A Security Enforcement Kernel for OpenFlow Networks
HotSDN 2012

Phillip Porras, **Vinod Yegneswaran**, Martin Fong, Mabry Tyson
(SRI International)

Seungwon Shin, Guofei Gu
(Texas A&M University)

Classic Network Perimeter Defense

- Security Policy Enforcement Methodology
  - Well-defined static security policy instantiated for a target topology
  - Deployed consistently across the network
  - Policy can only be altered by a small set of trusted elements
  - Policy modification events are audited and monitored for compliance

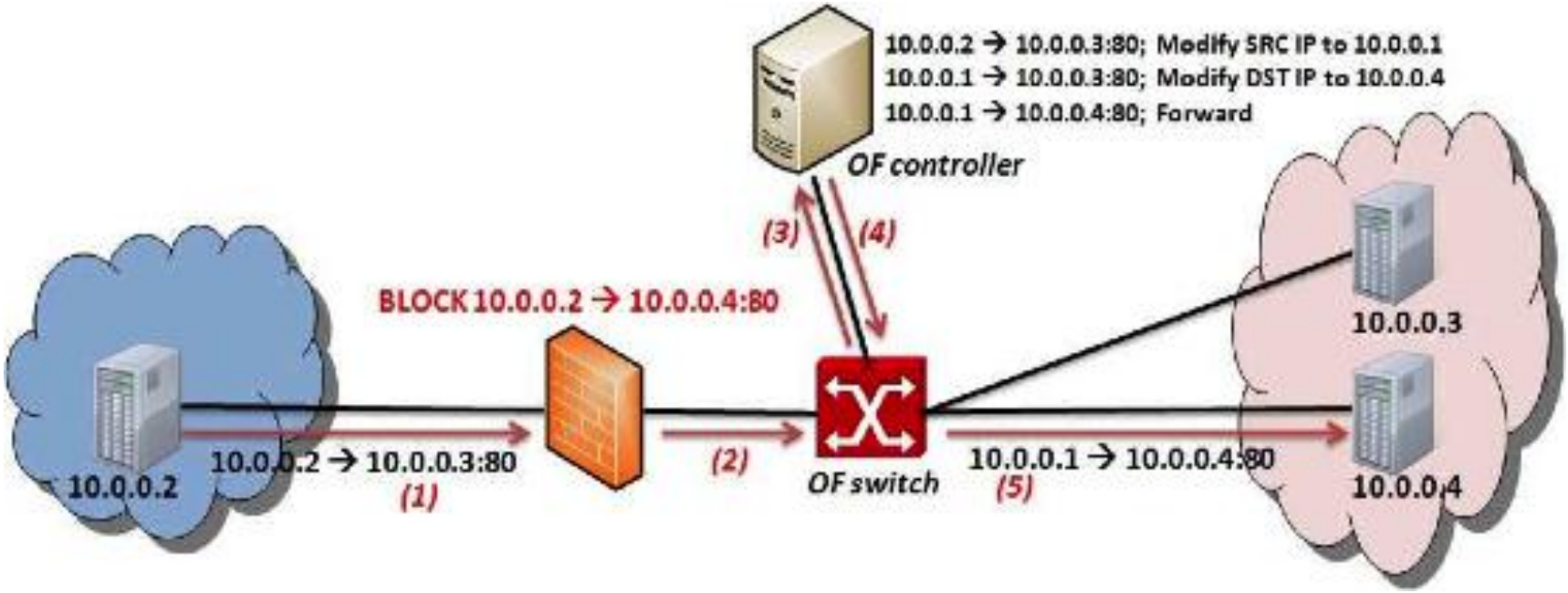
The OpenFlow [SDN] Network Model

- SDN / OpenFlow Network Model
  - Provides a set of continually and dynamically defined flow policies
  - Flow policies are embodied in the current set of *flow rules* instantiated into the switch
  - Flow rules are produced from OpenFlow applications that monitor and react to in and outbound packet flows
    - OF apps can compete, contradict, override one another, incorporate vulnerabilities
    - Worst case: an adversary can use the deterministic OF app to control the state of all OF switches in the network

OpenFlow Evasion Scenario

.

