



Network Management



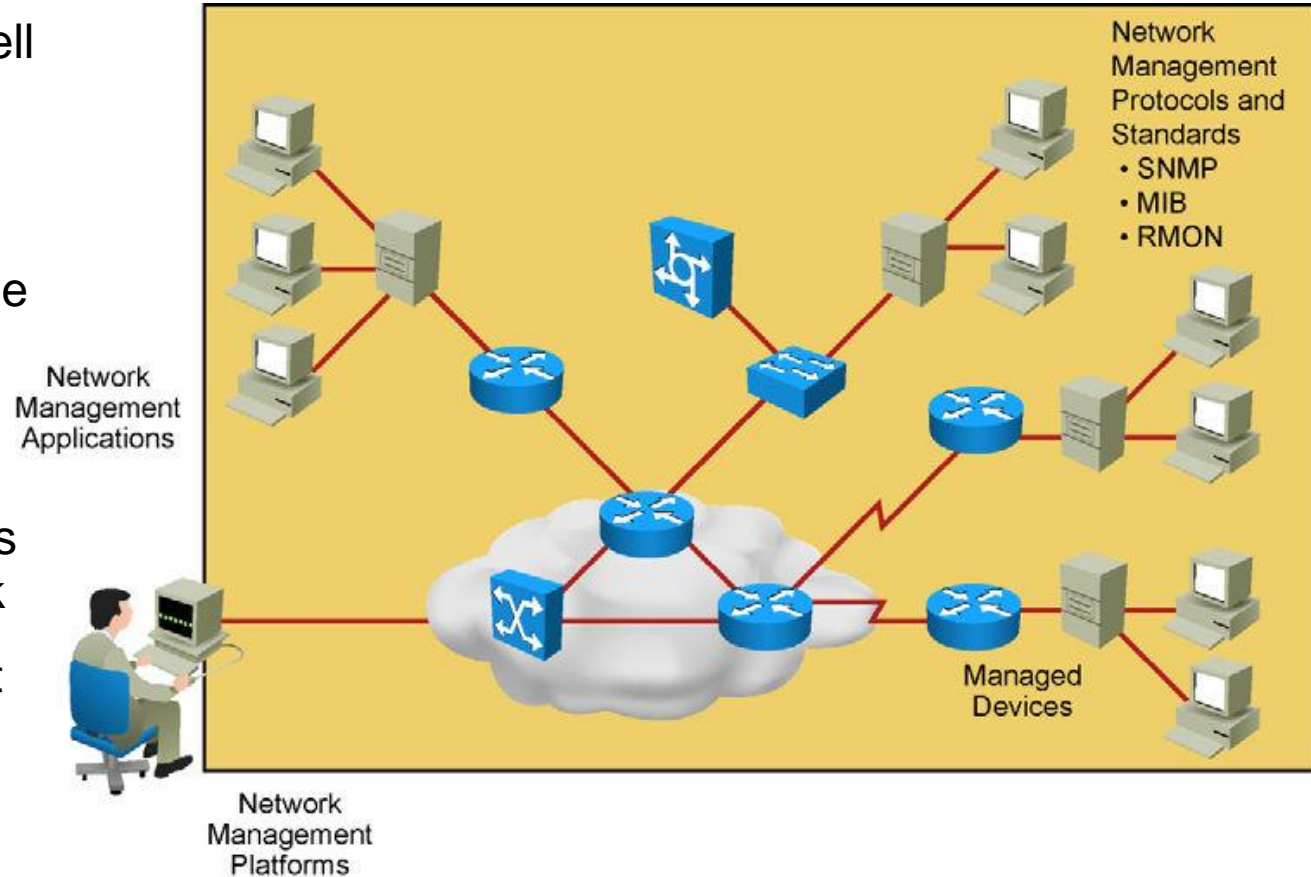
SWITCH Module 7

Agenda

- **NTP**
- **Syslog**
- **SNMP**
- **IP SLAs**
- **AAA**
- **Identity-based Routing**

Network Management Overview

- Ability to verify the network is working well and behaving in the planned manner
- Ability to characterize the performance of the network
- Ability to understand how much traffic is flowing and where it is flowing in the network
- Ability to troubleshoot the network



NTP



The Need for Accurate Time

- The need for accurate time is increasing year by year
- Coordinating events, marking logs, and kicking-off scripts all run based on a system clock
- Therefore, in today's network, coordination of system clocks and their accuracy is increasing in importance
- From a best practice perspective, it is recommended to set clocks on all network devices to UTC regardless of their location, and then configure the time zone to display the local time if desired. In this manner, global operations can fall back to UTC time for relative time

Configuring the System Clock Manually

```
Switch# show clock
10:10:03.979 UTC Thu Feb 22 2001
! Shows what the device thinks is the current time
Switch# clock set 12:13:00 10 January 2015
! Manual system clock reconfiguration
Switch# show clock detail
12:13:03.487 UTC Sat Jan 10 2015
Time source is user configuration
! Verification of how system clock has changed. Adding the detail keyword will tell
you what was the source of clock configuration
```

```
Switch(config)# clock timezone EDT -5
Switch(config)# clock summer-time EDT recurring
! Changes timezone and enables daylight savings time. In this example, EDT is used.
Switch# show clock detail
07:44:12.370 EDT Sat Jan 10 2015
Time source is user configuration
Summer time starts 02:00:00 EDT Sun Mar 8 2015
Summer time ends 02:00:00 EDT Sun Nov 1 2015
! Verifies how clock settings now reflect local time
```

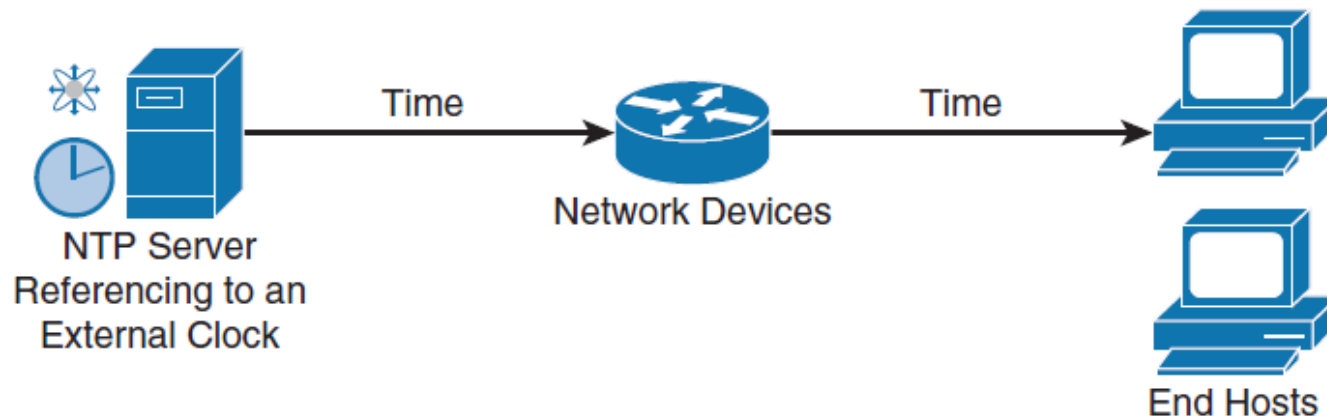
Setting Summer Time

- **clock summer-time zone recurring** [week day month hh:mm week day month hh:mm [offset]]
- **clock summer-time zone date** date month year hh:mm date month year hh:mm [offset]

Parameter	Description
<i>zone</i>	Name of the time zone (for example, PDT) to be displayed when summer time is in effect
<i>recurring</i>	Indicates that summer time should start and end on the corresponding specified days every year
<i>date</i>	Indicates that summer time should start on the first specific date that is listed in the command and end on the second specific date in the command
<i>week</i>	(Optional) Week of the month (1 to 5 or last).
<i>day</i>	(Optional) Day of the week (Sunday, Monday, and so on)
<i>date</i>	Date of the month (1 to 31)
<i>month</i>	(Optional) Month (January, February, and so on)
<i>year</i>	Year (1993 to 2035)
<i>hh:mm</i>	(Optional) Time (military format) in hours and minutes
<i>offset</i>	(Optional) Number of minutes to add during summer time (default = 60)

Network Time Protocol Overview

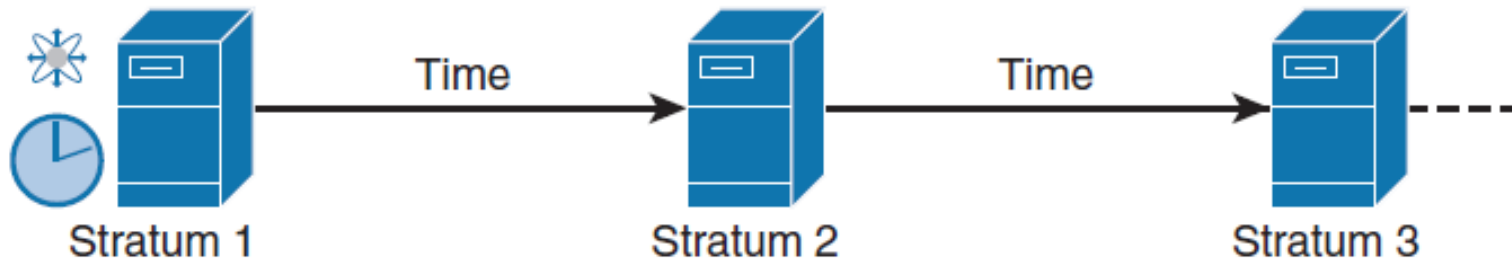
- Manually setting the clocks of any network device is neither accurate nor scalable
- The best practice is to use Network Time Protocol (NTP), Simple NTP (SNTP), or Precision Time Protocol (PTP)
- NTP is designed to synchronize the time throughout an entire network infrastructure, including servers, switches, routers, host machines, wireless access points, uninterruptible power supply (UPS), and so on
- NTP leverages UDP port 123 for both the source and destination by default.



Network Time Protocol Overview

- An NTP network usually gets its reference time from an authoritative time source, such as a radio clock, GPS, or an atomic clock attached to an NTP time server somewhere in the network.
- NTP then distributes this time across the network.
- Accurate timekeeping is made possible by exchanging NTP messages between each pair of machines (server/client) with an association.
- However, in a LAN environment, NTP can be configured to use IP broadcast messages instead.
- To keep accuracy of time, NTP uses the concept of a stratum to describe how many NTP hops away a machine is from an authoritative time source.
- A machine running NTP automatically chooses the machine with the lowest stratum number

