Open APIs for Open Minds

How to deploy Spark HA instance using Ansible 2.0 in FIWARE Lab

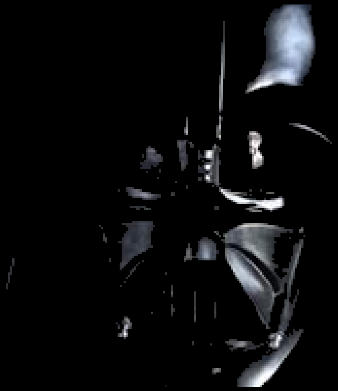
Fernando López Aguilar

Technological Expert. IT and Cloud Architect

fernando.lopezaguilar@telefonica.com

@flopezaguilar

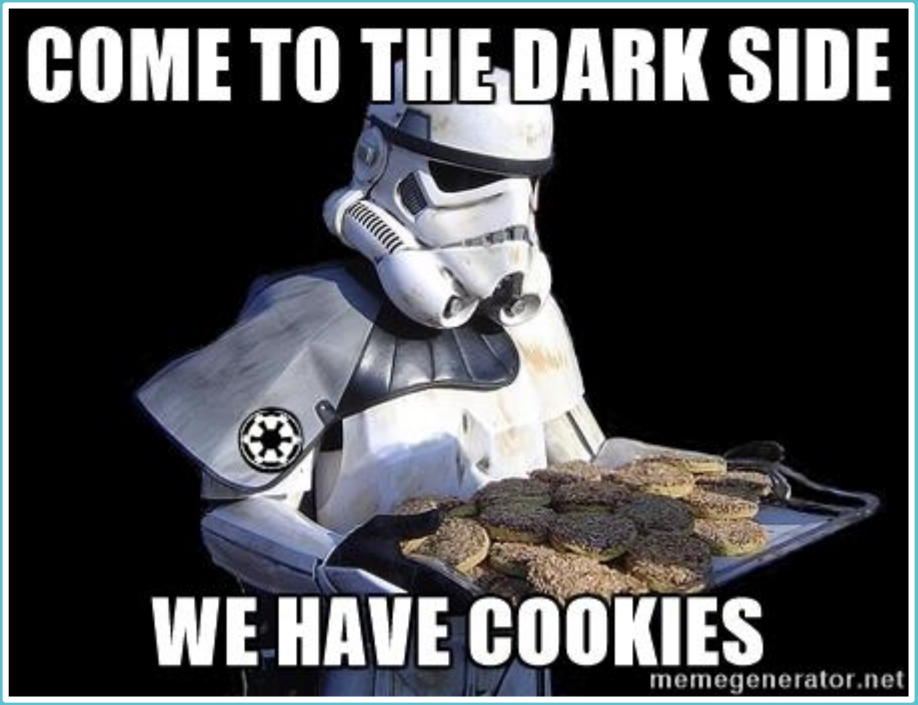
FIWARE



KEEP
CALM
AND
JOIN THE
DARK SIDE

Cookies of the Dark Side

- Use OpenStack locally.
- Use Ansible.
- Use both together.



Come to the Dark Side

We use OpenStackClient

OpenStack client

- Unified shell command structure and a common language to describe operations in OpenStack.
- Remote interaction with any OpenStack environment (if it is open).
- Easy install, just execute: pip install python-openstackclient.
- Easy to use, just execute: $ openstack --help

OpenStackClient hands-on

- Create a new directory (e.g. malaga)
- If you do not have, install pip and virtualenv.
- Create your virtualenv (virtualenv is a tool to create isolated Python environments).
- Activate the isolated python environment.
- Install the OpenStackClient

```
ubuntu@ubuntu:~/malaga$ sudo apt install python-pip
…

ubuntu@ubuntu:~/malaga$ sudo apt install virtualenv
…


ubuntu@ubuntu:~/malaga$ virtualenv env
New python executable in /home/ubuntu/malaga/env/bin/python
Installing setuptools, pip, wheel...done.


ubuntu@ubuntu:~/malaga$ source env/bin/activate


(env) ubuntu@ubuntu:~/malaga$ pip install \
> python-openstackclient
…
```

