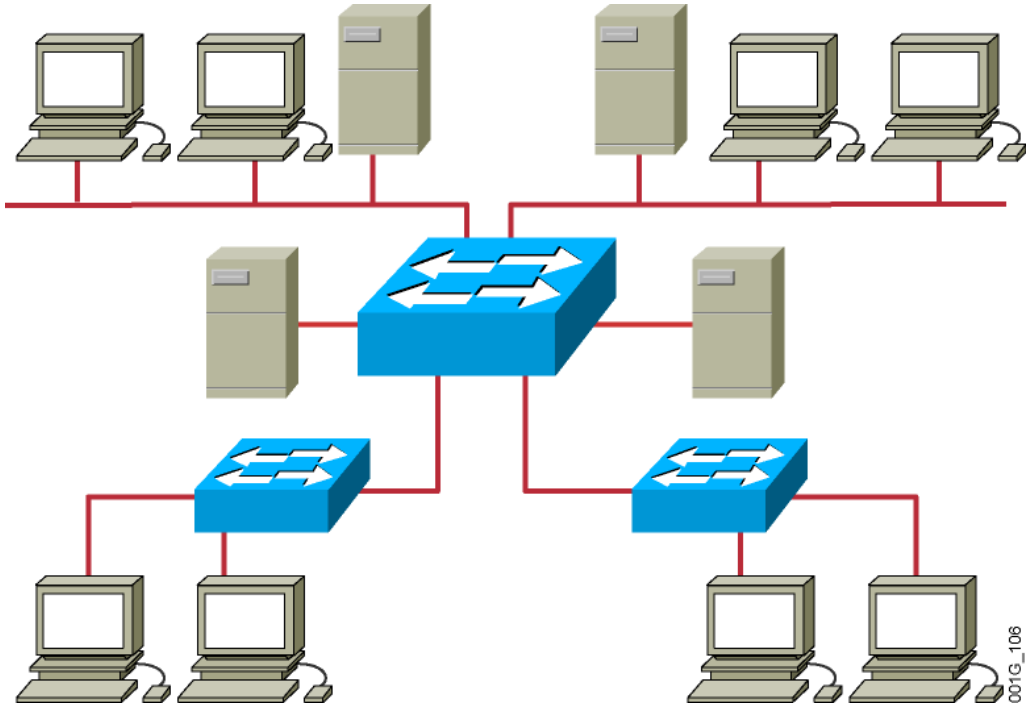
Extending Switched Networks with Virtual LANs

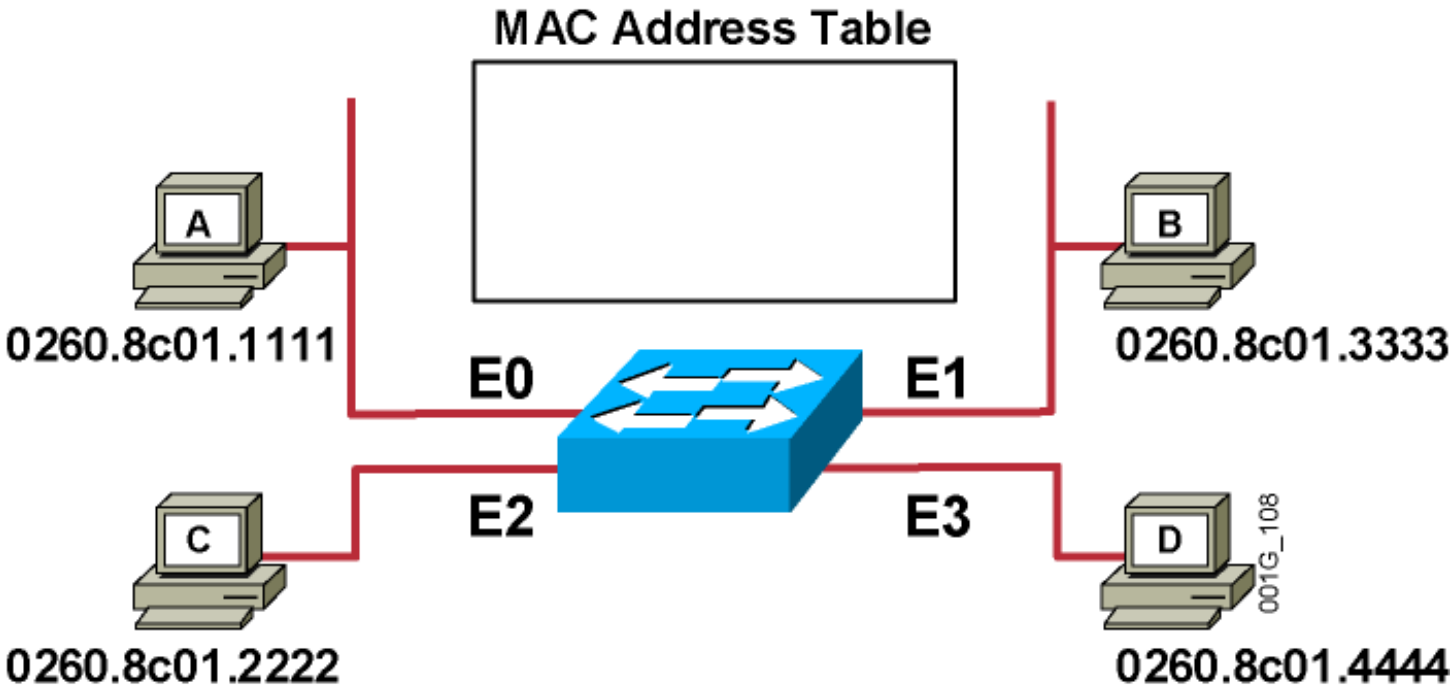
Introducing VLAN Operations

Functions of bridges and switches



- Address learning
- Forwarding the filtering decisions
- Loop avoidance

1- Address learning:



- Learns which MAC is connected on which port by checking the source MAC address in the frame.

- The initial MAC address table is empty.

2- Forwarding:

- Switch the frames to the port or ports where the destination is located by checking the destination MAC address in the frame and the MAC table.
- Frame types that are always flooded:

IF the destination MAC is:

1- Broadcast .

2- Multicast.

3- Unknown-Unicast.

- Forwarding Modes:

1- Store and Forward.

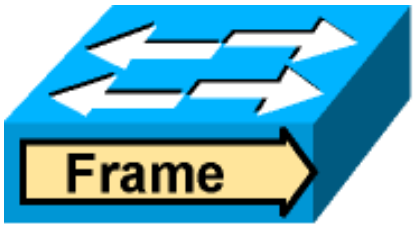
2- Cut through.

3- Fragment Free.

Transmitting Frames

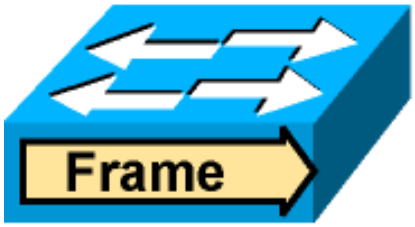
Cut-Through

- Switch checks destination address and immediately begins forwarding frame



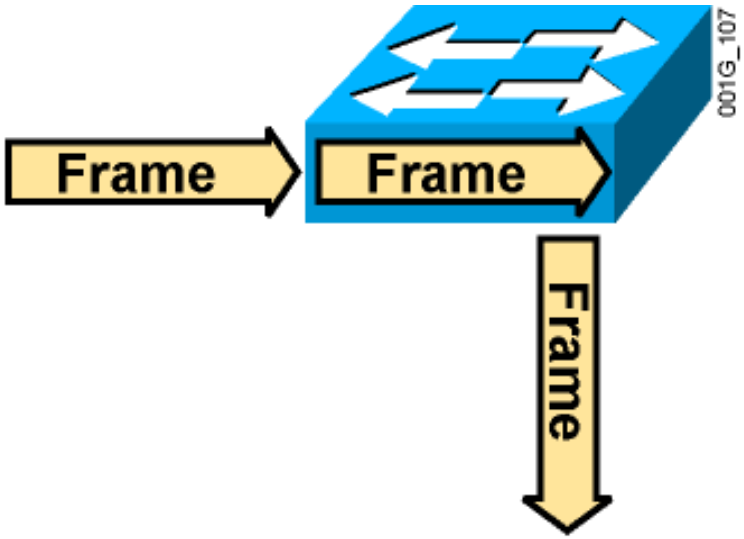
Fragment-Free

- Switch checks the first 64 bytes, then immediately begins forwarding frame

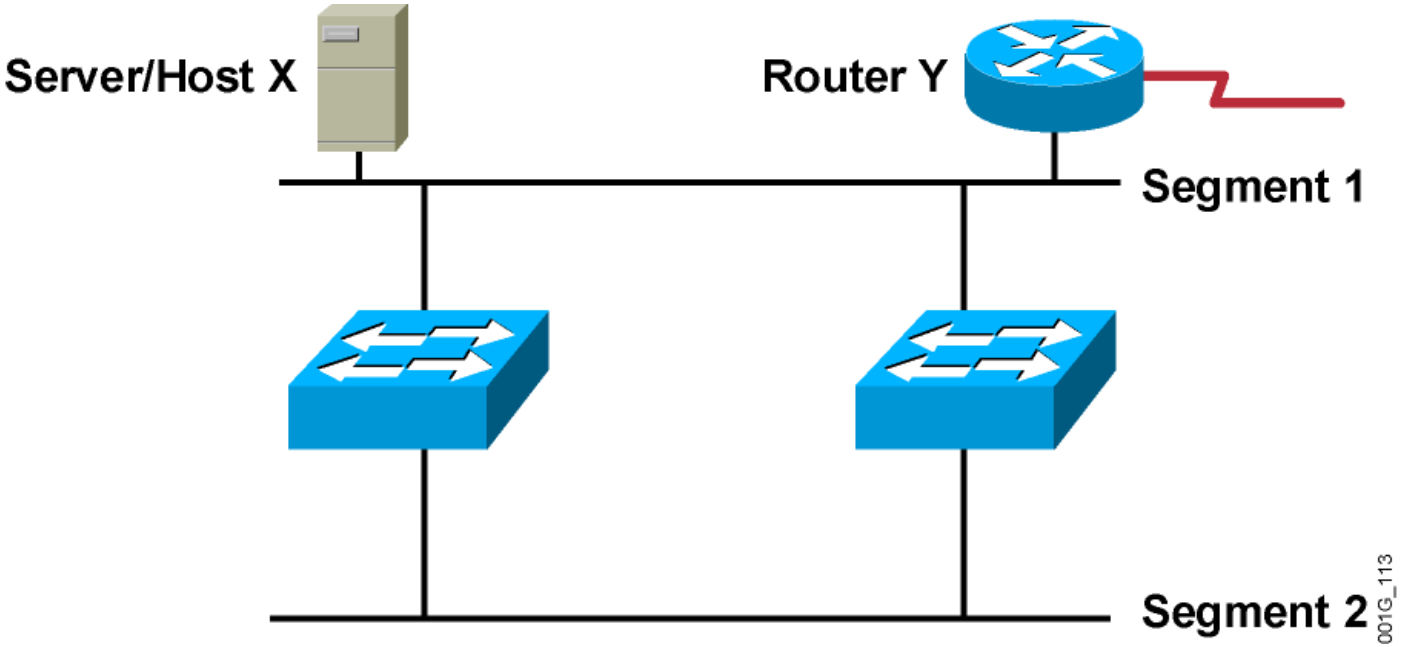


Store and Forward

- Complete frame is received and checked before forwarding

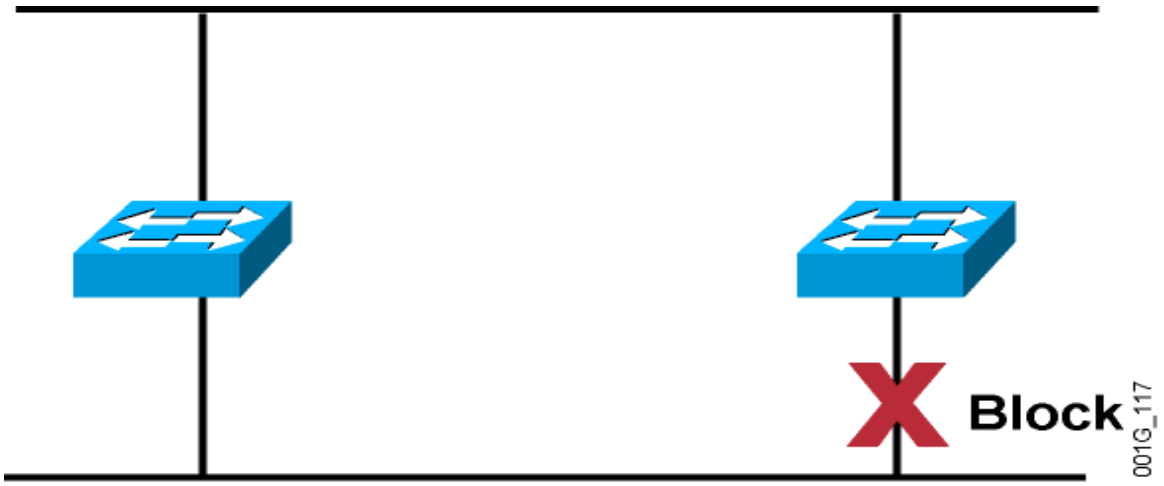


3- Removing L2 Loops:



- Redundant topology eliminates single points of failure.
- Redundant topology causes broadcast storms, multiple frame copies, and MAC address table instability problems.

Spanning Tree Protocol



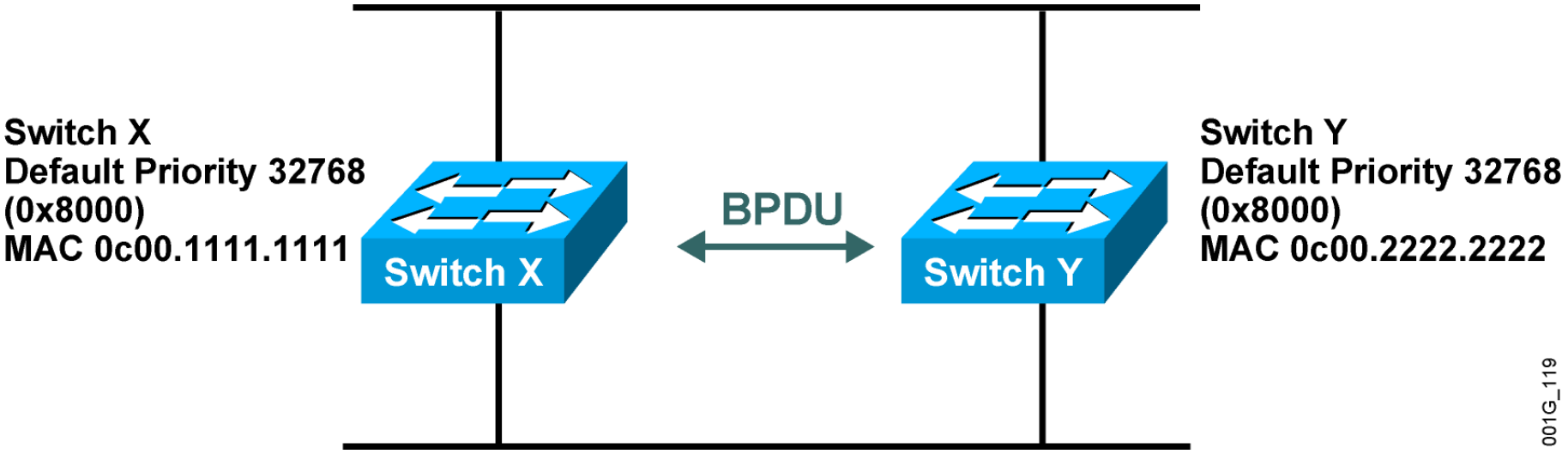
- **Provides a loop-free redundant network topology by placing certain ports in the blocking state**
- **STP protocol enables switches to become aware of each other so they can negotiate a loop free path.**
- **STP operates as switches communicate with one another, Data messages are exchanged in the form of BPDU (Bridge Protocol Data Unit).**

Spanning Tree Operation

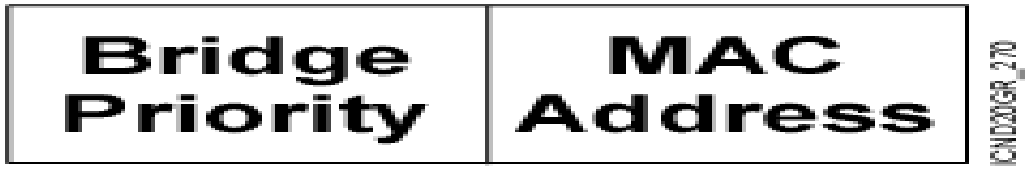
1- BPDU Flooding:

- BPDUs are flooded from each switch to the other switches on a well known multicast MAC address.
- Every switch will take a copy of the BPDU and resend it to other switches.
- Every switch will form a database from all the BPDUs.
- BPDU is sent every two seconds.

