.38 或许会成为下一年度最受期待 Linux 内核版本。

http://os.51cto.com/art/201011/233692.htm

【开源的商业模式】麦克尼利最近表示甲骨文 CEO 拉里埃里森是一个伟大的资本家，但并不是一个“好的”资本家。卖掉 SUN 之后麦克尼利立场有了很大的转变，似乎不再看好开源的发展前景。

http://os.51cto.com/art/201011/233627.htm

【Linux 病毒】第一个 Linux 系统上的病毒可以追溯到 1996 年，而之后的 15 年间，一直有新的 Linux 病毒出现，尝试攻击互联网上的服务器们和用户们。

http://os.51cto.com/art/201011/235491.htm

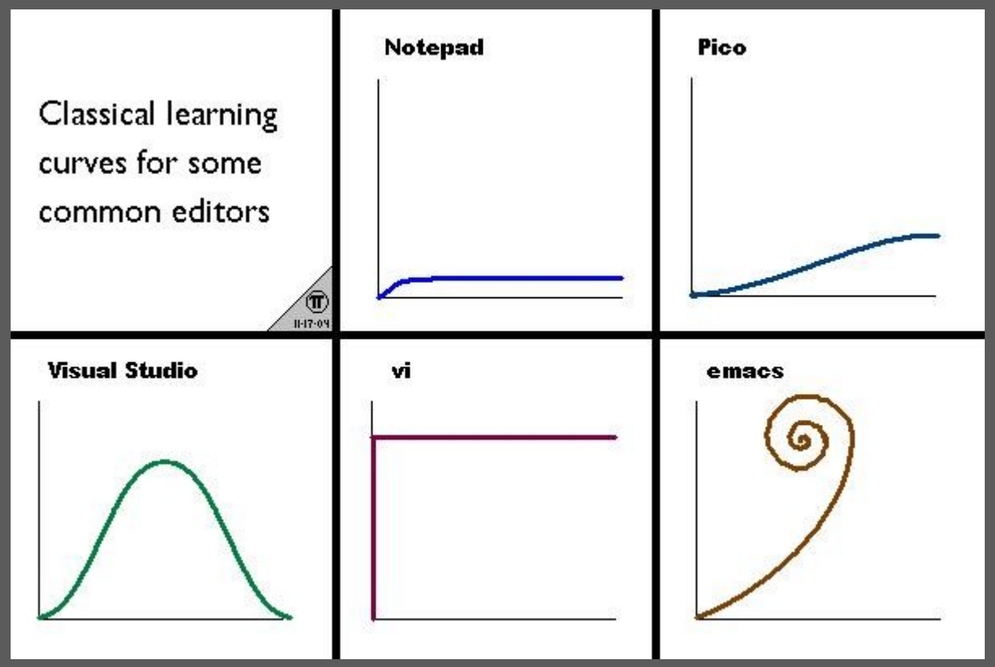
【RHEL 6】RHEL 6 的防火墙配置工具和 Firestater 相比不相上下，但是比之另外一个专业工具 Firewall Builder 功能上还有不足之处。

http://os.51cto.com/art/201011/235916.htm

【Solaris 11】Solaris 11 将在几秒钟立即启动，而不是把启动时间控制在分钟之内它还将采用无风险的更新模式，广泛的解决故障问题，还可以迅速重启失败的应用和服务项目。

http://os.51cto.com/art/201012/236785.htm

本期专题：运维与开发



运维与开发在 Linux 下使用的工具有相同的，也有不同的。比如 Terminal 往往是双方都是用的，而他们是用的文本编辑器则往往不同。比如，开发者常用 emacs 和 Eclipse，而运维则往往使用 Vi 或 Vim 即可。

上图是一个搞笑漫画，展示了常见编辑器的学习曲线。纵轴代表一个编辑器的使用熟练程度横轴代表使用该编辑器的时间长度。曲线坡度越陡的，就是该编辑器越难学习的意思。

图片来源：http://coolshell.cn/articles/3125.html

调查报告：运维与开发的常用工具

在 Linux 平台上，运维与开发人员往往是密切相关的，尤其是对于 Web 服务提供方而言更是如此。为了提高产品从开发到生产环境部署这个环节的效率，近年来有不少运维界的高手们开始提倡“敏捷运维”、“DevOps”等概念，主张运维与开发者之间更加紧密的在一起工作。

为了更方便的了解运维和开发在 Linux 使用习惯上的异同，51CTO 系统频道近日做了一些小调查。此次调查主要针对开发者与系统运维，收集到有效回复 50 余份，下面选取较有代表性的 30 份数据进行分析，覆盖人群包括 Web 开发者、移动开发者、网络游戏开发/运维、系统运维、网络工程师。

覆盖问题包括：

1、常用的 Linux 发行版

2、常用的编辑器

3、常用的语言

4、常用的浏览器

这份调查报告限于受访者范围，并不具备代表性，目的仅在于反映在职开发者和运维的技术关注点。

（下接第 12 页）

Python也是一个非常值得推荐的利器。这种语言具有非常简捷而清晰的语法特点，适合完成各种高层任务，几乎可以在所有的操作系统中运行。

运维人员应该掌握哪些常用技术

文/王文文

本着自己几年运维的工作经验和几个大型网站工程师的不吝赐教。这里将个人总结的一些体会发出来给大家参谋参谋（注不包含怎么做人）。如果看完本文您有任何问题或意见，欢迎和我交流。我的blog是：

http://verdureorange.blog.51cto.com

下面将运维中需要学习或能让你运维工作加分的技能稍做介绍。首先我们假设你是一个公司的IT运维经理。需要搞定五十台以上的服务器和若干杂七杂八的设备，哪些技术你会用的上？

一、微软系统

对于Windows的熟悉是最基本的。当然，作为一个运维经理，可不是整天玩个Windows 7或XP就可以交差的。你得掌握微软Active Directory及其上层各种服务和应用的搭建。一般常用的有ISA、Exchange、SQL Server。随着Windows 2008的大放异彩，Hyper-V又成了微软工程师不得不掌握的重型武器。

二、Linux/BSD系统

虽然Ubuntu现在很火，但是在公司里使用的大多还都是Redhat系列和Suse系列。你得熟悉DNS、NIS、Apache、SMB、DHCP、Sendmail、FTP、MySQL这些常规服务。如果公司的IT业务大规模对外，你还得学会LVS或Nginx等负载均衡技术。

友情提示：如果你将去人人网或豆瓣等新锐Web2.0公司，那你还得熟悉Cassandra之类混合型的非关系的数据库技术；Memcache之类高性能分布式的内存对象缓存系统（它通过在内存里维护一个统一的巨大的Hash表存储各种格式的数据）。

既然说了Linux，这里也顺带要提一下BSD，同样是开源的宠儿，BSD的安全性和高效让人印象深刻，目前包括Yahoo、Sina在内的很多公司都会用它来跑应用。这类系统熟悉之后，以后从事Solaris相关工作也会减少难度（同样的Unix血统）。

如果有中间件要求的，可以适当接触Weblogic（Oracle系）或WebSphere（IBM系）、Jboss（红帽系）。这些东西在目前流行的大型应用中非常广泛。

三、编程开发

混Windows系统的自觉一点学好Powershell吧。要是说前几年还得看看VBscript的话，未来就都是Powershell的天下了。

PowerShell是微软公司于2006年第四季度正式发布的。它的出现让Windows在运维方面拉近了与Unix，Linux等操作系统的距离。目前支持.Net Framework 2.0。能够运行在Windows XP SP2之后各种操作系统上。能够同时支持WMI，COM，ADO.NET，ADSI等已有的Windows管理模型。这项全新的技术提供了丰富的控制与自动化的系统管理能力；而“脚本语言”（scripting languages）则是用来编写程序的计算机语言。脚本语言通常都有简单、易学、易用的特性，目的就是希望能让写程序的人（开发者）快速完成程序的编写工作。

关于Linux平台下的运维人员，还是推荐Shell加Perl的组合，毕竟那么多年过来了

这个组合一直在为各大网站的工程师们稳定的工作着。另外，Python 也是一个非常值得推荐利器。这种语言具有非常简捷而清晰的语法特点，适合完成各种高层任务，几乎可以在所有的操作系统中运行。目前，基于这种语言的相关技术正在飞速的发展，用户数量急剧扩大相关的资源非常多。