NTP: Stratum



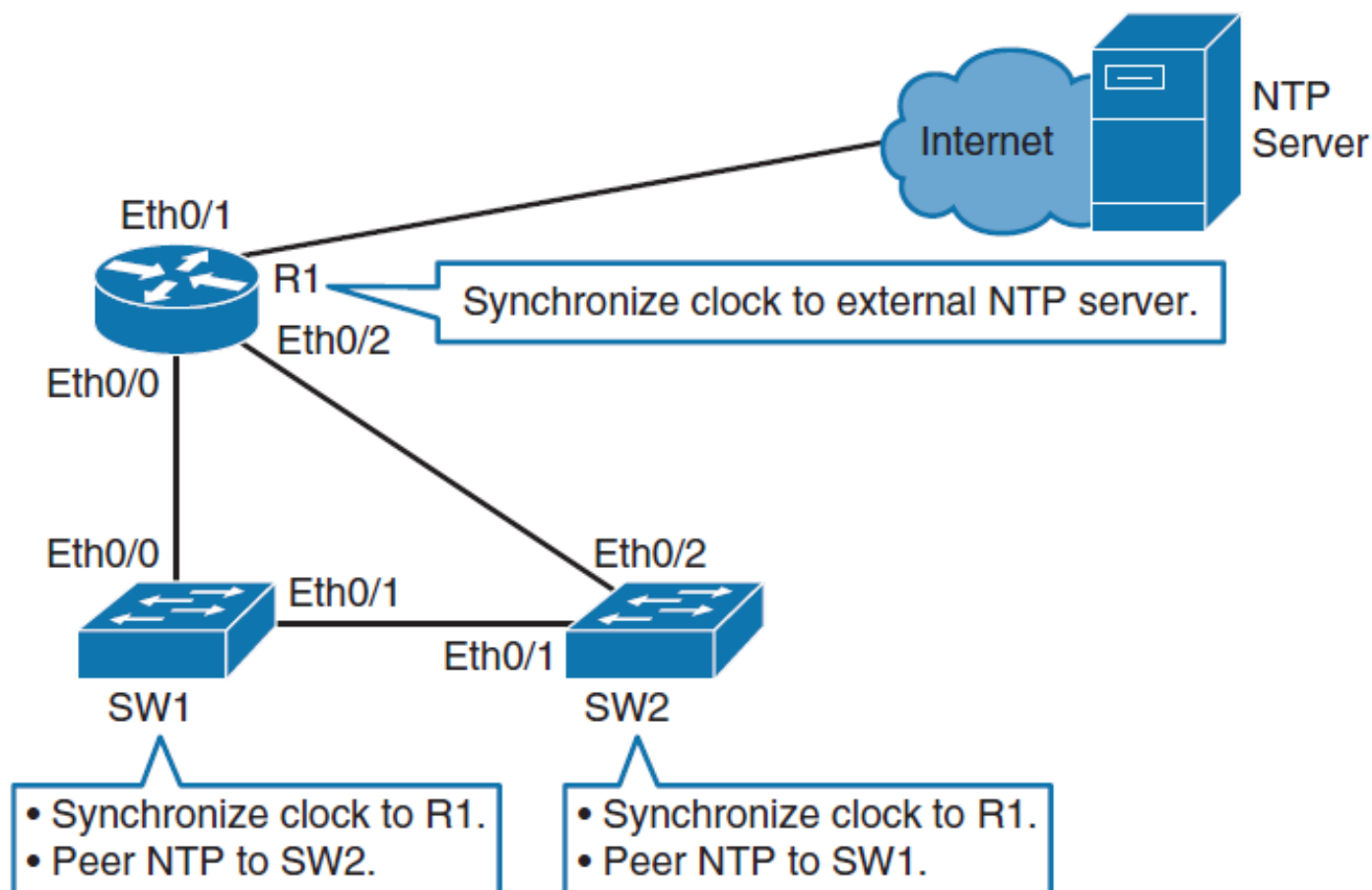
- NTP avoids in two ways synchronizing to a machine whose time may not be accurate.
 - NTP never synchronizes to a machine that is not synchronized itself.
 - NTP compares the time that is reported by several machines and will not synchronize to a machine whose time differs significantly from the others, even if its stratum is lower.

NTP Modes

- A device may take on more than one role at a time.
- **Server**
 - Provides accurate time information to clients on the network.
- **Client**
 - Synchronizes its time to an NTP server. This mode is most suited for file server and workstation clients that are not required to provide any form of time synchronization to other local clients. It can also provide accurate time to other devices.
- **Peers**
 - Peers only exchange time synchronization information.
- **Broadcast/multicast**
 - Special “push” mode of NTP server where the local LAN is flooded with updates; used only when time accuracy is not an issue.

NTP Example

```
ntp server 209.165.200.187
```



Verify NTP

```
R1# show ntp status
```

```
Clock is synchronized, stratum 2, reference is 209.165.200.187  
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10  
ntp uptime is 1500 (1/100 of seconds), resolution is 4000  
reference time is D67E670B.0B020C68 (05:22:19.043 PST Mon Jan 13 2014)  
clock offset is 0.0000 msec, root delay is 0.00 msec  
root dispersion is 630.22 msec, peer dispersion is 189.47 msec  
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s  
system poll interval is 64, last update was 5 sec ago.
```

```
R1# show ntp associations
```

address	ref clock	st	when	poll	reach	delay	offset	disp
*~209.165.200.187	.LOCL.	1	24	64	17	1.000	-0.500	2.820

* sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured

Setting and Verifying the Clock Time Zone and Daylight Savings Time

```
R1(config)# clock timezone EDT -5
R1(config)# clock summer-time EDT recurring

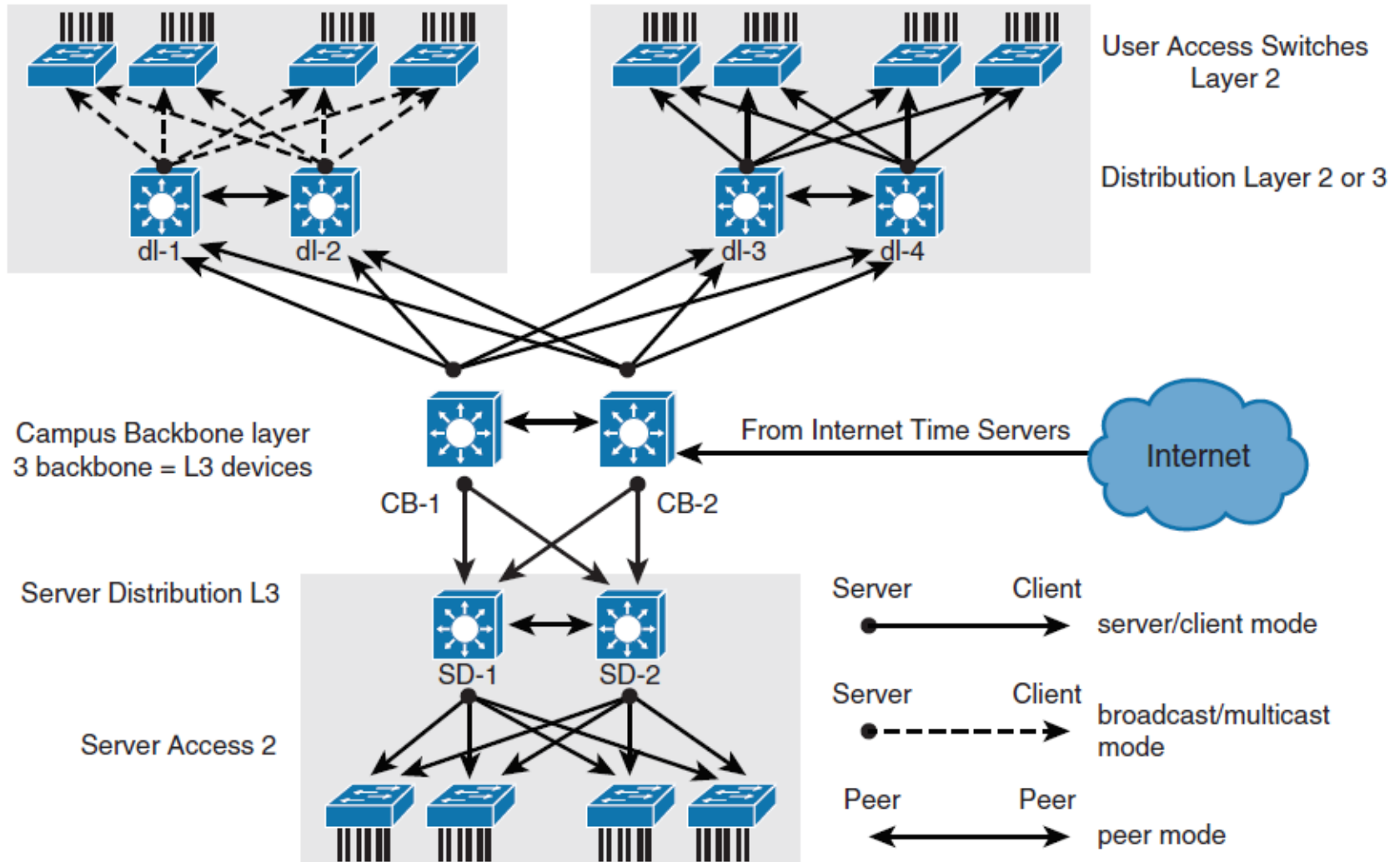
R1# show clock detail
08:01:54.470 EDT Tue Jan 14 2014
Time source is NTP
Summer time starts 02:00:00 EDT Sun Mar 9 2014
Summer time ends 02:00:00 EDT Sun Nov 2 2014
```

Downstream NTP Example

```
SW1(config)# ntp server 10.0.0.1  
SW1(config)# clock timezone EDT -5  
SW1(config)# clock summer-time EDT recurring
```

```
SW1# show ntp status  
Clock is synchronized, stratum 3, reference is 10.0.0.1  
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**18  
reference time is D67FD8F2.4624853F (10:40:34.273 EDT Tue Jan 14 2014)  
clock offset is 0.0053 msec, root delay is 0.00 msec  
root dispersion is 17.11 msec, peer dispersion is 0.02 msec
```

NTP Design Principles



Securing NTP

NTP authentication steps:

- **Step 1.** Define NTP authentication key or keys with `ntp authentication-key` command. Every number specifies a unique NTP key.
- **Step 2.** Enable NTP authentication using the `ntp authenticate` command.
- **Step 3.** Tell the Cisco device which keys are valid for NTP authentication using the `ntp trusted-key` command. The only argument to this command is the key that you defined in the first step.
- **Step 4.** Specify the NTP server that requires authentication by using the `ntp server ip-address key key-number` command. You can similarly authenticate NTP peers by using the `ntp peer ip-address key key-number` command.

NTP Authentication Example

```
NTPServer(config)# ntp authentication-key 1 md5 MyPassword
NTPServer(config)# ntp authenticate
NTPServer(config)# ntp trusted-key 1
NTPClient(config)# ntp authentication-key 1 md5 MyPassword
NTPClient(config)# ntp authenticate
NTPClient(config)# ntp trusted-key 1
NTPClient(config)# ntp server 10.0.1.22 key 1
```

NTP ACL's

For NTP, you can configure the following four restrictions through access lists:

- **Peer**

- Time synchronization requests and control queries are allowed. The device is allowed to synchronize itself to remote systems that pass the access list.

- **Server:**

- Time synchronization requests and control queries are allowed. The device is *not* allowed to synchronize itself to remote systems that pass the access list.

- **Server-only**

- Only allows synchronization requests.

- **Query-only**

- Only allows control queries.

NTP Access List Example

```
Router(config)# access-list 1 permit 10.0.1.0 0.0.255.255  
Router(config)# ntp access-group peer 1
```

```
Router(config)# access-list 1 permit 10.1.0.0 0.0.255.255  
Router(config)# ntp access-group server-only 1
```

NTP Source Address

- The source of the NTP packet will be the same as the interface the packet was sent out on.
- When implementing authentication and access lists, it is good to have a specific interface set to act as the source interface for NTP.
- It would be wise of you to choose a loopback interface to use as the NTP source.
- This is because the loopback will never be down like physical interfaces.
- If you configured loopback 0 to act as the NTP source for all communication and that interface has, for example, an IP address of 192.168.12.31, you can write up just one access list that will allow or deny based on one single IP address of 192.168.12.31.

NTP Versions

- NTPv4 is an extension of NTP Version 3. NTPv4 supports both IPv4 and IPv6 and is backward compatible with NTPv3.

