2010 年 9 月 第零期（测试刊）

本期主题：运维自动化

关键字：DevOps，开源运维工具，Kickstart，Cobbler，敏捷运维

本期内容索引

Linux 运维趋势 第 0 期 运维自动化

杂志策划：51CTO 系统频道

本期主编：lazycai

本期专家：李洋

技术顾问：抚琴煮酒、王文文

特别顾问：杨文飞

Logo 制作：高鹏飞

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）

投稿信箱：yangsai@51cto.com

（注：本杂志并不单独接受投稿，所有投稿的接受对象为 51CTO 系统频道）

【人物】运维专家李洋：漫谈基于开源服务的运维自动化实现

人物专访 | 李洋

采访&编辑：杨赛

随着各种业务对 IT 的依赖性渐重以及云计算技术的普及，企业平均的 IT 基础架构规模正不断扩张。有些 Web 2.0 企业可能会需要在两个星期内增加上千台服务器，因此对运维而言，通过手动来一个一个搭建的方法不仅麻烦、效率低下，而且非常不利于维护和扩展。即使是在传统的企业当中，日常的备份、服务器状态监控和日志，通过手动的方式来进行的效率也很低，是一种人力的浪费。因此，自动化早已是每个运维都必须掌握的看家本领。

在不同的企业中，自动化的规模、需求与实现方式都各不相同，因此在技术细节层面，运维之间很难将别的企业的方法整个套用过来。然而在很多情况下，自动化的思路是有共通之处的。因此，51CTO 系统频道最近邀请了中国移动通信研究院的项目经理李洋先生，就运维自动化实现，尤其是基于开源工具的运维自动化，谈了谈自己的经验和看法。

李洋，博士毕业于中科院计算所。10 多年来一直从事计算机网络信息安全研发工作，曾主持和参与多项国家重点项目以及信息安全系统和企业信息安全系统的研发工作。具有 Linux 系统应用、管理、安全及内核的研发经验，擅长网络安全技术、协议分析、Linux 系统安全技术、Linux 系统及网络管理、Linux 内核开发等。

51CTO：能否先大致谈谈您的运维经历？

李洋：我的运维经历分 3 个阶段：

（1）靠纯手工、重复地进行软件部署和运维；

（2）通过编写脚本方便地进行软件部署和运维；

（3）借助第三方工具高效、方便地进行软件部署和运维。

这几个阶段是随着我知识、经验、教训不断积累而不断演进的。而且，第 2 个阶段和第 3 个阶段也可以说是齐头并进的。Linux 下的第三方工具虽说已经不少了，但是 Linux 下的脚本编写对运维工作的促进是绝对不可以忽视的。所以我在实践中一直是两种方式都采用。

51CTO：在 Linux 下有哪些运维工具是让您感觉特别好用、眼睛一亮的？

李洋：其实 Linux 下的运维工具不算太多，我觉得比较好用的包括 RedHat 提供的 Kickstart Installations 自动安装解决方案，不过该方案相对比较繁琐；目前有的工程师认为 Cobbler 是让人眼前一亮的好工具，个人感觉也挺不错。另外，其实 Linux 下能够方便地通过编写 shell 脚本、使用 CronTab 等方法来进行运维，个人认为这也是非常不错的选择。

51CTO：有人理解自动化就是运维为了减少重复枯燥的工作而建立的流程方法，而除此之外，自动化还能够带来减少人为错误、及时报警与故障恢复、提高业务可用性等好处。您对运维工作自动化是如何理解的？您认为自动化的技能/意识对于运维的重要程度如何？

李洋：运维工作自动化确实包含上述 2 个方面，归纳总结来其实就是：把零碎的工作集中化，把复杂的工作简单有序化，把流程规范化，最大化地解放生产力，也就是解放运维人员。

自动化的技能/意识对于运维工作至关重要。运维工作不是简单的使用工具，这里面还有很多技巧和意识。具体的技巧/意识包括：

（1）如何驾驭这些琳琅满目的工具为己所用

（2）如何根据不同的应用环境来选用不同的工具

（3）如何根据应用来组合使用工具

等等等等。一定要记住一点：工具是来帮助人进行运维的，这中间还需要人的干预和决策，工具不能代替完全的运维工作。

51CTO：自动化针对的范围可以大致分为安装自动化、部署自动化、监控自动化等方面。除此之外，还有哪些方面是您比较关注的？（比如软件发布、更新、备份等）对于这几个方面的技术实现，您一般采用哪些工具？这些工具相比其他同类工具的优势在哪里？

李洋：自动化其实还包括软件发布自动化、升级自动化、安全管控自动化、优化自动化等等。我个人比较关注管理和安全方面的技术实现，比如说 HP 和 IBM 出品的一些 ITIL 和 ITSM 产品等我都在使用，比如 HP Openview，IBM Tivoli 等等。这些工具都有 Linux 的版本，与其他同类工具相比的优势应该在于他们的商业应用成熟度，都是老品牌了。

51CTO：针对一个小规模的网站，到百万量级、千万量级的网站，您在考虑工具的选择上会有怎样的不同？

李洋：我在选择上对于百万量级、千万量级的网站尤其会考虑选择成熟的工具、性能高的工具、熟悉的工具。而对于小规模的网站，则会考虑选择一些开源的、免费的工具。这个原则就是以应用为导向，百万量级、千万量级的网站牵涉的面广、要求高，不成熟的工具往往很难说服我使用，所以主要是在成熟度方面。

51CTO：很明显，自动化的实现不是单纯学习几个工具就能够做好的，甚至于规划不好的情况，自动化不仅没有节省人力，反而带来了更多的问题。您建议运维人员在考虑自动化流程的过程中应该遵循怎样的原则或思路？

李洋：其实前面多多少少也谈到这个问题了。归纳一下，包括如下几点原则：

（1）根据应用选择工具

（2）对于关键应用，选择成熟度高的工具

（3）不能过分依赖一种工具，需要进行对比和分析

（4）对工具的特性做到精通

（5）是人驾驭工具，人要监督工具，而不是工具来驾驭人

（6）别忘了脚本。

51CTO：您推荐运维人员可以通过哪些资源、渠道来学习有关自动化方面的经验知识？

据我个人的经验，很多专业网站，比如 51CTO、网管员世界等都是很不错的媒体，我看有这方面的很多文章，运维人员可以多多涉猎和学习；另外，运维人员还可以根据选择的自动化工具，登陆该工具的网站进行学习，一般这样的网站都会有一些运维的基础知识来辅助运维人员更快地熟悉工具使用。