四、网络设备

熟悉网络基础知识、网络通信协议和常见的网络设备是必须的。建议把思科和华为这两家的产品摸熟。

友情提示：很多人会把华为和 H3C 当成一家，其实他们两家设备的命令并不怎么兼容。

如果是在稍微大一点的公司工作或者哥们你就是在 IDC 混，那还得学会对企业局域网和广域网进行规划、实现和检查排错，VLAN 是必须的。其他就是视频、语音之类的网络服务了。

基础的都搞明白了之后还想提高自己朋友可以去学一下 CCNP 或者 H3CNE 的课程。

五、存储

这里顺带着提一下存储，给新手做一个概念上的介绍。当然，这类东西在数据量大的企业中也是经常用到的。

先说 SAN。SAN 是专门用于提供企业商务数据或运营商数据的存储和备份管理的网络。因为是基于网络化的存储，SAN 比传统的存储技术拥有更大的容量和更强的性能。

通过专门的存储管理软件，可以直接在 SAN 里的大型主机、服务器或其他服务端电脑上添加硬盘和磁带设备（现在大多数的 SAN 是基于光纤信道交换机和集线器的。相当于一个高速的子网，通常 SAN 由 RAID 阵列连接光纤通道组成，SAN 和服务器和客户机的数据通信通过 SCSI 命令而非 TCP/IP，数据处理是“块级”。

NAS 则以数据为中心，将存储设备与服务器彻底分离，集中管理数据，从而释放带宽、提高性能、降低总拥有成本、保护投资。其成本远远低于使用服务器存储，而效率却远远高于后者。这类设备相对来说还算简单。

SAN 和 NAS 的应用都非常广泛，现成的解决方案也有很多，它们可以混用，也可以单独使用，主要还是根据自己公司的实际情况来定。

原文：

http://netsecurity.51cto.com/art/201007/209096.htm

推荐阅读：

门户网站运维经验谈

http://os.51cto.com/art/201008/219699.htm

SA，神仙与装机男：运维的工作到底啥样儿？

http://os.51cto.com/art/201007/214401.htm

运维漫谈：半身半仙亦民工

http://os.51cto.com/art/201008/219611.htm

十大 x86 服务器常见故障：系统篇

http://os.51cto.com/art/201008/220004.htm

调查报告：常用的 Linux 发行版

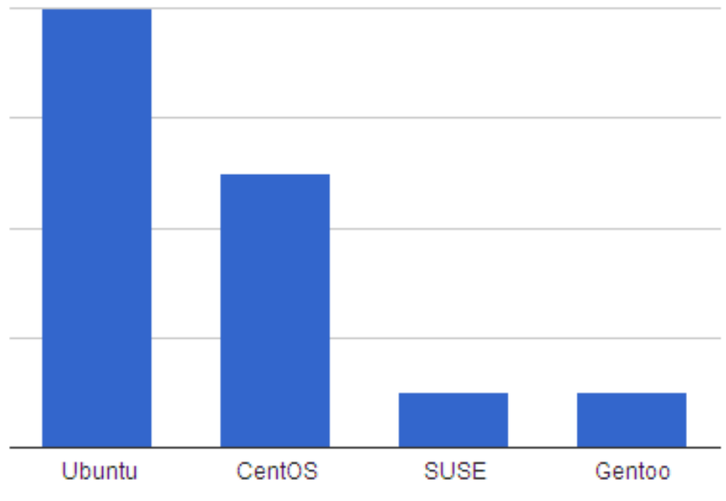


图 1 开发者常用的 Linux 发行版

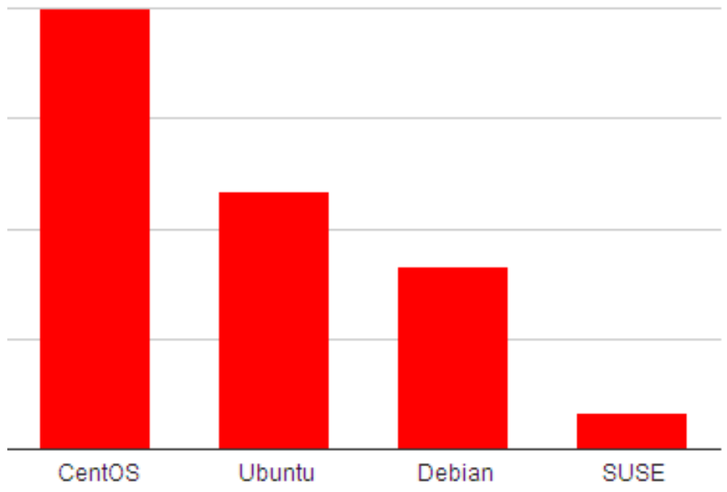


图 2 运维常用的 Linux 发行版

（下接第 14 页）

系统管理员通常被认为是生产环境的拥有者。他们是持续跟踪机器运行状态的人，是凌晨2点会被电话叫醒的人，基本上也是和生产环境问题距离最近的人。

是否应该允许开发人员进入生产环境？

文/外刊IT评论

在Web开发公司里，有个问题会被一而再、再而三的提出来，这就是：

“是否应该允许开发人员进入生产环境？如果是，允许到什么程度？”

对于此，我的观点是，整体上，应该限制开发人员访问生产环境。在论证我这个观点之前我想声明一下，我的立场绝对不是基于对开发人员的品质品行的判断——所以请不要往这方面想。首先我要公布一些常见的程序员们不喜欢或讨厌这个观点的论据：

“系统管理员什么都不让我们干，他又解决不了，我们根本没法完成任务。”

如果真是这种情况，那么他们应该是对的。如果是因为没有足够的管理员或管理员不称职那么瓶颈就会出现。然而，允许开发人员访问生产环境并不是一种解决方案，因为即使这一回你把问题解决了，以后还是会出现缺少管理员的情况。有时候一些管理上的规章制度会使事情变的繁琐，耽误时间，但我想这时间也不会是一种不可接受的漫长。

“我们以前就是这么干的。”

创业公司很少一成立就有一些系统管理员。出于某些原因，系统管理员会被认为是一种奢侈品。尽管这种状况在以前是可以的，但随着公司的成长，管理员应 该慢慢增多。事情会变的越来越复杂，这也是公司为什么需要系统管理员的原因。所以说，“我们以前就这么干”的说法并没有多少说服力。

“我们需要进入生产环境解决问题。”

也许是，也许不是。管理员应该能够给你想要的各种信息。如果这个过程成为瓶颈，那么允许有限范围内的访问是合适的。

还有很多的关于限制开发人员访问的讨论，但来让我们把目标转向我真正想讨论的——为什么这是一个好建议。

限制访问的产生由来：

如果开发人员不能够访问生产环境，一个最大的含义就是他们不能自己去安装程序。也就意味这管理员要去为他们安装程序。这其中会发生两件事情：

1）开发人员和系统管理员必须交流——相互的交流！管理员必须学会如何安装程序（我可不希望是由我去解释），这应该是件好事。

2）开发人员必须制作安装文件和编写简单有效的部署步骤说明。这也是很好的事情。能按步骤来重建系统是应对灾难恢复的重要的一部分。

所以，一旦开发人员被限制访问生产环境，我们就能避免那种因为程序过于复杂以致于只有开发人员出面才能部署安装的情况出现。同样，开发人员也避免了把时间浪费在部署和安装程序的工作上了，节省下的时间开发新程序岂不更好。虽然起初他们会多花一点时间，但以后会越来越方便。

这样管理员也能学到更多的关于安装过程中需要备份哪些东西的知识了。即使在管理员对程序不太了解的情况，他们只需对着开发人员提供的文档来执行他们的备份操作。

开发人员所关心的并不一定是系统管理员所关心的：

通常开发人员所关注的安全领域并不一定会是系统管理员所关注的领域。对于Web网站安全，他们只是擅长他们自己特有的领域。诸如

跨站脚本攻击，SQL 注入等安全问题是开发人员擅长的，对于系统管理员则不然。诸如账户权限，文件权限，web 服务器配置通常不是开发人员擅长的和感兴趣的，而这些对于生产环境确十分重要的，它们是系统管理员擅长的领域。我对这个领域的见解是越少人访问越安全还有，这能有效的避免凌晨两点被电话吵醒——因为系统管理员怀疑你们 15 个程序员中的某个人在服务器上做了不当的操作。

变更控制：

我不认为一个稳重的程序员会不把变更控制当作重要的事情。因为在程序中经常会有这样的事情。然而，我发现许多程序员却不会慎重的把在服务器上的每次改动都做上日志。（当然我也看到过一些配置文件是有版本控制的）

