
Manila plugin for Fuel User Guide

Release 1.0-1.0.0-1

Mirantis Inc.

October 20, 2016

CONTENTS

1	Overview	1
1.1	Overview of the Manila plugin for Fuel	1
1.2	Software prerequisites	1
1.3	Limitations	2
1.4	Licenses	2
2	Install and configure Manila plugin for Fuel	3
2.1	Install Manila plugin for Fuel	3
2.2	Uninstall Manila plugin for Fuel	4
2.3	Configure Manila plugin for Fuel	5
3	Use Manila plugin for Fuel	8
3.1	Using File Share as a Service possibility with Manila plugin for Fuel	8
3.2	Troubleshooting	13
3.3	Links	13

OVERVIEW

Overview of the Manila plugin for Fuel

The purpose of this document is to describe how to install, configure and use the Manila plugin 1.0.0 for Fuel 9.1

The Manila is the OpenStack project that provides “File Sharing as a Service”. Main goal of the project is providing coordinated access to shared or distributed file systems to OpenStack Compute instances. But as a many other OpenStack services it can be used independently according to modular design established by OpenStack.

The Manila based on that principles:

- Component based architecture: Quickly add new behaviors
- Highly available: Scale to very serious workloads
- Fault-Tolerant: Isolated processes avoid cascading failures
- Recoverable: Failures should be easy to diagnose, debug, and rectify
- Open Standards: Be a reference implementation for a community-driven api
- API Compatibility: Manila strives to provide API-compatible with popular systems like Amazon EC2

This plugin brings features of Manila into Mirantis OpenStack.

Software prerequisites

To use the Manila plugin for Fuel 9.1, verify that your environment meets the following prerequisites:

Prerequisites	Version/Comment
Fuel	9.1
manila-service-image	last
NetApp® ONTAP®	8 or later

The manila-service image is the service image for generic driver. It should be build from <https://github.com/openstack/manila-image-elements>.

Limitations

The Manila plugin for Fuel 9.1 has some known issues/limitations of usage:

- Manila CLI response with warnings if specific configuration <https://bugs.launchpad.net/fuel-plugin-manila/+bug/1633018>
- Manila services uses publicURL instead of internalURL <https://bugs.launchpad.net/fuel-plugin-manila/+bug/1633456>
- Manila share created from snapshot has error status (Generic) <https://bugs.launchpad.net/fuel-plugin-manila/+bug/1634818>

Licenses

Component	License
Manila plugin for Fuel	Apache 2.0

INSTALL AND CONFIGURE MANILA PLUGIN FOR FUEL

Install Manila plugin for Fuel

Before you install Manila plugin for Fuel 9.1, verify that your environment meets the requirements described in *Software prerequisites*. You must have the Fuel Master node installed and configured before you can install the plugin. This plugin is hotpluggable, so you can install the Manila plugin for Fuel after you deploy an OpenStack environment.

To install Manila plugin for Fuel:

1. Download Manila plugin for Fuel from the [Fuel Plugins Catalog](#).
2. Copy the plugin .rpm package to the Fuel Master node:

Example:

```
# scp fuel-plugin-manila-1.0-1.0.0-1.noarch.rpm root@fuel-master:/tmp
```

3. Copy the manila-service-vm iso to the Fuel Master node:

Example:

```
# scp manila-service-image.qcow2 root@fuel-master:/tmp
```

4. Log into Fuel Master node CLI as root.
5. Set path to manila service image into environment variable MANILA_IMAGE

Example:

```
# export MANILA_IMAGE=/tmp/manila-service-image.qcow2
```

6. Install the plugin by typing:

```
# fuel plugins --install fuel-plugin-manila-1.0-1.0.0-1.noarch.rpm
```

7. Verify that the plugin is installed correctly:

```
# fuel plugins
id | name | version | package_version
---|-----|-----|-----
1 | fuel-plugin-manila | 1.0.0 | 4.0.0
```

Uninstall Manila plugin for Fuel

To uninstall Manila plugin for fuel, follow the steps below:

1. Log in to the Fuel Master node CLI
2. Delete all environments in which Manila plugin for Fuel is enabled:

Example:

```
# fuel --env <ENV_ID> env delete
```

3. Uninstall the plugin:

```
# fuel --plugins --remove fuel-plugin-manila==1.0.0
```

4. Verify wheter the Manila plugin for Fuel was uninstalled successfully:

```
# fuel plugins
```

Proceed to *Configure Manila plugin for Fuel*.

