Lesson 6: Routing

Van-Linh Nguyen

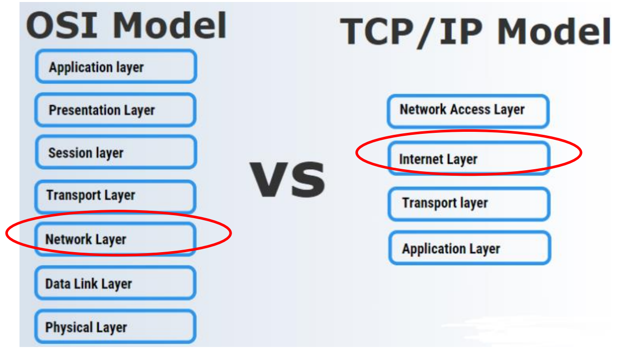
Fall 2024

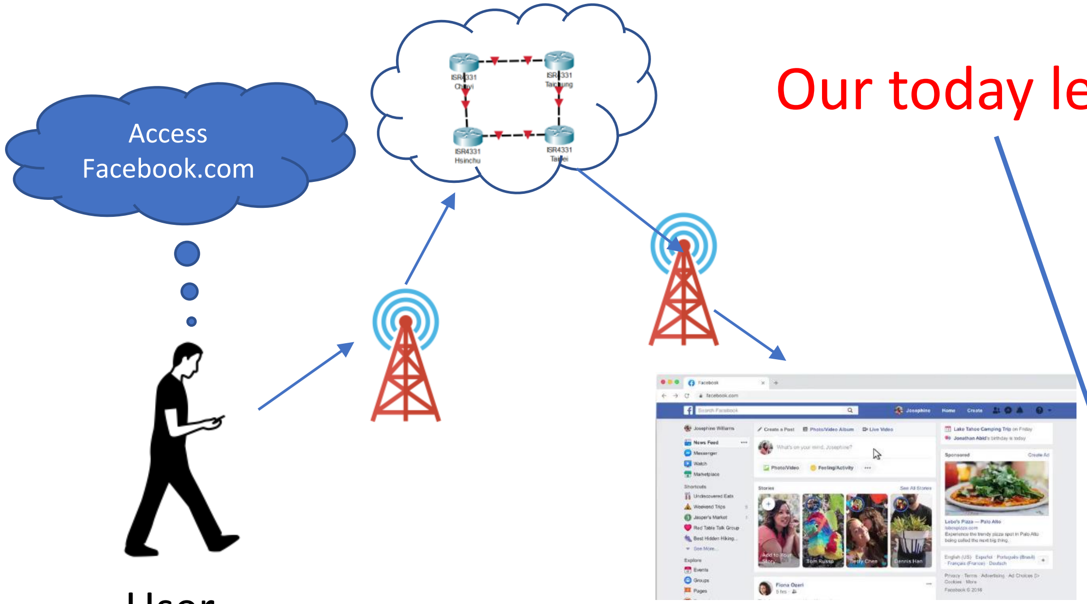
國立中正大學 Cybersecurity Lab

Outline

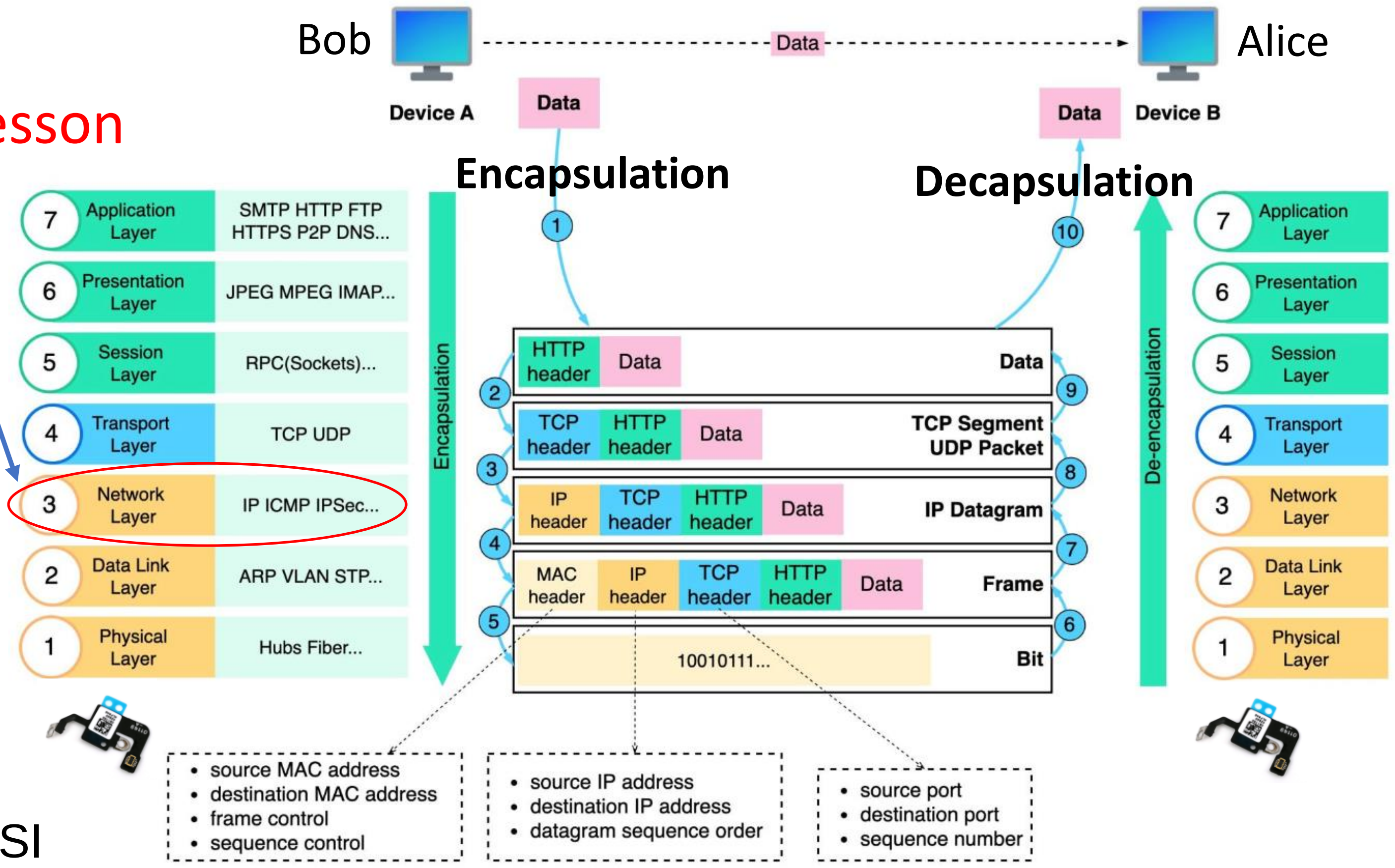
- IP Routing
  1. Internetworking
  2. Subnetting
  3. DHCP
  4. Routing Protocols
     - ✓ Distance vector
     - ✓ Link State

Encapsulation/Decapsulation





Our today lesson

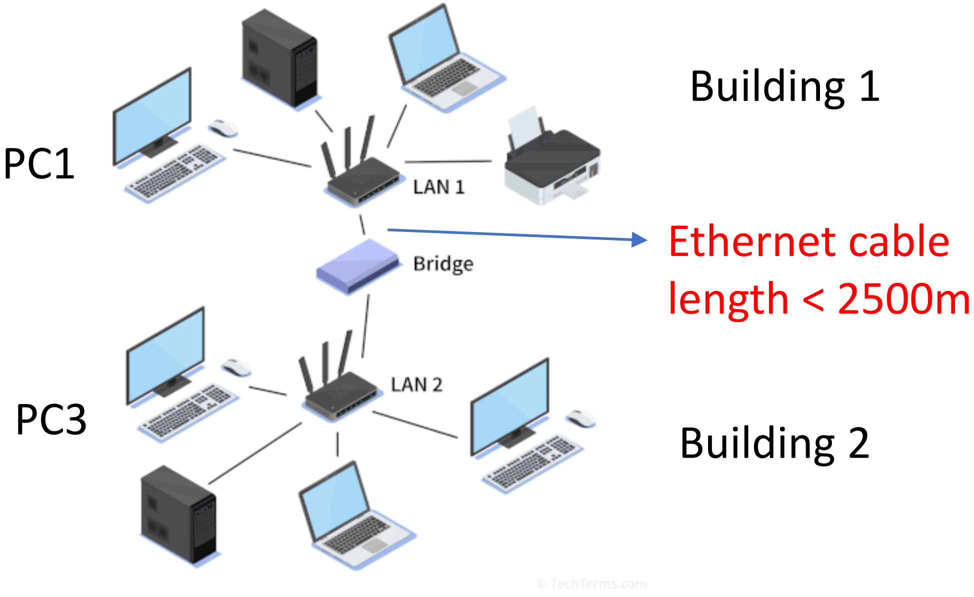


TCP/IP Model is more popular in industry than OSI
OSI is for academic research

https://blog.bytebytego.com/

Switching or Routing

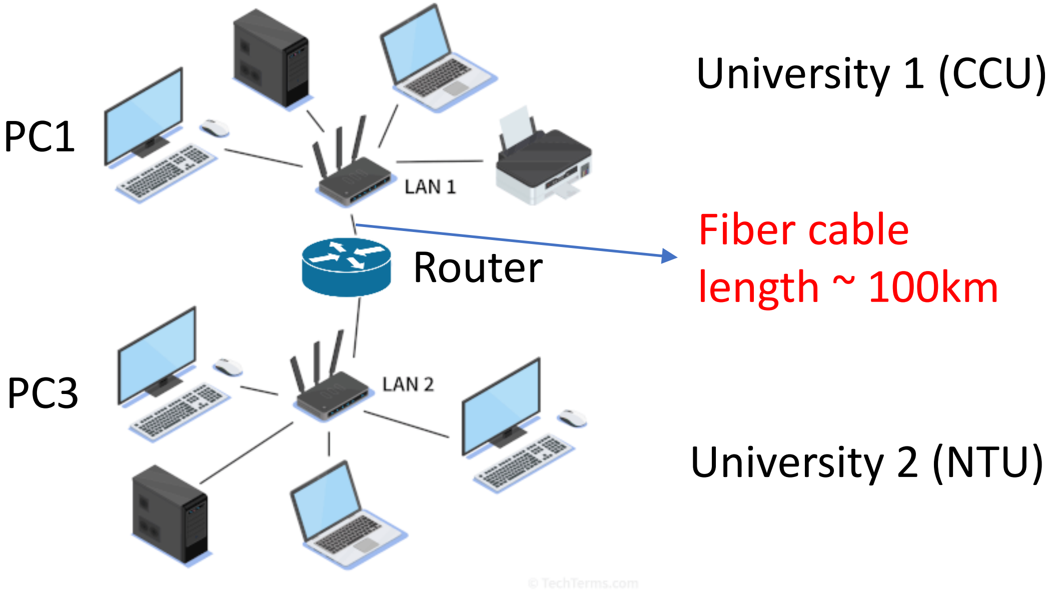
- Connect multiple networks use bridge



For small areas

Use Spanning Tree algorithm

- Connect multiple networks use router



For large areas → Internetworking

Use routing algorithm

Switching or Routing

- Switching do not offer any QoS services to enable effective packet switching (best effort).
- Routing support QoS capability offered that can prioritize different types of network traffic

Switching	Routing
Perform at Layer 2 (Data link layer)	Perform at Layer 3 (Network layer)
Use ASIC	Use software
If the destination is not known to switch, it will broadcast the frame	If the destination is not known to forward, it will drop the packet
Switching is done in the same network	Routing is done in different networks
Switching uses MAC address	Routing uses IP address
Fast and simple to setup (plug and play)	Slower and complicated to setup

Switch

Many LAN ports

Router

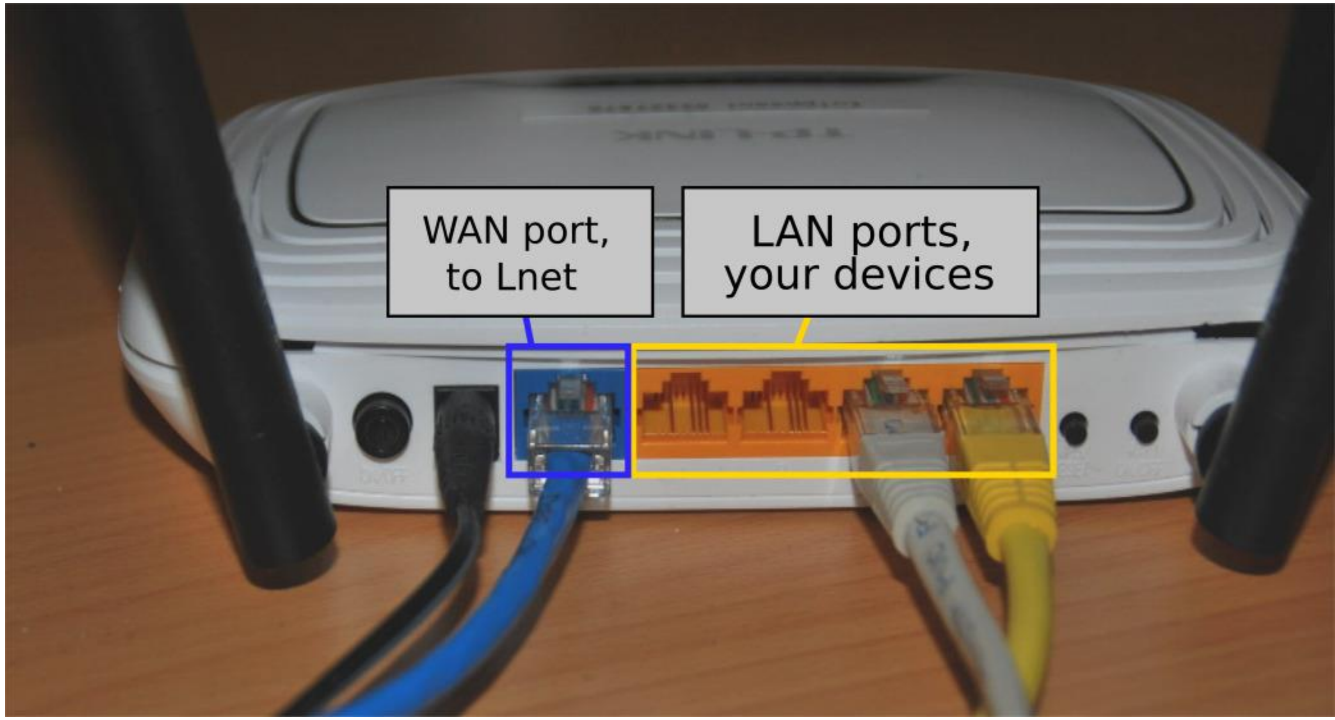
1-2 WAN ports, few LAN ports

Home Router

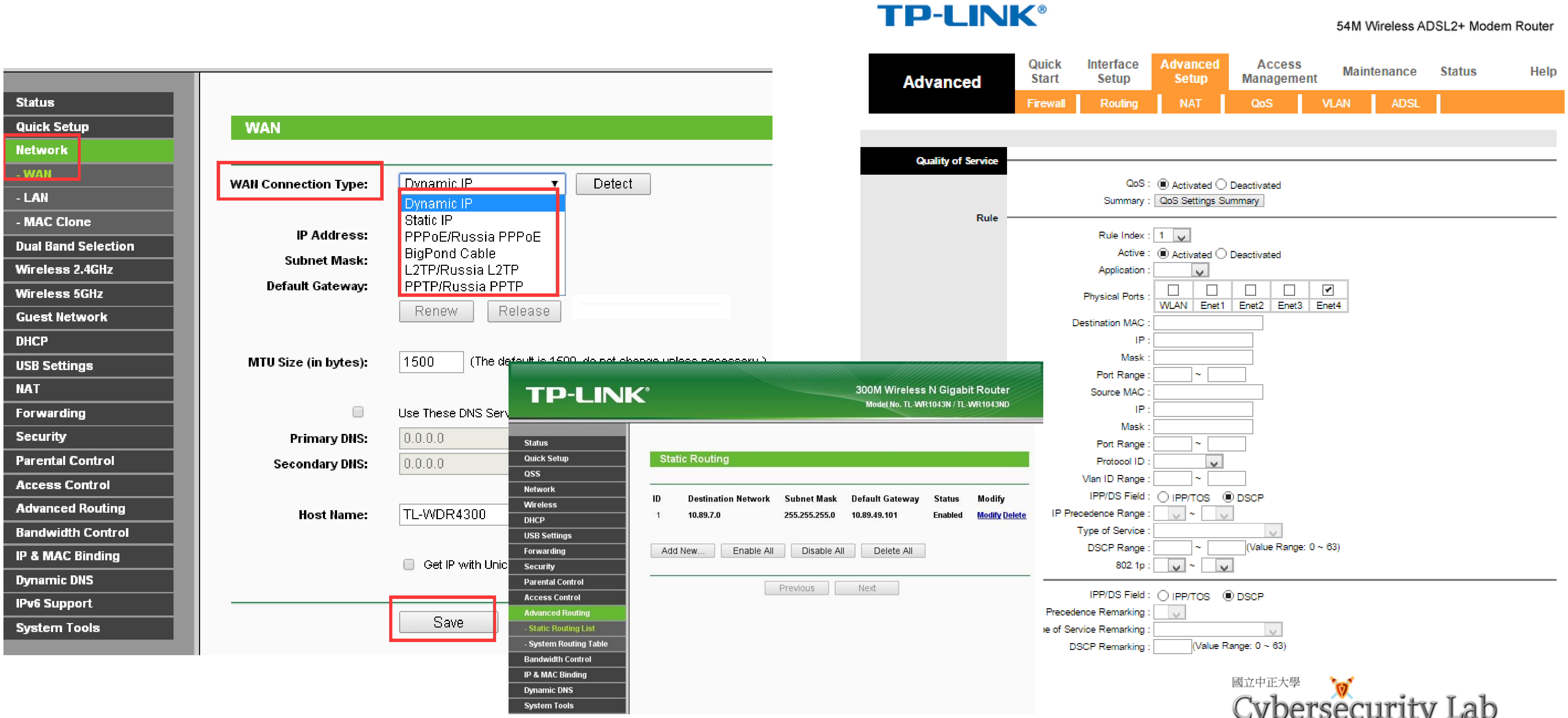
54Mbps - 1Gbps

WAN (Internet) port

LAN port
(to PC)

