Chef Cookbook Repositories

Overview

Artifactory supports Chef Cookbook repositories on top of its existing support for advanced artifact management.

Artifactory support for Chef Cookbook provides:

1. The ability to provision Cookbook packages from Artifactory to the Knife and Berkshelf command line tool from all repository types.
2. Calculation of metadata for Cookbook packages hosted in Artifactory local repositories.
3. Access to remote Cookbook repositories (in particular the Chef supermarket public repository) through remote repositories which provide proxy and caching functionality.
4. The ability to access multiple Cookbook repositories from a single URL by aggregating them under a Virtual Repository. This overcomes the limitation of the Knife client which can only access a single repository at a time.
5. Compatibility with the Knife command line tool to list, show and install Cookbooks. Compatibility with the Berkshelf command line to resolve Cookbook dependencies.
6. The ability to assign access privileges according to projects or development teams.

Chef Repository

Chef uses the concept of a Chef repository, to represent storing their own data objects on a workstation. This is different from the use of "repository" in Artifactory.

Chef provides an official "supermarket" for cookbook packages, so Chef repositories in Artifactory are actually Chef supermarkets in Chef terminology. This page refers to Chef Cookbook repositories and Chef supermarkets interchangeably.

Page contents

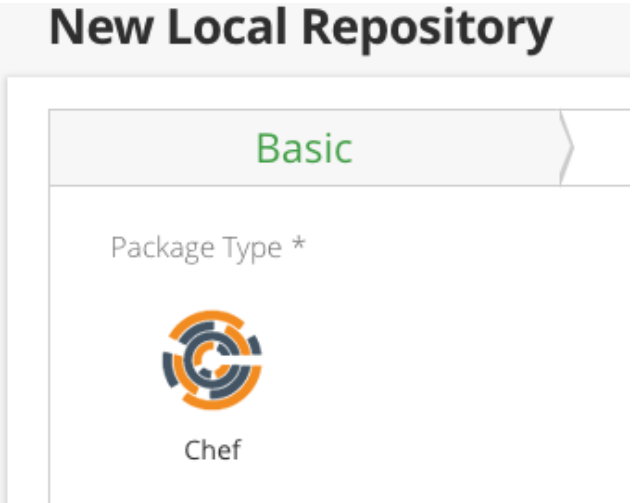
Integration Benefits

JFrog Artifactory and Chef Cookbook Repositories

Configuration

Local Chef Supermarket

To enable calculation of Chef package metadata in local repositories so they are, in effect, Chef supermarkets, set the **Package Type** to **Chef** when you create the repository:



Repository Layout

Artifactory allows you to define any layout for your Chef Cookbook repositories. In order to upload packages according to your custom layout, you need to package your Chef Cookbook files with Knife or Berkshelf and archive the files as **tar.gz**. Then you can upload to any path within your local Chef supermarket, see publishing Cookbooks.

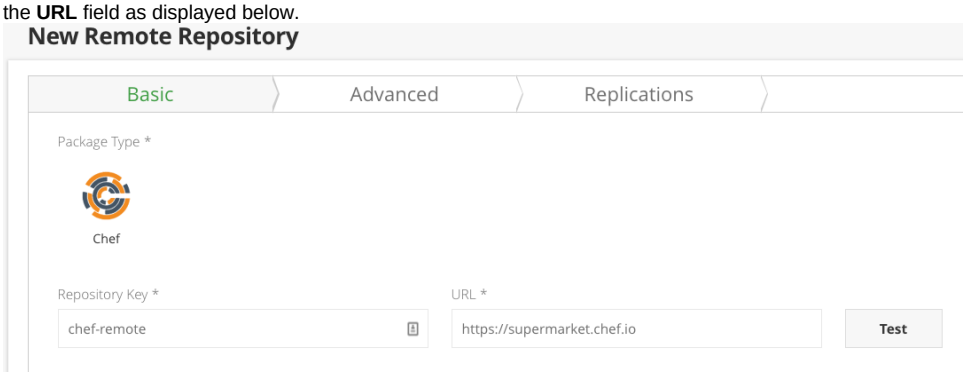
Remote Chef Supermarket

A Remote Repository defined in Artifactory serves as a caching proxy for a supermarket managed at a remote URL such as `https://supermarket.chef.io`.

Artifacts (such as tgz files) requested from a remote repository are cached on demand. You can remove downloaded artifacts from the remote repository cache, however, you can not manually deploy artifacts to a remote Chef repository.

