Network Statistics for OpenFlow

Miroslav Cupák, Masoud Hosseinkhani, Maryam Samizadeh

05/12/2012

Introduction

- OpenFlow provides some very basic statistics per flow
- Network Management requires wider statistics to make wise decisions about networks
- New metrics are needed to be measured

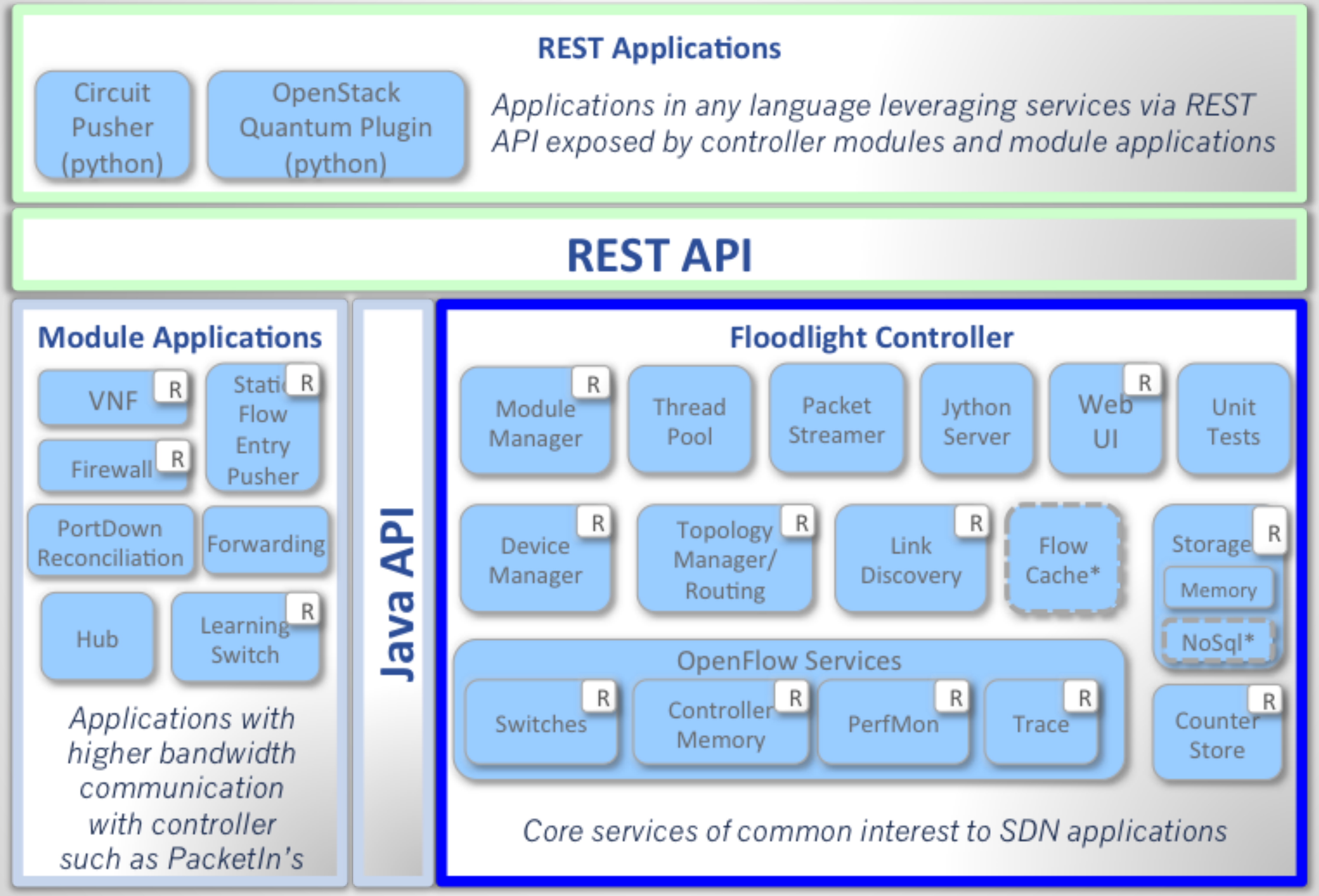
Floodlight Controller

- Our group decided to work on the Floodlight controller
- Floodlight is a Java-based controller
- Widely used and supported by the SDN community

Initial Thoughts!

