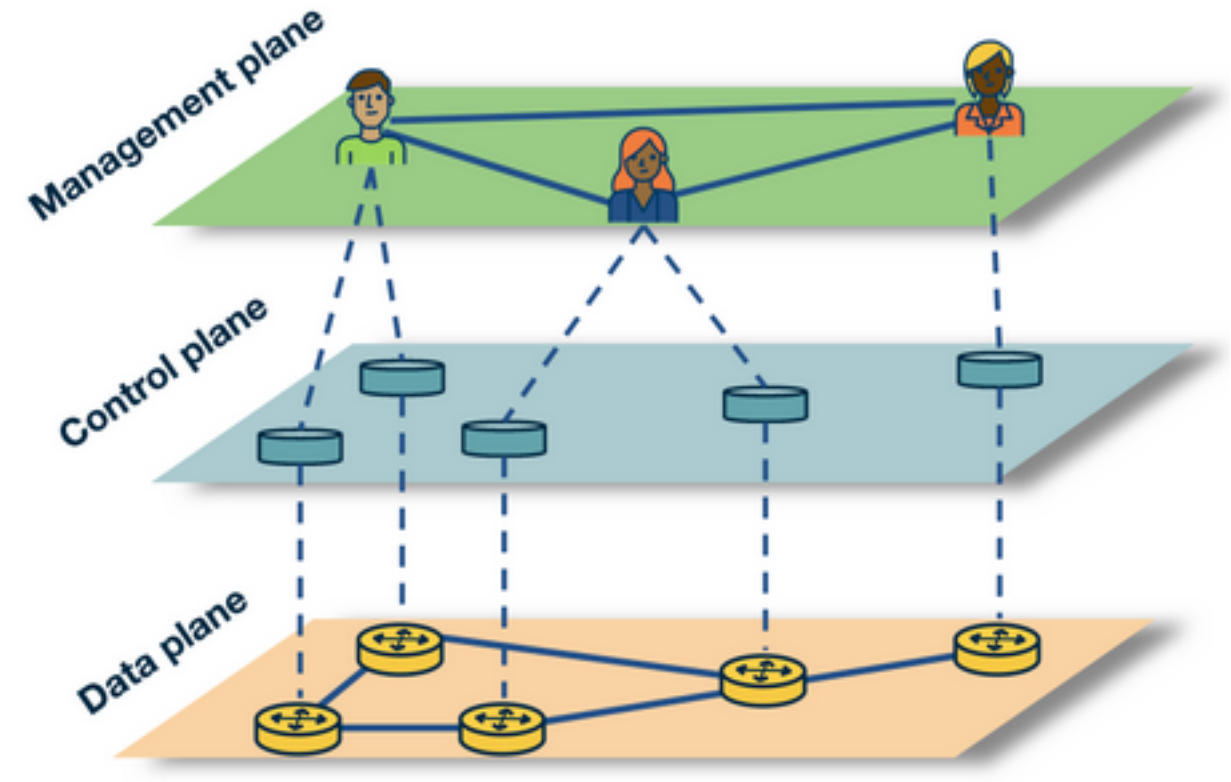
Software Defined Networking Part 2

Revisiting the Motivation for SDN

1. Challenges leading to SDN
   - Handling the ever-growing complexity and dynamic nature of networks
   - Tightly coupled architecture
2. Separation of control and data plane
   - Production-level SDNs need a physically distributed control plane to achieve performance, reliability, and scalability
   - Achieved by using a programming interface between the SDN controller and the switches
     - OpenFlow is an example of one such API
3. Layers of functionality
   - Data plane: Forward data in the form of packets or frames
   - Control plane: Determine which path to use by using protocols to populate forwarding tables of data plane elements
   - Management plane: Used to monitor and configure the control functionality, e.g. SNMP-based tools