2- Root Bridge election



- Root bridge = bridge with the lowest bridge ID
- Bridge ID =



Default = 32768

- After election, the root bridge only sends the BPDUs every 2 sec.

In this example, which switch has the lowest bridge ID?

3- Root port election: (RP)

- Each non-root switch will elect the best port to reach the rout switch.
- Best port is the port that having:

1- Least accumulative path cost to the root switch.

2- If equal costs, least sender BID.

Link Speed	Cost (Revised IEEE Spec)	Cost (Previous IEEE Spec)
10 Gbps	2	1
1 Gbps	4	1
100 Mbps	19	10
10 Mbps	100	100

001G_121

4- Designated port election: (DP)

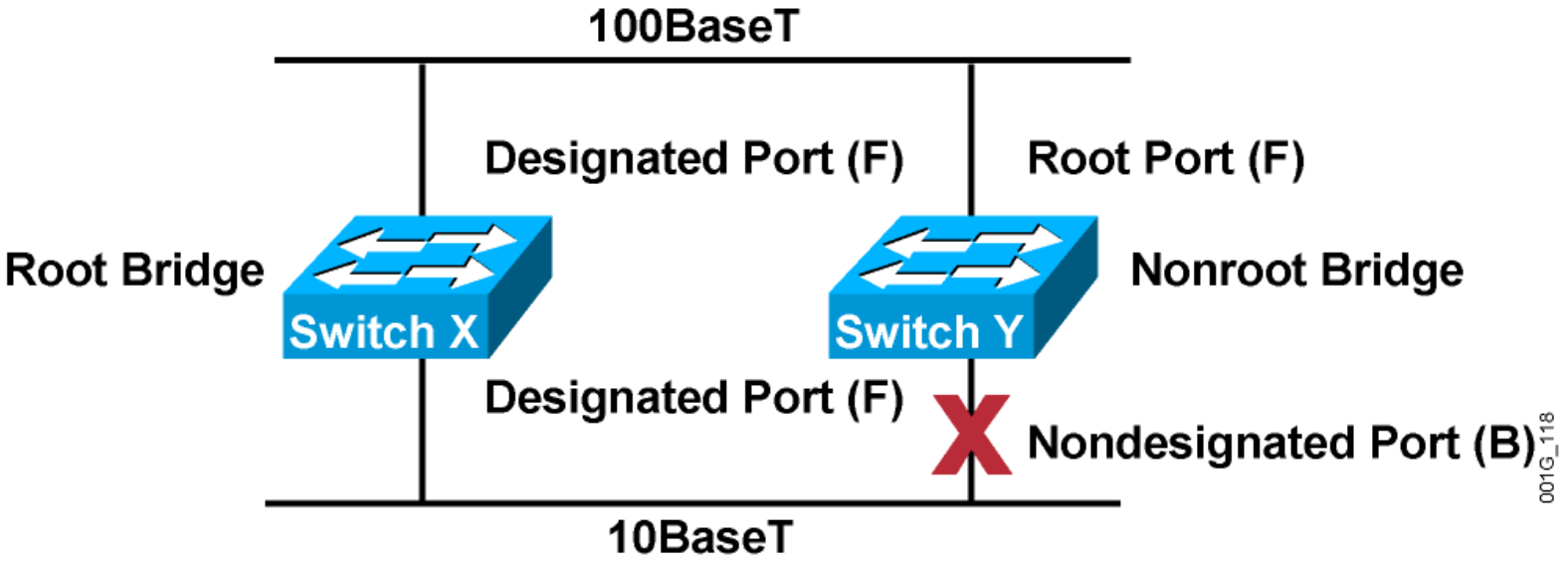
- DP is the best port in every LAN segment.

5- Blocked Port: (BP)

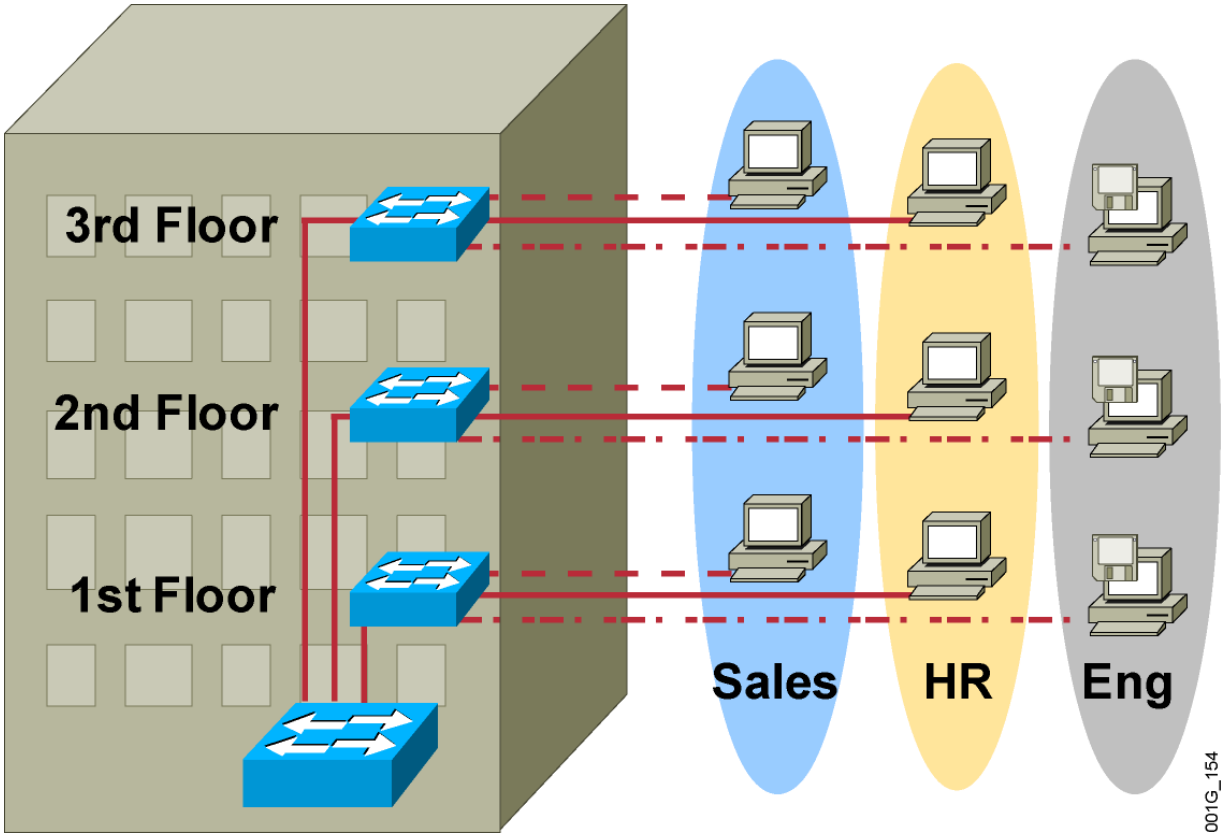
- It is the port that neither RP nor DP.
- BP will logically blocked till any change happen.

Spanning Tree Operation summary.

- One root bridge per network
- One root port per nonroot bridge
- One designated port per segment
- Nondesignated ports are unused



VLAN Overview



VLANs provides:

1- Segmentation

2- Flexibility

3- Security

Before VLANs:

- All switch ports:

1- Single broadcast domain.

2- Multiple collision domain.

After VLANs:

- Each VLAN is a single broadcast domain and one logical subnet.

- VLAN membership:

1- Static VLAN membership:

- Assign certain port to a certain VLAN.

(port based VLAN)

- By default, all ports of the switch are assigned to VLAN 1.

2- Dynamic VLAN membership:

- Assign certain MAC to a certain VLAN.

(MAC based VLAN)

- Even if the PC changes its port on the switch , the PC still be connected to its VLAN.
- This is done by using VMPS (VLAN membership policy server).

- VLAN connection (Port) types:

1- Access port:

- It is a port which is member in only one Vlan.

Ex: a switch port that connected to a pc.

2- Trunk port:

- Switch port that is member in all Vlans by default.

Ex: a switch port that connected to another switch.

- Problem:

-If host B sends a broadcast to Vlan 2, the frames will pass to the switch F over the trunk link.

-The switch F will broadcast the frames to all ports cause it doesn't know the Vlan of the traffic.

The solution:

-Trunk add field that identify the source Vlan ID to the frame.

- **VLAN trunking Methods:**

1- ISL (Inter switch Link) for Ethernet.

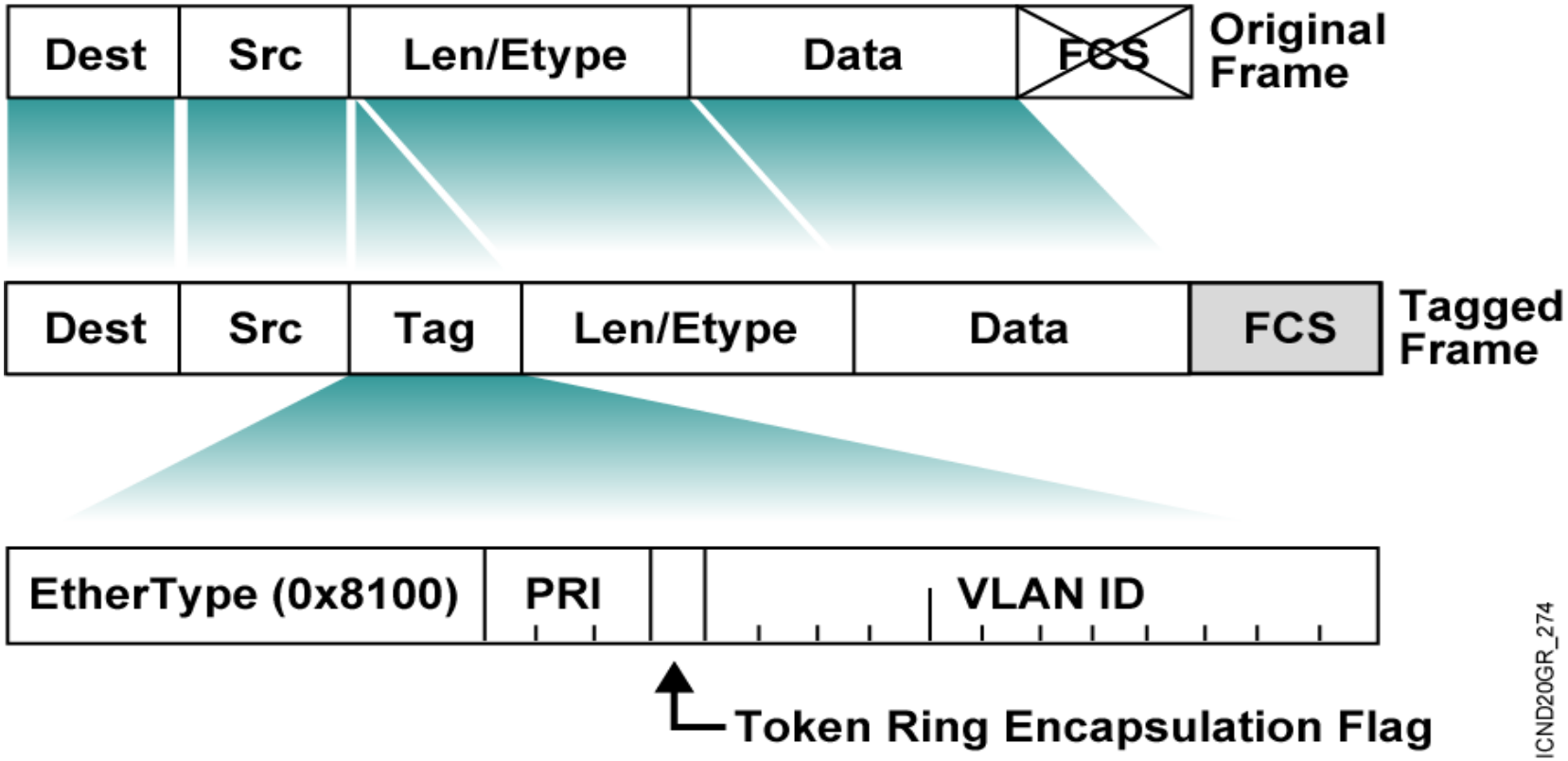
2- IEEE 802.1q for Ethernet.

3- LANE for ATM.

4- IEEE 802.10 for FDDI.

IEEE 802.1q (dot1q)

- Add 4 bytes tagging to the ethernet frame and recalculate new crc.



- Vlan ID is 12 bits in the Tag field. So, the Vlan range is 0 ----- 4095.
- Dot1q makes less overhead than ISL.
- Dot1q can support both tagged and untagged frames, where the untagged Vlan is called Native Vlan.
- By default, Native Vlan is a VLAN 1.
- Native Vlan is a management Vlan where all management data between switches are sent through this Vlan. (BPDU, STP, VTP).

- Inter VLAN routing:

- We have to use a router to route between different VLANs.

Method 1:

- Inter VLAN routing using access ports.
- Disadv.: for each Vlan you need 1 router interface and 1 switch.

Method 2:

- Router on stick.

Router(config)# int e0/0.1

Router(config-if)#encapsulation {isl / dot1q} <vlan id>

Router(config-if)#ip address <ip> <mask>

- **VLAN configuration:**

1- Create VLAN.

2- Naming VLAN (optional).

3- Assign ports to VLAN.

