ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Δίκτυα Καθοριζόμενα από Λογισμικό

Ενότητα 2.3: Security Policy Enforcement in SDNs

Ξενοφώντας Δημητρόπουλος
Τμήμα Επιστήμης Υπολογιστών

Security Policy Enforcement in SDNs

Xenofontas Dimitropoulos

3/11/2014

Credits: Seungwon Shin, Phil Porras, Vinod Yegneswaran

SDN Security Aspects

- Controller is single point of failure
  - DoS attacks?
  - Compromised controller?
- Enforce isolation between virtual networks
- Weaknesses in OpenFlow protocol
  - E.g. sneak in unauthentic rules
- New opportunities for security policy enforcement

Agenda

- FortNOX Security Enforcement Kernel

- FRESCO Security Application Development Environment

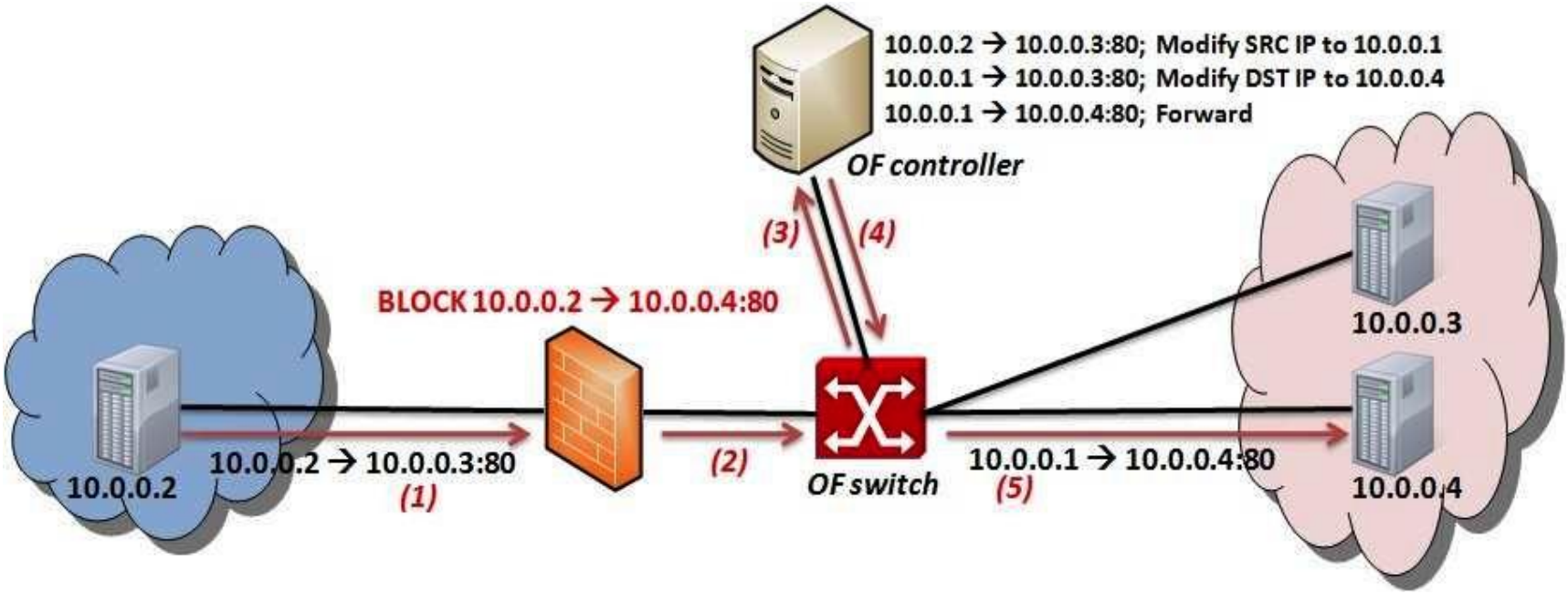
Classic Network Perimeter Defense

- 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

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

FortNOX 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

Phillip Porras, Vinod Yegneswaran, Martin Fong, Mabry Tyson, Seungwon Shin, Guofei Gu
"A Security Enforcement Kernel for OpenFlow Networks" HotSDN 2012