Dynamic Flow Tunneling



OpenFlow Security Policy Enforcement

- Dynamic control plane (policies) and data plane (flows) introduces new enforcement challenges
- OpenFlow could benefit from better mechanisms for
  - specifying and authenticating policies
  - dealing with rewrite rules
  - detecting and auditing policy violations

Research Objectives and Contributions

- Broad Objective
  - Provide mechanisms that support the development and integration of *traditional* and *new* security applications into Software-Defined Networks
- Specific Contributions
  - Development of a security enforcement kernel for the NOX OpenFlow controller
  - Role-based authorization
  - Rule conflict detection
  - Security directive translation

Motivating Security Applications

Tarpits: A Tarpit is an advanced anti-attack countermeasure designed to hold (reverse-DoS) inbound TCP connections from attackers

Reflector Nets (*): A security app that reprograms the OF network to forward an external entity into a remote honeynet

Phantom Nets: A technique in which a scanner is mislead into producing a false topology map for the network being scanned

Emergency Broadcast: When a switch-wide exceptional state is detected, this security app auto-inserts a high-priority forward rule for all connections originating from network operator owned addresses, while inserting drop filters to reject detected flooding sources/ports

White holes: A strategy for defeating sophisticated density-aware IP scanning techniques used by scan-and-infect malware to increase the rate at which viable infection targets are discovered

BotHunter: A method for diagnosing infections in internal network assets using dialog correlation to discover flow sequences that match coordination centric malware infections

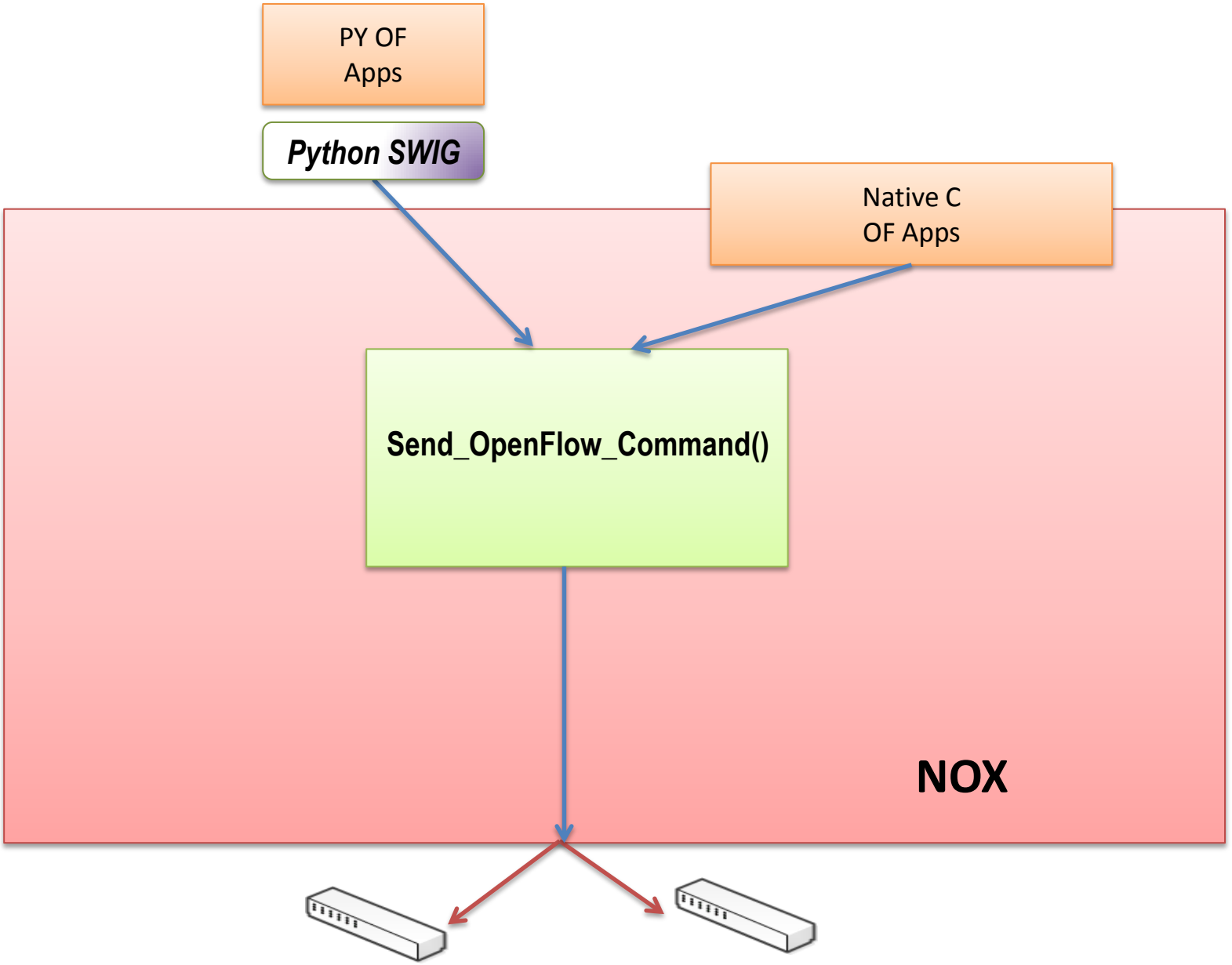
Many More: TRW (*), BotMiner (*), P2P Plotter (*)

