

# Chapter 22: Enterprise Network Architecture

## Chapter 23: Fabric Technologies

## Chapter 24: Network Assurance

Instructor Materials



CCNP Enterprise: Core Networking

# Hierarchical LAN Design Model

- A hierarchical LAN design model divides the enterprise network architecture into modular layers.
- Modular layers allow each layer to implement specific functions.
- Modular layers can be replicated throughout the network providing scaling and a consistent deployment method.
- Provides fault containment.
- Provides the ability to put network components in place or take them out of service with no impact on the rest of the network.

## Hierarchical LAN Design Model

### 3 Layer Design

- **Access layer** - Gives endpoints and users direct access to the network.
- **Distribution layer** - Provides an aggregation point for the access layer and acts as a services and control boundary between the access layer and the core layer.
- **Core layer (also referred to as the backbone)** - Provides connections between distribution layers for large environments.

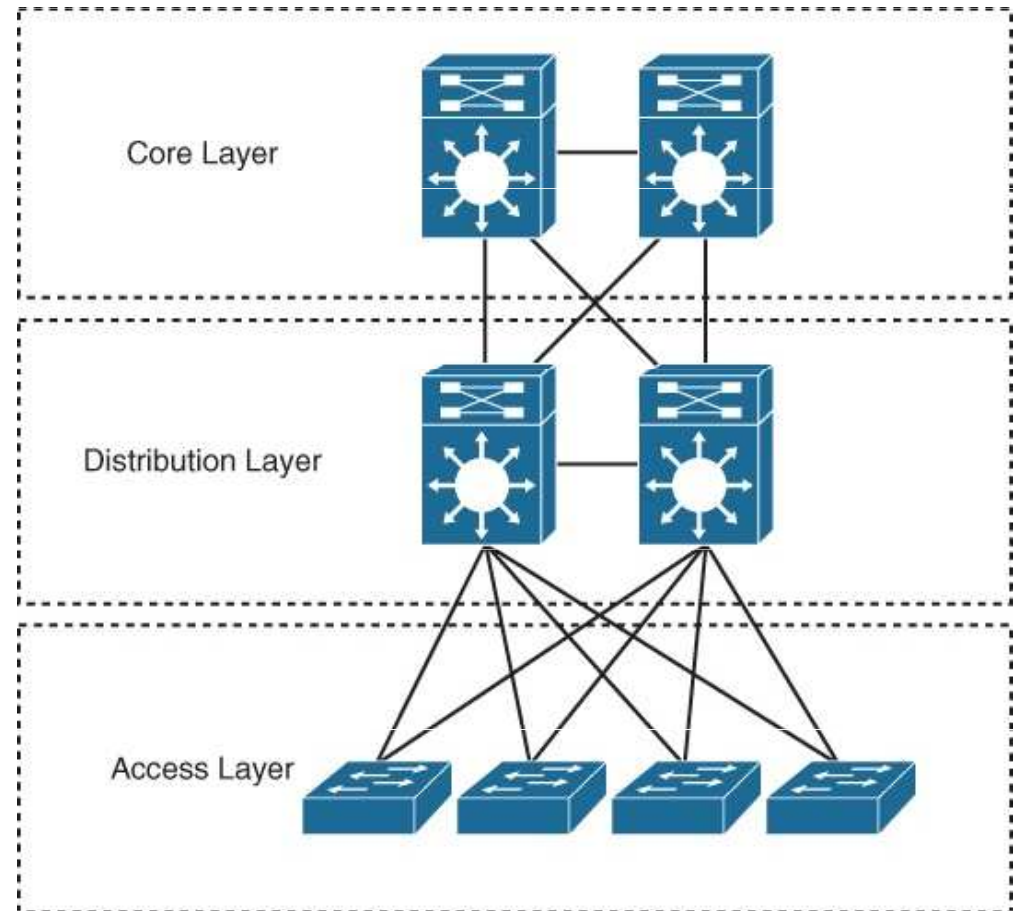


Figure 22-1 Hierarchical LAN Design

# Hierarchical LAN Design Model

## Scalable Layer Design

The number of layers needed depends on the characteristics of the network deployment site. As illustrated in Figure 22-2, a small campus in a single building might require only access and distribution layers, while a campus that spans multiple buildings will most likely require all three layers. The modularity of this design ensures that each layer will provide the same services and the same design methods.

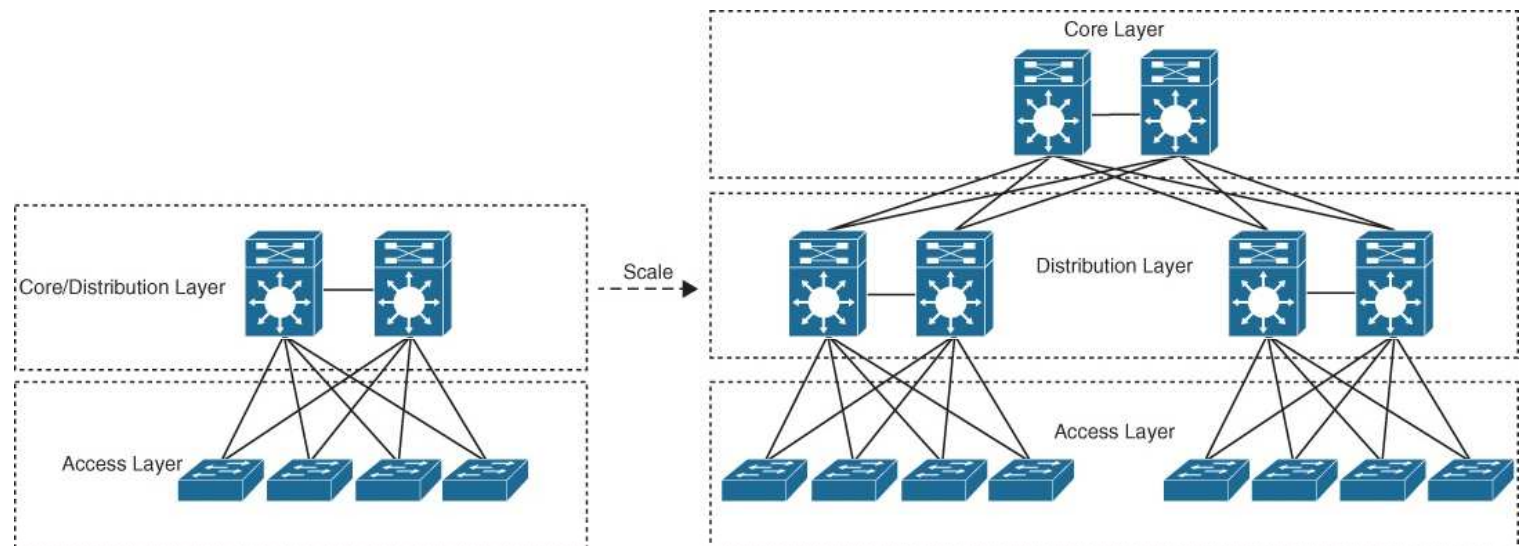


Figure 22-2 Modular Design Scalability

# Hierarchical LAN Design Model

## Access Layer

The access layer, also commonly referred to as the network edge, is where end-user devices or endpoints connect to the network.

It provides high-bandwidth device connectivity using wired and wireless access technologies such as Gigabit Ethernet and 802.11n and 802.11ac wireless.

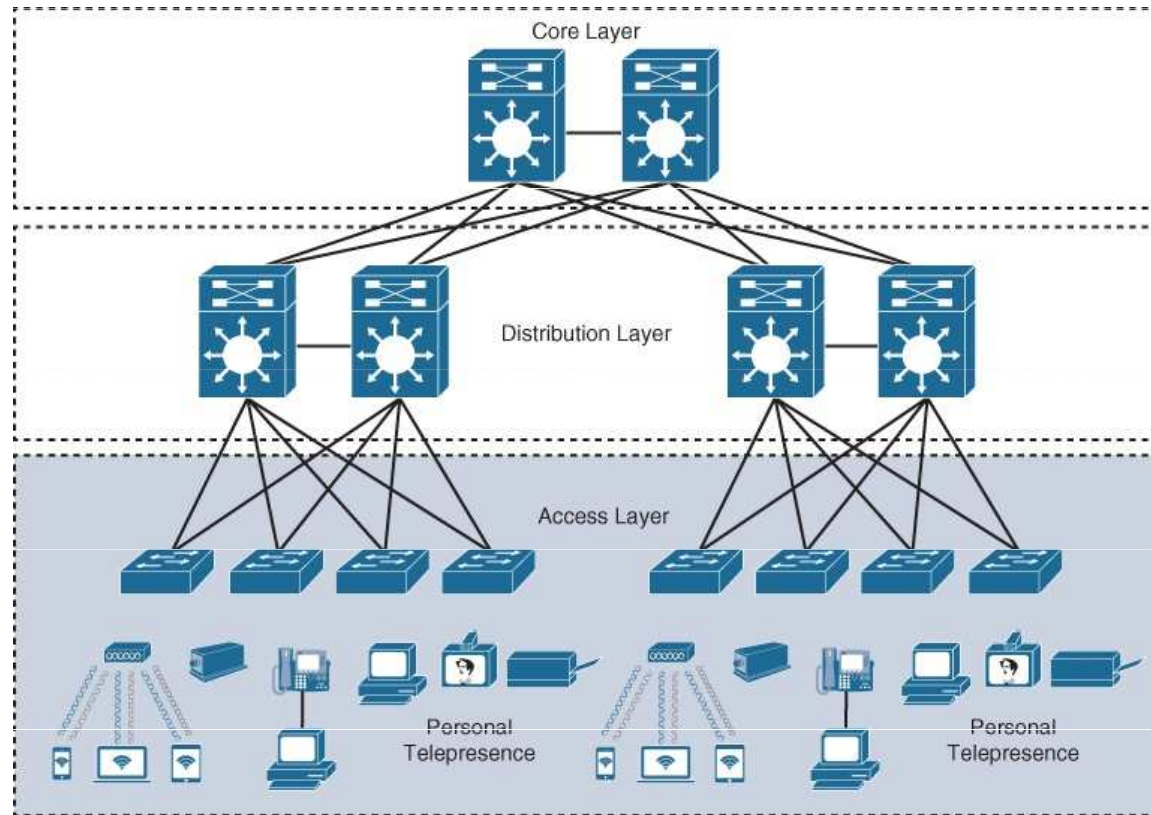


Figure 22-3 Access Layer Connectivity

## Hierarchical LAN Design Model Access Layer (Cont.)

- It can be segmented (for example, by using VLANs) so that different devices can be placed into different logical networks for performance, management, and security reasons.
- In the hierarchical LAN design, the access layer switches are not interconnected to each other. Communication between endpoints on different access layer switches occurs through the distribution layer.
- It plays a big role in ensuring that the network is protected from malicious attacks. This protection includes making sure the end users and endpoints connecting to the network are prevented from accessing services for which they are not authorized.
- Quality of service (QoS) trust boundary and QoS mechanisms are typically enabled on this layer to ensure that QoS is provided end-to-end to satisfy the end user's quality of experience (QoE).

## Hierarchical LAN Design Model

# Distribution Layer

- The primary function of the distribution layer is to aggregate access layer switches in a given building or campus.
- The distribution layer provides a boundary between the Layer 2 domain of the access layer and the core's Layer 3 domain.
- This boundary provides two key functions for the LAN:
  - On the Layer 2 side, the distribution layer creates a boundary for Spanning Tree Protocol (STP), limiting propagation of Layer 2 faults.
  - On the Layer 3 side, the distribution layer provides a logical point to summarize IP routing information when it enters the core of the network. The summarization reduces IP routing tables for easier troubleshooting and reduces protocol overhead for faster recovery from failures.

# Hierarchical LAN Design Model

## Distribution Layer

- The distribution switches need to be deployed in pairs for redundancy as displayed in Figure 22-4.
- The distribution layer switch pairs should be interconnected to each other using either a Layer 2 or Layer 3 link.
- When campus buildings are geographically dispersed, distribution layer switches can be located within the buildings in order to reduce the number of fiber-optic runs (which are costly) between buildings.

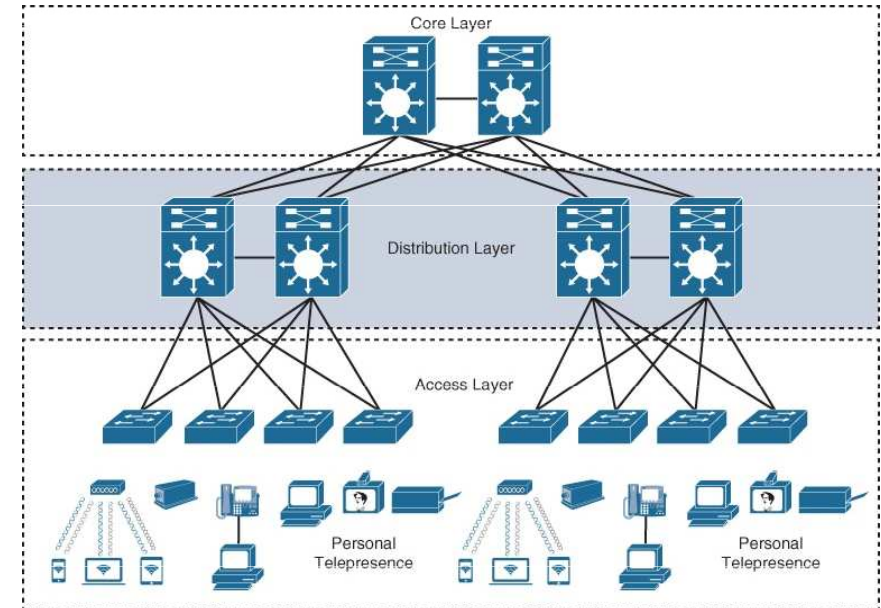


Figure 22-4 Distribution Layer Connectivity

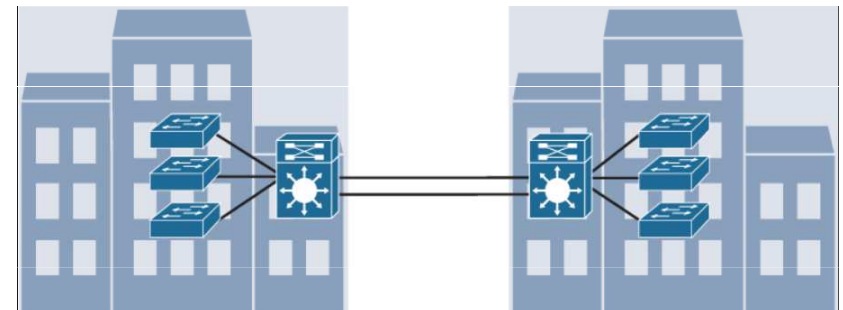


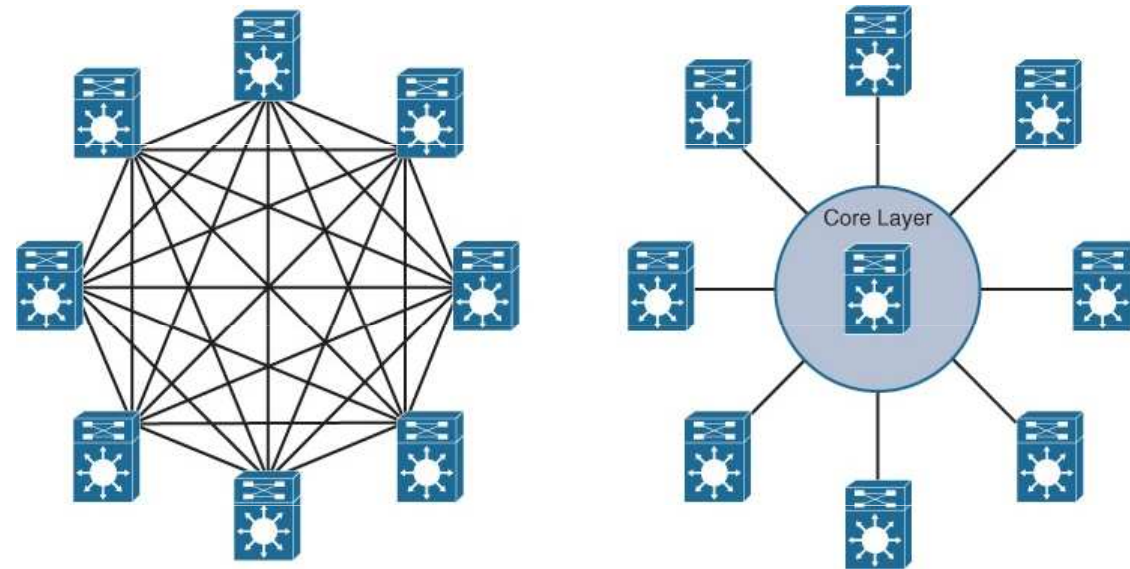
Figure 22-5 Distribution Layer Reducing Fiber Optic Runs



## Hierarchical LAN Design Model

# Core Layer

- As networks grow beyond three distribution layers in a single location, organizations should consider using a core layer to optimize the design.
- The core layer is the backbone and aggregation point for multiple networks and provides scalability, high availability, and fast convergence to the network.
- The core can provide high-speed connectivity for large enterprises with multiple campus networks distributed worldwide, and it can also provide interconnectivity between the end-user/endpoint campus access layer and other network blocks, such as the data center, the private cloud, the public cloud, the WAN, the internet edge, and network services.



**Figure 22-6** Core Layer Reduces Large Network Complexity

Use of the core to reduce the network complexity, from  $N \times (N - 1)$  to  $N$  links for  $N$  distribution layer switches.

# Enterprise Network Architecture Options

- Two-tier design (collapsed core)
- Three-tier design
- Layer 2 access layer (STP based)
- Layer 3 access layer (routed access)
- Simplified campus design
- Software-Defined Access (SD-Access)

## Two-tier design (Collapsed Core)

- Smaller campus networks may have multiple departments spread across multiple floors within a building. In these environments, a core layer may not be needed, and collapsing the core function into the distribution layer can be a cost-effective solution (as no core layer means no core layer devices) that requires no sacrifice of most of the benefits of the three-tier hierarchical model.
- Prior to selecting a two-tier collapsed core and distribution layers, future scale, expansion, and manageability factors need to be considered.

## Enterprise Network Architecture Options

# Two-tier design (Cont.)

Switches at the Core/Distribution layer provide two separate functions:

- connectivity to the WAN, Data Center, internet, and network services edge blocks
- LAN aggregation to the end-user access layer switches

The WAN Edge block connects to remote data centers, remote branches, or other campus networks or for connectivity to cloud providers.

The Data Center/Server Room block is where business-critical servers are placed.

The Internet Edge block is used for regular internet access, e-commerce, connection to remote branches, remote VPN access and non-dedicated cloud provider connections.

The Network Services block is where devices providing network services reside, such as wireless LAN controllers (WLCs).

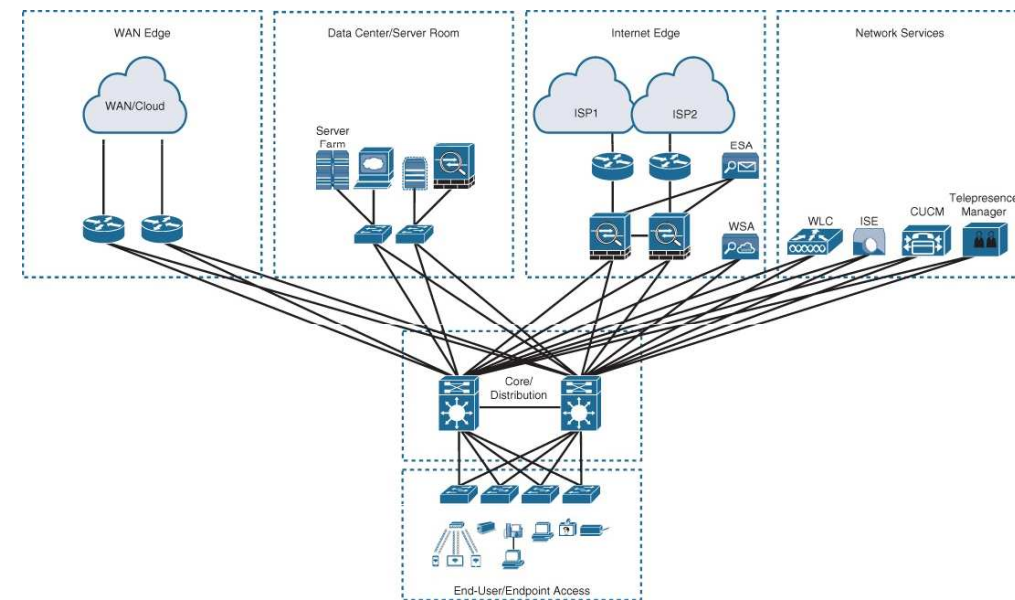


Figure 22-7 Two-Tier/Collapsed Core Design

## Enterprise Network Architecture Options

# Three-Tier Design

Three-tier designs separate the core and distribution layers and are recommended when more than two pairs of distribution switches are required.

