

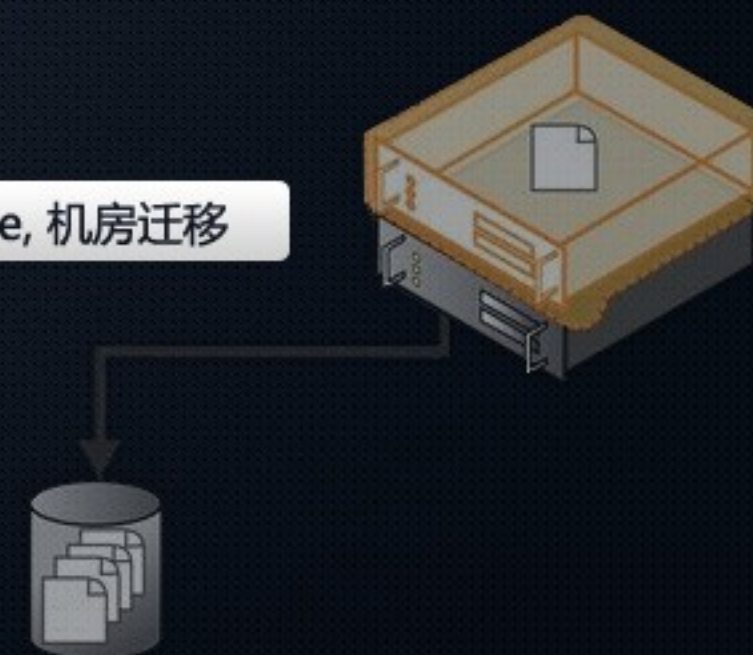
Linux 运维趋势

第七期

2011年4月号

关键字 Linux, Windows, MySQL, Oracle, 机房迁移

网站迁移



Linux 运维趋势

杂志策划：[51CTO 系统频道](#)

本期主编：李晶

责任总编：杨赛

封面制作：高鹏飞

交流圈子：

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

专题页面邮件订阅入口：

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

下载汇总页：

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

投稿邮箱：

yangsai@51cto.com

内容目录

【人物】

天涯社区运维刘天斯：因分享而快乐 因快乐所以坚持.....3

【交流】

VDI 情景下有关多核 CPU 必要性的探讨.....5

【八卦，趣闻与数字】2011.03 – 2011.04.....7

【本期专题：迁移】.....8

资料整理：服务器迁移经验谈.....9

阿里巴巴核心业务系统数据库平台迁移：Oracle -> MySQL.....12

数据库迁移几种方式简单介绍.....14

DNS Server 迁移笔记.....16

实战：如何实现从物理到虚拟基础架构迁移.....17

【工具】

通过 dsh 批量管理 Linux 服务器.....20

用 ClusterSSH 管理多台 Linux 服务器.....22

CairoPlot 让 Linux 服务器的日志文件更直观.....25

个人建议从前端应用、负载均衡、缓存层开始着手，理由是不仅不会给应用逻辑产生很大的干扰，同时还会给整个平台带来高可用性，保证在转型过程中平稳过渡。

天涯社区运维刘天斯：因分享而快乐 因快乐所以坚持

采访/李晶



人物简介：

刘天斯，天涯社区运维工程师。技术特长是架构设计、Linux 系统运维、运维开发等方面。目前关注的是云计算方面的知识。

个人博客：<http://blog.liuts.com>

大型论坛社区的海量存储、海量访问对于每一个运维来说都是挑战性的。对于天涯社区来说，是如何在过去数年间逐渐从 Windows 平台全面转移到开源架构的？天涯社区的运维是如何利用自己的知识和能力让这一过程变得平稳的？我们请天涯社区的运维刘天斯根据自己的工作经验来谈谈他的运维经验分享。

51CTO：首先，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，现在主要负责哪方面的工作等等。

刘天斯：03 年毕业后在一家网络公司从事开发兼美工设计的工作，一次偶然的机会接触到了 Linux，从一开始的系统安装到各类常用应用平台的搭建。一步一步从一个程序员到运维工程师的转变。2005 年加入天涯，正式成为一名运维工程师。目前主要负责的工作有

应用平台的上线、部署、监控、调优、事件处理、应急预案等，涉及的领域有架构设计与优化、运维开发、负载均衡、缓存应用、数据库分布式存储等。

51CTO：您在天涯社区做系统管理员的这段时间内遇到的最大困难是什么？您最难忘或最高兴的事是什么？

刘天斯：如何协调各部门与运维的关系，不要再当炮灰了。最高兴的是部门定期聚餐，喝喝小酒，聊聊人生，畅所欲言。

51CTO：您对开源是如何理解的？天涯社区在过去两年间陆续开源了包含 LVS 管理系统、Varnish 缓存推送平台、高性能数据引擎 memlink 等好几个项目，业内的同行们都十分关注，您认为这个对整个产业带来了哪些好处？身为一个天涯的运维，您认为这个过程对您自己的价值在哪里？

刘天斯：开源就是分享，让更多的人受益的同时自己也在提高。经常看到很多朋友都在做监控平台、运维工具。事实上功能都是惊人的相似，大家都在做同样重复的工作，为什么不能其中一个人开源出来大家一起来使用、完

善。对整个行业来讲这块的投入成本都会降低对个体来讲也是资源的整合。如果形成良性的循环，行业的生态环境将有很大程度的改善。本人热衷于开源技术，同样也愿意为开源贡献自己一分微薄之力，希望更多的人能支持开源参考开源。

51CTO：天涯社区在过去数年间逐渐从Windows平台全面转移到开源架构，很多朋友对天涯的这个转型过程很感兴趣，网上现在也有不少相关的资料。对于这个过程，您有哪些觉得特别好的资料想推荐给大家的吗？您自己在这个过程中有哪些心得？

刘天斯：这个过程是漫长的，是一个快速发展壮大的互联网公司的必经之路。转型之前需考虑资源的投入、人才的储备。下一步便是相关技术及经验的积累，这方面的资源互联网上已经很丰富了。个人建议从前端应用、负载均衡、缓存层开始着手，理由是不仅不会给应用逻辑产生很大的干扰，同时还会给整个平台带来高可用性，保证在转型过程中平稳过渡。

51CTO：作为《2010年度十大杰出IT博客》之一，能否简单介绍一下您开博的经历？一直坚持记录博客，是有什么动力在驱动自己吗？

刘天斯：开始写博客的目的只是单纯当成笔记本来用，后来慢慢变成交流沟通的平台，收到很多网友的评论、邮件。当中有很多是非常好的建议，比如文章“LAMP+logzilla2.9.9+syslog-ng实现集中日志管理(第二版)”，共收到5位网友提供后续问题案例，自己也从中得到学习。因分享而快乐，因快乐所以坚持。

51CTO：您现在最关注哪方面的技术？您自己是如何学习并掌握新技术的？

刘天斯：我现在关注的技术是云计算，思考如何构建一个高效的私有云。学习方法大家都会有自己的一套，工作期间每天安排出2个小时的学习时间就很不错了。掌握一门新技术是很不容易的，“边学边练”的方法在每个阶段都是适用的。当然，除了兴趣以外还需要有一定的毅力。

51CTO：最后，您认为在未来两年内，大型网站的系统管理员(或想成为大型网站系统管理员的同行们)最需要关注哪些方向？

刘天斯：是云计算，它除了给我们降低运营成本，还提供了更加弹性的资源分配，可伸缩的处理能力，满足大量的数据存储与计算，最终提高企业竞争力，这些都是未来几年我们所需要的。■※

原文：

<http://os.51cto.com/art/201103/251458.htm>

推荐阅读：

[将你 Windows 上的设置和数据迁移到 Linux](#)

[工业软件向云平台迁移是福音还是隐患](#)

[不受制于厂商 大型机迁移概念及优势](#)

[云迁移≠云转型 先迁移还是先转型？](#)

[保护你的 Ubuntu 服务器](#)

[51CTO 技术沙龙总结：LAMP 架构扩展](#)

[优秀的 Windows 管理员应该具有的 9 大品质](#)

对于虚拟桌面基础架构 (VDI) 解决方案来说, 多核芯片完全是浪费钱, 而且高可用性被荒废了。

VDI 情景下有关多核 CPU 必要性的探讨

文/Andre Leibovici
编译/布加迪

在旧金山召开的 IDF 2010 上, 英特尔发布了其下一代服务器至强处理器, 代号为 Westmere-EX。性能不可谓不惊人: 10 核 20 线程的处理器对三通道内存的寻址能力达到了 2TB, 这相当于前一代处理器的内存寻址能力的整整两倍。

然而, 对于虚拟桌面基础架构 (VDI) 解决方案来说, 多核芯片完全是浪费钱, 而且高可用性被荒废了。

我的客户经常问我有什么建议好提供, 同时又告诉我, 他们一直在考虑购买配备四核、六核甚至八核的 Nehalem 处理器。我的答复始终是反问一句: 为什么需要多核?

不妨想想 vSphere 4.1、vCenter 4.1、VMware View 4.5 和 View Composer 2.5 方面的一些局限性。这些是组成 View 解决方案的最新一代的几个组件。

vCenter 4.1

经过验证的架构支持 2000 个虚拟桌面

vSphere 4.1

每个主机最多支持 320 个虚拟机

在 Nehalem 系统中, 经过验证的架构每个核心支持 16 个虚拟桌面

View Composer 2.5

每个集群最大支持 8 个主机

每个数据存储区最多支持 128 个链接克隆虚拟桌面

在一个新部署的系统中, Windows7 很可能是 GuestOS, 而该操作系统的最大推荐内存容量是 2GB。为了本文的需要, 我们还假设不用到处理器密集型的应用程序, 单单一个虚拟处理器 (vCPU) 就足以满足大多数用户的需要。不过, 由于 Windows7 的地址空间布局随机化功能 (ASLR), 透明页面共享 (TPS) 比率减小到了大约 10% (如果使用 Windows XP, TPS 比率约为 40%)。

鉴于我们已作好了假定和限制, 现在运行几个模拟。完全出于好奇心, 第一个模拟将使用发布不久的 32 核处理器。

64 个核心 (2 个插座 x 32 个核心) x 每个主机 16 个虚拟机 = 每个主机 1024 个虚拟机 (禁用超线程技术)

这种场景需要每个主机约 2008GB 内存。

约 128 个核心 (2 x 32 个超线程插座) x 每个主机 16 个虚拟机 = 每个主机 2048 个虚拟机 (启用超线程技术)

