
ploy Documentation

Release 1.0

Florian Schulze

Feb 03, 2018

Contents

1	ploy	1
1.1	Overview	1
1.2	Installation	1
1.3	Configuration	1
1.4	Plugins	1
1.5	SSH integration	3
1.6	Instance names	3
1.7	Macro expansion	3
1.8	Massaging of config values	4
1.9	Buildout specifics	4
1.10	Changelog	5
2	Plugins	13
2.1	ploy_ansible	13
2.2	ploy_ec2	18
2.3	ploy_ezjail	21
2.4	ploy_fabric	25
2.5	ploy_openvz	28
2.6	ploy_virtualbox	28
3	Indices and tables	33

1.1 Overview

Ploy is a commandline-tool to provision, manage and control server instances. There are plugins for EC2 (`ploy_ec2`), FreeBSD Jails (`ploy_ezjail`) and more. You can create, delete, monitor and ssh into instances while ploy handles the details like ssh fingerprint checking. Additional plugins provide advanced functionality like integrating Fabric (`ploy_fabric`) and Ansible (`ploy_ansible`).

You can find the detailed documentation at <http://ploy.readthedocs.org/en/latest/>

1.2 Installation

ploy is best installed with `easy_install`, `pip` or with `zc.recipe.egg` in a buildout. It installs two scripts, `ploy` and `ploy-ssh`.

1.3 Configuration

All information about server instances is located in `ploy.conf`, which by default is looked up in `etc/ploy.conf`.

1.4 Plugins

Support for backends and further functionality is implemented by plugins. One plugin is included with ploy.

plain For regular servers accessible via ssh.

You can see which plugins are available in your current installation with `ploy -v`.

1.4.1 Plain

With plain instances you can put infos about servers into the configuration to benefit from some ploy features like ssh fingerprint checking and plugins like the Fabric integration.

Options

host or ip (required) The host name or address for the server.

user The default user for ssh connections. If it's set to * then the current local user name is used.

port The ssh port number.

ssh-host-keys or ssh-fingerprints (required) The public ssh host keys (or fingerprints) of the server.

ssh-host-keys It's best to use the public ssh host keys if available. You can either specify the path, or put in the contents directly (`ssh-rsa AAAA...`). Multiple host keys can be specified one per line, or separated by commas.

ssh-fingerprints If set to `ask` then manual interactive verification is enabled. If set to `ignore` then no verification is performed at all! Multiple fingerprints can be specified one per line, or separated by commas. The format of fingerprints is either 16 bytes as hex numbers separated by colons, the same 16 bytes prefixed by `MD5:`, or the hash type, followed by a colon and the base 64 encoded hash digest.

password-fallback If this boolean is true, then using a password as fallback is enabled if the ssh key doesn't work. This is off by default. You may be asked more than once for the password. The first time is by paramiko which always happens, but is remembered. The other times is by the ssh command line tool if it's invoked.

password Never use this directly! If password-fallback is enabled this password is used. This is mainly meant for Fabric scripts which have other ways to get the password. On use case is bootstrapping [FreeBSD](#) from an [mfsBSD](#) distribution where the password is fixed.

proxycommand The command to use in the ProxyCommand option for ssh when using the `ploy-ssh` command. There are some variables which can be used:

path The directory of the ploy.conf file. Useful if you want to use the `ploy-ssh` command itself for the proxy.

known_hosts The absolute path to the known_hosts file managed by ploy.

instances The variables of other instances. For example: `instances.foo.ip`

In addition to these the variables of the instance itself are available.

A full example for a proxycommand:

```
proxycommand = {path}/../bin/ploy-ssh vm-master -W {ip}:22
```

ssh-key-filename Location of private ssh key to use.

ssh-extra-args A list of settings separated by newlines passed on to ssh.

Example:

```
ssh-extra-args = ForwardAgent yes
```

1.5 SSH integration

ploy provides an additional tool `ploy-ssh` to easily perform SSH based operations against named instances. Particularly, it encapsulates the entire *SSH fingerprint* mechanism. For example EC2 instances are often short-lived and normally trigger warnings, especially, if you are using elastic IPs.

Note:: it does so not by simply turning off these checks, but by transparently updating its own fingerprint list using mechanisms provided by the backend plugins.

The easiest scenario is simply to create an SSH session with an instance. You can either use the `ssh` subcommand of the `ploy` tool like so:

```
ploy ssh INSTANCENAME
```

Alternatively you can use the `ploy-ssh` command directly, like so:

```
ploy-ssh INSTANCENAME
```

The latter has been provided to support `scp` and `rsync`. Here are some examples, you get the idea:

```
scp -S `pwd`/bin/ploy-ssh some.file demo-server:/some/path/
rsync -e "bin/ploy-ssh" some/path fschulze@demo-server:/some/path
```

1.6 Instance names

Instances have an **id** which is the part after the colon in the configuration. They also have a **unique id** which has the form `[masterid]-[instanceid]`. The `[masterid]` depends on the plugin. For plain instances it is `plain`. The `[instanceid]` is the **id** of the instance. So, if you have the following config:

```
[plain-instance:foo]
...
```

Then the **unique id** of the instance is `plain-foo`.

1.7 Macro expansion

In the `ploy.conf` you can use macro expansion for cleaner configuration files. That looks like this:

```
[ec2-instance:demo-server2]
<= demo-server
securitygroups = demo-server2

[ec2-securitygroup:demo-server2]
<= demo-server
```

All the options from the specified macro are copied with some exceptions depending on the backend plugin.

If you want to copy data from some other kind of options, you can add a colon in the macro name. This is useful if you want to have a base for instances like this:

```
[macro:base-instance]
keypair = default
region = eu-west-1
```

```
placement = eu-west-1a

[ec2-instance:server]
<= macro:base-instance
...
```

1.8 Messaging of config values

Plugins and ploy message certain string values from the config to convert them to other types and do formatting or expansion.

You can use that yourself, which is useful for the Fabric integration and other things.

Here is a simple example:

```
[section]
messagers =
    intvalue=ploy.config.IntegerMessenger
    boolvalue=ploy.config.BooleanMessenger
intvalue = 1
boolvalue = yes
```

If you now access those values from for example a fabric task, you get the correct type instead of strings.

The above syntax registers the messagers only for that section. You can register messagers for other sections or even section groups with this syntax:

```
messagers =
    [option]=[sectiongroup]:import.path.to.messenger
    [option]=[sectiongroup]:[section]:import.path.to.messenger
```

The parts have the following meaning:

- [option]** This is the name of the option which should be massaged
- [sectiongroup]** The name of the section group. That's the part before the optional colon in a section. To match sections without a colon, use `global`. To match every section, use `*`.
- [section]** The name of the section to which this messenger is applied. If empty, the current section is used.