To create and name VLAN:

- **New method**

(config)# vlan <vlan id>

(config-vlan)# name <name>

- **Old method**

#vlan database

(vlan)#vlan <valn id> [name <name>]

To assign port to vlan:

(config)# int <int. name>

(config-if)# switchport mode access

(config-if)# switchport access vlan <vlan id>

Verifying a Trunk

```
wg_sw_2950#show interfaces interface [switchport | trunk]
```

```
wg_sw_2950#show interfaces fa0/11 switchport
Name: Fa0/11
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
  . . .
```

```
wg_sw_2950#show interfaces fa0/11 trunk

Port        Mode         Encapsulation  Status        Native vlan
Fa0/11      desirable    802.1q         trunking      1

Port      Vlans allowed on trunk
Fa0/11      1-4094

Port        Vlans allowed and active in management domain
Fa0/11      1-13
```

Verifying a VLAN

Catalyst 2950 Series

```
wg_sw_2950#show vlan [brief | id vlan-id || name vlan-name]
```

```
wg_sw_2950#sh vlan id 2

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
2    switchlab99                      active    Fa0/2, Fa0/12

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
2    enet  100002     1500  -      -      -        -    -        0      0

.   .   .
wg_sw_2950#
```

Verifying VLAN Membership

```
wg_sw_2950#show vlan brief
```

```
wg_sw_2950#show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
--
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
2    vlan2                            active
3    vlan3                            active
4    vlan4                            active
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
--
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

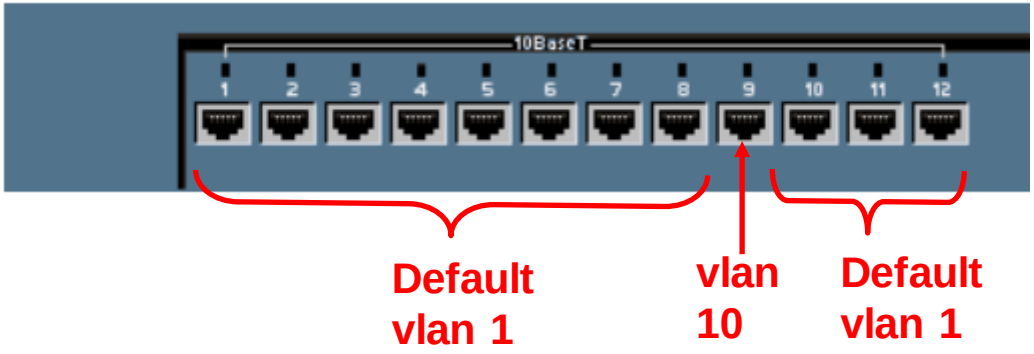
```
wg_sw_2950#show interfaces interface switchport
```

Verifying STP for a VLAN

```
wg_sw_2950#show spanning-tree [active | detail | vlan
vlan-id | summary]
```

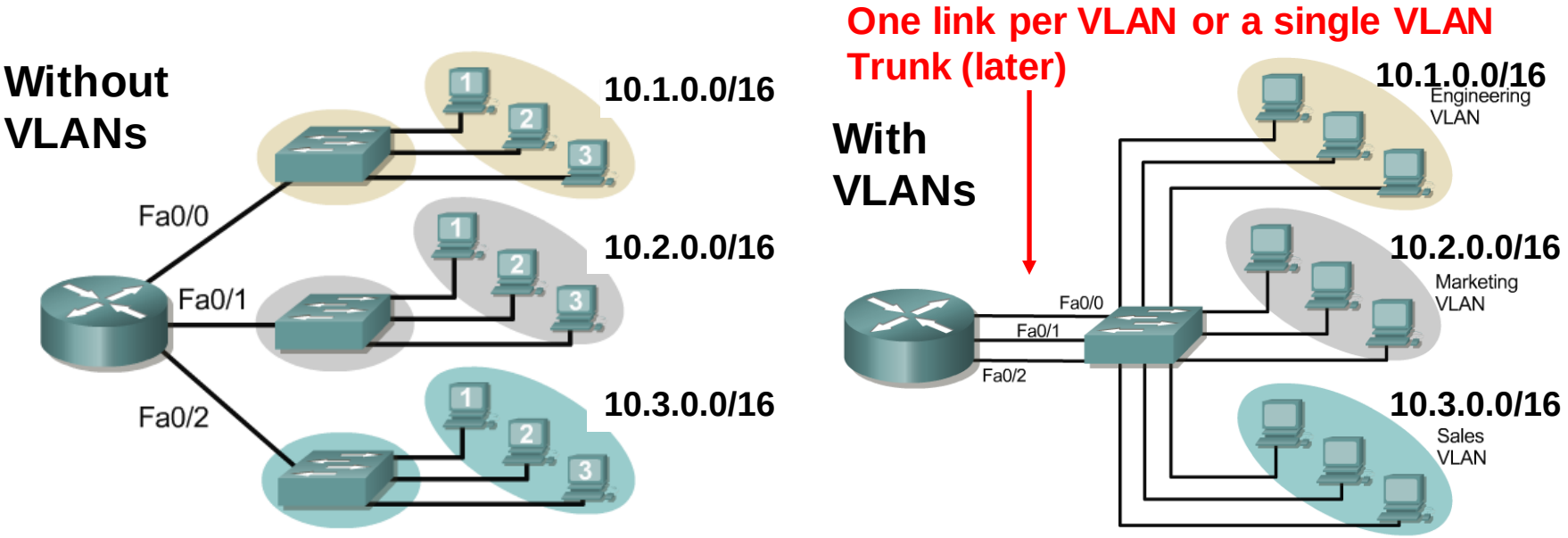
```
wg_sw_2950#sh spanning-tree vlan 2

VLAN0002
  Spanning tree enabled protocol ieee
  Root ID     Priority    2
              Address     0008.20fc.a840
              Cost        31
              Port        12 (FastEthernet0/12)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32770  (priority 32768 sys-id-ext 2)
              Address     0008.a445.9b40
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time 300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
---
Fa0/2            Desg FWD 100       128.2    Shr
Fa0/12           Root FWD 19        128.12   P2p
```

VLAN introduction



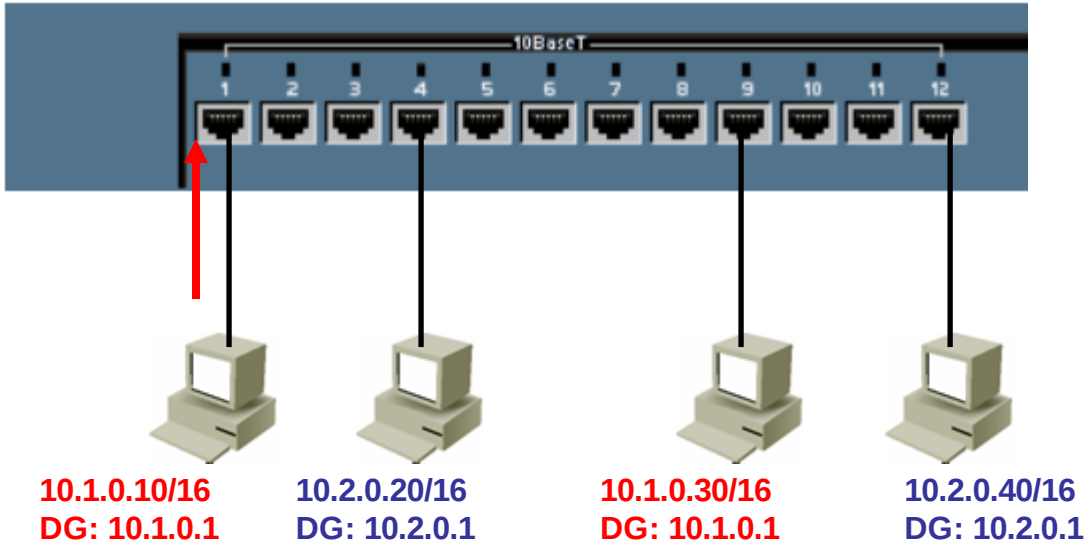
- **VLANs provide segmentation based on broadcast domains.**
- VLAN = Subnet
- VLANs can logically segment switched networks based on:
  - Physical location (Example: Building)
  - Organization (Example: Marketing)
  - Function (Example: Staff)

VLAN introduction



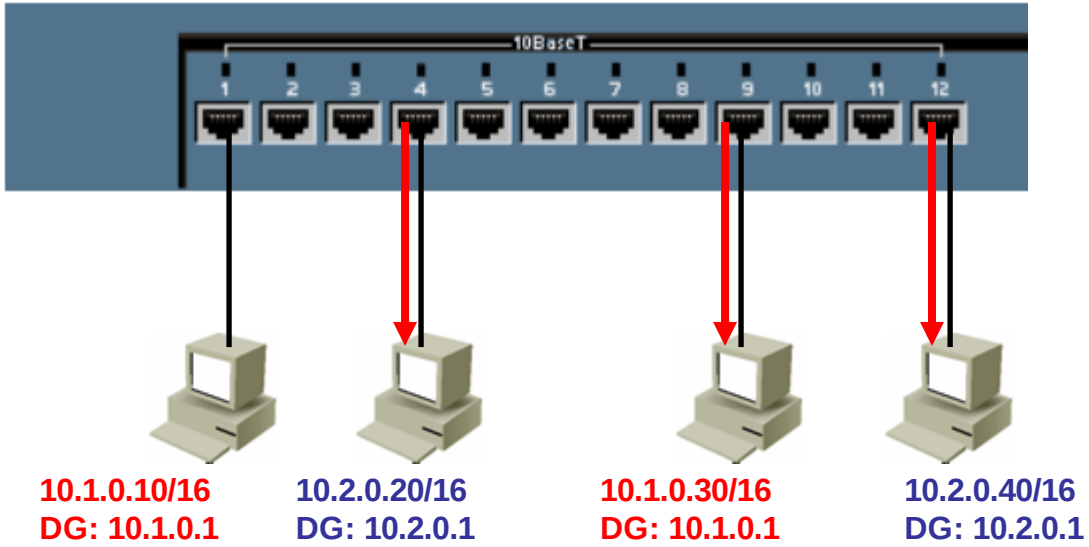
- VLANs are created to provide segmentation services traditionally provided by physical routers in LAN configurations.
- VLANs address scalability, security, and network management.

Two Subnets, One Switch, No VLANs



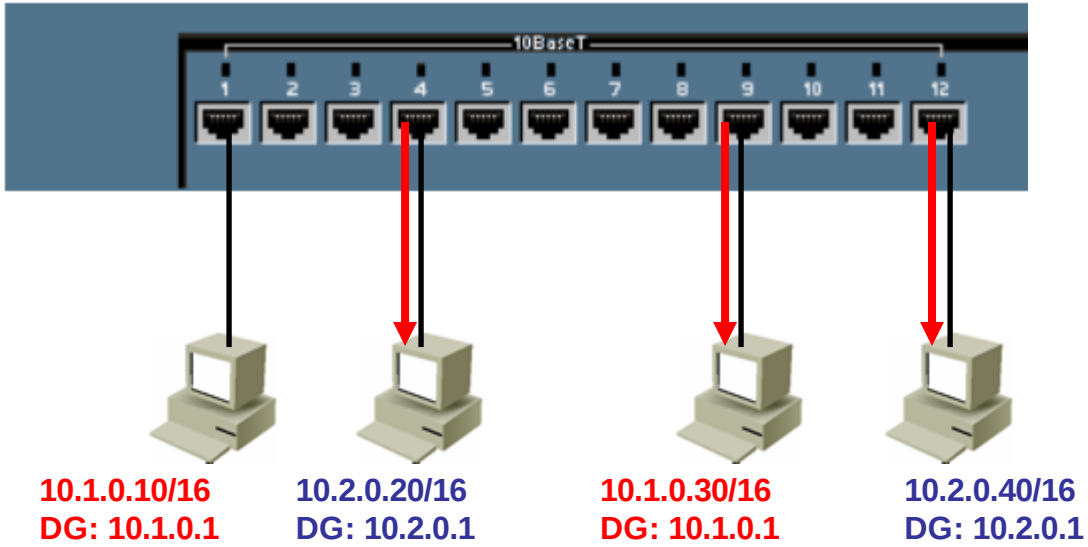
- Layer 2 Broadcasts
  - What happens when 10.1.0.10 sends an ARP Request for 10.1.0.30?

Two Subnets, One Switch, No VLANs



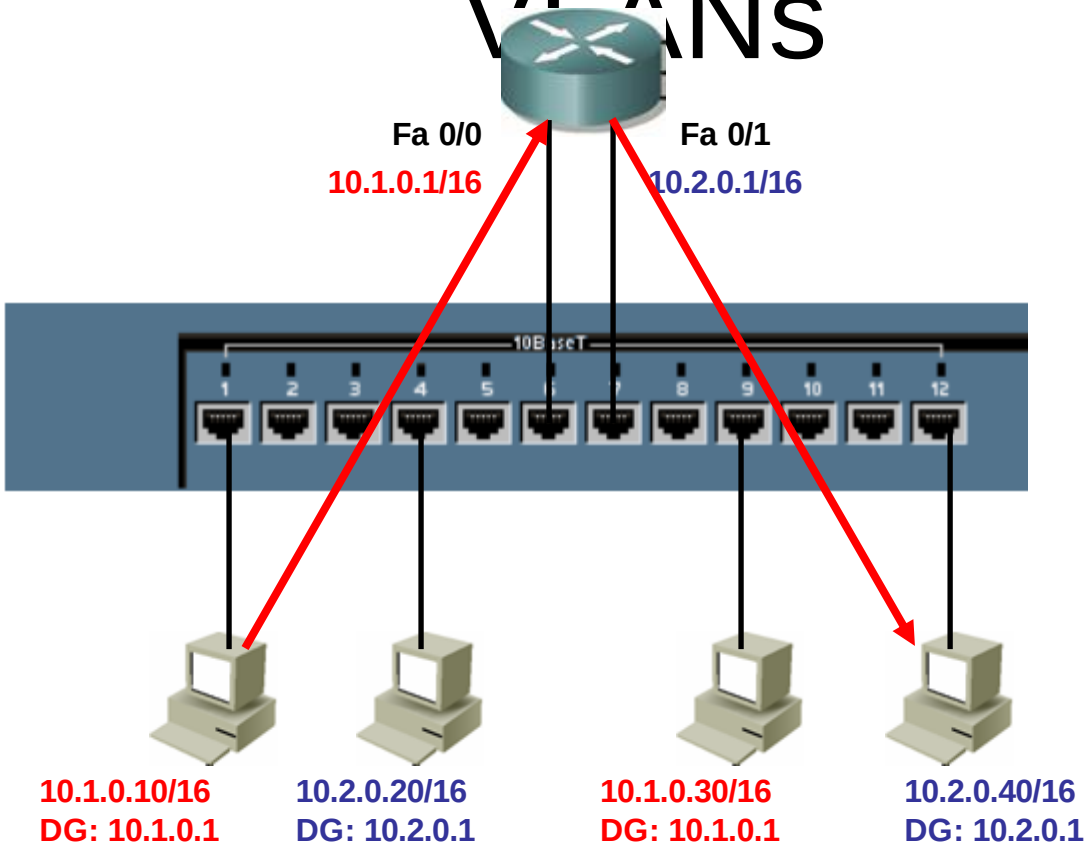
- Layer 2 Broadcasts
  - Switch floods it out all ports.
  - All hosts receive broadcast, even those on a different subnet.
  - Layer 2 broadcast should be isolated to only that network.
  - Note: If the switch supports VLANs, by default all ports belong to the same VLAN and it floods it out all ports that belong to the same VLAN as the incoming port (coming).

Two Subnets, One Switch, No VLANs



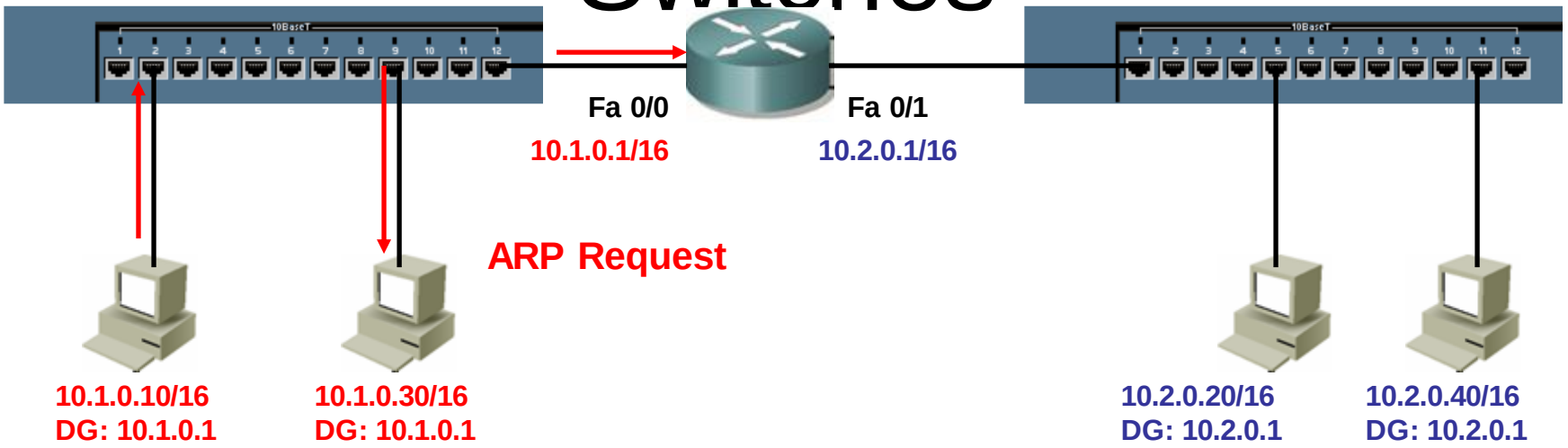
- Layer 2 Unknown Unicasts
  - This is the same for unknown unicasts.

Two Subnets, One Switch, No VLANs



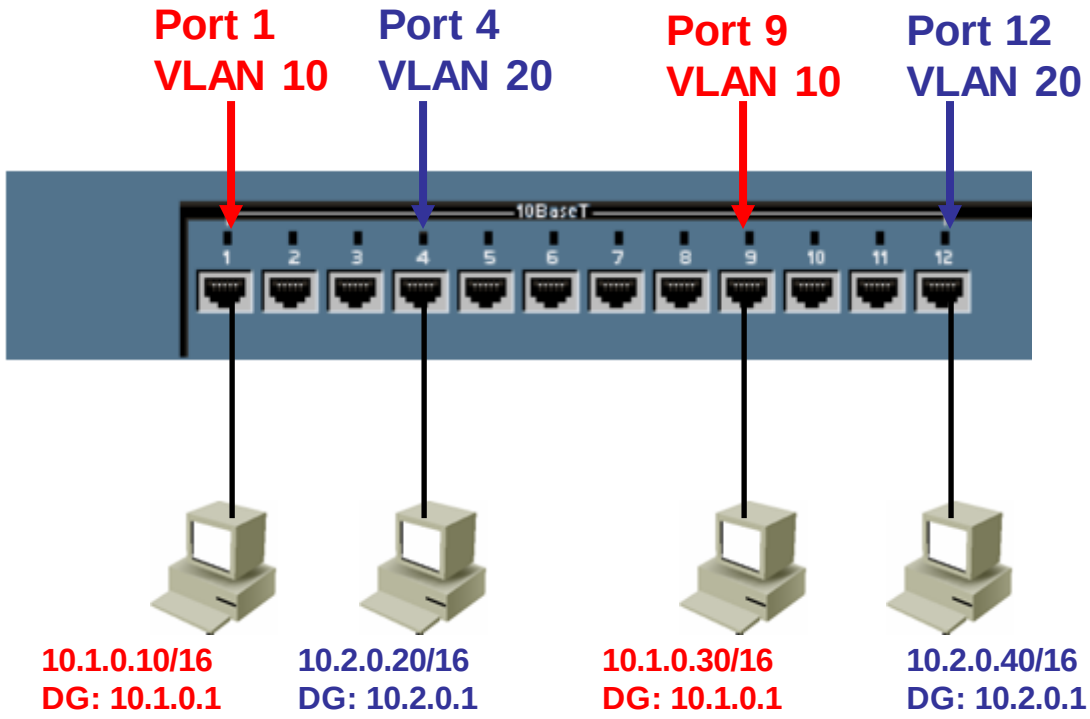
- Even though hosts are connected to the same switch (or even hub), devices on different subnets must communicate via a router.
- Remember a switch is a layer 2 device, it forwards by examining Destination MAC addresses, not IP addresses.

Traditional Solution: Multiple Switches



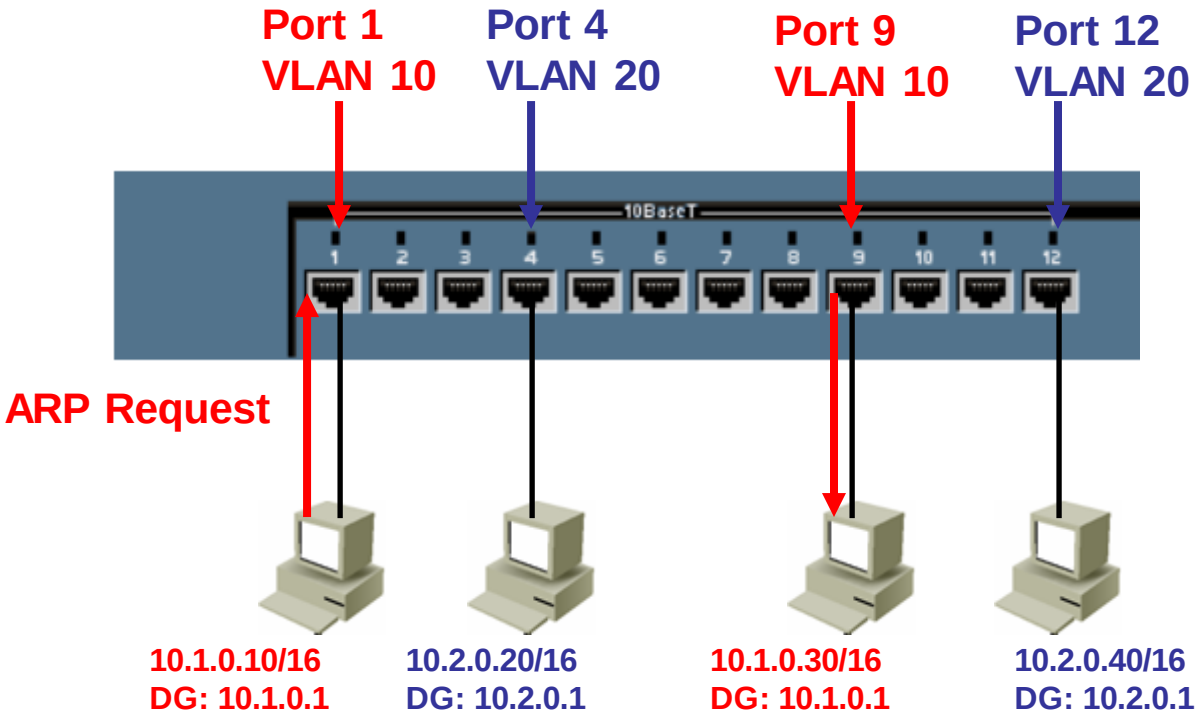
- The traditional solution is have devices on the same subnet connected to the same switch.
- This provides broadcast and unknown unicast segmentation, but is also less scalable.

Broadcast domains with VLANs and routers



- A **VLAN** is a **broadcast domain** created by one or more switches.
- VLANs are assigned on the switch and correspond with the host IP address.
- Each switch port can be assigned to a different VLAN.

Broadcast domains with VLANs and routers

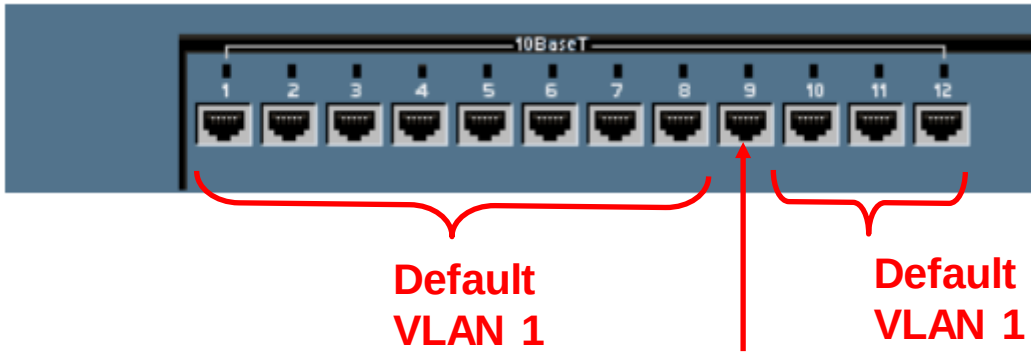


- Ports assigned to the same VLAN share the same broadcast domain.
- Ports in different VLANs do not share the same broadcast domain.

VLAN operation

Configuring VLANs	Description
Statically	Network administrators configure port-by-port. Each Port is associated with a specific VLAN. The network administrator is responsible for keying in the mappings between the ports and VLANs.
Dynamically	The ports are able to dynamically work out their VLAN configuration. Uses a software database of MAC address to VLAN mappings (which the network administrator must set up first).

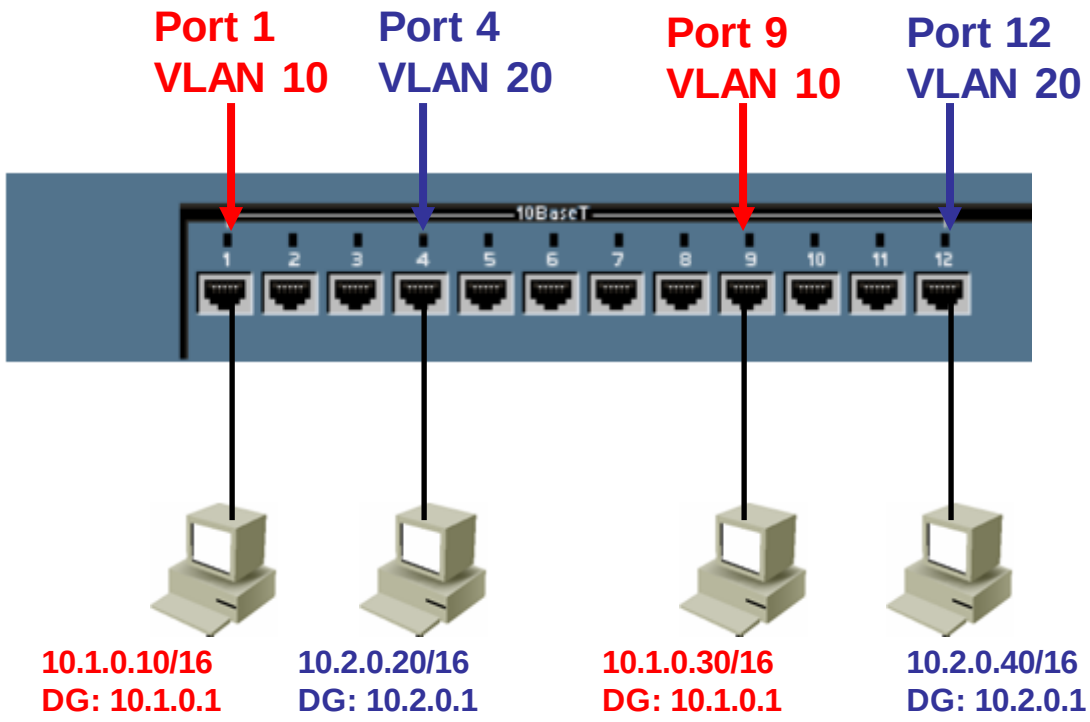
Static VLANS




