deploymachine Documentation

Release 0.3

Thomas Schreiber

December 16, 2011

CONTENTS

Contents:

CHAPTER

ONE

PREFACE

Deploymchine is:

- Tiny wrapper around configuration management tools such as chef, kokki, and puppet.
- Deployment framework for launching and maintaining websites.
- Slew of fabric commands for common tasks.

Documentation about the usage and installation of the deploymachine can be found at http://deploymachine.readthedocs.org

The source code and issue tracker can be found on GitHub. https://github.com/rizumu/deploymachine

CHAPTER

TWO

MOTIVATION

Launching a high performance web application has community supported best practices. Deploymachine attempts to put these best practices into code, simplifying launching and growing the application. Deploymachine follows the standards set forth by the Slicehost provisioning documentation, the Django Deployment Workshop, and the Arch Linux wiki.

- SLICEHOST PROVISIONING DOCS: http://articles.slicehost.com/ubuntu-10
- DJANGO DEPLOYMENT WORKSHOP: https://github.com/jacobian/django-deployment-workshop/
- ARCH LINUX WIKI: https://wiki.archlinux.org/

Each component and setting of the server stack really should be understood inside out. However, sometimes it is prudent to install now and learn later. Deploymachine attempts to help speed things along by means of sensible defaults. Designed with server scalability in mind, the idea is to start with one server and add or remove servers as needed.

Media files and database backups are stored in Cloudfiles or S3 and the application code and configuration is kept under version control. Therefore all the Deploymachine's servers can be killed and rebuilt in a matter of minutes, with no fear of data loss. Keep in mind that killing the database server safely will require some preparation and planning to ensure this.

CHAPTER

THREE

SERVER TYPES

Deploymachine supports three server types:

- `load balancer`
- `application node`
- `database server`

The current idea is to have one loadbalancer, one dbserver, and potentially multiple appnodes. However, in the case of a small app, it is probably best to start with a dbappblancer first. Or start with an appbalancer and a dbserver as I have done. One the app demands, split the loadbalncer off and scale each type individually.

Note: FUTURE FEATURE

A large app may need to add additional types of servers to the process such as a caching server, logging server, or a email server. A high traffic site will possibly need multiple database servers.

Deploymachine is intent on supporting the most common and practical cases, but not the kitchen sink.

CHAPTER

FOUR

INSTALLATION

Deploymachine runs locally on OSX or GNU/Linux.

Clone the deploymachine to an easily accesible folder on your local machine. Replacing `username` with your local username, the recommended place to clone is:

- `/home/username/www/lib/deploymachine/` for GNU/Linux
- `/Users/username/www/lib/deploymachine/` for OSX

Copy the example files and customize for your project. A suggestion is to instead store these private files a private git repository, and create symlinks to them. These files are set to be ignored in the deploymachine's .gitignore:

```
$ cp fabfile_local.dist.py fabfile_local.py
$ cp deploymachine_settings.py.dist deploymachine_settings.py
$ cp kokki-config.dist.j2 kokki-config.j2
```

You must also set some environment variables for your interactive shell. For example, add the following to your `~/.bashrc` and edit accordingly:

```
export OPENSTACK_COMPUTE_USERNAME=""  # your rackspace username
export OPENSTACK_COMPUTE_APIKEY=""  # your rackspace apikey
export VIRTUALENVS_LOCAL_ROOT=""  # typically /home/username/.virtualenvs/
export SITES_LOCAL_ROOT=""  # try /home/username/www/
```

CHAPTER

FIVE

EXTERNAL LIBRARIES

Deploymachine can be used with the following 3rd party libraries to enhance the possibilities.

CHAPTER

SIX

CLOUD PROVIDERS

Deploymachine supports both the Rackspace Openstack API and the Amazon Boto API via the bootmachine project.

CHAPTER

SEVEN

DNSIMPLE

Deploymachine harnesses the dnsimple python API for directing our domain names to our loadbalancer's ip, even after a kill/create. @@@TODO

CHAPTER

EIGHT

NEWRELIC

For Python application monitoring.

CHAPTER

NINE

VIRTUALBOX

There is a fabfile method to boot virtualbox instances. This is useful for testing in a developement environment.

CHAPTER

TEN

FIRST DEPLOYMENT

After private/personal configuration is complete, the general idea is as follows:

```
$ fab dbserver launch
$ fab loadbalancer launch
$ fab appnode launch
```

Warning: Currently untested with all possible combinations and quantities of server types.

CHAPTER

ELEVEN

SCALING UP AND DOWN

First start with a combined dbappbalancer, then when necessary split into an appbalancer and dbserver, followed by loadbalancer, appnode, dbserver.

Once application growth warrants more power, increase/decrease the number of appnodes, add a cachenode, lognode, or tasknode.

Growth beyond this can only suggest that the application is bringing in revenue. The good news is that the deploymachine has already prepared the application for further expansion by neatly separating repsonsibility between services.

Warning:

- Currently implemented are loadbalancer, appnode, dbserver and appbalancer.
- Not yet implemented are dbappbalacner, cachenode, lognode or tasknode.
- The splitting process is not yet implemented, stay tuned.

CHAPTER

TWELVE

ONGOING DEPLOYMENTS

First use the bootmachine to launch and provision your new machine:

```
$ fab openstack.boot appnode2
$ fab root provision
```

Next to add a new app node to the mix follow this process:

```
$ fab appnode launch
$ fab kokki:loadbalancer
```

Deploymachine provides a Fabric API for maintaining your application. Commands for git, pip, supervisor, nginx, django and more are included or type `fab -l`. Take a peek inside the `contrib` folder for all the available options, or help extend the existing library. Domain specific commands can be imported from your personal in `fab_local.py`. An example `fab_local.example.py` is included.

CHAPTER

THIRTEEN

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*