0	128	64	32	16	8	4	2	1
/24	/25	/26	/27	/28	/29	/30	/31	/32
	128	192	224	240	248	252	254	255
1Q	0.5Q	0.25Q	0.125Q					
	802.11 MAC Frame Format	FC	DUR	ADD1	ADD2	ADD3	SEQ	ADD4	DATA	FCS
	/23	/22	/21	/20	/19	/18	/17	/16
	254	252	248	240	224	192	128	0
	2Q	4Q	8Q	16Q	32Q	64Q	128Q	256Q
# of hosts	512	1,024	2,048	4,096	8,192	16,384	32,768	65,536

Private IPv4 addresses

10.0.0.0/8 10.0.0.0 →10.255.255.255
172.16.0.0/12 172.16.0.0→172.31.255.255
192.168.0.0/16 192.168.0.0→192.168.255.255

224.0.0.0 →239.255.255.255 Multicast

*Wildcard bit subtract 255

IPv6 Multicast Scope

Interface Local → FF01::/16
Link Local →FF02::/16(not route)
Site Local → FF05::/16
Organizational Local → FF08::/16
Global → FF0E::/16

EUI-64

48bits MAC → 64-bit interfaceID

1. Divide MAC add. half
1234 5678 90AB→123456 | 78 90AB
2. Insert FF FE in the middle
1234 56FF FE78 90AB
3. Invert 7th bit

IPv6 vs IPv4 Multicast

All hosts - FF02::1→224.0.0.1
All routers - FF02::2→224.0.0.2
All OSPF routers - FF02::5→224.0.0.5
All OSPF DRs/BDRs - FF02::6→224.0.0.6
All RIP routers - FF02::9→224.0.0.9
All EIGRP routers - FF02::A→224.0.0.10

IPV6 NDP,DAD

No Broadcast in IPv6
NDP replaces ARP
NS (ICMPv6 Type 135)
RS → all routers to ID-FF02::2
RA → all nodes via FF02::1
DAD→Duplicate Add. Detection

STP

ID→ {Bridge Priority (32,768) + MAC} (↑lowest)
BLOCKING(DISCARDING) Speed(Mbps) → Cost
~~LISTENING~~ 10→100
LEARNING 100→19
FORWARDING 1G→4
Hello 2 seconds 10G→2
Max age/dead timer (2*10) 20 seconds
IEEE→Cisco
STP 802.1D→PVST+
RSTP 802.1w→Rapid PVST+
Multiple STP(802.1s)

EtherChannel

PAGP (Cisco) → LACP(IEEE)
8 channels / interface → 16 (8 standby)
Auto + Auto (No negotiation)→ Passive + Passive
Auto + Desirable (Successful)→ Passive + Active
Auto + On (No negotiation) → Passive + On
Desirable + Desirable (Successful)→Active+Active

FHRP

HSRP(Active/Standy) 0000.0c07.acXX Group #
HSRP v2 0000.0c9F.FXXX (Cisco)
VRRP (Master/Backup) 0000.5e00.01XX (IEEE)
GLBP (AVG/AVF) 0007.b400.XXYY (Cisco)

ACL

Standard 1-99, 1300-1999 → Destination
Extended 100-199, 2000-2699 → Source
One ACL per direction per protocol (4/int)

CDP vs LLDP (802.1AB)

Enabled by default → Disabled by default
messages 60 secs → 30 secs
holdtime 180 secs → 120 secs
no cdp run → lldp run
no cdp enable → lldp transmit/receive

OSPF

Cost(100 default)=ref. bandwidth / interface bandwidth

Down	Init	2-way	Exstart	Exchange	Loading	Full
	DR/BDR			LSDB/LSA	LRS/LSAck	

HELLO 10seconds, DEAD 40seconds (by default)
DBD - summary of LSDB
LSR - request specific LSAs from neighbor
LSU - send specific LSAs to neighbor

Router-ID
1. manually configured
2. highest loopback int. IPs
3. highest physical int. IPs

	AD
Connect	0
Static	1
eBGP	20
EIGRP	90
OSPF	110
IS-IS	115
RIP	120
iBGP	200

NTP (UDP Port 123)