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

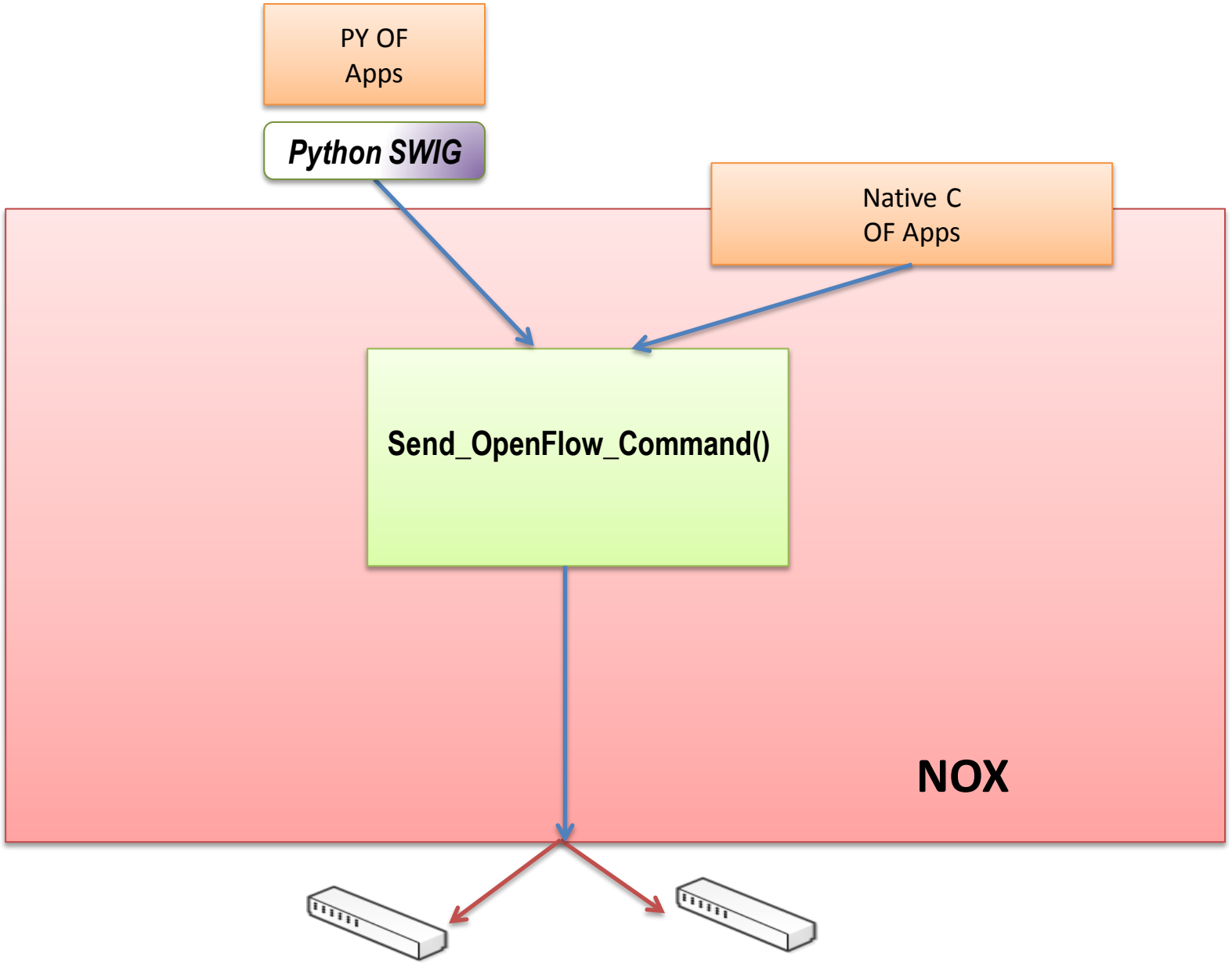
Many More: TRW (*), BotMiner (*), P2P Plotter (*)

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

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

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

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

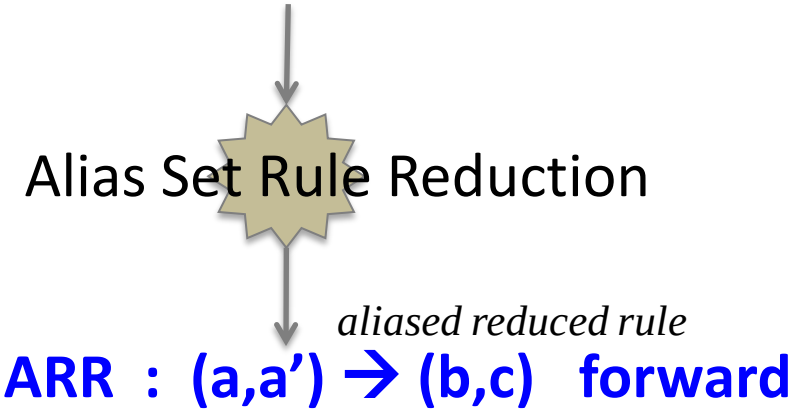
Match: a → b

Actions:

a ← a'

b ← c

forward



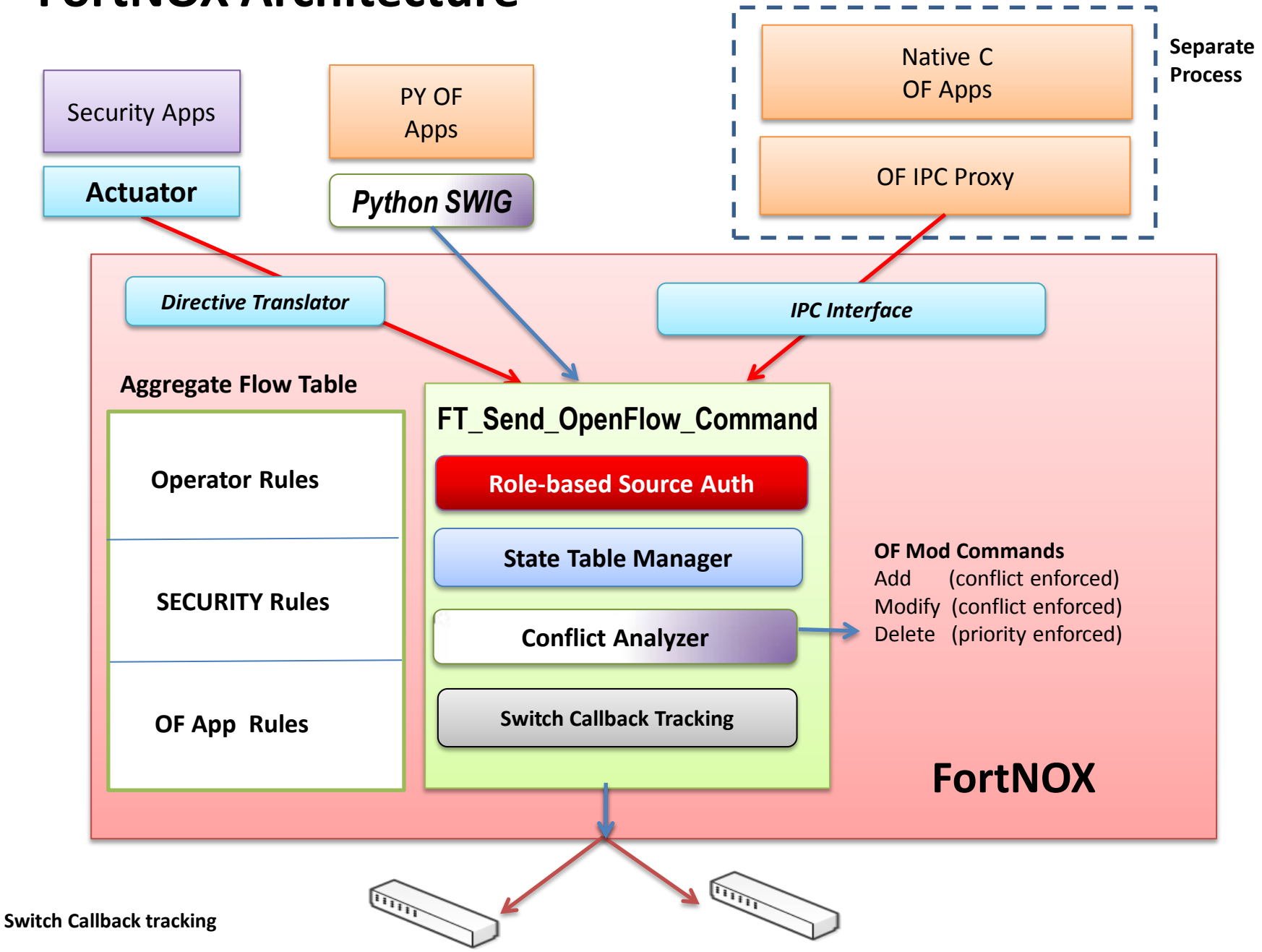
Conflict Resolution

- 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

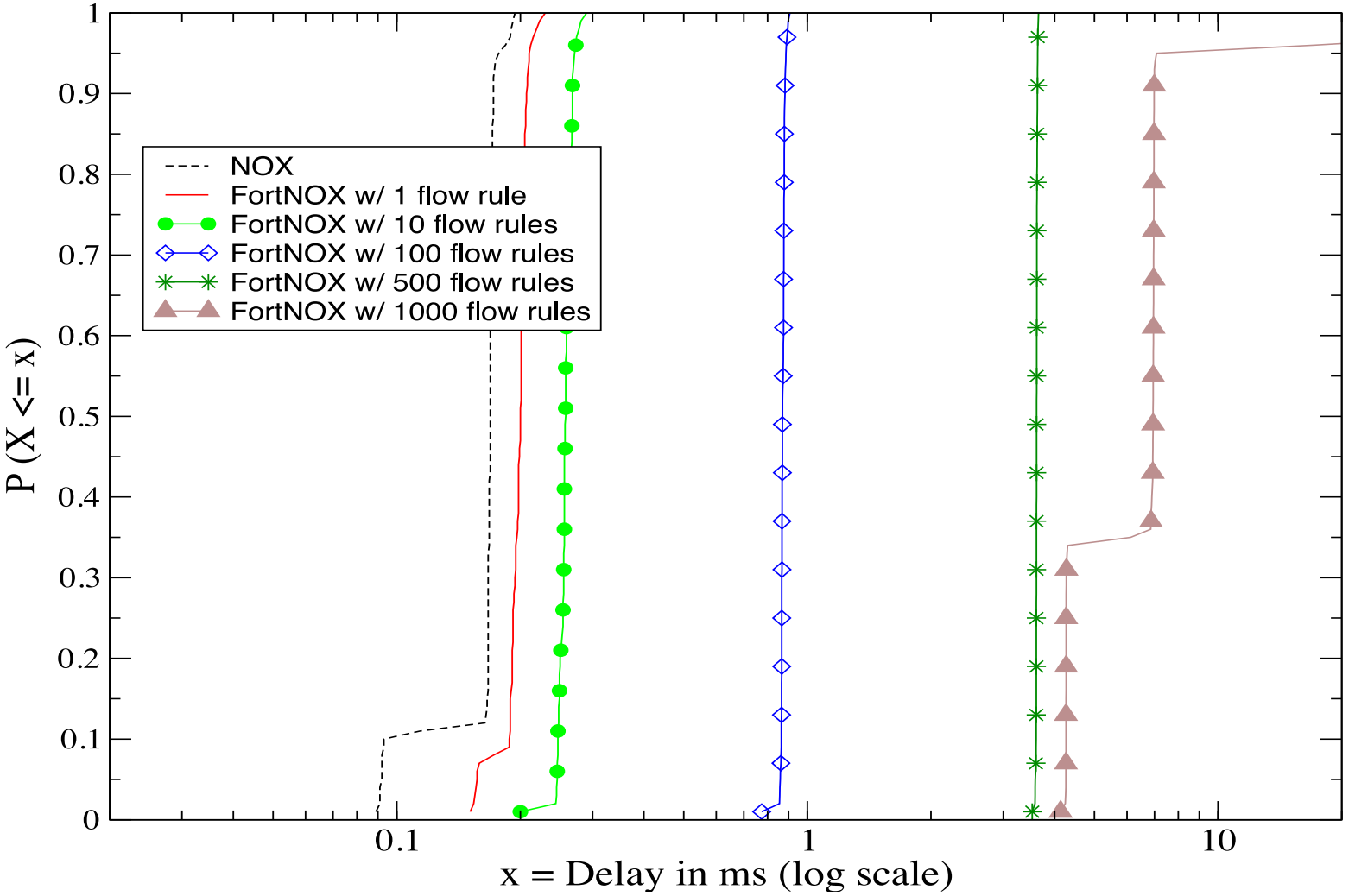
- Python interface for translating high level mitigation directives into flow rules
  - Seven new OF security directives currently implemented
  - block, deny, allow, redirect, quarantine, undo, constrain and info

