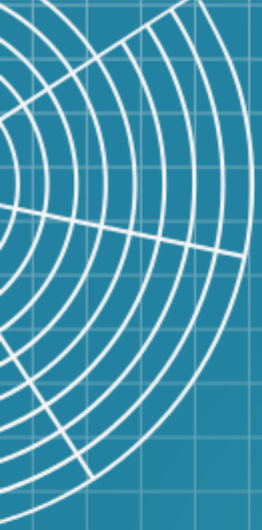
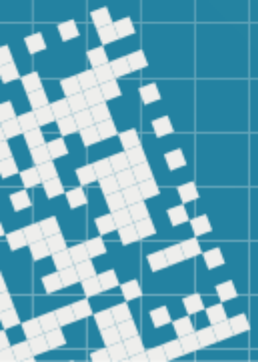
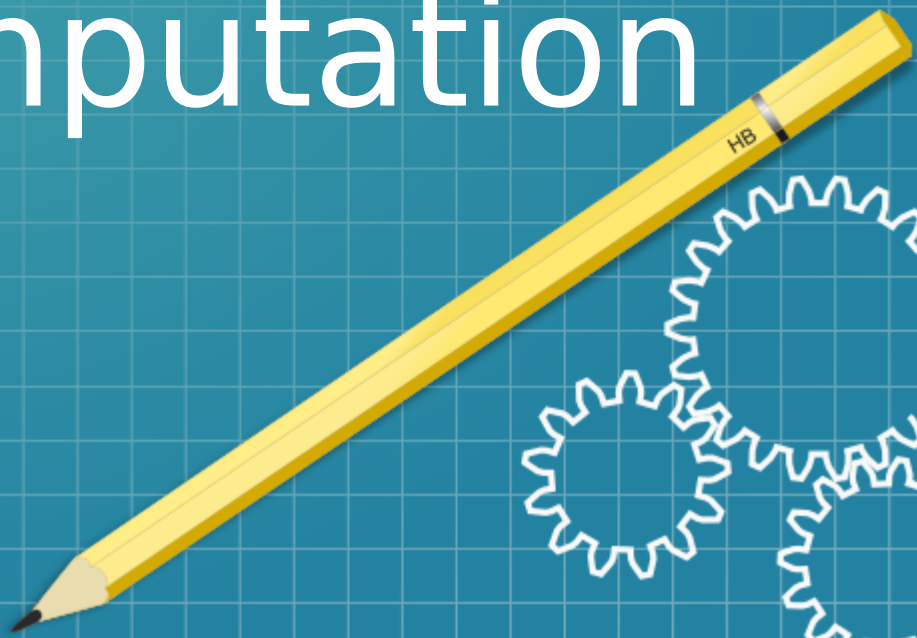
Software Defined Networking: Distributed systems and Trust computation







Abstract

The internet, since the advent of ARPANET, has come along a very long way. It has undoubtedly changed millions of lives and even now is in its infancy. Software Defined Networking (SDN) is presented as a paradigm shift in this regard. It strives to standardize the networking on all levels. This is an initiative to redesign the current networking stack and compartmentalize into three main planes, the data plane, the control plane and the management plane, respectively moving from bottom up. We, here, have put an effort to augment the idea of SDN to a more distributed framework. Using cleverly designed topologies like Spine leaf, we demonstrate the interconnection of controllers using relay system designed from bottom up as the first phase. The second phase, on other hand, acknowledges the need to secure such translations and we try to mitigate Denial of Service (DoS) attacks on the control plane.

What are Software Defined Networks?

- In a software-defined network, a network engineer or administrator can shape traffic from a centralized control console.
- The centralized SDN controller directs the switches to deliver network services wherever they're needed.
- A typical representation of SDN architecture comprises three layers: the application layer, the control layer and the infrastructure layer.

