HowTo use the

VTN-based RTM Network Service

(version 0.9)

The provided code contribution is taken from a deployment in 3 separate virtual machines.  
For a better understanding of the scenario and the deployment setup please review the accompanying slide set. Available at: <https://community.opensourcesdn.org/wg/Media_NBI/document/51>

Add a proper list of requirements and packages (incl. versions) this contribution depends upon.

**List of Virtual Machines:**

1. Main Virtual Machine
   1. IP address: 192.168.56.101
   2. Runs: mininet, RTM service, VLC
2. Controller for “non-SDN” controlled switch in the middle of topology.
   1. IP address: 192.168.56.107
   2. *E.g. vtn-554a337/manager/dist/target/distribution.vtn-manager-0.3.0-SNAPSHOT-osgipackage/opendaylight/ ; sudo ./run.sh*
3. SDN-controller: Used to run VTN Coordinator
   1. IP address: 192.168.56.102
   2. Restart VTN Coordinator: execute /usr/local/vtn/bin/vtn\_stop and then /usr/local/vtn/bin/vtn\_start to start the daemon

Make sure that the VM’s have the appropriate IP Addresses. If not then the Mininet topologies will be unable to connect to the correct SDN Controllers

**Steps:**

* Start the Virtual Machines
* Start RTM Service
  1. cd /rtm-api
  2. Before running the service make sure that the config file contents comply with your virtual network configuration. (see section below)
  3. Run the service using python manage.py runserver
* Start vlc-sdn-generator script (VLC wrapper, holds the app’s user credentials)
  1. cd vlc-logs
  2. sudo ./vlc-sdn-generator.sh
* Run the mininet topology

**Note:** If this is the first time that you are running the service you should:

* Setup and install flask (should be extended soon)
* Setup a database for the network service (needs flask setup and running)
  1. cd /rtm-api
  2. . env/bin/activate
  3. If this the first time do the following:
     1. python manage.py createdb
     2. python manage.py adduser <username>  
        (RTM service user application need to authenticate with these credentials, current default root:root)
     3. Add password for your user
* Add all the dscp mappings to the interface before using sessions and/or policies.

**VTN-Based RTM Service Configuration:**

In directory /rtm-api there is a file named config that you can modify to change the behavior of the VTN-based RTM Service. More specifically you can:

1. Specify the attributes of the SDN controller:
   1. type 🡪 Controller Type. Valid values “odl” for OpenDaylight (currently provided mininet configuration) and “pfc” for NEC’s ProgrammableFlow controller.
   2. ipAddress 🡪 Controller Ip Address (in case of “odl” specify the IP of VTN Manager and not VTN Coordinator)
   3. webApiPort 🡪 SDN Controller WebApi Port (default: 8080 for both controllers)
   4. webApiUserName 🡪 Web Api Username (in case of odl “admin”, in case of pfc “webapi”)
   5. webApiPassword 🡪 Web Api password (in case of odl “admin”, in case of pfc “webapi”)
2. Specify how many SDN controllers you have in your test topo (controller1 or/and controller2 element). If you want to use one controller delete the controller2 element
3. Define which markings you want to use (“priority” for vlan priority (OpenDaylight) and “dcsp” for dscp)

**Mininet Topology:**

1. Multiple\_vlan.py: To run the topology you have to specify an argument that will define if the VTN creation should take into account all the hosts or the hosts that are connected to the switch ports that are specified in the topo config file (arguments –all and --file).

**Note:** The above Mininet script configures the VTN topology and queues configuration automatically. You can specify the topology configuration using the input file topo\_config. With this file you can specify the number of edge switches (giving the name is equals to assigning dpid. For example s2 will get the dpid 0000-0000-0000-0002). Also you can specify the number of hosts that are connected to each switch and whether the link with the core of the network is limited and with how much bandwidth. Furthermore you have the flexibility to define the list of ports that you want to be mapped in the vtn topology . Note that you have to run the topology with –file argument to take into consideration the specified ports.

**Appendix:**

1. Running VLC from Mininet Hosts
   1. First of all decide which hosts are going to get involved to the video stream. For information about the topology you can use the command “net” while you are in the mininet CLI.
   2. Spawn xterm for each host you want using mininet CLI (“xterm h1” for example)
   3. First you have to specify the command for the receiver of the VLC Stream. The command is “vlc-wrapper udp://@:11223”. Note that with this command we only specify the protocol and the port number.
   4. In the host that will be sending the VLC stream the command that we used “vlc-wrapper /path/to/Big-Buck-Bunny-sd.mp4 –sout=udp/ts://10.0.0.4:11223 –extraintf logger –verbose=2 –logfile path/to/vlc-logs/vlc.log”. With this command we specify which file to stream, which is the destination host and port and the log file. Note that for the VLC Wrapper that we implemented to work the log file directory have to be the same with the above command (the name of the file doesn’t matter).
2. Testing RTM Service using POSTMAN Rest Client (*Chromium only*)
   1. Open chromium browser in Ubuntu 14.04
   2. Click the apps icon (top left toolbar)
   3. Click Postman-REST Client
   4. Click collections button
   5. From there you can see ready examples for each operation that the RTM Service Provides. We have created three collections. One for the Session Operations, one for the Policies Operations and one for the DSCP mappings operations.