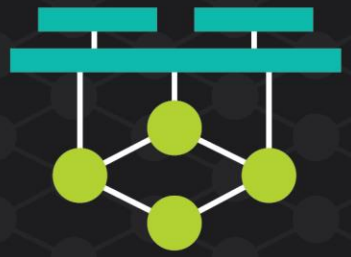


ons
EUROPE
OPEN NETWORKING //
Enabling Collaborative
Development & Innovation

SDN in the world of AI / ML

Abhijit Kumbhare
OpenDaylight Technical Steering Committee (TSC) Chair
Ericsson

Hosted By
THE LINUX FOUNDATION | LF NETWORKING

Agenda

- What is SDN?
- What is AI / ML?
- Role of AI/ML in SDN
- Acknowledgements

What is SDN?

Introduction: Why SDN?

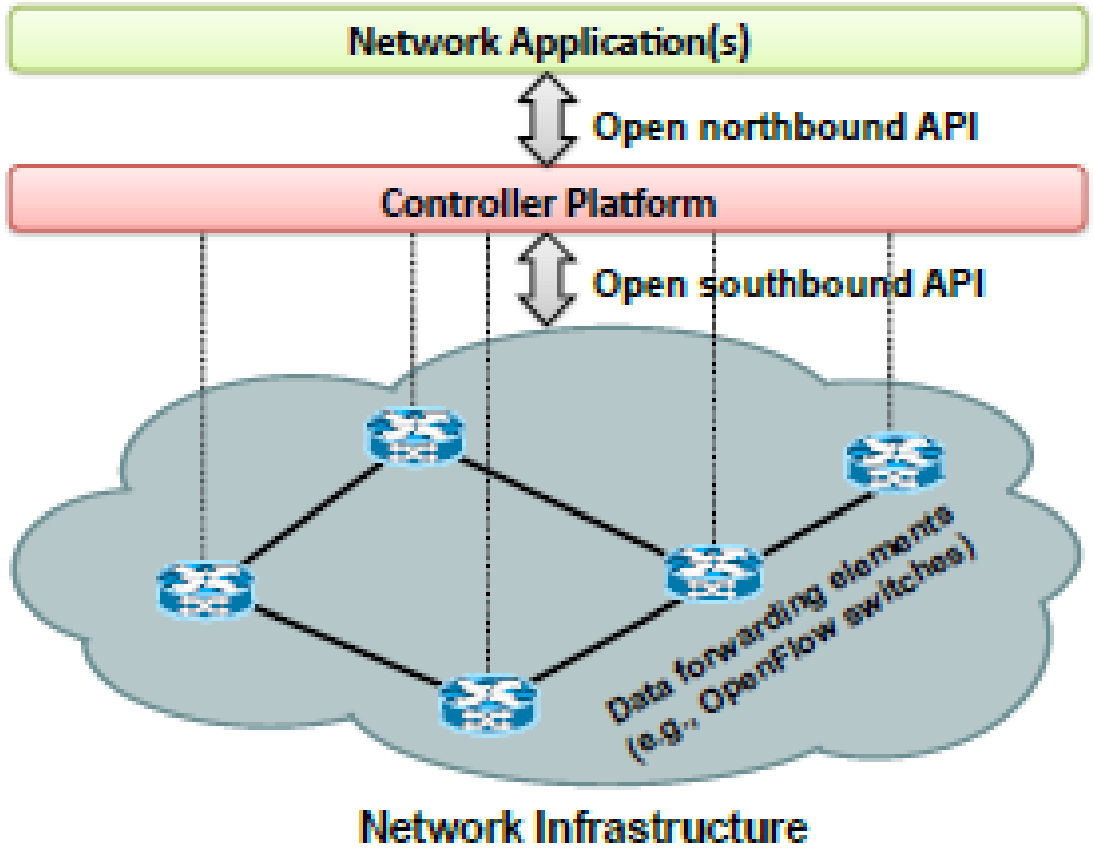
- Traditional IP networks are *Complex and hard to manage*
- Network operator need to configure each individual network device separately using low-level and often vendor-specific commands
- Networks are also *vertically integrated .*
  - the control plane and the data plane are bundled inside the networking devices. Reducing flexibility and hindering innovation and evolution of networking infrastructure.
    - Example: the transition from IPV4 to IPV6 started more than a decade ago and still largely incomplete.
    - A new routing protocol can take 5 to 10 years to be fully designed, evaluated and deployed .