SDN Architecture

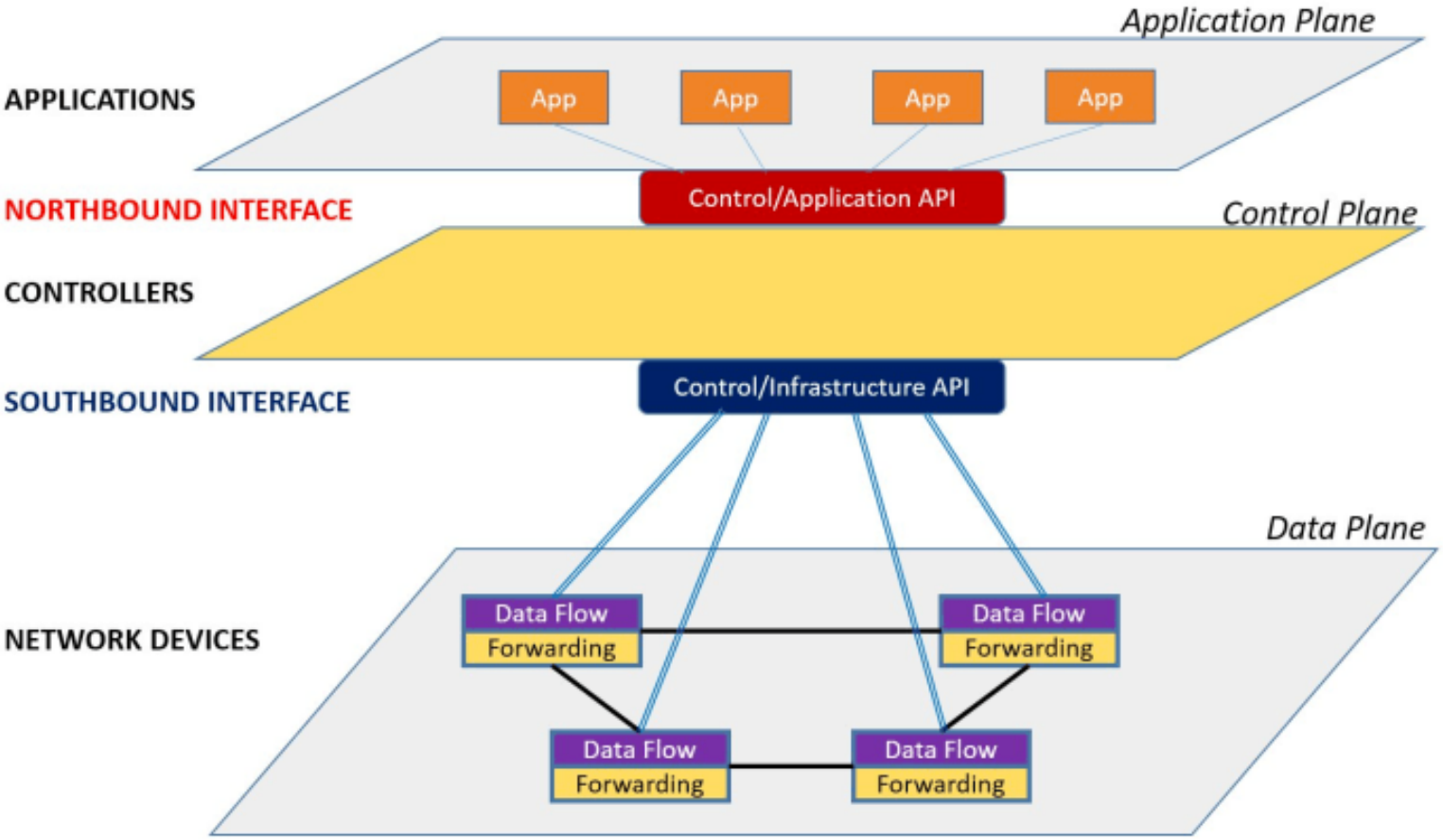


Figure 1 - Software-Defined Networking – A high level architecture

Introduction

- This is an initiative to redesign the current networking stack and compartmentalize into three main planes, the data plane, the control plane and the management plane.
- Using cleverly designed hybrid topologies, we demonstrate inter-controller connection in a distributed environment in first phase.
- The second phase acknowledges the need to secure such translations and we try to mitigate Denial of Service (DoS) attacks on the control plane.

Current Situation

1) Current SDN industry standard controller implementations do not inherently support distributed inter-controller communication.
2) Some of them who do, use infeasible and expensive mesh topologies.
3) SDN, thus, is limited to single controller and non-scalable networks.
4) Inter-controller communication of different types is also a problem.

