# DOCUMENTATION FOR DUILDING THE CODE AND EXECUTING IT.

**Project** – Monitoring automata
**Author** – Mamatha Mallikarjuna Jaldiranna
**Department** – DSP

## #Install ONL OS on your switch -

- reboot the switch using **$ sudo reboot** command.
- Don't press any key for any system setup when it asks. It asks twice. Don't do anything.
- If no OS - it enters ONIE directly. ELse it gives options 1.ONL (or any other OS Installed) 2. **ONIE**
- To install a new OS, select **ONIE option** and go **ONIE-Rescue** option.

**Through USB stick - I DID THIS. IT WORKS.**

- Copy the installer file into USB drive. Connect it to the switch (NOT PC)
- **$mkdir /mnt/media** - make this new directory to mount your installer file and install.
- **$mount /dev/sdb /mnt/media** - mount the device. Sometimes it might be other than sdb, like sdb1 ..etc. So check it. to check you can use **$sudo fdisk -l** //This lists all the devices and you can see which is yours. Also sometimes the file might be corrupted for that you need to run this – Its always better to run this – **********
- run this command - **$ onie-nos-install /mnt/media/installer_file_name**
  This will install the OS and automatically reboot the system. IN the menu it will show 1.ONL 2.ONIE

**Through link - internet-**
use ONIE to install ONL installer (one time installation for a device which has no ONL on the device yet or you need functions that only exist in latest ONL)

**ONIE:/ # wget http://opennetlinux.org/binaries/latest-DEB8-AMD64-installed.installer**
**ONIE:/ # sh latest-DEB8-AMD64-installed.installer**

For PowerPC:

**ONIE:/ # wget http://opennetlinux.org/binaries/latest-DEB8-PPC-installed.installer**
**ONIE:/ # sh latest-DEB8-PPC-installed.installer**

For Armel:

**ONIE:/ # wget http://opennetlinux.org/binaries/latest-DEB8-ARMEL-installed.installer**
**ONIE:/ # sh latest-DEB8-ARMEL-installed.installer**

(For AMD64 or PPC, you can use latest-DEB7-xxx-installed.installer if you wand.)

If your unit can't connect to Internet, you can download its installer file.
Save it in tftp server, then use below command to install in ONIE rescue:
ONIE:/ # install_url tftp://192.168.1.1/latest-DEB8-xxx-installed.installer
(You must configure management port first, example 192.168.1.3 and TFTP server IP is 192.168.1.1)

# # After installing ONL - internet was not working.

1**. sources.list** file was missing from /etc/apt folder. ITs a file with default servers to download softwares from.
So created it with default values -
**deb http://httpredir.debian.org/debian jessie main**
**#deb-src http://httpredir.debian.org/debian jessie main**

**deb http://httpredir.debian.org/debian jessie-updates main**
**#deb-src http://httpredir.debian.org/debian jessie-updates main**

**deb http://security.debian.org/ jessie/updates main**
**#deb-src http://security.debian.org/ jessie/updates main**
.....
This is not defaut - it was added by me from some website.
**deb http://ftp.us.debian.org/debian/ jessie main contrib non-free**

There should be a copy (of your distribution) in here:
**/usr/share/doc/apt/examples/sources.list** --- but this file is also not there.

2. I was not able to access internet even after doing all this shit.
**$ Sudo apt-get update** --and -- **$ ping google.com** was also giving an error. But **$ping 8.8.8.8** was working. So the problem was with **DNS name resolution**. AFter adding the below content to the file everything worked out fine.

**resolv.conf file from /etc/resolv.conf** - it was empty for us. But it shouldnt be.
Add these content to the file : Also it becomes empty everytime the switch reboots so have to make sure it doesn't do that. Else rewrite it everytime.
content of /etc/resolv.conf file -
.....
**# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)**
**# DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN**

**nameserver 208.67.222.222**
**nameserver 208.67.222.220**
**nameserver 192.168.0.254**
**nameserver 8.8.8.8**
**nameserver 8.8.4.4**
.....
//8.8.8.8 belongs to google.

This solved the issue and all was good.

.....

