Look up to the skies and

see

cloud-init

Mariusz Wołoszyn
Fudo Security

Think you know the cloud?

- IaaS
  - Vm
  - (v)CPU, RAM
  - block storage
  - vNet,
- PaaS
  - S3
  - Heroku
- SaaS
  - Kafka
  - Postgres
  - Redis

Compute
Storage
Database
Migration
Networking & Content Delivery
Developer Tools
Management Tools
Security, Identity & Compliance
Analytics
Artificial Intelligence
Mobile Services
Application Services
Messaging
Business Productivity
Desktop & App Streaming
Internet of Things
Game Development

Quiz: Do you know this terms?

- Cinder
- Swift
- AWS Lambda
- Neutron
- Glance
- Elastic Beanstalk
- Azure Functions

Why look up?

FACE IT:

YOU ARE OLD! REALLY OLD!

Installing an OS is ancient!

Now you just run “services”.

Everything is just a service, everything exists as an “aaS”.

< 404 Title not found >

- Automation
- Automation
- Automation
- Automation
- Automation
- Automation
- Automation
- Automation
- Automation
- Automation

- DevOps
- Continuous Integration
- Continuous delivery
- Infrastructure as Code

- It’s easier
  - docker run
  - vagrant up
- It’s ergonomic
  - eliminate repeated tasks

Ever wondered how cloud instance has your password or ssh keys already setup?

What we can do?

- Spawn full stack of applications
  - LAMP
  - WP/Joomla
- Configure your on-demand box
  - Vpn
  - Seedbox
- Configure clusters or multi-node architectures
  - Elastic-search
  - with logstash
  - with kibana
  - with rev-proxy...
- Harden environments
- Scale applications on demand
- There are recipes and tools that can do virtually anything.

What is clout-init?

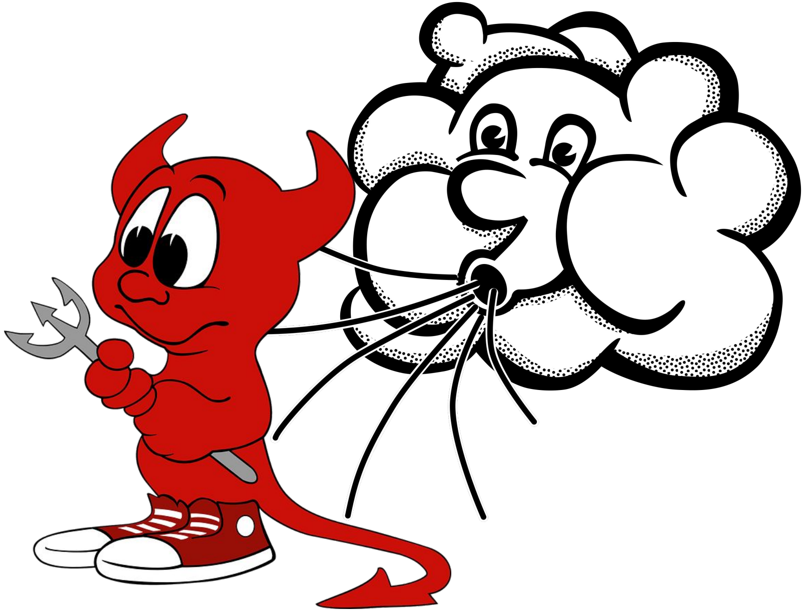
A de-facto standard for configuring cloud instances.

Not only cloud, can configure physical too (for example MaaS).

Understand the way the “cloud provider” tries to configure the machine and applies modification to provisioned OS.

There are MANY ways to communicate configuration to freshly installed OS.

Availability



- Ubuntu
- Fedora
- Debian
- RHEL
- CentOS
- *and more…*
- FreeBSD? (more on that later)

For most Linux distributions there are official builds of so called cloud-images with cloud-init preloaded:
https://cloud-images.ubuntu.com/
https://cloud.centos.org/centos/7/images/
https://cdimage.debian.org/cdimage/openstack/
https://alt.fedoraproject.org/cloud/

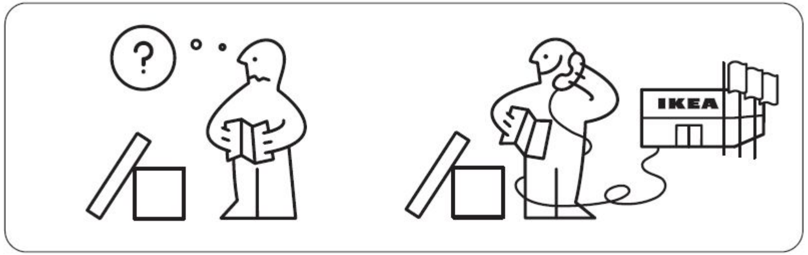
Low level what

Meta-data

- Setting a default locale
- Setting an instance hostname
- Generating instance SSH private keys
- Adding SSH keys to a user’s .ssh/authorized_keys so they can log in
- Setting up ephemeral mount points
- Configuring network devices

User-data:

- Running custom scripts
- Putting custom files/data



Other operations

- Growing fs
  - Cloud image is smallest possible.
  - Client configures an instance and chooses disk size.
  - Disk backing file is “resized” to match user requirements.
  - Cloud-init post boot grows the fs at first run.
- Creating partitions and fs
- Other ops performed by various modules:
  - Ntp
  - Phone home
  - Syslog
  - Timezone
  - Mount
  - Runcmd
  - ...