Limitations of Current System

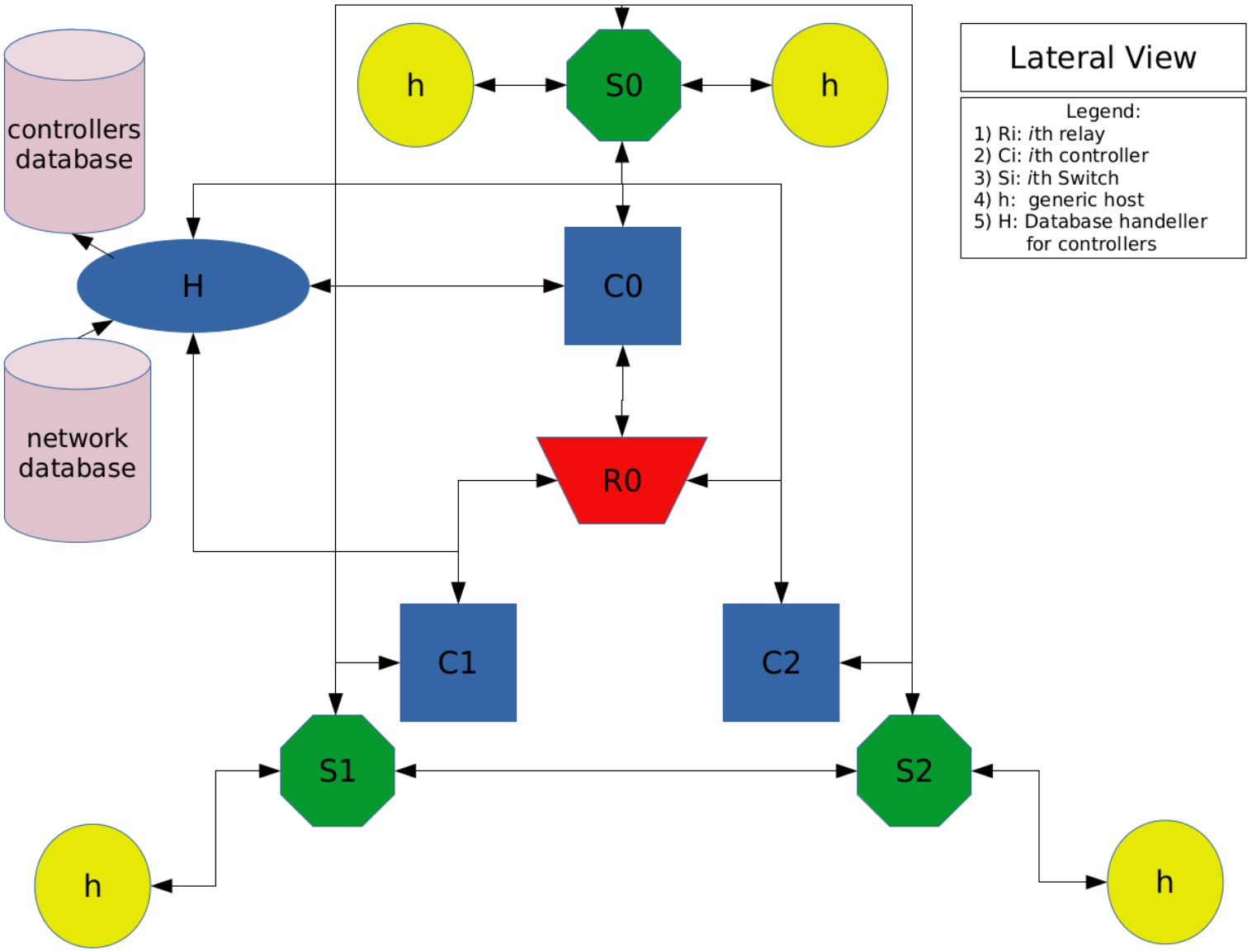
1) Less scalability factor.
2) More energy consumption.
3) Greater expense for physical cables.
4) More vulnerable area to secure.
5) $(n-1)^n$ connections within controllers due to Mesh topology.