## # Installing openNSL - on switch

- **copy opennsl-accton_3.2.0.5+cdp+accton1.0-1_amd64.deb** file to **mnt/onl/data** folder on the switch. - same file works for both switches.
  **$ scp /home/mamtha/Downloads/opennsl-accton_3.2.0.5+cdp+accton1.0-1_amd64.deb root@192.168.0.106:/mnt/onl/data**
- Install opennsl debian package (one time activity after power-on)
  **$ dpkg -i /mnt/onl/data/opennsl-accton_3.2.0.5+cdp+accton1.0-1_amd64.deb**
- run **$opensl_setup**
  you will get something like this **- linux-kernel-bde installed**.
- Sometimes you dont get anything, execute $opensl_setup again to get the message.

to check if opennsl is installed on the SWITCH or not.

**root@localhost:/# ldconfig -p | grep opennsl**

libopennsl.so.1 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libopennsl.so.1
libopennsl.so.1 (libc6,x86-64) => /usr/lib/libopennsl.so.1
libopennsl.so (libc6,x86-64) => /usr/lib/libopennsl.so

//atleast one file should be visible.

.....

## # Install Openvswitch on switch – INSTALL EVERYTHING IN THE /home FOLDER.

**https://github.com/openvswitch/of-config/blob/master/INSTALL.md** - HAS ALL STEPS - Download from here - http://openvswitch.org/download/

// requires a lot of dependencies - install them all. Check the link. It has everything.

- copy the file into switch - mnt/onl/data/libraries folder. If libraries folder not there then create it.
  **Scp /home/mamtha/Downloads/openvswitch-2.8.1.tar.gz root@192.168.0.106:/mnt/onl/libraries**

- Go to the folder and run ths –
  **./configure --disable-dbus --with-ovs-srcdir=/root/openvswitch-2.8.1 PKG_CONFIG_PATH=/usr/local/lib/pkgconfig/**
  It might give some error related to python six. Then install six. For that you need to install pip first.
  - **$ sudo easy_install pip**
  - **$ pip install six**
  - If it doesn't give any error then all good. ☺
- **$ make && make install**

When Open vSwitch is installed, it can be started:

**/usr/local/share/openvswitch/scripts/ovs-ctl start**

*** You have to specify the path of the this OVS in the makefile.am of src folder of the monitor-automata-v2 file.

.....

# # Install Libssh - on switch and also PC

If system is not supporting libssh version >=0.6.4, we have to install it from source code

**Prerequisites** :

Install following packages :
**sudo apt-get install libtool libltdl-dev libxml2-dev libxslt-dev libxslt-python libcurl4-openssl-dev cmake zlib1g-dev openssl pkg-config libnacl-dev openssh-client libssl-dev libkrb5-dev libtool-bin**

**sudo apt-get install libltdl-dev libxml2-dev libxslt1-dev python-libxslt1 libcurl4-openssl-dev libssh2-1 and libssh2-1-dev**

libssl-dev - cmake not able to find openssl error - so install this package.
libkrb5-dev - error related to GSSAPI.

Steps to install -

- **wget https://red.libssh.org/attachments/download/195/libssh-0.7.3.tar.xz**
- **unxz libssh-0.7.3.tar.xz**
- **tar -xvf libssh-0.7.3.tar**
- **cd libssh-0.7.3/**
- **mkdir build**
- **cd build/**
- **cmake ..**
- **make**
- **make install**

**resolving libssh.so error –**

1. I tried installing libssh also... i am getting a different error now –
   Linking C shared library libssh.so/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/4.9/../../../../lib/libnacl.a(crypto_scalarmult_curve25519_donna_c64-base.o): relocation R_X86_64_32 against `.rodata' can not be used when making a shared object; recompile with -fPIC/usr/lib/gcc/x86_64-linux-gnu/4.9/../../../../lib/libnacl.a: error adding symbols: Bad valuecollect2: error: ld returned 1 exit statussrc/CMakeFiles/ssh_shared.dir/build.make:1367: recipe for target 'src/libssh.so.4.4.1' failedmake[2]: * [src/libssh.so.4.4.1] Error 1CMakeFiles/Makefile2:122: recipe for target 'src/CMakeFiles/ssh_shared.dir/all'

failedmake[1]: * [src/CMakeFiles/ssh_shared.dir/all] Error 2Makefile:137: recipe for target 'all' failedmake: * [all] Error 2i tried a few solutions... none of them worked..

