SCTR's

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

THIRD YEAR INFORMATION TECHNOLOGY (2019 COURSE)

LABORATORY MANUAL

FOR

COMPUTER NETWORK AND SECURITY LABORATORY

SEMESTER - VI

[Subject code: 314456]

[Prepared By]

Mr. Naman V. Buradkar

Mr. Tushar A. Rane

Mr. Vishal R. Jaiswal

DEPARTMENT OF INFORMATION TECHNOLOGY

INSTITUTE VISION AND MISSION

VISION

Pune Institute of Computer Technology aspires to be the leader inhigher technical education and research of international repute.

MISSION

To be leading and most sought after Institute of education and research in emerging engineering and technology disciplines that attracts, retains and sustains gifted individuals of significant potential.

DEPARTMENT VISION AND MISSION

VISION

The department endeavors to be recognized globally as a center of academic excellence & research in Information Technology.

MISSION

To inculcate research culture among students by imparting information technology related fundamental knowledge, recent technological trends and ethics to get recognized as globally acceptable and socially responsible professionals.

Savitribai Phule Pune University, Pune
Third Year Information Technology (2019 Course)
314456: Computer Network Security Lab

Teaching Scheme:	Credit Scheme:	Examination Scheme:
Practical (PR) : 4 Hrs/week	02 Credit	OR: 50 Marks TW: 25 Marks

Prerequisites:

1. Fundamentals of Computer Networks.

Course Objectives:

1. To design and implement small size network and to understand various networking commands.
2. To learn various client/server environments to use application layer protocols.
3. To understand network layer routing protocols and its implementations.
4. To understand the network security by using public key cryptography algorithms.

Course Outcomes:

On completion of the course, students will be able to–

CO1: Design and configure small size network and associated networking commands.

CO2: Understand various client/server environments to use application layer protocols.

CO3: Use basic cryptographic techniques in software and system design.

CO4: Apply methods for authentication, access control, intrusion detection.

Guidelines for Instructor's Manual

1. The faculty member should prepare the laboratory manual for all the experiments and it shouldbe made available to students and laboratory instructor/assistant.

Guidelines for Student's Lab Journal

1. Student should submit term work in the form of handwritten journal based on specified list of assignments.
2. Practical Examination will be based on the term work.
3. Candidate is expected to know the theory involved in the experiment.
4. The practical examination should be conducted if and only if the journal of the candidate is complete in all respect.

Guidelines for Lab /TW Assessment

1. Examiners will assess the term work based on performance of students considering the parameters such as timely conduction of practical assignment, methodology adopted for implementation of practical assignment, timely submission of assignment in the form of handwritten write-up along with results of implemented assignment, attendance etc.
2. Examiners will judge the understanding of the practical performed in the examination by asking some questions related to theory & implementation of experiments he/she has carried out.
3. Appropriate knowledge of usage of software and hardware related to respective laboratoryshould be checked by the concerned faculty member.

Guidelines for Laboratory Conduction

As a conscious effort and little contribution towards Green IT and environment awareness, attaching printed papers of the program in journal may be avoided. There must be hand-written write-ups for every assignment in the journal. The DVD/CD containing student's programs should be attached to the journal by every student and same to be maintained by department/lab In-charge is highly encouraged. For reference one or two journals may be maintained with program prints at Laboratory.

List of Laboratory Assignments

Group A: Computer Network

1. **Using a Network Simulator (e.g. packet tracer) Configure Router for…**
   - **a)** Configure a router using router commands and Configure Routing Information Protocol(RIP).
   - **b)** Configure Access Control lists – Standard & Extended.
   - **c)** Network Address Translation: Static, Dynamic & PAT (Port Address Translation)
2. **Using a Network Simulator (e.g. packet tracer) Configure Routing Protocols,**
   - **a)** Configure EIGRP – Explore Neighbor-ship Requirements and Conditions, its K Values Metrics Assignment and Calculation.
   - **b)** OSPF – Explore Neighbor-ship Condition and Requirement, Neighbor-ship states, OSPF MetricCost Calculation.
   - **c)** WLAN with static IP addressing and DHCP with MAC security and filters.
3. **Socket Programming in C/C++ on Linux.**
   - **a)** TCP Client, TCP Server
   - **b)** UDP Client, UDP Server
4. **Introduction to server administration** (server administration commands and their applications) and configuration of below Server: (Study/Demonstration Only)
   - **a)** FTP **b)** Web Server

Group B: Network Security

1. Implement a client and a server on different computers using python. Perform the communication between these two entities by using RSA cryptosystem.
2. Implement a client and a server on different computers using python. Perform the authentication of sender between these two entities by using RSA digital signature cryptosystem.
3. Implement a client and a server on different computers using python. Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.
4. Use the snort intrusion detection package to analyze traffic and create a signature to identify problem traffic.

Reference Books:

1. Andrew S. Tanenbaum, David J. Wethrall, Computer Network, Pearson Education, ISBN: 978-0-13-212695-3.
2. Kurose Ross, Computer Networking: A Top Down Approach Featuring the Internet, Pearson Education, ISBN: 978-81-7758-878-1
3. William Stallings, Cryptography and Network Security, Pearson Education, 7th Edition, ISBN 978-0-13-444428-4

Group A: Computer Network

ASSIGNMENT NO : 1

Using a Network Simulator (e.g. packet tracer) Configure Router for...

a) Configure a router using router commands and Configure Routing Information Protocol(RIP).

b) Configure Access Control lists – Standard & Extended.

c) Network Address Translation: Static, Dynamic & PAT (Port Address Translation)

Access control list

ACLs are basically a set of commands, grouped together by a number or name that is used to filter traffic entering or leaving an interface.

When activating an ACL on an interface, you must specify in which direction the traffic should be filtered:

- **Inbound (as the traffic comes into an interface)**
- **Outbound (before the traffic exits an interface)**

Inbound ACLs: Incoming packets are processed before they are routed to an outbound interface. An inbound ACL is efficient because it saves the overhead of routing lookups if the packet will be discarded after it is denied by the filtering tests. If the packet is permitted by the tests, it is processed for routing.

Inbound ACL (Access Control List):
It controls what traffic is allowed to come into a network or device.

Outbound ACL:
It controls what traffic is allowed to go out of a network or device.

Outbound ACLs: Incoming packets are routed to the outbound interface and then processed through the outbound ACL.

Universal fact about Access control list

1. ACLs come in two varieties : **Numbered and named**
2. Each of these references to ACLs supports two types of filtering: **standard and extended.**
3. Standard IP ACLs can filter only on the **source IP address** inside a packet.

Standard Filtering (Standard ACL)
Definition: Filters traffic based solely on the source IP address.

xtended Filtering (Extended ACL)
Definition: Filters traffic based on multiple criteria, such as source IP, destination IP, protocol, and port numbers.

4. Whereas an extended IP ACLs can filter on the **source and destination IP addresses** in the packet.
5. There are two actions an ACL can take: **permit or deny.**
6. Statements are processed top-down.
7. Once a match is found, no further statements are processed—therefore, order is important.
8. If no match is found, the imaginary **implicit deny statement at the end of the ACL** drops the packet.
9. An ACL should have at least one permit statement; otherwise, all traffic will be dropped because of the hidden implicit deny statement at the end of every ACL.

No matter what type of ACL you use, though, you can have only one ACL per protocol, per interface, per direction. For example, you can have one IP ACL inbound on an interface and another IP ACL outbound on an interface, but you cannot have two inbound IP ACLs on the same interface.

Access List Ranges

Type	Range
IP Standard	1–99
IP Extended	100–199
IP Standard Expanded Range	1300–1999
IP Extended Expanded Range	2000–2699

Standard ACLs

A standard IP ACL is simple; it filters based on source address only. You can filter a source network or a source host, but you cannot filter based on the destination of a packet, the particular protocol being used such as the Transmission Control Protocol (TCP) or the User Datagram Protocol (UDP), or on the port number. You can permit or deny only source traffic.

Extended ACLs:

An extended ACL gives you much more power than just a standard ACL. Extended IP ACLs check both the source and destination packet addresses. They can also check for specific protocols, port numbers, and other parameters, which allow administrators more flexibility and control.

Named ACLs

One of the disadvantages of using IP standard and IP extended ACLs is that you reference

them by number, which is not too descriptive of its use. With a named ACL, this is not the case because you can name your ACL with a descriptive name. The ACL named Deny Mike is a lot more meaningful than an ACL simply numbered 1. There are both IP standard and IP extended named ACLs.

Another advantage to named ACLs is that they allow you to remove individual lines out of an ACL. With numbered ACLs, you cannot delete individual statements. Instead, you will need to delete your existing access list and re-create the entire list.

Configuration Guidelines

- Order of statements is important: put the most restrictive statements at the top of the list and the least restrictive at the bottom.
- ACL statements are **processed top-down until a match is found,** and then no more statements in the list are processed.
- If no match is found in the ACL, the packet is dropped (implicit deny).
- Each ACL needs either a unique number or a unique name.
- The router cannot filter traffic that it, itself, originates.
- You can have only one IP ACL applied to an interface in each direction (inbound and outbound)—you can't have two or more inbound or outbound ACLs applied to the same interface. (Actually, you can have one ACL for each protocol, like IP and IPX, applied to an interface in each direction.)
- Applying an empty ACL to an interface permits all traffic by default: in order for an ACL to have an implicit deny statement, you need at least one actual permit or deny statement.
- Remember the numbers you can use for IP ACLs.Standard ACLs can use numbers ranging **1–99 and 1300–1999,** and extended ACLs can use **100–199 and 2000– 2699.**
- Wildcard mask is not a subnet mask. Like an IP address or a subnet mask, a wildcard mask is composed of 32 bits when doing the conversion; subtract each byte in the subnet mask from 255.

There are two special types of wildcard masks:

0.0.0.0 and 255.255.255.255

A 0.0.0.0 wildcard mask is called a host mask

255.255.255.255. If you enter this, the router will cover the address and mask to the keyword any.

Placement of ACLs

Standard ACLs should be placed as close to the destination devices as possible.

Extended ACLs should be placed as close to the source devices as possible.

Standard access lists

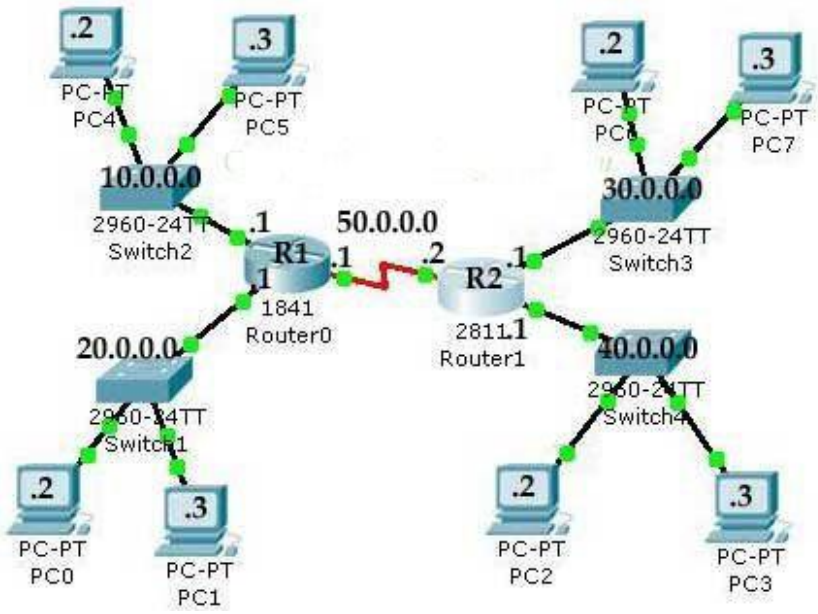
Because a standard access list filters only traffic based on source traffic, all you need is the IP address of the host or subnet you want to permit or deny. ACLs are created in global configuration mode and then applied on an interface. The syntax for creating a standard ACL is

access-list {1-99 | 1300-1999} {permit | deny} source-
address [wildcard mask]

In this article we will configure standard access list. If you want read the feature and characteristic of access list reads this previous article.

Access control list

In this article we will use a RIP running topology. Which we created in RIP routing practical.



Three basic steps to configure Standard Access List

- Use the access-list global configuration command to create an entry in a standard ACL.
- Use the interface configuration command to select an interface to which to apply the ACL.
- Use the ip access-group interface configuration command to activate the existing ACL on an interface.

With Access Lists you will have a variety of uses for the wild card masks, but typically For CCNA exam prospective you should be able to do following:

1. Match a specific host,
2. Match an entire subnet,
3. Match an IP range, or
4. Match Everyone and anyone

Match specific hosts

Task

You have given a task to block 10.0.0.3 from gaining access on 40.0.0.0. While 10.0.0.3 must be able to communicate with networks. Other computer from the network of 10.0.0.0 must be able to connect with the network of 40.0.0.0.

Decide where to apply ACL and in which directions.

Our host must be able to communicate with other host except 40.0.0.0 so we will place this access list on FastEthernet 0/1 of R2 (2811) connected to the network of 40.0.0.0. Direction will be outside as packet will be filter while its leaving the interface. If you place this list on R1(1841) then host 10.0.0.3 will not be able to communicate with any other hosts including 40.0.0.0.

To configure R2 double click on it and select CLI (Choose only one method result will be same)

R2>enable
R2#configure
terminal
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 1 deny host 10.0.0.3
R2(config)#access-list 1 permit
any R2(config)#interface
fastEthernet 0/1 R2(config-if)#ip
access-group 1 out

OR

R2>enable

R2#configure
terminal
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 1 deny 10.0.0.3 0.0.0.0
R2(config)#access-list 1 permit any
R2(config)#interface fastEthernet
0/1 R2(config-if)#ip access-group
1 out

To test first do ping from 10.0.0.3 to 40.0.0.3 it should be request time out as this packet will filter by ACL. Then ping 30.0.0.3 it should be successfully replay.

PC>ping 40.0.0.3

Pinging 40.0.0.3 with 32 bytes of
data: Request timed out.
Request timed
out. Request
timed out.
Request timed
out.