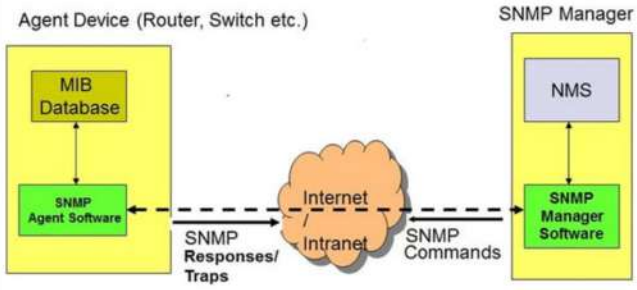
R1#show clock
R1#show clock detail
R1#calendar set
R1#clock update-calendar (clock read-c)
Stratum 0 (ref./atomic/GPS clock) 15max
server,client, symmetric active mode

DHCP (DORA)

Discover - client→server (broadcast)
Offer - server→client(broadcast/unicast)
Request - client→server(broadcast)
Ack - server→client(unicast)

ip helper-address 192.168.10.10 (server)
R1#show ip dhcp binding
1/2 lease time, client renews IP lease

SNMP Architecture



SNMP

SNMPv1 - original version
SNMPv2c - Community string as passwords
SNMPv3 - Encryption & Authentication
Read-NMS →Devices (Get,GetNext,GetBulk)
Write- NMS → Devices (Set)
Notification - Devices → NMS (Trap, Inform)
Response - (Response)

Syslog

Emergency	0
Alert	1
Critical	2
Error	3
Warning	4
Notice	5
Informational	6
Debugging	7 (default)

NAT

Inside local→Inside global→Outside global→Outside local
R1(config)#ip nat inside source list 1 interfaceg0/0 overload

Port Security

Shutdown (err disable, syslog/SNMP, violation counter +1)
Restrict (not disable, discard traffic, violation counter +1)
Protect (not disable, no syslog/SNMP, discard traffic, no count)

Redundancy

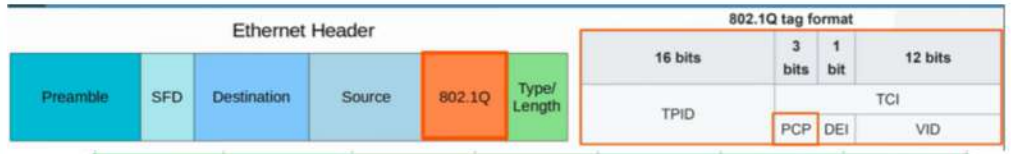
Single Homed → 1 SP, 1 Connection
Dual Homed → 1 SP, 2 Connections

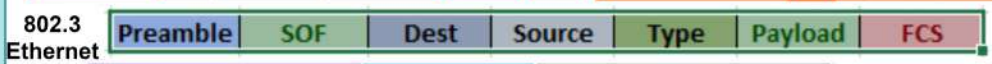
Multi-Homed → 2SPs, 1 Connection
Dual Multi-Homed→2SPs, 2 Connections

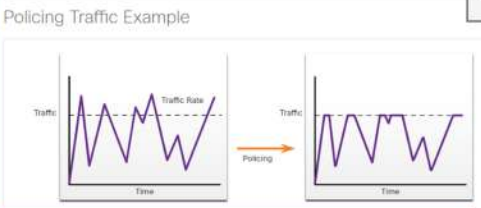
QoS

►One way delay - 150ms or less
►Jitter - 30ms or less
►Loss - 1% or less
►Bandwidth

PCP value	Traffic types
0	Best effort (default)
1	Background
2	Excellent effort
3	Critical applications
4	Video
5	Voice
6	Internetwork control
7	Network control







IPP (3 bits)& DSCP (6 bits)

	Lowest drop precedence →		High
High Priority	AF41 (34)	AF42 (36)	AF43 (38)
↑	AF31 (18)	AF32 (20)	AF33 (22)
↑	AF21 (18)	AF22 (20)	AF23 (22)
Lowest Priority	AF11 (10)	AF12 (12)	AF13 (14)

Wi-Fi

►2.4GHz non overlap → 1,6,11 (5GHz has none)
►CSMA/CD - wired avoidance half duplex
►CSMA/CA - wireless avoidance half duplex
►Signals are affected by Absorption, Reflection, Diffraction, and Scattering.

