Networking Monitoring Technique in SDN using RYU controllers

Fazil Sarfaraz, Twain Pereira, Prof. Chao Li Tarng
### Department of Electrical Engineering, San Jose State university, San Jose, California 95112.

Introduction

Current computer networks are in a phase of consistent growth and evolvement, the internet these days and usage of the internet applications has increased sharply since more than a decade. Keeping this increase in mind and the increase of service applications over the internet it is required to have a large, strong, balanced and reliable framework of network. To achieve this computer networking giants are competing in developing advanced switching, routing, load balancing and QoS techniques and various protocols to support these developments. This highly potential field networking has introduced Software Defined Networking (SDN) which is currently the hottest technology in computer networking and is getting much of attraction form computer networking companies. Being relatively new there is a vast potential of development in this field and is said to be the future of networking in which a command is used over the routers, switches to control the traffic and hence the routers and switches can be made less intelligent and cheap. This give rise to load balance the traffic in SDN which is very crucial to maintain the QoS." .

Methodology

The idea behind SDN is pulling the intelligence of the network away from the hardware. A similar concept has been used earlier. In the old days we had box that used to contain everything right from the hard-drives as memory units, the processing units, the operating system unit, a unit to have access control list etc. And the problem in having this kind of system was that once the hard drives failed, all the data was gone and the rest of the machine was useless. This led to the implementation of the storage network concept where all the data drives were removed and separated from the control mechanism which controlled how the data was stored. So now we have data centers that contain data and a management system like servers that control this data storage."

OpenFlow – A Clear Separation of Control and Data Plane
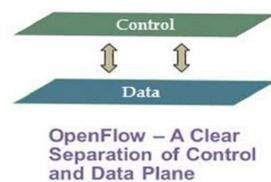
Methodology

OpenFlow is a protocol. This is the protocol followed by the controller to communicate with the switch. A switch following OpenFlow protocol is called as an SDN switch or a OpenFlow switch. The below diagram show how the connections actually look like. The OpenFlow connection is an SSL secure connection."
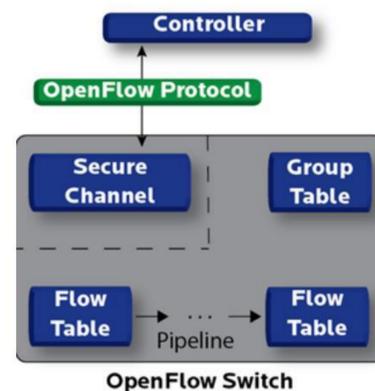
Architecture

The block diagram of our network topology consists of controller module which will do the controlling and managing part of the network as explained above and the second block consist of the distributed SDN controller part these are more than one and we are working on optimizing this part of the topology. OpenFlow enabled switches are connected which are programmed to create the routing policies and we will implement algorithm to help them migrate and control traffic whenever needed
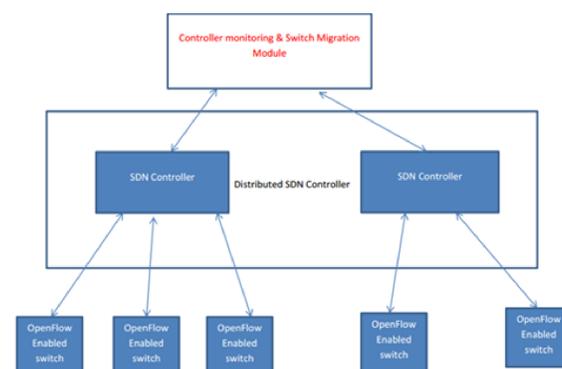
Rest API

This project uses Mininet VM to define default and custom topologies. Mininet VM provides command line interface to define these topologies. Along with that it provides pinging operations within the defined network. To extract the packet contents, RYU controller are used which provided REST API to fetch data. The API provided ease in programming.
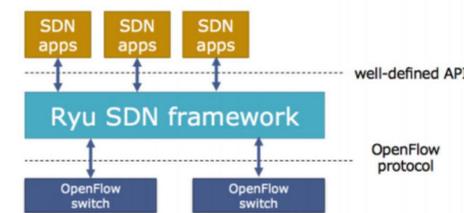
RYU Controller

RYU is a java based controller. This project contains a python script that invokes various RYU modules which extract the desired data from the virtual network. This is a simulation of an actual SDN deployed network wherein you can program your control plane to extract data from your forwarding plane and offers a variety of operations that can be performed in data extraction.
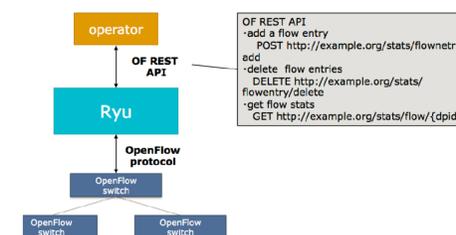
This project serves the purpose of having an application being developed on an SDN framework which utilizes the exciting features of SDN.

Results

Based on the work done we were able to achieve load monitoring and switch migration. The switch migration was done on a passive note, and was only achieved once. For it to occur hard-coded messages were needed to be sent. This prompted the creation of a few protocol based messages. Each of these commands would trigger a different action.

The first action would find the switch which has a higher level of load than the other controllers. The next command would identify the destination controller for the switch to move to. Another set of commands would relieve the switch from the initial controller and move the information and the controller to the new destination controller. After this all new data flows would be directed to this new controller instead of the old one.

Summary

The proposed work would allow for unlimited scaling in a SDN network and allow a single monitored controller to govern an entire network using other controllers. Thus the typical SDN network which is currently limited to small network companies employing private networks or a company employing a large network made up of multiple independent small networks would be released to the world.

Key References

1.N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, OpenFlow: Enabling innovation in campus networks, ACM Computer Communication Review, Apr. 2008.
2. M. Yu, J. Rexford, M. J. Freedman, and J. Wang. Scalable Flow-Based Networking with DIFANE. In Proc. SIGCOMM, 2010.
3. R. Bennesby, P. Fonseca, E. Mota, and A. Passito. An Inter-AS Routing Component for Software-Defined Networks. In Proc. NOMS, Apr. 2012
4. M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat. Hedera: Dynamic Flow Scheduling for Data Center Networks. In Proc. NSDI, Apr. 2010.
5. N. Gude, T. Koponen, J. Pettit, B. Pfa, M. Casado, N. McKeown, and S. Shenker. NOX: Towards an Operating System for Networks. In SIGCOMM CCR, July 2008.
6. A. Tavakoli, M. Casado, T. Koponen, and S. Shenker. Applying NOX to the Datacenter. In HotNets, 2009.

Acknowledgements