Ping statistics for 40.0.0.3:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>ping 30.0.0.3

Pinging 30.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 30.0.0.3: bytes=32 time=140ms
TTL=126 Reply from 30.0.0.3: bytes=32
time=156ms TTL=126 Reply from 30.0.0.3:
bytes=32 time=112ms TTL=126

Ping statistics for 30.0.0.3:
Packets: Sent = 4, Received = 3, Lost = 1 (25%
loss), Approximate round trip times in milli-
seconds:

Minimum = 112ms, Maximum = 156ms, Average = 136ms

As we applied access list only on specific host so other computer from the network of 10.0.0.0 must be able to connect with the network of 40.0.0.0. To test do ping from 10.0.0.2 to 40.0.0.3

PC>ipconfig

IP Address .....: 10.0.0.2
Subnet Mask. ..... : 255.0.0.0
Default Gateway ..... : 10.0.0.1

PC>ping 40.0.0.3

Pinging 40.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 40.0 0.3: bytes=32 time=141ms TTL=126
Reply from 40.0 0.3: bytes=32 time=140ms TTL=126
Reply from 40.0 0.3: bytes=32 time=125ms TTL=126

Ping statistics for 40.0.0.3:
Packets: Sent = 4, Received = 3, Lost = 1 (25%
loss), Approximate round trip times in milli-
seconds:
Minimum = 125ms, Maximum = 141ms, Average = 135ms

Match an entire subnet
Task
You have given a task to the network of 10.0.0.0 from gaining access on 40.0.0 0. While 10.0.0.0 must be able to communicate with networks .

Wildcards

Wildcards are used with access lists to specify an individual host, a network, or a certain range of a network or networks.

Formula to calculate wild card mask for access list

The key to matching an entire subnet is to use the following formula for the wildcard mask. It goes as follows:

Wildcard mask = 255.255.255.255 – subnet

So for example if my current subnet was 255.0.0.0, the mask would be 0.255.255.255.

255.255.255.255
255 .0 .0 .0 -
----
0. 255 .255.255

Once you have calculated the wild card mask rest is same as we did in pervious example

R2>enable
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 2 deny 10.0.0.0
0.255.255.255 R2(config)#access-list 2 permit any
R2(config)#interface fastethernet
0/1 R2(config-if)#ip access-group
2 out R2(config-if)#

To test first do ping from 10.0.0.3 to 40.0.0.3 it should be request time out as this packet will filter by ACL. Then ping 30.0.0.3 it should be successfully replay.

Now do ping from 10.0.0.2 to 40.0.0.3 and further 30.0.0.2 result should be same as the packet is filtering on **network based**

Match an IP range

You are a network administrator at ComputerNetworkingNotes.com. You task is to block an ip range of 10.3.16.0 – 10.3.31.255 from gaining access to the network of 40.0.0.0

Solutions

Our range is 10.3.16.0 – 10.3.31.255. In order to find the mask, take the higher IP and subtract from it the lower IP.

10.3.31.255

10.3.16.0 -

0.0.15.255

In this case the wildcard mask for this range is 0.0.15.255. To permit access to this range, you would use the following:

R2>enable
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 2 deny 10.3.16.0
0.0.15.255 R2(config)#access-list 2 permit any

R2(config)#interface fastethernet
0/1 R2(config-if)#ip access-group
2 out R2(config-if)#

One thing to note is that each non-zero value in the mask must be one less than a power of 2, i.e. 0, 1, 3, 7, 15, 31, 63, 127, 255.

Match Everyone and Anyone

This is the easiest of Access-Lists to create, just use the following: access-list 1 permit any **or**
access-list 1 permit 0.0.0.0 255.255.255.255

Secure telnet session via standard ACL

This is among the highly tested topic in CCNA exam. We could use extended ACL to secure telnet session but if you did that, you'd have to apply it inbound on every interface, and that really wouldn't scale well to a large router with dozens, even hundreds, of interfaces.Here's a much better solution:

Use a standard IP access list to control access to the VTY lines themselves.

To perform this function, follow these steps:

1. Create a standard IP access list that permits only the host or hosts you want to

be able to telnet into the routers.

2. Apply the access list to the VTY line with the **access-class** command

Secure R2 in a way that only 20.0.0.2 can telnet it beside it all other telnet session should be denied

R2>enable
R2#configure
terminal
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 3 permit host 20.0.0.2
R2(config)#line vty 0 4
R2(config-line)#password
vinita R2(config-line)#login
R2(config-line)#access-class 3 in

To test do telnet from 20.0.0.2 first is should be successful.
PC>ipconfig
IP Address .....: 20.0.0.2
Subnet Mask. ..... : 255.0.0.0

Default Gateway ..... : 20.0.0.1

PC>telnet 50.0.0.2
Trying 50.0.0.2 ...
User Access Verification
Password:
R2>

Now telnet it from any other pc apart from 20.0.0.2. it must be filter
and denied PC>ipconfig
IP Address .....: 20.0.0.3
Subnet Mask. ..... : 255.0.0.0
Default Gateway ..... : 20.0.0.1

PC>telnet 50.0.0.2
Trying 50.0.0.2 ...

% Connection refused by remote host PC>

Configure Extended Access Lists

An extended ACL gives you much more power than just a standard ACL. Extended IP ACLs check both the source and destination packet addresses. They can also check for specific protocols, port numbers, and other parameters, which allow administrators more flexibility and control.

```
access-list access-list-number {permit | deny}
protocol source source-wildcard [operator port]
destination destination-wildcard [operator port]
[established] [log]
```

Command Parameters	Descriptions
access-list	Main command
access-list-number	Identifies the list using a number in the ranges of 100–199 or 2000– 2699.
permit | deny	Indicates whether this entry allows or blocks the specified address.
protocol	IP, TCP, UDP, ICMP, GRE, or IGRP.
source and destination	Identifies source and destination IP addresses.
source-wildcard and destination-wildcard	The operator can be lt (less than), gt (greater than), eq (equal to), or neq (not equal to). The port number referenced can be either the source port or the destination port, depending on where in the ACL the port number is configured. As an alternative to the port number, well-known application names can be used, such as Telnet, FTP, and SMTP.
established	For inbound TCP only. Allows TCP traffic to pass if the packet is a response to an outbound-initiated session. This type of traffic has the acknowledgement (ACK) bits set. (See the Extended ACL with the Established Parameter example.)
log	Sends a logging message to the console.

Before we configure Extended Access list you should cram up some important port number

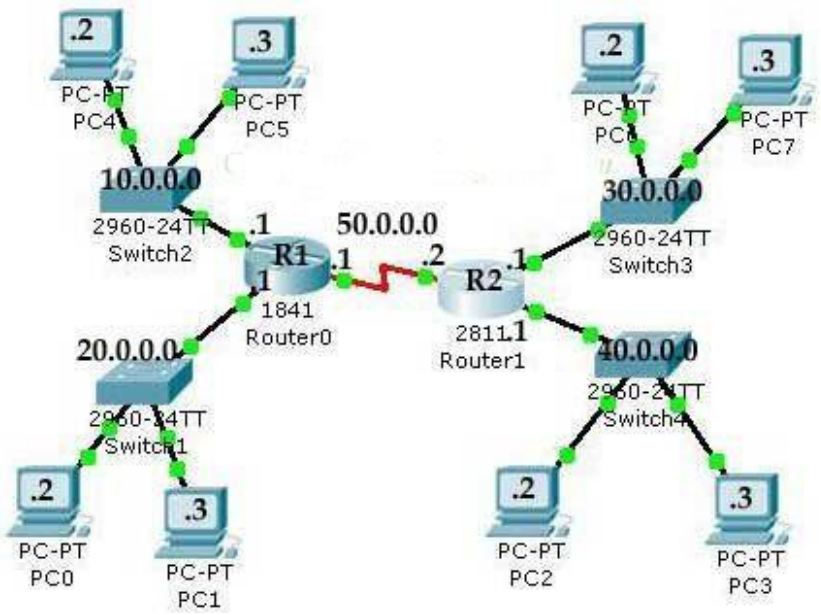
Well-Known Port Numbers and IP Protocols

Port Number	IP Protocol
20 (TCP)	FTP data
21 (TCP)	FTP control
23 (TCP)	Telnet
25 (TCP)	Simple Mail Transfer Protocol (SMTP)
53 (TCP/UDP)	Domain Name System (DNS)
69 (UDP)	TFTP
80 (TCP)	HTTP

In this article we will configure Extended access list. If you want to read the feature and characteristic of access list reads this previous article.

Access control list

In this article we will use a RIP running topology. Which we created in RIP routing practical.



Three basic steps to configure Extended Access List

- Use the access-list global configuration command to create an entry in a Extended ACL.
- Use the interface configuration command to select an interface to which to apply the ACL.
- Use the ip access-group interface configuration command to activate the existing ACL on an interface.

With Access Lists you will have a variety of uses for the wild card masks, but typically For CCNA exam prospective you should be able to do following:

1. ***Block host to host***
2. ***Block host to network***
3. ***Block Network to network***
4. ***Block telnet access for critical resources of company***
5. ***Limited ftp access for user***
6. ***Stop exploring of private network form ping***
7. ***Limited web access***
8. ***Configure established keyword***

Block host to host

Task

*You are the network administrator at **ComputerNetworkingNotes.com.** Your company hire a new employee and give him a pc 10.0.0.3. your company's critical record remain in 40.0.0.3. so you are asked to block the access of 40.0.0.3 from 10.0.0.3. while 10.0.0.3 must be able connect with other computers of network to perfom his task.*

Decide where to apply ACL and in which directions.

As we are configuring Extended access list. With extended access list we can filter the packed as soon as it genrate. So we will place our access list on F0/0 of Router1841 the nearest port of 10.0.0.3

To configure Router1841 (Hostname R1) double click on it and select CLI

R1>enable
R1#configure
terminal
Enter configuration commands, one per line. End with
CNTL/Z. R1(config)#access-list 101 deny ip host 10.0.0.3
40.0.0.3 0.0.0.0 R1(config)#access-list 101 permit ip any
any R1(config)#interface fastEthernet 0/0

R1(config-if)#ip access-group
101 in R1(config-if)#exit
R1(config)#

Verify by doing ping from 10.0.0.3 to 40.0.0.3. It should be reqest time out. Also ping other computers of network including 40.0.0.2. ping shuld be sucessfully.

Block host to network

Task

Now we will block the 10.0.0.3 from gaining access on the network 40.0.0.0. (if you are doing this practical after configuring pervious example don't forget to remove the last access list 101. With no access-list command. Or just close the packet tracer without saving and reopen it to be continue with this example.)

R1(config)#access-list 102 deny ip host 10.0.0.3 40.0.0.0
0.255.255.255 R1(config)#access-list 102 permit ip any any
R1(config)#interface fastEthernet
0/0 R1(config-if)#ip access-group
102 in R1(config-if)#exit
R1(config)#

Verify by doing ping from 10.0.0.3 to 40.0.0.3. and 40.0.0.2.It should be reqest time out. Also ping computers of other network. ping shuld be sucessfully.

Once you have calculated the wild card mask rest is same as we did in pervious example

R2>enable
Enter configuration commands, one per line. End with
CNTL/Z. R2(config)#access-list 2 deny 10.0.0.0
0.255.255.255 R2(config)#access-list 2 permit any
R2(config)#interface fastethernet
0/1 R2(config-if)#ip access-group
2 out R2(config-if)#

To test first do ping from 10.0.0.3 to 40.0.0.3 it should be request time out as this packet will filter by ACL. Then ping 30.0.0.3 it should be successfully replay.

Network to Network Access List

Task

Student's lab is configured on the network of 10.0.0.0. While management's system remain in the network of 40.0.0.0. You are asked to stop the lab system from gaining access in management systems

Now we will block the network of 10.0.0.0 from gaining access on the network 40.0.0.0. (if you are doing this practical after configuring pervious example don't forget to remove the last access list 101. With no access-list command. Or just close the packet tracer without saving and reopen it to be continue with this example.)

R1(config)#access-list 103 deny ip 10.0.0.0 0.255.255.255 40.0.0.0 0.255.255.255 R1(config)#access-list 103 permit ip any any R1(config)#interface fastethernet 0/0 R1(config-if)#ip access-group 103 in R1(config-if)#exit R1(config)#

Verify by doing ping from 10.0.0.3 and 10.0.0.2 to 40.0.0.3. and 40.0.0.2.It should be reqest time out. Also ping computers of other network. ping shuld be sucessfully.

Network to host

Task

For the final scenario you will block all traffic to 40.0.0.3 from the Network of 10.0.0.0 To accomplish this write an extended access list. The access list should look something like the following.

R1(config)#interface fastethernet 0/0 R1(config-if)#no ip access-group 103 in R1(config-if)#exit R1(config)#no access-list 103 deny ip 10.0.0.0 0.255.255.255 40.0.0.0 0.255.255.255 R1(config)#access-list 104 deny ip 10.0.0.0 0.255.255.255 40.0.0.3 0.0.0.0 R1(config)#access-list 104 permit ip any any

R1(config)#interface fastethernet 0/0 R1(config-if)#ip access-group 104 in

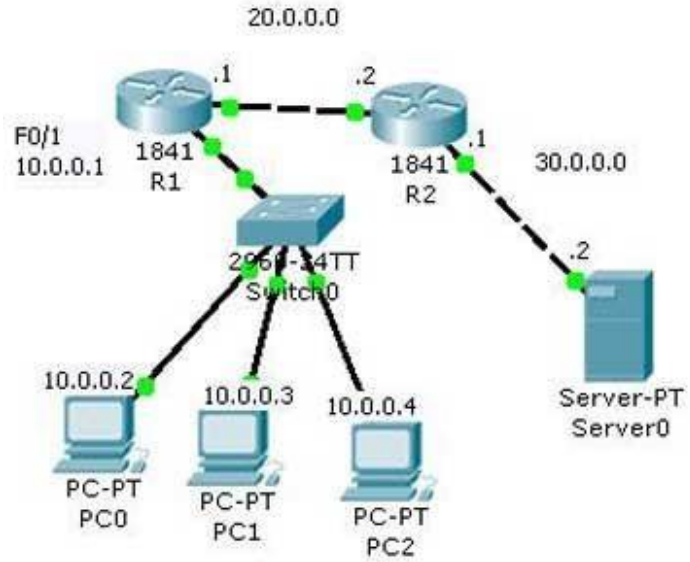
R1(config-if)#exit
R1(config)#

Verify by doing ping from 10.0.0.3 and 10.0.0.2 to 40.0.0.3.It should be reqest time out. Also ping computers of other network. ping shuld be sucessfully.