Gen.	IEEE	GHz
Wi-Fi 7	802.11be	2.4/5/6
Wi-Fi 6&6E	802.11ax	2.4/5/6
Wi-Fi 5	802.11ac	5
Wi-Fi 4	802.11n	2.4/5

LAN ARCHITECTURE

Three-tier (Core,Distribution, Access)
Two-tier (Collapsed Core) (Distribution, Access)
Spine-leaf(Spines↔Leaves,Not to each other)

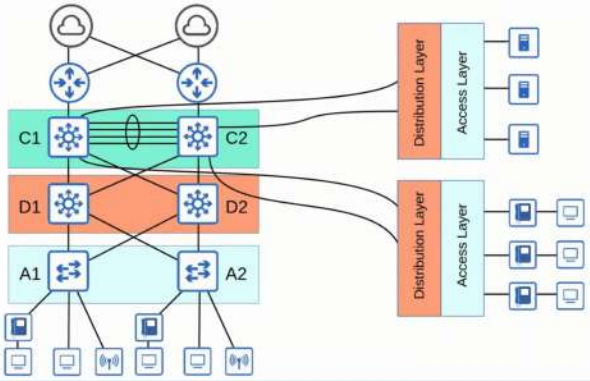
Core Layer → speed/fast transport →Dist. Layer
Dist. Layer → aggregates/connects to WAN & 🌐
Access Layer → connects end hosts, QoS, port security, DAI, PoE APs, IP phones

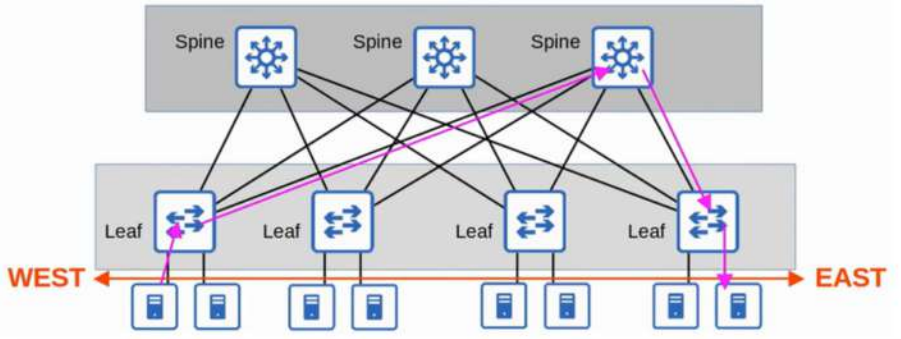
Wireless LAN

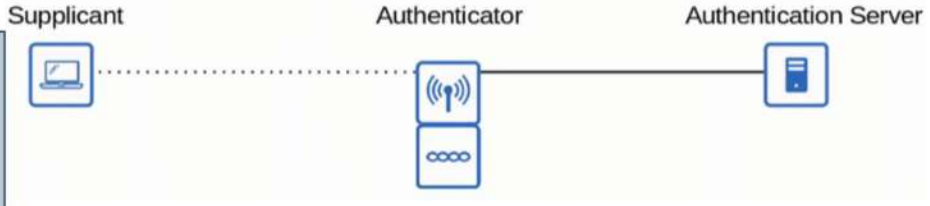
Three types of Service Sets
►IBSS (ad hoc) independent
►Infrastructure (BSS, ESS) passing btw APs in a ESS roaming
►Mesh (MBSS)
SSID (not unique human readable)
BSSID (unique MAC add. of AP)
upstream wired network is DS

Service Sets

•ESS (Extended Service Set)
•AP with own BSS connected by wired network
•Each BSS→ same SSID
•Each BSS has unique BSSID (MAC add. of APs)
•Each BSS uses different channel to avoid inf.
•BSA should overlap 10-15%
•Roaming seamlessly







802.1X authentication architecture

802.1X Probe Request

►Management: used to manage BSS
→Beacon
→Probe request/response
→Authentication
→Association request/response
►Control: used to control access to medium (radio F.)
→RTS (Request to Send)
→CTS (Clear to Send)
→ACK
►Data: used to send actual data packets.