Configure Manila plugin for Fuel

Configuring and deploying an environment with Manila plugin for Fuel involves creating an environment in Fuel and modifying the environment settings.

To configure OpenStack environment with Manila plugin:

1. Create an OpenStack environment as described in the [Fuel User Guide](#):
2. In the *Additional services* menu, select *Install Manila*:

Create a new OpenStack environment
✕

Name and Release

Compute

Networking Setup

Storage Backends

Additional Services

Finish

- Install Sahara** ⓘ
 Sahara enables on-demand provisioning of Hadoop clusters to be deployed on OpenStack utilizing a variety of vendor distributions.
- Install Murano** ⓘ
 Murano is an application catalog, which allows application developers and cloud administrators to publish various cloud-ready applications in a browsable categorized catalog, which may be used by the cloud users (including the inexperienced ones) to pick-up the needed applications and services and composes the reliable environments out of them in a "push-the-button" manner. Additional features may be enabled by installing the newest version of the Murano plugin.
- Install Ceilometer and Aodh** ⓘ
 Ceilometer provides metering and monitoring of an OpenStack cloud. Aodh is an alarming service which uses the data collected by Ceilometer.
- Install Ironic** ⓘ
 Ironic enables baremetal provisioning.
- Install Manila** ⓘ
 Manila is the Shared Network As A Service project

Cancel

← Prev

Next →

3. Follow next steps of the [Create a new OpenStack environment](#) instruction.

1. In the *Nodes* tab of the Fuel web UI [add](#) at least one node with roles manila-share and manila-data:

The screenshot displays two node entries in the Fuel web UI. The first entry is titled 'Manila share (1)' and contains a single node: 'Untitled (64:65)' with the role 'MANILA-SHARE'. The second entry is titled 'Manila data (1)' and contains a single node: 'Untitled (24:f9)' with the role 'MANILA-DATA'. Both nodes are in a 'DISCOVERED PENDING ADDITION' state and have hardware specifications of CPU: 4 (4), RAM: 4.0 GB, and HDD: 1.5 TB. Each node entry includes a selection checkbox, a 'Qemu' icon, a document icon, a refresh icon, and a settings gear icon.

2. In the *Settings* tab, click *OpenStack Services*:

- (a) Check that *Enable Manila service* is enabled.
- (b) Set the choosend backend for Manila.
 - i. For generic driver specify the *Image name* exactly same as you set on the plugin installation stage.
 - ii. For NetApp driver specify hostname, credential and parameters related to your environment.

enable Manila service 

Versions 1.0.0

Use the Generic driver
Use the cinder volumes as a backend for manila shares

Image name Name of the service image for generic driver

Use the NetApp driver
Use the NetApp onTap storage as a backend for manila shares

NetApp transport type

- https
Choose this protocol for encrypted connection
- http
Choose this protocol for not encrypted connection

NetApp server hostname Set the address, port and protocol for to the server access

NetApp server port

NetApp server username Set the username for to the server access

NetApp server password Set the password for to the server access

NetApp root volume aggregate Set the parameter netapp_root_volume_aggregate

NetApp search pattern for aggregation names Set the parameter netapp_aggregate_name_search_pattern

NetApp search pattern for storage port names Set the parameter netapp_port_name_search_pattern

USE MANILA PLUGIN FOR FUEL

Using File Share as a Service possibility with Manila plugin for Fuel

Once you deploy an OpenStack environment with Manila plugin for Fuel, you can start using it both from CLI and Horizon. The topic of CLI usage is too big to put it into this User Guide and it well described [here](#).

The Horizon usage is very obvious. You can notice that two new tabs appears on the Admin and Project/Compute sections:

The screenshot shows the OpenStack Horizon dashboard. The top header includes the OpenStack logo, the word "MIRANTIS" in a red box, and "OpenStack DASHBOARD". A user menu shows "admin" with a dropdown arrow. The left sidebar has a "System" section expanded, listing various system components. The main content area is titled "Shares" and has several tabs: "Shares", "Snapshots", "Share Networks", "Security Services", "Share Types", and "Share Servers". The "Shares" tab is active. Below the tabs, there is a table with two columns: "NAME" and "EXTRA SPECS". One row is visible, showing "default_share_type" with extra specs "snapshot_support=True" and "driver_handles_share_servers=True". Below the table, it says "Displaying 1 item". At the bottom of the sidebar, the word "Shares" is highlighted in red.