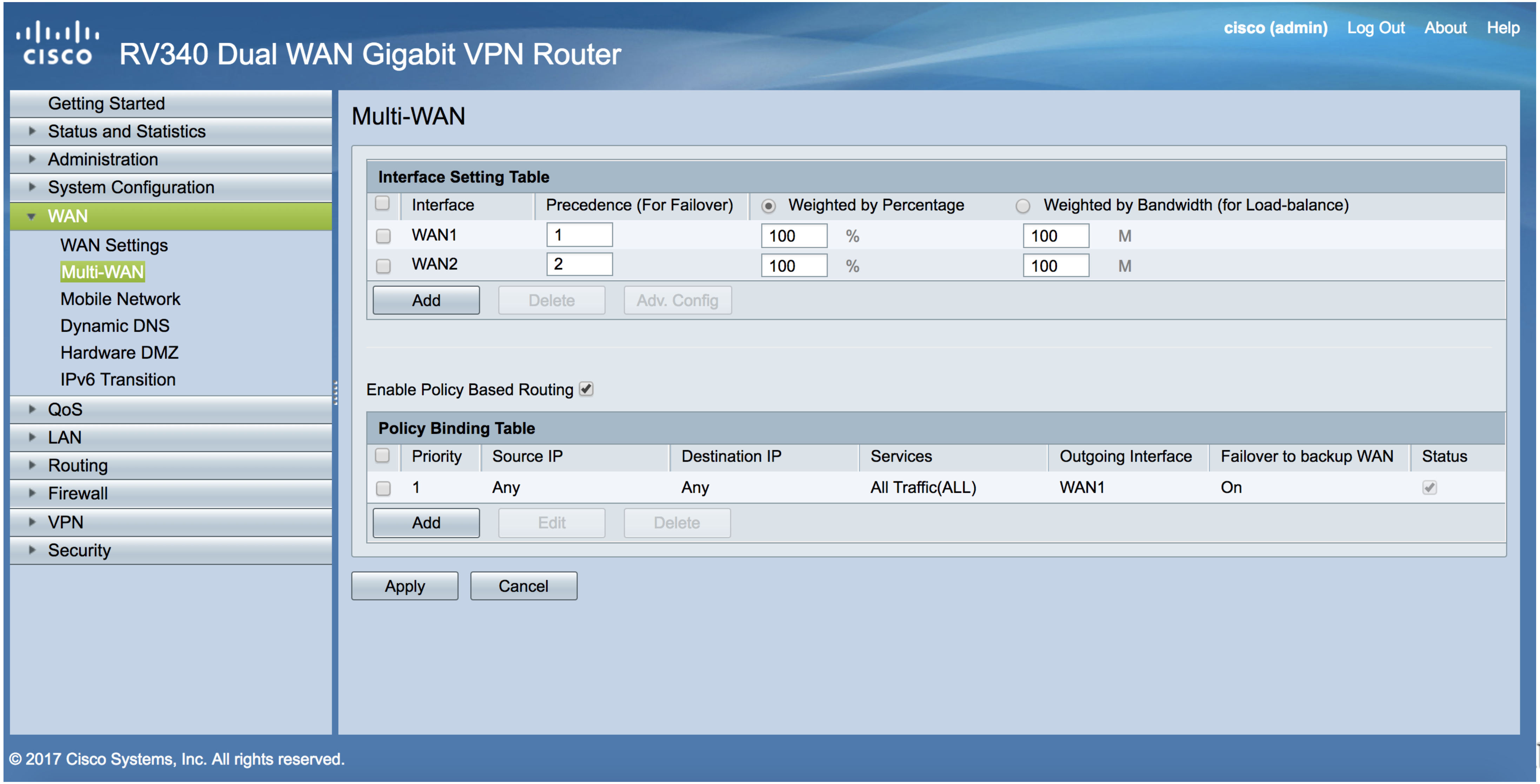


Router WAN/QoS configuration



國立中正大學 Cybersecurity Lab

Router Multi-WAN configuration



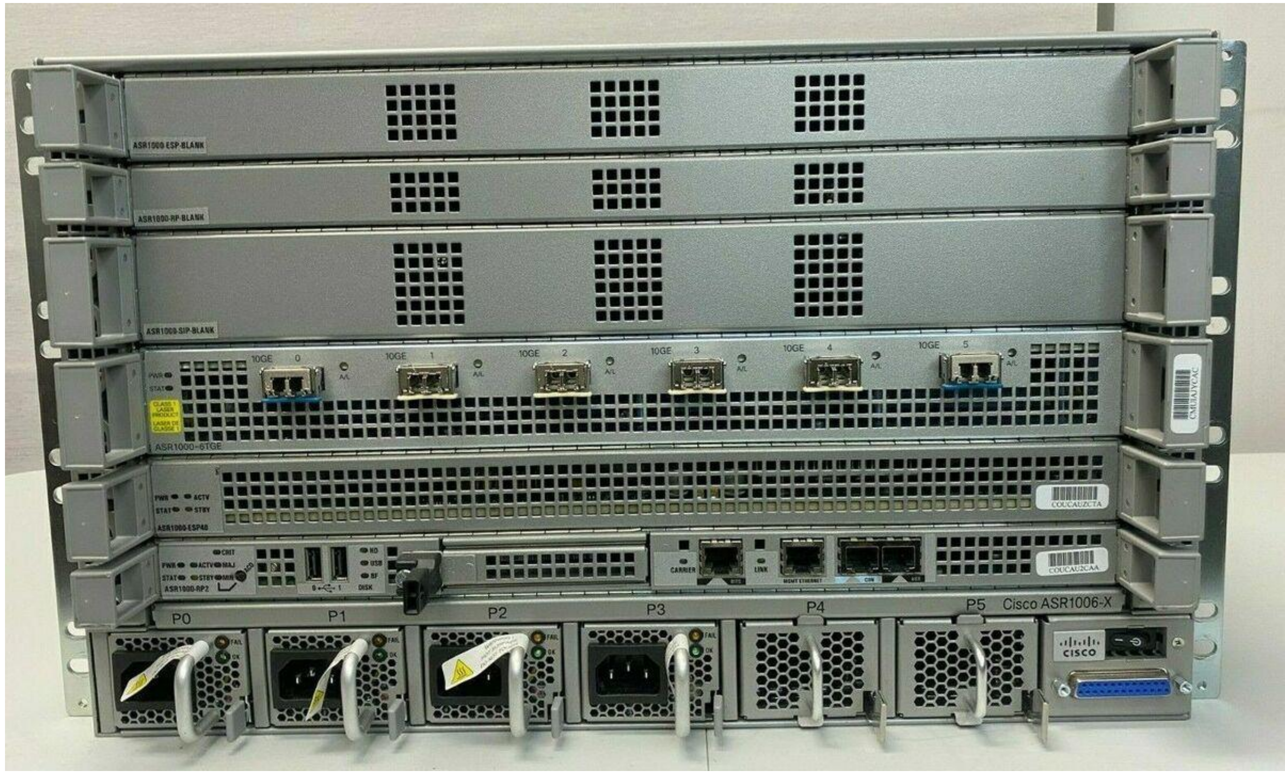
Enterprise Router

- Use to connect big networks

10Gbps

Configure via console command line

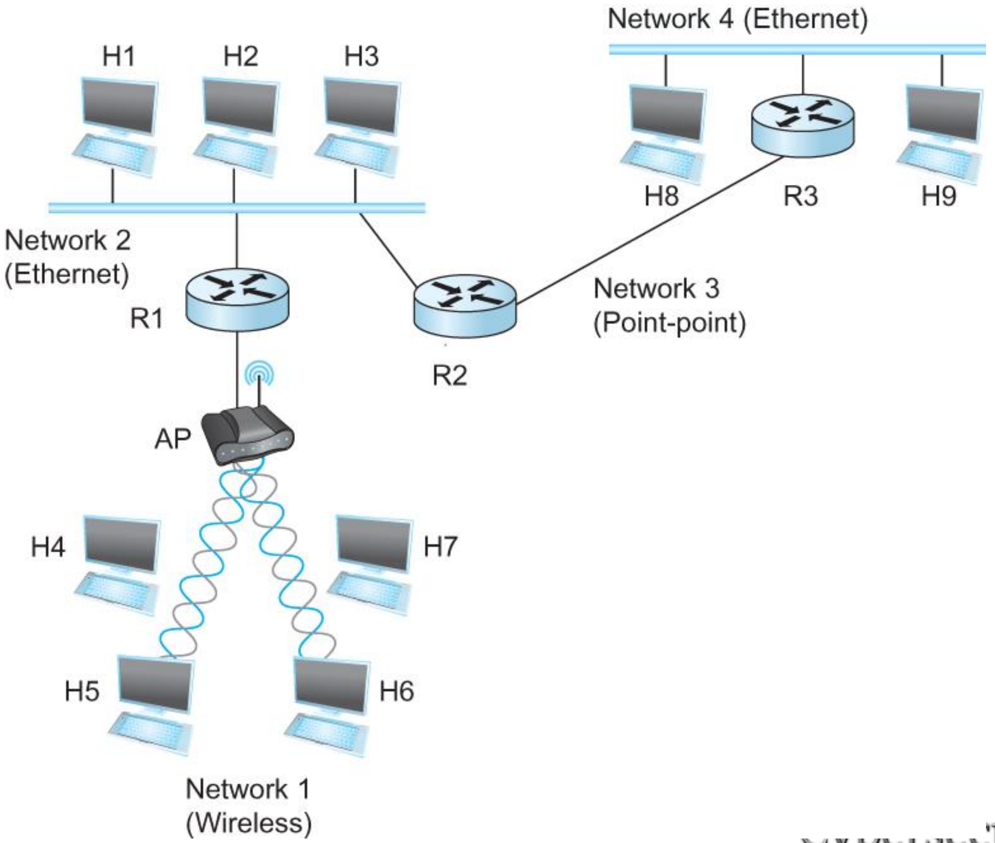
100Gbps



Internetworking

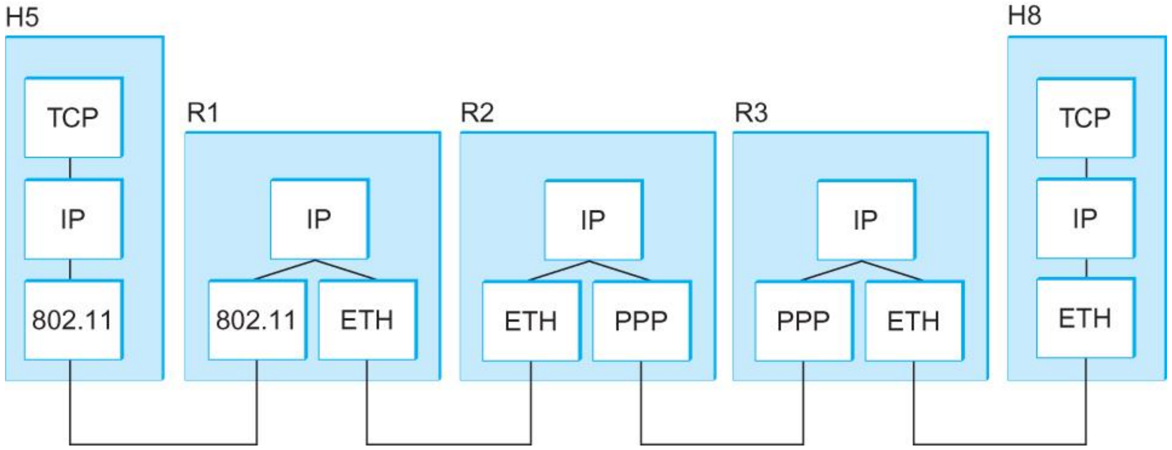
- What is internetwork
  - ✓ An arbitrary collection of networks interconnected to provide some sort of host-host to packet delivery service

A simple internetwork where H represents hosts and R represents routers



Internetworking

- What is IP
  - IP stands for Internet Protocol
  - Key tool used today to build scalable, heterogeneous internetworks
  - It runs on all the nodes in a collection of networks and defines the infrastructure that allows these nodes and networks to function as a single logical internetwork



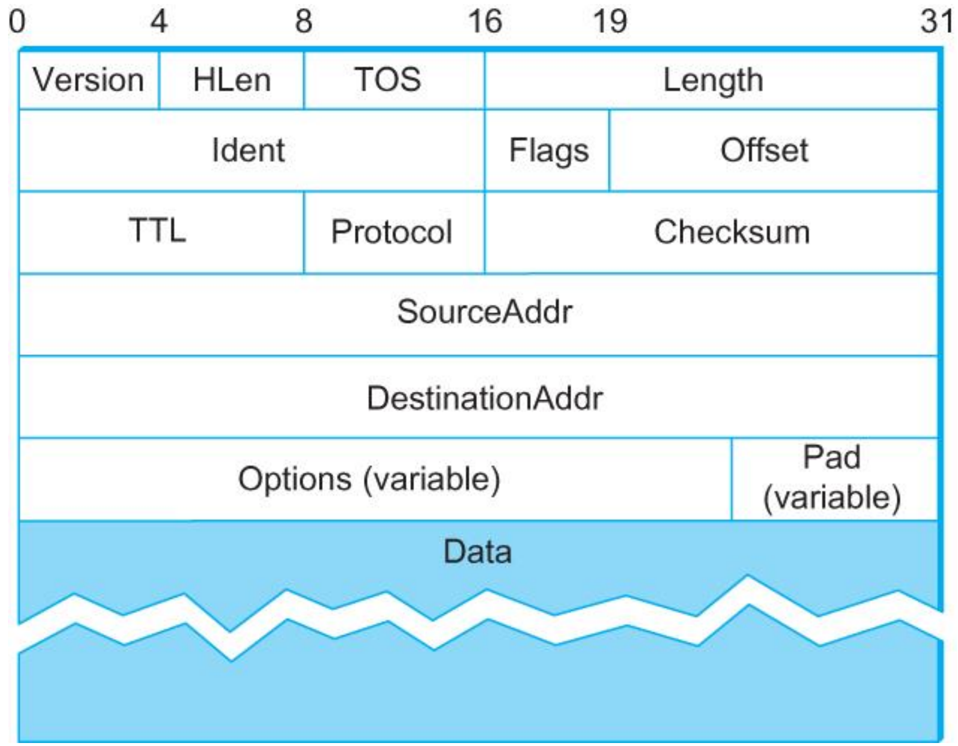
A simple internetwork showing the protocol layers

IP Service Model

- Packet Delivery Model
  - Connectionless model for data delivery
  - Best-effort delivery (unreliable service)
    - packets are lost
    - packets are delivered out of order
    - duplicate copies of a packet are delivered
    - packets can be delayed for a long time
- Global Addressing Scheme
  - Provides a way to identify all hosts in the network

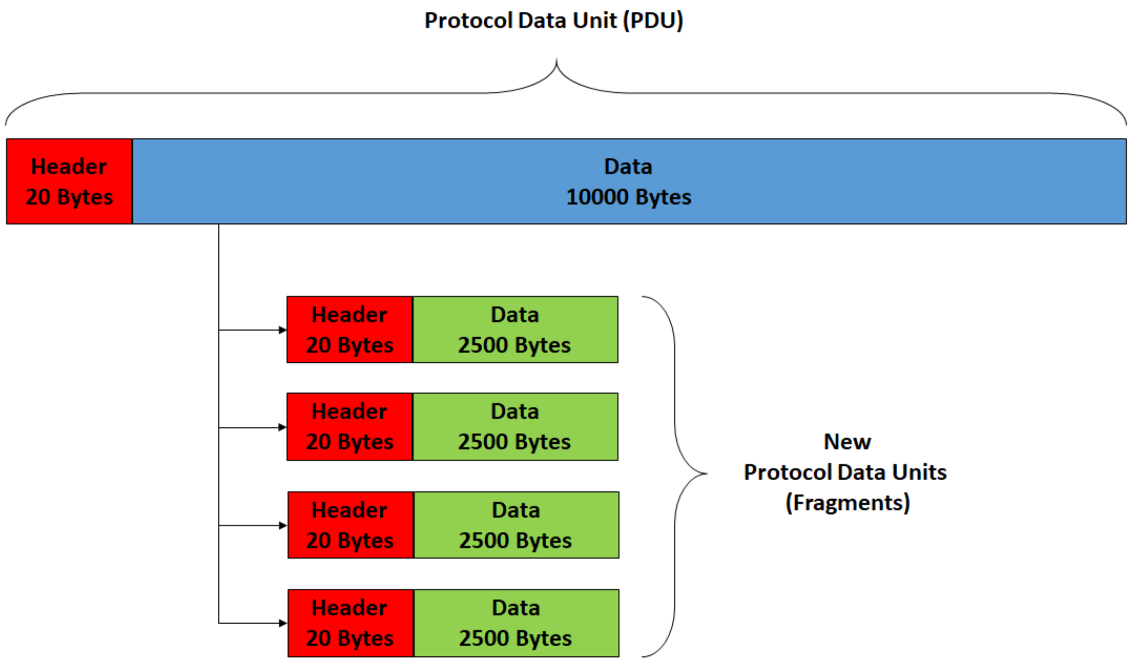
IP Packet Format

- Version (4): currently 4
- Hlen (4): number of 32-bit words in header
- TOS (8): type of service (not widely used)
- Length (16): number of bytes in this datagram
- Ident (16): used by fragmentation
- Flags/Offset (16): used by fragmentation
- TTL (8): number of hops this datagram has traveled
- Protocol (8): demux key (TCP=6, UDP=17)
- Checksum (16): of the header only
- DestAddr & SrcAddr (32)