WLC

►WLC ↔ lightweight APs use CAPWAP tunnels
►Control tunnel UDP port 5246
►Data tunnel UDP port 5247

Credits
@JeremysITLab
@CiscoNetAcad
@Any original authors

This cHeatsHeet is designed by
@Henori100

Network Automation

Logical Planes

►Data plane →forward data/traffic (802.1qVlantags, NAT, ACLs, port security

►Control plane → routing table, MAC tables, ARP, STP, etc

►Management → SSH/Telnet, Syslog, SNMP, NTP

Southbound Interface (SBI)

►SBI →communicates btw controller and all network devices

APIs facilitate data exchanges between programs

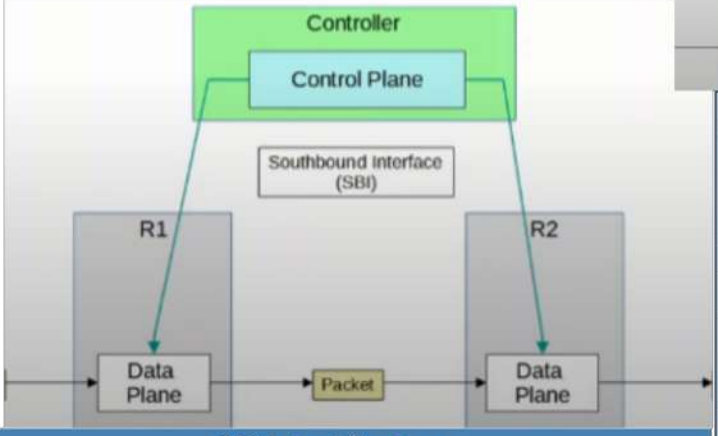
Some examples of SBIs:

→OpenFlow

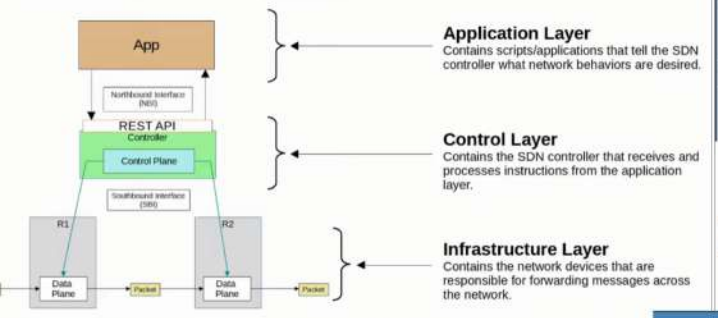
→Cisco OpFlex

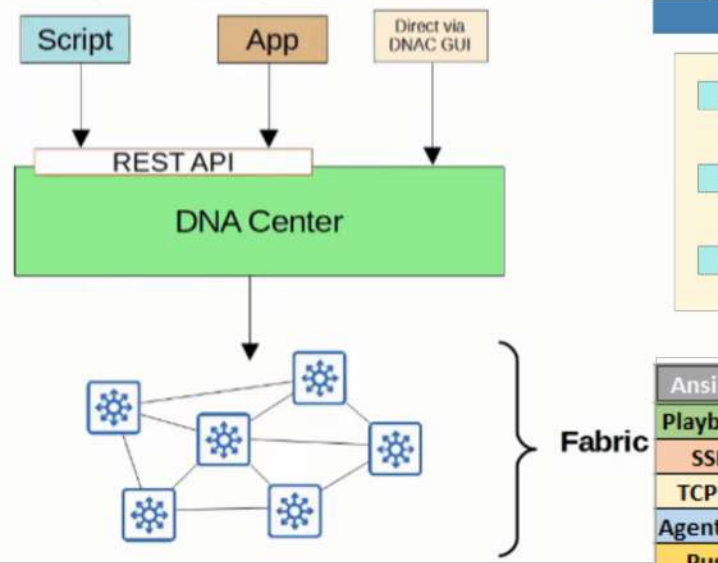
→Cisco onePK (Open Network Environment Platform Kit)

