Understanding the Technology

Getting to Know SDN Controllers

- **SDN is the brain of Software Defined Network. It uses existing technologies but with a different approach than how they were used in the past.**

- **An SDN controller is the centralized repository of policy and control instructions for the network or application infrastructure.**

- **When we write control software to implement the desired policies, it is typically the case that we're programming the controller and using APIs on the controller. The software can automate network administration tasks, reducing the need for manual operations. This helps achieve cloud-level scale, supporting tens of thousands of servers and millions of workloads.**

- **The controller bridges the gap between open, programmable network elements and the applications that communicate with them, automating the provisioning (the setup and management) of the entire infrastructure, including the network, services, and applications.**

- **The controller gives a programmatic interface (sometimes called a *northbound* interface) for setting policies and provisioning services across the network. We can use it to eliminate network complexity and simply run the physical and virtual network automatically.**

To reduce the burden of managing the network, the controller translates business policy directly into network device-level policy.

It automates the deployment, compliance checking, and enforcement of network policies across the environment.

A controller also provides:

✓Consistency across the enterprise network that keeps downtime to a minimum and lowers operational complexity and associated cost.
✓Automated end-to-end provisioning and configuration to enable rapid deployment of applications and services.
✓Support for both existing and new deployments that lets you implement programmability and automation with the infrastructure you already have.

For Ex: In Cisco ACI, the APIC(Application Policy Infrastructure Controller) Controller manages an application centric policy that unifies the network, security, server, storage, application, and cloud teams around the infrastructure requirements of the business-critical applications.

Figure 3-1: APIC provides a common SDN controller across IT teams.

Essentially, Cisco's SDN policies are the instructions that tell the controller how the various network elements are supposed to deal with network and application traffic.

At the heart of ACI is a concept called an application network profile (ANP). The ANP is a policy that defines the requirements of the application in terms of network resources and how application components are connected.

When new instances of an application are needed (for capacity expansion or in a new location), the ANP is used as the template. The ANP shows how to deploy the application and all the resources and services it needs wherever it can be optimally placed in the cloud fabric.

ACI translates this replicable template into a long series of commands to the individual devices/boxes to properly configure all the services and infrastructure components automatically. These tedious steps are no longer left to the administrators (or multiple teams of administrators, more likely!).

This automation accelerates IT processes, aligns them with business needs, and reduces errors in large scale deployments.

- To implement policies, Cisco uses an open architecture that includes the SDN controller and policy repository, as well as infrastructure nodes such as network devices, virtual switches, network services, and so on that are controlled by the controller.

- There is also an open communication protocol between the controller and the infrastructure nodes (also called a *southbound* protocol or interface).

- OpFlex is the name of the communication protocol between the controller and the managed devices in the Cisco ACI system.

- OpFlex was designed with an application-centric policy model in mind. With OpFlex, the devices like the switches and the firewalls still enforce the policies in Cisco ACI.

- This approach provides the centralization of policy management with distributed control, allowing automation of the entire infrastructure without limiting scalability through a centralized control point or creating a single point of catastrophic failure.

OpFlex uses high-level application-oriented policy statements sent to network nodes, and it relies on those devices to determine how to implement the policies.

After device configurations are completed, such as configuration of an application delivery controller to support a particular service, communication between the controller and the infrastructure may be required only in the event of policy updates.

OpFlex is a bidirectional communication protocol.

In addition to communication in the form of commands from the controller to the managed devices, useful state, performance, and health metrics can be sent from network nodes to either the controller or an external analytics engine (also called an observer).

There are two types of management models:

1. Declarative management model.
2. Imperative management model.

In declarative management model the policy is centralized, but the enforcement of the policy isn’t, it’s distributed (and performed by the network nodes which decide the best way to enforce the policy).

Cisco ACI and OpFlex implement what is called a *declarative* management model.

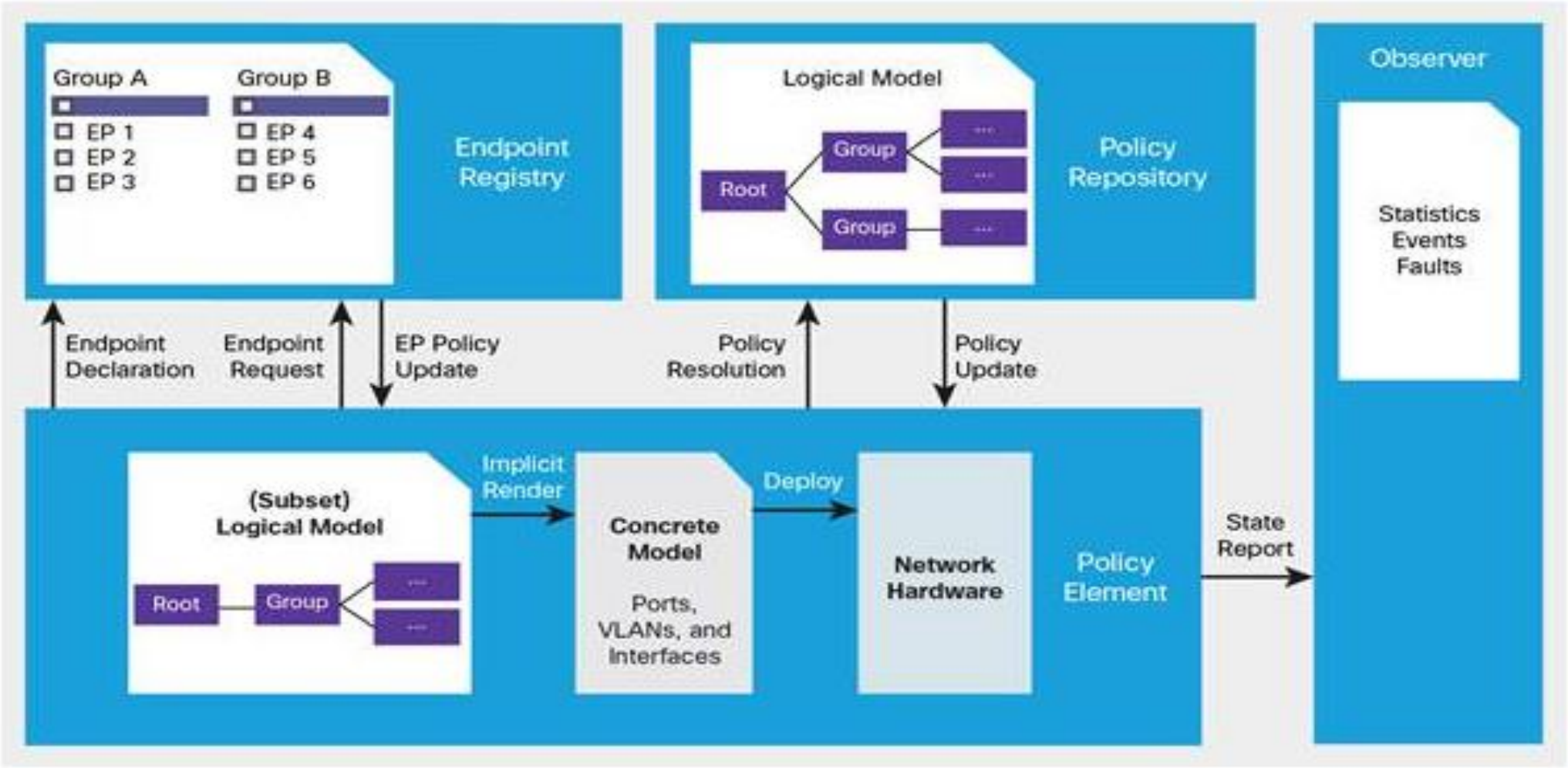
In *imperative management* model, the controller issues all the control instructions when they’re needed, which requires that the controller understand all the device states in real time, which may be impractical.