这种场景需要每个主机约 4000GB 内存。

闲话少说，vSphere 每个主机最多支持 320 个虚拟机。就第一个例子而言，每个处理器/核心最多支持的虚拟机数量必须是 5，那样虚拟机的数量才能保持在 320 个以下。在这种情况下，系统和处理器在很大程度上将得不到充分利用。

第二种场景使用 128 个超线程核心，它总共需要 4000GB 内存。该架构最多支持 2TB 内存，因而为了将内存控制在极限范围以内，每个处理器/核心支持的虚拟机数量最多是约 8 个。情况看起来比不用超线程技术要好一点，但是昂贵的硬件资源仍然基本上得不到充分利用。

这两种场景让人觉得像是在玩数字游戏。不妨看看在使用 2 个插座、每个插座有 6 个处理器/核心的实际部署环境下会发生什么情况。我在这种场景下没考虑超线程和睿频加速 (Turbo Boost) 技术，因为它们其实并不会让核心数量翻一番。

12 个核心 (2 个插座 x 6 个核心) x 每个主机 16 个虚拟机 = 每个主机 192 个虚拟机

这种场景需要每个主机约 312GB 内存。

12 个核心的场景对于每个主机支持的虚拟机数量来说很适合，不过要求每个主机约 384GB 内存。而现在外面的系统大多并不提供支持这么大内存容量的功能，只有少数几个例外，比如思科 ASIC 内存扩展架构。

如果你总共有约 550 个虚拟机，这代表停运时间接近 33%。单单就存储需求而言，拥有 192 个虚拟机的每个主机在每个用户每秒 10 次输入输出操作 (10 IOPS) 的正常工作负载、读写比为 20/80 的情况下，大概需要 6528 次 IOPS。另外别忘了考虑所有虚拟机的网络连接和吞吐量。

对于 VDI 解决方案而言，我的客户大多数目前采用配备 2 个插座、每个插座 4 个核心的 Nehalem 系统，也就是说每个主机总共有 8 个核心。对我来说，这种配置似乎在成本、性能与可用性之间提供了最佳组合。使用每个处理器/核心 16 个虚拟机的验证架构，就有可能让每个主机运行约 128 个虚拟机；而每个主机需要约 256GB 内存。

对于有些系统来说，256GB 的内存仍然行不通。在这种情况下，每个处理器/核心 12 个虚拟机就允许每个主机可支持 96 个虚拟机和 192GB 内存。

值得一提的是，如果一个包括 8 个主机的集群其每个处理器/核心运行 16 个虚拟机，允许最多有 1024 个虚拟机，但没有高可用性事件所需的备用容量。这种情况下，如果某个主机关闭或丢失，就没有备用容量来启动额外主机上的虚拟机。运行 8 个主机的集群，每个处理器/核心又运行 12 个虚拟机，允许支持 768 个虚拟机，又拥有正好可以支持主机故障的容量。

我个人的建议是，在装满 8 个主机的集群中，每个处理器/核心最多只用 12 个虚拟机。

❖

本文为节选，完整内容请参考英文原文：
[Cores and more Cores... We don't need them!](#)

译文全文：

<http://virtual.51cto.com/art/201104/253057.htm>

2011年3月到4月之间，发生了下面这些事.....

八卦，趣闻与数字 2011.03 – 2011.04

收集整理/51CTO 系统频道

【CentOS 4】CentOS 开发团队在3月4日发布了4.x 分支的最新版本4.9。

<http://os.51cto.com/art/201103/247292.htm>

【CentOS 5】CentOS 5.6 目前已经可以在部分镜像站获取下载地址。本次更新包含了很多错误修正、升级和新功能。

<http://os.51cto.com/art/201104/253965.htm>

【Debian】Debian 改名蝶变？经证实，“蝶变”这个译名并没有被社区人员认可，而是有人未和其他人打招呼而直接修改了 wiki 才有的。看看 Debian Developers 对这件事是怎么说的。

<http://os.51cto.com/art/201103/249473.htm>

【openSUSE】openSUSE 11.4 发布，基于 Linux 2.6.37 版内核，集成 KDE 4.6、GNOME 2.32 作为主要的桌面环境。

<http://os.51cto.com/art/201103/249135.htm>

【Ubuntu】Ubuntu 11.04 Natty 之后的 11.10 版开发代号已确定，代号名为 Oneiric Ocelot，意为“梦幻般的虎猫”。

<http://os.51cto.com/art/201103/247858.htm>

【内核】Linux 2.6.38 内核已经正式发布。该版本去掉了过去的全局锁，这一举措进一步提升了 Linux 的性能。

<http://os.51cto.com/art/201103/249821.htm>

【Debian 6.0】Debian 项目发布了 Debian 6.0 的第一个更新，该更新主要修复了一些安全方面的问题。

<http://os.51cto.com/art/201103/249967.htm>

【Ubuntu】Ubuntu 技术委员会在上周四下午针对 Ubuntu 是否要默认安装第三方软件包进行了投票，投票结果是否定的。也就是说以后 Ubuntu 将不会默认安装 Flash Player 等第三方应用程序。

<http://os.51cto.com/art/201103/251639.htm>

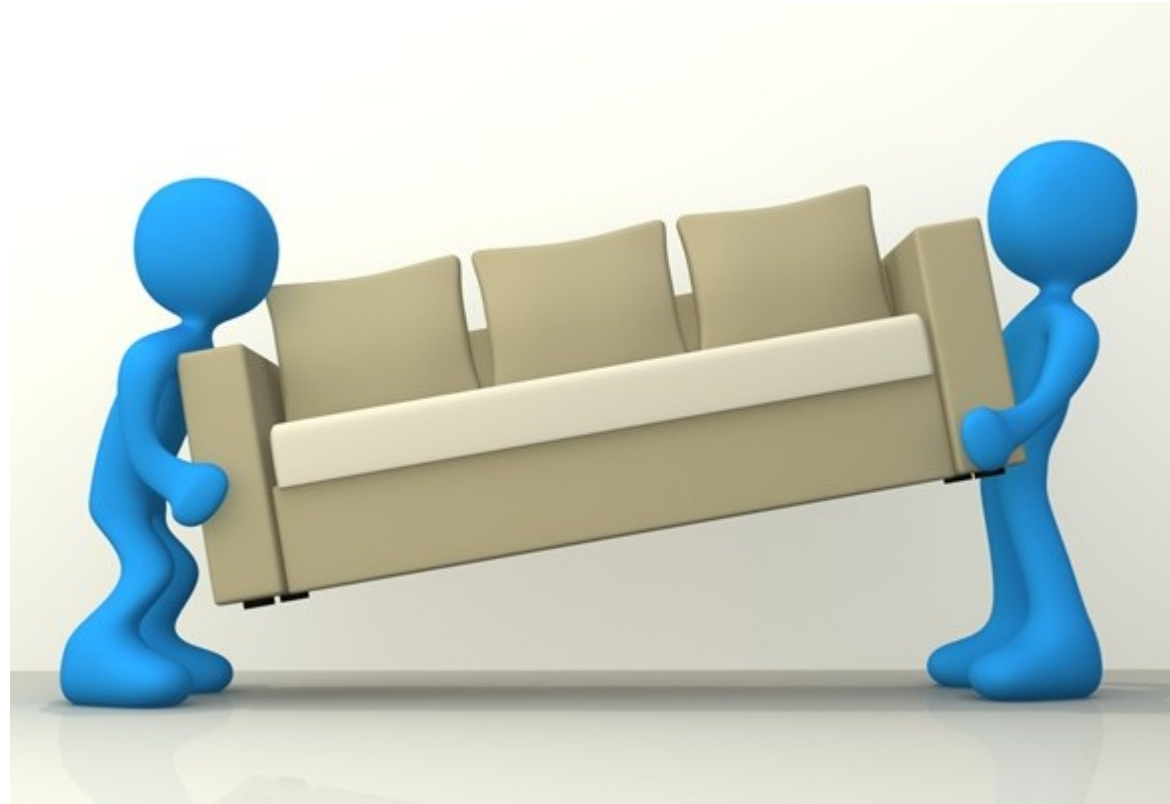
【Gnome 3】在 Gnome 2 发布 9 年之后，搭载全新用户界面 Gnome Shell 的新一代桌面环境 Gnome 3 终于正式发布。

<http://os.51cto.com/art/201104/253891.htm>

【愚人节】本文收录了作者最喜爱的一些恶作剧手段，它们可以让你在愚人节中充分向那些毫不知情的同事们演示自己疯狂而独特的“个人魅力”。

<http://os.51cto.com/art/201104/252890.htm>

本期专题：迁移



服务器迁移大致上有如下几种情况：

- 1、软件层架构的变动，如 Windows 迁移到 Linux，RHEL4 迁移到 RHEL5，CVS 迁移到 Git，Oracle 迁移到 MySQL 等
- 2、底层的变动，如硬件更换，虚拟机-物理机双向迁移，机房迁移，迁移到云平台等

为什么要迁移？服务器迁移有什么需要注意的地方？本专题将分享有关服务器迁移的一些经验。

如果配置文件比较多比较零散，怕出错的话，建议列一张清单，然后一条一条执行。

资料整理：服务器迁移经验谈

整理编译/杨赛

有关服务器迁移，一般的实际需求是怎样的？迁移过程中常见的问题又有哪些？以下内容采集编译自 [Server Fault 网站](#) 有关 migration 词条的 [FAQs](#)，整理了有关服务器迁移方面的常见问题（主要集中在 Linux 相关的内容）。

迁移单台服务器需要注意什么？

Q：我有台 Ubuntu Server 9.04，跑着 web，数据库和 mail，整个硬盘有 5GB 数据。打算换台好机子，但是我没做过迁移，能不能给些入门指导？

A：首先，备份所有的东东，并在新机子上恢复。这包括：

1、所有的数据库 dump 出来，在新机子上 restore

2、webserver、数据库、email 的配置文件复制过去

当然，免不了会有些停机时间。想减少停机时间，还有几个建议：

1、迁移前先对所有 email 数据做一次 rsync，在所有都配置、测试完毕之后，将旧服务器上的服务停掉，仅 rsync 最新的数据，启动新服务器

2、直到切换之前，使用 MySQL replication 确保两台服务器的数据库内容一致

3、rsync 可以用于一切数据，甚至数据库也可以，当然为了避免错误，数据库还是使用 dump 或 replication 吧。

如果配置文件比较多比较零散，怕出错的话，建议列一张清单，然后一条一条执行。另外，所有的配置文件都应该在 /etc 下面，除非没用软件包或者安装了专有软件的情况才有在 /opt 下面的。