Layered view of networking functionality

Quiz 1

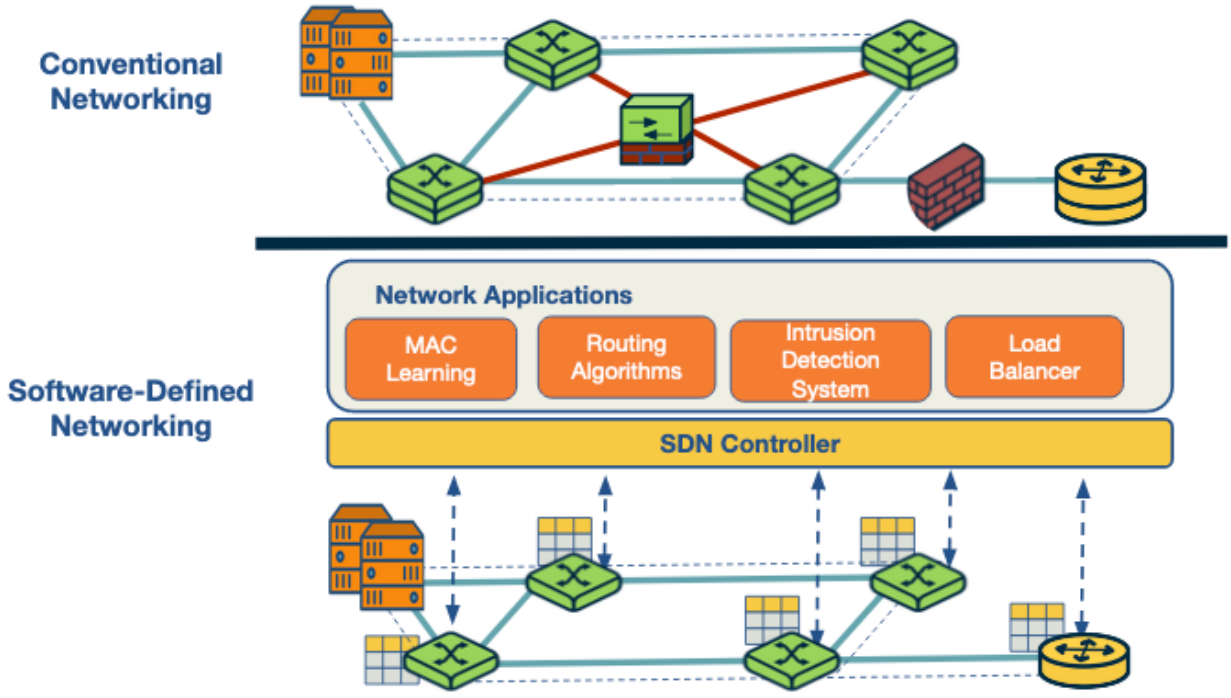
1. In a software defined networking, every device (switch, router, middlebox, etc.) must be able to make decisions in the forwarding process.
   - False
2. The transition to IPv6 would be faster with a software defined networking paradigm compared to a conventional networking paradigm.

- True

3. An OpenFlow switch may also be used for routing.
   - True
4. The management plane defines a network plocy.
5. The control plane enforces a network policy.
6. The data plane executes a network policy.

SDN Advantages

1. Conventional Networks
   - Networks had a tightly coupled data and control plane, meaning adding a new feature required going through the process of modifying all control plane devices (installing new firmware/hardware upgrades)
   - Middleboxes were used to introduce load balances, intrusion detection ssytems, firewalls, etc.
2. Advantages of Software-defined Networks
   - Shared abstractions: Middlebox services can be programmed easily
   - Consistency of same network information: Consistent policy decisions while reusing control plane modules
   - Locality of functionality placement: Middlebox applications can take actions from anywhere in the network
   - Simpler integration: Load balancing and routing applications can be combined sequentially



