Active monitoring framework for Software-Defined Networks

Candidate: Hanieh Rajabi

Tutor: Prof. Giuseppe Bianchi

University of Rome Tor Vergata

University of Rome "Tor Vergata"

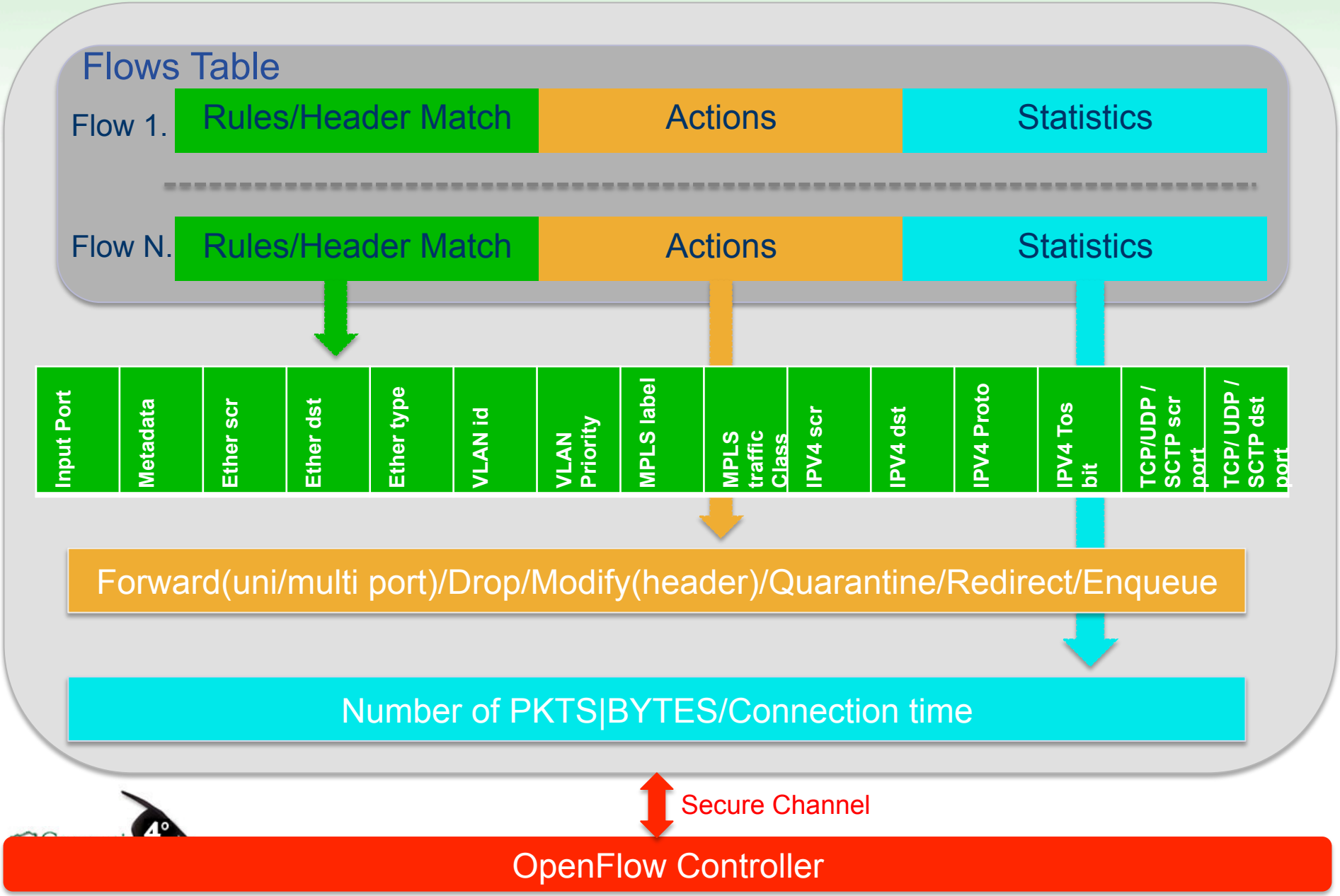
4° Borsisti Day *– 13/09/2013*

GARR

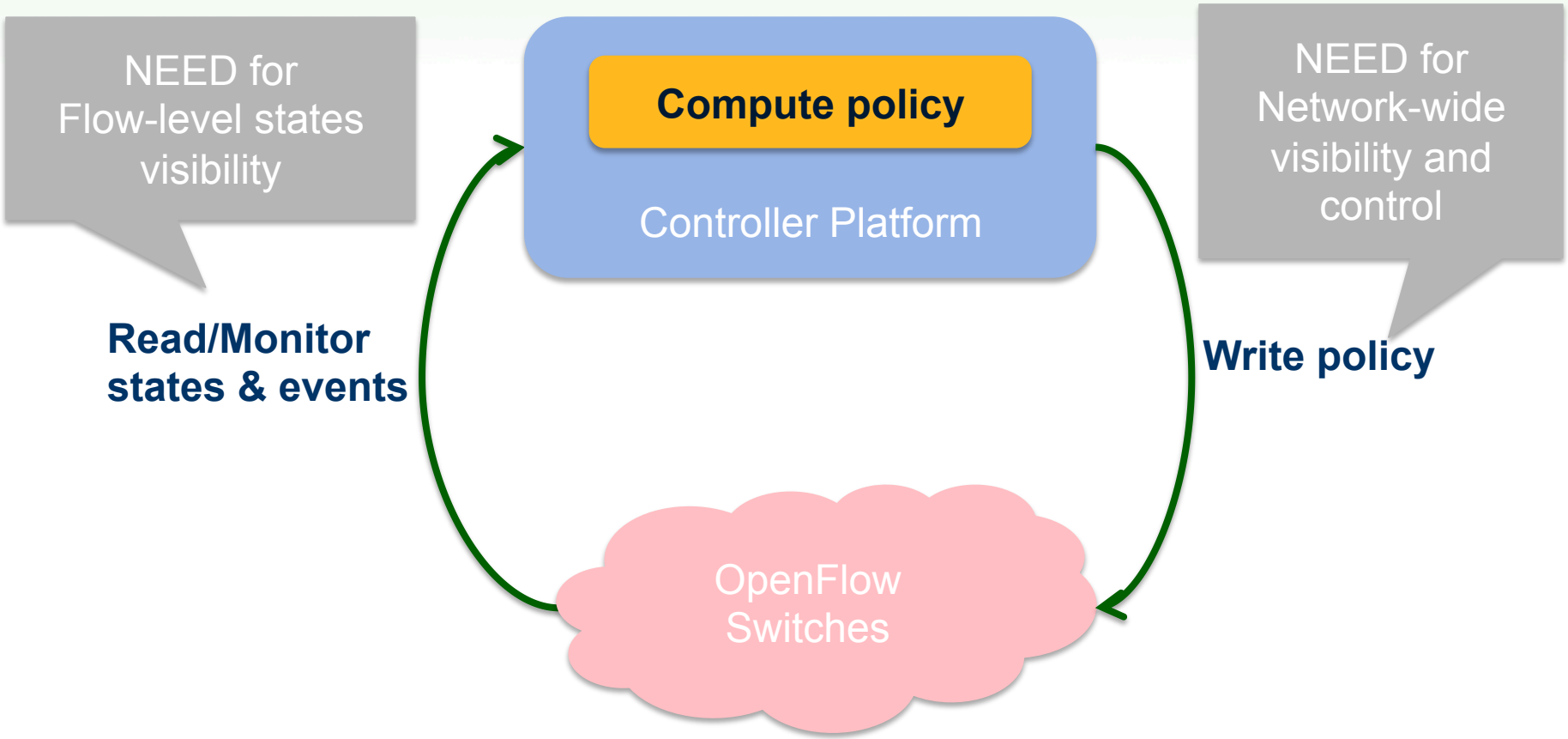
SDN: General overview

- Goal: Simplify networking and enable new applications
- Separation of control plane and data plane
  - Decouple the system that makes decisions about where traffic is sent (the control plane) from the underlying system that forwards traffic to the selected destination (the data plane)
  - Enable flow-based network programmability from controllers
- Features:
  - Increase network reliability, flexibility and Possibly Security
  - Automated management
  - Uniform policy enforcement
- Network operators and administrators can programmatically configure through network abstraction

