PolicyCop: An Autonomic QoS Policy Enforcement Framework for Software Defined Networks

Md. Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, and Raouf Boutaba

David R. Cheriton School of Computer Science, University of Waterloo

[mfbari|sr2chowdhury|r5ahmed|rboutaba]@uwaterloo.ca
Roadmap

• Motivation
• Our Contribution
• Our Approach
• Simulation Results
• Conclusion & Future Work
Motivation

- Network management systems are being continuously challenged to satisfy application QoS requirements
- Policy based management can tackle these challenges
- Recently emerging field of Software Define Networking (SDN) can provide features like:
  - Per flow control
  - Dynamic flow aggregation
  - Dynamic traffic classes
  - Avoid protocol clutter problem
  - Ease of deployment
- Policy based management can be coupled together with SDN to provide autonomic policy based management
Roadmap

• Motivation
• Our Contribution
• Our Approach
• Simulation Results
• Conclusion & Future Work
Our Contribution

• We have designed and implemented a prototype of an autonomic QoS policy enforcement framework, PolicyCop that:
  • Leverages the programmability offered by SDN for
    • Dynamic traffic steering
    • Flexible Flow level control
    • Dynamic traffic classes
    • Custom flow aggregation levels
  • Monitors the network to detect policy violations
  • Reconfigures the network to reinforce the violated policy
Roadmap

- Motivation
- Our Contribution
- Our Approach
- Simulation Results
- Conclusion & Future Work
Our Approach
PolicyCop: Control Plane

PolicyCop

Admission Control  Routing  Device Tracker  Statistics Collector  Rule DB

SDN Controller

Control Plane

OpenFlow

NB_API
PolicyCop: Management Plane

PolicyCop

Policy Enforcer
- Topology Manager
- Resource Manager

Policy Adapter
- Policy Provisioning

Management Plane

Policy Validator
- Event Handler
- Event Monitor

Policy DB

Manual Action
- Policy Manager

Autonomic Action

NB API
Our Approach (Workflow)

1. Start
2. Monitor traffic
3. Check policy
4. Has violations?
   - Yes: Check event type
     - Requires autonomous action?
       - Yes: Reprovision resources
       - No: Forward to Manager
     - No: Adapt to policy
   - No: Reprovision resources
5. Policy DB
6. Resource status
7. Topology
8. Event classifier
9. Manager takes action
Roadmap

• Motivation
• Our Contribution
• Our Approach
• Simulation Results
• Conclusion & Future Work
Experimental Setup

- 5 Open vSwitches (OVSs) & 4 hosts
- OVSs’ interconnected with GRE tunnels to simulate bandwidth and latency
- One floodlight controller
- Used `iperf` to generate traffic
Use Case 1: Link Failure

[Diagram of network topology with nodes H4, H1, S5, S1, S3, S2, S4, H3, H2, and Floodlight Controller. An 'X' indicates a link failure.]

[Graph showing throughput (Mbps) over time (seconds) with three traces labeled H1 to H2, H2 to H4, and H3 to H4, with a highlighted interval between 15 and 20 seconds.]
Use Case 2: Throughput Violation
Roadmap

• Motivation
• Our Contribution
• Our Approach
• Simulation Results
• Conclusion & Future Work
Conclusion & Future Work

• We have
  • Presented the design of PolicyCop, an autonomic QoS policy enforcement framework for SDN
  • Demonstrated the effectiveness of PolicyCop through a working prototype

• Our next step
  • Implement all component of PolicyCop
  • Interface with existing policy specification languages (e.g., Ponder)
  • Provide a collection of controller applications for other network management function
Questions?