The imperative model essentially centralizes both the policy and the enforcement, which can be a bottleneck or create a single point of failure.

The declarative design makes sense when you consider the ways that devices are configured and policies are enforced in network nodes that vary considerably across vendors and models. These details would introduce complexity into the policy model or result in a policy that doesn’t take full advantage of each device’s full capabilities if everything was enforced directly within the controller.

The likely result would be a proprietary system with controlled devices modeled on a small number of device types or a single vendor’s products.

Figure 2. OpFlex Logical Model



Cisco, along with partners including Microsoft, Red Hat, Citrix, F5, Canonical, and Embrane, developed OpFlex.

OpFlex is an open and extensible policy protocol for transferring abstract policy in XML or JavaScript Object Notation (JSON) between a network policy controller such as the Cisco APIC and any device such as a switch.

The protocol is designed to support XML and JSON and to use standard remote procedure call (RPC) mechanisms such as JSON-RPC over TCP.

The use of a secure channel through Secure Sockets Layer (SSL) or Transport Layer Security (TLS) is also recommended.

Although OpFlex is a Cisco Protocol, it is open source and any vendor can use it.

The protocol defines a number of logical constructs required for its operation:

Policy Repository

The policy repository (PR) is a logically centralized entity containing the definition of all policies governing the behavior of the system. In Cisco ACI, this function is performed by the Cisco APIC or by the leaf nodes of the network fabric. The policy authority handles policy resolution requests from each policy element.

Policy Element (Policy Agent)

A policy element (PE) is a logical abstraction for a physical or virtual device that implements and enforces policy. Policy elements are responsible for requesting portions of the policy from the policy authority as new endpoints connect, disconnect, or change. Additionally, policy elements are responsible for rendering that policy from an abstract form into a concrete form that maps to their internal capabilities. This process is a local operation and can function differently on each device as long as the semantics of the policy are honored.

Endpoint Registry

The endpoint registry (ER) stores the current operation state (identity, location, etc.) of each endpoint (EP) in the system. The endpoint registry receives information about each endpoint from the local policy element and then can share it with other policy elements in the system. The endpoint registry may be physically co-located with the policy authority, but it may also be distributed in the network fabric itself. In Cisco's ACI solution, the endpoint registry actually lives in a distributed database within the network itself to provide additional performance and resiliency.

Observer

The Observer serves as the monitoring subsystem that provides a detailed view of the system operational state and performance. It serves as a data repository for performance and state data that pertains to the devices under control. This could include information related to trending, forensics, and long-term visibility data such as statistics, events, and faults. Statistical data reported to the Observer at expiration of reporting intervals and statistics will be rolled up for longer-term trend analysis.

Table 1. RPC Methods Supported by OpFlex

Command	From	To	Description
Identity	Any	Any	Is the first OpFlex message between any entity and its peer (frequently the PE and PR)
Policy Resolution	PE	PR	Retrieves a set of policies for an object
Policy Update	PR	PE	Sent to PEs when the policy definition for policies for which the PE has requested resolution has changed
Policy Trigger	PE	PE	Sends a policy trigger from one PE to a peer PE to trigger a policy resolution on the peer; may be in response to an attachment event that affects an upstream switch
Endpoint Declaration	PE	ER	Indicates the attachment of a new endpoint
Endpoint Request	PE	ER	Queries for an endpoint using a set of identifiers; for example, endpoints may be looked up by MAC address
Endpoint Policy Update	ER	PE	Sent by the ER when a change relating to EP declaration has occurred
State Report	PE	Observer	Sent by each PE to an observer to pass faults, events, and statistics