1.9 Buildout specifics

With `zc.recipe.egg` you can set a custom configfile location like this:

```
[ploy]
recipe = zc.recipe.egg
eggs = ploy
arguments = configpath="${buildout:directory}/etc/", configname="servers.cfg"
```


1.10 Changelog

1.10.1 1.5.1 - 2017-12-17

- Fix getting fingerprints automatically when `ssh-host-keys` is used. [fschulze]

1.10.2 1.5.0 - 2017-10-03

- New `ssh-host-keys` option which allows to set host keys directly to force paramiko to use a specific key type to avoid fingerprint mismatches. [fschulze]

1.10.3 1.4.0 - 2017-10-02

- Allow instance implementations to return multiple fingerprints via `get_fingerprints` method. [fschulze]
- Support other key types than just `rsa`, because paramiko > 2 defaults to `ed25519`. [fschulze]
- Output more information about used keys on fingerprint mismatch. [fschulze]

1.10.4 1.3.1 - 2016-06-02

- Don't get `ssh-fingerprints` from master if not set in instance config. This fixes automatic fingerprint fetching for EC2, ezjail etc. [fschulze]

1.10.5 1.3.0 - 2016-05-21

- Add option `ssh-fingerprints` which allows to specify multiple fingerprints. [fschulze]
- Support new output of `ssh-keygen` which includes the hash type and defaults to `SHA256`. [fschulze]

1.10.6 1.2.1 - 2015-08-27

- Allow to specify multiple masters per instance. [fschulze]

1.10.7 1.2.0 - 2015-03-05

- Add `Executor` helper to handle local and remote command execution. It's also handling `ssh` agent forwarding enabled by either the users `ssh` config or the `ssh-extra-args` option. [fschulze]

1.10.8 1.1.0 - 2015-02-28

- Add `ssh-extra-args` option. [fschulze]
- Add `annotate` command to print the configuration with the source of each setting. [fschulze]
- Allow custom shebang in gzipped startup scripts. [fschulze]

1.10.9 1.0.3 - 2015-01-22

- Drop bad entries from our `known_hosts` file to prevent failures in paramiko. [fschulze]
- Set `StrictHostKeyChecking=yes` for all ssh connections to prevent interactive asking. [fschulze]

1.10.10 1.0.2 - 2014-10-04

- Ask before terminating an instance. [fschulze]
- Fix config setting propagation in some cases of proxied instances. [fschulze]
- Close all connections before exiting. This prevents hangs caused by open proxy command threads. [fschulze]
- Add option to log debug output. [fschulze]
- Add helpers to setup `proxycommand` in plugins. [fschulze]

1.10.11 1.0.1 - 2014-08-13

- Fix error output for plain instances on ssh connection failures. [fschulze]

1.10.12 1.0.0 - 2014-07-19

- Fix removal of bad host keys when using non standard ssh port. [fschulze]
- Renamed `plain-master` to `plain`, so the uids of instances are nicer. [fschulze]

1.10.13 1.0rc15 - 2014-07-16

- Only remove bad host key from `known_hosts` instead of clearing it completely. [fschulze]
- Removed support for `proxyhost` option. It caused hangs and failures on missing or invalid ssh fingerprints. [fschulze]
- Allow empty `startup_script` option to mean use no startup script. [fschulze]

1.10.14 1.0rc14 - 2014-07-15

- Allow `fingerprint` to be set to a public host key file. [fschulze]

1.10.15 1.0rc13 - 2014-07-08

- Better error message for instances missing because the plugin isn't installed. [fschulze]
- Fix tests when ploy itself isn't installed. [fschulze]

1.10.16 1.0rc12 - 2014-07-08

- Use plain `conftest.py` instead of `pytest` plugin. [fschulze]

1.10.17 1.0rc11 - 2014-07-05

- Fix uid method for master instances. [fschulze]

1.10.18 1.0rc10 - 2014-07-04

- Print plugin versions with `-v` and `--versions`. [fschulze]
- Python 3 compatibility. [fschulze]

1.10.19 1.0rc9 - 2014-06-29

- Let plugins add type of lists to show with the `list` command. [fschulze]
- Use `server` and `instance` consistently. [fschulze]
- Always make instances accessible by their full name in the form of “[master_id]-[instance_id]”. Only if there is no conflict, the short version with just “[instance_id]” is also available for convenience. [fschulze]
- Add instance id validator which limits to letters, numbers, dashes and underscores. [fschulze]
- Renamed from `mr.awesome` to `ploy`. [fschulze]

1.10.20 1.0rc8 - 2014-06-16

- Give a bit more info on ssh connection failures. [fschulze]

1.10.21 1.0rc7 - 2014-06-11

- Expose some test fixtures for reuse in plugins. [fschulze]
- Add `before_terminate` and `after_start` hooks and make it simple for plugins to add their own hooks. [fschulze]

1.10.22 1.0rc6 - 2014-06-10

- Add `get_path` method to `ConfigSection` class. [fschulze]

1.10.23 1.0rc5 - 2014-06-09

- Provide helper method `ssh_args_from_info` on `BaseInstance` to get the arguments for running the ssh executable from the info provided by `init_ssh_key`. [fschulze]
- Allow overwriting the command name in help messages for `bsdploy`. [fschulze]
- Make `debug` command usable for instances that don't have a startup script. [fschulze]
- Instances can provide a `get_port` method to return a default port. [fschulze]
- Catch socket errors in `init_ssh_key` of plain instances to print additional info for debugging. [fschulze]
- Delay setting of config file path to expose too early use of config in plugins. Refs #29 [fschulze]

1.10.24 1.0rc4 - 2014-05-21

- Fix massagers for [instance:...] sections. [fschulze]
- Copy massagers in ConfigSection.copy, so overrides in startup script work correctly. [fschulze]

1.10.25 1.0rc3 - 2014-05-15

- Fetch fingerprints only when necessary. This speeds up connections when the fingerprint in known_hosts is still valid. [fschulze]

1.10.26 1.0rc2 - 2014-05-14

- Moved setuptools-git from setup.py to .travis.yml, it's only needed for releases and testing. [fschulze]
- More tests. [fschulze]

