# Source Routed Forwarding with Software Defined Control, Considerations and Implications

Mourad Soliman, Biswajit Nandy,
Ioannis Lambadaris
Carleton University, ON, Canada
{msoliman, bnandy, ioannis}@sce.carleton.ca

Peter Ashwood-Smith
Huawei Canada
Kanata, ON, Canada
peter.ashwoodsmith@huawei.com

## ABSTRACT
The research introduced in this paper focuses on controller scalability and performance issues in Software-Defined Networks (SDNs), and discusses a new routing scheme that leverages a variation of Source Routing for use in OpenFlow-based networks. The research aims to reduce the state needed to be distributed to the network devices by the controller(s) in SDNs, and in return improve the scale, convergence time, fault tolerance and cost of such network architectures.

## Categories and Subject Descriptors
C.2.1 **[Computer-Communication Networks]**: Network Architecture and Design; C.2.2 **[Computer-Communication Networks]**: Network Protocols; C.4 **[Performance of Systems]** Design Studies; Performance attributes.

## Keywords
Software Defined Networks, Source Routing, OpenFlow, SDN.

## 1. INTRODUCTION
The Software-Defined OpenFlow network architecture is composed of the OpenFlow switches, the controller(s), and a secure channel for communication between the controller and the switches using the OpenFlow protocol [1]. In this architecture, the controller has knowledge of the topology of the network of forwarding devices that it manages, and is responsible for configuring the forwarding state on those devices. The forwarding state is stored in the OpenFlow switches in flow tables that contain entries that match flows to forwarding actions to be executed. New flows trigger new state distribution to all the devices along the paths that the controller decides the flows should traverse [1].

In WANs with long propagation delays, the need to distribute large amount of state places limits on network convergence time, and affects controller's ability to respond to network events in a very short time [2].

To address the problems related to limitations imposed by state distribution in SDN, a variation of source routing is proposed [3]. Since the controller knows the exact path between ingress and egress nodes, with some modifications to the OpenFlow model, the controller can relay path information to ingress nodes to be embedded in a header that can be inserted in the packets, and inspected at each node along the path to forward the packets accordingly [3]. This new approach is introduced in *Section 2*, and analyzed in the context of the Internet2 OS3E topology [4] in *Section 3*. The current state of the research and the future objectives are discussed in *Section 4*.


*CoNEXT Student'12*, December 10, 2012, Nice, France.


## 2. SOURCE ROUTING APPROACH
### 2.1 Path Expression
*Figure 1* shows a network composed of 5 OpenFlow switches and a controller. Each switch's interfaces are locally labeled. A possible path for a flow from *NodeX* to *NodeY* is through *SwitchA* →*SwitchB* →*SwitchD* →*SwitchE*. This means that the flow will be forwarded to interface 2 on *SwitchA*, then Interface 3 on *SwitchB*, *SwitchD*, and *SwitchE*. The path can be expressed as a sequence of interface numbers that the flow will be forwarded through. In the case explained above the path can be expressed as {2,3,3,3}, and can be embedded at the ingress node in a packet header with a hop count to indicate the position along the path.

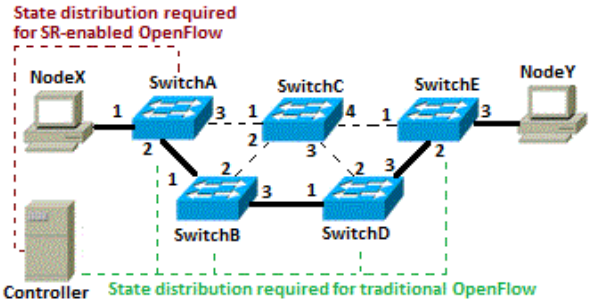


**Figure 1. 5-Switches SDN showing Source Routing approach.**

Using this approach, new state information will be pushed to only one node, the ingress switch, *SwitchA*, instead of having to distribute state to four nodes, i.e., the controller only needs to distribute one-fourth the amount of state distributed if traditional OpenFlow is used. *SwitchA* will form a packet header that carries the path information, and add the header to the packets of the flow as they come. The intermediate nodes won't need to receive state information as they will inspect the newly added header for forwarding decisions. The state reduction in this case is directly proportional to the number of links in the path, *Figure 2*. Source route packet headers are added and removed by trusted ingress nodes that authenticate with the controller to avoid security issues resulting from untrusted users routing as they want.

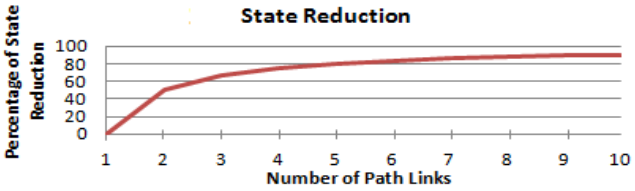


**Figure 2. State Reductions with different path lengths.**

## 2.2 Reverse Path Calculation

As the packets traverse the defined path, the reverse path can be calculated by changing the output interface number in the path packet header with the input interface number at each intermediate node. For example, before *SwitchB* forwards the packet to Interface 3, the entry "3" in the path header can be changed with the input interface number "1", and so on at *SwitchD* and *SwitchE*. The sequence {2,3,3,3} will be changed to {1,1,1,2} by the time it reaches *SwitchE* and can be stored for future use with flows from *NodeY* to *NodeX* without the need for the ingress node to contact the controller, i.e., achieving a 50% reduction in the burden of the controller.