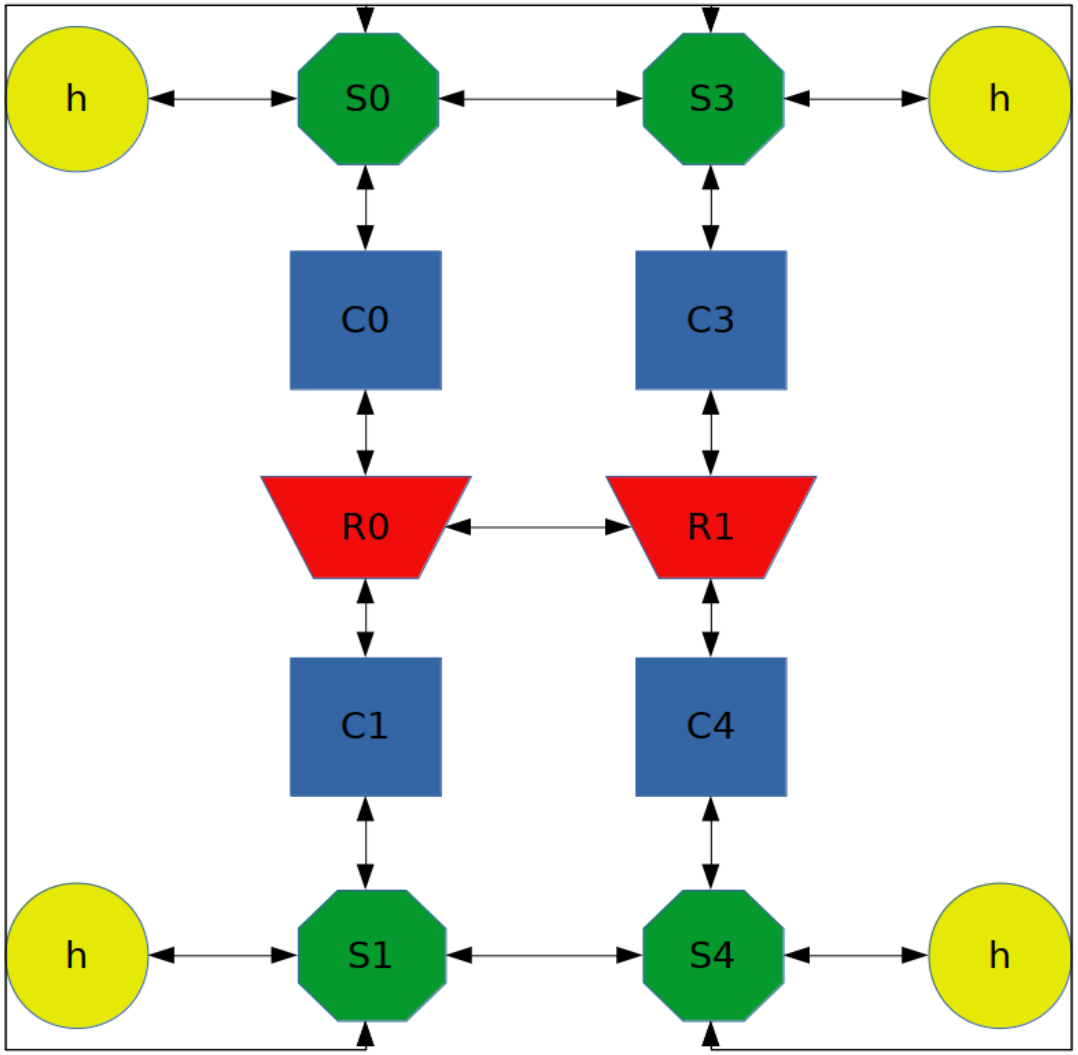
Proposed System

1) We propose a relay to act as a bridge between the controllers in a distributed system.
2) The relays can be sub-relayed as per geographical requirements.
3) Controllers use relay as proxy to broadcast flow query in the network.
4) A duplex connection between each controller and relay facilitates simultaneous broadcast and reply.
5) Bottlenecks are eliminated using frequent multi threaded constructs.
6) A TCP server listening for particular connections at controller handler, relay and controller as well for the duplex connection.
7) Python/java has been used as controller implementation language to interface with dynamic shared libraries coded in pure C, while relay engine remaining in pure C.

Review 1

Designed Architecture





Cross-Sectional View (Observer's left)

