CMPE 209 Network Security

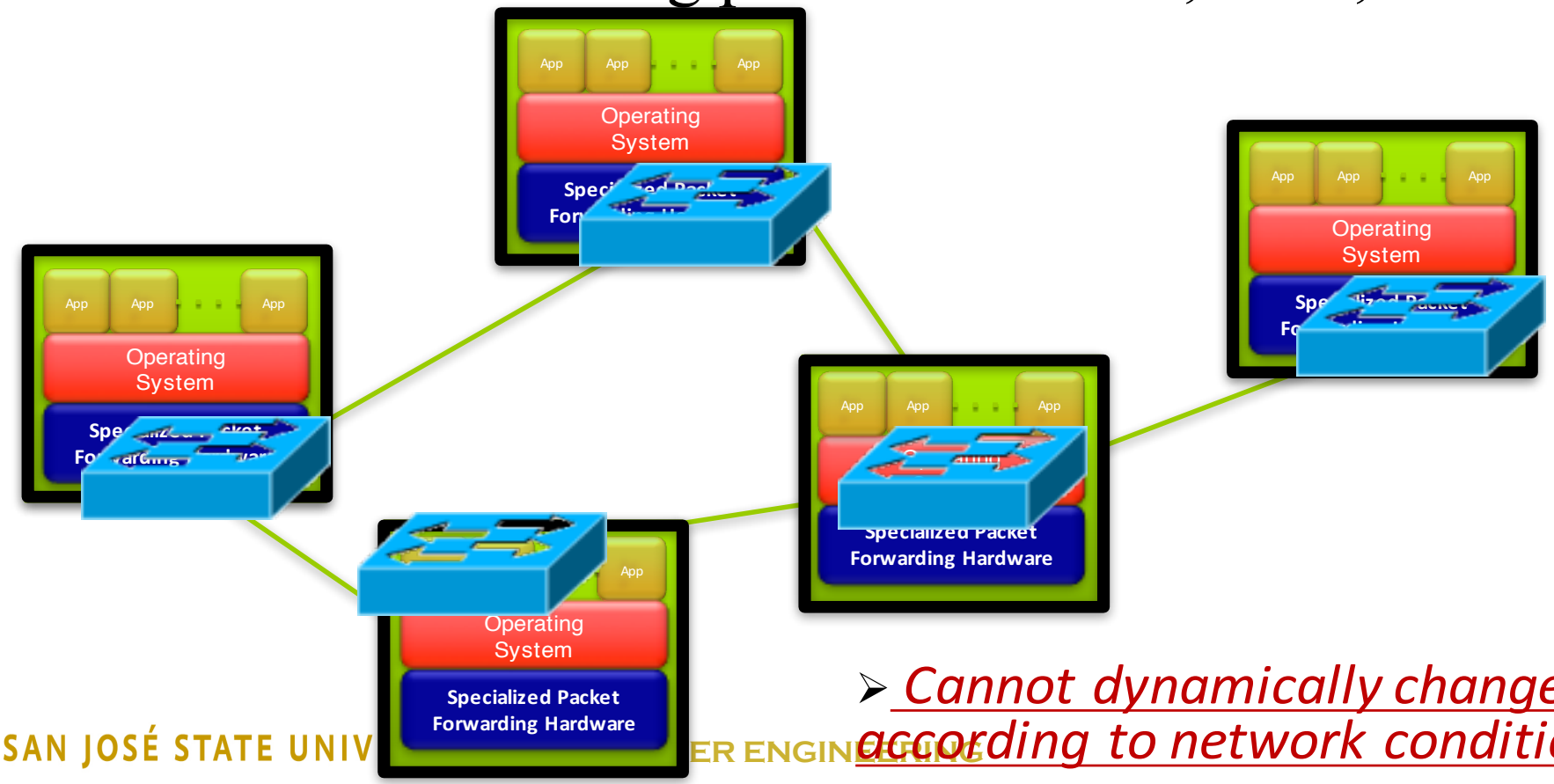
SDN and Security

Dr. Younghee Park

SAN JOSÉ STATE UNIVERSITY

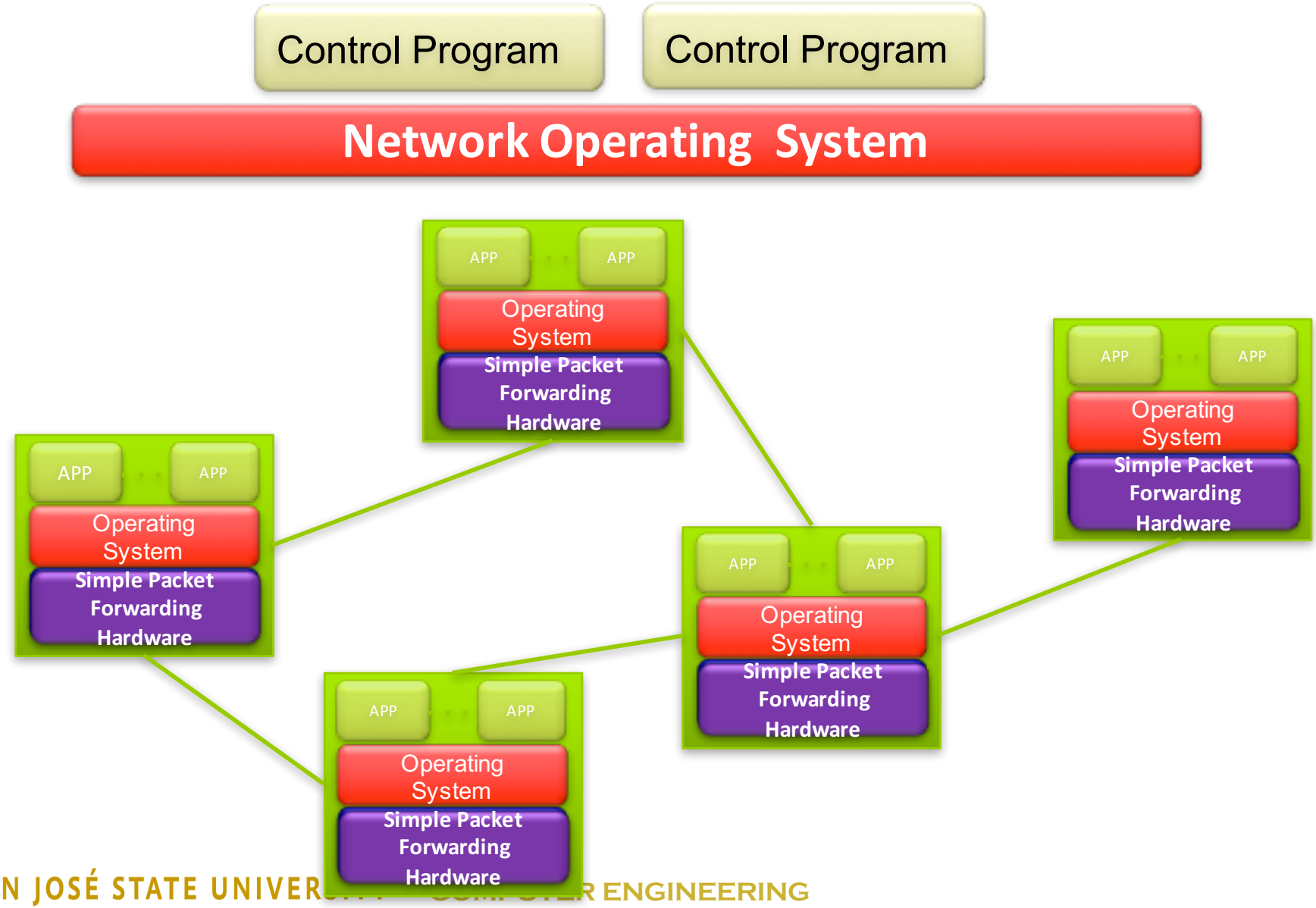
Limitation of Current Networks

- Classical network architecture
  - Distributed control plane
  - Distributed routing protocols: OSPF, BGP, etc.



➢ *Cannot dynamically change according to network conditions*

Reconstruction Networks -SDN



Requirement of the SDN Approach

Adaptability	•Networks must adjust and respond dynamically, based on application needs, business policy, and network conditions
Automation	•Policy changes must be automatically propagated so that manual work and errors can be reduced
Maintainability	•Introduction of new features and capabilities must be seamless with minimal disruption of operations
Model management	•Network management software must allow management of the network at a model level, rather than implementing conceptual changes by reconfiguring individual network elements
Mobility	•Control functionality must accommodate mobility, including mobile user devices and virtual servers
Integrated security	•Network applications must integrate seamless security as a core service instead of as an add-on solution
On-demand scaling	•Implementations must have the ability to scale up or scale down the network and its services to support on-demand requests