NTPv4 adds the following capabilities:

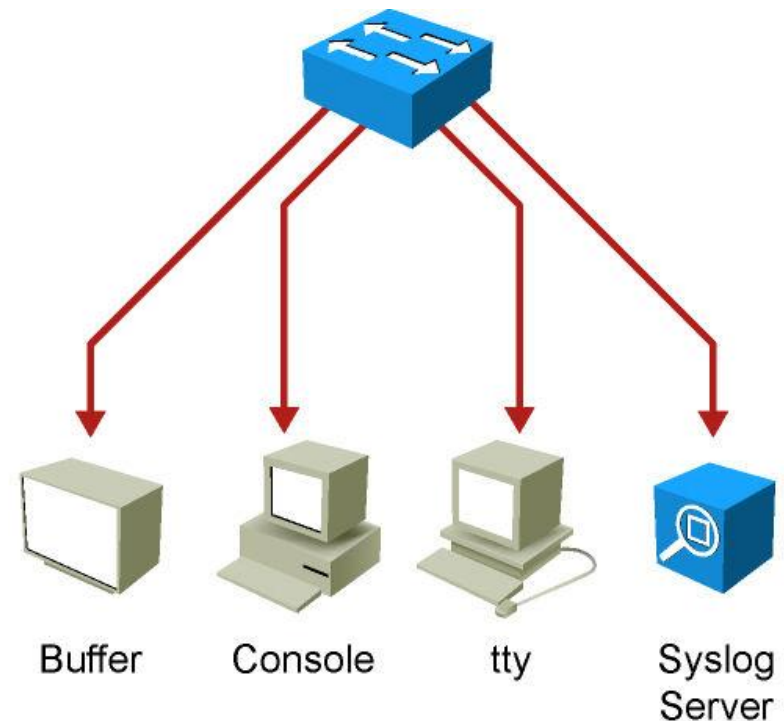
- Support for IPv6
- Better security
- Leverages multicast over broadcast for push modes

Syslog



Syslog

- Networking protocol which provides comprehensive reporting mechanism
- UDP port 514
- Logs are sent in plain text
- Supported almost on all network devices (also on Windows/MAC/Linux servers)



Syslog Severity Level


- The lower number, the more serious the situation

Syslog severity terminology	Syslog severity Cisco terminology
Emergency	Level 0, the most serious
Alert	Level 1
Critical	Level 2
Error	Level 3
Warning	Level 4
Notice	Level 5
Informational	Level 6
Debugging	Level 7, the least serious

Syslog Message Format

- **Facility:** A code consisting of two or more letters that indicates the message origin (hardware device, protocol, system software etc.)
 - Cisco IOS has more than 500 facilities (e.g. IP, OSPF, SYS, IPsec, RSP, IF)
- **Severity:** A single-digit code from 0 to 7 that reflects the severity
- **Mnemonic:** A code that uniquely identifies the error message
- **Message-text:** Text string describing the condition

%FACILITY-SUBFACILITY-SEVERITY-MNEMONIC: Message-text



The diagram illustrates the mapping of the example message to the format fields. Colored lines connect the fields to their corresponding parts of the message: a yellow line for Facility (SYS), a blue line for Subfacility (5), a red line for Severity (CONFIG), and a purple line for Mnemonic (I). The Message-text is the entire string following the colon.

```
%SYS-5-CONFIG_I: Configured from console by  
cwr2000 on vty0 (192.168.64.25)
```

Configuring Syslog

- Setup Syslog server and severity level:

```
! Syslog configuration
Switch(config)# logging host 192.0.2.1
Switch(config)# logging trap ?
<0-7>          Logging severity level
alerts         Immediate action needed          (severity=1)
critical       Critical conditions               (severity=2)
debugging      Debugging messages               (severity=7)
emergencies    System is unusable               (severity=0)
errors         Error conditions                  (severity=3)
informational  Informational messages           (severity=6)
notifications  Normal but significant conditions (severity=5)
warnings       Warning conditions                (severity=4)

Switch(config)# logging trap informational
```

- Setup circular local Syslog buffer:

```
! Local logs configuration
! Parameters: local log size and the severity level that has to be logged
Switch(config)# logging buffered 10000 6
```

Verifying

```
Switch# show logging
Syslog logging: enabled (11 messages dropped, 0 messages rate-limited,
                  0 flushes, 0 overruns, xml disabled, filtering disabled)
  Console logging: level debugging, 174 messages logged, xml disabled,
                  filtering disabled
  Monitor logging: level debugging, 0 messages logged, xml disabled,
                  filtering disabled
  Buffer logging: level informational, 3 messages logged, xml disabled,
                  filtering disabled
  Logging Exception size (4096 bytes)
  Count and timestamp logging messages: disabled
No active filter modules.
  Trap logging: level informational, 43 message lines logged
    Logging to 192.0.2.1(global) (udp port 514, audit disabled, link up),
    2 message lines logged, xml disabled, filtering disabled
Log Buffer (10000 bytes):

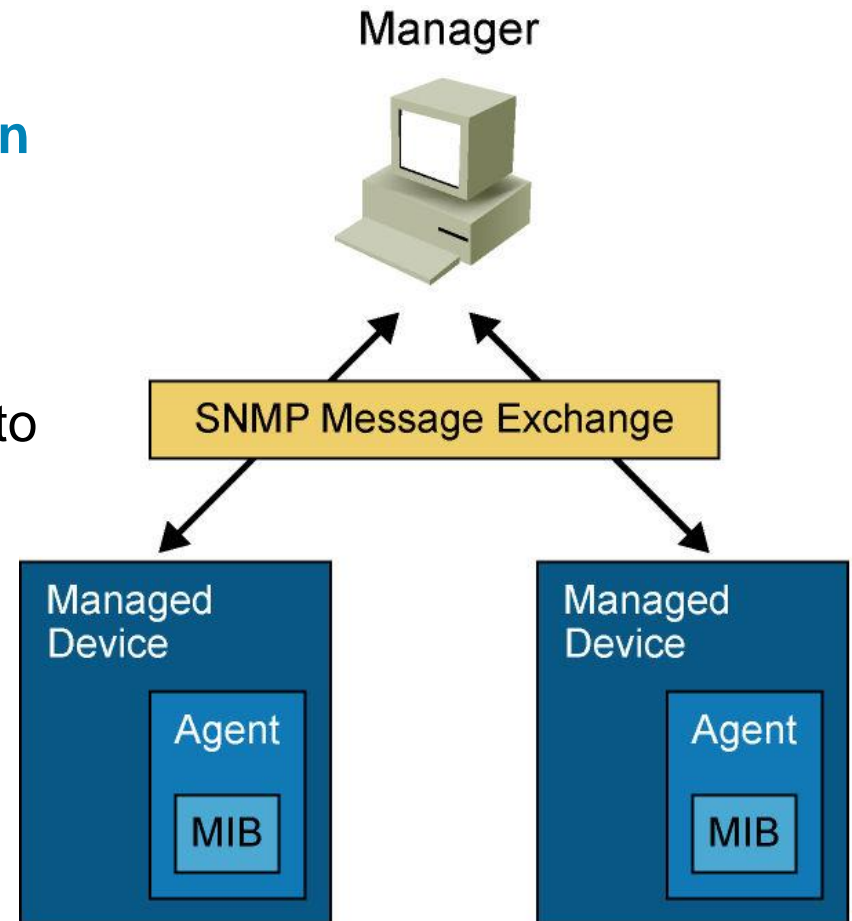
*Mar  1 01:40:47.395: %SYS-5-CONFIG_I: Configured from console by console
...
```

Simple Network Management Protocol



Simple Network Management Protocol (SNMP)

- SNMP contains three elements:
 - **Network Management Application**
 - **SNMP Agents** – running inside a managed device
 - **MIB database** – describes the information that the agent can use to populate the data
- SNMP modes:
 - **Pull model** – manager periodically polls the SNMP agents
 - **Push model** – agents inform the manager of certain events

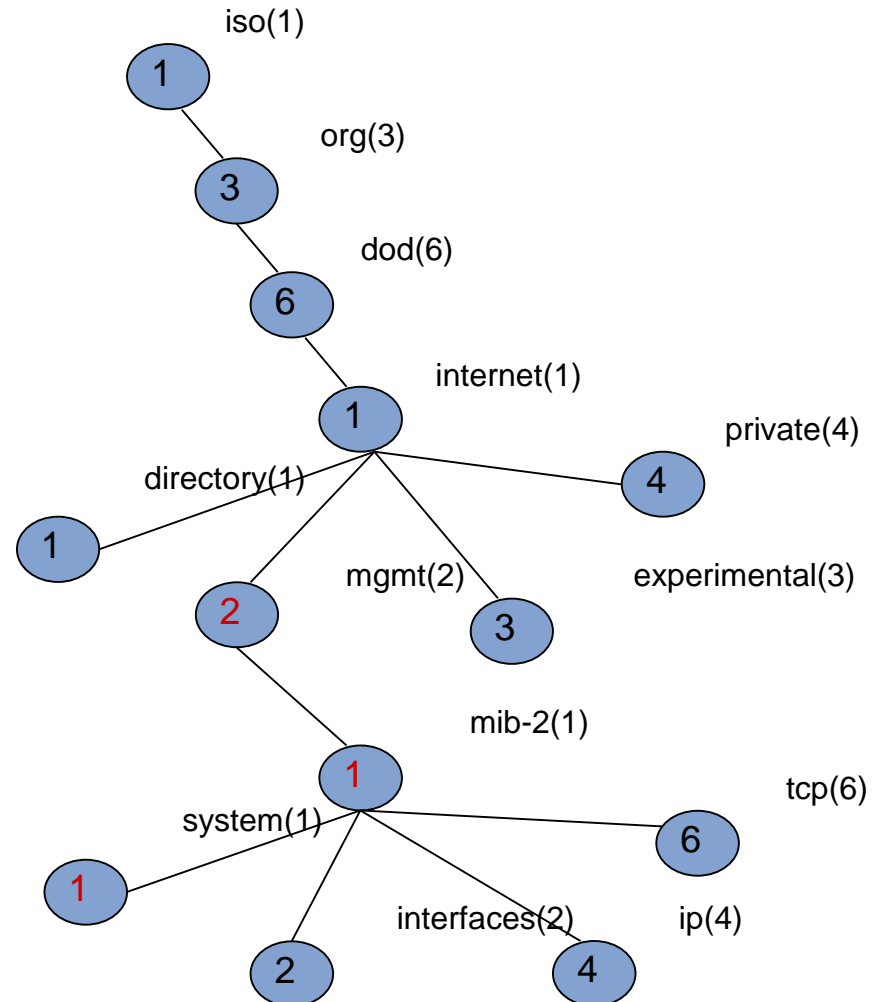


Management Information Base (MIB)

- Agent's objects have own identifiers **OID (Object Identifier)**
 - Hierarchical tree structure
 - Number + name
 - Object's address is a path from the root node

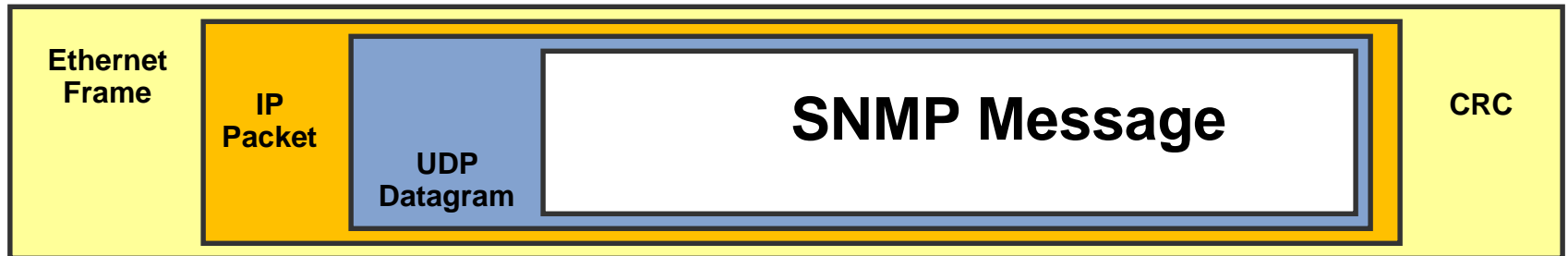
- E.g.:* OID = 1.3.6.1.2.1.1

iso(1)
org(3)
dod(6)
internet(1)
mgmt(2)
mib-2(1)
system (1)



