Cobbler 自动化部署实践

By 05月30日 2016 Automated Ops Linux Cobbler

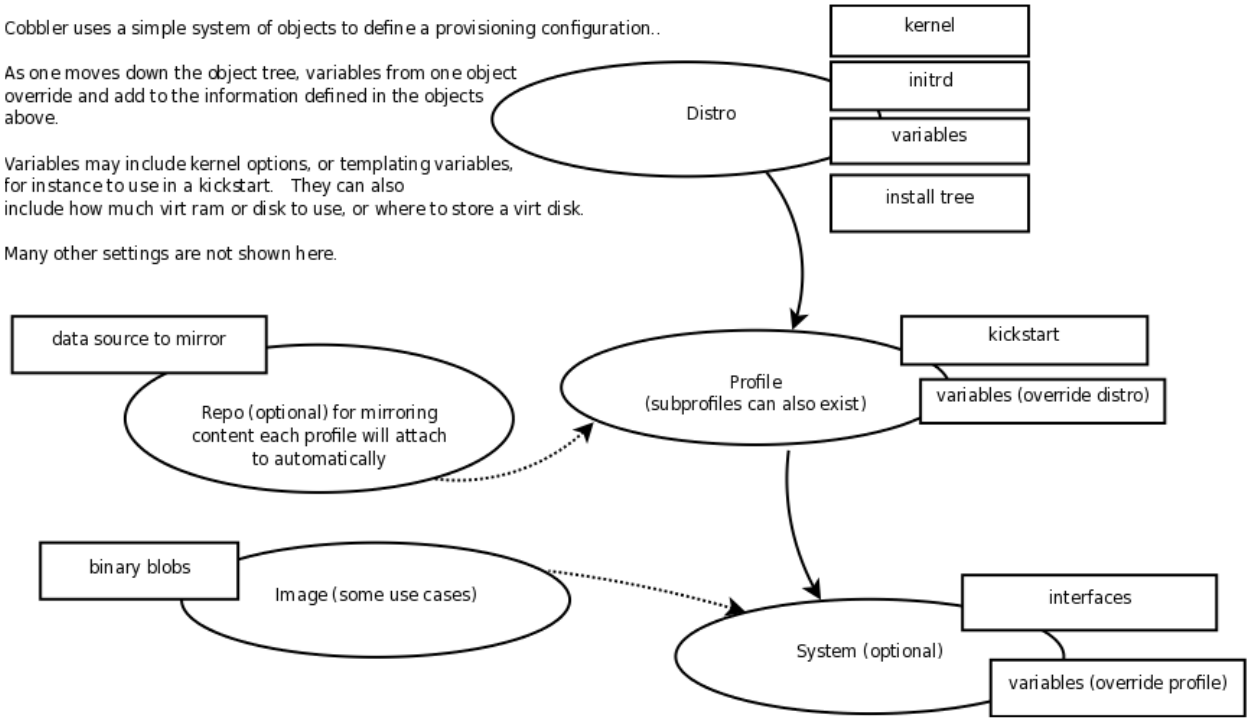
Cobbler 自动化部署实践

运维自动化在生产环境中占据着举足轻重的地位，尤其是面对几百台，几千台甚至几万台的服务器时，仅仅是安装操作系统，如果不通过自动化来完成， 根本是不可想象的。面对生产环境中不同服务器的需求，该如何实现批量部署多版本的操作系统呢？ Cobbler 便可以的满足这一实际需求，实现多版本操作系统批量部署。

1 Cobbler 简介

Cobbler 是一个快速网络安装 Linux 的服务，而且经过调整也可以支持网络安装 Windows。
该工具使用 Python 开发，小巧轻便（才 15 k 行 Python 代码），使用简单的命令即可完成PXE网络安装环境的配置， 同时还可以管理 DHCP、DNS、TFTP、RSYNC 以及 YUM 仓库、构造系统 ISO 镜像。
Cobbler 支持命令行管理，Web 界面管理，还提供了 API 接口，可以方便二次开发使用。 Cobbler 客户端 koan 支持虚拟机安装和操作系统重新安装，使重装系统更便捷。

官网 Cobbler 系统对象模型图



distro

- 发行版
- 面对不同的操作系统
- 面对同一个操作系统不同的版本

profile

- 核心特性是通过 kickstart 来部署

system

- 主要目的配置网络接口

2 Cobbler 功能

使用 Cobbler，您无需进行人工干预即可安装机器。Cobbler 设置一个 PXE 引导环境（它还可使用 yaboot 支并控制与安装相关的所有方面，比如网络引导服务（DHCP 和 TFTP）与存储库镜像。当希望安装一台新机器时，

使用一个以前定义的模板来配置 DHCP 服务（如果启用了管理 DHCP）
将一个存储库（yum 或 rsync）建立镜像或解压缩一个媒介，以注册一个新操作系统
在 DHCP 配置文件中为需要安装的机器创建一个条目，并使用您指定的参数（IP 和 MAC 地址）
在 TFTP 服务目录下创建适当的 PXE 文件
重新启动 DHCP 服务以反映更改
重新启动机器以开始安装（如果电源管理已启用）

Cobbler 支持众多的发行版：Red Hat、Fedora、CentOS、Debian、Ubuntu 和 SuSE。当添加一个操作系统（通
Cobbler 知道如何解压缩合适的文件并调整网络服务，以正确引导机器。

Cobbler 可使用 kickstart 模板。基于 Red Hat 或 Fedora 的系统使用 kickstart 文件来自动化安装流程。
通过使用模板，您就会拥有基本的 kickstart 模板，然后定义如何针对一种配置文件或机器配置而替换其中的
例如，一个模板可能包含两个变量 $domain 和 $machine_name。在 Cobbler 配置中，一个配置文件指定 doma
并且每台使用该配置文件的机器在 machine_name 变量中指定其名称。该配置文件中的所有机器都使用相同的
进行配置，但每台机器拥有其自己的机器名称。您仍然可以使用 kickstart 模板在不同的域中安装其他机器并

为了协助管理系统，Cobbler 可通过 fence scripts 连接到各种电源管理环境。

Cobbler 支持 apc_snmp、bladecenter、bullpap、drac、ether_wake、ilo、integrity、ipmilan、ipmitool、
要重新安装一台机器，可运行 reboot system foo 命令，而且 Cobbler 会使用必要的凭据和信息来为您运行恰

除了这些特性，还可使用一个配置管理系统（CMS）。您有两种选择：该工具内的一个内部系统，或者集成一个现
借助内部系统，您可以指定文件模板，这些模板会依据配置参数进行处理（与 kickstart 模板的处理方式一样
如果必须自动将配置文件部署到特定机器，那么此功能很有用。