Introduction: What is SDN? (Initial View - Evolving)

- Software-Defined Networking (SDN) is an emerging networking paradigm that gives hope to change the limitation of current network infrastructures.
  - First, it breaks the vertical integration by separating the network's control logic (the control plane) from the underlying routers and switches that forward the traffic (the data plane).
  - Second, with the separation of the control and data planes, network switches become simple forwarding devices and the control logic is implemented in a logically centralized controller (or network operating system), simplifying policy enforcement and network (re)configuration and evolution

Simplified view of an SDN architecture

Figure 1



Southbound API / Protocol

- The separation of the control plane and the data plane can be realized by means of a well-defined programming interface between the SDN controller and the switches. The controller exercises direct control over the state in the data plane elements via this well-defined API, as depicted in Figure 1.

- The most notable example of such an API is OpenFlow. An OpenFlow switch has one or more tables of packet-handling rules (flow table).

- Each rule matches a subset of the traffic and performs certain actions (dropping, forwarding, modifying, etc.) on the traffic. Depending on the rules installed by a controller application, an OpenFlow switch can – instructed by the controller – behave like a router, switch, firewall, or perform other roles (e.g., load balancer, traffic shaper, and in general those of a middlebox).

Separation of Concerns

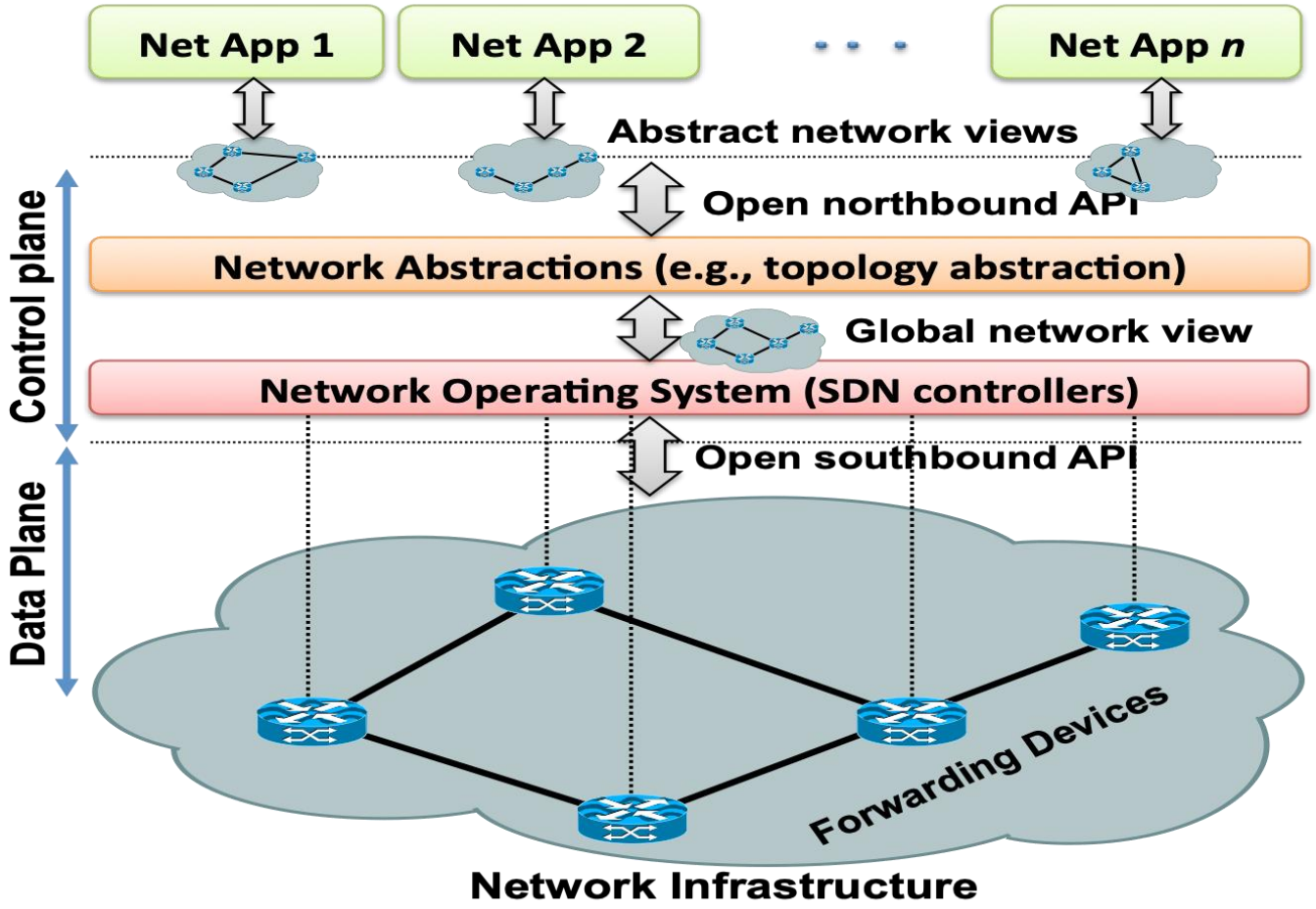
- An important consequence of the software-defined networking principles is the separation of concerns introduced between the definition of network policies, their implementation in switching hardware, and the forwarding of traffic.