Application based Extended Access list

In pervoius example we filter ip base traffic. Now we will filter applicaion base traffic. To do this practical either create a topology as shown in figure and enable telnet and http and ftp service on server or download this pre configured topology and load it in packet tracer.

Extended Access list



The established keyword

The **established** keyword is a advanced feature that will allow traffic through only if it sees that a TCP session is already established. A TCP session is considered established if the three-way handshake is initiated first. This keyword is added only to the end of extended ACLs that are filtering TCP traffic.
You can use TCP established to deny all traffic into your network except for incoming traffic that was first initiated from inside your network. This is commonly used to

block all originating traffic from the Internet into a company's network except for Internet traffic that was first initiated from users inside the company. The following configuration would accomplish this for all TCP-based traffic coming in to interface serial 0/0/0 on the router:

R1(config)#access-list 101 permit tcp any any established
R1(config)#interface serial 0/0/0
R1(config-if)#ip access-group
101 in R1(config-if)#exit

Although the access list is using a permit statement, all traffic is denied unless it is first established from the inside network. If the router sees that the three-way TCP handshake is successful, it will then begin to allow traffic through.

To test this access list double click on any pc from the network 10.0.0.0 and select web brower. Now give the ip of 30.0.0.2 web server. It should get sucessfully access the web page. Now go
30.0.0.2 and open command prompt. And do ping to 10.0.0.2 or any pc from the network the 10.0.0.0. it will request time out.

Stop ping but can access web server

We host our web server on 30.0.0.2. But we do not want to allow external user to ping our server as it could be used as denial of services. Create an access list that will filter all ping requests inbound on the serial 0/0/0 interface of router2.

R2(config)#access-list 102 deny icmp any any
echo R2(config)#access-list 102 permit ip any
any R2(config)#interface serial 0/0/0
R2(config-if)#ip access-group 102 in

To test this access list ping from 10.0.0.2 to 30.0.0.2 it should be request time out. Now open the web browser and access 30.0.0.2 it should be successfully retrieve

Grant FTP access to limited user

You want to grant ftp access only to 10.0.0.2. no other user need to provide ftp access on server. So you want to create a list to prevent FTP traffic that originates

from the subnet 10.0.0.0/8, going to the 30.0.0.2 server, from traveling in on Ethernet interface E0/1 on R1.

R1(config)#access-list 103 permit tcp host 10.0.0.2 30.0.0.2 0.0.0.0 eq 20
R1(config)#access-list 103 permit tcp host 10.0.0.2 30.0.0.2
0.0.0.0 eq 21 R1(config)#access-list 103 deny tcp any any eq 20
R1(config)#access-list 103 deny tcp any any
eq 21 R1(config)#access-list 103 permit ip
any any R1(config)#interface fastethernet
0/1
R1(config-if)#ip access-group
103 in R1(config-if)#exit

Grant Telnet access to limited user

For security purpose you don't want to provide telnet access on server despite your own system. Your system is 10.0.0.4. create a extended access list to prevent telnet traffic that originates from the subnet of 10.0.0.0 to server.

R1(config)#access-list 104 permit tcp host 10.0.0.4 30.0.0.2 0.0.0.0 eq 23
R1(config)#access-list 104 deny tcp 10.0.0.0 0.255.255.255 30.0.0.2
0.0.0.0 eq 23 R1(config)#access-list 104 permit ip any any
R1(config)#interface fast 0/1
R1(config-if)#ip access-group
104 in R1(config-if)#exit

N A T – P A T

This assignment explains basic concepts of static NAT, dynamic NAT, PAT, inside local, outside local, inside global and outside global in detail with examples.

Basic overview of NAT

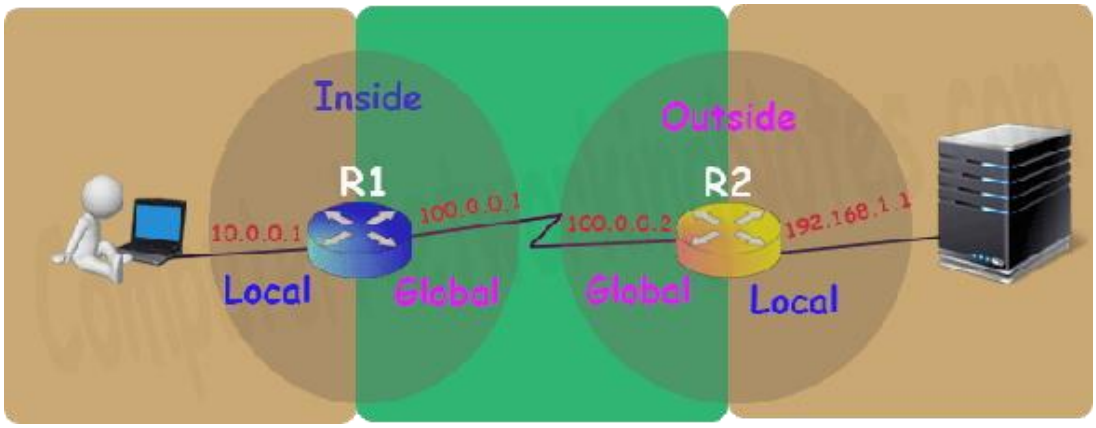
There are several situations where we need address translation such as, a network which do not have sufficient public IP addresses want to connect with the Internet, two networks which have same IP addresses want to merge or due to security reason a network want to hide its internal IP structure from the external world. NAT (Network Address Translation) is the process which translates IP address. NAT can be performed at firewall, server and router. In this assignment we will understand how it is performed at Cisco router.

NAT Terminology

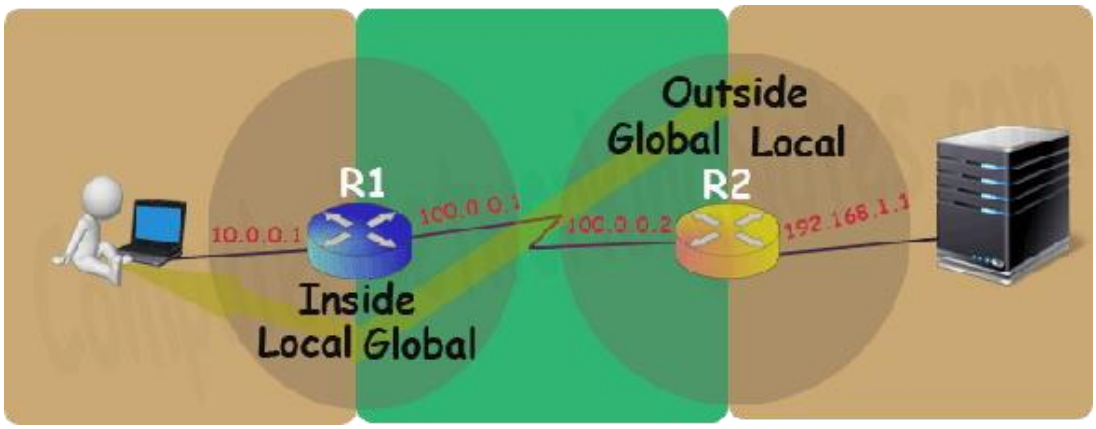
Before we understand NAT in details let's get familiar with four basic terms used in NAT.

Term	**Description**
Inside Local IP Address	Before translation source IP address located inside the local network.
Inside Global IP	After translation source IP address located outside the local network.
Address Outside	Before translation destination IP address located outside the remote network.
Global IP Address	After translation destination IP address located inside the remote network.
Outside Local IP Address	

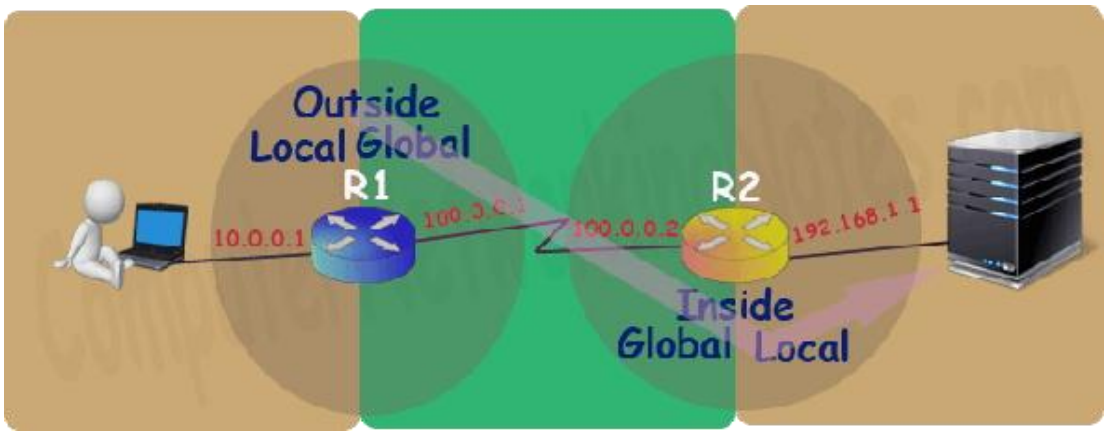
Let's understand these terms with an example. Suppose a user is browsing a website from his home computer. The network which connects his computer with internet is considered as a local network for him. Same as the network which connects the webserver where the website is located with internet is considered as a local network for webserver. The network which connects both networks on internet is considered as a global network.



On router the interface which is connected with local network will be configured with inside local IP address and the interface which is connected with global network will be configured with inside global IP address. Inside and outside depend on where we are standing right now. For example in above network for user router R1 is inside and router R2 is outside.



While for webserver router R2 is inside and router R1 is outside.

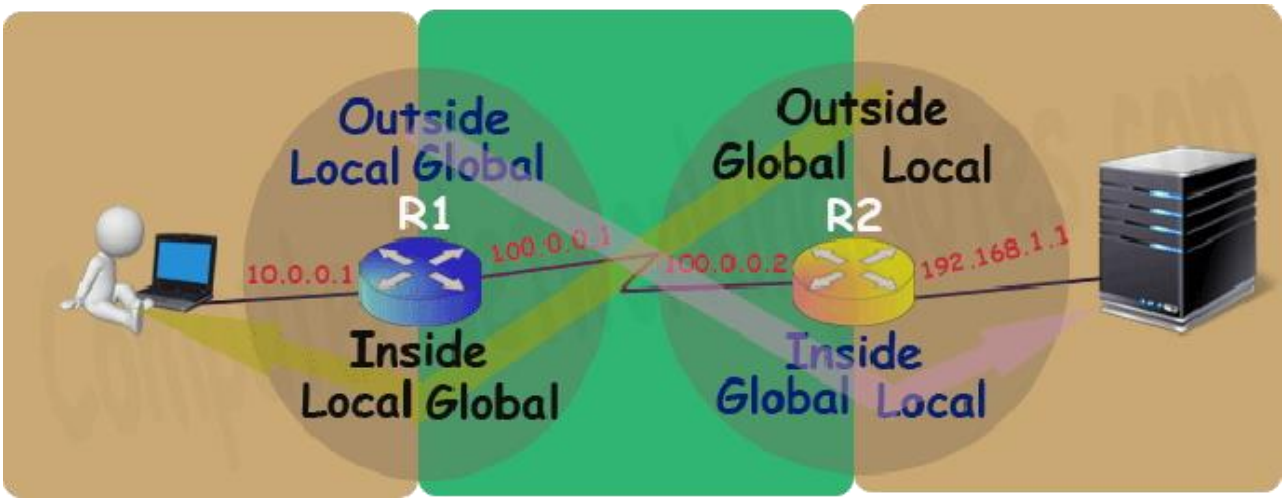


Basically on a NAT enabled router there are two types of interface inside local and inside global.

So, what about outside global and outside local? Well… these terms are used to explain the NAT process theoretically. Practically we never need to configure the outside local and outside global as they sound. For example let's discuss above example once again.

On R1 we will configure inside local address (10.0.0.1) and inside global address (100.0.0.1) which will become outside local address (10.0.0.1) and outside global address (100.0.0.1) for R2 respectively.

Same way on R2 we will configure inside local address (192.168.1.1) and inside global address (100.0.0.2) which will become outside local address (192.168.1.1) and outside global address (100.0.0.2) for R1 respectively.



So practically we only configure inside local and inside global. What is inside for one side is the outside for other side.

Types of NAT

There are three types of NAT; Static NAT, Dynamic NAT and PAT. These types define how inside local IP address will be mapped with inside global IP address.

Static NAT 1 PUBLIC IP --> 1 PRIVATE

In this type we manually map each inside local IP address with inside global IP address. Since this type uses one to one mapping we need exactly same number of IP address on both sides.

Dynamic NAT 1 PUBLIC POOL --> 1 PRIVATE POOL

In this type we create a pool of inside global IP addresses and let the NAT device to map inside local IP address with the available outside global IP address from the pool automatically.

PAT MULTIPLE PUBLIC --> MULTIPLE PRIVATE

In this type a single inside global IP address is mapped with multiple inside local IP addresses using the source port address. This is also known as PAT (Port Address Translation) or NAT over load.