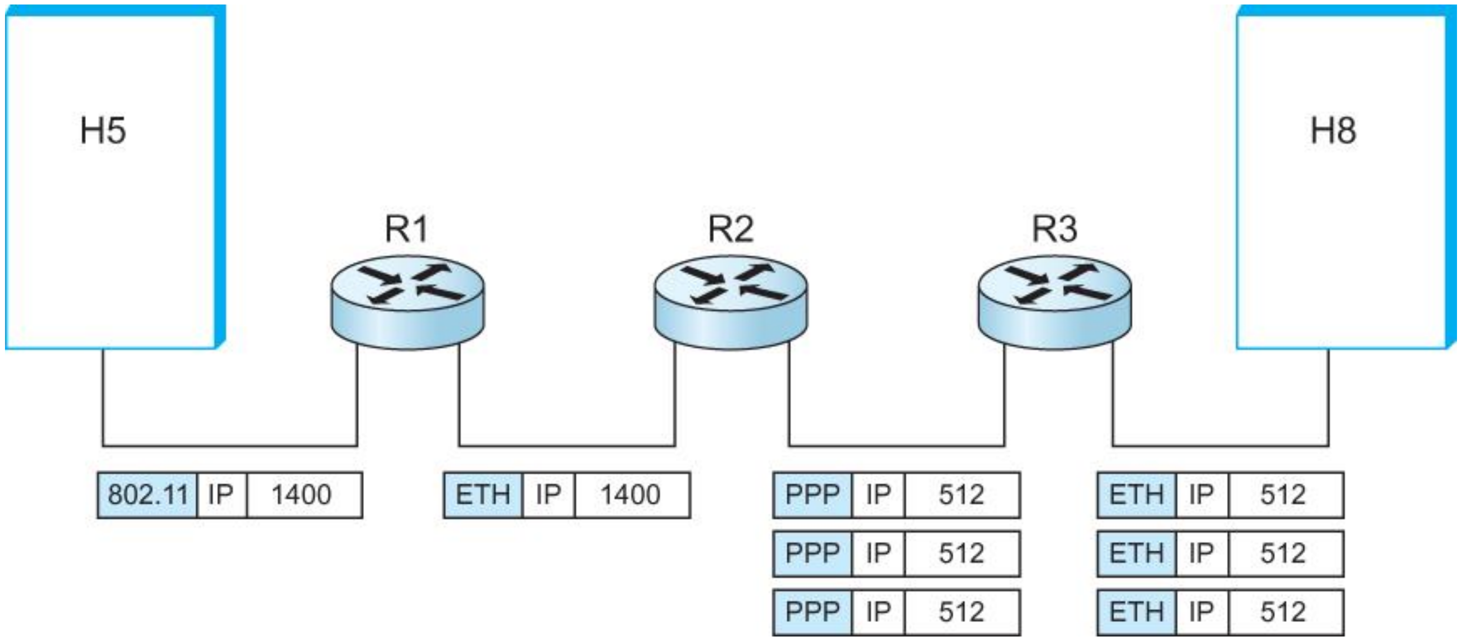
IP Fragmentation and Reassembly

- Each network has some MTU (Maximum Transmission Unit)
  - Ethernet (1500 bytes), FDDI (4500 bytes)
- Strategy
  - Fragmentation occurs in a router when it receives a datagram that it wants to forward over a network which has (MTU < datagram)
  - Reassembly is done at the receiving host
  - All the fragments carry the same identifier in the *Ident* field
  - Fragments are self-contained datagrams
  - IP does not recover from missing fragments



國立中正大學 Cybersecurity Lab

IP Fragmentation and Reassembly

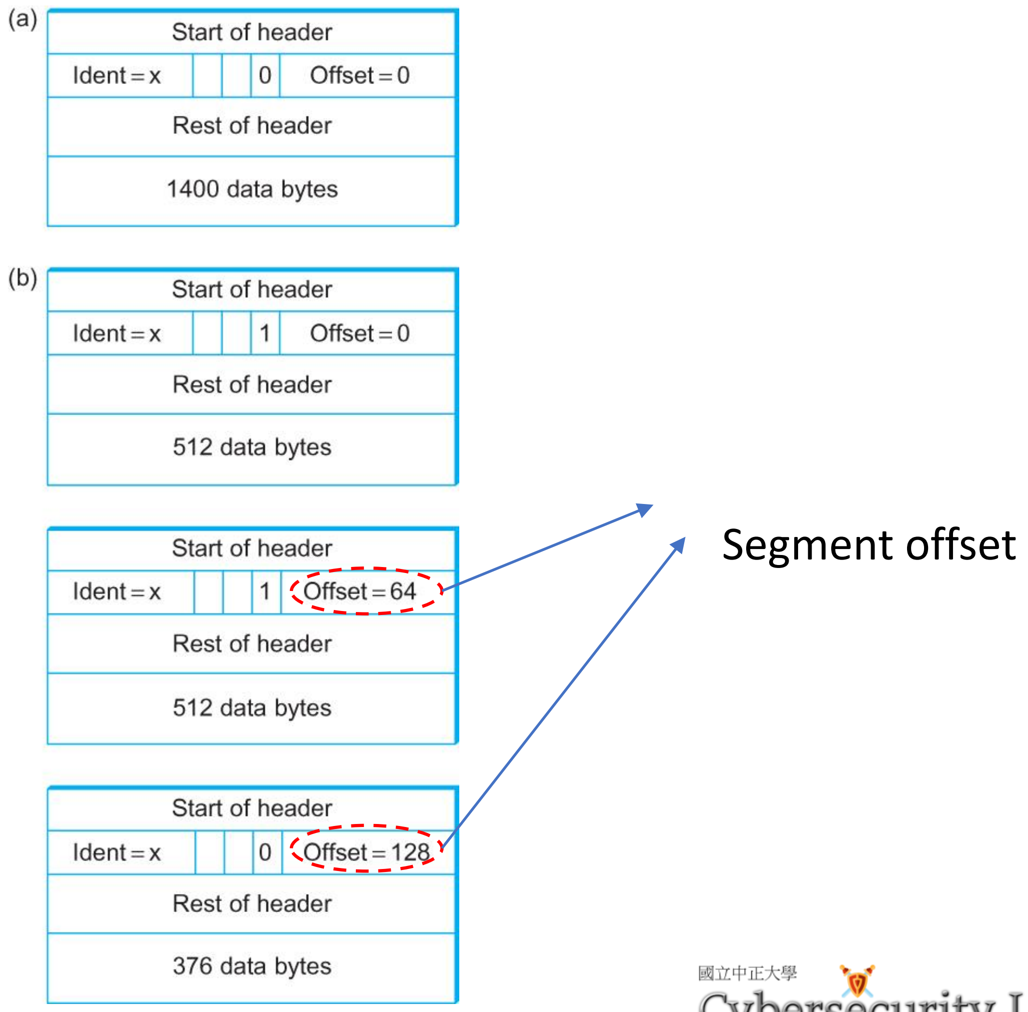


IP datagrams traversing the sequence of physical networks

IP Fragmentation and Reassembly

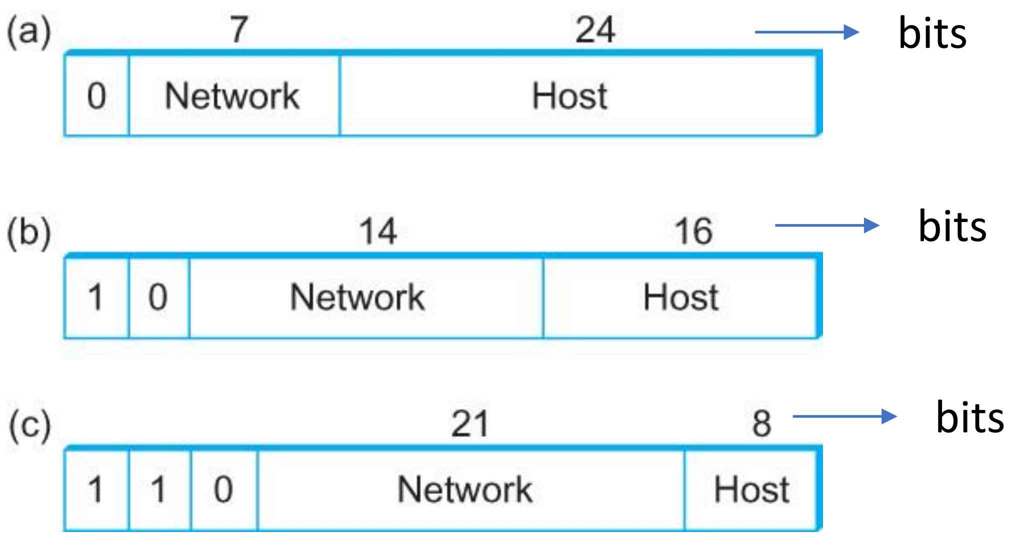
Header fields used in IP fragmentation.

(a) Unfragmented packet;
(b) Fragmented packets.



Global Addresses

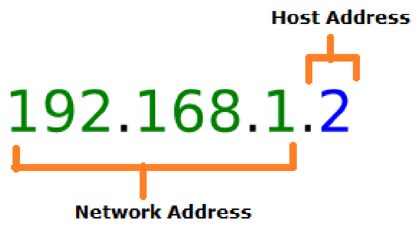
- Properties
  - globally unique
  - hierarchical: network + host
  - 4 Billion IP address, half are A type, ¼ is B type, and 1/8 is C type
- Format

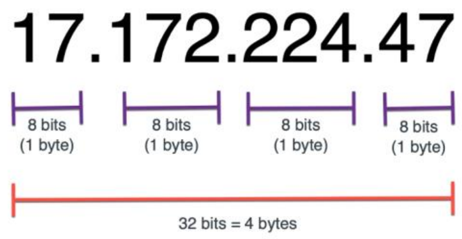


- Dot notation
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77

Internet Protocol address

- The unique identifying number assigned to every device connected to the internet





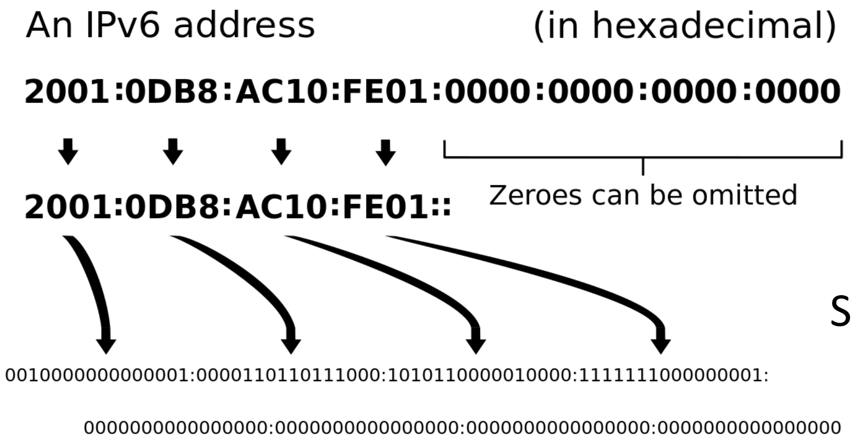
Src: Samsung

- IPv4 (32 bits): 192.168.1.2

(max 4.3 billion addresses)

- IPv6 (128 bits)

(max 7.9 x 10^{28} addresses)



Src: wikipedia

國立中正大學 Cybersecurity Lab

00000000. 00000000 . 00000000. 00000000

11111111. 00000000 . 00000000. 00000000

2^{24} ~ 16million

11111111. 11111111. 00000000. 00000000

2^{16} ~ 64k

IP classes

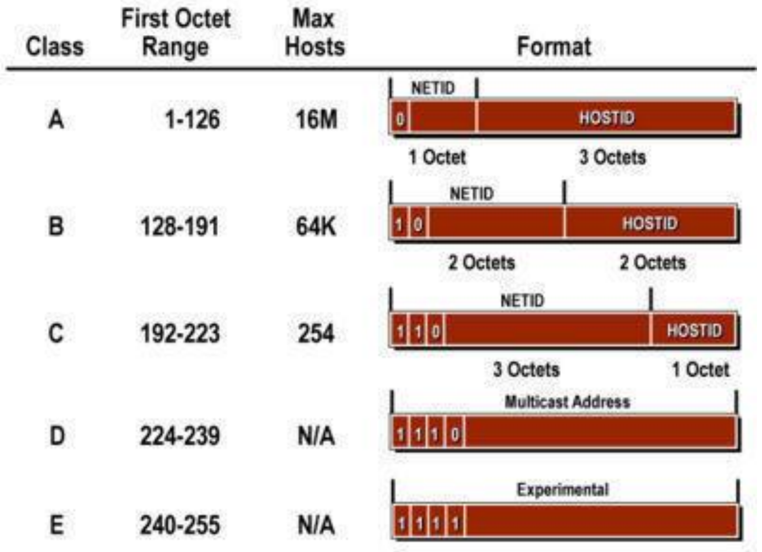
- There are five different IP classes

Since IPv4 is exceeded, we need to transfer to IPv6

- We often use Class A-C

Five Different Classes of IPv4 Addresses

Class	First Octet decimal (range)	First Octet binary (range)	IP range	Subnet Mask	Hosts per Network ID	# of networks
Class A	0 – 127	0XXXXXXX	0.0.0.0-127.255.255.255	255.0.0.0	$2^{24}-2$	2^{7}
Class B	128 – 191	10XXXXXX	128.0.0.0-191.255.255.255	255.255.0.0	$2^{16}-2$	2^{14}
Class C	192 – 223	110XXXXX	192.0.0.0-223.255.255.255	255.255.255.0	$2^{8}-2$	2^{21}
Class D (Multicast)	224 – 239	1110XXXX	224.0.0.0-239.255.255.255			
Class E (Experimental)	240 – 255	1111XXXX	240.0.0.0-255.255.255.255			



IP Datagram Forwarding

- Strategy
  - every datagram contains destination's address
  - if directly connected to destination network, then forward to host
  - if not directly connected to destination network, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table
- Example (router R2)

NetworkNum	NextHop
1	R1
2	Interface 1
3	Interface 0
4	R3

IP Datagram Forwarding

- Algorithm

```
if (NetworkNum of destination = NetworkNum of one of my interfaces) then
  deliver packet to destination over that interface
else
  if (NetworkNum of destination is in my forwarding table) then
        deliver packet to NextHop router
  else
        deliver packet to default router
```

For a host with only one interface and only a default router in its forwarding table, this simplifies to

```
if (NetworkNum of destination = my NetworkNum) then
  deliver packet to destination directly
else
  deliver packet to default router
```

IP address

- An IPv4 has 4 octets with 32 bits

140. 123. 1. 1

Octet 1 Octet 2 Octet 3 Octet 4

- An IP address has two parts: Network number and Host number

Network number

Define the network to host all PCs (i.e., township address)

Chiayi, Minhsiung, University Road

Host number

Define a specific PC address (e.g., a house address)

No 153

We can locate the house by finding this address

Chiayi, Minhsiung, University Road, No 153

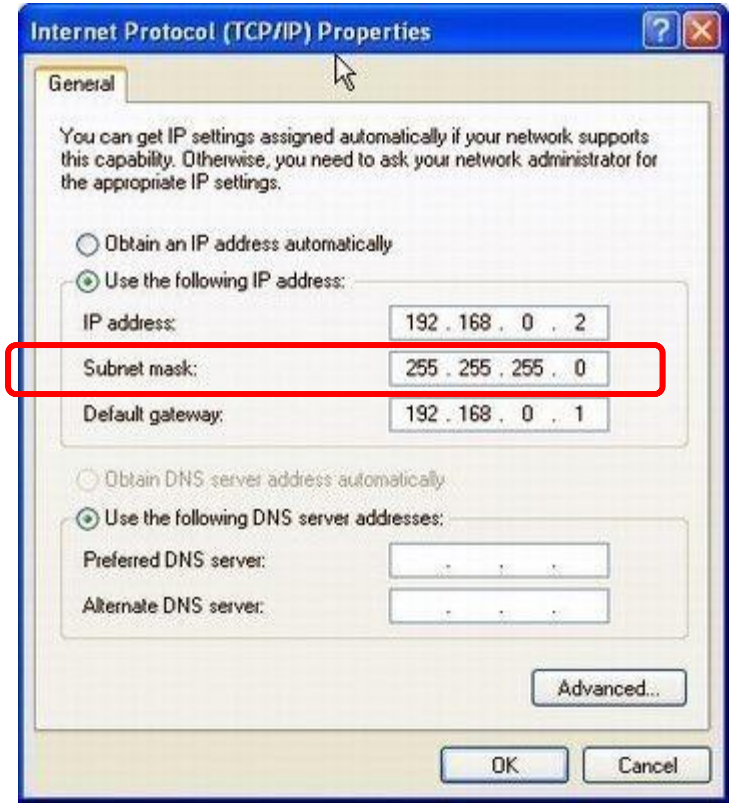
IPv4

- There are five different IP classes

Five Different Classes of IPv4 Addresses

Class	First Octet decimal (range)	First Octet binary (range)	IP range	Subnet Mask	Hosts per Network ID	# of networks
Class A	0 – 127	0XXXXXXX	0.0.0.0-127.255.255.255	255.0.0.0	$2^{24}-2$	2^{7}
Class B	128 – 191	10XXXXXX	128.0.0.0-191.255.255.255	255.255.0.0	$2^{16}-2$	2^{14}
Class C	192 – 223	110XXXXX	192.0.0.0-223.255.255.255	255.255.255.0	$2^{8}-2$	2^{21}
Class D (Multicast)	224 – 239	1110XXXX	224.0.0.0-239.255.255.255			
Class E (Experimental)	240 – 255	1111XXXX	240.0.0.0-255.255.255.255			

Subnet mask in your PC



What is a subnet mask?