Data sources

- NoCloud (literally)
- vSphere
- Azure
- CloudStack
- DigitalOcean
- Amazon EC2
- OpenNebula
- OpenStack
- Google Compute Engine
- … many more

Example sources

- Apache CloudStack
  - http://10.1.1.1/latest/user-data
    http://10.1.1.1/latest/meta-data
    http://10.1.1.1/latest/meta-data/{metadata type}
- Digital Ocean,
  - HTTP over the link local address 169.254.169.254
  - eg. http://169.254.169.254/metadata/v1/
- AWS EC2
  - http://169.254.169.254/**2009-04-04**/meta-data/
- vSphere
  - Injected into the VM as an ISO via the cdrom.
- Open Nebula
  - contextualized (parametrized) by CD-ROM image

- NoCloud
  - You can provide meta-data and user-data to a local vm boot via files on a vfat or iso9660 filesystem. The filesystem volume label must be cidata.
  - Alternatively, you can provide meta-data via kernel command line or SMBIOS “serial number” option.
  - e.g. you can pass this option to QEMU:

    ```
    -smbios
    type=1,serial=ds=nocloud-net;s=http://10.10.0.1:8000/
    ```

    to cause NoCloud to fetch the full meta-data from http://10.10.0.1:8000/meta-data after the network initialization is complete.

https://cloudinit.readthedocs.io/en/latest/topics/datasources/nocloud.html

p.s. cloud-init doc is a real mess

Example configuration

```
$ cat user-data
#cloud-config
password: qpqp01
chpasswd: { expire: False }
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDGLRAZuH3SIVjZo/kLlKkEgSS2FPvVUgOfBK6izIyOyuH9Fueb69lQKS7F+BU052xbeLnHhfrdx2nXxwbE
SLt9bHahdqtZDi/5lvaM2KtvtK69+MRLDUphsQp1IQpWd8UJxQNbjV/thgUbIhABsS98bY00DboPGJ2VHZ/IGc+7SC/K3dDwYG9LM2XhFsO3YLup
v+v1Q7Sot0b4a17++YW8J6T99cDzzFPXZdmh88VsDNjE25hhW0IDXhLfTCArGUPTEJoT0sdzFVJIMfhr+O34sTHXG1sWpZUydcSR6lk1Jab8Ag90
RyAtchGT4+nR9b5s8S4q7Bntjvj0zL9FKLLp mwoloszyn@mwoloszyn

$ cat meta-data
local-hostname: cloudevil
```

Example network-cofig

```
$ cat network-config

version: 2
ethernets:
     vnet0:
           match:
                mac_address: "52:54:00:12:34:00"
           set-name: interface0
           addresses:
           - 192.168.1.10/255.255.255.0
           gateway4: 192.168.1.254

           routes:

           - to: 0.0.0.0/0

                via: 10.23.2.1
                metric: 3
```

Using could-init with FreeBSD

There are “some” unofficial cloud images like:
http://images.openstack.nctu.edu.tw/bsd-cloudinit/

Or:
https://mirror.dionipe.net/cloud-image/

and some old and obsolete instructions:
https://pellaeon.github.io/bsd-cloudinit/

and some modern:
https://nanjj.github.io/cloudinit-bsd.html

But it seems the only viable option is to make your own image :(

Installation

All operations were performed on current (as of 2010-02-26) FreeBSD-12.0-RELEASE image obtained from:

https://download.freebsd.org/ftp/releases/VM-IMAGES/12.0-RELEASE/amd64/Latest/FreeBSD-12.0-RELEASE-amd64.qcow2.xz

```
pkg install net/cloud-init
echo 'cloudinit_enable="YES"' >> /etc/rc.conf
echo 'sshd_enable="YES"' >> /etc/rc.conf
```

No, we’re not done yet. :)

p.s. cloud-init depends on sudo

Configure

/usr/local/etc/cloud/cloud.cfg

```
disable_root: false
datasource_list: [ 'NoCloud', 'ConfigDrive', 'Azure', 'OpenStack', 'Ec2']

# System and/or distro specific settings
# (not accessible to handlers/transforms)
System_info:
   # This will affect which distro class gets used
   distro: freebsd
   # Default user name + that default users groups (if added/used)
   default_user:
      name: freebsd
      lock_passwd: false
      gecos: FreeBSD
      groups: [wheel]
      sudo: ["ALL=(ALL) NOPASSWD:ALL"]
      shell: /bin/tcsh
```