```
Switch(config)#interface fastethernet 0/9
Switch(config-if)#switchport access vlan 10
```

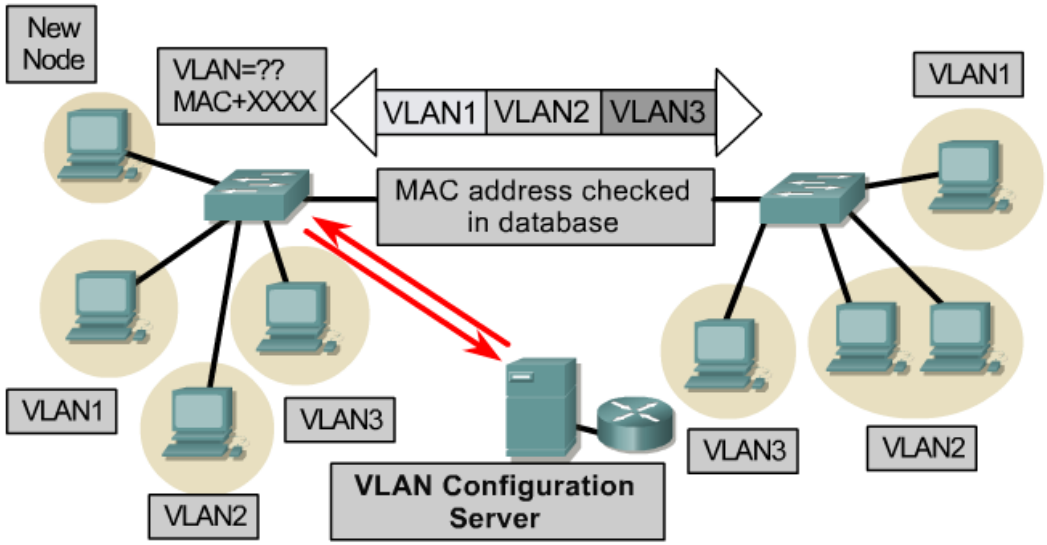
- Static membership VLANs are called **port-based** .
- This is the most common method of assigning ports to VLANs.
- As a device enters the network, it automatically assumes the VLAN membership of the port to which it is attached.
- There is a **default VLAN**, on Cisco switches that is VLAN 1.

VLAN operation



- VLANs are assigned on the switch port.
- In order for a host to be a part of that VLAN, it must be assigned an IP address that belongs to the proper subnet.
  - Remember: VLAN = Subnet

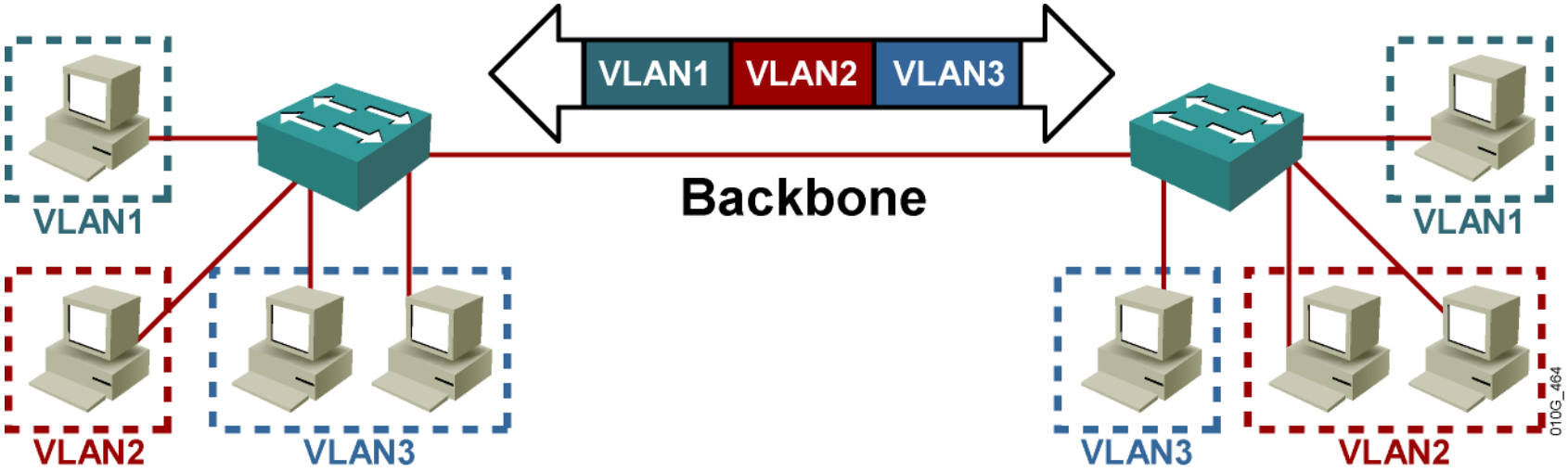
VLAN operation



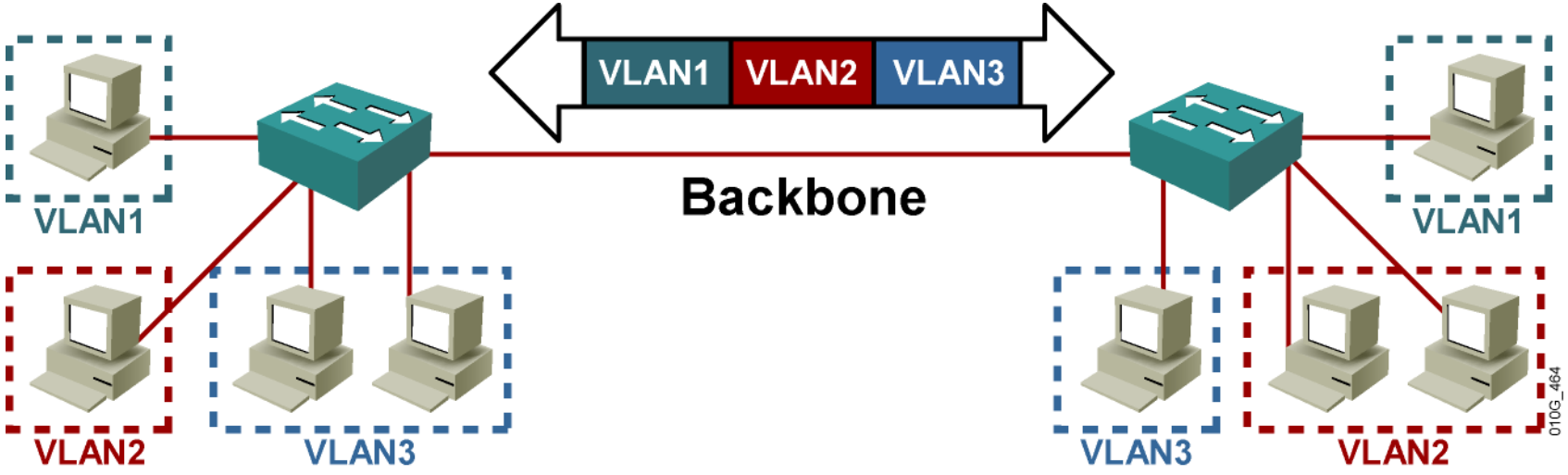
- **Dynamic membership** VLANs are created through network management software. (Not as common as static VLANs)
- CiscoWorks 2000 or CiscoWorks for Switched Internetworks is used to create Dynamic VLANs.
- Dynamic VLANs allow for membership based on the MAC address of the device connected to the switch port.
- As a device enters the network, it queries a database within the switch for a VLAN membership.

Quick Introduction to Trunking

- More in the next presentation.

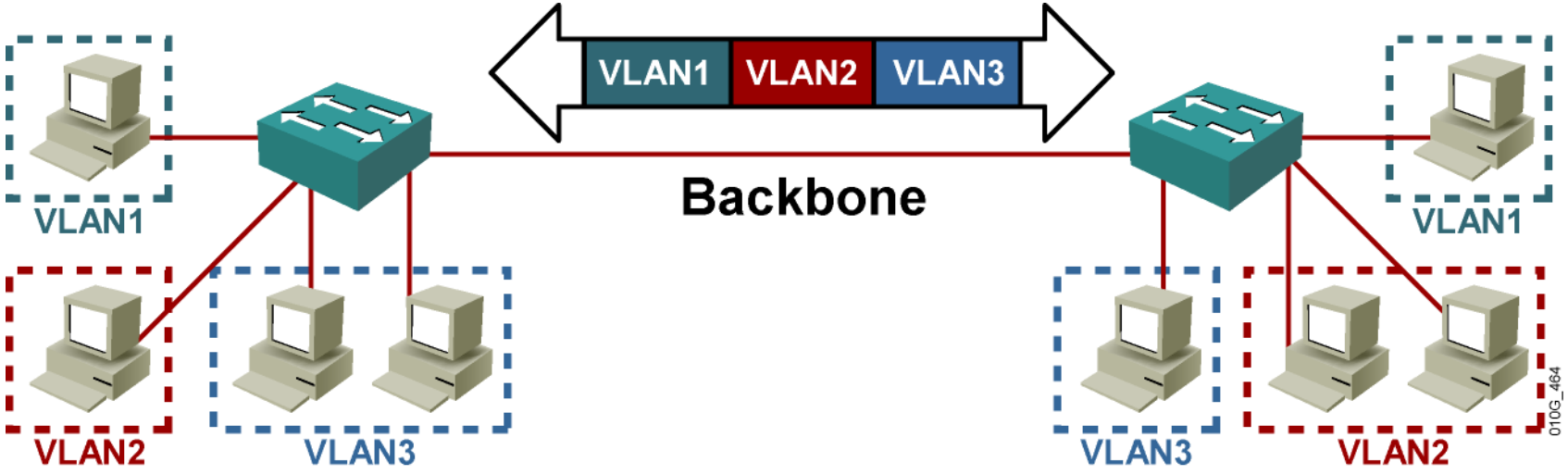


VLAN Trunking/Tagging



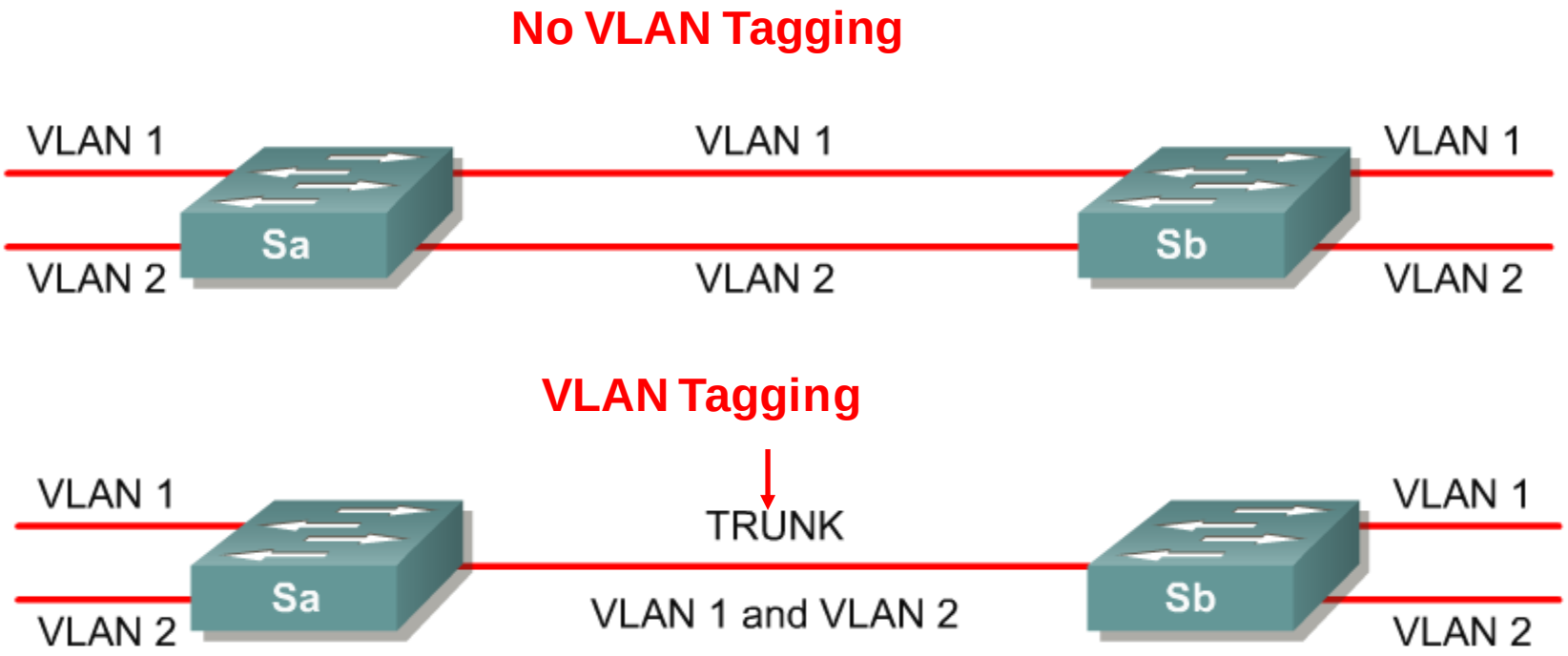
- **VLAN Tagging** is used when a link needs to carry traffic for more than one VLAN.
- **Trunk link:** As packets are received by the switch from any attached end-station device, **a unique packet identifier** is added within each header.
- This header information designates the VLAN membership of each packet.

VLAN Trunking/Tagging



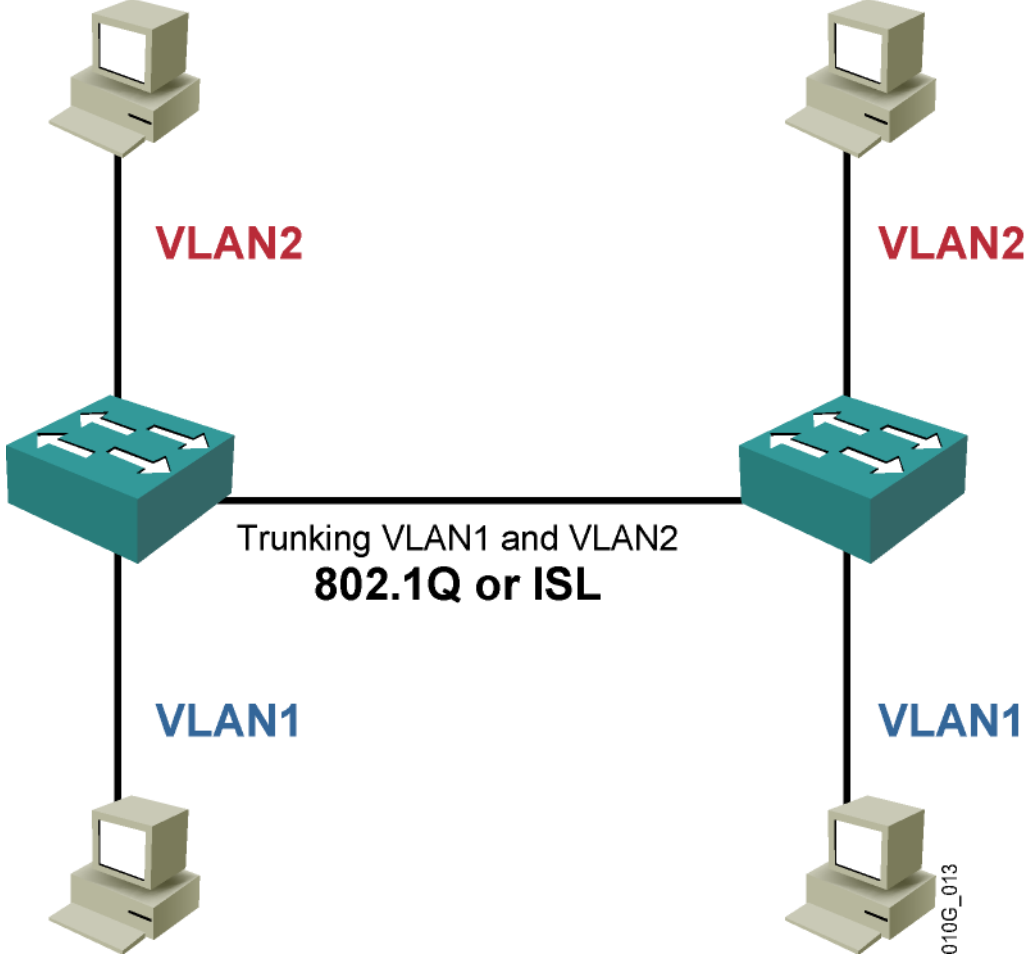
- The packet is then forwarded to the appropriate switches or routers based on the VLAN identifier and MAC address.
- Upon reaching the **destination node (Switch)** the **VLAN ID is removed** from the packet by the adjacent switch and forwarded to the attached device.
- Packet tagging provides a mechanism for controlling the flow of broadcasts and applications while not interfering with the network and applications.
- This is known as a trunk link or VLAN trunking.

VLAN Trunking/Tagging



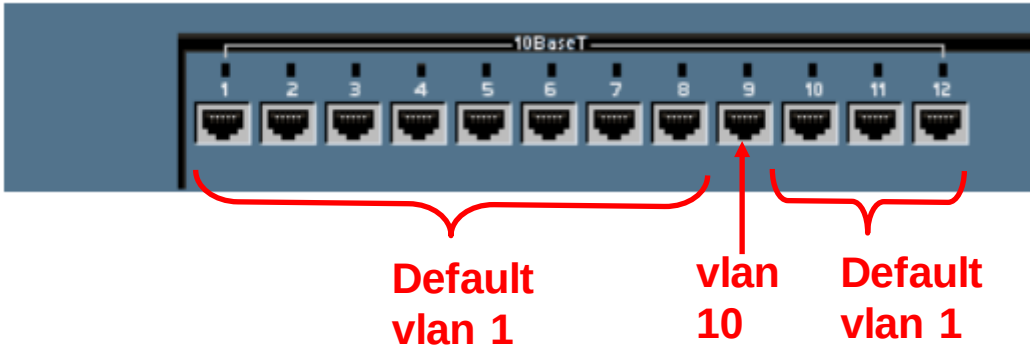
- VLAN Tagging is used when a single link needs to carry traffic for more than one VLAN.

VLAN Trunking/Tagging

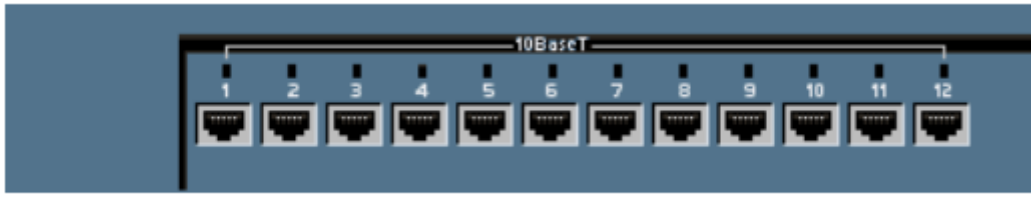


- There are two major methods of frame tagging, Cisco proprietary **Inter-Switch Link (ISL)** and **IEEE 802.1Q**.
- ISL used to be the most common, but is now being replaced by 802.1Q frame tagging.
- Cisco recommends using 802.1Q.

Configuring VLANs

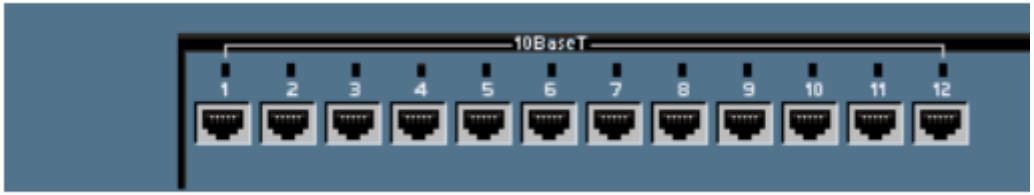


Configuring static VLANs



- The following guidelines must be followed when configuring VLANs on Cisco 29xx switches:
  - The maximum number of VLANs is switch dependent.
    - 29xx switches commonly allow 4,095 VLANs
  - VLAN 1 is one of the factory-default VLANs.
  - VLAN 1 is the default Ethernet VLAN.

Creating VLANs



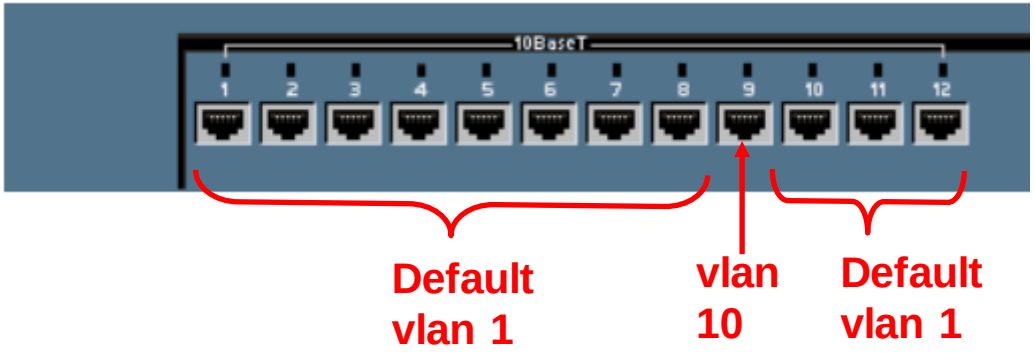
- **Assigning access ports (non-trunk ports) to a specific VLAN**

