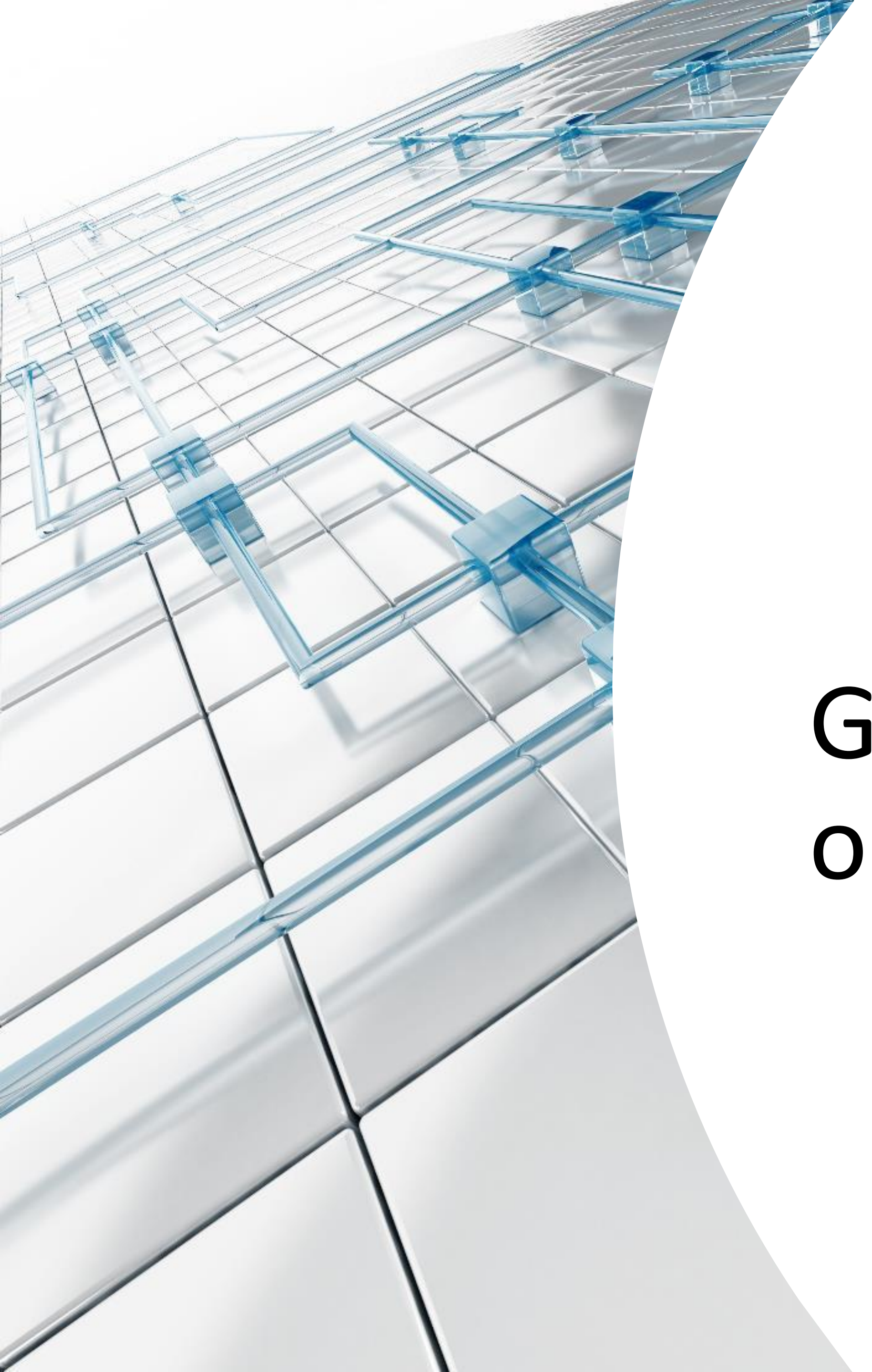
Open Source MANO by ETSI

Osmclient Hackfest Challenges

Gerardo García (Telefónica, OSM TSC Chair)