Ports

- UDP Port 161 - SNMP Messages
- UDP Port 162 - SNMP Trap Messages



SNMP Version 1 (SNMPv1)

- [RFC 1157](#)

- Five basic SNMP messages

- **Get Request (Get)**

- Used to request the value of a specific MIB variable

- **Get Next Request (GetNext)**

- Used after the initial Get Request to retrieve the next object instance from a table or a list

- **Set Request (Set)**

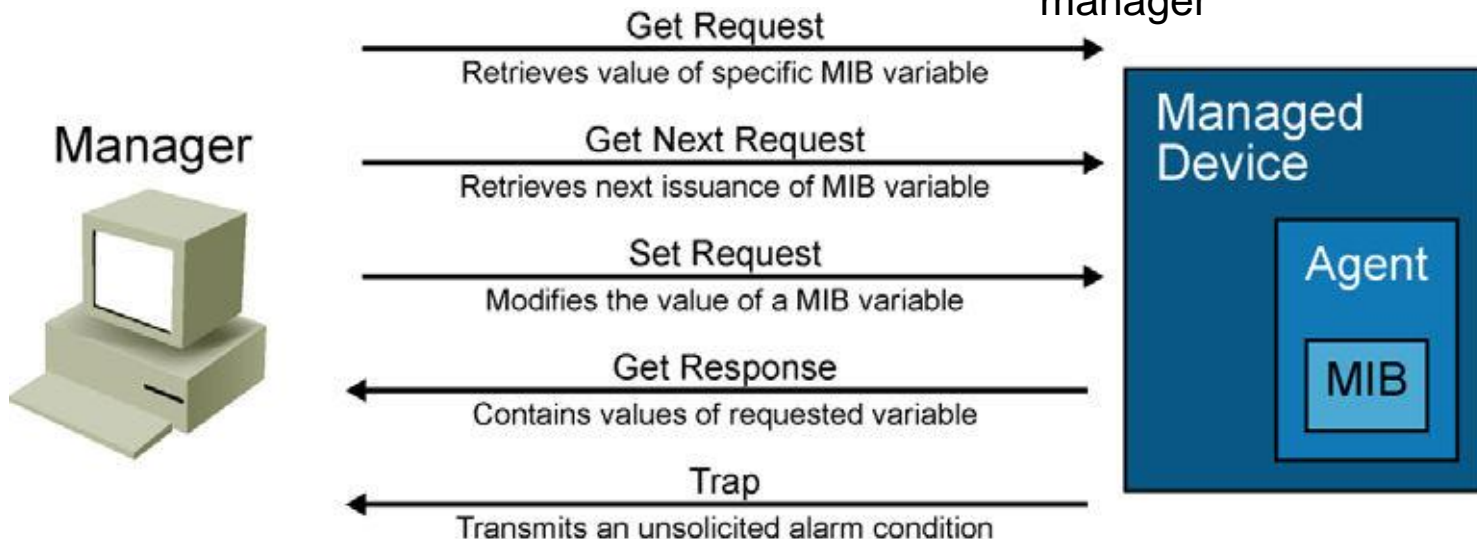
- Used to set a MIB variable on an agent

- **Get Response (Response)**

- Used by an agent to respond to a Get Request or Get Next Request from a manager

- **Trap**

- Transmit an unsolicited alarm to manager



SNMP Version 2 (SNMPv2)

- **Internet standard Management Framework** ([RFC 1441](#))
 - Lack of security – problems with standardization
 - Only experimental implementations, adds 64 bits counters
- Community-based **SNMPv2 (SNMPv2c)**
 - [RFC 1901](#)
 - The most common implementation of SNMPv2
 - SNMPv2c uses community strings for administrative access
 - **READ-ONLY**
 - **READ-WRITE**
 - **TRAP**
- SNMPv2 introduces two new message types:
 - **Get Bulk Request**
 - Reduces repetitive requests
 - Improve performance when you are retrieving large amounts of data
 - **Inform Request**
 - Alert an SNMP manager of specific conditions
 - Message is confirmed by Inform Response

SNMP Version 3

- [RFC 3410](#), [3411](#), [3412](#), [3413](#), [3414](#), [3415](#)
- It adds methods to ensure the secure transmission
- SNMPv3 introduces three levels of security
 - **noAuthNoPriv**
 - No authentication is required
 - No privacy is provided
 - **authNoPriv**
 - Authentication using MD5 or SHA
 - No privacy
 - **authPriv**
 - Authentication using MD5 or SHA
 - Privacy (encryption) using DES, 3DES or AES
- Cisco supports User-based Security Model (Authentication with name and password)

Recommendations

- SNMPv1 and SNMPv2 use community strings in clear text
 - Community strings should be changed at regular intervals
 - IF SNMP is used only to monitor devices
THEN use read-only communities
 - Use ACL to prevent SNMP messages from going beyond the required devices
- SNMPv3 is recommended because it provides authentication and encryption
- Restrict access to read-only.
- Use write access with separate credentials and careful consideration.
- Set up SNMP views to restrict manager to only access needed sets of MIBs.
- Configure ACLs to restrict SNMP access only by known managers.
- Use SNMPv3 authentication, encryption, and integrity where possible, including upgrading devices to support SNMPv3 if necessary.

SNMPv3 Best Practice Configuration

```
Switch(config)# access-list 99 permit 10.1.1.0 0.0.0.255
Switch(config)# snmp-server view OPS sysUpTime included
Switch(config)# snmp-server view OPS ifDescr included
Switch(config)# snmp-server view OPS ifAdminStatus included
Switch(config)# snmp-server view OPS ifOperStatus included
Switch(config)# snmp-server user userZ groupZ v3 auth sha secretpwd2 priv aes 256
                secondsecretpwd2
Switch(config)# snmp-server enable traps
Switch(config)# snmp-server host 10.1.1.50 traps version 3 priv userZ cpu port-
                security
Switch(config)# snmp-server ifindex persist
```

SNMP Command Reference

Command	Description
<code>snmp-server enable traps</code> [<i>notification-type</i>]	Enables SNMP notification types that are available on your system
<code>snmp-server group</code> <i>group-name</i> {v1 v2c v3 {auth noauth priv}} [context <i>context-name</i>] [read <i>read-view</i>][write <i>write-view</i>] [notify <i>notify-view</i>][access [<i>acl-number</i> <i>acl-name</i>]]	Configures a new SNMP group with specified authentication and optionally with the specified associated SNMP context, read, write, notify view, and associated ACL
<code>snmp-server host</code> { <i>ip-address</i> } [informs traps] version{1 2c 3 {auth noauth}}	Specifies the recipient of an SNMP notification operation
<code>snmp-server ifindex persist</code>	Enable interface index persistence
<code>snmp-server user</code> <i>username</i> <i>group-name</i> {v1 v2c v3 [encrypted][auth {md5 sha} <i>auth-password</i>]} [access [priv {des 3des aes {128 192 256}} <i>privpassword</i>]] { <i>acl-number</i> <i>acl-name</i> }]	Configures a new user to an SNMP group
<code>snmp-server view</code> <i>view-name</i> <i>oid-tree</i>	Creates a view entry

Configuring SNMPv2 and SNMPv3

- Configure SNMP access lists
- Configure SNMP community strings
- Configure SNMP Trap receiver
- Configure SNMP traps

```
Switch(config)# access-list 1 permit 10.1.1.0 0.0.0.255
Switch(config)# snmp-server community cisco RO 1
Switch(config)# snmp-server community xyz123 RW 1
Switch(config)# snmp-server host 10.1.1.50
Switch(config)# snmp-server enable traps ?
```

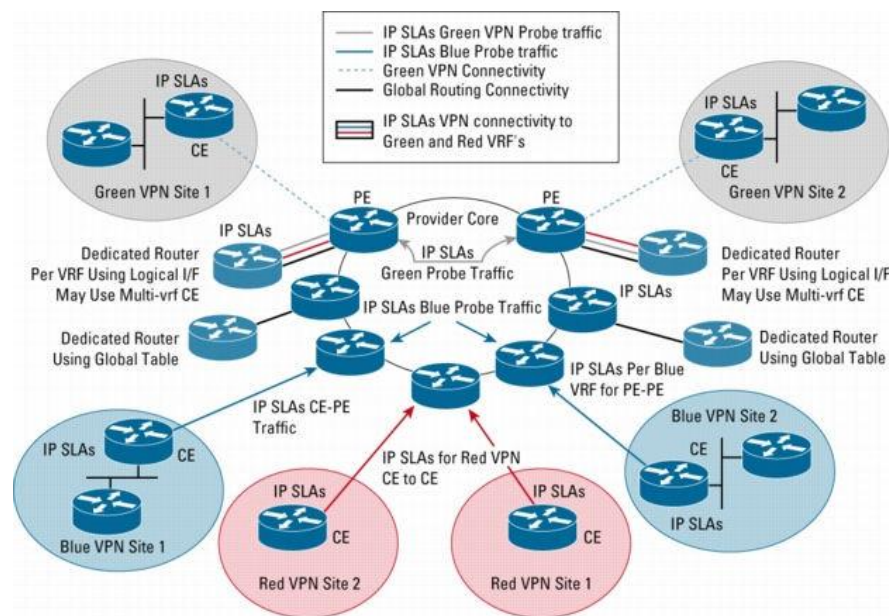
```
Switch(config)# snmp-server group MYGROUP v3 priv
Switch(config)# snmp-server user MYUSER MYGROUP v3 auth md5
MYPASS123 priv aes 128 MYKEY123
Switch(config)# snmp-server host 10.9.99.50 priv
Switch(config)# snmp-server enable traps snmp linkdown linkup
coldstart warmstart hsrp vrrp
```

IP Service Level Agreements



IP Service Level Agreements

- Cisco IOS **IP Service Level Agreements (IP SLAs)** use active traffic monitoring
- Cisco IOS IP SLAs tests send simulated data and measure performance between network locations
- Administrator can set conditions needed to satisfy the test



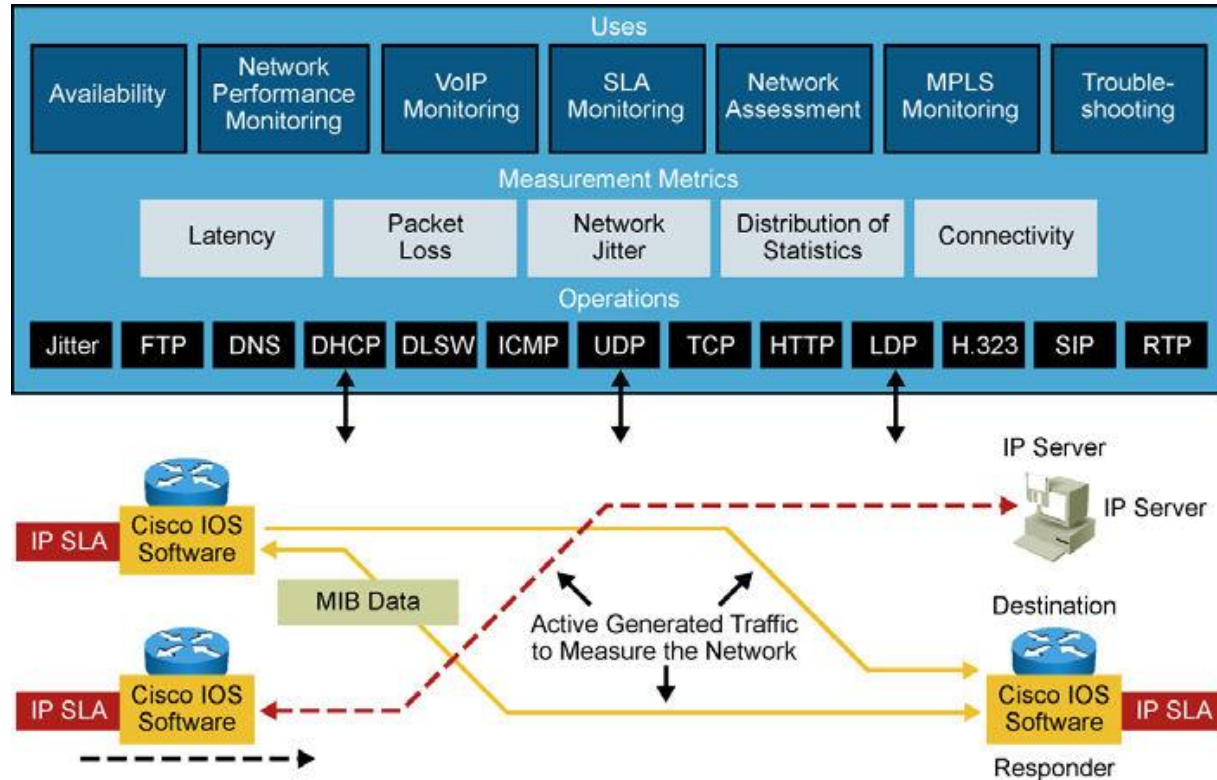
Cisco IOS IP SLAs

- Possible parameters:
 - Network resource availability
 - Response time
 - One-way latency
 - Jitter
 - Packet lost
 - Voice quality scoring
 - Application performance