- To define the network/host part

140. 123. 1. 1

Octet 1 Octet 2 Octet 3 Octet 4

- Subnet mask (default)
  - ✓ Class A: All bits of the first octet is 1 (i.e. \8, 1111 1111.0000 0000 . 0000 0000 . 0000 0000 = 255.0.0.0)
  - ✓ Class B: All bits of the first two octets is 1 (i.e., \16, 1111 1111. 1111 1111. 0000 0000 . 0000 0000 = 255.255.0.0)
  - ✓ Class C: All bits of the first three octets (i.e., \24, 1111 1111. 1111 1111. 1111 1111. 0000 0000 = 255.255.255.0)

When we type :

IP: 140. 123. 1. 1

Subnet mask: 255.255.255.0

→ Network address is 140.123.1.0

Host address is 140.123.1.1

Find the network address

- 140.123.1.3/24
- 140.123.1.128/26
- 128.123.12.11/16
- 128.123.12.11/22

Check network address and host address

- 140.123.1.3/24 → 24 Network ID | 8 Host ID

- To find the network address, you can do as follows
  - ✓ Step 1: Transfer the host octet to binary format

    140.123.1.3 → xx.xx.xx. 0000 0101
  - ✓ Step 2: Transfer the host octet of the subnet mask to binary format

    /24 ~ 255.255.255.0 → x.x.x. 0000 0000
  - ✓ Step 3: Use **AND** operand

xx.xx.xx. 0000 0101

x . X . x . 0000 0000

---

xx.xx.xx. 0000 0000

Add network ID part

Transfer to decimal

Check /24

/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2

140.123.1.0 is the network address of the host 140.123.1.3/24

Network address and host address

- 140.123.1.128/26 →

26	6
Network ID	Host ID

- To find the network address, you can do as follows
  - ✓ Step 1: Transfer the host octet to binary format

    140.123.1.128 → xx.xx.xx. 1000 0000
  - ✓ Step 2: Transfer the host octet of the subnet mask to binary format

    /26 ~ 255.255.255.192 → x.x.x.1100 0000
  - ✓ Step 3: Use AND operand

Check /26

/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2

xx.xx.xx. 1000 0000

x . x . x . 1100 0000

---

xx.xx.xx. 1000 0000

Add network ID part

Transfer to decimal

140.123.1.128 is the network address (not host address)

國立中正大學 Cybersecurity Lab

Example 3 (class B address)

- 128.123.12.11/16 →

16	16
Network ID	Host ID

- To find the network address, you can do as follows
  - ✓ Step 1: Transfer the host octet to binary format
    128.123.12.11 → xx.xx. 0000 1100. 0000 1011
  - ✓ Step 2: Transfer the host octet of the subnet mask to binary format
    /16 ~ 255.255.0.0 → x.x. 0000 0000.0000 0000
  - ✓ Step 3: Use AND operand

Check /16

/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2

xx.xx. 0000 1100. 0000 1011

x . x . 0000 0000. 0000 0000

Add network ID part

xx.xx. 0000 0000. 0000 0000

Transfer to decimal

128.123.0.0 is the network address of 128.123.12.11/16

Example 4 (class B address)

- 128.123.12.11/22 →

22	10
Network ID	Host ID

- To find the network address, you can do as follows
  - ✓ Step 1: Transfer the host octet to binary format

    128.123.12.11 → xx.xx. 0000 1100. 0000 1011
  - ✓ Step 2: Transfer the subnet mask to binary format

    /22 ~ 255.255.252.0 → x.x. 11111100. 0000 0000
  - ✓ Step 3: Use AND operand

xx.xx. 0000 1100. 0000 1011

x . x . 1111 1100. 0000 0000

Add network ID part

xx.xx. 0000 1100. 0000 0000

Transfer to decimal

Check /22

/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2

128.123.12.0 is the network address of 128.123.12.11/22

國立中正大學 Cybersecurity Lab

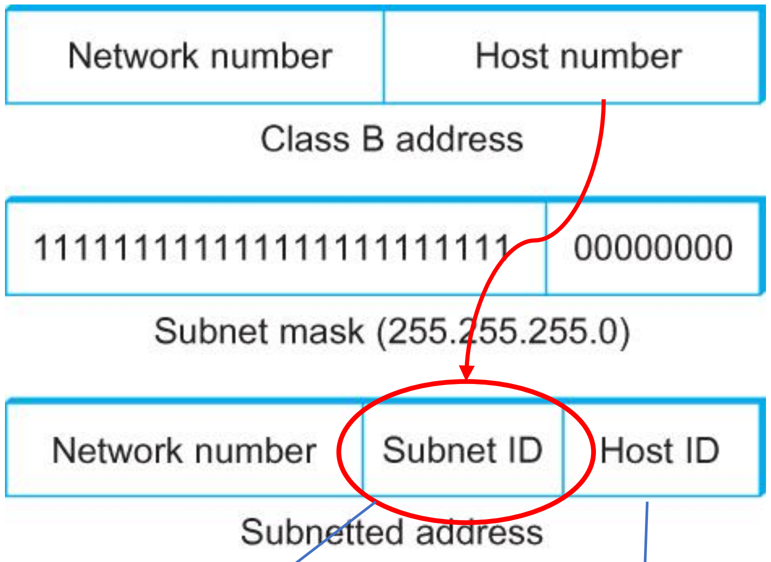
The new requirement

- ISP gives me 140.123. 101.0/24
- However, I wants to split this address into several networks (one for Department of Finance, one for Department of Computer Science, ….)
- Computers in those split networks cannot access to the each other
- I don’t need to request new IPs from ISP → We use subnetting

Subnetting

- We can split the host number into two parts: Subnet ID and Host ID

✓ Transfer binary bits of the network number + subnet number to 1
✓ Transfer binary bits of the host part to 0



For example, 140.123. 101.0/24

- Network number: 24 bits
- Host number: 8 bits → Split host number into 2 bits for subnet ID
  6 bits for host ID

We transfer the network number + subnet number = 24+2 = 26 bits to 1
The remaining bits for host number (32 – 26 = 6) is transferred to 0

Remaining bits for host number

1111 1111. 1111 1111. 1111 1111. 1100 0000

In decimal: 255. 255. 255. 192

Decide how many networks you want to split

Define the max number of hosts per subnetted network

An example to find the subnet mask

- From this IP 140.123.1.0, please split into four networks. Each subnet has a maximum 62 hosts

Five Different Classes of IPv4 Addresses

Class	First Octet decimal (range)	First Octet binary (range)	IP range	Subnet Mask	Hosts per Network ID	# of networks
Class A	0 – 127	0XXXXXXX	0.0.0.0-127.255.255.255	255.0.0.0	$2^{24}-2$	2^{7}
Class B	128 – 191	10XXXXXX	128.0.0.0-191.255.255.255	255.255.0.0	$2^{16}-2$	2^{14}
Class C	192 – 223	110XXXXX	192.0.0.0-223.255.255.255	255.255.255.0	$2^{8}-2$	2^{21}
Class D (Multicast)	224 – 239	1110XXXX	224.0.0.0-239.255.255.255			
Class E (Experimental)	240 – 255	1111XXXX	240.0.0.0-255.255.255.255			

- Solution:
  - ✓ 140.123.1.0 is a class B address
  - ✓ Default subnet mask is \24
  - ✓ The host part is 32-24 = 8 bits .
  - ✓ To split the network into four networks,

  we need to request at least 2 bits from the host number, since 2^2 = 4.

  So total bits for the network part is 24 + 2 = 26
  - ✓ Now, transfer bits of the network part into 1 (i.e., 26 bits), the remaining is 0:

    1111 1111. 1111 1111. 1111 1111. 1100 0000.

26 bits

6 bits

Total hosts supported = 2^6 - 2 = 62

Don't count the first address (network address) and the last address (broadcast address)

An example (cont.)

- Transfer the binary value into decimal

1111 1111. 1111 1111. 1111 1111. 1100 0000.

255. 255. 255. 192

$2^7 + 2^6 = 128+64$
$= 192$

- The final subnet mask is $\boxed{255.\ 255.\ 255.192}$

Check the final subnetted networks

- To find the subnetted networks, please select different options of the bits you extracted for the subnet ID from Host number

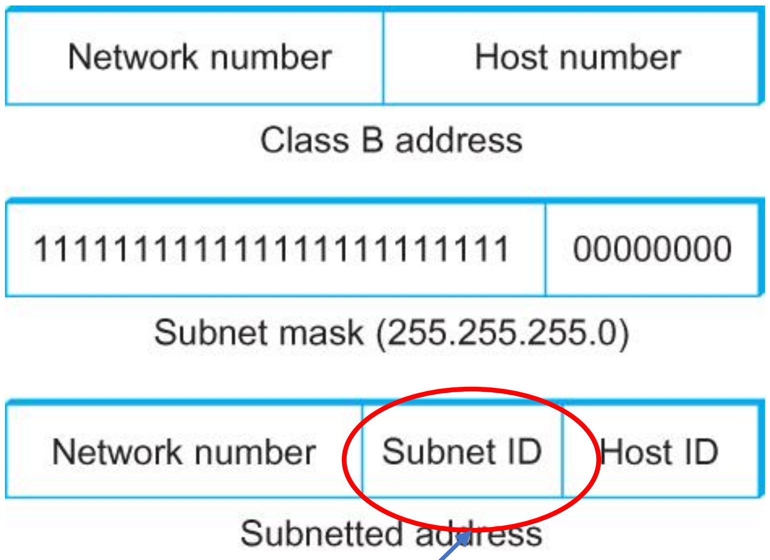
Original IP 140.123.1.0

xx.xx.xx. 0000 0000 → 140.123.1.0

xx.xx.xx. 0100 0000 → 140.123.1.64

xx.xx.xx. 1000 0000 → 140.123.1.128

xx.xx.xx. 1100 0000 → 140.123.1.192



Check all options of binary combination and convert to decimal

The final results

- We have four networks

Network IP	Possible host IP	Subnet mask
140.123.1.0	140.123.1.1~140.123.1.63	255.255.255.192
140.123.1.64	140.123.1.65~140.123.1.127	
140.123.1.128	140.123.1.129~140.123.1.191	
140.123.1.192	140.123.1.193~140.123.1.254	

The faster method to find the subnet mask

- Check this table

Network Bits	Subnet mask	Number of subnets	Number of Hosts
/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2

Step 1: Check number of hosts in the requirement

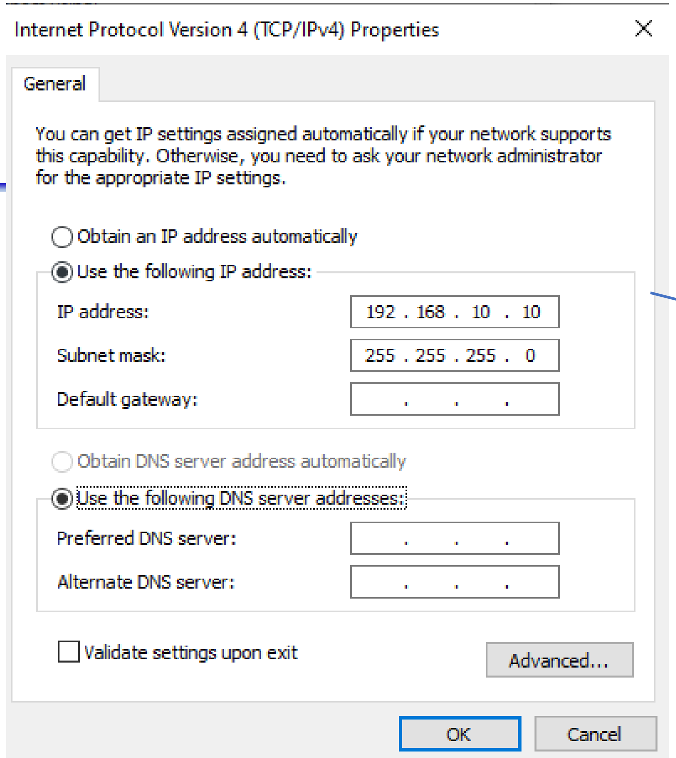
Step 2: Determine the corresponding subnet mask

國立中正大學 Cybersecurity Lab

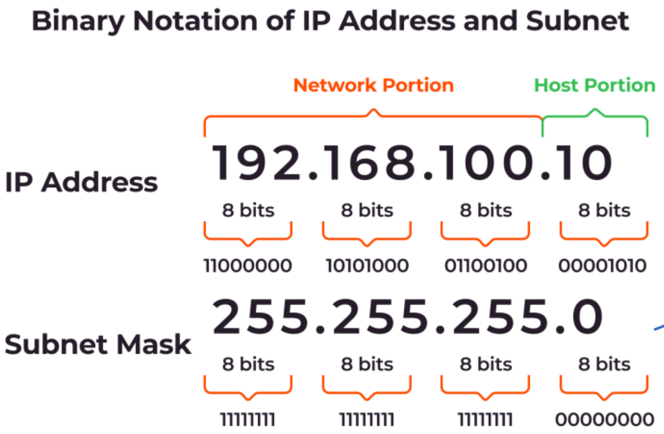
Common subnet masks

- Some famous subnet masks in class B

Network Bits	Subnet mask	Number of subnets	Number of Hosts
/16	255.255.0.0	256 (254)	65534
/17	255.255.128.0	512 (510)	32766
/18	255.255.192.0	1024 (1022)	16382
/19	255.255.224.0	2048 (2046)	8190
/20	255.255.240.0	4096 (4094)	4094
/21	255.255.248.0	8192 (8190)	2046
/22	255.255.252.0	16384 (16382)	1022
/23	255.255.254.0	32768 (32766)	510
/24	255.255.255.0	65536 (65534)	254
/25	255.255.255.128	131072 (131070)	126
/26	255.255.255.192	262144 (262142)	62
/27	255.255.255.224	524288 (524286)	30
/28	255.255.255.240	1048576 (1048574)	14
/29	255.255.255.248	2097152 (2097150)	6
/30	255.255.255.252	4194304 (4194302)	2