```
Switch(config)#interface fastethernet 0/9
Switch(config-if)#switchport access vlan vlan_number
Switch(config-if)#switchport mode access
```

- **Create the VLAN: (This step is <u>not</u> required and will be discussed later.)**

```
Switch#vlan database
Switch(vlan)#vlan vlan_number
Switch(vlan)#exit
```

Creating VLANs

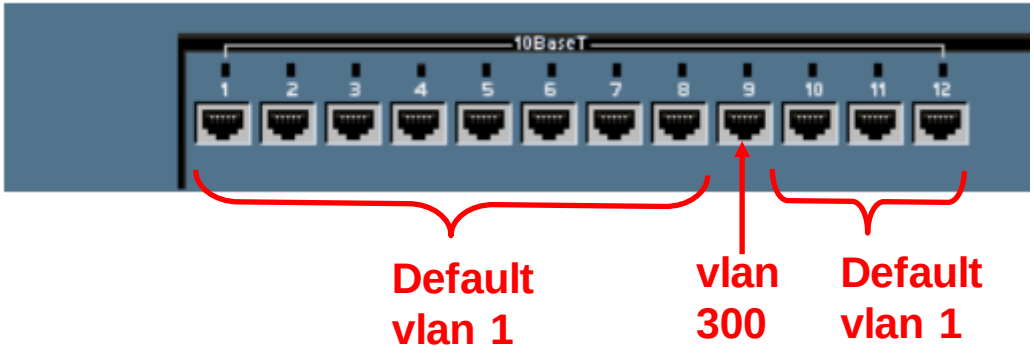


- Assign ports to the VLAN

