EnOS(Lib), The Library for Edge Computing Experiments

Ronan-Alexandre Cherrueau & Javier Rojas Balderrama

July 10th, 2019

IPL Discovery

Managing Resources of an Edge Infrastructure

Edge Infrastructure?

A kind of distributed Cloud infrastructure

Particularities

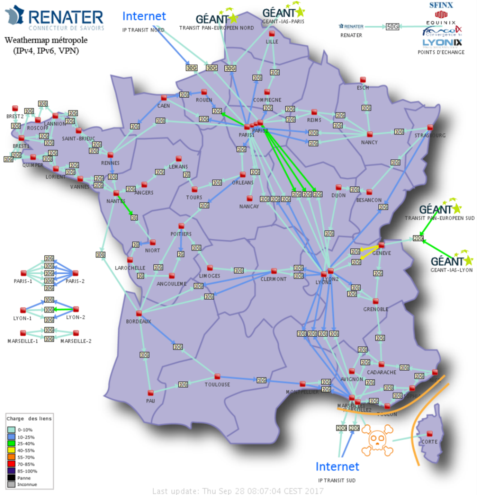
- 100s/1000s of locations (*i.e.*, Data Centers)
- Dozen of servers per Data Center
- Wan links (10 to 300 ms RTT)

- Intermittent connectivity
- Network partitioning issues

Example of an Edge Infrastructure

Renater backbone

- Point of Presence (PoP) in red
- Micro DC in each PoP
  - Dozens of servers
- WAN links interconnects PoPs
  - 10ms, Paris ↔ Marseille
  - 300ms, Paris ↔ Brasilia
- Net. partitions risks (☠) between PoPs
  - Marseille/Corte

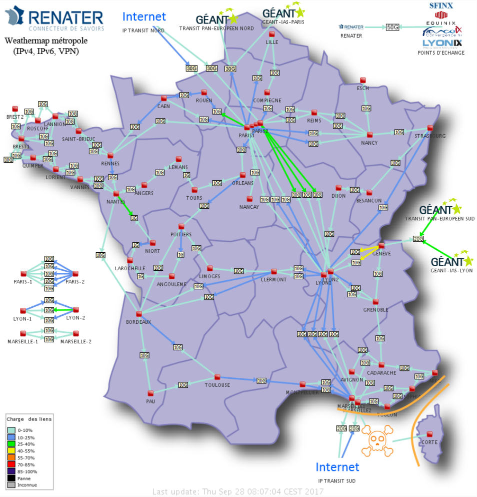


Last update: Thu Sep 28 08:07:04 CEST 2017

Managing Resources of an Edge Infrastructure?

Same as in Cloud Computing. Tuned for the Edge. [Cherrueau et al., 2018a]

1. Operate/use a single DC (1-DC)
   - Manage users, flavors, quotas
   - Provision computes, storages, nets
2. Operate/use several DCs
   - Cross-DC collaborative provisioning (x-DC)
   - Manage multiple DCs simultaneously (⋆-DC)
3. Support disconnections
   - Access/Manage reachable resources (full isolation)

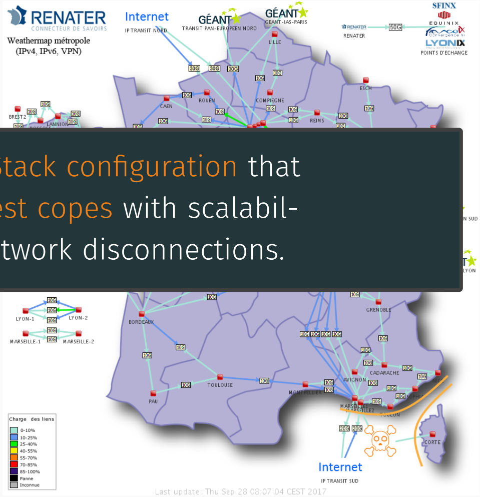


Managing Resources of an Edge Infrastructure?

Same as in Cloud Computing. Tuned for the Edge. [Cherrueau et al., 2018a]

1. Operate/use a single DC (1-DC)
   - Manage users, flavors, quotas
   - Provision computes, storages, nets
2.
   - Manage multiple DCs simultaneously (⋆-DC)
3. Support disconnections
   - Access/Manage reachable resources (full isolation)

Discovery project: Find an OpenStack configuration that supports 1-DC, x-DC, ⋆-DC and best copes with scalability, links latency/throughput, network disconnections.