如何把用户账户迁移到新的 Linux 机器？

Q：我们有个内部 Subversion 库运行在 Linux 机子上，Ubuntu 8，通过 svn+ssh 验证方式登录。最近入手了台新机子，也装了 Linux，Ubuntu 9，用 RAID 1+0 配置了更大磁盘空间，所以想把 Subversion 库迁移过去。

我怎么把所有的 user，group 和文件信息都迁移过去？我打算把 /etc/passwd 和 /etc/group 以及所有 /home 下的用户目录都复制过去，这样就可以了么？

A：你需要复制的内容包括：

```
/etc/passwd
/etc/shadow
/etc/group
/etc/gshadow
/var/spool/mail
```

/home

详细情况可参考[本篇文章](#)。

A：用户不多的话，直接 useradd (定义同样的 uid 和 gid) 会比较快一些。

另外，在新机子上创建号用户之后，记得将老机子上的登录禁止掉，并用 rsync 把用户的 home 目录同步过来。迁移完毕之后，重置一下密码即可。

之所以要避免复制 /etc/passwd 和 /etc/group ，是防止你把机子搞乱之后自己 (或 root) 登录不进去了。而去如果发行版不同，对这些文件的解读也会有区别，所以要谨慎。一个不小心，可能会开放了 guest 登录，或导致其他安全问题。当然了，虽然说复制密码文件要谨慎，但是也不用过于神经质。

从 Apache 迁移到 Nginx 好吗？有什么经验分享？

Q：我目前使用 Apache 的以下模块：多个虚拟主机，Server Side Include，以及 FastCGI，想问问各位有关迁移到 Nginx 的问题。大家有什么经验分享，比如迁移过程中

的问题，迁移之后有没有好处，nginx 上有用的模块等？

A：我个人经验而言是值得的。比如我有个 Magento 搭建的电子服务站 (大家都知道 Magento 是很慢的)。迁移到 nginx+php-fcgi/php-fpm+apc 之后，我这边的性能提高了 100%。所以，除非 Apache 上有必要的模块，否则我推荐 Nginx。具体情况可以参考[我的 Magento 调优笔记](#)。

A：你要的这三个模块：多个虚拟主机，SSI，还有 FastCGI，Nginx 都支持。我这边同时应用 Lighttpd，Apache 和 Nginx。无法彻底抛弃 Apache，是因为我们需要运行一些自定义模块 (包含一个改良版 mod_pubcookie)，而这些功能很难迁移到 Lighttpd/Nginx 上。

我用 Lighttpd 做轻量内容服务器，但是它在 FreeBSD 上的表现不是很好 (尤其是使用 FreeBSD 的 sendfile / kqueue syscalls 的时候 Lighttpd 会死掉，还连带着把整个服务器也弄宕机)。而 Nginx 方

面我就没遇到过任何问题。所以我正在用 Nginx 替换掉所有的 Lighttpd。

如何快速迁移 MySQL？

Q：我有大概 40 来个中小规模的 MySQL 数据库，需要把它们从一个 whm 服务器的数个 cpanel 帐号迁移到另一个服务器上。我本来的想法是手动一个一个 dump 然后 import，但是很费时间，有什么更快的办法么？

A：能用 ssh 的话，我知道一个很快的方法 mysqldump 配合几个参数，并和 ssh 链起来使用。这样可以让源数据库一边不中断的输出目标数据库一边不中断的导入，而去也不会用到任何临时文件：)

```
源服务器# mysqldump --user=user1 --all-databases | ssh 目标服务器 'mysql --user=user2'
```

如果你在源服务器上使用你的私人密钥和 ssh-agent 验证过，就可以使用 ssh 的 -A 参数来建立连接，就不用被目标服务器的验证信息烦来烦去了。当然，Agent forwarding 有安全隐患，要谨慎使用。

Q：我的 MySQL 备份文件怎么在 SQL Server 2008 里恢复？

A：你不能在 SQL Server 2008 里自动 restore 一个 MySQL 的备份文件。你可以写一个转换脚本，或者使用微软提供的 DTS 等工具。

服务器配置如何迁移？

Q：我有台 Debian Lenny 服务器，慢得很，所以最近买了台新机子，想要将整个 Debian 配置——包括用户账户、路径、安装的软件等——迁移过去。有什么比较快速的做法推荐？

A：网络安全而给力的话，netcat 配合 tar 是最好的方式。具体执行方法参考[这篇文章](#)。不要忘记用 live distro 重启服务器，chroot 进系统重装一下 MBR。如果用 ACL 的话，也不要忘记使用 bsdtar。

A：Puppet 或 Cfengine 这样的配置管理工具会比较简单些。如果之前做过部署，那么只要应用相同的 classes 就能让它自动部署你的配置。就算没做过，现在开始也不晚，给

puppet 做下描述，在原机子上应用检查一下，再应用到新机子上，这样以后就可以随时建立同样配置的新服务器了。虽然第一次使用 puppet 做配置会更加费时间，但长远来看，绝对是更加节省时间的。

A：直接把整台机器 rsync 过去。

虚拟机实时迁移的相关问题？

Q：以前一直以为 AMD 和英特尔服务器之间是没有虚拟系统支持实时迁移的，直到今天才从 KVM FAQ 上看到 KVM 是支持这个实时迁移，而去 2008 年就有案例的。现在 2011 年了，想问一下现在有哪些主流的虚拟系统（ESXi，Xen，Hyper-V 等）支持 AMD 和英特尔服务器之间的实时迁移的？

A：ESX/ESXi 的话，可以通过一系列“高级配置”和 VM CPU-bit 设置实现 AMD 和英特尔之间的 vmotion——不过，这样造成的问题比它带来的便利要多得多。

ESX/ESXi 之所以不默认允许跨 CPU 架构的实时迁移，因为不同的 CPU 家族的能力各有不同。比如一个现代的英特尔 CPU 上跑着

一个使用了 SSE4.2 指令的实例，那么你把它 vMotion 到一个不支持该指令的 CPU 上，实例就会崩溃。

解决方法之一是设定一个“最大公约数”，即在一个集群中，采用最烂的 CPU 来设定 VM 的运行方式，忽略那些新 CPU 里面的各种功能和指令。■※

资料来源：

<http://serverfault.com/questions/239138/migrating-from-one-dedicated-linux-server-to-another>

<http://serverfault.com/questions/253012/other-than-kvm-does-any-hypervisor-support-live-migration-between-amd-and-intel>

<http://serverfault.com/questions/19274/how-to-transfer-user-accounts-to-a-new-linux-machine>

<http://serverfault.com/questions/22987/experience-in-migrating-from-apache-to-nginx>

<http://serverfault.com/questions/163305/fastest-way-to-move-multiple-databases-to-a-new-server>

原文：

<http://os.51cto.com/art/201104/254412.htm>

这个项目无论是从技术角度还是业务角度来说，都对我们有着非常大的价值，也必定会带来非常深远的影响。

阿里巴巴核心业务系统数据库平台迁移：Oracle -> MySQL

文/Sky.Jian

为了对核心技术拥有更多的自主控制能力，为了解决数据库的线性扩展问题，为了尽量减少对商业软件的依赖，为了摆脱对高端硬件的依赖，为了... 基于以上多种原因，2年前，我们计划将公司某核心应用平台进行大手术：数据库平台从软件到硬件全部重构。当然，这其中应用架构的改造也不可避免的进行了大换血。

这个项目无论是从技术角度还是业务角度来说，都对我们有着非常大的价值，也必定会带来非常深远的影响。项目历时2年多，分4个阶段才完成：

应用接口统一

这一阶段主要是为了后面真正迁移的时候做准备工作，将该核心应用系统的所有数据访问入口统一到一起，全部以服务化的接口方式呈现给其他有需要的系统，一来方便后续变更的控制，二来也推进了服务化的进程。

Oracle 数据库中拆分（1拆16）

这个阶段本不是必要的，但是由于项目启动稍微晚了点，数据出现了爆发性增长，导致该系统的数据表太大（单表不带索引过500GB），原Oracle数据库已经快撑不住了。为了安全起见，先在Oracle中从一个主表以会员ID进行hash运算后再进行水平拆分，从1个表分拆成了16个。附表由于访

访问量稍小，而且全部是根据主键访问，暂时保留原样。

当然，这样的水平拆分，必然会带来数据访问路由以及数据合并的问题。我们专门为此开发了具有分布式数据库路由/数据合并，数据库读写分离，数据库链接管理等功能的数据访问中间层，专门解决拆分后给应用服务器带来的影响，使得应用服务器完全感受不到后端数据库的变化。

这个数据访问中间层，对前端应用服务器来说，就是一个完整的数据库，所有数据请求都从这里实现，以协议的方式和前端应用服务器的jdbc驱动进行交互，以便让数据库对应用服务器彻底透明。

Oracle 迁移至 MySQL（16拆128）

这个阶段是整个阶段中历时最长，复杂度最高，风险系数最高的，未知因素也最多的一个阶段。虽然MySQL数据库已经在互联网行业占据了大片江山，但是对于阿里巴巴来说，却仍然是一个新鲜玩意儿，因为之前我们一直

都用 Oracle 来提供所有的业务系统的数据库服务。

在此之前，我们从来没有在如此核心业务系统的数据库上使用过 PC Server 和本地硬盘来承载数据库，一直是使用 IBM 小型机和中高端存储设备来解决高性能和高可靠的问题。在更换成 PC Server 和本地硬盘来承载数据库之后，我们就必须面对 PC Server 本身硬件可能存在的不可靠性所带来的 Crash，所以我们必须有一套完善的 HA 切换机制，要比小型机厂商所提供的商业 HA 管理软件更加高效更加自动化更加可控，才能我们降低了设备本身可靠性之后达到原有的可用性要求。

对于一个需要满足 365 * 24 * 7 的核心业务系统来说，肯定是不可能给我们太多时间来进行数据迁移的，所以我们不得不设计出一个对现有系统影响尽可能小的迁移方案，这势必会造成方案的高度复杂化，带来更多的风险。最后的迁移方案要经历如下 4 个阶段：

1. Oracle 读/写；MySQL 初始化并增量写

2. Oracle 读/写；MySQL 写

3. Oracle 写；MySQL 读/写

4. Oracle 停访问；MySQL 读/写

当然，也正式由于有如此复杂的方案，才确保了在整个迁移过程中的的停机时间被控制在 10 分钟之类。

附属 Detail 信息迁移至 MySQL

从项目开始，至完成主表拆分结束，已经接近 2 年了。这 2 年时间内，数据量一直都在飞涨，这让即使仅仅只是按照主键访问的附表也快无法承受持续增长的业务压力，附表的拆分也就成了必行之势。由于在原来主表拆分的过程中，整个项目组已经积累了大量的经验，附表拆分过程非常顺利，基本没有出现任何问题。虽然附表的拆分过程与主表相比除了 1 拆 16 这个阶段外没有减少其他任何环节，但是整个拆分过程也才 2 个月就全部搞定了。