NAME	EXTRA SPECS
default_share_type	snapshot_support=True driver_handles_share_servers=True

The screenshot shows the OpenStack Dashboard interface. At the top, there is a header with the OpenStack logo and 'MIRANTIS DASHBOARD' on the left, and a user profile 'admin' on the right. The main content area is titled 'Shares' and has four tabs: 'Shares' (selected), 'Snapshots', 'Share Networks', and 'Security Services'. Below the tabs is a table with three columns: 'NAME', 'DESCRIPTION', and 'METADATA'. The table is currently empty. On the left side, there is a sidebar menu with categories: 'Project', 'Compute' (expanded to show Overview, Instances, Volumes, Images, and Access & Security), 'Shares' (highlighted in red), 'Network', 'Orchestration', 'Object Store', 'Admin', and 'Identity'.

Let's go over the basic usage of Manila:

1. Firstly we have to create a share network for access to new share:

Create Share Network ✕

Name *

Description:

Share networks contain network data, that will be used for creation of service VM, where will be hosted shares.

Description

Neutron Net *

Neutron Subnet *

Cancel Create Share Network

2. After that we can create new share:

Create Share

✕

Share Name *

Description

Share Protocol *

NFS

Size (GiB) *

1

▲
▼

Share Type *

default_share_type

Availability Zone

nova

Share Network *

default_share_network

Metadata

Make visible for all ⓘ

Description:

Select parameters of share you want to create.

Metadata:

One line - one action. Empty strings will be ignored.
To add metadata use:

key=value

Share Limits

Total Gibibytes (0 GiB)
1,000 GiB Available

Number of Shares (0)
50 Available

Cancel
Create Share

3. When the share becomes available it should be configured for future usage. At least necessary to add new rule in order to allow mounting new share.

Shares

Shares | Snapshots | Share Networks | Security Services

Filter Create Share Delete Shares

NAME	DESCRIPTION	METADATA	SIZE	STATUS	VISIBILITY	PROTOCOL	SHARE NETWORK	ACTIONS
share	-		1GIB	Available	private	NFS	default_share_network	Edit Share Extend Share Create Snapshot Delete Share Manage Rules Edit Share Metadata

Displaying 1 item

Add Rule ✕

Access Type *

Access Level *

Access To *

Description:

Add policy rule to share, 'ip' rule represents ipv4 address, 'user' rule represents username or usergroup.

Cancel Add Rule

After that new share could be consumed in your environment.

Troubleshooting

This section contains a guidance on how to ensure that the Manila plugin is up and running on your deployed environment.

To find logs

The Manila places its log by convenient path. On controller:

- /var/log/manila/manila-api.log
- /var/log/manila/manila-scheduler.log

On a manila-share node:

- /var/log/manila/manila-share.log

On a manila-data node

- /var/log/manila/manila-data.log

To verify Manila configuration files

Check that /etc/manila directory contains following files:

- -rw-r--r-- 1 manila manila 1.8K Oct 19 02:35 api-paste.ini
- -rw-r--r-- 1 manila manila 1.3K Oct 19 02:35 logging_sample.conf
- -rw-r--r-- 1 root root 2.6K Oct 19 03:44 manila.conf
- -rw-r--r-- 1 manila manila 5.2K Oct 19 02:35 policy.json
- -rw-r--r-- 1 root root 989 Oct 19 02:35 rootwrap.conf
- drwxr-xr-x 2 manila manila 4.0K Oct 19 02:35 rootwrap.d

To verify Manila services

Check output of the commands on any controller node:

```
# . /root/openrc
# manila service-list
```

All services should be in the *up* stage.

In case of using self signed certificates

Use the `--insecure` option for all console commands. For example:

```
# manila --insecure list
# manila --insecure type-create some_share_type True
```

Links

For more information about the software discussed in this document, see the following links:

- The Manila [wiki page](#).
- The Manila plugin for Fuel [repository](#).

- The LaunchPad project [URL](#) used to report bugs found in the plugin.