如果你没有这样做，这意味这生产环境不可能正确的重建。这也意味着如果某些改动导致了问题，对于来解决问题的人却未必能知道这些曾经发生过的改动。这就好像是一个系统管理员打开产品代码，在未通知任何人的情况下修改了程序或提交到产品里。哇，我估计程序员们会抓狂的。

拥有者对其有控制权：

Joel’s Spolsky 有句话放在管理工作上很合适：

“每人都有自己的一块领地。是谁的，就是谁的。如果一个管理者或其他人，想插手一个事情的管理方式，他必须保证自己是事情拥有者。拥有者有最终话语权。”

系统管理员通常被认为是生产环境的拥有者。他们是持续跟踪机器运行状态的人，是凌晨 2 点会被电话叫醒的人，基本上也是和生产环境问题距离最近的人。如果开发人员直接访问的生产环境，那这种管理控制无形中就给破坏了。

系统管理员的职责：

为了能在这种原则下正常的工作，管理员必须完成一些工作。

1）向开发人员们询问他们想从你这得到什么信息，你要很乐意的给他们想要的。

2）确保开发人员在他们自己的空间里有一个好的开发环境。

3）理论结合实践。每个公司都有自己的特殊情况，有些公司由于自身业务的原因决定了开发人员的无访问权限（比如金融类）。然而即使你那不是个金融 公司，取消开发人员的访问特权的作业制度也是最好的方案。有可能某些开发人员同时担任这系统管理员的工作，所以每个公司都有自己的情况。

原文：

http://blog.serverfault.com/post/893001713/

译文：

http://www.aqee.net/2010/08/16/should-developers-have-access-to-production/

调查报告：常用的编辑器

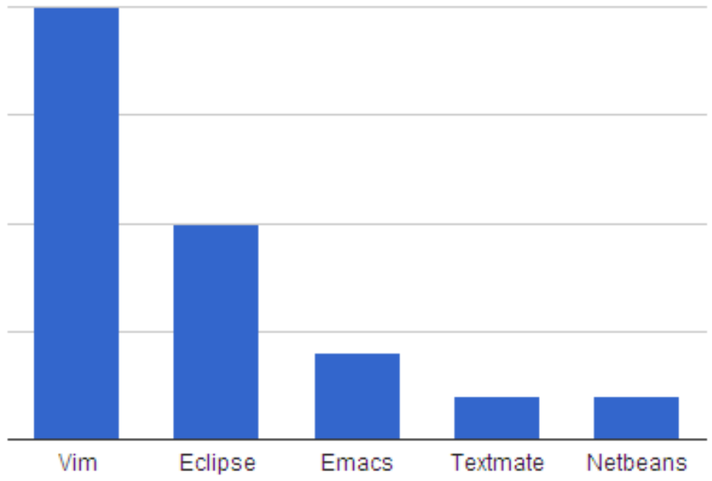


图 3 开发者常用的编辑器

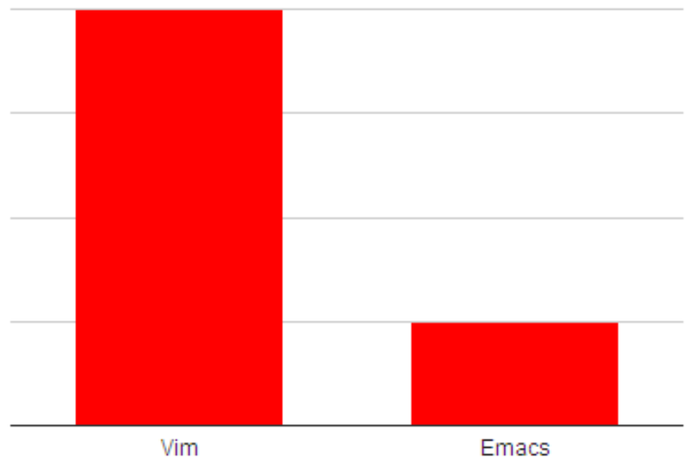


图 4 运维常用的编辑器

（下接第 17 页）

每条 JOB 执行完毕之后，系统会自动将输出发送邮件给当前系统用户。日积月累，非常的多，甚至会撑爆整个系统。

Linux 计划任务：听酒哥分享 Crontab 使用心得

文/抚琴煮酒

你用过 Crontab 吗？你对 Crontab 了解吗？你知道如何以秒为级别执行任务吗？你知道怎样让 Crontab 执行 PHP 脚本吗？你知道 Crontab 的一些特别的用途吗？在本文中，抚琴煮酒将向大家介绍自己多年来使用 Crontab 的一些心得。

服务器系统：64Bit CentOS 5.5

虽然关于 Crontab 的介绍到处都是，详细读了一遍这个词条，收获还是有的。Crontab 这个名字来自“chronos”，一个古希腊语，“时间”的意思（以下用法在生产环境下的服务器非常有用，抚琴煮酒强烈推荐）；关于 crontab 的基础用法和语法我就不推荐了，google 和 51cto 上到处都是，我说的是一些进阶技巧，里面也有可能是不知道的；Shell 脚本均取自于线上服务器。我安装的服务器一般都是采用最小化安装，安装以后就发现启动 cron 的管理服务 crond 默认就是启动的，

```
service crond status
crond (pid 3444) is running...
```

手快的小伙可用 chkconfig crond on 让其在 level3 和 level5 自动运行。

cron 把命令行保存在 crontab（crontable）文件里，这个文件通常在 /etc 目录下。每个系统用户都可以有自己的 crontab（在 /var/spool/cron/ 下）。要查看当前用户的 crontab，输入

```
crontab -l
```

要编辑 crontab，输入

```
crontab -e
```

要删除 crontab，输入

```
crontab -r
```

如当前是 root 身份，要查看/编辑/删除/某用户的 crontab，只需在相应的命令后加上 -u USERNAME（如 crontab -e -u USERNAME）即可。crontab 文件的默认编辑器是 vi，可以输入

```
export VISUAL='editor'
```

更改默认编辑器。

cron 服务每分钟不仅要读一次

```
/var/spool/cron
```

目录内的所有文件，还需要读一次

```
/etc/crontab
```

文件。配置这个文件也能让 cron 执行任务。使用 crontab 命令是对用户级任务的配置，而编辑 /etc/crontab 文件是对系统级任务的配置。

抚琴煮酒定义的 crontab 语法比较好记，推荐记忆学习：

```
分 时 日 月 星期 用户 带绝对路径的命令 脚本或 PHP 详细路径
```

细心的你发现没有，crontab 最小执行时间为分钟，如果要求任务是以秒为级别的怎么办，其实是有办法的，我等会以实例说明下；服务器我建议也以/etc/crontab 为主，它更方便控制用户管理；建议服务器上的计划任务时间错开，不要同时并行任务，不然会在某时间段造成系统负载过大，搞得你的 Nagios 狂发报警邮件。

图 Crontab 这个名字来自古希腊语 chronos，取“时间”之意

玩 windows 应该也会发现，windows2K 系列的每修改一次系统管理员 Administrator 的密码就是一次浩大的工程，windows 的每个计划任务 scheduler 都要更改密码；CentOS 下修改 root 密码就简单多了，一条 passwd 命令就解决问题了，而且不影响 crontab 的执行。

每条 JOB 执行完毕之后，系统会自动将输出发送邮件给当前系统用户。日积月累，非常的多，甚至会撑爆整个系统。所以每条 JOB 命令后面进行重定向处理是非常必要的：>>/dev/null 2>&1 。前提是对 Job 中的命令需要正常输出已经作了一定的处理，比如追加到某个特定日志文件；>> /dev/null 2>&1 表示把所有标准输出发送到 /dev/null（linux 的回收站），把标准错误输出（2）发送到和标准输出（1）同样的地方（即 /dev/null）。运行这行命令将不会产生任何输出。举例说明如下：

```
30 15 13 6 1 * root tar czf
 /usr/local/backups/daily/etc.tar.gz /etc
 >> /dev/null 2>&1
```

CentOS/FreeBSD 下用 crontab 定时执行 PHP 程序的正确方法如下：

1、使用 crontab -e 编辑定时任务

内容为：

xx:xx:xx 执行一个 test.php 文件

2、php 文件必须在文件头一行，加上解释器路径（就象 perl 做的那样）

```
#!/usr/local/bin/php
```

PHP 的执行需要 Apache 的支持，shell 脚本的执行需要 Linux 的支持，而 Linux 支持定时运行某个程序的功能。

要将 PHP 作为 Shell 脚本语言使用，我以前的方法就是直接在/etc/crontab 里直接带上 php 路径，如

```
*/5 * * * * root php test.php
```