OpenStack client hands-on

- After the installation you can execute the command shell openstack.
- To obtain information about the openstack command and its subcommands, run it:

```
$ openstack help [subcommand]

$ openstack --help
```

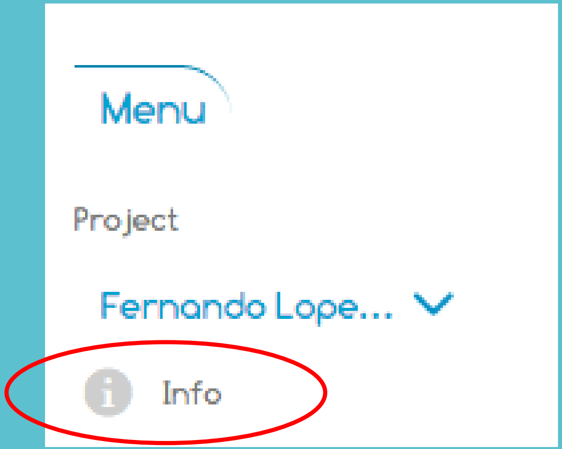
- Typing openstack switches to the interactive mode.

  Typing help [subcommand] for information.

  To exit the interactive mode, type quit.

OpenStack client hands-on

- Next steps: we need the credentials of your OpenStack environment.
- Just login into FIWARE Lab Cloud Portal (https://cloud.lab.fiware.org/) and get your data:



OpenStackClient hands-on

- File downloaded a file with format:

<your user name>-openrc

- Content:
  - OS_USERNAME
  - OS_PASSWORD
  - OS_TENANT_NAME
  - OS_REGION_NAME
  - OS_AUTH_URL

```
ubuntu@ubuntu:~/malaga$ more
fernando.lopezaguilar@telefonica.com-openrc


export
OS_USERNAME=fernando.lopezaguilar@telefonica.
com

export OS_PASSWORD=

export OS_TENANT_NAME="Fernando Lopez
cloud”

export OS_REGION_NAME=Spain2

export OS_AUTH_URL=http://130.206.84.8:4730/v3/
```

OpenStackClient hands-on

- You need to edit the file and add the following.

```
export OS_PROJECT_DOMAIN_NAME=default

export OS_USER_DOMAIN_NAME=default

export OS_IDENTITY_API_VERSION=3
```

- I suggest to add also the following line.

```
export PS1='(`basename
\"$VIRTUAL_ENV`)[\u@FIWARE Lab
\W(keystone_user)]\$ ’
```

- Move the file to your work directory and load it.

```
ubuntu@ubuntu:~/malaga$ source
fernando.lopezaguilar@telefonica.com-openrc
```

OpenStackClient hands-on

- Your turn…
  - Get help about openstack client
  - Create a keypair.
  - List of keypairs.

I’ve been to the dark side…

THEY HAVE ANIBLE… !!!

Ansible

- Ansible is an IT automation tool.
- Manages machines in an agent-less manner.
- Uses of OpenSSH for transport and a YAML language.
- Main focus in software deployment and system configuration.

What do you need to know?

- Connect to a remote machine using SSH.
- Interact with the bash command-line shell.
- Install packages.
- Use the sudo command.
- Check and set file permissions.
- Start and stop services.
- Set environment variables.
- Write scripts (any language).

Previous action

- Install ansible (v 2.2.0.0)
- Using openstackclient, deploy a server.
    - Create a keypair
    - Create security group an assign rule
    - Create a server

First steps with Ansible

- Define inventory file (*hosts).*
- INI-like format and looks like this:*:*
  - <server name>: name of the server
  - ansible_ssh_host: IP of the server
  - ansible_ssh_user: user to access via ssh
  - ansible_ssh_private_key_file: key pair to access the server