这个迁移项目算是彻底完成了，但是我们的迁移之路并不会就此止步，还有很多的系统仍然存在扩展性问题，还有很多的数据库应用等着我们去拆分。

注：同事们还为此送了我们一个虽不太雅但也意思相近的名称“拆迁队”。■※

原文：

<http://isky000.com/database/core-database-system-migration-from-oracle-to-mysql>

推荐专题资源：

[企业内网开发环境部署与管理全攻略](#)

[系统运维秘诀大分享](#)

[FreeBSD 入门指南](#)

推荐文章：

[FreeBSD 8 下如何最有效率的安装软件](#)

[安装 FreeBSD8 服务器后建议做的事](#)

[Linux 系统管理员都应该熟悉的工具](#)

[FreeBSD+Nginx+PHP 配置高性能 Web 平台](#)

[精简语句吧，让你的 MySQL 更有效](#)

[浅析 MySQL 数据碎片的产生](#)

[MySQL 数据库的优化（上）单机 MySQL 数据库的优化](#)

[MySQL 数据库的优化（下）MySQL 数据库的高可用架构方案](#)

rman 比较适合于跨文件系统的迁移，如同平台下的不同文件系统。

数据库迁移几种方式简单介绍

文/小荷

我们常常需要对数据进行迁移，迁移到更加高级的主机上、迁移到远程的机房上、迁移到不同的平台下.....

一、exp/imp

这也算是最常用最简单的方法了，一般是基于应用的 owner 级做导出导入。

操作方法为：在新库建立好 owner 和表空间，停老库的应用，在老库做

```
exp user/pwd owner=XXX
file=exp_xxx.dmp log=exp_xxx.log
buffer=6000000
```

传 dmp 文件到新库，在新库做

```
imp user/pwd fromuser=XXX touser=XXX
file=exp_xxx.dmp log=imp_xxx.log
ignore=y
```

优缺点：优点是可以跨平台使用；缺点是停机时间长，停机时间为从 exp 到网络传输到新库，再加上 imp 的时间。

二、存储迁移

这种情况下，数据文件、控制文件、日志文件、spfile 都在存储上（一般情况下是裸设备），我们可以直接把存储挂到新机器上，然后在新机器上启动数据库。

操作方法：将老库的 pfile（因为里面有指向裸设备的 spfile 链接）tnsnames.ora，listener.ora，密码文件传到新库的对应位置。将存储切至新机，或者用文件拷贝或 dd 的方式复制数据文件，启动数据库。

优缺点：优点是该迁移方式非常简单，主要的工作是主机工程师的工作，dba 只需配合即可，停机时间为当库、切存储、起库的时间缺点是要求新老库都是同一平台，是相同的数据库版本。

三、利用 data guard 迁移

用 dg 我们不仅可以用来做容灾，物理的 dg 我们还可以作为迁移的方式。

操作方法：可见[这篇文章](#)或者[这篇文章](#)或者其他相关网文。注意 switch over 之后，可以将 dg 拆掉，去掉

```
log_archive_dest_2
FAL_SERVER
FAL_CLIENT
standby_file_management
```

等参数。另外还要注意如果用 rman 做 dg，注意手工添加 tempfile。

优缺点：优点是停机时间短，停机时间为 switch over 的时间。缺点：主机必须双份、存储必须双份。

四、用 rman 做迁移

rman 比较适合于跨文件系统的迁移，如同平台下的不同文件系统。

操作方法：

1. 停第三方的归档备份，如 legato 或 dp

2. backup 数据库：

```
run {
  allocate channel t1 type disk;
  backup full format
  '$DIR_BAK/UNDOTBS1_2_%d_%s_%p.bak'
  datafile 2;
  .....
  release channel t1;
}
```

3. 备份控制文件

```
alter database backup controlfile to
'/tmp/mydb.ctl';
```

并到新数据库用 rman 恢复：

```
restore controlfile from
'/arch/sd168.ctl';
```

4. restore 备份文件：

```
run {
  allocate channel t1 type disk;
  restore datafile 2;
  releasechannel t1;
```

```
}
```

5. 传归档日志，并且对归档进行做 recover：

```
recover database until sequence = 归档
的序号 thread = 1;
```

6. 对数据库 open resetlogs：

```
RMAN>sql 'alter database open
resetlogs';
```

7.

```
alter tablespace temp add tempfile
'XXXXXX' size XXM reuse;
```

优缺点：优点是可以跨文件系统，停机时间少。缺点是要时刻关注这归档日志，做 recover 的时候一个都不能少！※

原文：

<http://www.oracleblog.org/working-case/some-method-for-migrate/>

相关推荐：

[使用 Microsoft Azure 让云迁移变得简便](#)

[VMware 迁移的教训：为什么备份如此重要](#)

[Domino 系统从 UNIX 平台到 windows 平台的迁移及备份](#)

[精通 Xen 虚拟机迁移技术](#)

Linux 新书推荐



《Linux 指令范例速查手册》

Linux 继承了 UNIX 强大而灵活的命令行工作方式。Linux 中常见的指令有好几百个，不管是初学者还是 Linux 专业人员，面对如此庞大的指令库，都需要一本比较全面的 Linux 指令查询书籍。本书共分 3 篇，讲解了 459 个 Linux 指令，给出了近 700 个典型示例。第 1 篇介绍了 172 个 Linux 基础指令；第 2 篇介绍了 200 个 Linux 系统管理指令；第 3 篇介绍了 87 个 Linux 网络管理指令；附录给出了按英文字母排序的 Linux 指令索引。 [查看>>](#)

本来迁移 DNS Server 就不是一件很有技术含量的工作，只不过各种手续比较繁琐，而且还要等待 DNS 解析生效。

DNS Server 迁移笔记

文/Johnny

本来迁移 DNS Server 就不是一件很有技术含量的工作，只不过各种手续比较繁琐，而且还要等待 DNS 解析生效。索性就把迁移过程简单记录下来，也算给大家指个路。

假设：在万网 (www.net.cn) 申请了几个域名

a.com/b.com/c.com

默认为 a.com/b.com/c.com 负责解析的 DNS Server 是万网的 DNS Server

dns1.hichina.com/dns2.hichina.com

现在：要把负责解析 c.com 的 DNS Server 由

dns1.hichina.com/dns2.hichina.com

改为自己建立的 DNS Server (安装 Bind9 Name Server) ，当然该 DNS Server 要托管在 IDC 机房或有互联网接入的地方。

步骤如下：

假设我们自己建立的 DNS Server 全称是 ns1.a.com 和 ns2.a.com。

登录 www.net.cn ，进入 a.com 的域名管理部分，选择“创建域名服务器”，分别建立 com 级的 ns1.a.com 和 ns2.a.com 即可，不过每个要收费 75.00 (不爽!!!)。

.com 域名和 .cn 域名的顶级域名管理 Server 分属不同机构，因此如果要迁移 .cn 域名，还需要创建 2 个 .cn 的域名服务器。

之所以在这里创建域名服务器，其实就是让万网通知 internic，要注册一个新的 DNS Server，这样别人访问你的网站的时候，.com 顶级 DNS Server 才能找到你自己的 DNS Server。

创建以后大概经过 9-12 小时即可生效，可以通过访问

<http://www.internic.net/whois.html>

查询 (选择 NameServer) 注册的 DNS Server 是否存在，如果存在就表示注册成功了；否则，会提示找不到，表示 DNS 还没有更新到 internic。

ns1.a.com 和 ns2.a.com 一旦注册完成即可以登录 www.net.cn，修改某个域名的 DNS Server，把 DNS Server 修改成自己建立的 DNS Server 即可，当然这里应该修改成 ns1.a.com 和 ns2.a.com。

大概 9-12 小时后 DNS 迁移即可生效，至此 DNS 迁移完成。■※

原文：

http://blogold.chinaunix.net/u/3959/showart_125185.html

每个基础架构都各有所异，所以在将服务器迁移到虚拟环境中去这个过程没有现成的计划可以仿照，但是还是有一些你可以遵循的规则。

实战：如何实现从物理到虚拟基础架构迁移

文/ Paul Venezia
编译/ 哲婷

在进行小型企业基础架构虚拟化的过程中，看上去让一切井然有序地运行起来并非一件容易的事情。然而，在很多情况下，这个过程中最困难的部分其实是把所有预算综合在一起，然后用有限的资金去选择必要的硬件和软件设施——而开始着手进入虚拟化则相对而言比较简单。

从物理基础架构迁移到虚拟基础架构，最重要的是，必须在迁移某台机器或者在投入生产之前，甚至在你开始测试之前，确保所有的部件都准备就绪。就像组装一张从宜家买的桌子一样，在进行安装之前把所有的工具都准备好，这样会让工作进展得更为顺利。所以，在

进行这次艰难的虚拟化之旅之前，确认准备好每一件必备的东西，这会让整个过程更为顺利和快捷，也会大大提高成品的质量。

为此，充分认识你所选择的虚拟化方案的功能和局限性至关重要。在某些情况下，有限的预算可能不能让你享受到一些高端的功能，所以你必须对这些功能进行了解以后做出合理的让步。

例如，你可能会需要主机之间的实时虚拟服务器迁移许可，但是可能就要舍弃自动负载平衡或者高可用性，或者不得不放弃高级内存优化和类似的功能。

在第一种情况下，你需要在多台物理主机中间手动平衡虚拟服务器，并且手动连接和重启这些服务器需要关闭一台物理服务器。在后一种情况下，你就需要为每台物理主机准备更多的内存否则高级内存共享就无法使用。

还有一些其它的例子，不过以上这些都是比较常见的。在规模较小的基础架构中，缺少这些功能可能无关紧要，因为由于虚拟服务器的数量较少，而且它们一般不会遇到负载不平衡或者高变量的工作负载。无论怎样，在开始工作之前，对你手里的东西做一下充分的了解是十分必要的。

建设网络

拥有足够的物理服务器处理能力、以太网交换机和足够的存储是极为重要的。在市场上有很多物美价廉的存储设备，它们可以处理虚拟化的工作负载以及多核服务，而且他们的价格非常合理。

如果可能的话，无论选择何种方案，你最好是都有一个合理的冗余方案，比如冗余电源供给和给予保护的冗余磁盘阵列（RAID）水

平，至少是 RAID5。如果基础架构非常小，没有什么存储共享计划，那么为物理主机服务器或者服务器配备电源支持的磁盘阵列控制器就非常重要了，最好是选择位于服务器内部的 RAID6。