Multiple pairs of distribution switches are typically required for the following reasons:

- When implementing a network for a large enterprise campus composed of multiple buildings, where each building requires a dedicated distribution layer.
- When the density of WAN routers, internet edge devices, data center servers, and network services are growing to the point where they can affect network performance and throughput.
- When geographic dispersion of the LAN access switches across many buildings in a larger campus facility would require more fiber-optic interconnects back to a single collapsed core.

## Enterprise Network Architecture Options Three-Tier Design (Cont.)

In Figure 22-8, the building blocks or places in the network (PINs) are each using the hierarchical design model.

Each block is deployed with a pair of distribution switches connected to the core.

The hierarchical LAN design is more appropriate for north–south traffic flows, such as endpoints communicating with the WAN edge, data center, Internet, or network services blocks.

The data center block is using the newer leaf–spine design, an alternative to the three-tier design when traffic is predominantly east–west between servers within the data center.

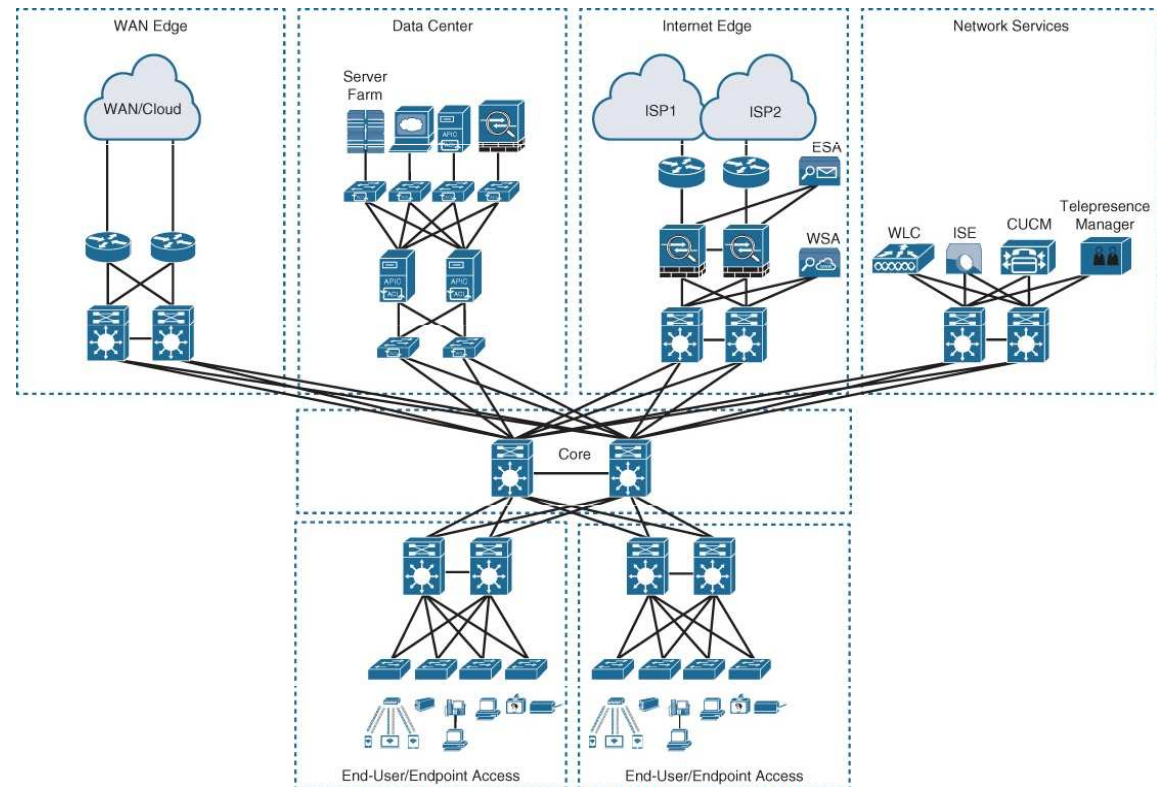


Figure 22-8 Three-Tier Design

## Enterprise Network Architecture Options

# Layer 2 Access Layer (STP Based)

- Traditional LAN designs use a Layer 2 access layer and a Layer 3 distribution layer.
- The distribution layer is the Layer 3 IP gateway for access layer hosts.
- Whenever possible, it is recommended to restrict a VLAN to a single access layer switch to eliminate topology loops.
- Restricting a VLAN to a single switch provides a loop-free design, but at the cost of network flexibility because all hosts within a VLAN are restricted to a single access switch.
- Some organizations require that the same Layer 2 VLAN be extended to multiple access layer switches to accommodate an application or a service.
- The looped design causes STP to block links, which reduces the bandwidth from the rest of the network and can cause slower network convergence.

# Enterprise Network Architecture Options Layer 2 Access Layer (STP Based) – Topology Design

Figure 22-9 illustrates a loop-free topology where a VLAN is constrained to a single switch, as well as a looped topology where a VLAN spans multiple access switches. The looped design causes STP to block links.

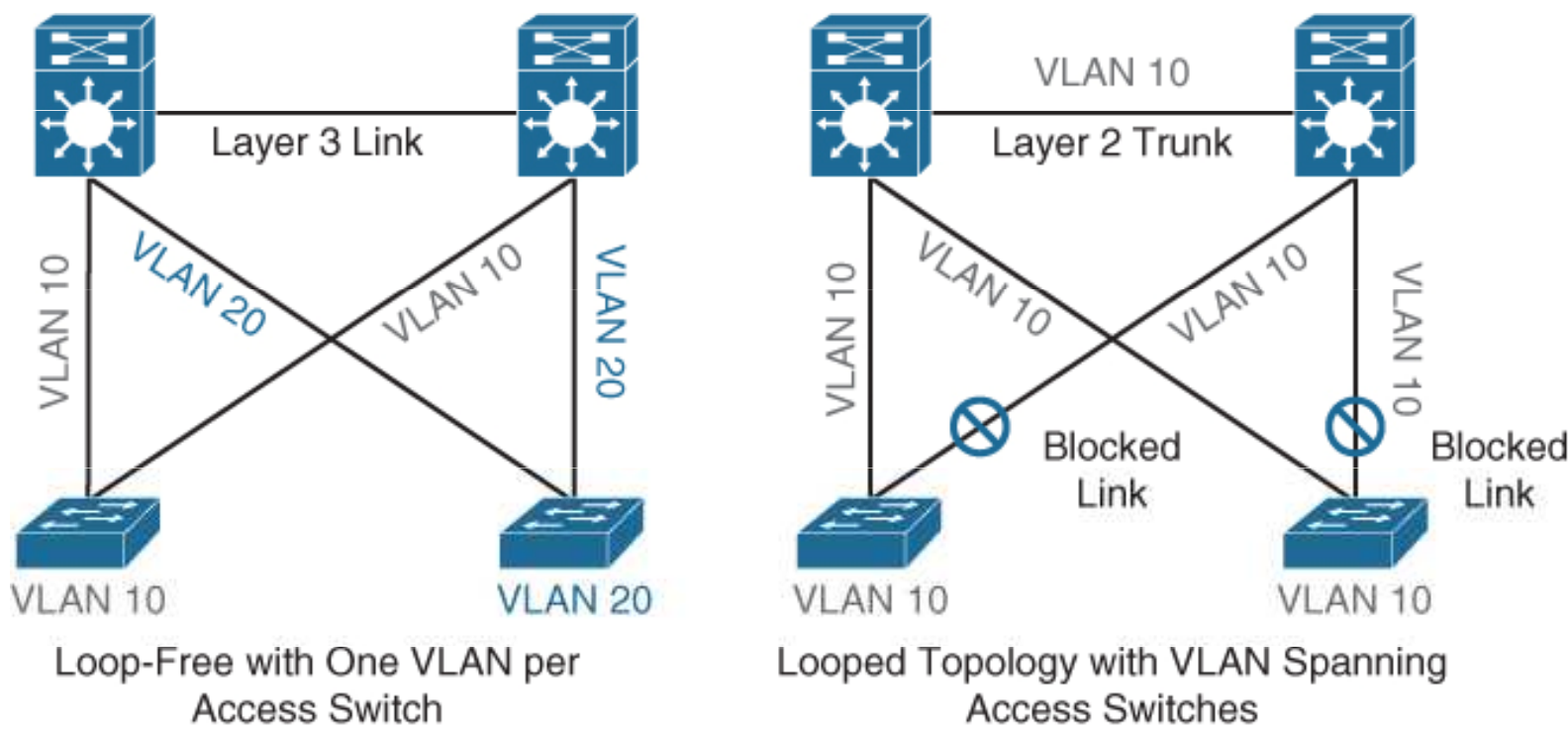


Figure 22-9 Loop-Free Topology and Looped Topology



## Layer 2 Access Layer (STP Based) – High Availability

- First-hop redundancy protocols (FHRPs) provide hosts with a consistent MAC address and gateway IP address for each configured VLAN. The distribution layer should have a pair of standalone switches configured with a FHRP.
- Hot Standby Router Protocol (HSRP) and Virtual Router Redundancy Protocol (VRRP) are the most common first-hop redundancy protocols.
  - The downside to these protocols is that they only allow hosts to send data out the active FHRP router through a single access uplink.
  - This leaves one of the access layer-to-distribution layer uplinks unutilized unless manual configuration of the distribution layer is performed.
- Gateway Load Balancing Protocol (GLBP) provides greater uplink utilization for access layer-to-distribution layer traffic by load balancing the load from hosts across multiple uplinks.
  - The downside is that it works only on loop-free topologies.

## Enterprise Network Architecture Options Layer 3 Access Layer (Routed Access)

Routed access is an alternative configuration in which Layer 3 is extended all the way to the access layer switches. In this design, access layer switches act as full Layer 3 routed nodes (providing both Layer 2 and Layer 3 switching), and the access-to-distribution Layer 2 uplink trunks are replaced with Layer 3 point-to-point routed links. Consequently, the Layer 2/Layer 3 demarcation point is moved from the distribution switch to the access switch, as illustrated in Figure 22-10.

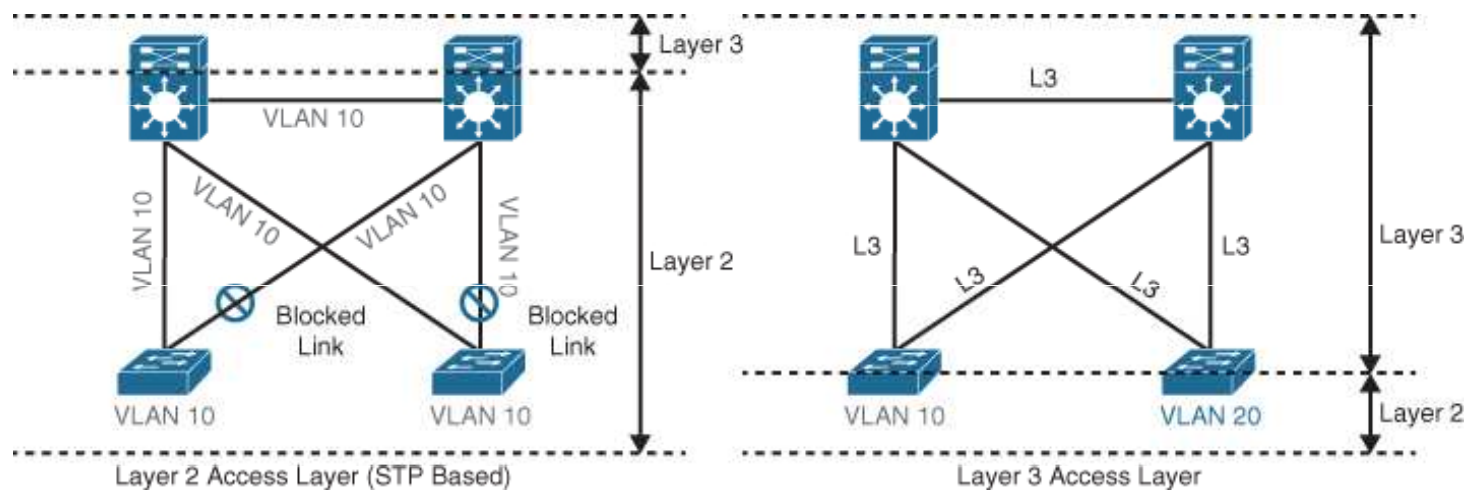


Figure 22-10 Layer 2 Access Layer and Layer 3 Access Layer

## Enterprise Network Architecture Options

# Layer 3 Access Layer (Routed Access)

The routed access-to-distribution block design advantages over the Layer 2 access layer design:

- **No first-hop redundancy protocol required** - It eliminates the need for first-hop redundancy protocols such as HSRP and VRRP.
- **No STP required** - Because there are no Layer 2 links to block, this design eliminates the need for STP.
- **Increased uplink utilization** - Both uplinks from access to distribution can be used, increasing the effective bandwidth available to the end users and endpoints connected to the access layer switches.
- **Easier troubleshooting** - It offers common end-to-end troubleshooting tools (such as **ping** and **traceroute**).
- **Faster convergence** - It uses fast-converging routing protocols such as Enhanced Interior Gateway Routing Protocol (EIGRP) and Open Shortest Path First (OSPF).

## Enterprise Network Architecture Options

# Simplified Campus Design

Relies on switch clustering such as a virtual switching system (VSS) and stacking technologies such as StackWise, in which multiple physical switches act as a single logical switch. Clustering and stacking technologies can be applied to any of the campus building blocks to simplify them even further. Using this design offers the following advantages:

- **Simplified design** - By using the single logical distribution layer design, there are fewer boxes to manage, which reduces the amount of time spent on ongoing provisioning and maintenance.
- **No first-hop redundancy protocol required** - It eliminates the need for first-hop redundancy protocols such as HSRP and VRRP because the default IP gateway is on a single logical interface.
- **Reduced STP dependence** - Because EtherChannel is used, it eliminates the need for STP for a Layer 2 access design. However, STP is still required as a failsafe in case multiple access switches are interconnected.

## Enterprise Network Architecture Options

# Simplified Campus Design (Cont.)

### Simplified Campus Design advantages:

- **Increased uplink utilization** - With EtherChannel, all uplinks from access to distribution can be used, increasing the effective bandwidth available to the end users and endpoints connected to the access layer switches.
- **Easier troubleshooting** - The topology of the network from the distribution layer to the access layer is logically a hub-and-spoke topology, which reduces the complexity of the design and troubleshooting.
- **Faster convergence** - With EtherChannel, all links are in forwarding state, and this significantly optimizes the convergence time following a node or link failure event. Provides fast sub-second failover between links in an uplink bundle.
- **Distributed VLANs** - With this design, VLANs can span multiple access switches without the need to block any links.

# Enterprise Network Architecture Options

## Simplified Campus Design (Cont.)

The simplified campus design is loop free, highly available, flexible, resilient, and easy to manage. Figure 22-11 illustrates how the network can be simplified by introducing VSS and StackWise into the design.

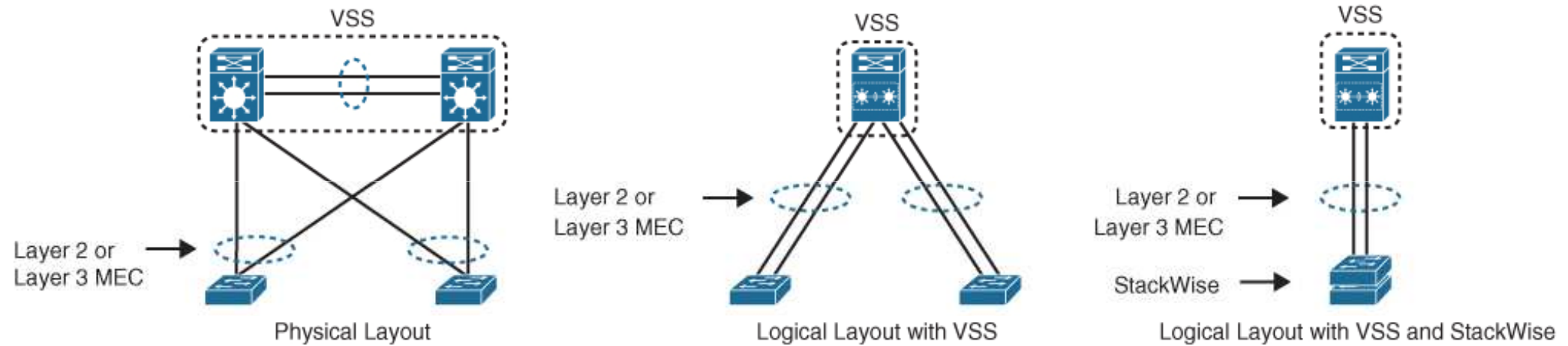


Figure 22-11 Simplified Campus Design with VSS and StackWise

## Enterprise Network Architecture Options Simplified Campus Design (Cont.)

Using the simplified campus design approach across all the campus blocks (when possible) can provide an optimized architecture that is easy to manage, resilient, and more flexible, with higher aggregated uplink bandwidth capacity.

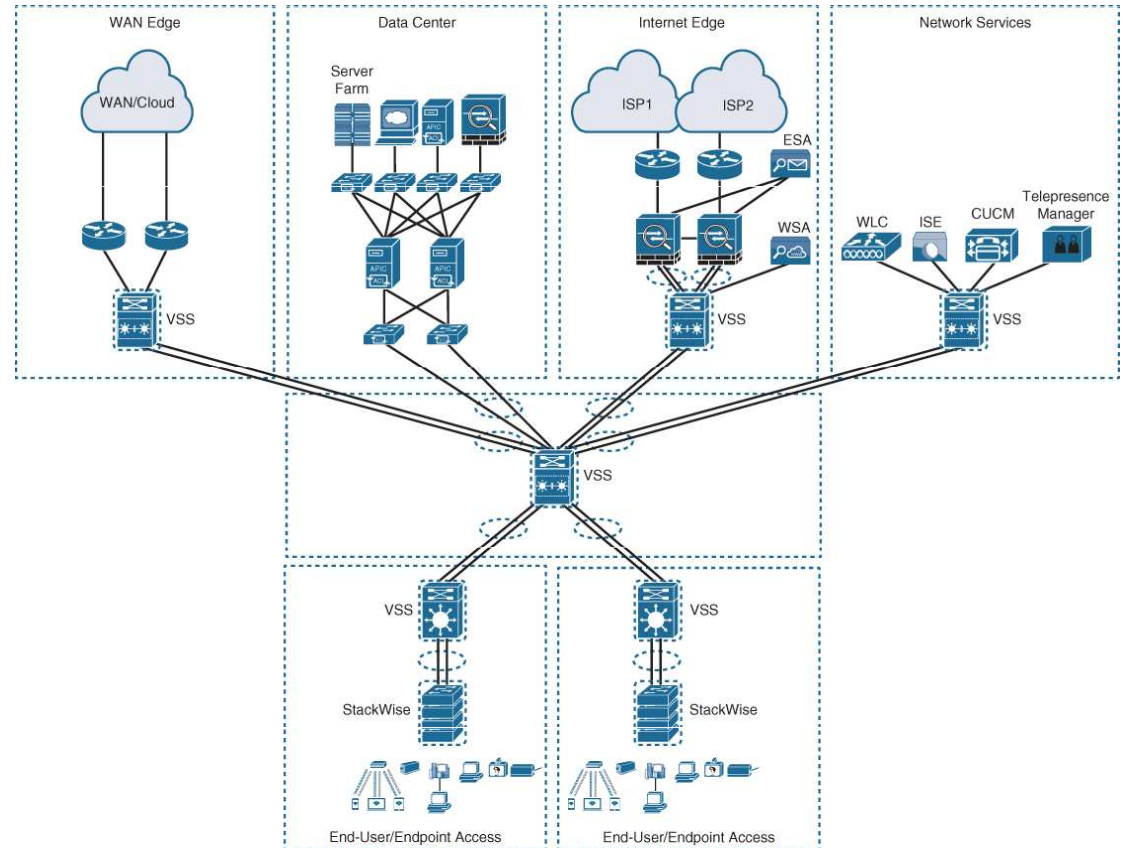


Figure 22-12 Applying VSS and StackWise in a Campus Network

## Software-Defined Access (SD-Access) Design

SD-Access - Industry's first intent-based networking solution for the enterprise

- Built on the principles of the Cisco Digital Network Architecture (DNA)
- A combination of the campus fabric design and the Digital Network Architecture Center (Cisco DNA or DNAC).
- Adds fabric capabilities to the enterprise network through automation
- Provides automated end-to-end segmentation to separate user, device, and application traffic without requiring a network redesign
- Fabric capabilities provide services such as host mobility and enhanced security in addition to the normal switching and routing capabilities.