## 2.3 Link Failure Recovery

Link failure conditions can be handled locally in this approach as the intermediate nodes don't hold state information about the network and rely on the path packet header in forwarding. During network initialization, the controller can push information to nodes regarding alternative routes to the next hop. For example, In *Figure 1*, if packet reaches *SwitchB* and Interface 3 is found down, the switch can change the entry "3" in the path sequence with {2,3} to reach *SwitchD* but through *SwitchC*, based on pre-stored information. As the failure is being handled locally temporarily, a notification would be sent to the controller about the failure condition.

# 3. ANALYSIS OF INTERNET2 OS3E

Internet2 is developing a 34-node SDN called the Open Science, Scholarship and Services Exchange (OS3E) across the US to support advanced global scientific research [4].

## 3.1 State Distribution

The OS3E network was analyzed considering the state distribution, assuming one controller is deployed. *Table 1* shows results relating the average number of links in the shortest paths from three main cities to the rest of the network to the reduction in state distribution if the Source Routing approach is in effect.

**Table 1. State Reduction in OS3E's three main cities.**

| City | Avg # Links in Shortest Path | % of Distributed State Reduction |
|---|---|---|
| Seattle | 4.5 | 77.8 % |
| Chicago | 3.2 | 69.2 % |
| New York | 5.4 | 81.5 % |

Averaging the results obtained from the rest of the 34 nodes in the network yields a 77.6% reduction in the state that has to be distributed to achieve network convergence. Applying the extra 50% state reduction that the reverse path calculation brings to the results above yields a total of 88.8% reduction in state distribution in the OS3E network, if the Source Routing approach is used. This significant state reduction directly improves the network convergence time.

## 3.2 Controller Placement

In Software-Defined WANs, the node-to-controller propagation latency bounds the ability to respond to network events. The best controller placement is the one that minimizes propagation delays, and consequently improves performance and fault tolerance [2]. The research in [2] finds that to minimize worst case latency in the OS3E network, the controller should be placed close to the geographic center of the network in Kansas City, while minimizing the average latency requires a placement in Chicago close to the high density areas in the network, *Figure 3*.

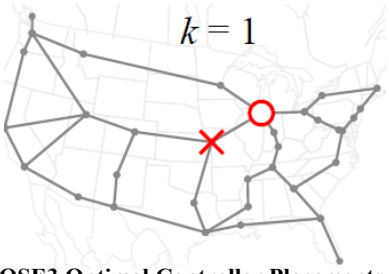


**Figure 3. OSE3 Optimal Controller Placements, (K=1, one controller used), *'X' is Kansas City and 'O' is Chicago*. [2]**

Using the Source Routing approach adds new dimensions to this problem. In the Source Routing approach the controller sends state information to ingress nodes only. Therefore, not all node-to-controller latencies have to be considered for optimizing the controller placement. Furthermore, not all the ingress nodes will communicate to the controller at the same rate, in particular with the use of the reverse path calculation approach. As a result, taking network traffic patterns into consideration, an optimal placement can be achieved based on the geographical distribution of the ingress nodes with the highest traffic demands, possibly with lower latency bounds than found in [2].

# 4. CONCLUSION AND FUTURE WORK

In this paper, we presented a new approach for routing in SDN that leverages variations of Source Routing and aims to open new possibilities for improvements in controllers' scalability, availability, and placement optimizations. Unlike previous source routing attempts, the proposed approach is simple, strict and orthogonal to IP, like MPLS tunneling. In addition, the approach utilizes the controller's knowledge of the network topology and adapts to the Software defined networks architecture.

Currently a model for Source-Routing-enabled OpenFlow switches is being developed to be used in NS3 simulation environment, to conduct performance analysis comparing the new approach to traditional OpenFlow. Our objective is to utilize the advantages that the Source Routing approach has in (1) Improving controller's scalability and network convergence time by reducing the state to be distributed in the network, (2) Improving controller's availability by locally handling link failures in real time, (3) Optimizing controller(s) placements to reduce the effects of propagation latencies in WANs.

# 5. REFERENCES

[1] Nick McKeown, et al., "OpenFlow: enabling innovation in campus networks", ACM SIGCOMM CCR, v.38 n.2, April 2008.

[2] Brandon Heller, Rob Sherwood, Nick McKeown, "The Controller Placement Problem", in HotSDN'12, August 2012, Helsinki, Finland.

[3] Peter Ashwood-Smith, "Software Defined Networking and centralized Controller State Distribution Reduction – Discussion of one Approach", IEEE Draft, July 2012.
Available at: http://www.ieee802.org/1/files/public/docs2012/new-ashwood-sdn-optimizations-0712-v01.pdf

[4] Internet2 Open Science, Scholarship, and Services Exchange. Web Site: http:://www.internet2.edu/network/ose