redhat.

How to Effectively Monitor, Analyze and Troubleshoot OpenDaylight

An ODLTools Tutorial

Vishal Thapar,
Principal Software Engineer

Agenda

- Story so far
- First few steps
- The Toolset
- Playtime
- Next steps

redhat.

Story so far

Evolution

- Shell scripts for easier debugging of ODL flows
- Move to Python for more complex analysis
- Code shared over ftp
- Code added to Github as odltools
- Devs added their own custom scripts
- Project added to OpenDaylight Flourine as ODLTools
- Used in Netvirt and Genius CSIT
- Work in progress

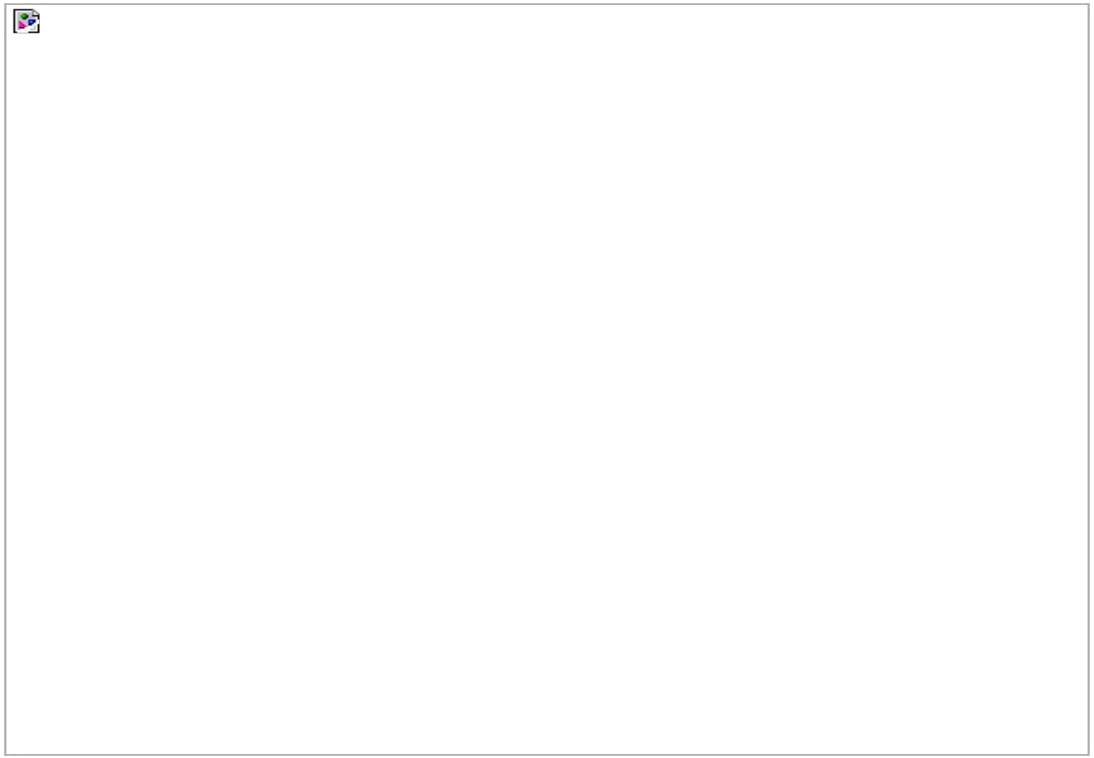
- Sam Hague, RedHat (shague)
- Josh Hershberg, RedHat (jhershberg)
- Tim Rozet, RedHat (trozet)
- Faseela K, Ericsson (faseelak)

ODL Tools

- Troubleshoot and monitor live systems
- Offline analysis of logs and data dumps collected
- Collect logs and dumps from live systems for offline analysis
  - ODL Datastore dumps
  - Karaf [ODL] logs
  - *OVS Dumps [flows, groups etc.]*
- More a collection of tools than individual tools

OpenDaylight and OpenStack

- OpenStack Neutron
  - networking-odl
  - ODL ML2 Driver
- OpenDaylight
  - Neutron Northbound
  - NetVirt
  - MD-SAL
  - Southbound Protocols
    - OVSDB
    - OpenFlow



redhat.