SDN

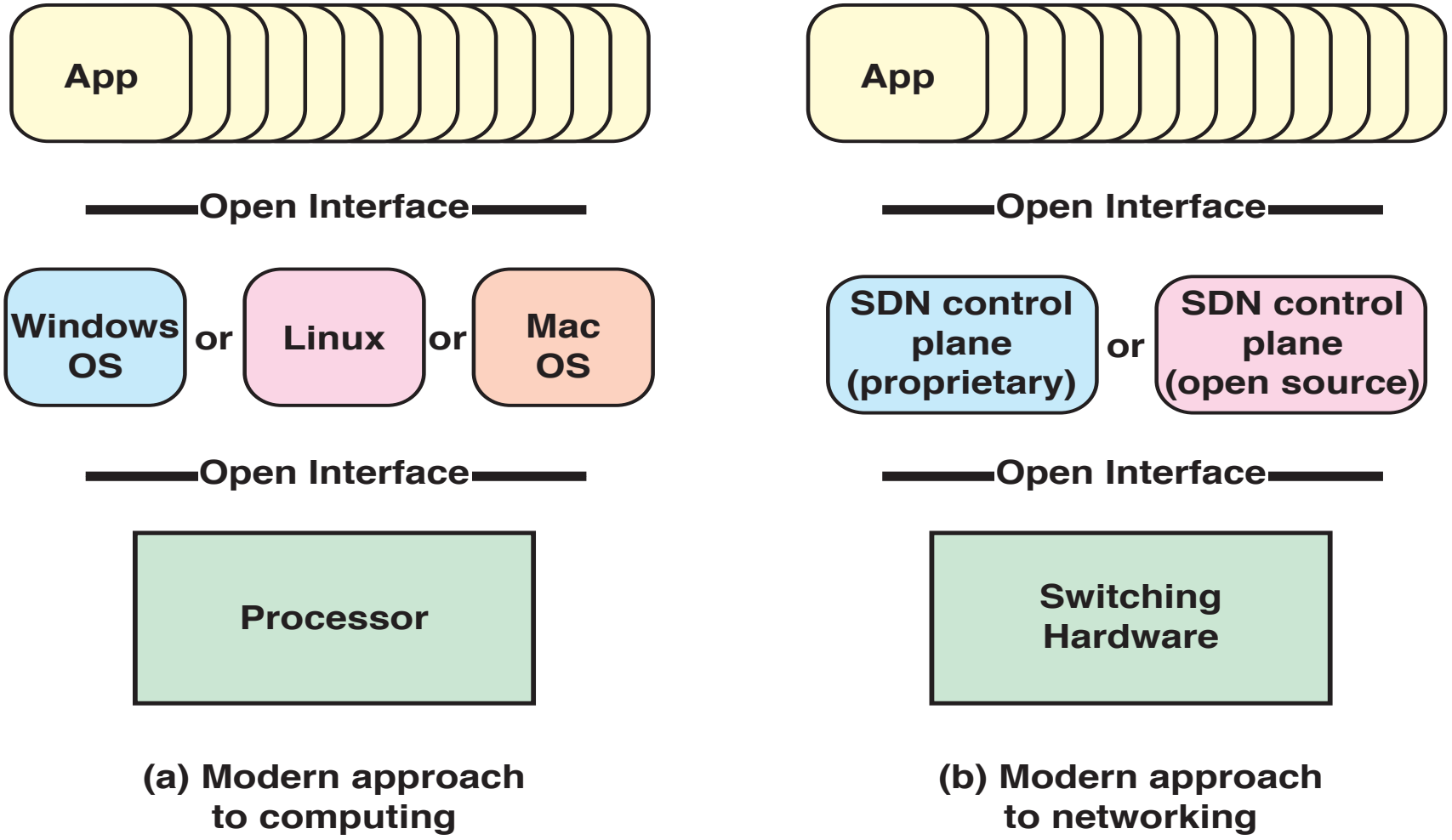


Figure 3.1 The Modern Approach to Computing and Networking

SDN

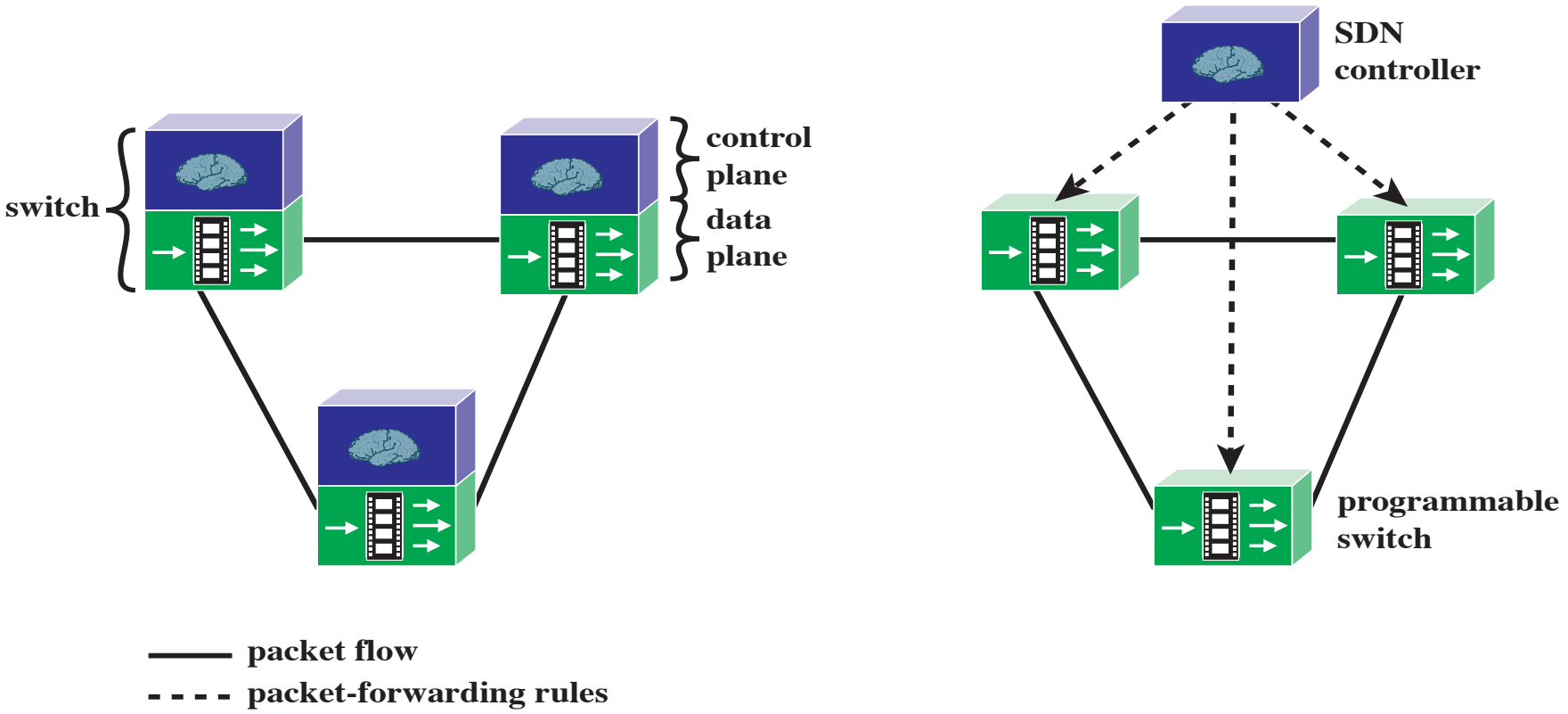


Figure 3.2 Control and Data Planes

SDN

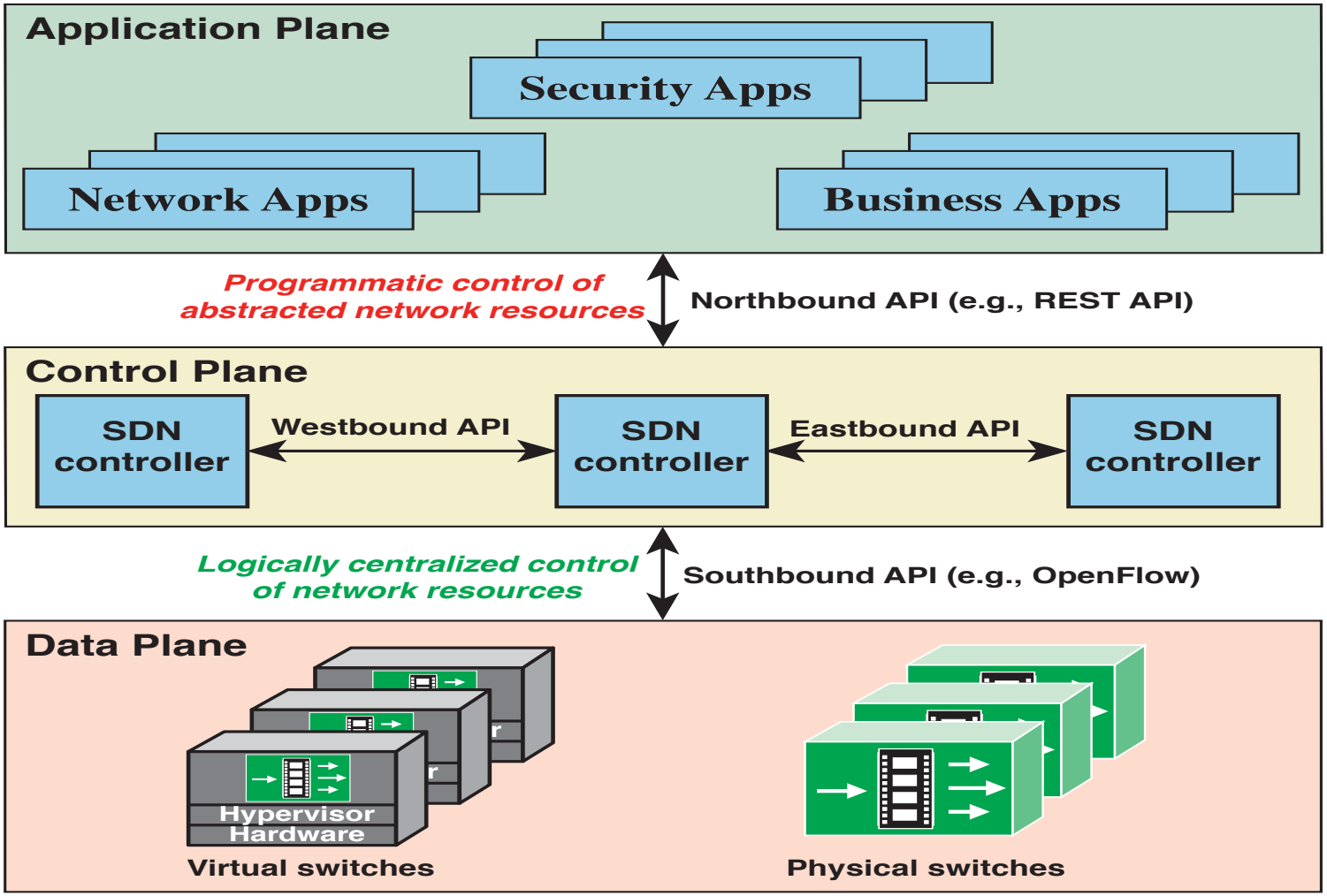


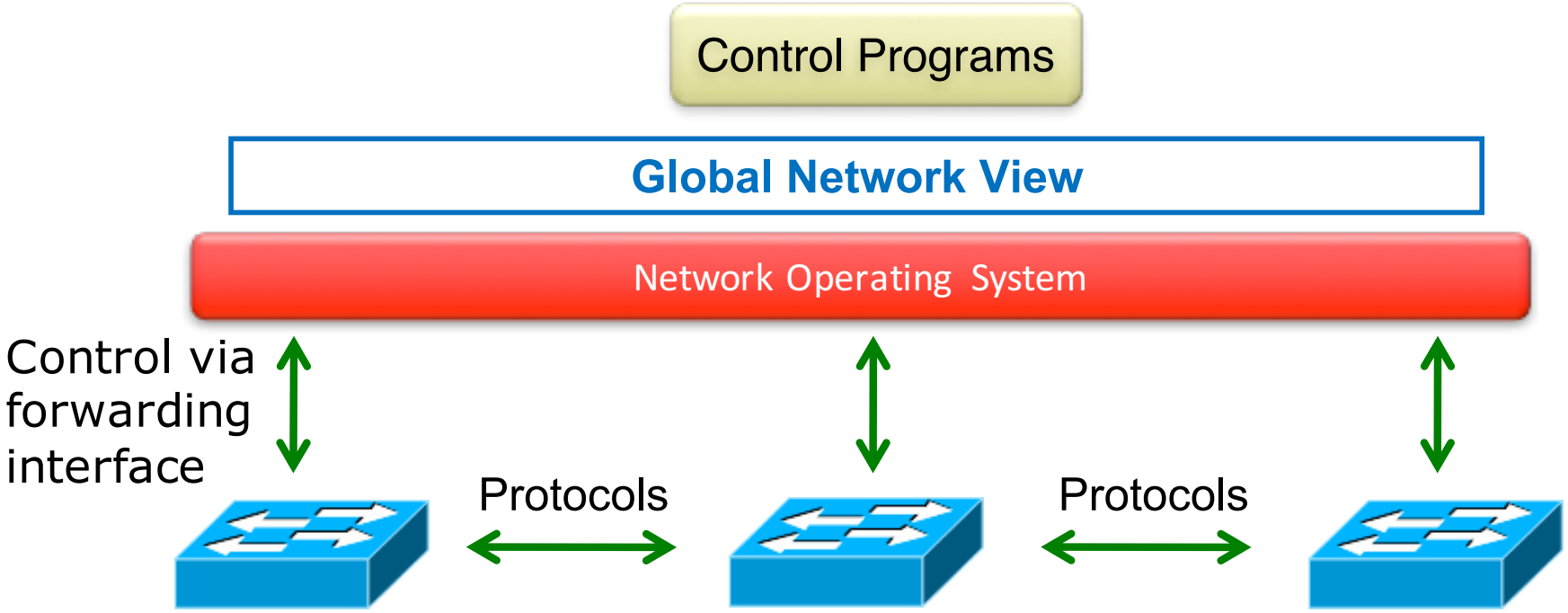
Figure 3.3 SDN Architecture

Characteristics of SDN

- The control plane is separated from the data plane; data plane devices become simple packet-forwarding devices
- The control plane is implemented in a centralized controller or set of coordinated centralized controllers
    - The SDN controller has a centralized view of the network or networks under its control
    - The controller is portable software that can run on commodity servers and is capable of programming the forwarding devices based on a centralized view of the network
- Open interfaces are defined between the devices in the control plane (controllers) and those in the data plane
- The network is programmable by applications running on top of the SDN controllers; the SDN controllers present an abstract view of network resources to the applications.

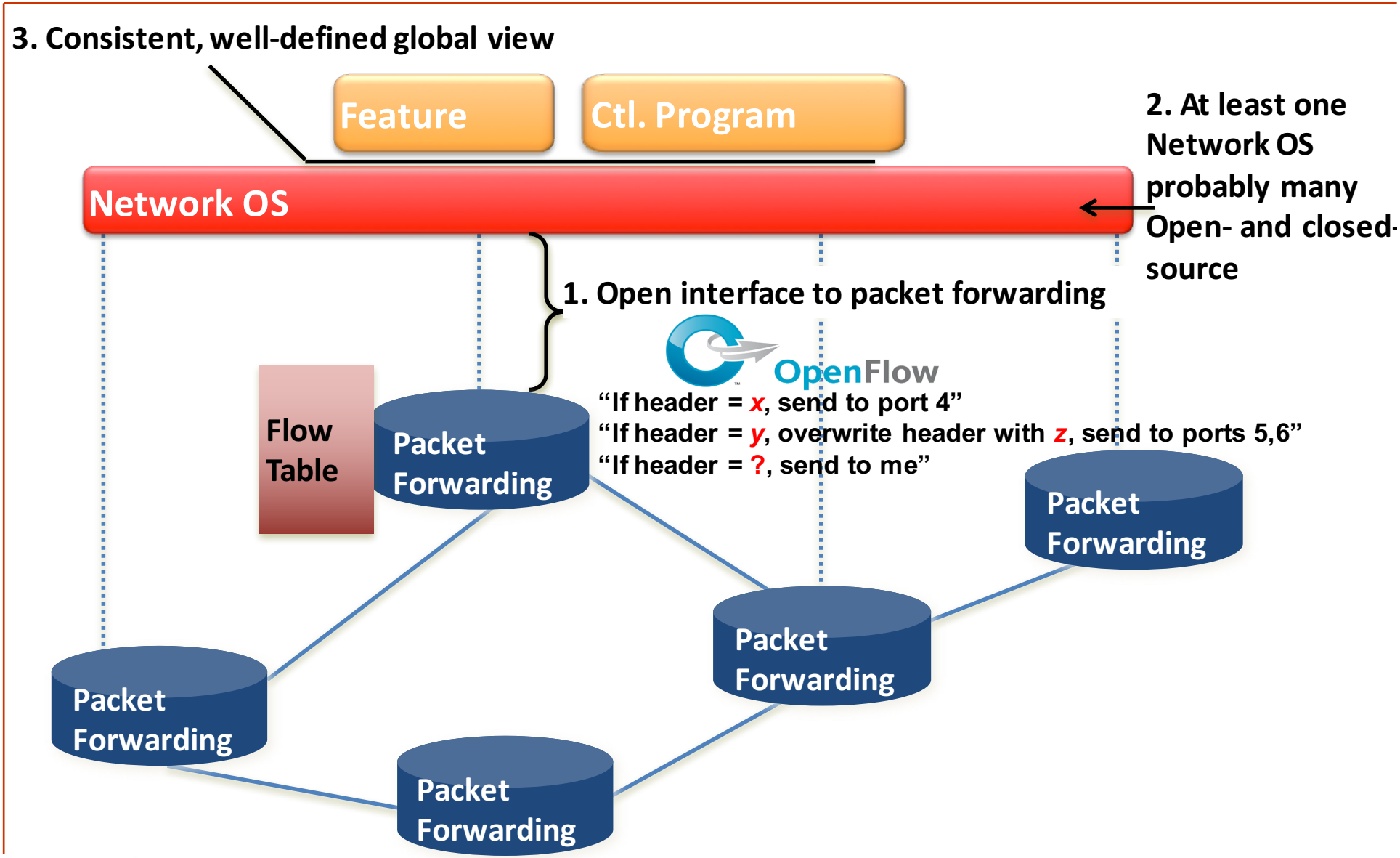
Restructuring Networks - SDN

- "NOX: Towards an Operating System for Networks"



- http://benpfaff.org/papers/nox.pdf

Separation of control, forwarding planes

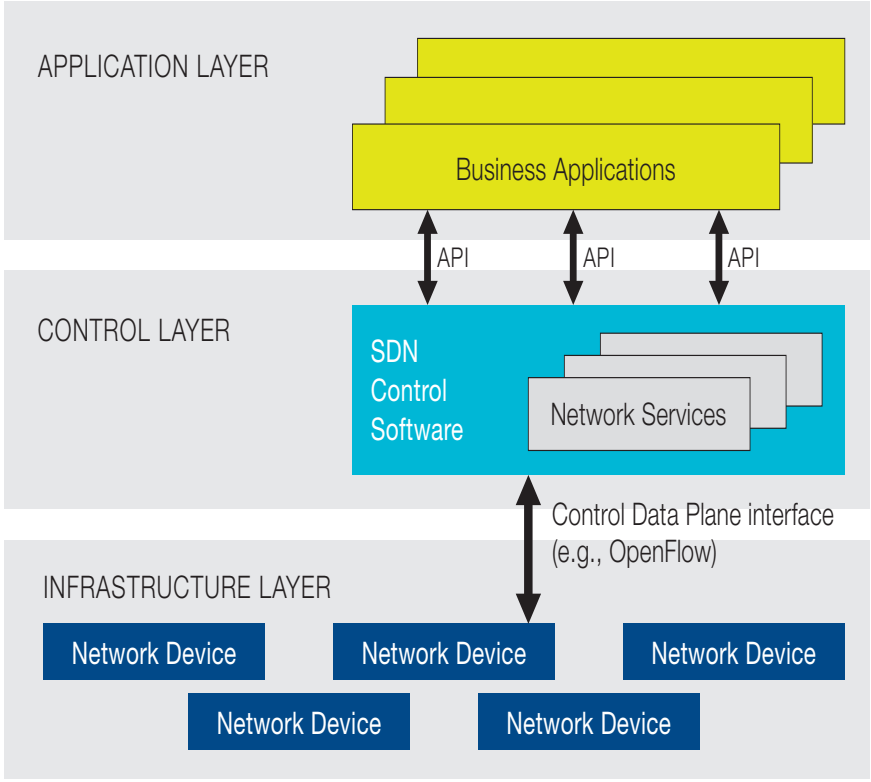


Useful Links for your study

- https://www.sdxcentral.com
- https://www.opennetworking.org/
- Open Networking Summit
- Openflow protocol https://osrg.github.io/ryu-book/en/html/openflow_protocol.html

SDN Architecture

- Separate the control plane from the data plane
- Three layers
  - Application layer
    - Application part
    - Implements logic
  - Control layer
    - Kernel part
    - Runs applications
  - Infrastructure layer
    - Data plane
    - Network switch or router