IP SLA Sources, Responders and Operation

- **IP SLA source** sends testing data to destination
 - All test are configured on SLA source
 - SLA source use **control protocol** to communication with responder before the test starts (agreement on TCP/UDP ports, test type etc.)
 - Source and responder have to use time synchronization (NTP)
- **IP SLA responder** allows to anticipate and respond to IP SLAs request packets
 - To increase security on IP SLA measurements control messages, responder can utilize MD5 authentication
- **IP SLA operation** is measurement that includes protocol, frequency, traps and thresholds
 - IP SLA operations are defined by capabilities of target devices

IP SLA Operations



- IP SLA against device **without SLA responder** (web server or IP station)
 - E.g. ping test
- IP SLA against device **with SLA responder** running
 - More powerful test or more accurate results

Configuration Steps

- 1) Define SLA operation (probe)
- 2) Scheduling IP SLA operation
- 3) Define at least one tracking object and define action associated with the tracking object
- 4) IF responder is Cisco device THEN enable wide range of IP SLA capabilities

```
Switch(config)# ip sla {monitor} responder
```

■ Note:

- Starting from IOS 12.4(4)T, 12.2(33)SB and 12.2(33)SXI , command **ip sla monitor** is replaced by command **ip sla**

Step 1 – Define SLA Operation ①

1) Defining SLA operation

```
Router(config)# ip sla operation-number
```

- Parameter *operation-number* is ID of operation

```
R1(config)# ip sla 1  
R1(config-ip-sla)# ?  
IP SLAs entry configuration commands:  
  dhcp          DHCP Operation  
  dns           DNS Query Operation  
  exit          Exit Operation Configuration  
  icmp-echo     ICMP Echo Operation  
  icmp-jitter   ICMP Jitter Operation  
  ...  
R1(config-ip-sla)#
```

Step 1 – Define SLA Operation ②

- PING probe against non-responder node:

```
Router(config-ip-sla) #  
icmp-echo {destination-ip-address | destination-hostname}  
[source-ip {ip-address | hostname} | source-interface IFACE]
```

Parameter	Description
<i>destination-ip-address</i> <i>destination-hostname</i>	Destination IPv4/IPv6 address
source-ip { <i>ip-address</i> <i>hostname</i> }	(Optional) Sets source IPv4/IPv6 address
source-interface <i>interface-name</i>	(Optional) Sets the interface IP address as a source

- Note:

- Starting from IOS 12.4(4)T, 12.2(33)SB and 12.2(33)SXI command **type echo protocol ipIcmpEcho** is replaced by command **icmp-echo**

The icmp-echo Command

```
R1(config-ip-sla)# icmp-echo 209.165.201.30
R1(config-ip-sla-echo)# ?
```

IP SLAs echo Configuration Commands:

default	Set a command to its defaults
exit	Exit operation configuration
frequency	Frequency of an operation
history	History and Distribution Data
no	Negate a command or set its defaults
owner	Owner of Entry
request-data-size	Request data size
tag	User defined tag
threshold	Operation threshold in milliseconds
timeout	Timeout of an operation
tos	Type Of Service
verify-data	Verify data
vrf	Configure IP SLAs for a VPN Routing/Forwarding in-stance

```
R1(config-ip-sla-echo)#
```

■ Bare minimum:

```
Router(config-ip-sla-echo)#
```

```
frequency seconds
```

```
timeout milliseconds
```


Step 2 – Scheduling IP SLA Operation

2) IP SLA operation needs to be scheduled

```
Router(config)#  
  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]]
```

- Note:

- Starting from IOS 12.4(4)T, 12.2(33)SB and 12.2(33)SXI the command **ip sla monitor schedule** is replaced by **ip sla schedule**

Step 3 – Creating Tracking Object

3) Creating tracking object:

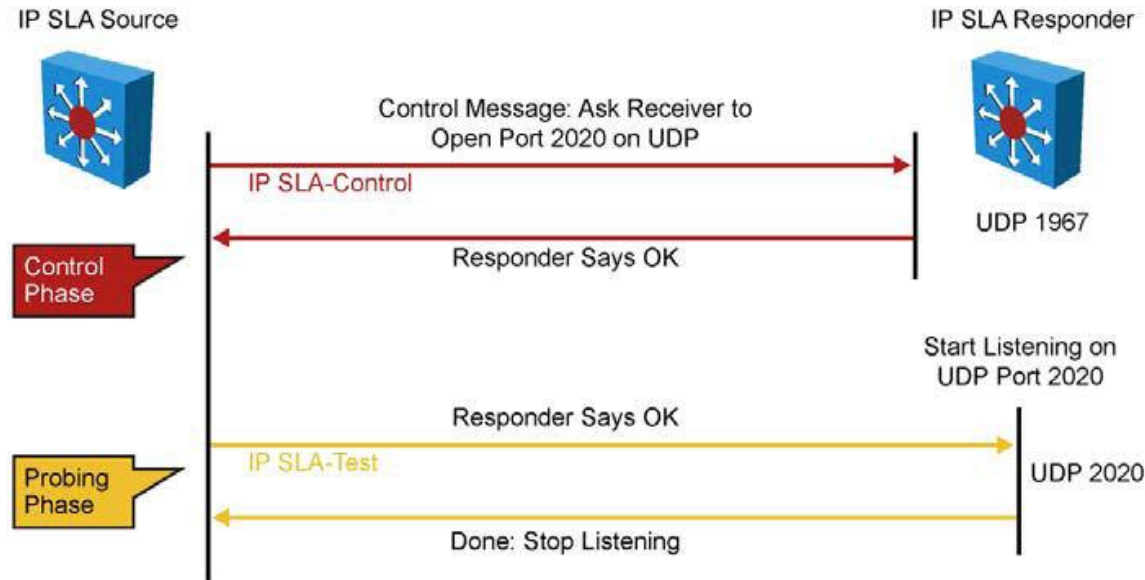
```
Router(config)#  
  track object-number ip sla operation-number {state |  
  reachability}
```

Parameter	Description
<i>object-number</i>	Tracking object number from 1 to 500
<i>operation-number</i>	Number used for the identification of the IP SLAs operation you are tracking
state	Tracks the operation return code
reachability	Tracks the reachability

■ Note:

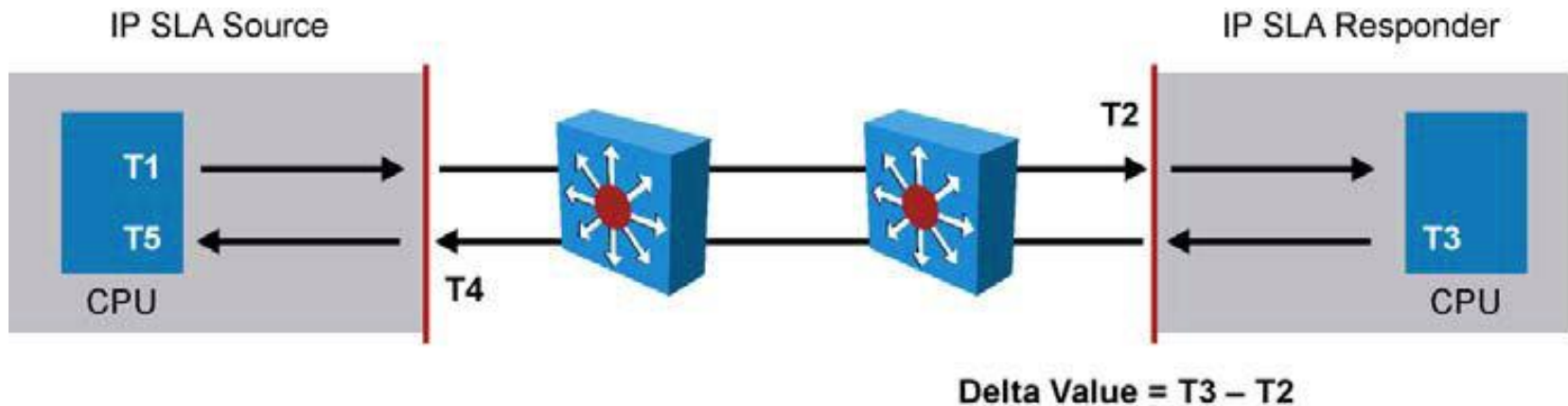
- Starting from IOS 12.4(20)T, 12.2(33)SXI1 and 12.2(33)SRE the command **track rtr** is replaced by **track ip sla**

IP SLA Operation with Responder



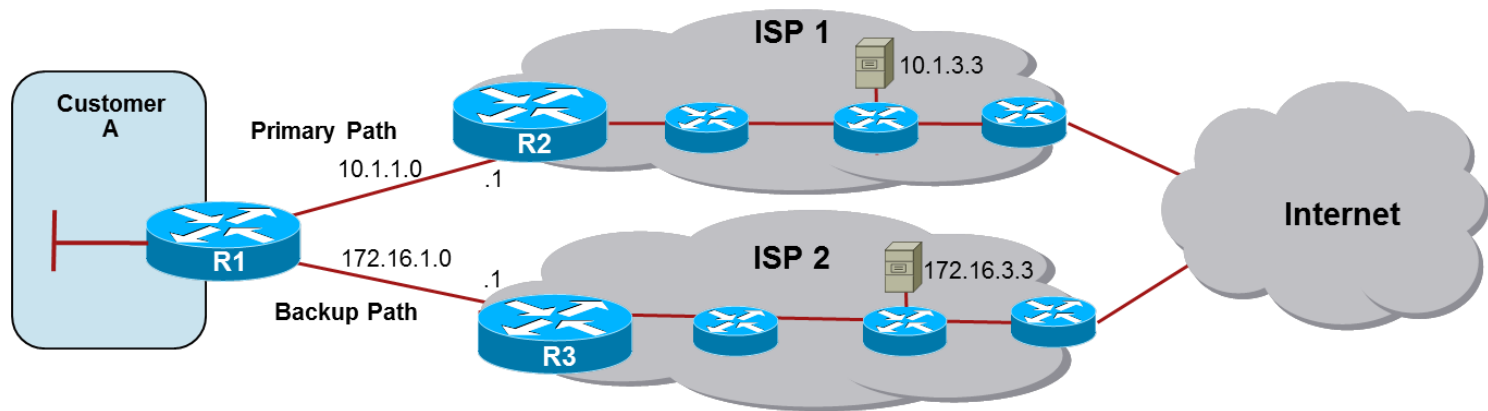
- 1) IP SLA source sends a control message with the configured IP SLA operation information (protocol, port number, and duration) to IP SLA **control port UDP 1967** on responder
- 2) IF the responder processes the control message THEN it sends an OK message to the source router and listens on the port specified in the control message for a specified duration
 - IF the responder cannot process the control message THEN it returns an error
- 3) IF the return code of control message is OK THEN the IP SLA operation moves to the probing phase, where it sends one or more test packets to the responder for response time computations.
 - The return code is available in IP SLA statistics
- 4) The responder accepts the test packets and responds
 - Based on the type of operation, the responder might add an "in" timestamp and an "out" timestamp in the response packet payload to account for CPU time spent in measuring unidirectional packet loss, latency, and jitter to a Cisco device.