SDN Advantages

Quiz 2

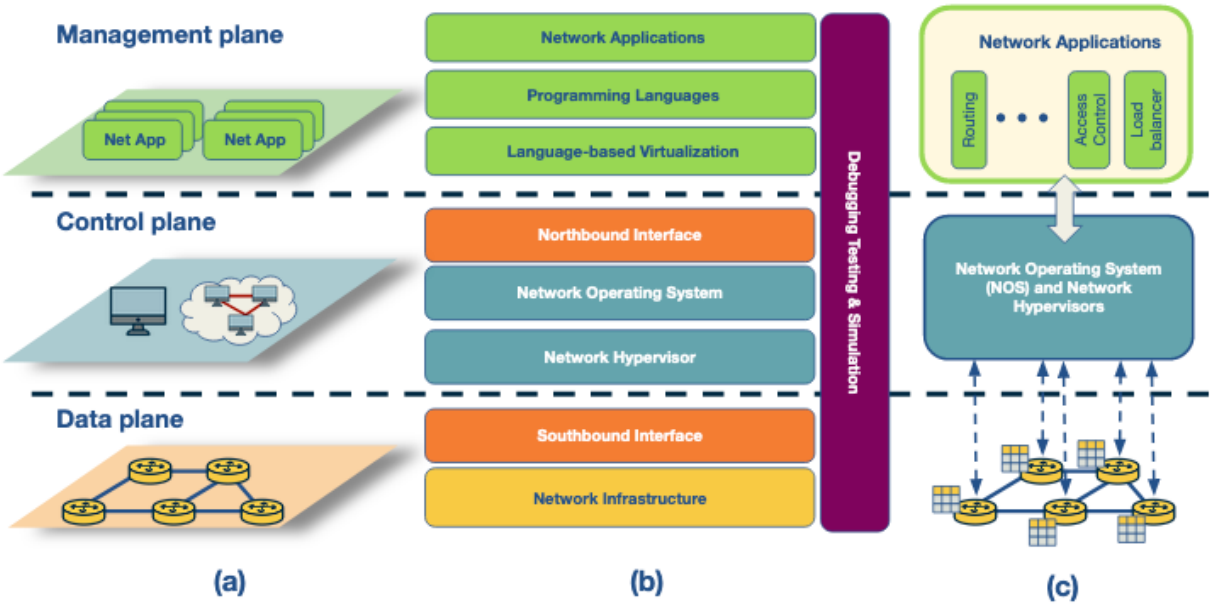
1. Load balancing is only possible with software defined networking.
   - False
2. In software-defined networking, integrations of networking applications are smoother. For example, load balancing and routing applications can be combined sequentially. Describe a scenario where routing would take precedence over load balancing, and vice versa.

- For example, in a network with high traffic volume and complex routing requirements, routing policies may take precedence over load balancing. Conversely, in a network with high availability requirements, load balancing may be a higher priority.

3. In conventional networking, which device can implement an intrusion detection system (IDS)?
   - Switches
   - Routers
   - Middleboxes (True)
   - All of the above

The SDN Landscape

1. SDN Design
   - SDN landscape is organized into three different view
     - Plane-oriented view
     - SDN layers
     - System design perspective
2. SDN Technologies
   - Infrastructure: Physical networking equipment are merely forwarding elements and any logic to operate them is directed from the centralized control system
     - OpenFlow
   - Southbound interfaces: Act as connected bridges between control and forwarding elements
     - Play a crucial role in separating control and data plane functionality
     - OpenFlow, ForCES, OVSDB, OpFlex, OpenState
   - Network virtualization: Need to provide support for arbitrary network topologies and addressing schemes, similar to the computing layer
     - VLAN, NAT, MLPS can provide full network abstractions, but are connected on a box-by-box basis and there is no unifying abstraction to configure them in a global manner
     - VxLAN, NVGRE, FlowVisor, FlowN, NVP
   - Network operating systems: Ease network management and solve networking problems by using a logically centralized controller by way of a network operating system
     - OpenDayLight, OpenContrail, Onix, Beacon, HP VAN SDN
   - Northbound interfaces: Abstraction that guarantees programming language and controller independence
     - Floodlight, Trema, NOX, Onix, and SFNet
   - Language-based virtualization: Express modularity and allow different levels of abstractions to view a single physical device in different ways
     - Pyretic, libNetVirt, AutoSlice, RadioVisor, OpenVirteX
   - Network programming languages: Network programmability can be achieved using high- or low-level programming languages
     - Low-level makes it difficult to write and reuse modular code
     - High-level provides abstractions, improves modularity, and does away with specific and low-level configurations
     - Pyretic, Frenetic, Merlin, Nettle, Procera, FML
   - Network applications: Functionalities that implement the control plane logic and translate to commands in the data plane
     - Hedera, Asterx, OSP, OpenQoS, Pronto, Plug-N-Serve, SIMPLE, FAMS, FlowSense, OpenTCP, NetGraph, FortNOX, FlowNAC, VAVE



SDN Landscape

Quiz 3

1. The northbound interfaces separate the management and control plane.
2. The southbound interfaces separate the data plane and control plane.
3. OpenFlow is used in the data plane, and it is an example of a southbound interface.