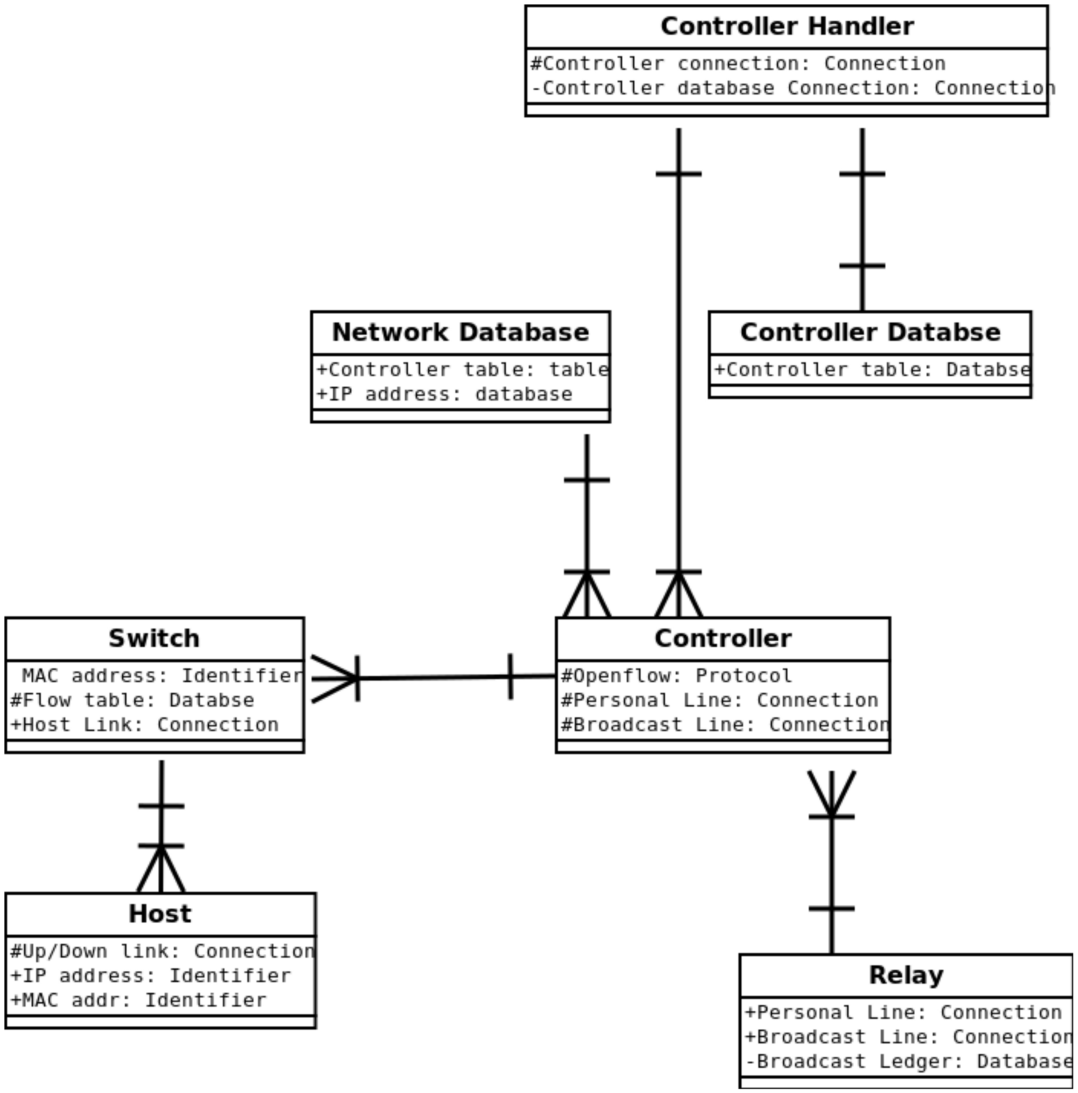
Legend:

1) Ri: *i*th relay
2) Ci: *i*th controller
3) Si: *i*th Switch
4) h: generic host
5) H: Database handeller for controllers

How the Architecture works?

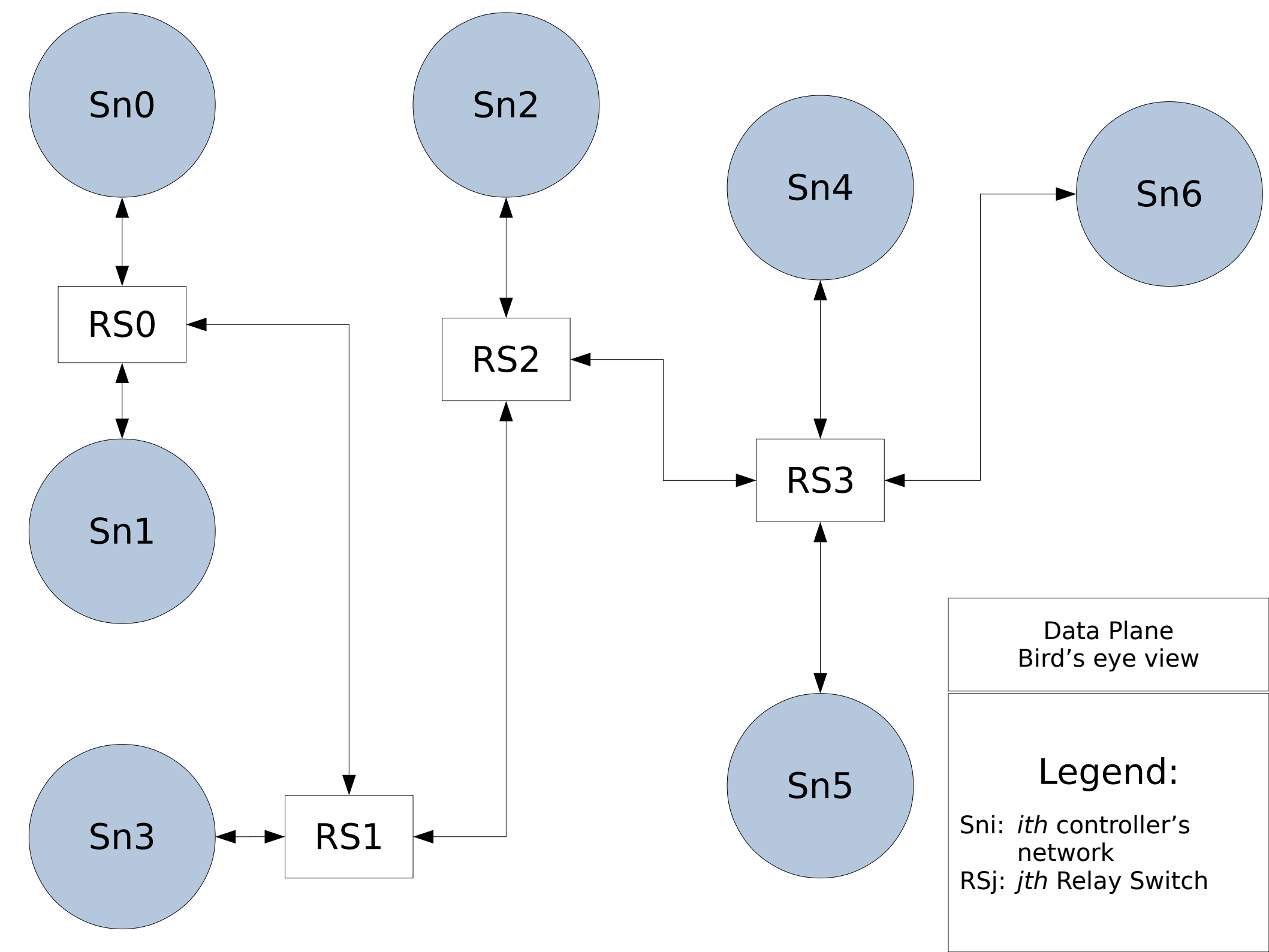
1) Database docker image starts
2) Controller handler start connecting to ‘controllers’ database.
3) Controllers boot sending information to controller handler to be registered.
4) Topology simulator fetches the Controller Database
5) User Input for the required number of hosts and switches.
6) Random pairing for switches and hosts using the libsodium library.
7) Random number of hosts connected to the switches.
8) Topology is described as a 3d array, where the entries in database are separate tables with each controller has a separate table with its IP as the name for each table.
9) Class A Ips are automatically assigned by the Mininet.
10) Static Ips assigned to each controller on connection.
11) Standalone controller connection through relay.
12) Network database accessed by controller upon the connection.

