Knife Quick Reference

Chef

About Knife

Knife is a command-line tool that provides an interface between a local Chef repository and the Chef server. Knife helps users manage cookbooks and recipes, nodes, roles, stores of JSON data, environments, cloud resources (using Knife plug-ins), and more. This quick reference covers the most common Knife actions. The basic syntax for Knife is:

```
$ knife subcommand [ARGUMENT] (options)
```

COMMON OPTIONS AVAILABLE TO ALL SUB-COMMANDS

--config --- The configuration file to use.
--editor --- The editor that Knife will use for interactive commands.
--environment --- The name of an environment on the Chef server.
--format --- The output format: *summary*, *text*, *json*, *yaml*, or *pp*.
--help --- Show the man page for a Knife sub-command.
--key --- The private key used to sign requests made to the Chef Server API.

bootstrap

Install Chef on a target system so that it can be run as a chef-client.

OPTIONS

--bootstrap-proxy --- The proxy server that is the target of the bootstrap.
--bootstrap-version --- The version of Chef to install.
--distro --- The distribution used for the boostrap. Default: *chef-full*.
--hint --- A hint set on the target of the bootstrap. This hint is used by Ohai.
--identity-file --- The SSH identity file used during authentication.
--json-attributes --- A JSON string that is added to the first run of a chef-client.
--node-name --- The name of a node on the Chef server.
--prerelease --- Use to install pre-release Chef gems.
--run-list --- A comma-separated list of roles and recipes.
--ssh-gateway, **--ssh-port** --- The SSH tunnel or gateway and the port.
--ssh-user, **--ssh-password** --- The SSH user name and password.
--sudo --- Use to run a bootstrap command using sudo.
--template-file --- The path to a template file.

EXAMPLES

```
$ knife bootstrap 192.168.1.1 --ssh-user user --ssh-password pwd --sudo
```

client

Manage an API client list and its associated RSA key-pairs.

bulk delete

Delete API clients using patterns and regular expressions.

create, delete, edit, list, show

Create and delete API clients (along with the RSA key-pair). View API clients in a list. View or edit information about a specific API client.

OPTIONS

--admin --- Create an admin client (create command only).

reregister

Regenerate the RSA key-pair for an API client.

OPTIONS

--file-name --- The file in which a private key is saved.

EXAMPLES

```
$ knife client create example-client --file-name "/etc/chef/client.pem"
```

cookbook

A cookbook is the fundamental unit of configuration and policy distribution in Chef. Each cookbook defines a configuration scenario and contains all of the components that are required to support that scenario, including recipes, providers, attributes, definitions, libraries, and templates.

bulk delete

Delete cookbooks using patterns and regular expressions.

create

Create new cookbook directories in the local repository.

delete

Delete cookbooks from the local repository.

OPTIONS

--all --- Delete a cookbook, along with all of its versions.
--purge --- Delete a cookbook, along with all of its versions, including copies of the cookbook on the Chef server.

download

Download a cookbook from the Chef server.

OPTIONS

--force --- Overwrite the directory in the local repository when downloading.
--latest --- Get only the latest version of a cookbook.

list

View a list of cookbooks.

OPTIONS

--all --- View a list of cookbooks, including all cookbook versions.
--with-uri --- Show the corresponding URI for all cookbooks and versions.

metadata, metadata from file

Regenerate the metadata for a cookbook from the repository or from a file.

show

Show the details for a cookbook or cookbook version.

OPTIONS

PART --- The part of the cookbook to show: *attributes*, *definitions*, *libraries*, *files*, *providers*, *recipes*, *resources*, and/or *templates*.
--platform, **--platform-version** --- The platform (or version) for which a cookbook is designed.
--with-uri --- Show the corresponding URI for all cookbooks and versions.

test

Test a cookbook for syntax errors.

OPTIONS

--all --- Test all cookbooks.

upload

Upload a cookbook to the Chef server.

OPTIONS

--all --- Upload all cookbooks.
--include-dependencies --- If another cookbook is required to use a cookbook, upload that cookbook too.
--force --- Upload the cookbook, even when it is frozen.
--freeze --- Use to prevent a cookbook from being modified and to require any future changes to be saved as a cookbook version, or to be uploaded using the force option.

EXAMPLES

```
$ knife cookbook delete smartmon version 0.8
```

```
$ knife cookbook download smartmon
```

```
$ knife cookbook metadata from file /path/to/file
```

```
$ knife cookbook show redis 0.1.6 recipes attributes
```

```
$ knife cookbook test getting-started
```

```
$ knife cookbook upload redis --freeze
```

configure

Create knife.rb and client.rb files for distribution to nodes.

OPTIONS

--client --- Creates the client.rb file (instead of the knife.rb file).

EXAMPLES

Create a knife.rb file:

```
$ knife configure 'directory_name/'
```

Create a client.rb file:

```
$ knife configure client 'directory_name/'
```

cookbook site

The Cookbooks Site API is used to provide access to cookbooks that are hosted at *cookbooks.opscode.com*. Knife uses this API to download cookbooks from that site, share those cookbooks with others, search for specific cookbooks, and also show the details for any cookbook.

download

Download a cookbook from *cookbooks.opscode.com*.

install

Install a downloaded cookbook to a git repository.

OPTIONS

--branch --- The name of the branch to be used.
--skip-dependencies --- Use to install only this cookbook (and none of its dependencies) into a git repository.

list

View a list of cookbooks currently available at *cookbooks.opscode.com*.