To define a remote repository to proxy a remote Chef Cookbook, repository follow the steps below:

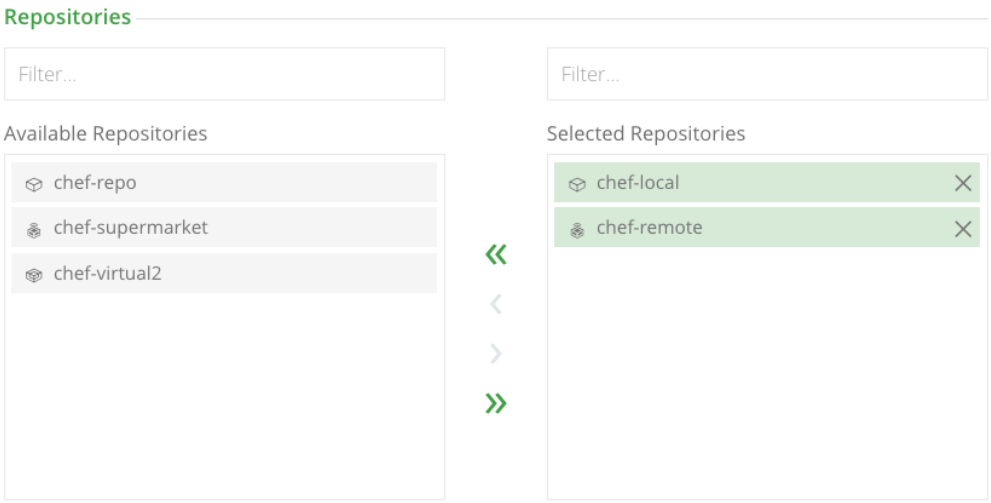
1. In the **Admin** module, under **Repositories | Remote,** click "New".
2. In the New Repository dialog, set the **Package Type** to **Chef**, set the **Repository Key** value, and specify the URL to the remote repository in the **URL** field as displayed below.

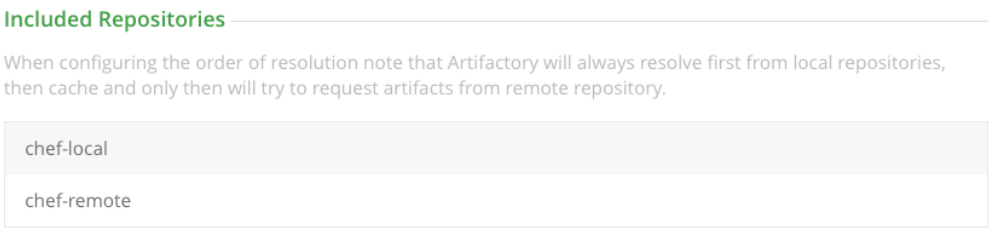


3. Click "Save & FInish".

Virtual Chef Supermarket

A Virtual Repository defined in Artifactory aggregates packages from both local and remote repositories.
This allows you to access both locally hosted Chef Cookbook packages and remote proxied Chef Cookbook repositories from a single URL defined for the virtual repository.
To define a virtual Chef Cookbook repository, create a virtual repository, set the **Package Type** to be **Chef,** and select the underlying local and remote Chef repositories to include in the **Basic** settings tab.





Using the Knife Command Line

Chef repositories must be prefixed with api/chef in the path

When accessing a Chef supermarket through Artifactory, the repository URL must be prefixed with **api/chef** in the path. This applies to all Knife commands.

For example, if you are using Artifactory standalone or as a local service, you would access your Chef supermarket using the following URL:

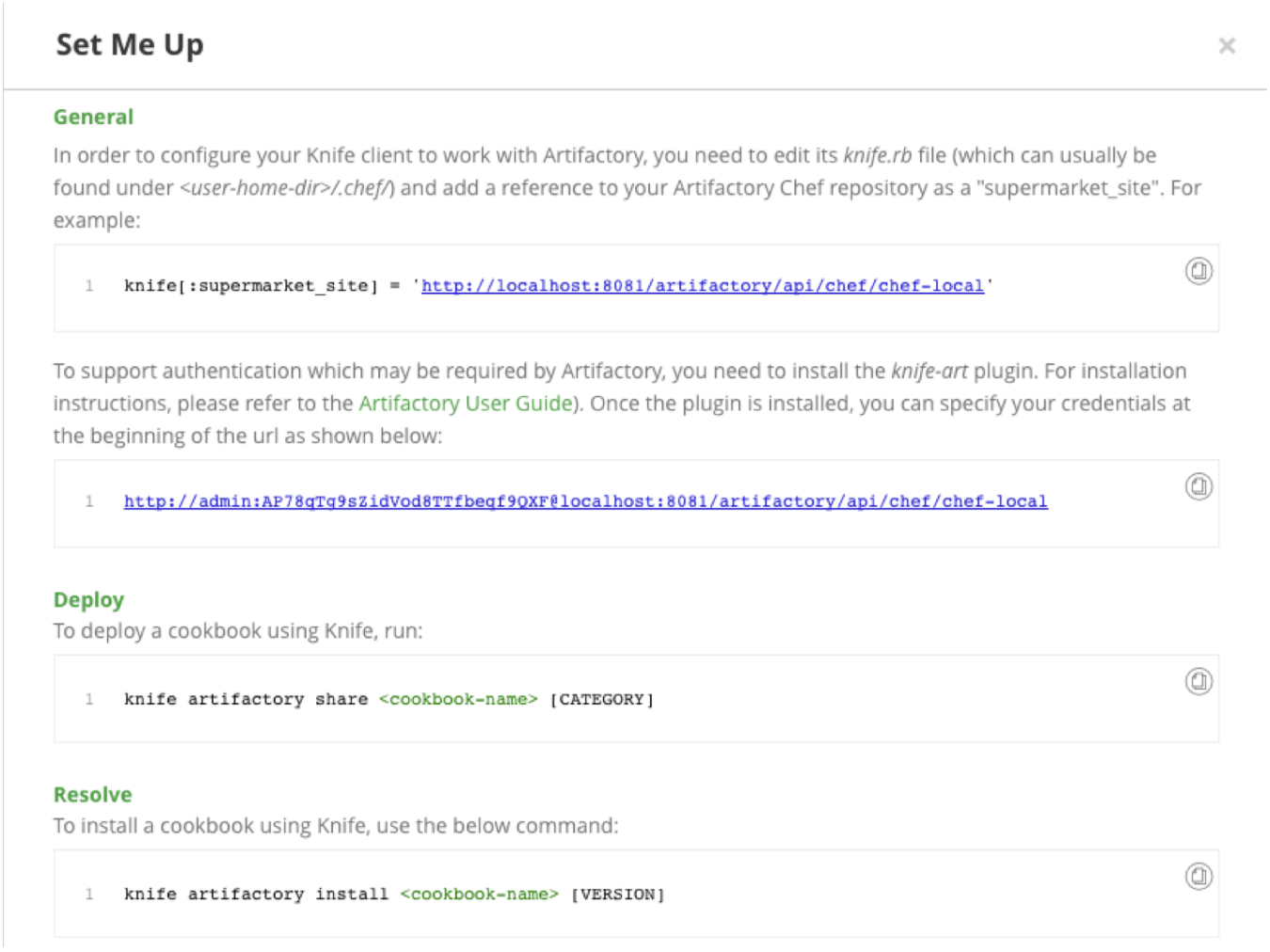
*http://localhost:8081/artifactory/**api/chef**/<repository key>*

Or, if you are using Artifactory SaaS the URL would be:

*https://<server name>.jfrog.io/<server name>/**api/chef**/<repository key>*

To use the Knife command line you need to make sure it's installed. It's part of ChefDK, that can be installed in various ways.

Once you have created your Chef supermarket, you can select it in the Tree Browser and click **Set Me Up** to get code snippets you can use to change your Chef supermarket URL, and deploy and resolve packages using the knife command line tool.



Set the default Chef supermarket with a URL pointing to a Chef supermarket in Artifactory by editing your `~/.chef/knife.rb` configuration file (the example below uses a repository with the key *`chef-virtual`*). You authenticate the request using your username and password or with your Artifactory API key :

Setting the default Chef supermarket for Knife with credentials

```
knife[:supermarket_site] = 'http://admin:password@localhost:8081/artifactory/api/chef/chef-virtual'
or
knife[:supermarket_site] = 'http://admin:<APIKEY>@localhost:8081/artifactory/api/chef/chef-virtual'
```

knife.rb file location

The knife.rb file doesn't exist by default. It can be created with the `knife configure` command. Refer to the knife documentation for possible knife.rb locations.

The location of this file can be overidden with the `--config` parameter when running a knife command

Working with Artifactory without Anonymous Access

By default, Artifactory allows Anonymous Access for Chef repositories. This is defined under **Security | General Configuration**. For details please refer to Allow Anonymous Access.
If you want to be able to trace how users interact with your repositories you need to uncheck the Allow Anonymous Access setting. This means that users will be required to enter their username and password.