- This separation is key to the desired flexibility, breaking the network control problem into tractable pieces, and making it easier to create and introduce new abstractions in networking, simplifying network management and facilitating network evolution and innovation.

Evolving View: Centralized vs Distributed

- Some people think of SDN as a programmatic control of the network as opposed to control/data plane separation
- In this view, a logically centralized programmatic model does not postulate a physically centralized system.
- In fact, many production-level SDN network designs resort to physically distributed control planes.

SDN Architecture and its Fundamental Abstractions



Role of AI/ML in SDN

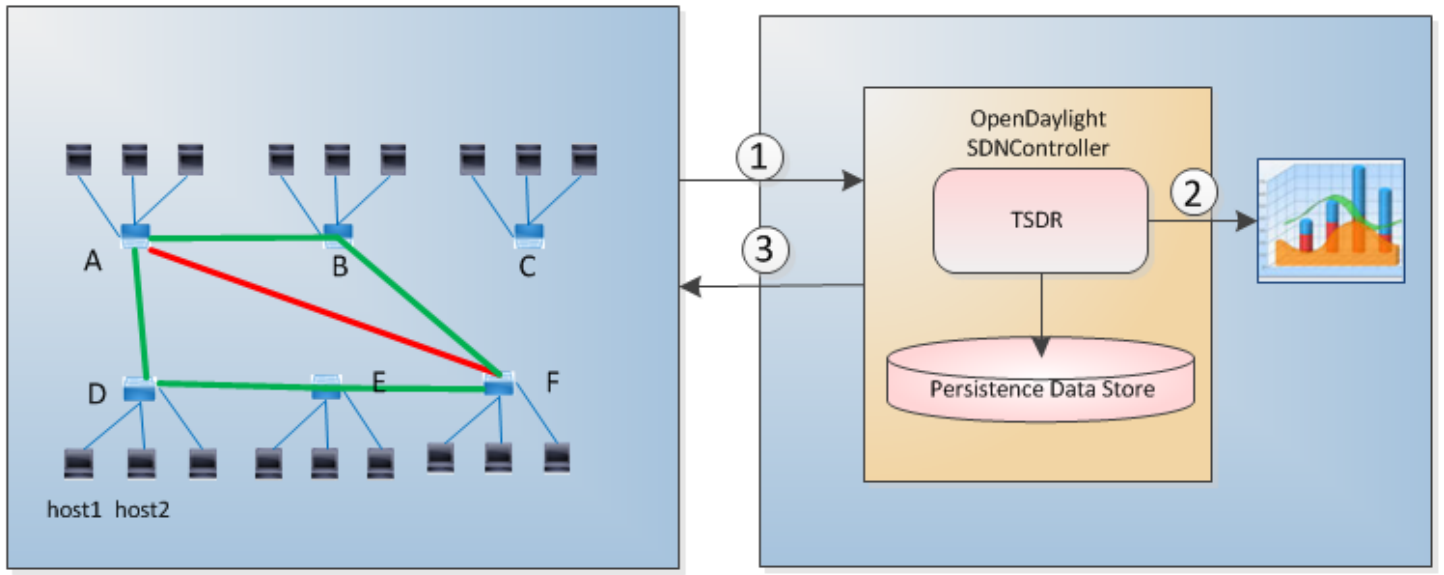
Why we need Machine Learning in SDN

- Software Defined Network needs to be intelligent.
  - To be aware of the runtime status of the network.
  - To make the right decisions to adjust the policies for traffic classification and traffic shaping.
    - Traffic Classification – Categorizes network traffic by packet or flow attributes.
    - Traffic Shaping – A bandwidth management technique to normalize/prioritize network resources according to a traffic profile
  - To dynamically change the policies according to the analytics results.
    - AI / MI can be used to establish normalized profiles and dynamically update the profiles based on a set of predetermined or dynamically learned rules.

Use Cases of a smart and intelligent SDN controller

- Traffic Control and Routing Optimization
  - Congestion Control
  - Traffic Pattern Prediction
  - Routing Optimization
- Security and Anomaly Detection
  - DDoS attack detection and mitigation
  - Detect and avoid Man in the Middle attacks
  - Identify altered IDS / IPS mechanisms
  - Identify illicit traffic replication, modification, fabrication
- Resource optimization
  - Networking resource allocation optimization
  - Cloud resource management optimization
- Troubleshooting and Self-healing

Example Use Case:
Traffic congestion prediction with automated control



SDN controlled network

OpenDaylight + TSDR

1. Collect stats from the network and store into TSDR
2. Data analysis through data analytics engines integration
3. Traffic flow redirection from A->F to A->B->F and A->D->E->F

Predicted congestion path in the next 24 hours

Healthy path in the next 24 hours

Use Cases of a smart and intelligent SDN controller

- Traffic Control and Routing Optimization
  - Congestion Control
  - Traffic Pattern Prediction
  - Routing Optimization
- Security and Anomaly Detection
  - DDoS attack detection and mitigation
  - Detect and avoid Man in the Middle attacks
  - Identify altered IDS / IPS mechanisms
  - Identify illicit traffic replication, modification, fabrication
- Resource optimization
  - Networking resource allocation optimization
  - Cloud resource management optimization
- Troubleshooting and Self-healing