This was one of the solutions - You can set the position independent code property on all targets:set(CMAKE_POSITION_INDEPENDENT_CODE ON)or in a specific library:add_library(lib1 SHARED lib1.cpp)set_property(TARGET lib1 PROPERTY POSITION_INDEPENDENT_CODE ON)
But this dint work for me, I went through a lot of shit and later tried to install this in /home folder and everything worked. I had not installed this in the /home folder in the beginning.

2. **$ sudo apt-get update && sudo apt-get install build-essential** - it was giving CMAKE_CXX_COMPILER-NOTFOUND - this fixed it.

3. Another one was a big one related to HMAC..etc. I dint find a solution for it. I just reinstalled my OS after trying it to resolve it for 4 days.

.....

## # Install Libnetconf - on switch

**Pre-requitse – usually all will be fulfilled when you install libssh. I had missed this - libtool-bin**

- **git clone https://github.com/cesnet/libnetconf**
- **cd libnetconf**
- **./configure**
- **make**
- **sudo make install**
- **[libnetconf] # cp dev-tools/lnctool/lnctool /usr/local/bin/**

P.S: If you face “configure: error: Missing libssh (>=0.6.4)” error during configure got back and install libssh.

.....

## # Pyang installation - on switch and PC

Prerequisites -
**sudo apt-get install python-setuptools**

- **git clone https://github.com/mbj4668/pyang**
- **cd pyang**
- **sudo python setup.py install**

.....

## # OFCONFIG INSTALLATION –

- copy the folder onto the switch. Its on the shrikanths CD. go to the /OFCONFIG folder.

- //if downloaded from the git then run ./boot.sh first to get the configure file.
- Path to configured OVS directory must be passed: -
- **$ ./configure --disable-dbus --with-ovs-srcdir=/root/openvswitch-2.3.1 PKG_CONFIG_PATH=/usr/local/lib/pkgconfig/**
- After successful configuration of OF-CONFIG, it can be build and installed using standard steps:
- **$ sudo make**
- **$ sudo make install**
- OF-CONFIG server can be started by: ofc-server
  **$ ofc-server -v 3 -f**

……

## # install czmq - on switch

**Pre-requiste - apt-get install libczmq-dev**

wget https://download.opensuse.org/repositories/network:/messaging:/zeromq:/release-stable/Debian_9.0/Release.key -O ----→ this might not work for you if the link has changed. So download from online. Put it on the switch and go to the folder.

- **cd czmq**
- **sh autogen.sh - fails - instal these autoconf automake libtool-bin libzmq-dev**
- **./configure**
- **make all**
- **sudo make install**
- **sudo ldconfig**

.....

## # Monitoring model -

copy this folder to the switch -

- **$ scp /home/mamtha/Desktop/CD/monitoring-model.tar.gz root@192.168.0.106:/**
- AFter copying run this -
  **root@localhost:/monitoring-model# pyang -f yin monitoring-model.yang**
  //shows some output like in xml format.
- Then run this
  **root@localhost:/monitoring-model# lnctool --model monitoring-model.yang transapi --paths /monitoring-model/paths**
  **root@localhost:/monitoring-model#**

monitoring-model*.yin, *_gdefs-config.rng,*-config.rng and *-schematron.xsl files were generated by lnctool. **Mainly - *.yin and *.rng and *.xsl files are created. Names will be different. check for extensions.**

- Yet to figure this out – I dint do this but shrikanth did. I worked for me without this. I dint really understand what he meant here. ☹
  Copied the monitoring-model-transapi.c file into server directory. (notice that appended transapi to the generated file)
  Modified the makefile.am in server directory to contain this new filename as well.