Managing Resources of an Edge Infrastructure with OpenStack

The Many Multifarious Moiling Configurations of OpenStack

- Independent OpenStack in each PoP
- Centralize control plane in one PoP + Remote computes in others
- Segregation techniques (Regions, Nova Cells)
- Federated/Brokering approaches
- OpenStack in each PoP with a global/geo-distributed database
- Something else. Something new...

OpenStack – the devil is in the detail

- 186 services
- 13,000,000 LoCs
- 6 months release cycle

Managing Resources of an Edge Infrastructure with OpenStack

The Many Multifarious Moiling Configurations of OpenStack

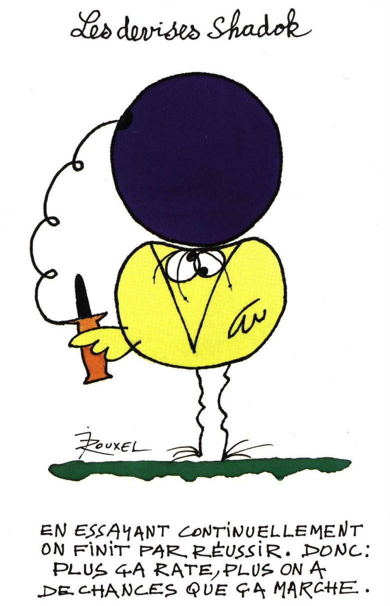
- Independent OpenStack in each PoP
- Centralize control plane in one PoP + Remote computes in others
- Segregation techniques (Regions, Nova Cells)

> Build an *analytic* tool to figure out which OpenStack conf. supports 1-DC, x-DC, ⋆-DC and best copes with scalability, links latency/throughput, network disconnections.

OpenStack – the devil is in the detail

- 186 services
- 13,000,000 LoCs
- 6 months release cycle

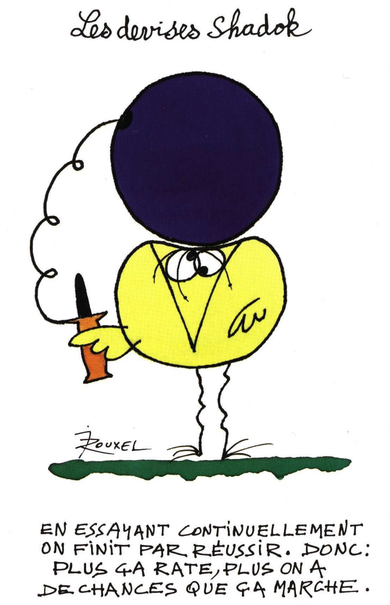
OpenStack Analysis


Les devises Shadok
ROUXEL
EN ESSAYANT CONTINUELLEMENT ON FINIT PAR RÉUSSIR. DONC: PLUS ÇA RATE, PLUS ON A DE CHANCES QUE ÇA MARCHE.

A tool to conduct perf. analysis of OpenStack

- Scientific and reproducible manner (automation)
- At small and large-scale
- Under different net. topologies
- Between different releases & configurations

OpenStack Analysis


Les devises Shadok
ROUXEL
EN ESSAYANT CONTINUELLEMENT ON FINIT PAR RÉUSSIR. DONC: PLUS ÇA RATE, PLUS ON A DE CHANCES QUE ÇA MARCHE.

A tool to conduct perf. analysis of OpenStack

- Scientific and reproducible manner (automation)
- At small and large-scale
- Under different net. topologies
- Between different releases & configurations

Objectives

- Understand OpenStack behavior
- Evaluate new proposals
- Find solution (1-DC, x-DC, $\star$-DC, scalabiltiy, latency, disconnections)

EnOS, the *Tool* for OpenStack Experiments at the Edge

EnOS: Experimental eNvironment for OpenStack

Workflow

1. `$ enos deploy`
   - Get testbed resources
   - Deploy OpenStack

2. `$ enos bench`
   - Perform benchmarks
   - Collect metrics

3. `$ enos backup`
   - Save bench results & metrics
   - Visualize & share

4. `$ enos destroy`
   - Remove OpenStack
   - Release testbed resources

Enos the first chimpanzee to achieve Earth orbit during the NASA Mercury program that then leads to the Apollo program

$ enos deploy – Resource/Topology Description

```
$ cat ./basic.yml
> resources:
>   clusterA:
>     control: 1
>     network: 1
>   clusterB:
>     compute: 50

$ enos deploy ./basic.yml
```