OSM#15 Hackfest

13/06/2023



Open Source MANO

General introduction to osmclient

About OSM client

- CLI tool based on Python and used to interact with OSM.
  - It interacts with OSM NBI, exposed normally in port 9999
- Click framework to model commands and options
- Pycurl library to interact with OSM NBI (REST interface)
- Python logging library to control logging (levels: INFO, VERBOSE, DEBUG)
- Presents output in pretty table format (or in yaml when supported)

Configure OSM client

- The following env vars need to be configured

```
export OSM_HOSTNAME=<OSM_HOST_IP_ADDRESS>
export OSM_USER=<OSM_USERNAME>
export OSM_PASSWORD=<OSM_PASSWORD>
export OSM_PROJECT=<OSM_PROJECT>
```

- In the Hackfest environment

```
export OSM_HOSTNAME=172.21.248.42
export OSM_USER=hackfest15groupX
export OSM_PROJECT=hackfest15groupX
export OSM_PASSWORD=hackfest15groupX
```

Typical OSM client options

- -h: to get help
  - Provides contextual help:
    - osm –help: global help
    - osm ns-list –help: help for command ns-list
- -v, --verbose: increase verbosity (-v INFO, -vv VERBOSE, -vvv DEBUG)
  - -v: logs HTTP requests and HTTP code responses
  - -vv: logs HTTP requests (URL + body) and HTTP responses (code + data)
  - -vvv: maximum verbosity; provides extra information

Install OSM client (1/2)

```
# Install dependencies
sudo apt-get update
sudo apt-get install -y libcurl4-openssl-dev libssl-dev \
        software-properties-common apt-transport-https
sudo apt-get install -y git wget make
sudo apt-get install -y python3 python3-setuptools python3-dev python3-pip
git clone https://osm.etsi.org/gerrit/osm/osmclient

# Upgrade pip to the latest version (with sudo, to install it globally for all users)
sudo -H python3 -m pip install -U pip

# Clone osmclient
git clone https://osm.etsi.org/gerrit/osm/osmclient
```

Install OSM client (2/2)

```
# Install osmclient directly from the repo for development purposes
python3 -m pip install --user -e osmclient \
    -r osmclient/requirements.txt -r osmclient/requirements-dev.txt

# Patch pyangbind
wget https://osm.etsi.org/gitlab/osm/im/-/raw/master/pyangbind.patch
patch ~/.local/lib/python3.10/site-packages/pyangbind/lib/yangtypes.py <pyangbind.patch

# Install charm to be able to build OSM packages with charm
sudo snap install charm --classic

# Logout and login so that PATH can be updated.
# Executable osm will be found in /home/ubuntu/.local/bin
which osm
```

Testing code

- Every commit in OSM is validated in a Jenkins CI/CD pipeline:
  - Test code format and linting for the change
  - Build docker images and run sanity tests
- To test code format and code linting:

```
./devops-stages/stage-test.sh
```

E2E Testing with Robot framework

- Configure envconfig.rc

```
OSM_HOSTNAME=<OSM_IP_ADDRESS>
OSM_USER=<OSM_USER>
OSM_PASSWORD=<OSM_USER_PASSWORD>
OSM_PROJECT=<OSM_IP_ADDRESS>
VIM_TARGET=<VIM_REGISTERED_AT_OSM>
VIM_MGMT_NET=<NAME_OF_THE_MGMT_NETWORK_IN_THE_VIM>
```

- To test hackfest_basic testsuite

```
docker run --rm=true --name tests -t --env-file envconfig.rc \
           -v ~/reports:/robot-systest/reports \
           opensourcemano/tests:testing-daily \
           -t hackfest_basic
```

Manual E2E Testing

```
git clone --recursive https://osm.etsi.org/gitlab/vnf-onboarding/osm-packages.git
osm vim-list
osm nfpkg-create osm-packages/hackfest_basic_vnf
osm nspkg-create osm-packages/hackfest_basic_ns
osm nfpkg-list
osm nspkg-list
osm ns-create --ns_name hfbasic --nsd_name hackfest_basic-ns --vim_account
<VIM_GROUP_X> --config '{vld: [ {name: mgmtnet, vim-network-name: osm-ext} ] }
osm ns-list
osm vnf-list --filter member-vnf-index-ref=vnf --ns <NS_ID>
osm ns-delete hfbasic
osm nspkg-delete hackfest_basic-ns
osm nfpkg-delete hackfest_basic-vnf
```