原文链接：

http://os.51cto.com/art/201009/224077.htm

相关阅读：

网站运维之道 之自动化管理 via dbanotes.net

怎样做一个优秀而懒惰的系统管理员 via ixpub

系统管理员与开发者界限日渐模糊 via 51CTO

【八卦，趣闻&数字】

【BTRFS】根据 Ubuntu 开发团队称，由于没有充足的时间，无法完成余下的 BTRFS 开发工作，因此也就是说在 10.10 中将不会完全支持 BTRFS 。 - wowubuntu

【ZFS】开发 Native ZFS for Linux 项目的 KQ Infotech 公司将在 9 月 15 日左右正式发布一个完全可用的 Linux Kernel ZFS 模块。- Phoronix

【Ubuntu】据说 Ubuntu 11.04 将舍弃一直以来的新立德（Synaptic）软件安装程序，而完全采用 Ubuntu 软件中心的机制。现在应用软件商店在各处大火特火，这也许是 Ubuntu 将来盈利模式的重点？ - Ubuntu

【Fedora】代号 Laughlin 的 Fedora 14 已经发布了 Alpha 版本，此次版本中，systemd 取代 upstart 成为进程管理，整合了 KDE 4.5 桌面，引入 D 语言支持，更新 Python, Perl 和 Erlang，并为 Perl 6 增加了 Rakudo 支持。 - Fedora

【Apache】到 2010 年 8 月为止，在互联网上的 213,458,815 个网站中，Apache 依然是全球最受欢迎的 Web 服务器，数量高达 670 万台。 - NetCraft

【Linux Web Server】根据 2010 年 7 月的数据，全球每十个基于 Linux 的 Web 服务器当中，就有三个 CentOS，两个半 Debian，一个半 RHEL，以及一个 Ubuntu。 - W3techs

【Servers】2007 年，Intel 约有 10 万台服务器，Facebook 约有 3 万台服务器，微软约有 13 万台服务器，而 Google 则据说已经超过了 100 万台服务器。2010 年，Google 的服务器数量到达千万了么？-Datacenterknowledge

【Varnish】Varnish 是一种状态艺术，高性能的 web 加速器。简单直白的 VCL 配置文件，和 nginx 的配置文件有几分相似，强大的监控和分析工具，还有强大的 CLI 管理工具，这些工具都可以大大的增强 varnish 的可管理和可分析性。并且由于 varnish 先进的架构，发展的趋势就是替代 squid 服务器。 - Andy Feng

【Sikuli】Sikuli 是一种利用图片（快照）去搜索和自动化 GUI 的视觉技术，Sikuli 脚本可以自动化你在屏幕上看到的一切，而这不需要内部 API 的支持。有些人说 Sikuli 看起来只是另一个按键向导或是 AutoHotkey，但其实 Sikuli 还有许多在这个直观意义之上的潜力。 - vgod(Sikuli 项目发起者)

【桌面环境】从调查可以看出来，GNOME 和 KDE 是目前的主流环境，而 Xfce、LXDE、Openbox 和 Fluxbox 等也已经得到了很高的应用。 - Distrowatch

【Cobbler】我一直怀疑 Cobbler 是中国人命名的项目，因为 PXE 发音为"pixie"(皮鞋)，而 Cobbler 的中文意思是"补鞋匠"。 - Fenng

【Linux 入门】硬体驱动程序是学习 Linux 者的恶梦。笔者认为 Ubuntu 在这方面比 SUSE 周全。在 Red Hat 或 SUSE 的环境中，对于太新或比较冷门的硬件，大多没有预设驱动程序，需要自己去找。但熟悉 Ubuntu 后，还是要去接触 SUSE 或 Red Hat，因为 SUSE 或 Red Hat 是企业级应用 Linux 的正统。

国际前沿 | DevOps

原文：What is DevOps?
作者：Damon Edwards
译者：吕欣

如果你对 IT 管理感兴趣，尤其是对 Web 运维感兴趣，那么最近一定会注意到 DevOps 这一热词的出现（注：在英文中，Developer 指开发者，Operations 指运维，而 DevOps 一词也就是开发+运维的意思）。

在许多方面，DevOps 是一个集合性概念，指的是能够理顺开发和运维之间相互配合关系的任何事物，但是 DevOps 背后的理念要比上述说法更深远。

什么是 DevOps？

人们越来越意识到传统意义上的开发行为和运维行为存在脱节现象，从而导致冲突和低效。正如 Lee Thompson 和 Andrew Shafer 所言，在开发和运维之间存在一面“混乱之墙”。相互冲突的动机、流程和工具导致了这面“墙”的存在。

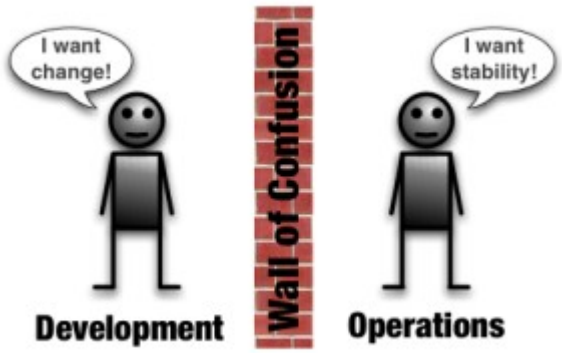

图 1 开发要变化，运维要稳定

以开发为中心的人通常认为，变化会带来回报。企业依靠他们来应对不断变化的需求。因此他们被鼓励尽可能进行变革。

而运维人员则往往视变化为敌人。企业依靠他们维持正常业务运维和实施让企业赚钱的服务。由于变化会影响稳定性和可靠性，运维业务有理由对它说不。我们已经多次听到过如下统计数字：在所有宕机事件中有 80%情况是源于自杀式的改变。