$ enos deploy – Resource/Topology Description

```
$ cat ./basic.yml
> resources:
>   clusterA:
>     control: 1
>     network: 1
>   clusterB:
>     compute: 50

$ enos deploy ./basic.yml
```

```
$ cat ./advanced.yml
> resources:
>   clusterA:
>     control: 1
>     network: 1
>     nova-conductor: 5
>   clusterB:
>     compute: 50

$ enos deploy ./advanced.yml
```

`$ enos deploy` – Resource/Topology Description

```
$ cat ./basic.yml
> resources:
>   clusterA:
>     control: 1
>     network: 1
>   clusterB:
>     compute: 50

$ enos deploy ./basic.yml
```

```
$ cat ./advanced.yml
> resources:
>   clusterA:
>     control: 1
>     network: 1
>     nova-conductor: 5
>   clusterB:
>     compute: 50

$ enos deploy ./advanced.yml
```

```
$ cat ./net—topo.yml
> resources:
>   grp1:
>     clusterA:
>       control: 1
>       network: 1
>       nova—conductor: 5
>   grp2:
>     clusterB:
>       compute: 50
>
> network-constraints:
>   - src: grp1
>     dst: grp2
>     delay: 100ms
>     rate: 10Gbit
>     loss: 0%

$ enos deploy ./net—topo.yml
```

`$ enos deploy` – Under the Hood

- Build upon abstract resource units
  - computation unit – *anything EnOS can SSH to and run Docker on*
  - network unit – *any Linux network device*

- EnOS deploys OpenStack with Kolla
  - *over abstract computation units*
- EnOS applies network constraints with tc
  - *over abstract network units*

- Provider gets concrete resources from a testbed
  - VBox, Libvirt, Grid’5000, VM on Grid’5000 (segregation), Chameleon, OpenStack, Static testbed, ...
  - ~500 LoCs

`$ enos deploy` – Grid'5000 Example

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

network-constraints:
  - src: grp1
    dst: grp2
    delay: 100ms
    rate: 10Gbit
    loss: 0%
```

```
$ enos deploy
```

⇒

1. Provider gets 2 nodes on paravance, 50 nodes on parasilo and returns node IP addresses
2. EnOS provisions nodes with Docker daemon (Kolla dependency)
3. EnOS installs OpenStack using Kolla
4. EnOS sets up bare necessities (flavors, images, router, ...)
5. EnOS applies network constraints between grp1 and grp2 using tc

$ enos bench