Overlay Networks in SDN Environments

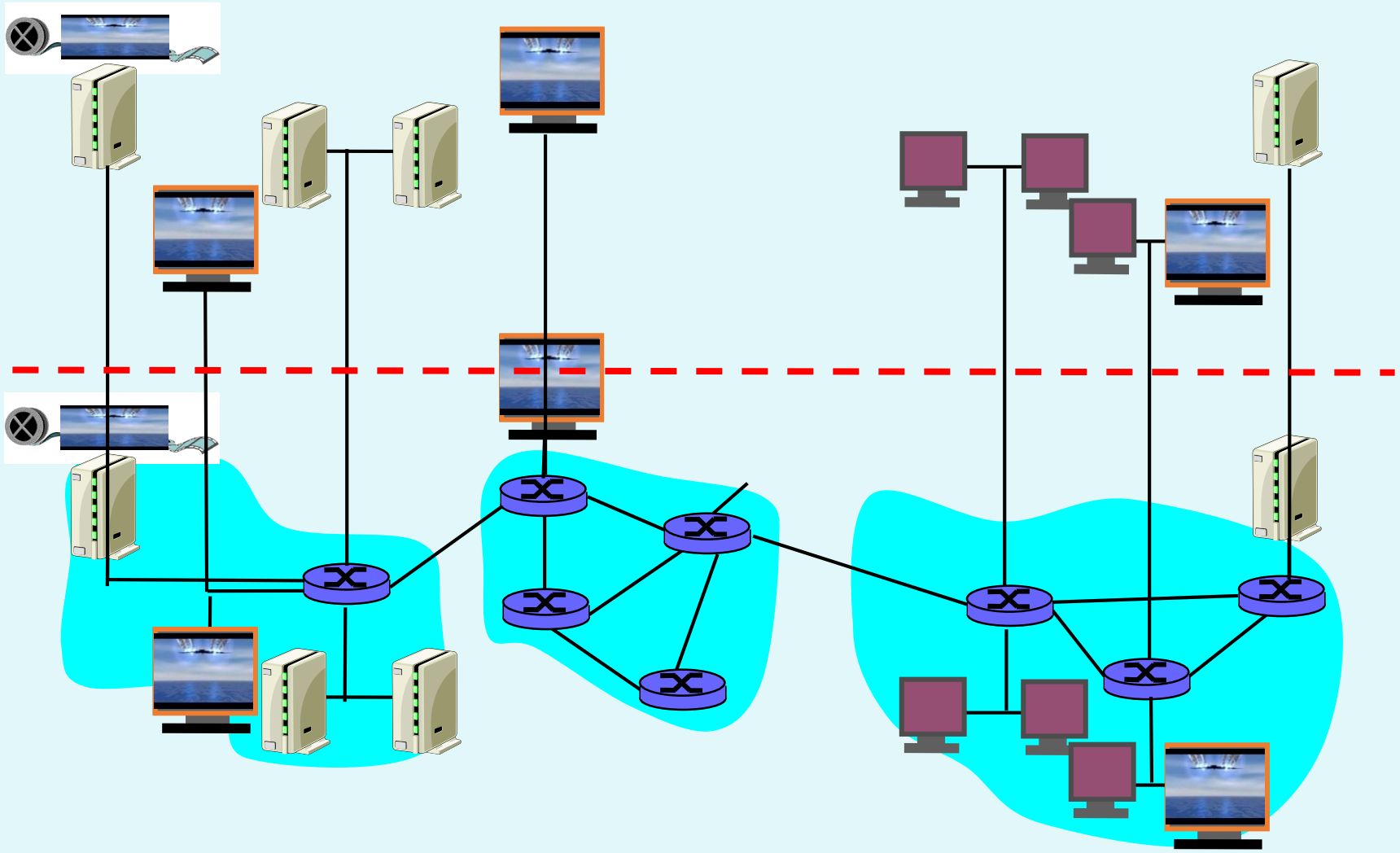
A virtual network can be called an overlay network because it defines a new logical topology on top of an underlying physical network.

Virtual network overlays are logically separated virtual networks of interconnected nodes that share an underlying physical network, allowing deployment of applications that require specific network topologies without the need to modify the underlying or physical network.

They’re frequently used to isolate tenants and applications that share the same physical network such as a cloud provider.

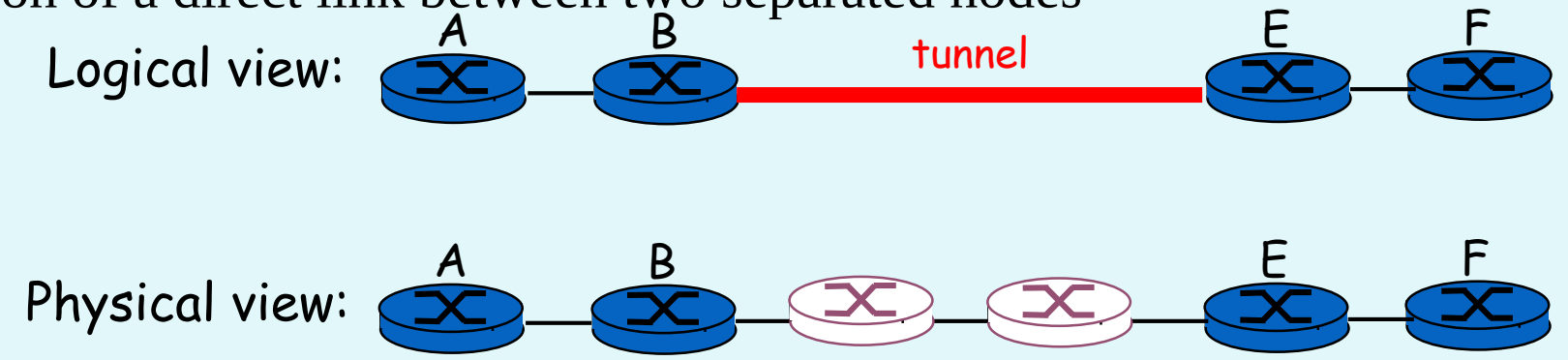
A *tenant* is a user of shared cloud resources. A tenant might be an organization with all its applications or an application comprised of a bunch of workloads that are all logically connected but isolated from all other application networks.