也能正确执行；有兴趣的朋友可写一个 hello 程序测试，将其所有输出到一个文件即可测试，我做了大量测试证明其可执行性；当然你的 php 要保证其正确性，建议你的每一个 crontab 计划任务先在 shell 中先正确运行一次后再写进 crontab 里，这是保证成功的必要条件。

crontab 如何以秒为执行你的 Linux 计划任务呢，许多同事和同学都问过我这个问题。其实这个问题很简单，我以实例说明下：

我以前的办公室是一台 ubuntu 服务器，连接 ADSL 作 NAT 带办公室的机器上网，很不幸的发现，这样 ADSL 爱掉线，一掉线网关 gateway 就没了，所以写了一个 shell 脚本：

```
#!/bin/bash
while :
do
if route | tail -1 | grep "0.0.0.0"
then
&>/dev/null
else
adsl-stop
adsl-start
fi
sleep 10
done
```

将程序放进后台执行。

执行脚本方法 nuhup sh route.sh &，注意前面要用上 nohup，这样避免 root 用户 logout 时此脚本也退出生效的问题。

程序的运行间隔时间是 10s，很多同学不明解为什么要加一个 sleep 10；大家可以拿虚拟机测试下，如果没有这条 sleep 10 的语句，你的 Linux 服务器会运行大量此 route.sh，很快耗光你的服务器资源，你的系统负载会很

快上去，Nagios 马上会叫起来的~

另一个就是监控负载均衡器 Nginx 的 shell 脚本，用于 Nginx+Keepalived 负载均衡高可用环境：

```
#!/bin/bash
while  :
do
nginxpid=`ps -C nginx --no-header | wc
 -l`
if [ $nginxpid -eq 0 ];then
/usr/local/nginx/sbin/nginx
sleep 5
if [ $nginxpid -eq 0 ];then
/etc/init.d/keepalived stop
fi
fi
sleep 5
done
```

另外，这里附带说下 crontab 的一个妙用：

我在配置一台服务器的 iptables 时，不小心设置了某一项错误参数，结果锁定了 SSH 会话，导致我们经理及另一系统管理员员连不上服务器，郁闷之余，看到此法特推荐给大家，极其有用，建议大家学习参考：可以配置一计划任务 crontab,每 5 分钟运行一次，即

```
*/5 * * * * root /bin/sh
 /root/firestop.sh
```

firestop.sh 内容为

```
#/bin/bash
service iptables stop
```

这样即使你的脚本存在错误设置（或丢失的）规则时，也不至于将你锁在计算机外而无法返回与计算机的连接，让你放心大胆的调试你的脚本。

最后说明下 Crontab 任务的执行环境问题。

尽管可以更改执行 cron 任务時使用的环境，但经常最好创建一個包装脚本，以在运行实际需要的命令前定义任何环境境变量（如 PATH）。

这样做的部分原因是出于安全考虑；向 crontab 作业开放的区域越多，越可能得到包含可疑内容的東西。另一个原因是，这样可确保即使更改了环境中的一个依赖关系，你的 crontab 作仍将执行。

掌握以上 crontab 的用法后，我相信大家用 Linux 更可以得心应手，工作更加 happy 和轻松了，希望这篇文章能给大家的运维工作带来帮助！

原文：

http://os.51cto.com/art/201011/233361.htm

推荐阅读：

Linux 计划任务——cron 服务入门与应用教程

http://os.51cto.com/art/201011/233945.htm

利用 crontab 系统每天定时备份 MySQL 数据库

http://database.51cto.com/art/200510/8433.htm

cron 介绍与安装配置笔记

http://iminmin.blog.51cto.com/689308/421180

调查报告：常用的编程语言

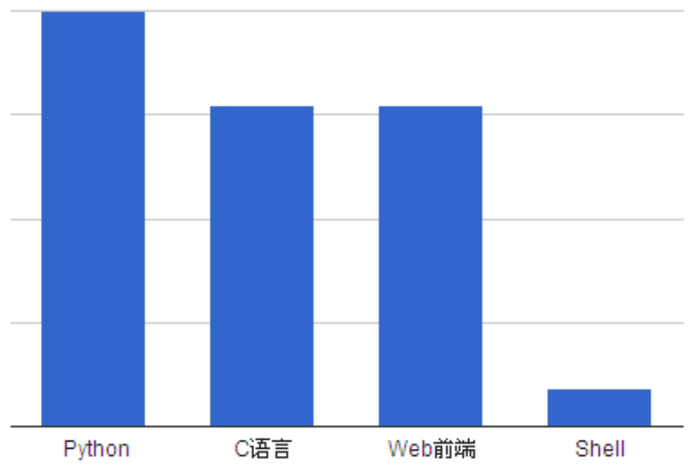


图 5 开发者在 Linux 下常用的语言

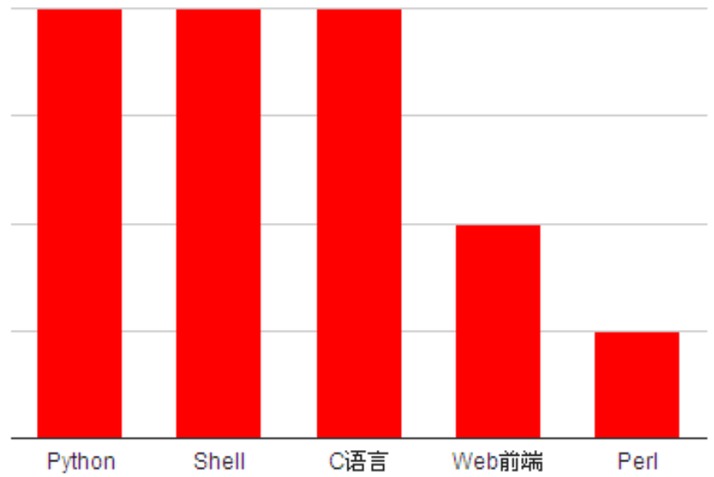


图 6 运维在 Linux 下常用的语言

（下接 20 页）

优秀管理员用 shell 脚本完成绝大多数工作，所以他们有足够的时间喝咖啡泡论坛。

Shell 学习笔记——总括篇

文/licong

Shell 脚本编程是 Unix/Linux 系统管理员应当具备的一项非常重要的技能，优秀管理员用 shell 脚本完成绝大多数工作，所以他们有足够的时间喝咖啡泡论坛。然而，要掌握这一技能并不十分容易，这需要了解相当数量的知识，并进行大量的练习和实践。笔者丝毫不敢说自己已经掌握了这一技能，我写这些文章只是为了把学习的历程记录下来，如果恰好也对你有所帮助我很荣幸。

首先，我们来解决必须回答的问题：shell 是什么？要回答这个问题必须先清楚 Unix/Linux 系统的结构。我们把这个结构简单分为两部分：系统内核——实用程序。系统内核是系统的心脏，从打开计算机自检时就驻留在计算机内存，直至计算机关闭；而实用程序驻留在计算机磁盘上，仅当需要时才调入内存。

那么 shell 是什么呢？shell 是一种实用程序，实际上所有 Unix/Linux 命令都是一个实用程序！每当系统允许用户登陆时，系统（准确地讲是 init 程序）为每个终端启动 getty，getty 做一些事情然后在分配给他的终端上显示”login:”等待用户输入信息。一旦用户键入信息并以回车结束，getty 程序就会消失，同时启动 login 程序完成登陆处理。用户成功登陆后，将会启动一个重量级的程序那就是 shell！

Shell 为什么那么重要呢？因为他有很多很强大的功能：

一、执行程序：shell 负责解释并执行终端请求的程序。

二、变量和文件名的替换：shell 会解释一些特殊的符号来进行替换。

三、I/O 重定向：将输入和输出重定向到别的地方（不再是标准的键盘和屏幕）。

四、管道线连接：shell 可以把多个命令连接在一起，就像管道一样。

五、环境控制：shell 控制着用户的环境，你可以根据需要通过 shell 来改变环境。

六、解释型程序设计语言：可以类似于 C（尽管没那么强大）那样编写复杂的程序。

下面我们的详细内容都是围绕上面六大功能展开的，在这之前先说明一下 shell 脚本又是什么？把 shell 能够解释的命令一条接一条写到文件里，加入上面提到的一些功能，并给该文件可以执行的权限，那么一个 shell 脚本就诞生了。换句话说，当你掌握了这些功能也就基本上掌握了编写 shell 脚本的技能了！