```
$ 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/cluster

$ enos bench

```
$ 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/cluster

EnOS makes the list of computation/network units accessible for an integration with outside tools

`$ enos backup`

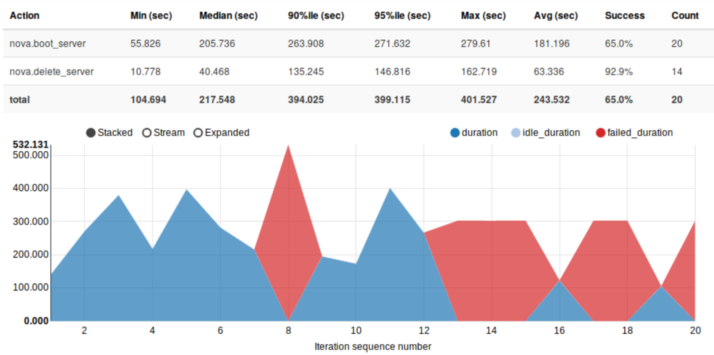
Produces a tarball with

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

$ enos backup

Produces a tarball with

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



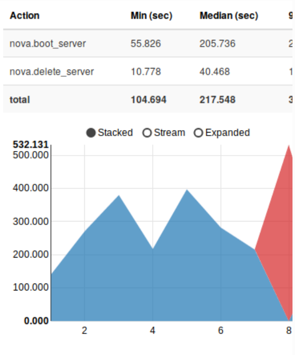
Action	Min (sec)	Median (sec)	90%ile (sec)	95%ile (sec)	Max (sec)	Avg (sec)	Success	Count
nova.boot_server	55.826	205.736	263.908	271.632	279.61	181.196	65.0%	20
nova.delete_server	10.778	40.468	135.245	146.816	162.719	63.336	92.9%	14
total	**104.694**	**217.548**	**394.025**	**399.115**	**401.527**	**243.532**	**65.0%**	**20**

● Stacked ○ Stream ○ Expanded
● duration ● idle_duration ● failed_duration
532.131
500.000
400.000
300.000
200.000
100.000
0.000
2
4
6
8
10
12
14
16
18
20
Iteration sequence number

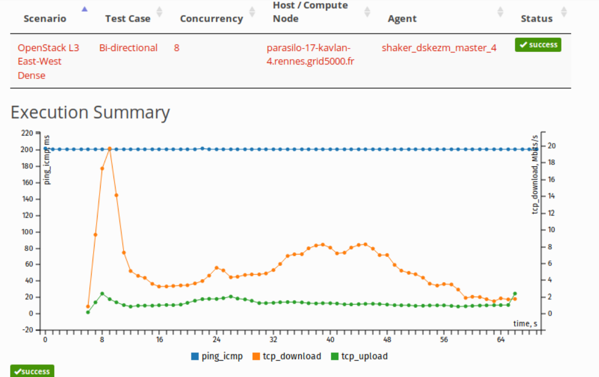
$ enos backup

Produces a tarball with

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



Action	Min (sec)	Median (sec)	9
nova.boot_server	55.826	205.736	2
nova.delete_server	10.778	40.468	1
total	**104.694**	**217.548**	3



Scenario	Test Case	Concurrency	Host / Compute Node	Agent	Status
OpenStack L3 East-West Dense	Bi-directional	8	parasilo-17-kavlan-4.rennes.grid5000.fr	shaker_dskezm_master_4	✓ success

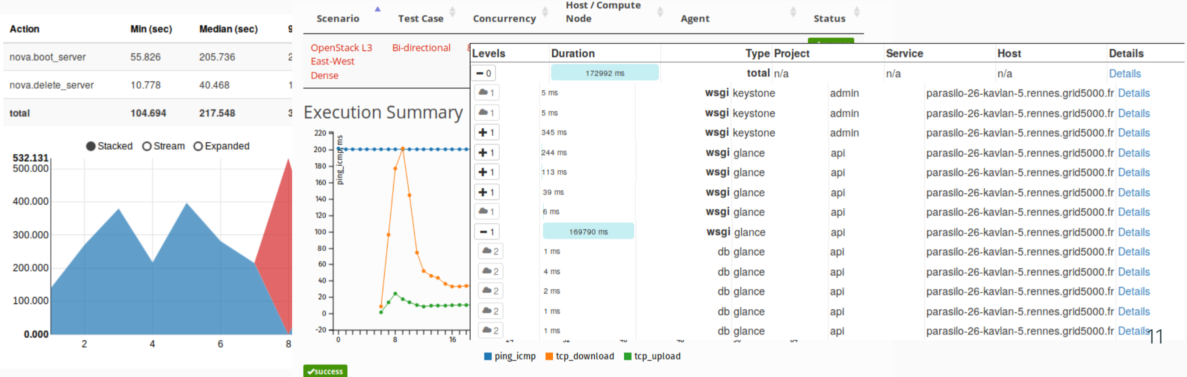
Execution Summary

✓success

$ enos backup

Produces a tarball with

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



Action	Min (sec)	Median (sec)	9
nova.boot_server	55.826	205.736	2
nova.delete_server	10.778	40.468	1
total	**104.694**	**217.548**	**3**

Scenario	Test Case	Concurrency	Host / Compute Node	Agent	Status
OpenStack L3 East-West Dense	Bi-directional	8			

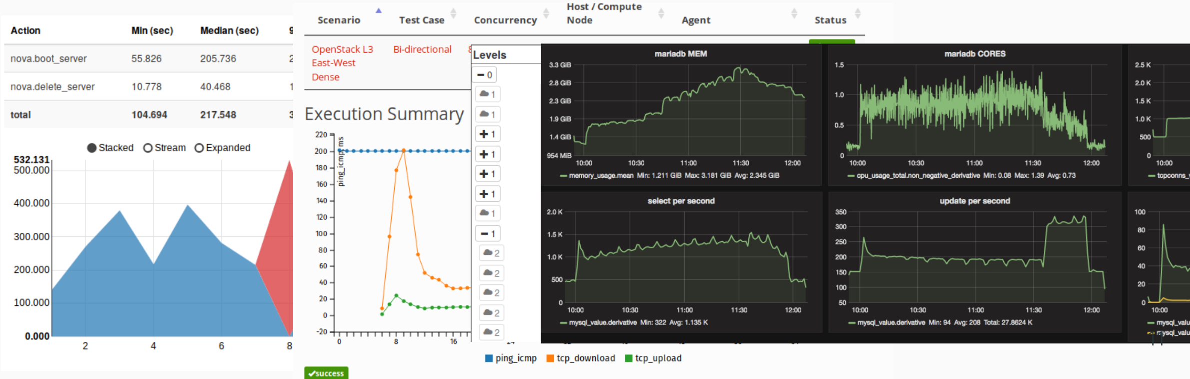
Execution Summary

Levels	Duration	Type	Project	Service	Host	Details
− 0	172992 ms	total	n/a	n/a	n/a	Details
1	5 ms	wsgi	keystone	admin	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
1	5 ms	wsgi	keystone	admin	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
+ 1	345 ms	wsgi	keystone	admin	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
+ 1	244 ms	wsgi	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
+ 1	113 ms	wsgi	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
+ 1	39 ms	wsgi	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
1	6 ms	wsgi	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
− 1	169790 ms	wsgi	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