开发人员和运维人员认识世界的方法存在根本性的差别。他们都认为自己的做法是正确的。的确，孤立的来看他们都是正确的。

更糟糕的是，开发和运维团队通常处于公司组织架构的不同部分，通常具有不同管理者的和竞争关系，而且通常工作在不同的地点。

让混乱之墙更坚固的还包括开发和运维工具之间的错位。看一下开发者要求和日常使用的常见工具，再看一下系统管理员，你会发现两者存在很大不同，开发人员没有兴趣使用运维人员的工具，反之亦然；而且两部分工具之间也不存在重要的集成。即使在某些工具类型上有一些重叠之处，使用方式也完全不同。

当应用程序变动需要从开发团队推向运维团队时，混乱之墙的存在则将变得更加明显。有人将其称为一个“版本发布(Release)”，有人则称其为一次“部署(deployment)”，但有一件事情是公认的，问题可能会随之而来。下图虽然是一个抽象化场景，但是如果你经历过这一过程，一定会感觉到它的真实性。

开发人员把一个软件版本“扔”给墙对面的运维人员。后者拿到该版本产品后开始准备将其部署。运维人员手动修改由开发者提供的部署脚本或创建自己的脚本。他们还需要修改配置文件来适应与开发环境大不相同的真实生产环境。最完美的情况是，他们重复在此前环境中已完成的工作；而糟糕的情况是，他们将

引入或发现新的漏洞。

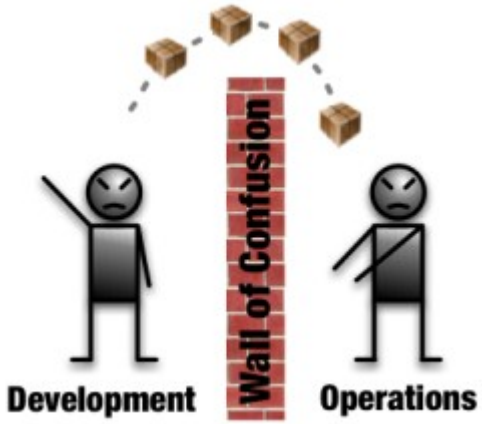

图 2 从开发到运维

运维人员然后开始进行他们自认为正确的部署过程。由于开发和运维之间的脚本、配置、过程和环境存在差别，这一部署过程实际上也是首次被执行。当然，期间如果发生一个问题，开发人员会被要求来帮助进行排障。运维人员会说开发团队给的产品存在问题。而开发人员则会回应称该产品在他们的环境下运行良好，因此一定是运维人员在部署的过程中做错了什么。由于配置、文件存储位置和过程的不同，开发人员诊断问题也并非一件易事。

没有一个可靠的方式来把环境回滚到此前已知的正常状态。本来应该一帆风顺的部署过程最后变成一场救火行动，经过反复测试之后才让生产环境恢复到正常状态。

部署阶段已经非常明显的需要 DevOps 理念来解决问题，但需要 DevOps 的绝不仅仅是这一阶段。正如 John Allspaw 所指出的那样，开发和运维之间的协作需求在部署之前就已存在，同时也会在部署之后的长时间之内继续存在。

DevOps 所带来的好处

DevOps 是一个非常强大的概念，因为它可以在众多不同层面上产生共鸣。

从开发或运维人员来看，DevOps 可以让他们从众多烦恼中解脱出来。它虽然不是具有魔力的万灵药，但是如果你能够让 DevOps 奏效，则会节省大量时间，而且不至于打击自己的士气。显而易见，投入精力将 DevOps 落到实处，我们应该会更加高效、更加敏捷和减少挫败感。

对于企业来说，DevOps 直接有助于实现两个强大战略性企业品质，“业务敏捷性”和“IT 融合”。它们可能不是 IT 人士所担忧的事情，但是却应该获得掌握财政大权的管理者的注意。

【名词定义】IT 融合：企业渴望达到的一个状态，能够高效的使用信息技术来达到企业目标——通常是提高公司业绩或市场竞争力。

通过从共同企业目标角度出发来校准开发和运维的职责和流程，DevOps 有助于实现 IT 融合。开发和运维人员需要明白，它们仅仅是一个统一业务流程中的一部分。DevOps 思想确保个体决策和行为应力求支持和改进这个统一的业务流程，无论你是来自哪一个组织架构。

【名词定义】业务敏捷性：一个机构以高效、经济的方式迅速适应市场和环境变化的能力。

当然对于开发人员来说，“敏捷”有专门的含义，但目标是非常类似的。敏捷开发方法旨在保持软件开发工作与客户/公司的目标同步，尽管需求不断变化，也可以产生高品质软件。对于多数机构来说，迭代项目管理方法 Scrum 是敏捷的代名词。

业务敏捷性承诺，在企业权益集团作出决策和开发者进行响应之间能够紧密互动和快速反馈。看一下一家运转良好的敏捷开发团体的产品，你会看到一个与业务需求保持一致的稳

定持续改进。但是，当你从企业角度回顾一下整个开发-运维周期，敏捷方法的相关优势通常会变得非常模糊。混乱之墙导致了应用程序生命周期的分裂，开发和运维分别按照不同的节奏进行。实际上，产品部署之间的长期间隔使得一个团体的敏捷工作变成了它一直试图避免的瀑布生命周期。当存在混乱之墙时，无论开发团体有多么敏捷，改变企业缓慢和迟钝的特点是极其困难的。