→NETCONF



SDN Architecture



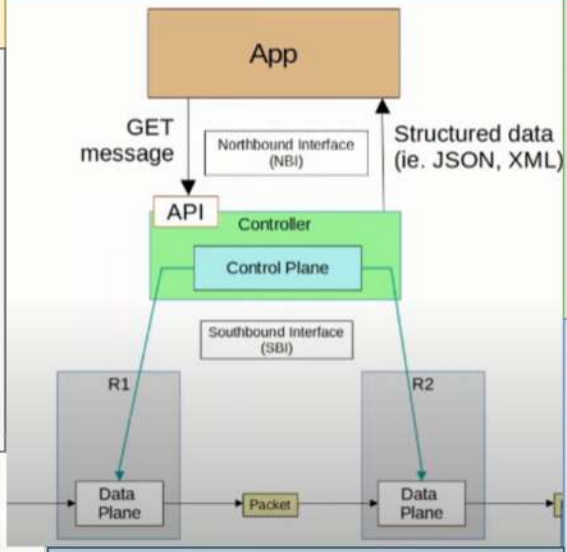


Northbound Interface (NBI)

►NBI →uses SBI to communicate with all managed devices and gathers →networked devices, topology, available interfaces, their configurations

A Rest API is used on the controller as an interface for apps to interface with it.

→Data is sent in a structured (serialized) format such as JSON or XML



Cisco SD Access

►SD-Access is SDN solution for automating campus LANs

→ACI automating Data Center

→SD-WAN automating WANs

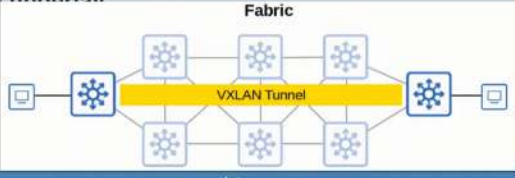
►Cisco DNA is the controller at the center of SD-Access.

►Underlay→physical network of devices and connections (IP connectivity IS-IS)

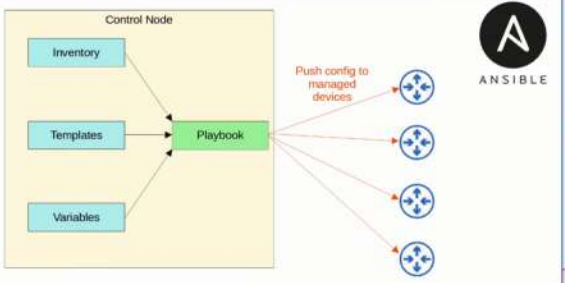
►Overlay→virtual network built ↑ physical underlay network

→SD-Access uses VXLAN to build tunnels

►Fabric→combination of overlay and underlay



Ansible



Ansible	Puppet	Chef
Playbook	Manifest	Recipe,Run-list
SSH	HTTPS (via REST API)	HTTPS (via REST API)
TCP 22	8140	10002
Agentless	Agent-based	Agent-based
Push	Pull	Pull

REST APIs

►Uniform interface

►Client-server

►Stateless

►Cacheable or non-cacheable

►Layered system

►Code-on-demand(optional)

JSON

►REST API often uses JSON

►Whitespace is insignificant

►Four 'primitive' data types:

→String, Number, Boolean, Null

►Two structured data types

→Object and Array

YAML

►Used in Ansible

►Whitespace is Significant

Ansible

►written in Python (Red Hat)

►Push Model, Agentless, Uses SSH to connect to devices, make configuration changes, extract info.

►Playbooks (written in YAML)

►Inventory (written INI, YAML)

►Templates (in Jinja2)

►Variables (in YAML)

Puppet

►written in Ruby

►Pull Model, Agent-based

►Puppet master/server

►Client uses TCP8140→Master

►Proprietary language for files

►Manifast, Templates

Chef

►written in Ruby

►Pull Model, Agent-based

►Not all Cisco devices support

►Server→TCP10002, files →DSL

►Resources: Recipes, Cookbooks, Run-list