Overlay Networks



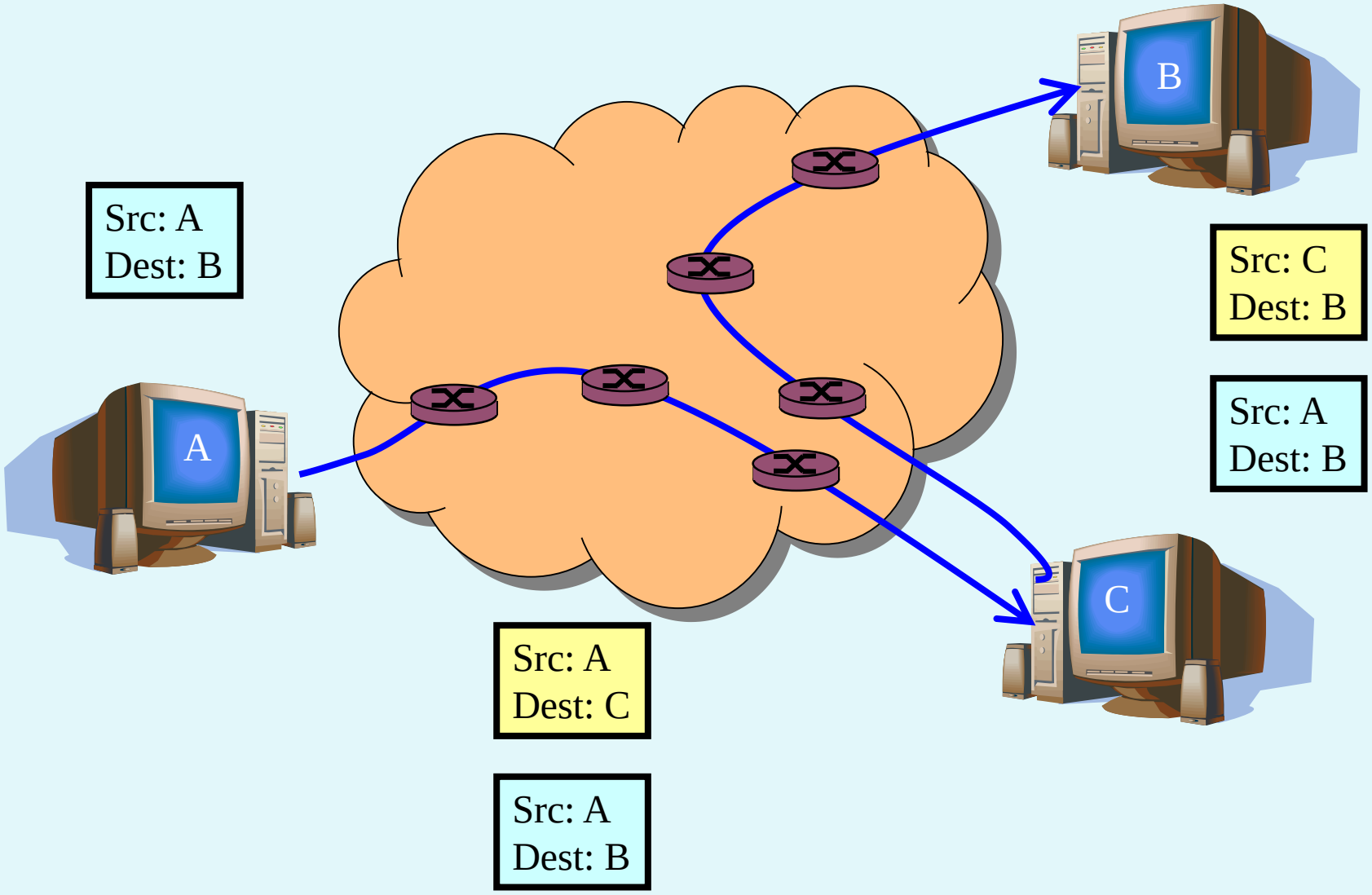
IP Tunneling to Build Overlay Links

- IP tunnel is a virtual point-to-point link
  - Illusion of a direct link between two separated nodes



- Encapsulation of the packet inside an IP datagram
  - Node B sends a packet to node E
  - … containing another packet as the payload

Tunnels Between End Hosts



A Historical Example

The Internet is an overlay network

- – goal: connect local area networks
- – built on local area networks (e.g., Ethernet), phone lines
- – add an Internet Protocol header to all packets

Overlay Networks

- A logical network built on top of a physical network
  - Overlay links are tunnels through the underlying network

- **Many logical networks may coexist at once**
  - **Over the same underlying network**
  - **And providing its own particular service**

- Nodes are often end hosts
  - Acting as intermediate nodes that forward traffic
  - Providing a service, such as access to files

- **Who controls the nodes providing service?**
  - **The party providing the service**
  - **Distributed collection of end users**

VXLAN (Virtual Extensible Local Area Network) is an overlay protocol initially developed by Cisco and promoted as a standard by VMware, Red Hat, Citrix, and other virtualization infrastructure vendors.

It is used in many SDN installations, particularly those that rely on VMware.