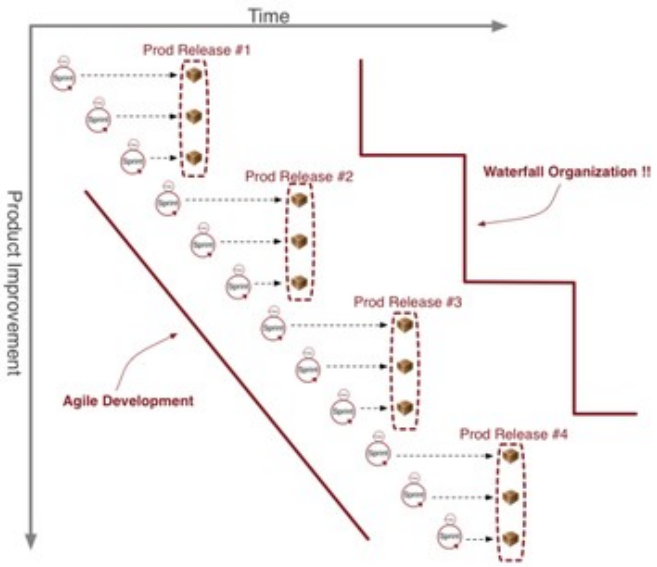

图 3 敏捷的开发，瀑布的组织

DevOps 能将敏捷开发的优势体现在机构层面上。通过考虑到快速、反应灵敏但稳定的业务运维，使其能够与开发过程的创新保持同步，DevOps 可以做到这一点。

如何将 DevOps 落到实处？

和多数新出现的话题一样，发现问题的共性特点要比找到解决方案容易的多。

要想实现 DevOps 相关解决方案，以下三方面需要关注：

1、评价和鼓励改变文化

改变文化和激励系统从来不是一件易事。但是，如果你不改变企业文化，兑现 DevOps 的承诺将非常困难。考察一个企业的主导文化时，你需要紧密关注如何评价和判断企业业绩。评价的内容将影响和刺激行为的发生。开发-运维生命周期中的所有当事方需要明白，在更大的企业流程中自己只是其中一部分。个体和团队的成功都要放在整个开发-运维生命周期内来进行评价。对于许多机构来说，这是一个转变，不再是孤立的来进行业绩评价，每一个团队不再是基于自己的团队来评价和判断业绩好坏。

2、统一标准化的流程

这是 DevOps 的一个重要主题，整个开发-运维生命周期必须被看作一个端对端的流程。流程的不同阶段可以采取不同的方法，只要这些流程可以被组合到一起创建一个统一的流程。与评价和激励的问题相似的是，实现这个统一的流程时每个组织可能会有略微不同的需求。

3、统一的工具

这是大多数 DevOps 讨论一直在关注的领域。这一点不令人吃惊，因为当技术专家在考虑解决一个问题时，第一反应往往就是直接跳转到工具讨论上。如果你关注 Puppet、Chef 或 ControlTier 等工具社区，那么你可能已经意识到人们对在开发和运维工具之间建立桥梁的重大关注。“基础设施即代码”、“模型驱动自动化”和“持续性部署”都是可以划归 DevOps 旗下的概念。

关于把 DevOps 变为现实需要哪些类型的工具，Jake Sorofman 提出如下建议：

一个版本控制软件库

它可以确保所有系统产品在整个版本发布生命周期中被很好的定义，且能够实现一致性共享，同时保持最新信息。开发和 QA 机构能够从中取得相同平台版本，生产机构部署已经

被 QA 机构验证过的相同版本。

深层模型系统

它的版本系统清晰的描述了软件系统相关的所有组件、策略和依赖性，从而可以简单的根据需要复制一个系统或在无冲突的情况下引入变化。

人工任务的自动化

在依赖关系发现、系统构造、配置、更新和回滚等过程中，减少人工干涉。自动操作变为高速、无冲突和大规模系统管理的命令和控制基础。

在从开发到运维的生命周期中存在许多不同的工具。工具选择和执行决策需要根据它们对端到端生命周期的影响来决定。

关于 DevOps 的澄清

现在某些系统管理员正在试图把自己的岗位名称改为“DevOps”。但是，DevOps 不应该是一个单一的位置或职称。把 DevOps 变成一个新职位名称或特定角色是一件非常危险的事情。例如这会导致以下错误端点：你是一个 DBA？或者是一个安全专家？那么不用担心 DevOps，因为那是 DevOps 团队的问题。

设想一下，你不会说“我需要招聘一个 Agile”或“我需要招聘一个 Scrum”或“我需要招聘一个 ITIL”，而只是会说需要招聘了解这些概念或方法的开发人员、项目经理、测试人员或系统管理员。DevOps 也是同样道理。

与 DevOps 具有相同理念的术语很多，例如敏捷运维（Agile Operations）、敏捷基础设施（Agile Infrastructure）和 Dev2Ops。还有很多人虽然没有提及“DevOps”，但却在遵循着类似的理念。

原文地址：

http://os.51cto.com/art/201009/223199.htm

相关阅读：

开发者与系统管理员的争执:不要碰我的生产环境! via 外刊 IT 评论

敏捷运维概述 via InfoQ 中文站

初探敏捷开发 via 51CTO 开发频道

【运维漫画阁】正则表达式有什么用？

漫画国际来源：

http://imgs.xkcd.com/comics/regular_expressions.png

有关 Perl

Perl（Practical Extraction and Report Language）是一种脚本语言，最初由 Larry Wall 于 1987 年 12 月 18 日发表。被称之为“一种拥有各种语言功能的梦幻脚本语言”、“Unix 中的王牌工具”。

推荐阅读：

善用脚本 让你的 Nagios 记录系统监控日志

自动分析黑名单及白名单的 iptables 脚本

正则表达式完全学习手册：菜鸟入门指导

全面分析 Linux 正则表达式（一）

全面分析 Linux 正则表达式（二）

全面分析 Linux 正则表达式（三）

全面分析 Linux 正则表达式（四）

全面分析 Linux 正则表达式（五）

全面分析 Linux 正则表达式（六）

命令行&工具 | 开源工具链

原文：Open Source Toolchains for Linux Systems Administrators

作者：Mark Hinkle

译者：哲婷

今天的系统管理有两大明显趋势，它们是 DevOps 和进展稍微迟缓的敏捷运维运动。这些措施流行于许多 Web 2.0 和云计算公司，比如 Twitter、谷歌还有雅虎，当然也不乏像 Facebook 这样其产品高度依赖 IT 技术的公司。但是，实际上这种做法也同样非常适合传统企业中的 IT 管理员们，这些企业往往都有大规模的基础架构和不切实际的工作负荷，需要提高工作效率来达成其业务目标。

