Toward Fog/Edge Deployments: Evaluating OpenStack WANwide with Enos

❧

Ronan-Alexandre Cherrueau (`@rcherrueau`)

Inria, Discovery Initiative

OpenStack

A IaaS manager

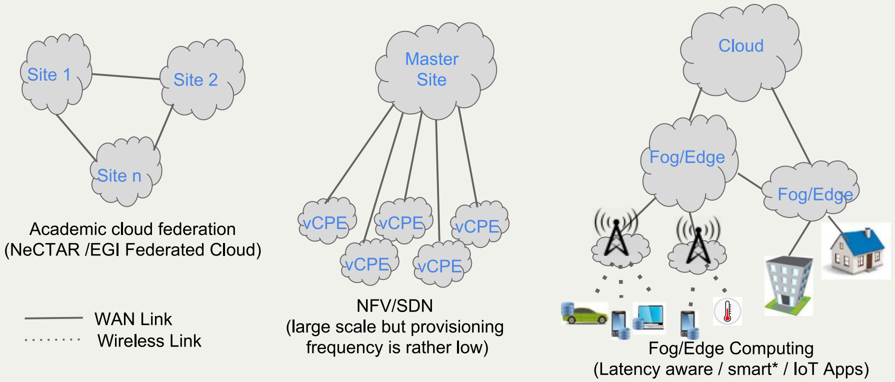
- Compute with Nova (VM), Ironic (bare metal) and Magnum (container)
- Network with Neutron
- Storage with Cinder (volume) and Swift (object)

The *de facto* solution

- EC2, Compute Engine, Azure: Closed Source Public Cloud
- Cloudwatt, Numergy …
- *OpenStack*: Open Source Private and Public Cloud 👍

Question: is OpenStack the de facto solution for distributed Cloud?

Distributed Cloud


Site 1
Site 2
Site n
Academic cloud federation
(NeCTAR /EGI Federated Cloud)
WAN Link
Wireless Link
Master Site
vCPE
vCPE
vCPE
vCPE
vCPE
NFV/SDN
(large scale but provisioning frequency is rather low)
Cloud
Fog/Edge
Fog/Edge
Fog/Edge Computing
(Latency aware / smart* / IoT Apps)

Management services interconnected through WAN Link (OpenStack WANwide)

OpenStack Mechanisms for Distributed Cloud

Native Mechanisms

- Centralized: 1 control node and *n* computes
- Mutli-regions: *n* OpenStack Clouds managed by Keystone (no global view)

Research work

- Broker: *n* OpenStack Clouds managed by an external service that builds the global view (Tricircle)
- P2P: *n* OpenStack Clouds that collaborate to build the global view (Discovery Initiative)

Which one is the most interesting?

OpenStack lacks of a tool for performance evaluations

- Latency/throughput impact?
- Message characterization: distinction between LAN and WAN traffic?
- Changes between OpenStack releases
- Deployment complexity

A Sandbox for Conducting Performance Analysis of OpenStack?

Enos: Experimental Env. for OpenStack

Motivation: Conducting performance analysis

- In a scientific and reproducible manner (automation)
- At small and large-scale
- Under different network topologies (traffic shaping)
- Between different releases

Workflow

1. `enos deploy`: Get testbed resources; Deploys OpenStack
2. `enos bench`: Runs benchmarks; Measures CPU/RAM/Network consumption per service/node
3. `enos backup`: Get benchmarks results

`enos deploy` - Resource/Topology Description

```
$ cat ./basic.yml
resources:
  parasilo:
    control: 1
    network: 1
  paravance:
    compute: 50

$ enos deploy -f ./basic
     .yml
```

```
$ cat ./advanced.yml
resources:
  parasilo:
    control: 1
    network: 1
    nova-conductor: 5
  paravance:
    compute: 50

$ enos deploy -f ./
     advanced.yml
```

```
$ cat ./network-topo.yml
resources:
  grp1:
    parasilo:
      control: 1
      network: 1
      nova-conductor: 5
  grp2:
    paravance:
      compute: 50

network_constraints:
  - src: grp1
    dst: grp2
    delay: 100ms
    rate: 10Gbit
    loss: 0%
    symetric: yes

$ enos deploy -f ./
     network-topo.yml
```

`enos deploy` – Under the Hood

```
resources:
  grp1:
    parasilo:
      control: 1
      network: 1
  grp2:
    paravance:
      compute: 50

network_constraints:
    delay: 100ms
    rate: 10Gbit
    loss: 0%
```

$\Rightarrow$

1. Provider gets 2 nodes on parasilo, 50 nodes on paravance and returns node's IP addresses
2. Enos provisions nodes with Docker daemon
3. Enos installs OpenStack using *Kolla-ansible*
4. Enos sets up bare necessities (flavors, cirros image, router, ...)
5. Enos applies network constraints between grp1 and grp2 using tc