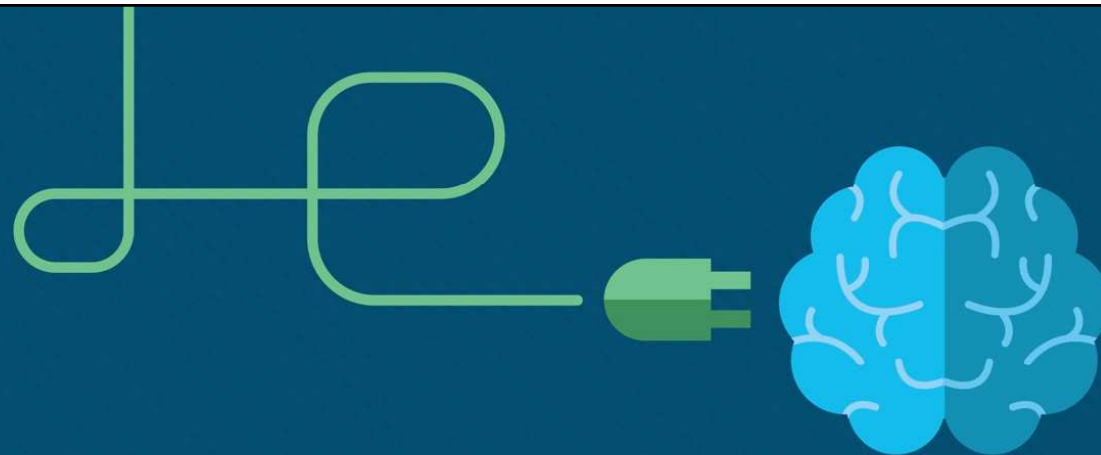




# Chapter 23: Fabric Technologies

Instructor Materials

CCNP Enterprise: Core Networking



# Software-Defined Access (SD-Access) - Cisco

- A fabric network is an overlay network (virtual network) built over an underlay network (physical network) using overlay tunneling technologies such as VXLAN (L2 to L4/UDP encapsulation). Fabric networks improve upon traditional physical networks by enabling host mobility, network automation, network virtualization, and segmentation. They are more manageable, flexible, secure (by means of encryption), and scalable than traditional networks.
- Manual network configuration changes are slow and can lead to misconfigurations that cause service disruptions on the network. The situation is exacerbated in a constantly changing environment where more users, endpoints, and applications are being added.
- The increase in users and endpoints makes configuring user credentials and maintaining a consistent policy across the network very complex.

# SD Access Features

With SD-Access, an evolved campus network can be built that addresses the needs of existing campus networks by leveraging the following capabilities, features, and functionalities:

**Network automation** - SD-Access replaces manual network device configurations with network device management through a single point of automation, orchestration, and management of network functions through the use of Cisco DNA Center.

**Network assurance and analytics** - SD-Access enables proactive prediction of network-related and security-related risks by using telemetry to improve the performance of the network, endpoints, and applications, including encrypted traffic.

**Host mobility** - SD-Access provides host mobility for both wired and wireless clients.

**Identity services** - Cisco Identity Services Engine (ISE) identifies users and devices connecting to the network and provides the contextual information required for users and devices to implement security policies for network access control and network segmentation.

# SD Access Features (Cont.)

**Policy enforcement** - Traditional access control lists (ACLs) can be difficult to deploy, maintain, and scale because they rely on IP addresses and subnets. Creating access and application policies based on group-based policies using Security Group Access Control Lists (SGACLs) provides a much simpler and more scalable form of policy enforcement based on identity instead of an IP address.

**Secure segmentation** - With SD-Access it is easier to segment the network to support guest, corporate, facilities, and IoT-enabled infrastructure.

**Network virtualization** - SD-Access makes it possible to leverage a single physical infrastructure to support multiple virtual routing and forwarding (VRF) instances, referred to as virtual networks (VNs), each with a distinct set of access policies.

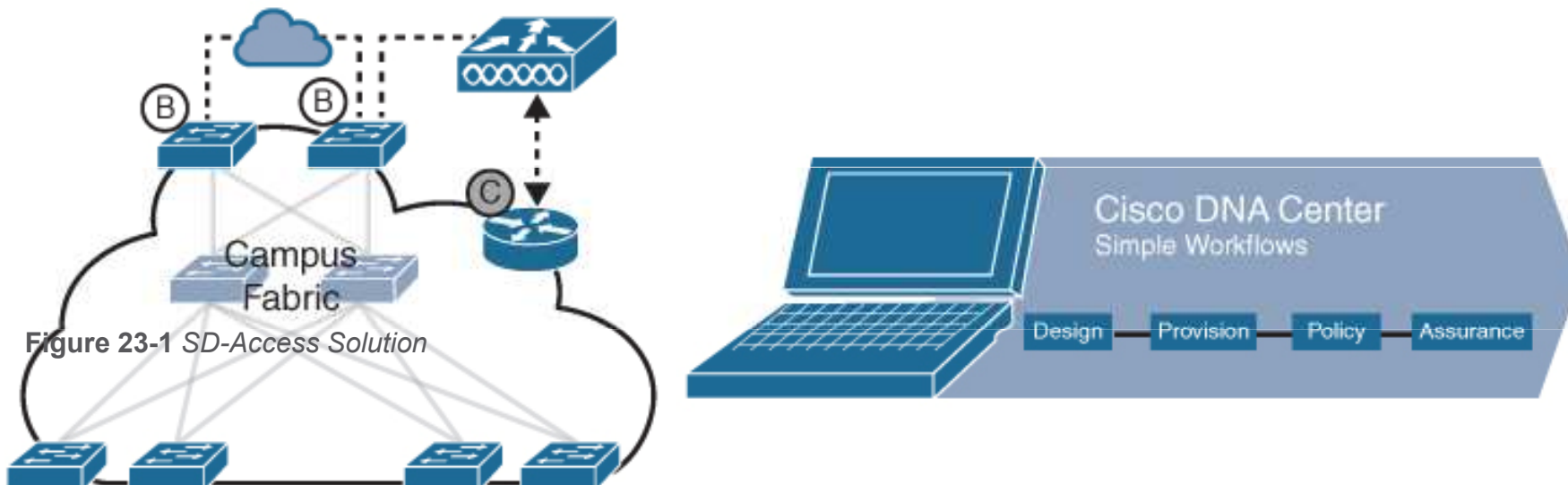
## Software-Defined Access (SD-Access)

# What is SD Access?

SD-Access has two main components:

- **Cisco Campus fabric solution** - The campus fabric is a Cisco-validated fabric overlay solution that includes all of the features and protocols (control plane, data plane, management plane, and policy plane) to operate the network infrastructure.
- **Cisco DNA Center** – management solution. When the campus fabric solution is managed via the Cisco DNA Center, the solution is considered to be SD-Access, as illustrated in Figure 23-1.

SD-Access = Campus Fabric + Cisco DNA Center



Software-Defined Access (SD-Access)

# What is SD Access Architecture

Cisco SD-Access is based on existing hardware and software technologies. What makes Cisco SD-Access special is how these technologies are integrated and managed together. The Cisco SD-Access fabric architecture can be divided into four basic layers, as illustrated in Figure 23-2.

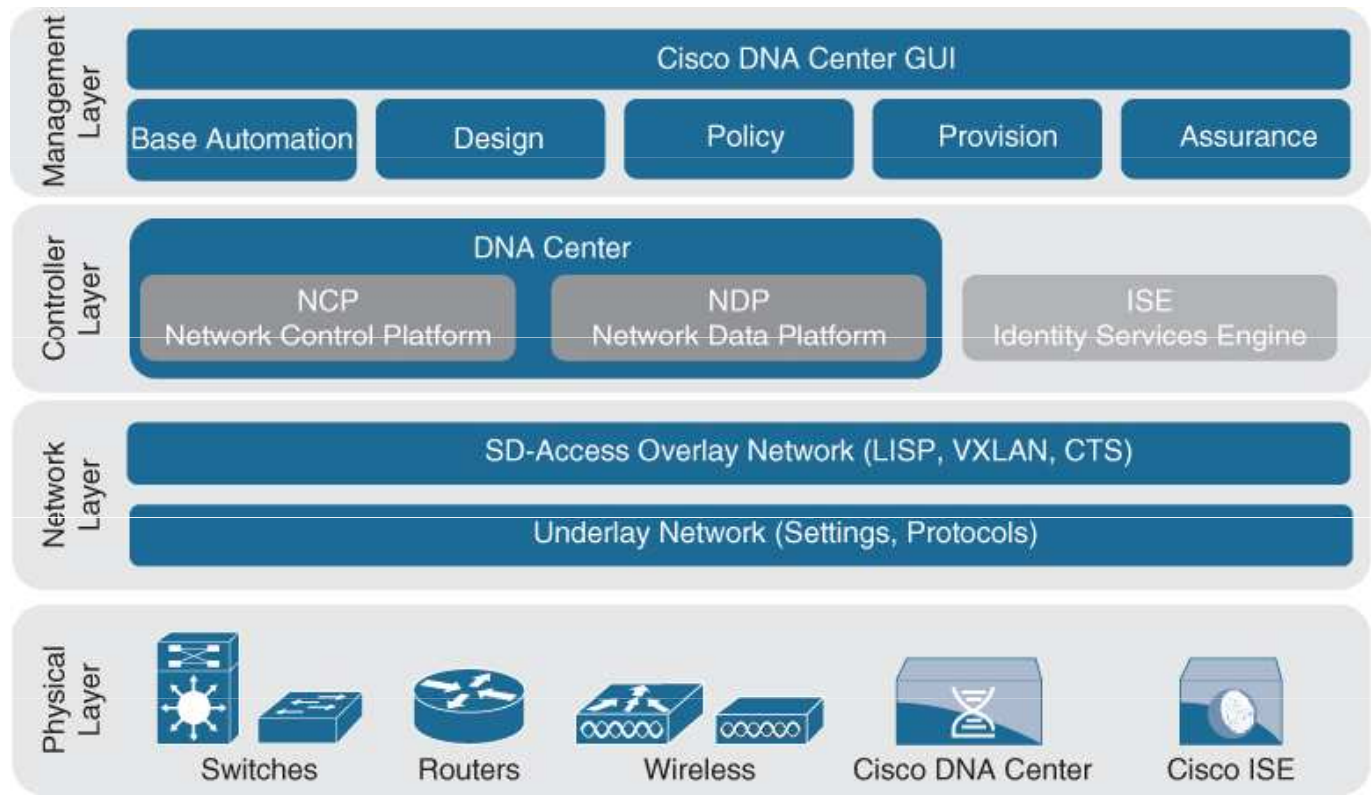


Figure 23-2 Cisco SD-Access Architecture

# Physical Layer

While Cisco SD-Access is designed for user simplicity, abstraction, and virtual environments, everything runs on top of physical network devices (switches, routers, servers, wireless LAN controllers (WLCs), and wireless access points (APs). Cisco access layer switches that do not actively participate in the SD-Access fabric but that are part of it because of automation are referred to as SD-Access extension nodes.

The following are the physical layer devices of the SD-WAN fabric:

**Cisco switches** - Switches provide wired (LAN) access to the fabric. Multiple types of Cisco Catalyst switches are supported, as well as Nexus switches.

**Cisco routers** - Routers provide WAN and branch access to the fabric. Multiple types of Cisco ASR 1000, ISR, and CSR routers, including the CSRv and ISRv cloud routers, are supported.

**Cisco wireless** - Cisco WLCs and APs provide wireless (WLAN) access to the fabric.

**Cisco controller appliances** - Cisco DNA Center and Cisco ISE are the two controller appliances required.

# Software-Defined Access (SD-Access) Network Layer

The network layer consists of the underlay network and the overlay network. These two sublayers work together to deliver data packets to and from the network devices participating in SD-Access. All this network layer information is made available to the controller layer.

The **network underlay** is the **underlying physical layer**, and its sole purpose is to transport data packets between network devices for the SD-Access fabric overlay.

The **overlay network** is a **virtual (tunneled) network** that virtually interconnects all of the network devices forming a fabric of interconnected nodes. **It abstracts the inherent complexities and limitations of the underlay network.**

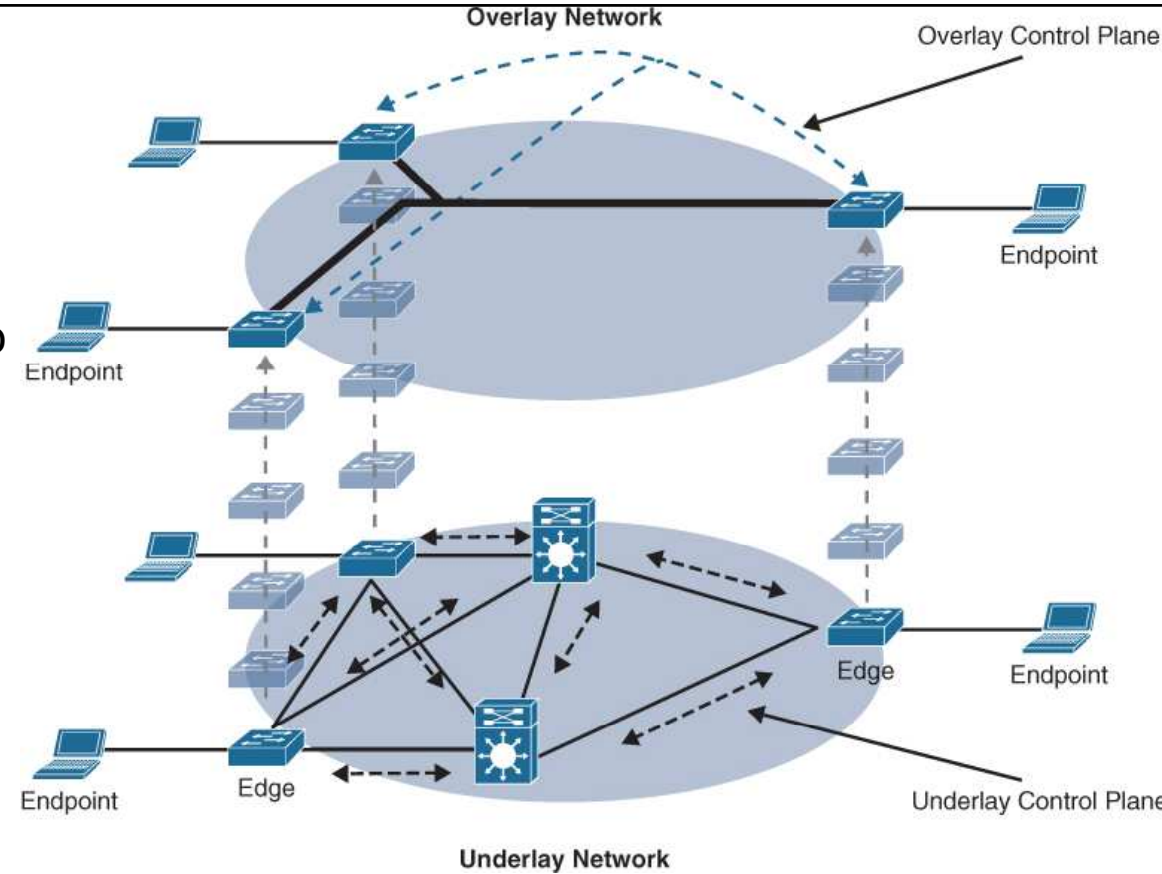


Figure 23-3 Underlay and Overlay Networks

Figure 23-3 shows a visual representation of the relationship between an overlay network and the network underlay.



# Underlay Network

Two models of underlay that are supported are as follows:

**Manual underlay** - This type of underlay network is configured and managed **manually** (such as with a CLI or an API) rather than through Cisco DNA Center. An advantage of the manual underlay is that it allows customization of the network to fit any special design requirements (such as changing the IGP to OSPF). It allows SD-Access to run on the top of a legacy (or third-party) IP-based network.

**Automated underlay** - In a fully automated network underlay, all aspects of the underlay network are **configured and managed by the Cisco DNA Center LAN Automation feature**. The LAN Automation feature creates an IS-IS routed access campus design. It uses the Cisco Network Plug and Play features to deploy both unicast and multicast routing configuration in the underlay to improve traffic delivery efficiency for SD-Access. An automated underlay eliminates misconfigurations and reduces the complexity of the network underlay. It also greatly simplifies and speeds the building of the network underlay. A downside to an automated underlay is that it does not allow manual customization for special design requirements.

Software-Defined Access (SD-Access)

# Overlay Network (SD-Access Fabric)

The SD-Access fabric is the overlay network, and it provides **policy-based network segmentation**, **host mobility for wired and wireless hosts**, and **enhanced security** beyond the normal switching and routing capabilities of a traditional network.

In SD-Access, the fabric overlay is fully automated, regardless of the underlay network model used (manual or automated). It includes all necessary overlay control plane protocols and addressing, as well as all global configurations associated with operation of the SD-Access fabric.

There are three basic planes of operation in the SD-Access fabric:

- **Control plane**, based on **Locator/ID Separation Protocol (LISP)**
- **Data plane**, based on **Virtual Extensible LAN (VXLAN)**
- **Policy plane**, based on **Cisco TrustSec**

# SD-Access Control Plane

The SD-Access fabric control plane is based on Locator/ID Separation Protocol (LISP).

- LISP is an IETF standard protocol defined in RFC 6830 that is based on a **simple endpoint ID (EID) to routing locator (RLOC) mapping system** to **separate the identity** (*endpoint IP address*) **from its current location** (*network edge/border router IP address*).
- LISP dramatically simplifies traditional routing environments by **eliminating the need for each router to process every possible IP destination address and route**. It does this by **moving remote destination information to a centralized mapping database** called the **LISP map server (MS)** (*a control plane node in SD-Access*), which **allows each router**
  - to manage only its local routes and
  - query the map system to locate destination EIDs.
- This technology provides many advantages for Cisco SD-Access, such as smaller routing tables, dynamic host mobility for wired and wireless endpoints, address-agnostic mapping (IPv4, IPv6, and/ or MAC), and built-in network segmentation through VRF instances.

Software-Defined Access (SD-Access)

# SD-Access Fabric Data Plane

The **tunneling technology** used for the fabric data plane is based on **Virtual Extensible LAN (VXLAN)**. **VXLAN encapsulation is IP/UDP based**, meaning that it **can be forwarded by any IP-based network** (legacy or third party) and creates the overlay network for the SD-Access fabric.

Although LISP is the control plane for the SD-Access fabric, it does not use LISP data encapsulation for the data plane.

The differences between the LISP and VXLAN packet formats are illustrated in Figure 23-4.

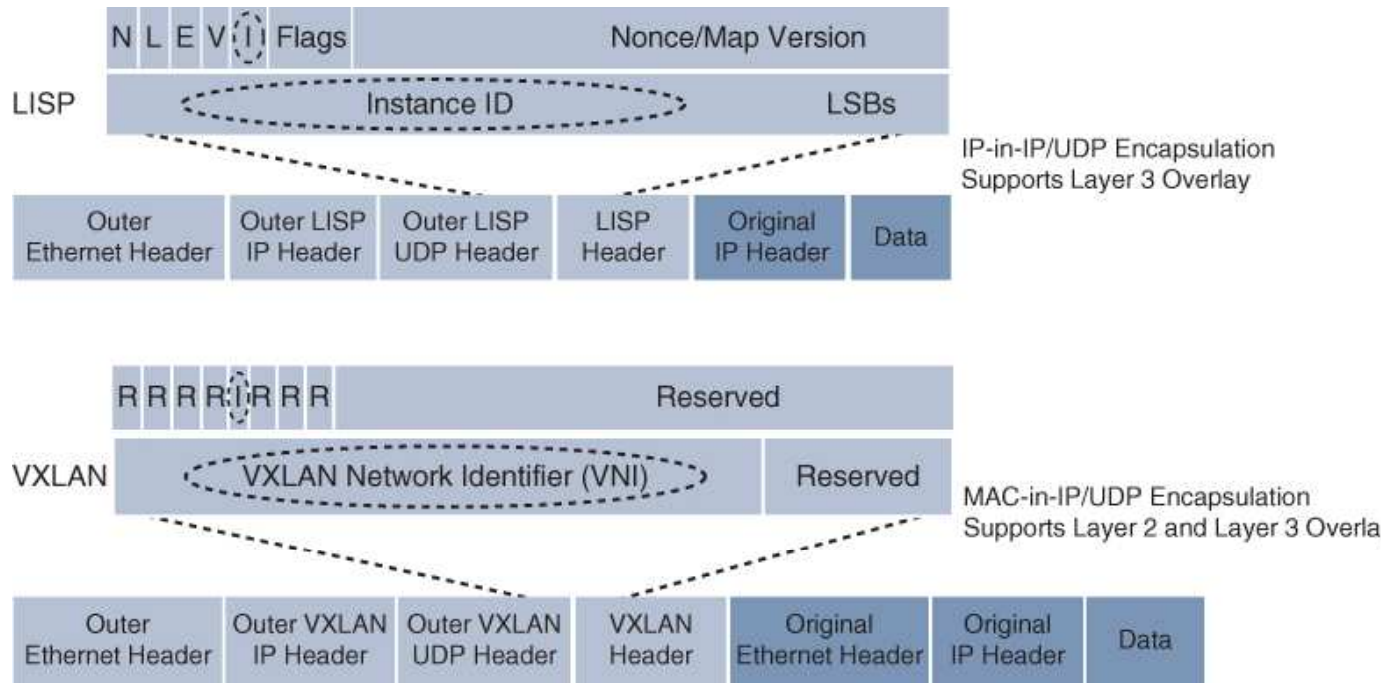


Figure 23-4 LISP and VXLAN Packet Format Comparison



## SD-Access Fabric Plane (Cont.)

The original VXLAN specification was enhanced for SD-Access to support Cisco TrustSec Scalable Group Tags (SGTs). This added **new fields** to the first 4 bytes of the **VXLAN header** in order to transport up to 64,000 SGT tags. The new VXLAN format is called VXLAN Group Policy Option (VXLAN-GPO), and it is defined in the IETF draft draft-smith-vxlan-group-policy-05. The new fields in the VXLAN-GPO packet format include the following:

- **Group Policy ID** - 16-bit identifier that is used to carry the SGT tag.
- **Group Based Policy Extension Bit (G Bit)** - 1-bit field that, when set to 1, indicates an SGT tag is being carried within the Group Policy ID field and is set to 0 when it is not.
- **Don't Learn Bit (D Bit)** - 1-bit field that when set to 1 indicates that the egress virtual tunnel endpoint (VTEP) must not learn the source address of the encapsulated frame.
- **Policy Applied Bit (A Bit)** - 1-bit field that is only defined as the A bit when the G bit field is set to 1. When the A bit is set to 1, it indicates that the group policy has already been applied to this packet, and further policies must not be applied by network devices. When it is set to 0, group policies must be applied by network devices, and they must set the A bit to 1 after the policy has been applied.