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

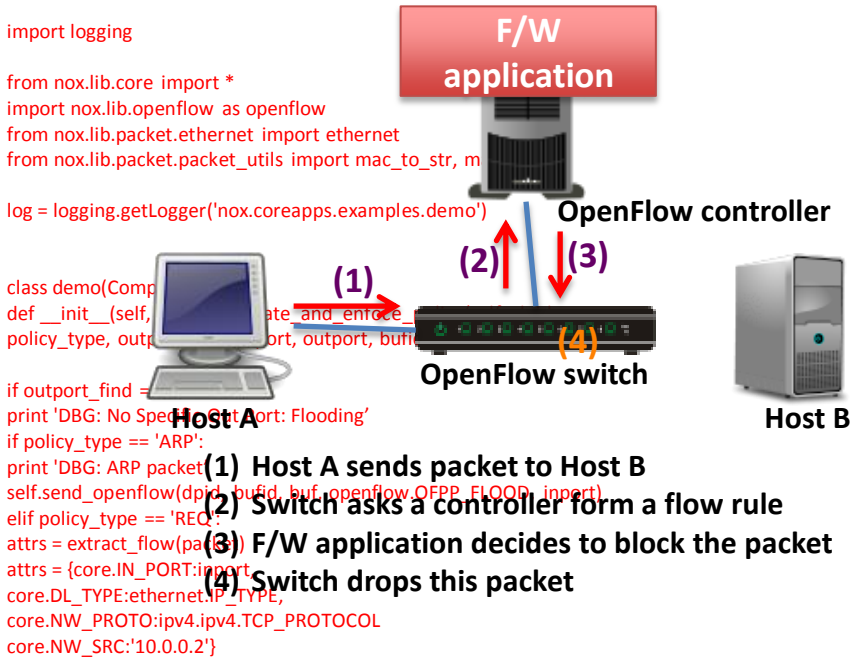
Agenda

- FortNOX Security Enforcement Kernel

- FRESCO Security Application Development environment

Security Functions with SDN

- Security functions can be applications of SDN
  - Firewall
  - DDoS detection
  - Scan detection
  - and more...



Security Functions with SDN

- However, it is not easy to create security applications in SDN

- Why?
  - lack of convenience
    - need to understand many low level things
  - lack of information
    - E.g., TCP session, network status

- **How to address these issues?**

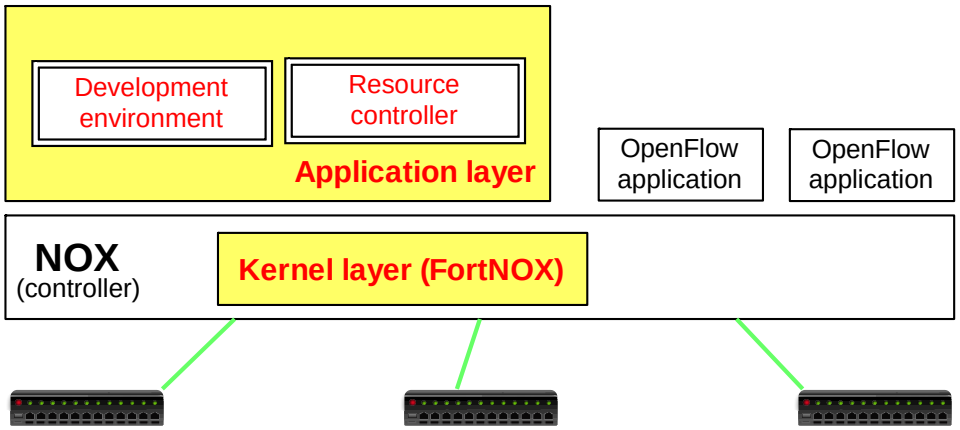
FRESCO

- It is a framework to
  - Provide development environment for security applications
  - Manage resources for security applications
  - Deploy security policies
- With this framework, we can
  - Create and compose our own network security functions easily
  - Deploy network security functions easily and dynamically
- **Finally, FRESCO can help security people focus on devising security applications**

Seungwon Shin, Phil Porras, Vinod Yegneswaran, Martin Fong, Guofei Gu, and Mabry Tyson
"FRESCO: Modular Composable Security Services for Software-Defined Networks" NDSS 2013

