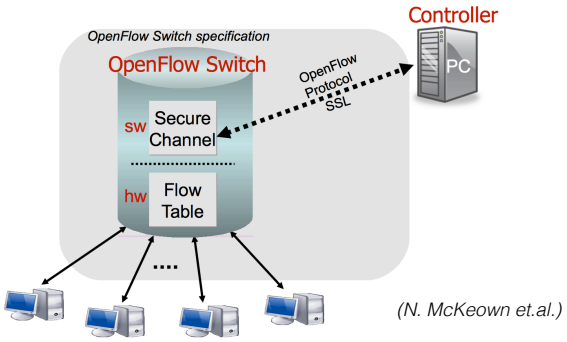
NOSIX

A Lightweight Portability API for the SDN OS

Andreas Wundsam (ICSI / UC Berkeley) · Minlan Yu (UC Berkeley / USC)

OpenFlow enables control plane programmability...



(N. McKeown et.al.)

... but programming SDNs with OF is hard

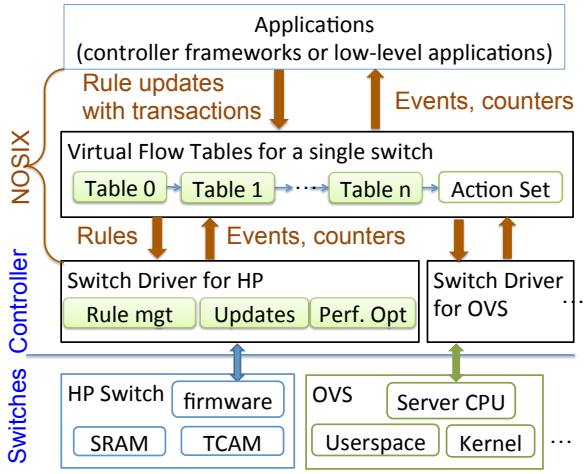
- Mismatch Between Application Expectations and Reality
- **Expectations**:
  - Homogeneous forwarding model
  - Sufficiently large flow tables
  - Predictable feature set and performance
  - Switch state known / deltas efficiently reconcilable
  - Support for fail-over
- **Reality**: Switch landscape is highly **heterogeneous**
  - Data Plane:
    - Hardware vs. software
    - Supported Matches + Actions
    - Table Sizes and Counts
  - Control Plane:
    - Rule updates
      - Supported consistency
      - Maximum churn
    - Counters
    - OpenFlow version
- **Reality** (II): **OpenFlow primitives** do not match expectations
  - With switch-side flow-expirations, flow table state is unknown
  - Barrier semantics switch dependent
  - No efficient reconciliation of changes after disconnect
- Tradeoff between **portability** and **performance**?
- **So far: Onix, POX, Frenetic, ...**
  - Manage the **entire** network
  - Provide a simplified network-wide programming model, controller distribution, consistent updates, composability,...
  - Optimize for a particular programming model
  - **All** have to be adapted for each individual switch [class]

    ➡ **Duplication of effort**

A Missing Piece in the Stack?

- A common API that provides the common primitives
  - **local -** controlling **one** switch
  - **ubiquitous** - services (almost) everyone needs
  - **expressive** - does not constrain the OpenFlow programming model
- Principles
  - Applications expose expectations to the switch
  - Vendors provide switch drivers in the controller

Architecture



Core Concepts

- **VFT:** Virtualized Flow Tables
  - Created by the Application
  - Full Feature Set
  - No resource constraints
  - Per table consistent updates
  - *Annotations* describe application expectations
- **Switch Drivers**
  - Map the Virtual Flow Table to the switch reality
  - Can virtualize resource constraints
    - E.g., rule paging to map 50k rules to 2k table entries
  - Optimize for switch specifics
    - E.g., knowledge of exact `BARRIER` Semantics

Benefits

- Portable Development of Low-Level-Controller Applications unsuitable for a high level SDN stack (e.g., Load-Balancing)
- Building Block for Higher Level Controller Platforms
- Decouples
  - Rule update semantics and mechanism at the switches
  - Application-specific and switch-specific performance optimizations
- Annotations
  - provide a lever to choose between portability and performance
- Enable protocol innovations by the vendors, e.g.,
  - built-in transactions for updates
  - efficient ruleset reconciliation after disconnect