Workflow


Controller Handler
#Controller connection: Connection
-Controller database Connection: Connection
Network Database
+Controller table: table
+IP address: database
Controller Databse
+Controller table: Databse
Switch
MAC address: Identifier
#Flow table: Databse
+Host Link: Connection
Controller
#Openflow: Protocol
#Personal Line: Connection
#Broadcast Line: Connection
Host
#Up/Down link: Connection
+IP address: Identifier
+MAC addr: Identifier
Relay
+Personal Line: Connection
+Broadcast Line: Connection
-Broadcast Ledger: Database

Review 2

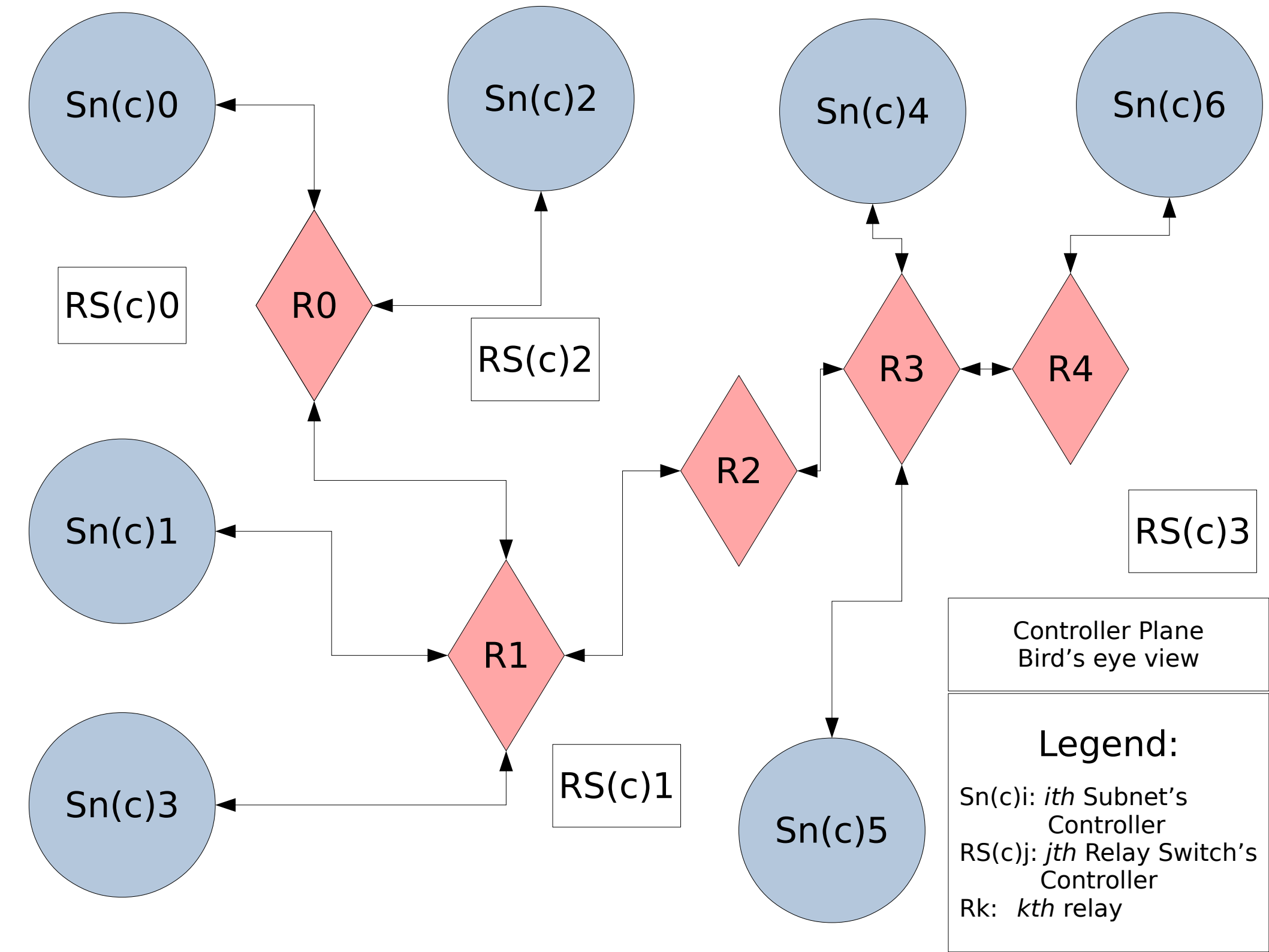
Some updations in Architecture


Sn0
Sn2
Sn4
Sn6
RS0
RS2
RS3
Sn1
Sn3
RS1
Sn5
