Author Details

Mani Prashanth Varma Manthena
M.Sc. in Electrical Engineering - Telecommunications Track (2012 - 2014)
Faculty of EEMCS
TU Delft
Graduate Intern (2013 - 2014)
Service Enabling and Management Department
TNO - Delft
Email: me@prashanthvarma.com, M.P.V.Manthena@student.tudelft.nl
Ph. No.: +31684401806

SNMP Web Application

SNMP Web Application is a custom built SNMP based network interface monitoring application, it exposes network interface failure and high bandwidth utilization events with configurable utilization thresholds via its REST API to the applications running on top of it. This web application was built as a part of the Proof of Concept (PoC) for my M.Sc. thesis/graduation project at TU Delft and TNO. My M.Sc. thesis/graduation project involved proposing and validating an evolutionary approach to NaaS architecture with SDN and NFV for service provider networks. In Figure 1, my PoC NaaS architecture is shown. As it can be seen in the Figure 1, in one of the scenarios (i.e. Option 1 : Testbed Setup A), NaaS Platform interacts with this application (i.e. SNMP Web Application) through REST API in my PoC.

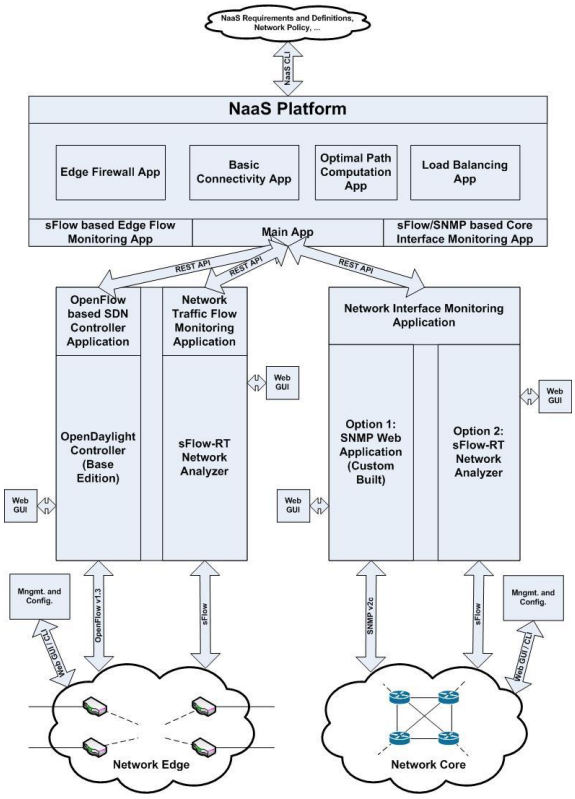

Figure 1. PoC NaaS Architecture

PoC NaaS Architecture Description

Note: This PoC is implemented mainly in UNIX/LINUX (i.e. Ubuntu) based systems/VMs. However, it can also be implemented in other OS environments without any changes to the PoC - Source Code. Further, all the required applications and dependencies for the PoC can either be implemented in a single system/VM (i.e. refer PoC - VM) or in multiple systems/VMs (i.e. in the above PoC NaaS architecture, each block along with its external interfaces represents a single system/VM) for scalability reasons.

PoC NaaS Architecture - NaaS Platform and Virtualized Network Functions:

- Basic Connectivity
- Optimal Path Computation
- Load Balancing
- Edge Firewall

PoC NaaS Architecture - OpenFlow based SDN Controller Application:

- OpenDaylight Controller (Base Edition)

PoC NaaS Architecture - Network Edge Traffic Flow Monitoring Application:

- sFlow-RT Network Analyzer

PoC NaaS Architecture - Network Core Interface Monitoring Application:

- Option 1 (e.g. Testbed Setup A): Custom built SNMP Web Application
- Option 2 (e.g. Testbed Setup B): sFlow-RT Network Analyzer

PoC - Testbed Setups:

- Testbed Setup A: Proposed solution with legacy switches at the network core
- Testbed Setup B: Proposed solution with OpenFlow/Open vSwitch enabled switches at the network core

Proof of Concept (PoC) Resources:

- For my Proof of Concept (PoC) - Architecture, go to the folder PoC - NaaS Architecture
- For my Proof of Concept (PoC) - Source Code, go to the folder PoC - Source Code
- For my Proof of Concept (PoC) - Testbed Setups, go to the folder PoC - Testbed Setups
- For my Proof of Concept (PoC) - VM, go to the folder PoC - VM
- For my Proof of Concept (PoC) - References, go to the folder PoC - References

SNMP Web Application Contents

- Main Application
- Interface Status Monitoring Applications
- Interface Utilization Monitoring Applications
- Configuration and log files

Note: As it can be clearly seen from the SNMP Web Application's source code, this application is written for network core interface monitoring of my PoC - Testbed Setup A. However, If you want to use this application for other testbed setups (i.e. custom testbed setups), you need to make some changes to the scripts and the

configuration files in the source code of this application. Information regarding these required changes to the source code scripts and files is mentioned in the section "Custom Testbed Setups" at the end of this document.