SDN Infrastructure Layer

1. SDN infrastructure is composed of networking equipment performing simple forwarding tasks
   - Physical devices do not have embedded intelligence or control
     - Relegated to NOS
   - Built on top of open and standard interfaces that ensure configuration and communication compatibility and interoperability among different control plane and data plane devices
2. SDN architecture
   - Data plane device is a hardware or software entity that forwards packets
   - Controller is a software stack running on commodity hardware
   - OpenFlow flow table entry
     - Matching rule
     - Actions to be executed on matching packets
     - Counters that keep statistics of matching packets
   - OpenFlow device possible actions
     - Forward the packet to outgoing port
     - Encapsulate the packet and forward it to the controller
     - Drop the packet
     - Send the packet to normal processing pipeline
     - Send the packet to next flow table

SDN Southbound Interfaces

1. Southbound interfaces overview
   - APIs separating medium between the control plane and data plane functionality

- OpenFlow standards promote interoperability and deployment of vendor-agnostic devices

2. OpenFlow
   - Most widely accepted southbound standard for SDNs
     - Event-based messages that are sent by forwarding devices to controller when there is a link or port change
     - Flow statistics are generated by forwarding devices and collected by controller
     - Packet messages are sent by forwarding devices to controller when they do not know what to do with a new incoming flow
   - OpenFlow is most accepted, but ForCES and OVSDB are alternatives that attempt to provide a more flexible approach

Quiz 4

1. What action does an OpenFlow device take when an incoming flow does NOT match any rules in any of the flow tables in the pipeline?
   - Sends a message to the controller
2. Which type of message sent by an OpenFlow device to the network OS allows for quality of service (QoS) policies to be implemented?
   - Flow statistics
3. Which type of message would be sent by an OpenFlow device to the network OS in when it receives new routing information?
   - An event-based message

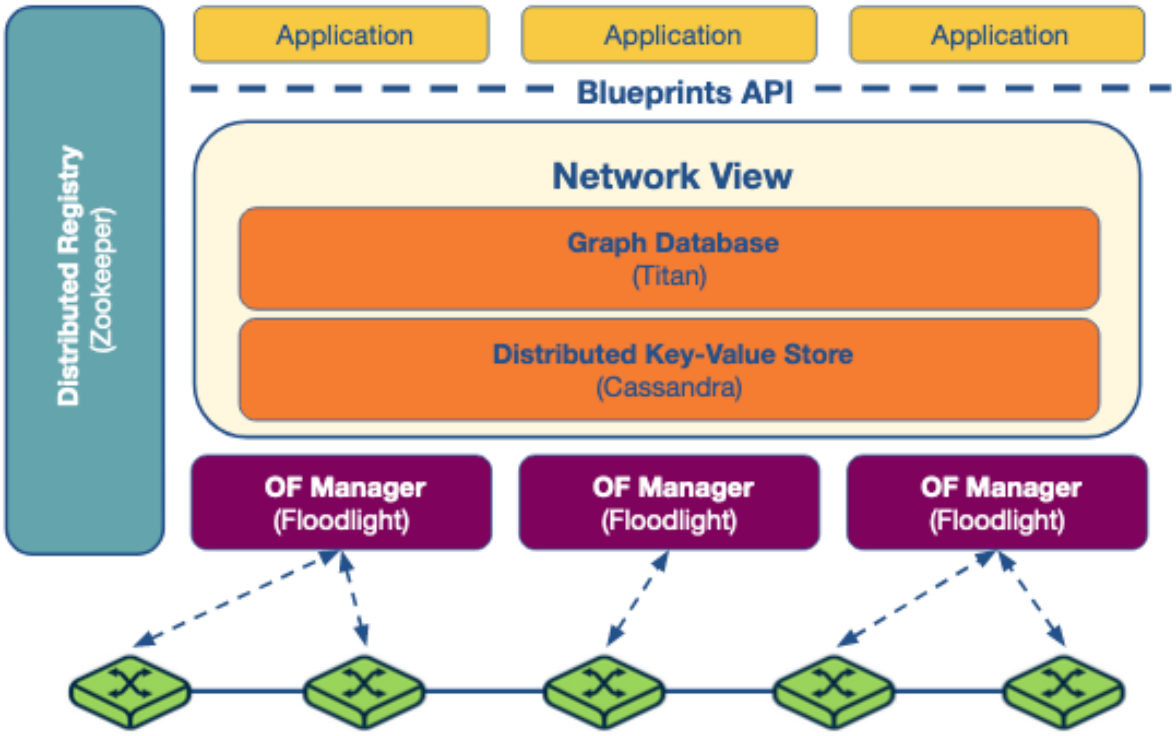
SDN Controllers: Centralized vs Distributed

1. Overview
   - Controller is a critical element in an SDN architecture as it is the key supporting piece for control logic to generate network configuration based on the policies defined by the network operator
2. Core Controller Functions
   - Topology, statistics, notifications, device management, shortest path forwarding, security mechanisms
     - High priority rules should always take precedence over low priority
3. Centralized controllers
   - Single entity that manages all forwarding devices in the network, which is a single point of failure and may have scaling issues
     - May not be enough to handle a large number of data plane elements
4. Distributed controllers
   - Can be scaled to meet the requirements of potentially any environment (small or large networks) unlike a centralized controller
   - Can occur in two ways
     - Centralized cluster of nodes
     - Physically distributed set of elements
   - Properties of distributed controllers
     - Weak consistency semantics
     - Fault tolerance