```
Switch(config)#interface fastethernet 0/9
Switch(config-if)#switchport access vlan 10
Switch(config-if)#switchport mode access
```

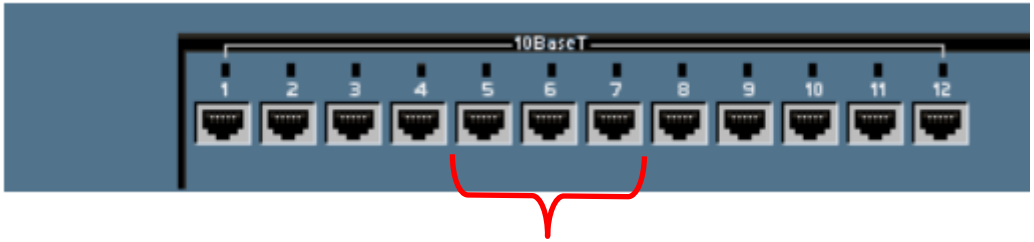
- **`access`** – Denotes this port as an access port and not a trunk link (later)

Creating VLANs



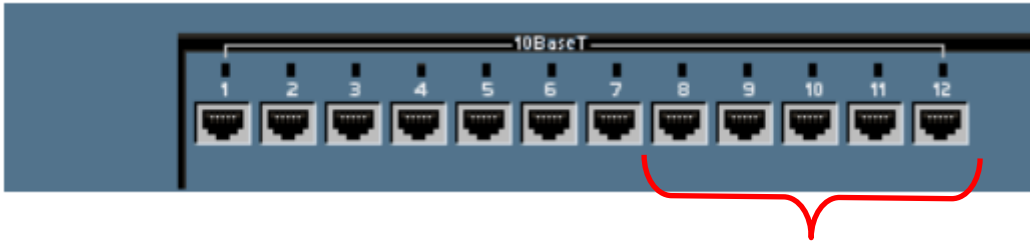

```
Switch(config)#interface fastethernet 0/9
Switch(config-if)#switchport access vlan 300
Switch(config-if)#switchport mode access
```

Configuring Ranges of VLANs



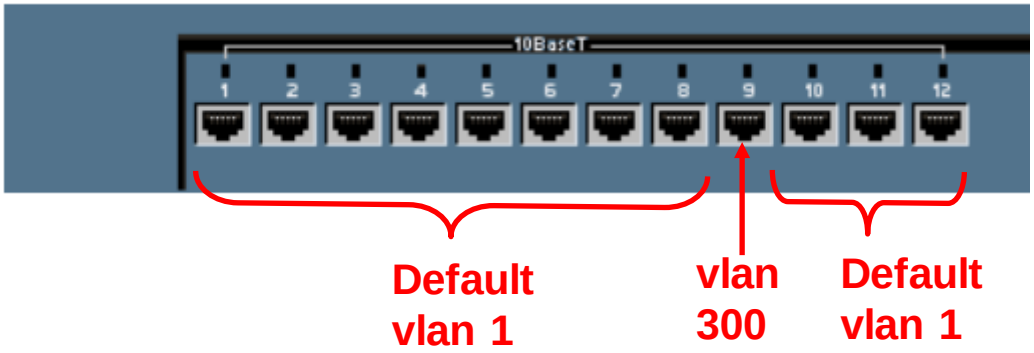

```
Switch(config)#interface fastethernet 0/5
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
Switch(config-if)#exit
Switch(config)#interface fastethernet 0/6
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
Switch(config-if)#exit
Switch(config)#interface fastethernet 0/7
Switch(config-if)#switchport access vlan 2
Switch(config-if)#switchport mode access
```

Configuring Ranges of VLANs




```
Switch(config)#interface range fastethernet 0/8 - 12
Switch(config-if)#switchport access vlan 3
Switch(config-if)#switchport mode access
Switch(config-if)#exit
```

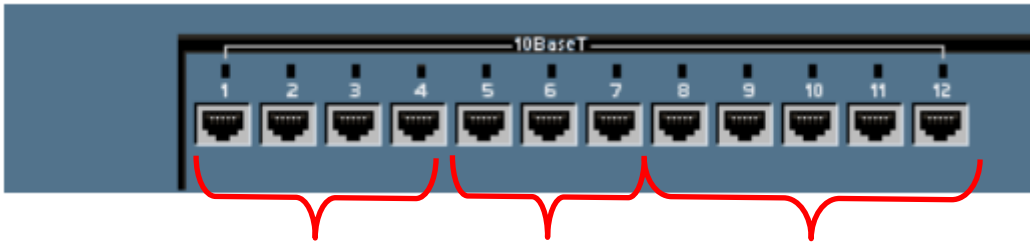
Creating VLANs




```
SydneySwitch(config)#interface fastethernet 0/1
SydneySwitch(config-if)#switchport mode access
SydneySwitch(config-if)#exit
```

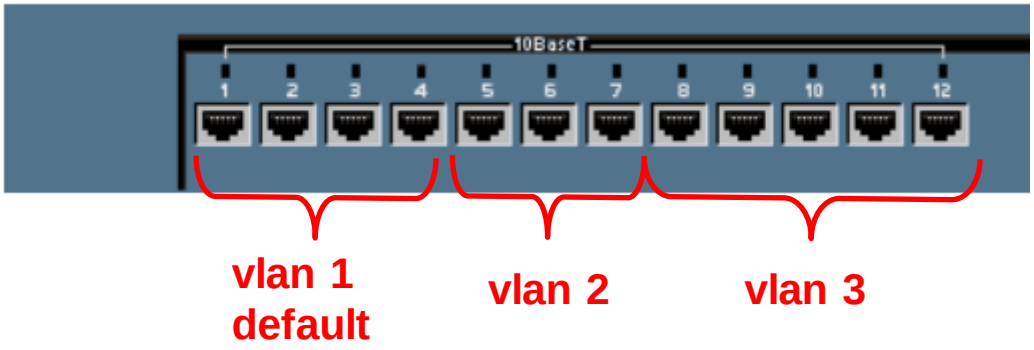
Note: The **`switchport mode access`** command should be configured on all ports that the network administrator does not want to become a trunk port.

Verifying VLANs – show vlan




```
SydneySwitch#show vlan

VLAN Name                 Status    Ports
---- -------------------- ------- -------------------------------

VLAN Name                 Status    Ports
---- -------------------- ------- -------------------------------
1    default              active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
2    VLAN2                active    Fa0/5, Fa0/6, Fa0/7
3    VLAN3                active    Fa0/8, Fa0/9, Fa0/10, Fa0/11,
                                    Fa0/12
1002 fddi-default         active
1003 token-ring-default   active
1004 fddinet-default      active
1005 trnet-default        active

VLAN Type SAID MTU  Parent RingNo BridgeNo Stp BrdgMode Trans1 Trans2
---- ---- ---- ---- ------ ------ -------- ---------- ------ -------
1    enet 100001 1500  -      -      -        -   -     1002   1003
2    enet 100002 1500  -      -      -        -   -     0      0
```

Verifying VLANs – show vlan brief

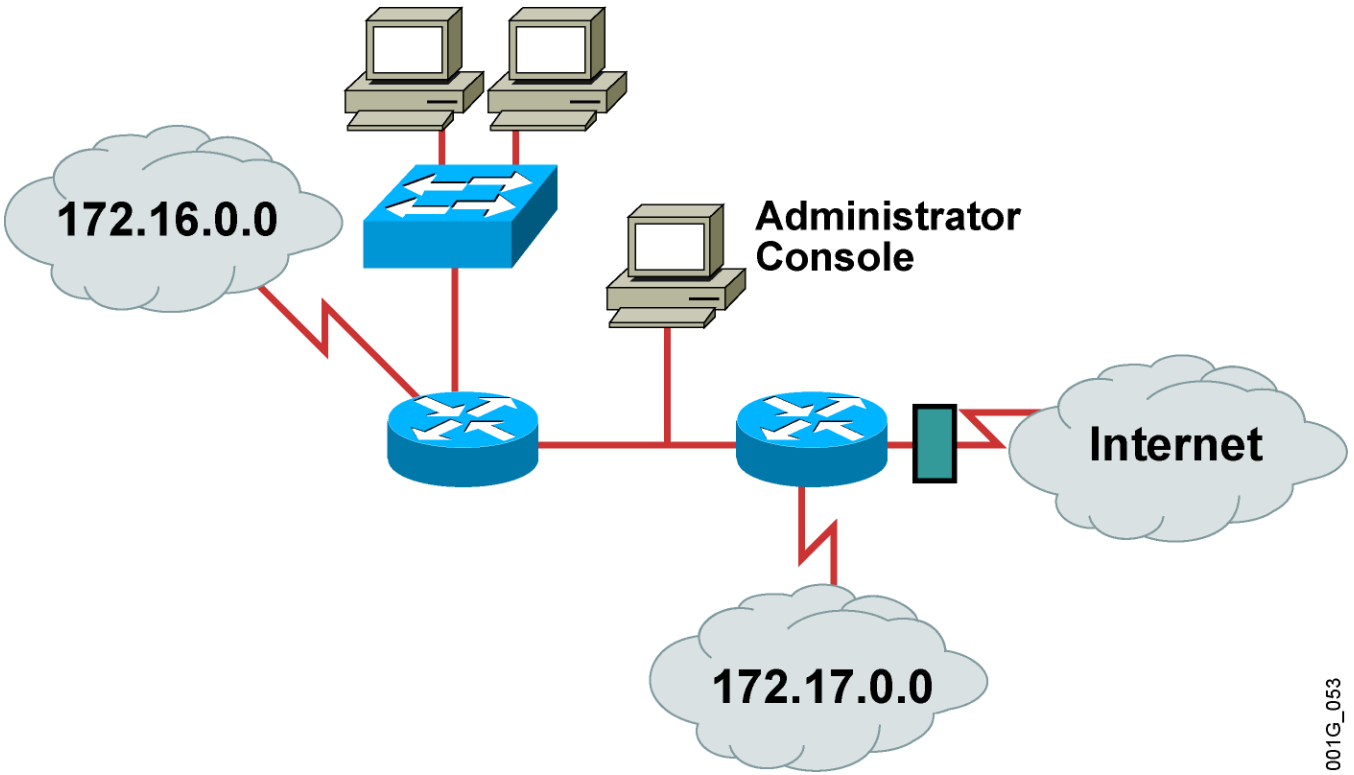



```
SydneySwitch#show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
2    VLAN2                            active    Fa0/5, Fa0/6, Fa0/7
3    VLAN3                            active    Fa0/8, Fa0/9, Fa0/10, Fa0/11,
                                                Fa0/12
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active
```

Managing IP Traffic with ACLs

Introducing ACLs

Why Use ACLs?



– Manage IP traffic as network access grows
– Filter packets as they pass through the router

- IP Access control list: (ACL)
- ACL is a set of commands that are grouped under certain Name or Number to control traffic flow (permit or deny).
- Access list is configured on the router then activated on interfaces.

- ACL processing:

1- Statements are checked from up to down.

2- Once a match found, no further checking.

3- If no match found, the packet will be dropped due to the “ implicit deny “ statement at the end of the ACL.

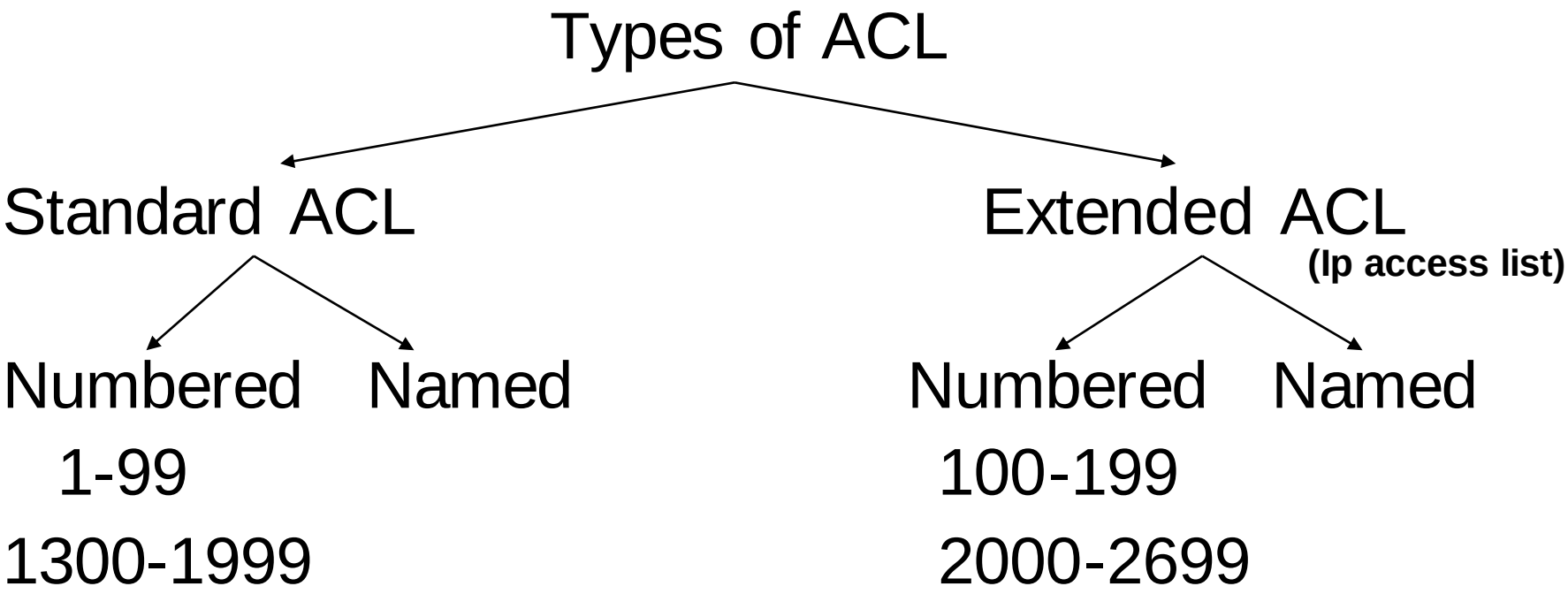
4- ACL must contain at least one permit statement otherwise all packets will be dropped.

5- In any ACL , you can not add statement between statements (any new statements can only be added to the end of ACL).

6- In Numbered ACL, you can not delete a certain statement , only delete the whole ACL.

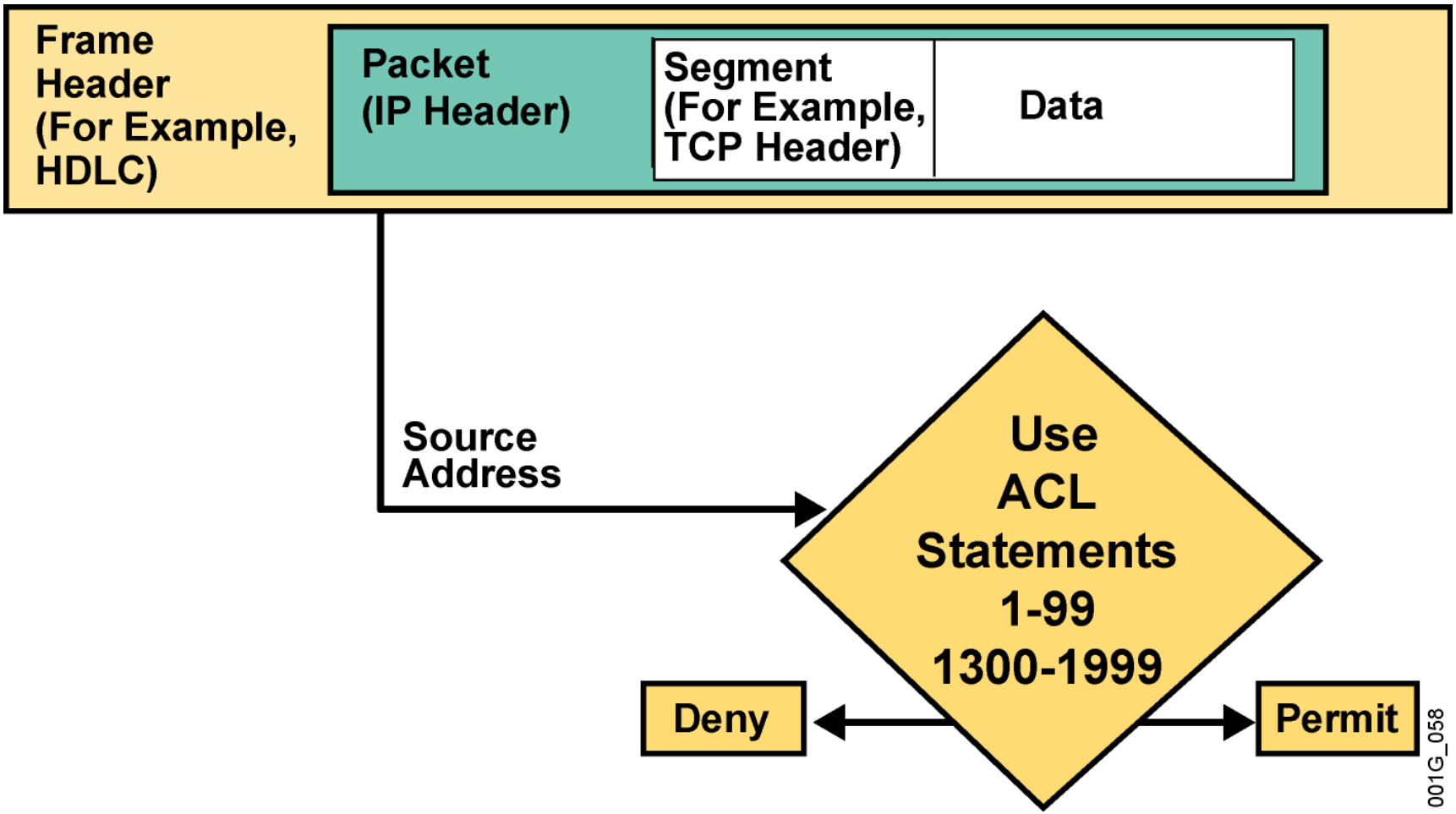
7- In Named ACL, you can delete a certain statement between statements.

- Note:
- You can have one ACL per interface per protocol per direction.



Standard ACLs

It filters the packets based on source ip address in the packet header.



Standard IP ACL Configuration