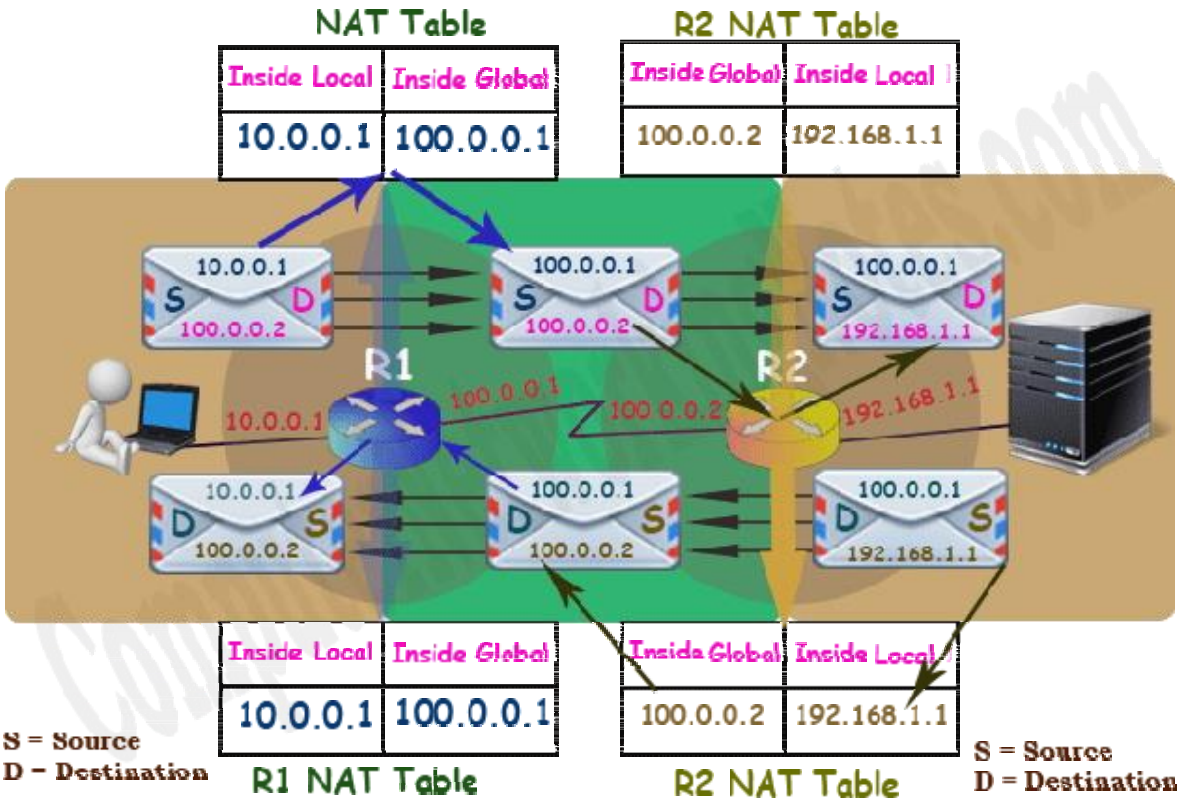
Situations where NAT is used

There are no hard and fast rules about where we should use NAT or where we should not use the NAT. Whether we should use the NAT or not is purely depends on network requirement for example NAT is the best solution in following situations: -

- Our network is built with private IP addresses and we want to connect it with internet. As we know to connect with internet we require public IP address. In this situation we can use NAT device which will map private IP address with public IP address.
- Two networks which are using same IP address scheme want to merge. In this situation NAT device is used to avoid IP overlapping issue.
- We want to connect multiple computers with internet through the single public IP address. In this situation NAT is used to map the multiple IP addresses with single IP address through the port number.

How NAT Works

To understand how NAT works, let's take one more example. In this example a user is accessing a web server. User and Webserver both are connected through the NAT devices. Both user and webserver are using private IP addresses which are not routable on the internet. Now let's understand how NAT makes this communication possible.



User generates a data packet for web server. This packet has source address 10.0.0.1 and destination address 100.0.0.2.

Here source address is the correct address but why the packet has destination address 100.0.0.2 instead of actual destination address 192.168.1.1?

When a system needs to connect with the website, it uses DNS server to resolve the IP address of the website. DNS server advertises the global IP address of the website. Outsider can connect with the website through the advertised IP address only. In our example the global IP address of web server is 100.0.0.2. For this reason the packet has the destination address 100.0.0.2 instead of 192.168.1.1.

This packet reaches at R1. Since this packet contains private IP address in source filed which is not routable on internet, R1 has to update the private IP address with a routable public IP address before forwarding this packet.

R1 checks NAT table for available public IP addresses. Depending on what type of NAT (Static, Dynamic or PAT) is configured one routable public IP will be picked from NAT table for this packet.

In our example 100.0.0.1 is picked for this packet. Now R1 will replace 10.0.0.1 with 100.0.0.1 in the source filed of the packet and forward it to the R2.

R2 receives this packet and reads the destination IP address. R2 looks in NAT table to find out the actual IP address of the destination. Since the NAT table of R2 has an entry for the address
100.0.0.2 which maps it with the address 192.168.1.1, R2 will replace the destination address
100.0.0.2 with the address 192.168.1.1 and forward it to the web server.

Webserver will process this packet and reply with its own packet. This packet has source address 192.168.1.1 and destination address 100.0.0.1.

Since webserver received this packet from 100.0.0.1 so it will reply to it instead of 10.0.0.1.

R2 receives this packet. Before forwarding this packet R2 will replace the source IP address with the mapped IP address in NAT table. In this example 192.168.1.1 will be replaced with 100.0.0.2.

R1 receives this packet and checks its destination address. R1 will perform a query in NAT table to figure out the IP address which is associated with this destination IP address. Since this destination IP address 100.0.0.1 is mapped with 10.0.0.1, R1 will replace this destination IP address 100.0.0.1 with 10.0.0.1 and forward it to the PC.

From user's point of view the IP address of the webserver is 100.0.0.2. While from web server's point of view the IP address of the user is 100.0.0.1. This way both user and webserver will never know to whom they are communicating actually.

Advantages and disadvantages of NAT

Nat provides following advantages: -

- NAT solves IP overlapping issue.
- NAT hides internal IP structure from external world.
- NAT allows us to connect with any network without changing IP address.
- NAT allows us to connect multiple computers with internet through the single the public IP address.

NAT has following disadvantages: -

- NAT adds additional delay in network.

- Several applications are not compatible with NAT.
- End to end IP traceability will not work with NAT.
- NAT hides actual end device.

That's all for this article. In next part of this assignment we will learn how to configure static NAT and dynamic NAT in Cisco router.

How to Configure Static NAT in Cisco Router

This assignment explains how to configure static NAT (Network Address Translation) in Cisco Router step by step with Packet Tracer examples.

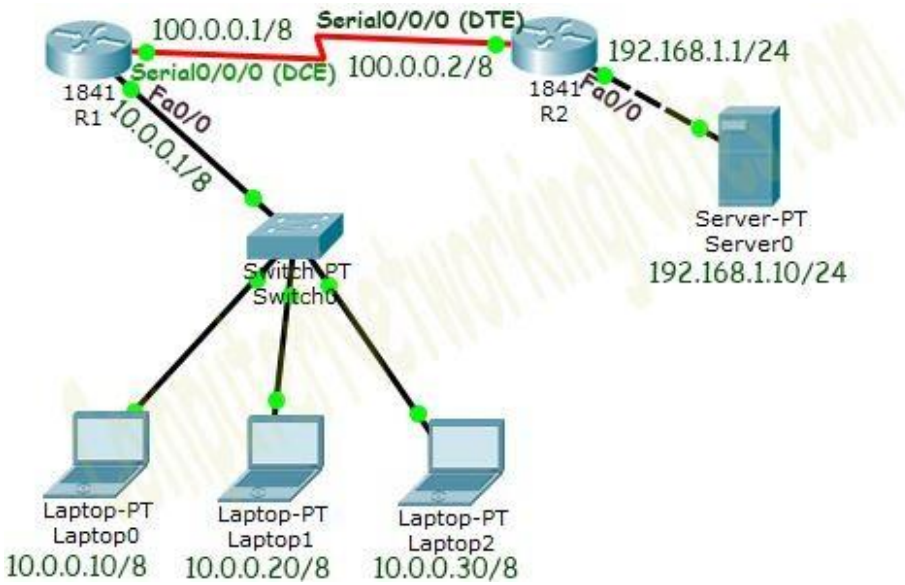
In order to configure NAT we have to understand four basic terms; inside local, inside global, outside local and outside global. These terms define which address will be mapped with which address.

Term	**Description**
Inside Local IP Address	Before translation source IP address located inside the local network.
Inside Global IP Address	After translation source IP address located outside the local network.
Outside Global IP Address	Before translation destination IP address located outside the remote network.
Outside Local IP Address	After translation destination IP address located inside the remote network.

Static NAT Practice LAB Setup

In this assignment I will use Packet Tracer network simulator software for

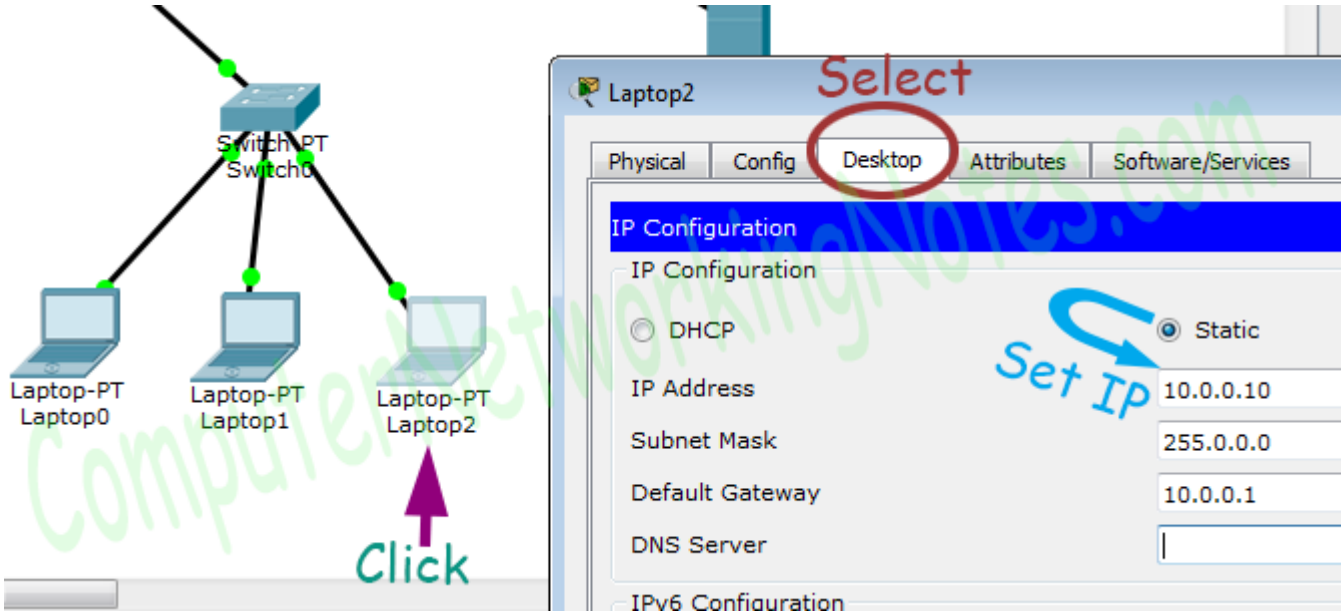
demonstration. Create a lab as illustrates in following figure.



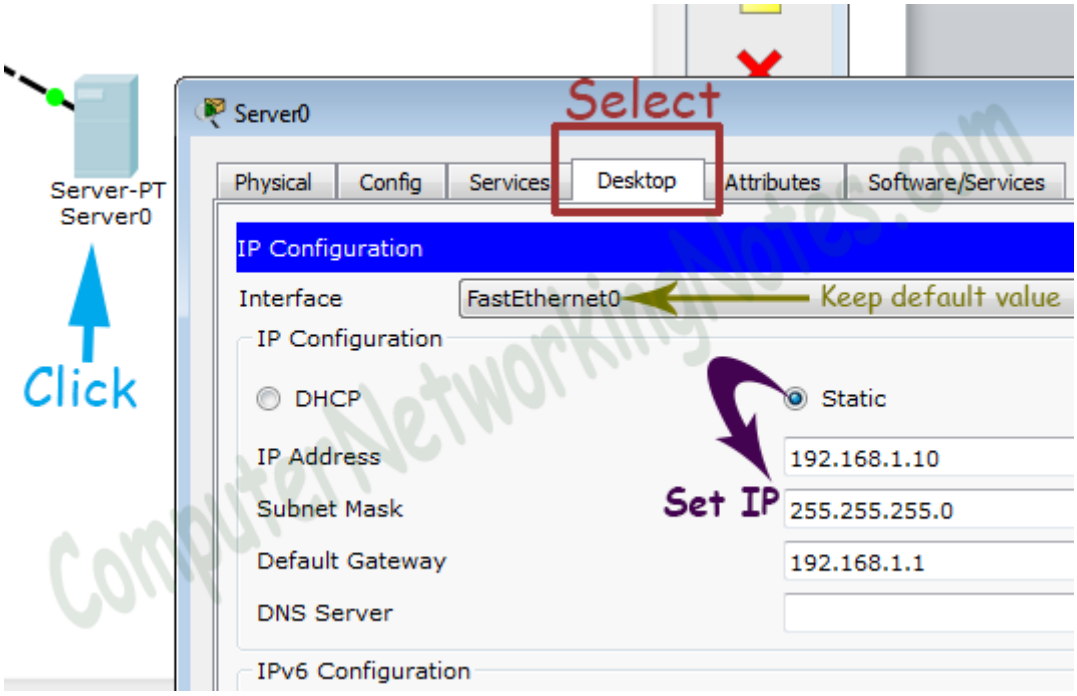
Initial IP Configuration

Device / Interface	**IP Address**	**Connected With**
Laotop0	10.0.0.10/8	Fa0/0 of R0
Laptop1	10.0.0.20/8	Fa0/0 of R0
Laptop2	10.0.0.30/8	Fa0/0 of R0
Server0	192.168.1.10/24	Fa0/0 of R1
Serial 0/0/0 of R1	100.0.0.1/8	Serial 0/0/0 of R2
Serial 0/0/0 of R2	100.0.0.2/8	Serial 0/0/0 of R2

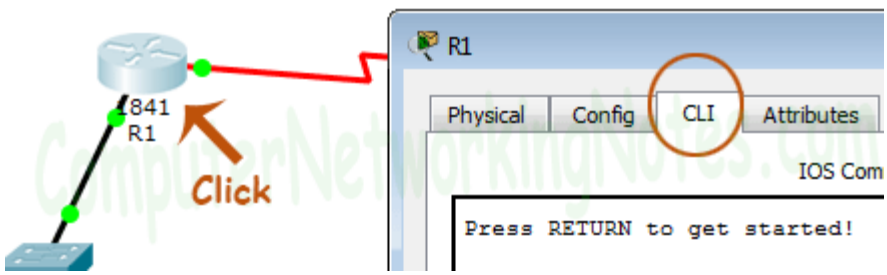
To assign IP address in Laptop click **Laptop** and click **Desktop** and **IP configuration** and Select **Static** and **set IP address** as given in above table.



