Self–Adaptive Routing

Arthur van Kleef
Marvin Rambhadjan

System and Network Engineering

June 30, 2010

Introduction

Network management involves complex tasks.

Congestion Control	Load Balancing, Rerouting
Quality of Service	i.e. Voice and Video
Provisioning	Resource Reservation

Existing protocols

Review of existing protocols based on adaptivity

Protocol	Layer	Capabilities
OSPF	3	Path Cost
BGP	3	Local Pref, MED, Next-Hop, AS-Path
MPLS(-TE)	2.5	Explicit LSP's
PBB(-TE)	2	Service and Trunk coupling
STP	2	Path Cost, Priority

The control plane becomes more complex by adding new protocols and protocol extensions.

Control Plane

The control plane performs the following functions:

- Control Connections
- Disseminate connectivity information
- Calculate optimal path

Software programs are good at handling complex tasks.

Separate forwarding plane from control plane (ForCES).

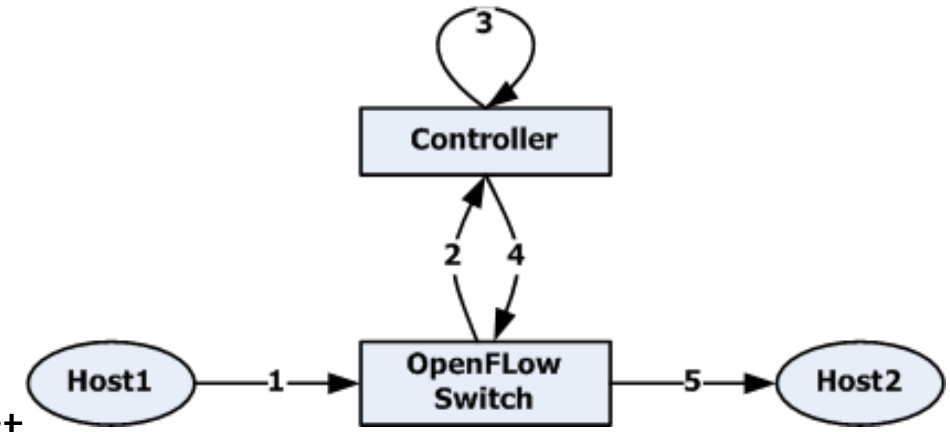
Research Questions

- What is the architecture of a network that supports a software control plane?
- If the control plane becomes software, what is the general pattern of the programs that implement routing and network management?

OpenFlow

OpenFlow Operation

1. Packet sent to switch
2. Switch passes packet to controller
3. Control decides action
4. Install flow entry
5. Packet forwarded to the host



OpenFlow Tuple

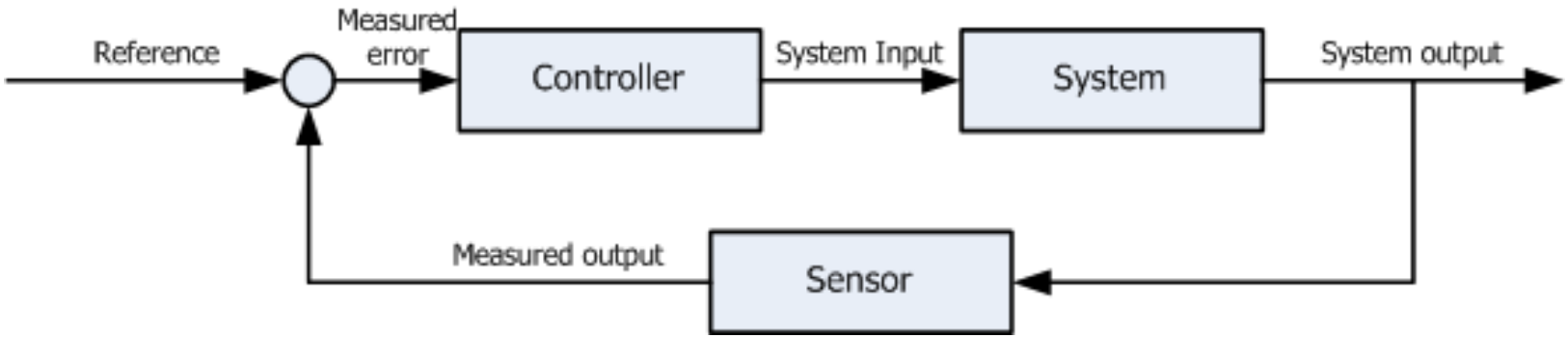
Flows describing traffic.