```
Router(config)#access-list access-list-number
{permit | deny} source [mask]
```

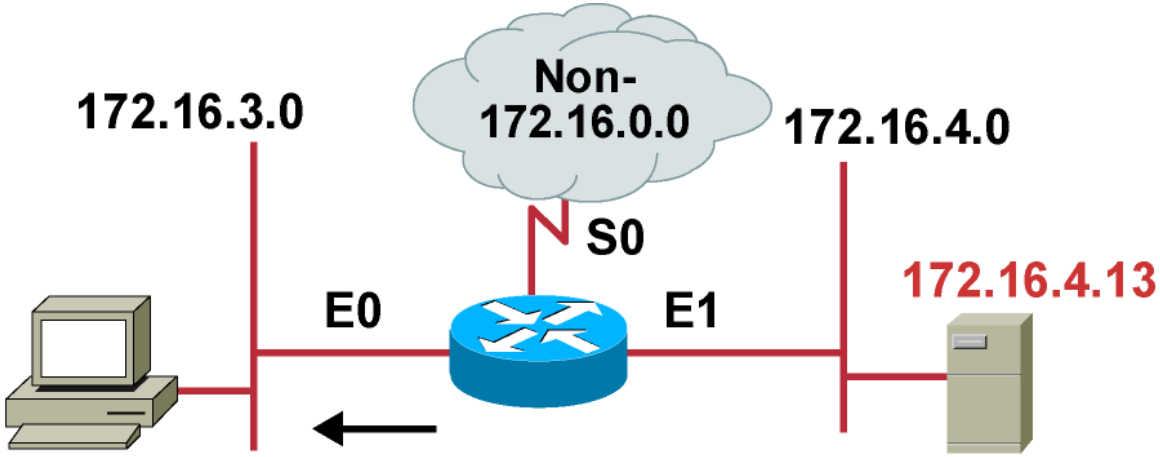
Action

- **Sets parameters for this list entry**
- **IP standard ACLs use 1 to 99**
- **Default wildcard mask = 0.0.0.0**
- no access-list *access-list-number* **removes entire ACL**

```
Router(config-if)#ip access-group
access-list-number  {in | out}
```

  - Activates the list on an interface
  - Sets inbound or outbound testing
  - **no ip access-group *access-list-number*** removes ACL from the interface

Standard IP ACL Example 1

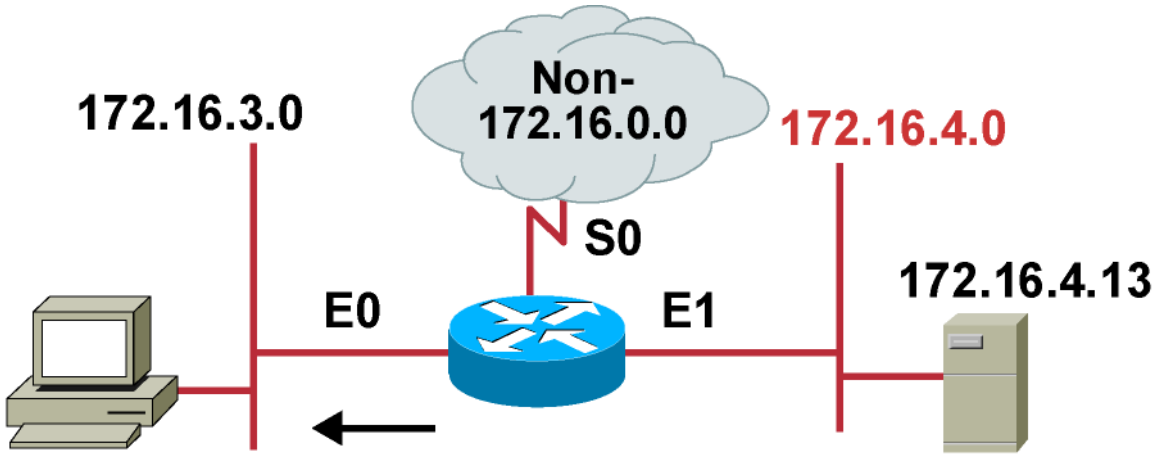



```
Router(config)#access-list 1 deny 172.16.4.13 0.0.0.0
Router(config)#access-list 1 permit 0.0.0.0  255.255.255.255
(implicit deny all)
(access-list 1 deny 0.0.0.0   255.255.255.255)

Router(config)#interface ethernet 0
Router(config)#ip access-group 1 out
```

001G_068

- **Deny a specific host.**

Standard IP ACL Example 2

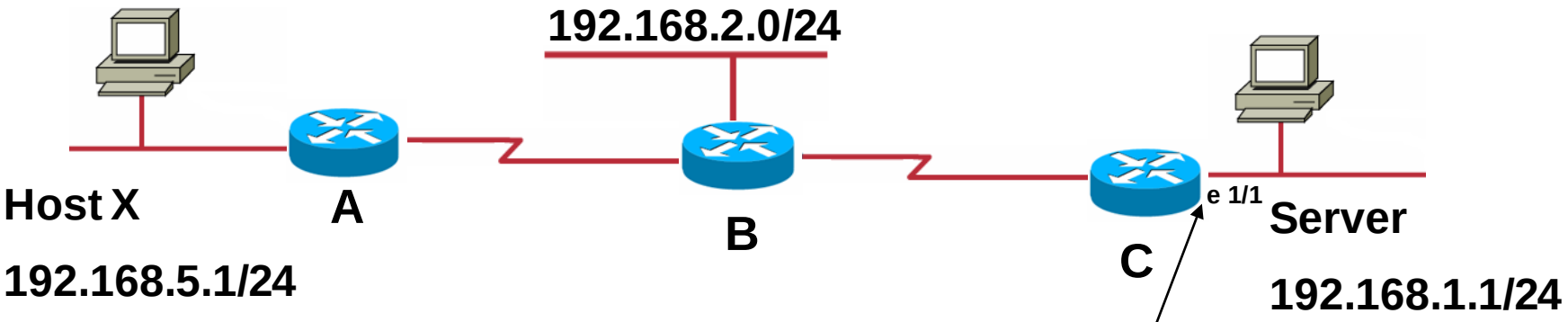



```
Router(config)#access-list 1 deny 172.16.4.0  0.0.0.255
Router(config)#access-list 1 permit any
(implicit deny all)
(access-list 1 deny 0.0.0.0   255.255.255.255)

Router(config)#interface ethernet 0
Router(config)#ip access-group 1 out
```

001G_069

- **Deny a specific subnet.**

Placement of standard ACL:



- We want to restrict the user X from accessing the server.

```
(config)# access-list 1 deny 192.168.5.1
        # access-list 1 permit any
C(config)# int e 1/1
(config-if)# ip access-group 1 out
```

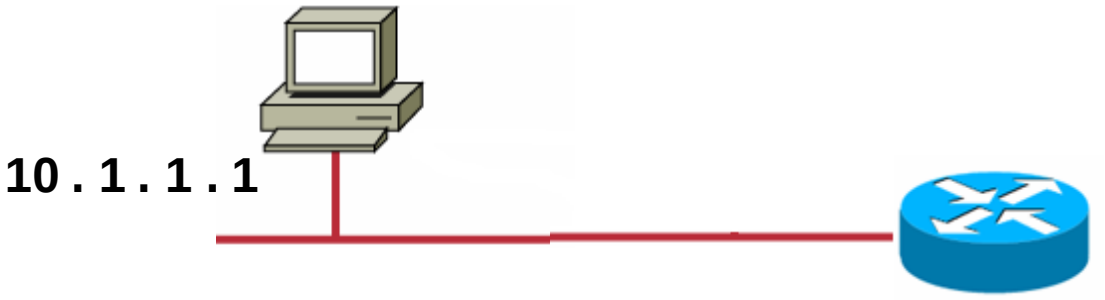
0.0.0.0 255.255.255.255

- Rule:

- Standard ACL is placed as close as possible to destination.

- To control telnet access to router:

We want to restrict the telnet access from host 10.1.1.1 to the router.



(config)# access-list 1 deny 10.1.1.1

(config)# access-list 1 permit any

(config)# line vty 0 4

(config-line)# access-class 1 in

Note: Router can not filter IP packets that sourced by router itself.

Using Named IP ACL (Standard)

```
Router(config)#ip access-list {standard | extended} name
```

- **Alphanumeric name string must be unique.**

```
Router(config-std-nacl)#{permit | deny}
{ip access list test conditions}
Router(config-std-nacl)#no {permit | deny}
{ip access list test conditions}
```

- **Permit or deny statements have no prepended number.**
- **“no” removes the specific test from the named ACL.**

```
Router(config-if)#ip access-group name {in | out}
```

- **Activates the named IP ACL on an interface.**

Extended ACLs

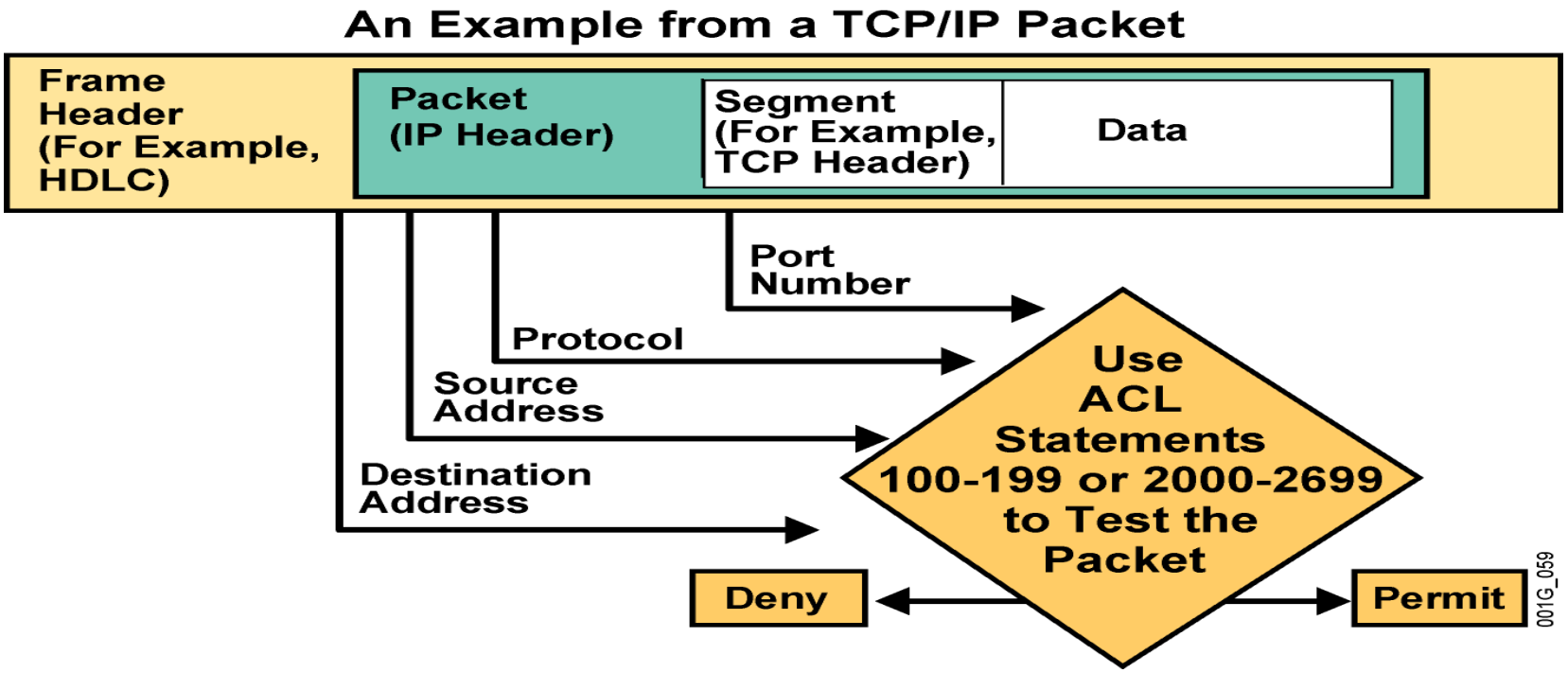
-It is more flexible than standard ACL.

-Extended ACL can match on:

1- Source IP , Destination IP.

2- TCP/IP protocols. (IP, TCP, UDP, ICMP,…….).

3- Protocol information (port no.).



Extended IP ACL Configuration

```
Router(config)#access-list access-list-number
{permit | deny} protocol source source-wildcard
[operator port] destination destination-wildcard
[operator port]
```

- **Sets parameters for this list entry**

```
Router(config-if)#ip access-group access-list-number
{in | out}
```

- **Activates the extended list on an interface**

- Note:

- There are two special types of W.C.M:

1- 0.0.0.0 is called host mask.

Ex: 1.1.1.1 0.0.0.0 = host 1.1.1.1

2- 255.255.255.255 is called any.

Ex: 0.0.0.0 255.255.255.255 = any

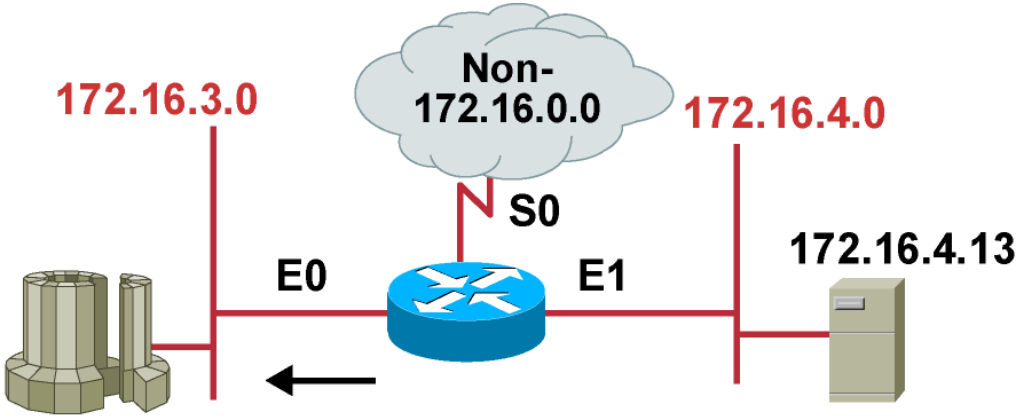
- The operators

eq 80 = eq http.

(Lt) operator means less than or equal.

(gt) operator means greater than or equal.

Extended ACL Example 1

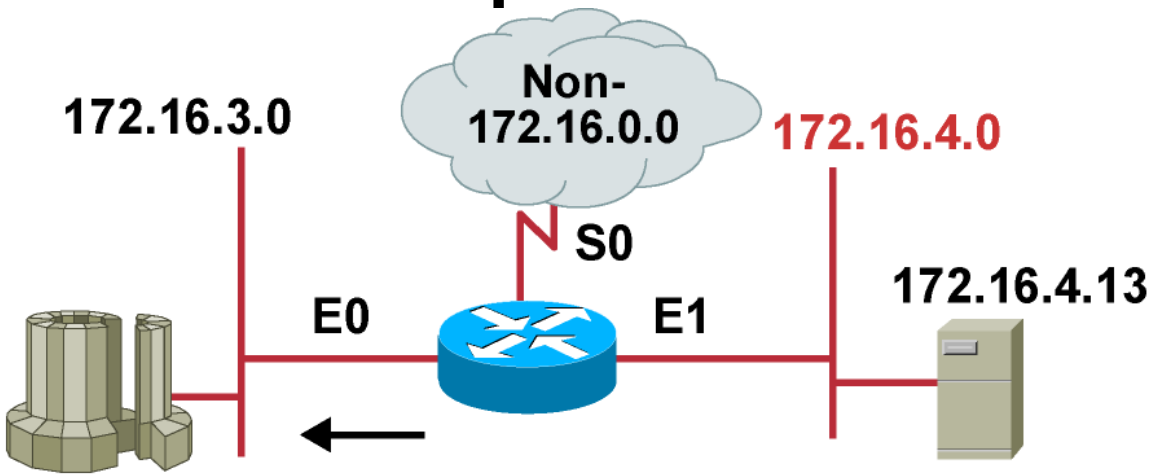