2	1 ms	db	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
2	4 ms	db	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
2	2 ms	db	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
2	1 ms	db	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details
2	1 ms	db	glance	api	parasilo-26-kavlan-5.rennes.grid5000.fr	Details

$ enos backup

Produces a tarball with

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



Action	Min (sec)	Median (sec)	9
nova.boot_server	55.826	205.736	2
nova.delete_server	10.778	40.468	1
total	**104.694**	**217.548**	**3**

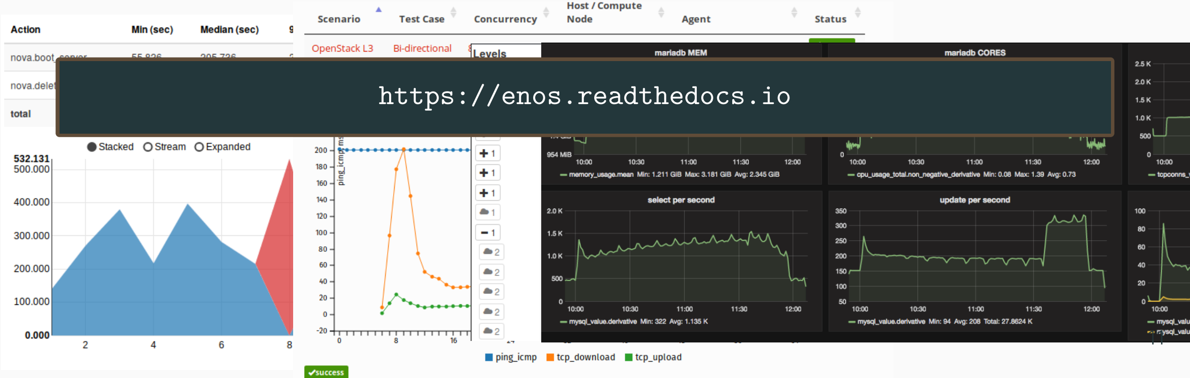
Scenario	Test Case	Concurrency	Host / Compute Node	Agent	Status
OpenStack L3 East-West Dense	Bi-directional	8			

Execution Summary

$ enos backup

Produces a tarball with

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



https://enos.readthedocs.io

Action	Min (sec)	Median (sec)
nova.boot_server	55.826	205.726
nova.delet		
total		

Scenario	Test Case	Concurrency	Host / Compute Node	Agent	Status
OpenStack L3	Bi-directional				

EnOS Slide of Fame – Users

- Chasing 1000 nodes scale [Simonin et al., 2016]
    - Mirantis/Inria – OpenStack Summit Barcelona
    - Test OpenStack scalability for computes
- OpenStack WANwide [Lebre et al., 2017]
    - University of Chicago/Inria – OpenStack Summit Boston
    - Study network latency/throughput impacts on functional behavior and performance degradations
- OpenStack IoT
    - FBK (Italy) – OpenStack Days Italy 2017
    - Test OpenStack support for IoT
- OpenStackoïd [Cherrueau et al., 2019]
    - Inria – OpenStack Summit Denver
    - Study of a new proposal for managing the collaboration in OpenStack
- Mutli-Level Elasticity for Data Stream Processing [Marangozova-Martin et al., 2019]
    - Université de Grenoble – IEEE TPDS
    - Stream processing under low latency
- ...

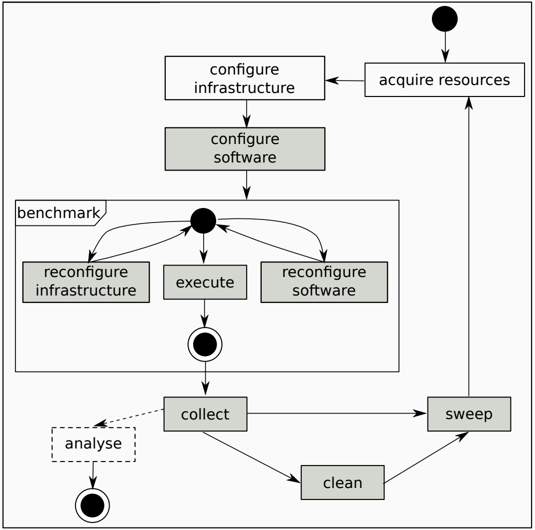
EnOSLib, the *Library* for Experiments with Edge Computing

Towards a generalisation of EnOS concepts

EnOSLib: *A task-oriented library for experimenters*

Main Features

- Vital organs of EnOS
- Common tasks to organise the experimenter's pipeline
- Multi-infrastructure support (extensible to new providers)
- Emulation of network constraints
- Built-in *idempotency*
- Remote launcher



How does EnOSLib help us to make experiments?

- Bootstraping an experiment with high-level abstractions:
  - Tasks
  - Resources
  - Roles
- Idempotent execution to iterate safely on experimental code
- Fully implemented in Python
- Continuous integration (CI): py35/py36 support, PEP8, and functional tests
- Continuous documentation (CD): automated pipeline, release on PyPi

A SDK for extensible development