An Example Controller: ONOS

1. Open Networking Operating System
   - Distributed SDN control platform
   - Aims to provide a global view of the network to the applications, scale-out performance, and fault tolerance
   - Several ONOS instances running in a cluster
     - Management and sharing of network state is achieved by maintaining a global network view

- To make forwarding and policy decisions, applications consume information from the view and then update these decisions back to the view
  - OpenFlow managers receive changes and appropriate switches are programmed
  - View is implemented with Titan, a graph database, and Cassandra, a distributed key-value store
- Distributed architecture offers scale-out performance and fault tolerance
  - Propagation of state changes between switch and network view is handled solely by the master instance of that switch
  - Workload can be distributed by adding more instances to the ONOS cluster
- ONOS acheives fault tolerance by redistributing work of a failed instance to other remaining instances
  - Each switch connects to multiple ONOS instances with only one instance acting as its master
  - Each ONOS instance acts as a master for a subset of switches
  - Consensus election to choose a new master when one fails
- Zookeeper is used to maintain mastership between the switch and controller



ONOS

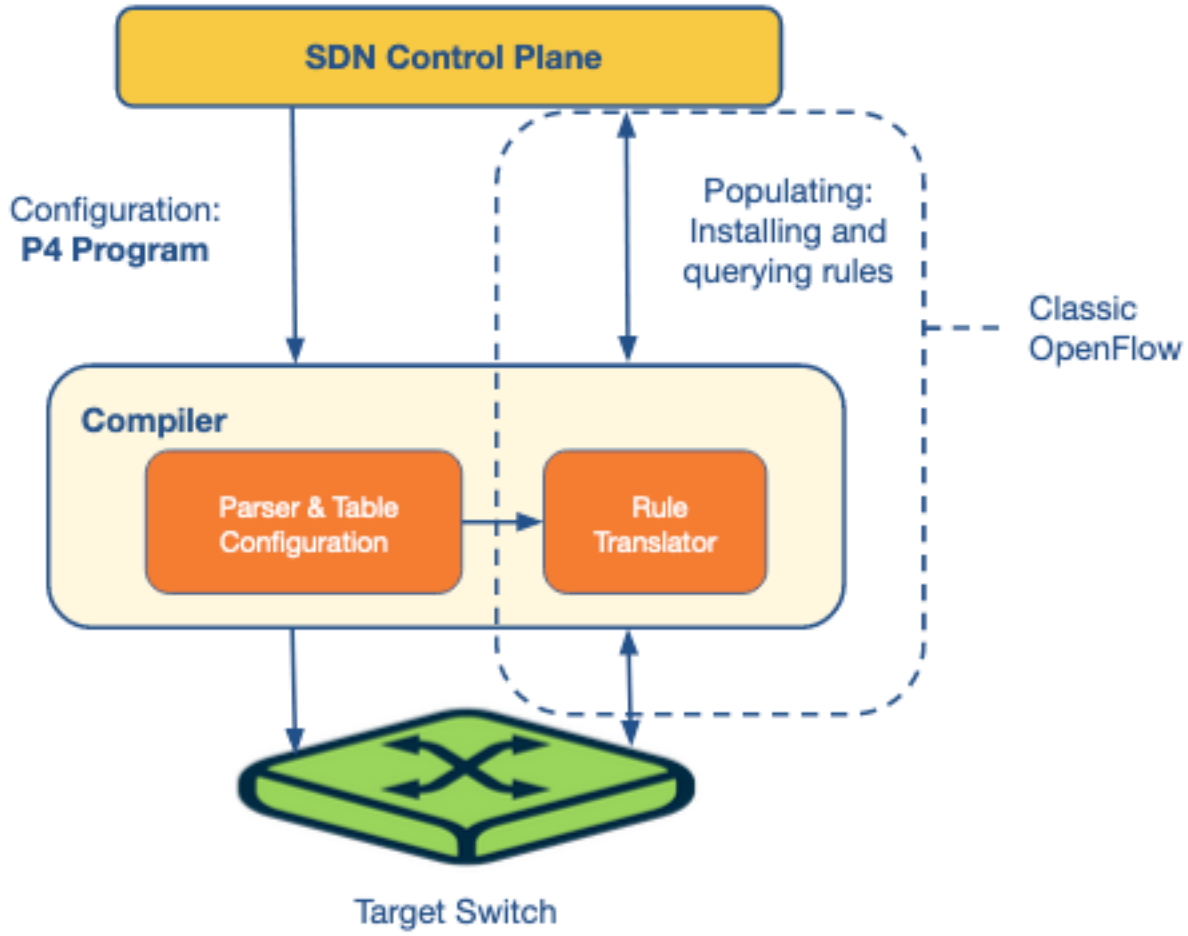
Quiz 5

1. A network controller prioritizes the rules generated by various services.
   - True
2. Which architecture provides the best throughput?
   - A distributed controller with a centralized cluster of nodes
