图书馆数据中心基础架构部署自动化系统

郑海山
(厦门大学信息与网络中心　厦门 361005)

摘要: 【目的】解决高校图书馆数据中心在服务器数量膨胀下的自动化部署问题。【应用背景】高校图书馆数据中心承担的任务越来越重，服务器数量急剧攀升，自研任务增多，人工管理已无法适应。【方法】引入自动化理念，对数据中心的基础架构配置全部使用自动化脚本部署。引入 Vagrant 工具使得开发环境和生产环境一致。【结果】解决服务器和虚拟机的操作系统依赖组件自动化部署的问题，以及开发和生产环境不一致的问题。运维人员工作量减少，提高开发效率。【结论】应用自动化运维理念和方法后，图书馆数据中心对内管理更加清晰化、规范化、自动化。

关键词: 数据中心　运维管理　Puppet　Vagrant
分类号: TP393

1　引　言

随着高校信息化的兴起，高校图书馆数据中心的实体和虚拟服务器数量不断膨胀，数据中心运维工作复杂性加大。国内对数据中心运维方面存在一个共同的观点，即需要提高自动化水平[1]。李倩[2]对图书馆的运维自动化归纳出 10 个工作内容。而针对数据安全，杨义先等[3]总结了数据容灾、系统容灾和应用容灾的关键技术和实现方法。基于故障发现后的处理机制，有些高校使用 ITIL[4]对运维故障管理进行标准化。李艳霞等[5]制定规章制度，部署自动化监控预警系统。

本文介绍厦门大学图书馆数据中心在服务器开发和部署方面的自动化经验，通过引入自动化理念，定义一套服务器采购安装部署的标准化规范，对服务器的全生命周期进行规范化管理。以开源 Puppet 为核心，结合 Fabric、Bat 和 PowerShell 等自动化脚本工具，并在自行研发的项目中应用 Vagrant，使得开发、测试、部署全部实现自动化。结合笔者前期研究[6]，对图书馆数据中心进行了有效的管理。

2　研究背景

2.1　数据中心面临的问题

(1) 新服务器部署困难：新采购的服务器根据不同的用途需要安装不同的操作系统，光盘安装模式较为繁琐。操作系统安装后，诸如 IP 地址、防火墙、自动更新、运行库、数据库、监控、备份等配置环节使用传统方法手动部署浪费时间较多。

(2) 无法快速获知已部署的服务器的状况：对数据中心内已经部署并正在运行的虚拟机和物理服务器，只有进入操作系统才能看到其上面运行的各种服务、依赖的组件、组件的配置等信息。工程师无法快速了解数据中心内的所有服务器和业务，无法达到对服务器和系统自主可控的要求。

(3) 数据中心文档缺乏：技术工程师偏爱写程序但不愿写文档，这导致数据中心缺乏规范的文档描述，工作交接和应急处理无章可循。

(4) 基础架构的备份和异地重建较难：数据中心整体基础架构的备份依赖于虚拟机。虽然备份软件的重复数据删除功能极大减少了备份集的大小，然而备

通讯作者：郑海山，ORCID: 0000-0001-5242-2414，E-mail: haishion@xmu.edu.cn。

份还是占用了备份软件的许可证数量。数据中心备份难以在异地重建。

(5) 开发环境搭建周期长：开发环境和生产环境的不一致极大阻碍了业务的创新。新加入的开发人员可以克隆一台已经部署好的虚拟机，通过更改 IP 地址、更新代码库和数据库版本加入到开发进程。然而部署好的虚拟机需要随时根据开发进程调整设置。如果是集群环境，还需克隆多台虚拟机，开发环境搭建成本较高。

2.2 解决思路

(1) 引入自动化理念：开发一个运维系统对数据中心部署进行规范。需要更新观念，虽然有时手工操作比自动化脚本更快更直接，然而为了今后维护方便，对操作可追溯，应尽量对所有操作脚本化。整个运维系统应当对基础架构平台全部使用文本定义，使用大量的自动化脚本配合少量的手动操作。

