Specialization Software-defined Networking (Winter 2014/2015))

Institute of Computer Science
Georg-August-Universität Göttingen

Lecturer: Dr.Mayutan Arumaithurai

Author: Hari Raghavendar Rao Bandari
11334055
h.bandari@stud.uni-goettingen.de

Date: April 30, 2015

Contents

1 Paper Review for Group Discussion **3**

1.1 Paper Information 3

1.2 Summary of the paper 3

1.3 Contributions of the Paper 4

1.4 Strengths of the Paper 4

1.5 Weaknesses of the Paper 4

1.6 Open Issues and Future Work 5

1.7 Your Questions to the Authors 5

2 Paper Review for Group Discussion **6**

2.1 Paper Information 6

2.2 Summary of the paper 6

2.3 Contributions of the Paper 7

2.4 Strengths of the Paper 7

2.5 Weaknesses of the Paper 7

2.6 Open Issues and Future Work 7

2.7 Your Questions to the Authors 7

3 Paper Review **8**

3.1 Paper Information 8

3.2 Summary of the paper 8

3.3 Contributions of the Paper 9

3.4 Strengths of the Paper 9

3.5 Weaknesses of the Paper 10

3.6 Open Issues and Future Work 10

3.7 Your Questions to the Authors 10

1 Paper Review for Group Discussion

1.1 Paper Information

Title: SDN traceroute: Tracing SDN Forwarding without Changing Network Behavior
Type of Paper: Short Paper

1.2 Summary of the paper

They introduce a tool, SDN traceroute that can question the current path of the packet through an SDN-enabled network. The packet path will be traced out using actual forwarding mechanisms at each SDN-enabled device without changing the forwarding rules being measured. This provides administrator to track down the path of the current packet in SDN enabled network. It also help to debug the packet at switch and controller logic. For this implementation they required high-priority rules per device, run on the commodity hardware using required features of the openflow 1.0, and can generate traces in about one millisecond per hop. The mechanism of the SDN Traceroute interface the input will be Ethernet frame with user-specified packet header fields and Injection point: (Switch id, Port) to SDN- enabled Network and output will be the route taken by the packet with the ordered list of (Switch id , Port). Therefore, It work in two stages in the first stage it color each switch in the network using graph coloring algorithmi.e vertex coloring of network topology. Then install high- priority rules in switches and send to controller if tag is adjacent switch color by this they can trap the packet coming from respective neighbours. Second stage it will install a probe packet into the network for conducting traceroute now packet-out message with tag will sent to next hop with color tag of it, now the hop which receive the packet it will send packet-in message to the controller with port number and tag. Therefore controller will record the hop, and it will modify the probe packet and resend to the same hop with packet-out with tag, action so that it forwards to next hop with color tag.

Although, It is and alternate tool for measuring SDN enabled network. It will modify the rules in measured switch.

In conclusion, the key characteristics which use to trace the path of given packet using actual forwarding rules in the network. They are In Non-invasive, the existing rules in forwarding table does not required no modification. Therefore SDN traceroute will not change the forwarding behaviour but it is trying to measure it. It required very low overhead with very few(1-2)ternary content-addressable memory(TCAM) rules per switch.It can trace arbitrary Ethernet packets.In practical SDN traceroute is Compatible with existing SDN Switches and protocols in SDN network.

1.3 Contributions of the Paper

They introduced alternative tool to trace the path of the current packet in sdn-enabled network. They implement using color graphing algorithm to color switches so that neighbour switch is not colored with same color to trace out the path. Therefore, insert a probe packet into the network it will forward it next hop with the color tag then before forwarding to next hop it will send to controller then it modify it again it will resend to same hop and then it can forward it to adjacent switch with the color tag. Therefore this approach is good way of identifying a packet path. Creativity stands where they used color graphs algorithms for switches. it does not modify the existing rule in forwarding tables. The major Impact about this tool is to trace out the packet current path and install high-priority rules to trap and re-inject the packet to each hop.

The technically depth of the paper is not clear as the paper is more about their research work.

1.4 Strengths of the Paper

1. It is compatible with existing SDN switches and protocols. 2. It requires very few(1-2) TCAM. 3. No Modification is required in production rules.

1.5 Weaknesses of the Paper

1. It reserves k bits in the packet header for carrying color tags. 2. It reserves highest priority rules for color tag matching.

1.6 Open Issues and Future Work

What is the scalability of the SDN Traceroute tool.

1.7 Your Questions to the Authors

How SDN traceroute provide consistency and security to install or update rules in switches.

2 Paper Review for Group Discussion

2.1 Paper Information

Title: Controller-agnostic SDN Debugging
Type of Paper: Short Paper

2.2 Summary of the paper

The paper states the problems about the SDN developers facing problem while debugging the network. As there are several factors such as traffic behavior and interaction of application will not allow for debugging. This paper provides OFf, which is help SDN developers for debugging and testing. OFf is basically built on fs-sdn simulator, it basically offers debugging features for controller applications such as stepping, break points and watch variables. Also it provides option in packets tracing, packet replay and virtualization and alerts when configuration changes. OFf is accessed through a text through a text interface and is designed to interoperate with any standard SDN controller platform.

Therefore, OFf consist of two main parts 1. OFf proxy and OFf controller/ debugger runtime interface that connects to the fs-sdn. The OFf proxy basically acts as a bridge between fs-sdn and real controller platform. It translates simulated messages between switches and control plane and vice versa. 2. OFf control debugger is a platform that connects to language-level debugging through an OFf runtime interface component and handles starting a controller using a library inter -position agent that ensure that any time related calls by the controller will return fs-sdn.