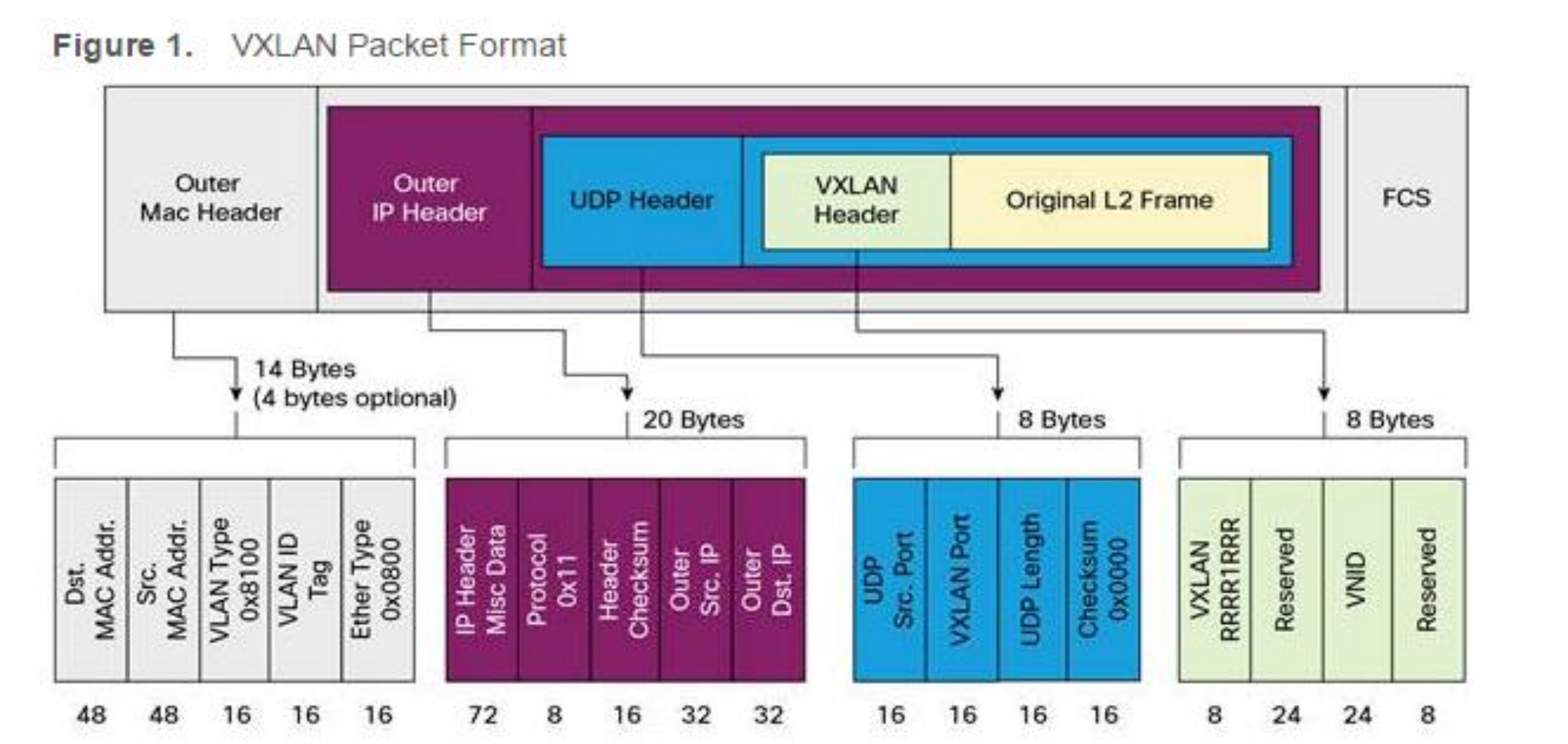


Figure 2. VTEP

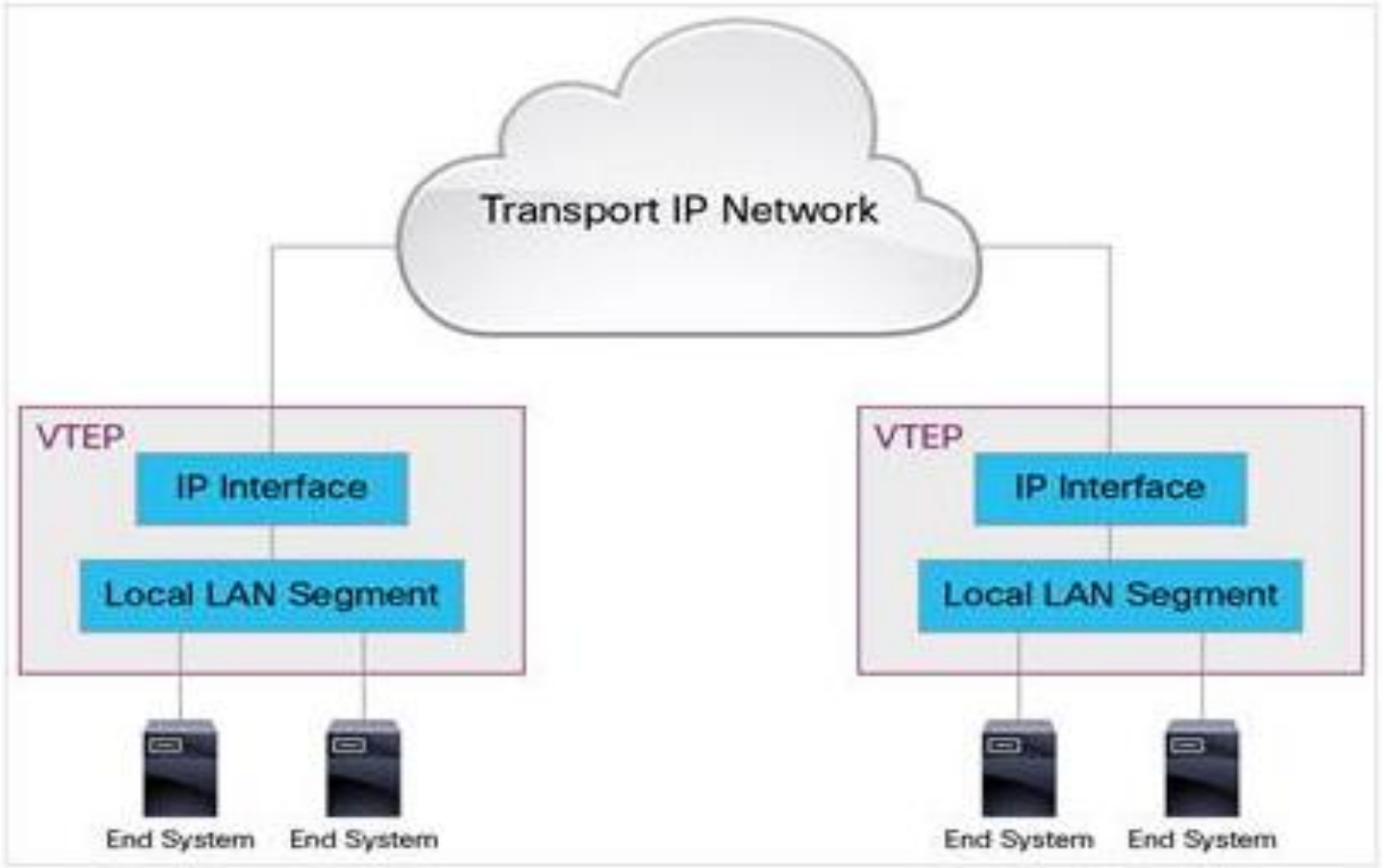
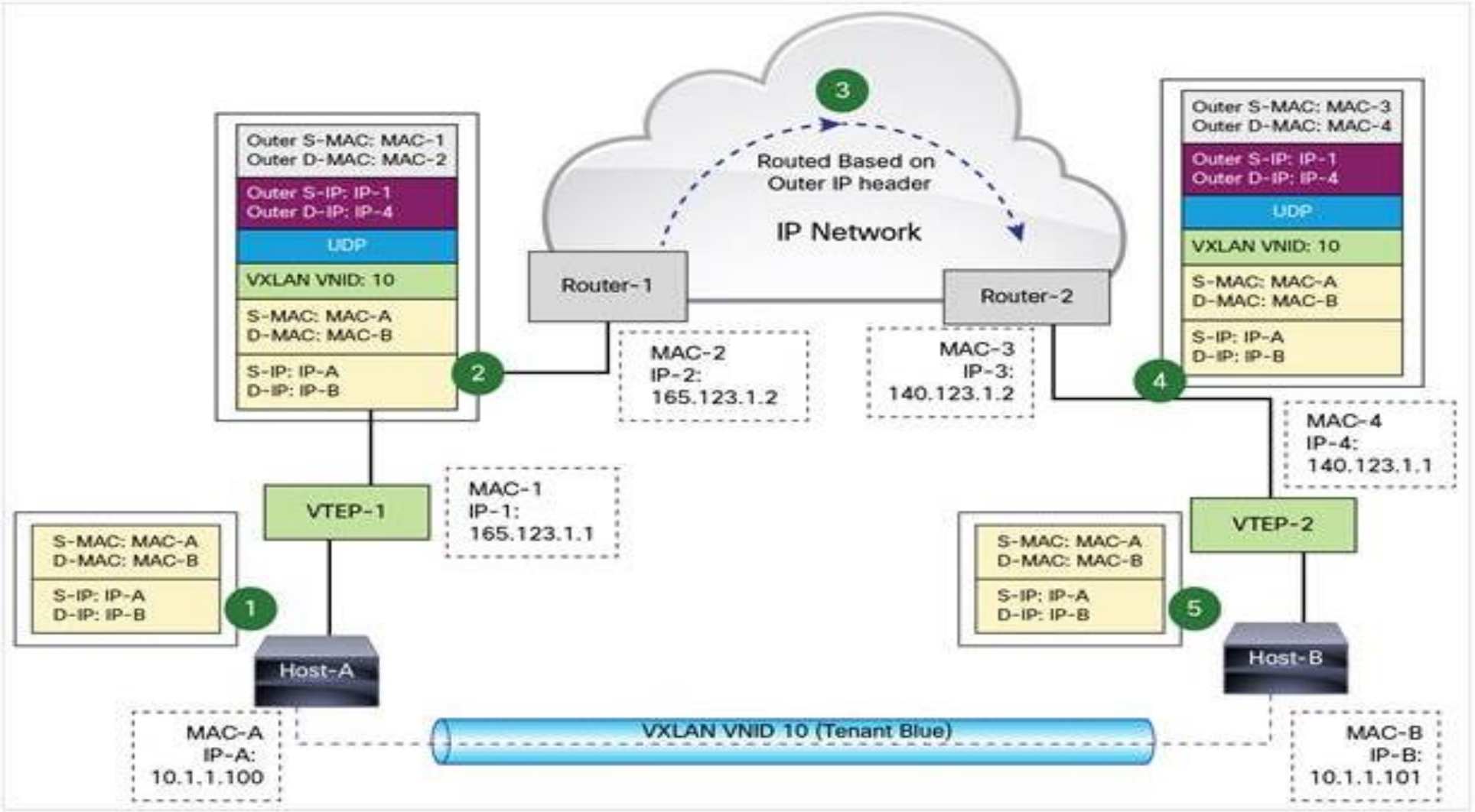


Figure 3. VXLAN Unicast Packet Forwarding Flow



Another common overlay network technology is NVGRE (network virtualization using generic routing encapsulation), common in Microsoft data center environments.

VXLAN and NVGRE are both examples of overlay networks that span not only physical Layer 2 networks, but Layer 3 routed networks.

This means that you can have one VXLAN network that runs across Layer 3 connections to other data centers and to cloud providers. It can go wherever you might need it.

These overlay networks are all implemented by encapsulating network packets within a new overlay packet header that may or may not be recognized by network devices.

Only the edge switches that represent the beginning and end of the overlay tunnel need to encapsulate or de capsulate the packet.

The intermediary network devices can pass the packets unimpeded.

Any cloud automation platform running on an SDN architecture will necessarily need to understand how to set up and configure these virtual network overlays in support of application deployments and to establish new tenants in cloud environments.

Virtual switches, virtual firewalls, and so on will also need to integrate with the SDN infrastructure.

Virtual or Overlay networks provide a flexible, programmable infrastructure that SDN can take advantage of.

NVGRE: Salient Features

- Does not follow flooding concept to learn unknown MAC addresses.
- Jumbo frames.
- GRE header.
- TNI(Tenant Network Identifier field of 24 bits).

Advantages of Overlay Networks

✓**Fabric scalability and flexibility:**

Overlay technologies allow the network to scale by focusing on the network overlay edge devices.

With overlays used at the fabric edge, the spine and core devices are freed from the need to add end-host information to their forwarding tables.

Additionally, the placement of end hosts is more flexible because the overlay virtual network no longer needs to be constrained to a single physical location.