Add blkid

```
cat > /usr/bin/blkid << EOF
#!/bin/sh
if [ -e /dev/iso9660/config-2 ]; then
        echo /dev/iso9660/config-2
fi
if [ -e /dev/iso9660/cidata ]; then
        echo /dev/iso9660/cidata
fi
EOF
chmod a+x /usr/bin/blkid
```

Patch this and that

```
cd /usr/local/lib/python2.7/site-packages/cloudinit
patch -p0 << EOF
--- /root/util.py    2019-02-27 10:48:01.478742000 +0000
+++ /usr/local/lib/python2.7/site-packages/cloudinit/util.py    2019-02-27
10:48:05.932087000 +0000@@ -1654,6 +1654,7 @@
         if mtypes is None:
             mtypes = ["auto"]
    elif platsys.endswith("bsd"):
+   sync = False
         if mtypes is None:
             mtypes = ['ufs', 'cd9660', 'vfat']
         for index, mtype in enumerate(mtypes):
EOF
```

Make config image

Manually:

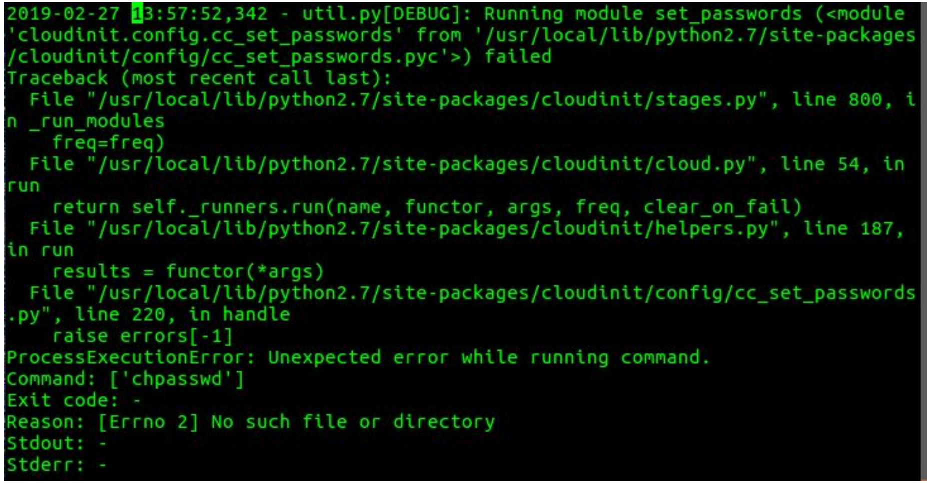
```
## create user-data and meta-data files that will be used
## to modify image on first boot
$ { echo instance-id: iid-local01; echo local-hostname: cloudimg; } > meta-data

$ printf "#cloud-config\npassword: passw0rd\nchpasswd: { expire: False }\nssh_pwauth: True\n" > user-data

## create a disk to attach with some user-data and meta-data
$ genisoimage  -output seed.iso -volid cidata -joliet -rock user-data meta-data
```

Simply using cloud-localds (on Linux ;):

```
cloud-localds config.img my-user-data my-meta-data
```

Problems

- Password changing
    - using just ssh keys won’t give you access via console
    - however I never managed to make password changing work on FreeBSD :/
    - even though the very same config worked on Linux,
    - After some digging I found that set_passwords module is not FreeBSD compatible (in this version)



```
2019-02-27 13:57:52,342 - util.py[DEBUG]: Running module set_passwords (<module
'cloudinit.config.cc_set_passwords' from '/usr/local/lib/python2.7/site-packages
/cloudinit/config/cc_set_passwords.pyc'>) failed
Traceback (most recent call last):
  File "/usr/local/lib/python2.7/site-packages/cloudinit/stages.py", line 800, i
n _run_modules
    freq=freq)
  File "/usr/local/lib/python2.7/site-packages/cloudinit/cloud.py", line 54, in
run
    return self._runners.run(name, functor, args, freq, clear_on_fail)
  File "/usr/local/lib/python2.7/site-packages/cloudinit/helpers.py", line 187,
in run
    results = functor(*args)
  File "/usr/local/lib/python2.7/site-packages/cloudinit/config/cc_set_passwords
.py", line 220, in handle
    raise errors[-1]
ProcessExecutionError: Unexpected error while running command.
Command: ['chpasswd']
Exit code: -
Reason: [Errno 2] No such file or directory
Stdout: -
Stderr: -
```

Documentation

Use docs for your version

```
# cloud-init --version
/usr/local/bin/cloud-init 18.3
```

https://cloudinit.readthedocs.io/en/**18.3**/