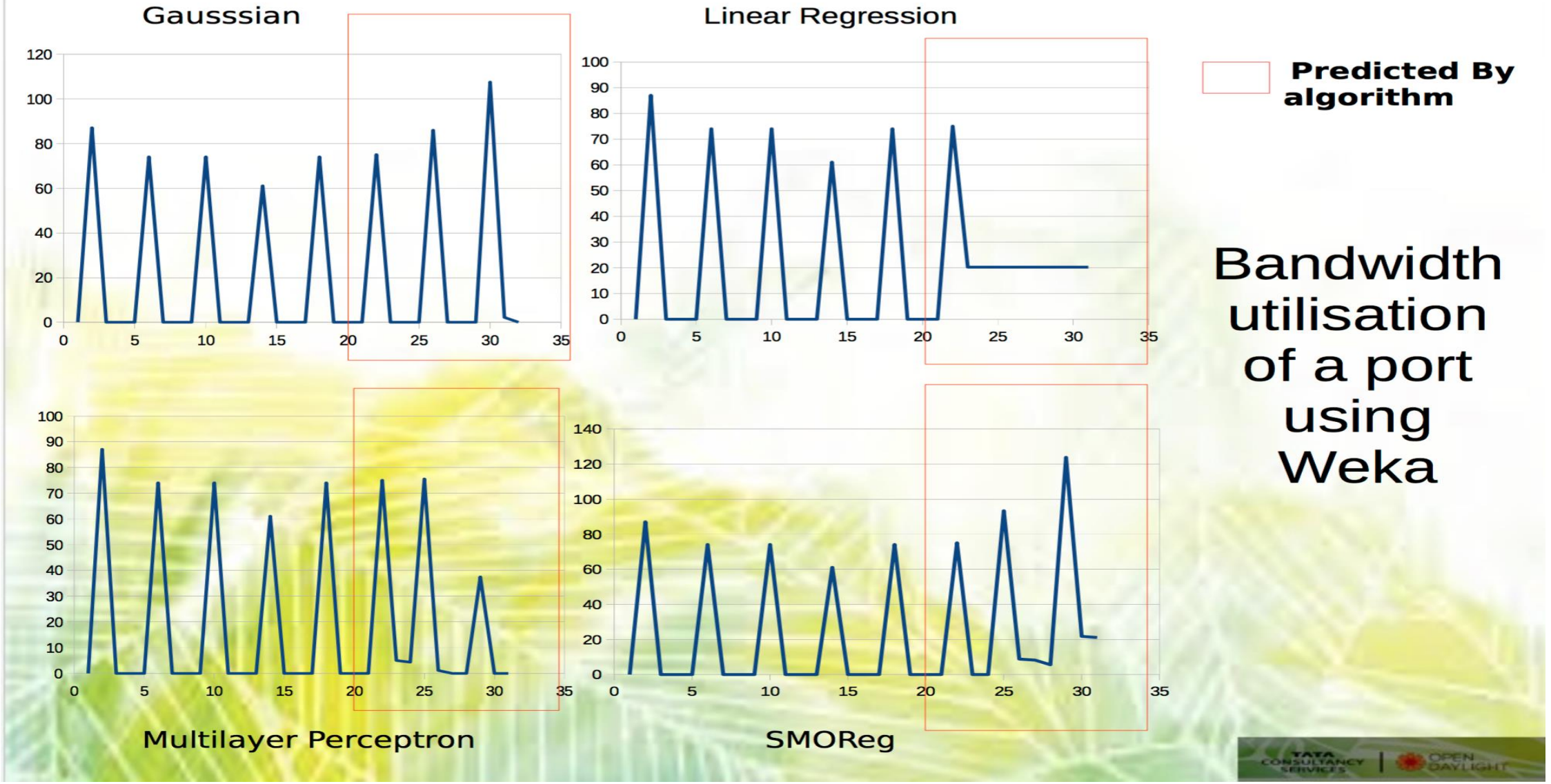
Security and Anomaly Detection

- DDoS attach detection and mitigation
  - example: AI/MI algorithms detect HTTP POST attack. Attacker sends thousands of valid HTTP POST headers that specify “Content-Length”, then the attacker very slowly sends very small packets to force the server to wait on the entire “Content-Length”, causing server side DDOS. Profile monitors can detect the patterns this causes (source address plus subsequent very small packets over time) and mitigate by dropping the packets causing a time out and recovery on the server.
- Detect and avoid Man in the Middle attacks
  - Inter/intra switch packet latency, as determined by metrics from several different sources can indicate that a flow has been held up while the packets are being tampered with. AI time series analysis of the latency can detect trends and point the IDS to the malicious actor (or network).