OpenFlow Protocol Architecture



SDN Monitoring: 3 steps



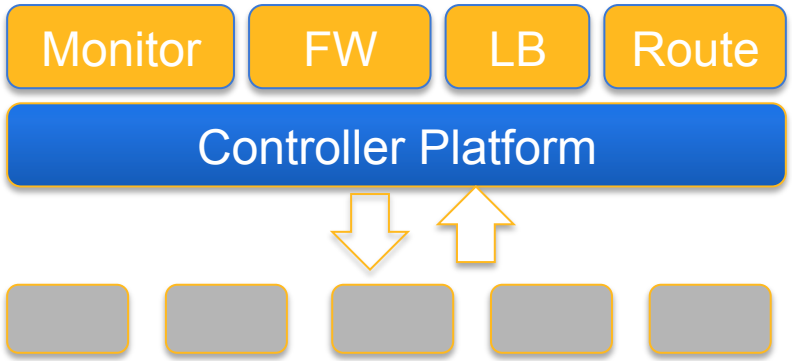
Monitoring tasks challenges

- OpenFlow is a low level of abstraction
  - Event handling: which event for which flow
  - Sophisticated event
  - Controller visibility
    - sees events that the switches do not know how to handle
- Matching rules
  - List of rules could be OK for DPI (e.g. SNORT), but not sufficiently expressive for a general purpose monitoring.
  - Limit number of rules (limited space not enough for all possible patterns)

Monitoring tasks challenges

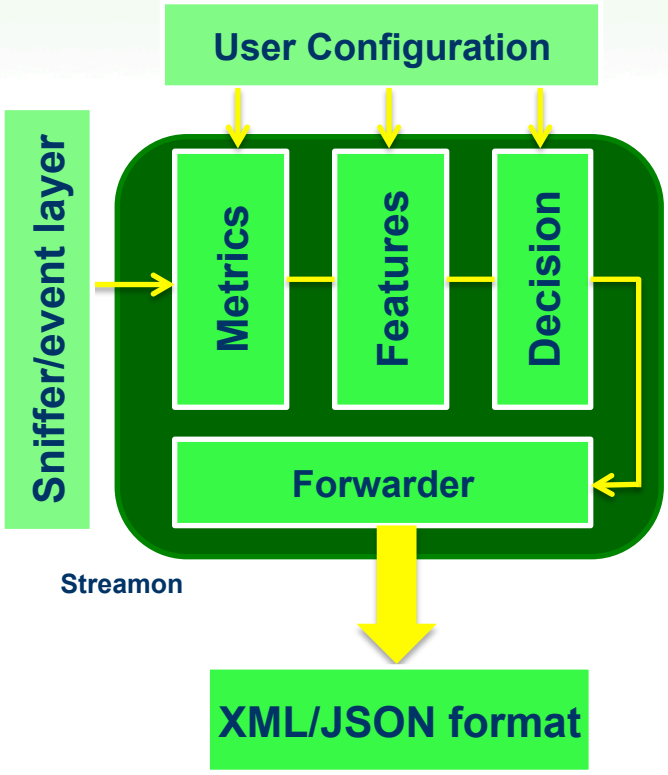
- Feedback loop from actions
  - state handling is needed to track attack evolution