还需要注意的是，如果你放弃了共享存储，那么你就无法利用类似于实时迁移的功能，你也不能快速启动依存于故障物理主机内本地存储的虚拟服务器。此外，在以太网交换机方面确保你的交换器能够链路聚合。

一旦处理好这些问题，建设网络就变得轻而易举了。对于一个共享存储的解决方案而言每台物理主机都应该有至少四个网络接口：两个为故障转移配置——以便在紧急情况下切换到备份系统；另外两个则是为前端的链路聚合而配备的。对于非共享部署，你可以只选择两个聚合的前端接口。

为了预防任何单线的故障，你应该像配置多条网络连接路线一样配置存储阵列。

一旦网络建设完成，你就可以准备在物理主机上着手进行虚拟化，然后在合适的时候把它连接到你的共享存储中去了。

处理虚拟化迁移

每个基础架构都各有所异，所以在将服务器迁移到虚拟环境中去这个过程没有现成的计划可以仿照，但是还是有一些你可以遵循的规则。

首先需要考虑的问题是使用物理到虚拟 (P2V) 迁移工具。这些工具从很多供应商那里都可以买到，它们也有可能包括在你所选择的虚拟化产品之中。

这些工具有优劣之分，不过大多数服务器都可以通过此类方式成功迁移，节省你的时间和精力。不过，有的服务器上可能安装了什么特别的软件；有的服务器可能要求使用硬件的验证方式；有的物理零件（如 MAC 地址）上可能绑定了一些许可协议……等等。在此类服务器面前，P2V 的方式可能反而比重建虚拟服务器更加麻烦，但是这个只能通过实际操作来验证。

当然，在大多数情况下，你都可以选择尝试 P2V 进行服务器迁移而不受到任何物理服务器问题的干扰。而且，如果迁移失败，重新启动物理服务器后不至于导致数据丢失。

这就是说，在任何迁移进行之前，确保先对你的备份进行测试。在某些可能出错的地方要经常保留备份计划。

有些服务器的迁移是不应该使用 P2V 工具的。最常见的例子是 Windows 域控制器。相比之下，在一台虚拟服务器上建立一个新的域控制器并把它作为一个完整的域名服务器的做法是比较合理的。

保留一个单独的物理服务器作为域控制器也是一个不错的办法，这样不至于让所有的域控制器都被虚拟化。这种做法不是必须的，但是，这么做缺乏高可用性特点，却能为未来提供一个相当安全的网络。

其它服务器可以使用 P2V 进行迁移，或者简单地进行虚拟服务器重建。在某些情况下，重建服务器是清除旧的物理服务器遗留下来的碎片的不错的办法，可以为你提供一个干净的过渡环境。记住，使用 P2V 来迁移物理服务器不可能解决任何现有的问题，有时候可能让它们变得更糟。不过，你仍然可以选择一直尝试 P2V，而把重建服务器作为备用方案。

重要的是要保持 IP 地址以及物理和虚拟服务器状态的记录。在你使用 P2V 的时候，确保不会出现物理服务器和它的虚拟分身同时运作的情况。P2V 过程保留了物理服务器的整个状态，包括名称、域成员和 IP 地址。因此如果两者同时运行将会制造很大的麻烦。最好的办法是关闭物理服务器，然后启动新的虚拟服务器。

将物理服务器基础架构转换到虚拟基础架构的过程并非只是一夜之功。事实上，它也不能够一味追求速度。你可以从某一点入手，比如选择一到两个物理服务器进行转移，让它们以虚拟服务器的状态运行一段时间，这样可以确定它们的可行性。你可以每天或者每周转换一到两台服务器——一般情况下，没有必要尝试一次性完成全部的转换。

通力合作

将虚拟化与软件或者操作系统的升级结合起来可能会让你受益匪浅，这样做可以让你在留有现有基础架构退路的情况下同时测试新的虚拟化平台和新服务器的预期行为。

这也可以让你开始使用新的解决方案，就像是在与物理世界中形形色色的问题做斗争的战场上呼吸到的一丝新鲜的空气。

最后，在迁移的过程中，一定要在重组的某些问题上花些时间，确保一切按照原计划进行。另外，应该保证你的计划中包含关于新的虚拟基础架构备份的实施和测试。

一旦你已经完成了所有的转换和重建，你可能会惊叹自己怎么可能会曾经生活在没有虚拟化的环境里。届时，所有在迁移过程中所付出的艰辛和对于这个过程持有的怀疑态度都将消失殆尽。■※

原文：

[How to Move From a Physical to Virtual Infrastructure](#)

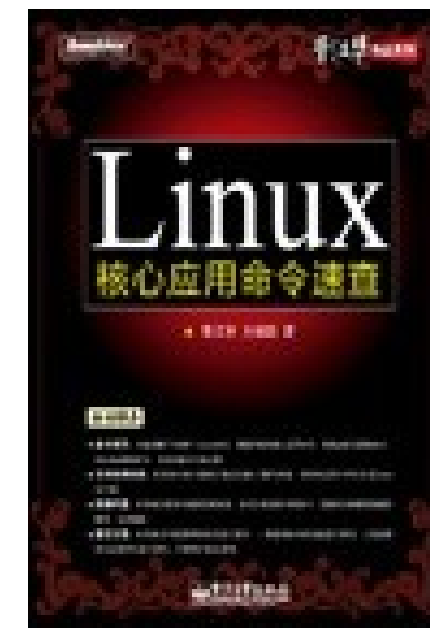
译文：

<http://virtual.51cto.com/art/201011/234844.htm>

相关推荐：

[在 Debian 下通过 OpenVZ 实现虚拟化入门精华中的精华](#) [20 个最基本的虚拟化技巧](#)

Linux 新书推荐



《Linux 核心应用命令速查》

本书汇集了 Linux 命令行下核心管理命令（包括最新的虚拟化管理命令、SELinux 管理命令）的功能说明、语法说明、选项介绍、典型应用实例和注意事项等，对每一个命令都做了非常详尽的介绍，并列举了大量的实例进行说明，可以使读者对 Linux 下的命令有快速深入的认识。全书按照 Linux 命令的功能进行分类，便于读者查询。 [查看>>](#)

dsh 是专为在远程系统上运行 Shell 命令设计的工具，通过 dsh 可以简化对大量计算机的操作。

通过 dsh 批量管理 Linux 服务器

文/张勤

目前在企业网络中越来越多的出现 Linux 服务器，而如何方便高效的管理大量的 Linux 服务器是系统管理员非常关心的一个问题。现在有大量的开源管理工具，可以实现这样的管理工具，现在给大家介绍一个通过命令行有效地管理大量 Linux 的工具——dsh。

dsh 是专为在远程系统上运行 Shell 命令设计的，通过 dsh 可以简化对大量计算机的操作。dsh 命令语法如下：

```
dsh [-m machinename | -a | -g
groupname] [-f machinefile] [-M] [-q]
[--wait-shell]--commandline
```

常用选项：

-M：在显示远程命令执行的输出时，在前面加上主机名。

-a：如果经常操作同一组计算机，可以创建一个全局集合的组。

\$HOME/.dsh/machines.list 文件是全局集合的定义。在该文件中每行一个计算机的 IP 地址，在指定 -a 后，dsh 就会在 machines.list 中列出的所有计算机上执行指定的命令。

-q：指定使用安静模式输出。

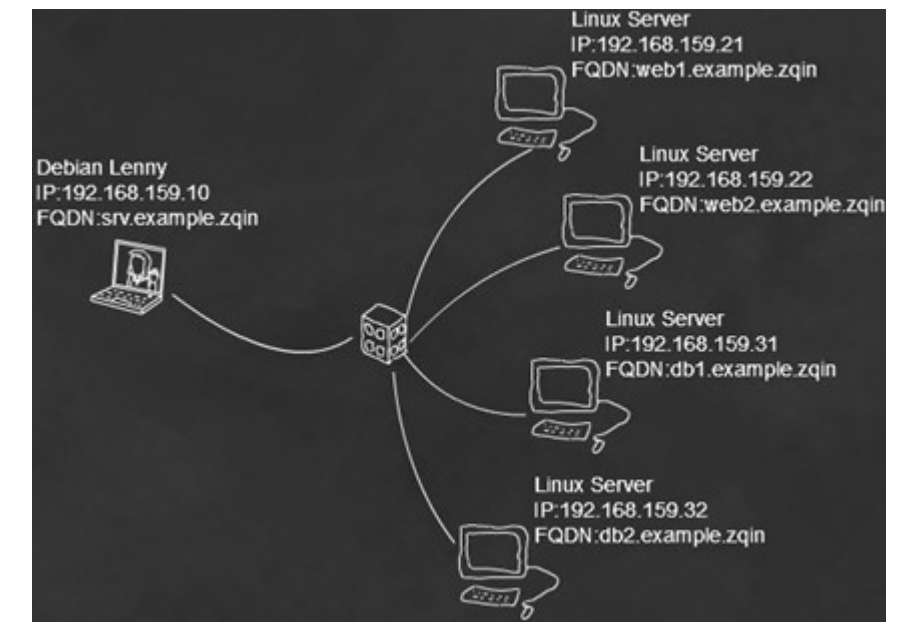
-m machinename：指定需要执行指定命令的计算机。

-g groupname：指定需要执行指定命令的计算机组，主机名组在 \$HOME/.dsh/group/ 目录是定义，每个计算机组一个文件，文件名即是组外，在文件中每行一个计算机 IP 地址。

-f machinefile：指定计算机列表文件。

-wait-shell：在默认情况下，dsh 是并行地在计算机上运行命令。如果希望顺序地运行命令则指定 --wait-shell。

下面在我们一起来看看在如下图的网络中如何通过 dsh 有效的管理 Linux 服务器。



1、dsh 是通过 SSH 方式连接到服务器，所以需要在所有服务器上安装 SSH。

2、在 srv.example.zqin 上通过如下命令安装 dsh。

```
srv:~# apt-get -y install libdshconfig1
libdshconfig1-dev dsh
```

