Chef Configuration Management

Table of Contents:

Introduction

Chef is a configuration management tool (like Puppet, Ansible, SaltStack, CFEngine). It is a tool which manages the configuration of centrally managed Linux servers, compute clusters, and desktops at SLAC. Examples of configuration items Chef manages include: sudo privileges, login access privileges, logging, software repositories, cronjobs, baseline security configuration. Chef is the configuration management tool for CentOS 7 and later, Red Hat Enterprise Linux (RHEL) 7 and later, and Ubuntu 16.04 and later. Operating systems earlier than those (RHEL 5 and 6, Solaris) are centrally managed using Taylor (a locally written configuration management tool).

Installing Chef

To get Chef installed on a SLAC owned Linux server, contact unix-admin@slac.stanford.edu . To get Chef installed on a SLAC owned Linux desktop, contact ithelp@slac.stanford.edu .

If you prefer to install Chef yourself, that is also possible. Run this command as root (or sudo):

```
curl -s yum.slac.stanford.edu/go-chef | sudo -i /bin/sh
```

You can also place the above command in your kickstart %post script if you are doing automated network installations. If you want to use a non-default chef configuration, you can create a json file named /root/kickstart-chef.json with your configuration options and it will be used by the go-chef script. You can email unix-admin@slac.stanford.edu for help with this.

Chef Architecture

- Chef Server - the Chef Server has the cookbook code which is run on each client, and information about each client manged by Chef which is updated after every chef-client run. Each host which runs chef contacts the Chef Server before the client run to get any updated code, and after each client run, to report success or failure, and provide a full report of the updated status based on the most recent run.
- Chef Client - there is a chef-client cronjob on each host managed by Chef which runs the chef-client code on a routine basis.
- Chef Automate Server - the Chef Automate server is an Operations Dashboard which provides details about everything managed by Chef, including status and error messages. The chef-client run on each host sends information to the Chef Automate Server, so the various dashboards provide OCIO with the current state of all centrally managed hosts.
- GitHub Organization - the central repository for Chef code is maintained in a SLAC OCIO GitHub organization: https://github.com/SLAC-CHEF/
- Jenkins Server - all Chef code goes through a Continuous Integration / Continuous Delivery Pipeline before being automatically delivered to the Chef Server. Automated tests using Test Kitchen are run on Vagrant VMs which are spun up by Jenkins. If all tests pass, and an OCIO system administrator presses the "Proceed" button, then updated code is delivered to the Chef Server.

Chef log files and directories

Chef-client logs are sent to syslog and a local log file. You can view the logs using these methods on each host which is managed:

```
sudo less -r /var/log/chef/client.log
sudo journalctl -t chef-client
sudo grep -w chef-client /var/log/everything
```

The logs are also sent to the central syslog server, and to Splunk. On the central syslog server, the log can be viewed here (this is for OCIO staff only):

```
ssh loghost
grep -w chef-client /u2/today/SYSLOG/daemon
```

Directories with Chef information, and some useful Chef commands:

/var/chef/cache/cookbooks/	This directory contains the cookbooks downloaded from the chef server.
/var/chef/cache/backup/	This directory contains backup files of any changes made by chef.
sudo -i /root/knife-node-show	This script will show configuration details for the current host.
/afs/slac/g/scs/systems/report/chef/system.info/	This directory contains information about each host managed by chef

Common configuration options

(this is a work in progress....)

Configuration Item	Chef attribute name for this item	Notes	How to enable it
sudo access	`authorization.sudo.users`	Request sudo access using this form: https://www.slac.stanford.edu/comp/unix/auth/superuser-req.shtml	
restrict local login access	`limit_login`		
do not update the default boot kernel	kernel_updatedefault		
home directory location	override_homedir	The default home directory location is /home (local to the host). The home directory listed in the LDAP directory service can be enabled instead. Be sure the directory services home directory location is actually available on this host first.	

Steps to provide an OCIO admin full access to Chef

These are the steps to grant an OCIO admin full access to the Chef Infrastructure.
Note: for minimal access (eg, to create GitHub Pull Requests and GitHub Issues), just steps 1a and 1b are needed.
Everyone in unix-admin should have steps 1a and 1b done, and optionally all steps for full access.

1. GitHub: Grant access to https://github.com/orgs/SLAC-CHEF organization
    a. Log into github as the admin for the SLAC-CHEF organization
        i. escrow -c systems display slac-chef-admin
    b. Invite the person using their github account on this page
        i. https://github.com/orgs/SLAC-CHEF/people
    c. The default permission level for members of SLAC-CHEF is read-only
        i. this is so we can invite any SLAC user to be a member so they can collaborate with us
        ii. add the unix-admin user to the scs-unix team on this page so they have write access to repos
            1. https://github.com/orgs/SLAC-CHEF/teams/scs-unix/members
    d. The unix-admin person should create an ssh key pair for github use if necessary
        i. https://github.com/settings/keys
2. Chef Server: create account on Chef Server
    a. Log into https://chef01.slac.stanford.edu
    b. ...
3. Automate Server: create account on Chef Automate 2 Server
    a. Log into https://chef-automate2.slac.stanford.edu
    b. ...
4. Jenkins Server: create account on Jenkins Server
    a. escrow -c systems display chef-jenkins
    b. Log into http://chef-build01.slac.stanford.edu:8080
    c. ...
5. Provide instructions on how to do the following:
    a. clone an existing cookbook, make a change, push changes to github, press the "approve" button on Jenkins
    b. create a new cookbook, with Jenkinsfile
        i. new github repo, with scs-unix team write access
        ii. new jenkins project
        iii. verify cookbook permissions on chef server (write access for jenkins user only)
        iv. add cookbook to a role, if appropriate
    c. how to test changes on cookbook before pushing to production
    d. how to bypass Jenkins pipeline for emergencies (eg, Jenkins server is down)