——程序执行

shell 负责解释并执行终端请求的程序，这里的程序大多数情况都是指 Unix/Linux 命令。Shell 所涉及到的每条命令都遵循相同的基本格式：

程序名　参数表

Shell 扫描命令行，并判断要执行的程序名字以及要传给程序什么参数（这里我们认为选项是参数的一部分）。

Shell 用特定的字符去判断程序名的起止位置。这些字符被称为空白字符，有空格、TAB 和换行符（回车），shell 会忽略掉多出的空白字符。键入命令：mv file temp 时 shell 扫描命令行，并提取从该行开始到第一个空白字符为止的字符串作为将执行的程序名：mv；接下去，到下一个空白字符之前的字符序列作

为传给 mv 的第一个参数：file；再接着到下一个空白字符（这里是换行符之前的字符序列是传给命令 mv 的第二个参数：temp。分析完命令之后，shell 就会执行 mv 命令，同时传给它两个参数 file 和 temp。后面的任务就交给 mv 去完成了。注意，Unix/Linux 系统绝大多数情况是严格区分大小写的！

前面提过，多个空白字符将被 shell 忽略。这就是说，当 shell 处理命令行：

```
echo whendowe  eat?
```

时，该命令行给 echo 程序 4 个参数：when、do、we 和 eat？

因为 echo 得到参数后，只是将他们显示在终端上，并用一个空格字符分隔各个参数，因此下面的输出就很容易理解了：

```
$echo whendowe  eat?
When do we eat?
$
```

事实上，echo 命令根本看不见那些空白字符：他们已被 shell“吞”掉了！

——变量和文件名的替换

与其他程序设计语言一样，shell 允许你给变量赋值。任何时候，在命令行中指定一个变量，并且在其前面冠以美元符号，shell 都会在这个位置用已赋给该变量的值来替换。我们后面还会详细讨论这个主题。

Shell 也在命令行执行文件名替换。实际上，在判定要执行的程序名及其参数之前，它先扫描命令行，寻找文件名替换字符*，？，或[…]。假如当前目录包含如下文件：

```
$ls
file1
file2
file3
$
```

现在对 echo 命令做文件名替换：

```
$echo *
file1 file2 file3
$
```

有多少个参数传给了 echo 程序，一个还是 4 个？因为我们知道 shell 会执行文件名替换，所以答案是 4 个。当 shell 分析命令行：

```
echo *
```

时，它识别出特殊字符 *，并在命令行中将*替换为当前工作目录下的所有文件：

```
echo file1 file2 file3
```

然后 shell 确定传给命令的参数。因此 echo 是看不见*的，当轮到它来处理时，看到的是命令行中键入 4 个参数。

——I/O 重定向

在命令行中执行输入输出重定向也是 shell 的职责。它扫描命令行，看是否有特殊字符<、>和>>出现（还有字符<<，将在后面讨论）。

键入命令：

```
echo Remember to order Law >reminder
```

时，shell 识别出特殊的重定向字符>，并提取命令行中的下一个词作为重定向输出结果的文件名，在上例中是 reminder。如果 reminder 已经存在，而你也有对它的写入权限，原来的内容就丢失了（如果没有写入权限 shell 会提示一条错误信息）。

在 shell 开始程序的执行之前，它就将程序的标准输出重定向到指定文件。就程序而言它根本不知道自己的输出被重定向了，它只是按照自己的方式将输出写入标准输出（通常是屏幕），并没有意识到 shell 已经将输出重定向到一个文件。

让我们来看两条几乎一样的命令：

```
$wc -l users
5 users
$wc -l < users
5
```

第一种情况下，shell 分析命令行，并判断要执行的程序名是 wc 以及要传给它的两个参数：-l 和 users。当 wc 开始执行时，它被传给了两个参数。第一个参数是-l，表示它将对行进行计数。第二个参数指明了要进行计数的文件名。所以，wc 打开文件 users，统计行数，并在终端上打印出统计后的行数和相应文件名。

第二种情况下，wc 的操作稍微有一点不同。扫描命令行时，shell 识别出输入重定向字符<，紧跟着的那个词表示重定向数据院文件的名字。Shell 从命令行“吞”掉<users，启动 wc 程序的执行，将他的标准输入重定向到文件 users，并传递给它一个参数-1。这时

wc 开始执行，它发现只传给了自己一个参数-l。因为没有指定文件名，wc 将此作为统计标准输入上出现的行数的一种指示。因此，wc 在计数标准输入上的行数时，并未意识到，它实际上是在为文件 users 中的行进行计数。最终的结果显示在终端上——没有文件名，因为没有给过 wc 文件名。

——管道线连接

就像在命令行中搜索重定向字符一样，shell 也在搜索管道字符 | 。对每个它发现的管道字符，它将位于 | 前面的命令的标准输出连接到位于 | 后面的命令的标准输入。然后为这两个程序执行初始化（判断程序及其参数）。

因此在键入

```
$who | wc -l
```

时，shell 发现，管道字符隔开了命令 who 和 wc。它将前面命令的标准输出连接到后面命令的标准输入，然后为执行这两个程序做初始化。当执行 who 时，它产生一个登录人员的列表，并将结果写到标准输出：但它没有意识到结果将不会送到终端，而是送到另一个命令。

当 wc 命令执行时，它识别出没有指定文件名，并在标准输入上统计行数。没有意识到标准输入不是来自于终端，而是来自于 who 命令的输出。

——环境控制

Shell 提供某些命令用于定制环境。环境包括用户的宿主目录、shell 用于提醒用户键入命令的提示符号、每当请求运行某个程序而进行搜索的目录列表等等。详细的内容留到后续专门的章节继续讨论。

——解释型程序设计语言

Shell 有着自己的内部程序语言。它和 C 等其他高级程序设计语言最大的区别在于，这种语言是解释型的，他只是分析每条语句并执行而不像其他语言那样需要经过编译才能执行。解释型的 shell 比编译型的程序更易于调试和修改，但他们的运行比等价的编译程序更耗时。

与大部分程序设计语言类似，shell 也具有 if、case、for、while、until 等判断、分支、循环结构，以及变量、数组和函数。这部分内容是脚本编程的重点和难点之一，后续也会有专门的章节来进行讨论。

好了，shell 总括就到这里了。总结一下，就是我们上面提到的六大功能。每一个功能都值得我们深入挖掘！事实上，你把这些功能挖掘得越透彻，你也就越容易写出更高质量的脚本。