  All about generating new data-model.
  ==========
  Make changes to the YANG file present in the monitoring-model folder presented with the DVD. "monitoring-model.yang"
  and from witin the folder monitoring-model run the below command.
  **lnctool --model /home/shrikanth/monitoring-model/monitoring-model.yang transapi --paths /home/shrikanth/monitoring-model/paths**

  This will generate files required for NETCONF server.
  monitoring-model.yang
  monitoring-model.yin
  monitoring-model-config.rng
  monitoring-model-gdefs-config.rng
  monitoring-model-schematron.xsl
  monitoring-model.c --> This file is also generated with skeleton code. Merge the changes carefully into monitoring-model-transapi.c

  For more information on lnctool refer to LIBNETCONF website:
  lnctool is generated when LIBNETCONF library is installed.

  paths --> file contains the path to which a callback function will be generated by the lnctool. This will be used by the NETCONF server to trigger monitoring data model.
  Merge the generated files with the files present at OFCONFIG/Mon-model/server-src/model
  Note that monitoring-model.c equivalent code is present in monitoring-model-transapi.c file within the folder OFCONFIG/Mon-model/server-src/server

.....

# # Netopeer Installation - ON your PC - Need this to connect to the switch.

https://anukulverma.wordpress.com/2016/03/27/netopeer-cli-installation-and-configuration/

# Install required software

- **sudo apt-get update**
- **sudo apt-get install -y git libxml2 libxml2-dev libxslt-dev libssh2-1-dev libcurl4-gnutls-dev libdbus-1-dev doxygen libevent-dev libreadline-dev libncurses-dev libxml++2.6-dev libtool python-libxml2 openssh-server xsltproc cmake build-essential libssl-dev libtool-bin wget python-setuptools vim**

- **sudo apt-get clean**
- **sudo apt-get purge**

**//INSTALL LIBSSH, PYANG, LIBNETCONF**

- **git clone https://github.com/CESNET/netopeer.git**
- **cd netopeer/server/**
- **./configure –disable-dbus**
- **make && make install**
- **cd ../cli**
- **./configure**
- **make && make install**

# Run netopeer-cli

**$ netopeer-cli**
**$ connect --login root 192.168.0.106**
**$ subscribe**
**$ edit-config --config /home/shrikanth/TestCases/FP_MON_HELLO_5.xml running**
Root – username
I92.168.0.106 – ip of the switch.

.....
**Running - issues 1**

**1. Monitor-automata-v2 - 3d step -$ sudo make**
**zlog.h file not found while running make**

fix - So installed zlog -

- wget https://github.com/HardySimpson/zlog/archive/latest-stable.tar.gz
- $ tar -zxvf latest-stable.tar.gz
- $ cd zlog-latest-stable/
- $ make
- $ sudo make PREFIX=/usr/local/ install

.....
**Running - issues 2**

**ERROR –**
root@localhost:/Monitor-automata-v2# make
make all-recursive
make[1]: Entering directory '/Monitor-automata-v2'
Making all in src
make[2]: Entering directory '/Monitor-automata-v2/src'
gcc -std=gnu99 -I../src -Wall -O3 -I/usr/include/libxml2 -I../src/include -Wall -O3 -I/mnt/onl/data/libraries/openvswitch-2.8.1/lib/ -I/mnt/onl/data/libraries/openvswitch-2.8.1/include/ -I/mnt/onl/data/libraries/openvswitch-2.8.1/include/openvswitch/ -g -O2 -o

Monitor-automata-v2 monitor-automata.o util.o monitoring_utilities.o l3-interface-config.o vector.o lists.o -lzmq -lczmq -lpthread -lxml2 -lopenvswitch -lssl -lcrypto -latomic -lrt -lm -lzlog -lopennsl
**/usr/bin/ld: cannot find -lopennsl**
**collect2: error: ld returned 1 exit status**
Makefile:332: recipe for target 'Monitor-automata-v2' failed
make[2]: *** [Monitor-automata-v2] Error 1
make[2]: Leaving directory '/Monitor-automata-v2/src'
Makefile:349: recipe for target 'all-recursive' failed
make[1]: *** [all-recursive] Error 1
make[1]: Leaving directory '/Monitor-automata-v2'
Makefile:289: recipe for target 'all' failed
make: *** [all] Error 2

fix – main error was this - **/usr/bin/ld: cannot find -lopennsl**
**collect2: error: ld returned 1 exit status –** it had to be fixed.