Prerequisites for a Secure OpenFlow Platform

Must be resilient to

- Vulnerabilities in OF applications
- Malicious code in 3rd party OF apps
- Complex interaction that arise between OF app interactions
- State inconsistencies due to switch garbage collection or policy coordination across distributed switches
- Sophisticated OF applications that employ packet modification actions
- Adversaries who might directly target our security services to harm the network

Classic NOX Architecture



The FortNOX Security Enforcement Kernel

FortNOX:

A Non-bypassable mediation service that performs inline vetting of the OpenFlow Application flow rules against the current set of network flow constraints defined by administrators or OpenFlow Security applications

Least privilege mediation of flow insertions for policy consistency

- The FortNOX controller executes independently, in a separate process space (and ideally from a separate user account), from that of the OpenFlow applications it services
- NOX C libraries are wrapped using a Proxy App. They must not be run within the FortNOX process space
- All interactions between the controller and the switch must be mediated by the controller
- ~ 500 lines of C++ extension of the NOX source code

Authenticating Rule Producers

FortNOX implements source authentication through the use of digital signatures

- Rule producers export a public key, which administrators may choose to install into FortNOX, assigning this key to an authorization role

- FortNOX accepts FLOW_MOD commands with an extra digital signature
- Legacy OF application rules assigned default roles and lowest priorities

Role-Based Authorization

FortNOX extends the controller to recognize 3 standard authorization roles among flow rule producers

- **OF Operator Role** – define authoritative security policy
- **OF Security Role** - add flow constraints to combat live threat activity
- **OF Application Role** – legacy OF Apps, may remain security unaware

Authorization roles inform

- rule priority assignments
- conflict resolution when conflicts are detected

Rule Conflict Analysis

FortNOX incorporates a live rule conflict detection engine

- **Rule Conflict**: arises when a new candidate rule enables or disables a network flow that is otherwise inversely prohibited (or allowed) by existing rules
- ***Alias set rule reduction*** – a method detecting flow rule conflicts, even when OF set operations are used

Rule Conflict Analysis

Candidate Rules

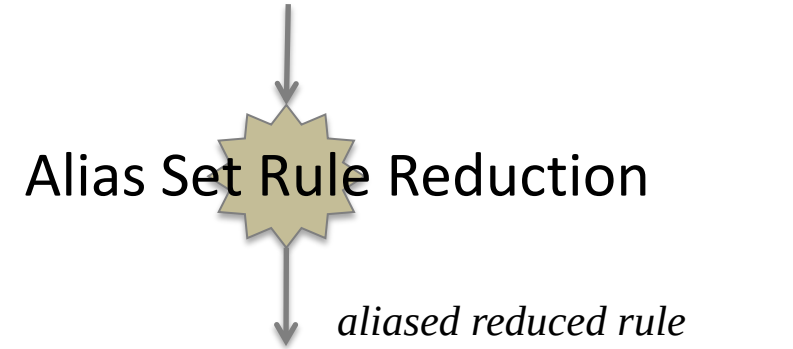
Match: a → b

Actions:

a ← a'