✓**Arbitrary Layer 2 connectivity over Layer 3 underlay:**

It is possible to decouple the network services provided to end hosts from the topology used in the physical network.

✓ **Overlapping addressing:**

Most overlay technologies used in the data center allow virtual network IDs to uniquely scope and identify individual private networks.

This scoping allows potential overlap in MAC and IP addresses between tenants. The overlay encapsulation also allows the underlying infrastructure address space to be administered separately from the tenant address space.

✓ **Separation of roles and responsibilities:**

With the encapsulation used in overlay technologies, separation of administration domains can also be achieved.

The administrator of the cloud fabric (infrastructure) can be responsible for fabric addressing, availability, and load balancing, while the individual tenants can be responsible for their own addressing policies and services without affecting infrastructure policies.

✓**Support for unlimited VNs**

Support for essentially unlimited numbers of VNs as the 24 bits that are typically used by network overlays to identify VNs can identify slightly more than 16 million VN IDs. While theoretically VN solutions can support 16 million VNs, practical limits are often in the range of 16,000 to 32,000 VNs.

✓**Decoupling of VN**

Decoupling of the virtual network topology from the physical network infrastructure and decoupling of the "virtual" MAC and/or IP addresses used by VMs from the infrastructure IP addresses used by the physical data center core network. The decoupling avoids issues such as limited MAC table size in physical switches.

✓**Support for VM mobility**

Support for virtual machine (VM) mobility independent of the physical network. If a VM changes location, even to a new subnet in the physical network, the switches at the edge of the overlay simply update mapping tables to reflect the new physical location of the VM.

Disadvantages of the Overlay-Based Approach

- Gateways between the virtual network systems and network service points on the physical network may need to pass high volumes of traffic. If a software gateway running on a VM or a dedicated appliance has insufficient processing power, hardware support for the gateway functionality may be required in physical switches or network service appliances.

There are silicon switching chips that support gateway functionality for VXLAN which is the most popular encapsulation protocol.

- Some value-added features in existing networks cannot be leveraged due to encapsulation. For example, the physical network loses its ability to provide differentiated services based on the content of the packet header.

- Adds complexity

- Adds Overhead

Automating Cloud via SDN

SDN solutions can provide a powerful framework for hosting and adding value to cloud automation solutions.

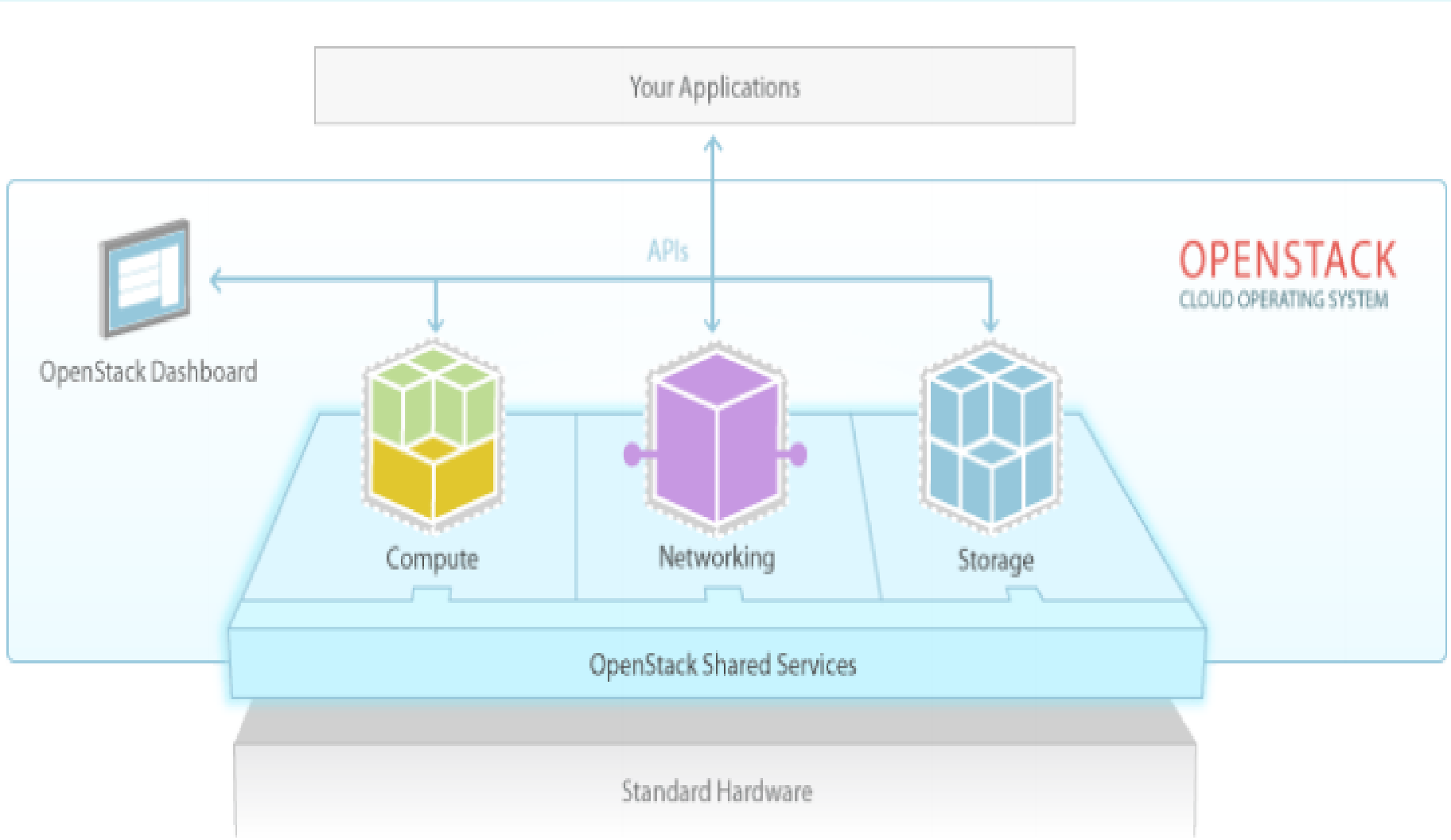
The ideal SDN framework will be capable of hosting a range of cloud automation and workflow automation tools through the open northbound interfaces on the SDN controller.