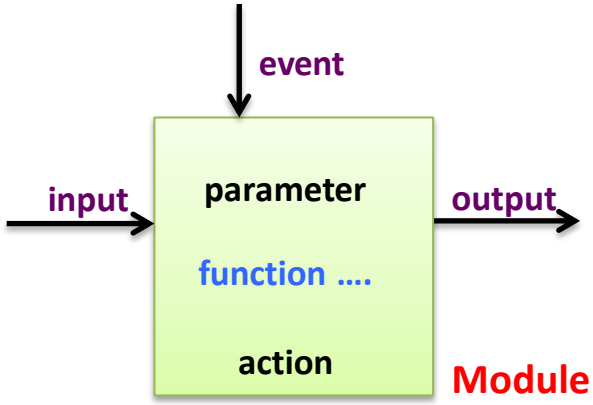
Architecture

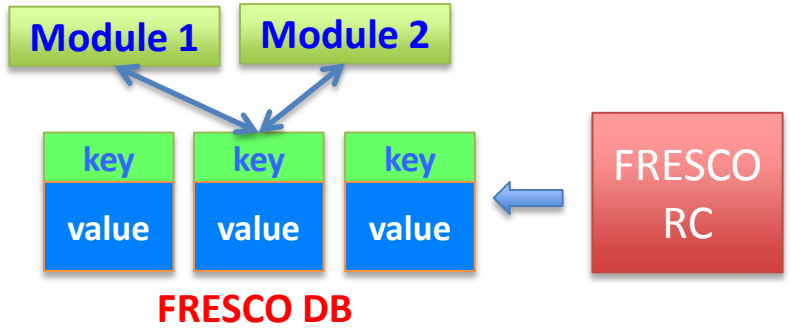
- Component
  - **Application layer**
    - **Development env. (DE)**
    - **Resource controller (RC)**
  - Kernel layer
    - Security enforcement kernel
    - FortNOX
      - paper in HotSDN 2012



Development Environment

- FRESCO Module
  - Basic operation unit
- FRESCO DB
  - Simple database
    - (key,value) pairs
- FRESCO script
  - Define interfaces
  - Connect multiple modules





Development Environment: Fresco Script

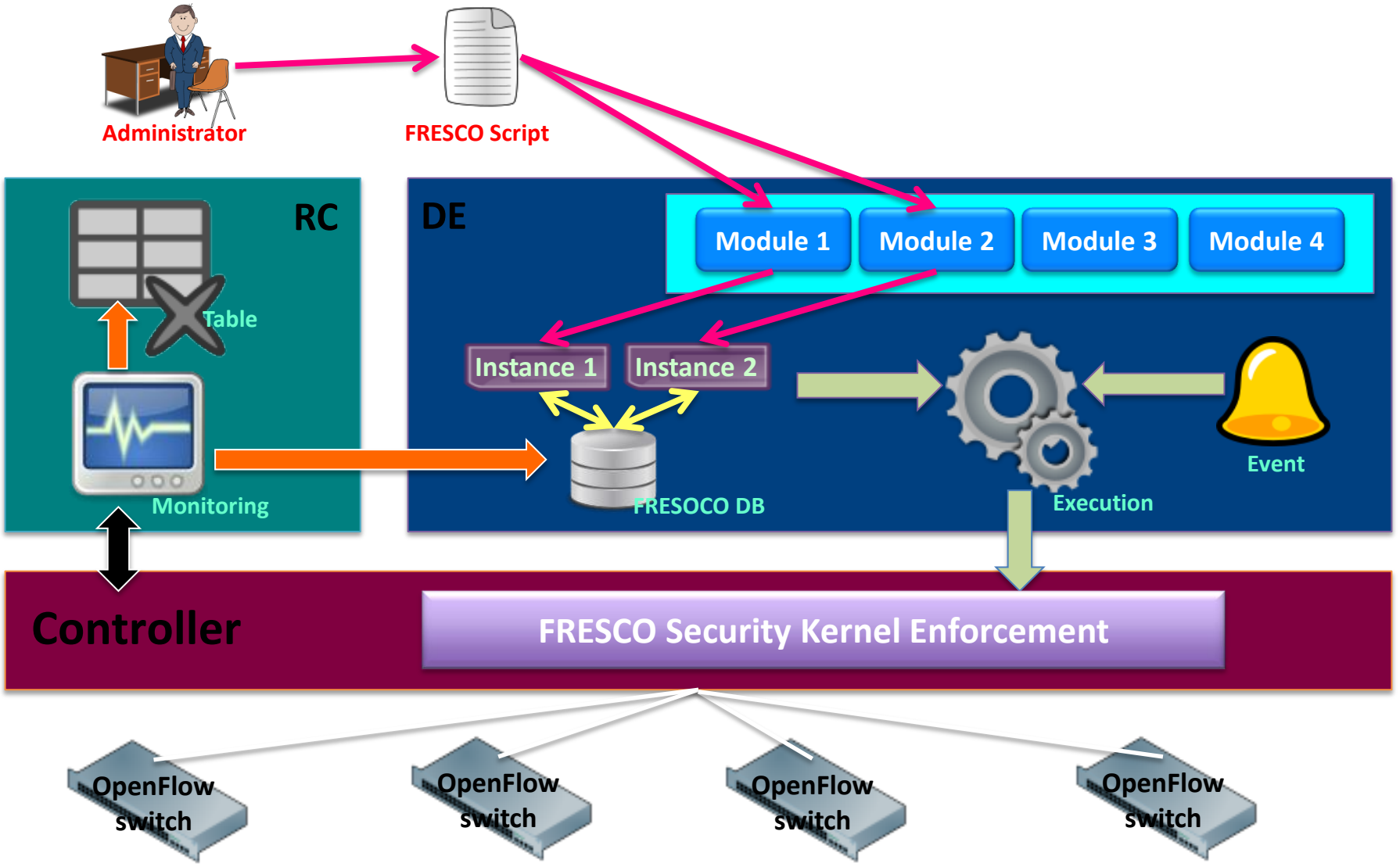
```
instance_name (#inputs) (#outputs) {
  type: name of existing module of which this is an instance
  event: triggering event
  input: name of input item, name of input item, ...
  output: name of output item, name of output item, ...
  parameter: value
  action: action to be performed
}
```

