nuka Documentation

Bearstech

Sep 19, 2018

Contents

1 Quickstart 3

2 Index 5

3 Indices and tables 17

Because ops can dev.

nuka is a provisioning tool focused on performance. It massively uses Asyncio and SSH. It is compatible with docker vagrant and apache-libcloud.

CHAPTER 1

Quickstart

Install nuka (See Installation for detailled steps):

```
$ pip install "nuka[full]"
```

Then start a script:

```
#!/usr/bin/env python3.5
import nuka
from nuka.hosts import DockerContainer
from nuka.tasks import (shell, file)

# setup a docker container using the default image
host = DockerContainer('mycontainer')


async def do_something(host):

    # we just echoing something using the shell.command task
    await shell.command(['echo', 'it works'], host=host)

    # if no host is provided, then a var named `host` is searched
    # from the stack. Mean that this will works to
    await shell.command(['echo', 'it works too'])


async def do_something_else(host):

    # log /etc/resolv.conf content
    res = await file.cat('/etc/resolv.conf')
    host.log.info(res.content)


# those coroutines will run in parallell
nuka.run(
```

(continues on next page)

(continued from previous page)

```
    do_something(host),
    do_something_else(host),
)
```

Run it using:

```
$ chmod +x your_file.py
$ ./your_file.py -v
```

The first run will be slow because we have to pull the docker image. The next run will take approximately 1s.

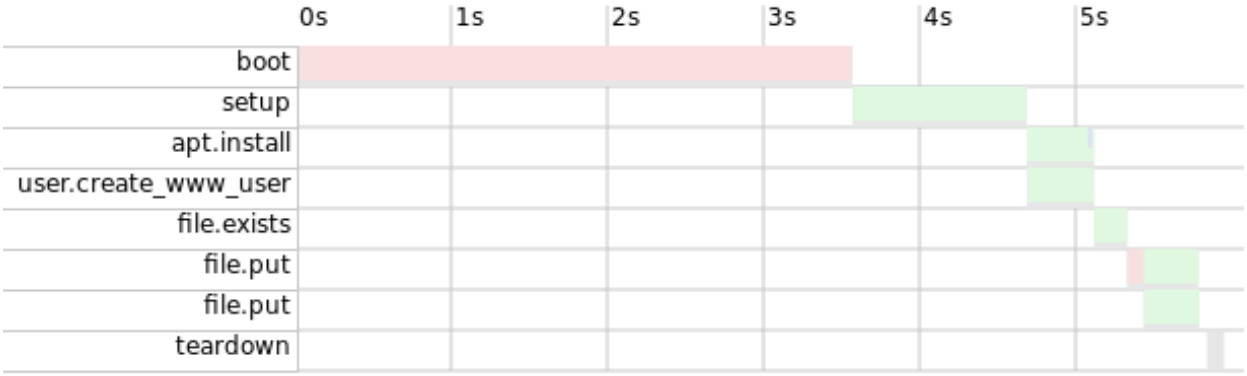
Get some help:

```
$ ./your_file.py -h
```

Look at the generated gantt of your deployement:

```
$ firefox .nuka/reports/your_file_gantt.html
```

You'll get a dynamic report like this screenshot:



CHAPTER 2

Index

2.1 Installation

You'll need *python3.5+*.

2.1.1 Install python3.5 using pyenv

You can install python3.5 using pyenv

You may need a few packages:

```
$ sudo apt-get install build-essential libssl-dev libreadline-dev libbz2-dev python-
↪virtualenv
```

Install pyenv:

```
$ git clone https://github.com/yyuu/pyenv.git ~/.pyenv
```

Then install python3.5 (or python3.6):

```
$ ~/pyenv/bin/pyenv install 3.5.3
```

2.1.2 Install nuka in a virtualenv

Basic install:

```
$ virtualenv -p $(which python3.5) myproject
$ cd myproject
$ source bin/activate
```

Check that your virtualenv use the correct version:

```
$ bin/python --version
Python 3.5.3
```

Install nuka in your virtualenv using pip:

```
$ pip install nuka
```

If you're planning to use libcloud or docker then you'll need some extra dependencies. Replace the last line by:

```
$ pip install "nuka[full]"
```