原文：

http://licong.blog.51cto.com/542131/205615

调查报告：常用的浏览器

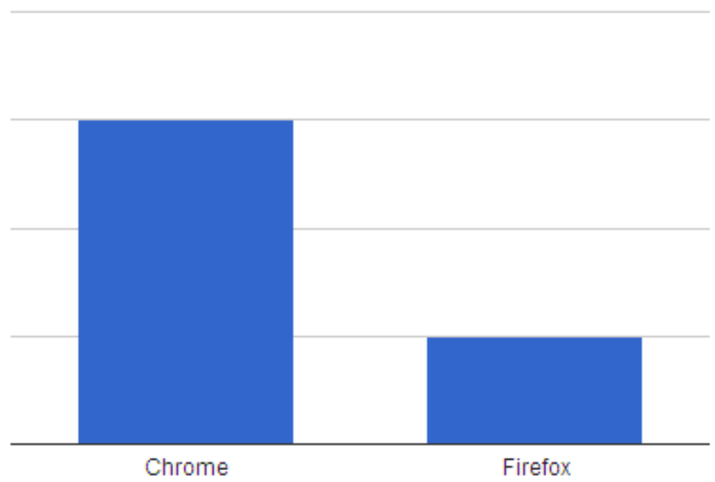


图 7 开发者在 Linux 上常用的浏览器

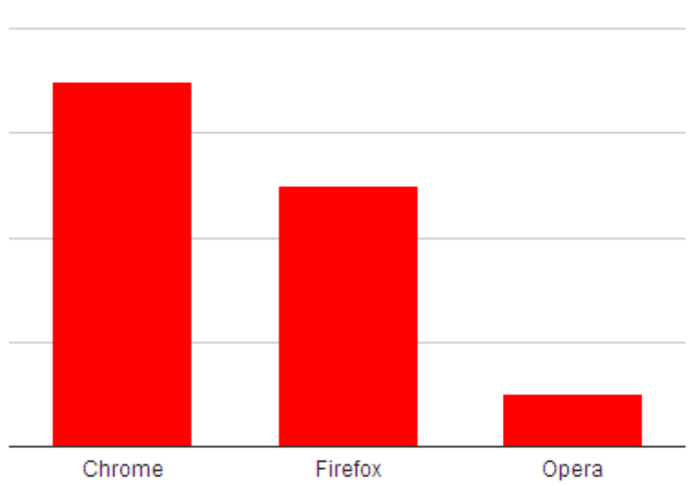


图 8 运维在 Linux 上常用的浏览器

（完）

SSH是一个非常伟大的工具，如果你要在互联网上远程连接到服务器，那么SSH无疑是最佳的候选。

25个必须记住的SSH命令

文/URFIX
编译/黄永兵

OpenSSH是SSH连接工具的免费版本。telnet，rlogin和ftp用户可能还没意识到他们在互联网上传输的密码是未加密的，但SSH是加密的，OpenSSH加密所有通信（包括密码），有效消除了窃听，连接劫持和其它攻击。此外，OpenSSH提供了安全隧道功能和多种身份验证方法，支持SSH协议的所有版本。

SSH是一个非常伟大的工具，如果你要在互联网上远程连接到服务器，那么SSH无疑是最佳的候选。下面是通过网络投票选出的25个最佳SSH命令，你必须牢记于心。

1、复制SSH密钥到目标主机，开启无密码SSH登录

```
ssh-copy-id user@host
```

如果还没有密钥，请使用ssh-keygen命令生成。

2、从某主机的80端口开启到本地主机2001端口的隧道

```
ssh -N -L2001:localhost:80 somemachine
```

现在你可以直接在浏览器中输入http://localhost:2001访问这个网站。

3、将你的麦克风输出到远程计算机的扬声器

```
dd if=/dev/dsp | ssh -c arcfour -C
 username@host dd of=/dev/dsp
```

这样来自你麦克风端口的声音将在SSH目标计算机的扬声器端口输出，但遗憾的是，声音质量很差，你会听到很多嘶嘶声。

4、比较远程和本地文件

```
ssh user@host cat /path/to/remotefile |
 diff /path/to/localfile -
```

在比较本地文件和远程文件是否有差异时这个命令很管用。

5、通过SSH挂载目录/文件系统

```
sshfs name@server:/path/to/folder
 /path/to/mount/point
```

从http://fuse.sourceforge.net/sshfs.html下载sshfs，它允许你跨网络安全挂载一个目录。

6、通过中间主机建立SSH连接

```
ssh -t reachable_host ssh
 unreachable_host
```

Unreachable_host表示从本地网络无法直接访问的主机，但可以从reachable_host所在网络访问，这个命令通过到reachable_host的“隐藏”连接，创建起到unreachable_host的连接。

7、将你的SSH公钥复制到远程主机，开启无密码登录 - 简单的方法

```
ssh-copy-id username@hostname
```

8、直接连接到只能通过主机B连接的主机A

```
ssh -t hostA ssh hostB
```

当然，你要能访问主机A才行。

9、创建到目标主机的持久化连接

```
ssh -MNf <user>@<host>
```

在后台创建到目标主机的持久化连接，将这个命令和你~/.ssh/config中的配置结合使用：

```
Host host
ControlPath ~/.ssh/master-%r@%h:%p
ControlMaster no
```

所有到目标主机的SSH连接都将使用持久化SSH套接字，如果你使用SSH定期同步文件（使用rsync/sftp/cvs/svn），这个命令将非常有用，因为每次打开一个SSH连接时不会创建新的套接字。

10、通过SSH连接屏幕

```
ssh -t remote_host screen -r
```

直接连接到远程屏幕会话（节省了无用的父bash进程）。

11、端口检测（敲门）

```
knock <host> 3000 4000 5000 && ssh -p
 <port> user@host && knock <host> 5000
 4000 3000
```

在一个端口上敲一下打开某个服务的端口（如SSH），再敲一下关闭该端口，需要先安装knockd，下面是一个配置文件示例。

```
[options]
logfile = /var/log/knockd.log
[openSSH]
sequence = 3000,4000,5000
seq_timeout = 5
command = /sbin/iptables -A INPUT -i eth0
 -s %IP% -p tcp -dport 22 -j ACCEPT
tcpflags = syn
[closeSSH]
sequence = 5000,4000,3000
seq_timeout = 5
command = /sbin/iptables -D INPUT -i eth0
 -s %IP% -p tcp -dport 22 -j ACCEPT
tcpflags = syn
```

12、删除文本文件中的一行内容，有用的修复

```
ssh-keygen -R <the_offending_host>
```

在这种情况下，最好使用专业的工具。

13、通过SSH运行复杂的远程shell命令

```
ssh host -l user $(<cmd.txt)
```

更具移植性的版本：

```
ssh host -l user “`cat cmd.txt`”
```

14、通过SSH将MySQL数据库复制到新服务器

```
mysqldump -add-drop-table -extended-
 insert -force -log-error=error.log
 -uUSER -pPASS OLD_DB_NAME | ssh -C
 user@newhost “mysql -uUSER -pPASS
 NEW_DB_NAME”
```

通过压缩的SSH隧道Dump一个MySQL数据库，将其作为输入传递给mysql命令，我认为这是迁移数据库到新服务器最快最好的方法。

15、删除文本文件中的一行，修复“SSH主机密钥更改”的警告

```
sed -i 8d ~/.ssh/known_hosts
```

16、从一台没有SSH-COPY-ID命令的主机将你的SSH公钥复制到服务器

```
cat ~/.ssh/id_rsa.pub | ssh user@machine
 “mkdir ~/.ssh; cat >>
 ~/.ssh/authorized_keys”
```

如果你使用Mac OS X或其它没有ssh-copy-id命令的*nix变种，这个命令可以将你的公钥复制到远程主机，因此你照样可以实现无密码SSH登录。

17、实时SSH网络吞吐量测试

```
yes | pv | ssh $host “cat > /dev/null”
```

通过SSH连接到主机，显示实时的传输速度，将所有传输数据指向/dev/null，需要先安装pv。

如果是Debian：

```
apt-get install pv
```

如果是Fedora：

```
yum install pv
```

（可能需要启用额外的软件仓库）。

18、如果建立一个可以重新连接的远程GNU screen

```
ssh -t user@some.domain.com
 /usr/bin/screen -xRR
```

人们总是喜欢在一个文本终端中打开许多shell，如果会话突然中断，或你按下了“Ctrl-a d”，远程主机上的shell不会受到丝毫影响，你可以重新连接，其它有用的screen命令有“Ctrl-a c”（打开新的shell）和“Ctrl-a a”（在shell之间来回切换），请访问

http://aperiodic.net/screen/quick_reference

阅读更多关于screen命令的快速参考。

19、继续 SCP 大文件

```
rsync -partial -progress -rsh=ssh
 $file_source $user@$host:
 $destination_file
```

它可以恢复失败的 rsync 命令，当你通过 VPN 传输大文件，如备份的数据库时这个命令非常有用，需要在两边的主机上安装 rsync。

```
rsync -partial -progress -rsh=ssh
 $file_source $user@$host:
 $destination_file local -> remote
```

或

```
rsync -partial -progress -rsh=ssh
 $user@$host:$remote_file
 $destination_file remote -> local
```

20、通过 SSH W/ WIRESHARK 分析流量

```
ssh root@server.com 'tshark -f "port !22"
 -w -' | wireshark -k -i -
```

使用 tshark 捕捉远程主机上的网络通信，通过 SSH 连接发送原始 pcap 数据，并在 wireshark 中显示，按下 Ctrl+C 将停止捕捉，但也会关闭 wireshark 窗口，可以传递一个“-c #”参数给 tshark，让它只捕捉“#”指定的数据包类型，或通过命名管道重定向数据而不是直接通过 SSH 传输给 wireshark，我建议你过滤数据包，以节约带宽，tshark 可以使用 tcpdump 替代：

```
ssh root@example.com tcpdump -w - 'port !
 22' | wireshark -k -i -
```

21、保持 SSH 会话永久打开

```
autossh -M50000 -t server.example.com
 'screen -raAd mysession'
```