Data Plane
Bird's eye view
Legend:
Sni: *ith* controller's network
RSj: *jth* Relay Switch

The New Data Plane

The data plane now consists of three separate components, namely:

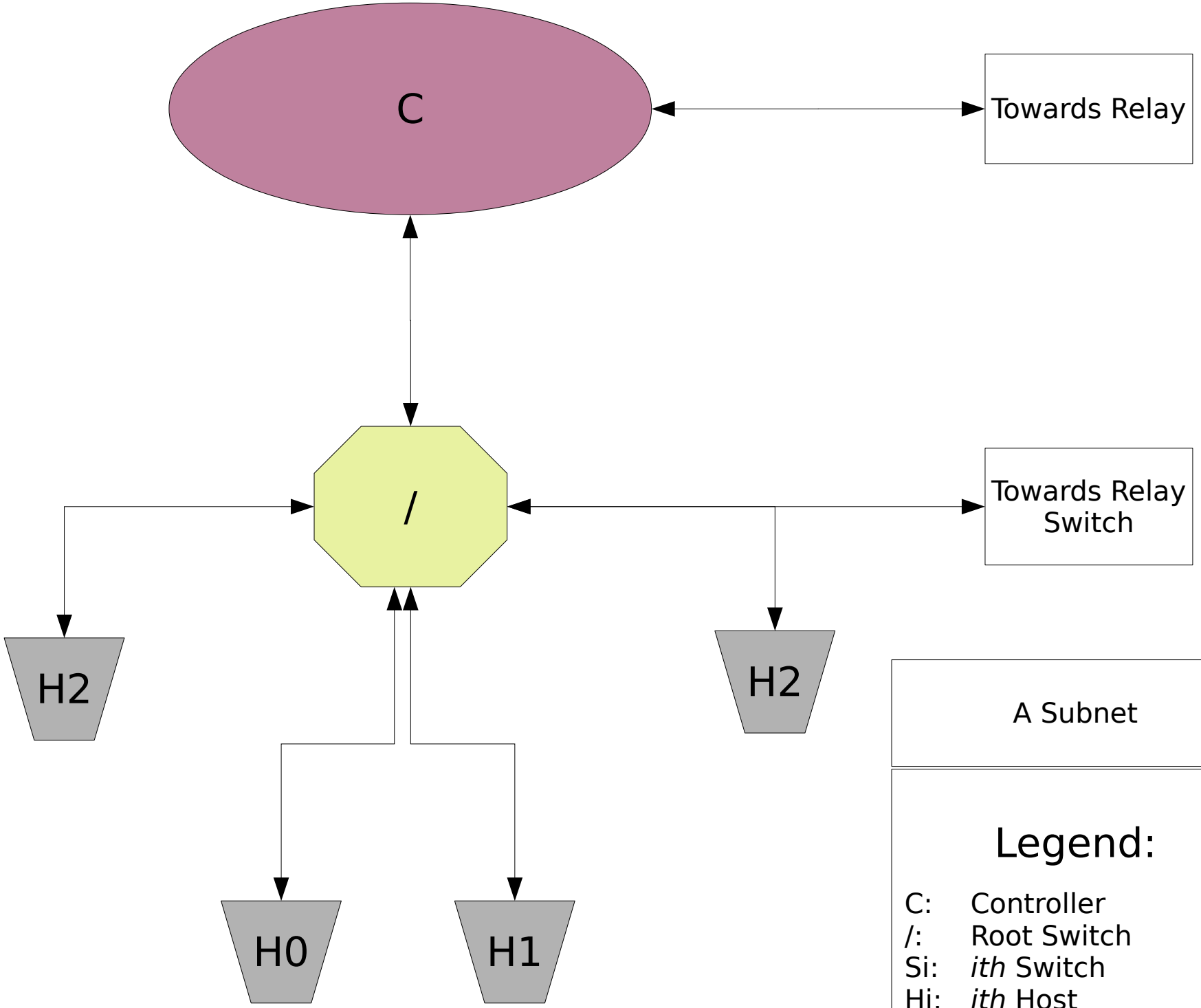
- The Root Switch:
  - A unique and mandatory entity for every controller subnet.
  - Every host/switch within a subnet have a connection to it.
- The Relay Switch:
  - The communicator between the subnets.
  - *n* Relay Switches in whole network are connected via (*n-1)* connections
  - Any number of subnets can be managed by a Relay Switch.
  - They form a straight chain within themselves.
- A generic host:
  - Represents a host which is a generic node.


