Hyperconvergence meets Big Data

Rafael Monnerat

rafael (at) nexedi (dot) com

@ramonnerat

https://lab.nexedi.com/u/rafael

▼ Details

This presentation aims to demonstrate how to use SlapOS (Hyperconverged OS) to deploy an entire Big Data Infrastrucure and show how “data life cycle” can be managed with Wendelin - covering ingestion, analysis, visualization and weaving it into an application.

Agenda

- Hyperconvergence with SlapOS
- Big Data with Wendelin
- How to deploy?
- Upload Data
- Jupyter Quick Demos

▼ Details

We’ll show how Wendelin and SlapOS could handle acquisition, analysis and exploitation of data, making it a potential solution for IOT scenarios where data is available and needs some logic applied before being presented as web application, possibly on a commercial basis.

The agenda of the presentation includes an introduction on SlapOS, as a tool used to deploy a wide range of different services and an introduction of Wendelin, as a tool in order to make out-of-core python applications.

After a short introduction, we progress to show the steps to deploy SlapOS infrastructure and later to deploy Wendelin on the just deployed SlapOS, including an use case which shows SlapOS deploying a fluentd instance to ingest data to the Wendelin Database.

To conclude, we make a live demo with an Jupiter using out-of-core python to handle wav files stored on Wendelin, and a second short demo on handle computer resources consumption data.

Nexedi: largest OSS publisher in Europe

▼ Details

Stack 100% Open Source

SlapOS - HyperconvergenceOS

http://community.slapos.org/

▼ Details

SlapOS architecture uses two types of servers: SlapOS Master and SlapOS Nodes.

SlapOS Nodes are hosted in multiple data centers, providers or at Homes/Offices. SlapOS Nodes exchange information with the SlapOS Master through the SLAP protocol. SLAP stands for “Simple Language for Accounting and Provisioning”.

SlapOS Master tells to each SlapOS Node which software it should install and which instances of which software each should run. Each SlapOS Node tells SlapOS Master how much resources were consumed by each instance of each software.

Current architecture of SlapOS, based on a single redundant master node, is designed to manage thousands of servers. Future versions of SlapOS – such as the SafeOS proposal of the RESILIENCE project – may rely on a distributed implementation SlapOS Master.

SlapOS - HyperconvergenceOS

▼ Details

SlapOS Node

▼ Details

An SlapOS Node is composes by 3 distinct layers:

- "SlapOS Core" or SlapOS Kernel:
- Software Release:
- Software Intances:

"One System to Rule them All"

- CDN/Mesh Networking (Grandenet)
- KVM Clusters for Big Data (Teralab)
- Wendelin Environments for Big Data (Wendelin)
- Development PaaS for Developers (Nexedi)
- Distributed Test Nodes to run Unit Test (Nexedi)
- Automated Ready to Use VMs (VIFIB)
- ChromiumOS images Builder (NayuOS)

▼ Details

Wendelin - Out-of-core Pydata

http://www.wendelin.io/architecture
▼ Details

Data Ingestion

▼ Details

SlapOS Deployment (with token)

```
wget https://deploy.erp5.cn/slapos && bash slapos
[... Install Ansible ...]
Starting Ansible playbook:
What is this computer name?  (...): [noname]: COMPUTER-NAME
If you have slapos token if you have (...): [notoken]: 20010101-ABDC
```

Keep it simple with single command to type...

▼ Details

SlapOS Deployment Standalone

```
# Leave the computer name and token empty
wget https://deploy.erp5.cn/slapos && bash slapos
[... Ansible is installed...]
Starting Ansible playbook:
What is this computer name?  (...): [noname]:
```

If you have slapos token if you have (...): [notoken]:

Them..

```
# Configura Local Master
slapos configure local
```

```
# Prepare the computer to run services.
slapos node format --now
```

▼ Details

You can work on standalone mode, when you don't need to manage more them one computer

Easy Deployment (client-only)

&nbps;

```
# You can use easy_install or pip
easy_install slapos.core
```

```
pip install slapos.core
```

```
slapos configure client
```

▼ Details

All packages already contains a client builtin however you can install a client...