Following same way configure IP address in Server.



To configure IP address in Router1 click **Router1** and select **CLI** and press **Enter key**.



Two interfaces of Router1 are used in topology; FastEthernet0/0 and Serial 0/0/0.

By default interfaces on router are remain administratively down during the start up. We need to configure IP address and other parameters on interfaces before we could actually use them for routing. Interface mode is used to assign the IP address and other parameters. Interface mode can be accessed from global configuration mode. Following commands are used to access the global configuration mode.

```
Router>enable
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

Before we configure IP address in interfaces let's assign a unique descriptive name to router.

```
Router(config)#hostname R1
R1#
```

Now execute the following commands to set IP address in FastEthernet 0/0 interface.

```
R1(config)#interface FastEthernet0/0
R1(config-if)#ip address 10.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
```

interface FastEthernet 0/0 command is used to enter in

interface mode. ***ip address 10.0.0.1 255.0.0.0*** command assigns

IP address to interface. ***no shutdown*** command is used to bring

the interface up.

exit command is used to return in global configuration mode.

Serial interface needs two additional parameters clock rate and bandwidth. Every serial cable has two ends DTE and DCE. These parameters are always configured at DCE end.

We can use show controllers interface command from privilege mode to check the cable's end.

```
R1(config)#exit
R1#show controllers serial 0/0/0
Interface Serial0/0/0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 2000000
[Output omitted]
```

Fourth line of output confirms that DCE end of serial cable is attached. If you see DTE here instead of DCE skip these parameters.

Now we have necessary information let's assign IP address to serial interface.

```
R1#configure terminal
R1(config)#interface Serial0/0/0
```

```
R1(config-if)#ip address 100.0.0.1 255.0.0.0
R1(config-if)#clock rate 64000
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
```

Router#configure terminal Command is used to enter in global configuration

mode. ***Router(config)#interface serial 0/0/0*** Command is used to enter in

interface mode. ***Router(config-if)#ip address 100.0.0.1 255.0.0.0*** Command

assigns IP address to interface. **Router(config-if)#clock rate 64000**

In real life environment this parameter controls the data flow between serial links and need to be set at service provider's end. In lab environment we need not to worry about this value. We can use any valid rate here.

Router(config-if)#bandwidth 64

Bandwidth works as an influencer. It is used to influence the metric calculation of EIGRP or any other routing protocol which uses bandwidth parameter in route selection process.

Router(config-if)#no shutdown Command brings interface up.

Router(config-if)#exit Command is used to return in global configuration mode.

We will use same commands to assign IP addresses on interfaces of Router2. We need to provided clock rate and bandwidth only on DCE side of serial interface. Following command will assign IP addresses on interface of Router2.

Initial IP configuration in R2

```
Router>enable
Router#configure terminal
Router(config)#hostname R2
R2(config)#interface FastEthernet0/0
R2(config-if)#ip address 192.168.1.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
```

```
R2(config)#interface Serial0/0/0
R2(config-if)#ip address 100.0.0.2 255.0.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
```

That's all initial IP configuration we need. Now this topology is ready for the practice of static nat.

Configure Static NAT

Static NAT configuration requires three steps: -

- Define IP address mapping
- Define inside local interface
- Define inside global interface

Since static NAT use manual translation, we have to map each inside local IP address (which needs a translation) with inside global IP address. Following command is used to map the inside local IP address with inside global IP address.

```
Router(config)#ip nat inside source static [inside local ip address] [inside global IP address]
```

For example in our lab Laptop1 is configured with IP address 10.0.0.10. To map it with 50.0.0.10 IP address we will use following command

```
Router(config)#ip nat inside source static 10.0.0.10 50.0.0.10
```

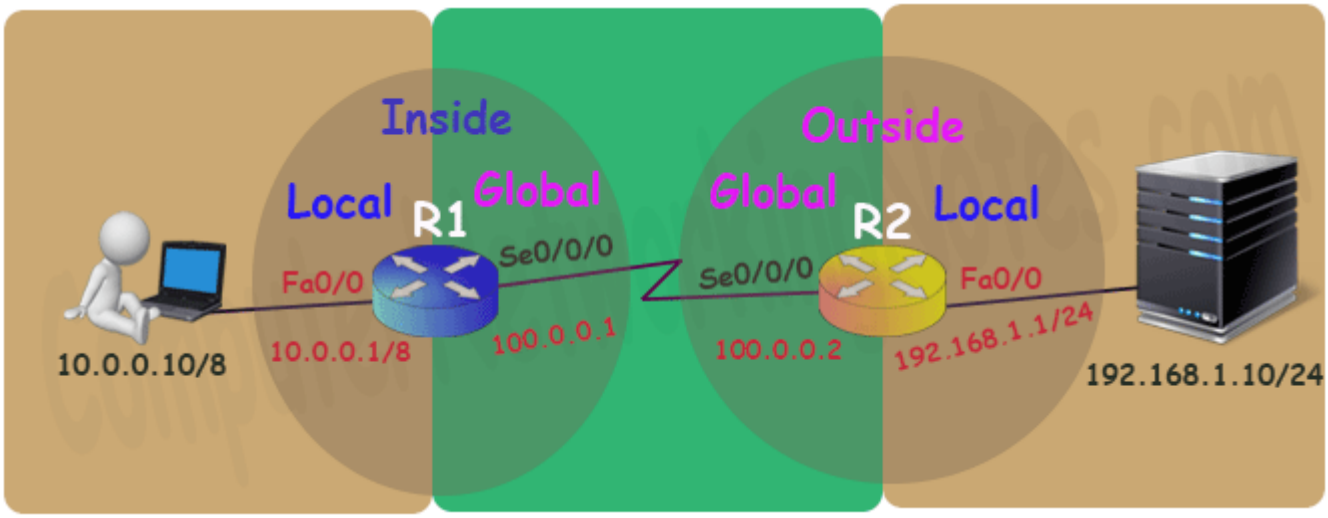
In second step we have to define which interface is connected with local the network. On both routers interface Fa0/0 is connected with the local network which need IP translation. Following command will define interface Fa0/0 as inside local.

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

In third step we have to define which interface is connected with the global network. On both routers serial 0/0/0 interface is connected with the global network. Following ommand will define interface Serial0/0/0 as inside global.

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

Following figure illustrates these terms.



Let's implement all these commands together and configure the static NAT.

R1 Static NAT Configuration

```
R1(config)#ip nat inside source static 10.0.0.10 50.0.0.10
R1(config)#interface FastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#
R1(config)#interface Serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```

For testing purpose I configured only one static translation. You may use following commands to configure the translation for remaining address.

```
R1(config)#ip nat inside source static 10.0.0.20 50.0.0.20
R1(config)#ip nat inside source static 10.0.0.30 50.0.0.30
```

R2 Static NAT Configuration

```
R2(config)#ip nat inside source static 192.168.1.10 200.0.0.10
R2(config)#interface FastEthernet 0/0
R2(config-if)#ip nat inside
R2(config-if)#exit
R2(config)#
R2(config)#interface Serial 0/0/0
```

```
R2(config-if)#ip nat outside
R2(config-if)#exit
```

Before we test this lab we need to configure the IP routing. IP routing is the process which allows router to route the packet between different networks. Following assignment explain routing in detail with examples

Routing concepts Explained with Examples

Configure static routing in R1

```
R1(config)#ip route 200.0.0.0 255.255.255.0 100.0.0.2
```

Configure static routing in R2

```
R2(config)#ip route 50.0.0.0 255.0.0.0 100.0.0.1
```

Testing Static NAT Configuration

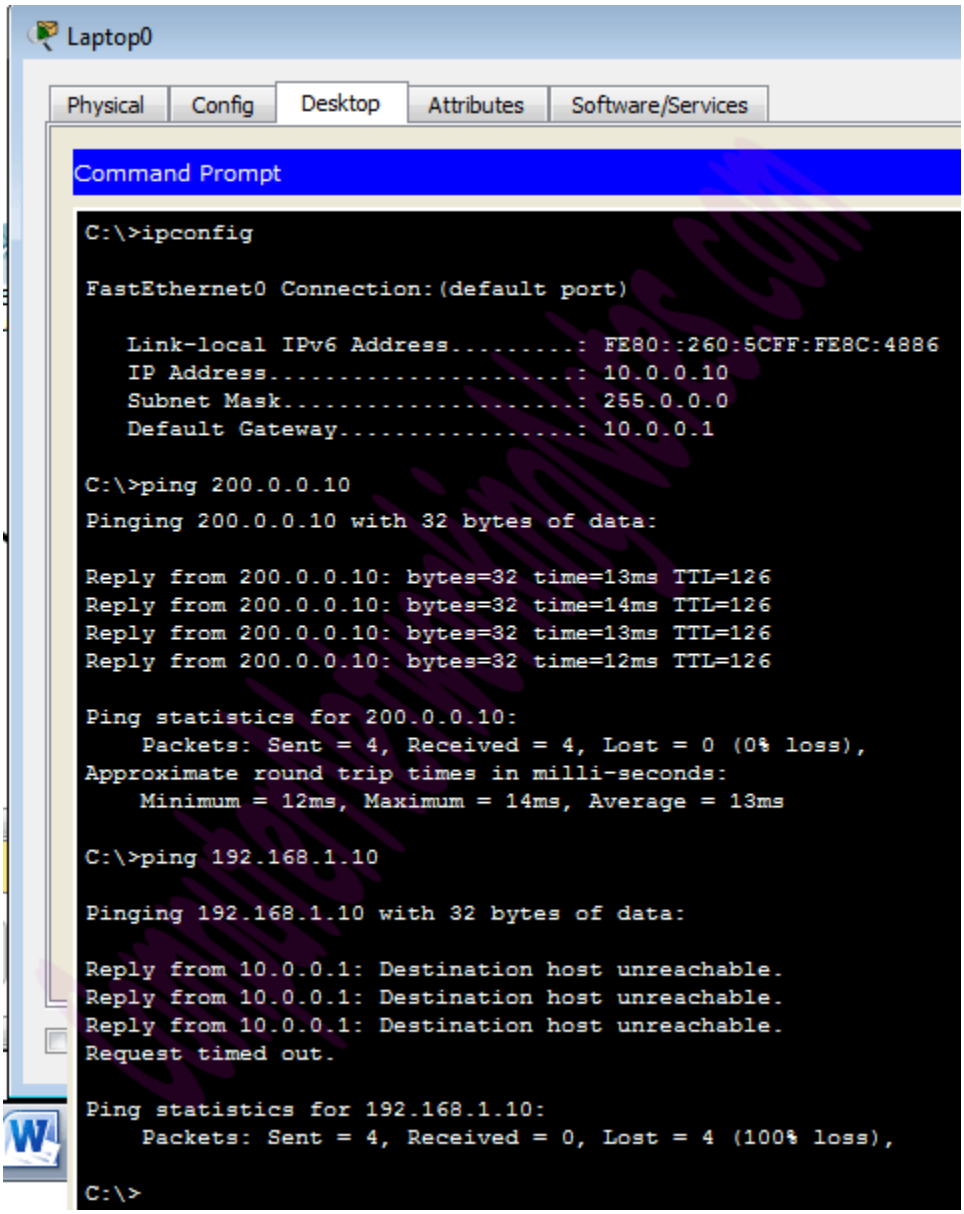
In this lab we configured static NAT on R1 and R2. On R1 we mapped inside local IP address 10.0.0.10 with inside global address 50.0.0.10 while on R2 we mapped inside local IP address 192.168.1.10 with inside global IP address 200.0.0.10.

Device	Inside Local IP Address	Inside Global IP Address
Laptop0	10.0.0.10	50.0.0.10
Server	192.168.1.10	200.0.0.10

To test this setup click Laptop0 and Desktop and click Command Prompt.

- Run **ipconfig** command.
- Run **ping 200.0.0.10** command.

- Run **ping 192.168.1.10** command.

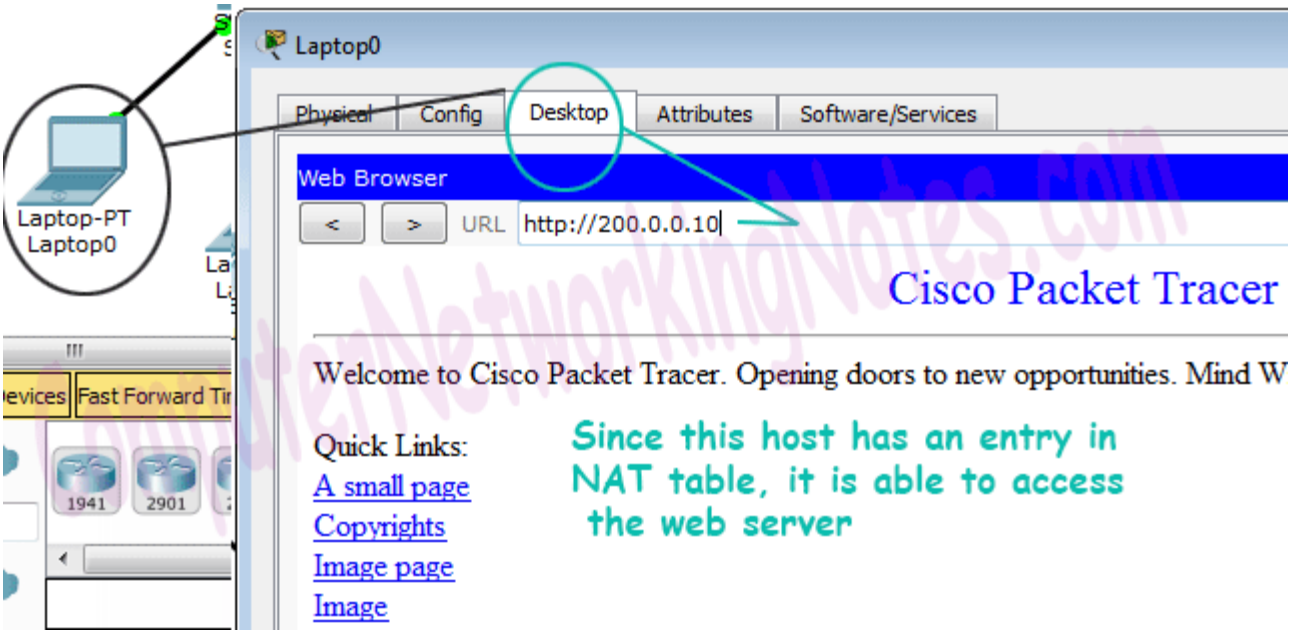


First command verifies that we are testing from correct NAT device.