- Our focus at first was on latency and jitter
- Data-plane metrics!
- OpenFlow is really a control-plane protocol
- How accurate they can be!?
- Let's work on new useful metrics!

Architecture



REST Applications

Circuit Pusher (python)

OpenStack Quantum Plugin (python)

Applications in any language leveraging services via REST API exposed by controller modules and module applications

REST API

Module Applications

VNF R

Firewall R

Static Flow Entry Pusher R

PortDown Reconciliation

Forwarding

Hub

Learning Switch R

Applications with higher bandwidth communication with controller such as PacketIn's

Java API

Floodlight Controller

Module Manager R

Thread Pool

Packet Streamer

Jython Server

Web UI R

Unit Tests

Device Manager R

Topology Manager/ Routing R

Link Discovery R

Flow Cache*

Storage R

Memory

NoSql*

OpenFlow Services

Switches R

Controller Memory R

PerfMon R

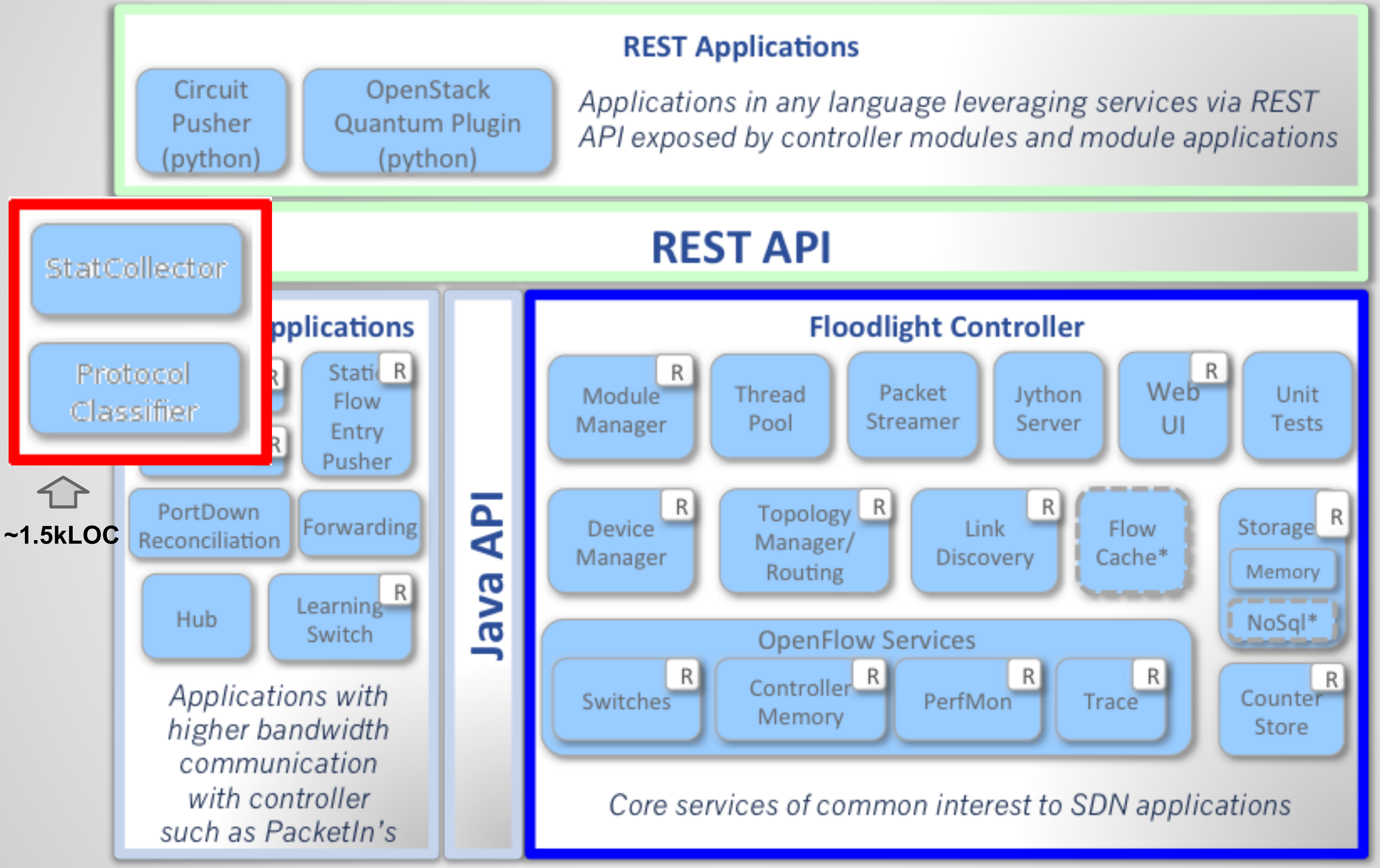
Trace R

Counter Store R

Core services of common interest to SDN applications

* Interfaces defined only & not implemented: FlowCache, NoSql

Architecture



REST Applications

Circuit Pusher (python)