First few steps

Get Ready

- Python2 or Python3 installed
- Pull and install odltools
  - `git clone https://git.opendaylight.org/gerrit/odltools`
  - `python setup.py install`
- Install from pip: `pip install odltools`
- Verify installation with odltools -h or `odltools -V`
- Get netvirt csit log file:
  - Go to https://jenkins.opendaylight.org/releng/view/netvirt-csit/
  - Select a 3node job with multiple nodes
  - Get any of the output_* files from robot-plugin
  - e.g. ‘`wget https://logs.opendaylight.org/releng/vex-yul-odl-jenkins-1/netvirt-csit-3node-0cmb-1ctl-2cmp-openstack-queens-upstream-stateful-snat-conntrack-oxygen/67/robot-plugin/output_01_l2.xml.gz`’

Get Ready (Optional)

- Java 8 installed
- Get OpenDaylight distribution
  - `https://docs.opendaylight.org/en/stable-fluorine/downloads.html`
- Run ODL and install netvirt
  - https://docs.opendaylight.org/en/stable-fluorine/getting-started-guide/installing_opendaylight.html
  - `feature:install odl-netvirt-openstack`

Play Around


```
odltools -h
usage: python -m odltools [-h] [-v] [-V]
{csit,karaf,model,monitor,analyze,show} ...

OpenDaylight Troubleshooting Tools

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         verbosity (-v, -vv)
  -V, --version         show program's version number and exit

Subcommands:
  Command Tool
  {csit,karaf,model,monitor,analyze,show}
```

redhat.

The Toolset

ODL Tools

- Monitoring
- Show
- Analyze
- Incubator
  - OVS Replay

redhat.

Playtime

Monitor


```
odltools monitor -h
usage: python -m odltools monitor [-h] [-d {Config,Operational}] cluster.json

Graphical tool for monitoring an OpenDaylight cluster
positional arguments:

  cluster.json          JSON Cluster configuration file in the following
                        format: { "cluster": { "controllers": [ {"ip":
                        "172.17.10.93", "port": "8181"}, {"ip":
                        "172.17.10.93", "port": "8181"}, {"ip":
                        "172.17.10.93", "port": "8181"} ], "user": "username",
                        "pass": "password", "shards_to_exclude": ["prefix-
                        configuration-shard"] } }

optional arguments:
  -h, --help            show this help message and exit
  -d {Config,Operational}, --datastore {Config,Operational}
                        polling can be done on "Config" or "Operational" data
                        stores
```

Show


```
odltools show -h
usage: python -m odltools show [-h]


{elan-instances,flows,id-pools,groups,stale-bindings,tables,neutron,eos}
                                ...

positional arguments:
  {elan-instances,flows,id-pools,groups,stale-bindings,tables,neutron,eos}

optional arguments:
  -h, --help            show this help message and exit
```

Analyze


```
odltools analyze -h
usage: python -m odltools analyze [-h] {interface,inventory,nodes,trunks} …

positional arguments
  {interface,inventory,nodes,trunks}


optional arguments:
  -h, --help            show this help message and exit
```

Others

```
odltools model -h
usage: python -m odltools model [-h] {get} …
Tools for MDSAL models

optional arguments:
  -h, --help  show this help message and exit

Subcommands:
  Model tools
  {get}
    get       Get and write all mdsal models
```

```
odltools karaf -h
usage: python -m odltools karaf [-h] {format} ...

Karaf log tools

optional arguments:
  -h, --help  show this help message and exit
Subcommands:
  Karaf tools
  {format}
    format    Dump a karaf log with pretty printing of MDSAL
objects
```

```
odltools csit -h
usage: python -m odltools csit [-h] [-g] [-d] infile path

positional arguments:
  infile        XML output from a Robot test, e.g. output_01_l2.xml.gz
  path          the directory that the parsed data is written into

optional arguments:
  -h, --help    show this help message and exit
  -g, --gunzip  unzip the infile
  -d, --dump    dump extra debugging, e.g. ovs metadata
```

OVS Replay

- Recreate OVS instance from offline dumps
- Only one instance currently supported
- Incubator project - not pushed to pypi

redhat.

Next Steps

Our wishlist for ODLTools

- Make odltools a library so others can reuse in their own code
- Add support for rest interface to ODLTools
- Add any analysis gaps e.g. comparing ODL and OVS flows

Your wishlist for ODLTools

- Raise feature request at ODL Tools Jira
  - https://jira.opendaylight.org/projects/ODLTOOLS
- Write your code and contribute
-

Links

- https://wiki.opendaylight.org/view/ODLTools
- https://jira.opendaylight.org/browse/ODLTOOLS
-

Missing co-conspirators

Thank You