Sn(c)0
Sn(c)2
Sn(c)4
Sn(c)6
RS(c)0
R0
RS(c)2
R3
R4
R2
Sn(c)1
RS(c)3
R1
Sn(c)3
RS(c)1
Sn(c)5
Controller Plane
Bird's eye view
Legend:
Sn(c)i: ith Subnet's Controller
RS(c)j: jth Relay Switch's Controller
Rk: kth relay

The New Controller Plane

The redesigned controller plane has three separate entities, namely:

- The Root Switch Controller:
  - It serves as the OpenFlow controller for every controller subnet.
  - Are interconnected via Relay for real time controller communication.
- The Relay:
  - This is a standalone multi-threaded TCP server.
  - Helps in real time connection between the Root Switch Controllers.
  - Forms duplex connections to every Root Switch Controller.
- The Relay Switch Controller:
  - A generic L2 learning switch controller template.
  - Enforces OpenFlow protocol on the Relay Switches.


C
Towards Relay
/
Towards Relay
Switch
H2
H2
H0
H1
A Subnet
Legend:
C: Controller
/: Root Switch
Si: *ith* Switch
Hj: *jth* Host

So what is new?

Old Topology	New Topology
Relay served as a **host lookup**.	**Relay** serves as **communication entity**.
Controller **Plane congestion was high**.	Controller plane **congestion is low.**
Absence of Relay Switch might result in controller flow loops (infinite).	**Presence of Relay Switches** mitigates controller flow loops.
Subnet **planning was difficult and asymmetrical.**	Due to mandated **Root Switch**, **subnet planning is symmetric and symmetrical.**

Lets Talk Numbers!

n(Subnets)/ n(Hosts/ Subnet)	Throughput (Bytes/sec)	Bandwidth (Bytes/Sec)	Delay (Hz)	Packet Loss (%)	Packet Delivery Ratio	Flow request Rate (Hz)
1/300	91.744	868.198	22.114	0.552	1	(22.579)
2/50	95.232	898.652	7.264	0.525	1	(25.115) (21.685)
2/150	95.168	846.332	22.837	0.526	1	(23.605) (20.311)
3/50	94.484	893.565	10.958	0.559	1	(22.466) (24.053)
4/75	95.744	845.227	23.018	0.586	1	(19.425) (23.220)

TODO

1) A blacklisting/whitelisting procedure to dynamically remove, reinsert or re-route the traffic of a particular host in whole network.
2) To develop an Intrusion Detection System and report real time statistics to the whole network.
3) To discover new types of applications for the Relay server.
4) To extensively test and report the results and values derived from observing topology under attack.
5) To implement similar concept in various types of topologies.

Thank You!

Prepared and Presented by:
Naman Arora
RA1511003010235
Nikhil Gupta
RA1511003010245