(2) 操作系统自动安装：服务器的操作系统安装可采用 PXE 网络启动安装[7]，系统的配置通过自动安装脚本使用文本文件定义，打开物理或者虚拟服务器即可实现自动安装。安装后通过 Puppet 脚本定义安装后的操作系统的各个组件的配置信息和业务的运行环境。这些自动安装脚本和 Puppet 脚本通过规范的文档书写，代码则变成数据中心的文档。

(3) 部署脚本可重复运行：对数据中心架构备份只需备份脚本即可，无需对整个虚拟机进行备份。有了这些文档，服务器在公有云内重建也较简单，只需要克隆一个简单的操作系统运行 Puppet 等脚本即可全部配置部署完毕。

(4) 开发环境自动化：开发环境使用 Vagrant 工具，只要一人编写了部署环境定义，参与者只需要运行一条命令，即可在几分钟内部署好所有环境，而且部署可在个人笔记本上实现，方便离线开发。

3 系统实现

3.1 系统框架及功能

系统整体框架如图 1 所示：

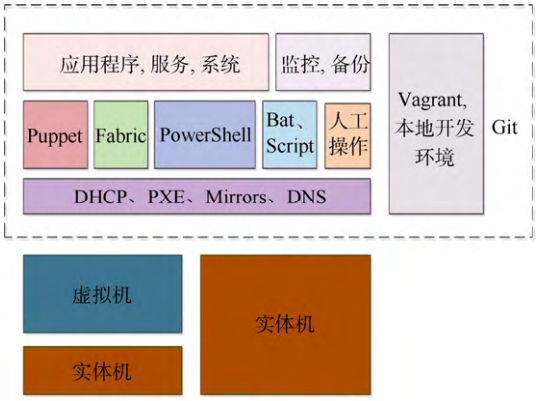

图 1 系统框架图

多个 Git 源代码库和一个贯穿始终的自动化理念组成了整个运维系统。依托实体机和虚拟机，通过 DHCP、PXE、Mirrors 服务器以及 DNS 服务器分配地址自动安装操作系统，配合 Puppet、Fabric、PowerShell、Bat、Script 脚本和少量人工操作定义运行环境，最终实现应用系统和服务顺利运行。并通过脚本自动将机器 IP 地址和描述加入位置管理、监控服务器和备份策略内。而本地开发环境则使用 Vagrant 定义。所有代码均放入 Git 源代码版本库，能全部实现自动化操作，需要手工介入的已在文档内进行详细说明。

应用规范后的服务器全生命周期流程如图 2 所示：

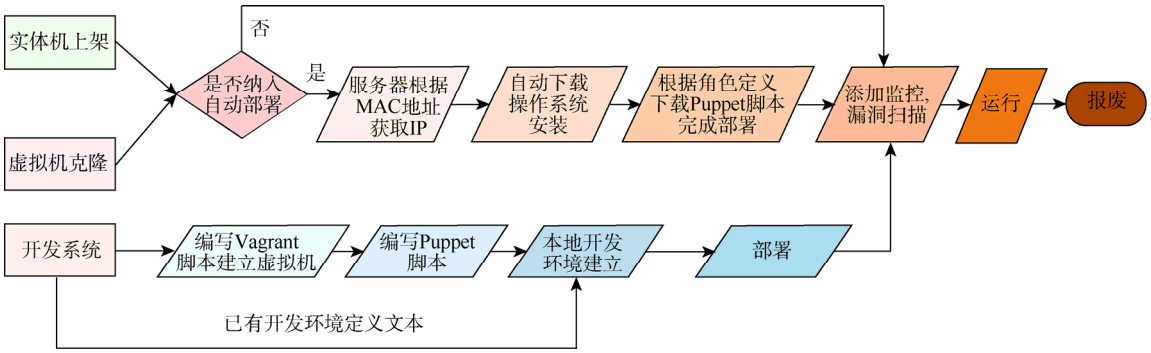

图 2 服务器的全生命周期流程

依托规范和自动化理念，运维系统实现的功能如下:

(1) 对基础架构配置文本化，采用版本管理，变更可追朔。

(2) 部署自动化。

(3) 开发和测试自动化。

(4) 文档自动化，代码即清晰的文档。

3.2 Git 源代码版本库

Git 是整个项目协作的基础，对于数据中心的相关项目，可以选择全部放入一个大的Git代码库，也可将每个子项目分开，笔者分析了所有项目的差别，如表 1 所示:

表 1　项目各项差别

项目名称	代码量	修改频率	更改周期	与其他项目耦合度	安全性要求
DNS 配置	小	高	新机器部署时, 日常	中	高
DHCP 配置	小	高	新机器部署时, 日常	中	高
TFTP 配置	小	中	新机器部署时	中	一般
NPT, Mirrors	小	低	无	低	低
业务系统	大	高	日常	耦合度低, 首次部署需要修改 DNS 和 DHCP	各个业务系统授权不同的开发工程师
监控系统	小	中	新机器部署时	中	低

经过比较和整合，笔者将整个运维系统分为以下源代码仓库:

数据中心主项目，包含 DNS、DHCP、TFTP 等项目，授权专门的部署工程师。NTP、Mirrors、位置管理系统整合成一个项目。监控系统、备份系统整合成一个项目。各个业务系统为独立项目，授权开发者团队。所有的 Puppet 资源定义、自动化脚本、单次运行的脚本、在测试中可能多次运行的脚本全部放入业务项目内部。

3.3 服务器初始安装

服务器上架或者虚拟机建立以后，打开电源，通过PXE网络启动，自动从DHCP服务器获取IP地址，再从TFTP服务器下载操作系统镜像，自动安装操作系统[7]。安装后，自动从Puppet主服务器下载服务器应有的配置，自动部署需要的组件和设置。

Windows 下，无人值守安装的配置文件可使用“自动安装工具包”生成。Ubuntu 可使用 debconf-get-selections 命令得到安装 Ubuntu 系统所有可选择的选项，节选部分如下:

```
d-i debian-installer/locale select en_US.UTF-8
d-i console-setup/ask_detect boolean false
d-i netcfg/get_hostname string dns3
tasksel tasksel/first multiselect standard, ubuntu-server
```

3.4 Puppet 自动化

Puppet[8]是一个配置管理工具, Puppet 使用一种简洁的语言，对系统内的资源所要达到的最终状态进行定义。Puppet 使用主机名区分一台服务器，所以 DNS 服务器是 Puppet 多节点管理的基础。对于 DNS，学校内网络中心也有 DNS 服务器，然而建立数据中心自有的 DNS 服务器有其必要性，通过网络中心授权图书馆主域名到图书馆的 DNS 服务器，使得图书馆对馆内其他业务系统的 DNS 条目控制权较高，可自行方便地添加修改 DNS 条目而无需联系网络中心。同时数据中心有些服务器为内部 IP 地址，无法直接连接外网，无法获取 DNS 记录。而自有的 DNS 服务器可建立外网无法访问的 DNS 条目，使得任何一台服务器均可有一个 DNS 地址，方便工程师记忆和 Puppet 管理。

所有的基础架构，诸如 DNS、DHCP、TFTP、监控、备份系统等配置也均使用 Puppet 管理。

对于业务系统，以一个典型的提供 ASP 运行环境的 Windows 服务器为例，安装完操作系统后，需添加 IIS 角色、ASP 角色服务，启用 ASP 父路径，设置调试时，将错误发送到浏览器，安装连接 SQL 客户端程序等。以往需要手工在 GUI 操作，只需使用类似以下 Puppet 脚本即可实现:

```
dism{'IIS-WebServerRole': ensure=>present}
dism{'IIS-WebServer': ensure=>present, require=>Dism
  ['IIS-WebServerRole']}
dism{'IIS-ApplicationDevelopment':ensure=>present,require=>
  Dism['IIS-WebServerRole']}
dism{'IIS-ISAPIExtensions':ensure=>present,require=>Dism
  ['IIS-ApplicationDevelopment']}
dism{'IIS-ASP': ensure=>present,require=>Dism
  ['IIS-ISAPIExtensions']}
exec{'Enable ASP Parent Path': command=>'C:\Windows\
  System32\inetsrv\appcmd.exe set config -section:system.
  webServer/asp /enableParentPaths:"True" /commit:apphost',
  require => Dism['IIS-ASP']}
```

```
package{'SQL Server Native Client':ensure=>present,
  source=>'\\192.168.1.150\software\sqlncli_x64.msi',provider
  =>'windows',install_options=>[{'USERNAME'=>'LibRoot'},
  {'COMPANYNAME'=>'Lib'},{'IACCEPTSQLNCLILICENSE
  TERMS'=>'YES'}]}
```