IP SLA Responder Timestamps



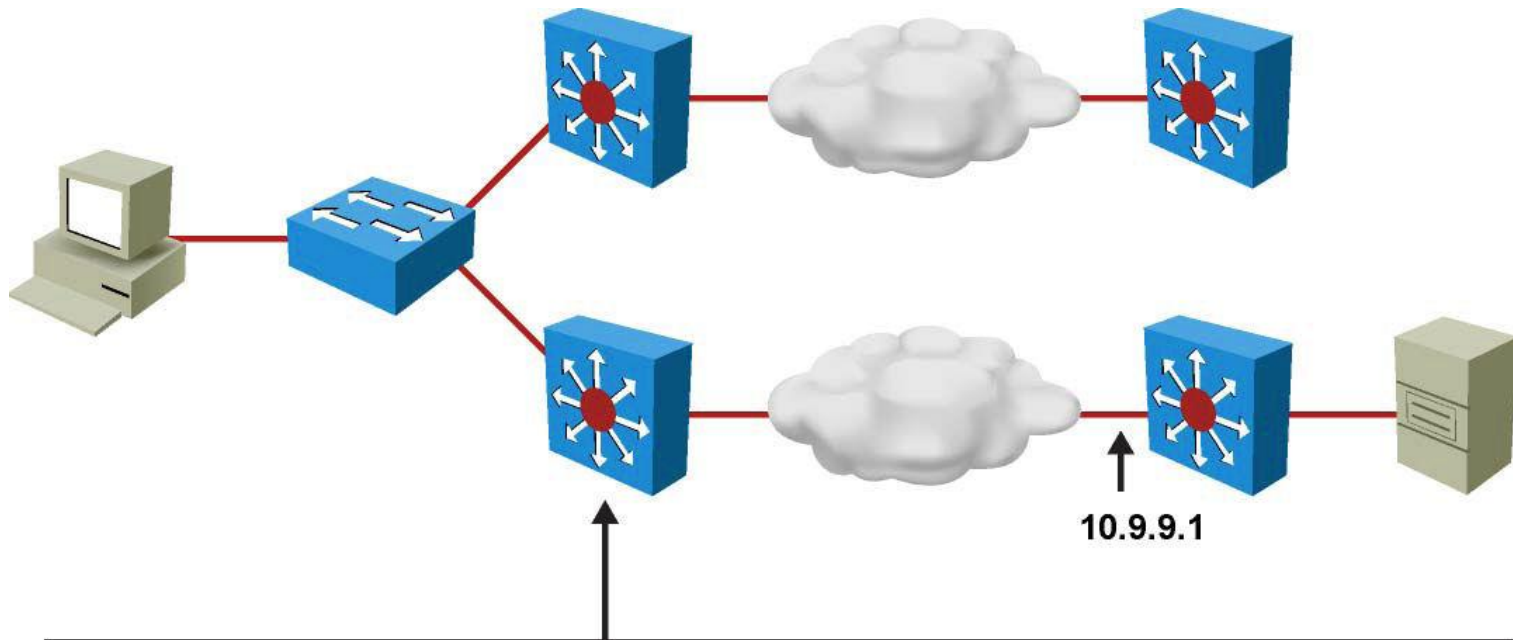
- IP SLA responder timestamps are used in round-trip calculations
- IP SLA source sends test packet at time T1
- IP SLA responder includes receipt time (T2) and transmitted time (T3)

Example: IS SLAs on Multi-homed Connection



```
R1(config)# ip sla 11
R1(config-ip-sla)# icmp-echo 10.1.3.3
R1(config-ip-sla-echo)# frequency 10
R1(config-ip-sla-echo)# exit ! 2x
R1(config)# ip sla 22
R1(config-ip-sla)# icmp-echo 172.16.3.3
R1(config-ip-sla-echo)# frequency 10
R1(config-ip-sla-echo)# exit ! 2x
R1(config)# track 1 ip sla 11 reachability
R1(config-track)# delay down 10 up 1
R1(config-track)# exit
R1(config)# track 2 ip sla 22 reachability
R1(config-track)# delay down 10 up 1
R1(config-track)# exit
R1(config)# ip sla schedule 11 life forever start-time now
R1(config)# ip sla schedule 22 life forever start-time now
R1(config)# ip route 0.0.0.0 0.0.0.0 10.1.1.1 2 track 1
R1(config)# ip route 0.0.0.0 0.0.0.0 172.16.1.1 3 track 2
```

Example: HSRP and IP SLA Tracking



```
sw(config)# ip sla 18
sw(config-sla)# icmp-echo 10.9.9.1
sw(config)# ip sla schedule 18 start-time now life forever
sw(config)# track 90 rtr 18 state
sw(config)# interface vlan10
sw(config-if)# ip address 10.1.1.2 255.255.255.0
sw(config-if)# standby 10 ip 10.1.1.1
sw(config-if)# standby 10 priority 110
sw(config-if)# standby 10 preempt
sw(config-if)# standby 10 track 90 decrement 20
```

Verify IP SLA Operation

- When IP SLA is configured, the test is conducted as per the scheduled configuration - the test might succeed or fail
- IF you do not monitor the test results THEN it might fail silently

```
Switch# show ip sla statistics
Round Trip Time (RTT) for Index 1
Latest RTT: NoConnection/Busy/Timeout
Latest operation start time: 11:11:22.533 eastern Thu Jul 9 2010
Latest operation return code: Timeout
Over thresholds occurred: FALSE
Number of successes: 177
Number of failures: 6
Operation time to live: Forever
Operational state of entry: Active
Last time this entry was reset: Never
```

Verify IP SLA Configuration

```
Switch# show ip sla configuration
IP SLAs, Infrastructure Engine-II
Entry number: 1
Owner:
Tag:
Type of operation to perform: echo
Target address/Source address: 10.1.3.10/10.1.253.1
Type Of Service parameter: 0x0
Request size (ARR data portion): 28
Operation timeout (milliseconds): 5000
Verify data: No
Vrf Name:
Schedule:
    Operation frequency (seconds): 5
    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
<output omitted>
```


AAA



AAA

- ■ **Authentication**

- Authentication is the process of identifying a user before that user is allowed access to a protected resource.

- ■ **Authorization**

- After the user gains access to the network, authorization is performed.
- Authorization allows you to control the level of access users have.

- ■ **Accounting**

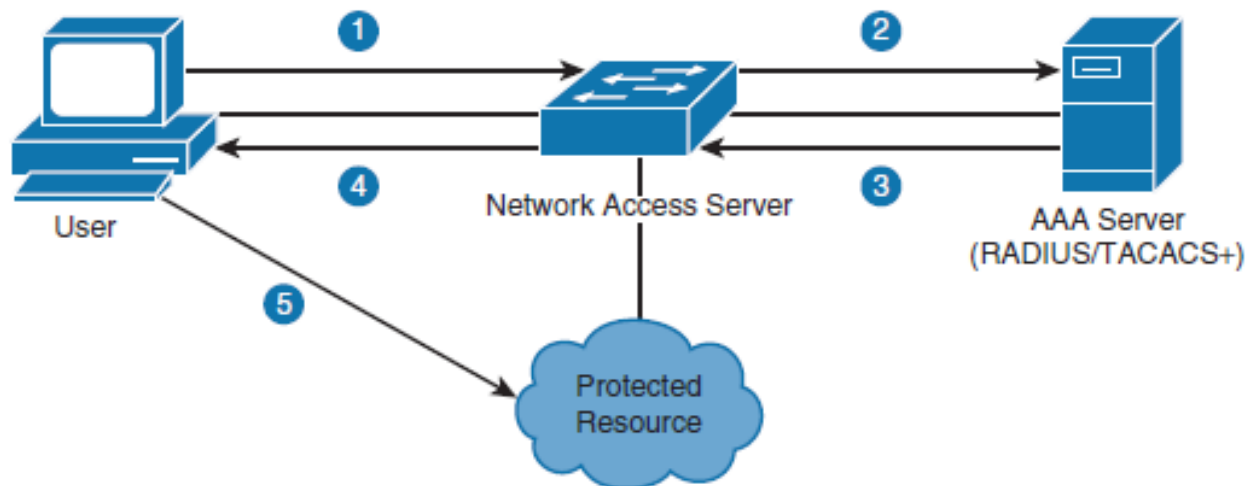
- Accounting is performed after authentication. Accounting enables you to collect information about the user activity and resource consumption.

AAA Benefits

- **Increased flexibility and control of access configuration**
 - AAA offers additional authorization flexibility on a per-command or per-interface level, which is unavailable with local credentials.
- **Scalability**
 - As the network grows, managing a large number of users on multiple devices becomes highly impractical and error-prone, with a lot of administrative burden.
- **Standardized authentication methods**
 - AAA supports the RADIUS protocol, which is an industry open standard. This ensures interoperability and allows flexibility because you can mix and match different vendors.
- **Multiple backup systems**
 - You may specify multiple servers when configuring authentication options on the method list, combining them in a server group.