b ← c

forward



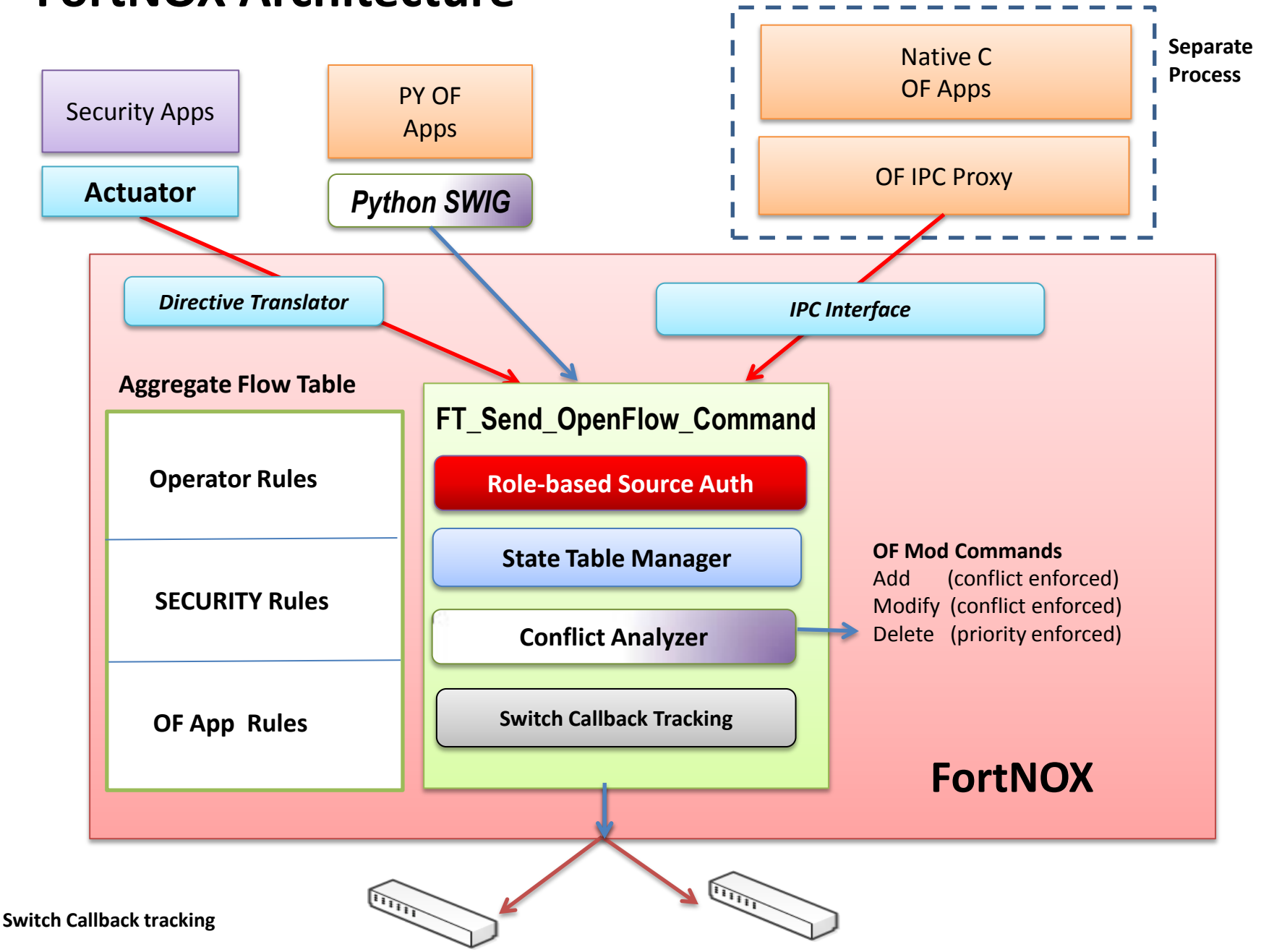
Conflict Resolution

- Derive ARRs per candidate rule
- Compare each ARR against FortNox's Aggregate Flow Table
- **IF** ARR intersects with registered rule
  Then flag candidate rule if ARR conflicts
  - Possible Resolution
    - Based on role-based priority
      - EQ - *policy*
      - GR - DEL, ADD
      - LT - REJECT