打开一个 SSH 会话后，让其保持永久打开，对于使用笔记本电脑的用户，如果需要在 Wi-Fi 热点之间切换，可以保证切换后不会丢失连接。

22、更稳定，更快，更强的 SSH 客户端

```
ssh -4 -C -c blowfish-cbc
```

强制使用 IPv4，压缩数据流，使用 Blowfish 加密。

23、使用 cstream 控制带宽

```
tar -cj /backup | cstream -t 777k | ssh
 host 'tar -xj -C /backup'
```

使用 bzip 压缩文件夹，然后以 777k bit/s 速率向远程主机传输。Cstream 还有更多的功能，请访问

http://www.cons.org/cracauer/cstream.html#usage

了解详情，例如：

```
echo w00t, i'm 733+ | cstream -b1 -t2
```

24、一步将 SSH 公钥传输到另一台机器

```
ssh-keygen; ssh-copy-id user@host; ssh
 user@host
```

这个命令组合允许你无密码 SSH 登录，注意，如果在本地机器的~/.ssh 目录下已经有一个 SSH 密钥对，ssh-keygen 命令生成的新密钥可能会覆盖它们，ssh-copy-id 将密钥复制到远程主机，并追加到远程账号的 ~/.ssh/authorized_keys 文件中，使用 SSH 连接时，如果你没有使用密钥口令，调用 ssh user@host 后不久就会显示远程 shell。

25、将标准输入（stdin）复制到你的 X11 缓冲区

```
ssh user@host cat /path/to/some/file |
 xclip
```

你是否使用 scp 将文件复制到工作用电脑上，以便复制其内容到电子邮件中？xclip 可以帮到你，它可以将标准输入复制到 X11 缓冲区，你需要做的就是点击鼠标中键粘贴缓冲区中的内容。

原文：

http://blog.urfix.com/25-ssh-commands-tricks/

译文：

http://os.51cto.com/art/201011/235252.htm

推荐阅读：

五步建立一个 VNC Linux 服务器

http://server.51cto.com/sCollege-185750.htm

网络安全工具百宝箱

http://netsecurity.51cto.com/art/201011/235748.htm

Linux 系统管理员都应该熟悉的工具

http://os.51cto.com/art/201009/225721.htm

Tcprstat 是一个免费开源的 TCP 分析工具，用于监测网络流量，并计算请求与响应之间的延迟。

调查服务器响应时间的利器 tcprstat

文/Yu Feng

我们在做服务器程序的时候，经常要知道一个请求的响应时间，借以优化或者定位问题。

这时候来自 percona 的 tcprstat 来救助了！ 这个工具原本开发用来调查 mysqld 的性能问题，所以不要奇怪它的默认端口是 3306，但是我们可以用这个工具来调查典型的 request->response 类型的服务器。

什么是 tcprstat:

Tcprstat 是一个免费开源的 TCP 分析工具，用于监测网络流量，并计算请求与响应之间的延迟。该工具由此推算响应时间的数据并将记录输出，输出结果类似 Unix 下的其他 -stat 工具，如 vmstat ， iostat 和 mpstat 。该工具还可以被限定为仅仅监视特定端口的流量，很适合针对单独守护进程的响应时间监控，如 mysqld ， httpd ， memcached 等。

文档很详细，请参考：

http://www.percona.com/docs/wiki/tcprstat:start

不愿意编译的同学直接从这里下载 64 位系统的编译好的二进制：

http://github.com/downloads/Lowercases/tcprstat/tcprstat-static.v0.3.1.x86_64

源码编译也挺容易的： 由于它自带 libpcap 包， 这个包有可能在 configure 的时候没认识好 netlink， 只要把 config.h 里面的 netlink 那个 define 注释掉就好。

编译好了， 典型使用很简单：

```
# tcprstat -p 3306 -t 1 -n 5
timestamp   count   max min avg med
 stddev  95_max  95_avg  95_std  99_max
 99_avg  99_std
1283261499  1870    559009  39  883 153
 13306   1267    201 150 6792    323 685
1283261500  1865    25704   29  578 142
 2755    889 175 107 23630   333 1331
1283261501  1887    26908   33  583 148
 2761    714 176 94  23391   339 1340
1283261502  2015    304965  35  624 151
 7204    564 171 79  8615    237 507
1283261503  1650    289087  35  462 146
 7133    834 184 120 3565    244 358
```

但是这个 tcprstat 在 bonding 的网卡下有点问题：

```
# /sbin/ifconfig
bond0     Link encap:Ethernet  HWaddr
 A4:BA:DB:28:B5:AB
          inet addr:10.232.31.19
 Bcast:10.232.31.255  Mask:255.255.255.0
          inet6 addr:
 fe80::a6ba:dbff:fe28:b5ab/64 Scope:Link
          UP BROADCAST RUNNING MASTER
 MULTICAST  MTU:1500  Metric:1
          RX packets:19451951688 errors:0
 dropped:4512 overruns:0 frame:0
          TX packets:26522074966 errors:0
 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:6634368171533 (6.0 TiB)
 TX bytes:32576206882863 (29.6 TiB)
...
# tcprstat -p 3306 -t 1 -n 5
pcap: SIOCGIFFLAGS: bonding_masters: No
 such device
```

解决方案是：

```
# sudo tcprstat -p 3306 -t 1 -n 0 -l
 `/sbin/ifconfig | grep 'addr:[^ ]\+' -o
 | cut -f 2 -d : | xargs echo | sed -e
 's/ /,/g'`
```

用 IP 方式，而不是网络接口方式搞定。

原文：

http://blog.yufeng.info/archives/963

掌握 Shell，通常能够让任务在数秒钟内完成，这就让 Shell 跟 C、Perl、Python 这些语言区别开来。

系列连载：最牛 B 的 Linux Shell 命令（1）

文/Peteris Krumins
编译/BOY PT

Shell 作为 Unix 系操作系统当中最有魅力且不可或缺的组件，经过数十载的洗礼不仅没有被淘汰，而且愈加变得成熟稳健究其原因，大概因为它是个非常稳固的粘合剂能够把大量功能强大的组件任意配搭，总能很好很快地完成用户的任务。

本文的一些命令很可能看起来是”雕虫小技”，我们只好仰慕一下 Shell 大牛了，但是有些细节我会稍加发掘加以说明，遇到有趣的地方希望能博您一笑了。

1.以 SUDO 运行上条命令

```
$ sudo !!
```

大家应该都知 sudo，不解释。但通常出现的情况是，敲完命令执行后报错才发现忘了 sudo。这时候，新手用户就会：按上箭头，按左箭头，盯着光标回到开始处，输入 sudo，回车；高手用户就蛋定多了，按 Ctrl-p，按 Ctrl-a，输入 sudo，回车。

这里介绍这个是天外飞仙级别的，对，就直接 sudo !!。

当然这几种解决方式效果是完全一样的，只是款不一样，嗯，不解释。

两个感叹号其实是 bash 的一个特性，称为事件引用符（event designators）。!!其实相当于!-1，引用前一条命令，当然也可以 !-2，!-50。默认情况下 bash 会在 ~/.bash_history 文件内记录用户执行的最近 500 条命令，history 命令可以显示这些命令。

关于事件引用符的更多用法可以深入阅读 The Definitive Guide to Bash Command Line History。

2.以 HTTP 方式共享当前文件夹的文件

```
$ python -m SimpleHTTPServer
```

这命令启动了 Python 的 SimpleHTTPServer 模块，考虑到 Python 在绝大多数的 Linux 发行版当中都默认安装，所以这个命令很可能是最简单的跨平台传文件的方法。

命令执行后将在本机 8000 端口开放 HTTP 服务，在其他能访问本机的机器的浏览器打开 ttp://ip:8000 即打开一个目录列表，点击即可下载。

3.在以普通用户打开的 VIM 当中保存一个 ROOT 用户文件

```
:w !sudo tee %
```

这题目读起来纠结，其实是很常见的，常常忘记了 sudo 就直接用 vim 编辑/etc 内的文件，（不过也不一定，vim 发现保存的文件无法保存时候会提示）等编辑好了，保存时候才发现没权限。曲线方法是先保存个临时文件，退出后再 sudo cp 回去。不过实际上在 vim 里面可以直接完成这个过程的，命令就是如此。