Osmclient Hackfest Challenges
Steps to be followed

Osmclient challenges

- Objective: to update OSM Python client to add new capabilities or improve maintenance
- 4 challenges
  - **Challenge 1**. Global option `-o <FORMAT>` in osmclient to adapt output format
  - **Challenge 2**. Global option `-c` in osmclient to filter output and show only selected fields in xxx-show and xxx-list commands
  - **Challenge 3**. Deprecation framework for old commands in osmclient
  - **Challenge 4**. Replace pycurl library in osmclient by requests library
- Difficulty: low to medium
- They will allow to get a general picture of how OSM work and how to interact with OSM from an external system (in this case, OSM client)

DISCLAIMER

Challenges have been designed, trying to take into account all the steps required to implement them.

However, since this is new code to be done, unexpected steps might be necessary

Hackfest Challenges

TITLE	MODULES	MENTORS	DIFFICULTY
Challenge 1. Global option \`-o <FORMAT>\` in osmclient to adapt output format	osmclient	Gerardo	LOW
Challenge 2. Global option \`-c\` in osmclient to filter output and show only selected fields in xxx-show and xxx-list commands	osmclient	Gerardo	LOW
Challenge 3. Deprecation framework for old commands in osmclient	osmclient	Gerardo	LOW
Challenge 4. Replace pycurl library in osmclient by requests library	osmclient, devops (installer)	Gerardo	MEDIUM
Challenge 5. Remove the duplicated information from VDUR in VNFR for Vertical-scaling operation	RO	Gulsum	MEDIUM
Challenge 6. NS Delete Operation should raise error if the deletion operation is not completed successfully	RO	Gulsum	MEDIUM
Challenge 7. Remove EE Charms when VNF has only day-1 operations	LCM, N2VC	Guillermo	HIGH
Challenge 8. Vulnerability fix of bug 2088 regarding Remote Code Execution in N2VC/LCM	N2VC, NBI, LCM	Pedro	MEDIUM/HIGH

Challenge 1. Global option in osmclient to adapt output format

- Description
  - A new global option `-o <FORMAT>` will be added to osmclient to adapt output format
  - The output formats could be: table, csv, yaml, json, jsonpath (default:table)
    - Reference example (kubectl): https://medium.com/codex/kubectl-output-101-851f8e61fd51
  - It would only apply to xxx-show and xxx-list commands
  - The code should be properly written so that all commands use the same call in all the cases: *print_output(format, header, rows)*
- Definition of Done
  - All sanity tests should pass. Nothing is broken
  - All the xxx-show and xxx-list for NS, VNF and VIM should work with all the format options
  - (Optional). A new Robot test is added or an existing one is updated to check all the options work for a specific osmclient command

Challenge 1. Global option in osmclient to adapt output format

Setup: clone the repo, run devops-stages/stage-test.sh	Everything passes. Environment is ready
Getting familiar. Run osmclient commands to test its current behaviour	osm ns-list displaying output
Add global option "-o <FORMAT>", so that the code does not break. The global option must work only for a group of commands	osm ns-list without -o option keeps working
First command: osm ns-list -o table 1. Identify the code in osmclient/cli_commands/ns.py that is related to printing the output 2. Create new function print_output(format, header, rows) in osmclient/common to print the output in different formats 3. Change the code in osmclient/cli_commands/ns.py to use that common function print_output	osm ns-list and osm ns-list -o table prints the output in table format
Add new format (yaml) : osm ns-list -o yaml	osm ns-list -o yaml prints the output in yaml format
Add new format (json) : osm ns-list -o json	osm ns-list -o json prints the output in json format
Add new format (csv) : osm ns-list -o csv	osm ns-list -o csv prints the output in csv format