RADIUS and TACACS+ Overview

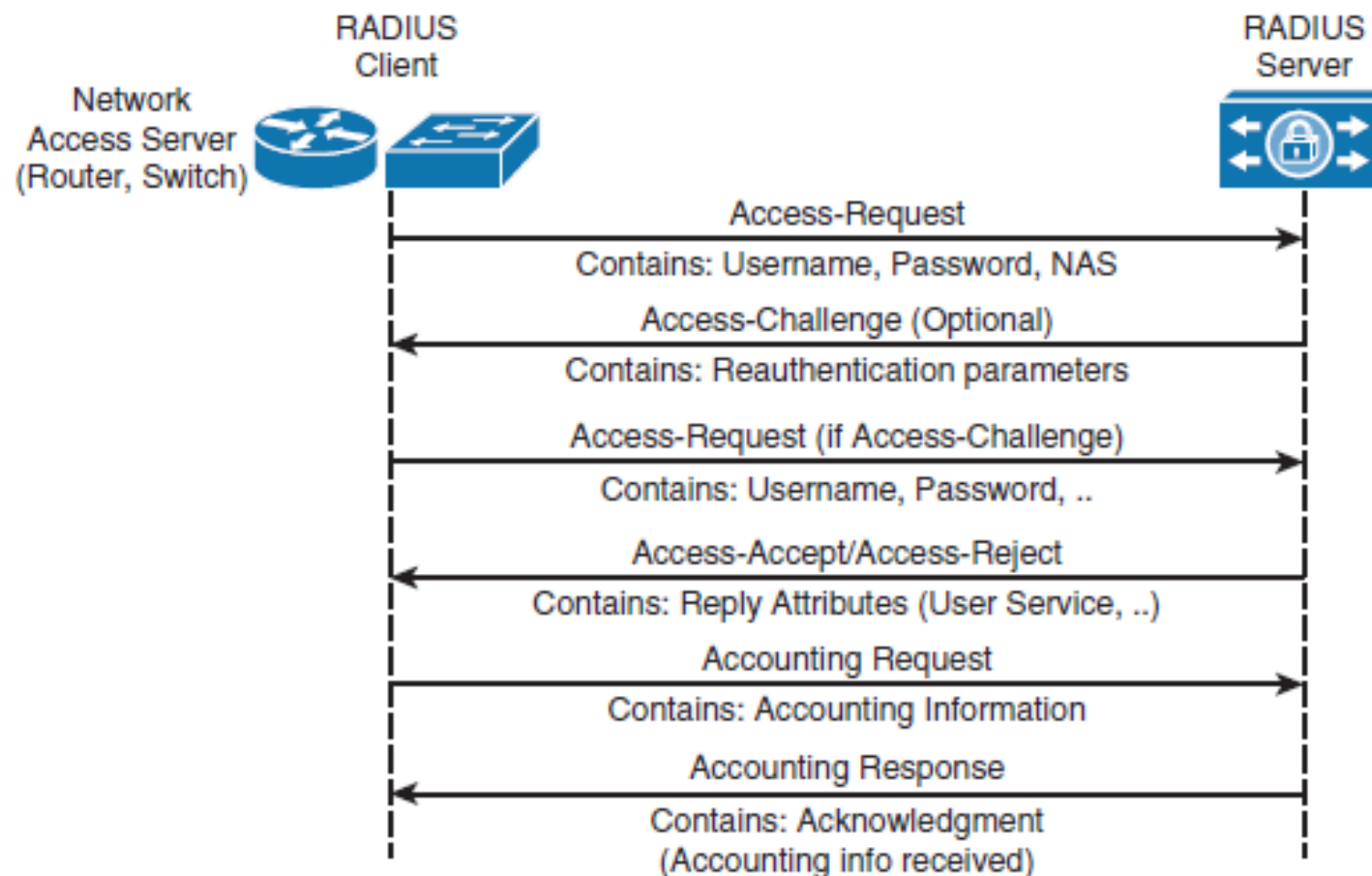
- RADIUS and TACACS+ are AAA protocols.
- Both use the client/server model.
- As shown in Step 1, a user or machine sends a request to a networking device such as a router that acts as a network access server when running AAA.
- The network access server then communicates (2, 3) with the server exchanging RADIUS or TACACS+ messages.
- If authentication is successful, the user is granted (4) an access to a protected resource (5), such as a device CLI, network, and so on.



TACACS+ Versus RADIUS

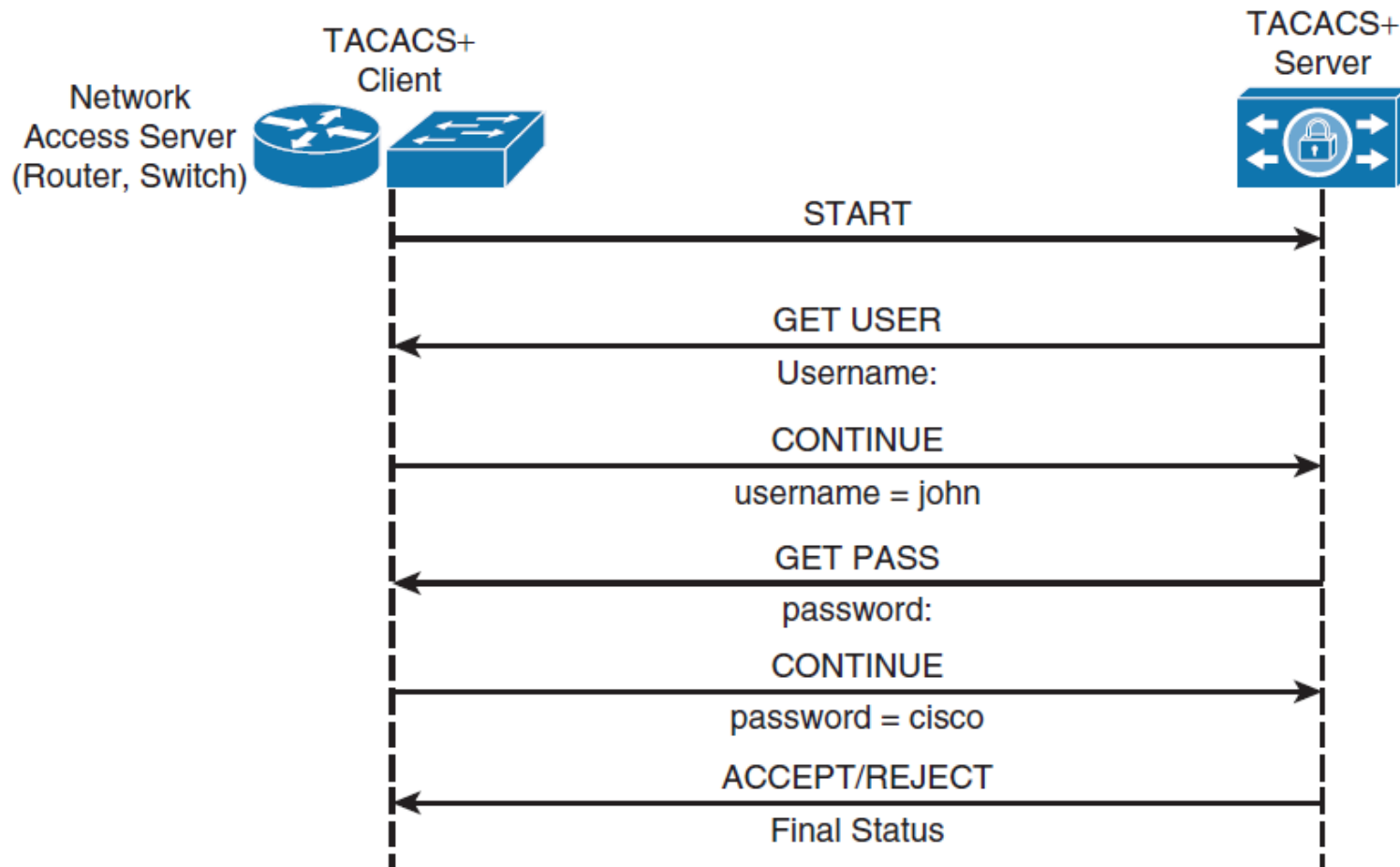
Feature	RADIUS	TACACS+
Developer	Livingston Enterprise (now industry standard)	Cisco (proprietary)
Transport protocol	UDP ports 1812 and 1813	TCP port 49
AAA support	Combines authentication and authorization and separates accounting	Uses the AAA model and separates all three services
Challenge response	One-way, unidirectional (single challenge response)	Two-way, bidirectional (multiple challenge responses)
Security	Encrypts only the password in the packet	Encrypts the entire packet body

RADIUS Authentication Process



RADIUS AAA Communication

TACACS+ Authentication Process



TACACS+ Authentication Communication

Configuring AAA

- To enable AAA, the first step is to configure the `aaa new-model` command in global configuration mode.
- This step essentially enables AAA capability.
- In addition, until this command is enabled, all other AAA commands are hidden.
- The **`aaa new-model`** command immediately applies local authentication to all lines and interfaces (except console line con 0).
- To avoid being locked out of the router, it is a best practice to define a

```
Switch(config)# username User123 secret Secretpwd
```

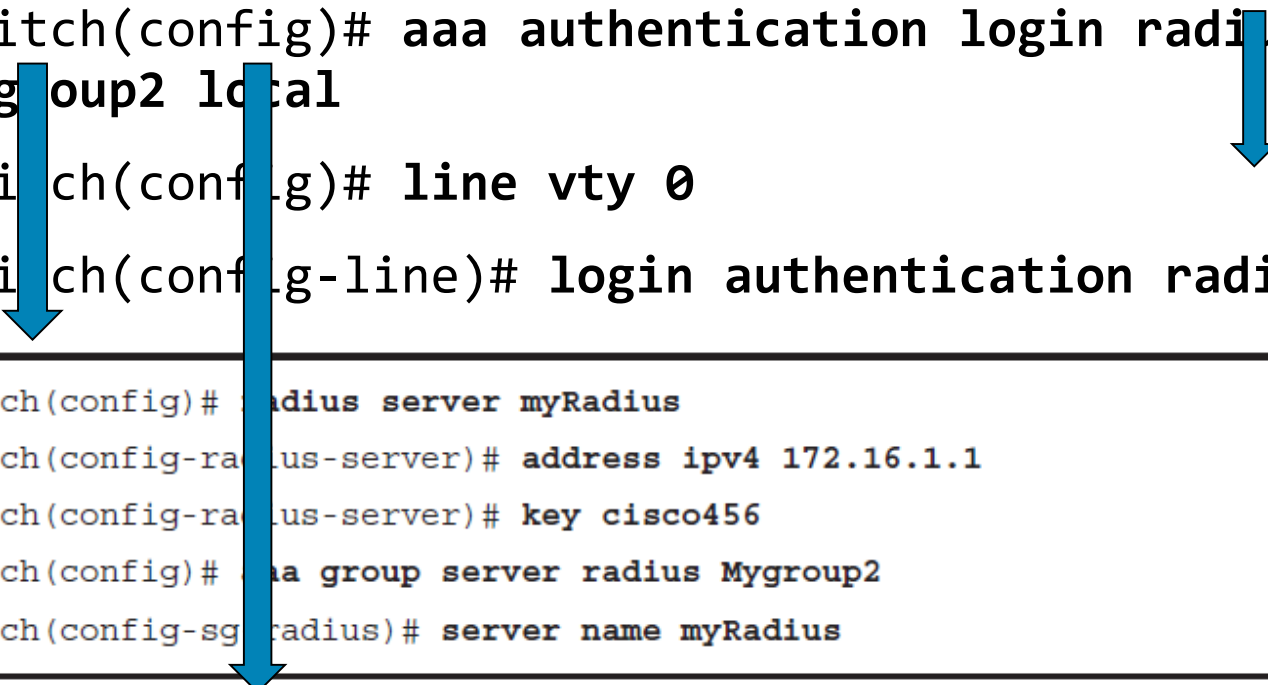

Configuring RADIUS Access

- Switch(config)# **radius server** *configuration-name*
- Switch(config-radius-server)# **address ipv4** *hostname* [**auth-port integer**] [**acct-port integer**]
- Switch(config-radius-server)# **key** *string*
- Switch(config)# **aaa group server radius** *group-name*
- Switch(config-sg-radius)# **server name** *configuration-name*

```
Switch(config)# radius server myRadius
Switch(config-radius-server)# address ipv4 172.16.1.1
Switch(config-radius-server)# key cisco456
Switch(config)# aaa group server radius Mygroup2
Switch(config-sg-radius)# server name myRadius
```

Apply RADIUS Method List to vty

- Switch(config)# **aaa authentication login radius_list group Mygroup2 local**
- Switch(config)# **line vty 0**
- Switch(config-line)# **login authentication radius_list**



```
Switch(config)# radius server myRadius
Switch(config-radius-server)# address ipv4 172.16.1.1
Switch(config-radius-server)# key cisco456
Switch(config)# aaa group server radius Mygroup2
Switch(config-sg radius)# server name myRadius
```

```
Switch(config)# username User123 secret Secretpwd
```

Configuring TACACS+ for Console and vty Access

- Switch(config)# **tacacs server** *configuration-name*
- Switch(config-server-tacacs)# **address ipv4** *hostname*
- Switch(config-server-tacacs)# **port** *integer*
- Switch(config-server-tacacs)# **key** *string*
- Switch(config)# **aaa group server tacacs+** *group-name*

```
Switch(config)# tacacs server myTacacs
Switch(config-server-tacacs)# address ipv4 192.168.1.1
Switch(config-server-tacacs)# key cisco123
Switch(config)# aaa group server tacacs+ Mygroup1
Switch(config-sg-tacacs+)# server name myTacacs
```

```
Switch(config)# aaa authentication login default group Mygroup1 local
Switch(config)# aaa authorization exec default group Mygroup1 local
```

AAA Authorization

To configure authorization, complete the following steps:

- **Step 1.** Define a named list of authorization methods.
 - **Step 2.** Apply that list to one or more interfaces (except for the default method list).
 - **Step 3.** The first listed method is used. If it fails to respond, the second one is used, and so on until all listed methods are exhausted. Once the method list is exhausted, a failure message is logged.
-
- Switch(config)# **aaa authorization** *authorization-type list-name method-list*
 - Switch(config)# **line** *line-type line-number*
 - Switch(config)# **authorization** { **arap** | **commands** *level* | **exec** | **reverse-access** } *list-name*

AAA Accounting

AAA accounting has the same rules and configuration steps as authentication and authorization:

- **Step 1.** You must first define a named list of accounting methods.
 - **Step 2.** Apply that list to one or more interfaces (except for the default method list).
 - **Step 3.** The first listed method is used; if it fails to respond, the second one is used, and so on.
-
- Switch(config)# **aaa accounting** *accounting-type list-name { start-stop | stop-only | none } method-list*
 - Switch(config)# **interface** *interface-type interface-number*
 - Switch(config-if)# **ppp accounting** *list-name*

Limitations of TACACS+ and RADIUS

RADIUS may not be the optimal choice in the following situations:

- **Device-to-device situations**

- RADIUS does not offer two-way authentication.