Our router address



/24

Subnetting

Forwarding Algorithm

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop>
   D1 = SubnetMask & D
   if D1 = SubnetNum
      if NextHop is an interface
         deliver datagram directly to destination
      else
         deliver datagram to NextHop (a router)
```

Subnetting

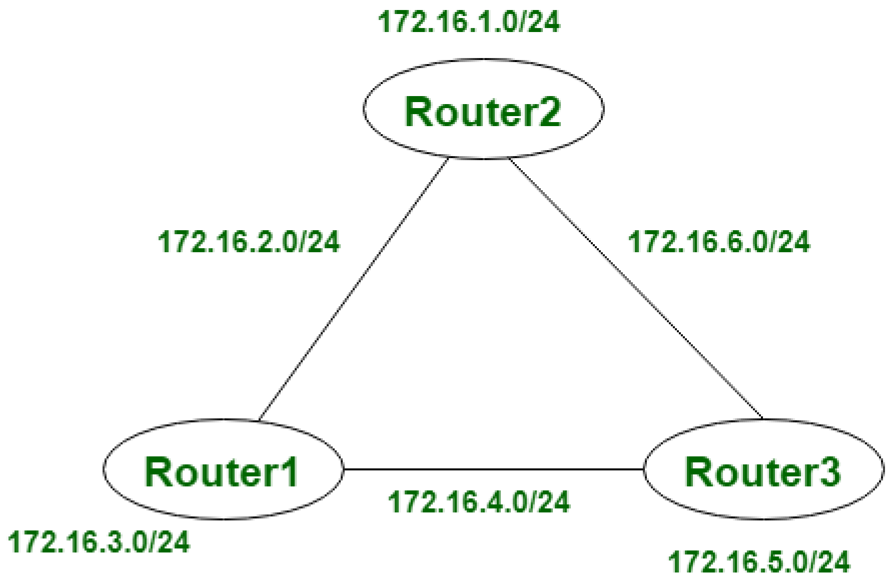
Notes

- Would use a default router if nothing matches
- Not necessary for all ones in subnet mask to be contiguous
- Can put multiple subnets on one physical network
- Subnets not visible from the rest of the Internet

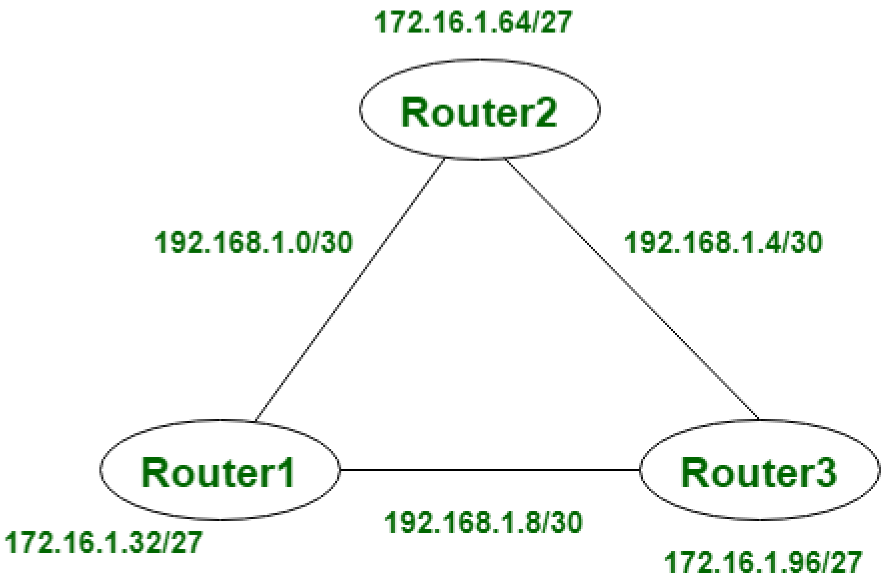
Classless Addressing

- Classless Inter-Domain Routing (CIDR)
  - A technique that addresses two scaling concerns in the Internet
    - The growth of backbone routing table as more and more network numbers need to be stored in them
    - Potential exhaustion of the 32-bit address space
  - Address assignment efficiency
    - Arises because of the IP address structure with class A, B, and C addresses
    - Forces us to hand out network address space in fixed-size chunks of three very different sizes
      - A network with two hosts needs a class C address
        - Address assignment efficiency = 2/255 = 0.78
      - A network with 256 hosts needs a class B address
        - Address assignment efficiency = 256/65535 = 0.39

Classless Addressing



Classful: Subnet mask is same throughout the topology



Classless: Subnet mask can change in the topology

Classless Addressing

- Exhaustion of IP address space centers on exhaustion of the class B network numbers
- Solution
  - Say “NO” to any Autonomous System (AS) that requests a class B address unless they can show a need for something close to 64K addresses
  - Instead give them an appropriate number of class C addresses
  - For any AS with at least 256 hosts, we can guarantee an address space utilization of at least 50%
- What is the problem with this solution?

Classless Addressing

- Problem with this solution
  - Excessive storage requirement at the routers.
- If a single AS has, say 16 class C network numbers assigned to it,
  - Every Internet backbone router needs 16 entries in its routing tables for that AS
  - This is true, even if the path to every one of these networks is the same
- If we had assigned a class B address to the AS
  - The same routing information can be stored in one entry
  - Efficiency = 16 × 255 / 65, 536 = 6.2%

Classless Addressing

- CIDR tries to balance the desire to minimize the number of routes that a router needs to know against the need to hand out addresses efficiently.

- CIDR uses aggregate routes
  - Uses a single entry in the forwarding table to tell the router how to reach a lot of different networks
  - Breaks the rigid boundaries between address classes

Classless Addressing

- Consider an AS with 16 class C network numbers.
- Instead of handing out 16 addresses at random, hand out a block of contiguous class C addresses
- Suppose we assign the class C network numbers from 192.4.16 through 192.4.31
- Observe that top 20 bits of all the addresses in this range are the same (11000000 00000100 0001)
  - We have created a 20-bit network number (which is in between class B network number and class C number)
- Requires to hand out blocks of class C addresses that share a common prefix

Classless Addressing

- Requires to hand out blocks of class C addresses that share a common prefix
- The convention is to place a /X after the prefix where X is the prefix length in bits
- For example, the 20-bit prefix for all the networks 192.4.16 through 192.4.31 is represented as 192.4.16/20

- By contrast, if we wanted to represent a single class C network number, which is 24 bits long, we would write it 192.4.16/24

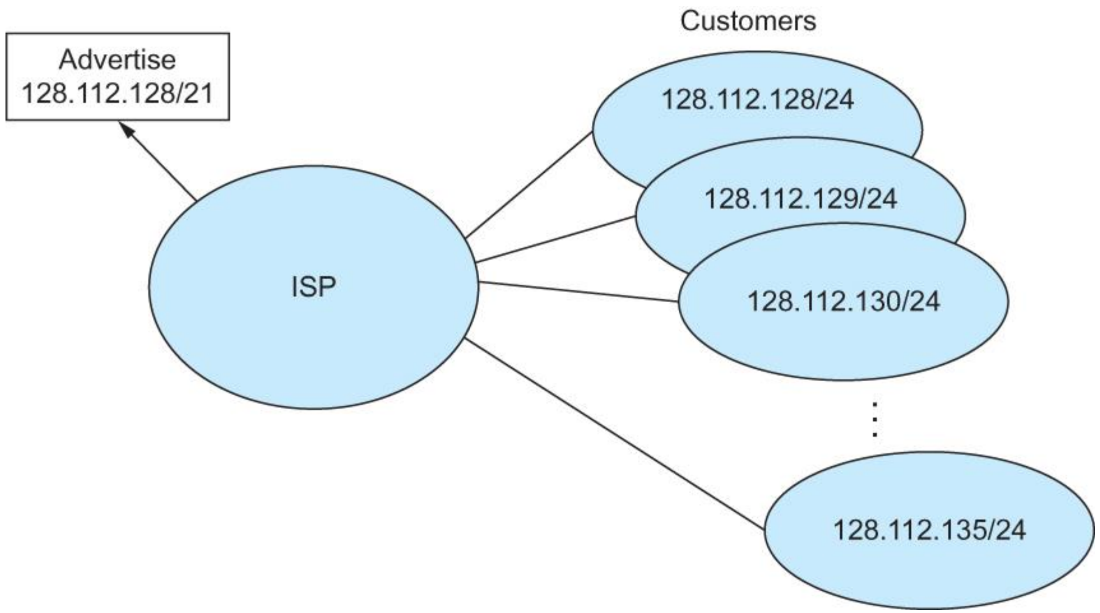
Classless Addressing

- How do the routing protocols handle this classless addresses
  - It must understand that the network number may be of any length
- Represent network number with a single pair

  `<length, value>`

- All routers must understand CIDR addressing

Classless Addressing



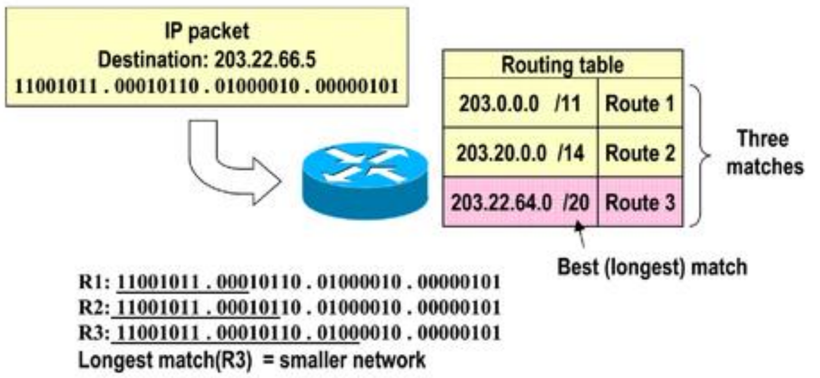
Route aggregation with CIDR

IP Forwarding Revisited

- IP forwarding mechanism assumes that it can find the network number in a packet and then look up that number in the forwarding table

- We need to change this assumption in case of CIDR

- CIDR means that prefixes may be of any length, from 2 to 32 bits

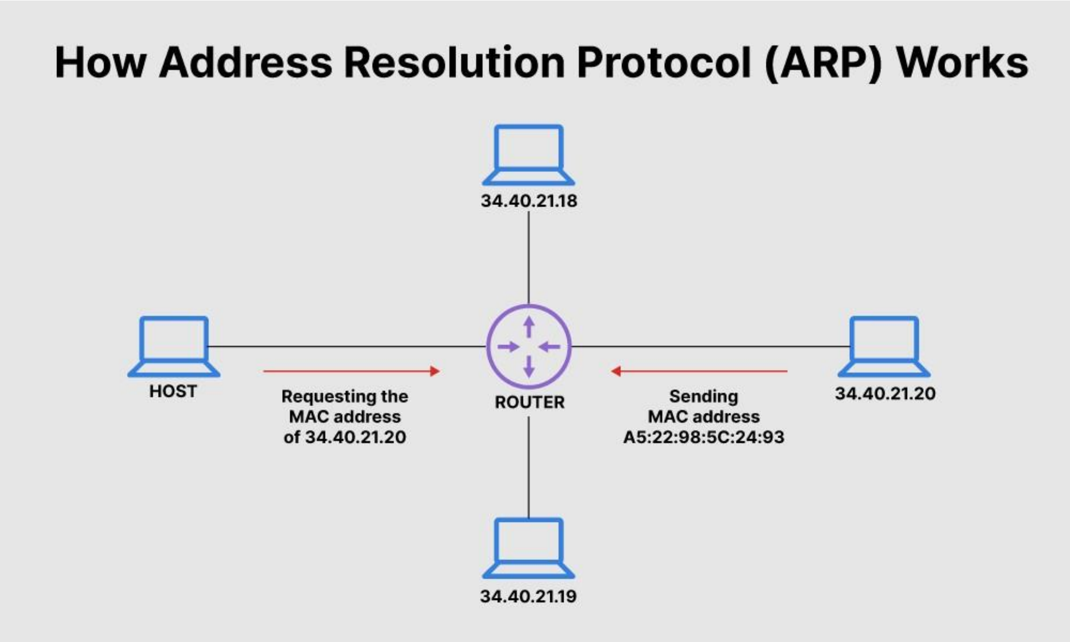


IP Forwarding Revisited

- It is also possible to have prefixes in the forwarding tables that overlap
  - Some addresses may match more than one prefix
- For example, we might find both 171.69 (a 16 bit prefix) and 171.69.10 (a 24 bit prefix) in the forwarding table of a single router
- A packet destined to 171.69.10.5 clearly matches both prefixes.
  - The rule is based on the principle of “longest match”
    - 171.69.10 in this case
- A packet destined to 171.69.20.5 would match 171.69 and not 171.69.10

Address Translation Protocol (ARP)

- Map IP addresses into physical addresses
  - destination host
  - next hop router
- Techniques
  - encode physical address in host part of IP address