# SD-Access Fabric Policy Plane

The fabric policy plane is based on Cisco TrustSec. Cisco TrustSec SGT tags are assigned to authenticated groups of users or end devices. Network policy (for example, ACLs, QoS) is then applied throughout the SD-Access fabric, based on the SGT tag instead of a network address (MAC, IPv4, or IPv6).

TrustSec SGT tags provide several advantages for Cisco SD-Access, such as:

- Support for both network-based segmentation using VNs (VRF instances) and group-based segmentation (policies)
- Network address-independent group-based policies based on SGT tags rather than MAC, IPv4, or IPv6 addresses, which reduces complexity
- Dynamic enforcement of group-based policies, regardless of location for both wired and wireless traffic
- Policy constructs over a legacy or third-party network using VXLAN
- Extended policy enforcement to external networks (such as cloud or data center networks) by transporting the tags to Cisco TrustSec-aware devices using SGT Exchange Protocol (SXP)

# Software-Defined Access (SD-Access)

## SD-Access Fabric Roles and Components

The operation of the SD-Access fabric requires multiple different device roles, each with a specific set of responsibilities. Each SD-Access-enabled network device must be configured for one (or more) of the five basic device roles in the fabric overlay:

- Control plane node
- Fabric border node
- Fabric edge node
- Fabric WLAN controller (WLC)
- Intermediate nodes

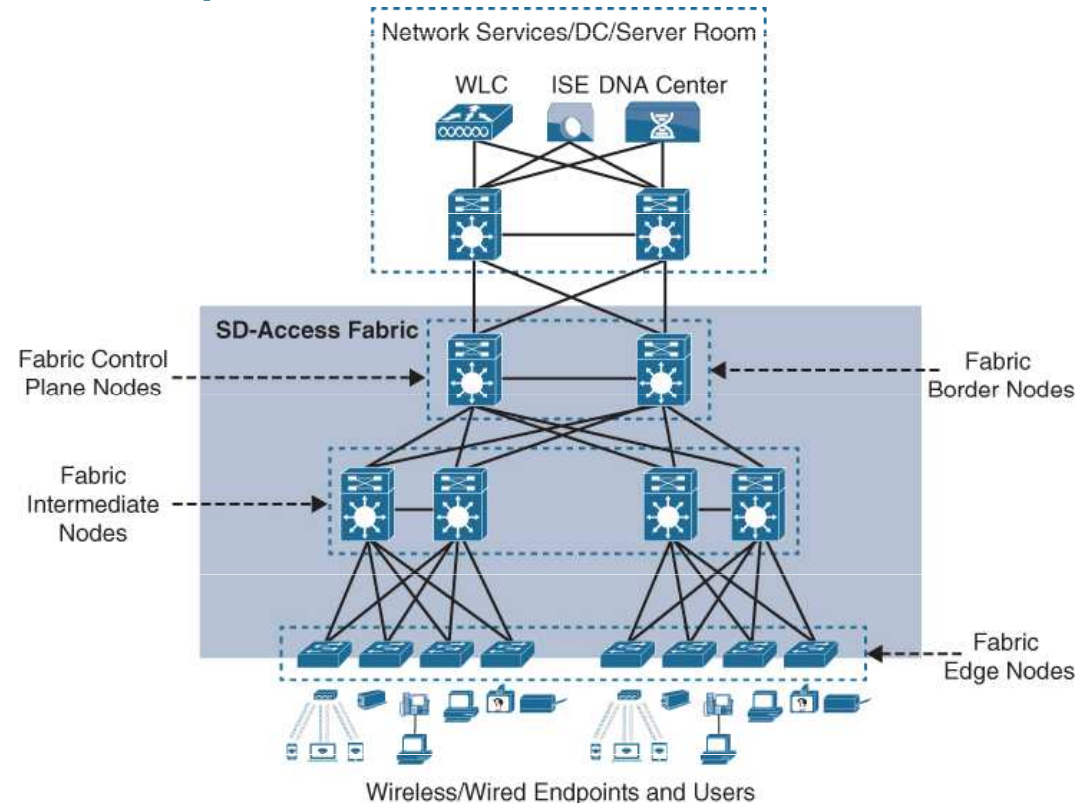


Figure 23-6 SD-Access Fabric Roles

Figure 23-6 illustrates the different SD-Access fabric design roles and how nodes in the fabric can play multiple roles. For example, the core layer routers in this figure are acting as fabric border nodes and control plane nodes.

## Software-Defined Access (SD-Access)

# Cisco DNA Policy Workflow

The Cisco DNA policy workflow provides all the tools to logically define Cisco DNA policies. The following are some of the Cisco DNA policy tools:

- **Dashboard** - Used to monitor all the VNs, scalable groups, policies, and recent changes.
- **Group-Based Access Control** - Used to create group-based access control policies, which are the same as SGACLs. Cisco DNA Center integrates with Cisco ISE to simplify the process of creating and maintaining SGACLs.
- **IP-Based Access Control** - Used to create IP-based access control policy to control the traffic going into and coming out of a Cisco device in the same way that an ACL does.
- **Application** - Used to configure QoS in the network through application policies.
- **Traffic Copy** - Used to configure Encapsulated Remote Switched Port Analyzer (ERSPAN) to copy the IP traffic flow between two entities to a specified remote destination for monitoring or troubleshooting purposes.
- **Virtual Network** - Used to set up the virtual networks (or use the default VN) and associate various scalable groups



## Software-Defined Access (SD-Access) Cisco DNA Provision Workflow

The Cisco DNA provision workflow provides all the tools to deploy the Cisco SD-Access fabric. The following are some of the Cisco DNA provision tools:

- **Devices** - Used to assign devices to a site ID, confirm or update the software version, and provision the network underlay configurations.
- **Fabrics** - Used to set up the fabric domains (or use the default LAN fabric).
- **Fabric Devices** - Used to add devices to the fabric domain and specify device roles (such as control plane, border, edge, and WLC).
- **Host Onboarding** - Used to define the host authentication type (static or dynamic) and assign host pools (wired and wireless) to various VNs.

# Software-Defined WAN (SD-WAN)

Managing enterprise networks is becoming more complex, with customers embracing a multicloud approach, applications moving to the cloud, mobile and IoT devices growing exponentially in the network, and the internet edge moving to the branch. This digital transformation is powering the adoption of SD-WAN by customers looking to do the following:

- Lower costs and reduce risks with simple WAN automation and orchestration.
- Extend their enterprise networks (such as branch or on-premises) seamlessly into the public cloud.
- Provide optimal user experience for SaaS applications.
- Leverage a transport-independent WAN for lower cost and higher diversity. This means the underlay network can be any type of IP-based network, such as the internet, MPLS, 3G/4G LTE, satellite, or dedicated circuits.
- Enhance application visibility and use that visibility to improve performance with intelligent path control to meet SLAs for business-critical and real-time applications.
- Provide end-to-end WAN traffic segmentation and encryption for protecting critical enterprise compute resources.

## Software-Defined WAN (SD-WAN) SD-WAN Principle

- SD-WAN is an acronym for software-defined networking (SDN) in a wide area network (WAN).
- SD-WAN simplifies the management and operation of a WAN by decoupling the networking hardware from its control mechanism.
- It allows companies to build higher-performance WANs using lower-cost and commercially available Internet access, enabling businesses to partially or wholly replace more expensive private WAN connection technologies such as MPLS.

## Software-Defined WAN (SD-WAN) SD-WAN Solutions

### Cisco currently offers two SD-WAN solutions:

- **Cisco SD-WAN (based on Viptela)** - This is the preferred solution for organizations that require an SD-WAN solution with cloud-based initiatives that provides granular segmentation, advanced routing, advanced security, and complex topologies while connecting to cloud instances.
- **Meraki SD-WAN** - This is the recommended solution for organizations that require unified threat management (UTM) solutions with SD-WAN functionality or that are existing Cisco Meraki customers looking to expand to SD-WAN. UTM is an all-in-one security solution delivered in a single appliance and typically includes the following security features: firewall, VPN, intrusion prevention, antivirus, antispam, and web content filtering.

# Software-Defined WAN (SD-WAN) Cisco SD-WAN Architecture

Cisco SD-WAN (based on Viptela - acquired in 2017) is a cloud-delivered overlay WAN architecture that facilitates digital and cloud transformation for enterprises. It addresses all the customer requirements mentioned earlier.

Figure 23-13 shows how SD-WAN can be used to provide secure connectivity to remote offices, branch offices, campus networks, data centers, and the cloud over any type of IP-based underlay transport network, such as the internet, 3G/4G LTE, and MPLS. It also illustrates how some of the components to manage the SD-WAN fabric can be deployed on a data center, private cloud, or public cloud.

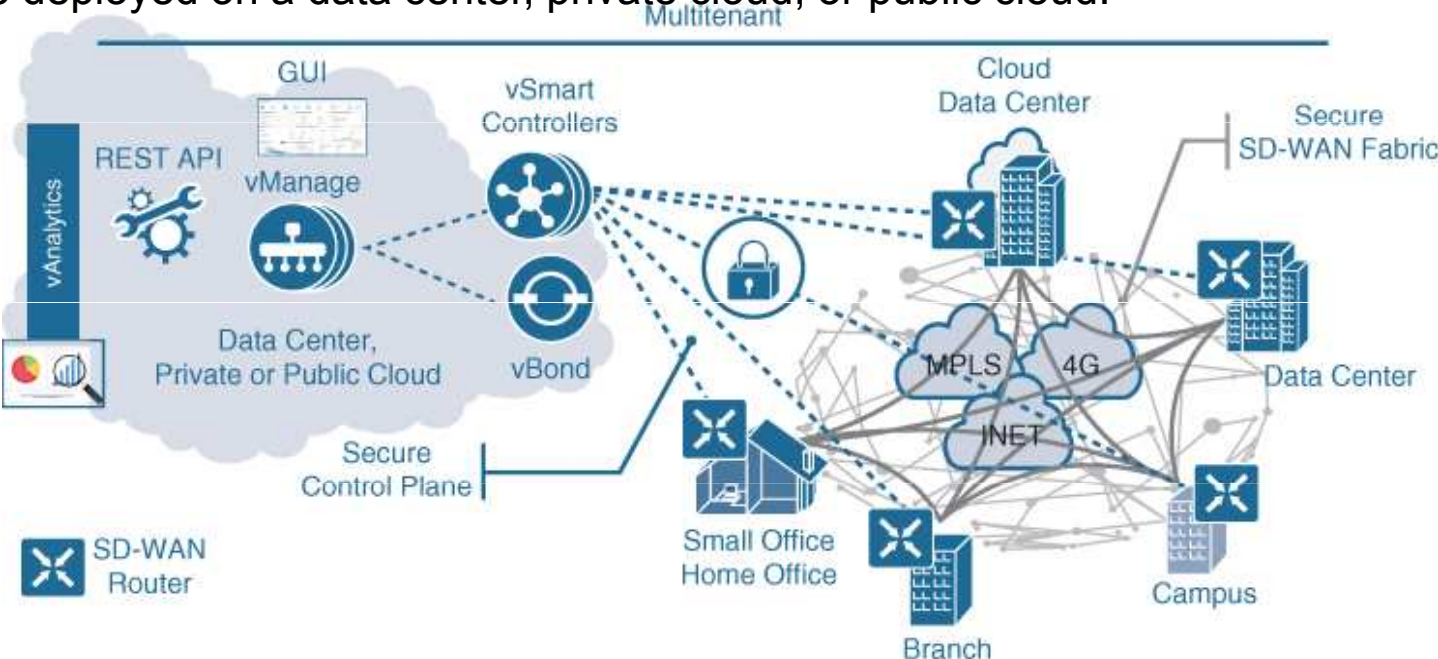


Figure 23-13 SD-WAN Solution Architecture



## Software-Defined WAN (SD-WAN) Cisco SD-WAN Solution

The Cisco SD-WAN solution has four main components and an optional analytics service:

- **vManage NMS** - The vManage NMS is a GUI that is used to configure and manage the full SD-WAN solution. It enables centralized provisioning and simplifies network changes.
- **vSmart Controller** - vSmart controllers have pre-installed credentials that allow them to authenticate every SD-WAN router that comes online. These credentials ensure that only authenticated devices are allowed access to the SD-WAN fabric. After successful authentication, each vSmart controller establishes a permanent DTLS tunnel to each SD-WAN router in the SD-WAN fabric and uses these tunnels to establish Overlay Management Protocol (OMP) neighbor relationships with each SD-WAN router. OMP is a proprietary routing protocol similar to BGP that can advertise routes, next hops, keys, and policy information needed to establish and maintain the SD-WAN fabric.

## Software-Defined WAN (SD-WAN) Cisco SD-WAN Solution (Cont.)

**Cisco SD-WAN Routers (vEdge and cEdge):** Cisco SD-WAN routers deliver the essential WAN, security, and multicloud capabilities of the Cisco SD-WAN solution. They are available as hardware, software, cloud, or virtualized routers that sit at the perimeter of a site, such as a remote office, branch office, campus, or data center. SD-WAN routers support standard router features, such as OSPF, BGP, ACLs, QoS, and routing policies, in addition to the SD-WAN overlay control and data plane functions.

There are two different SD-WAN router options available for the Cisco SD-WAN solution:

- **vEdge:** The original Viptela platforms running Viptela software.
- **cEdge:** Viptela software integrated with Cisco IOS-XE. This is supported on CSR, ISR, ASR1K, ENCS, and the cloud-enabled CSRv and ISRv platforms.

## Software-Defined WAN (SD-WAN) vBond Orchestrator

The vBond orchestrator **authenticates** the **vSmart controllers** and the **SD-WAN routers** and **orchestrates connectivity between them**. It is the **only device that must have a public IP address** so that all SD-WAN devices in the network can connect to it. A vBond orchestrator is an SD-WAN router that only performs vBond orchestrator functions.

The major components of the vBond orchestrator are:

- **Control plane connection** - Each vBond orchestrator has a permanent control plane connection over a DTLS tunnel with each vSmart controller.
- **NAT traversal** - The vBond orchestrator facilitates the initial orchestration between SD-WAN routers and vSmart controllers when one or both of them are behind NAT devices.
- **Load balancing** - In a domain with multiple vSmart controllers, the vBond orchestrator automatically performs load balancing of SD-WAN routers across the vSmart controllers when routers come online.

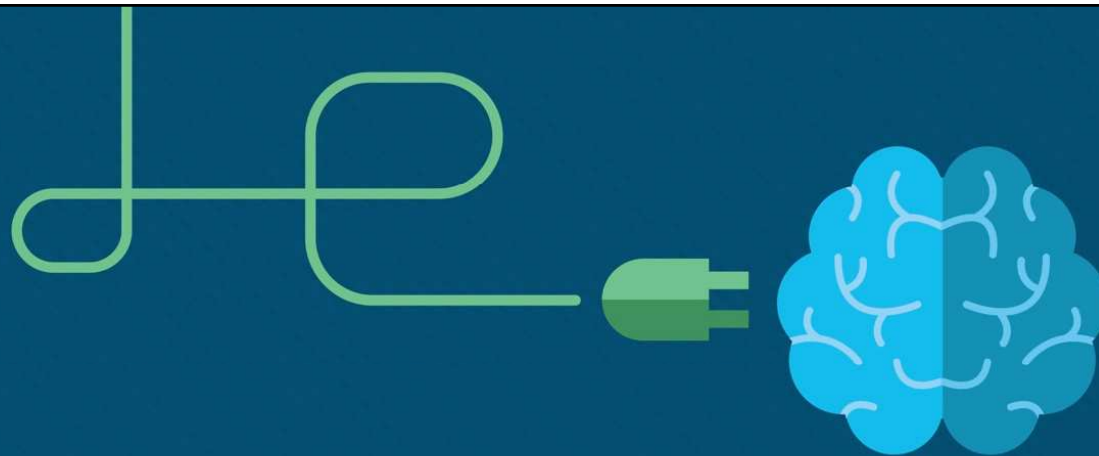


## Software-Defined WAN (SD-WAN) vAnalytics

vAnalytics is an optional analytics and assurance service that has many advanced capabilities, including the following:

- Visibility into applications and infrastructure across the WAN
- Forecasting and what-if analysis
- Intelligent recommendations

These capabilities benefit SD-WAN in ways that are not possible without v Analytics. For example, if a branch office is experiencing latency or loss on its MPLS link, vAnalytics detects it and compare that latency or loss with information on other organizations in the area that it is also monitoring to see if they are also having that same issue in their circuits. If they are, vAnalytics can then report the issue with confidence to the SPs. vAnalytics can also help predict how much bandwidth is truly required for any location. This aids in deciding whether a circuit can be downgraded to a lower bandwidth to reduce costs.



# Chapter 24: Network Assurance

Instructor Materials

CCNP Enterprise: Core Networking



# Network Diagnostic Tools

Many network diagnostic tools are readily available. This section covers some of the most common tools available and provides use cases and examples of when to use them.

# Ping

Ping is one of the most useful troubleshooting.

The following troubleshooting flow is a quick way to check reachability and try to determine what the issue may be:

**Step 1.** Gather the facts.

**Step 2.** Test reachability by using the **ping** command.

**Step 3.** Record the outcome of the **ping** command and move to the next troubleshooting step. If **ping** is successful, then the issue isn't likely related to basic reachability. If **ping** is unsuccessful, the next step could be checking something more advanced, such as interface issues, routing issues, access lists, or intermediate firewalls.

# Successful and Unsuccessful Pings

Example 24-1 illustrates a successful and 24-2 an unsuccessful ping between R1 and R2. This example shows five 100-byte ICMP echo request packets sent to 10.1.12.2 with a 2-second timeout. The results are five exclamation points (!!!!!) or periods (.....).

**Example 24-1** Successful ping Between R1 and R2

**Example 24-2** Unsuccessful ping Between R1 and R2

```
R1# ping 10.1.12.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.12.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

```
R1# ping 10.1.12.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.12.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Result	Meaning
!	Each exclamation point indicates receipt of a reply.
.	Each period indicates the network server timed out while waiting for a reply.
U	A destination unreachable error PDU was received.
I	User interrupt (if you hit CTRL+^)
A	Access list denied packet
H	Host unreachable
P	Protocol unreachable
N	Network unreachable
Q	Source quench (destination too busy).
M	Could not fragment.
?	Unknown packet type.
&	Packet lifetime exceeded



## Network Diagnostic Tools

# Ping Options

The ping command can be changed and manipulated to aid in troubleshooting. Example 24-3 shows some of the available options for the **ping** command on a Cisco device. These options can be seen by using the context sensitive help (?) after the IP address that follows the ping command. This section specifically focuses on the **repeat**, **size**, and **source** options.

### Example 24-3 ping 10.1.12.2 Options

```
R1# ping 10.1.12.2 ?
data      specify data pattern
df-bit    enable do not fragment bit in IP header
repeat    specify repeat count
size      specify datagram size
source    specify source address or name
timeout   specify timeout interval
tos       specify type of service value
validate  validate reply data
<cr>
```

## Network Diagnostic Tools

# Ping Size

Another very common use case for the **ping** command is to send different sizes of packets to a destination. An example might be to send 1500-byte packets with the DF bit set to make sure there are no MTU issues on the interfaces or to test different quality of service policies that restrict certain packet sizes. Example 24-5 shows a ping destined to R2's Ethernet0/0 interface with an IP address 10.1.12.2 and a packet size of 1500 bytes.