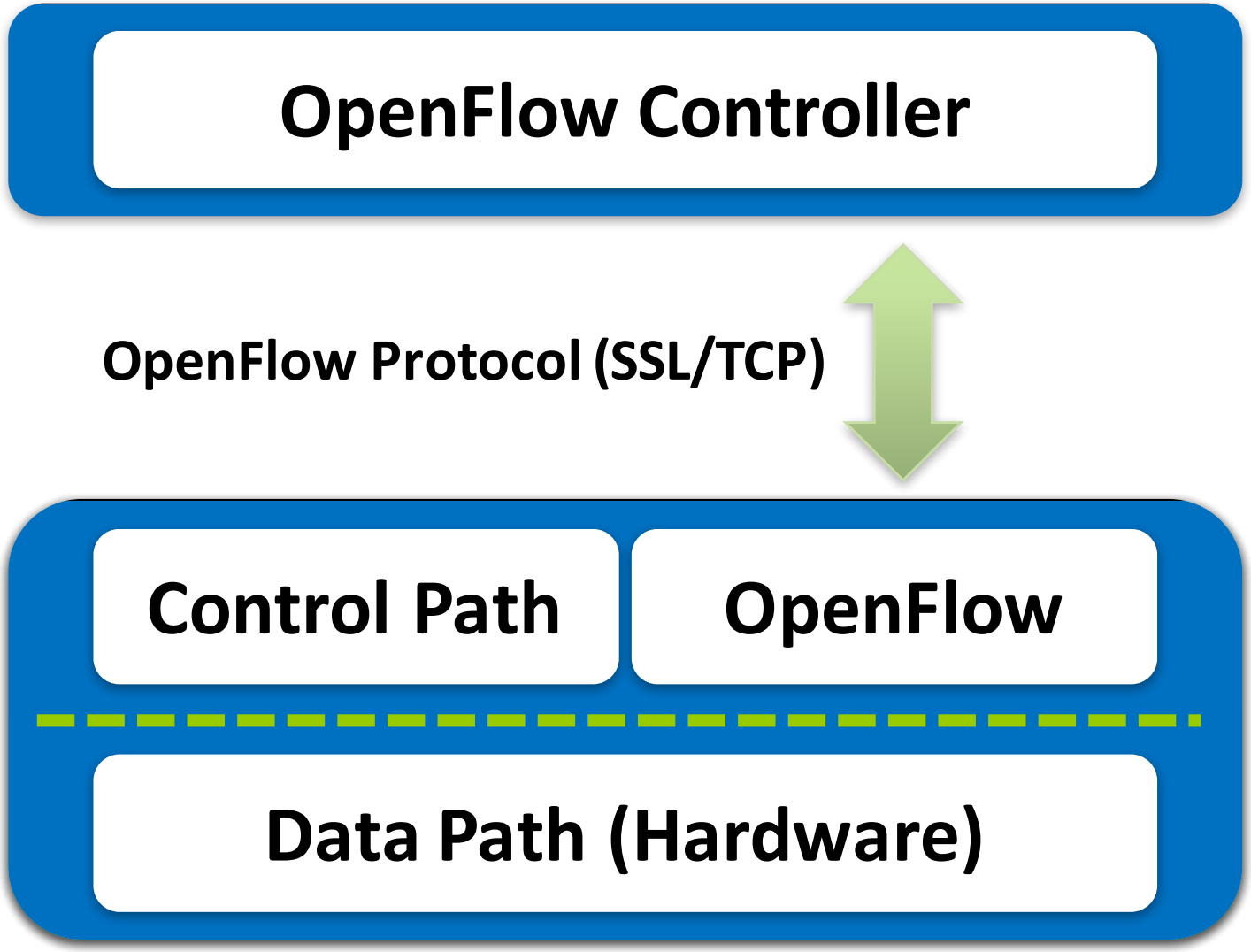


From Open Networking Foundation

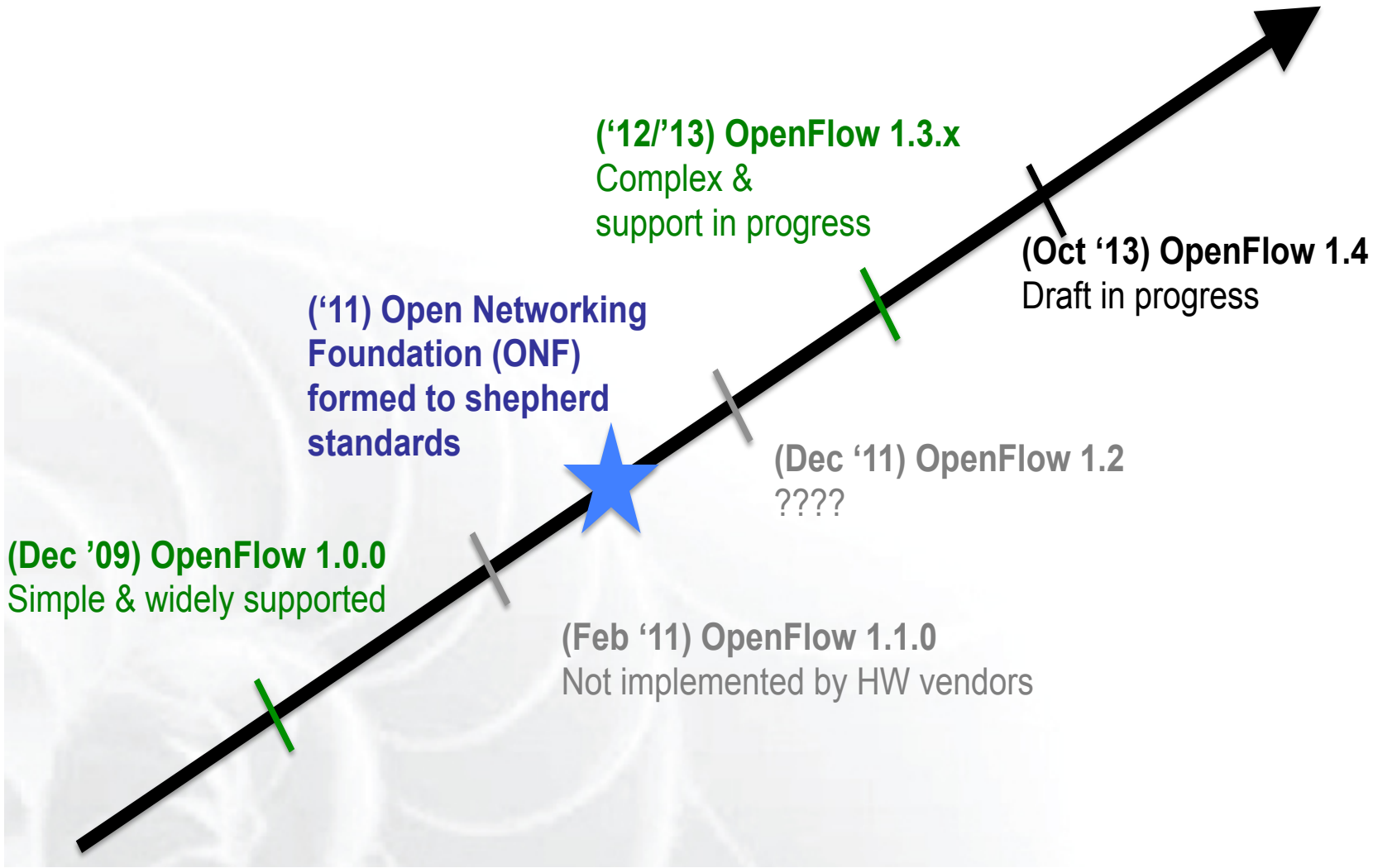
SDN Definition

- SDN is a framework to allow network administrators to automatically and dynamically manage and control a large number of network devices, services, topology, traffic paths, and packet handling (quality of service) policies using high-level languages and APIs.
- Management includes provisioning, operating, monitoring, optimizing, and managing FCAPS (faults, configuration, accounting, performance, and security) in a multi-tenant environment.
  - Key: Dynamic & Quick
    - Legacy approaches such as CLI were not quick particularly for large networks

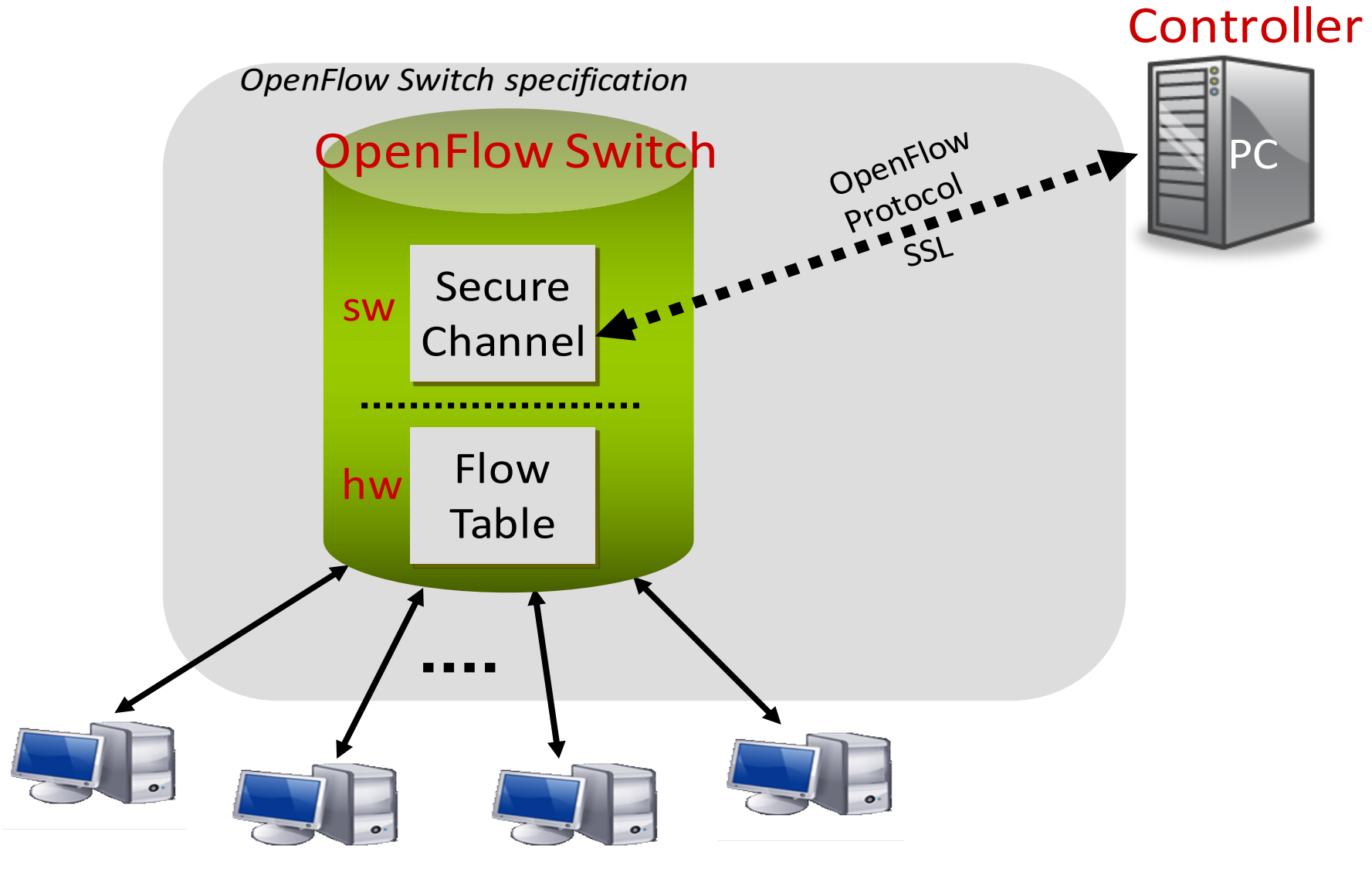
OpenFlow Concept



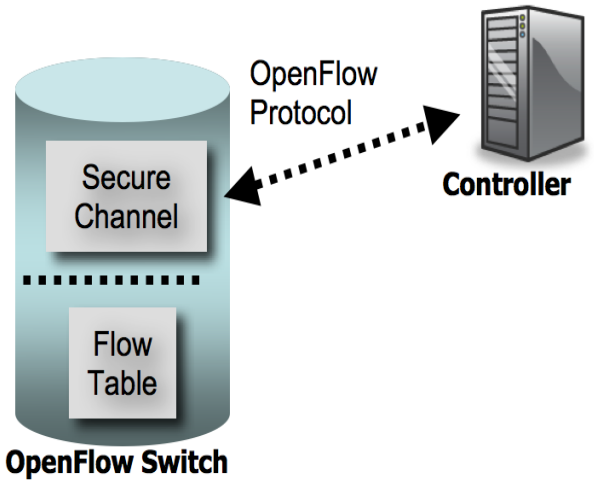
OpenFlow Versions



OpenFlow Network Architecture

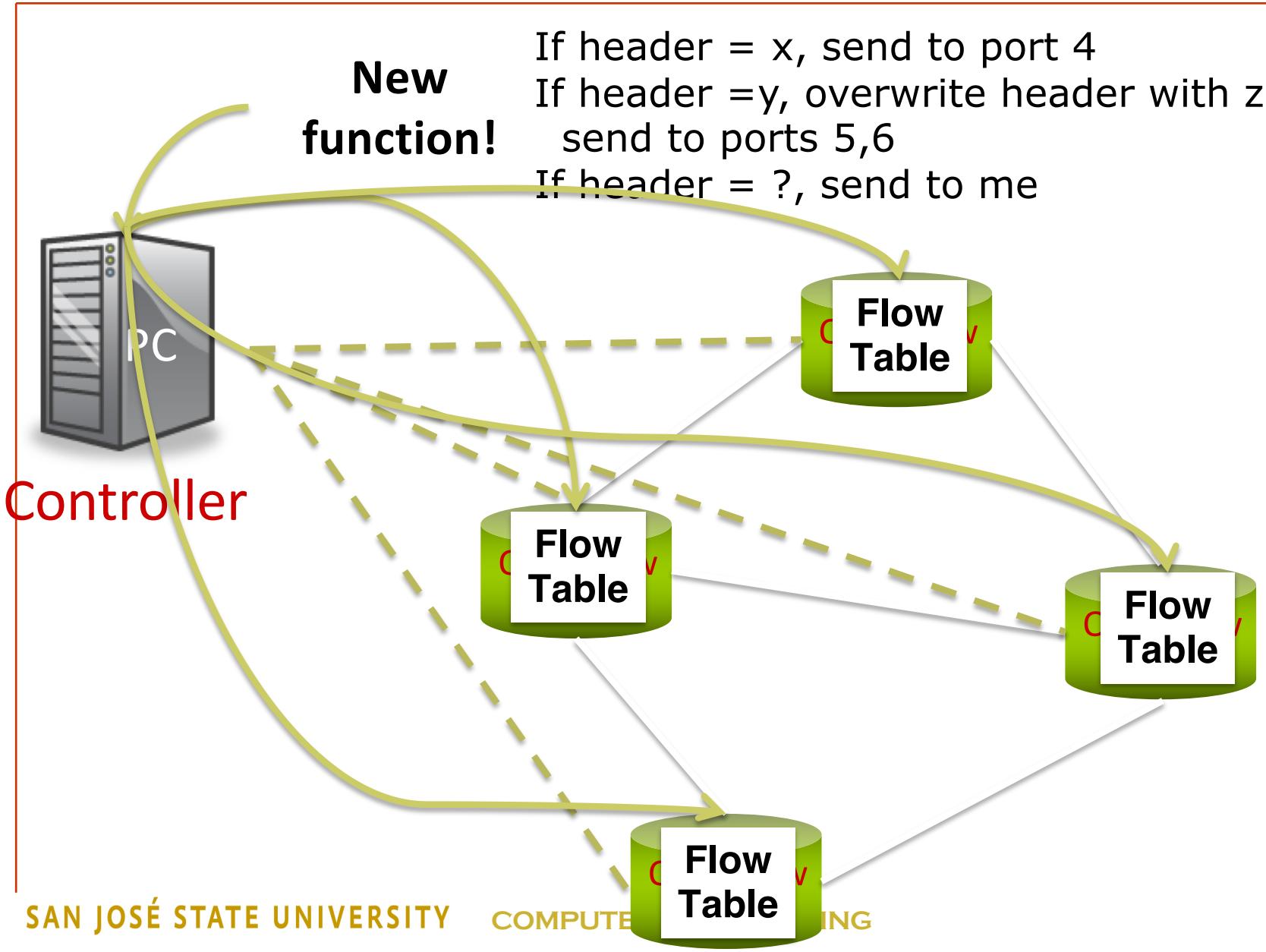


OpenFlow (OF) protocol



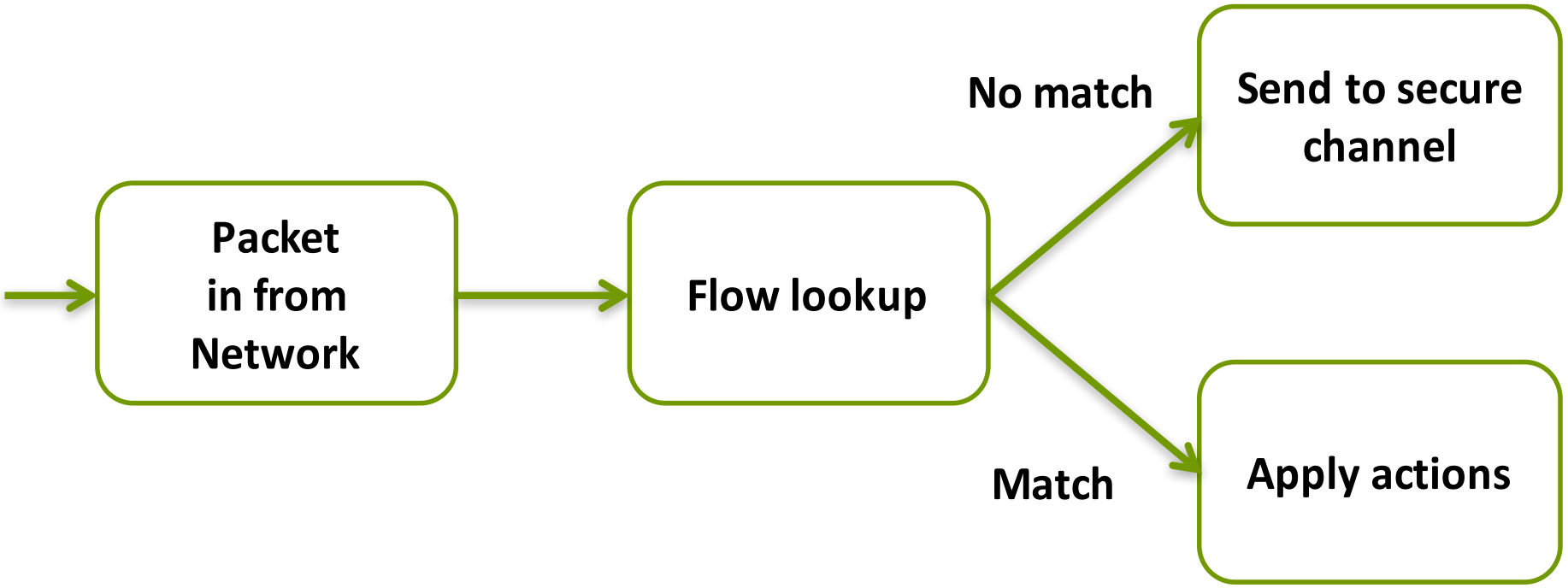
- A communications protocol that gives access to the forwarding plane of a network switch or router over the network
- An enabler of Software defined networking (SDN)
- It enables controllers to determine the path of network packets through the network of switches.
  - An application program which gives instructions as to which flows go on which elements, W/O which Controller cannot do anything.
- It is layered on top of the Transmission Control Protocol (TCP), and prescribes the use of Transport Layer Security (TLS).
- Controllers should listen on TCP port 6653 for switches that want to set up a connection.