  - table-based
- ARP (Address Resolution Protocol)
  - table of IP to physical address bindings
  - broadcast request if IP address not in table
  - target machine responds with its physical address
  - table entries are discarded if not refreshed



ARP Packet Format

<table>
<tr><td>0</td><td>8</td><td>16</td><td>31</td></tr>
<tr><td colspan="2">Hardware type = 1</td><td colspan="2">ProtocolType = 0x0800</td></tr>
<tr><td>HLen = 48</td><td>PLen = 32</td><td colspan="2">Operation</td></tr>
<tr><td colspan="4">SourceHardwareAddr (bytes 0–3)</td></tr>
<tr><td colspan="2">SourceHardwareAddr (bytes 4–5)</td><td colspan="2">SourceProtocolAddr (bytes 0–1)</td></tr>
<tr><td colspan="2">SourceProtocolAddr (bytes 2–3)</td><td colspan="2">TargetHardwareAddr (bytes 0–1)</td></tr>
<tr><td colspan="4">TargetHardwareAddr (bytes 2–5)</td></tr>
<tr><td colspan="4">TargetProtocolAddr (bytes 0–3)</td></tr>
</table>

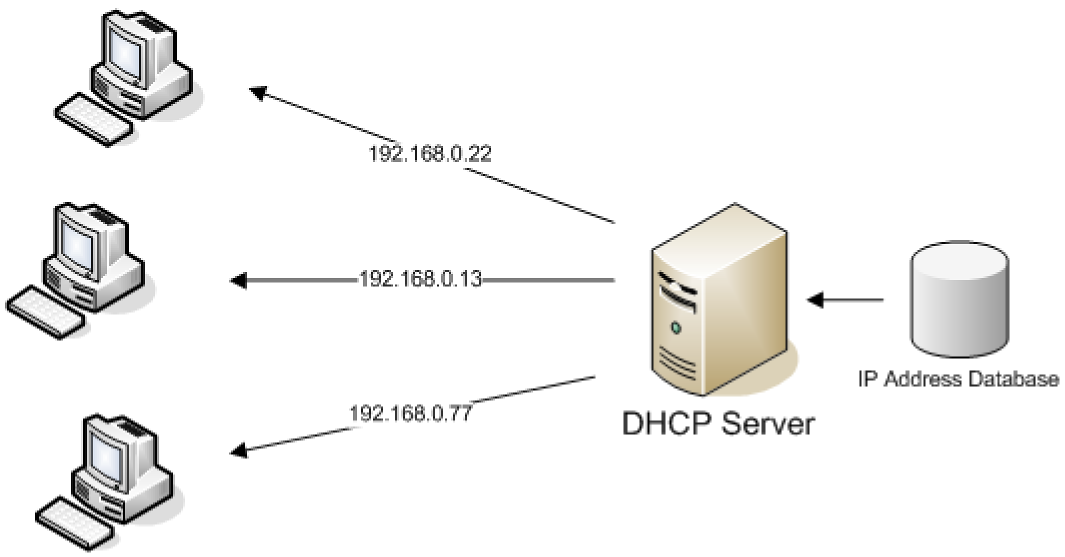
- HardwareType: type of physical network (e.g., Ethernet)
- ProtocolType: type of higher layer protocol (e.g., IP)
- HLEN & PLEN: length of physical and protocol addresses
- Operation: request or response
- Source/Target Physical/Protocol addresses

Host Configurations

- Notes
  - Ethernet addresses are configured into network by manufacturer and they are unique
  - IP addresses must be unique on a given internetwork but also must reflect the structure of the internetwork
  - Most host Operating Systems provide a way to manually configure the IP information for the host
  - Drawbacks of manual configuration
    - A lot of work to configure all the hosts in a large network
    - Configuration process is error-prune
  - Automated Configuration Process is required

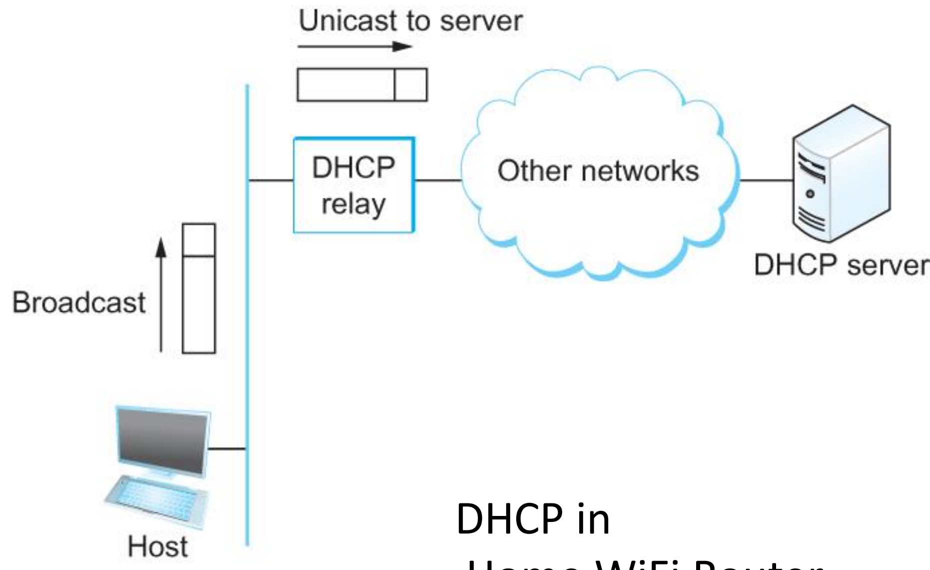
Dynamic Host Configuration Protocol (DHCP)

- DHCP server is responsible for providing configuration information to hosts
- There is at least one DHCP server for an administrative domain
- DHCP server maintains a pool of available addresses



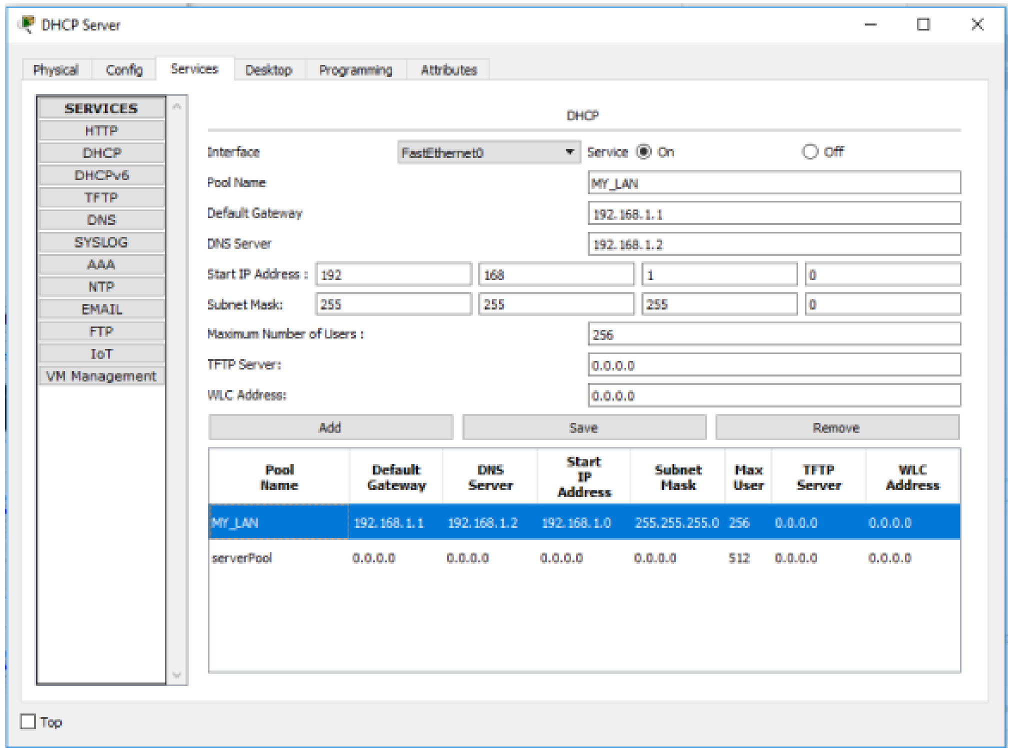
DHCP

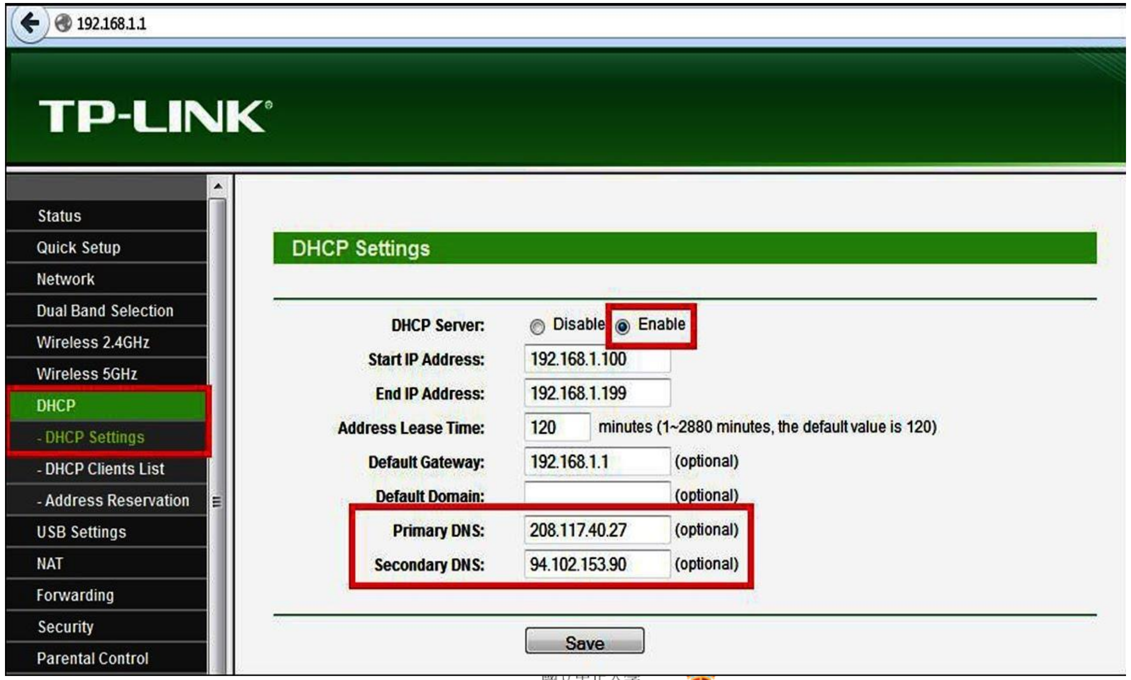
- Newly booted or attached host sends DHCPDISCOVER message to a special IP address (255.255.255.255)
- DHCP relay agent unicasts the message to DHCP server and waits for the response



DHCP in Home WiFi Router

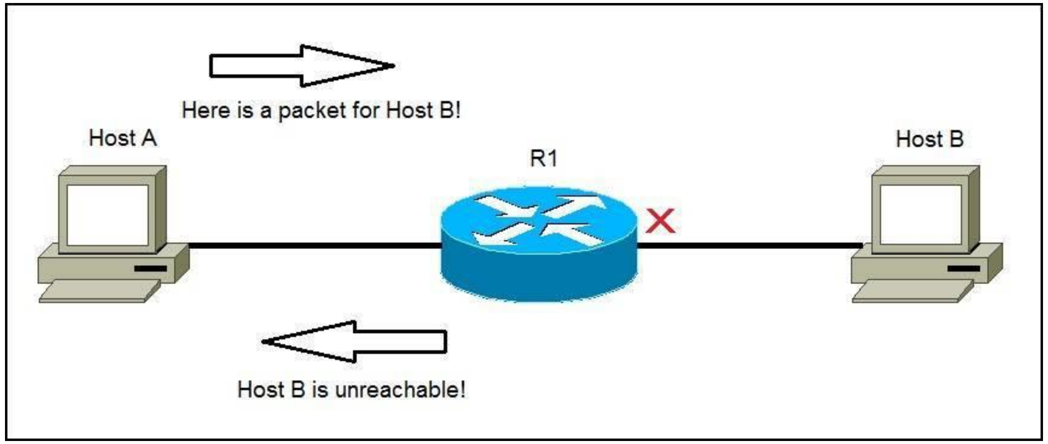
DHCP in Packet Tracer





Internet Control Message Protocol (ICMP)

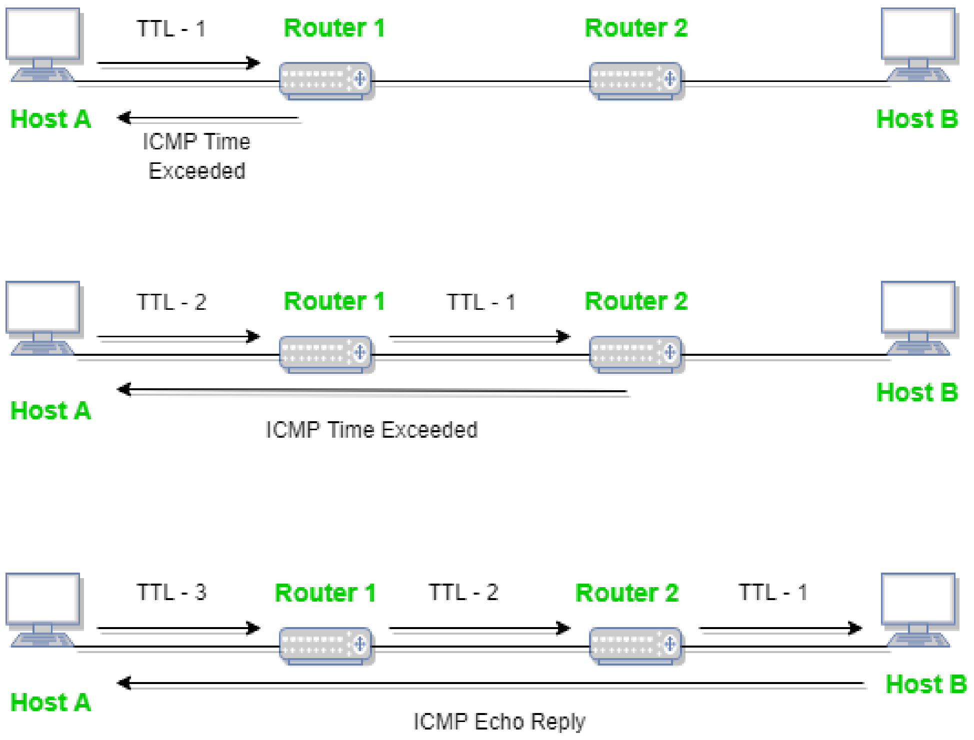
- Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
  - Destination host unreachable due to link /node failure
  - Reassembly process failed
  - TTL had reached 0 (so datagrams don't cycle forever)
  - IP header checksum failed
- ICMP-Redirect
  - From router to a source host
  - With a better route information



Internet Control Message Protocol (ICMP)

- Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
  - Destination host unreachable due to link /node failure
  - Reassembly process failed
  - TTL had reached 0 (so datagrams don't cycle forever)
  - IP header checksum failed

- ICMP-Redirect
  - From router to a source host
  - With a better route information



Routing

Forwarding versus Routing

- Forwarding:
  - to select an output port based on destination address and routing table
- Routing:
  - process by which routing table is built

- Forwarding table VS Routing table
  - Forwarding table
    - Used when a packet is being forwarded and so must contain enough information to accomplish the forwarding function
    - A row in the forwarding table contains the mapping from a network number to an outgoing interface and some MAC information, such as Ethernet Address of the next hop
  - Routing table
    - Built by the routing algorithm as a precursor to build the forwarding table
    - Generally contains mapping from network numbers to next hops

Routing

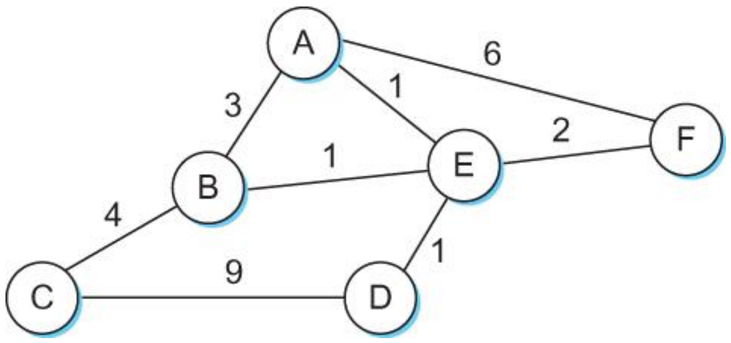
(a)	
Prefix/Length	**Next Hop**
18/8	171.69.245.10

(b)		
Prefix/Length	**Interface**	**MAC Address**
18/8	if0	8:0:2b:e4:b:1:2

Example rows from (a) routing and (b) forwarding tables

Routing

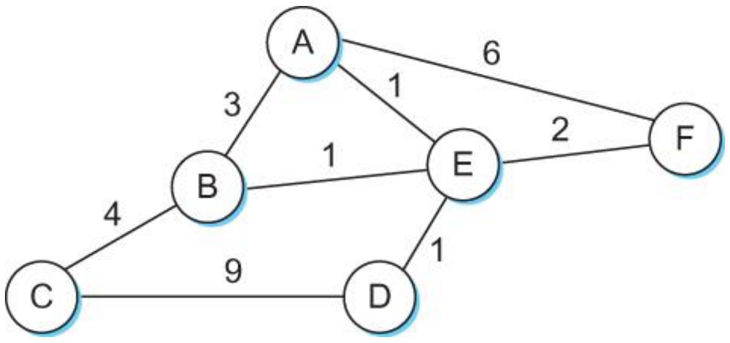
- Network as a Graph



- The basic problem of routing is to find the lowest-cost path between any two nodes
  - Where the cost of a path equals the sum of the costs of all the edges that make up the path

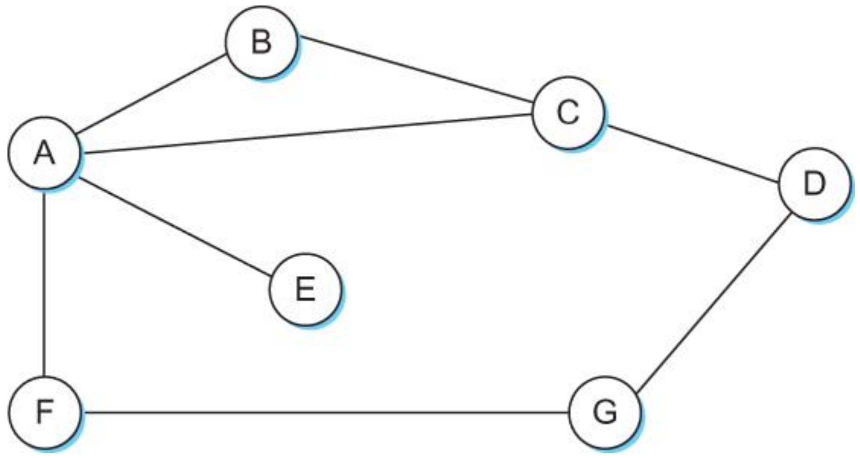
Routing

- For a simple network, we can calculate all shortest paths and load them into some nonvolatile storage on each node.
- Such a static approach has several shortcomings
  - It does not deal with node or link failures
  - It does not consider the addition of new nodes or links
  - It implies that edge costs cannot change
- What is the solution?
  - Need a distributed and dynamic protocol
  - Two main classes of protocols
    - Distance Vector
    - Link State



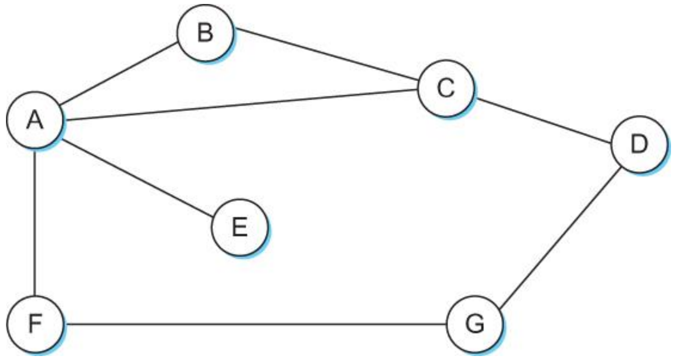
Distance Vector

- Each node constructs a one dimensional array (a vector) containing the “distances” (costs) to all other nodes and distributes that vector to its immediate neighbors
- Starting assumption is that each node knows the cost of the link to each of its directly connected neighbors