3、在使用 dsh 进行管理时，需要输入被管理服务器的用户名及密码，为了使用起来更加

方便可使用如下命令将 SSH 的公钥复制到被管理服务器。

```

srv:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
(/root/.ssh/id_rsa):
Enter passphrase (empty for no
passphrase):
Enter same passphrase again:
Your identification has been saved
in /root/.ssh/id_rsa.
Your public key has been saved in
/root/.ssh/id_rsa.pub.
The key fingerprint is:
dd:e9:d3:84:fc:4c:ff:b4:b0:fa:12:fa:fd:
49:3d:4d root@testsrv
The key's randomart image is:
+--[ RSA 2048 ]-----+
| |
| |
| |
| . o o |
| S . = o E|
| o * oo|
| . +. +o=|
| . ...+. =|
| .o+++.+.|
+-----+

```

```

srv:~# scp ~/.ssh/id_rsa.pub
192.168.159.21:/root/.ssh/authorized_k
eys
srv:~# scp ~/.ssh/id_rsa.pub
192.168.159.22:/root/.ssh/authorized_k
eys
srv:~# scp ~/.ssh/id_rsa.pub
192.168.159.31:/root/.ssh/authorized_k
eys
srv:~# scp ~/.ssh/id_rsa.pub
192.168.159.32:/root/.ssh/authorized_k
eys

```

如果被管理的服务器比较多也可以编写个脚本来复制公钥。下面是一个复制公钥到多个服务器上的脚本。

```

for i in $(seq 200 253)
do
ssh 192.168.159.$i -C mkdir /root/.ssh
scp ~/.ssh/id_rsa.pub
192.168.1.$i:/root/.ssh/authorized_key
s
done

```

4、为了方便使用 dsh，可以将所有被管理服务器分类并存放对应文件中。

将所有被管理服务器的 IP 地址(或 FQDN)加入 \$HOME/.dsh/machines.list 文件中(每行一个)。

在 \$HOME/.dsh/group/ 目录下建立名为 web 的文件，并将 web1、web2 的 IP 地址(或 FQDN)加入其中(每行一个)。

在 \$HOME/.dsh/group/ 目录下建立名为 db 的文件，并将 db1、db2 的 IP 地址(或 FQDN)加入其中(每行一个)。

在上述配置完成后就可以在 srv 上通过 dsh 进行批量操作了，下面我们一起来看几个例子。

1、在 db1.example.zqin 上执行 reboot 命令。

```

srv:~# dsh -M -m db1.example.zqin --
reboot

```

2、在 \$HOME/.dsh/machines.list 文件中定义的所有服务器上同时执行 updatedb 命令。

```

srv:~# dsh -M -a -- updatedb

```

3、在 \$HOME/.dsh/group/web 文件中定义的所有服务上面同时执行命令。

```

srv:~# dsh -M -g -- /etc/init.d/apache2
restart

```

原文：

<http://os.51cto.com/art/201103/249087.htm>

假如想要发现哪个进程使用了大量内存的话，我通常会使用 `htop` 或 `top` 而非 `ps`。

用 ClusterSSH 管理多台 Linux 服务器

文/ Bill Childers
译/ 黄永兵

Cluster SSH 是一个可以用来通过 SSH 协议同时管理多台远程计算机的工具。它非常适合用来快速配置一个集群中的所有运行相同服务和具备相同配置的计算机节点。现在有大量的开源管理工具，都可以实现这样的管理，比如 `dsh`、`SUSE Manager` 等。下面是用 ClusterSSH 管理多台 Linux 服务器的具体过程。

如果你是一名 Linux 系统管理员，那你每天一定会和许许多多的机器打交道，因为你要定期监测和维护这些机器，如一批 Web 服务器，如果你要同时在一台或多台机器上敲入相同的命令，你可能会通过 SSH 登录，然后逐台敲入，如果使用 ClusterSSH，可以为你节省不少类似的工作时间。

ClusterSSH 是用 Tk/Perl 包装 XTerm 和 SSH 后形成的新工具，就其本身而言，它可以运行在任何兼容 POSIX 的操作系统上，我曾经在 Linux, Solaris 和 Mac OS X 上运行过它，它需要 Perl 库 Tk(在 Debian 或 Ubuntu 上就是 `perl-tk`) 和 `X11::Protocol`(在 Debian 或 Ubuntu 上就是 `libx11-protocol-perl`)，此外，`xterm` 和 `OpenSSH` 是必不可少的。

安装

在 Debian 或 Ubuntu 上安装 ClusterSSH 是相当简单的，只需要敲入 `sudo apt-get install clusterssh` 就可以安装好，至于依赖包你也不必担心，一切都会为你装好的，它也提供了适合 Fedora 的 rpm 包，在

FreeBSD 上可通过 `port` 系统安装，还为 Mac OS X 准备了 MacPort 版本，因此你可以在你的苹果电脑上安装 ClusterSSH，当然，如果你是极客，也可以下载源代码自己编译。

配置

可以通过 ClusterSSH 的全局配置文件 `/etc/cluster`，或用户 home 目录下的 `.cshrc` 文件来配置它，我喜欢用户级的配置方式，这样同一个系统中的不同用户可以根据自己的喜好进行配置，ClusterSSH 定义了一个“cluster”机器组，你可以通过一个界面来控制这个组中的所有机器，在配置文件的顶端“clusters”部分，你可以详尽地列出你的集群，然后用独立的段落来描述每个集群。

例如，假设我有两个集群，每个集群由两台机器组成，“Cluster1”由“Test1”和“Test2”两台机器组成，“Cluster2”由“Test3”和“Test4”两台机器组成，`~/.cshrc`(或 `/etc/cluster`) 配置文件的内容看起来应该是：

```
clusters = cluster1 cluster2
cluster1 = test1 test2
cluster2 = test3 test4
```

你也可以创建中间集群(包含其它集群的集群),如果你想创建一个名叫“all”的集群包含所有的机器,有两种实现手段,首先,你可以创建一个包含所有机器的集群,如:

```
clusters = cluster1 cluster2 all
cluster1 = test1 test2
cluster2 = test3 test4
all = test1 test2 test3 test4
```

但我更喜欢的方法是使用一个包含其它集群的中间集群:

```
clusters = cluster1 cluster2 all
cluster1 = test1 test2
cluster2 = test3 test4
all = cluster1 cluster2
```

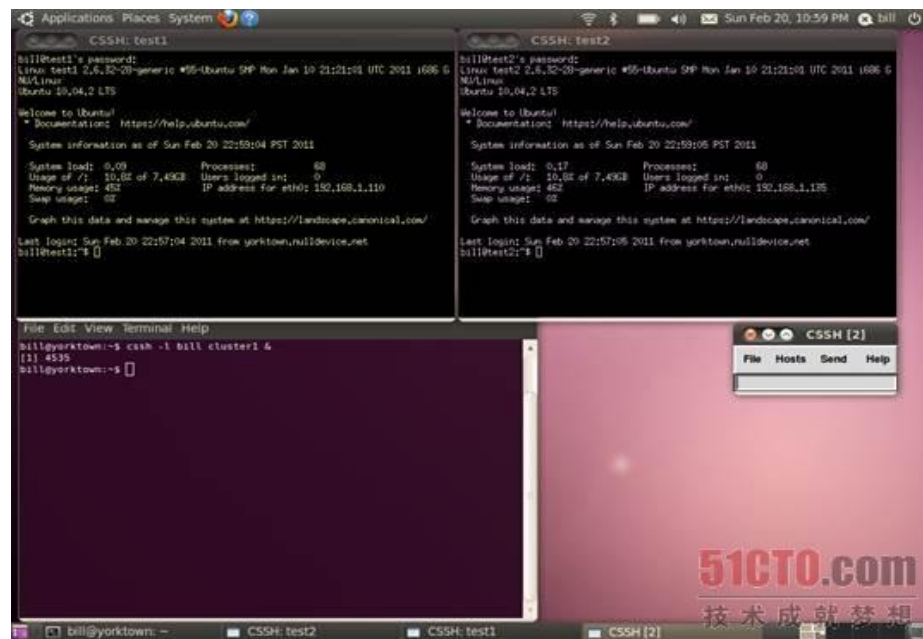


图 1 启动中的 ClusterSSH

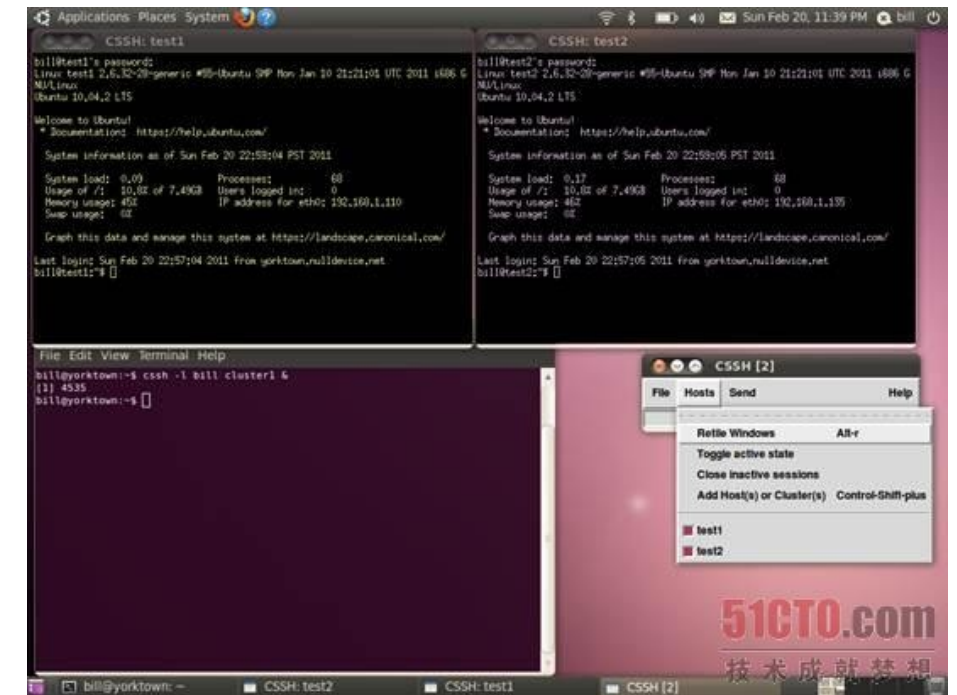
通过调用包含 cluster1 和 cluster2 的“all”集群,这些集群发生的任何变化都能自动捕捉到,因此你不必更新“all”定义,当.csshrc 文件变得很大时,此举可以帮你节省大量的时间。

使用 ClusterSSH

ClusterSSH 的使用方法和 SSH 类似,只需要运行 `cssh -l <用户名> <集群名>` 就可以启动 ClusterSSH,并以你输入的用户名登录到集群,在下图中,你可以看到我已经登录到“cluster1”集群,窗口标题为“CSSH[2]”的小窗口是 ClusterSSH 的控制台窗口,在它里面敲入的任何命令都会回显在集群中的所有机器上,在这个例子中是“Test1”和“Test2”,必要时,你也可以登录到.csshrc 文件中未列举的机器,使用的命令是 `cssh -l <用户名> <机器名 1> <机器名 2> <机器名 3>`。