Operation Step



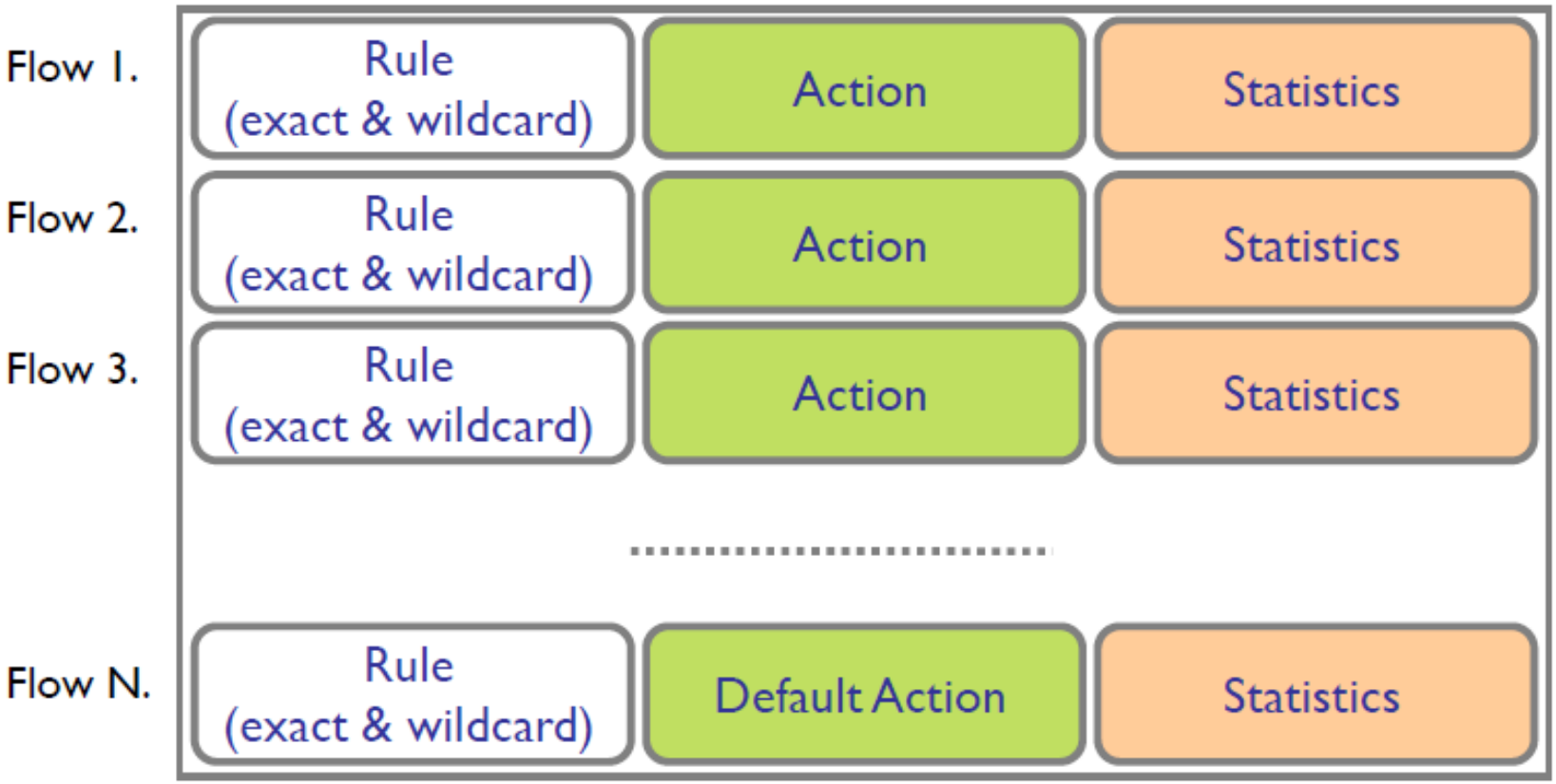
Packet Processing

- OpenFlow SW's Packet Processing
  - Search a matched entry of flow table with arriving packet's information

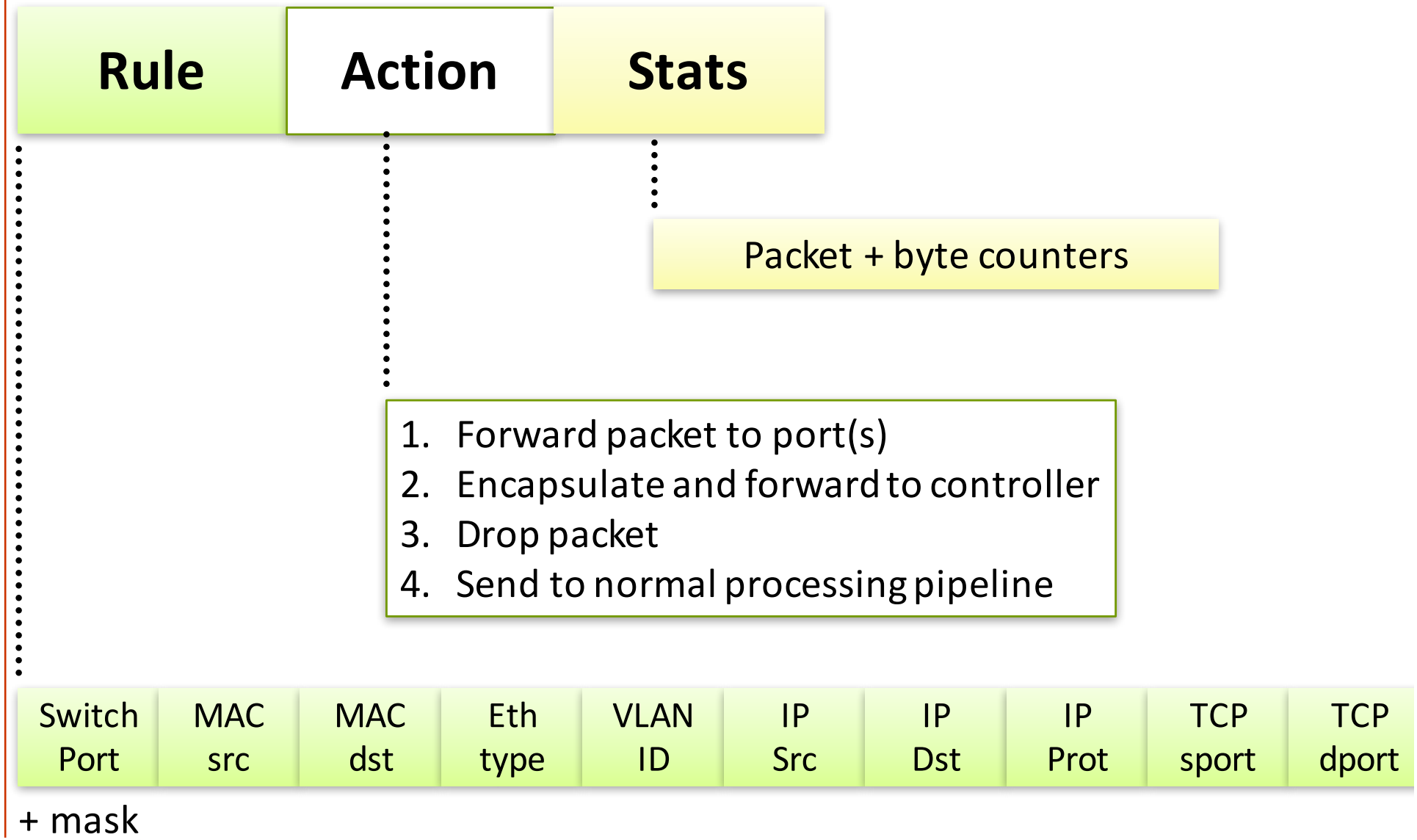


Flow Table Structure

- Exploit flow table in switches, routers, and chipsets



Flow Table Entry



Flow Table Entry Examples

Ethernet Switching

Switch Port	MAC src	**MAC dst**	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	**00:1F:.**	*	*	*	*	*	*	*	port6

IP Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	**IP Dst**	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	**5.6.7.8**	*	*	*	port6

Application Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	**TCP dport**	Action
*	*	*	*	*	*	*	*	*	**22**	drop

Flow Table Entry Examples

Flow Switching

Switch Port	**MAC src**	**MAC dst**	**Eth type**	**VLAN ID**	**IP Src**	**IP Dst**	**IP Prot**	**TCP sport**	**TCP dport**	Action
port3	00:2E:..	00:1F:.	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

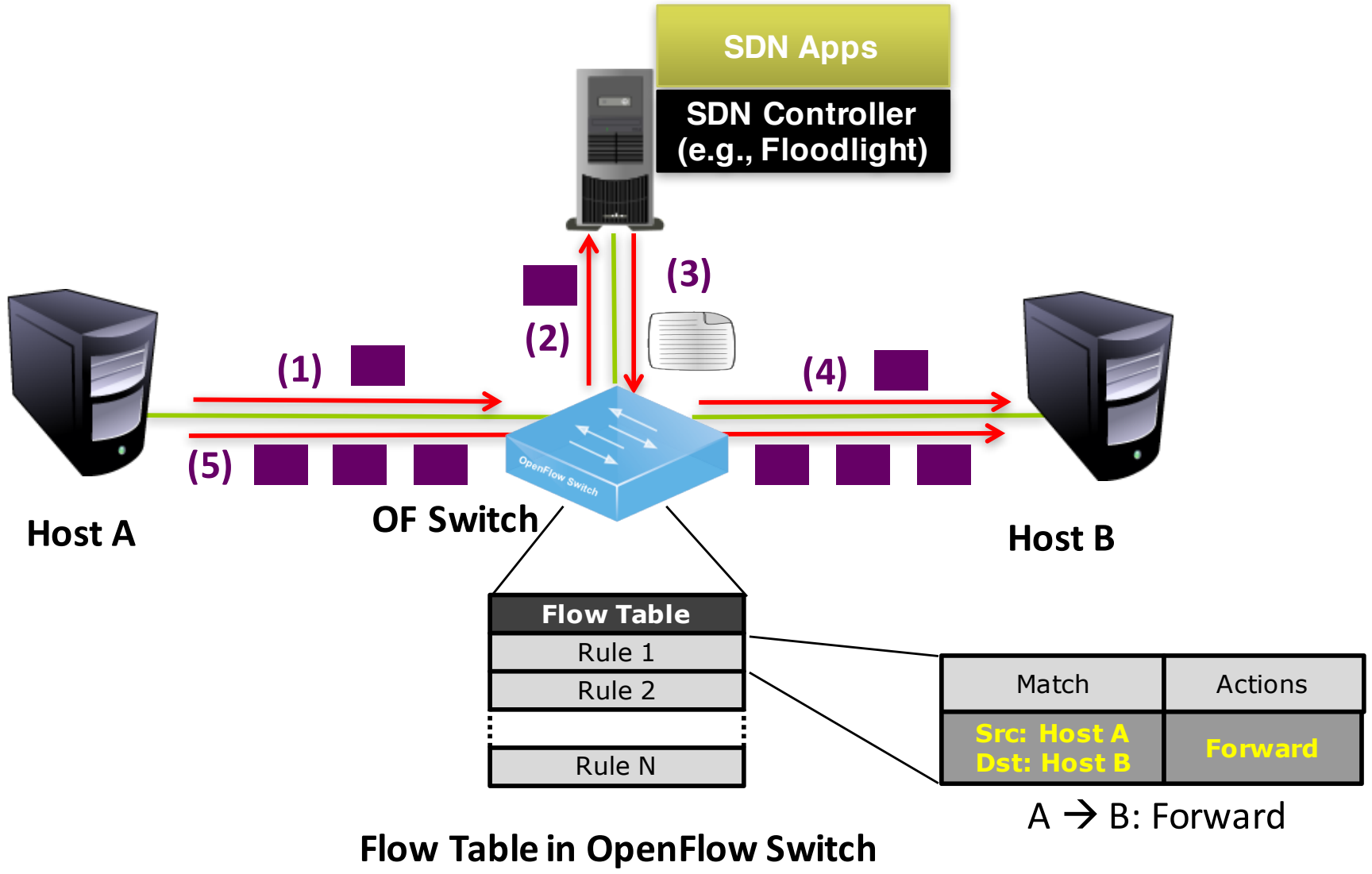
VLAN + App

Switch Port	MAC src	MAC dst	Eth type	**VLAN ID**	IP Src	IP Dst	IP Prot	TCP sport	**TCP dport**	Action
*	*	*	*	**vlan1**	*	*	*	*	**80**	port6

Port + Ethernet + IP

Switch Port	**MAC src**	MAC dst	**Eth type**	VLAN ID	IP Src	**IP Dst**	**IP Prot**	TCP sport	TCP dport	Action
port3	**00:2E:..**	*	**0800**	*	*	**5.6.7.8**	**4**	*	22	drop