使用 koan 客户端，Cobbler 可从客户端配置虚拟机并重新安装系统。我不会讨论配置管理和 koan 特性，因为
但是，它们是值得研究的有用特性。

3 基础环境准备

```
[root@linux-node1 ~]# cat /etc/redhat-release   # 查看系统版本
CentOS Linux release 7.2.1511 (Core)
[root@linux-node1 ~]# uname -r          # 查看内核版本
3.10.0-327.18.2.el7.x86_64
[root@linux-node1 ~]# getenforce        # 确认 SELinux 关闭
Disabled
[root@linux-node1 ~]# systemctl status firewalld        # 确认防火墙关闭
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled
   Active: inactive (dead)
[root@linux-node1 ~]# ip a | grep eth0  # 查看 IP 地址
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    inet 192.168.56.11/24 brd 192.168.56.255 scope global eth0
[root@linux-node1 ~]# hostname          # 查看主机名
linux-node1
[root@linux-node1 ~]# rpm -ivh http://mirrors.aliyun.com/epel/epel-release-latest-7.noarch.rpm
Retrieving http://mirrors.aliyun.com/epel/epel-release-latest-7.noarch.rpm
Preparing...                          ################################# [100%]
Updating / installing...
   1:epel-release-7-6                 ################################# [100%]
```

注意：

1. 虚拟机网卡采用 NAT 模式并关闭，不要使用桥接模式，因为我们会搭建 DHCP 服务器，在同一局域网多个 DHCP 服务会有冲突，并且导致实践失败。
2. VMware 的 NAT 模式的 DHCP 服务也关闭，避免干扰。

4 Cobbler 安装配置

4.1 安装 Cobbler 及相关软件包

```
[root@linux-node1 ~]# yum install dhcp tftp-server xinetd httpd cobbler cobbler-web pykickstar
[root@linux-node1 ~]# rpm -qa | grep -E "dhcp|tftp-server|xinetd|httpd|cobbler|cobbler-web|pyk
dhcp-common-4.2.5-42.el7.centos.x86_64
cobbler-2.6.11-1.el7.x86_64                    # Cobbler 程序包
pykickstart-1.99.66.6-1.el7.noarch      # Cobbler 检查 kickstart 语法错误
httpd-tools-2.4.6-40.el7.centos.1.x86_64
dhcp-libs-4.2.5-42.el7.centos.x86_64
tftp-server-5.2-12.el7.x86_64           # tftp 服务
dhcp-4.2.5-42.el7.centos.x86_64         # dchp 服务
xinetd-2.3.15-12.el7.x86_64
httpd-2.4.6-40.el7.centos.1.x86_64      # Apache Web 服务
cobbler-web-2.6.11-1.el7.noarch         # Cobbler 的 Web 程序包
[root@linux-node1 ~]# rpm -ql cobbler   #  查看安装的文件，下面列出部分
/etc/cobbler                    # 配置文件目录
/etc/cobbler/settings           # Cobbler 主配置文件，这个文件是 YAML 格式，Cobbler 是 Python 写的程
/etc/cobbler/dhcp.template      # DHCP 服务的配置模板
/etc/cobbler/tftpd.template     # tftp 服务的配置模板
/etc/cobbler/rsync.template     # rsync 服务的配置模板
/etc/cobbler/iso                # ISO 模板配置文件目录
/etc/cobbler/pxe                # PXE 模板文件目录
/etc/cobbler/power              # 电源的配置文件目录
/etc/cobbler/users.conf         # Web 服务授权配置文件
/etc/cobbler/users.digest       # 用于 Web 访问的用户名密码配置文件
/etc/cobbler/dnsmasq.template # DNS 服务的配置模板
/etc/cobbler/modules.conf       # Cobbler 模块配置文件
/var/lib/cobbler                # Cobbler 数据目录
/var/lib/cobbler/config         # 配置文件
/var/lib/cobbler/kickstarts     # 默认存放 kickstart 文件目录
/var/lib/cobbler/loaders        # 存放的各种引导程序
/var/www/cobbler                # 系统安装镜像目录
/var/www/cobbler/ks_mirror      # 导入的系统镜像列表
/var/www/cobbler/images         # 导入的系统镜像启动文件
/var/www/cobbler/repo_mirror  # yum 源存储目录
/var/log/cobbler                # 日志目录
/var/log/cobbler/install.log  # 客户端系统安装日志
/var/log/cobbler/cobbler.log  # Cobbler 日志
```

