Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico Di Milano

Network Automation

Projects

Sebastian Troia and Guido Maier

Politecnico di Milano, Milan, Italy

{sebastian.troia}{guido.maier}@polimi.it

Network Automation

13/12/2022

Agenda

- Projects
  - SDN – Openflow (7 projects)
  - SDN - NETCONF/YANG (2 projects)
  - NFV - ADVA Ensemble (3 projects)
- Project outcome

Agenda

- Projects
  - **SDN – Openflow (7 projects)**
  - SDN - NETCONF/YANG (2 projects)
  - NFV - ADVA Ensemble (3 projects)
- Project outcome

SDN - Openflow

Traffic management functions implemented as NetApps over the SDN controller

1. Load balancing
2. Node failure detection
3. Link failure detection
4. Monitoring
5. MPLS TE disjoint path routing
6. MPLS TE constraint based routing
7. SD-WAN monitoring and traffic engineering

Load balancing

Team members:

Objectives:

1. Design a multipath network where there are multiple paths from one switch to another
    1. Using mininet exploit different network topologies
2. Measure the traffic load on the links in real time
    1. Using openflow functions, you can get the status of the link with the actual or historical load (e.g. meter tables)
3. Implement an algorithm to load balance the traffic based on link load
    1. Implement an algorithm that, given the amount of byte flowing in a congested link (or links), balance the traffic
4. Test the algorithm with different traffic generators (D-ITG, iperf)
    1. D-itg: http://www.grid.unina.it/software/ITG/
    2. Iperf: https://iperf.fr/
5. Display the results
    1. Display the results in terms of load in the links, specifically show how the link load change over time (e.g. matplotlib)
6. Implement the algorithm into the BONSAI SDN testbed

Node failure detection

Team members:

Objectives:

1. Design a ring network and monitor the switches status through openflow functions
   1. Create a mininet topo, and save the switch status at each *t* (time interval)
2. Implement an algorithm to react to switch failures
   1. Look at the openflow messages and reconfigure the Openflow tables accordingly (e.g. group tables)
3. Test the algorithm with different traffic generators (D-ITG, iperf)
   1. D-itg: http://www.grid.unina.it/software/ITG/
   2. Iperf: https://iperf.fr/
4. Display the results
   1. Delay, pkt loss with and without the implemented algorithm
5. Implement the algorithm into the BONSAI SDN testbed

Link failure detection

Team members:

Objectives:

1. Design a ring network and monitor the link status through openflow functions
    1. Create a mininet topo, and save the link status at each *t* (time interval)
2. Implement an algorithm to react to the link failure
    1. Look at the openflow messages and reconfigure the Openflow tables accordingly (e.g. group tables)
3. Test the algorithm with different traffic generators (D-ITG, iperf)
    1. D-itg: http://www.grid.unina.it/software/ITG/
    2. Iperf: https://iperf.fr/
4. Display the results
    1. Delay, pkt loss with and without the implemented algorithm
5. Implement the algorithm into the BONSAI SDN testbed

Monitoring

Team members:

Objectives:

1. Design a multipath network where there are multiple paths from one switch to another
   1. Create a mininet topo
2. Implement a monitoring algorithm able to collect periodically the statistics of the flows from the switches
3. Test the algorithm with different traffic generators (D-ITG, iperf)
   1. D-itg: http://www.grid.unina.it/software/ITG/
   2. Iperf: https://iperf.fr/
4. Display the results in terms of flow statistics
5. Implement the algorithm into the BONSAI SDN testbed

MPLS TE disjoint path routing

Team members:

Objectives:

1. Design a multipath network where there are multiple paths from one switch to another
    1. Using mininet exploit different network topo
2. Implement an algorithm to calculate the first K disjoint shortest path and implement the LSPs for each of them
    1. Create a function that returns the first K (e.g. K=5) disjoint shortest paths in a list
    2. Implement the LSP for each path
3. Test the algorithm with different traffic generators (D-ITG, iperf)
    1. D-itg: http://www.grid.unina.it/software/ITG/
    2. Iperf: https://iperf.fr/
4. Display the results
    1. Print the results of the k-shortest path algorithm (paths with different K values)
    2. Delay and throughput for each LSP
5. Implement the algorithm into the BONSAI SDN testbed

MPLS TE constraint based routing

Team members:

Objectives:

1. Design a multipath network where there are multiple paths from one switch to another
    1. Using mininet exploit different network topo
2. Implement an algorithm to calculate the shortest path between two end hosts based on the available bandwith on the links and implement the LSP
    1. Generate background traffic in the network
    2. Find the shortest path between two end hosts based on the available bandwith on the links and implement the LSP
3. Test the algorithm with different traffic generators (D-ITG, iperf)
    1. D-itg: http://www.grid.unina.it/software/ITG/
    2. Iperf: https://iperf.fr/
4. Display the results
    1. Print the selected shortest path and the current available bandwidth on the links
5. Implement the algorithm into the BONSAI SDN testbed

SD-WAN monitoring and traffic engineering

Team members:

Objectives:

1. Design a multipath network where there are multiple tunnels between two switches (or CPE)
    1. Create a mininet topo
2. Implement an algorithm able to monitor the delay of different flows and re-route them according to a specified threshold
3. Test the algorithm with different traffic generators (D-ITG, iperf)
    1. D-itg: http://www.grid.unina.it/software/ITG/
    2. Iperf: https://iperf.fr/
4. Display the results in terms of end to end performance with and without SD-WAN
    1. Delay, throughput and packet loss
5. Implement the algorithm into the BONSAI SDN testbed

Agenda

- Projects
  - SDN – Openflow (7 projects)
  - **SDN - NETCONF/YANG (2 projects)**
  - NFV - ADVA Ensemble (3 projects)
- Project outcome

SDN – NETCONF/YANG

Automation of configuration functions implemented as NetApps over the SDN controller

1. Creation of an optical service
2. Monitoring and analysis of an optical service

Creation of an optical service

Team members:

Objectives:

- The goal of this project is to create an optical service through the SDN controller provided by SM-Optics.
    - Deploy an optical service through the graphical user interface of the SDN controller
    - Design a list of REST API that create the optical service
    - Retrieve the topology through the REST API
    - Automate the creation of the service by implementing a python program able to create the service and check if the creation has been performed successfully

Monitoring and analysis of an optical server

Team members:

Objectives:

- The goal of this project is to monitor an optical service through the SDN controller provided by SM-Optics.
  - Deploy an optical service through the graphical user interface of the SDN controller
  - Design a list of REST API that monitor the optical parameters of a service
  - Retrieve the topology through the REST API
  - Automate the collection of monitored parameters by implementing a python program able to store and visualize the collected data

Agenda

- Projects
  - SDN – Openflow (7 projects)
  - SDN - NETCONF/YANG (2 projects)
  - **NFV - ADVA Ensemble (3 projects)**
- Project outcome

NFV – ADVA Ensemble

Setup and configure a service chain on the ADVA testbed using the Ensemble orchestrator

1. Setup and configure a service chain with MikroTik routers
2. Setup and configure a service chain with Firewalls
3. Setup and configure an SD-WAN based on FlexiWAN

Setup and configure a service chain with Microtik routers

Team members:

Objectives:

1. Ensemble Orchestrator:
   - Create Network Service Template NST with virtual MikroTik router and CentOS VM with mapping of services and physical ports based on Base-NST
2. Ensemble Virtualization Director:
   - Import NST in Ensemble Virtualization Director for tenant: PoliMI1
   - Create Application that uses ZTP template and imported NST
   - Create new Connector endpoint and select application
3. VNF configuration
   - Initial configuration of VNFs via VNC console
   - Generate test traffic from CentOS VM to test the MikroTik router

Setup and configure a service chain with Firewall

Team members:

Objectives:

1. Ensemble Orchestrator:
   - Create Network Service Template NST with virtual firewall and CentOS VM with mapping of services and physical ports based on Base-NST
2. Ensemble Virtualization Director:
   - Import NST in Ensemble Virtualization Director for tenant: PoliMI2
   - Create Application that uses ZTP template and imported NST
   - Create new Connector endpoint and select application
3. VNF configuration
   - Initial configuration of VNFs via VNC console
   - Generate test traffic from CentOS VM to test the firewall router

Setup and configure an SD-WAN based on FlexiWAN

Team members:

Objectives:

1. Ensemble Orchestrator:
   - Create Network Service Template NST with FlexiWAN routers and CentOS VM with mapping of services and physical ports based on Base-NST
2. Ensemble Virtualization Director:
   - Import NST in Ensemble Virtualization Director for tenant: PoliMI3
   - Create Application that uses ZTP template and imported NST
   - Create new Connector endpoint and select application
3. VNF configuration
   - Initial configuration of VNFs via VNC console
   - Configure the FlexiWAN manager
   - Generate test traffic from CentOS VM to test the SD-WAN

Agenda

- Projects
  - SDN – Openflow (7 projects)
  - SDN - NETCONF/YANG (2 projects)
  - NFV - ADVA Ensemble (3 projects)
- **Project outcome**

Project outcome

The outputs of the project are the following:

1. **A power point** presentation to be held in English by all the members of the group (maximum 20 slides). You can find the template in the gitlab repository of the course
   - https://gitlab.com/network-automation-code-repository/2022-2023/code-template/
2. **A Demo** in which you show the project running without errors ☺
3. **Code** of the project (**commented**)
4. **Documentation** of the code written as README.md file inside the gitlab repo

THE PROJECT IS CONSIDERED COMPLETED ONLY IF YOU HAVE ACHIEVED ALL THE 4 OUTPUTS

You will have **access to a Gitlab repository** where you will commit your code together with the final power point presentation and the complete documentation to be done in *Readme.md*

Following, a useful *Readme.md* template:
https://docs.github.com/en/free-pro-team@latest/github/writing-on-github/basic-writing-and-formatting-syntax

THANK YOU