Distance Vector

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

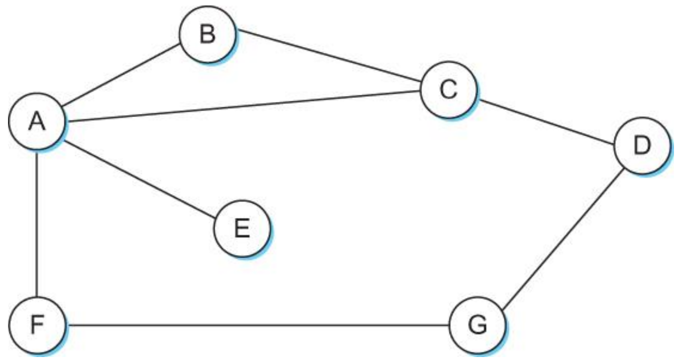


Initial distances stored at each node (global view)

Distance Vector

Initial routing table at node A

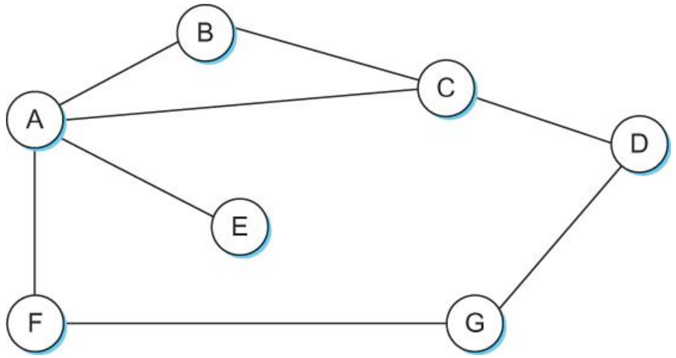
Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—



Distance Vector

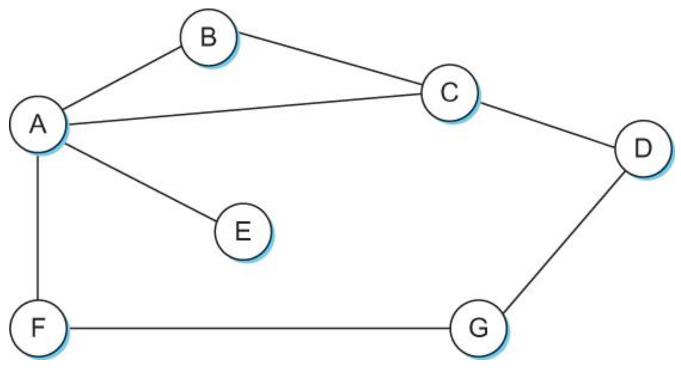
Final routing table at node A

Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F



Distance Vector

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0



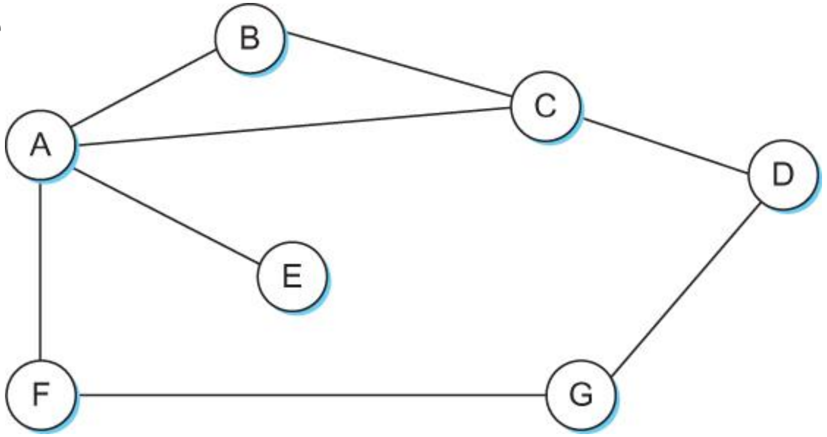
Final distances stored at each node (global view)

Distance Vector

- The distance vector routing algorithm is sometimes called as Bellman-Ford algorithm
- Every T seconds each router sends its table to its neighbor each each router then updates its table based on the new information
- Problems include fast response to good new and slow response to bad news. Also too many messages to update

Distance Vector

- When a node detects a link failure
  - F detects that link to G has failed
  - F sets distance to G to infinity and sends update to A
  - A sets distance to G to infinity since it uses F to reach G
  - A receives periodic update from C with 2-hop path to G
  - A sets distance to G to 3 and sends update to F
  - F decides it can re



Distance Vector

- Slightly different circumstances can prevent the network from stabilizing
  - Suppose the link from A to E goes down
  - In the next round of updates, A advertises a distance of infinity to E, but B and C advertise a distance of 2 to E
  - Depending on the exact timing of events, the following might happen
    - Node B, upon hearing that E can be reached in 2 hops from C, concludes that it can reach E in 3 hops and advertises this to A
    - Node A concludes that it can reach E in 4 hops and advertises this to C
    - Node C concludes that it can reach E in 5 hops; and so on.
    - This cycle stops only when the distances reach some number that is large enough to be considered infinite
      - **Count-to-infinity problem**

Count-to-infinity Problem

- Use some relatively small number as an approximation of infinity
- For example, the maximum number of hops to get across a certain network is never going to be more than 16

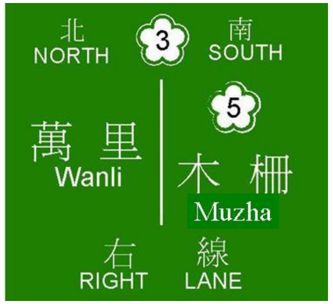
- One technique to improve the time to stabilize routing is called *split horizon*
  - When a node sends a routing update to its neighbors, it does not send those routes it learned from each neighbor back to that neighbor
  - For example, if B has the route (E, 2, A) in its table, then it knows it must have learned this route from A, and so whenever B sends a routing update to A, it does not include the route (E, 2) in that update

Count-to-infinity Problem

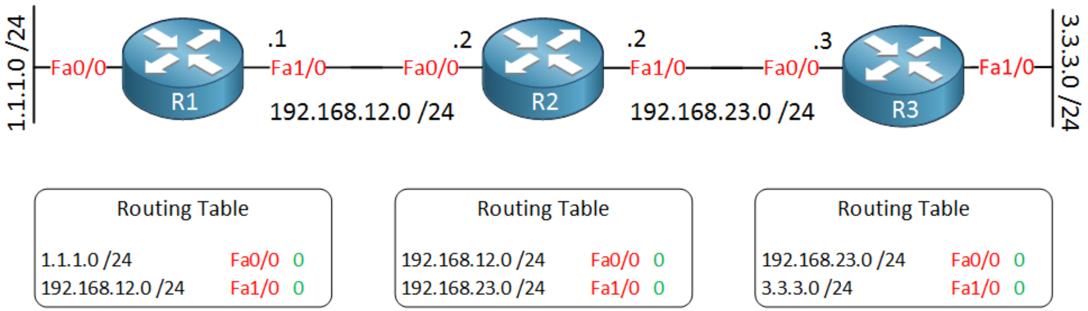
- In a stronger version of split horizon, called *split horizon with poison reverse*
  - B actually sends that back route to A, but it puts negative information in the route to ensure that A will not eventually use B to get to E
  - For example, B sends the route (E, ∞) to A

Routing Information Protocol (RIP)

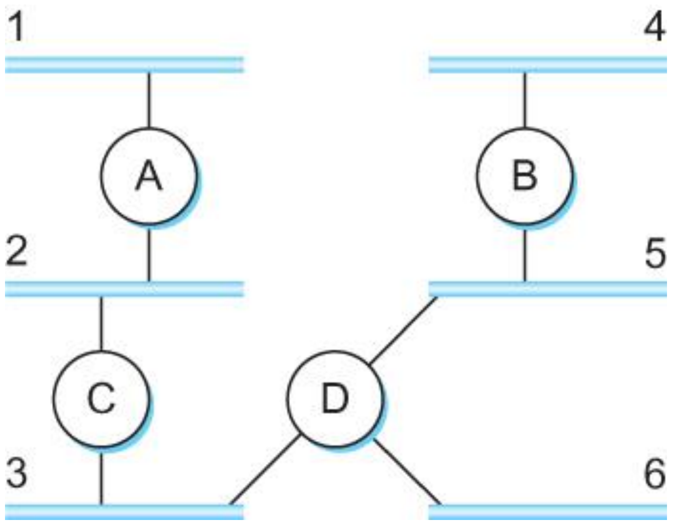
- RIP is a typical distance-vector routing protocol
- Two versions: RIP, RIPv2
- Similar to build road sign direction



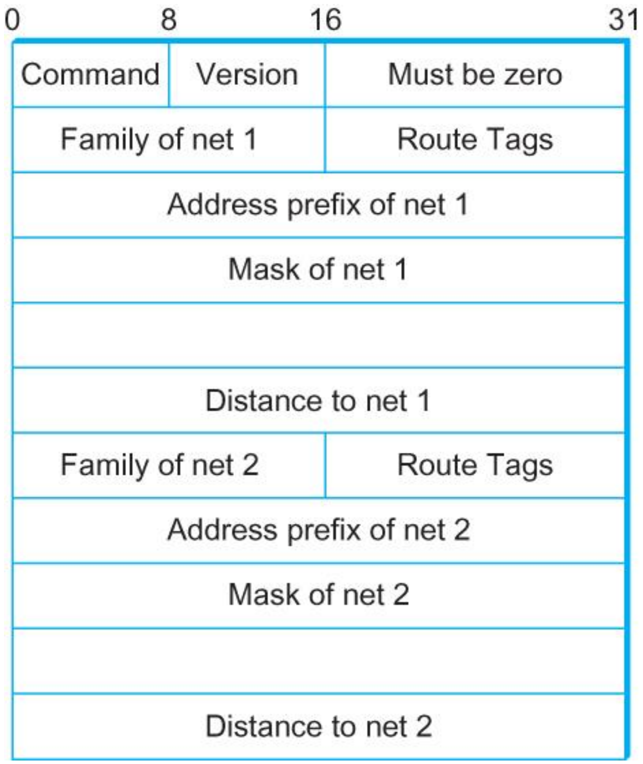
Feature	RIPv1	RIPv2
class	distance vector	distance vector
hop count	15	15
addressing	classful	classless
authentication	none	none / text / MD5
routing updates	255.255.255.255	224.0.0.9



RIP packet format



Example Network
running RIP



RIPv2 Packet Format

Lab7: Configure RIP in Packet Tracer

- Create three networks for three universities (CCU, FCU, NTU)
- Each network has 1 PC and WiFi Access point
- Configure RIP to connect the networks together

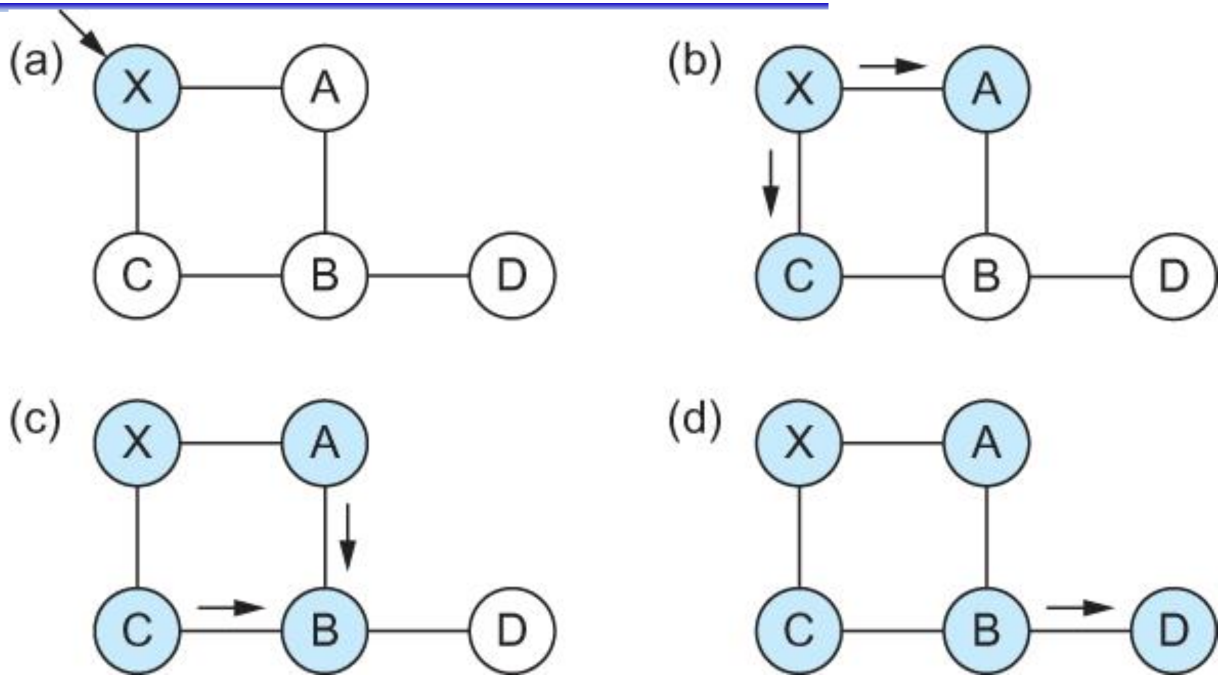
Link State Routing

Strategy: Send to all nodes (not just neighbors) information about directly connected links (not entire routing table).

- Link State Packet (LSP)
  - id of the node that created the LSP
  - cost of link to each directly connected neighbor
  - sequence number (SEQNO)
  - time-to-live (TTL) for this packet
- Reliable Flooding
  - store most recent LSP from each node
  - forward LSP to all nodes but one that sent it
  - generate new LSP periodically; increment SEQNO
  - start SEQNO at 0 when reboot
  - decrement TTL of each stored LSP; discard when TTL=0

Link State

Reliable Flooding



Flooding of link-state packets. (a) LSP arrives at node X; (b) X floods LSP to A and C; (c) A and C flood LSP to B (but not X); (d) flooding is complete

Shortest Path Routing

- Dijkstra's Algorithm - Assume non-negative link weights
  - N: set of nodes in the graph
  - l((i, j): the non-negative cost associated with the edge between nodes i, j $\in$N and l(i, j) = $\propto$ if no edge connects i and j
  - Let s $\in$N be the starting node which executes the algorithm to find shortest paths to all other nodes in N
  - Two variables used by the algorithm
    - M: set of nodes incorporated so far by the algorithm
    - C(n) : the cost of the path from s to each node n
    - The algorithm