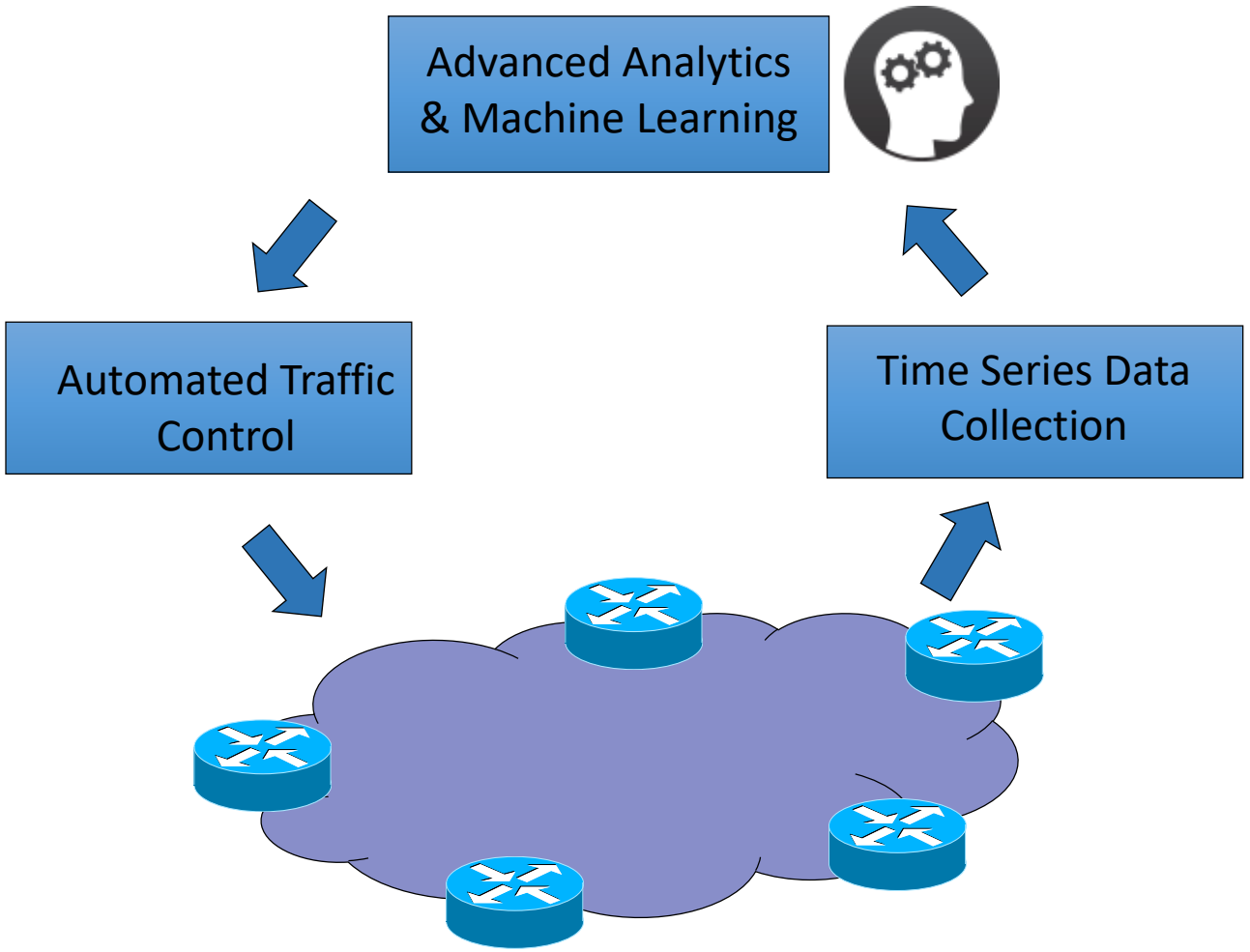
Use Cases of a smart and intelligent SDN controller

- Traffic Control and Routing Optimization
  - Congestion Control
  - Traffic Pattern Prediction
  - Routing Optimization
- Security and Anomaly Detection
  - DDoS attack detection and mitigation
  - Detect and avoid Man in the Middle attacks
  - Identify altered IDS / IPS mechanisms
  - Identify illicit traffic replication, modification, fabrication
- Resource optimization
  - Networking resource allocation optimization
  - Cloud resource management optimization
- Troubleshooting and Self-healing