It meant that the linker is missing –

https://stackoverflow.com/questions/16710047/usr-bin-ld-cannot-find-lnameofthelibrary

To figure out what the linker is looking for, run it in verbose mode.

For example, I encountered this issue while trying to compile MySQL with ZLIB support. I was receiving an error like this during compilation:

**/usr/bin/ld: cannot find -lzlib**
I did some Googl'ing and kept coming across different issues of the same kind where people would say to make sure the .so file actually exists and if it doesn't, then create a symlink to the versioned file, for example, zlib.so.1.2.8. But, when I checked, zlib.so DID exist. So, I thought, surely that couldn't be the problem.

I came across another post on the Internets that suggested to run make with LD_DEBUG=all:

LD_DEBUG=all make
Although I got a TON of debugging output, it wasn't actually helpful. It added more confusion than anything else. So, I was about to give up.

Then, I had an epiphany. I thought to actually check the help text for the ld command:

**ld --help**
From that, I figured out how to run ld in verbose mode (imagine that):

**ld -lzlib --verbose**
This is the output I got:

==========
attempt to open /usr/x86_64-linux-gnu/lib64/libzlib.so failed
attempt to open /usr/x86_64-linux-gnu/lib64/libzlib.a failed
attempt to open /usr/local/lib64/libzlib.so failed
attempt to open /usr/local/lib64/libzlib.a failed
attempt to open /lib64/libzlib.so failed
attempt to open /lib64/libzlib.a failed
attempt to open /usr/lib64/libzlib.so failed
attempt to open /usr/lib64/libzlib.a failed
attempt to open /usr/x86_64-linux-gnu/lib/libzlib.so failed
attempt to open /usr/x86_64-linux-gnu/lib/libzlib.a failed
attempt to open /usr/local/lib/libzlib.so failed
attempt to open /usr/local/lib/libzlib.a failed
attempt to open /lib/libzlib.so failed
attempt to open /lib/libzlib.a failed
attempt to open /usr/lib/libzlib.so failed
attempt to open /usr/lib/libzlib.a failed
/usr/bin/ld.bfd.real: cannot find -lzlib
Ding, ding, ding...

So, to finally fix it so I could compile MySQL with my own version of ZLIB (rather than the bundled version):

**sudo ln -s /usr/lib/libz.so.1.2.8 /usr/lib/libzlib.so**
Voila!

Do the the same for the opennsl library. As I told before we can find out where the opennsl is located using this command -

**root@localhost:/# ldconfig -p | grep opennsl**
libopennsl.so.1 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libopennsl.so.1
libopennsl.so.1 (libc6,x86-64) => /usr/lib/libopennsl.so.1
libopennsl.so (libc6,x86-64) => /usr/lib/libopennsl.so

Point the link to your .so or .so1 file appropriately. In our case –

**$ sudo ln -s /usr/lib/x86_64-linux-gnu/libopennsl.so.1 /usr/lib/libopennsl.so**

……

# # Now everything is done. So you can execute your code.

**1. Compiling and running the code.**

1. Copy the OpenVswitch library onto the path /mnt/onl/data/libraries/openvswitch-2.3.1/include/openvswitch/ and the zlog configuration file onto the path /etc/**zlog.configuration**

2. Uncomment the line in monitor-automata.h  //#define opennsl
3. Comment the line **#define CHECK_POLLING_ACCURACY** in monitor-automata.h to not to compile the instrumentation code added for checking the polling accuracy.
4. Update the Makefile.am file present in the path **src/Makefile.am** of the Monitor-automata code repository. // there are two make files, one is src and one in main folder. Make sure to check which one.

   From

   AM_CFLAGS += -I/mnt/onl/data/libraries/openvswitch-2.3.1/lib/
   AM_CFLAGS += -I/mnt/onl/data/libraries/openvswitch-2.3.1/include/
   AM_CFLAGS += -I/mnt/onl/data/libraries/openvswitch-2.3.1/include/openvswitch/

   TO

   M_CFLAGS += -I/home/mamtha/OVS/openvswitch-2.3.1/lib/
   AM_CFLAGS += -I/home/mamtha/OVS/openvswitch-2.3.1/include/
   AM_CFLAGS += -I/home/mamtha/OVS/openvswitch-2.3.1/include/openvswitch/

   Where 'mamtha' in the above path -> could be replaced by your name or the name of the home folder owned by the user logged into the machine.