3. Which architecture provides the highest level of fault tolerance?
   - A distributed controller with a physically distributed set of elements
4. Which architecture has the strongest consistency semantics?
   - A centralized controller
5. How does ONOS handle faults?

- If an ONOS instance fails, the other instances elect a new master for each of the switches that were previously controlled by the failed instance.

Programming the Data Plane: The Motivation

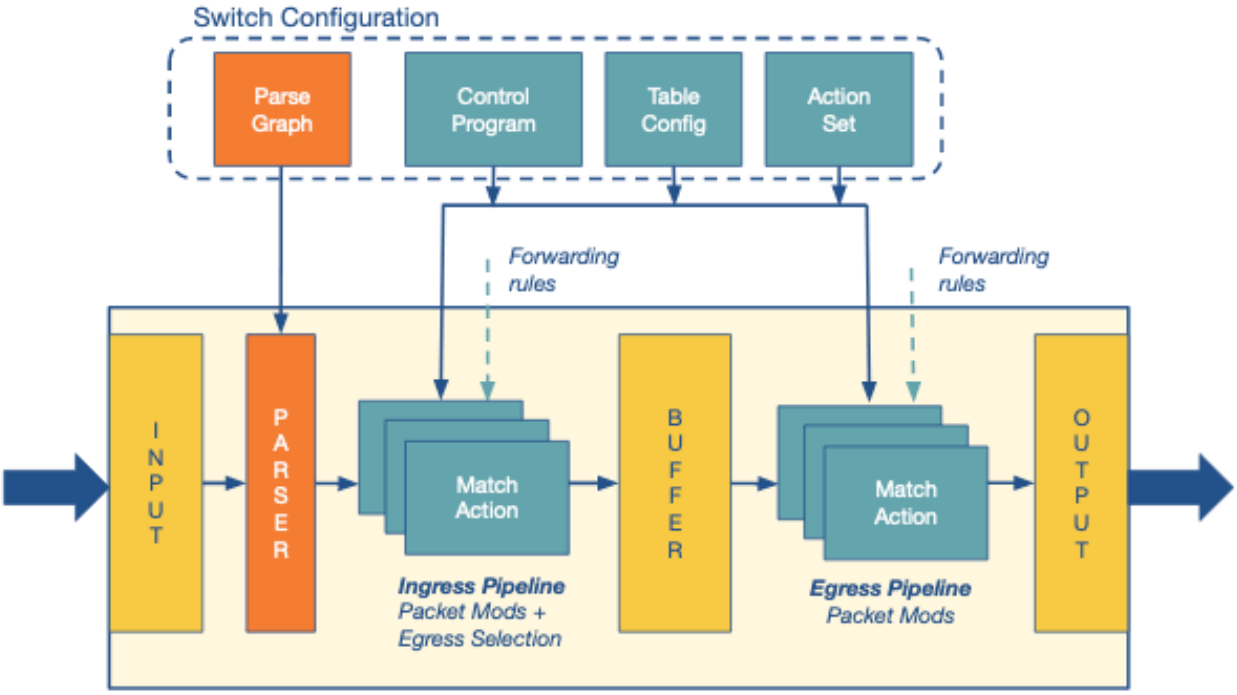
1. P4: Programming Protocol-Independent Packet Processors
   - High-level programming language to configure switches which works in conjunction with SDN control protocols
   - OpenFlow includes multiple stages of rule tables with increasing number of header fields to allow better exposure to a switch's functionalities to the controller
   - Need for an extensible, flexible approach to parse packets and match header fields while also exposing an open interface to the controllers to leverage these capabilities
   - P4 is used to configure the switch programmatically and acts as a general interface between the switches and the controller with its main aim of allowing the controller to define how the switches operate
2. Primary Goals of P4
   - Reconfigurability: Controller should be able to modify how parsing and processing of packets takes place in the switches
   - Protocol independence: Switches should not be tied to a single protocol
   - Target independence: Packet processing programs should be programmed independent of the underlying target devices