Fresco Script: Drop HTTP traffic

```
port_comparator (1) (1) {
  type: Comparator
  event: INCOMING_FLOW
  input: destination_port
  output: comparison_result
  parameter: 80
  action: -
}

do_action (1) (0) {
  type: ActionHandler
  event: PUSH
  input: comparison_result
  output: -
  parameter: -
  action: comparison_result ? DROP : FORWARD
}
```

Operational Scenario



Implementation

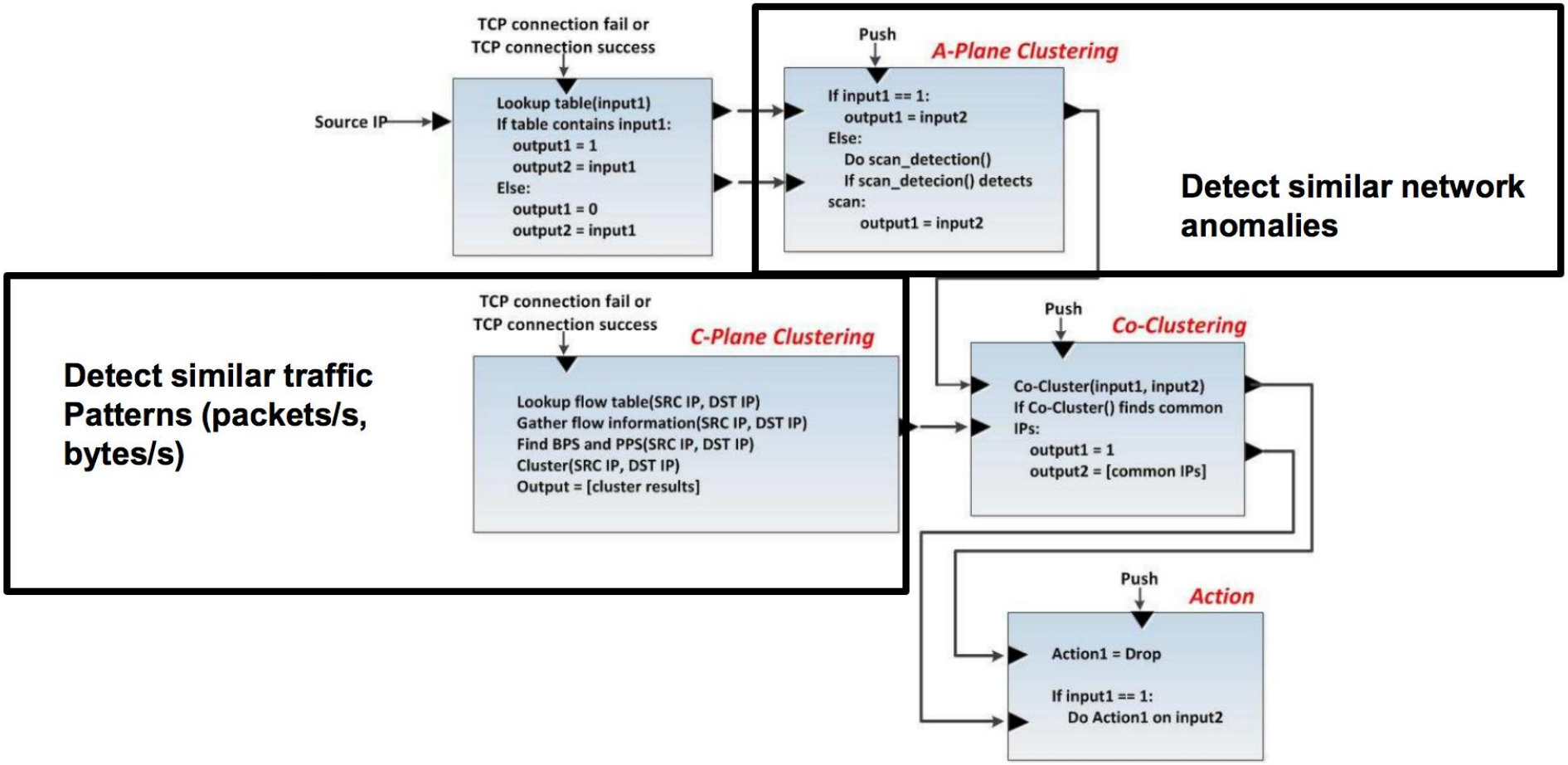
- NOX (open source OpenFlow controller) based
  - Development environment
    - NOX based Python application
  - Resource controller
    - NOX based Python application
  - Security enforcement kernel
    - Modify NOX (C++)