DevOps 鼓励与开发者们共同进行产品研发。系统管理员的角色经历了不断的变化，已经从被动执行重复系统建设任务的基础架构维护者演变成现在的系统设计和建设工程师。在过去，系统管理员可能在一批随机的脚本中搜集他们的专业领域知识；但是现在，精明的系统工程师们已经开始编写基础架构，并可以确保对于这些编码的理解和制度化能够遍及整个公司。

DevOps 和 Agile 并不被定义为是一次技术性改变，而是专业和文化上的改变。它们重新定义了 IT 管理员的职能：从系统和 IT 架构维护向管理和定义可复制的、有弹性的和高度可用的 IT 系统转变。

这一变化势必会带来一些战术上的改善，用以达成更高的可用性和工作效率。以下做法是敏捷系统工程师们最常使用的办法：

1、基础架构自动化

相比起执行重复任务而言，管理员们需要创建机械化进程，让它们有效的使用工具。这些工具可以用来生成一致的结果并被团队中的其他成员所分享。

2、服务器版本控制

任何变化并不直接应用到服务器上，而是应用到一个中央软件库内。所有改变从中央软件库推送到服务器上，从而建立一个在出现错误的时候的回滚机制。

3、频繁的改进和发布/更新

旧的思维方式是，尽可能减少对服务器和架构进行改变（无论是次数还是规模）来降低风险。而敏捷的运维思路是，通过频繁的微小变化来改善服务器的性能，而由于每次的变化都很小，当出现问题的时候，他们可以轻易地跟踪那些影响架构的不利变化。

为了实现这些目标，系统工程师需要一个更强大的工具包。幸运的是，开源界为了这一目标已经研发了大量的工具。

开源工具链

软件开发者们对于工具链相当熟悉：一个程序的输出构成了另一个程序的录入，从而组成了一系列程序。比如说，使用 GNU Emacs 编辑器，GNU bin-utils 和 GNU 编译器集（GCC）的组合。软件开发者们编写很多在其它程序中调用的程序与流程，而不是将同一段代码在所有的地方都重复一遍。

有了新生的 DevOps 和敏捷运维运动，运维人员们组成了社区，帮助人们用自己所喜爱的工具来定义属于自己的工具链。DevOps 的工具链项目就是这些社区当中的一员。

正如软件开发商有不同的专用工具来串联

成一条软件工具链（比如编辑、编译、构建脚本等），系统管理员也可以运用由这些能够实现自动管理功能和维护 Linux 服务器的工具组成的工具链。它们可以分为三大类：预备、配置管理和监控。

Provisioning	Configuration Management and Automation	Monitoring
• Cobbler • Kickstart • OpenQRM • Spacewalk	• Chef • ControlTier • Func • Puppet	• Nagios • OpenNMS • Zabbix • Zenoss Core

1、预备类工具可以使 Linux 服务器上的软件安装包自动化。它们借助服务器上的软件包系统比如 rpm 或者 apt 来安装软件包，有些甚至会做一些粗略的配置工作。

2、配置管理和自动化是用来设置参数或者开启一个新服务器上的服务。它们也可以用来把系统还原到遇到错误之前的状态。

3、监控工具用来收集服务器数据，从而生成可用性、性能和其它系统状态的报告。

整合系统管理工具

开源工具链十分善于维护高服务水平，因为自动化和准确无误的运作比手动解决问题更快、更有效率。如果你想保持五个九的可用服务水平（99.999%的正常运行时间），那么你在每一年中只允许有 5 分 15 秒的停机时间。对于一个管理员而言，这样的时间甚至来不及接收错误报告页，更不用说登陆到服务器进行问题诊断了。

构建工具链可以从使用服务器自动构建工具开始。服务器自动构建工具可以加速部署速度，并可以在短时间内大规模部署服务器，同时也可以让构建过程更容易复制。在发生严重故障时，还可以重建架构。

在早期，Linux 用户可能会整理出一个软件包列表，发至 rpm 进行批量软件安装。后来，我们用 Kickstart 来执行无人值守的 Linux 安装。现在，Cobbler 把这个功能提升到了一个新的高度：它实现了物理机与虚拟机的并行系统构建，并且可以进行 DHCP 和 DNS 的配置。

Cobbler 还集成了其它的工具，比如用于配置管理自动化的 Puppet 等软件。这个工具可以在系统安装完毕后进行服务的自动更新。

开源工具链的另外一个例子是将预备、配置和自动化集成到一起的工具，比如由红帽资助的 Genome 项目。

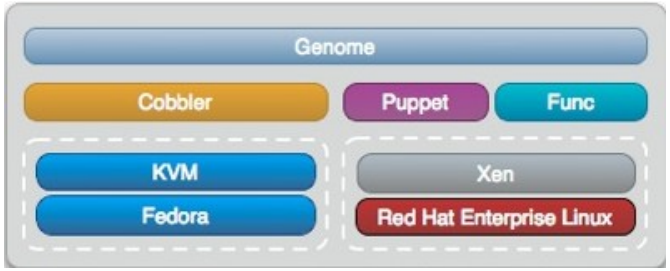

图 1 Genome 项目组成

Genome 是一系列用于维持云架构的工具，可以部署多层次的网络应用，包括 Apache 反向代理层，JBoss 应用服务器层，以及 PostgresSQL 数据库层。

上述大部分工具都是主动类的工具，可以用来进行变动并执行任务。然而，它们却缺乏关于当前系统状况的信息，于是监控工具就有了用武之地。对于传统系统管理员而言，监控无非是在发生错误的时候通过一个页面或者一封邮件提醒他们。但是，监控工具（如 Nagios / OpenNMS / Zenoss Core）能够提供更为完善的服务器性能检测，可以告诉管理员们所

有运行中的服务的状态。它们中的一些甚至可以提供在其它工具中开启程序的界面，比如 Zenoss Core 可以在一个检测状态台上通过 Cfengine、Chef 或者 Puppet 来重新配置服务。

在一次 O'Reilly Velocity 的会议上，我们曾经进行过一次叫做 DevOps GameDay 的演示。一个网络应用程序被部署在东海岸和西海岸的亚马逊 EC2 数据中心。