1.10.27 1.0rc1 - 2014-03-23

- Test, enhance and document adding massagers via config. [fschulze]
- Moved ec2 and fabric integration into separate plugins. [fschulze]
- You can now have instances with the same name if they belong to different masters, they will then get the name of the master as a prefix to their name. [fschulze]
- Add possibility to overwrite the default config name. [tomster]
- Improved proxycommand and documented it. [fschulze]
- Make the AWS instance available in masters. This changes the get_masters plugin interface. [fschulze]
- Use os.execvp instead of subprocess.call. This allows the use of ssh in the proxycommand option, which greatly simplifies its use. [fschulze]
- Added command plugin hooks. [fschulze]
- The variable substitution for the proxycommand option now makes the other instances available in a dict under instances. And adds known_hosts. [fschulze]
- Load plugins via entry points instead of the plugin section in the config. [fschulze]
- Allow fallback to password for ssh to plain instances. [fschulze]
- Add option to ask for manual fingerprint validation for plain instances. [fschulze]

1.10.28 0.13 - 2013-09-20

- Use os.path.expanduser on all paths, so that one can use ~ in config values like the aws keys. [fschulze]

1.10.29 0.12 - 2013-09-11

- There is no need to add the AWS account id to security group names anymore. [fschulze]
- Rules are removed from security groups if they aren't defined in the config. [fschulze]
- Allow adding of custom config massagers from inside the config. [fschulze]

- Support block device maps to enable use of more than one ephemeral disk. [fschulze]
- Added `do` method on `ec2` and `plain` instances which allows to call fabric commands. [fschulze]
- Use `PathMassager` for `access-key-id` and `secret-access-key` in the `ec2-master` section. This might break existing relative paths for these options. [fschulze]
- Added support for EBS boot instances. [fschulze]
- Add option `ssh-key-filename` to point to a private ssh key for `ec2` and `plain` instances. [fschulze]
- Fix Fabric integration for newer versions of Fabric. [fschulze]
- Support `proxycommand` option for `plain` instances. This also caused a change in the `init_ssh_key` API for plugins. [fschulze]
- Support `ProxyCommand` from `~/.ssh/config` for `plain` instances. Requires Fabric 1.5.0 and Paramiko 1.9.0 or newer. [fschulze]

1.10.30 0.11 - 2012-11-08

- Support both the `ssh` and `paramiko` libraries depending on which Fabric version is used. [fschulze]

1.10.31 0.10 - 2012-06-04

- Added `ec2-connection` which helps in writing Fabric scripts which don't connect to a server but need access to the config and AWS (like uploading something to S3). [fschulze]
- Fix several problems with using a user name other than `root` for the `do` and `ssh` commands. [fschulze]
- Require Fabric `>= 1.3.0`. [fschulze]
- Require boto `>= 2.0`. [fschulze]
- Added hook for startup script options. [fschulze]
- Added possibility to configure hooks. [fschulze]
- Refactored to enable plugins for different virtualization or cloud providers. [fschulze]
- Added lots of tests. [fschulze]

1.10.32 0.9 - 2010-12-09

- Overwrites now also affect server creation, not just the startup script. [fschulze]
- Added `list` command which supports just listing `snapshots` for now. [fschulze]
- Added `delete-volumes-on-terminate` option to delete volumes created from snapshots on instance termination. [fschulze]
- Added support for creating volumes from snapshots on instance start. [natea, fschulze]
- Added support for `~/.ssh/config`. This is a bit limited, because the paramiko config parser isn't very good. [fschulze]
- Added `help` command which provides some info for `zsh` autocompletion. [fschulze]

1.10.33 0.8 - 2010-04-21

- For the `do` command the Fabric options `reject_unknown_hosts` and `disable_known_hosts` now default to true. [fschulze]
- Allow adding normal servers to use with `ssh` and `do` commands. [fschulze]
- Refactored `ssh` connection handling to only open network connections when needed. Any fabric option which doesn't need a connection runs right away now (like `-h` and `-l`). [fschulze]
- Fix status output after `start`. [fschulze]

1.10.34 0.7 - 2010-03-22

- Added `snapshot` method to `Server` class for easy access from fabfiles. [fschulze]

1.10.35 0.6 - 2010-03-18

- It's now possible to specify files which contain the aws keys in the `[aws]` section with the `access-key-id` and `secret-access-key` options. [fschulze]
- Added `-c/--config` option to specify the config file to use. [fschulze]
- Added `-v/--version` option. [tomster (Tom Lazar), fschulze]
- Comment lines in the startup script are now removed before any variables in it are expanded, not afterwards. [fschulze]
- Use `argparse` library instead of `optparse` for more powerful command line parsing. [fschulze]

1.10.36 0.5 - 2010-03-11

- Added gzipping of startup script by looking for `gzip:` prefix in the filename. [fschulze]
- Added macro expansion similar to `zc.buildout 1.4`. [fschulze]

1.10.37 0.4 - 2010-02-18

- Check console output in `status` and tell user about it. [fschulze]
- Friendly message instead of traceback when trying to `ssh` into an unavailable server. [fschulze]
- Remove comment lines from startup script if it's starting with `#!/bin/sh` or `#!/bin/bash`. [fschulze]
- Removed `-r` option for `start` and `debug` commands and replaced it with more general `-o` option. [fschulze]
- Made startup script optional (not all AMIs support it, especially Windows ones). [fschulze]
- The `stop` command actually only stops an instance now (only works with instances booted from an EBS volume) and the new `terminate` command now does what `stop` did before. [fschulze]
- Better error message when no console output is available for `ssh` finger print validation. [fschulze]
- Fixed indentation in documentation. [natea (Nate Aune), fschulze]

1.10.38 0.3 - 2010-02-08

- Removed the `[host_string]` prefix of the `do` command output. [fschulze]

1.10.39 0.2 - 2010-02-02

- Snapshots automatically get a description with date and volume id. [fschulze]
- The `ssh` command can now be used with `scp` and `rsync`. [fschulze]

1.10.40 0.1 - 2010-01-21

- Initial release [fschulze]

2.1 ploy_ansible

2.1.1 Overview

The `ploy_ansible` plugin provides integration of `Ansible` with `ploy`. It automatically builds an `inventory` and provides a custom connection plugin.

2.1.2 Installation

`ploy_ansible` is best installed with `easy_install`, `pip` or with `zc.recipe.egg` in a buildout.