First steps with Ansible

- Connect to the server named <*server name*> and invoke the ping module.

```
ubuntu@ubuntu:~/malaga$ ansible ansible-test -i hosts -m ping

ansible-test | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Time to do bigger things

Ansible

- Process description in YAML description file.
- YAML is a human friendly data serialization standard for all programming languages.
- A deployment is described in a “playbook” (e.g.: deploy a web application).
- A “playbook” can define several processes o deployments

```
---
invoice: 12345
product:
  - id: 987
    amount: 2
    description: basketball ball
  - id: 760
    amount: 1
    description: football ball
```

Ansible inventory

- Ansible works against multiple systems in your infrastructure at the same time.
- By defaults, saved in the location /etc/ansible/hosts but can be specified a different inventory with –i command.
- Not only is this inventory configurable, but you can also use multiple inventory files at the same time.
- You can use a script to generate dynamically the inventory.

Using Variables: About Jinja2

- Ansible allows you to reference variables in your playbooks using the Jinja2 templating system.
- Example: in a simple template, you can do something like:

```
My amp goes to {{ max_amp_value }}
```

- This is also valid directly in playbooks:

```
template: src=foo.cfg.j2 dest={{ remote_install_path }}/foo.cfg
```

- Variables might be defined in YAML file (e.g. vars/main.yml)

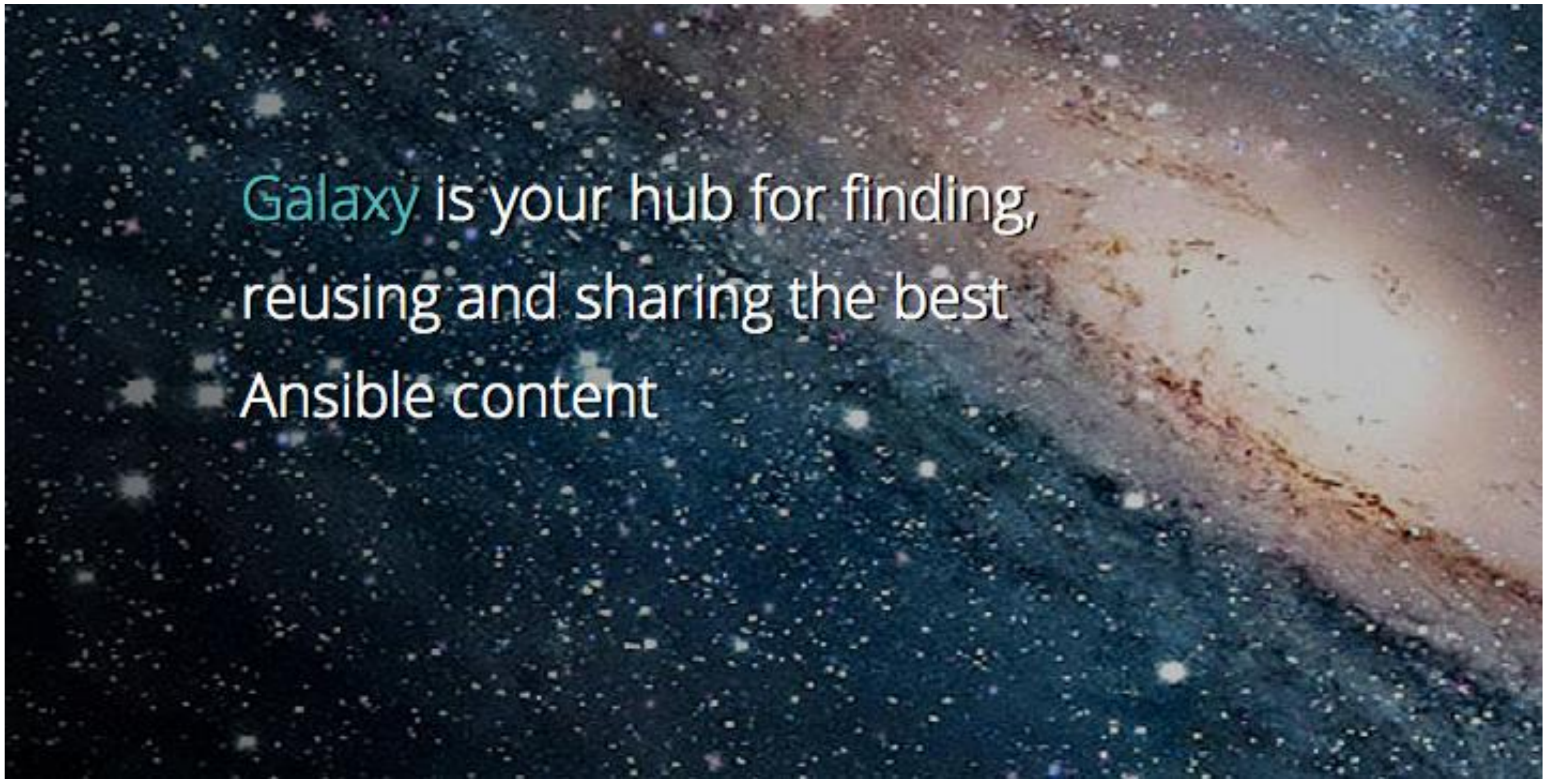
Ansible modules

- Ansible ships with a number of modules (called the ‘module library’).
- They can be executed directly (-m option) on remote hosts or through Playbooks.
- Users can also write their own modules.