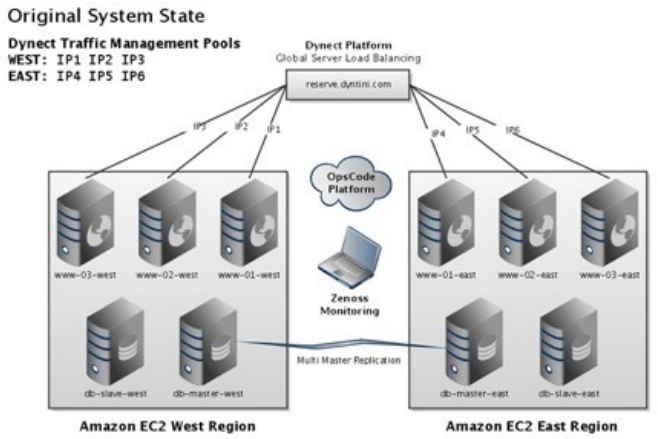

图 2 DevOps GameDay

在西海岸的管理员制造了一些服务器故障。Zenoss Core 对架构进行了监测，发现了故障并且通知 OpsCode 的 Chef 采取行动。Chef 随后通过 Dynect API 更新了 Dynect 平台上的服务，从而恢复了向东海岸设备的正常数据传输。每次在新的服务器上线时，Chef 都能够将新情况推送至 Zenoss，并开始进行监测。

这次演示让大家认识到架构自动化改进系统恢复的能力。在这种情况下，系统能够保持内置冗余和多个网路服务器以及 DNS 之间的平衡。当系统发现了故障时，架构可以在 90 秒内进行自动恢复。虽然这只是一个简单的演示，但是同样的设计可以运用到其它的工具和情景中去。

总结

多年以来，专有软件管理厂商一直在尝试通过广泛的管理套件来提供完整的服务周期管理。这些产品通常在一个秘密环境下进行研发，然后通过收购规模较小的公司的技术来进行不断地补充。这种东拼西凑的方式往往让他们出售给大多数用户的产品有着一长串的功能，但事实上总是效果少于承诺。

开源工具对于管理架构的好处不言而喻。首先，它们为用户提供了选择余地。其次，技术得到了开放的发展，而终端用户能够更好的诠释功能集成软件的发展。第三，开源工具促进创新，并且极具包容性。最后，开源软件的氛围利于开放标准、互操作性和开放 API，这使得集成变得更为简单。

无论你的基础架构如何，都可以试试看使用开源工具创建属于你自己的工具链，也许会有想象不到的收获。

原文地址：

http://os.51cto.com/art/201008/215653.htm

相关阅读：

"全世界系统管理员日"的由来 via Microsoft

Linux 批量安装 五大开源软件挨个看 via 51CTO 系统频道

系统管理员应该定期完成的九件事 via 51CTO 系统频道

【命令行&工具】自动化开源工具一览

预备类

Kickstart

【描述】针对红帽 Linux/Fedora 等发行版的自动化安装方式，简单的讲就是让系统在安装过程中从一个 ks.cfg 配置文件中自动获取所有需要配置的参数。源于 Anaconda 项目。

【技术支持】红帽/Fedora 社区

【起始于】2004 年之前

【维基】
http://fedoraproject.org/wiki/Anaconda/Kickstart

Cobbler

【描述】为了实现快速网络安装环境的 Linux 安装服务器，可以为数量众多的 Linux 服务自动化执行任务

【发起人】Michael DeHaan

【技术支持】红帽/Fedora 社区

【起始于】2007 年之前

【官方网站】
https://fedorahosted.org/cobbler/

配置管理类

Chef

【描述】一个系统集成框架，可以用 Ruby 等代码完成服务器的管理配置并编写自己的库。

【技术支持】OpsCode

【起始于】2009 年 1 月

【官方网站】
http://www.opscode.com/chef/

ControlTier

【描述】一个开源、跨平台的构建/部署服务器的自动化框架，可以在多个节点、多个应用层上进行服务扩展及管理等工作。

【技术支持】ControlTier 社区（Google Group）

【起始于】2007 年之前

【官方网站】
http://controltier.org

监控类

Nagios

【描述】一个强大的监控预警系统，可以监控系统、应用、服务以及各种进程的运行状况，并提供了多种警报机制。

【技术支持】Nagios Enterprises

【发起人】Ethen Galstad

【起始于】1999 年

【官方网站】
http://www.nagios.org/

OpenNMS

【描述】一个网络管理应用平台，可以自动识别网络服务，事件管理与警报，性能测量等任务。

【技术支持】openNMS group

【起始于】2005 年之前

【官方网站】
http://www.opennms.org/

【命令行&工具】自动化开源工具一览

预备类

OpenQRM

【描述】针对数据中心管理的开源平台，针对设备的部署、监控等多个方面通过可插拔式架构实现自动化的目的，尤其面向云计算/基于虚拟化的业务。

【发起人】Matt Rechenburg

【起始于】2005 年之前

【官方网站】
http://www.openqrm.com/

Spacewalk

【描述】针对红帽/Fedora 等发行版的软件更新管理软件，同时也提供预备和监控的功能。这个项目衍生了红帽 Network Satellite 产品。

【技术支持】红帽

【起始于】2001（Red Hat Network）/2008

【官方网站】
http://spacewalk.redhat.com/

配置管理类

Func

【描述】全称为 Fedora Unified Network Controller，Fedora 统一网络控制器，用于自动化的远程服务器管理。

【发起人】Michael DeHaan 等

【技术支持】红帽/Fedora 社区

【官方网站】
https://fedorahosted.org/func/

Puppet

【描述】一个开源的数据中心自动化/配置管理框架，用于 Puppet 自己的声明语言自动化重现任意的系统配置。

【技术支持】Puppet Labs

【官方网站】
http://www.puppetlabs.com/

监控类

Zabbix

【描述】用于监控网络上的服务器/服务以及其他网络设备状态的网络管理系统，后台基于 C，前台由 PHP 编写，可与多种数据库搭配使用。提供各种实时报警机制。

【技术支持】Zabbix 公司

【发起人】Alexei Vladishev

【起始于】1998 年

【官方网站】
http://www.zabbix.com/

Zenoss Core

【描述】一个基于 Zope 应用服务器的应用/服务器/网络管理平台，提供了 Web 管理界面，可监控可用性、配置、性能和各种事件。

【技术支持】Zenoss Inc.