2.1.3 Installing from source

```
$ pip install -e "git+https://github.com/bearstech/nuka.git#egg=nuka[full]"
```

2.1.4 Installing docker

You should have a recent docker version. See Install docker

2.2 API

2.2.1 nuka.hosts

2.2.2 nuka

2.2.3 nuka.utils

2.3 Builtin tasks

2.3.1 nuka.tasks.apache

nuka.tasks.apache.a2disconf

nuka.tasks.apache.a2dismod

nuka.tasks.apache.a2dissite

nuka.tasks.apache.a2enconf

nuka.tasks.apache.a2enmod

nuka.tasks.apache.a2ensite

2.3.2 nuka.tasks.apt

nuka.tasks.apt.debconf_set_selections

Example:

```
res = await apt.debconf_set_selections(
    [('adduser', 'adduser/homedir-permission', 'true')]
)
assert bool(res)
```

nuka.tasks.apt.install

Example:

```
res = await apt.install(['python'])
assert bool(res)
```

nuka.tasks.apt.list

nuka.tasks.apt.search

nuka.tasks.apt.source

Example:

```
if 'wheezie' in host.hostname:
    n = 'wheezie'
elif 'jessie' in host.hostname:
    n = 'jessie'
else:
    n = 'stretch'
src = 'deb http://apt.dockerproject.org/repo/ debian-{0} main'.format(n)
res = await apt.source(
    name='docker',
    key='https://yum.dockerproject.org/gpg',
    src=src,
)
assert bool(res)
src = 'deb https://deb.bearstech.com/debian {0}-bearstech main'.format(n)
res = await apt.source(
    name='bearstech',
    key='https://deb.bearstech.com/bearstech-archive.gpg',
    src=src,
)
assert bool(res)
```

nuka.tasks.apt.update

Example:

```
res = await apt.update(cache=3600)
assert bool(res)
```

nuka.tasks.apt.upgrade

2.3.3 nuka.tasks.archive

nuka.tasks.archive.untar

2.3.4 nuka.tasks.file

nuka.tasks.file.cat

Example:

```
res = await file.cat('/etc/default/useradd')
assert res.content
```

nuka.tasks.file.chmod

nuka.tasks.file.chown

nuka.tasks.file.exists

Example:

```
res = await file.exists('/tmp')
assert bool(res) is True

res = await file.exists('/nope')
assert bool(res) is False
```

nuka.tasks.file.mkdir

Example:

```
if not await file.exists('/tmp/doc'):
    await file.mkdir('/tmp/doc')
```

nuka.tasks.file.mkdirs

nuka.tasks.file.mv

nuka.tasks.file.put

Example:

```
await file.put([
    dict(src='/etc/resolv.conf', dst='/tmp/resolv.conf'),
    dict(src='docs/utils.py', dst='/tmp/utils.py', executable=True),
    # jinja2 template
    dict(src='example.j2', dst='/tmp/xx1', mod='600'),
    # symlink
    dict(linkto='/etc/hosts', dst='/etc/hosts2'),
], ctx=dict(name='example'))
```

nuka.tasks.file.rm

Example:

```
await file.rm('/tmp/doc')
```

nuka.tasks.file.scripts

nuka.tasks.file.update

Example:

```
await file.update(
    dst='/etc/default/useradd',
    replaces=[(r'^\# HOME=/home', 'HOME=/new_home')])
```

2.3.5 nuka.tasks.git

nuka.tasks.git.git

2.3.6 nuka.tasks.http

nuka.tasks.http.fetch

2.3.7 nuka.tasks.mysql

nuka.tasks.mysql.create_db

nuka.tasks.mysql.execute

nuka.tasks.mysql.my_cnf

nuka.tasks.mysql.set_root_password

2.3.8 nuka.tasks.postgresql

nuka.tasks.postgresql.create_db

nuka.tasks.postgresql.psql

2.3.9 nuka.tasks.service

nuka.tasks.service.reload

nuka.tasks.service.restart

Example:

```
await service.restart('rsync')
```

nuka.tasks.service.start

Example:

```
await service.start('rsync')
```

nuka.tasks.service.stop

Example:

```
await service.stop('rsync')
```

2.3.10 nuka.tasks.shell

nuka.tasks.shell.command

nuka.tasks.shell.commands

nuka.tasks.shell.shell

2.3.11 nuka.tasks.user

nuka.tasks.user.authorized_keys

Example:

```
await user.create_user('myuser')
await user.authorized_keys(
    username='myuser', keysfile='~/.ssh/authorized_keys')
```

nuka.tasks.user.create_user

Example:

```
await user.create_user('myuser')
```

nuka.tasks.user.create_www_user

nuka.tasks.user.delete_user

Example:

```
await user.delete_user('myuser')
```

2.3.12 nuka.tasks.virtualenv

nuka.tasks.virtualenv.virtualenv

Example:

```
res = await venv.virtualenv('/tmp/venv')
assert res
```

2.3.13 nuka.tasks.yum

nuka.tasks.yum.install

Example:

```
res = await yum.install(['python'])
assert bool(res)
```

nuka.tasks.yum.update

Example:

```
res = await yum.update(cache=3600)
assert bool(res)
```

2.4 Use your own tasks

nuka's tasks are just python class that inherit from `nuka.task.Task`.