Provider to get testbed resources

- Resource ≡ Anything running a Docker daemon and Enos can SSH to.
- Existing provider: Vagrant (VBox), Grid'5000, Chameleon, OpenStack
- ~500 LoC

enos bench

- Benchmarks description

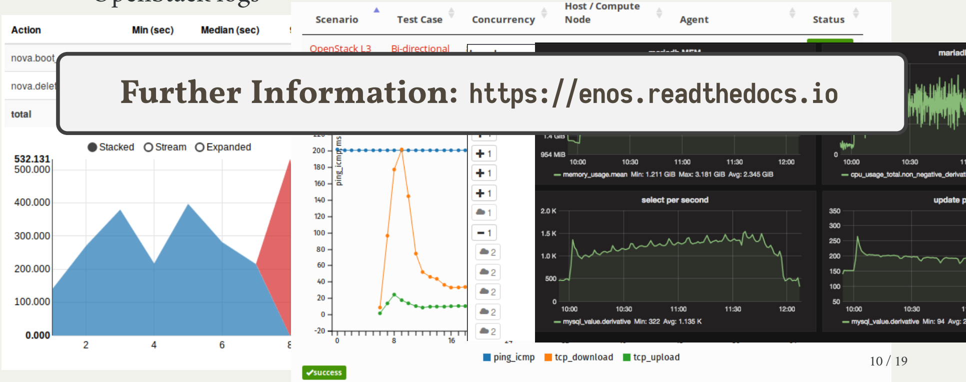
```
$ cat ./run.yml
rally:
  args:
    concurrency: 5
    times: 100
  scenarios:
    - name: boot and list servers
      file: nova-boot-list-cc.yml
      osprofiler: true
    - ...
shaker: ...

$ enos bench --workload=run.yml
```

- Under the hood
  - Rally: control plane benchmark
  - Shaker: data plane benchmark
  - OSProfiler: code profiling
  - Monitoring stack:
    - cAdvisor/Collectd: CPU/RAM/Network consumption per service/node
    - InfluxDB
    - Grafana

enos backup

enos backup produces a tarball with:

- Rally/Shaker reports
- OSProfiler traces
- InfluxDB database with cAdvisor/Collectd measures
- OpenStack logs



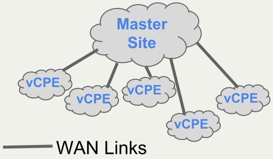
Further Information: `https://enos.readthedocs.io`

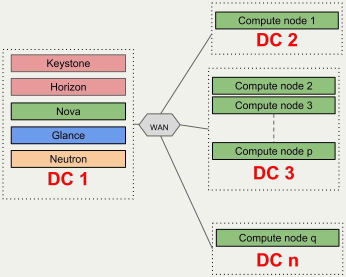
Enos Example: Evaluation of OpenStack WANwide

OpenStack WANwide

A single OpenStack to operate remote compute resources deployed at the edge

- Pros: simple
- Cons:
  - Single point of failure
  - Scalability (not addressed in this presentation, see “Chasing 1000 Nodes Scale”, Barcelona Summit 2016 – Done with Enos)
  - **Network latency/throughput impacts on functional behaviour and performance degradation**.



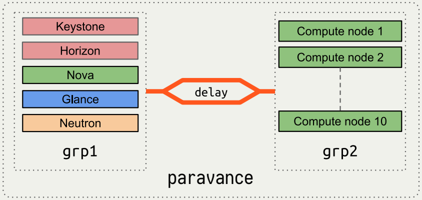


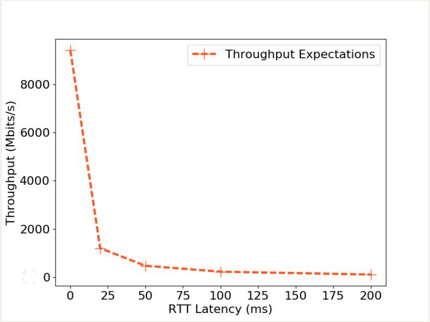
TestBeds: Grid'5000 + Chameleon

- Experiments runs independently on Grid'5000 and Chameleon in a fully automatized manner (software defined experiments leveraging Enos).
- 250 benchmarks (approx. 100 running hours) on each testbed.
- Results lead to the same conclusion whatever the testbed (collected performance are almost identical).
- Experimental setup: `https://github.com/BeyondTheClouds/enos-scenarios/`
- Results: `http://enos.irisa.fr/html/`

Latency Impact (Experiment #1)