OpenFlow Operation



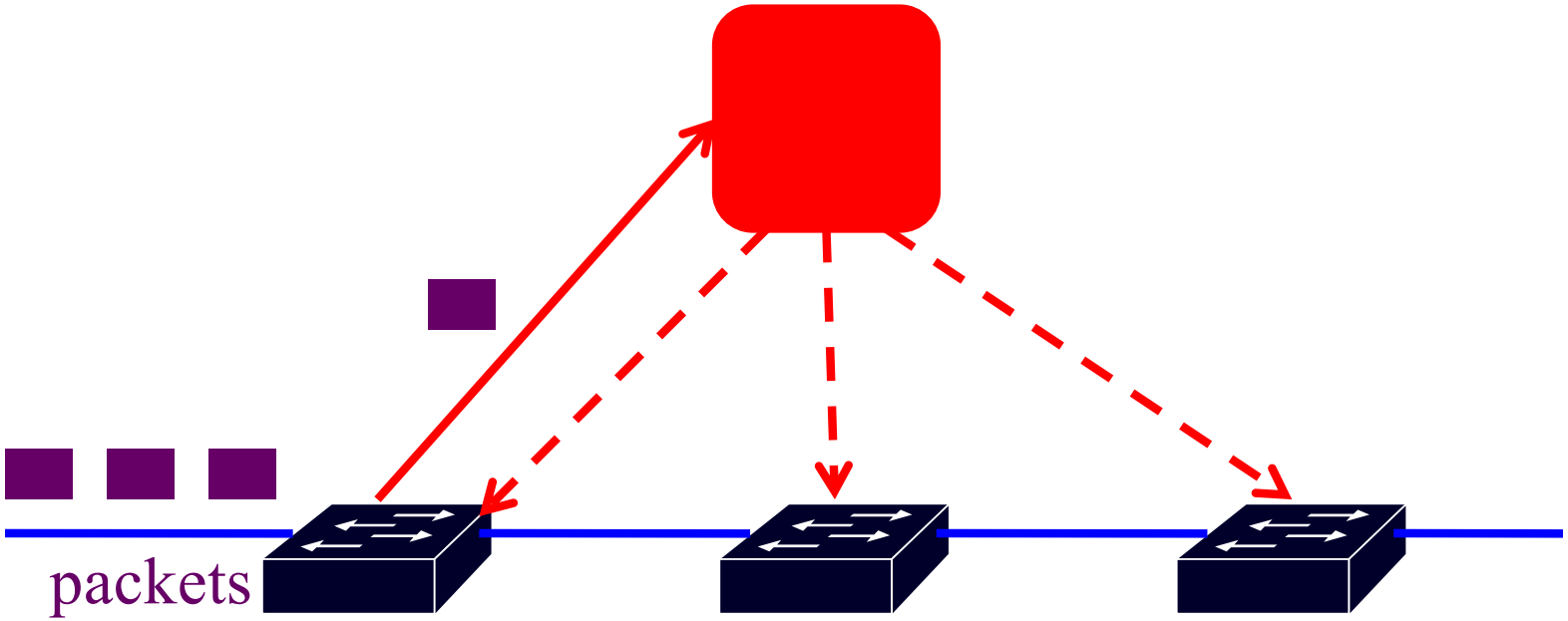
Flow Table in OpenFlow Switch

Challenges in SDNs

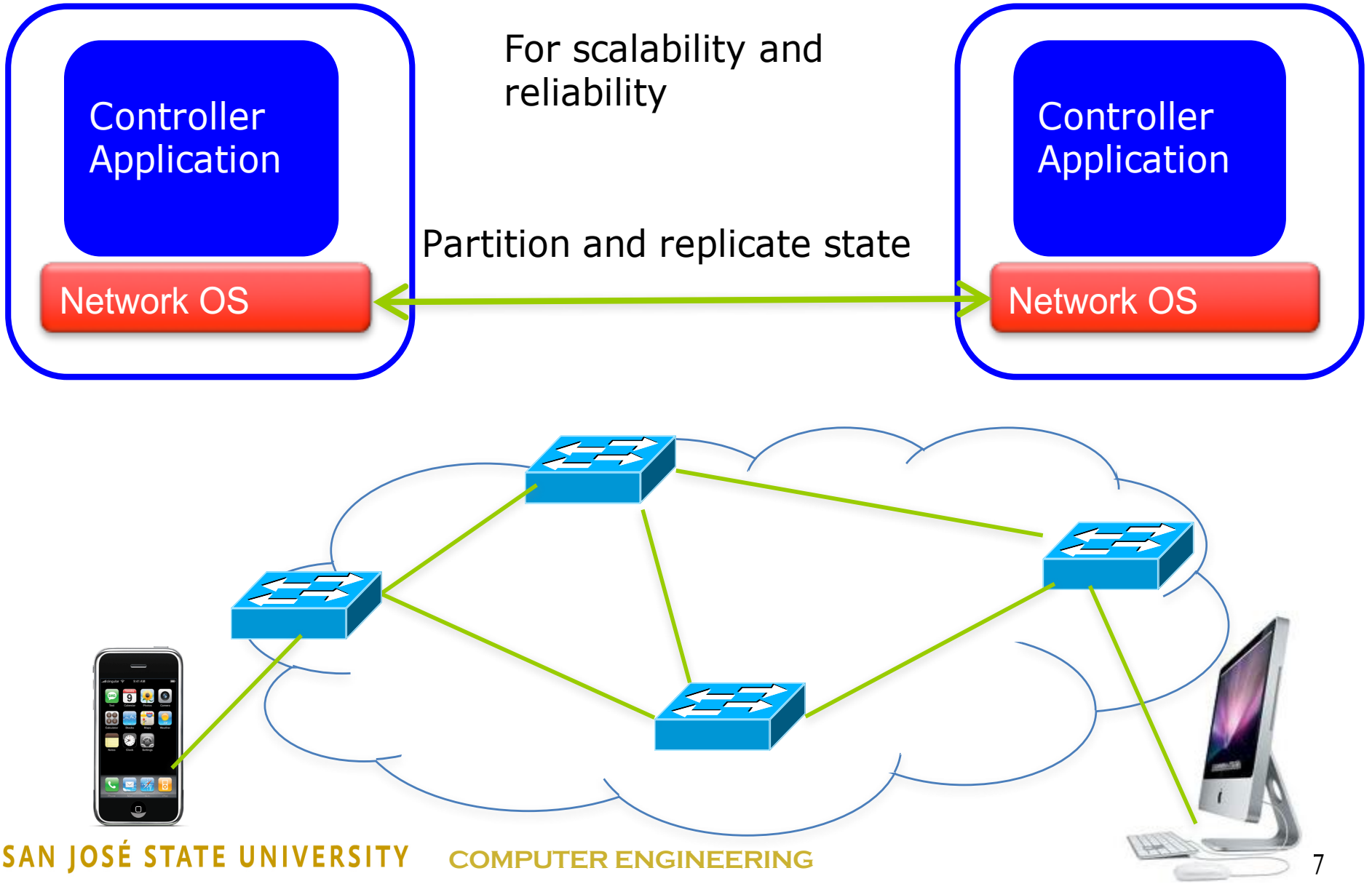
- Controller Delay and Overhead
- Distributed Controller
- Testing and Debugging
- Programming Abstractions
- …

Controller Delay and Overhead

- Controller is much slower the switch
- Processing packets leads to delay and overhead
- Need to keep most packets in the “fast path”



Distributed Controller



Testing and Debugging

- OpenFlow makes programming possible
  - Network-wide view at controller
  - Direct control over data plane
- Plenty of room for bugs
  - Still a complex, distributed system
- Need for testing techniques
  - Controller applications
  - Controller and switches
  - Rules installed in the switches

Summary of SDN

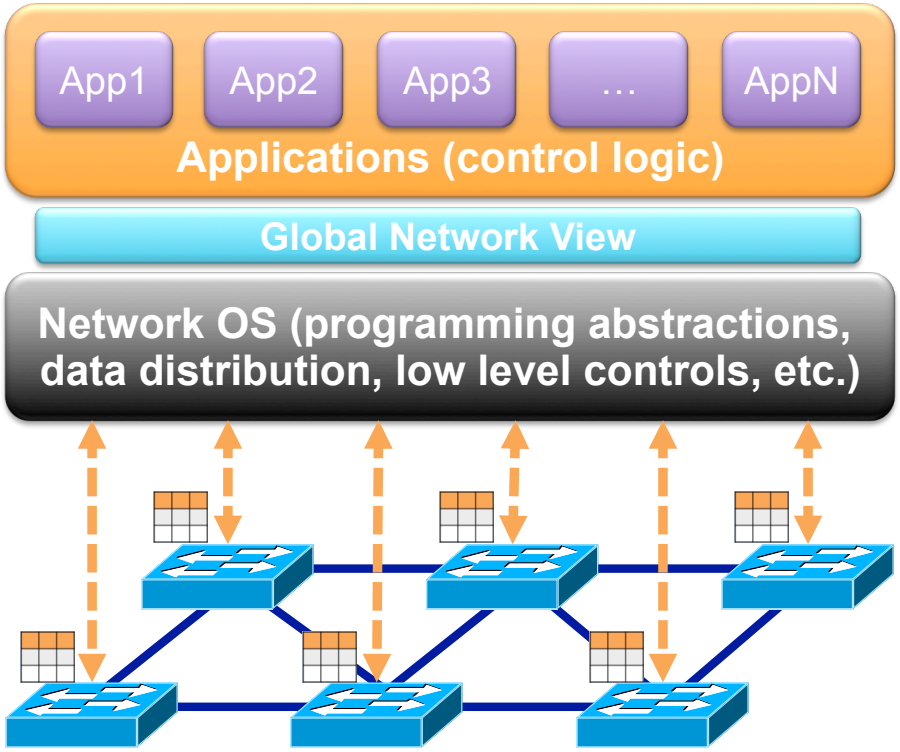
- Control and data planes **decoupled**
  - An enabler for innovation
  - More flexibility
- **Logical centralization** of network control
  - Easier to observe/infer and reason about network behavior
- Ability to **program** the network
  - Instead of configuring it (in a tedious, error-prone process)

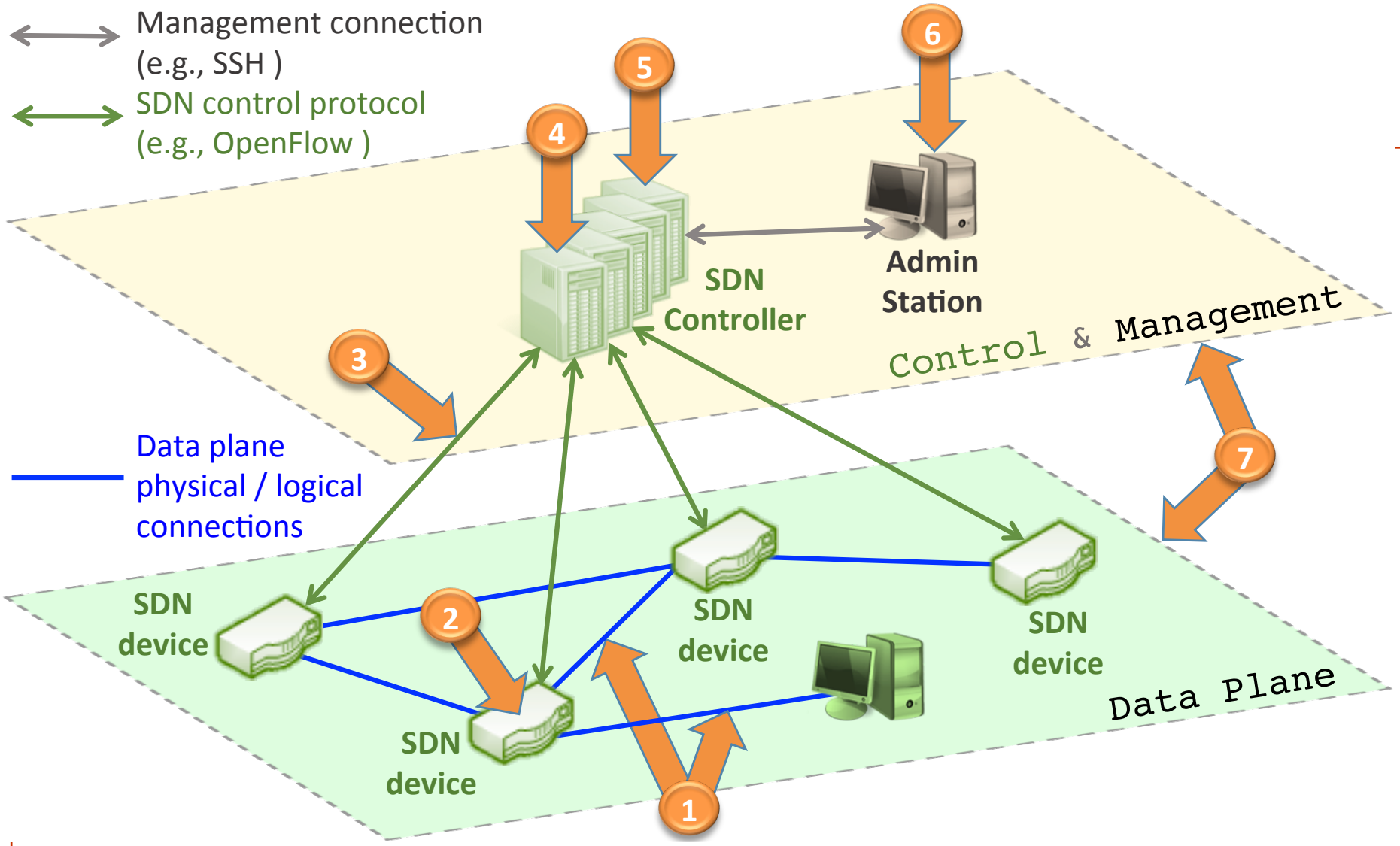
Useful Resources

- SDN/OpenFlow Reading List
  - https://sites.google.com/site/sdnreadinglist/
  - https://www.opennetworking.org/sdn-resources/sdn-reading-list
- Open Network Foundation
  - https://www.opennetworking.org/
- Floodlight - an Open SDN Controller
  - http://www.projectfloodlight.org/floodlight/
  - http://www.projectfloodlight.org
- RYU - https://osrg.github.io/ryu/
- SDN Simulation Environment
  - Please install Mininet on your laptop.
  - Mininet: http://mininet.org

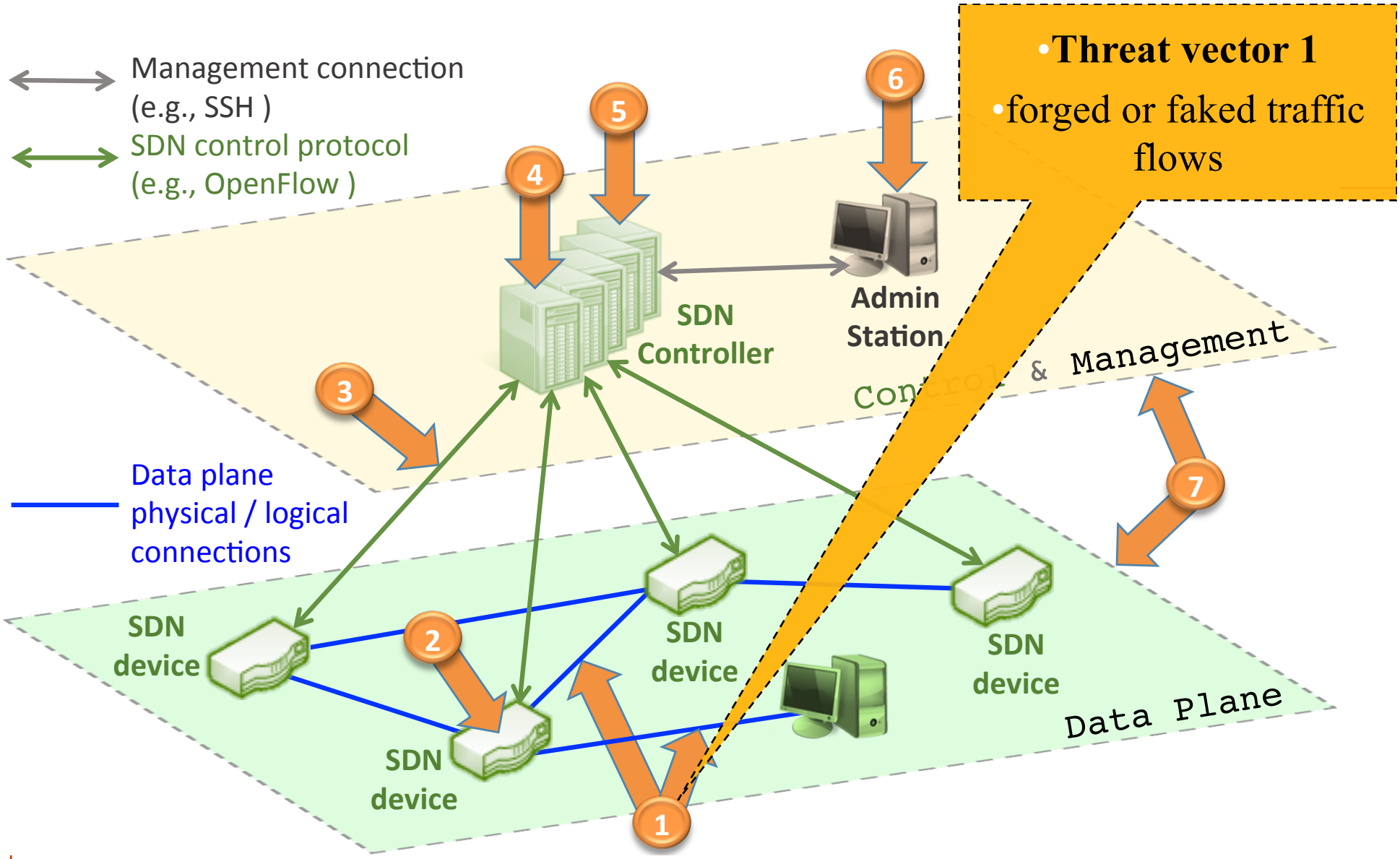
SDN Secuirty

Wait, now **others (attackers)** can program the network!