P4

Programming the Data Plane: P4's Forwarding Model

1. P4 Design
   - Programmable parser and set of match+action tables to forward packets
     - Tables can be accessed in multiple stages in series or parallel
   - OpenFlow only supports fixed parsers based on predetermined header fields and only a series combination of match+action tables
   - P4 allows for generationalization of packet processing across various forwarding devices such as routers, load balancers, etc. using multiple technologies such as fixed function switches, NPUs, etc.
2. Operations of the P4 Forwarding model
   - Configure: Used to program the parser
     - Specify header fields to be processed in each match+action stage and define the order of the stages
   - Populate: Entries in the match+action tabels specified during configuration may be altered using the populate operations
     - Allows addition and deletion of the entries in the tables



P4 Abstract Forwarding Model

An Introduction to the P4 Programming Language

1. Characteristics of P4
   - Legal header types are declard to let the parser be aware of the possible packet formats
   - Control flow program uses the declared header types and a set of actions to specify how the headers are processed
   - Table Dependency Graphs (TDGs) are used to identify dependencies between the header fields and help determine the order in which the tables can be executed
     - Tables with no dependencies can be executed in parallel

Quiz 6

1. The P4 programming language can also be used with a conventional network paradigm.
   - False
2. The P4 language is being developed as a replacement for OpenFlow.
   - False
3. The P4 language allows programmers to use multiple header fields to parse, match, and perform actions on packets.
   - True
4. The P4 language is used to program the data plane.
5. A multiport switch and a SmartNIC are two devices that can be programmed using P4. This is possible to which of the three primary goals of the language?
   - Target independence
6. The forwarding model used by P4 is a pipeline.
   - True
7. The match+action tables in P4 are more flexible than those in current version of OpenFlow.
   - True

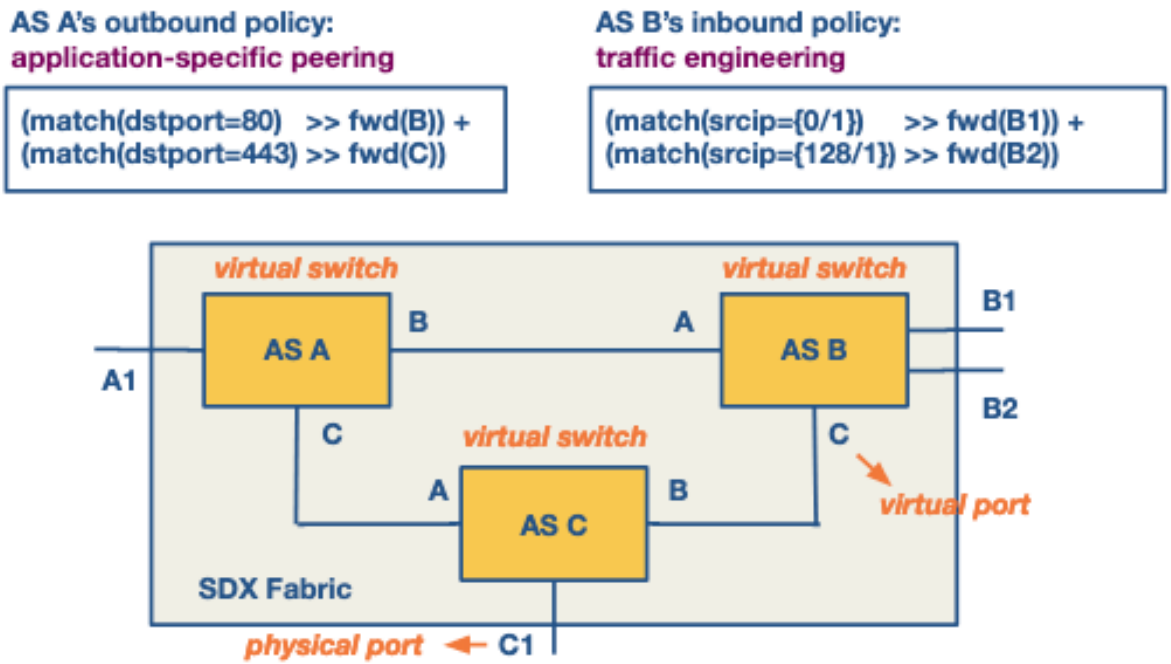
8. What are the two operations in the P4 forwarding model?
    - Configure, Populate

