



A Comparative Study to Measure the Performance of Spanning Tree Protocol on Software Defined Networks

M. A. Arham and K. Thabotharan

Department of Computer Science, Faculty of Science, University of Jaffna

arham3648@gmail.com & thabo@univ.jfn.ac.lk



1. Introduction

- This work focuses on investigating the performance of the Spanning Tree Protocol in Software Defined Networking and proposes an improved Spanning Tree Protocol Algorithm.
- We used the Mininet emulator to emulate the network topology of a given SDN and carried out our experiments.



Broadcast streams in a network having a loop structure logically disconnect to avoid loop using STP.

2. Methodology

We carry out our work in two parts: first we implemented the basic STP method for preventing loops in SDN architecture and then in second part we propose a new method for preventing loops in the SDN architecture.

2.1 STP method for preventing loop in SDN architecture

2.1.1 SDN without STP Configuration

A broadcast storm occurs on the network. Finally, the network melts down, causing failure in all network links.

```
arham@arham-Inspiron-15-3567:~/Desktop$ cd Simple_Topology/
arham@arham-Inspiron-15-3567:~/Desktop/Simple_Topology$ sudo python
[sudo] password for arham:
Unable to contact the remote controller at 127.0.0.1:6633
(100.00Mbit 1ms delay 0.00000% loss) (100.00Mbit 1ms delay 0.00000%
100.00Mbit 1ms delay 0.00000% loss) (100.00Mbit 1ms delay 0.00000%
0.00Mbit 1ms delay 0.00000% loss) (10.00Mbit 1ms delay 0.00000% los
Mbit 1ms delay 0.00000% loss) (8.00Mbit 1ms delay 0.00000% loss) (8
ing hosts
h1 h2 h3
(100.00Mbit 1ms delay 0.00000% loss) (10.00Mbit 1ms delay 0.00000%
0.00Mbit 1ms delay 0.00000% loss) (10.00Mbit 1ms delay 0.00000% los
Mbit 1ms delay 0.00000% loss) (10.00Mbit 1ms delay 0.00000% loss) (
CLI
*** Starting CLI:
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
```

Hosts becomes unreachable

Methodology

2.1.2 SDN with STP Configuration

The switches exchange information among themselves using bridge protocol data units (BPDU) and will then listen in on all ports for this BPDU message.

```
View Search Terminal Help
arham@arham-Inspiron-15-3567:~/Desktop/ryu-manager$ ryu-manager simple_sw
ing app simple_switch_stp_13.py
ing app ryu.controller.ofp_handler
Initiating app None of Stp
ing context stplib
Initiating app simple_switch_stp_13.py of SimpleSwitch13
Initiating app ryu.controller.ofp_handler of OFPHandler
[INFO] dpid=0000000000000001: Join as stp bridge.
[INFO] dpid=0000000000000001: [port=1] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000001: [port=2] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000002: Join as stp bridge.
[INFO] dpid=0000000000000002: [port=1] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000002: [port=2] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: Join as stp bridge.
[INFO] dpid=0000000000000003: [port=1] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: [port=2] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: [port=3] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: [port=1] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: [port=2] DESIGNATED_PORT / LISTEN
[INFO] dpid=0000000000000003: [port=3] Receive superior BPDU.
[INFO] dpid=0000000000000003: [port=1] DESIGNATED_PORT / BLOCK
[INFO] dpid=0000000000000003: [port=2] DESIGNATED_PORT / BLOCK
```

Select Root Bridge & Ports

Hosts becomes reachable

2.2 The Proposed Algorithm

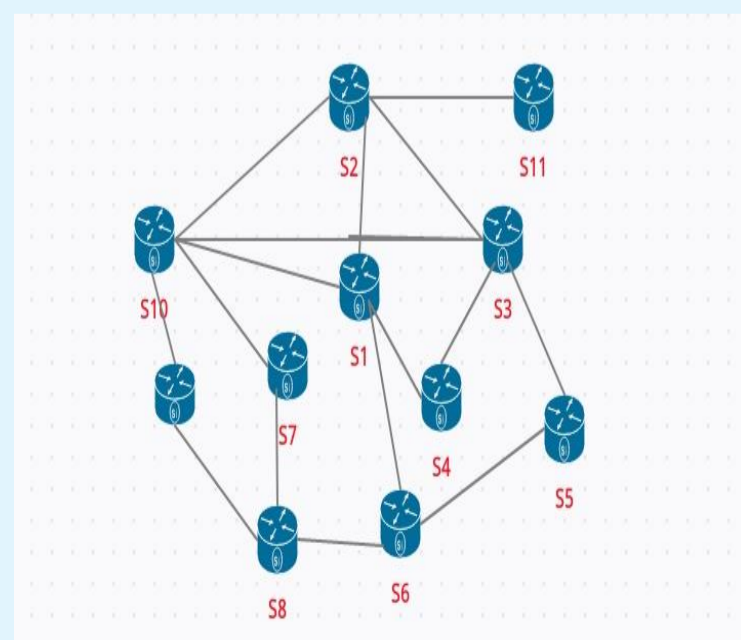
As a result, only activated switches will remain as switches(S). Here we check the following two conditions:

Condition-1: New set S (after deleting S1) has intersection with all sets of N(Si) or not. { N(Si) set of neighbors of the switch Si}

(If in the condition 1, switch Si is a non-activated switch, then it will be deleted and then it checks the condition 2)

Condition-2: In addition, the SDN controller has to check the sub-graph to find out that it is a connected graph or not.

3. Test Results



Input: a set of activated switches & non-activated switches

Testing the Proposed Algorithm

Input S = {S1, S2,, S11}
Neighbors of Si (i = 1 to 11)
N(S1) = {S2, S3, S4, S6, S10},
N(S2) = {S1, S3, S10},,
N(S11) = {S2}

- Step-1= new S = {S2, S3,, S11}
- Step-2 = Check condition 1
- Step-3 = Check condition 2
- Step-4 = go to Step-1

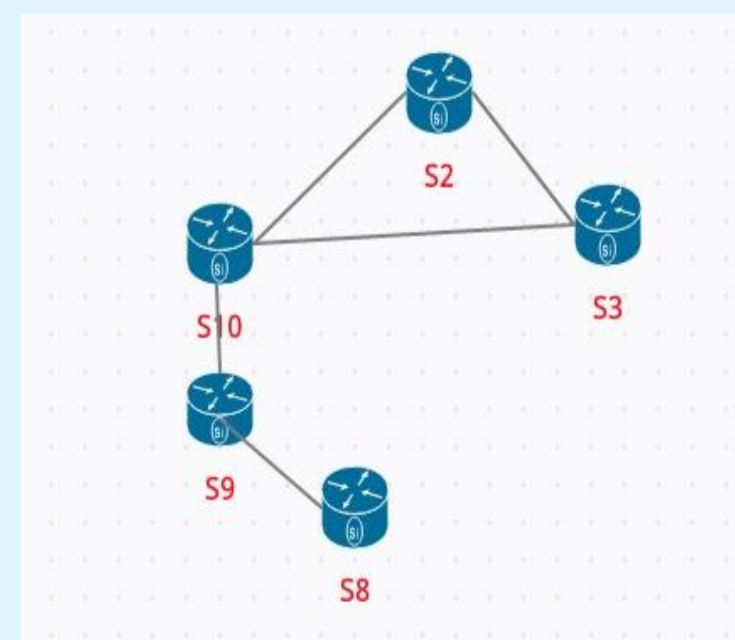
Output = Set of Activated Switches S = {S2,S3,S8,S9,S10}

Algorithm for selecting activated switches

Input: S - The set of switches,
N(Si) - the neighbors of the switch Si,
The network graph

Output: The set of activated switches

- for each Si in S // Si = S1, S2,
- newS ← remove Si from S
- for all N(Si):
- if intersection of the newS and N(Si) = ∅:
- add Si to the newS
- break
- End
- End
- if Si is not in the newS: // Check Condition 2
- H ← sub-graph from the network containing nodes of the newS
- if H is not connected:
- add Si to the newS
- End
- End
- The set of activated switches ← S



Output: Sub-graph containing only activated switches

4. Experimental setup

- To evaluate the proposed method we used Ryu as controller of the network.
- Here we used Mininet Simulator to create the topology of the network.
- The "pingall" command in Mininet can be seen all the hosts could communicate together or not.
- Implemented the basic STP method and proposed method for preventing loops in SDN architecture.

5. Discussion and Conclusion

- SDN is a novel networking paradigm in which the control plane is decoupled from the forwarding plane.
- STP is used to prevent loop in layer 2.
- In our method, the SDN controller received much fewer broadcast packets and CPU utilization.

6. References

[1]. Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S. and Uhlig, S., Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1), pp. 14-76. 2014.

[2]. "Spanning Tree" [online]. Available: <https://osrg.github.io/ryu-book/en/html/spanning-tree.html>

[3]. Irawati, I.D. and Nuruzzamanirridha, M., "Spanning Tree Protocol Simulation Based on Software Defined Network Using Mininet Emulator." In *International Conference on Soft Computing, Intelligence Systems, and Information Technology*, pp. 395-403, 2015.