2.1.3 Commands

The plugin adds the following commands to `ploy`.

configure Configures an instance. There are three ways to specify how to configure an instance. Applying the roles given by the `roles` option of an instance, a playbook set by the `playbook` option or a playbook with the unique name of the instance found in the `playbooks-directory`. Using `roles` or a `playbook` is mutually exclusive. If you specify a `playbook` and there is also a `playbook` in the default location, you will get a warning.

inventory Lists all known groups and their associated hosts, including regular default groups, such as `all` but also implicit, `ploy_ansible` groups such as instances of a particular master (i.e. all `ez`-instances of an `ez-master`)

ansible Runs an Ansible command. This basically reflects the `ansible` script of Ansible.

playbook Applies a playbook. This basically reflects the `ansible-playbook` script of Ansible.

vault Manages file encryption. This basically reflects the `ansible-vault` script of Ansible, but handles the encryption key source via `ploy.conf`.

vault-key Manages the vault key.

2.1.4 Options

Global

playbooks-directory

The `playbooks-directory` option of the `ansible` section allows you to specify the directory where playbooks, roles, `host_vars` etc are looked up. If you specify a relative path, then it's always relative to the `ploy.conf` directory. If you have a structure like this:

```
project
|-- deployment
| |-- roles
| |-- host_vars
|
|-- etc
|
|-- ploy.conf
```

Then you would put the following into your `ploy.conf`:

```
[ansible]
playbooks-directory = ../deployment
```

By default it is set to the parent directory of the directory the `ploy.conf` is located at like this:

```
project
|-- roles
|-- host_vars
|-- etc
|
|-- ploy.conf
```

vault-password-source

Using the `keyring` library, you can store the encryption key for the Ansible vault in your keychain.

The `vault-password-source` option is the id used in your keychain. The id must be unique among all people who have to use the feature, as it is used as an identifier in their keychain. If in doubt, use a speaking prefix and add a guid by running `python -c "import uuid; print(uuid.uuid4()).hex)"`.

If you want to rekey your files, you have to put the old id into the `vault-password-old-source` option and set a new id in `vault-password-source`. Just incrementing a number or appending a new guid is best.

Example:

```
[ansible]
vault-password-old-source = my-domain-deployment-0da2c8296f744c90a236721486dbd258
vault-password-source = my-domain-deployment-042a98b666ec4e4e8e06de7d42688f3b
```

You can manage your key with the `vault-key` command. For easy exchange with other developers, you can also export and import the key via `gpg` using the `vault-key export` and `vault-key import` commands.

Per instance

groups Whitespace separated list of Ansible group names this instance should be added to in addition to the default ones.

roles Used by the `configure` command. This allows you to configure an instance by applying the whitespace separated roles. This is like creating a playbook which only specifies a host and a list of roles names. If the `sudo` option is set, it's also set for the generated playbook.

playbook Allows you to explicitly specify a playbook to use for this instance. If you need `sudo`, then you have to add it yourself in that playbook.

Any option starting with `ansible_` is passed through to Ansible as is. This can be used for settings like `ansible_python_interpreter`.

Any option starting with `ansible-` is stripped of the `ansible-` prefix and then passed through to Ansible. This is the main way to set Ansible variables for use in playbooks and roles.

All other options are prefixed with `ploy_` and made available to Ansible.

2.1.5 Ansible inventory

All instances in `ploy.conf` are available to Ansible via their **unique id**.

The variables for each instance are gathered from `group_vars`, `host_vars` and the `ploy.conf`.

2.1.6 Ansible lookup plugins

The `ploy_crypted` lookup plugin can be used in playbooks to read the content of encrypted files. This is another way to access encrypted data where you don't have to move that data into yml files. An added benefit is, that the file is only decrypted when it is actually accessed. If you run tasks filtered by tags and those tasks don't access the encrypted data, then it's not decrypted at all.

Warning: This lookup plugin only works with files that are plain ascii or utf-8. It's a limitation caused by the way ansible handles variable substitution.

2.1.7 API usage

On the Python side, each `ploy` instance gains the following methods:

apply_playbook(self, playbook, *args, **kwargs) Applies the `playbook` to the instance.

has_playbook Return `True` if the instance has either of the `roles` or a `playbook` option set.

get_playbook(*args, **kwargs) Returns an instance of the Ansible internal `PlayBook` class. This is either from a file (from `playbook` option or the `playbook` kwarg), or dynamically generated from the `roles` option.

configure(*args, **kwargs) Configures the instance with the same semantics as the `configure` command.

get_ansible_variables Returns the Ansible variables from the inventory. This does not include `facts`, as it doesn't connect to the instance. This is particularly useful in Fabric scripts.

get_vault_lib Returns a readily usable Ansible `VaultLib` class. Use the `encrypt` and `decrypt` methods to encrypt/decrypt strings.

2.1.8 Changelog

1.4.0 - 2017-12-17

- Look for `[instance-name].yaml` in addition to `[master-name]-[instance-name].yaml`. This allows using the same playbook for the same instance on multiple masters. [fschulze]

1.3.2 - 2016-06-02

- Don't add empty search path when no additional role or library paths are defined. This prevents the current working directory from being searched. [fschulze]

1.3.1 - 2015-09-03

- Update Ansible requirement to `< 2.dev0`. The upcoming 2.0.0 has way too many internal changes to be supported. [fschulze]
- Add hosts only once in Inventory. [fschulze]

1.3.0 - 2015-04-10

- Added handling of `groups` option of instances to allow definition of additional Ansible groups. [fschulze]
- Get host variables on demand instead of at startup. If you have many hosts with encrypted yaml files, this speeds things up considerably in most cases. [fschulze]
- Fixes for changes in ansible 1.9. [fschulze]
- Added `inventory` command to list all known groups and their associated hosts. [fschulze]

1.2.4 - 2015-02-28

- Pass on the `sudo` setting if the `roles` option is used. [fschulze]

1.2.3 - 2015-02-28

- Fix sudo support for ansible `> 1.6`. [fschulze]
- Print warning when using an untested version of ansible. [fschulze]
- If ansible isn't installed, then require `>= 1.8` as that doesn't violate the sandbox of buildout anymore. [fschulze]

1.2.2 - 2015-02-18

- Test and fixes for changes in ansible 1.8. [fschulze]

1.2.1 - 2015-01-06

- Limit Ansible to pre 1.8, as `> 1.8` breaks stuff. [fschulze]

1.2.0 - 2014-10-27

- Always set `ansible_ssh_user` in inventory. [fschulze]
- Clear host and pattern cache after calling original `Inventory.__init__` method. [fschulze]
- Add `--extra-vars` option to `configure` command. [witsch (Andreas Zeidler)]
- Provide `ploy_crypted` lookup plugin to load encrypted files into Ansible variables. Only ascii and utf8 encoded files will work. [fschulze]
- Expand Ansible variables in `get_ansible_variables` method. [fschulze]
- Support Ansible vault with safe key storage via keyring library, so you don't have to type it in or have it in an unprotected file. [fschulze]