- **Networks using multiple service**

- RADIUS generally binds a user to a single service model.

TACACS+ may not be the optimal choice in the following situations:

- **Multivendor environment**

- TACACS+ is a Cisco proprietary protocol

- **When speed of response from the AAA services is of concern**

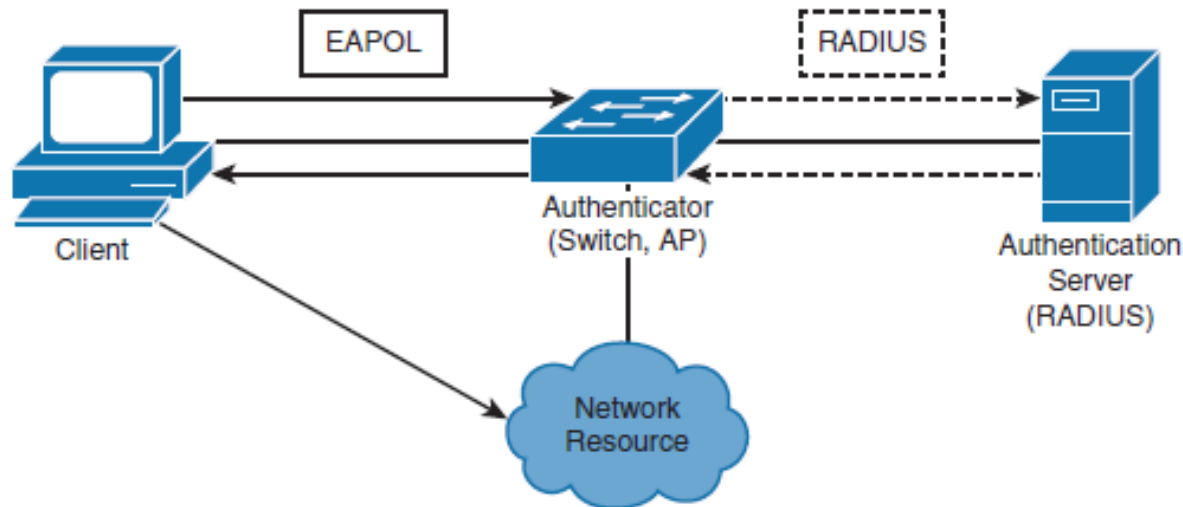
- TACACS+ uses TCP as a transport protocol mechanism.

Identity-Based Networking

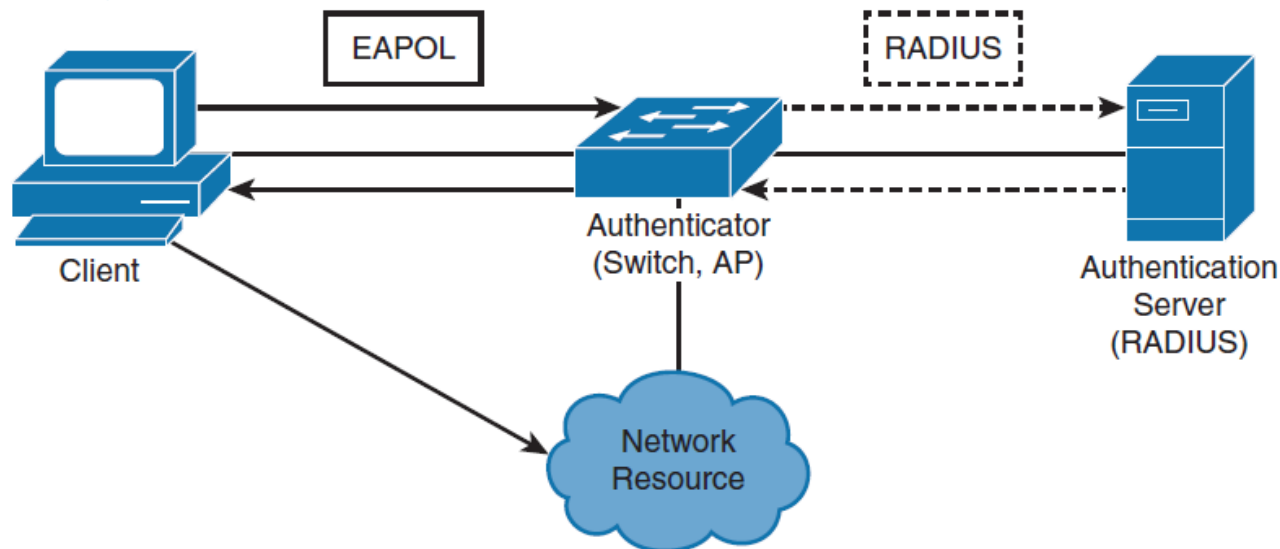


Identity-Based Networking

- Identity-based networking is a concept that unites several features to include authentication, access control, mobility, and user policy components with the aim to provide and restrict users with the network services that they are entitled to.
- From a switch perspective, identity-based networking allows you to verify users once they connect to a switch port.



IEEE 802.1X Port-Based Authentication Overview



- Until the client is authenticated, 802.1X access control allows only EAPOL, Cisco Discovery Protocol (CDP), and Spanning Tree Protocol (STP) traffic to pass through the port to which the client is connected. After authentication is successful, normal traffic can pass through the respective port.

802.1X Client/Server Model

■ Client

- Usually a workstation or laptop with 802.1X-compliant client software.
- Most modern operating systems include native 802.1X support.
- The client is also referred to as a *supplicant* in 802.1X terminology.

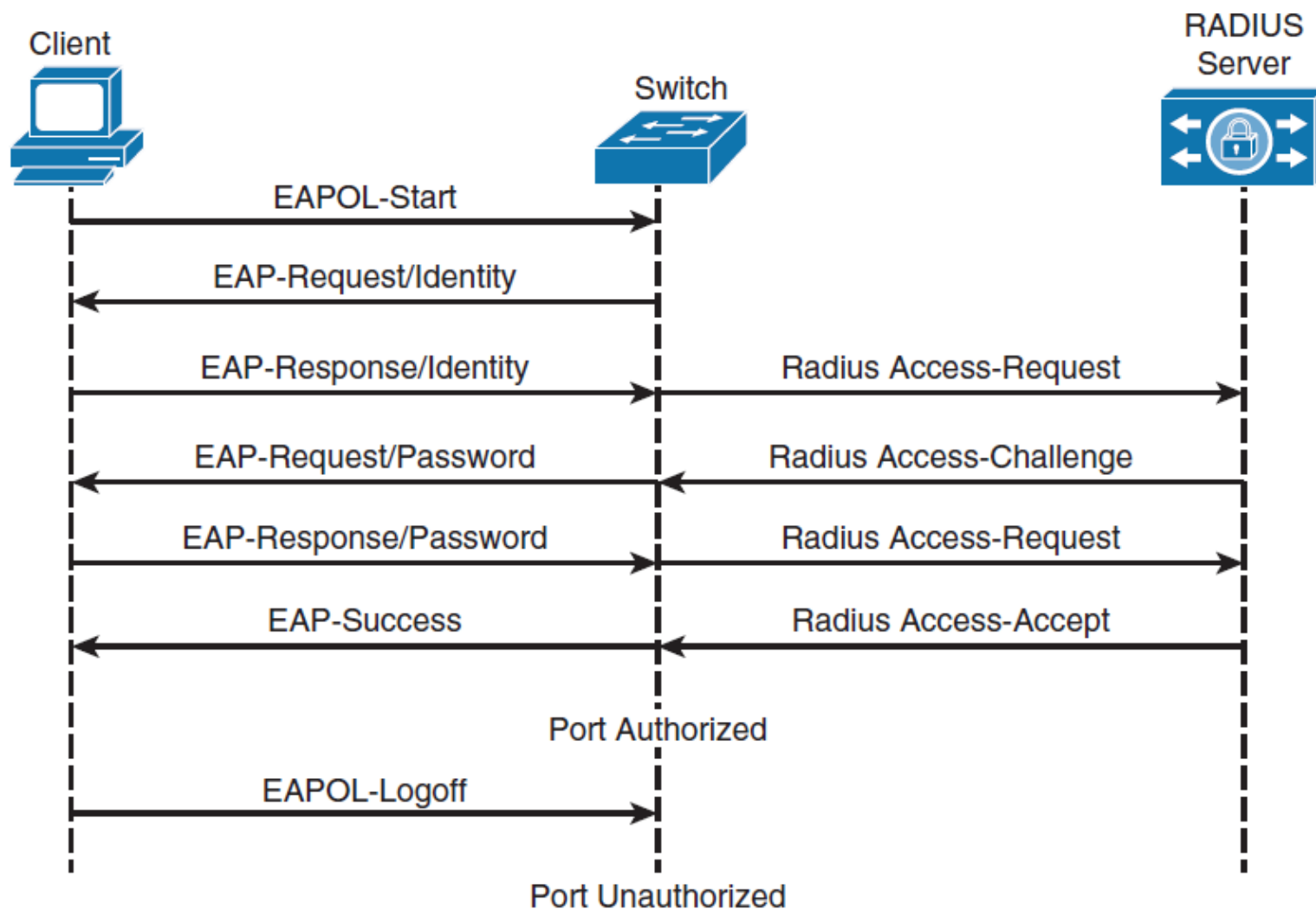
■ Authenticator

- Usually an edge switch or wireless access point (AP), the authenticator controls the physical access to the network based on the authentication status of the client.
- Authenticator includes a RADIUS client, which is responsible for encapsulation and decapsulation of Extensible Authentication Protocol (EAP) frames and interaction with the authentication server.

■ Authentication server

- A server that performs the actual authentication of the client.
- Currently, a RADIUS server with EAP extensions is the only supported authentication server.

802.1X Port-Based Authentication Overview



802.1X Configuration Example

```
Switch(config)# aaa new-model
Switch(config)# radius server host 172.16.1.1 key cisco456
Switch(config)# aaa group server radius Mygroup3
Switch(config-sg-radius)# server 172.16.1.1
Switch(config)# aaa authentication dot1x default group Mygroup3
Switch(config)# dot1x system-auth-control
Switch(config)# interface GigabitEthernet0/2
Switch(config-if)# dot1x port-control auto
```

- You will not be able to issue **dot1x** commands on the interface if it is not set to switchport mode access prior.
- The default state of switch ports varies between switches, but it is not commonly set to the access mode.



Slides adapted by Matěj Grégr and tuned by [Vladimír Veselý](#)
partially from official course materials
but the most of the credit goes to CCIE#23527 Ing. Peter Palúch, Ph.D.

The last update: 2016-11-02