In conclusion, paper provides three scenarios were described to find the bugs and security issues. The first scenario includes incorrect ordering of updates, in this barrier messages are induced in the network to get sequential ordering of updates and avoid race conditions which leads to poor performance and provide consistant network state. second scenario, Bad multi-app interaction problem leads to when SDN programs are not composed properly. It notify that inconsistencies in the network may arise. By using OFf diff reports as it can report changes. we can exactly determine the overheads of OFf are reduced. Third Scenario, Unexpected Rule Expiration. it play a role when unexpected interactions can occur when wildcard rules overlap or specific microflow rules are shadowed by wildcard rules.

2.3 Contributions of the Paper

Major contribution providing OFf tool, which is basically a debugging tool for the SDN developers that helps in reducing the workload and provides a better and efficient way of debugging capabilities to assess stepping and watch variables. Its an important tool as it helps in reflecting the problems that are often a problem in the networks. This definitely creates an impact for developers that basically waste a huge amount of time in finding/ searching out the traces out the packet path of the leads to different problems. The paper highlights these issues by introducing and implementing these on the scenarios. In conclusion, OFf that is built on the top of fs-sdn, provides implementations that helps in improving the OFfs accuracy and scalability. OFf mainly used to identify and eliminate bugs in a traffic engineering application, and to identify and remove a security vulnerability.

2.4 Strengths of the Paper

1. Papers was written in details how OFF works. 2. It helps sdn developers to debug and resolve race condition. 3. It identify and removes a security vulnerability.

2.5 Weaknesses of the Paper

The paper does not provided tracking down the root cause of a bug.

2.6 Open Issues and Future Work

What happens when we have more middleboxes in the network how we can maintain consistency?

2.7 Your Questions to the Authors

How to maintain consistency in the application level debugging

3 Paper Review

3.1 Paper Information

Title: A Network-State Management Service
Type of Paper: Full Paper

3.2 Summary of the paper

This paper provides deep insight into how we can multiple network management applications on shared network such as traffic engineering, load balancing, link corruption and device firmware update. If we run all these respective management application independently we undergo two issues 1. application conflict and 2. safety violation. Therefore, they introduced three view model observed state, it maintains an update view of the actual network state. Applications read this state and propose state changes based on their respective goals. Second target state is the desired state of the network, and Statesman is responsible for updating the network to match the Target State.Therefore, success or failure of updating the network towards the Target State will be reflected into the Observed State, and the applications will notify about the network conditions. Third proposed state that captures the state desired by applications,it writes its own Proposed State. It also detects the conflicts among them with target state. They even proposed dependency model of state variables which deals with applications read and write different state variables of the network,but state variables are not independent. The writability of one state variable can depend on the values of other state variables. Therefore, Statesman does not treat network state as a collection of independent variables but includes a model of dependencies among them. These dependencies are used when checking for conflicts and invariant violations. it also resolves conflicts occur in the network state due to the dynamic nature. TS-OS :The TS can conflict with the latest update OS when changes in the network behaves some state variables uncontrollable, although they have new values in the TS, PSOS. When the checker examines a PS, the OS may have changed from the time that the PS was generated, and some variables in the PS may not be controllable. PSTS: The TS is essentially the accumulation of all accepted PSes in the past. A PS can conflict with the TS due to an accepted PS from another application.The first two types of conflicts are because of the changing OS.To resolve TSOS conflicts, we introduce a flag called SkipUpdate.For PSTS conflicts, which are caused by the conflicting appli- cation proposals, Statesman supports an extensible set of conflict resolution mechanisms. It currently offers two mechanisms. The basic one is last-writer-wins The more advanced mechanism is priority-based locking. and also sufficient for current deployment.

The storage service is implemented as a HTTP web service with RESTful APIs. There is a freshness parameter in the read API because Statesman offers different freshness modes for reading the network states. Many management applications do not need the most up-to-date network states and can safely tolerate some staleness in state data. At the same time, applications that cannot tolerate staleness can use the up-to-date freshness mode.

In conclusion, Statesman enables multiple loosely-coupled network management applications to run on a shared network infrastructure, while preserving network safety and performance. It safely composes uncoordinated and sometimes-conflicting application actions using three distinct views of network state, inspired by version control systems, and a model of dependencies among different parts of network state.

3.3 Contributions of the Paper

They proposed three model view and resolved conflicts and migration problems between management network application and maintains network-wide variants. The statesman also handles operations failure. They also provided end-end latency breakdown along with it its is network state scale and checker performance and also read write micro benchmark performance. It also offer to run multiple network management application on shared network.

3.4 Strengths of the Paper

They implemented all high availability RESTful web service with persistent storage. They resolved application conflicts and failure migration. It also handles operational failures.Most important it is implemented in Microsoft data center.

3.5 Weaknesses of the Paper

1. Software implementation and deployment overheads. 2. Asynchronous components are seen to yield lot of idle time. 3. Invariant set is very limited and the specifies of adding/addressing them is not deleted.

3.6 Open Issues and Future Work

what happens when we have multiple middleboxes in the network how does statesman model behave?

3.7 Your Questions to the Authors

whether statesman have flexibility to scale if so how consistency and security will be handled.