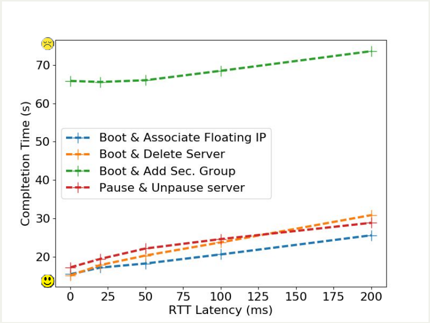
```
$ cat ./wan-exp1.yml
resources:
  grp1:
    paravance:
      control: 1
  grp2:
    paravance:
      compute: 10

network_constraints:
  - src: grp1
    dst: grp2
    delay: 0ms # 10ms, 25ms, 50ms, 100ms
    rate: 10Gbit
    loss: 0%
    symetric: yes

$ enos deploy -f ./wan-exp1.yml
```





Latency Impact - Control Plane (Rally Vision)

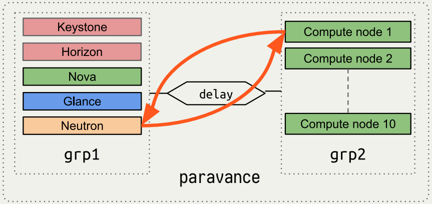
```
$ cat ./run.yml
rally:
  args:
    concurrency: 1
    times: 20
  scenarios:
    - file: nova-boot-and-assoc-fip.yml
    - file: nova-boot-and-delete.yml
    - file: nova-boot-and-add-sec.yml
    - file: nova-pause-and-unpause.yml
shaker: ...

$ enos bench --workload=run.yml
```

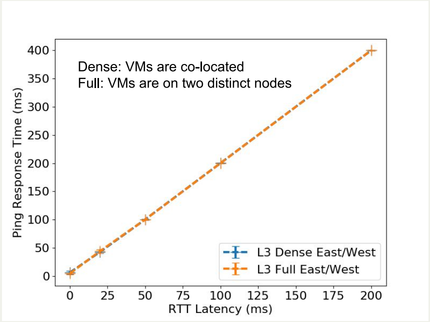


Completion time increases with latency (factor 2 between 0 and 200ms)

Latency Impact - Data Plane (Shaker Vision)

```
$ cat ./run.yml
rally: ...
shaker: ...
  - file: openstack/dense_l3_est_west.yml
  - file: openstack/full_l3_est_west.yml

$ enos bench --workload=run.yml
```



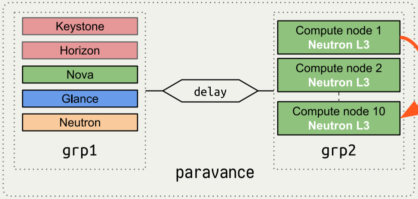


Ping response time is twice the RTT

Latency Impact with DVR (Experiment #2)

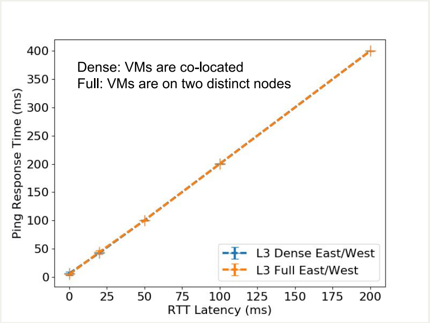
You say DVR?

- Distributed Virtual Routing
- L3 forwarding/NAT distributed to compute nodes

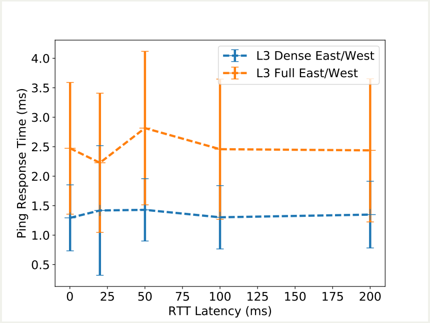


```
$ cat ./wan-exp2.yml
resources: ...
network_constraints: ...
kolla:
  enable_neutron_dvr: yes

$ enos deploy -f ./wan-exp2.yml
```

Latency Impact with DVR - Data Plane



Without DVR (2 $*$ RTT)



With DVR (LAN RTT)

Activating DVR is a Critical Change in WAN Context

Conclusion

You wanna conduct performance analysis of OpenStack: Use Enos!

Conducted/Ongoing Experiments with Enos

- Chasing 1000 nodes scale (OS Summit Barcelona) – `https://youtu.be/XURkQ3biF6w`
- Toward Fog, Edge and NFV Deployments (OS Summit Boston) – `https://youtu.be/xwT08H02Nok`
- Substitute MariaDB with CockroachDB
- Substitute RabbitMQ with QPid-dispatch

Important links

- Enos: `https://enos.readthedocs.io/en/stable/`
- Discovery Initiative: `https://beyondtheclouds.github.io/`