1.1.0 - 2014-08-13

- Test and fixes for changes in ansible 1.7. [fschulze]
- Add verbosity argument to `configure` command. [fschulze]

1.0.0 - 2014-07-19

- Added documentation. [fschulze]

1.0b8 - 2014-07-15

- Add ansible as dependency if it can't be imported already. [fschulze]

1.0b7 - 2014-07-08

- Packaging and test fixes. [fschulze]

1.0b6 - 2014-07-04

- Use unique instance id to avoid issues. [fschulze]
- Renamed `mr.awesome` to `ploy` and `mr.awesome.ansible` to `ploy_ansible`. [fschulze]

1.0b5 - 2014-06-16

- Set user in playbook to the one from the config if it's not set already. [fschulze]
- Change default playbook directory from the `aws.conf` directory to it's parent. [fschulze]

1.0b4 - 2014-06-11

- Added `playbook` and `roles` config options for instances. [fschulze]
- Added `has_playbook` and `configure` methods to instances. [fschulze]
- Added `before/after_ansible_configure` hooks. [fschulze]

1.0b3 - 2014-06-09

- Use `execnet` for connections. There is only one ssh connection per host and it's reused for all commands. [fschulze]
- Make sure the playbook directory is always absolute. [fschulze]
- Prevent use of persistent ssh connections, as that easily results in connections to wrong jails because of the proxying. This makes ansible a lot slower at the moment. [fschulze]
- Add support for `su` and `vault` (ansible 1.5) as well as `--force-handlers` (ansible 1.6). [fschulze]
- Removed `ansible` from install requirements. It won't install in a buildout so it needs to be installed in a `virtualenv` or via a system package. [fschulze]

1.0b2 - 2014-05-15

- Add `configure` command which is a stripped down variant of the `playbook` command with assumptions about the location of the yml file. [fschulze]
- Warn if a playbook is requested for a host that is not configured in the playbook hosts list. [fschulze]
- Allow `mr.awesome` plugins to add ansible variables. [fschulze]
- Inject the ansible paths sooner as they may not apply in some cases otherwise. [fschulze]
- Moved `setuptools-git` from `setup.py` to `.travis.yml`, it's only needed for releases and testing. [fschulze]

1.0b1 - 2014-03-24

- Initial release [fschulze]

2.2 ploy_ec2

2.2.1 Overview

The `ploy_ec2` plugin provides integration of [Amazon EC2](#) with `ploy`.

2.2.2 Installation

`ploy_ec2` is best installed with `easy_install`, `pip` or with `zc.recipe.egg` in a buildout.

2.2.3 Masters

To use `ploy_ec2` you need an Amazon account and [AWS keys](#).

Once you got your keys, you should put them in a secure location and reference them in your `ploy.conf`. Additionally you need to set the region of the master:

```
[ec2-master:ec2eu]
access-key-id = ~/.aws/ec2.id
secret-access-key = ~/.aws/ec2.key
region = eu-west-1
```

You can also set the `AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY` environment variables instead.

You need to define a master for each `region` you want to use.

2.2.4 Instances

Each instance has the following mandatory settings:

image The [Amazon Machine Image \(AMI\)](#) that this instance will start up with.

keypair The name of the [SSH keypair](#) to use.

placement The [availability zone](#) in which to launch the instances.

securitygroups The name of the [Securitygroups](#) this instance should be assigned to.

The following settings are optional:

`instance_type`

`ip`

startup_script Path to a script which will be run right after creation and first start of the instance. This uses the [User Data](#) feature and needs to be supported by the AMI.

`volumes`

`snapshots`

`device_map`

`delete-volumes-on-terminate`

2.2.5 Securitygroups

`description`

connections

```
[ec2-securitygroup:app-server]
description = The production server
connections =
  tcp      22      22      0.0.0.0/0
  tcp      80      80      0.0.0.0/0
```

2.2.6 Volumes

You can define volumes via `ec2-volume` sections. The id of the section must not start with `vol-`. You can declare the `size` as a number of GB.

If the volume doesn't exist, it is automatically created.

```
[ec2-volume:a-volume-name]
size = 100

[ec2-instance:foo]
...
volumes = a-volume-name /dev/sdf
```

2.2.7 Macro expansion

For instances the `ip` and `volumes` options aren't copied when expanding macros.

2.2.8 Fingerprint verification

Automatic ssh fingerprint verification works by checking whether the fingerprint is in the console output of the instance.

After reboot or stop/start of an instance, the console output is refreshed. The problem with that is, that the fingerprint isn't included in the console anymore by default. To fix that you need to log the fingerprint on reboot somehow. One way to do that with Ubuntu is to add a script at `/var/lib/cloud/scripts/per-boot/ssh-keys` with this content:

```
#!/bin/sh
/usr/bin/ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
```

Make sure that script is executable.

2.2.9 Changelog

1.2.1 - Unreleased

- Support yet another ssh key output format from console output of instance. [fschulze]

1.2.0 - 2015-09-03

- Check status of volume to give helpful error message if it's still attached. [fschulze]
- Allow volume definition via `ec2-volume` sections. [fschulze]
- Fix support of console output for the `ploy debug -c` command. [fschulze]
- Reuse `init_ssh_key` from `ploy.plain` to get way more options and error checking. [fschulze]

1.1.1 - 2015-01-22

- Only set `device_map` if it's in the config, the previous `None` default didn't always work. [fschulze]
- Fixed console output availability test for the `status` command. [fschulze]
- Better error message if fingerprint isn't in console output. [fschulze]
- There can be multiple instances for the same name if they were quickly started and stopped. Handle that case when requesting status of master. [fschulze]

1.1.0 - 2014-10-27

- Print status of all `ec2` instances when requesting status of master. [fschulze]

1.0.0 - 2014-07-19

- Added documentation. [fschulze]

1.0b4 - 2014-07-15

- Fix confusion between instance from ploy and ec2 instance. [fschulze]

1.0b3 - 2014-07-08

- Moved `snapshots` list command here after ploy enabled it. [fschulze]
- Renamed `mr.awesome` to `ploy` and `mr.awesome.ec2` to `ploy_ec2`. [fschulze]

1.0b2 - 2014-05-15

- Renamed `conn` to `ec2_conn` to allow reuse of `conn` from `BaseInstance`. [fschulze]
- Moved `setuptools-git` from `setup.py` to `.travis.yml`, it's only needed for releases and testing. [fschulze]

1.0b1 - 2014-03-24

- Initial release [fschulze]

2.3 ploy_ezjail

2.3.1 Overview

The `ploy_ezjail` plugin provides integration of `ezjail` with `ploy` to manage `FreeBSD` jails.