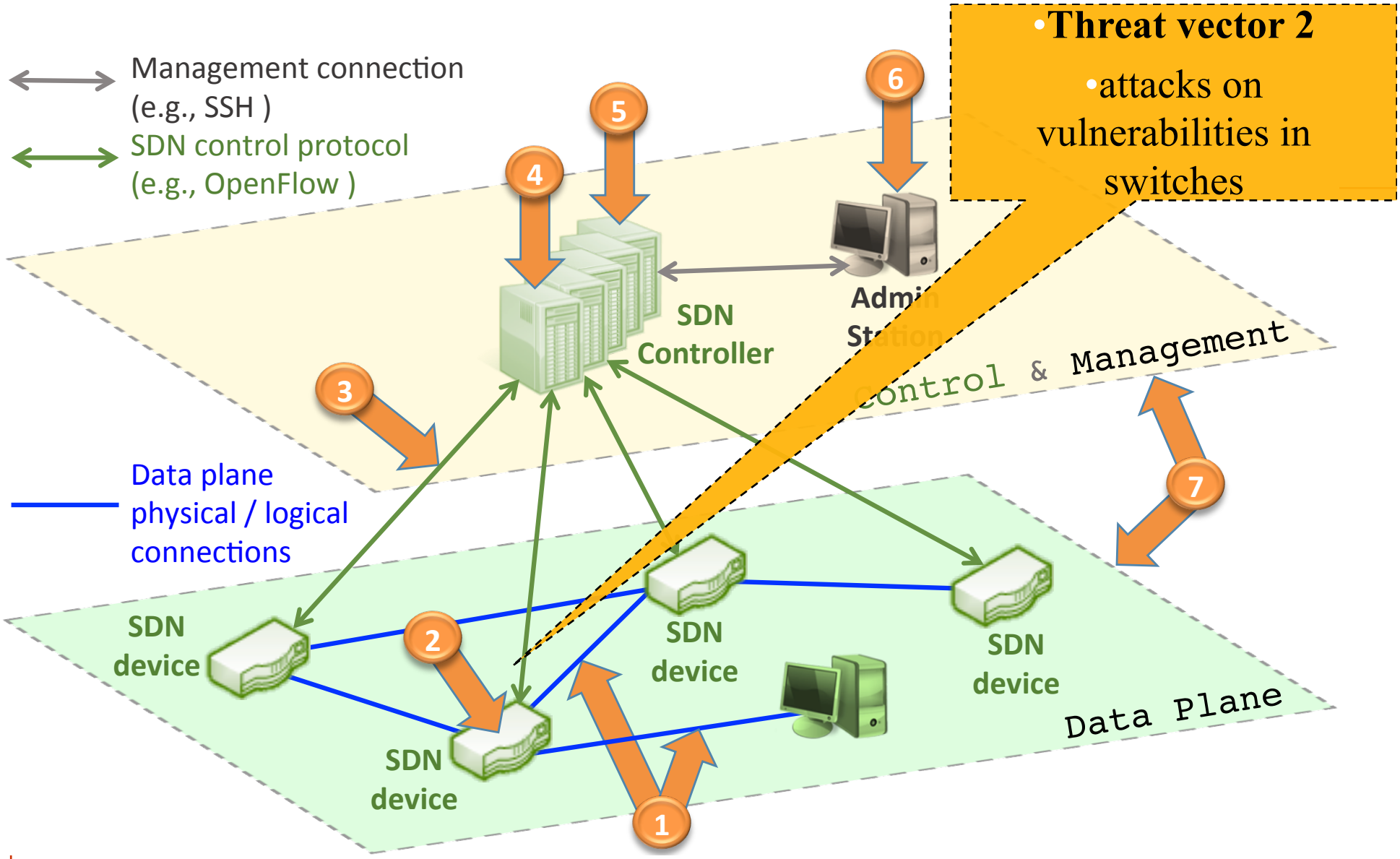


See "Towards Secure and Dependable Software-Defined Networks" (HotSDN'13)



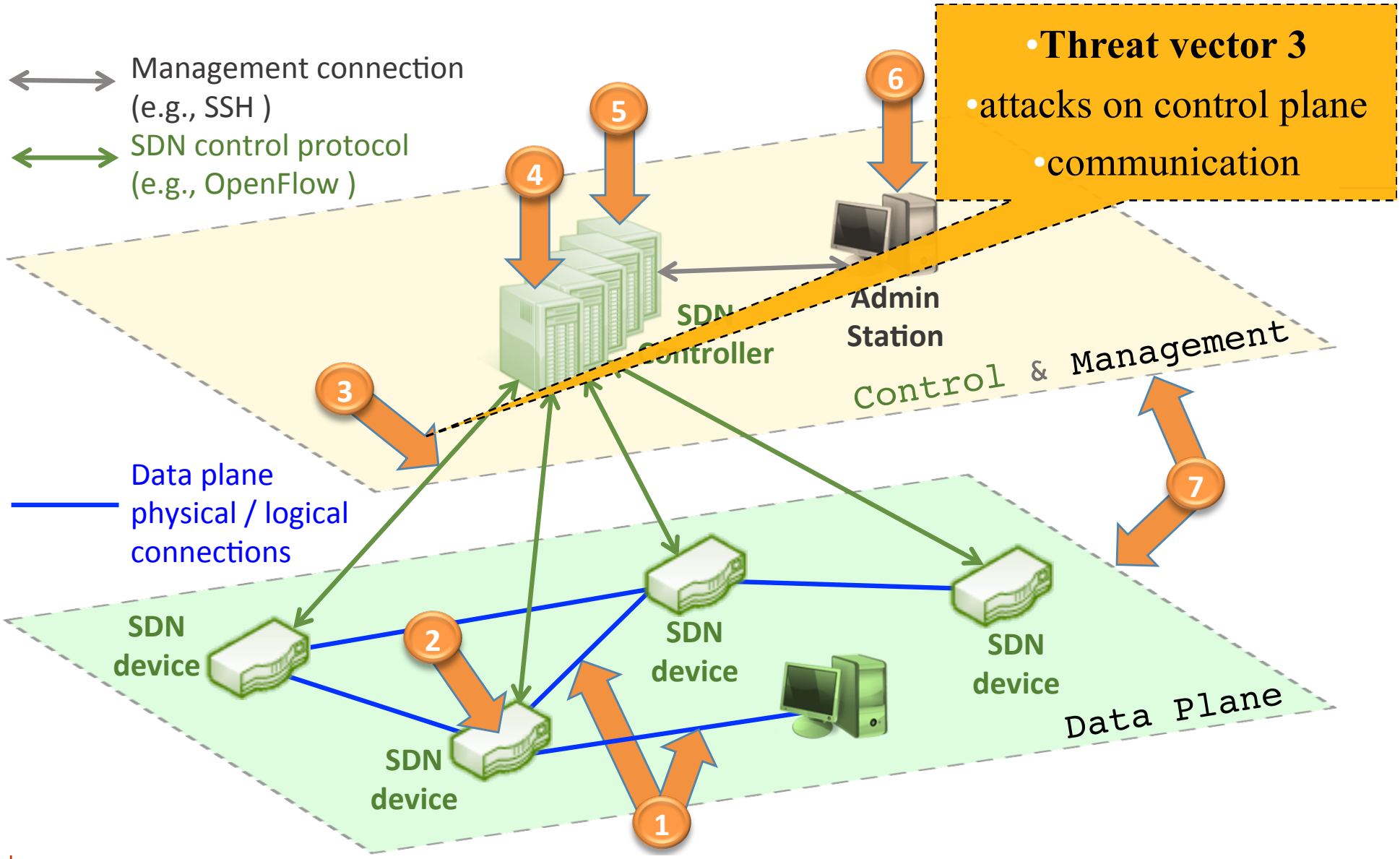
Not specific to SDNs, but can be a door for augmented DoS attacks.

•*Possible solutions:* IDS + rate bounds for control plane requests



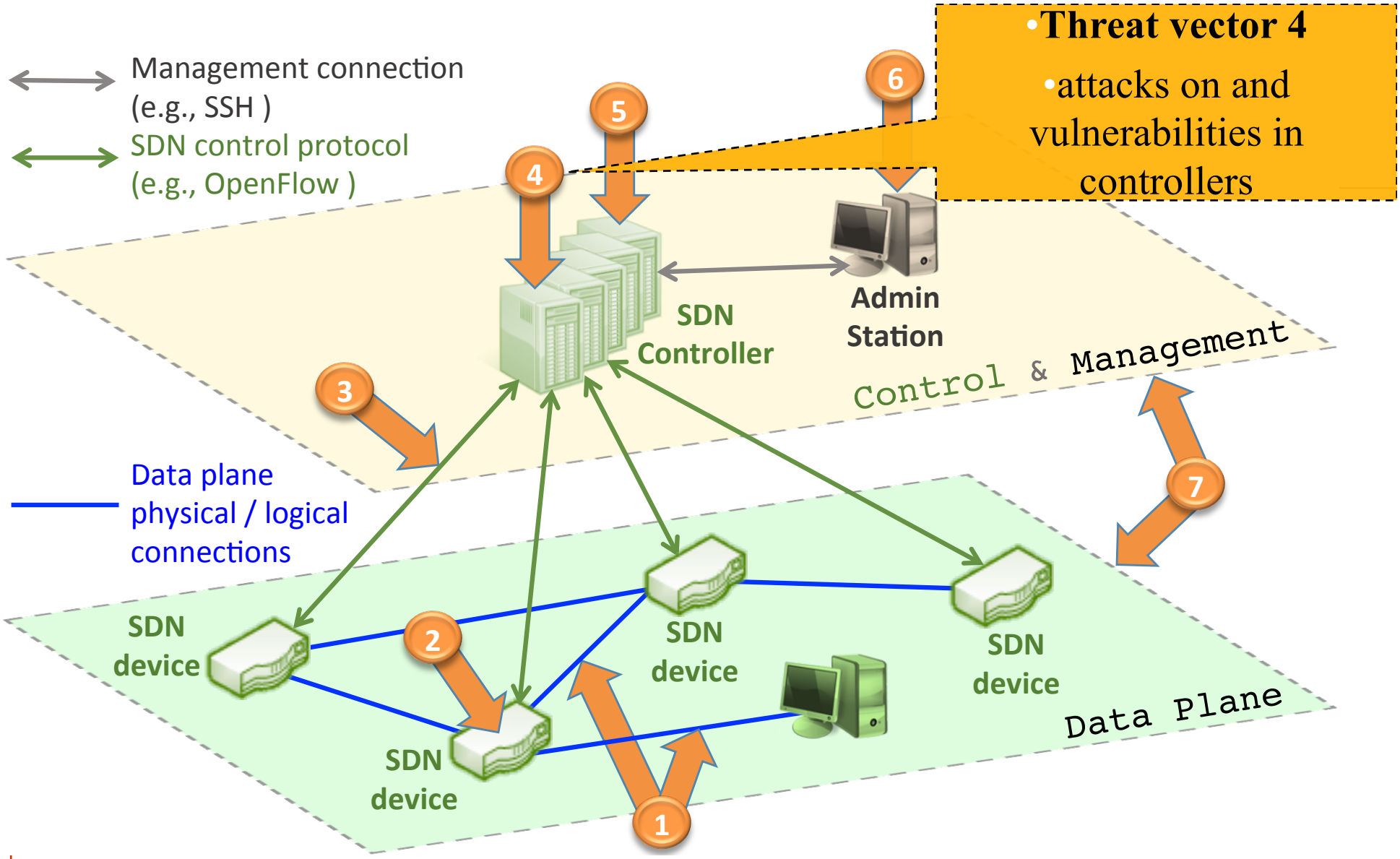
Not specific to SDNs, but now the impact is potentially augmented.

•*Possible solutions:* sw/hw attestation with autonomic trust management



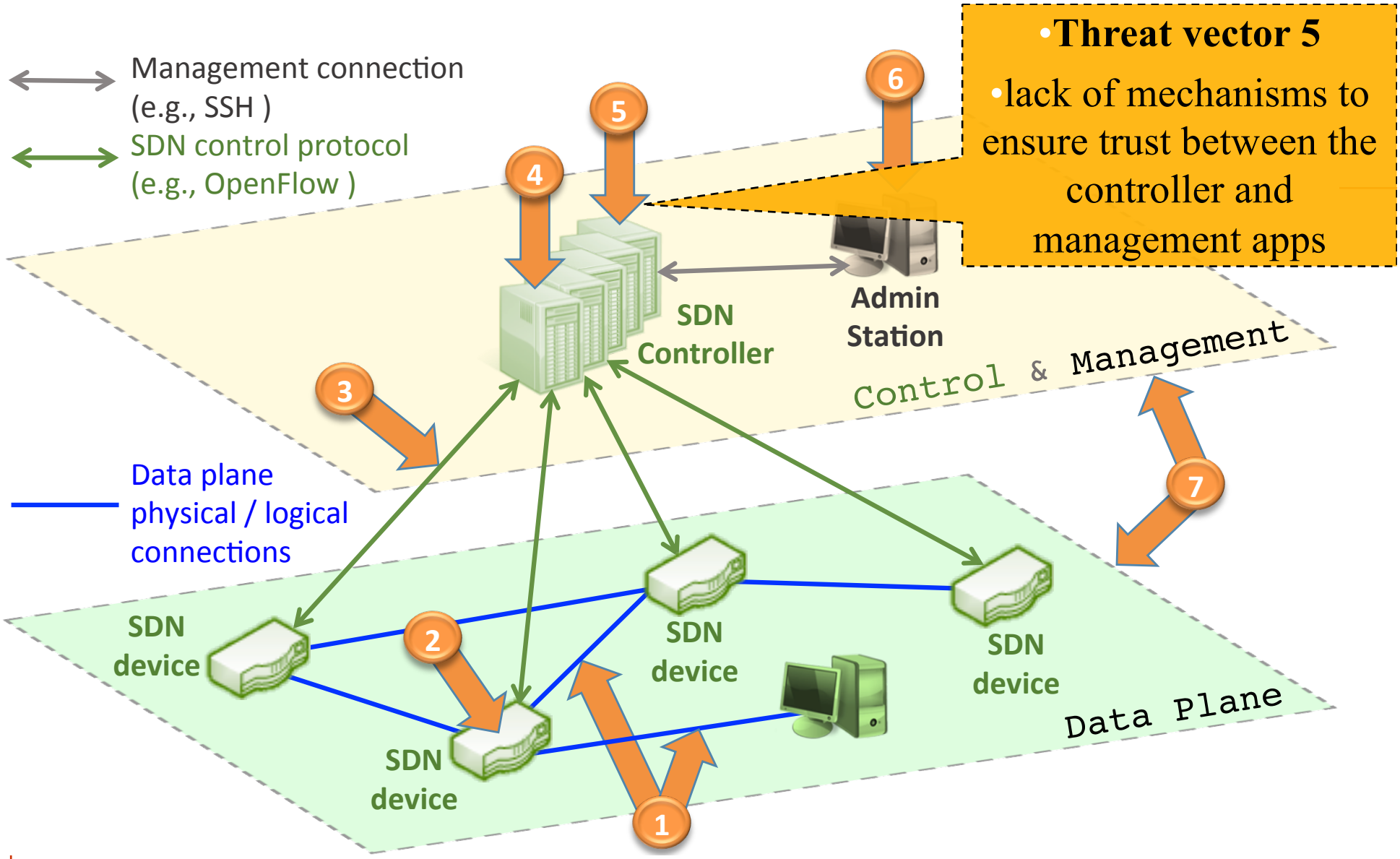
Specific to SDNs: communication with logically centralized controllers can be exploited.

•*Possible solutions:* threshold cryptography across controller replicas



Specific to SDNs, controlling the controller may compromise the entire network.