其中，MSI 静默安装的设置变量名可通过 Orca MSI Editor 导出。

3.5 相关脚本

Puppet 是针对资源所达到的最终状态定义的，有些操作无法通过 Puppet 定义完成，有些命令是一次性不能重复运行的，这时就需要补充。比如 Fabric、Shell 脚本、Windows 下的 Bat 脚本、PowerShell 命令等。

Windows 下设置防火墙的 Bat 脚本如下:

```
netsh advfirewall reset
netsh advfirewall firewall delete rule name=all
netsh advfirewall set allprofiles state on
netsh advfirewall set allprofiles firewallpolicy blockinbound,
  allowoutbound
netsh advfirewall firewall add rule name="Open Port" dir=
  in protocol=tcp localport=80 action=allow
```

通过类似的脚本，无需使用 GUI 界面即可完成需要的任务，规避了人工操作多个步骤中可能造成的错误，并且此脚本可在多台服务器中重复利用。

3.6 Vagrant 搭建开发测试环境

Vagrant 是搭建虚拟机开发环境的工具，它可将部署提前到开发阶段，使得开发环境和生产环境一致，减少了“在我的电脑上运行得很正常”[9]的错误。

之前的开发场景可能是：开发者开发，经历一段时间，部署到生产环境。继续开发，变更后再到生产环境部署。这个过程中任何的人工错误都会导致系统产生 Bug。通过引入 Vagrant，开发人员从项目建立时即开始关注部署，编写部署代码，使得项目从第一天开发就是可部署的。

使用 Vagrant 后，合作开发人员只需要到 Git 下载项目源代码，“vagrant up”命令即可自动建立虚拟机、配置 IP 地址、运行 Puppet 命令部署，启动一个同生成环境一样的开发环境，即使生产环境是由多台服务器组成的[10]。

部署时工程师只需要拿到 Vagrant 内的 Puppet 脚本在服务器上运行“puppet apply default.pp”即可建立运行环境。甚至可在安装操作系统时指定 Puppet 脚本完成全自动部署。

4 结果分析

通过引入自动化理念，减少大量的人工操作步骤，减少人工操作过程可能造成的错误，减轻运维工程师的工作量，实现单个运维工程师可以管理更多的服务器，多个运维工程师之间可以形成互补，优化了运维环境。而开发环境和生产环境的一致，使得整个开发过程顺畅，部署成本减低，业务创新加快。

对各个服务器，通过查看配置文件，可获知服务器的功能。通过查看 Git 日志，可了解服务器配置变更的历史。为验证效果，对部分项目进行测试，比较结果如表 2 所示:

表 2 应用自动化后效果比较

比较项目	未采用自动化	采用自动化
数据中心基础架构源代码	无	压缩后只有 20MB，方便备份。
DNS 服务器备份占用存储大小	5GB	10KB
微信平台开发环境搭建时间	新参与开发人员需要学习环境配置技术，部署和测试，建立环境需要 60 分钟。	新参与开发人员无需熟悉环境配置技术，建立环境需要 5 分钟。
随书光盘镜像服务器安装时间	刻录光盘，安装过程操作，设置防火墙、自动更新、添加监控、备份，验证，3 小时。	复制已有配置文件，修改个性化配置，网络自动安装，1 小时，含人工介入 20 分钟。

5 结 语

数据中心规范化管理是一个长期过程，如果初期没有采用自动化方法，编写自动化脚本短期内会增加工作量，但是从长远分析则是保证了部署的质量，相当于为数据中心编写部署文档，同时也为今后的变更提供可追朔的途径。

在自动化运维中，由于全部组件使用标准化包管理器自带的二进制包，减少了维护的工作量，然而也带来版本更新不够及时等缺点。同时，开源社区 Docker 容器等新的部署打包工具也在演进中。这都是今后可研究和改进的方向。

参考文献：

[1] 梁春丽. IT 运维管理自动化是关键[J]. 金融科技时代，2012(2): 35-39. (Liang Chunli. Automation is the Key Factor in IT Operation and Maintenance [J]. Financial Technology

Time, 2012(2): 35- 39.)