【起始于】2002 年

【官方网站】
http://zenoss.com/

实战 | Kickstart

作者：抚琴煮酒

本文记录了用 Kickstart 搭建 RHCE 实验室环境的详细步骤。文中使用的环境为 RHEL 5，也同样适用于 CentOS。文章作者抚琴煮酒（Andrew Yu）是经验丰富的 Linux/Unix 系统工程师，并曾经担任过一段时间的 RHCE 讲师的工作。

原理和概念

什么是 PXE

严格来说，PXE 并不是一种安装方式，而是一种引导的方式。进行 PXE 安装的必要条件是要安装的计算机中包含一个 PXE 支持的网卡（NIC），即网卡中必须要有 PXE 客户端。PXE（Pre-boot Execution Environment，直译为预启动执行环境）协议使计算机可以通过网络启动。协议分为 client 和 server 端，PXE client 在网卡的 ROM 中，当计算机引导时，BIOS 把 PXE client 调入内存执行，由 PXE client 将放置在远端的文件通过网络下载到本地运行。运行 PXE 协议需要设置 DHCP 服务器和 TFTP 服务器。DHCP 服务器用来给 PXE client（将要安装系统的主机）分配一个 IP 地址，由于是给 PXE client 分配 IP 地址，所以在配置 DHCP 服务器时需要增加相应的 PXE 设置。此外，在 PXE client 的 ROM 中，已经存在了 TFTP Client。PXE Client 通过 TFTP 协议到 TFTP Server 上下载所需的文件。

什么是 KickStart

KickStart 是一种无人职守安装方式，其工作原理是通过记录典型的安装过程中所需人工干预填写的各种参数，并生成一个名为 ks.cfg 的文件；在其后的安装过程中（不局限于生成 KickStart 安装文件的机器）当出现要求填写参数的情况时，安装程序会首先去查找 KickStart 生成的文件，当找到合适的参数时，就采用找到的参数，当没有找到合适的参数时，才需要安装者手工干预。这样，如果所有的 Kickstart 参数都设置好，安装者完全可以只告诉安装程序从何处取 ks.cfg 文件，然后去忙自己的事情。等安装完毕，安装程序会根据 ks.cfg 中设置的重启选项来重启系统，并结束安装。

PXE + KickStart 安装的条件

执行 PXE + KickStart 安装需要的设备为：

1. DHCP 服务器
2. TFTP 服务器
3. KickStart 所生成的 ks.cfg 配置文件
4. 一台存放系统安装文件的服务器，如 NFS、HTTP 或 FTP 服务器
5. 带有一个 PXE 支持网卡的将安装的主机

下面介绍具体的安装步骤。

【实战】Kickstart 无人值守安装搭建 RHCE 实验室

一、安装 httpd

```
yum -y install httpd*
```

二、挂载 RHEL5 的 DVD 光盘，并复制第一张光盘下的所有内容（文件和文件夹）到/var/html/www 下

```
mount /dev/cdrom   /mnt
cp -rf   /mnt/*   /var/html/www
```

三、安装 tftp-server，并启用 tftp 服务，重启 xinetd 进程

```
rpm -ivh tftp-server-0.39-1.i386.rpm
vi /etc/xinetd.d/tftp
# default: off
# description: ...
{
   socket_type = dgram
   protocol    = udp
   wait        = yes
   user        = root
   server      = /usr/sbin/in.tftpd
   server_args = -s /tftpboot
   disable     = no #disable 的值由 yes 变为 no
   per_source  = 11
   cps         = 100 2
   flags       = IPv4
}
service xinetd restart
```

四、配置支持 PXE 启动

注意我已经把第一张光盘的内容复制到/var/www/html 目录中了，所以所需要的文件我只需要从/var/ftp 目录中复制就行了。

1、进入 tftpboot 文件夹，没有就建一个，有了就不用建了。

```
cd /tftpboot/
```

2、把 pxelinux.0 复制到/tftpboot/中

```
cp /usr/lib/syslinux/pxelinux.0 /tftpboot
```

3、把 Linux 第一张安装光盘上/image/pxeboot/initrd.img 和 vmlinux 复制到/tftpboot/中

```
cp /var/ftp/image/pxeboot/initrd.img   /tftpboot
cp /var/ftp/image/pxeboot/vmlinux   /tftpboot
```

4、 复制第一张安装光盘上的 isolinux/*.msg 到/tftpboot/中

```
cp /var/ftp/isolinux/*.msg   /tftpboot
```

5、 在 tftpboot 中新建一个 pxelinux.cfg 目录

```
mkdir pxelinux.cfg
```

6、 把 Linux 第一张安装光盘上 isolinux 目录中的 isolinux.cfg 复制到 pxelinux.cfg 目录中，并同时更改文件名称为 default

```
cd pxelinux.cfg
cp /var/ftp/isolinux/isolinux.cfg
/tftpboot/pxelinux.cfg/default
```

五、安装 dhcp 服务，同时修改配置

1、 安装

```
rpm -ivh dhcp-3.0.1-12_EL.i386.rpm
```

2、 复制配置模板文件到指定的目录中，并重命名

```
cp /usr/share/doc/dhcp-3.0.1/dhcpd.conf.sample
/etc/dhcpd.conf
```

3、 修改配置文件，添加一行：filename "/pxelinux.0"，其他的修改自己完成就行了。这文件的位置一定要注意,不然会失败,切记

```
[root@localhost isolinux]# vim /etc/dhcpd.conf
ddns-update-style interim;
ignore client-updates;
next-server 192.168.1.14;#PXE 服务器 IP 地址

filename "/pxelinux.0";#注意此行的位置,写在 subnet 下面的话会失败

subnet 192.168.1.0 netmask 255.255.255.0 {
# --- default gateway
      option routers                  192.168.1.254;
      option subnet-mask              255.255.255.0;
      option nis-domain               "example.com";
      option domain-name              "example.com";
      option domain-name-servers      192.168.1.254;
      option time-offset -18000;
# 其他配置参数
#
}
```