Here is a simple example:

```
# -*- coding: utf-8 -*-
import codecs
from nuka.task import Task


class timezone(Task):

    def __init__(self, tz=None, **kwargs):
        # ensure we have a name to get a better repr() in logs
        kwargs.setdefault('name', tz)
        super(timezone, self).__init__(tz=tz, **kwargs)

    def do(self):
        """do the job: change the timezone file if needed"""
        tz = self.args['tz']
        changed = False
        with codecs.open('/etc/timezone', 'r', 'utf8') as fd:
            current_tz = fd.read().strip()
        if current_tz != tz:
            changed = True
            with codecs.open('/etc/timezone', 'w', 'utf8') as fd:
                current_tz = fd.write(tz + '\n')
        # we must return a dictionary with at least a return code and
        # the change state
        return dict(rc=0, changed=changed)

    def diff(self):
        """generate diff between actual state and task value.
        Implementing this method is not required but recommended"""
        tz = self.args['tz']
        with codecs.open('/etc/timezone', 'r', 'utf8') as fd:
            current_tz = fd.read().strip()
        diff = ''
        if current_tz != tz:
            diff = self.text_diff(current_tz, tz)
        return dict(diff=diff)
```

You must be sure that your code is compatible with the python binaries you use locally and remotely (2.x vs 3.x).

nuka's builtin tasks support python 2.7 and 3.4+

As a good practice your task should be isolated in a tasks package and must only use python's stdlib.

Once it's done, you can use it:

```
# -*- coding: utf-8 -*-
import nuka
from nuka.hosts import DockerContainer

from tasks.timezone import timezone

host = DockerContainer(hostname='debian_jessie')


async def change_timezone(host):
    await timezone(tz='Europe/Paris')


nuka.run(change_timezone(host))
```

2.5 Examples

We have more at: https://github.com/bearstech/nuka/tree/master/examples

2.5.1 Docker container

```
# -*- coding: utf-8 -*-
from nuka.hosts import DockerContainer
from nuka.tasks import shell
import nuka

host = DockerContainer(hostname='debian', image='bearstech/nukai')


async def my_tasks(host):
    await shell.shell('whoami')

nuka.run(my_tasks(host))
```

2.5.2 Docker compose

```
version: '2'
services:
    debian_stretch:
        image: bearstech/nukai:latest
```

```
#!../bin/python
from nuka.hosts import DockerCompose
from nuka.tasks import shell
import nuka

hosts = DockerCompose(project_name='myproject')
nuka.run(hosts.boot())

host = hosts['myproject_debian_stretch_1']
```

(continues on next page)

(continued from previous page)

```
async def my_tasks(host):
    await shell.shell('whoami')

nuka.run(my_tasks(host))
```

2.5.3 Vagrant

You need to run `vagrant up` manually.

```
from nuka.hosts import Vagrant
from nuka.tasks import shell
import nuka

host = Vagrant()


async def my_tasks(host):
    await shell.command('whoami')

nuka.run(my_tasks(host))
```

2.5.4 GCE

```
#!../bin/python
# -*- coding: utf-8 -*-
import nuka
from nuka.hosts import Cloud
from nuka.tasks import shell

nuka.config['gce'] = {
    'driver': '~/.gce/nuka.json',
    'user': 'gawel',
    'create_node_args': {
        'size': 'f1-micro',
        'image': 'debian-8-jessie-v20161215',
        'location': 'europe-west1-d',
        'ex_tags': ['nuka'],
        'ex_disk_auto_delete': True,
        'ex_service_accounts': [{
            'scopes': [
                'https://www.googleapis.com/auth/cloud.useraccounts.readonly',
                'https://www.googleapis.com/auth/devstorage.read_only',
                'https://www.googleapis.com/auth/logging.write',
                'https://www.googleapis.com/auth/monitoring.write',
                'https://www.googleapis.com/auth/service.management.readonly',
                'https://www.googleapis.com/auth/servicecontrol'
             ],
        }]
    },
}

nuka.config['ssh'].update({
```

(continues on next page)

(continued from previous page)

```
    'extra_options': ['-C', '-oStrictHostKeyChecking=no'],
    'keysfile': '~/.ssh/authorized_keys',
})


cloud = Cloud('gce')
host = cloud.get_or_create_node('myhost')


async def my_tasks(host):
    await shell.command(['ls'])

nuka.cli.add_argument('--destroy', action='store_true', default=False)
nuka.cli.parse_args()

if nuka.cli.args.destroy:
    nuka.run(cloud.destroy())
else:
    nuka.run(my_tasks(host))
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search