- These modules can control system resources, like services, packages, or files (anything really), or handle executing system commands.

Ansible roles

- Instead of creating giant playbooks with hundreds of tasks we can use roles to organize tasks.
- A role breaks task into smaller more discrete units of work.
- A role is all the tasks, variables and handlers needed to complete the unit of work.
- This allows a role to be completely self contained or encapsulated and completely reusable.
- Example of role is the installation and configuration of NTPd service.

Ansible Galaxy



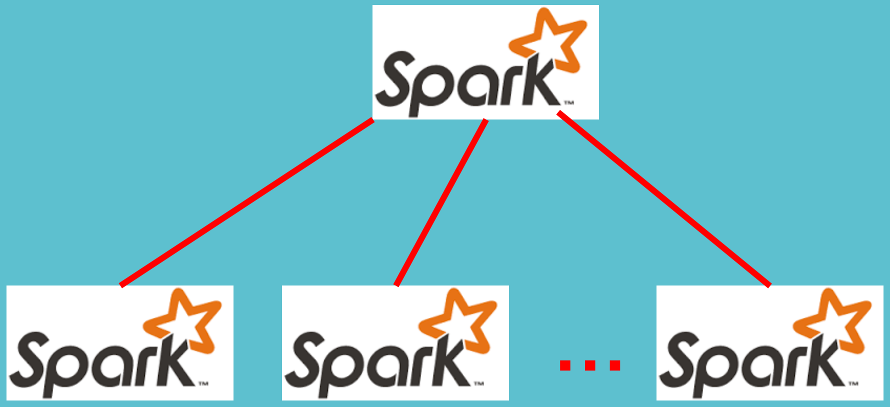
Galaxy is your hub for finding, reusing and sharing the best Ansible content

Where the hell is Spark…?

Spark

- Apache Spark is a fast and general-purpose cluster computing system.
- Provide high-level APIs in Java, Scala, Python and R.
- Support a rich set of higher-level tools:
  - Spark SQL for SQL and structured data processing.
  - MLlib for machine learning.
  - GraphX for graph processing, and Spark Streaming.

Spark in HA



- Creation of a Master node and several Slaves nodes.
- Configuration of the instances.
- Configuration of Spark nodes.

Hands-on Spark deployment with Ansible

- Clone the github repository

  https://github.com/flopezag/ansible_spark_openstack

- Follow the instructions in README.md file
- Take a look into /vars/main.yml file.

Yeah, if you could just,
come to the dark side…

References

- OpenStack Client Command List:

  http://docs.openstack.org/developer/python-openstackclient/command-list.html

- Ansible documentation:

  http://docs.ansible.com/ansible/

- FIWARE Lab Account portal:

  https://account.lab.fiware.org/idm/

DREAM BIG
IT WORKS
FIWARE
Open APIs for Open Minds

Thank you!

http://fiware.org
Follow @FIWARE on Twitter