### Example 24-5 ping 10.1.12.2 size 1500 Command

```
R1# ping 10.1.12.2 size 1500
Type escape sequence to abort.
Sending 5, 1500-byte ICMP Echos to 10.1.12.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

## Network Diagnostic Tools

# Extended Ping Options

An extended ping can take advantage of the same options already discussed, as well as some more detailed options for troubleshooting. These options are listed in Table 24-2.

Option	Description
Protocol	IP, Novell, AppleTalk, CLNS, and so on; the default is IP
Target IP address	Destination IP address of ping packets
Repeat Count	Number of ping packets sent; the default is 5 packets
Datagram Size	Size of the ping packet; the default is 100 bytes
Timeout in seconds	How long a echo reply response is waited for
Extended Commands	Yes or No to use extended commands; the default is No, but if Yes is used, more options become available
Source Address or Interface	IP address of the source interface or the interface name



## Network Diagnostic Tools

# Extended Ping Options (Cont.)

Option	Description
Type of Service (ToS)	The Type of Service to be used for each probe; 0 is the default
Set DF bit in IP header	Sets the Do Not Fragment bit in the IP header; the default is No
Data Pattern	The data pattern used in the ping packets; the default is 0xABCD
Loose, Strict, Record, Timestamp, Verbose	The options set for the ping packets: <b>Loose</b> - Specifies hops that ping packets should traverse <b>Strict</b> - Same as Loose with the exception that packets can only traverse specified hops <b>Record</b> - Displays IP addresses of first nine hops that the ping packets traverse <b>Timestamp</b> - Displays the round-trip time to the destination for each ping <b>Verbose</b> - Default option that is automatically selected with any and all other options

# Extended Ping with Multiple Options

Setting the **MTU** in an extended ping and setting the DF bit in the IP header can help determine whether there are MTU settings in the path that are not set appropriately.

With tunneling it is important to account for the overhead of the tunnel technology, which can vary. Specifying a **Type of Service** of **184** in decimal translates to **Expedited Forwarding** or (EF) **per-hop behavior** (PHB). This can be useful **when testing real-time quality of service** (QoS) policies.

Setting **Data Patterns** can help when troubleshooting framing errors, line coding, or clock signaling issues on serial interfaces.

Finally, a **timestamp** is set in this example, in addition to the default Verbose output. This gives a clock timestamp of when the destination sent an echo reply message back to the source.

### Example 24-7 *Extended ping with Multiple Options*

```
R1# ping
Protocol [ip]:
Target IP address: 22.22.22.23
Repeat count [5]: 1
Datagram size [100]: 1500
Timeout in seconds [2]: 1
Extended commands [n]: yes
Source address or interface: Loopback101
Type of service [0]: 184
Set DF bit in IP header? [no]: yes
Validate reply data? [no]:
Data pattern [0xABCD]: 0xABBA
Loose, Strict, Record, Timestamp, Verbose[none]: Timestamp
Number of timestamps [ 9 ]: 3
```

## Network Diagnostic Tools

# Traceroute

**Traceroute** is often used to troubleshoot when trying to determine where traffic is failing as well as what path traffic takes through-out the network.

**Traceroute** shows the IP addresses or DNS names of the hops between the source and destination.

It also shows how long it takes to reach the destination at each hop, measured in milliseconds. This tool is frequently used when more than one path is available to the destination or when there is more than one hop to the destination.

### **Example 24-8** *Basic traceroute to R2 Loopback102*

```
R1# traceroute 22.22.22.22
Type escape sequence to abort.
Tracing the route to 22.22.22.22
VRF info: (vrf in name/id, vrf out name/id)
 1 10.1.12.2 0 msec * 1 msec
```

## Adding a Less Specific Route

A less specific route is added to R1 that points to 22.0.0.0/8 or 22.0.0.0 255.0.0.0, the traceroute returns a “host unreachable” message. This is because there is a route to the next hop, R2 (10.1.12.2), but once the traceroute gets to R2, there is no interface or route to 22.22.22.23/32, and the traceroute fails. Example 24-11 shows this scenario.

### **Example 24-11** *Adding a Less Specific Route on R1*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip route 22.0.0.0 255.0.0.0 10.1.12.2
R1(config)# end
R1# traceroute 22.22.22.23
Type escape sequence to abort.
Tracing the route to 22.22.22.23
VRF info: (vrf in name/id, vrf out name/id)
 1 10.1.12.2 0 msec 0 msec 0 msec
 2 10.1.12.2 !H * !H
```

## Network Diagnostic Tools

# Traceroute Options

Another benefit of **traceroute** is that it has options available. These options can also be discovered by leveraging the context-sensitive help (?) from the command-line interface. Example 24-14 shows the list of available options to the traceroute command.

### Example 24-14 Available traceroute Options

```
R1# traceroute 22.22.22.23 ?
  numeric  display numeric address
  port     specify port number
  probe    specify number of probes per hop
  source   specify source address or name
  timeout  specify time out
  ttl      specify minimum and maximum ttl
<cr>
```

There are times when using some of the options available with **traceroute** may be . There are also times when there might be a reason to send a different number of probes per hop with different timeout timers rather than the default of three probes.

## Network Diagnostic Tools

# Extended Traceroute Options

Much like the extended **ping** command covered earlier in this chapter, there is an extended **traceroute** command, and it has a number of detailed options available. Those options are listed in Table 24-3.

Option	Description
Protocol	IP, Novell, AppleTalk, CLNS, and so on; the default is IP
Target IP address	Destination IP address of ping packets
Numeric display	Shows only the numeric display rather than numeric and symbolic display
Timeout in Seconds	Time that is waited for a reply to a probe; the default is 3 seconds
Probe count	Number of probes sent at each hop; the default is 3
Source Address	IP address of the source interface

## Extended Traceroute Options (Cont.)

Option	Description
Minimum Time-to-live	TTL value of the first set of probes; can be used to hide topology information or known hops
Maximum Time-to-live	Maximum number of hops; the default is 30
Port number	Destination port number of probes; the default is 33434
Loose, Strict, Record, Timestamp, Verbose	The options set for the traceroute probes: <b>Loose</b> - Specifies the hops that ping packets should traverse <b>Strict</b> - Same as Loose with the exception that packets can traverse only specified hops <b>Record</b> - Displays IP addresses of the first nine hops that the traceroute packets traverse <b>Timestamp</b> - Displays the round-trip time to the destination for each ping <b>Verbose</b> - Default option that is automatically selected with any and all other options

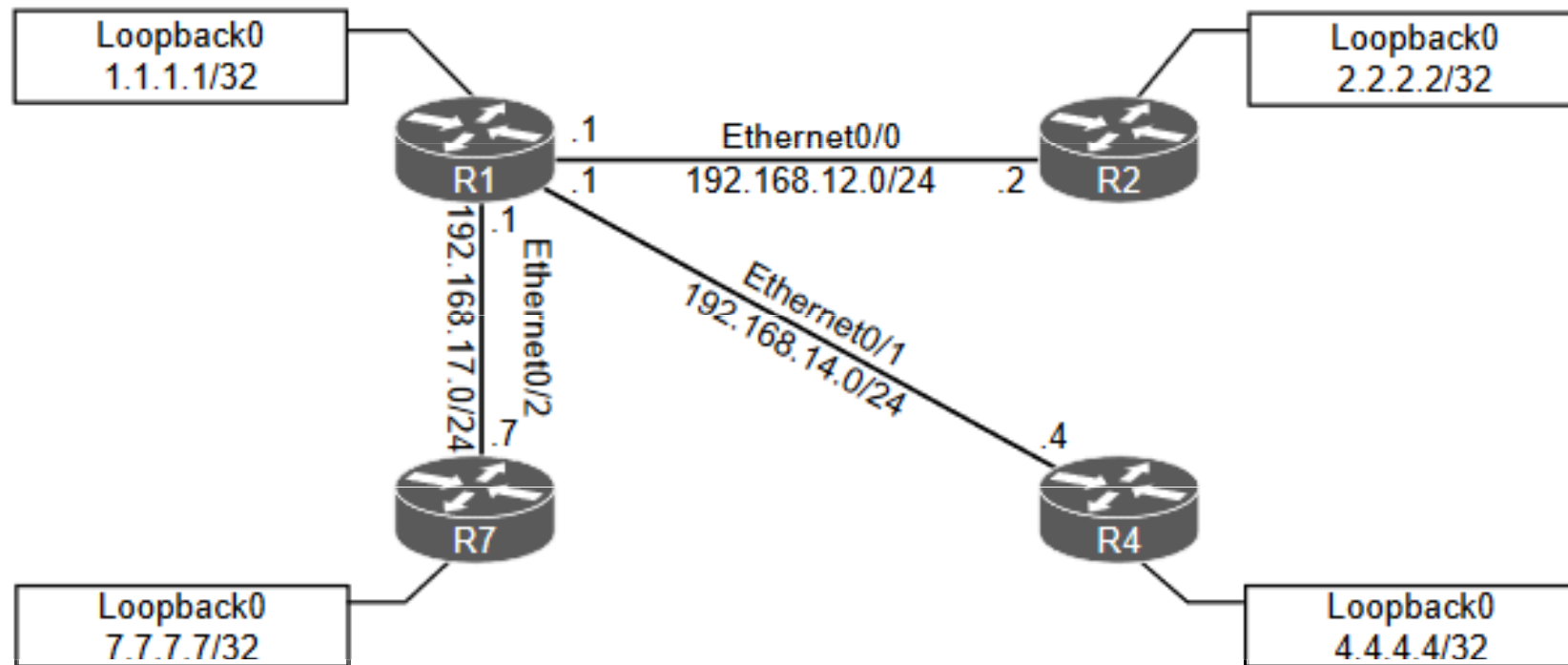
# Debugging

- Debugging can be a very powerful part of troubleshooting complex issues in a network.
- One of the most common use cases for debugging is when there is a need to see things at a deeper level (such as when routing protocols are having adjacency issues).
- There is a normal flow that is taken from a troubleshooting perspective, depending on the routing protocol.



# Debugging Debugging Topology

Using the simple topology shown in Figure 24-2, in this section, debugging is used to fix a couple issues in the OSPF area 0.



**Figure 24-2** Debugging Topology

## Debugging OSPF Adjacency Issues

Some of the common OSPF **adjacency issues** such as **MTU** issues, **incorrect interface types**, and **improperly configured network mask** can be resolved by using debugging.

From the output of the **show ip ospf neighbor** command on R1 in Example 24-17, it can be seen that the neighbor adjacency to R4 is in the **INIT** state. If the command is run after a few seconds, the state changes to **EXCHANGE** but quickly cycles back to the **INIT** state when the command is run again

**Example 24-17** Output of the show ip ospf neighbor Command

```
R1# show ip ospf neighbor

Neighbor ID    Pri   State           Dead Time   Address      Interface
7.7.7.7        0     FULL/ -         00:00:31   192.168.17.7 Ethernet0/2
4.4.4.4        0     INIT/ -         00:00:37   192.168.14.4 Ethernet0/1
2.2.2.2        0     FULL/ -         00:00:33   192.168.12.2 Ethernet0/0
R1# show ip ospf neighbor

Neighbor ID    Pri   State           Dead Time   Address      Interface
7.7.7.7        0     FULL/ -         00:00:33   192.168.17.7 Ethernet0/2
4.4.4.4        0     EXCHANGE/ -       00:00:37   192.168.14.4 Ethernet0/1
2.2.2.2        0     FULL/ -         00:00:32   192.168.12.2 Ethernet0/0
R1# show ip ospf neighbor

Neighbor ID    Pri   State           Dead Time   Address      Interface
7.7.7.7        0     FULL/ -         00:00:31   192.168.17.7 Ethernet0/2
4.4.4.4        0     INIT/ -         00:00:38   192.168.14.4 Ethernet0/1
2.2.2.2        0     FULL/ -         00:00:39   192.168.12.2 Ethernet0/0
```

## Debugging Output of Debug IP OSPF ADJ Command

Debugging is used on **R1** to try to determine what the issue is. Example 24-18 shows the output of the **debug ip ospf adj** command. This command is used to reveal messages that are exchanged during the OSPF adjacency process.

The output of the debug ip ospf adj command in Example 24-18 clearly states that it received a Database Descriptor packet from the neighbor 4.4.4.4, and that the **neighbor 4.4.4.4 has a smaller interface MTU of 1400**.

### Example 24-18 *Output of the debug ip ospf adj Command on R1*

```
R1# debug ip ospf adj
OSPF adjacency debugging is on
R1#
19:20:42.559: OSPF-1 ADJ   Et0/1: Rcv DBD from 4.4.4.4 seq 0x247A opt 0x52 flag 0x7
len 32  mtu 1400 state EXCHANGE
19:20:42.559: OSPF-1 ADJ   Et0/1: Nbr 4.4.4.4 has smaller interface MTU
19:20:42.559: OSPF-1 ADJ   Et0/1: Send DBD to 4.4.4.4 seq 0x247A opt 0x52 flag 0x2
len 152
R1#un all
All possible debugging has been turned off
```

## Debugging Output of Debug IP OSPF ADJ Command (Cont.)

Example 24-19 shows the output of the **debug ip ospf adj** command on **R4** with the relevant fields highlighted. The output shows that **R1 has an MTU size of 1500, which is larger than the locally configured MTU of 1400 on R4**. This is a really quick way of troubleshooting this type of issue with adjacency formation

### Example 24-19 *Output of the debug ip ospf adj Command on R4*

```
R4# debug ip ospf adj
OSPF adjacency debugging is on
R4#
19:28:18.102: OSPF-1 ADJ   Et0/1: Send DBD to 1.1.1.1 seq 0x235C opt 0x52 flag 0x7
len 32
19:28:18.102: OSPF-1 ADJ   Et0/1: Retransmitting DBD to 1.1.1.1 [23]
19:28:18.102: OSPF-1 ADJ   Et0/1: Rcv DBD from 1.1.1.1 seq 0x235C opt 0x52 flag 0x2
len 152  mtu 1500 state EXSTART
19:28:18.102: OSPF-1 ADJ   Et0/1: Nbr 1.1.1.1 has larger interface MTU
R4#un all
All possible debugging has been turned off
```

## Debugging

# Output of the Debug IP OSPF Hello Command

The issue with OSPF network type mismatch, which is a very common reason for neighbor adjacency issues. Often this is simply a misconfiguration issue when setting up the network.

When the **debug ip ospf hello** command is used on R1, everything appears to be normal: Hellos are sent to the multicast group 224.0.0.5 every 10 seconds.

Example 24-20 shows the output of the **debug** command on R1.

**Example 24-20** *Output of the debug ip ospf hello Command on R1*

```
R1# debug ip ospf hello
OSPF hello debugging is on
R1#
19:47:46.976: OSPF-1 HELLO Et0/0: Send hello to 224.0.0.5 area 0 from 192.168.12.1
19:47:47.431: OSPF-1 HELLO Et0/1: Send hello to 224.0.0.5 area 0 from 192.168.14.1
19:47:48.363: OSPF-1 HELLO Et0/2: Send hello to 224.0.0.5 area 0 from 192.168.17.1
R1#
19:47:50.582: OSPF-1 HELLO Et0/0: Rcv hello from 2.2.2.2 area 0 192.168.12.2
19:47:51.759: OSPF-1 HELLO Et0/2: Rcv hello from 7.7.7.7 area 0 192.168.17.7
R1#
19:47:56.923: OSPF-1 HELLO Et0/0: Send hello to 224.0.0.5 area 0 from 192.168.12.1
19:47:57.235: OSPF-1 HELLO Et0/1: Send hello to 224.0.0.5 area 0 from 192.168.14.1
19:47:58.159: OSPF-1 HELLO Et0/2: Send hello to 224.0.0.5 area 0 from 192.168.17.1
R1#
19:47:59.776: OSPF-1 HELLO Et0/0: Rcv hello from 2.2.2.2 area 0 192.168.12.2
19:48:01.622: OSPF-1 HELLO Et0/2: Rcv hello from 7.7.7.7 area 0 192.168.17.7
R1#un all
All possible debugging has been turned off
```

## Debugging Output of the Debug IP OSPF Hello Command (Cont.)

Example 24-21 shows the issue output on R4.

Based on the output, we can see that the **hello parameters are mismatched**. The output shows that **R4 is receiving a dead interval of 40**, while it has a **configured dead interval of 120**. We can also see that the **hello interval R4 is receiving is 10**, and the **configured hello interval is 30**. *By default, the dead interval is 4 times the hello interval.*

**Example 24-21** Output of the debug ip ospf hello Command on R4

```
R4# debug ip ospf hello
OSPF hello debugging is on
R4#
19:45:45.127: OSPF-1 HELLO Et0/1: Rcv hello from 1.1.1.1 area 0 192.168.14.1
19:45:45.127: OSPF-1 HELLO Et0/1: Mismatched hello parameters from 192.168.14.1
19:45:45.127: OSPF-1 HELLO Et0/1: Dead R 40 C 120, Hello R 10 C 30
19:45:45.259: OSPF-1 HELLO Et0/3: Rcv hello from 7.7.7.7 area 0 192.168.47.7
R4#
19:45:48.298: OSPF-1 HELLO Et0/0: Send hello to 224.0.0.5 area 0 from 192.168.34.4
19:45:48.602: OSPF-1 HELLO Et0/0: Rcv hello from 3.3.3.3 area 0 192.168.34.3
R4#un all
All possible debugging has been turned off
```

## Debugging Hello and Dead Intervals

Different network types have different hello intervals and dead intervals. Table 24-4 highlights the different hello and dead interval times based on the different OSPF network types.

Table 24-4 OSPF Network Types and Hello/Dead Intervals

Network Type	Hello Interval (in seconds)	Dead Interval (in seconds)
Broadcast	10	40
Non-broadcast	30	120
Point-to-point	10	40
Point-to-Multipoint	30	120

## Debugging Output of the Show IP OSPF Interface

The issue could be simply mismatched network types or mismatched hello or dead intervals.

The **show ip ospf interface** command shows what the configured network types and hello and dead intervals are.

Example 24-22 shows the output of this command on R4.

### Example 24-22 Output of the show ip ospf interface Command on R4

```
R4# show ip ospf interface ethernet0/1
Ethernet0/1 is up, line protocol is up
  Internet Address 192.168.14.4/24, Area 0, Attached via Network Statement
  Process ID 1, Router ID 4.4.4.4, Network Type POINT_TO_MULTIPOINT Cost: 10
  Topology-MTID      Cost      Disabled      Shutdown      Topology Name
  0                  10        no            no            Base
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    oob-resync timeout 120
    Hello due in 00:00:05
  Supports Link-local Signaling (LLS)
  Cisco NSF helper support enabled
  IETF NSF helper support enabled
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 2
  Last flood scan time is 0 msec, maximum is 1 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
```



## Debugging Changing the Network Type

Simply changing the network type on R4 interface Ethernet0/1 back to the default of Broadcast fixes the adjacency issue in this case. This is because R1 is configured as Broadcast, and now the hello and dead intervals will match.

Example 24-23 shows the **ip ospf network-type broadcast** command issued to change the network type to Broadcast on the Ethernet0/1 interface and the neighbor adjacency coming up. It is also verified with the do **show ip ospf neighbor** command.

### Example 24-23 *Changing the Network Type on R4*

```
R4# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R4(config)# interface ethernet0/1
R4(config-if)# ip ospf network broadcast
R4(config-if)#
20:28:51.904: %OSPF-5-ADJCHG: Process 1, Nbr 1.1.1.1 on Ethernet0/1 from LOADING to
FULL, Loading Done
R4(config-if)# do show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
7.7.7.7	0	FULL/ -	00:00:32	192.168.47.7	Ethernet0/3
1.1.1.1	1	FULL/BDR	00:00:39	192.168.14.1	Ethernet0/1
3.3.3.3	0	FULL/ -	00:00:33	192.168.34.3	Ethernet0/0

## Debugging Show and Debug Commands on R1

The final use case for using debugging to solve OSPF adjacency issues involves improper configuration of IP addresses and subnet masks on an OSPF interface.

To troubleshoot this without having to look through running configurations or at a specific interface, use the **debug ip ospf hello** command covered earlier in this section.

Example 24-24 shows the output of running the **show ip ospf neighbor** command on R1. It indicates that there is no OSPF adjacency to R4 when there certainly should be one.

### Example 24-24 show ip ospf neighbor, debug ip ospf hello, and debug ip ospf adj Commands on R1

```
R1# show ip ospf neighbor

Neighbor ID    Pri   State           Dead Time   Address      Interface
7.7.7.7        0     FULL/ -         00:00:34   192.168.17.7 Ethernet0/2
4.4.4.4        0     INIT/ -         00:00:30   192.168.14.4 Ethernet0/1
2.2.2.2        0     FULL/ -         00:00:37   192.168.12.2 Ethernet0/0
R1#
R1# deb ip os hello
OSPF hello debugging is on
R1# deb ip ospf adj
OSPF adjacency debugging is on
R1#
20:55:02.465: OSPF-1 HELLO Et0/0: Send hello to 224.0.0.5 area 0 from 192.168.12.1
20:55:03.660: OSPF-1 HELLO Et0/0: Rcv hello from 2.2.2.2 area 0 192.168.12.2
20:55:04.867: OSPF-1 HELLO Et0/1: Send hello to 224.0.0.5 area 0 from 192.168.14.1
20:55:05.468: OSPF-1 HELLO Et0/1: Rcv hello from 4.4.4.4 area 0 192.168.14.4
20:55:05.468: OSPF-1 HELLO Et0/1: No more immediate hello for nbr 4.4.4.4, which has
been sent on this intf 2 times
R1#
20:55:06.051: OSPF-1 HELLO Et0/2: Send hello to 224.0.0.5 area 0 from 192.168.17.1
R1#
20:55:08.006: OSPF-1 HELLO Et0/2: Rcv hello from 7.7.7.7 area 0 192.168.17.7

R1#
R1# undebg all
All possible debugging has been turned off
```

## Debugging Ping

The **network mask** on the Ethernet0/1 interface of R4 **needs to be changed** to match the one that R1 has configured and is sending to R4 through OSPF hellos.

Example 24-26 shows the network mask being changed on the R4 Ethernet0/1 interface and the OSPF adjacency coming up. This is then verified with the **do show ip ospf neighbor** command.

### Example 24-26 *Network Mask Change and show ip ospf neighbor on R4*

```
R4# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R4(config)# interface ethernet0/1
R4(config-if)# ip address 192.168.14.4 255.255.255.0
R4(config-if)#
21:14:15.598: %OSPF-5-ADJCHG: Process 1, Nbr 1.1.1.1 on Ethernet0/1 from LOADING to FULL, Loading Done
R4(config-if)# do show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
1.1.1.1	1	FULL/BDR	00:00:38	192.168.14.1	Ethernet0/1
7.7.7.7	0	FULL/ -	00:00:37	192.168.47.7	Ethernet0/3
3.3.3.3	0	FULL/ -	00:00:30	192.168.34.3	Ethernet0/0

```
R4(config-if)#
```

## Debugging

# Conditional Debugging

Conditional debugging can be used to **limit the scope of the messages that are being returned to the console or syslog server**. A great example of this is the **debug ip packet** command. Issuing this command on a router that is in production could send back a tremendous number of messages.