- Policy composition
  - Modularize controller

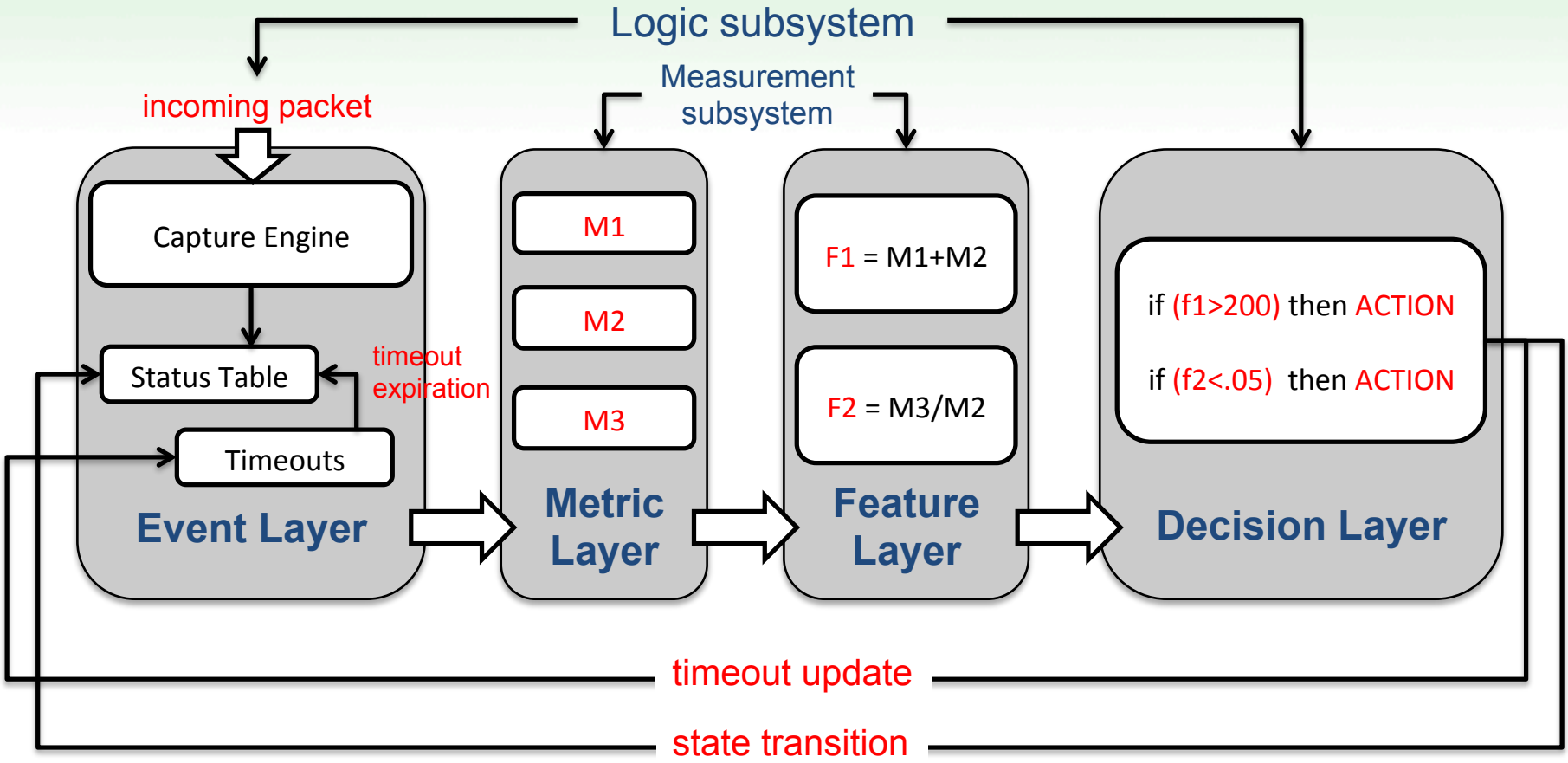


Monitoring Application

- **a software-defined stream-based monitoring API**
- Programmable monitoring probes
  - exploits eXtended Finite State Machines (XFSM)
  - Using compact data structure(bloom filter and counter bloom filter)
  - simple high level Application Programming Interface
    - Users can easily configure the application with their own metrics/features and the decision entries(high-level XML- like language).
  - Configurable output.