Challenge 1. Global option in osmclient to adapt output format

Add support of option "-o" to command osm ns-show	osm ns-show -o yaml prints the output in yaml format
Add support of option "-o" to command osm vnf-list	osm vnf-list -o csv prints the output in csv format
Add support of option "-o" to command osm vnf-show	osm vnf-show -o json prints the output in json format
Add support of option "-o" to command osm vim-list	osm vim-list -o yaml prints the output in yaml format
Add support of option "-o" to command osm vim-show	osm vim-show -o csv prints the output in csv format
Deprecate "--literal" option in favor of "-o yaml" option	osm xxx-show --literal should give a warning, and internally should call to print_output function
Add new format (jsonpath) : osm ns-list -o=jsonpath=…	osm ns-list -o=jsonpath='{@}' osm ns-list -o=jsonpath='{.items[0].id}' prints the desired output

Challenge 2. Global option in osmclient to filter output and show only selected fields

- Description
  - A new global option `-c <FORMAT>` will be added to osmclient to filter output, based on first-level fields
  - Reference example (openstack): `openstack server list -c Id`
  - The argument “-c” must be stackable, e.g. `-c Name –c Id`
  - It would only apply to xxx-show and xxx-list commands
  - The code should be properly written so that all commands use the same call in all the cases: *filter_output(fields, dict)*
- Definition of Done
  - All sanity tests should pass. Nothing is broken
  - All the xxx-show and xxx-list for NS, VNF and VIM should work with the new option
  - (Optional). A new Robot test is added or an existing one is updated to check that the new option works for a specific osmclient command

Challenge 2. Global option in osmclient to filter output and show only selected fields

Setup: clone the repo, run devops-stages/stage-test.sh	Everything passes. Environment is ready
Getting familiar. Run osmclient commands to test its current behaviour	osm ns-list displaying output
Add global option "-c <FIELD>", so that the code does not break. The global option must work only for a group of commands. It must be stackable	osm ns-list without -c option keeps working
First command: osm ns-list -c Id 1. Identify the code in osmclient/cli_commands/ns.py that is related to getting the object from OSM and the code related to printing the output. The filtering code must be placed between both. 2. Create new function filter_output(fields, dict) in osmclient/common to print the output in different formats 3. Change the code in osmclient/cli_commands/ns.py to use that common function filter_output	osm ns-list -c Name prints a table with the NS id
Add support of option "-c" to command osm ns-show	osm ns-show -c Name -c Id prints only the name and id of the NS
Add support of option "-c" to command osm vnf-list	osm vnf-list -c Name prints a table with the VNF id

Challenge 2. Global option in osmclient to filter output and show only selected fields

Add support of option "-c" to command osm vnf-show	osm vnf-show -c Id prints only the id of the VNF
Add support of option "-c" to command osm vim-list	osm vim-list -c Name -c Id prints only the name and id of the VIM
Add support of option "-c" to command osm vim-show	osm vim-show -c Id prints only the id of the VIM
Explore its applicability to other commands	New feature in OSM, together with design document

Challenge 3. Deprecation framework for old commands in osmclient

- Description
  - A deprecation framework will be defined and implemented in osmclient
  - Some commands will be selected to be deprecated in favor of new commands (or no command) for a given release.
  - The code should be properly written so that all commands check if they are deprecated, ideally without defining that code on a per-command basis
- Definition of Done
  - All sanity tests should pass. Nothing is broken.
  - The deprecation framework is implemented and documented.
  - A selected number of commands (e.g. vnfd-xxx, nsd-xxx) will give a warning.
  - A new Robot test is added or an existing one is updated to check that a deprecated command shows a warning

Challenge 3. Deprecation framework for old commands in osmclient

Setup: clone the repo, run devops-stages/stage-test.sh	Everything passes. Environment is ready
Getting familiar. Run osmclient commands to test its current behaviour	osm ns-list displaying output
Capture the challenge as a feature in Gitlab	New feature in OSM Gitlab
Define deprecation framework, including: 1. Unified deprecation message 2. Structure to manage deprecation commands	Etherpad with the design of the feature in Markdown format, specifying the deprecation framework
Implement deprecation framework: 1. Create structure to manage deprecation commands 2. As an example, include in the structure the command vnfd-list to be deprecated in Rel SIXTEEN by the method nfpkg-list. 3. Create new function check_deprecation(cmd_name) in osmclient/common to check if a command has been deprecated 4. Make sure that all commands call the method check_deprecation	osm vnfd-list should give the expected warning

Challenge 3. Deprecation framework for old commands in osmclient

Find a Robot Test using the command vnfd-list and run it	A Robot Test should be executed successfully
Update the Robot Test to do one call to vnfd-list and another call to nfpkg-list	The test should check that first call gives a warning, and that the second call doesn't give a warning. Both calls should present the same result.