5. Now to compile the code – go to the /Monito-automata folder and follow the steps –
   - **$ autoreconf -iv**
   - **$ ./configure**
   - **$ sudo make**
   - **$ sudo make install**
   - **Monitor-automata** - To start the first version of the code
   - **Monitor-automata-v2** - To start the second version of the code

6. After compiling the code, you should start the netconf server.
   - **sudo ofc-server -v 3 -> starts the server as a background process.**
   - **sudo ofc-server -f -v 3  -> starts the server as a foreground process**

7. Now we come to the controller side. , you can run the controller by 2 ways. Netopeer-cli and RYU controller. But in this you need both as you need to send RPC msgs to the switch through Netopeer-cli to set the device id and threshold and send HELLO msgs.

**1. Netopeer-cli -**

But even if you don't use netopeer as a controller you need it to run the RPC scripts.
Install netopeer on the system first.
https://anukulverma.wordpress.com/2016/03/27/netopeer-cli-installation-and-configuration/

- **$ sudo apt-get update**

- **$ sudo apt-get install -y git libxml2 libxml2-dev libxslt-dev libssh2-1-dev libcurl4-gnutls-dev libdbus-1-dev doxygen libevent-dev libreadline-dev libncurses-dev libxml++2.6-dev libtool python-libxml2 openssh-server xsltproc cmake build-essential libssl-dev libtool-bin wget python-setuptools vim** // these are the dependencies.
- **$ apt-get clean**
- **$ apt-get purge**
- Install Libssh >= 0.6.4.
- Install pyang
- Install libnetconf and lnctool
- Install netopeer (cli)

After installing, connect to the switch/netconf server through netopeer.

- **$ netopeer-cli**
    - It opens like this – netconf>
- **Netconf> connect - -login root 10.162.96.101** // root- username, ip is of the switch.
- It gets connect , now you have to run the rpc scripts.
- **Netconf> user-rpc** // below are the scripts
- Now, it opens and editor - //open editor - copy paste these - one by one. not all at once.
  // editor sucked big time- was not able to copy pasted - frankly annoyed me to hell. so i changed the ediot to nano and used it.
  // ANother issue - I copy pasted the msgs from soft copy of shrikanths thesis. But the "-" was in a different format as it was copied from latex so it wasnt ruuning. So check this before running.
  **netconf> editor nano** // this sets the editor of netopeer to nano.
  // if you want to know all the options then use help. it lists down all the options and you can choose. -> **netconf> help**
- The scripts are below –

  ********** DON'T INCLUDE <rpc> TAGS *******************

** You need to send only – configure device id and set fp tcam threshold msgs not all.

**FP GROUP STATUS Operation**

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<get_field_processor_group_status xmlns = "http://monitoring-automata.net/sdn-mon-automata">
<fp_monitoring_group> TCAM-MONITORING-TABLE </fp_monitoring_group>
</get_field_processor_group_status>
</rpc>
```

**CONFIGURE TCAM Threshold Operation**

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<set-fp-entry-threshold xmlns = "http://monitoring-automata.net/sdn-mon-automata">
<fp-entry-threshold>
<total-entries-threshold>245</total-entries-threshold>
<total-counters-count>100</total-counters-count>
<min-free-entries-per-device>3755</min-free-entries-per-device>
</fp-entry-threshold>
</se-fp-entry-threshold>
```

</rpc>

**MONITORING STATUS Operation**

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<mon_status xmlns="http://monitoring-automata.net/sdn-mon-automata">
<mon-id>200</mon-id>
<device-id>50</device-id>
</mon_status>
</rpc>
```

**PORT STATISTICS CLEAR Operation**

```
 <rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<port_statistics_clear xmlns="http://monitoring-automata.net/sdn-mon-automata">
<clear-port-stats>
<port-index>2</port-index>
</clear-port-stats>
</port_statistics_clear>
</rpc>
```

**CONFIGURE DEVICE ID Operation** // if not given then it takes the deault value -1

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<configure-device-id xmlns="http://monitoring-automata.net/sdn-mon-automata">
<switch-identification>10</switch-identification>
</configure-device-id>
</rpc>
```