```
# Resources description
conf = (Configuration()
          .add_machine(roles=[”control”], flavour=”large”, number=1)
          .add_machine(roles=[”compute”], flavour=”tiny”, number=3)
          .add_network(roles=[”network–interface”],  cidr=”192.168.42.0/24”)
          .finalize())

# Get resources
provider = Enos_vagrant(conf)
roles, networks = provider.init()

# Deploy generic monitoring
Monitoring(collector=roles[”control”],
           agent=roles[”control”]+roles[”compute”],
           ui=roles[”control”],
           network=”network–interface”).deploy()

# experimenter’s business code here ...

# Release resources
provider.destroy()
```

A SDK for extensible development

```
# Resources description
conf = (Configuration()
         .add_machine(roles=[”control”], flavour=”large”, number=1)
         .add_machine(roles=[”compute”], flavour=”tiny”, number=3)
         .add_network(roles=[”network—interface”], cidr=”192.168.42.0/24”)
         .finalize())

# Get resources
provider = Enos_vagrant(conf)
roles, networks = provider.init()
```

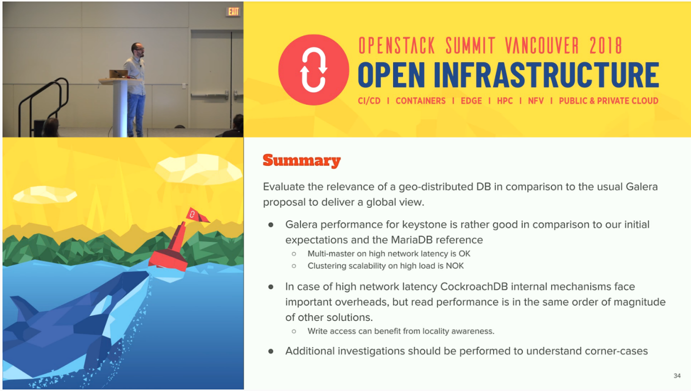
https://discovery.gitlabpages.inria.fr/enoslib/

```
           agent=roles[”control”]+roles[”compute”],
           ui=roles[”control”],
           network=”network—interface”).deploy()