4.2 配置 Cobbler

Cobbler 的运行依赖于 DHCP、TFTP、RSYNC 及 DNS 服务，其中 DHCP 可由 dhcpd（ISC）提供，也可由 dnsmasq 提供；TFTP 可由 tftp-server 程序包提供，也可由 Cobbler 功能提供， rsync 由 rsync 程序包提供，DNS 可由 bind 提供，也可由 dnsmasq 提供。
Cobbler可自行管理这些服务中的部分甚至是全部，但需要配置文件 /etc/cobbler/settings 中的“manange_dhcp”、“manager_tftpd”、“manager_rsync”、“manager_dns”分别来进行定义， 另外，由于各种服务都有着不同的实现方式，如若需要进行自定义，需要通过修改 /etc/cobbler/modules.conf 配置文件中各服务的模块参数的值来实现。

```
[root@linux-node1 ~]# systemctl start httpd             # 启动 Apache httpd 服务
[root@linux-node1 ~]# systemctl start cobblerd  # 启动 Cobbler 服务
[root@linux-node1 ~]# systemctl status httpd cobblerd # 查看 Apache httpd、Cobbler 服务状态
```

```
# 检查配置文件，需要在 cobblerd 和 httpd 启动的情况下检查

[root@linux-node1 ~]# cobbler check
The following are potential configuration items that you may want to fix:

1 : The ‘server’ field in /etc/cobbler/settings must be set to something other than localhost,
hostname or IP for the boot server as reachable by all machines that will use it.
2 : For PXE to be functional, the ‘next_server’ field in /etc/cobbler/settings must be set to
server on the PXE network.
3 : change ‘disable’ to ‘no’ in /etc/xinetd.d/tftp
4 : some network boot-loaders are missing from /var/lib/cobbler/loaders, you may run ‘cobbler
x86/x86_64 netbooting, you may ensure that you have installed a *recent* version of the syslin
Files in this directory, should you want to support all architectures, should include pxelinux
command is the easiest way to resolve these requirements.
5 : enable and start rsyncd.service with systemctl
6 : debmirror package is not installed, it will be required to manage debian deployments and r
7 : The default password used by the sample templates for newly installed machines (default_pa
and should be changed, try: “openssl passwd -1 -salt ‘random-phrase-here’ ‘your-password-here'
8 : fencing tools were not found, and are required to use the (optional) power management feat

Restart cobblerd and then run ‘cobbler sync’ to apply changes.

# 如上各问题解决方法如下：
# 第 1、2、7 个问题，顺便修改其他功能
[root@linux-node1 ~]# cp /etc/cobbler/settings{,.ori}   # 备份

# 修改 /etc/cobbler/settings 文件中的 server 参数的值为提供 cobbler 服务的主机相应的 IP 地址或主机
[root@linux-node1 ~]# sed -i "s#server: 127.0.0.1#server: 192.168.56.11#" /etc/cobbler/setting
[root@linux-node1 ~]# grep "^server: " /etc/cobbler/settings
server: 192.168.56.11

# 修改 /etc/cobbler/settings 文件中的 next_server 参数的值为提供 PXE 服务的主机相应的 IP 地址
[root@linux-node1 ~]# sed -i "s#next_server: 127.0.0.1#next_server: 192.168.56.11#" /etc/cobbl
[root@linux-node1 ~]# grep "^next_server: " /etc/cobbler/settings
next_server: 192.168.56.11

# 防止循环装系统，适用于服务器第一启动项是 PXE 启动
[root@linux-node1 ~]# sed -i "s#pxe_just_once: 0#pxe_just_once: 1#" /etc/cobbler/settings
[root@linux-node1 ~]# grep "^pxe_just_once: " /etc/cobbler/settings
pxe_just_once: 1

# 设置新装系统的默认 root 密码 cobbler
[root@linux-node1 ~]# openssl passwd -1 -salt 'cobbler' 'cobbler'
$1$cobbler$M6SE55xZodWc9.vAKLJs6.
# 修改 /etc/cobbler/settings 文件中的 default_password_crypted 参数的值为上面生成的密码串
[root@linux-node1 ~]# vim /etc/cobbler/settings

# 第 3 个问题，修改 /etc/xinetd.d/tftp 文件中的 disable 参数修改为 disable = no
[root@linux-node1 ~]# sed -i "/disable/ {s#yes#no#}" /etc/xinetd.d/tftp
[root@linux-node1 ~]# grep "disable" /etc/xinetd.d/tftp
        disable                 = no

# 第 4 个问题，执行 cobbler get-loaders 命令即可，会自动从官网下载
[root@linux-node1 ~]# cobbler get-loaders

# 第 5 个问题，配置 rsyncd 开机启动，启动 rsyncd
[root@linux-node1 ~]# systemctl enable rsyncd
[root@linux-node1 ~]# systemctl start xinetd rsyncd
[root@linux-node1 ~]# systemctl status rsyncd

# 第 6 个问题，和 Debian 系统相关，不需要
```

```
# 第 8 个问题，fence设备相关，暂不需要

# 上面相关问题配置完后，重启 Cobbler，并再次执行 cobbler check 检查确认（如下算解决了相关问题）
[root@linux-node1 ~]# systemctl restart cobblerd
[root@linux-node1 ~]# cobbler check
The following are potential configuration items that you may want to fix:

1 : debmirror package is not installed, it will be required to manage debian deployments and r
2 : fencing tools were not found, and are required to use the (optional) power management feat

Restart cobblerd and then run 'cobbler sync' to apply changes.
```

4.3 配置 DHCP

```
# 配置使用 Cobbler 管理 DHCP
[root@linux-node1 ~]# sed -i "s#manage_dhcp: 0#manage_dhcp: 1#" /etc/cobbler/settings
[root@linux-node1 ~]# grep "^manage_dhcp: " /etc/cobbler/settings
manage_dhcp: 1
# 修改 Cobbler 的 DHCP 模版，不要直接修改 DHCP 本身的配置文件，因为 Cobbler 会覆盖
[root@linux-node1 ~]# cp /etc/cobbler/dhcp.template{,.ori}
[root@linux-node1 ~]# vim /etc/cobbler/dhcp.template    # 列出修改过的相关字段
...
subnet 192.168.56.0 netmask 255.255.255.0 {
     option routers             192.168.56.2;
     option domain-name-servers 192.168.56.2;
     option subnet-mask         255.255.255.0;
     range dynamic-bootp        192.168.56.100 192.168.56.254;
     default-lease-time         21600;
     max-lease-time             43200;
     next-server                $next_server;
...
```

4.4 同步 Cobbler 配置

```
# 同步最新 Cobbler 配置，它会根据配置自动修改 DHCP 等服务

[root@linux-node1 ~]# systemctl restart cobblerd
[root@linux-node1 ~]# cobbler sync              # 同步所有配置，可以仔细看一下 sync 做了什么
task started: 2016-05-28_091148_sync
task started (id=Sync, time=Sat May 28 09:11:48 2016)
running pre-sync triggers
cleaning trees
removing: /var/www/cobbler/images/CentOS-7-x86_64
removing: /var/lib/tftpboot/pxelinux.cfg/01-00-0c-29-1d-bf-f1
removing: /var/lib/tftpboot/pxelinux.cfg/default
removing: /var/lib/tftpboot/grub/images
removing: /var/lib/tftpboot/grub/grub-x86.efi
removing: /var/lib/tftpboot/grub/grub-x86_64.efi
removing: /var/lib/tftpboot/grub/01-00-0C-29-1D-BF-F1
removing: /var/lib/tftpboot/grub/efidefault
removing: /var/lib/tftpboot/images/CentOS-7-x86_64
removing: /var/lib/tftpboot/s390x/profile_list
copying bootloaders
trying hardlink /var/lib/cobbler/loaders/grub-x86.efi -> /var/lib/tftpboot/grub/grub-x86.efi
```

```
trying hardlink /var/lib/cobbler/loaders/grub-x86_64.efi -> /var/lib/tftpboot/grub/grub-x86_64
copying distros to tftpboot
copying files for distro: CentOS-7-x86_64
trying hardlink /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/vmlinuz -> /var/lib/
trying hardlink /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/initrd.img -> /var/l
copying images
generating PXE configuration files
generating: /var/lib/tftpboot/pxelinux.cfg/01-00-0c-29-1d-bf-f1
generating: /var/lib/tftpboot/grub/01-00-0C-29-1D-BF-F1
generating PXE menu structure
copying files for distro: CentOS-7-x86_64
trying hardlink /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/vmlinuz -> /var/www/
trying hardlink /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/initrd.img -> /var/w
Writing template files for CentOS-7-x86_64
rendering DHCP files
generating /etc/dhcp/dhcpd.conf
rendering TFTPD files
generating /etc/xinetd.d/tftp
processing boot_files for distro: CentOS-7-x86_64
cleaning link caches
running post-sync triggers
running python triggers from /var/lib/cobbler/triggers/sync/post/*
running python trigger cobbler.modules.sync_post_restart_services
running: dhcpd -t -q
received on stdout:
received on stderr:
running: service dhcpd restart
received on stdout:
received on stderr: Redirecting to /bin/systemctl restart  dhcpd.service

running shell triggers from /var/lib/cobbler/triggers/sync/post/*
running python triggers from /var/lib/cobbler/triggers/change/*
running python trigger cobbler.modules.scm_track
running shell triggers from /var/lib/cobbler/triggers/change/*
*** TASK COMPLETE ***

[root@linux-node1 ~]# head -7 /etc/dhcp/dhcpd.conf              # 再看一下 DHCP 的配置文件
# ******************************************************************
# Cobbler managed dhcpd.conf file
# generated from cobbler dhcp.conf template (Sat May 28 01:11:49 2016)
# Do NOT make changes to /etc/dhcpd.conf. Instead, make your changes
# in /etc/cobbler/dhcp.template, as /etc/dhcpd.conf will be
# overwritten.
# ******************************************************************
```

