Chef VPC Toolkit 2.0

Use Chef to automate and configure VPC server groups in the cloud

Dan Prince

Rackspace

October 25th, 2010

Table of Contents

Introduction

The Chef VPC Toolkit is a set of Rake tasks that provide a framework to help automate the creation and configuration of “identical” VPC server groups in the cloud. The Chef VPC Toolkit provides the ability to create “toolkit projects” that can be used by team members and continuous integration systems to spin up groups of servers for development and testing. Configuration information is stored in JSON and YAML formats which can be easily parsed, edited, and version controlled.

Cloud Servers VPC is used to create secure groups of servers in the cloud. Each server group has a unique server namespace. The Chef configuration framework is used to configure servers via recipes and roles that are stored in Chef cookbook repositories.

Target Distribution Support

The Chef VPC Toolkit currently supports the following operating systems:

- CentOS 5.4 and 5.5 (Chef Server and Client)
- Fedora 12/13 (Chef Client Only)
- RHEL 5.4 and 5.5 (Chef Server and Client)
- Ubuntu 10.04 (lucid) (Chef Server and Client)
- Ubuntu 9.10/9.04 (karmic/jaunty) (Chef Client Only)

Supported Cloud Server Image IDs

The following image IDs are supported in Rackspace Cloud Servers when using the toolkit:

OS Name	Image ID
CentOS 5.5	51
CentOS 5.4	187811
Fedora 13	53
Fedora 12	17
RHEL 5.5	62
RHEL 5.4	14
Ubuntu 10.04 (lucid)	49
Ubuntu 9.10 (karmic)	14362
Ubuntu 9.04 (jaunty)	8

Prerequisites

Package Requirements

The Chef VPC Toolkit is currently deployed via Rubygems. The following packages are required:

- Ruby (1.8.6 or 1.8.7)

- RubyGems
- Rubygem-builder
- Rubygem-json
- Rubygem-rake (or rake)
- Rsync

Version of these packages are available in the most recent versions of Fedora and Ubuntu. Older version of RHEL (RHEL 5.4) may need to use EPEL and/or other third party sources to obtain the required RPMs.

SSH disable host key checking

SSH is used to connect to cloud servers and bootstrap the configuration process. By default SSH prompts you to accept new SSH keys as they are encountered. In order to streamline the configuration process it is recommended that you disable SSH host key checking. Adding the following option to your *$HOME/.ssh/config file* to enable the setting:

```
StrictHostKeyChecking no
```

SSH key pair

A SSH public/private key pair in your $HOME/.ssh directory is also required. If you haven't created a SSH key pair you can do so by running the *ssh-keygen* command. The Chef VPC Toolkit works best with SSH keys which do not require you to continually type in your password when spawning new SSH connections. Use of an SSH agent or not requiring an SSH (private key) password is recommended.

Installation

The Chef VPC Toolkit is deployed as a Gem package and can be installed via RubyGems.

RubyGem Installation

```
gem install chef-vpc-toolkit-2.0.0.gem
```

Configuration

Once the Chef VPC Toolkit has been configured you will need to create a configuration file in your home directory with settings for your environment. Create a file called .chef_vpc_toolkit.conf in your home directory with the following settings:

```
# Cloud Servers VPC settings
cloud_servers_vpc_url: https://<your server name>/
cloud_servers_vpc_username:
cloud_servers_vpc_password:

# knife text editor
#knife_editor: vim
```

You will need to add the username and password for Cloud Servers VPC to the configuration file. A custom knife editor may also be specified (Defaults to vim). Contact your Cloud Servers VPC administrator for this information.

Tutorial: message of the day

The tutorial describes how to create a Chef VPC Toolkit project and modify the configuration to customize the motd (the message of the day) using a Chef recipe.

Creating a new toolkit project

New projects can be created on the command line with the following command:

```
chef-vpc-toolkit -n <project name>
```

Once you have executed this command a new project directory should exist.