SDN Applications: Overview

1. Traffic Engineering
    - Optimize traffic flow to minimize power consumption, judiciously use network resources, perform load balancing, etc.
    - ElasticTree, Plug-n-Serve, Asterx, ALTO VPN
2. Mobility and Wireless
    - Existing wireless network face various challenges in its control plane including management of the limited spectrum, allocation of radio resources, and load balancing
    - OpenRadio is like OpenFlow for wireless
3. Measurement and Monitoring
    - First class of applications aims to add features to other networking services
    - Second class aims to improve existing features of SDNs using OpenFlow such as reducing the load on the control plane arising from collection of data plane statistics using various sampling and estimation techniques
    - Monitor traffic to allow system to respond to changes
4. Security and Dependability
    - Use SDN to impose security policies on the entry point to the network
    - Use programmable devices to enforce security policies on a wider network
    - Detect anomalies in traffic
        - DDoS detection
5. Data Center Networking
    - Use SDN to offer services such as live migration of networks, troubleshooting, real-time monitoring of networks among various other features
    - Detect anomalous behavior in data centers by defining different models and building application signatures from observing the information collected from network devices in the data center
    - Perform dynamic reconfigurations of virtual networks involved in a live virtual network migration

SDN Application Example: A Software Defined Internet Exchange

1. Limitations of BGP
    - Routing only on destination IP prefix
    - Networks have little control over end-to-end paths
2. Use SDN to address BGP Limitations
    - Perform multiple actions on traffic by matching over various header fields
3. SDX: SDN-based Architecture for IXPs
    - Application specific peering
    - Traffic engineering
    - Traffic load balancing
    - Traffic redirection through middleboxes
4. SDX Architecture
    - Each AS has the illusion of its own virtual SDN switch that connects its border router to every other participant AS
    - Each AS can define forwarding policies as if it is the only participant at the SDX, without influencing how other participants forward packets on their own virtual switches
    - Each AS can have its own SDN applications for dropping, modifying, or forwarding their traffic



SDX Application Example

SDN Applications: Wide Area Traffic Delivery

1. Application-specific Peering
   - ISPs prefer dedicated ASes to handle high volumes of traffic flowing from high bandwidth applications such as YouTube and Netflix
   - Use packet classifiers to direct traffic in a different path
2. Inbound Traffic Engineering
   - SDN enabled switch can be installed with forwarding rules based on the source port and IP address, enabling an AS to control how traffic enters its network
   - BGP performs routing based solely ont he destination address of the packet
3. Wide-area Server Load Balancing
   - Typical approach is for client to issue a DNS request to the service's DNS server
     - Service returns IP address of a server such that it balances the load in its system
     - Involves DNS caching which can lead to slower responses
   - More efficient approach is to modify the destination IP addresses at the exchange point to the desired backend server based on the request load
4. Redirection Through Middleboxes
   - Middleboxes are usually targeted at important junctions, such as the boundary of the enterprise networks with their upstream ISPs
   - To avoid placing many middleboxes, traffic is directed through a fixed set of middleboxes by their ISPs
   - Done by manipulating routing protocols such as internal BGP to send a subset of traffic to a middlebox
     - Can result in unnecessary traffic being redirected
   - Instead, use SDX to identify and redirect the desired traffic

Quiz 7

1. Most SDN applications can be grouped into one of the five categories: traffic engineering, mobility and wireless, measurement and monitoring, security and dependability, and data center networking. Classify each application into an appropriate category.
   - Load balancing: Traffic engineering
   - Interference management: Mobility and wireless
   - Traffic matrix estimation tool: Measurement and monitoring
   - Optimize network utilization: Data center networking
   - Minimizing power consumption: Traffic engineering
   - DoS attack mitigation: Security and dependability
2. The SDX architectures provides which of the following benefits?
   - It provides an AS more flexibility for managing traffic. (True)
   - It scales easily with more participants.
   - It provides faster peering compared to a conventional IXP.
   - All of the above
3. In the SDX architecture, each participant AS must use the same network applications for traffic engineering.
   - False
4. In the SDX architecture, two autonomous systems can choose to only exchange video traffic.
   - True
5. Which of the follow are applications of the SDX architecture?
   - Inbound traffic engineering
   - Wide-area server load balancing
   - Redirection of traffic through middleboxes
   - All of the above (True)