Security Directive Translation

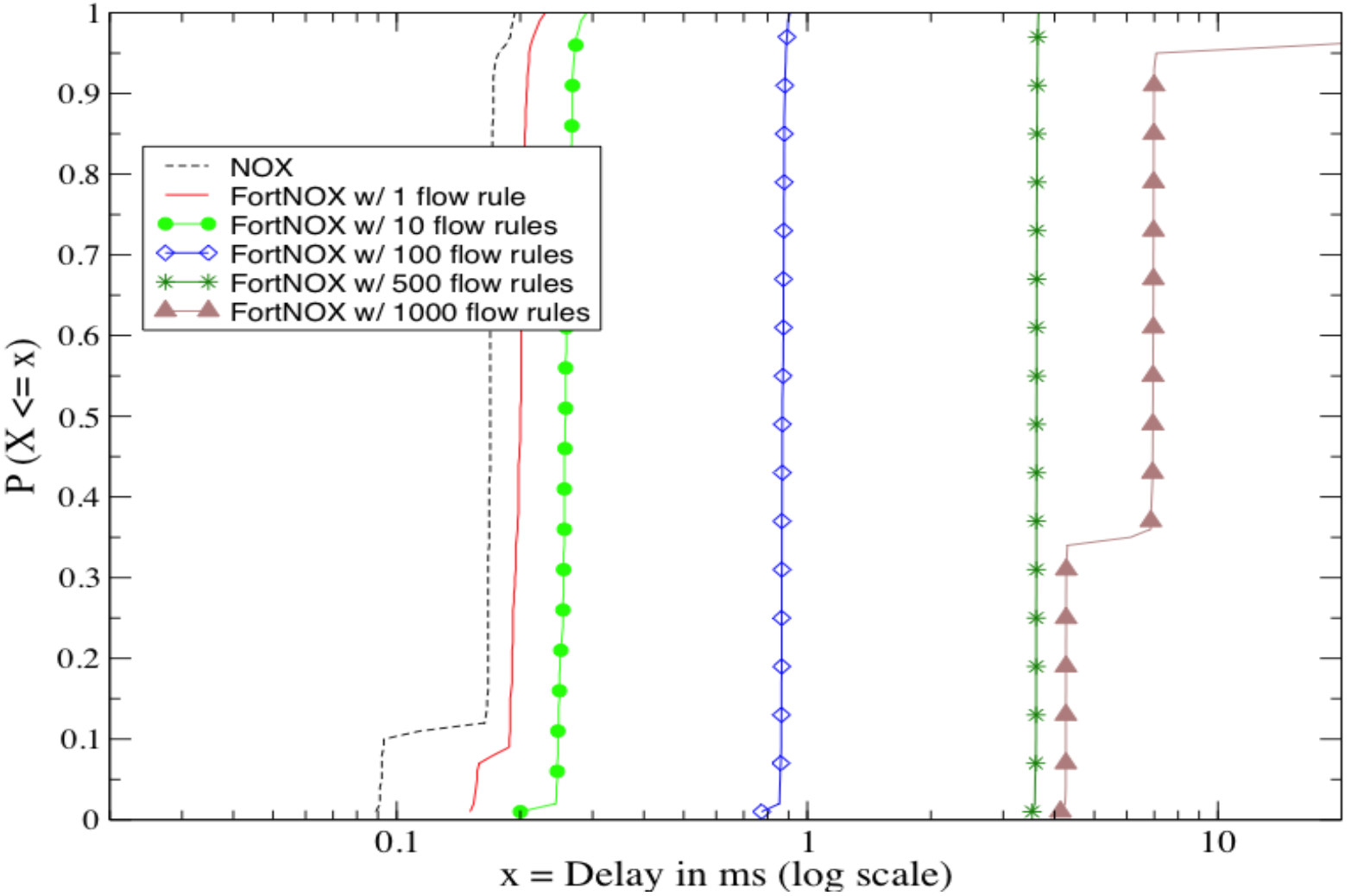
- Python interface for translating high level mitigation directives into flow rules
  - Seven new OF security directives currently implemented
  - block, deny, allow, redirect, quarantine, undo, constrain and info

FortNOX Architecture



Switch Callback tracking

Performance



Other Issues

- Distributed Policy Synchronization
  - FortNOX extends NOX to use barrier messages and switch callbacks to track flow rule removal
  - Distributed policy insertion must be atomically synchronized
  - Distributed policy removal must be atomically committed: harder

- Accountability: Audit accountability is a requirement for most sensitive computing environments. FortNOX produces a security audit trail for

  - all flow rule commands, with authenticated producer IDs
  - detected rule conflicts and resolution outcomes

Summary and Future Work

- FortNOX – A new security enforcement kernel for OF networks
  - Role-based Authorization
  - Rule-Authentication
  - Conflict Detection and Resolution
  - Security Directive Translation
- Ongoing Efforts and Future Work
  - Prototype implementations for newer controllers (Floodlight, POX)
  - Security enforcement in multicontroller environments
  - Improving error feedback to OF applications
  - Optimizing rule conflict detection
  - FRESCO: Modular language environment for composing OF security applications

Demonstrations

- **www.openflowsec.org**
  - Technical reports and publications
  - DEMO videos
    - Demo 1: **Constraints Enforcement** [high res .mov or Youtube!]
    - Demo 2: **Reflector Nets** [high res .mov or Youtube!]
    - Demo 3: **Automated Quarantine** [high res .mov or Youtube!]
  - FortNOX beta, single switch (multi-switch will follow)
- Acknowledgements:
  - Army Research Office