Prediction using Weka leveraging data collected in TSDR



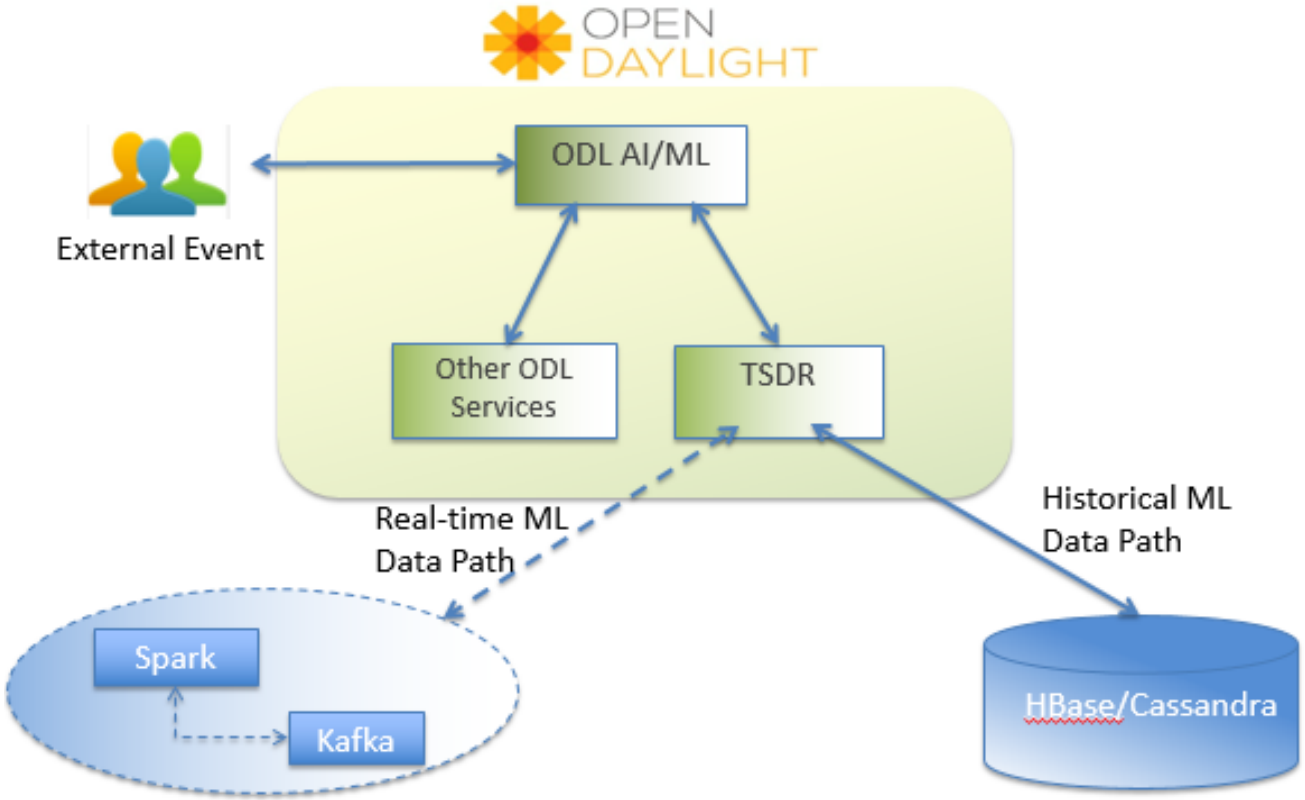
How to realize a smart and intelligent SDN Controller



- Network status awareness
  - Rely on time series data collected from the network
- Traffic Control Policy Change decision making
  - Based on the advanced analytics and machine learning.
- Dynamic change of Control policies
  - Automatically change the traffic control policies based on the analytics results.