Second command checks whether we are able to access the remote device or not. A ping reply confirms that we are able to connect with remote device on this IP address.

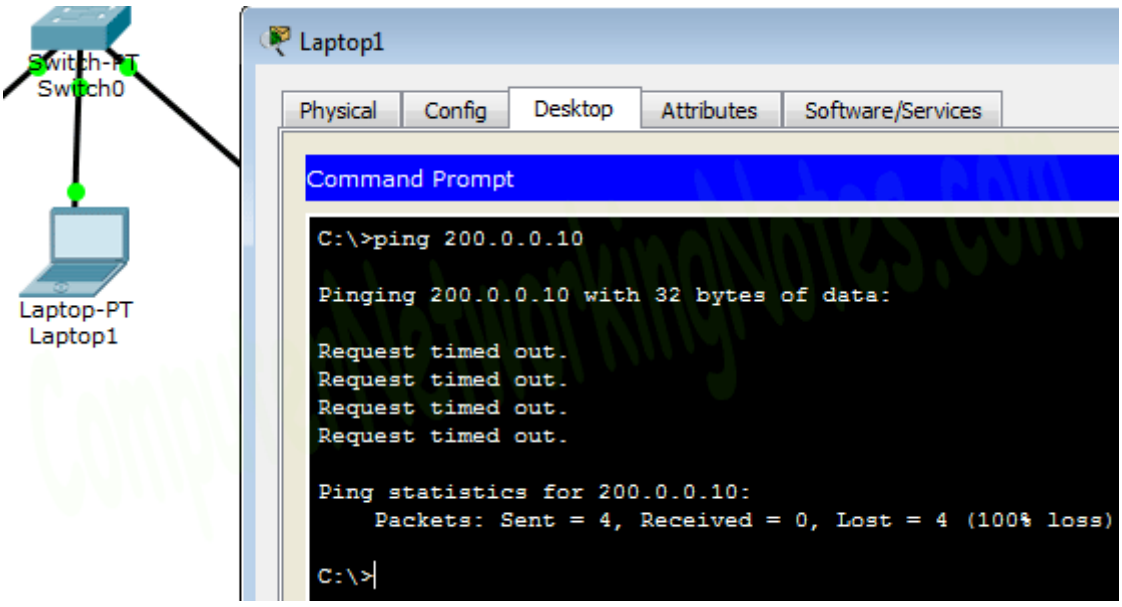
Third command checks whether we are able to access the remote device on its actual IP address or not. A ping error confirms that we are not able to connect with remote device on this IP address.

Let's do one more testing. Click **Laptop0** and click **Desktop** and click **Web server** and access 200.0.0.10.



Above figure confirms that host 10.0.0.10 is able to access the

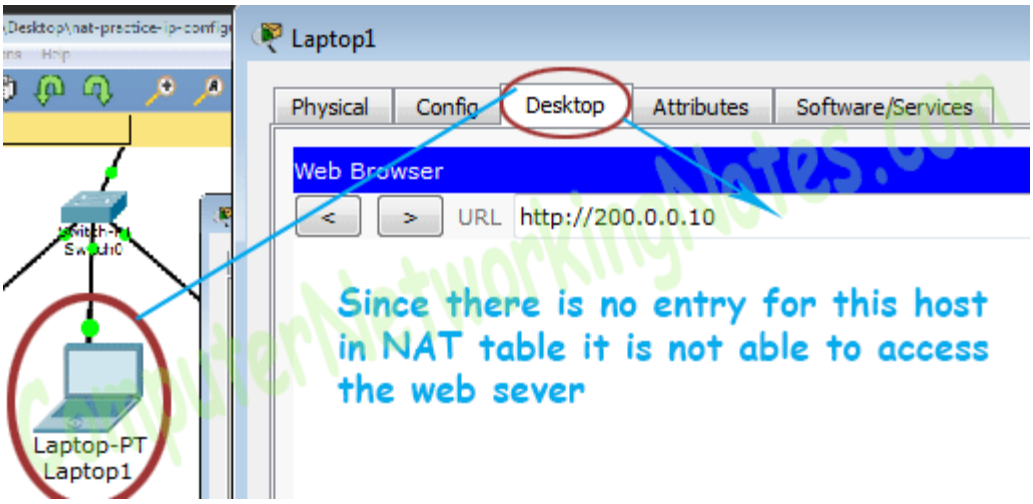
200.0.0.10. Now run **ping 200.0.0.10** command from Laptop1.



Why we are not able to connect with the remote device from this host?

Because we configured NAT only for one host (Laptop0) which IP address is 10.0.0.10. So only the host 10.0.0.10 will be able to access the remote device.

To confirm it again, let's try to access web service from this host.



If you followed this assignment step by step, you should get the same output of testing. Although it's very rare but some time you may get different output. To figure out what went wrong you can use my practice topology with all above configuration. Download my practice topology

Download NAT Practice LAB with Static NAT configuration

We can also verify this translation on router with ***show ip nat translation***

command. Following figure illustrate this translation on router R1.

```
R1#show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
icmp 50.0.0.10:13      10.0.0.10:13       200.0.0.10:13      200.0.0.10:13
icmp 50.0.0.10:14      10.0.0.10:14       200.0.0.10:14      200.0.0.10:14
icmp 50.0.0.10:15      10.0.0.10:15       200.0.0.10:15      200.0.0.10:15
icmp 50.0.0.10:16      10.0.0.10:16       200.0.0.10:16      200.0.0.10:16
tcp 50.0.0.10:1030     10.0.0.10:1030     200.0.0.10:80      200.0.0.10:80
tcp 50.0.0.10:1031     10.0.0.10:1031     200.0.0.10:80      200.0.0.10:80

R1#
```

Following figure illustrate this translation on router R2

```
R2#show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
icmp 200.0.0.10:13     192.168.1.10:13    50.0.0.10:13       50.0.0.10:13
icmp 200.0.0.10:14     192.168.1.10:14    50.0.0.10:14       50.0.0.10:14
icmp 200.0.0.10:15     192.168.1.10:15    50.0.0.10:15       50.0.0.10:15
icmp 200.0.0.10:16     192.168.1.10:16    50.0.0.10:16       50.0.0.10:16
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.10:1030     50.0.0.10:1030
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.10:1031     50.0.0.10:1031

R2#
```

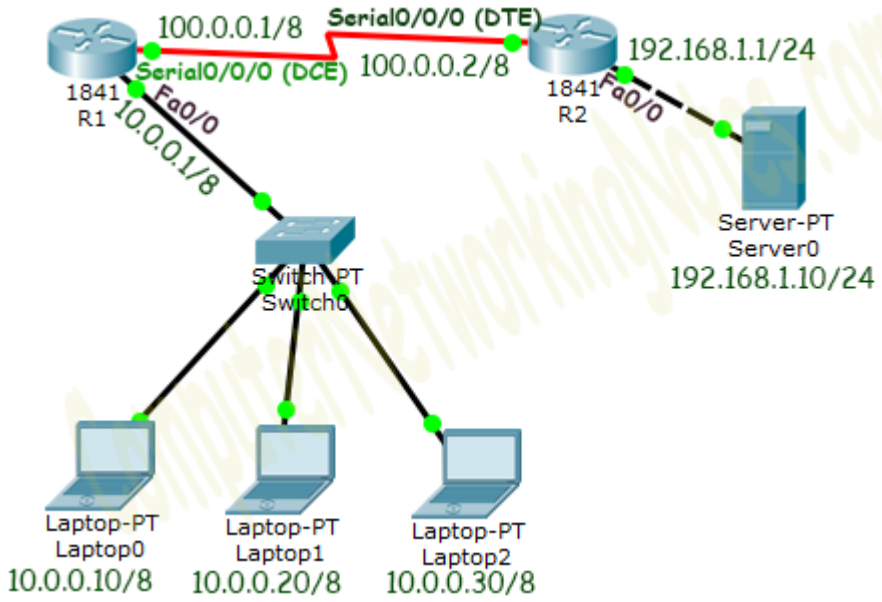
Pay a little bit extra attention on outside local address filed. Have you noticed one interesting feature of NAT in above output? Why actual outside local IP address is not listed in this filed?

The actual IP address is not listed here because router is receiving packets after the translation. From R1's point of view remote device's IP address is 200.0.0.10 while from R2's point of view end device's IP address is 50.0.0.10.

This way if NAT is enabled we would not be able to trace the actual end device.

That's all for this assignment. In next part we will learn dynamic NAT configuration step by step with examples.

How to Configure Dynamic NAT in Cisco Router

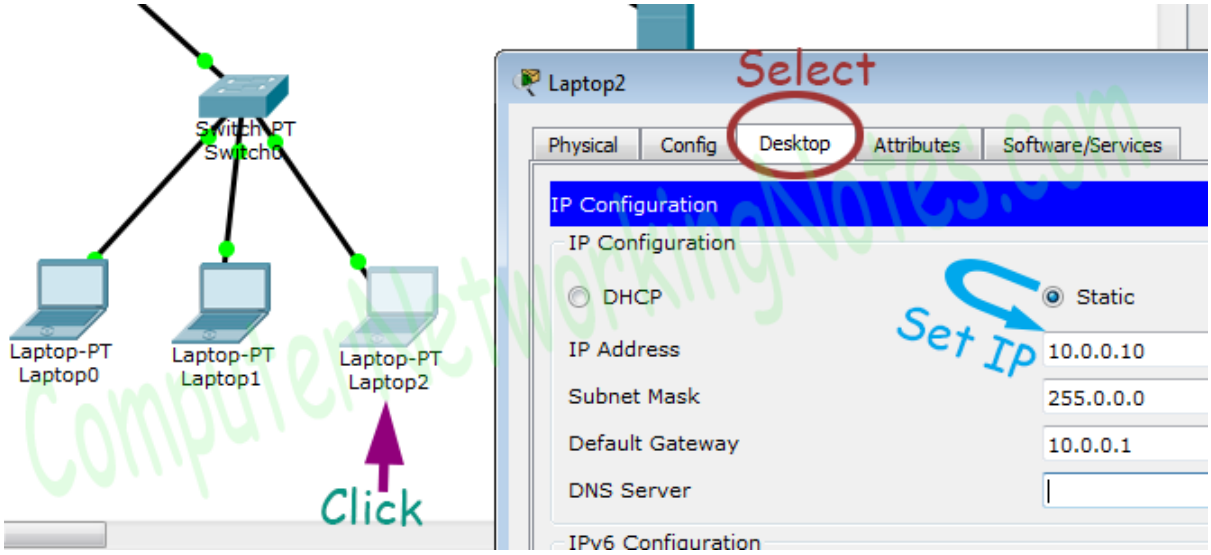


Initial IP Configuration

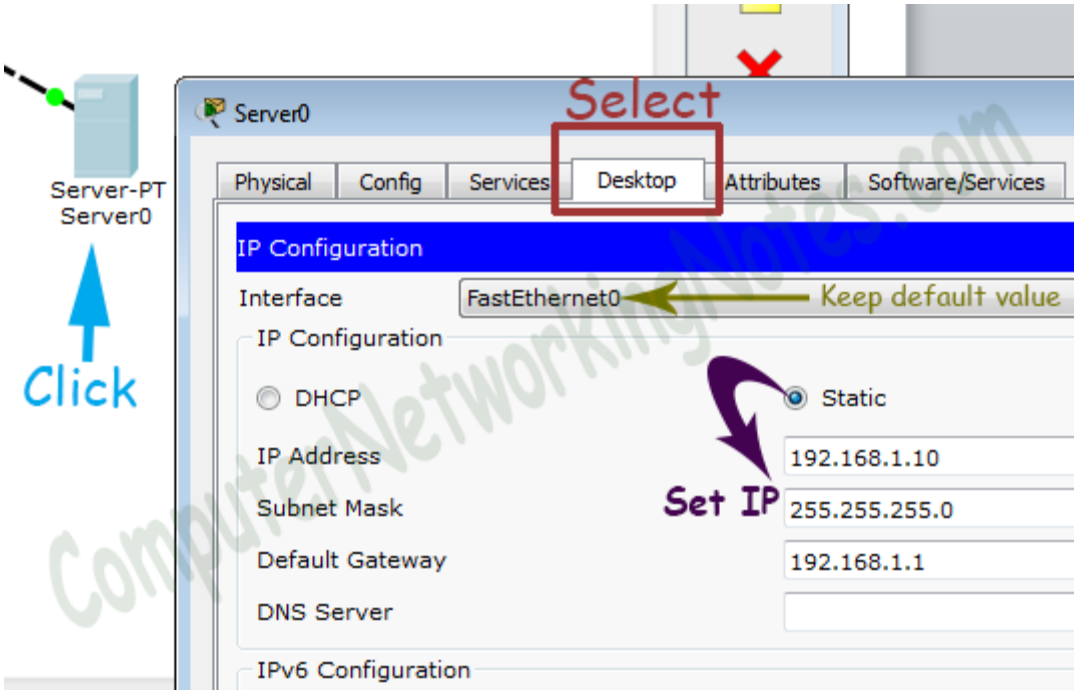
Device / Interface	IP Address	Connected With
Laotop0	10.0.0.10/8	Fa0/0 of R0
Laptop1	10.0.0.20/8	Fa0/0 of R0
Laptop2	10.0.0.30/8	Fa0/0 of R0
Server0	192.168.1.10/24	Fa0/0 of R1
Serial 0/0/0 of R1	100.0.0.1/8	Serial 0/0/0 of R2
Serial 0/0/0 of R2	100.0.0.2/8	Serial 0/0/0 of R2

Alternatively you can download my practice topology which is configured with this initial IP configuration.