4.5 配置开机启动

```
[root@linux-node1 ~]# systemctl enable dhcpd xinetd httpd cobblerd
[root@linux-node1 ~]# systemctl is-enabled dhcpd xinetd httpd cobblerd
enabled
enabled
enabled
enabled
```

5 Cobbler 命令行管理

5.1 查看命令帮助

```
[root@linux-node1 ~]# cobbler
usage
=====
cobbler <distro|profile|system|repo|image|mgmtclass|package|file> ...
        [add|edit|copy|getks*|list|remove|rename|report] [options|--help]
cobbler <aclsetup|buildiso|import|list|replicate|report|reposync|sync|validateks|version|signa

[root@linux-node1 ~]# cobbler import --help             # 导入镜像帮助
Usage: cobbler [options]

Options:
  -h, --help            show this help message and exit
  --arch=ARCH           OS architecture being imported
  --breed=BREED         the breed being imported
  --os-version=OS_VERSION
                        the version being imported
  --path=PATH           local path or rsync location
  --name=NAME           name, ex 'RHEL-5'
  --available-as=AVAILABLE_AS
                        tree is here, don't mirror
  --kickstart=KICKSTART_FILE
                        assign this kickstart file
  --rsync-flags=RSYNC_FLAGS
                        pass additional flags to rsync


cobbler check    核对当前设置是否有问题
cobbler list     列出所有的 cobbler 元素
cobbler report   列出元素的详细信息
cobbler sync     同步配置到数据目录，更改配置最好都要执行下
cobbler reposync 同步 yum 仓库
cobbler distro   查看导入的发行版系统信息
cobbler profile  查看配置信息
cobbler system   查看添加的系统信息
```

5.2 管理 distro

> Cobbler 变得可用的第一步为定义 distro，其可以通过为其指定外部的安装引导内核及 ramdisk 文件的方式实现。

如果已经有完成的安装树（如 OS 的安装镜像）则推荐使用 improt 导入的方式进行。

导入镜像

```
[root@linux-node1 ~]# mount /dev/cdrom /mnt             # 挂载 CentOS 7 的系统镜像
mount: /dev/sr0 is write-protected, mounting read-only
[root@linux-node1 ~]# cobbler import --path=/mnt/ --name=CentOS-7-x86_64 --arch=x86_64
# --path 镜像路径
# --name 为安装源定义一个名字
# --arch 指定安装源是 32 位、64 位、ia64，目前支持的选项有：x86|x86_64|ia64
# 安装源的唯一标示就是根据 name 参数来定义，本例导入成功后，安装源的唯一标示就是：CentOS-7-x86_64，如
[root@linux-node1 ~]# cobbler distro list               # 查看镜像列表
   CentOS-7-x86_64
# 镜像存放目录，Cobbler 会将镜像中的所有安装文件拷贝到本地一份，放在 /var/www/cobbler/ks_mirror 下的
# 因此 /var/www/cobbler 目录必须具有足够容纳安装文件的空间。
```

```
[root@linux-node1 ~]# ll /var/www/cobbler/ks_mirror/
total 4
dr-xr-xr-x 8 root root 4096 Dec 10 07:14 CentOS-7-x86_64
drwxr-xr-x 2 root root   35 May 25 04:27 config
[root@linux-node1 ~]# cobbler profile list              # 导入 distro 会自动生成 profile
```

如果有 kickstart 文件，也可以使用–kickstart=/path/to/kickstart_file 进行导入，import 会自动为导入的 distro 生成一个 profile

```
# 查看安装镜像文件信息
[root@linux-node1 ~]# cobbler distro report --name=CentOS-7-x86_64
Name                           : CentOS-7-x86_64
Architecture                   : x86_64
TFTP Boot Files                : {}
Breed                          : redhat
Comment                        :
Fetchable Files                : {}
Initrd                         : /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/ini
Kernel                         : /var/www/cobbler/ks_mirror/CentOS-7-x86_64/images/pxeboot/vml
Kernel Options                 : {}
Kernel Options (Post Install)  : {}
Kickstart Metadata             : {'tree': 'http://@@http_server@@/cblr/links/CentOS-7-x86_64'}
Management Classes             : []
OS Version                     : rhel7
Owners                         : ['admin']
Red Hat Management Key         : <<inherit>>
Red Hat Management Server      : <<inherit>>
Template Files                 : {}
```