The Knife command line tool does not support basic authentication (it only supports authentication with RSA keys).
To enable basic authentication, you will need to install the **knife-art.gem** plugin.

Install knife-art plugin

```
chef gem install knife-art
```

If properly installed you should see the following specific Artifactory commands:

Knife Artifactory plugin commands

```
** ARTIFACTORY COMMANDS **
knife artifactory download COOKBOOK [VERSION] (options)
knife artifactory install COOKBOOK [VERSION] (options)
knife artifactory list (options)
knife artifactory search QUERY (options)
knife artifactory share COOKBOOK [CATEGORY] (options)
knife artifactory show COOKBOOK [VERSION] (options)
knife artifactory unshare COOKBOOK VERSION
```

These commands are a wrapper around the standard Knife supermarket commands, that enable basic authentication. To add these credentials, pre-pend them to the URL of the Chef supermarket configured in your **knife.rb** file:

Setting the default Chef supermarket for Knife with credentials

```
knife[:supermarket_site] = 'http://admin:password@localhost:8081/artifactory/api/chef/chef-virtual'
```

Use an encrypted password

Use an encrypted password instead of clear-text; see Centrally Secure Passwords.

Publishing Cookbooks

You can use the UI or a simple REST API call to upload the **tgz/tar.gz** containing the Cookbook to a Chef repository.

Artifactory will automatically extract the relevant information from the `metadata.json` to later serve the index and respond properly to client calls. This `metadata.json` file is mandatory. If it does not exist in your cookbook, you can use a Knife command to generate it and then publish it to Artifactory. For example:

Publishing a new Cookbook

```
$ chef generate cookbook myapp
$ knife artifactory share myapp tool
```

Using the Berkshelf Command Line

From version 6.1.0, Berkshelf supports authenticated access to Artifactory using an API Key. Previous versions of Berkshelf only supported Anonymous access to Artifactory.

Berkshelf is a dependency manager for Chef Cookbooks, and is a part of the **ChefDK**.

To resolve dependencies from a Chef supermarket in Artifactory, first configure your Artifactory API Key using **config.rb**:

Setting the Artifactory API key for use by Berkshelf

```
artifactory_api_key "<APIKEY>"
```

Then set the default supermarket in your Berksfile's Cookbook:

Setting the default Chef supermarket for Berkshelf

```
source artifactory: 'http://localhost:8081/artifactory/api/chef/chef-virtual'
```

Then you can execute the `berks` command to download the required dependencies from Artifactory:

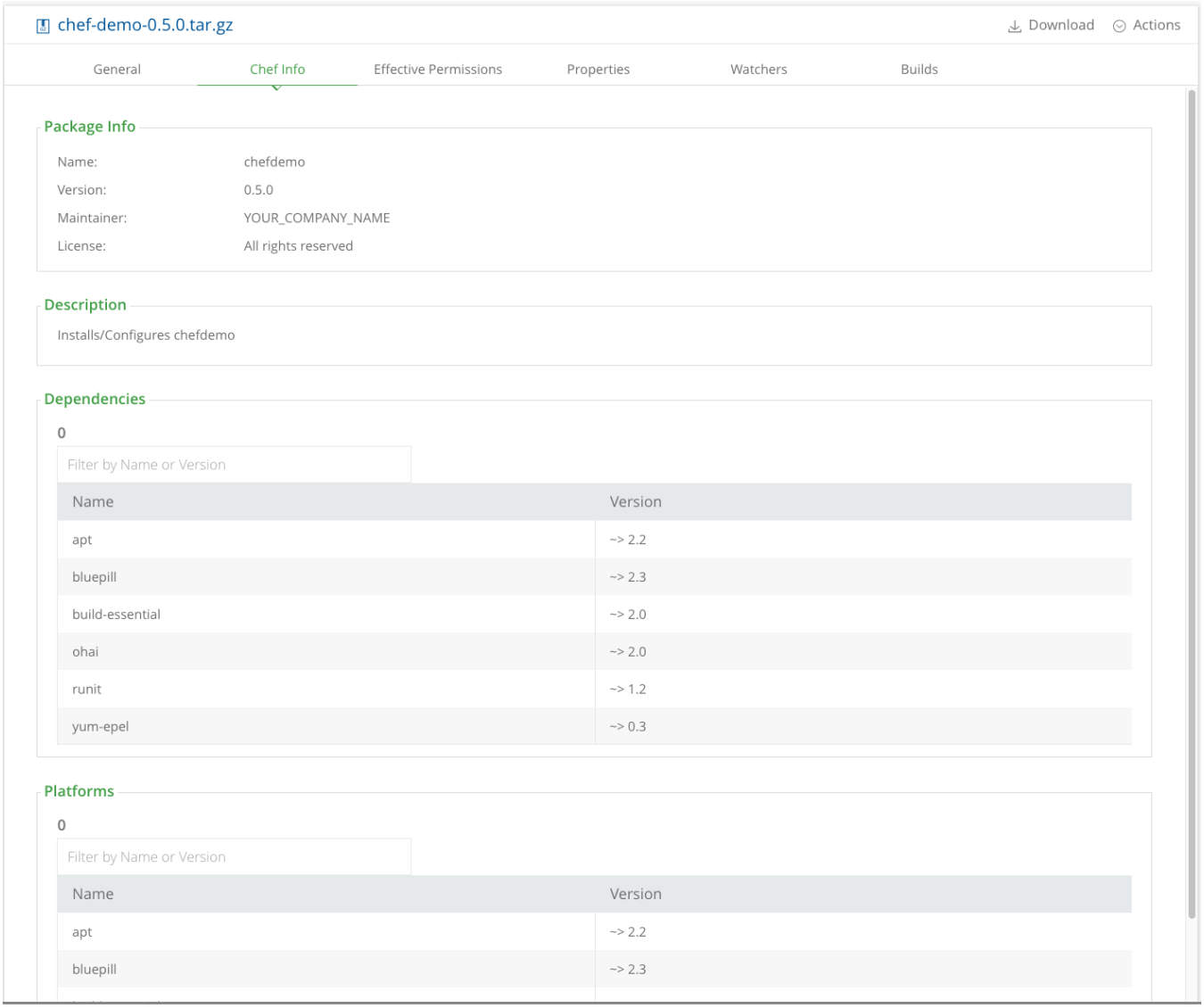
Resolving dependencies with Berkshelf

```
vagrant@default-ubuntu-1404:~/chef-zero/mycookbook$ berks
Resolving cookbook dependencies...
Fetching 'mycookbook' from source at .
Fetching cookbook index from http://localhost:8081/artifactory/api/chef/chef-virtual...
Installing apt (5.0.0) from http://localhost:8081/artifactory/api/chef/chef-virtual ([opscode]
http://localhost:8081/artifactory/api/chef/chef-virtual/api/v1)
Installing chef-apt-docker (1.0.0) from http://localhost:8081/artifactory/api/chef/chef-virtual ([opscode]
http://localhost:8081/artifactory/api/chef/chef-virtual/api/v1)
Installing chef-yum-docker (1.0.1) from http://localhost:8081/artifactory/api/chef/chef-virtual ([opscode]
http://localhost:8081/artifactory/api/chef/chef-virtual/api/v1)
Installing compat_resource (12.16.2) from http://localhost:8081/artifactory/api/chef/chef-virtual ([opscode]
http://localhost:8081/artifactory/api/chef/chef-virtual/api/v1)
Using mycookbook (0.1.0) from source at .
Installing yum (4.1.0) from http://localhost:8081/artifactory/api/chef/chef-virtual ([opscode]
http://localhost:8081/artifactory/api/chef/chef-virtual/api/v1)
```

Viewing Individual Chef Cookbook Information

Artifactory lets you view selected metadata of a Chef Cookbook directly from the UI.

In the **Artifacts** tab, select **Tree Browser** and drill down to select the *tgz/tar.gz* file you want to inspect. The metadata is displayed in the **Chef Info** tab.



Searching Chef Cookbooks

Artifactory supports a variety of ways to search for artifacts.
Artifactory also supports `knife search [search terms ...]`:

- For local repositories, it will look for the given terms in the name, description and maintainer fields.
- For remote repositories, the search will be done on the local cache, then the search query will be forwarded to the external repository and the results merged before returned to the client.
- For virtual repositories, the search will be done on local repositories and then on remote repositories, the results merged before returning to the client.

Properties

Artifactory annotates each deployed or cached Chef Cookbook package with at least 3 properties: *chef.name*, *chef.version* and *chef.maintainer*. If available, it will also add *chef.dependencies*, *chef.platforms* multi-valued properties.

You can use Property Search to search for Chef Cookbook according to their name, version, maintainer, dependencies or platforms requirements.