**GET PORT STATISTICS Operation**

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<port_statistics xmlns="http://monitoring-automata.net/sdn-mon-automata">
<port-statistics-get>
<port-index>2</port-index>
</port-statistics-get>
</port_statistics>
</rpc>
```

// each type you finish each one of them you can see the ouput on the **monitor-application.log** and also you can an **OK**. Msg on the netopeer-cli terminal.
// subscribe to the session by using "**subscribe**" command.

- Similarly, one can set the device IDs to the switches.
- NOTE: The SDN controller code expects that the **AS5712 switch is assigned ID 10 and AS7712 switch is assigned with the ID 20 always**. Also, the controller code always configures a monitoring agent with ID-5. So, if you are trying monitoring-agent movement scenario then make sure that the TCAM Device IDs are assigned as expected by the controller. and then set the TCAM threshold on both the switches.
- After this you send the HELLO msg to connect with the switch using the following command **edit-config --config /home/shrikanth/TestCases/FP_MON_HELLO_5.xml running FP_MON_HELLO_5.xml** is a xml file. It can be found on the CD of shrikanth.

- Finally, configure monitoring agents on the switch (through netopeer-cli) using the above commands meant to perform edit-config operation.
- The TestCases folder is also supplied with the DVD. So, change the path (from /home/shrikanth)to the folder containing test cases accordingly. Most importantly, do not configure the monitoring agent ID 5 through netopeer-cli console as it will be configured from the SDN controller itself and the agent will be moved to AS7712 switch later.
- To stop a monitoring agent, execute the corresponding test-case as **edit-config --config /home/shrikanth/TestCases/FP_MON_STOP_1.xml running**
- To change the polling interval of a monitoring agent, **edit-config --config home/shrikanth/TestCases/sample_test_state_mon_param_change.xml running** NOTE: the monitoring-agent ID and poll-time should be changed in the above test case file so that the poll time is changed for the desired monitoring agents.

**2. RYU Controller –**

- Requires ncclient program supplied with the DVD as well as present in the code repository at https://shrikanthmd1986@bitbucket.org/shrikanthmd1986/ncclient.git
- just goto the folder that contains setup.py, in the DVD goto -> Controller-Code/ncclient
- execute sudo python setup.py install

BINGO!!! ncclient is installed.

It is expected that all the dependencies are installed for RYU Controller following the link:

- To install RYU SDN controller,
- Goto the path Controller-Code/RYU/ryu-4.10 and execute
- sudo python setup.py install
- To run the RYU Controller monitoring application, run the below command
- **ryu-manager /Desktop /Controller-Code/RYU/ryu-4.10/ryu/app/monitoring_conf/monitoring_application.py** path can change depending on where you have stored the code.
- When You run this command, RYU controller starts and it connects to the switch. All msgs sent from the switch can be seen on the terminal of RYU.

The above command will run the application, installs a monitoring agent with ID 5 and listens to the switch for notifications. It handles MON_SWITCH notifications at the moment and ignores all other notifications. The code has been written in such a way.

NOTE: /media/shrikanth/OPENNSL/Final-Submission/ should be replaced bz which ever path you copy the Controller-Code/ folder to.

**3. ostinato-bin-win32-0.8** -> Contains the Traffic Generator application.

To use the traffic generator, follow the below steps on a windows machine:

- Run drone.exe as an administrator
- Run ostinato.exe as an administrator as well.

- Select any of the sessions by clicking on File-> Open Session : All the sessions are ending with the file name *.ossn.
- It will have atleast one flow already present. 100flows_using_one_stream.ossn is a good example to use.

It sends VLAN tagged (with VLAN ID-10 as expected by the Accton AS5712 switch) packets with source IP 50.0.0.1 to 50.0.0.N where N can be configured from the tool and the destination IP will be always 20.1.1.2

https://userguide.ostinato.org/ for more information on Ostinato.
Important to Click on APply button before starting the traffic on a port.