One way to alleviate this issue is to attach an access list to the **debug** command to limit the scope of messages to the source or destination specified within the access list. This can be done using standard or extended access lists.

The **options** for the **debug ip packet** command are as follows:

- **<1-199>**: Standard access list
- **<1300-2699>**: Access list with expanded range
- **detail**: More debugging detail

## Debugging

# Debugging on a Specific Interface

Another common method of conditional debugging is to **debug on a specific interface**. This is extremely useful when trying to narrow down a packet flow between two hosts.

Imagine that a network engineer is trying to debug a traffic flow between R1's Ethernet0/1 interface with source IP address 192.168.14.1/24 that is destined to R4's Loopback0 interface with IP address 4.4.4.4/32.

One way to do this would certainly be to change the access list 100 to reflect these source and destination IP addresses. However, because the access list is looking for any traffic sourced or destined to the 192.168.14.0/24 network, this traffic flow would fall into matching that access list. Using conditional debugging on the Loopback0 interface of R4 would be a simple way of meeting these requirements.

Even if all debugging has been turned off using the `undebug all` command, the interface conditions set for Loopback0 on R4 remain. The way to remove this condition is to use the `undebug interface loopback0` command on R4.

## Debugging Simple Network Management Protocol (SNMP)

The typical tool for reactive alerting from network devices is Simple Network Management Protocol (SNMP).

SNMP sends unsolicited traps to an SNMP collector or network management system (NMS). These traps are in response to something that happened in the network.

SNMP agent („client“)  
SNMP manager

For example, traps may be generated for link status events, improper user authentication, and power supply failures. These events are defined in the SNMP Management Information Base (MIB).

The MIB can be thought of as a repository of device parameters that can be used to trigger alerts.

# Debugging SNMP Versions

There are currently three versions of SNMP. Table 24-5 lists the versions and their differences.

Table 24-5 SNMP Version Comparison

Version	Level	Authentication	Encryption	Result
SNMPv1	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv2c	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv3	noAuthNoPriv	Username	No	Uses a username match for authentication.
SNMPv3	authNoPriv	Message Digest 5 (MD5) or Secure Hash Algorithm (SHA)	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms.

## Debugging SNMP Versions (Cont.)

Version	Level	Authentication	Encryption	Result
SNMPv3	authPriv (requires the cryptographic software image)	MD5 or SHA	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Allows specifying the User-based Security Model (USM) with these encryption algorithms:DES 56-bit encryption in addition to authentication based on the CBC-DES (DES-56) standard.3DES 168-bit encryptionAES 128-bit, 192-bit, or 256-bit encryption



## Debugging SNMP Operations

Using SNMP is considered best practice in production. SNMPv1 and SNMPv2c use access lists and a community password or string.

These community strings can be read-only (RO) or read/write (RW). It is critical to limit SNMP access to these devices by using access lists.

By default, if no version is specified in configuration, SNMPv1 is used. However, to better show how SNMP works, this chapter focuses on SNMPv2c.

Operation	Description
<b>get-request</b>	Retrieves a value from a specific variable
<b>get-next-request</b>	Retrieves a value from a variable within a table
<b>get-bulk-request</b>	Retrieves large blocks of data, such as multiple rows in a table
<b>get-response</b>	Replies to a get request, get next request, and set request sent by an NMS
<b>set-request</b>	Stores a value in a specific variable
<b>trap</b>	Sends an unsolicited message from an SNMP agent to an SNMP manager when some event has occurred

## Debugging MIBs

It is important to look at a MIB to understand some of the information or values that can be polled or send traps from SNMP.

The SNMPv2-MIB.my file shows what values can be polled in the MIB and to illustrate sending traps from SNMP. To see a list of available Cisco MIBs, visit <https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index>.

Although configuring an NMS is not covered in this chapter, the device side that points to an NMS is covered in this section.

The following list shows a handful of measures involved in setting up SNMP on a device to allow the device to be polled and send traps to an NMS:

- Define the SNMP host or the NMS to send traps to
- Create an access list to restrict access via SNMP
- Define the read-only community string
- Define the read/write community string
- Define the SNMP location
- Define the SNMP contact

## Debugging MIBs (Cont.)

These settings do not need to be configured in any particular order. It is best practice to configure the access list first and then the read-only and read/write strings to have security in place.

On R1, a standard access list is configured to only permit access from an NMS host on the 192.168.14.0/24 subnet, 192.168.14.100. Once the access list is configured, the read-only and read/write community strings are configured and bound to that access list.

It is important to try to use SNMP strings that are not easy to guess from a security perspective.

### **Example 24-30** *SNMP Access List on R1*

```
R4(config)# access-list 99 permit 192.168.14.100 0.0.0.0
R4(config)# snmp-server community READONLY ro 99
R4(config)# snmp-server community READONLY rw 99
```

At this point, the device is configured to be polled from an NMS host with the IP address 192.168.14.100.

## Debugging SNMP Traps

To send SNMP traps to an NMS, traps first must be enabled on the device. All available traps can be enabled by issuing the **snmp-server enable traps** command.

- However, this may enable unnecessary traps that have no significance to the network operations team. It might be more appropriate to be selective about which traps to enable.
- The traps that are available to be enabled is platform specific.
- A common approach to determining what traps are available is to look at the documentation for the device. It may be easier to simply issue the **snmp-server enable traps** command followed by ? to leverage the context-sensitive help and determine what traps are available on the device.

A significant number of traps can be enabled to send to an NMS.

# Debugging Syslog

Devices can generate useful information to the console, to the logging buffer, and to off-box syslog collectors. In fact, all three can be sent the same or different message types.

It is critical to note that prior to configuring any device to send log information, the date and time of the clock must be properly configured for accurate time.

Level Keyword	Level	Description	Syslog Definition
<b>emergencies</b>	0	System unstable	LOG_EMERG
<b>alerts</b>	1	Immediate action needed	LOG_ALERT
<b>critical</b>	2	Critical conditions	LOG_CRIT
<b>errors</b>	3	Error conditions	LOG_ERR
<b>warnings</b>	4	Warning conditions	LOG_WARNING
<b>notifications</b>	5	Normal but significant conditions	LOG_NOTICE
<b>informational</b>	6	Informational messages only	LOG_INFO
<b>debugging</b>	7	Debugging message	LOG_DEBUG

## Debugging Syslog (Cont.)

Having syslog configured doesn't mean that an issue will be found. It still takes the proper skill to be able to look at the messages and determine the root cause of the issue.

The logging buffer is the first area to focus on. You can enable logging to the buffer as follows:

1. Enable logging to the buffer.
2. Set the severity level of syslog messages to send to the buffer.
3. Set the logging buffer to a larger size.

The **logging buffered ?** command is issued from the global configuration mode to see the available options. It is important to note that the severity level can be configured by simply specifying the level with a number from 0 to 7 or the name of the severity (listed next to the severity level number).

The default size of the logging buffer is 4096 bytes. This can get overwritten quite quickly. It is good practice to expand the buffer size so you can capture more logging information.

# Debugging Logging Buffer Size and Severity Level

Debugging or severity 7 is the level that will be configured in this example; with this configuration, any debugging can be sent to the logging buffer instead of the console.

In Example 24-34, the logging is configured to the debugging level, 7, and it is set to 100000 bytes.

**Example 24-34** *Configuring the Logging Buffer Size and Severity Level on R1*

```
R1(config)# logging buffer 100000
R1(config)#
R1(config)# logging buffer debugging

R1(config)# do show logging
Syslog logging: enabled (0 messages dropped, 4 messages rate-limited, 0 flushes,
  0 overruns, xml disabled, filtering disabled)

No Active Message Discriminator.

No Inactive Message Discriminator.

Console logging: disabled
Monitor logging: level debugging, 0 messages logged, xml disabled,
  filtering disabled
Buffer logging: level debugging, 1 messages logged, xml disabled,
  filtering disabled
Exception Logging: size (4096 bytes)
Count and timestamp logging messages: disabled
Persistent logging: disabled

No active filter modules.

Trap logging: level informational, 108 message lines logged
Logging Source-Interface: VRF Name:

Log Buffer (100000 bytes):

*Jul 10 19:41:05.793: %SYS-5-LOG_CONFIG_CHANGE: Buffer logging: level debugging, xml
  disabled, filtering disabled, size (100000)
```

# Sending Logging to a Host for Debugging

Sending these same logs to an off-box collector, that could be configured as well. By default, these messages are sent to the logging host through UDP port 514, but this can be changed if necessary.

Configuring logging to a host is very similar to configuring logging on the console or buffer. In this case, it is configured by using the following steps:

1. Enable logging to host 192.168.14.100.
2. Set the severity level of syslog messages to send to host.

Syslog can be used to notify of power supply failures, CPU spikes, and a variety of other things.

It is important not to underestimate the level of granularity and detail that can be achieved by setting up proper notification policies in a network. It is ultimately up to the network operations team to determine how deep is appropriate to meet the business's needs.



# NetFlow and Flexible NetFlow

- Gathering statistical information on traffic flows is necessary for a number of reasons.
- NetFlow is very versatile and provides a wealth of information without much configuration burden.
- NetFlow has two components that must be configured: NetFlow Data Capture and NetFlow Data Export.

## NetFlow Ingress/Egress Collected Traffic Types

NetFlow captures traffic on ingress and egress—that is, traffic that is coming into the devices as well as traffic that is leaving them.

NetFlow collects traffic based on flows. A flow is a unidirectional traffic stream that contains a combination of the following key fields:

- Source IP address
- Destination IP address
- Source port number
- Destination port number
- Layer 3 protocol type
- Type of service (ToS)
- Input logical interface

Versions!

## NetFlow and Flexible NetFlow

# Enable NetFlow

R1's Ethernet0/1 interface for NetFlow Data Capture and exporting the data to the 192.168.14.100 collector. Example 24-37 illustrates the process of configuring NetFlow Data Capture and NetFlow Data Export on R1.

### Example 24-37 *Configuring NetFlow and NetFlow Data Export on R1*

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ip flow-export version 9
R1(config)# ip flow-export destination 192.168.14.100 9999
R1(config)# interface Ethernet0/1
R1(config-if)# ip flow ingress
R1(config-if)# ip flow egress
R1(config-if)# end
R1#
```

To verify that NetFlow and NetFlow Data Export were configured properly, a few commands can be run. The first is **show ip flow interface**, which shows the interfaces that are configured for NetFlow. The second is **show ip flow**, which shows the destination for the NetFlow data to be exported to. Finally, **show ip cache flow** shows the traffic flows that NetFlow is capturing.

## NetFlow and Flexible NetFlow

# Configuring and Verifying the Top Talkers on R1

NetFlow is able to configure the top specified number of talkers on the network.

A very useful and quick configuration allows you to gain a great snapshot of what is going on in a device from a flow perspective.

This view can be enabled by issuing the global configuration mode command **ip flow-top-talkers** and configuring the **top** command for the number of talkers (1–200) and the **sort-by** command to sort by bytes or packets, depending on the use case.

### Example 24-39 Configuring and Verifying the Top Talkers on R1

```
R1# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ip flow-top-talkers
R1(config-flow-top-talkers)# top 10
R1(config-flow-top-talkers)# sort-by bytes
R1(config-flow-top-talkers)# end
R1#
R1#
R1# show ip flow top-talkers
```

SrcIf	SrcIPAddress	DstIf	DstIPAddress	Pr	SrcP	DstP	Bytes
Et0/1	192.168.14.4	Null	224.0.0.2	11	0286	0286	9610
Et0/1	192.168.14.4	Null	224.0.0.5	59	0000	0000	5820

2 of 10 top talkers shown. 2 of 2 flows matched.

```
R1#
```

## NetFlow and Flexible NetFlow

# Flexible NetFlow

Flexible NetFlow was created to aid in more complex traffic analysis configuration than is possible with traditional NetFlow.

Flexible NetFlow allows for the use and reuse of configuration components.

Flexible NetFlow allows for the use of multiple flow monitors on the same traffic at the same time. This means that multiple different flow policies can be applied to the same traffic as it flows through a device.

Cisco proprietary

Table 24-9 Flexible NetFlow Components

Component Name	Description
Flow Records	Combination of key and non-key fields. There are predefined and user-defined records.
Flow Monitors	Applied to the interface to perform network traffic monitor
Flow Exporters	Exports NetFlow Version 9 data from the Flow Monitor cache to a remote host or NetFlow collector.
Flow Samplers	Samples partial NetFlow data rather than analyzing all NetFlow data.

## NetFlow and Flexible NetFlow

# Sampled NetFlow Data Trade-offs

kompromisy

There are trade-offs in using sampled NetFlow data.

The biggest one is that there is a reduced load on the device in terms of memory and CPU. However, by sampling NetFlow data only at specific intervals, something could be missed as the accuracy goes down with sampling compared to when gathering all data.

Security has been a huge driver in the adoption of Flexible NetFlow due to its ability to track all parts of the IP header, as well as the packet and normalize it into flows.

- Flexible NetFlow can dynamically create individual caches for each type of flow.
- Flexible NetFlow can filter ingress traffic destined to a single destination.

You can use the **collect** and **match** commands to create a customized flow record.

To create a custom flow record, certain key and non-key fields must be matched so the flow record is usable. The **match** command is used to select key fields, and the **collect** command is used to select non-key fields.

## Flow Record Key and Non-Key Fields

Table 24-10 shows a list of the key and non-key fields that can be used to mimic the original NetFlow capabilities when building a custom flow record.

Table 24-10 Flow Record Key and Non-Key Fields

Field	Key or Non-Key Field	Definition
IP ToS	Key	Value in the type of service (ToS)
IP ToS	Key	Value in the IP protocol field
IP source address	Key	IP source address
Transport source port	Key	IP destination address
Transport destination port	Key	Value of the transport layer source port field
Interface input	Key	Value of the transport layer destination port
Flow sampler ID	Key	ID number of the flow sampler (if flow sampling is enabled)

## Flow Record Key and Non-Key Fields (Cont.)

Field	Key or Non-Key Field	Definition
IP source AS	Non-key	Source autonomous system
IP destination AS	Non-key	Destination autonomous system number
IP next-hop address	Non-key	IP address of the next hop
IP source mask	Non-key	Mask for the IP source address
IP destination mask	Non-key	Mask for the IP destination address
TCP flags	Non-key	Value in the TCP flag
Interface output	Non-key	Interface on which the traffic is transmitted
Counter bytes	Non-key	Number of bytes seen in the flow
Counter packets	Non-key	Number of packets seen in the flow
Time stamp system uptime first	Non-key	System uptime (time, in milliseconds)
Time stamp system uptime last	Non-key	System uptime (time, in milliseconds)



## NetFlow and Flexible NetFlow

# Configuring Flow Records

Configuring flow records is an important step in enabling Flexible NetFlow. because the flow record defines what type of traffic will be analyzed or monitored.

- There are predefined flow records, and you can also create custom flow records.
- Custom flow records can have hundreds of different combinations to meet the exact needs of the business.

Configuring a custom flow record involves the following steps:

1. Define the flow record name.
2. Set a useful description of the flow record.
3. Set match criteria for key fields.
4. Define non-key fields to be collected.

# NetFlow and Flexible NetFlow

## Configuring the Custom Flow Record

Although many of the predefined flow records that are available may be suitable for many use cases, there are too many of them to cover here.

Having the ability to build a custom flow record for a specific and unique use case makes it extremely powerful.

Example 24-40 shows a custom flow record called CUSTOM1.

**Example 24-40** *Configuring and Verifying the Custom Flow Record on R4*

```
R4# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R4(config)# flow record CUSTOM1
R4(config-flow-record)# description Custom Flow Record for IPv4 Traffic
R4(config-flow-record)# match ipv4 destination address
R4(config-flow-record)# collect counter bytes
R4(config-flow-record)# collect counter packets
R4(config-flow-record)# exit
R4(config)#
R4(config)# do show flow record CUSTOM1
flow record CUSTOM1:
  Description:      Custom Flow Record for IPv4 Traffic
  No. of users:    0
  Total field space: 12 bytes
  Fields:
    match ipv4 destination address
    collect counter bytes
    collect counter packets

R4(config)#
R4(config)#do show running-config flow record
Current configuration:
!
flow record CUSTOM1
  description Custom Flow Record for IPv4 Traffic
  match ipv4 destination address
  collect counter bytes
  collect counter packets
!
R4(config)#
```

# NetFlow and Flexible NetFlow

## Configuring the Custom Flow Exporter

Now that a custom flow record has been configured, the flow exporter can be created.

There are a few important steps to complete when building a flow exporter:

1. Define the flow exporter name.
2. Set a useful description of the flow exporter.
3. Specify the destination of the flow exporter to be used.
4. Specify NetFlow version to export.
5. Specify the UDP port.

Example 24-41 illustrates the configuration of the flow exporter as well as how to verify the configuration.

**Example 24-41** *Configuring and Verifying the Custom Flow Exporter on R4*

```
R4# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R4(config)# flow exporter CUSTOM1
R4(config-flow-exporter)# description EXPORT-TO-NETFLOW-COLLECTOR
R4(config-flow-exporter)# destination 192.168.14.100
R4(config-flow-exporter)# export-protocol netflow-v9
R4(config-flow-exporter)# transport UDP 9999
R4(config-flow-exporter)# exit
R4(config)# exit
R4# sh run flow exporter
Current configuration:
!
flow exporter CUSTOM1
  description EXPORT-TO-NETFLOW-COLLECTOR
  destination 192.168.14.100
  transport udp 9999
!
R4#
R4# show flow exporter CUSTOM1
Flow Exporter CUSTOM1:
  Description:          EXPORT-TO-NETFLOW-COLLECTOR
  Export protocol:      NetFlow Version 9
  Transport Configuration:
    Destination IP address: 192.168.14.100
    Source IP address:     192.168.14.4
    Transport Protocol:    UDP
    Destination Port:      9999
    Source Port:           50192
    DSCP:                  0x0
    TTL:                   255
    Output Features:       Not Used
R4#
```

## NetFlow and Flexible NetFlow

# Configure a Flow Monitor

Now that a custom flow exporter has been configured, the flow monitor must be created.

- Each flow monitor requires a flow record to be assigned to it.
- Each flow monitor has its own cache, and the flow record provides the layout and how to carve up the cache for the defined traffic defined in the flow record.

To configure a flow monitor, the following high-level steps must be taken:

1. Define the flow monitor name.
2. Set a useful description of the flow monitor.
3. Specify the flow record to be used.
4. Specify a cache timeout of 60 for active connections.
5. Assign the exporter to the monitor.

The cache timeout tells the device to export the cache to the collector every 60 seconds. It is important when creating a flow monitor for the description of the flow monitor to be useful and to map back to the flow record.

# NetFlow and Flexible NetFlow

## Configuring the Custom Flow Monitor

When configuring QoS, it is nice to have the descriptions self-document the intent of what the policy is doing.

This helps when configuring the flow monitor and when using context sensitive help, as the description that is configured shows in the output.

Example 24-42 shows this as well as the configuration and verification for the flow monitor called CUSTOM1.

### Example 24-42 *Configuring and Verifying the Custom Flow Monitor on R4*