Supplying and Requesting monitor (fluentd)

```
# Supply will provide make the computer deploy the
# "product.monitor" software on COMPUTER with refernce COMP-1239
slapos supply https://lab.nexedi.com/nexedi/slapos/raw/1.0.33/software/monitor/software.cfg COMP-1239
```

```
# The Request will ask to the to the COMP-1239 instantiate one instance
# of the Software Release "product.monitor"

 slapos request my_first_instance\
 https://lab.nexedi.com/nexedi/slapos/raw/1.0.33/software/monitor/software.cfg \
 --parameters item=True --node computer_guid=COMP-1239
```

```
# You can also use alias for give me the latest monitor release
slapos supply product.monitor COMP-1239
```

```
# By not passing --node , your instance will be allocated on any computer
# has the wanted software release (respecting security roles of your user)
slapos request my_first_instance product.monitor --parameters item=True
```

Monitor contains fluentd

▼ Details

Supplying and Requesting Wendelin Stack

```
# Supply will provide make the computer deploy the
# "product.monitor" software on COMPUTER with refernce COMP-1239

 slapos supply https://lab.nexedi.com/nexedi/slapos/raw/1.0.33/software/wendelin/software.cfg COMP-1239
```

```
# The Request will ask to the to the COMP-1239 instantiate one instance
# of the Software Release "product.monitor"
```

slapos request my_first_instance
**https://lab.nexedi.com/nexedi/slapos/raw/1.0.33/software/wendelin/software.cfg **
--parameters item=True --node computer_guid=COMP-1239

Monitor contains fluentd

▼ Details

Deploying Wendelin (Standalone)

```
wget https://deploy.erp5.cn/wendelin-standalone && bash wendelin-standalone
```

▼ Details

Ready to use VMs (soon)

Not ready yet but soon images will be released for qemu, ec2, digital ocean, VMware...

▼ Details

Uploading your Wavs

Create configuration file

```
@type bin
format none
path
/srv/slapgrid/slappart9/srv/runner/PUT_YOUR_WAV_HERE/*.wav
pos_file
/srv/slapgrid/slappart9/srv/runner/Demo.pos
enable_watch_timer false
read_from_head true
tag wavdemo
```

```
@type wendelin
@id wendelin_out

  streamtool_uri https://softinst11111.host.vifib.net/erp5/portal_ingestion_policies/wavdemo
  user      zzz
  password  yyy

buffer_type       memory
flush_interval    1s
disable_retry_limit    true
```

and them run **fluentd -c configuration.cfg**

▼ Details

Files Uploaded!

The files are uploaded...

▼ Details

Wendelin Modules Overview

- Ingestion Policies
- Data Stream
- Data Arrays

▼ Details

Jupyter Wav Demos

Wav Demo Jupyter Notebook
▼ Details

This presentation continue down at the Jupyter

Jupyter Wendelin Async

CMFActivities Jupyter Notebook

▼ Details

Javascript-Based Gadgets

NXD-Presentation.Hyperconvergence.Big.Data.Small.App Full tutorial
▼ Details

pip install wendelin.core

$ **ipython**
Python 2.7.11+ (default, Jun 2 2016, 19:34:15)
...
 # imports
In [1]:
from wendelin.bigarray.array_zodb import ZBigArray
In [2]:
from wendelin.lib.zodb import dbopen, dbclose
In [3]:
import transaction
In [4]:
import numpy as np
 # open/create database for tests (on local disk for now)
In [5]:
root = dbopen('test.fs')
 # create 10 items 1d array object
In [6]:
root['A'] = A = ZBigArray((10,), np.int)
In [7]:
transaction.commit()
 # see what it is
In [8]:
A
Out[8]:

In [9]:
a = A[:]
In [10]:
a
Out[10]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
In [11]:
type(a)
Out[11]: numpy.ndarray

Wendelin Core Quick Tutorial
▼ Details

Thank you

Rafael Monnerat

rafael (at) nexedi (dot) com

@ramonnerat

We are hiring! https://www.nexedi.com/jobs

▼ Details