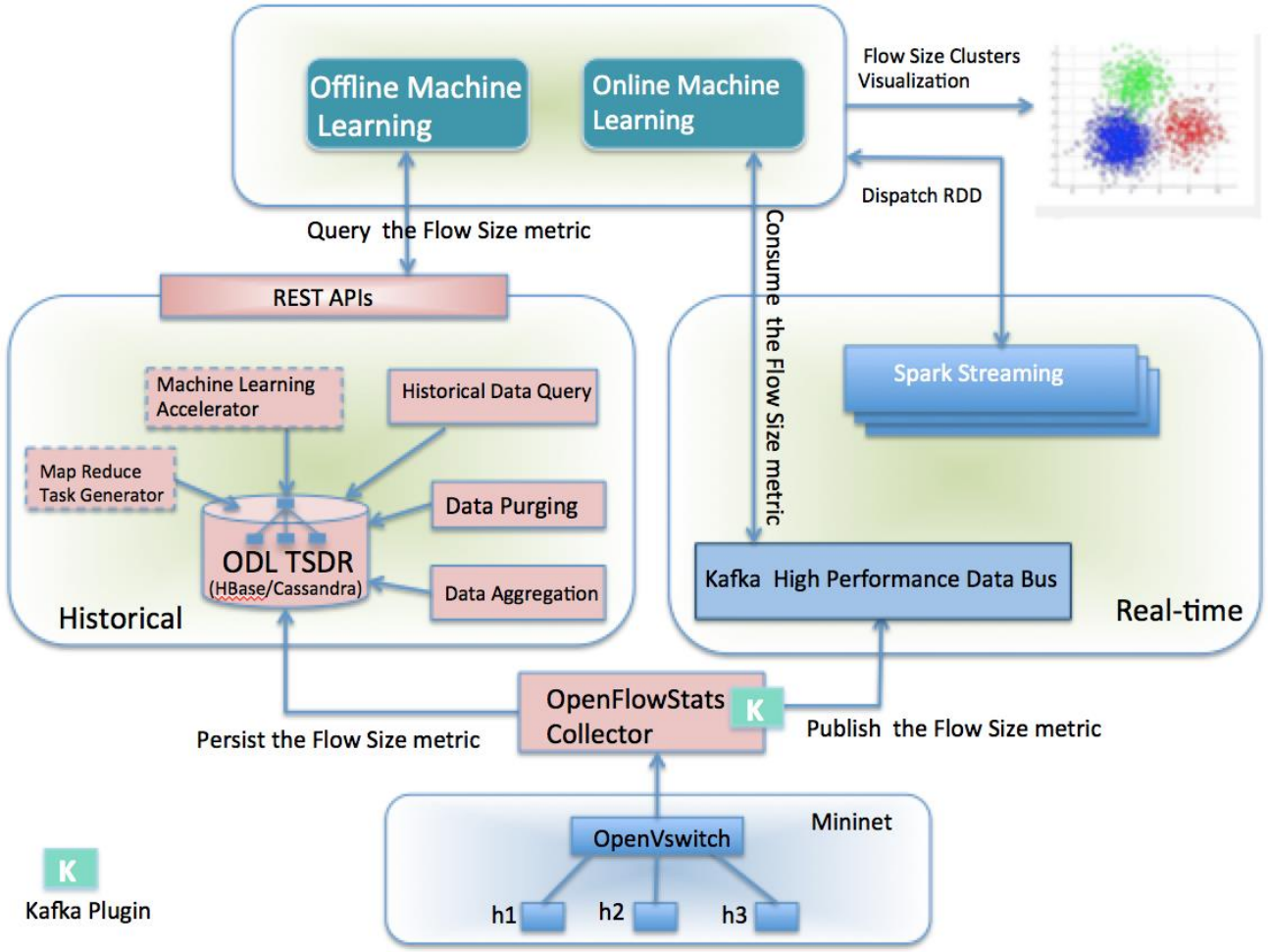
AI/ML framework in the ODL ecosystem

- Enable AI/ML on both historical and real-time data paths.
- Many use cases would require both offline and online ML on the time series data.
- External events could be additional input for accurate machine learning results.
- Feed back the results to SDN control path for automatic traffic steering and policy placement.
- Well-defined interface among the components towards future standardization of advanced analytics in SDN.



ODL AI/ML framework PoC Architecture

- PoC of both historical offline machine learning and real-time online machine learning
  - Collect the time series data
  - Persist into scalable data storage