```
R4(config)# flow monitor CUSTOM1
R4(config-flow-monitor)# description Uses Custom Flow Record CUSTOM1 for IPv4$
R4(config-flow-monitor)# record ?
  CUSTOM1      Custom Flow Record for IPv4 Traffic
  netflow      Traditional NetFlow collection schemes
  netflow-original Traditional IPv4 input NetFlow with origin ASs
R4(config-flow-monitor)# record CUSTOM1
R4(config-flow-monitor)# cache active timeout 60
R4(config-flow-monitor)# end
R4# show run flow monitor CUSTOM1
Current configuration:
!
flow monitor CUSTOM1
  description Uses Custom Flow Record CUSTOM1 for IPv4 Traffic
  cache timeout active 60
  record CUSTOM1
!
R4# show flow monitor CUSTOM1
Flow Monitor CUSTOM1:
  Description:      Uses Custom Flow Record CUSTOM1 for IPv4 Traffic
  Flow Record:     CUSTOM1
  Cache:
    Type:           normal
    Status:         not allocated
    Size:           4096 entries / 0 bytes
    Inactive Timeout: 15 secs
    Active Timeout: 60 secs
    Update Timeout: 1800 secs
    Synchronized Timeout: 600 secs
R4#
```

# NetFlow and Flexible NetFlow

## Configuring the Flow Exporter Mapping

The next step is to map the flow exporter CUSTOM1 to the flow monitor CUSTOM1.

You need to essentially map the two together so the traffic that is being collected by the flow record can be exported to the NetFlow collector at 192.168.14.100.

Example 24-43 shows the process and verification for adding the flow exporter CUSTOM1 to the flow monitor CUSTOM1 on R4.

**Example 24-43** *Configuring and Verifying the Flow Exporter Mapping to the Flow Monitor on R4*

```
R4# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R4(config)# flow monitor CUSTOM1
R4(config-flow-monitor)# exporter ?
CUSTOM1 EXPORT-TO-NETFLOW-COLLECTOR
R4(config-flow-monitor)# exporter CUSTOM1
R4(config-flow-monitor)# end
R4# show run flow monitor
Current configuration:
!
flow monitor CUSTOM1
description Uses Custom Flow Record CUSTOM1 for IPv4 Traffic
exporter CUSTOM1
cache timeout active 60
record CUSTOM1
!
R4# show flow monitor CUSTOM1
Flow Monitor CUSTOM1:
Description:      Uses Custom Flow Record CUSTOM1 for IPv4 Traffic
Flow Record:     CUSTOM1
Flow Exporter:   CUSTOM1 (inactive)
Cache:
Type:            normal
Status:         not allocated
Size:           4096 entries / 0 bytes
Inactive Timeout: 15 secs
Active Timeout: 60 secs
Update Timeout: 1800 secs
Synchronized Timeout: 600 secs
R4#
```

# NetFlow and Flexible NetFlow

## Configuring the Flow Monitor Interface

The final step necessary in enabling Flexible NetFlow is to apply the flow monitor to the interfaces.

This step turns on the collection of NetFlow statistics, and it can be enabled for ingress or egress or both.

Example 24-44 illustrates the process as well as how to verify that Flexible NetFlow is working by issuing the **show ip flow monitor CUSTOM1 cache** command.

**Example 24-44** *Configuring and Verifying the Flow Monitor Interface Commands on R4*

```
R4(config)# interface ethernet0/1
R4(config-if)# ip flow monitor ?
CUSTOM1          Uses Custom Flow Record CUSTOM1 for IPv4 Traffic
R4(config-if)# ip flow monitor CUSTOM1 input
R4(config-if)# interface ethernet0/2
R4(config-if)# ip flow monitor CUSTOM1 input
R4(config-if)# end
R4# show flow monitor CUSTOM1 cache
```

Cache type:	Normal
Cache size:	4096
Current entries:	3
High Watermark:	3
Flows added:	8
Flows aged:	5
- Active timeout ( 60 secs)	5
- Inactive timeout ( 15 secs)	0
- Event aged	0
- Watermark aged	0
- Emergency aged	0

IPV4 DST ADDR	bytes	pkts
=====	=====	=====
224.0.0.5	560	7
224.0.0.2	372	6
4.4.4.4	674	11

# Switched Port Analyzer (SPAN) Technologies

When the problem appears to be a Layer 2 issue, there are a few options:

- Insert a splitter between the devices.
- Configure the network device to mirror the packets.
- Insert a switch between the two devices and then configure the switch to mirror the transient traffic to a traffic analyzer.

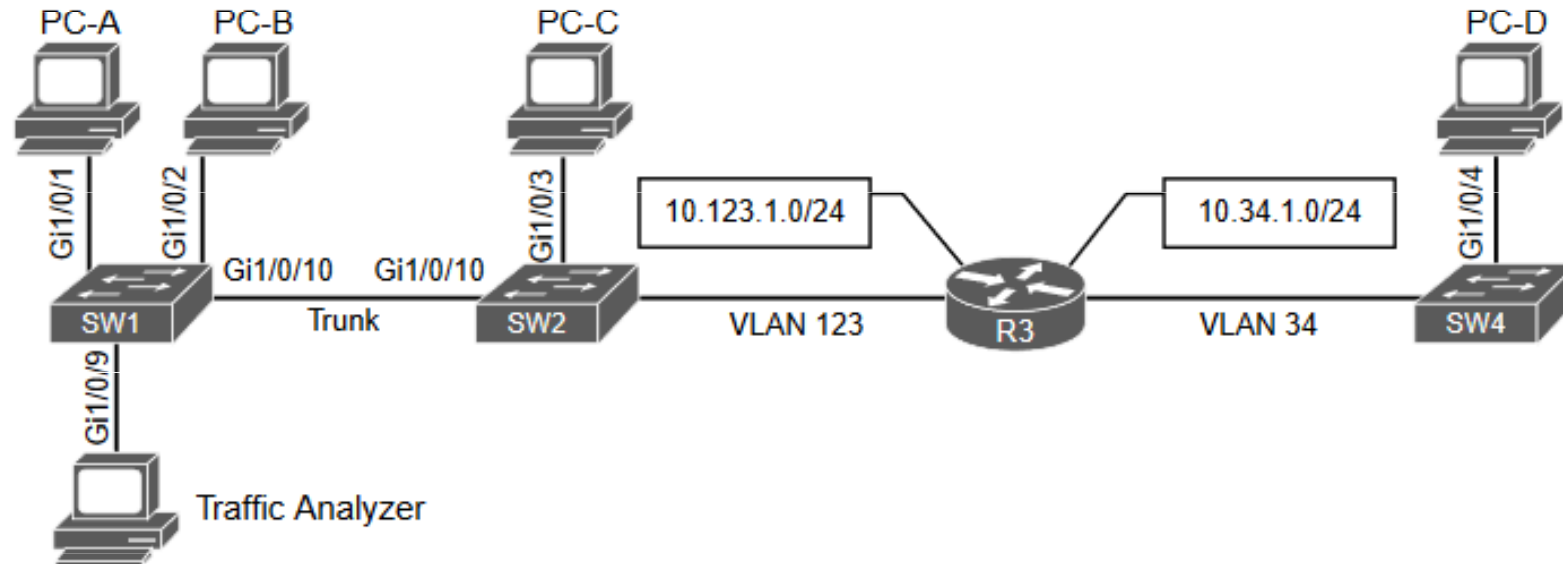
Catalyst switches provide the Switched Port Analyzer (SPAN), which makes it possible to capture packets using the second two options above by using the following techniques:



## Switched Port Analyzer (SPAN) Technologies

# Sample Topology for Packet Captures

Figure 24-4 shows a sample topology with four computers (PC-A, PC-B, PC-C, and PC-D) spread across three switches and a traffic analyzer connected to SW1. PC-A, PC-B, and PC-C are all connected to VLAN 123 on the 10.123.1.0/24 network, and PC-D is connected to VLAN 34, which is on the 10.34.1.0/24 network. This topology is used to demonstrate the concepts of SPAN, RSPAN, and ERSPAN.



**Figure 24-4** *Sample Topology for Packet Captures*

## Switched Port Analyzer (SPAN) Technologies

# Local SPAN

A local SPAN session is the most basic form of packet capture as all the configuration occurs on a single switch. The destination of the mirrored traffic can be one or more local ports. The source of the packet capture can be only one of the following:

- One or more specific switch ports
- A port channel (also known as an EtherChannel)
- A VLAN

Also consider the following:

- Most switches support at least two SPAN sessions, but newer hardware can support more than two sessions.
- The source port cannot be reused between two different SPAN sessions.
- Source ports can be switched or routed ports.
- The destination cannot be reused between two different SPAN sessions.
- It is possible to saturate the destination port if the source ports are receiving more data than the destination port can transmit.

## Switched Port Analyzer (SPAN) Technologies

# Specifying the Source Ports

The source ports are defined with the global configuration command **monitor session *session-id* source {interface *interface-id* | vlan *vlan-id*} [rx | tx | both]**. The SPAN *session-id* allows for the switch to correlate the source ports to specific destination ports.

The direction of the traffic can be specified as part of the configuration. With the optional **rx** keyword you capture only traffic received on that source, with the optional **tx** key word you capture traffic sent by that source, and with the **both** keyword you capture all traffic.

You can specify a trunk port as a source port to capture traffic for all VLANs that traverse that port. This might provide too much data and add noise to the traffic analysis tool. The VLANs can be filtered on the capture with the command **monitor session *session-id* filter vlan *vlan-range***.

## Switched Port Analyzer (SPAN) Technologies

# Specifying the Destination Ports

The destination port is specified with the global configuration command:

```
monitor session session-id destination interface interface-id [encapsulation  
{dot1q [ingress {dot1q vlan vlan-id | untagged vlan vlan-id | vlan vlan-id}  
| replicate [ingress {dot1q vlan vlan-id | untagged vlan vlan-id]}]} |  
ingress]
```

As you can see, there are a lot of different nested options.

A SPAN session normally copies the packets without including any 802.1Q VLAN tags or Layer 2 protocols. Using the **encapsulation replicate** keywords includes that information. The full global configuration command is:

```
monitor session session-id destination interface interface-id [encapsulation  
replicate]
```

## Switched Port Analyzer (SPAN) Technologies

# Specifying the Destination Ports (Cont.)

If the traffic analyzer is a Windows PC and is accessed using RDP, the port must be able to send and receive traffic for the Windows PC in addition to the traffic from the SPAN session. Situations like this require the following global configuration command:

```
monitor session session-id destination interface interface-id ingress {dot1q  
vlan vlan-id | untagged vlan vlan-id}
```

Selecting the **dot1q** keyword requires the packets to be encapsulated with the specified VLAN ID. Selecting the **untagged** keyword accepts incoming packets and associates them to the specified VLAN ID.

STP is disabled on the destination port to prevent extra BPDUs from being included in the network analysis. Great care should be taken to prevent a forwarding loop on this port.

## Switched Port Analyzer (SPAN) Technologies

# Local SPAN Configuration Examples

Example 24-45 shows how to monitor both PC-A's and PC-B's communication on SW1 and send it toward the local traffic analyzer.

### Example 24-45 *Enabling a SPAN Session on SW1*

```
SW1(config)# monitor session 1 source interface gi1/0/1 - 2
SW1(config)# monitor session 1 destination interface gi1/0/9
```

A specific SPAN session can be viewed, or the output can be restricted to the local SPAN session, as shown in Example 24-46.

### Example 24-46 *Verifying the Configured SPAN Session*

```
SW1# show monitor session local
Session 1
-----
Type                : Local Session
Source Ports        :
    Both             : Gi1/0/1-2
Destination Ports   : Gi1/0/9
Encapsulation       : Native
    Ingress          : Disabled
```

## Switched Port Analyzer (SPAN) Technologies

# Local SPAN Configuration Examples (Cont.)

The next example illustrates monitoring the trunk port Gi1/0/10 and provides the output to PC-B for PC-A and PC-B communication on SW1 and sending it toward the local traffic analyzer.

Example 24-47 shows the commands that are entered on SW1 and then shows the configuration verified by examining the SPAN session.

### Example 24-47 *Configuring and Verifying SPAN for the SW1 Gi1/0/10 Source*

```
SW1(config)# monitor session 1 source interface gi1/0/10
! Some of the following command keywords were shortened for autocomplete
! so they all appear on the same line.
SW1(config)# monitor session 1 destination interface Gi1/0/9 encapsulation replicate
SW1(config)# monitor session 1 filter vlan 123
```

```
SW1# show monitor session 1
Session 1
-----
Type                : Local Session
Source Ports        :
    Both             : Gi1/0/10
Destination Ports   : Gi1/0/9
    Encapsulation    : Replicate
    Ingress          : Disabled
Filter VLANs        : 123
```

## Switched Port Analyzer (SPAN) Technologies

# Local SPAN Configuration Examples (Cont.)

In the last scenario, the switch is configured to monitor PC-A's traffic, and it uses an already installed network traffic analysis tool on PC-B.

When the switch is configured, PC-B can be accessed remotely to view the network traffic by using RDP.

Example 24-48 lists the commands that are entered on SW1 to capture the ingress traffic and shows the configuration being verified.

### Example 24-48 *Configuring and Verifying SPAN for the SW1 Gi1/0/1 Source*

```
SW1(config)# monitor session 1 source interface gi1/0/1
! Some of the following command keywords were shortened for autocomplete
! so they all appear on the same line.
SW1(config)# monitor session 1 destination interface gi1/0/2 ingress untagged vlan
123
```

```
SW1# show monitor session 1
Session 1
-----
Type                : Local Session
Source Ports        :
    Both             : Gi1/0/1
Destination Ports   : Gi1/0/2
Encapsulation       : Native
    Ingress          : Enabled, default VLAN = 123
Ingress encap       : Untagged
```



## Switched Port Analyzer (SPAN) Technologies

# Remote SPAN (RSPAN)

In large environments, it might not be possible to move a network analyzer to other parts of the network. Example 24-49 shows the RSPAN VLAN being created on SW1 and SW2.

### **Example 24-49** *Creating the RSPAN VLAN*

```
SW1(config)# vlan 99
SW1(config-vlan)# name RSPAN_VLAN
SW1(config-vlan)# remote-span
```

```
SW2(config)# vlan 99
SW2(config-vlan)# name RSPAN_VLAN
SW2(config-vlan)# remote-span
```

Example 24-50 shows the configuration of RSPAN on the source switch, SW2. Traffic from PC-C is sent to SW1 for analysis.

### **Example 24-50** *Configuring a Source RSPAN Switch*

```
SW2(config)# monitor session 1 source interface gi1/0/3
SW2(config)# monitor session 1 destination remote vlan 99
```

# Switched Port Analyzer (SPAN) Technologies

## Configuration of RSPAN on the Destination Switch

Example 24-51 shows the configuration of RSPAN on the destination switch, SW1. The traffic is sent to the traffic analyzer for analysis.

### Example 24-51 *Configuring a Destination RSPAN Switch*

```
SW1(config)# monitor session 1 source remote vlan 99
SW1(config)# monitor session 1 destination interface gi1/0/9
```

Example 24-52 verifies the configuration of RSPAN on both SW1 and SW2.

### Example 24-52 *Verifying the RSPAN Settings*

```
SW1# show monitor session 1
Session 1
-----
Type                : Remote Destination Session
Source RSPAN VLAN   : 99
Destination Ports   : Gi1/0/9
  Encapsulation     : Native
  Ingress           : Disabled

SW2# show monitor session remote
Session 1
-----
Type                : Remote Source Session
Source Ports        :
  Both              : Gi1/0/3
Dest RSPAN VLAN     : 99
```

## Switched Port Analyzer (SPAN) Technologies

# Encapsulated Remote SPAN (ERSPAN)

In large environments, it might not be possible to move a network analyzer to other parts of the network. ERSPAN provides the ability to monitor traffic in one area of the network and route the SPAN traffic to a traffic analyzer in another area of the network through Layer 3 routing.

### Specifying the Source Ports

A source and destination must be configured. To configure a source, the following command is issued: **monitor session *span-session-number* type erspan-source**. This defines the session number as well as the session type, **erspan-source**.

Once the initial session is created, the source must be defined in the session. This is accomplished by issuing the **source { interface *type number* | vlan *vlan-ID* } [ , | - | both | rx | tx ]** command.

## Encapsulated Remote SPAN (ERSPAN) (Cont.)

When the source has been configured, it is necessary to configure the destination of the ERSPAN session. To enter the destination subconfiguration mode, the **destination** command is used. The rest of the commands will be issued in the destination subconfiguration mode to specify the destination of the ERSPAN session as well as any parameters associated with the configuration of the destination.

The next step is to identify the IP address of the destination for the ERSPAN session. Because this is a Layer 3 SPAN session, this IP address is where the traffic will be sent to be analyzed. The command to configure this action is simply **ip address** *ip-address*.

The final step is to assign a ToS or TTL to the ERSPAN traffic. This is done with the **erspan { tos tos-value | ttl ttl-value }** command from global configuration mode.

# Switched Port Analyzer (SPAN) Technologies

## ERSPAN Process

Example 24-53 illustrates this whole process. In addition, to verify the configured sessions, the **show monitor session erspan-source session** is issued on SW1.

### Example 24-53 *Configuring ERSPAN on SW1*

```
SW4# configure terminal
SW4(config)# monitor session 1 type erspan-source
SW4(config-mon-erspan-src)# description SOURCE-PC-D-TRAFFIC
SW4(config-mon-erspan-src)# source interface GigabitEthernet 1/0/4 rx
SW4(config-mon-erspan-src)# filter vlan 34
SW4(config-mon-erspan-src)# no shutdown
SW4(config-mon-erspan-src)# destination
SW4(config-mon-erspan-src-dst)# ip address 10.123.1.100
SW4(config-mon-erspan-src-dst)# erspan-id 2
SW4(config-mon-erspan-src-dst)# origin ip address 10.34.1.4
SW4(config-mon-erspan-src)# exit
SW4(config)# erspan ttl 32
SW4(config)# end
SW4#
SW4# show monitor session erspan-source session

Type : ERSPAN Source Session
Status : Admin Enabled
Source Ports :
RX Only : Gi1/0/4
Destination IP Address : 10.123.1.100
Destination ERSPAN ID : 2
Origin IP Address : 10.34.1.4
IPv6 Flow Label : None
```

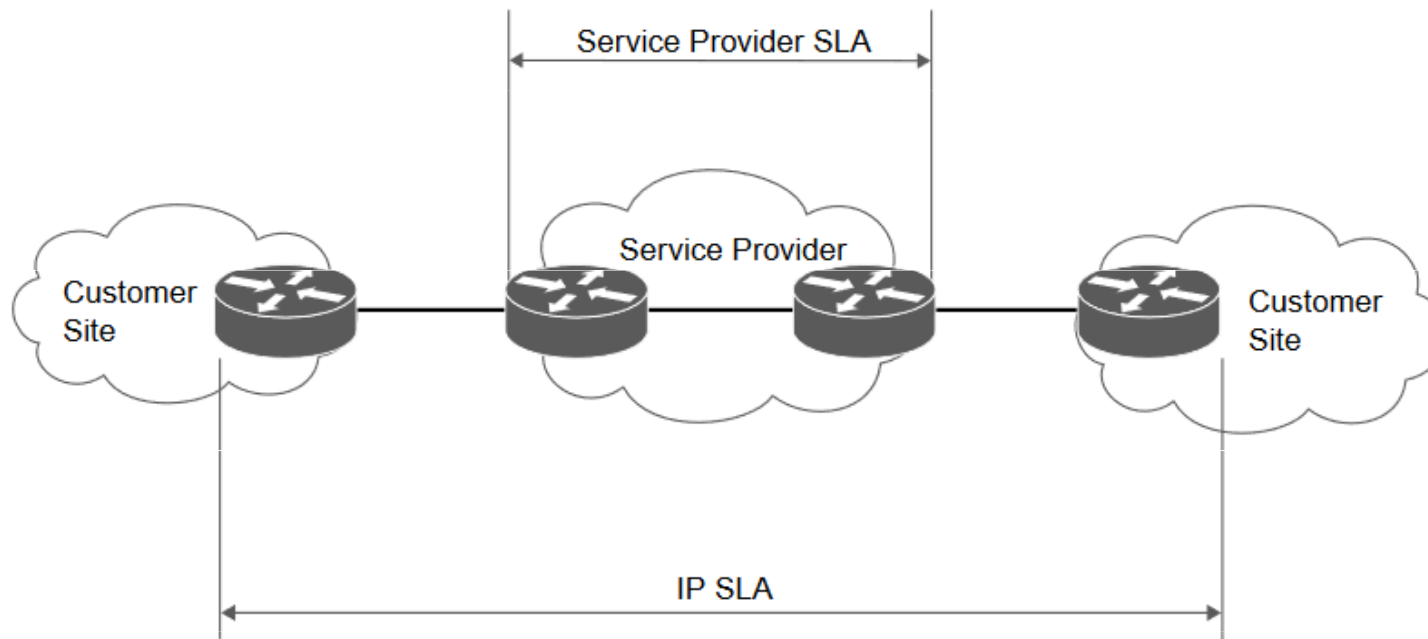
# IP SLA

IP SLA is a tool built into Cisco IOS software that allows for the continuous monitoring of various aspects of the network.

# IP SLA

## IP SLA and Typical Service Provider SLA

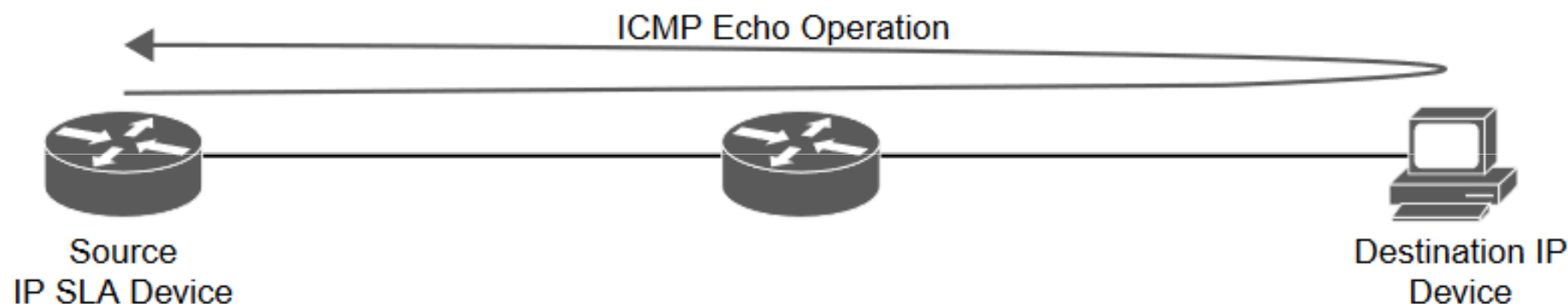
Typically, any SLA received from a service provider only monitors or guarantees the traffic as it flows across the service provider's network. Figure 24-5 shows this scenario and illustrates why IP SLA provides more visibility than a typical service provider SLA.



**Figure 24-5** *IP SLA and Typical Service Provider SLA*

# IP SLA IP SLA Echo Operation

Although there are many different options and probes available for IP SLA, this section focuses only on the ICMP echo and HTTP operations of IP SLA. The ICMP echo operation can functionally be thought of as testing reachability by leveraging ICMP echo and echo replies or pings. Figure 24-6 illustrates how the ICMP echo operation works in IP SLA.



**Figure 24-6** *IP SLA Echo Operation*



# IP SLA

## Configuring IP SLA ICMP Echo Operation on R1

To configure any IP SLA operation, the **ip sla operation-number** command must be used to enter IP SLA configuration mode, where *operation-number* is the configuration for the individual IP SLA probe. Once in IP SLA configuration mode, the command **icmp-echo {destination-ip-address | destination-hostname} [source-ip {ip-address | hostname} source-interface interface-name]** is used to configure the destination IP address of the device or host to be monitored.

The next step is to specify how often the ICMP echo operation should run. This is accomplished by issuing the **frequency seconds** command. Example 24-54 shows the process covered so far on R1.

### Example 24-54 *Configuring IP SLA ICMP Echo Operation on R1*