Project directory structure

toolkit projects use the following directory structure.

- config: configuration files for server groups, chef recipes and chef installer settings
- cookbook-repos: Chef cookbook repositories. By default only the 'local' Chef repo exists.
- share: files in this directory are automatically copied to /mnt/share in the dev/test cloud
- tmp: temporary XML file cache for running clouds

Creating a server group

After creating a new toolkit project you can immediately create a server group in the cloud with a single login server by running the following command:

```
rake create
```

This command will run the default task for toolkit projects which is 'create'. It will create a server group with a single login server, install the Chef server and sync share data. This operation will take several minutes to complete. You can monitor the process via the command line and/or use a web browser to connect to the Cloud Servers VPC web UI to monitor the creation of servers within the cloud.

Accessing the server group via ssh

After creating a new toolkit project you can ssh into the VPN gateway server *once it has booted* by using the following task inside of your toolkit project directory:

```
rake ssh
```

Running this command will give you a root ssh terminal on the login server.

At this point you can look around a bit if you wish.

When you are finished type 'exit' to exit the login ssh shell and go back to the toolkit project directory.

Deleting a server group

When you are finished using a server group you may want to delete it. Execute the following command inside the toolkit project directory to delete the most recent server group instance within the group:

```
rake group:delete
```

To delete a specific group ID you can use the following commands:

```
rake group:list
rake group:delete GROUP_ID=1
```

Inline Chef repositories and the motd recipe

The Chef VPC Toolkit uses Chef for configuration management. By default each toolkit project contains an inline chef repository which exists in the 'cookbook-repos/local' directory. Inline cookbooks are an optional part of toolkit projects however they are useful when developing new recipes and may also be used if a cookbook is tightly coupled to a specific toolkit project. *Inline cookbooks are automatically deployed to the login Chef server* and available for use within the server group when the 'create' task is used. For now let's have a look at the example motd recipe which is created in the 'local' Chef repository. Run the following command to view the default motd recipe:

```
cat cookbook-repos/local/cookbooks/motd/recipes/default.rb
<snip>
template "/etc/motd" do
  source "motd.erb"
  mode 0644
end
```

Our motd recipe instructs Chef to write out the /etc/motd file on a server by using a Chef template called motd.erb. The motd.erb file contains the following:

```
cat cookbook-repos/local/cookbooks/motd/templates/default/motd.erb
<%= @node[:motd] %>
```

The motd.erb template contains a single line which instructs the Chef template engine to write out the value of the 'motd' attribute to a file. Our motd recipe uses a template which is one of the many types of Chef resources to dynamically configure the /etc/motd file on a server.

Adding motd to nodes.json

In order to add the motd recipe to our login server we must edit the login section of the nodes.json file within the toolkit project. Make the following changes to the config/nodes.json file (changes highlighted in yellow):

```
{
    "login": {
        "run_list": [
            "recipe[motd]"
        ],
        "motd": "Hello World!"
    }
}
```

Creating a server group with the Hello World motd

After editing the nodes.json file so the login server uses the motd recipe you can create a new server group with the following command:

```
rake create
```

Once the server group is finished being created SSH into the login machine by running a 'rake ssh' command. You should be greeted by the following message:

```
$ rake ssh
Hello World!
```

Toolkit Project Tasks

Running a 'rake –tasks' command inside the project directory will display the available project tasks. Example:

```
[dan.prince@my_project]$ rake --tasks
rake chef:databags       # Create/Update databags on the Chef server.
rake chef:install        # Install and configure Chef on the server group
rake chef:sync_repos     # Sync the local cookbook repos directory to the C...
rake chef:tail_logs      # Tail the Chef client logs
rake chef:validate_json  # Validate the Chef JSON config file.
rake create              # Create a server group, install chef, sync share ...
rake group:create        # Create a new group of cloud servers
rake group:delete        # Delete a cloud server group
rake group:force_clean   # Force clean the cached server group files
rake group:list          # List existing cloud server groups
rake group:poll          # Poll/loop until a server group is online
rake group:show          # Print information for a cloud server group
rake rechef              # Rebuild and Re-Chef the specified server.
rake server:rebuild      # Rebuild a server in a server group.
rake share:sync          # Sync the share data.
rake ssh                 # SSH into the most recently created VPN gateway s...
rake usage               # Print help and usage information
```