OpenStack Quantum Plugin (python)

Applications in any language leveraging services via REST API exposed by controller modules and module applications

REST API

StatCollector

Protocol Classifier

~1.5kLOC

pplications

R

R

Stati R Flow Entry Pusher

PortDown Reconciliation

Forwarding

Hub

Learning R Switch

Applications with higher bandwidth communication with controller such as PacketIn's

Java API

Floodlight Controller

Module R Manager

Thread Pool

Packet Streamer

Jython Server

Web R UI

Unit Tests

Device R Manager

Topology R Manager/ Routing

Link R Discovery

Flow Cache*

Storage R

Memory

NoSql*

OpenFlow Services

Switches R

Controller R Memory

PerfMon R

Trace R

Counter R Store

Core services of common interest to SDN applications

* Interfaces defined only & not implemented: FlowCache, NoSql

Link Bandwidth

- what?
  - consumed data resources, current speed of the links
  - not theoretical maximal throughput, real bandwidth!
- why?
  - planning/scheduling, load balancing, troubleshooting
- how?
  - controller configured to load StatCollector module at startup
  - a periodic task scheduled to run in a separate thread
  - stats for the last interval serialized and available via REST as JSON

Link Bandwidth

- how?
  - finding links - parsing the topology
    - based on LinkDiscoveryManager and TopologyService
    - LLDP packets (a link established if an LLDP is sent out one port and the same LLDP is received on another port)
  - determining real bandwidth
    - based on byte per-port counters
    - periodic queries of switches to get the total value
    - subtract the values measured last time to track values over time
    - average over sent+received data on both ends and divide by time

Link Bandwidth

- how accurate?
  - based on per-port switch counters
  - switches might update the counters only once every couple of seconds
  - limited to switches attached to the controller
  - out of date by the time they're processed by the controller (insignificant)
  - UI displays the last completed interval
  - different delay for switches (reduced by average)

Port Bandwidth

- what?
  - more detailed view on each end of a link
- why?
  - step towards most of the other stats
- how?
  - periodic task started at startup exposing data (REST)
  - no need to search the topology
- how accurate?
  - like link bandwidth except for switch differences

Flow Bandwidth

- what?
    - data over time link usage by flows
- why?
    - proposed metric, interesting insight into network usage
- how?
    - periodic task started at startup exposing data (REST)
    - active polling of switches for flows (duration, B count)
    - average bandwidth over the total duration of the flow
- how accurate?
    - same out-of-date risk as port bandwidth
    - might miss some flows in case of big intervals

Switch Load

- what?
    - number of bytes processed by each switch
- why?
    - network upgrade planning, bottleneck finding
    - computing the list of busiest switches in a network
- how?
    - periodic task started at startup exposing data (REST)
    - aggregating bandwidth data of all the ports of the switch over an interval
- how accurate?
    - same out-of-date risk as port bandwidth

Device Activity

- what?
  - list of the hosts that are most active in the network
- why?
  - useful for network management by providing hints (weak points of the network, hosts with malicious actions)
- how?
  - analyzing incoming packets to find the host creating the flow + its port
  - querying the port for the total number of bytes
  - subtract the values measured last time

Protocol Classification

- what?
  - providing classification of the network traffic based on the higher-level protocols.
- why?
  - knowledge of the type of traffic in the network
  - performance improvements
    - tailoring the structure for more specific traffic.
- how?
  - parsing IP header for the protocol field, mapping the value against the list of L3+ protocols
- how Accurate?
  - very, but only analyzing 1 packet/flow on the controller

Conclusions

- measured metrics that are useful to network management for planning, scheduling, and load balancing
- link/port/flow bandwidth, switch load, device activity, protocol classification
- more focused on the control-plane metrics to be more accurate
- future work: based on the feedback from Floodlight community

Questions?