如果我想向终端发送点什么,只需要点击目标 Xterm,切换焦点,输入想要的内容即可。ClusterSSH 提供了许多有用的菜单项,在管理混合型机器环境时特别有效,如下图所示,

在 ClusterSSH 控制台的“Hosts”菜单中,有许多方便的选项可以调节。



“Retime Windows”选项只有当你手动调整了窗口大小或移动了位置时才会显示,如果你想添加其它机器或集群到运行中的 ClusterSSH 会话中,“Add Host(s) or Cluster(s)”就派上用场了。最后,你将会在“Hosts”菜单的底部看到每个主机的列表,通过主机名前的复选框,你可以选择 ClusterSSH 控制台在那台机器上执行,如果你想排除某台主机,这个功能特别管用。最后我想介绍的是位于“Send”菜单下的

“Hostname”选项，它可以在命令行中回显每台机器的主机名，相信你一定会用得着。

使用 ClusterSSH 时需小心

和许多 UNIX 工具一样，如果你不小心使用 ClusterSSH，它也可能会犯下可怕的错误，我就曾看到过因 Apache 配置文件文字输入和排版的一个小失误，导致全部 Web 服务器当掉的故事——因为 ClusterSSH 老实地传播了错误。

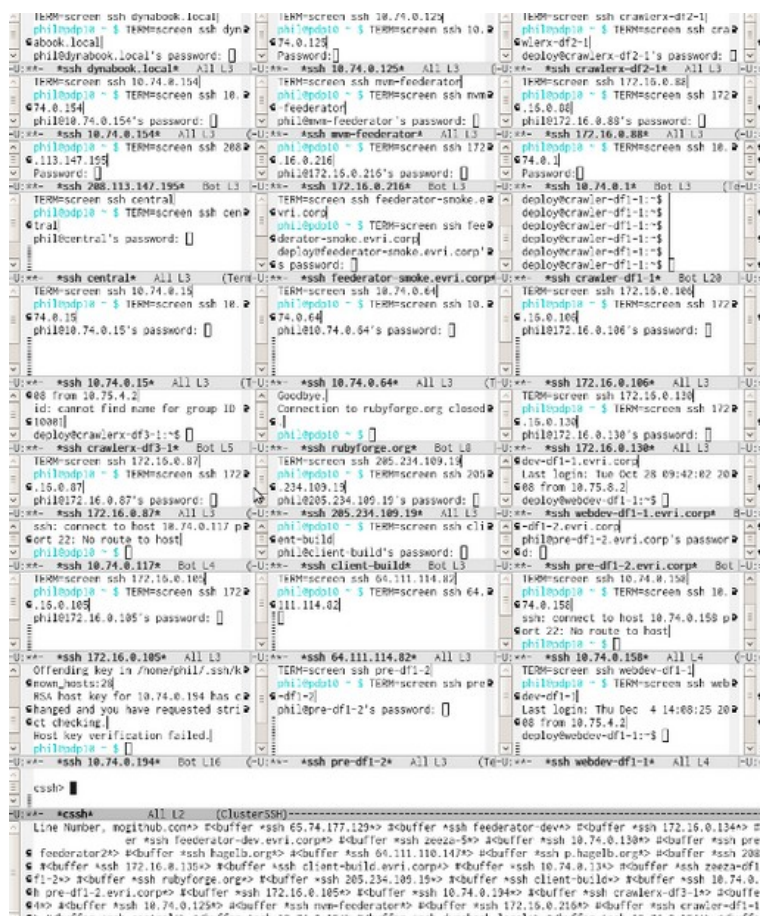


图 3 如果不小心犯下错误的话.....

特别是用特权用户操作 ClusterSSH 时更应该小心，一个小小的错误可能会引起巨大的损失，请谨记，在按下回车键之前，一定要仔细检查你所做的一切。

小结

ClusterSSH 不是配置管理系统的替代品，也不能代替管理多台机器时的最佳实践，但如果你需要在这些常用工具箱或程序之外做一些事情，或你在执行一些标准性工作，ClusterSSH 是不可缺少的，当你需要在多台机器上重复执行相同的任务时，它可以为你节省大量的时间，但和任何强大的工具一样，如果使用不当，也会带来许多危险。❗

原文：

[Managing Multiple Linux Servers with ClusterSSH](http://managing-multiple-linux-servers-with-cluster-ssh)

译文：

<http://os.51cto.com/art/201103/250014.htm>

相关推荐：

[用脚本简化部署 DenyHosts 防 SSH 暴力破解](#)

[25 个必须记住的 SSH 命令，你用过了吗？](#)

[20 个 OpenSSH 服务器最佳安全实践](#)

Linux 新书推荐



《鸟哥的 Linux 私房菜

基础学习篇 (第三版)》

本书是最具知名度的 Linux 入门书《鸟哥的 Linux 私房菜基础学习篇》的最新版，全面而详细地介绍了 Linux 操作系统。全书分为 5 个部分：第一部分着重说明 Linux 的起源及功能，如何规划和安装 Linux 主机；第二部分介绍 Linux 的文件系统、文件、目录与磁盘的管理；第三部分介绍文字模式接口 shell 和管理系统的好帮手 shell 脚本，另外还介绍了文字编辑器 vi 和 vim 的使用方法；第四部分介绍了对于系统安全非常重要的 Linux 账号的管理..... [查看>>](#)

为了找到答案，我将创建一个 Python 索引系统，然后使用 CairoPlot 工具来测绘出一幅扇形图。

CairoPlot 让 Linux 服务器的日志文件更直观

文/Akkana Peck
译/核子可乐

确实有些 Linux 服务器管理员很享受阅读及核对日志文件的艰辛过程，为什么不创建一个美观的列表及图形体系来突出那些故障和问题，而非要受这份罪呢？试试这款优秀的工具——CairoPlot 吧，它会提供给你美观且信息可视化的服务器日志文件分析途径。

作为一个需要整天跟数据打交道的从业者，我一直致力于寻找更好的方法来将纷繁复杂的数据显示为列表和图形，尤其是利用 Python 来实现这一目标。时下存在很多利用 Python 制作的整合软件包可供使用，但如果你希望输出的结果不会因为粗糙的视觉效果而遭到那些

苹果使用者们的耻笑，那么我向你强力推荐 CairoPlot。

CairoPlot 并未像大多数发行版软件那样进行封包，但它的安装过程依然简便易行。目前在 CairoPlot Launchpad page 上提供的最新版本是 1.1 版。你可以在那里下载到 `cairoplot-1.1.tar.gz` 这个文件，或是根据你的喜好，从 BZR 上搜索（一旦 1.2 版本发布，CairoPlot 项目可能会整体转移到 Sourceforge.net 上）。

解压压缩包：

```
$ tar xvf cairoplot-1.1.tar.gz
```

然后复制下面这个文件：`cairoplot-1.1/CairoPlot.py`，并粘贴到你开发的 Python 脚本所在的目录下。

用扇形图说明：谁在发送垃圾邮件？

展开测绘工作之前，找到一个良好的数据源永远是我们的首要任务。针对这个项目，让我们先来分析一个 Postfix 日志文件，`/var/log/mail.info`，借以观察一系列垃圾邮件的众多来源。

通过对文件的随机检查，我们会发现有许多提出接收请求的邮件都来自一个客观上根本不存在的地址，举例说明：

```
Mar 5 15:05:45 mailserver
postfix/smtpd[29764]: NOQUEUE: reject:
RCPT from
212.199.94.45.static.012.net.il[212.19
9.94.45]: 450 4.7.1
<ex02.maccabiworld.org>: Helo command
rejected: Host not found; from=<>
to=<aiglace@mydomain.com> proto=ESMTP
helo=<ex02.maccabiworld.org>
```

我们的 postfix 服务器一般会拒绝这样的邮件，因为通常情况下它们都是垃圾邮件。配置正确的邮件服务器应该不会编造这些虚假的

地址——当然在有些配置有误的服务器上会发生这种状况。

但是这些虚假的接收请求从何而来?他们是否来自特定的一些国家?而在这些特定国家的垃圾邮件来源中,存在多少.com 类型网站,又有多少.net 类型网站?

为了找到答案,我将创建一个 Python 索引系统,然后使用 CairoPlot 工具来测绘出一幅扇形图。索引中的每个关键字都将涵盖一个顶级域,例如“.com”;而其数值则为从该类型域中发来的被拒收邮件的数量。

剖析日志文件

要填充索引系统中的词条项目,意味着我们需要对/var/log/mail.info 文件进行剖析。每封邮件的真正发出地址能够从 RCPT 中查询到;将结果应用到 Python 的 re 模块中。因为这一过程是针对 CairoPlot 的,因此我们不必遵循 Python 的描述方式,只需按照以下代码的形式表达:

```
#!/usr/bin/env python
import CairoPlot, re
MAIL_INFO = "/var/log/mail.info"
```

```
# Dictionary to store the results as
# (domain : number of rejects)
rejected = {}
# Parse mail.info to find all the
# 'NOQUEUE: reject' lines and
# figure out what top-level domains
# (TLDs) they're coming from.
f = open(MAIL_INFO)
for line in f :
    if line.find('status=sent') > 0 :
        pass
    elif line.find('NOQUEUE: reject') > 0 :
        # An attempt we rejected. Look for a
        # pattern like
        # RCPT from
        # foo.example.com[nnn.nnn.nnn.nnn]
        rcpt = re.search("RCPT from ([^[]*)\
        [[([0-9\.]+)\]", line)
        if not rcpt :
            continue
        # Now rcpt.group(1) is the reverse-DNS
        # hostname (if any)
        # from the log file, rcpt.group(2) is
        # the IP address.
        if rcpt.group(1) and rcpt.group(1) !=
        'unknown' :
            hostname = rcpt.group(1)
        else :
            hostname = None
        # Find the part after the last "."
```

```
tld = "Unknown" # default there's no
"." in the hostname
if hostname :
    dot = hostname.rfind(".")
    if dot >= 0 :
        tld = hostname[dot+1:]
    if tld in rejected :
        # We've seen this TLD before; add 1.
        rejected[tld] += 1
    else :
        # First time we've seen this TLD.
        rejected[tld] = 1
f.close()
```