OPTIONS

--with-uri --- Show the corresponding URI for all cookbooks and versions.

search

Search for cookbooks at *cookbooks.opscode.com*.

share

Add a cookbook to *cookbooks.opscode.com*. Use unshare to stop sharing.

show

View information about one (or more) cookbooks at *cookbooks.opscode.com*.

EXAMPLES

```
$ knife cookbook site install getting-started
```

```
$ knife cookbook site search apache*
```

```
$ knife cookbook site share "apache2" "Web Server"
```

```
$ knife cookbook site show haproxy
```

Knife Quick Reference

data bag

Manage global variables (stored as JSON) on the Chef server. Nodes and recipes can access this data whenever they need to. A data bag item can be encrypted.

OPTIONS

--secret --- The encryption key used for a data bag item.
--secret-file --- The path to the file that contains the encryption key.

create, delete, edit, list, show

Create and delete data bags or data bag items. View a list of data bags. View or edit information about a specific data bag or data bag item.

from file

Use a template to create a data bag item.

EXAMPLES

```
$ knife data bag from file DATA_BAG_NAME "/path/to/file.json"
```

diff

Check for differences between files and directories on the Chef server and in the Chef repository.

EXAMPLES

```
$ knife diff
```

```
$ knife diff roles/base.json roles/webserver.json
```

environment

An environment is a way to map an organization's real-life workflow to what can be configured and managed when using Chef server.

bulk delete

Delete environments using patterns and regular expressions.

create, delete, edit, list, show

Create and delete environments. View environments in a list. View or edit information about a specific environment.

from file

Use a template to create an environment.

exec

Execute Ruby scripts on behalf of a chef-client. This is useful when a script only needs to be run infrequently or when a command does not require the full Knife syntax library.

OPTIONS

--exec --- The string of code that will be executed.
--script-path --- The path to the Ruby script.

EXAMPLES

```
$ knife exec --exec 'puts api.get("search").keys'
```

Use a script file to check status. A file named "status.rb" that contains:

```
printf "%-5s %-12s %-8s %s\n", "Check In", "Name", "Ruby", "Recipes"
nodes.all do |n|
   checkin = Time.at(n['ohai_time']).strftime("%F %R")
   rubyver = n['languages']['ruby']['version']
   recipes = n.run_list.expand.recipes.join(", ")
   printf "%-20s %-12s %-8s %s\n", checkin, n.name, rubyver, recipes
end
```

node

A node is any server or virtual server that is configured to be maintained by a chef-client.

bulk delete

Delete nodes using patterns and regular expressions.

create, delete, edit, list, show

Create and delete nodes. View nodes in a list. View or edit information about a specific node.

from file

Use a template to create a node.

run_list add, run_list remove

Add and remove roles and recipes from a node's run-list.

recipe list

View all recipes on the Chef server. Use a regular expression to limit the results to those that match a specific pattern.

EXAMPLES

```
$ knife recipe list 'couchdb::*'
```

role

A role is a way to define certain patterns and processes that exist across nodes as if they belong to a single job function.

bulk delete

Delete roles using patterns and regular expressions.

create, delete, edit, list, show

Create and delete roles. View roles in a list. View or edit information about a specific role.

from file

Use a template to create a role.

search

Search for anything that is indexed by the Chef server, including data bags, data bag items, environments, nodes, and roles. Use search patterns like exact, fuzzy, range, and wildcard.

OPTIONS

--attribute --- Search for a specific attribute.
--id-only --- Search for a specific identifier.
--run-list --- Show only run-list data.

EXAMPLES

```
$ knife search node 'platform:ubuntu'
```

Use the Chef search syntax to define your own search queries.

```
$ knife search node 'chef_environment:production AND platform:centos'
```

```
$ knife search node 'ec2:*' --attribute ec2.instance_type
```

ssh

Invoke SSH commands based on the results of a search query.

EXAMPLES

```
$ knife ssh "role:webserver" "sudo chef-client"
```

```
$ knife ssh name:* "sudo aptitude upgrade -y"
```

```
$ knife ssh "role:web" "uptime" -a ec2.hostname
```

status

Get the current status for all nodes on the Chef server.

OPTIONS

--run-list --- Add run-lists to the status report.

EXAMPLES

```
$ knife status --run-list
```

tag

Apply custom descriptions to nodes and custom groups of data.

create, delete, list

Create and delete tags. View all tags on the Chef server as a list.

user

Manage users and their associated RSA key-pairs.

create, delete, edit, list, show

Create and delete users. View a list of users. View or edit information about a specific user.

reregister

Regenerate the RSA key-pair for a user.

OPTIONS

--file --- The file in which a private key is saved.

EXAMPLES

```
$ knife user create example-user --file-name "/etc/chef/user.pem"
```

VERBS: delete, download, list, show, upload

Use a common set of verbs to interact with cookbooks, environments, data bags, nodes, and roles. Work with more than one type at a time.

- Work with objects in the Chef repository and the Chef server in the same way.
- Use the repository on the Chef server as if it were a file system.
- Knife knows what directory it is in. For example, roles do not need to be specified in the command when Knife is run from the roles/ directory.

EXAMPLES

```
$ knife download environments/production
```

```
$ knife list roles/ environments/ nodes/
```

```
$ knife show cookbooks/
```

```
$ knife show roles/base.json
```

For more information ...

Visit these links for more information about Knife---including plug-ins, more examples, and more options---the Chef community, cookbooks, and more!

LINKS

- docs.opscode.com
- http://getchef.com/community/
- cookbooks.opscode.com
- www.opscode.com

CHEF
DOCUMENTS