Example: Reflector Net

```
find_scan (1) (2) {
  type: ScanDetector
  event: TCP_CONNECTION_FAIL
  input: source_IP
  output: source_IP, scan_result
  parameter: 5 /* Five failed connection attempts */
  action: -
}

do_redirect (2) (0) {
  type: ActionHandler
  event: PUSH
  input: source_IP, scan_result
  output: -
  parameter: -
  action: scan_result ? REDIRECT : FORWARD
}
```

Bot Minner


TCP connection fail or
TCP connection success
Source IP
Lookup table(input1)
If table contains input1:
output1 = 1
output2 = input1
Else:
output1 = 0
output2 = input1
Push
A-Plane Clustering
If input1 == 1:
output1 = input2
Else:
Do scan_detection()
If scan_detecion() detects
scan:
output1 = input2
Detect similar network anomalies
Detect similar traffic Patterns (packets/s, bytes/s)
TCP connection fail or
TCP connection success
C-Plane Clustering
Lookup flow table(SRC IP, DST IP)
Gather flow information(SRC IP, DST IP)
Find BPS and PPS(SRC IP, DST IP)
Cluster(SRC IP, DST IP)
Output = [cluster results]
Push
Co-Clustering
Co-Cluster(input1, input2)
If Co-Cluster() finds common
IPs:
output1 = 1
output2 = [common IPs]
Push
Action
Action1 = Drop
If input1 == 1:
Do Action1 on input2

Evaluation

Source code length comparison

Algorithm	Implementation		
	Standard	OpenFlow	FRESCO
TRW-CB	1,060	741	66
Rate Limit	991	814	69

Results for Standard and OpenFlow are obtained in the following paper,
S. A. Mehdi, J. Khalid, and S. A. Khayam.
Revisiting Traffic Anomaly Detection Using Software Defined Networking, In Proceedings of Recent Advances in Intrusion Detection, 2011.

Flow rule setup time

	NOX	Simple Flow Tracker	Simple Scan Detector	Threshold Scan Detector	BotMiner	P2P Plotter Detector
Time (ms)	0.823	1.374	2.461	7.196	15.461	11.775

Please refer to our paper for the explanation of each test case

Summary and Future Work

- FRESCO
  - Create security applications easily
  - Deploy security applications easily
  - Focus on creating security applications

- Future work
  - Port FRESCO to other controllers for open source release
    - E.g., POX or Floodlight
  - Create more modules (now 16 basic modules)

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)

Τέλος Ενότητας

BY NC ND

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση, Όχι Παράγωγο Έργο 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

- Ως **Μη Εμπορική** ορίζεται η χρήση:
  - που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
  - που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
  - που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

- Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Κρήτης, Ξενοφώντας Δημητρόπουλος. «**Δίκτυα Καθοριζόμενα από Λογισμικό. Ενότητα 2.3: Security Policy Enforcement in SDNs**». Έκδοση: 1.0. Ηράκλειο/Ρέθυμνο 2015. Διαθέσιμο από τη δικτυακή διεύθυνση: http://www.csd.uoc.gr/~hy436/