Server group tasks

The server_group.json configuration file controls the names, types, sizes, and number of servers each cloud contains. By default only a single login server exists in the server_group.json configuration file.

To spin up a cloud with only the single login server we can run the following command inside of a newly created toolkit project directory:

```
rake group:create
```

After running this command the a request to Cloud Servers VPC will be submitted to create the server group. You can monitor the progress of the creation process by running the following task:

```
rake group:poll
```

The 'group:poll' task will run and display updates about the dev/test cloud until it comes online.

It is important to note that by default Chef VPC Toolkit tasks operate on the most recently created dev/test cloud. If you have created multiple clouds that are running concurrently you can specify the GROUP_ID environment variable to control which cloud is used for each command. To obtain a list of server group IDs you can run the following task:

```
rake group:list
```

Once you have the GROUP_ID you can use it to show status information (and other things) using the following syntax:

```
rake group:show GROUP_ID=101
```

Chef tasks

To install Chef on all machines in the most recent server group run the following command:

```
rake chef:install
```

This command will take care of bootstrapping the Chef installation on the Chef Server and installing Chef clients on each machine. The chef installation takes several minutes.

'Create' task

Rather than running individual 'group:create' and 'chef:install' commands each time you create a dev/test cloud you can use the 'create' task to automatically run the necessary tasks to create a server group and install the Chef server. This task can be executed with the following command:

```
rake create
```

After running this task the following operations will occur:

- A server group with a single login server will be created.
- The server group is polled until the all of the servers within the group come online.
- A Chef Server is installed on the configured machine.
- Chef clients are installed on all other machines within the server group.
- Share data is synchronized.

Some of these operations are optional and will execute only if data exists or they have been configured to execute. Share data is only copied if it exists in the share folder.

Chef Cookbooks

The Chef VPC Toolkit installs a fully functioning Chef infrastructure. A single Chef server is created in each server group instance. Chef clients are installed on all other nodes that are specified in the nodes.json configuration file.

Chef servers are configured to host cookbooks and roles as specified in the chef_installer.conf file. Recipes are stored in cookbooks that reside in cookbook repositories. In order to use recipes within a toolkit projects you will need to deploy them inside of cookbook repositories. There are several ways to manage cookbook repositories and use them within toolkit projects:

- <u>Shared cookbook repositories</u>: Shared repositories can be maintained in a version control system (Git, Subversion, etc.) and uploaded into Cloud Files where toolkit projects can access them directly. Shared cookbooks are useful for things that may apply to multiple toolkit projects or cookbooks that are also used to configure production machines. Shared cookbooks URL's are configured via the config/chef_installer.conf file in each toolkit project. Absolute URL's are downloaded directly and are useful for public cookbooks that reside on the web or in a CDN (Content Distribute Network). Relative URL's can be used to securely download “private” files via the Rackspace Cloud Files API.

- Inline cookbook repositories: Inline repositories are stored inside of toolkit projects. Inline cookbook repositories are uploaded to the Chef server via SSH each time you create a new cloud server group instance. Inline cookbooks can be useful for recipes that apply only to a specific set of tests. Using inline cookbooks is also a handy way to test out new Chef recipes in an isolated cloud.

Each toolkit project contains a nodes.json configuration file which is used to configure nodes within the server group. Recipes and roles can be assigned to nodes by adding them to the 'run_list' for each node within nodes.json. Chef node attributes are also configured in nodes.json.