5.3 管理 profile

Cobbler使用 profile 来为特定的需求类别提供所需要安装的配置，即在 distro 的基础上通过提供 kiskstart 文件来生成一个特定的系统安装配置。 distro 的 profile 可以出现在 PXE 的引导菜单中作为安装的选择之一

```
# 查看所有的 profile 设置
[root@linux-node1 ~]# cobbler profile report

# 查看指定的 profile 设置
[root@linux-node1 ~]# cobbler profile report --name=CentOS-7-x86_64
[root@linux-node1 ~]# cobbler profile report --name=CentOS-7-x86_64
Name                           : CentOS-7-x86_64
TFTP Boot Files                : {}
Comment                        :
DHCP Tag                       : default
Distribution                   : CentOS-7-x86_64
Enable gPXE?                   : 0
Enable PXE Menu?               : 1
Fetchable Files                : {}
Kernel Options                 : {'biosdevname': '0', 'net.ifnames': '0'}
Kernel Options (Post Install)  : {}
Kickstart                      : /var/lib/cobbler/kickstarts/sample_end.ks  # 默认 ks 文件
Kickstart Metadata             : {}
Management Classes             : []
Management Parameters          : <<inherit>>
```

```
Name Servers                    : []
Name Servers Search Path        : []
Owners                          : ['admin']
Parent Profile                  :
Internal proxy                  :
Red Hat Management Key          : <<inherit>>
Red Hat Management Server       : <<inherit>>
Repos                           : []
Server Override                 : <<inherit>>
Template Files                  : {}
Virt Auto Boot                  : 1
Virt Bridge                     : xenbr0
Virt CPUs                       : 1
Virt Disk Driver Type           : raw
Virt File Size(GB)              : 5
Virt Path                       :
Virt RAM (MB)                   : 512
Virt Type                       : kvm

# Cobbler 的 ks.cfg 文件存放位置
[root@linux-node1 ~]# ls /var/lib/cobbler/kickstarts/
CentOS-7-x86_64.cfg  esxi4-ks.cfg  install_profiles  pxerescue.ks         sample_end.ks（默认使
default.ks           esxi5-ks.cfg  legacy.ks         sample_autoyast.xml  sample_esx4.ks  samp
# 在第一次导入系统镜像后，Cobbler 会给镜像指定一个默认的 kickstart 自动安装文件在 /var/lib/cobbler/

[root@linux-node1 ~]# cd /var/lib/cobbler/kickstarts/
[root@linux-node1 kickstarts]# rz               # 上传准备好的 ks 文件

# 编辑 profile，修改关联的 ks 文件
[root@linux-node1 kickstarts]# cobbler profile edit --name=CentOS-7-x86_64 --kickstart=/var/li

# 修改安装系统的内核参数，在 CentOS 7 系统有一个地方变了，就是网卡名变成 eno16777736 这种形式，但是为
# 我们需要将它变成我们常用的 eth0，因此使用下面的参数。但要注意是 CentOS 7 才需要下面的步骤，CentOS 6
[root@linux-node1 kickstarts]# cobbler profile edit --name=CentOS-7-x86_64 --kopts='net.ifname
[root@linux-node1 kickstarts]# cobbler profile report CentOS-7-x86_64
Name                            : CentOS-7-x86_64
TFTP Boot Files                 : {}
Comment                         :
DHCP Tag                        : default
Distribution                    : CentOS-7-x86_64
Enable gPXE?                    : 0
Enable PXE Menu?                : 1
Fetchable Files                 : {}
Kernel Options                  : {'biosdevname': '0', 'net.ifnames': '0'}
Kernel Options (Post Install)   : {}
Kickstart                       : /var/lib/cobbler/kickstarts/CentOS-7-x86_64.cfg
Kickstart Metadata              : {}
Management Classes              : []
Management Parameters           : <<inherit>>
Name Servers                    : []
Name Servers Search Path        : []
Owners                          : ['admin']
Parent Profile                  :
Internal proxy                  :
Red Hat Management Key          : <<inherit>>
Red Hat Management Server       : <<inherit>>
Repos                           : []
Server Override                 : <<inherit>>
Template Files                  : {}
Virt Auto Boot                  : 1
Virt Bridge                     : xenbr0
Virt CPUs                       : 1
```

```
Virt Disk Driver Type          : raw
Virt File Size(GB)             : 5
Virt Path                      :
Virt RAM (MB)                  : 512
Virt Type                      : kvm

# 每次修改完都要同步一次
[root@linux-node1 ~]# cobbler sync
```

5.4 管理 repo

```
# 新部署机器安装 yum 源，并同步。建议使用内网 yum 源，在这里使用阿里云 yum 源
# 1. 添加 repo
cobbler repo add --name=centos-7.2-openstack-mitaka --mirror=http://mirrors.aliyun.com/centos/
# 2. 同步repo
cobbler reposync
# 3. 添加 repo 到对应的 profile
cobbler profile edit --name=Centos-7-x86_64  --repos="centos-7.2-openstack-mitaka"
# 4. 修改 kickstart文件。添加 $yum_config_stanza 到 %post %end 中间，如下：
%post
$yum_config_stanza
%end
# 5. 添加定时任务，定期同步 repo
echo "0 3 * * * /usr/bin/cobbler reposync --tries=3 --no-fail" >> /var/spool/cron/root

# 重装系统完成后的机器配置的 cobbler repo 如下：
[root@localhost ~]# cat /etc/yum.repos.d/cobbler-config.repo
[core-0]
name=core-0
baseurl=http://192.168.56.11/cobbler/ks_mirror/CentOS-7-x86_64
enabled=1
gpgcheck=0
priority=1


[centos-7.2-openstack-mitaka]
name=centos-7.2-openstack-mitaka
baseurl=http://192.168.56.11/cobbler/repo_mirror/centos-7.2-openstack-mitaka
enabled=1
priority=99
gpgcheck=0
```

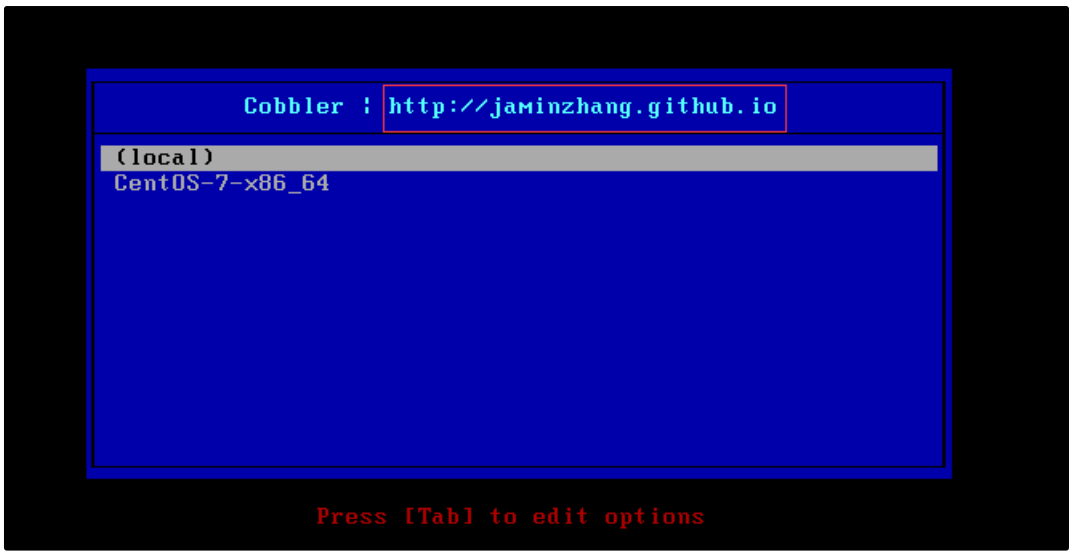
6 安装系统

6.1 安装新系统

新建一台虚拟机，通过网络启动即可。


```
# 修改 Cobbler 提示
[root@linux-node1 ~]# vim /etc/cobbler/pxe/pxedefault.template  # 修改成自定义
MENU TITLE Cobbler | http://jaminzhang.github.io
[root@linux-node1 ~]# cobbler sync # 修改配置都要同步
```