4、 启动 dhcp 服务

```
service dhcp start
```

六、 安装 kickstart 并进行配置

首先安装 Kickstart：

```
rpm -ivh system-config-kickstart-2.5.16-2.noarch.rpm
```

在 gnome 环境下配置 kickstart 的指令如下：

```
system-config-kickstart
```

之后的配置步骤如下：

1. 基本配置，按自己需求来就好
2. 安装方法，选择 httpd 安装，切记不要输入任何的帐号，我们采用的匿名安装
3. 引导安装程序选项，不需要做更改
4. 分区信息，创建三个分区
5. 网络配置，我使用的静态分配地址(动态同样如此)
6. 显示配置，按自己需求来就好
7. 软件包的选择，我选择了 Kernel Developent 和 Development Tools 安装（但千万不要选择这两个软件包，不然的话在安装的时候会报错的）
8. 其他的都是默认设置，没有做修改
9. 生成文件 ks.cfg，保存到/var/www/html 下

七、修改/tftpboot/pxelinux.cfg/default 文件，指定读取 ks.cfg 的方法（ks=http://192.168.1.40/ks.cfg）

```
vi /tftpboot/pxelinux.cfg/default
auth --useshadow --enablemd5
key --skip #这行一定要，跳过注册号输入，不然会失败
bootloader --location=mbr
clearpart –all –initlabel
text
firewall --disabled
firstboot --disable
keyboard us
lang en_US
logging --level=info
url --url=http://192.168.1.14/
network --bootproto=dhcp --device=eth0 --onboot=on
reboot
rootpw --iscrypted $1$HEJKfwF9$r1l0JoPz74ToF9NbE3Qs1
selinux --disabled
timezone --isUtc Asia/Shanghai
intall
xconfig --defaultdesktop=GNOME -depth=8
--resolution=640x480
part swap --bytes-per-inode=4096 --fstype="swap"
--size=512
part /boot --bytes-per-inode=4096 --fstype-"ext3"
--size=200
part / --bytes-per-inode=4096 --fstype="ext3" --grow
--size=1
%packages
#下面指定软件包列表即可
```

最后，重新引导安装就可以了。

以上是我的配置步骤，如果没有出入的话，是一定可以成功的！

原文地址：

http://hi.baidu.com/yuhongchun027/blog/item/8ebd44b7102ee1f331add1b1.html

推荐阅读：

RHEL5 无人值守安装图文攻略 via 51CTO 论坛

实战 | Cobbler

作者：hutuworm

从前，我们一直在做装机民工这份很有前途的职业。自打若干年前 Red Hat 推出了 Kickstart，此后我们顿觉身价倍增。不再需要刻了光盘一台一台地安装 Linux，只要搞定 PXE、DHCP、TFTP，还有那满屏眼花缭乱不知所云的 Kickstart 脚本，我们就可以像哈里波特一样，轻点魔棒，瞬间安装上百台服务器。这一堆花里胡哨的东西可不是一般人都能整明白的，没有大专以上学历，通不过英语四级，根本别想玩转。总而言之，这是一份多么有前途，多么有技术含量的工作啊。

很不幸，Red Hat 发布了网络安装服务器套件 Cobbler（补鞋匠），它已将 Linux 网络安装的技术门槛，从大专以上文化水平，成功降低到初中以下，连补鞋匠都能学会。对于我们这些在装机领域浸淫多年，经验丰富，老骥伏枥，志在千里的民工 兄弟们来说，不啻为一个晴天霹雳（}雷{）。

Cobbler 声称可以快速建立网络安装环境，那么到底有多快呢？我在一台装有 Fedora 9 的服务器上进行了测试，步骤如下：

1. 安装相关软件：

```
yum -y install cobbler tftp-server dhcp httpd xinetd
```

注意 /var/www/cobbler 目录必须具有足够容纳 Linux 安装文件的空间（移动，建软链接）

2. 检查 cobbler 配置：

```
cobbler check
```

按提示解决相关问题，把 /etc/cobbler/settings 中的 server 和 next_server 设为本服务器的 IP 地址，manage_dhcp 设为 1，以便管理 DHCP

3. 导入 Fedora 9 安装 DVD ISO 中的文件：

```
mount -o loop Fedora9/x86_64/Fedora-9-x86_64-DVD.iso /mnt/dvd/
```

将 ISO 文件挂载到 /mnt/dvd 目录

```
cobbler import --mirror=/mnt/dvd --name=FC9-x86-64
```

从 /mnt/dvd 目录导入所有安装文件，命名为 FC9-x86-64

```
cobbler distro list
```

查看导入结果，应显示 FC9-64-i386 和 FC9-64-xen-i386

4. 修改 DHCP 和 Kickstart 配置模板：

```
vi /etc/cobbler/dhcp.template
```

DHCP 配置模板，如果已经有一个 dhcpd.conf，可参照修改此模板

```
vi /etc/cobbler/sample.ks
```

Kickstart 配置模板

5. 生成并同步所有配置：

```
cobbler sync
```

6. 启动相关服务：

```
service xinetd start
```

/etc/xinetd.d/tftp 中 disable = no

service dhcpd start

```
service cobblerd start
```

曹植七步成诗，而 Cobbler 居然只需要六步。启动另一台新服务器，通过 PXE 启动进入蓝色的 Cobbler 安装界面，选择 Fedora 9 安装项，几分钟之内就能一气呵成，自动完成系统安装。

原文地址：

http://hutuworm.blogspot.com/2008/08/cobblerlinux.html

相关阅读：

漫谈运维：半神半仙亦民工

下期内容预告：监控与报警机制

时刻掌握服务器以及其上运行的应用服务的运行状态，是运维工程师们确保服务可用性的一大关键。下期内容将关注 Linux/Unix 运维人员在监控方面需要注意的方面，届时我们将请一位资深运维来谈谈对于监控与报警方面的心得，介绍一些国际上近期监控理念/技术方面的进展，并交流一些具体的操作心得，敬请关注！

有关《Linux 运维趋势》

《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，内容侧重中、高端的运维技术趋势与理念，并以一些基础的技巧心得、实际操作案例作为辅助。

十分感谢您的阅读！