[2] 李倩. 图书馆运维自动化的研究和规划[J]. 图书馆学研究, 2014(1): 25-27, 70. (Li Qian. Research and Planning in Operation and Maintenance in the Data Center of Library [J]. Researches in Library Science, 2014(1): 25-27, 70.)

[3] 杨义先, 姚文斌, 陈钊. 信息系统灾备技术综论[J]. 北京邮电大学学报, 2010, 33(2): 1- 6. (Yang Yixian, Yao Wenbin, Chen Zhao. Review of Disaster Backup and Recovery Technology of Information System [J]. Journal of Beijing University of Posts and Telecommunications, 2010, 33(2): 1-6.)

[4] 焦婧, 岳江红. 校园 IT 服务台的研究与实践[J]. 实验技术与管理, 2011, 28(5): 295-298. (Jiao Jing, Yue Jianghong. Study and Practice of Campus IT Service Desk in Colleges [J]. Experimental Technology and Management, 2011, 28(5): 295-298.)

[5] 李艳霞, 张倩, 齐芸芸, 等. 信息系统“主动式”运维保障工作的研究与实践[J]. 实验技术与管理, 2015, 32(2): 224-227. (Li Yanxia, Zhang Qian, Qi Yunyun, et al. Research and Practice of “Proactive” Operation and Maintenance Support for Information Systems [J]. Experimental Technology and Management, 2015, 32(2): 224-227.)

[6] 郑海山, 林俊伟. 图书馆数据中心运维中开源软件的应用[J]. 现代图书情报技术, 2014(6): 100-106. (Zheng Haishan, Lin Junwei. Application of Open Source Software in Operation and Maintenance in the Data Center of Library [J]. New Technology of Library and Information Service, 2014(6): 100-106.)

[7] 李爽. 操作系统自动化部署探究[D]. 广州: 华南理工大学, 2014. (Li Shuang. The Research on the Automation Deployment of Operation Systems [D]. Guangzhou: South China University of Technology, 2014.)

[8] 李新虎, 刘正伟, 刘俊朋. 基于 Puppet 工具的软件批量部署的实现[J]. 信息技术与标准化, 2014(6): 70-72, 75. (Li Xinhu, Liu Zhengwei, Liu Junpeng. Realization of Software Batch Deployment Based on Puppet Tool [J]. Information Technology & Standardization, 2014(6): 70-72, 75.)

[9] WhyVagrant [EB/OL]. [2015-04-19]. https://docs.vagrantup.com/v2/why-vagrant/index.html.

[10] 卞孟春. 基于 Jenkins 的持续集成方案设计与实现[D]. 北京: 中国科学院大学, 2014. (Bian Mengchun. Design and Application of Continuous Integration Scheme Based on Jenkins [D]. Beijing: University of Chinese Academy of Sciences, 2014: 26-27.)

收稿日期: 2015-05-04
收修改稿日期: 2015-05-20

The Automatic System for Infrastructure Deployment in the Data Center of Library

Zheng Haishan
(Information & Network Center, Xiamen University, Xiamen 361005, China)

Abstract: [**Objective**] This paper aims to solve problems of automatic deployment in the data center of university library with the increased servers. [**Context**] With the explosion of servers and more development projects, the data center of university library can’t undertake the heavy task. [**Methods**] Introducing the concept of automatic control, this paper uses automated scripts to maintain the infrastructure configuration of the data center. Use Vagrant to guarantee the coherence between development environment and production environment. [**Results**] This paper successfully solves two problems, including servers and operating systems of virtual machines reliance on components automatic deployment, and the inconsistency between development environment and production environment. Ultimately, reduce staff workload, and improve the development efficiency. [**Conclusions**] Applying automatic control strategies, the library’s data center reaches the target of clearness, standardization, automation.

Keywords: Data center　Operation and maintenance management　Puppet　Vagrant