2.3.2 Installation

`ploy_ezjail` is best installed with `easy_install`, `pip` or with `zc.recipe.egg` in a buildout.

2.3.3 Masters

To use `ploy_ezjail` you need a host running `FreeBSD` on which you want to manage jails.

You declare a master with `[ez-master:masterid]` where `masterid` is the name you want to use for this master. Now you can either add options like for a plain ploy instance, or you can use the `instance` option to refer to another instance from your config like this:

```
[ez-master:master1]
host = myhost.example.com

[plain-instance:foohost]
host = foohost.example.com

[ez-master:master2]
instance = foohost
```

The latter is most useful in conjunction with other ploy backend plugins, as it allows you to easily switch between provisioners, i.e. to have an `ez-master` provisioned on `VirtualBox` during development and on a `plain` instance in production.

Options

debug-commands If set to `yes`, the commands executed on the host are echoed locally.

instance The instance to use as host for this master. If empty, the local machine is used without an ssh connection.

ezjail-admin Path to the `ezjail-admin` script on the host. Defaults to `/usr/local/bin/ezjail-admin`.

sudo Use `sudo` to run commands on the host.

2.3.4 Instances

At the moment all jails will be created using ZFS (the `-c zfs` option of `ezjail-admin`), so the host needs to be setup accordingly.

Options

ip The ip address to use for the jail.

This can either be a single IPv4 address:

```
ip = 10.0.0.3
```

or any number of IPv4 and IPv6 addresses attached to different devices:

```
ip = lo1|10.0.0.3,vtnet0|2a03:b0c0:3:d0::3a4d:c002
```

The latter format is `ezjail`'s own. **Required**

flavour The **flavour** to use for this jail. This is explained in the [ezjail docs](#).

ezjail-name The **name** to use for the jail. By default the id of the instance is used.

mounts Additional mount points for the jail. You can specify one mount point per line. The format is:

```
src=SRC dst=DST [ro=true] [create=true]
```

The `src` is the path on the host, `dst` is the path inside the jail.

If `ro` is set to `true`, then the mount is read only.

When `create` is enabled, then the `src` path is created with `mkdir -p`. The `dst` path is always created inside the jail with `mkdir -p`.

You can reference [ZFS sections](#) inside `src` with `{zfs[name]}` where `name` is the `ez-zfs` section name. You can use the name of the jail instance with `{name}` in both `src` and `dst`. Examples:

```
src=/foo dst=/foo
src={zfs[backup]} dst=/bak
src={zfs[data]}/{name} dst=/mnt/data create=true
src={zfs[static]} dst=/mnt/static ro=true
```

no-terminate If set to `yes`, the jail can't be terminated via `ploy` until the setting is changed to `no` or removed entirely.

startup_script Path to a local script (relative to the location of the configuration file) which will be run inside the jail right after creation and first start of the jail.

rc_require String that indicates which other jails this jail requires to start up, effectively allowing you to define the startup order of jails. See `rcorder(8)` for more details. This value is written upon each startup of the jail not just when it is created initially, so to have changes take effect, it's sufficient to restart it. **Optional**

rc_provide String that indicates what this jail provides. `ezjail` itself always sets its jails to provide `standard_ezjail` to which `ploy_ezjail` adds the name of the jail. IOW if you simply want to build a startup order using the names of the jails, you will not need to set this value. If you want this jail to provide any additional values, set them here. This value is written upon each startup of the jail not just when it is created initially, so to have changes take effect, it's sufficient to restart it. **Optional**

2.3.5 ZFS sections

You can specify ZFS filesystems via `[ez-zfs:name]` sections. This is used in mounts of jails to get the mountpoint and verify that the path exists and is it's own ZFS filesystem. You can also create new ZFS filesystems with the `create` option.

Options

create If set to `yes`, the filesystem is created when first used.

path Specifies the path of this filesystem. This is not the mountpoint, but the ZFS path. You can reference other ZFS sections with `{zfs[name][path]}`. The name is the name of the referenced ZFS section. The `[path]` at the end is mandatory, as otherwise you would get the mountpoint of the referenced ZFS section. Examples:

```
[ez-zfs:data]
path = tank/data

[ez-zfs:shared]
path = {zfs[data][path]}/shared

[ez-zfs:jails]
path = {zfs[data][path]}/jails

[ez-zfs:backup]
create = true
path = tank/backup
```

2.3.6 Changelog

1.5.2 - Unreleased

1.5.1 - 2018-02-03

- Fix startup script for FreeBSD 11.x by replacing dots in it's name with underscores. [fschulze]

1.5.0 - 2017-12-17

- Add `get_fingerprints` to support host keys with all key types. [fschulze]
- Fix jail host `status` command for various cases. [fschulze]
- Output `stdout` in addition to `stderr` on various errors. This let's one see additional debug info when jails don't start. [fschulze]

1.4.0 - 2015-10-16

- Allow setting the jail name via `ezjail-name` instead of using the instance id as the default. [fschulze]

1.3.0 - 2015-09-03

- Improved error handling with useful error messages instead of tracebacks. [fschulze]
- Allow setting startup order of jails. [tomster]

1.2.0 - 2015-03-05

- Use new `Executor` helper from ploy 1.2.0 which handles ssh agent forwarding. [fschulze]
- Enable “local mode” where if the `instance` option is empty all commands are executed locally. [fschulze]

1.1.0 - 2014-10-27

- Print status of all jails when requesting status of master. [fschulze]
- Check jail status before trying to connect. [fschulze]
- Use new helper in ploy 1.0.2 to setup `proxycommand`. [fschulze]

1.0.0 - 2014-07-19

- Added documentation. [fschulze]

1.0b9 - 2014-07-08

- Packaging and test fixes. [fschulze]

1.0b8 - 2014-07-04

- Python 3 compatibility. [fschulze]
- Renamed `mr.awesome` to `ploy` and `mr.awesome.ezjail` to `ploy_ezjail`. [fschulze]

1.0b7 - 2014-06-16

- Provide default values for `proxyhost` and `proxycommand` options. [fschulze]
- Merge config of `ez-master` with the instance it's using. [fschulze]

1.0b6 - 2014-06-11

- Pass changes of proxy instance config on to the proxied instance config. [fschulze]

1.0b5 - 2014-06-10

- Forcefully destroy jail. Together with ezjail 3.4.1 this solves the issue that sometimes the ZFS filesystem wasn't removed and the jail couldn't be started without manual intervention. [fschulze]

1.0b4 - 2014-05-22

- Clear out massagers after copying the config for the proxy instance to prevent conflicts when the proxy instance is created. [fschulze]

1.0b3 - 2014-05-21

- Fixes to make [instance:...] using an ez-master work. [fschulze]

1.0b2 - 2014-05-15

- Added instance option to ez-master section to use another instance as the jail host. [fschulze, tomster]
- Moved setuptools-git from setup.py to .travis.yml, it's only needed for releases and testing. [fschulze]

1.0b1 - 2014-03-24

- Initial release [fschulze]

2.4 ploy_fabric

2.4.1 Overview

The ploy_fabric plugin provides integration of Fabric with ploy.