选择第二项就可以继续安装系统了。



ks.cfg 模板文件: Cobbler-CentOS-7.2-x86_64.cfg

6.2 使用 koan 客户端进行系统重装

```
# 安装 koan 客户端
[root@linux-node1 ~]# yum install koan -y
# 查询 Cobbler 服务器上 profile
[root@linux-node1 ~]# koan --server=192.168.56.11 --list=profiles
- looking for Cobbler at http://192.168.56.11:80/cobbler_api
CentOS-7-x86_64
CentOS-6-x86_64
# 指定要重装的系统
[root@linux-node1 ~]# koan --replace-self --server=192.168.56.11 --profile=CentOS-6-x86_64
```

如何避免 koan 安装错误机器，或者 cobbler 自动化安装错误机器。
解决方法：环境设计：配置装机 VLAN

6.3 定制化安装

可能从学习 kickstart 开始就有人想怎样能够指定某台服务器使用指定 ks 文件，kickstart 实现这功能可能比较复杂，但是 Cobbler 就很简单了。 区分一台服务器的最简单的方法就是物理MAC地址。
物理服务器的 MAC 地址在服务器上的标签上写了

物理服务器的 MAC 地址在服务器上的标签上写了。

根据机器的 MAC 地址，自动绑定 IP，网关，DNS 等

```
[root@linux-node1 ~]# cobbler system add --name=jaminzhang.github.io --mac=00:50:56:35:46:6E -
--interface=eth0 --static=1 --hostname=jaminzhang.github.io --name-servers="223.5.5.5 223.6.6.

# --name 自定义，但不能重复
# 查看定义的列表c
[root@linux-node1 ~]# cobbler system list
   jaminzhang.github.io

[root@linux-node1 ~]# cobbler sync
```

不进行询问,自动安装

```
Network boot from Intel E1000
Copyright (C) 2003-2014  VMware, Inc.
Copyright (C) 1997-2000  Intel Corporation

CLIENT MAC ADDR: 00 50 56 35 46 6E  GUID: 564DCFD6-5F8A-79B7-15EE-EB3F8002CDCB
CLIENT IP: 192.168.56.118  MASK: 255.255.255.0  DHCP IP: 192.168.56.11
GATEWAY IP: 192.168.56.1

PXELINUX 3.86 2010-04-01  Copyright (C) 1994-2010 H. Peter Anvin et al
!PXE entry point found (we hope) at 9E0E:0106 via plan A
UNDI code segment at 9E0E len 0BCE
UNDI data segment at 9878 len 5960
Getting cached packet 01 02 03
My IP address seems to be C0A83876 192.168.56.118
ip=192.168.56.118:192.168.56.11:192.168.56.1:255.255.255.0
TFTP prefix: /
Trying to load: pxelinux.cfg/564dcfd6-5f8a-79b7-15ee-eb3f8002cdcb
Trying to load: pxelinux.cfg/01-00-50-56-35-46-6e
Loading /images/CentOS-7-x86_64/vmlinuz.....................................
.....................................
Loading /images/CentOS-7-x86_64/initrd.img.....................................
.....................................
.....................................
..............
```

登陆系统检查相关信息

```
[root@jaminzhang ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:50:56:35:46:6e brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.118/24 brd 192.168.56.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe35:466e/64 scope link
       valid_lft forever preferred_lft forever
[root@jaminzhang ~]# hostname
jaminzhang.github.io
[root@jaminzhang ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search jaminzhang.github.io
nameserver 223.5.5.5
nameserver 223.6.6.6
[root@jaminzhang ~]# _
```

6.4 服务器采购及系统安装流程

1. 服务器采购
2. 服务器验收并设置 RAID
3. 服务商提供验收单，运维验收负责人签字。

4. 服务器上架
5. 资产录入。
6. 开始自动化安装。

    1. 将新服务器划入装机 VLAN

    2. 根据资产清单上的 MAC 地址，自定义安装。 1.机房 2. 机房区域 3. 机柜 4. 服务器位置 5. 服务器网线接入端口 6. 该端口 MAC 地址 7.profile ks.cfg 中指定操作系统 分区等、预分配的 IP 地址、主机名、子网、网关、DNS、角色等。

    3. 自动化装机平台，安装。 例子：
    MAC: 00:50:56:35:46:6E
    IP: 192.168.56.118
    掩码：255.255.255.0
    网关：192.168.56.2
    主机名：jaminzhang.github.io
    DNS：223.5.5.5 223.6.6.6
    根据 6.3 中的定制化安装配置如下 system:
    cobbler system add –name=jaminzhang.github.io –mac=00:50:56:35:46:6E –profile=CentOS-7-x86_64 –ip-address=192.168.56.118 –subnet=255.255.255.0 –gateway=192.168.56.2 \
    –interface=eth0 –static=1 –hostname=jaminzhang.github.io –name-servers=”223.5.5.5 223.6.6.6” –kickstart=/var/lib/cobbler/kickstarts/CentOS-7-x86_64.cfg

7 Cobbler Web 管理配置

新版 Cobbler 的 Web 界面使用的是 https
登录URL: https://192.168.56.11/cobbler_web

7.1 配置账号密码

cobbler_web 支持多种认证方式，如 authn_configfile、authn_ldap 或 authn_pam 等，默认为authn_denyall，即拒绝所有用户登陆。

下面说明三种能认证用户登录 cobbler_web 的方式

1. 使用 authn_pam 模块认证 cobbler_web 用户

首先修改 modules 中的 [authentication] 段中的 module 参数的值为 authn_pam 接着创建系统用户，并为用户设定密码，而后将设定的系统用户添加至 cobbler_web 的 admin 组中，修改 /etc/cobbler/users.conf文件， 将设定的用户添加为 admin 参数的值即可