Ingress Port	Ethernet src	Ethernet dst	Ethernet type	VLAN id	VLAN prior-ity	IP src	IP dst	IP proto	IP ToS	TCP UDP src port	TCP UDP dst port

- Actions
  - Send to controller
  - Forward
  - Flood
  - Drop

If switch does not have a flow–entry that matches a certain packet header then it forwards a packet to the controller.

Network Control Program



Reference Desired behavior

Controller Controller responsible for taking forwarding decisions

System Programmable network infrastructure

Sensor Monitoring for switch statistics

Test Environment

Resources were limited, so can we virtualize OpenFlow networks?

- **Hosts** User Mode Linux
- **OpenFlow Switches** User Mode Linux + Open vSwitch
- **Connections** Virtual Distributed Ethernet (VDE)
- **Controller** NOX OpenFlow Controller

Large topologies, runs on moderate hardware, flexible

Control Program components

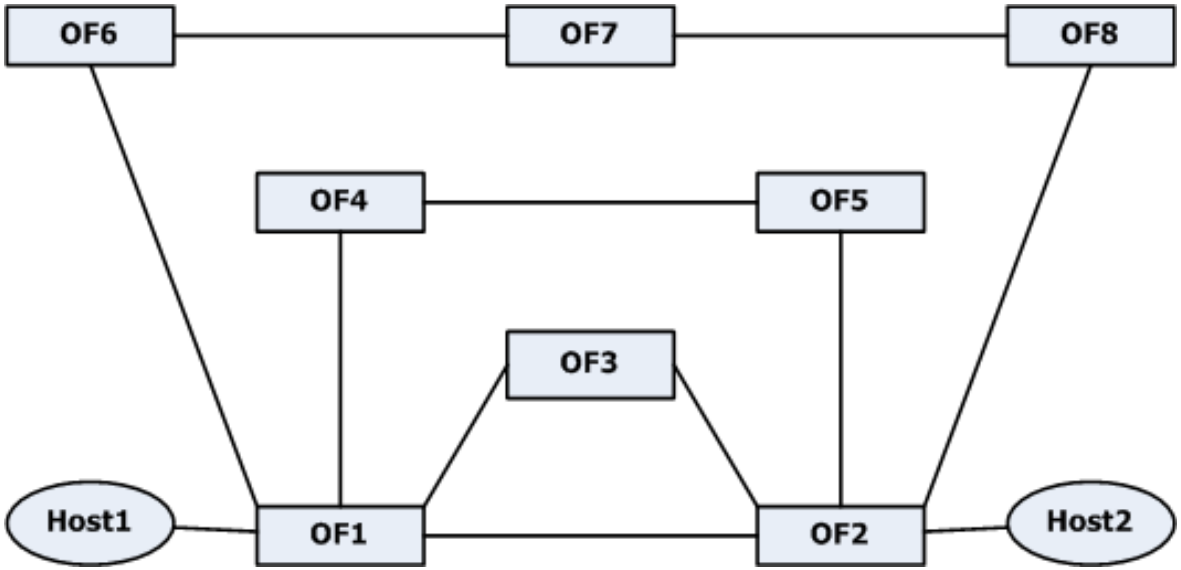
Control Program consists of different components that:

- Maintain network topology in a graph
- Track the location of end hosts
- Monitors traffic and link utilization
- Calculates paths between source and destination
  - Shortest Path
  - All possible paths
- Installs flow entries on OpenFlow switches

NCP Operation

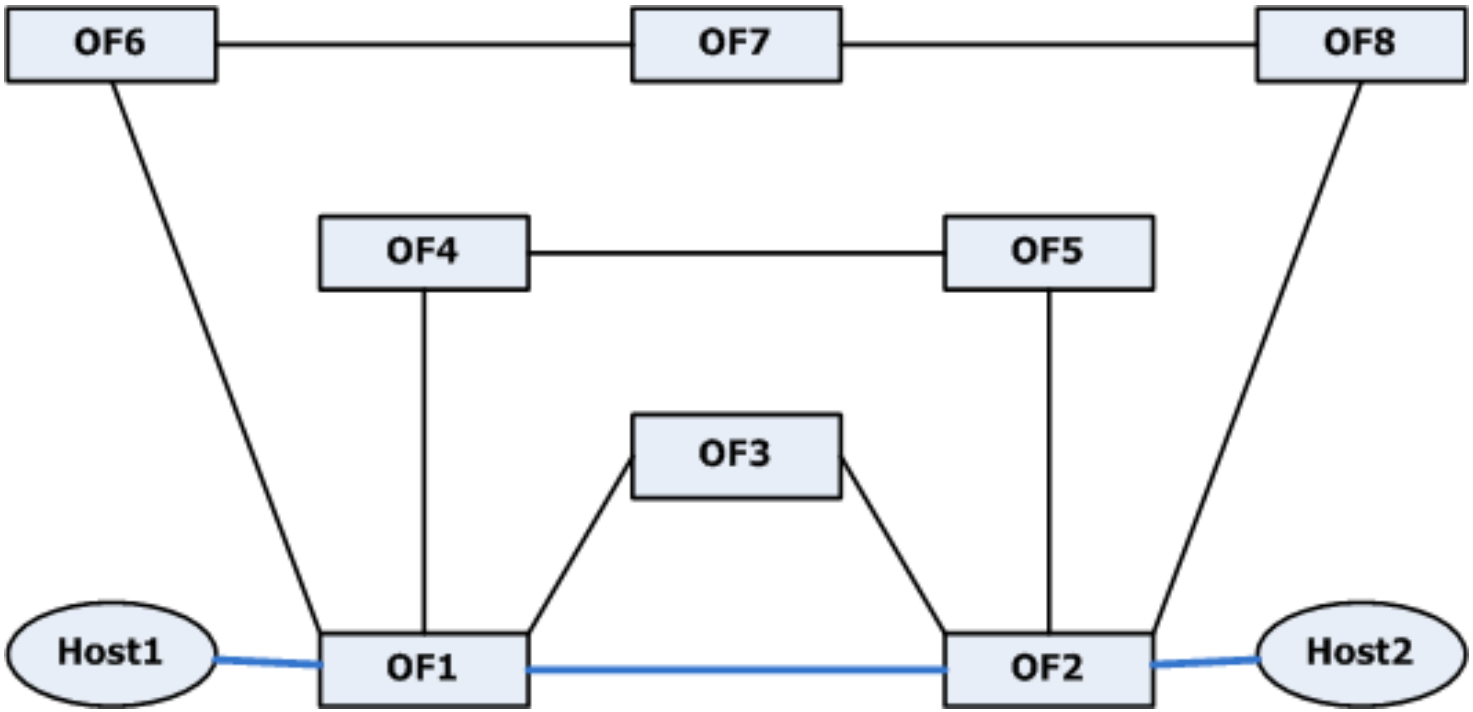
1. Receive packet_in event
2. Learn ethertype
3. Locate destination node in the topology
4. Add or consider active policy
5. Calculate path to destination
6. Return flow entry