在结尾部分,通过以下内容将索引系统中的“拒收”标准传达给 CairoPlot。

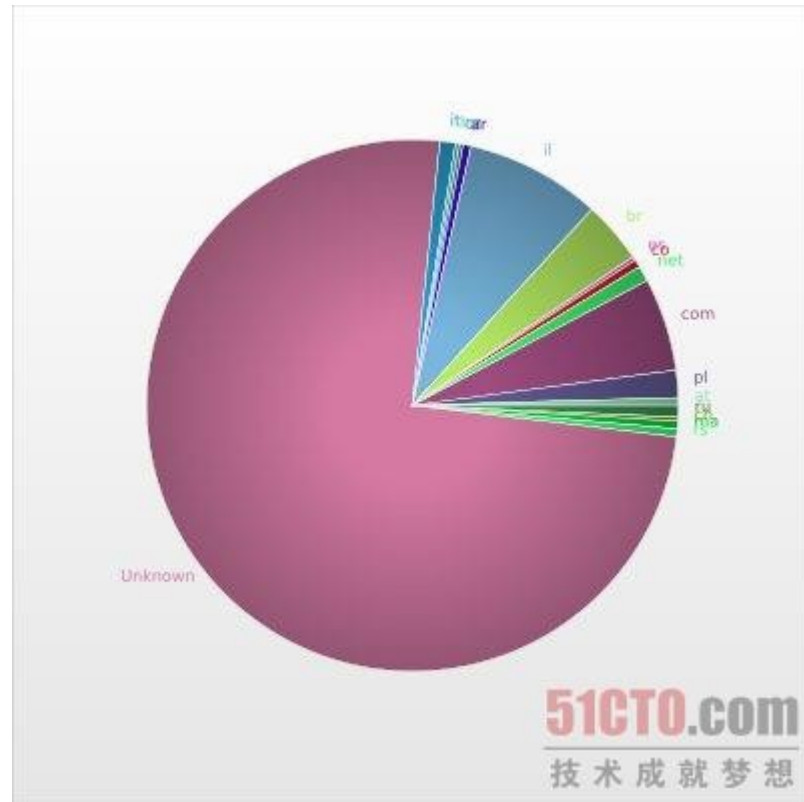
```
{'ru': 3, 'ch': 1, 'ma': 2, 'rs': 2,
'it': 4, 'hu': 1, 'cz': 1, 'ar': 2,
'il': 35, 'br': 16, 'es': 1, 'co': 2,
'net': 4, 'com': 24, 'pl': 7, 'at': 2}
```

创建扇形图

你要如何依据索引系统来创建一个扇形图?事实上一行命令即可实现:

```
CairoPlot.pie_plot("piechart",
rejected, 500, 500, None, True, False,
None)
```

CairoPlot 将生成一个名为 pie.svg 的图形文件(如图一所示)。



图一

其参数为：

```
pie_plot(name,  
         data,  
         width, height,  
         background=None,  
         gradient=False, shadow=False,  
         colors=None)
```

Name(名称)指文件名：

如果你希望加入一个诸如.jpg之类的扩展名，那么CairoPlot将使用你所设定的格式来取代svg格式，因为在某些情况下你可能会

需要一个IE用户能够通过页面正常浏览的图像。

Data(数据)：

代表索引系统中的数值。

Width(宽度)和Height(高度)：

代表你所希望的图形绘制尺寸。需要注意的是，CairoPlot预留给扇形图周边的空白区域是相当有限的，所以一定要注意整体规划。

Background(背景)：

你需要指定一种背景颜色，其选色方法为标准的RGB形式。因此，通过

```
background=(0,1,0)
```

这一指令，你将获得一个全绿的背景。你在这里也可以使用Cairo gradient(色阶)来进行设定。Gradient(色阶)功能可以让你选择是否将你扇形图中的某一块显示出色彩渐变的效果，以使整个图形更加美观。Shadow(阴影)功能将让你可以为整个扇形图增加底部阴影效果，并且如果你不喜欢系统默认的阴影颜色，也可以随意为其定义新的颜色。当然，阴影颜色同样即可以是单色也可以包含色阶。需

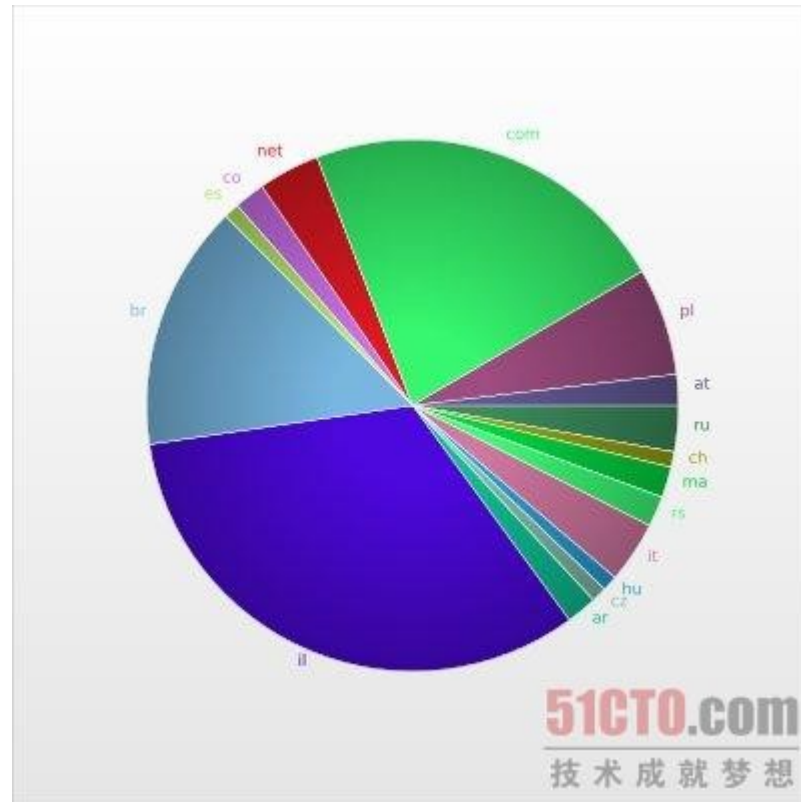
要强调的是，所选颜色的数量必须与索引系统中的项目数量相同。

图一中的示例存在一个小问题：它显示绝大多数无效的接收请求来自根本无法解析的服务器地址，而表示状况的图形被压成极细小的一块，根本无法解读。而且显然你从那一大块“无法解析”的服务器来源中根本折腾不出什么有用的信息。在这种情况下，你可以在指令中if hostname:后面添加如下内容

```
if hostname :  
    dot = hostname.rfind(".")  
    if dot >= 0 :  
        ext = hostname[dot+1:]  
    else :  
        continue
```

运行上述命令，这时扇形图变为如图二所示。很有意思吧。直到编写这个实例，我才意识到相比起其它国家，我从以色列和巴西收到了这么多的垃圾邮件。有时候一幅清晰的示意图绝对胜过千言万语。

(接下页)



图二

条形图

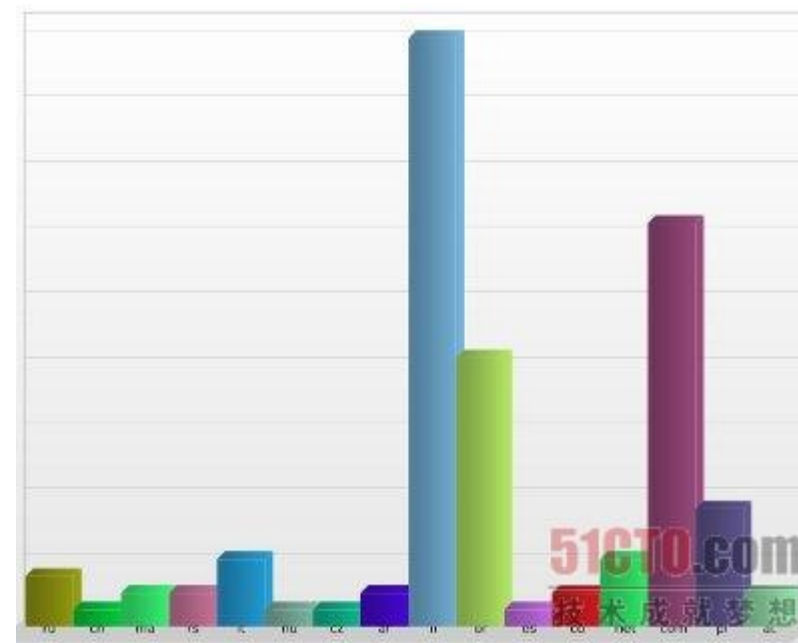
CairoPlot 也可以制作出很棒的条形图。但遗憾的是，CairoPlot 的各种规则不太适合条形图数据的导入。条形图需要的是列表，而非索引。

这都不叫事儿！只要把上文中的索引系统转换为两个列表——一个包含标签信息，一个包含具体数据——再进行条形图绘制（如图三所示）：

```
h_labels = [ k for k in rejected.keys()
]

rejlist = [ rejected[k] for k in
rejected.keys() ]

CairoPlot.bar_plot ('bars', rejlist,
500, 400, border=5,
three_dimension=True,
h_labels=h_labels)
```



图三

同制作扇形图一样，你可以导入一个颜色列表来使用自定义颜色，而且还有其它一些诸如背景、风格、圆滑边角、沙盘模型高度、象限

体积以及必然具备的标签体积和标签高度等调节选项。

当然，CairoPlot 同样可以制作其它类型的图形。这有一些实例文档，你也可以使用 Python 的交互式解析工具并输入如下内容：

```
import CairoPlot
help(CairoPlot.pie_plot)
```

CairoPlot 站点将很可能迁移至 Sourceforge 网站，并提供更加完备的访问页面。与此同时，如果你已经有过一些实践经验，你一定会深切体会到 CairoPlot 在制作美观艳丽的图形方面绝对称得上是顶尖工具之一。❖

原文：

<http://www.linuxplanet.com/linuxplanet/tutorials/7317/1/>

译文：

<http://os.51cto.com/art/201103/250253.htm>

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) 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，或加入我们 51CTO 的翻译团队！

投稿邮箱：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：双机操作。欢迎关注！

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新刊发布：

1、通过电子邮件订阅：

订阅方式请参考本杂志的专题页：

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

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。

《Linux 运维趋势》是开放的非盈利性电子杂志，其中所有内容均收集整理自国内外互联网（包含 51CTO 系统频道本身的内容）。对于来自国内的内容，编辑都会事先征求原作者的许可（八卦，趣闻&数字栏目例外）。如果您认为本杂志的内容侵犯到了您的版权，可发信至 yangsai@51cto.com 进行投诉。