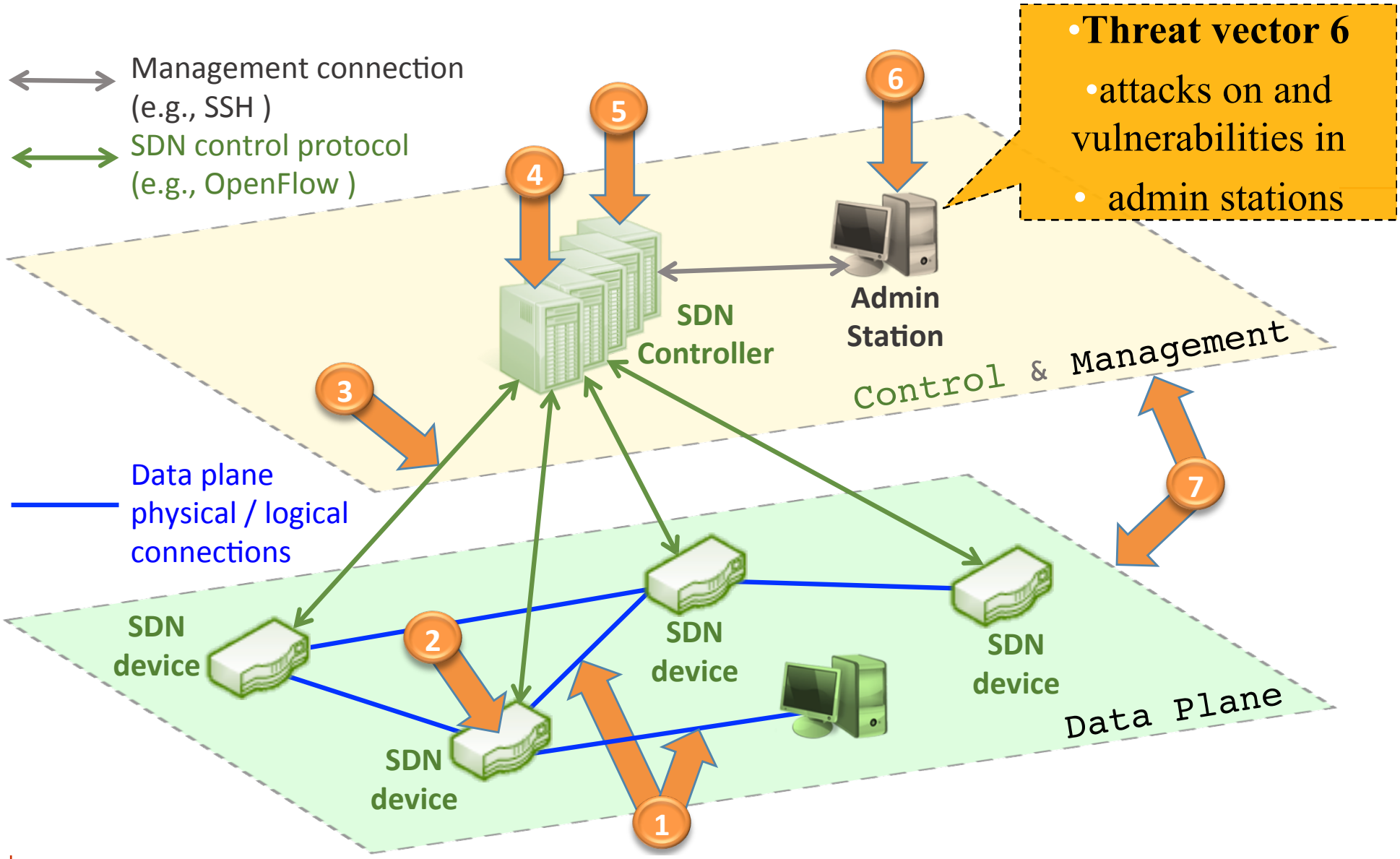
•*Possible solutions:* replication + diversity + recovery



- **Threat vector 5**
- lack of mechanisms to ensure trust between the controller and management apps

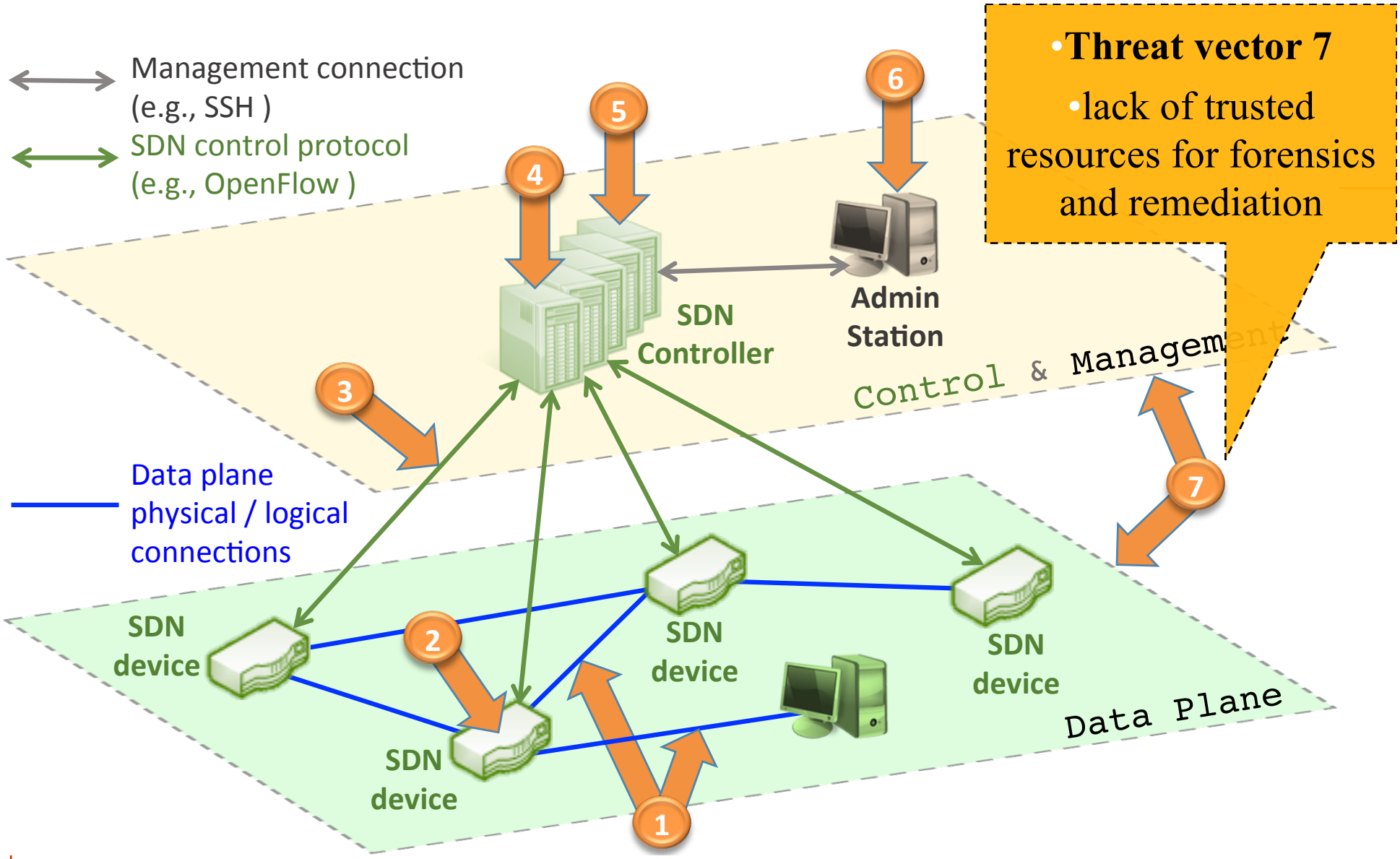
Specific to SDNs, malicious applications can now be easily developed and deployed on controllers.

- *Possible solutions:* sw attestation with autonomic trust management



Not specific to SDNs, but now the impact is potentially augmented.

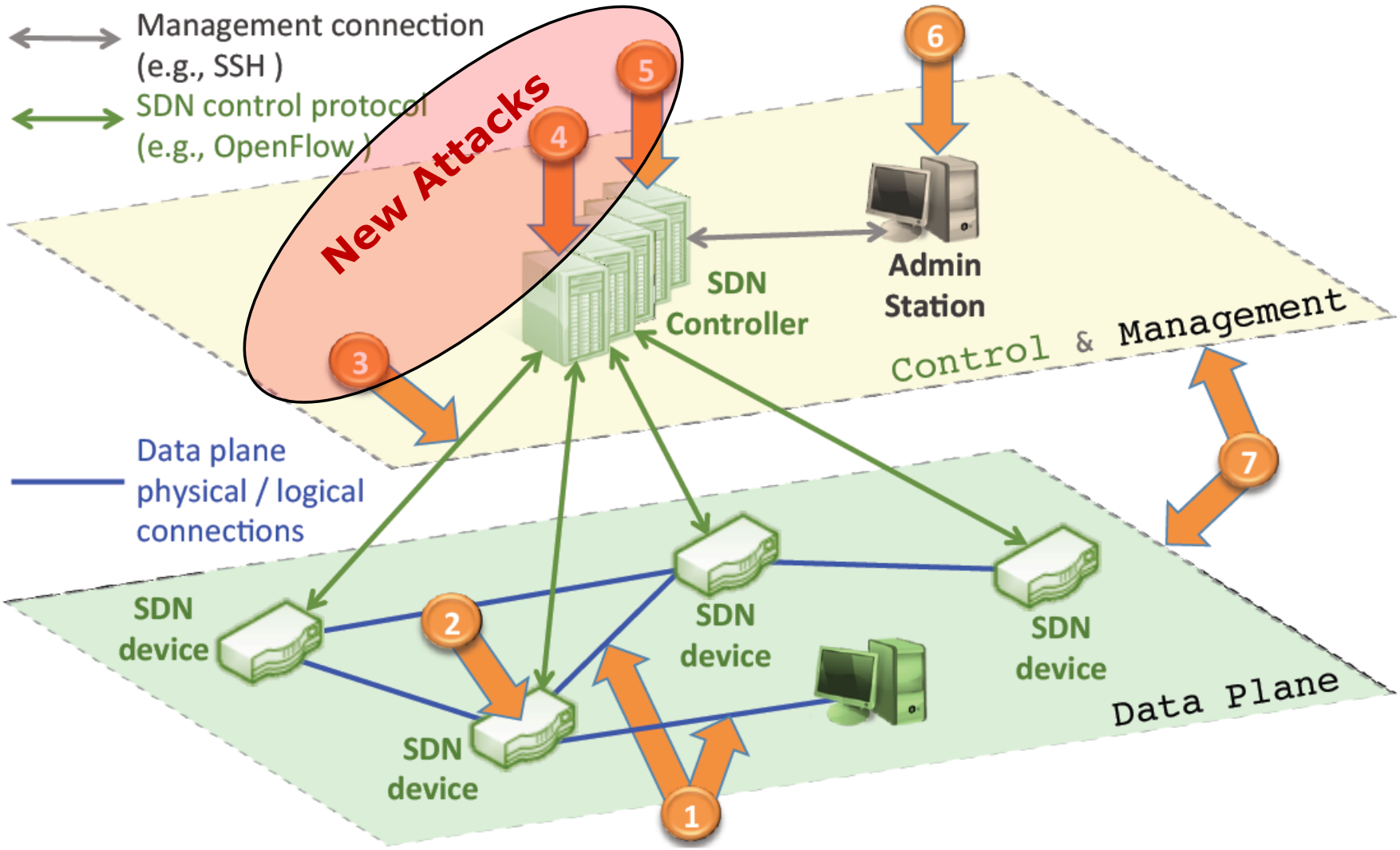
•*Possible solutions:* double credential verification



Not specific to SDNs, but it is still critical to assure fast recovery and diagnosis when faults happen.

•*Possible solutions:* indelible logging

Main Threat Vectors to SDNs



See "Towards Secure and Dependable Software-Defined Networks" (HotSDN'13)

SDN/NFV Security

- Security in SDN/NFV from My Research
  - Enabling Dynamic Access Control for Controller Applications in Software-Defined Networks
  - Fast address hopping at the switches: Securing access for packet forwarding in SDN
  - Machine-learning based Threat-aware System in Software Defined Networks
  - Dynamic Defense Provision via Network Functions Virtualization

Motivation

- The previous literatures
  - have either addressed specific attacks or proposed specific defense methods
  - have not addressed the fundamental need to detect and control malicious or suspicious traffic.
- Many undiscovered vulnerabilities in the SDN controller still remain
- It is essential to create a comprehensive security design that can defend against the various vulnerabilities in the SDN in order to ward off the wide range of potential attacks.

System Design

- Machine-learning based threat-aware system against network intrusion in SDN
  - It leverages the programmability of SDN with network intrusion detection and response system by using machine-learning techniques.

System Design

- It consists of three subsystems:
  - Data preprocessing,
  - Predictive data modeling
  - Decision making and response.

System Design

- Data Processing Subsystem
  - filters out redundant or irrelevant traffic from raw traffic data.
  - reduces the dimensionality of data to provide valuable information for the next step.
  - handles both historical archival traffic and real-time incoming traffic for further processing.

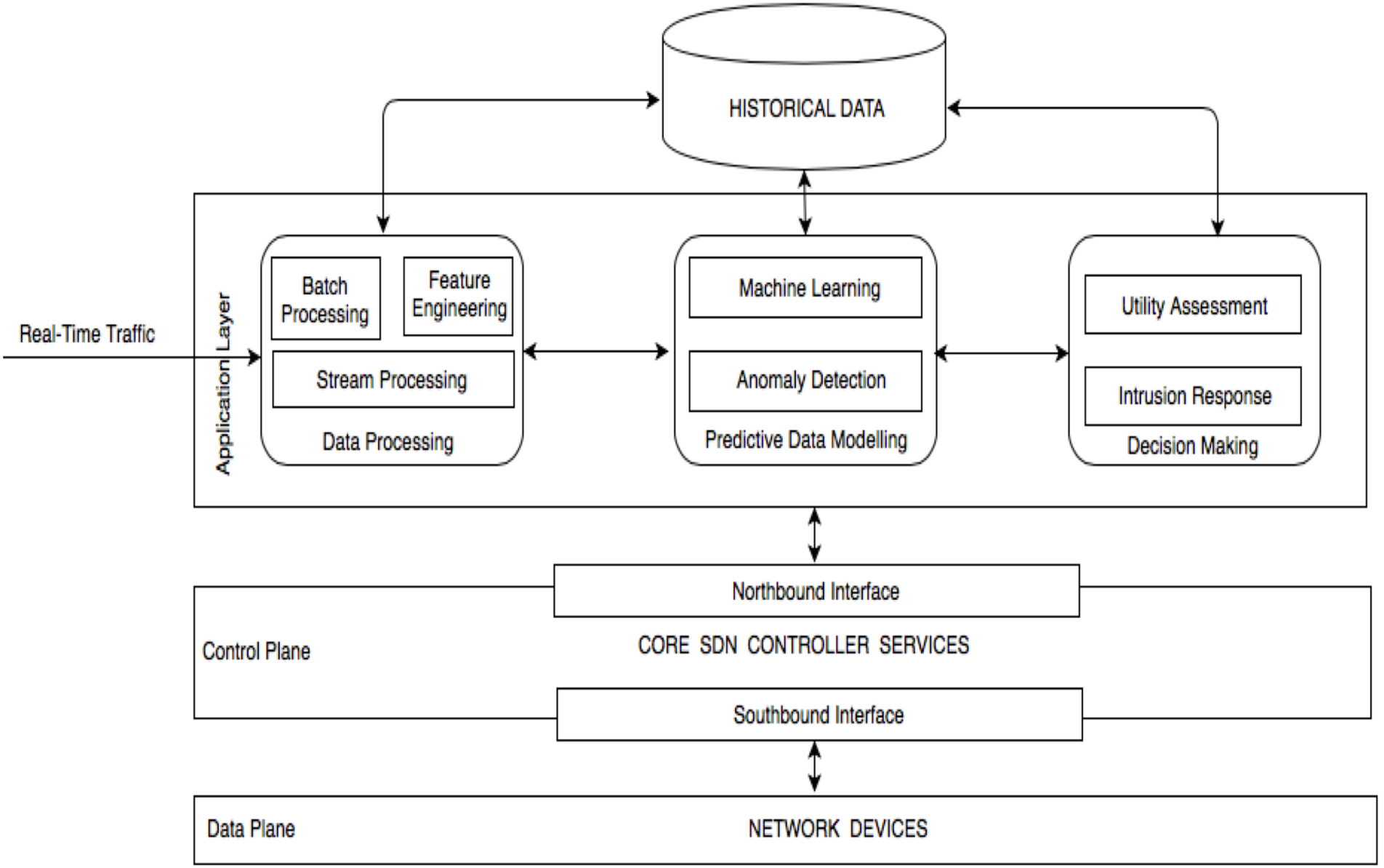
System Design

- Data Modeling
  - gathers sufficient normal and abnormal audit data.
  - applies a classification algorithm to train a classifier to label or predict new unseen audit data as belonging to the normal class or the abnormal class.
  - identifies malicious traffic in real time with negligible overhead.

System Design

- Decision-making and Response Subsystem
  - reduces uncertainty in primary classification through an active learning process
  - helps the SDN controller to respond to analysis results by using reactive routing in SDN.

System Architecture



System Architecture

- Data Preprocessing
  - Batch processing for archival historical data
  - Stream processing for real-time incoming traffic
  - Feature engineering for selecting appropriate feature sets
    - Forward feature selection strategy consists of selecting a subset of features from the training set that best predicts the test data by sequentially selecting features until there is no improvement in prediction.

System Design

- Predictive Data Modeling
  - intrusion detection is viewed as a classification problem differentiating audit events as belonging to either normal or abnormal classes
  - utilizes a machine-learning algorithm:
    - Random Forest

System Architecture

- Decision-making Process
  - chooses as “ambiguous” datasets the most uncertain ones.
  - re-evaluates them with feature sets that are ranked according to their information gains.
  - The goal of the algorithm is to learn to choose follow-up observations that are considered most interesting

System Architecture

- "ambiguous" datasets
  - We define "ambiguous" datasets as those receiving mixed decision votes in the multiple classification system
  - Most misclassified datasets belong to these ambiguous datasets.
  - The ambiguous data set simply queries the instance whose posterior probability of being positive is nearest 0.5.