Example: Case Network



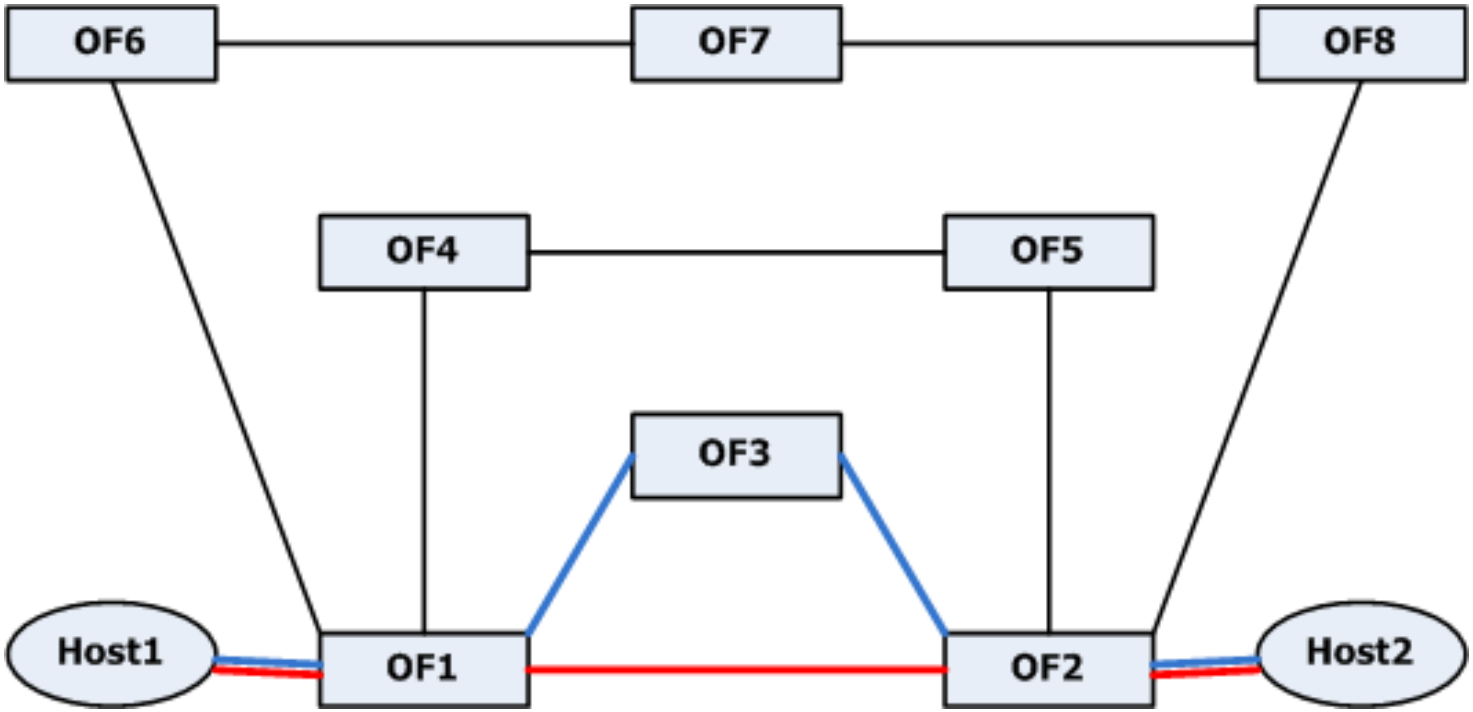
- **OF***N* Open vSwitches
- **Host***N* network hosts
- Virtual Distributed Switches