# experimenter’s business code here ...

# Release resources
provider.destroy()
```

EnOSLib Highlights

Keystone in the context of Fog-Edge Massively Distributed Clouds

@Vancouver OpenStack Summit, 2018: `https://youtu.be/HqwaA_if9Kc`


EXIT
OPENSTACK SUMMIT VANCOUVER 2018
OPEN INFRASTRUCTURE
CI/CD I CONTAINERS I EDGE I HPC I NFV I PUBLIC & PRIVATE CLOUD

Summary

Evaluate the relevance of a geo-distributed DB in comparison to the usual Galera proposal to deliver a global view.

- Galera performance for keystone is rather good in comparison to our initial expectations and the MariaDB reference
  - Multi-master on high network latency is OK
  - Clustering scalability on high load is NOK
- In case of high network latency CockroachDB internal mechanisms face important overheads, but read performance is in the same order of magnitude of other solutions.
  - Write access can benefit from locality awareness.
- Additional investigations should be performed to understand corner-cases

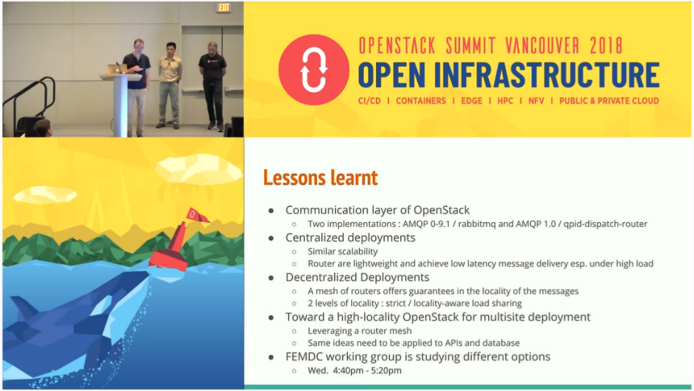
34

Develompent testing framework with EnOSLib

EnOSLib Highlights

OpenStack internal messaging at the edge: In depth evaluation

@Vancouver OpenStack Summit, 2018: `https://youtu.be/xGTW3FvJYI4`


OPENSTACK SUMMIT VANCOUVER 2018
OPEN INFRASTRUCTURE
CI/CD | CONTAINERS | EDGE | HPC | NFV | PUBLIC & PRIVATE CLOUD
EXIT

Lessons learnt

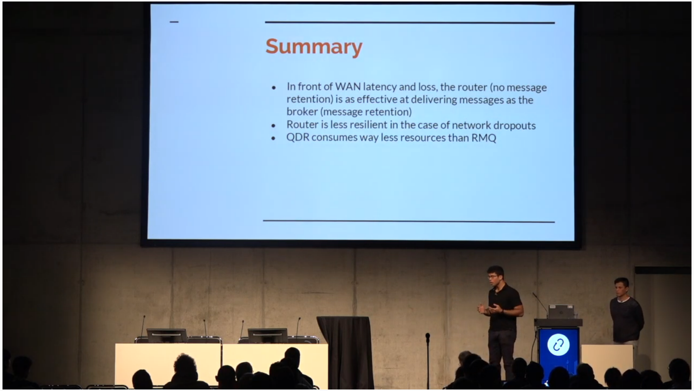
- Communication layer of OpenStack
  - Two implementations : AMQP 0-9.1 / rabbitmq and AMQP 1.0 / qpid-dispatch-router
- Centralized deployments
  - Similar scalability
  - Router are lightweight and achieve low latency message delivery esp. under high load
- Decentralized Deployments
  - A mesh of routers offers guarantees in the locality of the messages
  - 2 levels of locality : strict / locality-aware load sharing
- Toward a high-locality OpenStack for multisite deployment
  - Leveraging a router mesh
  - Same ideas need to be applied to APIs and database
- FEMDC working group is studying different options
  - Wed. 4:40pm - 5:20pm

Development and use of EnOS + Joint presentation with Red Hat

EnOSLib Highlights

RabbitMQ or Qpid Dispatch Router: Pushing OpenStack to the Edge

@Berlin OpenStack Summit, 2018: `https://youtu.be/i3neropoI9w`



Summary

- In front of WAN latency and loss, the router (no message retention) is as effective at delivering messages as the broker (message retention)
- Router is less resilient in the case of network dropouts
- QDR consumes way less resources than RMQ

Development of testing framework + Joint collaboration with Orange Labs

EnOS(Lib) Today & Tomorrow

Today

- Within the OpenStack ecosystem:
    - Several experimental frameworks
    - EnOS adoption in the community
- EnOS-Kubernetes
- EnOSLib + IPFS
- EnOSLib + BlockChain
- ...

Enos(Lib) allows us to experiment with Edge computing environments in a controlled fashion. This is aligned to the permanent researchers' need of tools to study large and complex infrastructures that are impossible to simulate due to their permanent evolution.

EnOS(Lib) Today & Tomorrow

Tomorrow

- Monitoring by roles in detail
    - Observe and identify application patterns
    - Profiling of CPU, memory, I/O (storage and network)