```
R1(config)# ip sla 1
R1(config-ip-sla)# icmp-echo 192.168.14.100 source-interface Loopback0
R1(config-ip-sla-echo)# frequency 300
R1(config-ip-sla-echo)# end
R1(config)#
```

# IP SLA Scheduling IP SLA 1 on R1

When the IP SLA configuration is complete, an important step is to schedule and activate the IP SLA operation that has been configured. This is where the **ip sla schedule operation-number [life {forever | seconds}] [start-time {[hh:mm:ss] [month day | day month] | pending | now | after hh:mm:ss}] [ageout seconds] [recurring]** command comes into play.

When the IP SLA operation is scheduled, it can be verified with the **show ip sla configuration** command.

Example 24-55 illustrates the configuration steps to schedule the IP SLA 1 operation with a start time of now and a lifetime of forever.

## Example 24-55 Scheduling IP SLA 1 on R1

```
R1(config)# ip sla schedule 1 life forever start-time now
R1(config)# do show ip sla configuration 1
IP SLAs Infrastructure Engine-III
Entry number: 1
Owner:
Tag:
Operation timeout (milliseconds): 5000
Type of operation to perform: icmp-echo
Target address/Source interface: 192.168.14.100/Loopback0
Type Of Service parameter: 0x0
Request size (ARR data portion): 28
Verify data: No
Vrf Name:
Schedule:
  Operation frequency (seconds): 300 (not considered if randomly scheduled)
  Next Scheduled Start Time: Start Time already passed
  Group Scheduled : FALSE
  Randomly Scheduled : FALSE
  Life (seconds): Forever
  Entry Ageout (seconds): never
  Recurring (Starting Everyday): FALSE
  Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 5000
Distribution Statistics:
  Number of statistic hours kept: 2
  Number of statistic distribution buckets kept: 1
  Statistic distribution interval (milliseconds): 20
Enhanced History:
History Statistics:
  Number of history Lives kept: 0
  Number of history Buckets kept: 15
  History Filter Type: None
```

## IP SLA

# Configuring the IP SLA HTTP GET Operation

Another very common use case for IP SLA is to monitor HTTP destinations for operation. This can be done by using the HTTP GET operation of IP SLA. In order to configure this type of monitor, the **ip sla operation-number** command must be used to enter IP SLA configuration mode.

When the operation number is specified, the next step is to configure the HTTP GET probe by issuing the command **http {get | raw} url[name-server ip-address] [version version-number] [source-ip {ip-address | hostname}] [source-port port-number] [cache {enable | disable}] [proxy proxy-url]**.

When the probe is configured, as with any other IP SLA operation, this operation needs to be scheduled by using the command **ip sla schedule operation-number [life {forever | seconds}] [start-time {[hh:mm:ss] [month day | day month] | pending | now | after hh:mm:ss}] [ageout seconds] [recurring]**.

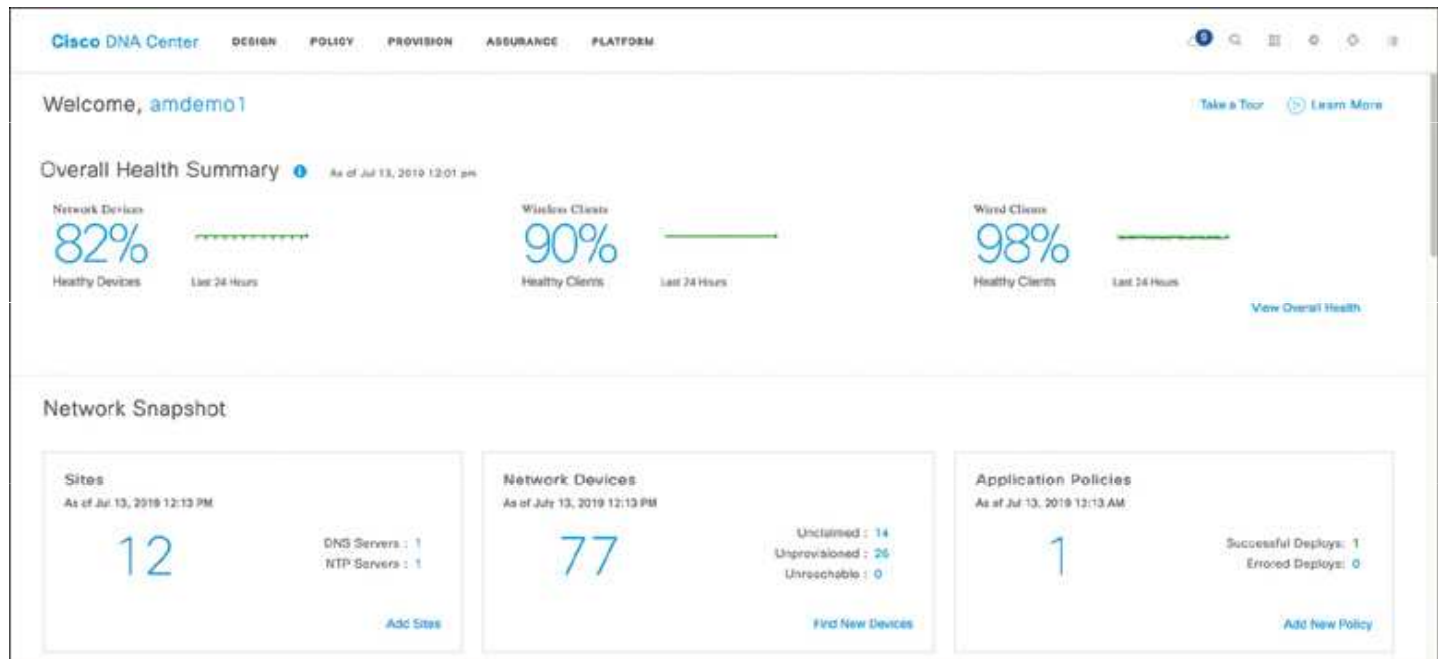
# Cisco Digital Network Architecture (DNA) Center Assurance

- Security has become one the most important pieces of the network, and users expect a better experience.
- Customers demand a simple way to manage Day 0–2 operations and require a scalable and simple approach to running the net-work.
- Cisco DNA Center Assurance provides a tool for handling the most relevant customer requirements.

# Cisco DNA Center Assurance Cisco Digital Network Architecture (DNA) Center

Cisco DNA Center is a powerful network controller and management dashboard, or it is the network management and command center for Cisco DNA, the intent-based network for the enterprise. It can provision and configure all network components, it uses advanced artificial intelligence (AI) and machine learning (ML) to proactively monitor, troubleshoot, and optimize network. It can be integrated with third-party systems for improved operational processes.

Figure 24-7 shows the main Cisco DNA Center page.

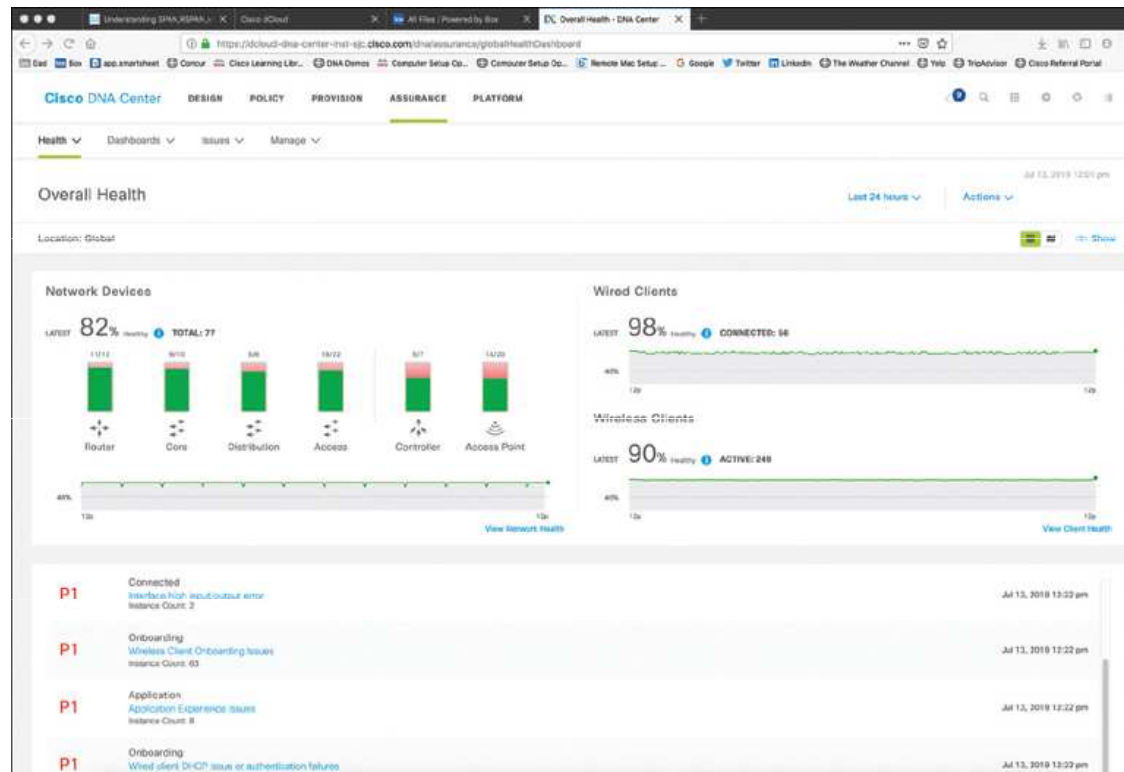


# Cisco DNA Center Assurance

## Cisco DNA Assurance

Cisco DNA Assurance is part of Cisco DNA Center.

- Assurance takes 30+ years of Cisco Technical Assistance Center (TAC) experience and puts it into a tool that uses machine learning to diagnose issues within a network.
- In addition to finding and diagnosing the issues, Assurance gives guided remediation steps to fix the issue. The Assurance tab is shown in Figure 24-8.



# Cisco DNA Center Assurance Cisco DNA Center Client 360 View

From this view, you can do many different things.

- You can click the user's name to see details related to that specific user.
- You can click each device to see specifics about that device in the Client 360 view.

Figure 24-10 shows the entire Client 360 view for the user Grace Smith.



## Cisco DNA Center Assurance

# Cisco DNA Center Search Results

The amount of information that this screen provides is tremendous. The following pieces of information have been gathered automatically:

- Device type
- OS version
- MAC address
- IPv4 address
- VLAN ID
- Connectivity status
- When the device was last seen on the network
- What device it is connected to
- Wireless SSID
- Last known location

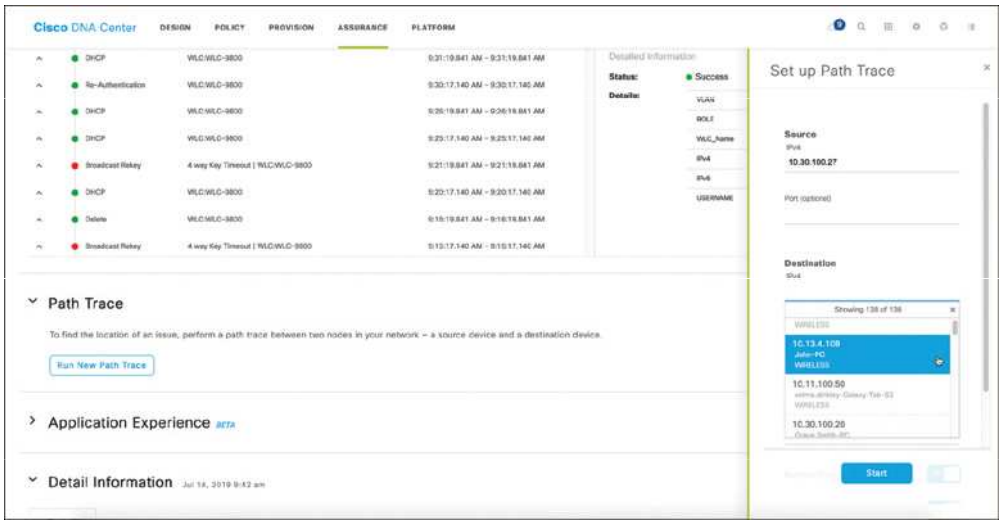
The timeline in the Client 360 view shows issues. This is also a Network Time Travel capability. Since Assurance records the telemetry, it is possible to search back in time to see exactly what has affected the user.



# Cisco DNA Center Assurance Path Trace

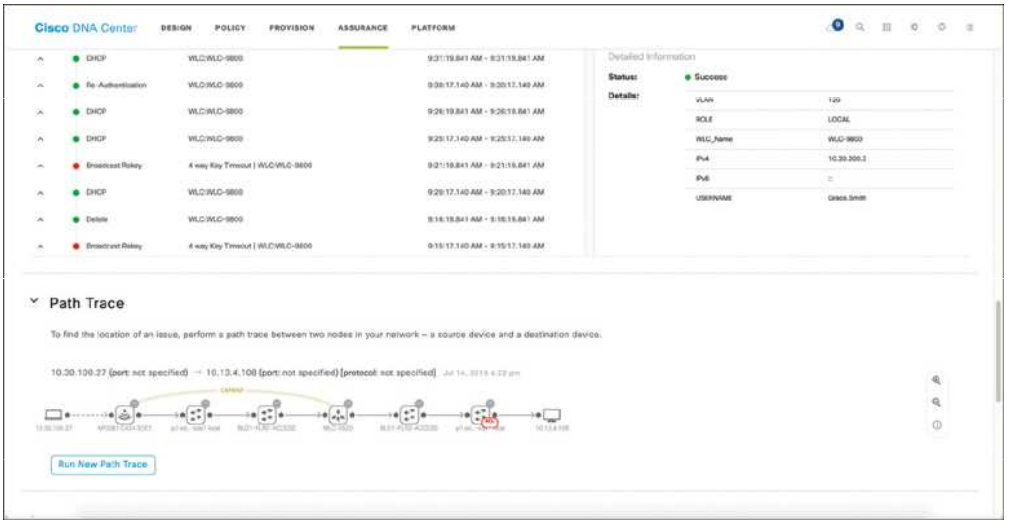
Assurance has a tool called Path Trace. Path Trace is a visual traceroute and diagnostic tool that can be run periodically or continuously, with a specific refresh interval.

Figure 24-11 Cisco DNA Center Assurance Client 360 Path Trace



The path trace output shows a topology view of the traceroute, and in this instance, Path Trace has also detected that there is an access control list (ACL) blocking the traffic from Grace's iPad to John's PC (see Figure 24-12).

Figure 24-12 Cisco DNA Center Assurance Client 360 Path Trace Output

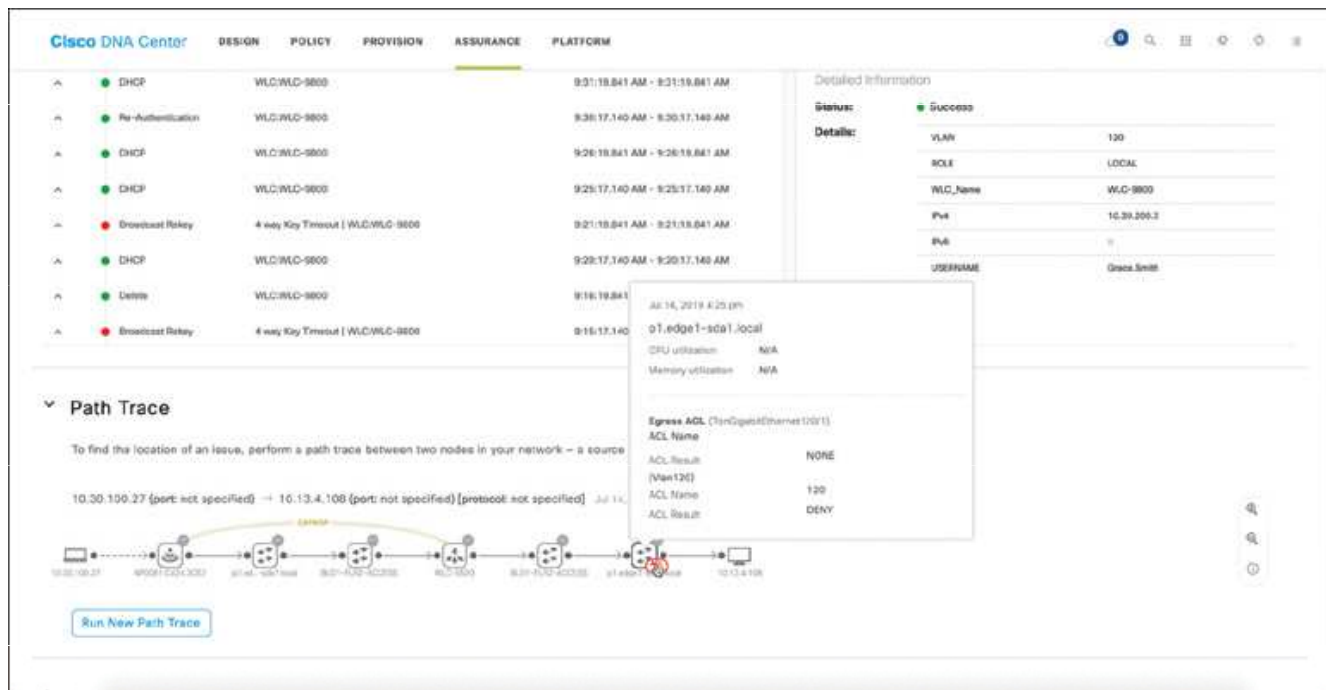


# Cisco DNA Center Assurance Path Trace ACL Information

By hovering over the ACL entry, the following information can be seen:

- The ACL's name
- The interface the ACL is applied to
- The direction (ingress or egress)
- The ACL result (permit or deny)

Figure 24-13 shows the access list information found in this example.



The screenshot displays the Cisco DNA Center Assurance interface. The top navigation bar includes tabs for DESIGN, POLICY, PROVISION, ASSURANCE (highlighted), and PLATFORM. A table lists various ACL entries with columns for status, name, interface, and time range. A 'Detailed Information' panel on the right shows the status as 'Success' and provides details for the selected ACL, including VLAN (120), ACL type (LOCAL), WLC Name (WLC-9800), IP address (10.30.100.2), and Username (Grace Smith). A 'Path Trace' section below the table shows a network diagram with a path from source to destination. A tooltip is visible over the path trace, displaying the following information:

Egress ACL (CiscoSwitchEthernet1/0/1)	
ACL Name	
ACL Result	NONE
(Vlan120)	
ACL Name	120
ACL Result	DENY

# Cisco DNA Center Assurance Client 360 Issues

By clicking on one of the issues listed under Grace's Client 360 view, such as the P1 Onboarding issue, a user can investigate the root cause of the issue.

Figure 24-14 shows the issues that are impacting Grace.