System Architecture

- In the Random Forest classification,
  - ambiguous datasets consist of the datasets that receive half positive and half negative votes.
  - "positive score"

$$P^{+} = \frac{Number\ of\ Positive\ Votes}{Total\ Number\ of\ Votes}$$

  - (cf) for evaluation
    - Positive" ($0.65 < P^{+}$), "Negative" ($P^{+} < 0.35$) and "Ambiguous," ($0.35 < P^{+} < 0.65$)

System Architecture

- Response System
  - evaluate the attack impact on the network and automatically take proper actions in time
  - Reactive Routing in SDN
    - An on-demand routing protocol that computes the routing path of incoming traffic by installing flow rules for different flow types in real time.
    - For true alerts that match developed attack models, SDN will drop the traffic according to installed *drop* flow rules
    - In the case of false alerts or undecided alerts, the system must reactively respond to incoming traffic according to flow type and network cost function.

System Architecture

- Cost Function
  - Using a cost function the SDN controller determines the best routing path to deliver the high-volume traffic from source to destination for minimal impact in network
  - Two metrics, utilization U_i and network latency D_i are used
    - The utilization is calculated by a used bandwidth of link *i* divided by the link capacity.
    - The network latency is computed by transmission rate.

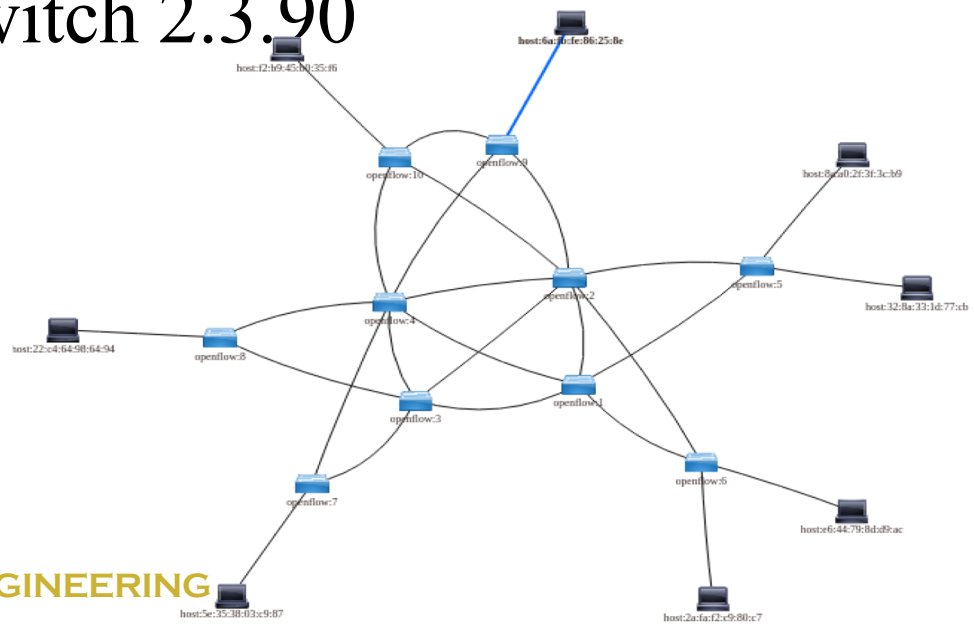
$$C_i = \sum_k c_k \times f_k(x_i^k)$$

System Implementation

- Implementation and Testing Environment
  - We implemented the proposed system by using Floodlight version 1.3, an open source SDN controller.
  - A function based on jNetPcap API is implemented in order to capture and extract the feature sets of incoming traffic.
  - The proposed framework is evaluated using the KDD99 intrusion detection dataset based on the 1998 DARPA initiative in order to generate attack models.

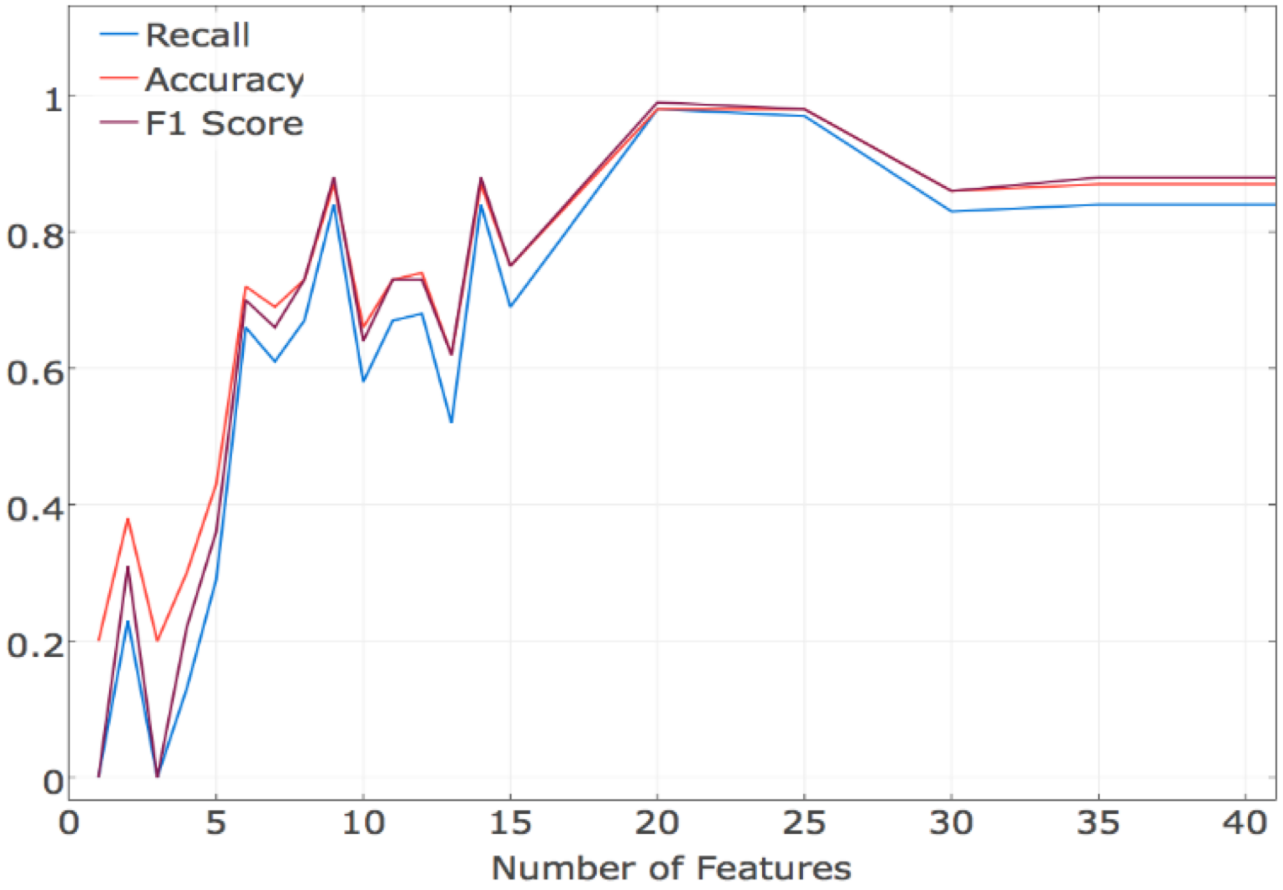
System Implementation

- Network Topology in Mininet
  - We create one large network topology in Mininet, a realistic virtual network.
  - The network topology consists of ten switches and eight hosts that are interconnected to each other.
  - Switch uses Open vSwitch 2.3.90



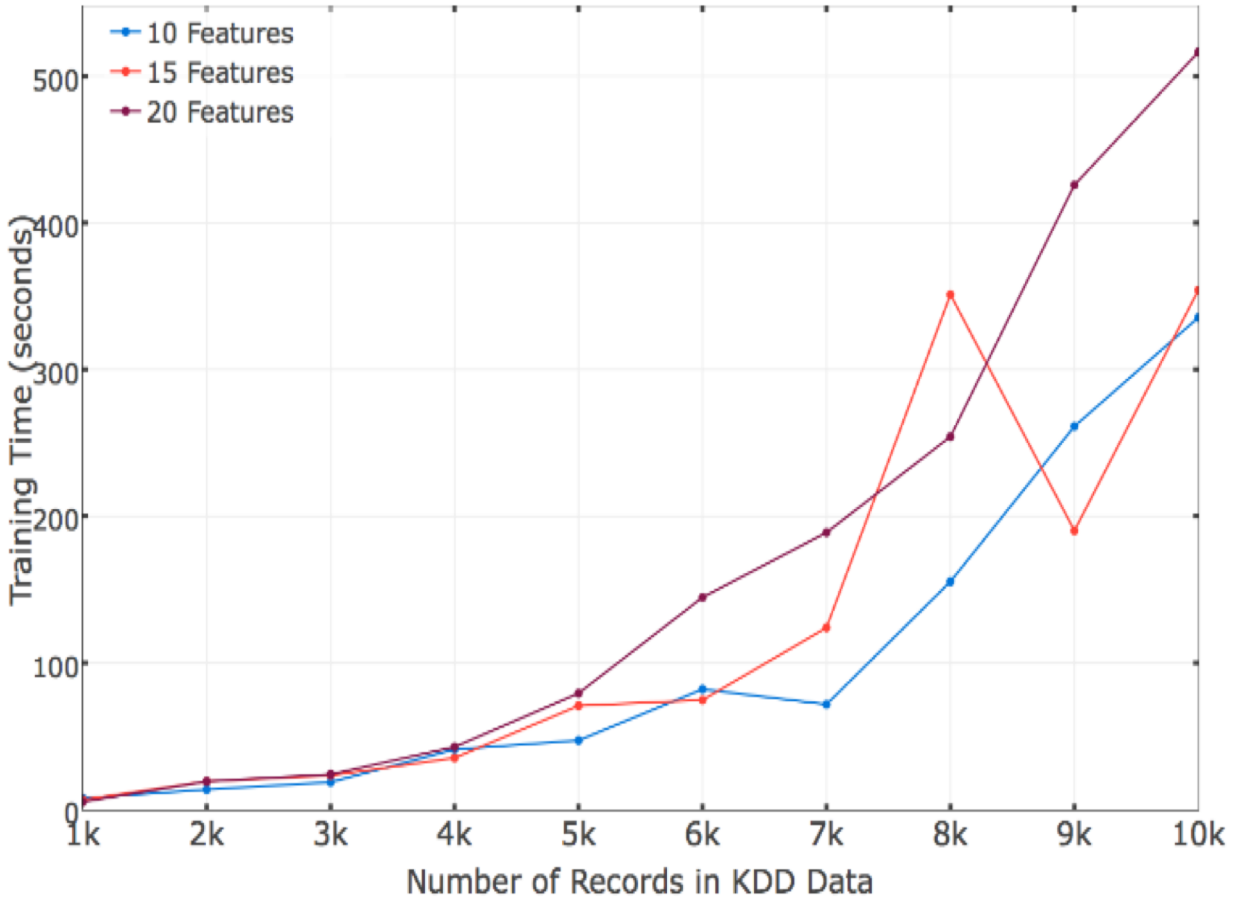
Evaluation

- Performance evaluation of detection according to the number of features



Evaluation

- Training time according to the number of feature sets



Evaluation

- Performance Evaluation
  - The performance metrics for the ambiguous dataset and the rest of the dataset excluding the ambiguous set are calculated and compared

	Recall	Precision	Accuracy	F1 Score
All Data Sets	0.98011	0.99860	0.98750	0.98927
All Data Sets Excluding Ambiguous Data	0.99030	0.99964	0.99409	0.99495
Only Ambiguous Data	0.24630	0.76879	0.45018	0.37307

Evaluation

- Performance Evaluation of the Ambiguous Dataset

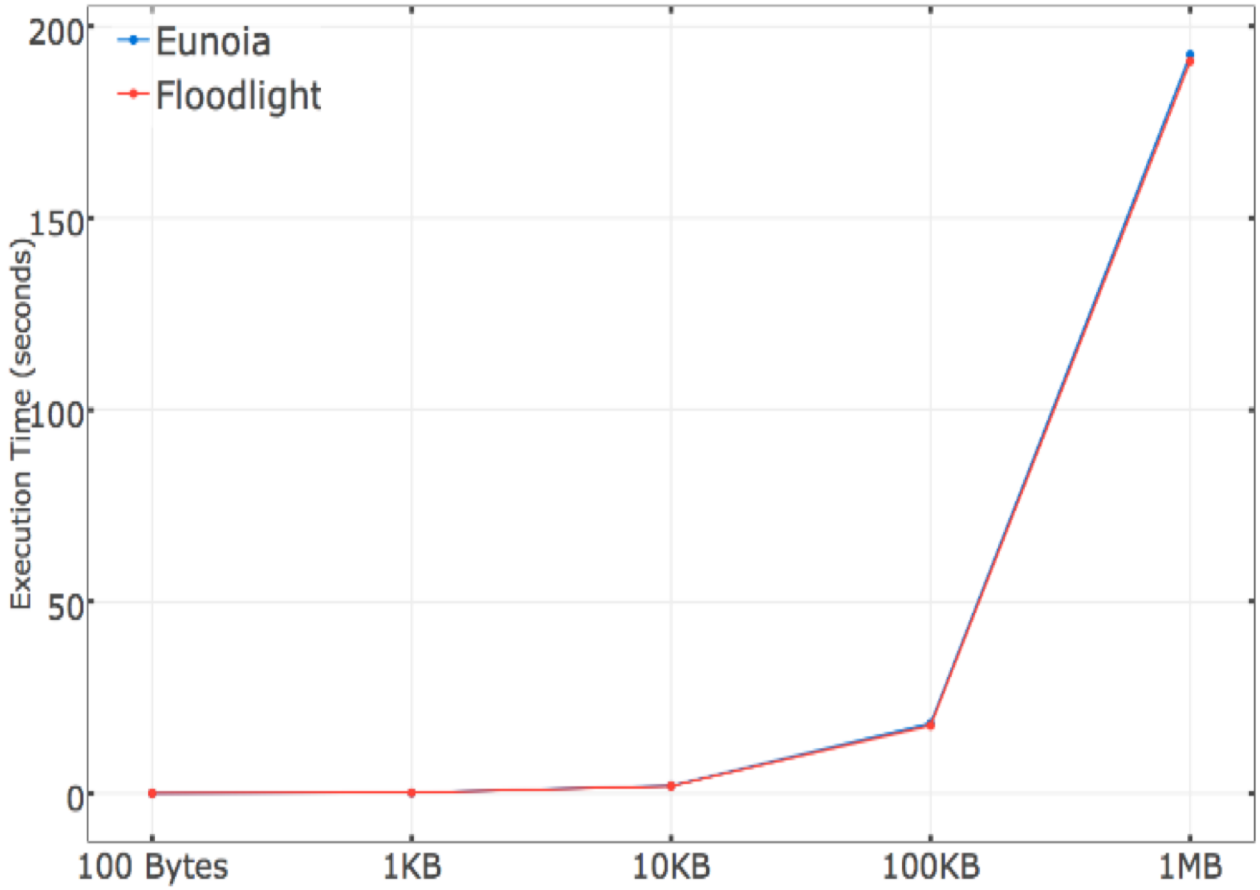
	Recall	Precision	Accuracy	F1 Score
Ambiguous Data (The result from Table II)	0.24630	0.76879	0.45018	0.37307
10 Features Used	0.88421	0.86598	0.82482	0.87500
15 Features Used	0.94681	0.92708	0.91176	0.93684

Evaluation

- Performance Evaluation of the Ambiguous Dataset
  - The detection accuracy for the ambiguous data set was improved by 83% and 103% with 10 and with 15 features, respectively.(Baseline was 45%)
  - Uncertainty in the ambiguous dataset was now resolved
    - segregating the uncertain dataset
    - calculating the important features by calculating the information gain
    - applying a simple classification to the uncertain dataset.

Evaluation

- Results of the execution time between the original Floodlight and our system



Conclusion

- We have proposed a network intrusion detection and response system in SDN using machine-learning techniques and reactive routing.
- We have proposed a method to select appropriate features to deal with decreased system performance caused by ambiguous data.
- We have also introduced an intrusion response system using reactive routing to solve problems with undecided alerts

Q & A