Example: Single Traffic Flow



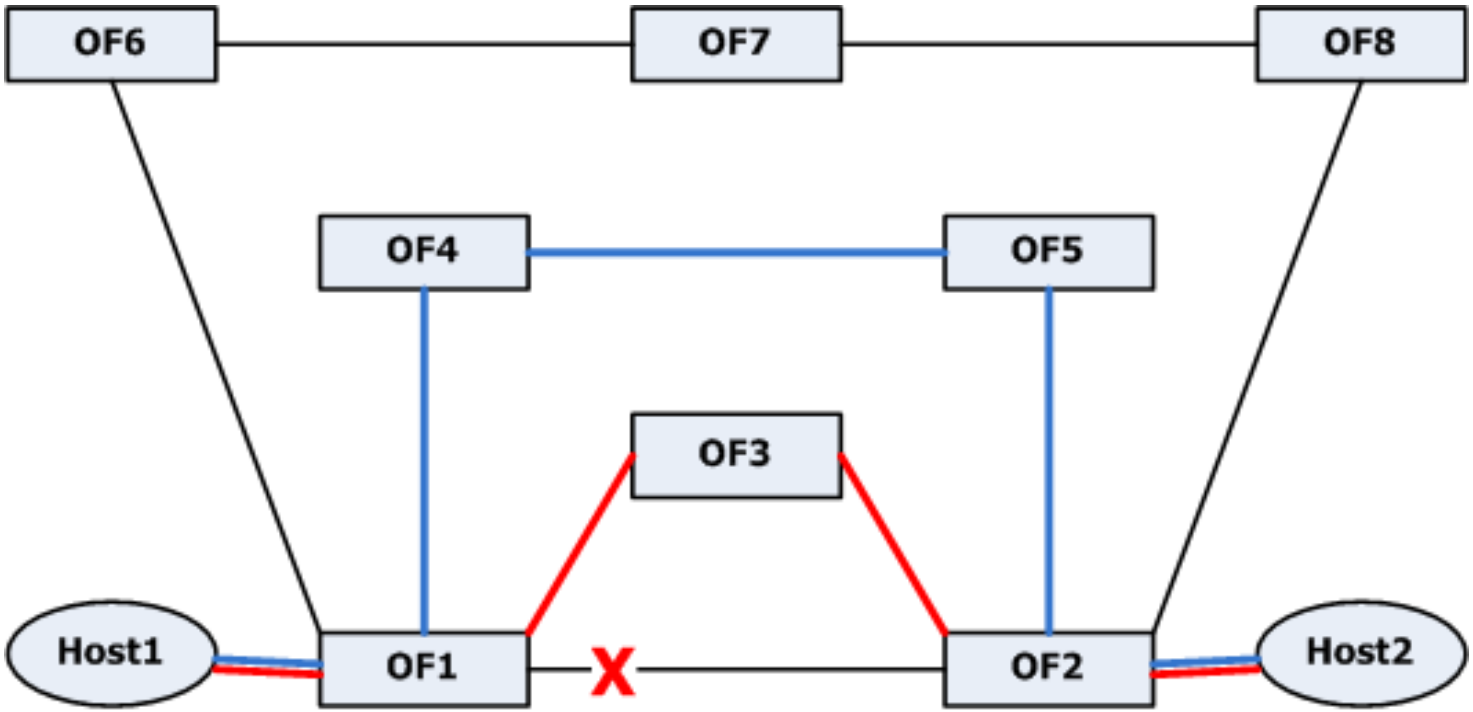
One traffic flow

Example: Extra high priority traffic flow



Another traffic flow with a higher priority

Example: Link Failure



Two traffic flows with a link down

Scalability of Programmable Networks

- FlowVisor is a transparent proxy between OpenFlow switches and controller(s).
- Multiple FlowVisor controllers can be added to balance the load between the controllers.
- The network can be "sliced" and the control is delegated to a controller and are based on the following criteria:

Layer	Options
4	src / dst TCP or UDP, ICMP code
3	src / dst IP address, IP Protocol, IP TOS
2	src / dst Ethernet address, VLAN
1	Physical switch port

Conclusion

- OpenFlow provides an architecture that support separation of the control plane and the forwarding plane.
- The general pattern of the software control plane is a feedback control loop.
- The test environment presented is very useful for OpenFlow experiments.

Questions

?