```
Router(config)#access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
Router(config)#access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
Router(config)#access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)

Router(config)#interface ethernet 0
Router(config)#ip access-group 101 out
```

001G_070

- Deny FTP from subnet 172.16.4.0 to subnet 172.16.3.0 out E0.
- Permit all other traffic.

Extended ACL Example 2

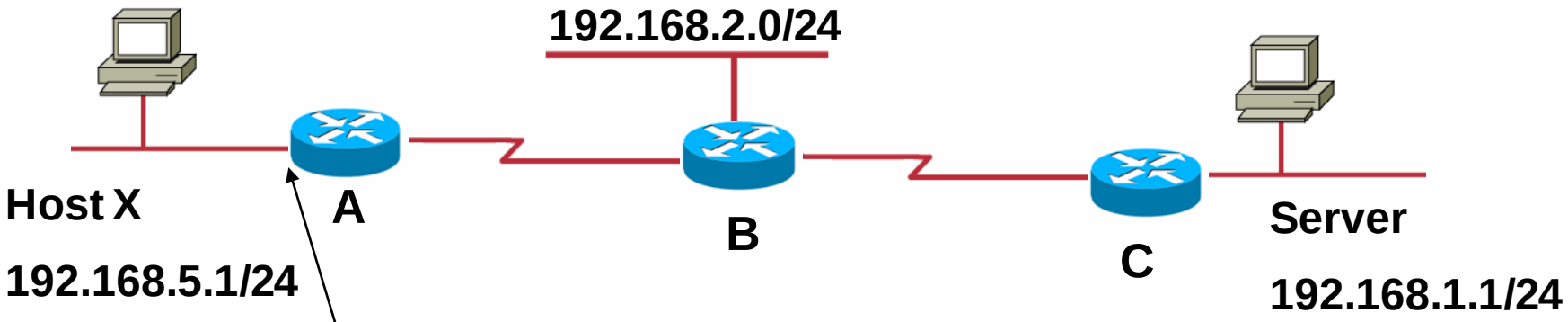



```
Router(config)#access-list 101 deny tcp 172.16.4.0  0.0.0.255  any eq 23
Router(config)#access-list 101 permit ip any any
(implicit deny all)

Router(config)#interface ethernet 0
Router(config)#ip access-group 101 out
```

001G_071

- Deny only Telnet from subnet 172.16.4.0 out E0.
- Permit all other traffic.

Placement of Extended ACL:



- We want to restrict the user X from accessing the server.

```
(config)# Access-list  100 deny ip 192.168.5.1  0.0.0.0
(config)# Access-list  100 permit any any
A (config)#  int e0/0
A (config)# ip access-group 100 in
```

- Rule:

- Extended ACL is placed as close as possible to source.

Using Named IP ACL (Extended)

```
Router(config)#ip access-list {standard | extended} name
```

- **Alphanumeric name string must be unique.**

```
Router(config-ext-nacl)#{permit | deny}
{ip access list test conditions}
Router(config-ext-nacl)#no {permit | deny}
{ip access list test conditions}
```

- **Permit or deny statements have no prepended number.**
- **“no” removes the specific test from the named ACL.**

```
Router(config-if)#ip access-group name {in | out}
```

- **Activates the named IP ACL on an interface.**

Monitoring ACL Statements

```
wg_ro_a#show {protocol} access-list {access-list number}
```

```
wg_ro_a#show access-lists {access-list number}
```

```
wg_ro_a#show access-lists
Standard IP access list 1
    permit 10.2.2.1
    permit 10.3.3.1
    permit 10.4.4.1
    permit 10.5.5.1
Extended IP access list 101
    permit tcp host 10.22.22.1 any eq telnet
    permit tcp host 10.33.33.1 any eq ftp
    permit tcp host 10.44.44.1 any eq ftp-data
```

Verifying ACLs

```
wg_ro_a#show ip interfaces e0
Ethernet0 is up, line protocol is up
  Internet address is 10.1.1.11/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  access list is 1
  Proxy ARP is enabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is
disabled
  IP Feature Fast switching turbo vector
  IP multicast fast switching is enabled
  IP multicast distributed fast switching is
disabled
  <text ommitted>
```

Scaling the Network with NAT and PAT

- **NAT (Network address translation)**

- Address translation allows you to translate your internal private address to a public address before the packets leave your local network to the public network.

- **NAT terminologies:**

1- Inside local IP: an internal device that has a private IP.

2- Inside global IP: an internal device that has a public IP.

3- Outside local IP: an outside device that has a private IP.

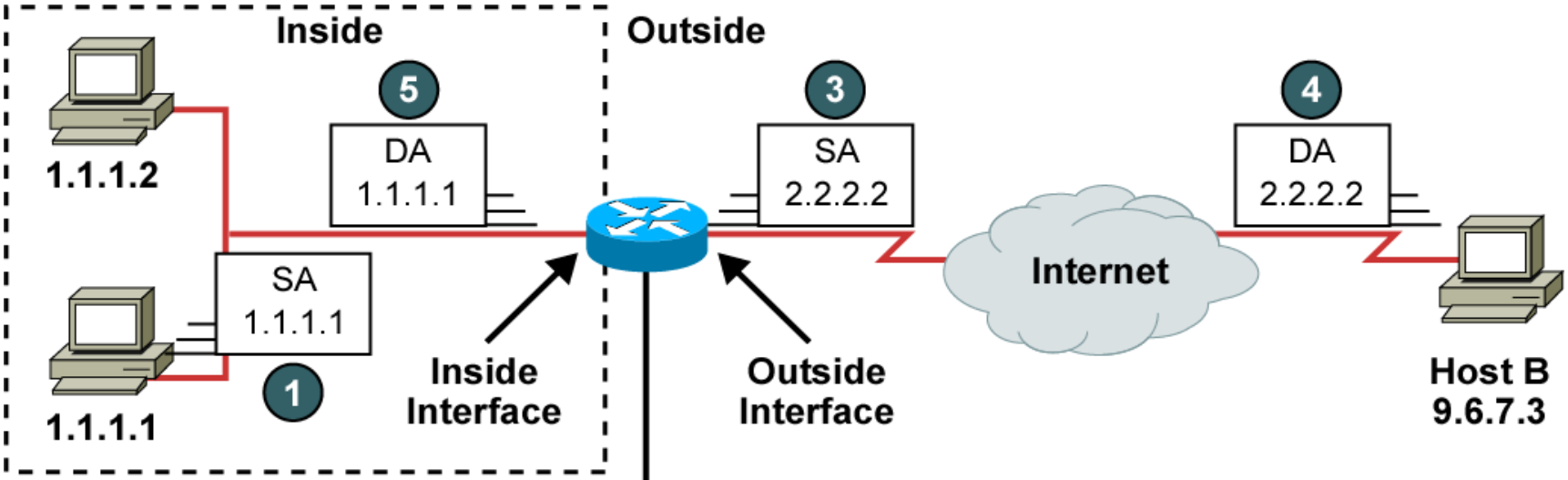
4- Outside global IP: an outside device that has a public IP.

Types of Address Translation:

- **1- Static Translation.**
- **2- Dynamic Translation.**

1- Static NAT:

- NAT table is formed manually translating private IPs to public IPs.



Inside Local IP Address	Inside Global IP Address
1.1.1.2 1.1.1.1	2.2.2.3 2.2.2.2

ICND20GR_177

Configuring Static Translation

```
Router(config)#ip nat inside source static local-ip global-ip
```

- **Establishes static translation between an inside local address and an inside global address**

```
Router(config-if)#ip nat inside
```

- **Marks the interface as connected to the inside**

```
Router(config-if)#ip nat outside
```

- **Marks the interface as connected to the outside**

2- Dynamic NAT:

- The router is given a pool of IPs that contains global IPs, so every user tries to access a public network will be given an IP from the pool.
- To configure Dynamic NAT:

1- Define the pool of IPs.

2- Define which inside addresses are allowed to be translated. (ACL)

Configuring Dynamic Translation

```
Router(config)#ip nat pool name start-ip end-ip
{netmask netmask | prefix-length prefix-length}
```

- **Defines a pool of global addresses to be allocated as needed.**

```
Router(config)#access-list access-list-number permit
source [source-wildcard]
```

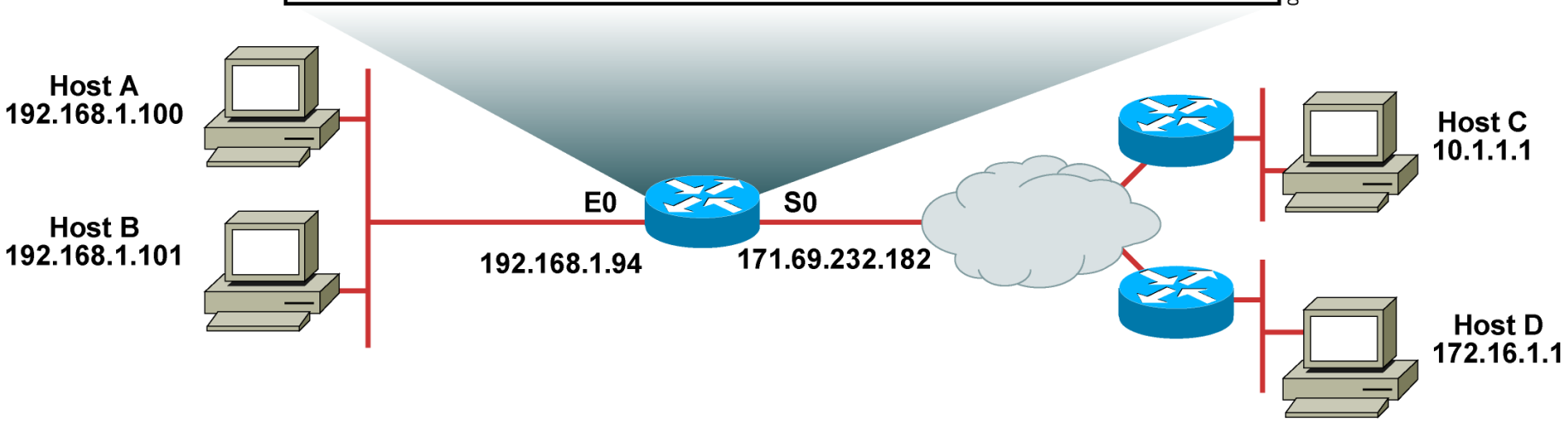
- **Defines a standard IP ACL permitting those inside local addresses that are to be translated.**

```
Router(config)#ip nat inside source list
access-list-number pool name
```

- **Establishes dynamic source translation, specifying the ACL that was defined in the prior step.**

Dynamic Address Translation Example

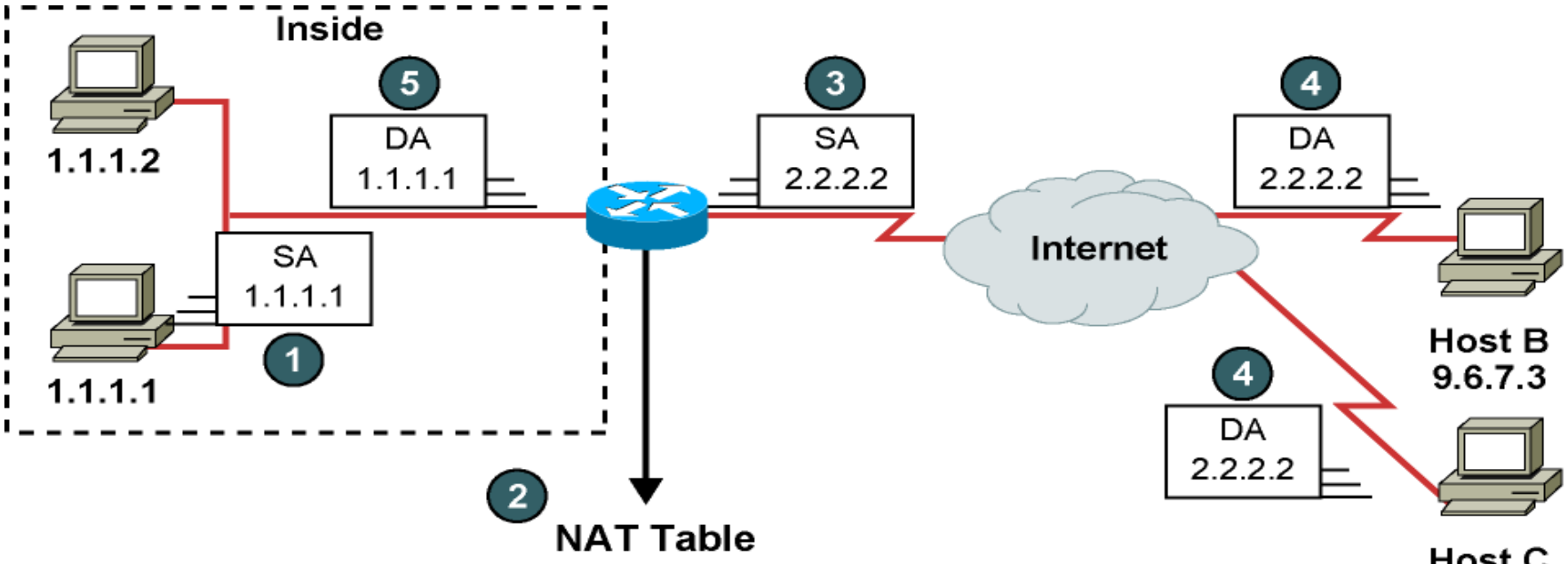
```
ip nat pool net-208 171.69.233.209 171.69.233.222 netmask
255.255.255.240
ip nat inside source list 1 pool net-208
!
interface serial 0
 ip address 171.69.232.182 255.255.255.240
 ip nat outside
!
interface ethernet 0
 ip address 192.168.1.94 255.255.255.0
 ip nat inside
!
access-list 1 permit 192.168.1.0 0.0.0.255
```



001G_186

- PAT: (port address translation)

- Static or dynamic NAT provide only one to one translation while PAT supports many to one translation.



Protocol	Inside Local IP Address:Port	Inside Global IP Address:Port	Outside Global IP Address:Port
TCP	1.1.1.2:1723	2.2.2.2:1723	6.5.4.7:23
TCP	1.1.1.1:1024	2.2.2.2:1024	9.6.7.3:23

001G_178

Configuring Overloading

```
Router(config)#access-list access-list-number permit
source source-wildcard
```

- **Defines a standard IP ACL that will be permit the inside local addresses that are to be translated**

```
Router(config)#ip nat inside source list
access-list-number interface interface overload
```

- **Establishes dynamic source translation, specifying the ACL that was defined in the prior step**

Displaying Information with show Commands

```
Router#show ip nat translations
```

- **Displays active translations (NAT & PAT tables)**

```
Router#show ip nat translation
     Pro Inside global      Inside local       Outside local      Outside global
     --- 172.16.131.1        10.10.10.1         ---                ---
```

```
Router#show ip nat statistics
```

- **Displays translation statistics**

```
Router#show ip nat statistics
      Total active translations: 1 (1 static, 0 dynamic; 0 extended)
      Outside interfaces:
      Ethernet0, Serial2.7
      Inside interfaces:
      Ethernet1
      Hits: 5  Misses: 0
      …
```