1. 使用 authn_configfile 模块认证 cobbler_web 用户 首先修改 modules 中的 [authentication] 段中的 module 参数的值为 authn_configfile 添加用户时，需要为 htdigest 命令使用 “-c” etc/cobbler/users.digest，后续添加其他用户则不能再使用，同步 cobbler，重启 httpd 以及 cobbler

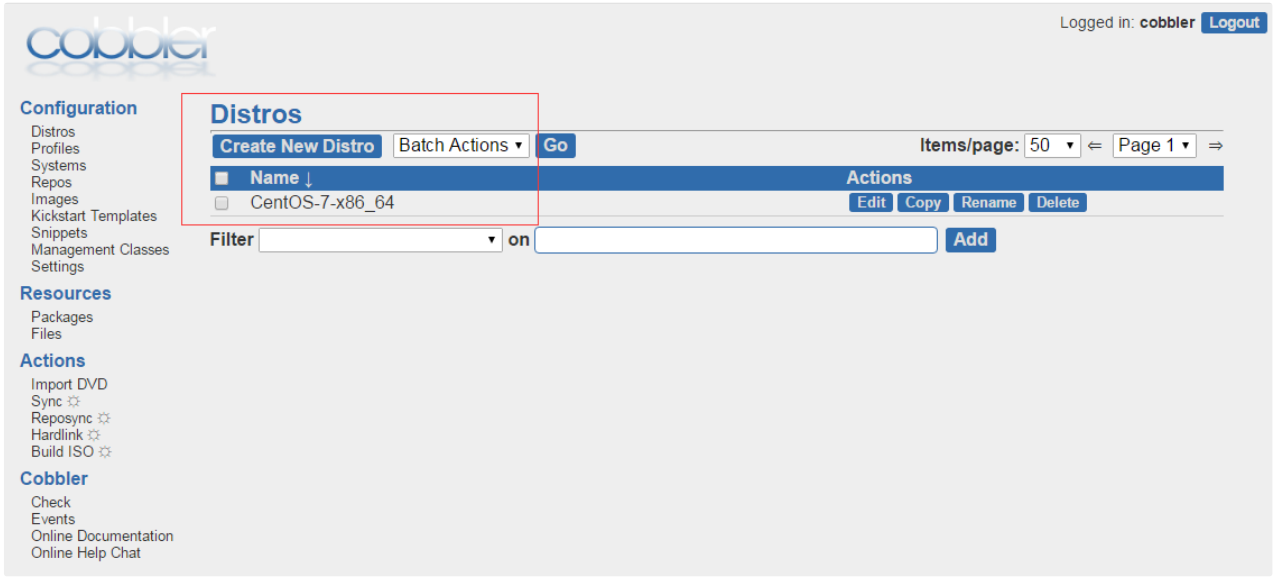
2. 使用 Cobbler 默认的 Web 账号密码认证

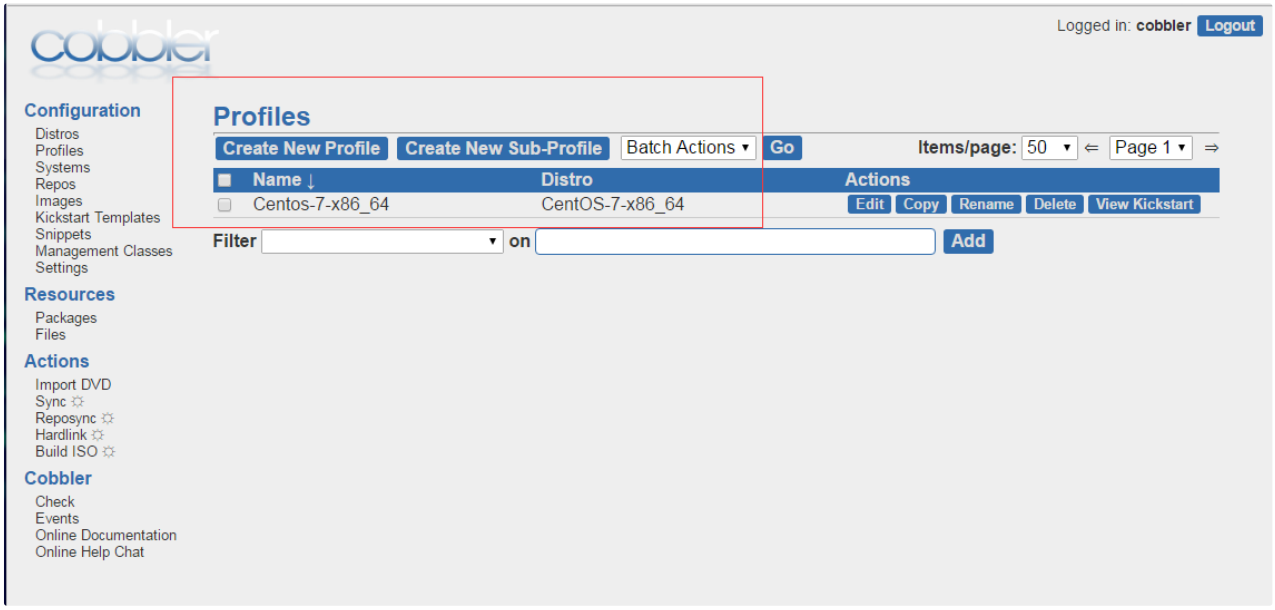
    user:cobbler
    password:cobbler

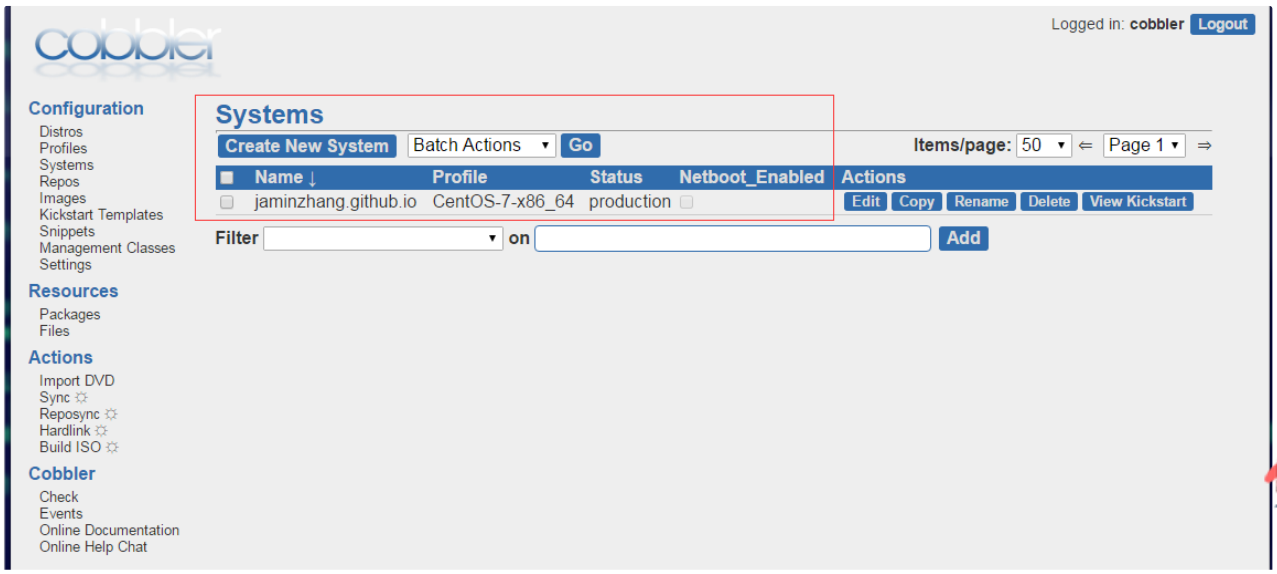
```
/etc/cobbler/users.conf       # Web服务授权配置文件
/etc/cobbler/users.digest     # 用于web访问的用户名密码配置文件
[root@linux-node1 ~]# cat /etc/cobbler/users.digest
cobbler:Cobbler:a2d6bae81669d707b72c0bd9806e01f3
# 设置 Cobbler Web 用户登陆密码
[root@linux-node1 ~]# htdigest /etc/cobbler/users.digest "Cobbler" cobbler
```

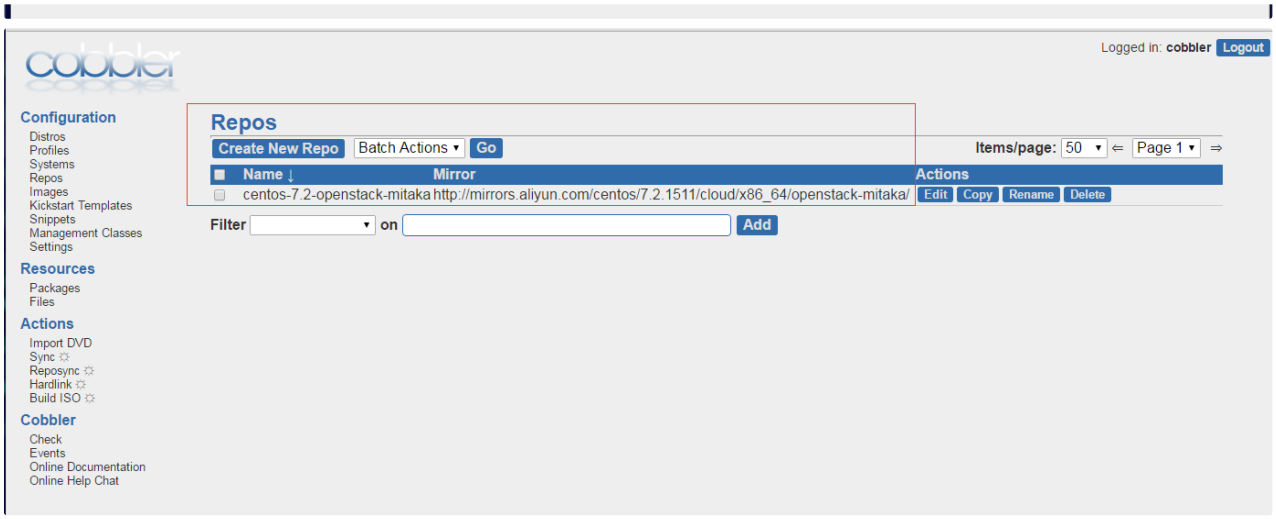
```
Changing password for user cobbler in realm Cobbler
New password: 123456
Re-type new password:123456
[root@linux-node1 ~]# cobbler sync
[root@linux-node1 ~]# systemctl restart httpd cobblerd
```

7.2 Cobbler Web 功能界面









8 Cobbler API 使用

cobbler_list.py 获取 Cobbler 服务器中的 distros/profiles/systems/images/repos 信息，代码如下：

```
#!/usr/bin/python
import xmlrpclib
server = xmlrpclib.Server("http://192.168.56.11/cobbler_api")
print server.get_distros()
print server.get_profiles()
print server.get_systems()
print server.get_images()
print server.get_repos()
```

cobbler_api.py 通过 Cobber API 添加 system

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import xmlrpclib

class CobblerAPI(object):
    def __init__(self,url,user,password):
        self.cobbler_user= user
        self.cobbler_pass = password