```
M = {s}
For each n in N - {s}
    C(n) = l(s, n)
while ( N ≠ M)
  M = M ∪ {w} such that C(w) is the minimum
                            for all w in (N-M)
  For each n in (N-M)
         C(n) = MIN (C(n), C(w) + l(w, n))
```

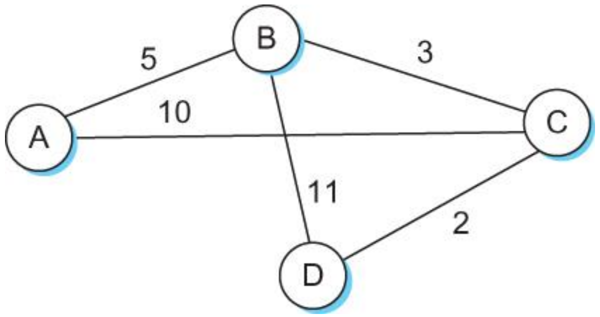
Shortest Path Routing

- In practice, each switch computes its routing table directly from the LSP's it has collected using a realization of Dijkstra's algorithm called the *forward search algorithm*
- Specifically each switch maintains two lists, known as **Tentative** and **Confirmed**
- Each of these lists contains a set of entries of the form (Destination, Cost, NextHop)

Shortest Path Routing

- The algorithm
  - Initialize the **Confirmed** list with an entry for myself; this entry has a cost of 0
  - For the node just added to the **Confirmed** list in the previous step, call it node **Next**, select its LSP
  - For each neighbor (Neighbor) of **Next**, calculate the cost (Cost) to reach this Neighbor as the sum of the cost from myself to Next and from Next to Neighbor
    - If Neighbor is currently on neither the **Confirmed** nor the **Tentative** list, then add (Neighbor, Cost, Nexthop) to the **Tentative** list, where Nexthop is the direction I go to reach Next
    - If Neighbor is currently on the **Tentative** list, and the Cost is less than the currently listed cost for the Neighbor, then replace the current entry with (Neighbor, Cost, Nexthop) where Nexthop is the direction I go to reach Next
  - If the **Tentative** list is empty, stop. Otherwise, pick the entry from the **Tentative** list with the lowest cost, move it to the **Confirmed** list, and return to Step 2.

Shortest Path Routing



Step	Confirmed	Tentative	Comments
1	(D,0,–)		Since D is the only new member of the confirmed list, look at its LSP.
2	(D,0,–)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
3	(D,0,–) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
4	(D,0,–) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
5	(D,0,–) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
6	(D,0,–) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
7	(D,0,–) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

Open Shortest Path First (OSPF)

0	8	16	31
Version	Type	Message length	
SourceAddr			
AreaId			
Checksum		Authentication type	
Authentication			

OSPF Header Format

LS Age			Options	Type = 1
Link-state ID				
Advertising router				
LS sequence number				
LS checksum			Length	
0	Flags	0	Number of links	
Link ID				
Link data				
Link type		Num_TOS	Metric	
Optional TOS information				
More links				

OSPF Link State Advertisement

OSPF

Example of OSPF configuration in Packet Tracer:
https://community.cisco.com/legacyfs/online/legacy/8/9/9/68998-OSPF%20Lab11-1.pdf

OSPF is an Link-State Interior Gateway Protocol used to distribute routing information within a single Autonomous System.

OSPF is built on the Dijkstra algorithm

0	8	16 31
Version	Type	Message length
SourceAddr		
AreaId		
Checksum		Authentication type
Authentication		

OSPF Header Format

LS Age			Options	Type=1
Link-state ID				
Advertising router				
LS sequence number				
LS checksum			Length	
0	Flags	0	Number of links	
Link ID				
Link data				
Link type		Num_TOS	Metric	
Optional TOS information				
More links				

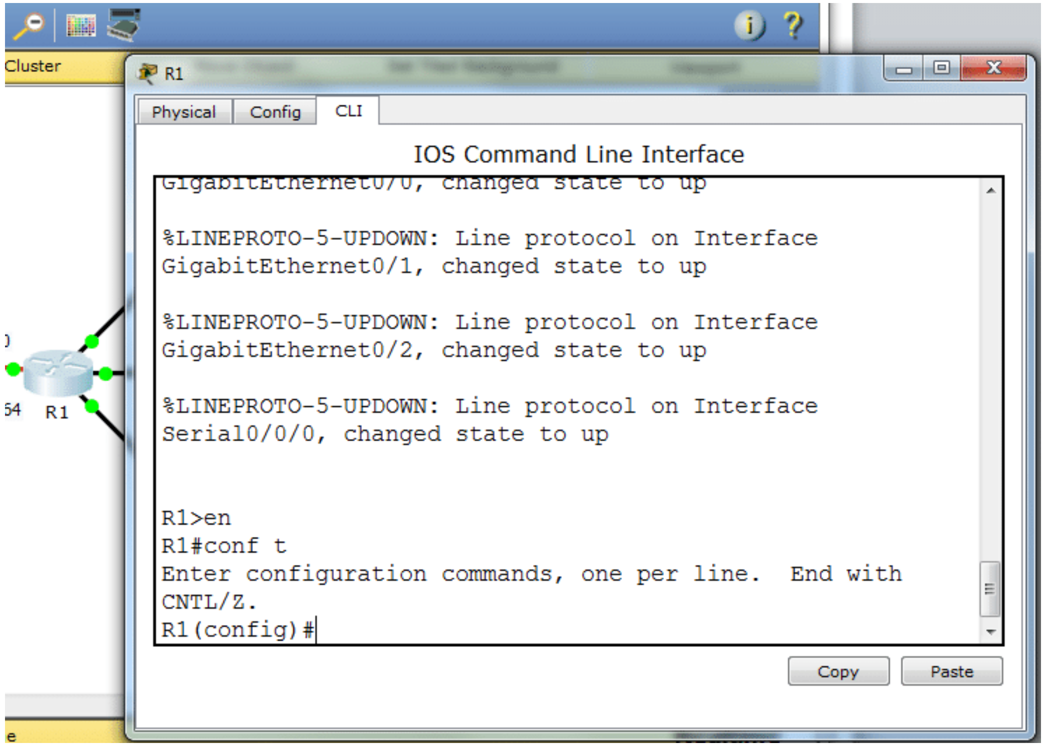
OSPF Link State Advertisement

Configure a router via CLI

- CLI is only the method to configure the advance routing protocols (OSPF, IGRP, BGP) on advanced routers (>10Gbps)

- Provide many options to configure a router which GUI may not fully support.

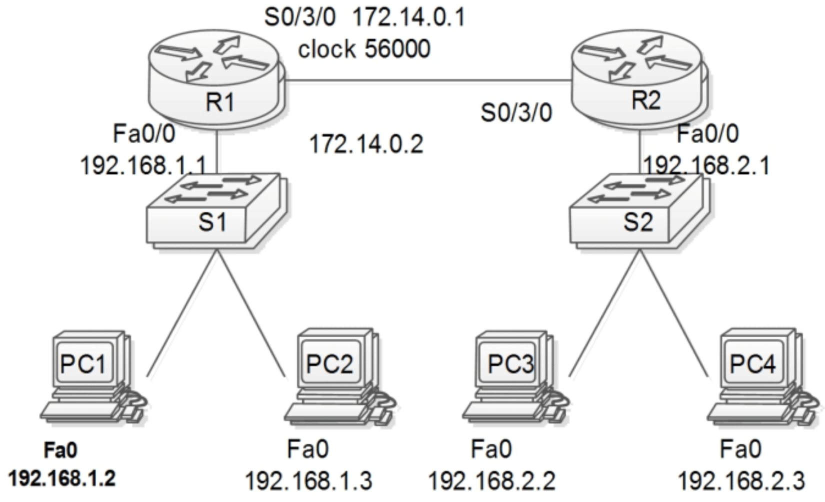
Some basic CLI commands in Packet Tracer: https://www.cisco.com/c/en/us/td/docs/routers/access/800M/software/800MSCG/routconf.html



Configure OSPF via CLI

- Example of a network topology with two routers

- Sample CLI commands for OSPF configuration in Packet Tracer



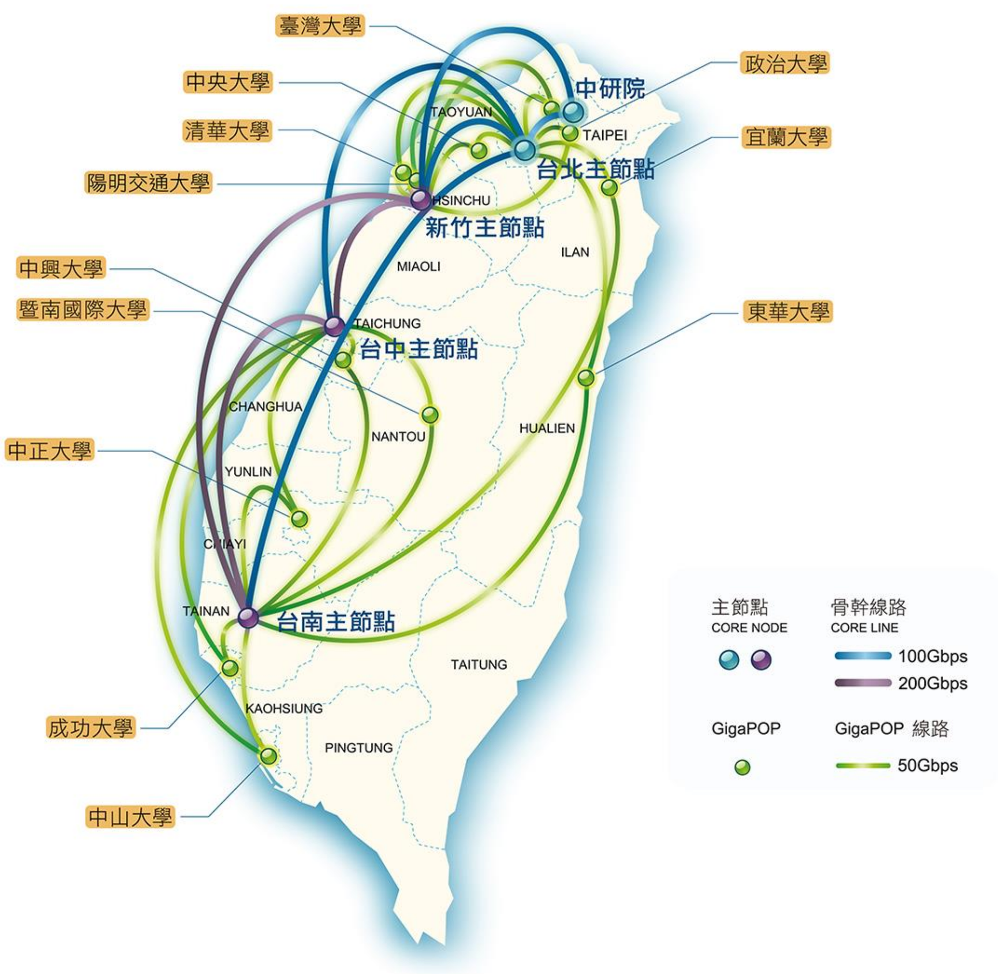
Commands
R1:
R1> enable
R1# configure terminal
R1(config)# interface Fa0/0
R1(config-if)# ip address 192.168.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config)# interface s0/3/0
R1(config-if)# ip address 172.14.0.1 255.255.0.0
R1(config-if)# no shutdown
R1(config-if)# clock rate 5000
R1(config)# router ospf 1
R1(config-router)# network 192.168.1.0 0.0.0.255 area 0
R1(config-router)# network 172.14.0 0.0.255.255 area 0

R2:
R2> enable
R2# configure terminal
R2(config)# interface Fa0/0
R2(config-if)# ip address 192.168.2.1 255.255.255.0
R2(config-if)# no shutdown
R2(config)# interface s0/3/0
R2(config-if)# ip address 172.14.0.2 255.255.0.0
R2(config-if)# no shutdown
R2(config)# router ospf 1
R2(config-router)# network 192.168.2.0 0.0.0.255 area 0
R2(config-router)# network 172.14.0 0.0.255.255 area 0

Lab8: Configure OSPF in Packet Tracer

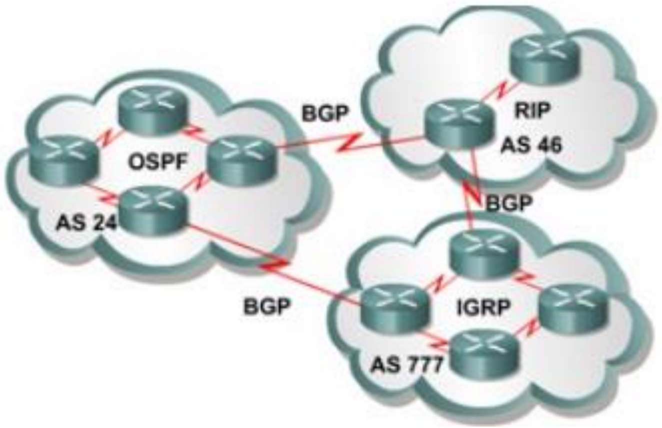
- Build a network as the figure
- Each network has a PC
- Configure OSPF for the network
- PCs can ping to each other

Bonus task: 1%



Other routing protocols

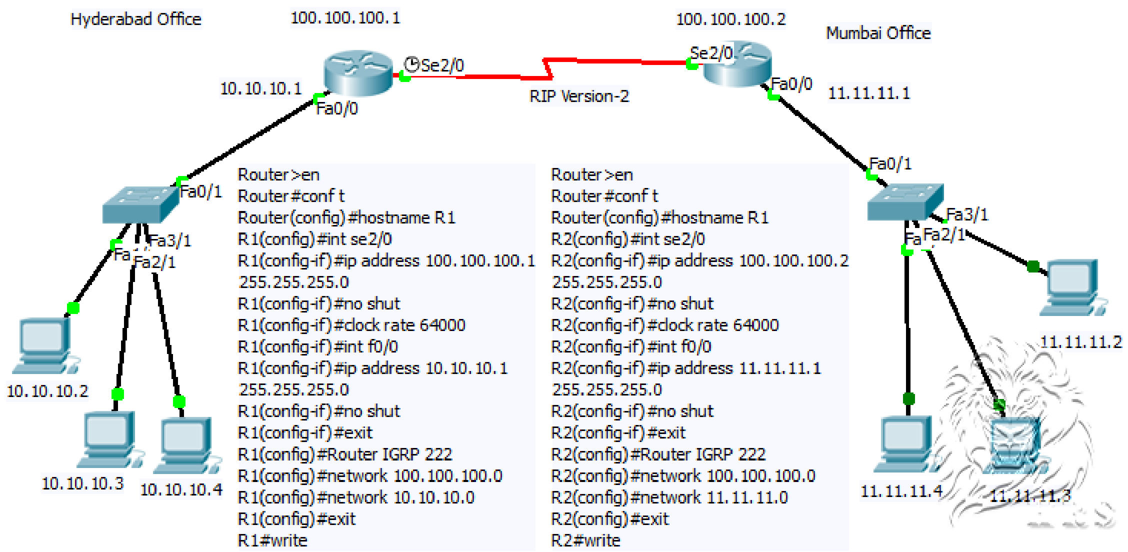
- Interior gateway protocol (IGRP) invented by Cisco
  - ✓ Enhanced interior gateway routing protocol (EIGRP)
- Exterior Gateway Protocol (EGP)
- Border gateway protocol (BGP)
- Intermediate System to Intermediate System (IS-IS)



	Interior Gateway Protocols				Exterior Gateway Protocols
	Distance Vector Routing Protocols		Link State Routing Protocols		Path Vector
Classful	RIPv1 (1982/1988)	IGRP (1985)			EGP (1982)
Classless	RIPv2 (1994)	EIGRP (1992)	OSPFv2 (1991)	IS-IS (1990)	BGPv4 (1995)
IPv6	RIPng (1997)	EIGRP for IPv6 (not yet released)	OSPFv3 (1999)	IS-IS for IPv6 (2000)	BGPv4 for IPv6 (1999)

Interior gateway protocol (IGRP)

- Cisco proprietary protocol
- Distance vector routing protocol
- Use metrics: delay, bandwidth, reliability, load balancing
- Use TCP to exchange routing updates



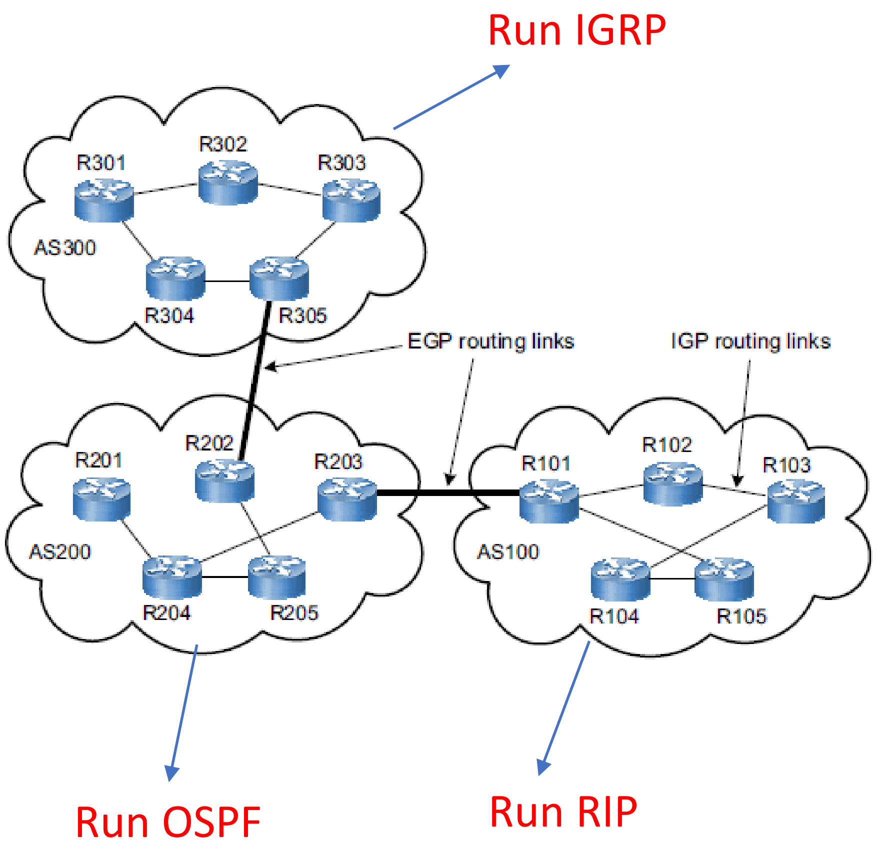
Source: Habib

Example of IGRP configuration in Packet Tracer: http://www.utez.edu.mx/curriculas/ccna2_EN/en-knet-311053022482417/ccna3theme/ccna3/pdf/knet-ESBEJ4gWBQNSMyJQ/CCNA2_lab_inst_7_3_5_en.pdf

Exterior Gateway Protocol (EGP)

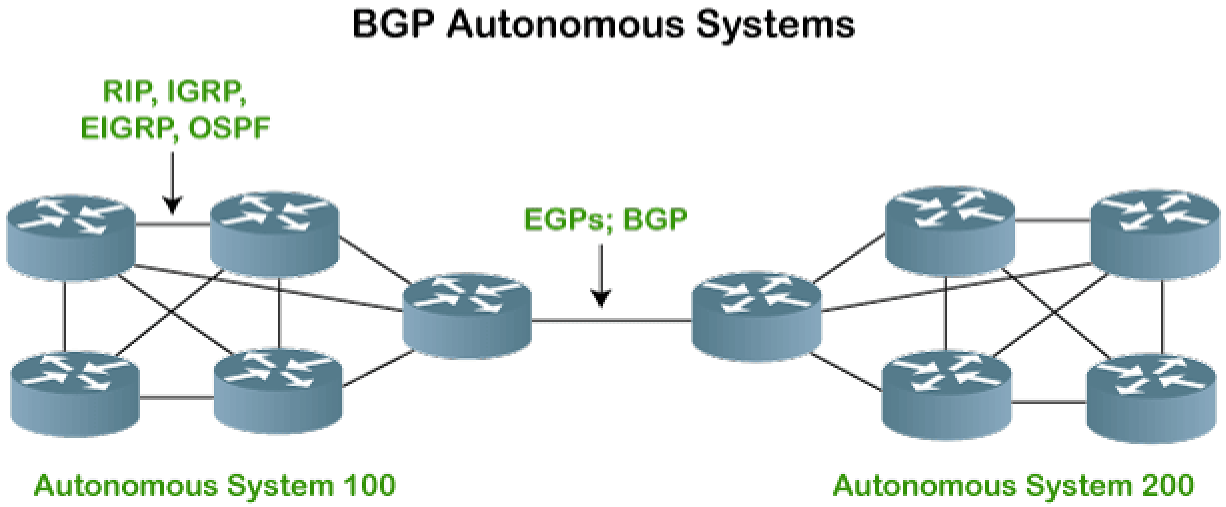
- **Autonomous system** (AS): A collection of networks under the same administrative authority and share a common routing strategy

- Exchange routing information among two routers in a network of autonomous systems.



Border gateway protocol

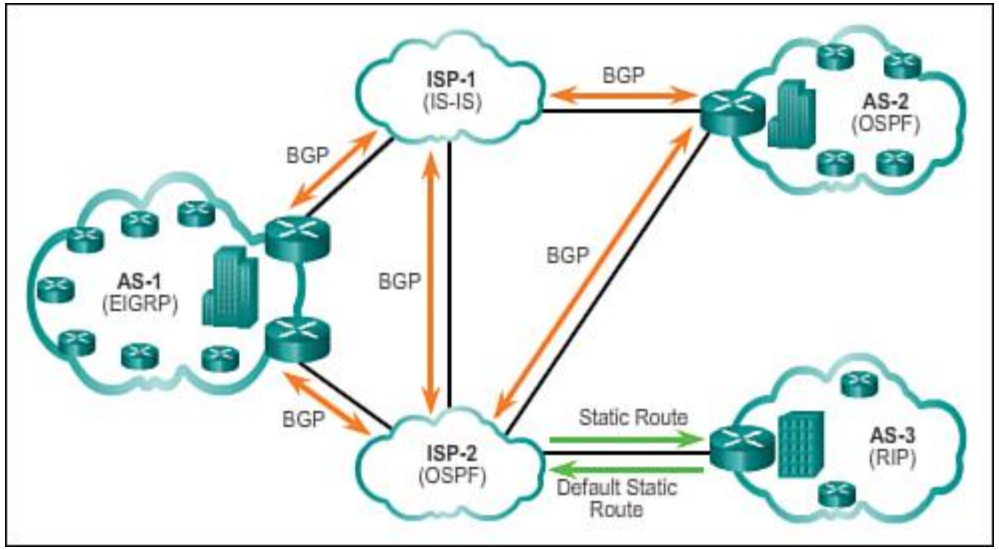
- BGP is a standardized exterior gateway protocol designed to exchange routing and reachability information among autonomous systems



Example of configuring BGP in Packet Tracer: https://ipcisco.com/lesson/bgp-configuration-example-on-packet-tracer/

IS-IS routing protocol

- A link-state routing protocol
- Routers exchange topology information with their nearest neighbors
- Using the shortest-path algorithm to determine routes
- Different from OSPF (IP): IS-IS runs on Layer 2



Lab9: Multiple routing protocols

- Build a network as the figure
- Blue/Green networks use OSPF
- Purple networks use BGP
- PCs in the network can access each other

Bonus task: 1%

Summary

- We have looked at some of the issues involved in building scalable and heterogeneous networks by using switches and routers to interconnect links and networks.
- To deal with heterogeneous networks, we have discussed in details the service model of Internetworking Protocol (IP) which forms the basis of today's routers.
- We have discussed in details two major classes of routing algorithms
  - Distance Vector
  - Link State