  - Publish to high performance data bus
- Integrate with external machine learning libraries
  - Spark MLlib
  - DeepLearning4J
- Collect OpenFlow Stats and apply machine learning algorithms
  - *k*-means clustering



Acknowledgements

- Yuling Chen
- Scott Melton

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

- Click to add Text

Click to add Title

- Click to add Text

- Click to add Text

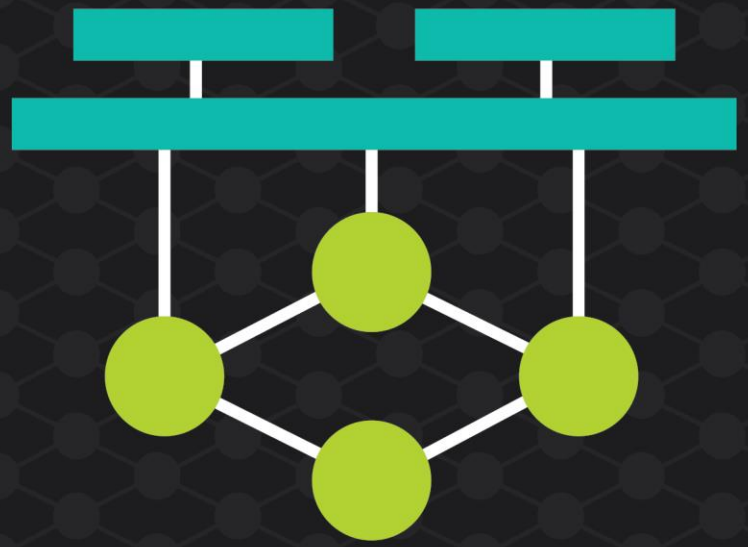
Click to add Title

- Click to add Text

Click to add Title

- Click to add Text

- Click to add Text



ons

EUROPE

OPEN NETWORKING //
Enabling Collaborative
Development & Innovation

Hosted By

THE LINUX FOUNDATION | LF NETWORKING