        self.cobbler_url = url

    def add_system(self,hostname,ip_add,mac_add,profile):
        '''
        Add Cobbler System Infomation
        '''
        ret = {
            "result": True,
            "comment": [],
        }
        #get token
        remote = xmlrpclib.Server(self.cobbler_url)
        token = remote.login(self.cobbler_user,self.cobbler_pass)

                #add system
        system_id = remote.new_system(token)
        remote.modify_system(system_id,"name",hostname,token)
        remote.modify_system(system_id,"hostname",hostname,token)
```

```
        remote.modify_system(system_id,'modify_interface', {
            "macaddress-eth0" : mac_add,
            "ipaddress-eth0" : ip_add,
            "dnsname-eth0" : hostname,
        }, token)
        remote.modify_system(system_id,"profile",profile,token)
        remote.save_system(system_id, token)
        try:
            remote.sync(token)
        except Exception as e:
            ret['result'] = False
            ret['comment'].append(str(e))
        return ret

def main():
    cobbler = CobblerAPI("http://192.168.56.11/cobbler_api","cobbler","cobbler")
    ret = cobbler.add_system(hostname='cobbler-api-test',ip_add='192.168.56.11',mac_add='00:50
    print ret

if __name__ == '__main__':
    main()
```

执行此 Python 脚本正确后输出为：

```
[root@linux-node1 ~]# python cobbler_api.py
{'comment': [], 'result': True}
[root@linux-node1 ~]# cobbler system list
   cobbler-api-test
   jaminzhang.github.io
```

Ref

COBBLER无人值守安装
Cobbler自动化部署最佳实践

操作系统安装流程及初始化规范 ↩ | ↪ HTTP 服务器性能测试工具-ab 学习