查阅 vim 的文档（输入:help :w），会提到命令:w!{cmd}，让 vim 执行一个外部命令 {cmd}，然后把当前缓冲区的内容从 stdin 传入。

tee 是一个把 stdin 保存到文件的小工具。

而%，是 vim 当中一个只读寄存器的名字，总保存着当前编辑文件的文件路径。

所以执行这个命令，就相当于从 vim 外部修改了当前编辑的文件，好完工。

4.切换回上一个目录

```
$ cd -
```

应该不少人都知道这个，横杆-代表上一个目录的路径。

实际上 cd - 就是 cd $OLDPWD 的简写，bash 的固定变量$OLDPWD 总保存着之前一个目录的路径。

相对地，$PWD 总保存着当前目录的路径。这些变量在编写 shell 脚本时候相当有用。

5.替换上一条命令中的一个短语

```
$ ^foo^bar^
```

又是另外一个事件引用符（event designator），可以把上一条命令当中的 foo 替换成 bar。

在需要重复运行调试一道长长的命令，需要测试某个参数时候，用这个命令会比较实用；但多数人会首先选择按上箭头提出上道命令，再移动光标去修改某参数，这样更直观，但效率上就不够使用引用符高，而且在脚本中用这个方法可以简化很多。

这道命令的原始样式应该是这样的：

```
!!:s/foo/bar/
```

本文一开始介绍过!!，后面的一段大家应该很熟悉，vim、sed 的替换操作都是这样的语法。

关于事件引用符的更多用法可以深入阅读 The Definitive Guide to Bash Command Line History

6.快速备份一个文件

```
$ cp filename{,.bak}
```

这道命令把 filename 文件拷贝成 filename.bak，大家应该在一些比较复杂的安装教程里面见过这样的用法。其原理就在于 bash 对大括号的展开操作，filename{,.bak} 这一段会被展开成 filename filename.bak 再传给 cp，于是就有了备份的命令了。

大括号在 bash 里面是一个排列的意义，可以试试这个：

```
$ echo {a,b,c}{a,b,c}{a,b,c}
```

将输出三个集合的全排列：

```
aaa aab aac aba abb abc aca acb acc
baa bab bac bba bbb bbc bca bcb bcc
caa cab cac cba cbb cbc cca ccb ccc
```

关于 shell 当中的集合操作，可深入阅读”Set Operations in the Unix Shell”

7.免密码 SSH 登录主机

```
$ ssh-copy-id remote-machine
```

这个命令把当前用户的公钥串写入到远程主机的~/.ssh/authorized_keys 内，这样下次使用 ssh 登录的时候，远程主机就直接根据这串密钥完成身份校验，不再询问密码了。前提是你当前用户有生成了公钥，默认是没有的先执行 ssh-keygen 试试吧！

这个命令如果用手工完成，是这样的：

```
your-machine$ scp ~/.ssh/identity.pub
 remote-machine:
your-machine$ ssh remote-machine
remote-machine$ cat identity.pub >>
 ~/.ssh/authorized_keys
```

如果你想删掉远程主机上的密钥，直接打开 authorized_keys，搜索你的用户名，删除那行，即可。

8.抓取 LINUX 桌面的视频

```
$ ffmpeg -f x11grab -s wxga -r 25 -i :0.0
 -sameq /tmp/out.mpg
```

我们在一些视频网站上看到别人的 3D 桌面怎么怎么酷的视频，通常就是这么来的，ffmpeg 可以直接解码 X11 的图形，并转换到相应输出格式。

ffmpeg 的通常用法是，根据一堆参数，输出一个文件，输出文件通常放最后，下面解析下几个参数：

-f x11grab 指定输入类型。因为 x11 的缓冲区不是普通的视频文件可以侦测格式，必须指定后 ffmpeg 才知道如何获得输入。

-s wxga 设置抓取区域的大小。wxga 是 1366*768 的标准说法，也可以换成 -s 800×600 的写法。

-r 25 设置帧率，即每秒抓取的画面数。

`-i :0.0` 设置输入源，本地 X 默认在 0.0

`-sameq` 保持跟输入流一样的图像质量，以用来后期处理。

后记

说 Shell 是一种编程语言，可能有些尴尬，虽然很多人每天都在用 Shell，但从来没见它荣登 TIOBE 编程语言排行榜之类的，可以说毫无名分，因为很多用户没意识到它是一种语言只当做这是一个能够很好完成任务的工具，基本得理所当然，就好像 GUI 程序的菜单、按钮一样。

掌握 Shell，通常能够让任务在数秒钟内完成，这就让 Shell 跟 C、Perl、Python 这些语言区别开来，没人否认后者更能胜任更多的任务，但是他们是在不同的层面上去做，Shell 依赖大量的系统组件黏合调用，而后者依赖各种库，各所擅长不同的应用领域，比喻就是，Shell 是混凝土，可以很方便地粘合一些建筑组件而成为稳固的高楼大厦；但同样是粘合剂，粘玻璃窗、粘书报、粘皮鞋，混凝土是绝对不合适的，Shell 并不擅长一些细致操作，比如它连浮点运算都不支持，更别提什么图形运算什么的。但这并不妨碍 Shell 来帮我们完成很多粗重任务。

Shell 的工作方式，大多数入门用户会觉得枯燥难学，而所谓的经典教材也离不开《Advanced Bash-Scripting》、《Bash Guide for Beginners》，但类似本文这样的一些”雕虫小技”因为难登大雅之堂绝不会收录进去。这情况如果象国外一些 unix 用户比较多的地方会有很好改善，即使是新手，偶尔看看别人的操作都能”偷师”一手，我编译本系列文章其实也就希望稍微改善一下这个状况。

原文：

http://www.catonmat.net/blog/top-ten-one-liners-from-commandlinefu-explained

译文：

http://www.isspy.com/most_useful_linux_commands_1/

推荐阅读：

实战 Linux Shell 编程与服务器管理
http://book.51cto.com/art/201003/191285.htm

不看后悔的 Linux 生产服务器 Shell 脚本分享
http://os.51cto.com/art/201010/229129.htm

从 Solaris 迁移 Perl 脚本到 Linux
http://os.51cto.com/art/201011/234181.htm

几个常用的 Linux 监控脚本
http://os.51cto.com/art/201010/229618.htm

勘误声明

《Linux 运维趋势》第一期的第 11 页：《资料篇：Linux 常用监控命令简介 - top》一文中，有两处错误：

1、top 命令的解释中，有关-i 参数的说明，写成了“显示空闲进程”，实际应为“排除空闲进程跟僵尸进程，只显示出运行的进程”；

2、有关 uptime 输出数值的说明，写成了“最近一秒，五秒，十五秒”，实际应为“最近一分钟，五分钟，十五分钟”；

特此勘误。

感谢读者星痕指出此错误！希望大家能够继续积极反馈您对《趋势》的意见与建议，帮助我们做的更好！

招募启事

《Linux 运维趋势》的建设需要您的加入！

您可以通过如下方式参与我们杂志的建设：

1、推荐文章

无论是您在互联网上看到的好文章，还是您自己总结/整理的资料；无论是英文还是中文；无论是入门的还是高端的，都欢迎推荐！推荐方式包括：

a）在技术圈中分享：http://g.51cto.com/linuxops

b）在邮件群中分享：linuxops-cn@googlegroups.com

c）发邮件给编辑：yangsai@51cto.com

2、投稿

如果您认为自己在 Linux 方面具有专家级别的能力，并且有与大家分享您技术经验的热诚，同时也有兴趣挣点稿费花花，那么欢迎您的投稿！

如果您在 IT 技术方面的翻译有很高的能力，能够快速、高质量的完成译文，并且也经常浏览到一些 Linux 方面的优秀外文，那么也欢迎您的投稿！

投稿信箱：yangsai@51cto.com

3、推广与意见

如果您喜欢我们的杂志，认为这本杂志对于您的工作有所帮助，请向您的 Linux 好友、同事们推荐它！

如果您觉得这份杂志还有什么地方需要改进或补充，也希望您能够提出您的宝贵意见！

联系人：yangsai@51cto.com

下期预告

下期主题为：Linux 服务器的性能瓶颈分析。敬请期待！

本刊为月刊，预定每月发布日期为：

每个月的第二个星期五

您可以通过如下方式检查是否有新刊发布：

1、加入电子邮件群组：
linuxops-cn@googlegroups.com

获得邮件提醒

2、经常光顾 51CTO Linux 频道：
http://os.51cto.com/linux/

《Linux 运维趋势》是由 51CTO 系统频道策划、针对 Linux/Unix 系统运维人员的一份电子杂志，内容从基础的技巧心得、实际操作案例到中、高端的运维技术趋势与理念等均有覆盖。