Monitoring Architecture



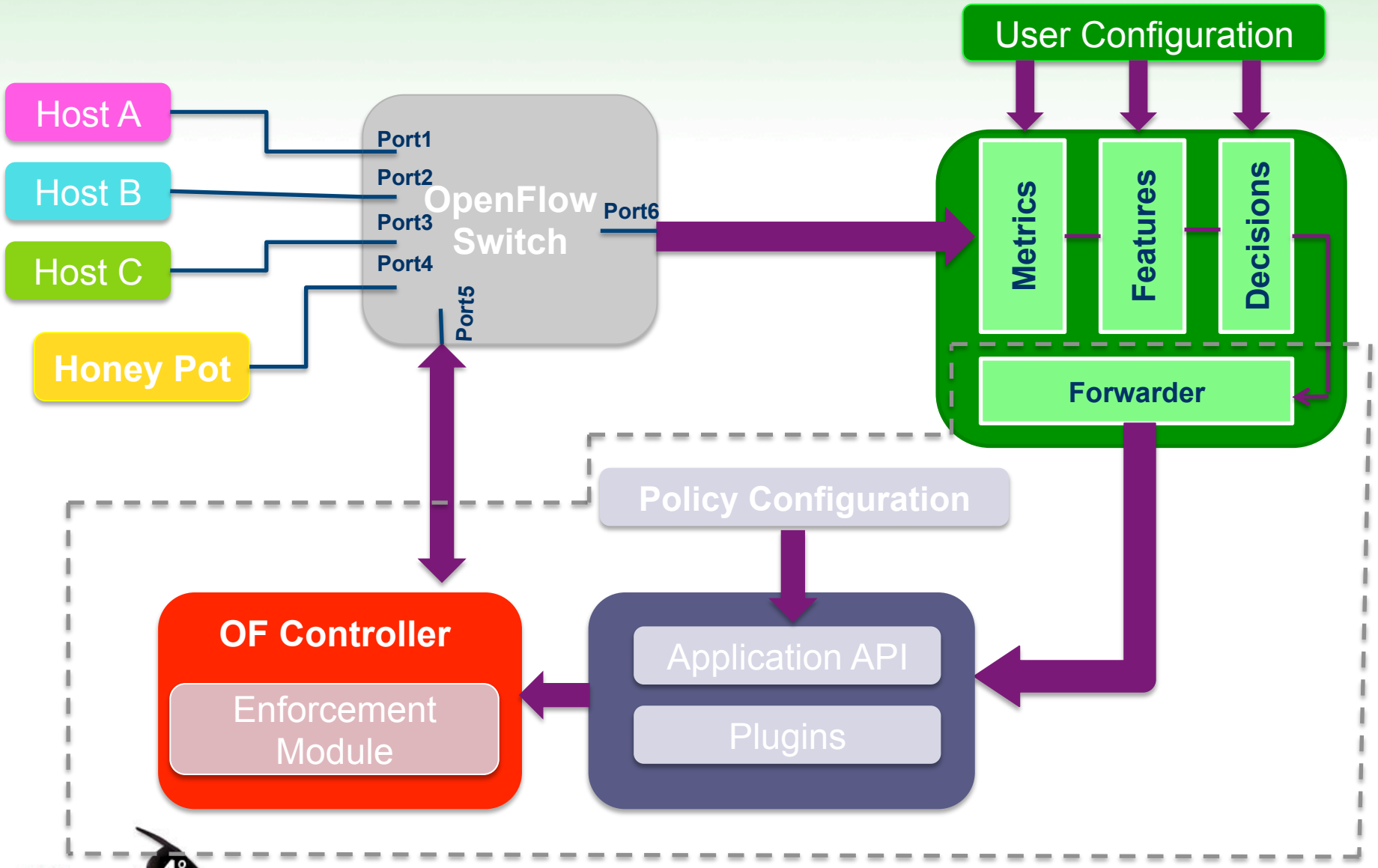
What to describe in configuration

- **EVENTS**
  - Triggered by packet arrival:
    - matching rule (e.g. DDoS: TCP protocol with SYN flag)
    - extract flow key (e.g. ip.dst)
  - Set by internal timeouts
- **METRICS & FEATURES**
  - Scalable hard-coded metrics (CBF)
    - M1= number of TCP SYN addressed to a same target in 60 seconds.
    - M2=TCP Syn rate over specific time window (rate of each host contacts each destination IP addresses;(ipsrc|ipdst)
  - Flexible features (arbitrary operations of metric outputs)
    - What to set (or export) when an event/ transition occurs
    - F1=M1; *if (F1 < threshold) then (state=default) else (state=monitor && set timeout)*
    - Timeout expires; *if (F1(t) > 1.2 * F1(t-1)) then (state=attack)*

What to describe in configuration

- **XFSM (eXtended Finite State Machines)**
  - TRACKING STATES: user-defined (if needed)
  - STATE TRANSITIONS
    - Which events cause transition, AND under which measurement conditions
  - associated ACTIONS (DROP,FORWARD,MARK,UPDATE/SET TIMEOUT)
  - E.g. filter traffic at attack state (i.e. using M2, the user starting activities before the DDoS attack is not filtered)

High Level System Architecture



Logic of Procedure

- Traffic monitoring application sniff the network traffic.
- Traffic Feature extraction for every flow
  - User based configuration
- Exported message from Monitoring
  - Base on the Type of Event allows to configure the output message.
    - E.g. TCP syn DDOS, report (scr.addr & dst.addr), attack type, status
- Policy User Configuration
  - Defines Rules/Policy for different type of Event.
- Develop module for mitigation response.
  - Policy enforcement in controller
- Flow Rules Installation in Switch

Work Flow

- Define interfaces between monitoring and OF controller
- Setup OF network (real or emulated)
- Define some policies for different type of threats
- Develop the monitoring-interfaced OF controller application
- Tests and measurements

Thank you

Q & A