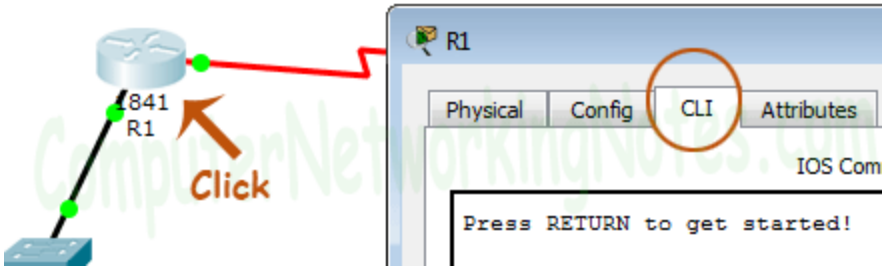
To assign IP address in Laptop click **Laptop** and click **Desktop** and click **IP configuration** and Select **Static** and set **IP address** as given in above table.



Following same way configure IP address in Server.



To configure IP address in Router1 click **Router1** and select **CLI** and press **Enter key**.



Run following commands to set IP address and hostname.

```
Router>enable
Router# configure terminal
Router(config)#
Router(config)#hostname R1
R1(config)#interface FastEthernet0/0
R1(config-if)#ip address 10.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface Serial0/0/0
R1(config-if)#ip address 100.0.0.1 255.0.0.0
R1(config-if)#clock rate 64000
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
```

Same way access the command prompt of R2 and run following commands to set IP address and hostname.

```
Router>enable
Router#configure terminal
Router(config)#hostname R2
R2(config)#interface FastEthernet0/0
R2(config-if)#ip address 192.168.1.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#interface Serial0/0/0
R2(config-if)#ip address 100.0.0.2 255.0.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
```

That's all initial IP configuration we need. Now this topology is ready for the practice of dynamic nat.

Configure Dynamic NAT

Dynamic NAT configuration requires four steps: -

1. Create an access list of IP addresses which need translation
2. Create a pool of all IP address which are available for translation
3. Map access list with pool
4. Define inside and outside interfaces

In first step we will create a standard access list which defines which inside local addresses are permitted to map with inside global address.

To create a standard numbered ACL following global configuration mode command is used:-

```
Router(config)# access-list ACL_Identifier_number permit/deny matching-parameters
```

Let's understand this command and its options in detail.

Router(config)#

This command prompt indicates that we are in global configuration mode.

access-list

Through this parameter we tell router that we are creating or accessing an access list.

ACL_Identifier_number

With this parameter we specify the type of access list. We have two types of access list; standard and extended. Both lists have their own unique identifier numbers. Standard ACL uses numbers range 1 to 99 and 1300 to 1999. We can pick any number from this range to tell the router that we are working with standard ACL. This number is used in groping the conditions under a single ACL. This number is also a unique identifier for this ACL in router.

permit/deny

An ACL condition has two actions; permit and deny. If we use permit keyword, ACL will allow all packets from the source address specified in next parameter. If we use deny keyword, ACL will drop all packets from the source address specified in next parameter.

matching-parameters

This parameter allows us to specify the contents of packet that we want to match. In a standard ACL condition it could be a single source address or a range of addresses. We have three options to specify the source address.

- Any
- host
- A.B.C.D

Any

Any keyword is used to match all sources. Every packet compared against this condition would be matched.

Host

Host keyword is used to match a specific host. To match a particular host, type the keyword host and then the IP address of host.

A.B.C.D

Through this option we can match a single address or a range of addresses. To match a single address, simply type its address. To match a range of addresses, we need to use wildcard mask.

Wildcard mask

Just like subnet mask, wildcard mask is also used to draw a boundary in IP address. Where subnet mask is used to separate network address from host address, wildcard mask is used to distinguish the matching portion from the rest. Wildcard mask is the invert of Subnet mask. Wildcard can be calculated in decimal or in binary from subnet mask.

We have three hosts in lab. Let's create a standard access list which allows two hosts and denies one host.

```
R1(config)#access-list 1 permit 10.0.0.10 0.0.0.0
R1(config)#access-list 1 permit 10.0.0.20 0.0.0.0
R1(config)#access-list 1 deny any
```

To learn standard ACL in detail you can use following tutorial.

Standard ACL Configuration Explained

In second step we define a pool of inside global addresses which are available for translation.

Following command is used to define the NAT pool.

```
Router(config)#ip nat pool [Pool Name] [Start IP address] [End IP address] netmask [Subnet mask]
```

This command accepts four options pool name, start IP address, end IP address and Subnet mask.

Pool Name: - This is the name of pool. We can choose any descriptive name here.

Start IP Address: - First IP address from the IP range which is available for translation.

End IP Address: - Last IP address from the IP range which is available for translation. There is no minimum or maximum criteria for IP range for example we can have a range of single IP address or we can have a range of all IP address from a subnet.

Subnet Mask: - Subnet mask of IP range.

Let's create a pool named ccna with an IP range of two addresses.

```
R1(config)#ip nat pool ccna 50.0.0.1 50.0.0.2 netmask 255.0.0.0
```

This pool consist two class A IP address 50.0.0.1 and 50.0.0.2.

In third step we map access list with pool. Following command will map the access list with pool and configure the dynamic NAT.

```
Router(config)#ip nat inside source list [access list name or number] pool [pool name]
```

This command accepts two options.

Access list name or number: - Name or number the access list which we created in first step.

Pool Name: - Name of pool which we created in second step.

In first step we created a standard access list with number **1** and in second step we created a pool named **ccna**. To configure a dynamic NAT with these options we will use following command.

```
R1(config)#ip nat inside source list 1 pool ccna
```

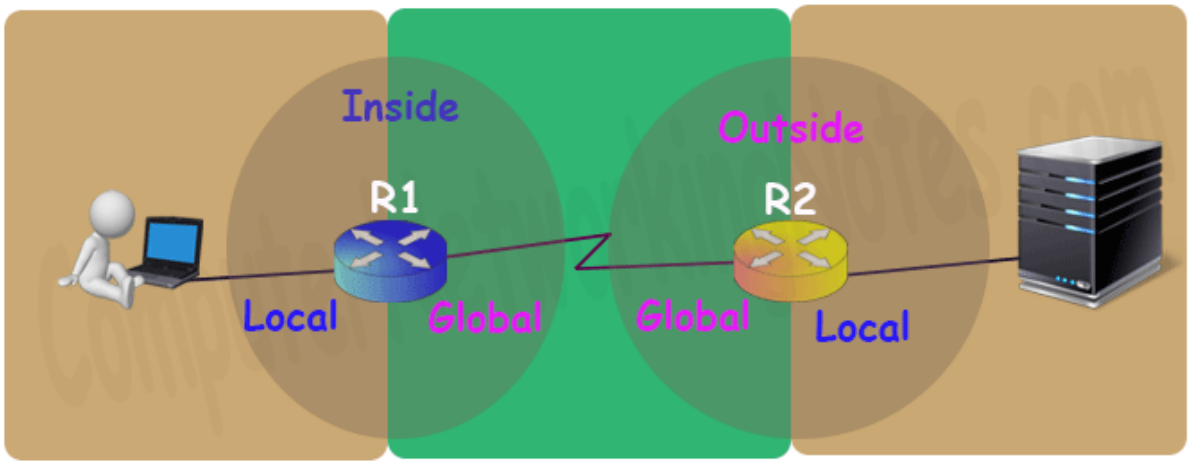
Finally we have to define which interface is connected with local network and which interface is connected with global network.

To define an inside local we use following command

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

Following command defines inside global

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



Let's implement all these commands together and configure the dynamic NAT.

R1 Dynamic NAT Configuration

```
R1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
```

```
R1(config)#access-list 1 permit 10.0.0.10 0.0.0.0
R1(config)#access-list 1 permit 10.0.0.20 0.0.0.0
R1(config)#access-list 1 deny any
R1(config)#ip nat pool ccna 50.0.0.1 50.0.0.2 netmask 255.0.0.0
R1(config)#ip nat inside source list 1 pool ccna
R1(config)#interface FastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface Serial0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
R1(config)#
```

For testing purpose I configured dynamic translations for two addresses only.

On R2 we can keep standard configuration or can configure dynamic NAT as we just did in R1 or can configure static NAT as we learnt in pervious part of this article.

Let's do a quick recap of what we learnt in previous part and configure static NAT on R2.

```
R2>enable
R2#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#ip nat inside source static 192.168.1.10 200.0.0.10
R2(config)#interface Serial 0/0/0
R2(config-if)#ip nat outside
R2(config-if)#exit
R2(config)#interface FastEthernet 0/0
R2(config-if)#ip nat inside
R2(config-if)#exit
R2(config)#
```

To understand above commands in detail please see the second part of this tutorial.

Before we test this lab we need to configure the IP routing. IP routing is the process which allows router to route the packet between different networks. Following tutorial explain routing in detail with examples

Routing Protocols Explained in details

Configure static routing in R1

```
R1(config)#ip route 200.0.0.0 255.255.255.0 100.0.0.2
```

Configure static routing in R2

```
R2(config)#ip route 50.0.0.0 255.0.0.0 100.0.0.1
```

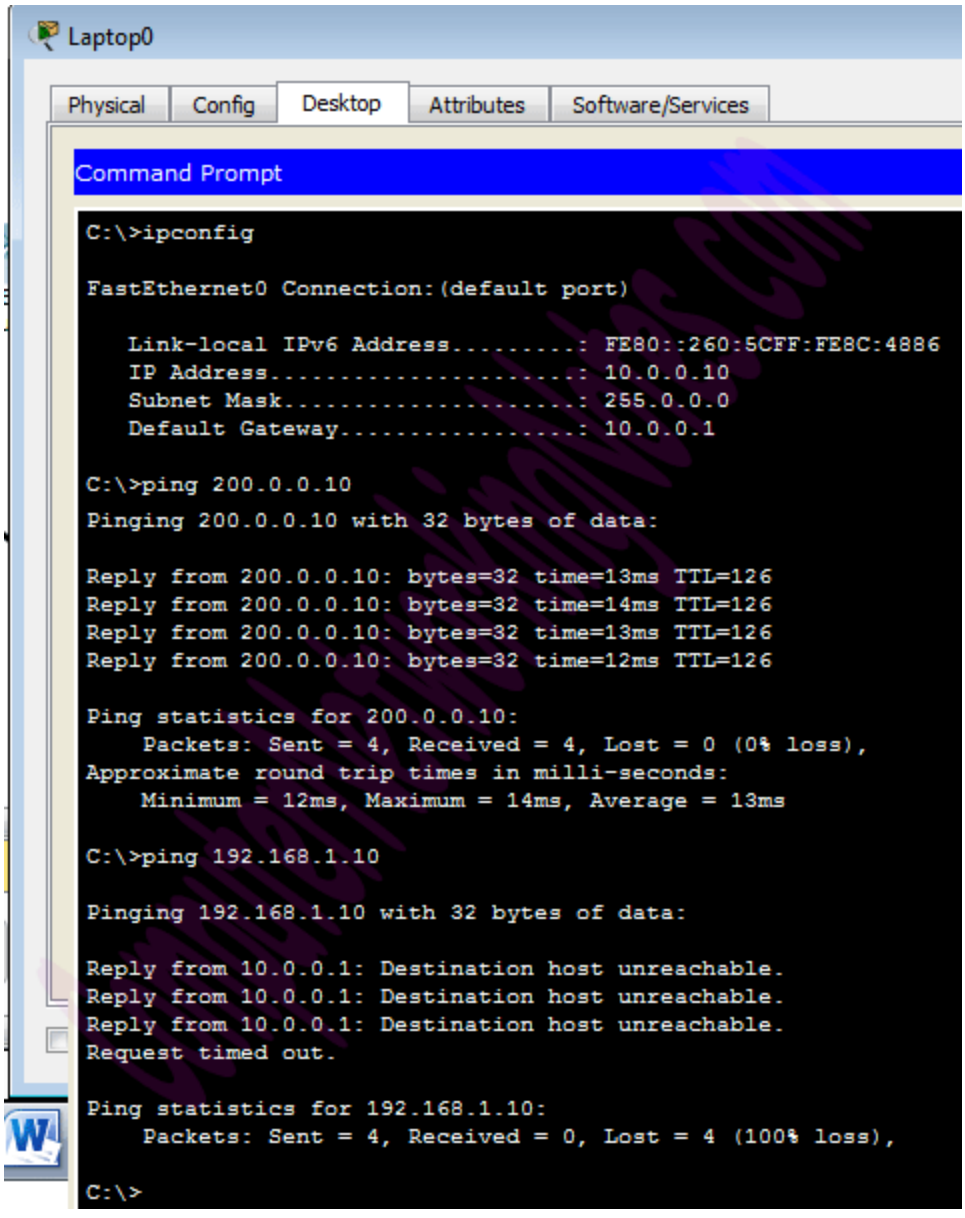
Testing Dynamic NAT Configuration

In this lab we configured dynamic NAT on R1for 10.0.0.10 and 10.0.0.20 and static NAT on R2 for 192.168.1.10.

Device	Inside Local IP Address	Inside Global IP Address
Laptop0	10.0.0.10	50.0.0.1
Laptop1	10.0.0.20	50.0.0.2
Server	192.168.1.10	200.0.0.10

To test this setup click **Laptop0** and **Desktop** and click **Command Prompt**.

- Run **ipconfig** command.
- Run **ping 200.0.0.10** command.
- Run **ping 192.168.1.10** command.