This flexibility will give customers a choice in what automation tools they can choose for their needs.

OpenStack is a leading open source cloud orchestration tool available today and is backed by more than 125 companies.

OpenStack is designed as a building block for public and private clouds, allowing automated management of computing, storage, and networking resources in a very flexible manner.

- OpenStack is mostly deployed as infrastructure-as-a-service (IaaS).
- This software platform consists of interrelated components that control hardware pools of processing, storage, and networking resources throughout a data center.
- Users either manage it through a web-based dashboard, or through command-line tools.
- OpenStack.org released it under the terms of the Apache License.
- OpenStack began in 2010 as a joint project of Rackspace Hosting and NASA.
- As of 2016, it is managed by the OpenStack Foundation.


Your Applications
APIs
OPENSTACK
CLOUD OPERATING SYSTEM
OpenStack Dashboard
Compute
Networking
Storage
OpenStack Shared Services
Standard Hardware

OpenStack, which is freely available under the Apache 2.0 license, is often referred to in the media as "the Linux of the Cloud"

OpenStack has a modular architecture that currently has eleven components:

- Nova - provides virtual machines (VMs) upon demand.
- Swift - provides a scalable storage system that supports object storage.
- Cinder - provides persistent block storage to guest VMs.
- Glance - provides a catalog and repository for virtual disk images.

- Keystone - provides authentication and authorization for all the OpenStack services.
- Horizon - provides a modular web-based user interface (UI) for OpenStack services.
- Neutron - provides network connectivity-as-a-service between interface devices managed by OpenStack services.
- Ceilometer - provides a single point of contact for billing systems.
- Heat - provides orchestration services for multiple composite cloud applications.
- Trove - provides database-as-a-service provisioning for relational and non-relational database engines.
- Sahara - provides data processing services for OpenStack-managed resources.

Block Storage

Old concept. Data is grouped into blocks. Block identifiers identify individual blocks.

Block storage devices provide fixed-sized raw storage capacity. Each storage volume can be treated as an independent disk drive and controlled by an external server operating system.

The most common examples of Block Storage are SAN, iSCSI, and local disks.

Block storage is the most commonly used storage type for most applications.

Block storage devices typically are formatted with a file system like FAT32, NTFS, EXT3, and EXT4.

Use cases

- Block storage is ideal for databases, since a DB requires consistent I/O performance and low-latency connectivity.

- Any application which requires server side processing, like Java, PHP, and .net will require block storage.

- Required for running mission-critical applications like Oracle, SAP, Microsoft Exchange, and Microsoft SharePoint.

✓ **Block storage options in the cloud: AWS Elastic Block Storage**

Object Storage

- Relatively newer concept. Block storage volumes can only be accessed when they're attached to an operating system.
- But data kept on object storage devices, which consists of the object data and metadata, can be accessed directly through APIs or http/https.
- You can store any kind of data, photos, videos, and log files. The object store guarantees that the data will not be lost.
- Object storage data can be replicated across different data centers and offer simple web service interfaces for access.

Use Cases:

- Storage of unstructured data like music, image, and video files.
- Storage for log files.
- Large data sets. Whether you're storing pharmaceutical or financial data, or multimedia files such as photos and videos, storage can be used as your **big data** object store.

Use Cases

- Archive files in place of local tape drives. Media assets such as video footage can be stored in object storage and archived to AWS glacier.

- As data stores scale to hundreds of terabytes and then into the petabyte range and beyond, object storage becomes even more attractive.

✓ **Object storage options in the Cloud: Amazon S3**:

Object Storage Versus Block Storage

	OBJECT STORAGE	BLOCK STORAGE
PERFORMANCE	Performs best for big content and high stream throughput	Strong performance with database and transactional data
GEOGRAPHY	Data can be stored across multiple regions	The further the distance between storage and application, the higher the latency
SCALABILITY	Can scale infinitely to petabytes and beyond	Addressing requirements limit scalability
ANALYTICS	Customizable metadata allows data to be easily organized and retrieved	No metadata

Key SDN Considerations and Requirements

The emergence of software defined networking (SDN) promises a new era of centrally managed, policy-based automation tools that could accelerate network management, optimization, and remediation.

However to reach those goals you need to be aware of certain SDN-related considerations and requirements. They are;

- ***Focusing on Applications***
- ***Keeping Things Open***
- ***Keeping What You Have by Using Cisco ACI***

Key SDN Considerations and Requirements

- ***Focusing on Applications***

Organizations are looking at cloud automation tools and SDN architectures to accelerate application delivery, reduce operating costs, and increase business agility.

To achieve this you need a more business relevant application policy language, a distributed enforcement system rather than centralized control, and greater network visibility.

- ***Keeping Things Open***

The ***open*** or ***nonproprietary*** approach reduces the ability of a single vendor to lock you into buying everything from it, thus increasing market competitiveness.

Most open source proponents feel that this openness leads to better products, lower costs, more innovation, and higher return on investment.

SDN is an open and transparent approach to the evolving needs in cloud networking and data center management.

- Many industries including Cisco, Citrix, Ericsson, HP, IBM, Juniper Networks, Microsoft, Red Hat, Vmware etc., founded the OpenDaylight Project in 2013 to collaborate on an open source SDN controller architecture.

- But even now some SDN vendors may lock you into certain components if you adopt their SDN offerings.

Organizations require an open, extensible, multivendor architecture that incorporates a wide range of infrastructure products and open source solutions. Ex: Cisco ACI.

- ***Keeping What You Have by Using Cisco ACI***

For the vast majority of organizations, it's simply not reasonable to throw out the existing investment in network infrastructure and start fresh.

Yes, you want the benefits of implementing an SDN solution, but you also need to make use of what you already have. There are SDN controllers that facilitate this by using the existing switches.

Thank You