2.4.2 Installation

ploy_fabric is best installed with easy_install, pip or with zc.recipe.egg in a buildout.

Once installed, it's functionality is immediately usable with ploy.

2.4.3 Commands

The plugin adds the following commands to ploy.

do Runs a Fabric task with simplified syntax for arguments. You can just put positional arguments on the command line behind the task name. For keyword arguments use the name=value syntax. For example:

```
ploy do something arg key=value
```

fab Runs a Fabric task and passes on command line options to Fabric. This basically reflects the fab script of Fabric.

2.4.4 Options

Instances only get the new `fabfile` option to specify which file to look in for tasks. The location is relative to `ploy.conf`.

2.4.5 Instance methods

For the Python side, each instance gains the `do(task, *args, **kwargs)` method. The `task` argument is the name of a task from the Fabric script which should be run. The remaining arguments are passed on to that task.

Another helper added to each instance is a context manager accessible via the `fabric` attribute on instances. With that you can switch to a new ssh connection with a different user in your Fabric tasks:

```
from fabric.api import env, run

def sometask():
    run("whoami") # prints the default user (root)
    with env.instance.fabric(user='foo'):
        run("whoami") # prints 'foo' if the connection worked
    run("whoami") # prints the default user (root)
```

All changes to the Fabric environment are reverted when the context manager exits.

2.4.6 Fabric task decorator

With `ploy_fabric.context` you can decorate a task to use a specific user with a separate connection. All changes to the Fabric environment are reverted when the context manager exits. This is useful if you want to run a task from inside another task.

```
from fabric.api import env, run
from ploy_fabric import context

@context # always run with the default user
def sometask():
    run("whoami") # prints the default user (root)

@context(user=None) # always run with the default user (alternate syntax)
def someothertask():
    env.forward_agent = True
    run("whoami") # prints the default user (root)

@context(user='foo') # always run as foo user
def anothersetask():
    env.forward_agent = False
    run("whoami") # prints the default user (user)
    someothertask()
    assert env.forward_agent == False
```

2.4.7 Fabric environment

The Fabric environment has the following settings by default.

reject_unknown_hosts Always set to `True`, ssh connections are handled by this plugin and ploy.

disable_known_hosts Always set to `True`, handled by ploy.

host_string The **unique id** of the current instance, only manipulate if you know what you do!

known_hosts Path to the `known_hosts` file managed by ploy.

instances Dictionary allowing access to the other instances to get variables or call methods on.

instance The current instance to access variables from the `config` attribute or other things and methods.

config_base The directory of `ploy.conf`.

Any option of the instance starting with `fabric-` is stripped of the `fabric-` prefix and overwrites settings in the environment with that name.

2.4.8 Changelog

1.1.1 - Unreleased

1.1.0 - 2014-10-27

- Require Fabric \geq 1.4.0 and vastly simplify the necessary patching. [fschulze]
- Close all newly opened connections after a Fabric call. [fschulze]
- Add context manager and decorator to easily switch fabric connections. [fschulze]

1.0.0 - 2014-07-19

- Added documentation. [fschulze]

1.0b6 - 2014-07-15

- Allow overwriting of fabric env from config with options prefixed by `fabric-`, i.e. `fabric-shell = /bin/sh -c`. [fschulze]

1.0b5 - 2014-07-08

- Packaging and test fixes. [fschulze]
- Fix task listing for `do` command. [fschulze]

1.0b4 - 2014-07-04

- Use unique id for host string to avoid issues. [fschulze]
- Added `fab` command which is just a wrapper for Fabric with all it's options and reworked `do` command into a simple version to just run a task. [fschulze]
- Renamed `mr.awesome` to `ploy` and `mr.awesome.fabric` to `ploy_fabric`. [fschulze]

1.0b3 - 2014-06-09

- When depending on Fabric, skip 1.8.3 which added a version pin on paramiko. [fschulze]
- Only add Fabric to `install_requires` if it can't be imported. That way we don't get problems if it's already installed as a system packages or in a virtualenv. [fschulze]

1.0b2 - 2014-05-15

- Register `fabfile` massager for all instances. [fschulze]
- Use context manager for output filtering and filter in `do` helper. [fschulze]
- Moved `setuptools-git` from `setup.py` to `.travis.yml`, it's only needed for releases and testing. [fschulze]

1.0b1 - 2014-03-24

- Initial release [fschulze]

2.5 ploy_openvz

2.5.1 Changelog

1.0b3 - unreleased

- Renamed `mr.awesome` to `ploy` and `mr.awesome.openvz` to `ploy_openvz`. [fschulze]

1.0b2 - 2014-05-15

- Moved `setuptools-git` from `setup.py` to `.travis.yml`, it's only needed for releases and testing. [fschulze]

1.0b1 - 2014-03-24

- Initial release [fschulze]

2.6 ploy_virtualbox

2.6.1 Overview

The `ploy_virtualbox` plugin provides integration of `VirtualBox` with `ploy`.

2.6.2 Installation

`ploy_virtualbox` is best installed with `easy_install`, `pip` or with `zc.recipe.egg` in a buildout.

2.6.3 Master

The default master for `ploy_virtualbox` is `virtualbox` and has the following options:

headless Whether to start instances in headless mode by default. If not set, the system default is used.

use-acpi-powerbutton Whether to use acpi power off by default when stopping instances. If not set, the system default is used.

basefolder The basefolder for VirtualBox data. If not set, the VirtualBox default is used. This varies depending on the OS that ploy is running in. When not provided as absolute path, then it's relative to `ploy.conf`.

instance Name of instance to use to execute VirtualBox commands instead of the default local machine.

Example:

```
[vb-master:virtualbox]
headless = true
```

2.6.4 Instances

headless Whether to start this instance in headless mode. If not set, the setting of the master is used.

use-acpi-powerbutton Whether to use acpi power off for this instance when stopping. If not set, the setting of the master is used.

basefolder The basefolder for this instances VirtualBox data. If not set, the setting of the master is used. When not provided as absolute path, then it's relative to `ploy.conf`.

no-terminate If set to `yes`, the instance can't be terminated via ploy until the setting is changed to `no` or removed entirely.