Getting Started

Note: This application is implemented mainly in UNIX/LINUX (i.e. Ubuntu) based systems/VMs. However, it can also be implemented in other OS environments without any changes to the application's source code. This application is completely written in Python 2.7, due to its simplicity, interoperability, support, and platform agnostic nature. Thus, the NaaS Platform will run in all the systems/VMs with Python version 2.x.

Step 1) Installing the required dependencies

- Requires Python version 2.x, this application was tested and implemented with Python version 2.7.

  Installing Python pip and other dependencies in Ubuntu, Python pip is a tool for installing and managing python packages. Ubuntu terminal command for installing Python pip tool:

  $ sudo apt-get install python-pip python-dev build-essential

  $ sudo pip install --upgrade pip

  $ sudo pip install --upgrade virtualenv

- As this application involves simultaneously executing several python scripts at the start of the application, the system/VM which hosts this application mush have the xterm terminal emulator installed in it. Ubuntu terminal command for installing xterm terminal emulator:

  $ sudo apt-get install xterm

- Requires a Python package called Bottle for enabling RESTful web services in the application. Bottle is a fast, simple and WSGI micro web-framework for Python. Ubuntu terminal command for installing Bottle package:

  $ sudo pip install bottle

- Requires a Python package called PySNMP for SNMP based network interface monitoring in the application. PySNMP is a cross-platform, pure-Python SNMP engine implementation. Ubuntu terminal command for installing PySNMP package:

  $ sudo pip install pysnmp

- Additionally, you might want to install the Net-SNMP software suite in your system/VM for performing SNMP based operations from your terminal window. Ubuntu terminal command for installing Net-SNMP software suite:

  $ sudo apt-get install snmp

Step 2) Configuring the SNMP Web Application

- At first, check whether the SNMP protocol is configured properly in the underlying network devices (i.e. legacy MPLS LSRs). You can verify this by perform snmpwalk on all the underlying network devices in the terminal window.

- Copy the SNMP Web Application source code to the system/VM

- Configure the Ip address and port number of the SNMP Web Application, this can be done by editing the last line of the SNMP_Main_App Python script in the application's source code.

  Last line of the SNMP_Main_App Python script:

  run(host='localhost', port=8090, debug=True)

  In the above line, replace 'localhost' with the IP address of the system/VM hosting the SNMP Web Application . Further, you can also change the port number.

  Example configuration:

  run(host='192.63.245.211', port=8090, debug=True)

Step 3) Installing and launching the SNMP Web Application

- Before launching the SNMP Web Application, check whether the SNMP protocol is configured properly in the underlying network devices (i.e. legacy MPLS LSRs).

- After copying the SNMP Web Application source code to the system/VM, open a new terminal window and go (i.e. "cd") to the application's main directory, where you need to enter "./start.sh" with administrator privileges to launch the SNMP Web Application.

  $ sudo ./start.sh

  Note: The "./start.sh" command launches all the required scripts in the source code of the SNMP Web Application

- After launching the OpenDaylight controller, verify the controller's operational status by navigating to its web GUI.

  Default URL:
  http://*snmp-ip***:8090**

  Note: "*snmp-ip*" in the above URL is the IP address of the system hosting the SNMP Web Application, this URL is also the base URL for the SNMP Web Application's REST API.

Custom Testbed Setups

If you want to implement this SNMP We Application for custom testbed setups (i.e. other than the PoC - Testbed Setup A), you need to make some changes to the scripts and the configuration files in the source code of this application.

- At first, based on your underlying network devices you need to replace the existing interface status and utilization monitoring python scripts.

  For this, just copy the contents of the existing scripts (e.g. BA_Interface_Status_Monitoring_App and BA_Interface_Utilization_Monitoring_App) while making the necessary configuration changes to it (e.g.

switch IP address inside the script, SNMP community name, etc.) and place them in the application source code main directory.

- Further, as per your testbed network devices, edit the configuration files (i.e. agent names, device interfaces, interface MAC addresses) in the Config folder of the application's source code. These configuration files are in simple and readable JSON format.
  For this, you can gather network devices information by performing snmpwalk on all the underlying network devices in the terminal window by using the corresponding SNMP OIDs.

- Finally, edit the startup script (i.e. start.sh) to run the replaced network interface status/utilization monitoring scripts by including their respective names in it.

Final Note

For further information and references relating to the SNMP Web Application, go to the folder PoC - References\SNMP based Network Interface Monitoring or contact me via my email id.