Challenge 4. Replace pycurl lib in osmclient by requests lib

- Description
  - The library pycurl is currently used to interact with OSM North Bound Interface (NBI)
  - The library requests is widely used
  - This challenge will replace all calls using to pycurl by requests calls
- Definition of Done
  - All sanity tests should pass. Nothing is broken.
  - Osmclient will work without pycurl as a requirement
  - Devops installer will work without installing libcurl4-openssl-dev and libssl-dev

Challenge 4. Replace pycurl lib in osmclient by requests lib

Setup: clone the repo, run devops-stages/stage-test.sh	Everything passes. Environment is ready
Getting familiar. Run osmclient commands to test its current behaviour. Use the options -v, -vv and –vvv	osm ns-list displaying output
Update HTTP Read operations 1. Identify the methods in \`common/http.py\` and \`sol005/http.py\` dealing with GET requests 2. Update those methods to use requests instead of pycurl	osm ns-list displaying output
Update HTTP Create operations 1. Identify the methods in \`common/http.py\` and \`sol005/http.py\` dealing with POST requests 2. Update those methods to use requests instead of pycurl	osm ns-create working
Update HTTP Delete operations 1. Identify the methods in \`common/http.py\` and \`sol005/http.py\` dealing with DELETE requests 2. Update those methods to use requests instead of pycurl	osm ns-delete working
Update HTTP Put operations 1. Identify the methods in \`common/http.py\` and \`sol005/http.py\` dealing with DELETE requests 2. Update those methods to use requests instead of pycurl	osm vim-update working (or any command relying on HTTP PUT)

Challenge 4. Replace pycurl lib in osmclient by requests lib

Setup: clone the devops repo, run devops-stages/stage-test.sh	Everything passes. Environment is ready
Getting familiar. Find references to libcurl4 in devops repo	At least it must be referenced in several Dockerfiles and in full_install_osm.sh
Update Dockerfiles using libcurl4	The docker images should be built
Push the change to gerrit and check Jenkins validation	Jenkins should pass successfully

Additional information

Relevant links

- OSM client in OSM user guide: https://osm.etsi.org/docs/user-guide/latest/10-osm-client-commands-reference.html?highlight=osm_project#annex-2-reference-of-osm-client-commands-and-library

How to build a docker image of a component as it is done in Jenkins CI/CD pipeline (1/2)

- Build your project as it is done in Jenkins
- Clone devops repo at the same level that your project

```
git clone ssh://${MY_EOL}@osm.etsi.org:29418/osm/devops.git
(cd "devops" && curl
https://osm.etsi.org/gerrit/tools/hooks/commit-msg >
.git/hooks/commit-msg ; chmod +x .git/hooks/commit-msg)
```

- Start HTTP server

```
./devops/tools/local-build.sh --run-httpserver
```

- Build the debian packages associated to your project (e.g. NBI) and its dependencies

```
rm -f ~/.osm/httpd/*.deb
./devops/tools/local-build.sh --module NBI,IM,common stage-2
```

How to build a docker image of a component as it is done in Jenkins CI/CD pipeline (2/2)

- Build Docker image

```
./devops/tools/local-build.sh --module NBI stage-3
```

- Patch the image of the component

```
kubectl -n osm patch deployment nbi --patch '{"spec":
{"template": {"spec": {"containers": [{"name": "nbi", "image":
"opensourcemano/nbi:devel"}]}}}}'
kubectl -n osm scale deployment nbi --replicas=0
kubectl -n osm scale deployment nbi --replicas=1
```

Open Source MANO by ETSI

Thank You!