Any option starting with `vm-` is stripped of the `vm-` prefix and passed on to `VBoxManage`. Almost all of these options are passed as is. The following options are handled differently or have some convenience added:

storage One storage definition per line. See `VBoxManage storageattach` documentation for details.

If you don't specify the `type`, then `hdd` is used by default.

You don't have to specify the `port`, if not set the line number starting at zero is used.

If you don't set `--storagectl` then `sata` is used as default and if that controller doesn't exist it's created automatically.

If `medium` references a local file that path will be passed directly to `VBoxManage storageattach`.

If it takes the form `vb-disk:NAME` which refers to a *Disk section* called `NAME` that will be used instead.

If it takes the form of an URL, the filename of that URL is assumed to be located at `~/ploy/downloads/` (this default can be overridden in the `[global]` section of the configuration file with an entry `download_dir`). If the file does not exist it will be downloaded.

When using the URL notation it is strongly encouraged to also provide a checksum using the `--medium_sha1` key (currently only SHA1 is supported).

Example for using a local ISO image as DVD drive:

```
storage =
  --type dvddrive --medium ~/downloads/archives/mfsbsd-se-9.2-RELEASE-amd64.iso
  --medium vb-disk:boot
```

Example for referencing an external URL:

```
storage =
  --type dvddrive --medium http://mfsbsd.vx.sk/files/iso/11/amd64/mfsbsd-se-11.
  ↪1-RELEASE-amd64.iso --medium_sha1 2fd80caf57c29d62859bccfa3b8ec7b5b244406e
  --medium vb-disk:boot
```

hostonlyadapter If there is a matching *Host only interface section*, then that is evaluated.

2.6.5 Disk sections

These section allow you to describe how disks should be created.

You can set the `size`, `variant` and `format` options as described in the `VBoxManage createhd` documentation.

The `filename` option allows you to set a filename for the disk, the extension is automatically added based on the `format` option. If you use a relative path, then it's base is the `basefolder` setting of the instance.

When the `delete` option is set to `false`, the disk is not deleted when the instance using it is terminated. The default is to delete the disk upon instance termination.

Example:

```
[vb-disk:boot]
size = 102400
```

2.6.6 Host only interface sections

If you want to use host only network interfaces, then this allows you to make sure your settings are as expected and the interface exists. For now only the `ip` option is supported. See `VBoxManage hostonlyif` documentation for details.

Example:

```
[vb-hostonlyif:vboxnet0]
ip = 192.168.56.1
```

DHCP

If a `vb-dhcpserver` section with the same name exists, then it is checked and if needed configured as well. See `VBoxManage hostonlyif` documentation for details.

Example:

```
[vb-dhcpserver:vboxnet0]
ip = 192.168.56.2
netmask = 255.255.255.0
lowerip = 192.168.56.100
upperip = 192.168.56.254
```

The combination of `vb-hostonlyif` with `vb-dhcpserver` allows to configure a hostonly network with a deterministic IP address. In the above example you could configure an instance with a static IP address of `192.168.56.99` which would be addressable from the host. The important part is to chose an address that is *within* the DHCP server network but *outside* its DHCP pool, which is defined by `lowerip` and `upperip` respectively.

2.6.7 SSH

Depending on the setup we can't get the IP address or host name automatically.

Unfortunately VirtualBox doesn't provide a way to see which instance got which IP address from its own DHCP servers for example.

If you know which host name or ip address your instance will have, then set the `host` or `ip` option as explained above in the `hostonly` section.

As a workaround you can also setup a NAT port forwarding like this:

```
vm-nic2 = nat
vm-natpf2 = ssh,tcp,,47022,,22
```

For this case `ply_virtualbox` knows how to get the port and uses it for SSH access via localhost.

If you install the VirtualBox guest additions in your instance, then the `status` command can show you the current IP address of the instance.

2.6.8 Example config

```
[vb-master:virtualbox]
# use-acpi-powerbutton = false

[vb-disk:boot]
size = 102400

[vb-hostonlyif:vboxnet0]
ip = 192.168.56.1

[vb-dhcpserver:vboxnet0]
ip = 192.168.56.2
netmask = 255.255.255.0
lowerip = 192.168.56.100
upperip = 192.168.56.254

[vb-instance:foo]
# headless = true
vm-ostype = FreeBSD_64
vm-memory = 512
vm-accelerate3d = off
vm-acpi = on
vm-rtcuseutc = on
vm-boot1 = disk
vm-boot2 = dvd
vm-nic1 = hostonly
vm-hostonlyadapter1 = vboxnet0
vm-nic2 = nat
vm-natpf2 = ssh,tcp,,47022,,22
storage =
  --type dvddrive --medium ~/downloads/archives/mfsbsd-se-9.2-RELEASE-amd64.iso
  --medium vb-disk:boot
```

2.6.9 Changelog

1.2.1 - Unreleased

1.2.0 - 2015-09-03

- Add `delete` option to disks. If set to `false`, the disk is kept in place upon instance termination and not deleted as per default. [fschulze]
- Ask user whether to continue or not when checksum of downloaded iso image doesn't match. [tomster]

1.1.0 - 2015-01-20

- Log info when starting an instance. [fschulze]
- Handle instances in `aborted` state. [fschulze]
- Print error output of commands on failures. [fschulze]
- Use new helper in ploy 1.0.2 to setup proxycommand. [fschulze]
- Added possibility to specify a remote instance to use for a virtualbox master. [fschulze]
- Added ability to reference disk images via external URL for virtualbox instances storage. [tomster]

1.0.0 - 2014-07-19

- Added documentation. [fschulze]
- Renamed `vb-master` to `virtualbox`, so the uids of instances are nicer. [fschulze]
- Enable DHCP server when creating or modifying it. [fschulze]

1.0b4 - 2014-07-15

- Verify and if possible create host only interfaces and dhcpservers. [fschulze]
- Add support for instances that have manually been put into `saved` state. [fschulze]

1.0b3 - 2014-07-08

- Packaging and test fixes. [fschulze]

1.0b2 - 2014-07-04

- Python 3 compatibility. [fschulze]
- Renamed `mr.awesome` to `ploy` and `mr.awesome.virtualbox` to `ploy_virtualbox`. [fschulze]

1.0b1 - 2014-06-16

- Initial release [fschulze]

CHAPTER 3

Indices and tables

- `genindex`
- `modindex`
- `search`