- Event injection to infrastructures likely common in edge platforms

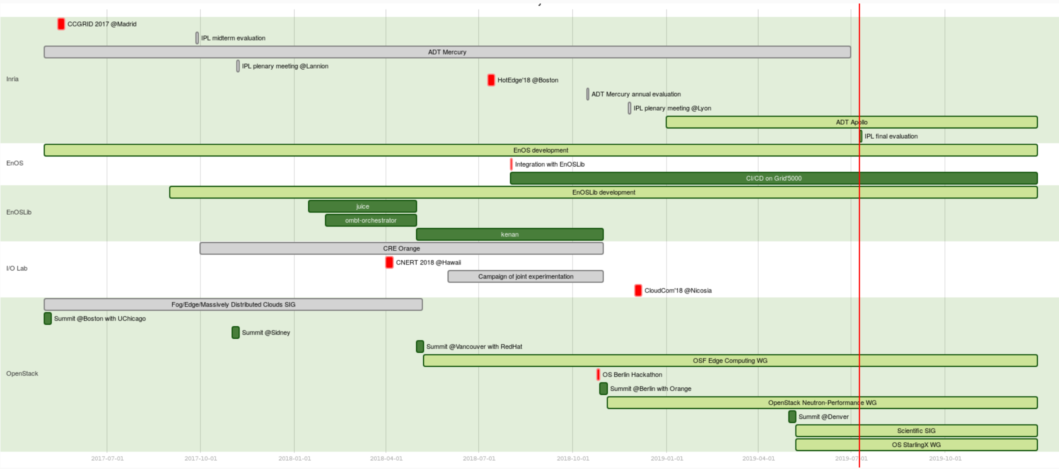
Wishlist

- Fine-grained simulation of small devices (CPU, memory or energy)
- Management of broader network configurations such as wireless and xSDL
- On-demand attachment/detachment of external nodes (servers, backups)

EnOS(Lib) Slide of Fame – Research

- Toward a Holistic Framework for Conducting Scientific Evaluation of OpenStack [Cherrueau et al., 2017]
    - CCGRID short paper
- Reproducible performance evaluations of different OpenStack deployments with EnOS [Pertin, 2017]
    - OpenStack Summit Sydney
- EnosStack: A LAMP-like stack for the experimenter [Cherrueau et al., 2018b],
    - INFOCOM Workshop
- Scalability and Locality Awareness of Remote Procedure Calls: An Experimental Study in Edge Infrastructures [Balderrama and Simonin, 2018]
    - CloudCom
- EnosLib: A library for reproducible experiment-driven research in distributed computing
    - Ongoing submission to TCC journal

Timeline Summary



References

- Balderrama, J. R. and Simonin, M. (2018).
  Scalability and locality awareness of remote procedure calls: An experimental study in edge infrastructures.
  In *2018 IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2018, Nicosia, Cyprus, December 10-13, 2018*, pages 40–47.

- Cherrueau, R., Lebre, A., Pertin, D., Wuhib, F., and Soares, J. M. (2018a).
  Edge computing resource management system: a critical building block! initiating the debate via openstack.
  In *USENIX Workshop on Hot Topics in Edge Computing, HotEdge 2018, Boston, MA, July 10, 2018*. USENIX Association.

- Cherrueau, R., Pertin, D., Simonet, A., Lebre, A., and Simonin, M. (2017).
  Toward a holistic framework for conducting scientific evaluations of openstack.
  In *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pages 544–548.

Cherrueau, R., Simonin, M., and van Kempen, A. (2018b).
Enosstack: A lamp-like stack for the experimenter.
In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 336–341.

Cherrueau, R.-A., Balderrama, J. R., and Lebre, A. (2019).
OpenStackoïd: Collaborative OpenStack Clouds On-Demand.
Presentation, OpenStack Summit, Denver, USA – `https://www.openstack.org/videos/summits/denver-2019/implementing-localization-into-openstack-cli-for-a-free-collaboration-`

Lebre, A., Cherrueau, R.-A., and Riteau, P. (2017).
Toward Fog, Edge, and NFV Deployments: Evaluating OpenStack WANwide.
Presentation, OpenStack Summit, Boston, USA – `https://www.openstack.org/videos/boston-2017/toward-fog-edge-and-nfv-deployments-evaluating-openstack-wanwide`.

Marangozova-Martin, V., El Rheddane, A., and De Palma, N. (2019).

Multi-level elasticity for data stream processing.
IEEE Transactions on Parallel and Distributed Systems, pages 1–1.

Pertin, D. (2017).
Reproducible performance evaluations of different OpenStack deployments with EnOS.
Presentation, OpenStack Summit, Sydney, Australia – `https://www.openstack.org/videos/summits/sydney-2017/reproducible-performance-evaluations-of-different-openstack-deployments`

Simonin, M., Shaposhnikov, A., and Belova, D. (2016).
Chasing 1000 nodes Scale.
Presentation, OpenStack Summit, Barcelona, Spain – `https://www.openstack.org/videos/barcelona-2016/chasing-1000-nodes-scale`.