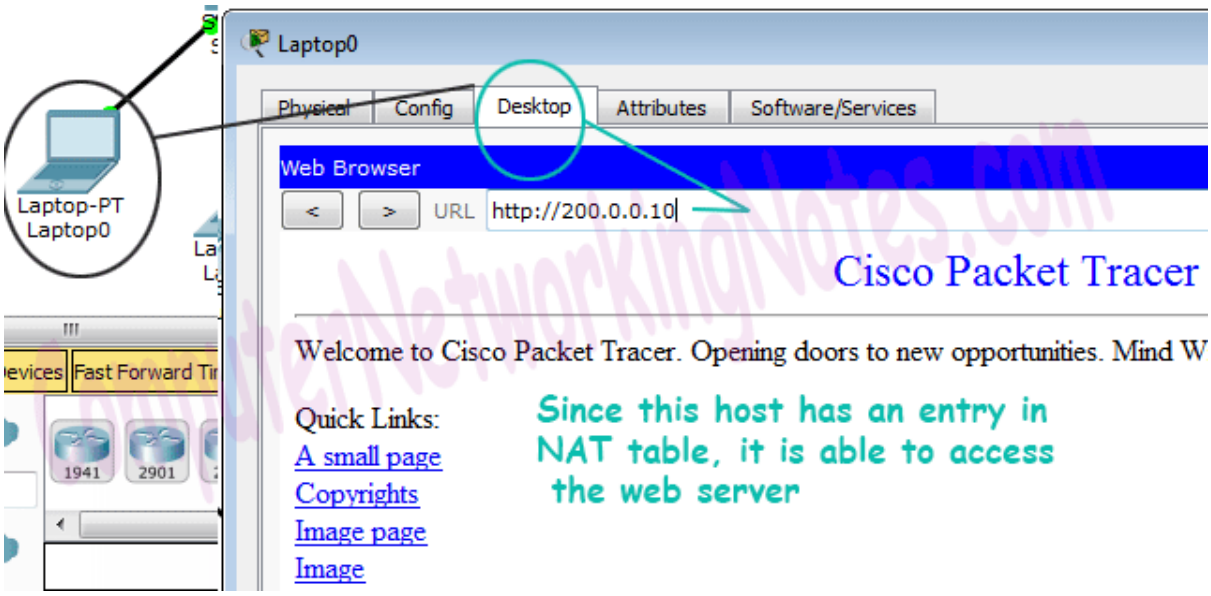


First command verifies that we are testing from correct NAT device.

Second command checks whether we are able to access the remote device or not. A ping reply confirms that we are able to connect with remote device on this IP address.

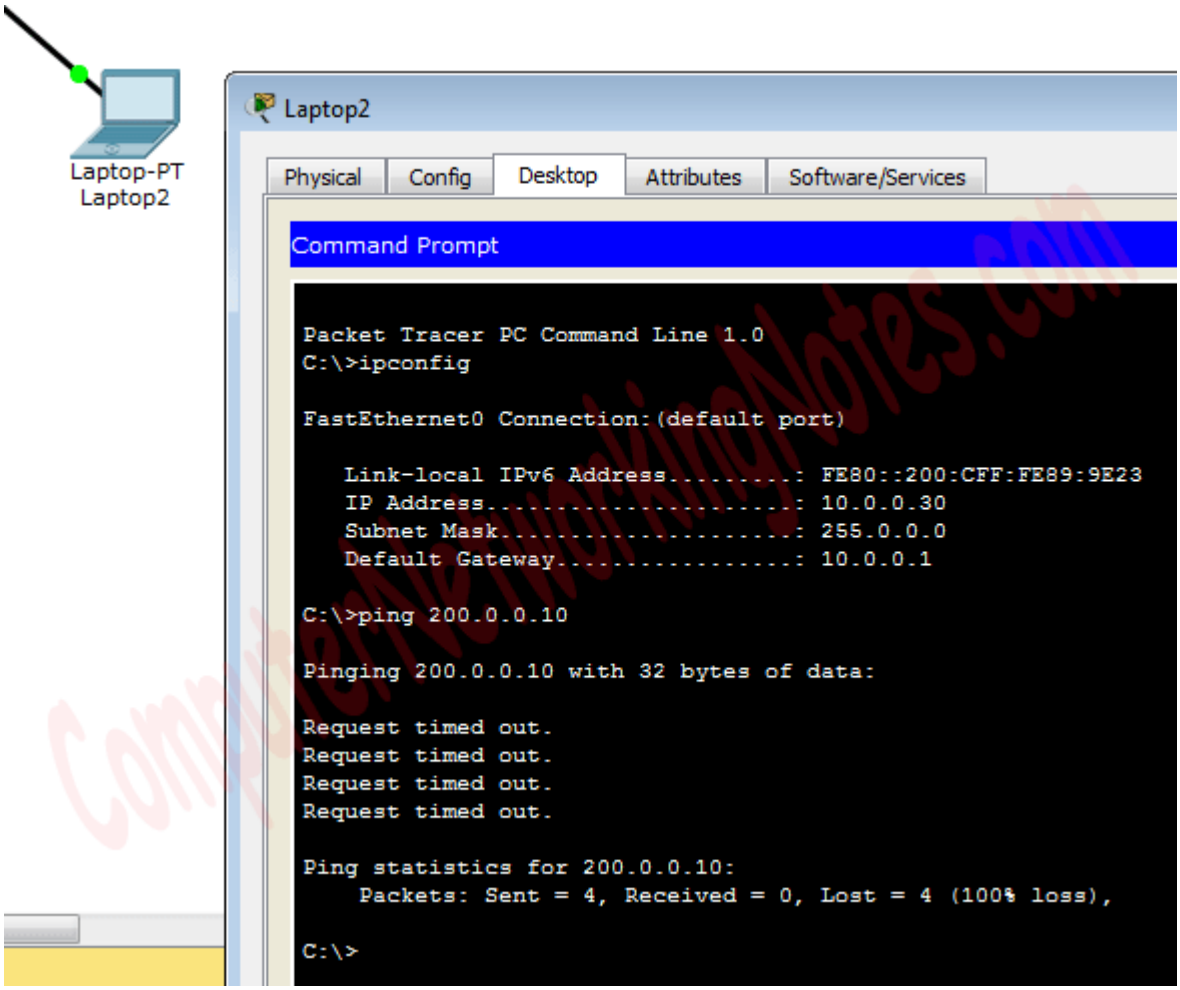
Third command checks whether we are able to access the remote device on its actual IP address or not. A ping error confirms that we are not able to connect with remote device on this IP address.

Let's do one more testing. Close the command prompt and click web server and access 200.0.0.10.

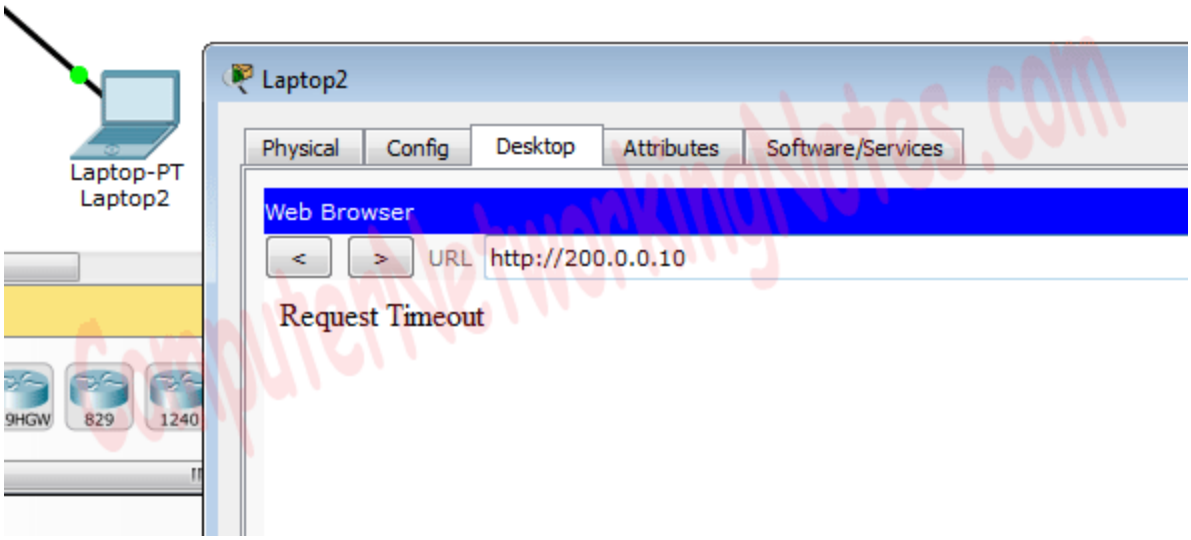


Above figure confirms that host 10.0.0.10 is able to access the 200.0.0.10. You can also do the same testing from Laptop1, result will be same.

Now run ping 200.0.0.10 command from Laptop2.



Close the command prompt and access web server from this host.



Why we are not able to connect with the remote device from this host?

Because we configured NAT only for two hosts (Laptop0 and Laptop1) which IP addresses are 10.0.0.10 and 10.0.0.20. So only the host 10.0.0.10 and 10.0.0.20 will be able to access the remote device.

If you followed this tutorial step by step, you should get the same output of testing. Although it's very rare but some time you may get different output. To figure out what went wrong you can use my practice topology with all above configuration. Download my practice topology.

Download NAT Practice LAB with Dynamic NAT configuration

We can also verify this translation on router with *show ip nat translation* command.

Following figure illustrates this translation on router R1.

```
R1>en
R1#show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
tcp 50.0.0.1:1025      10.0.0.10:1025     200.0.0.10:80      200.0.0.10:80
tcp 50.0.0.2:1025      10.0.0.20:1025     200.0.0.10:80      200.0.0.10:80

R1#
```

We did three tests one from each host, but why only two tests are listed here? Remember in first step we created an access list. Access list filters the unwanted traffic before it reaches to the NAT. We can see how many packets are blocked by ACL with following command

R1#show ip access-lists 1

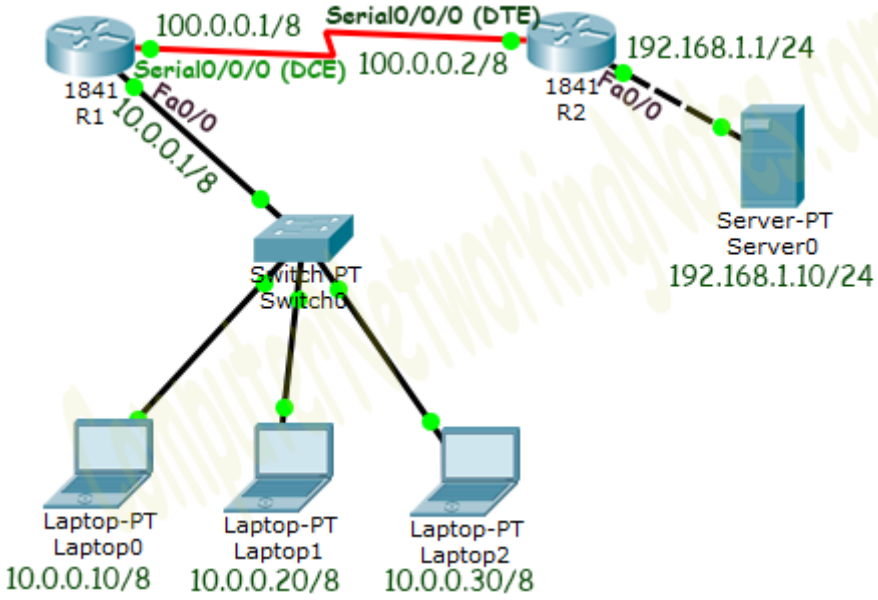
```
R1#show ip access-lists 1
Standard IP access list 1
    permit host 10.0.0.10 (8 match(es))
    permit host 10.0.0.20 (2 match(es))
    deny any (3 match(es))

R1#
```

Basically it is access list which filters the traffic. NAT does not filter any traffic it only translate the address.

Following figure illustrate NAT translation on router R2

```
R2>enable
R2#show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
---  200.0.0.10        192.168.1.10       ---                ---
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.1:1025      50.0.0.1:1025
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.2:1025      50.0.0.2:1025

R2#
```

PAT (NAT Overload)

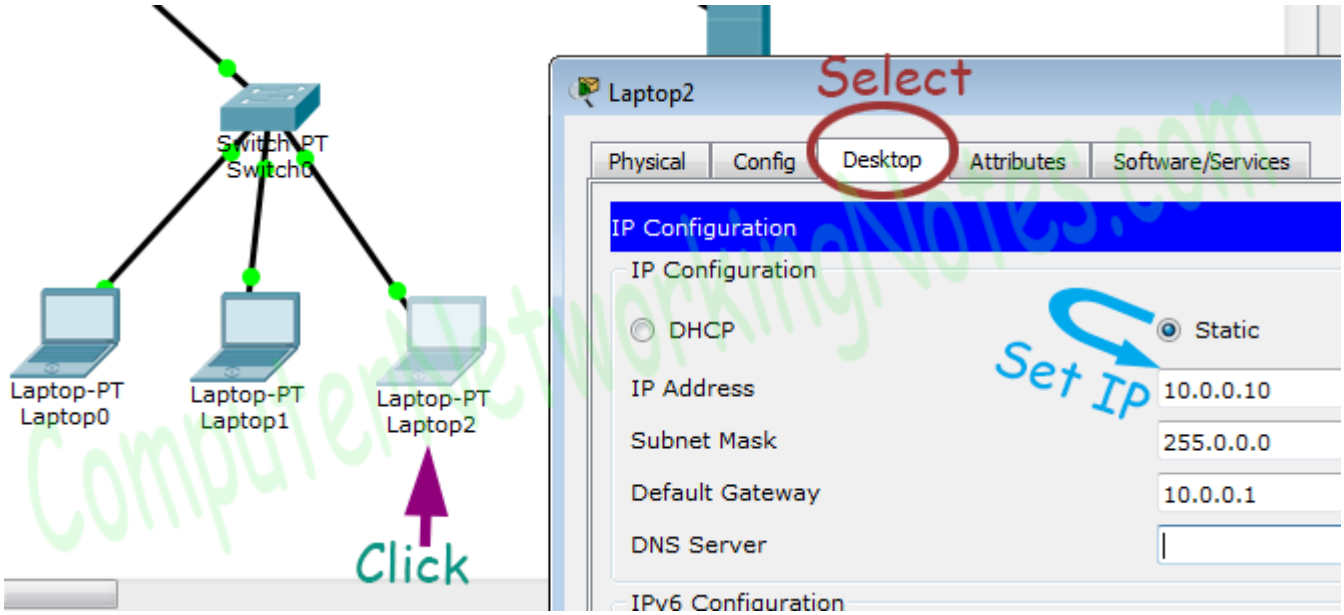


Initial IP Configuration

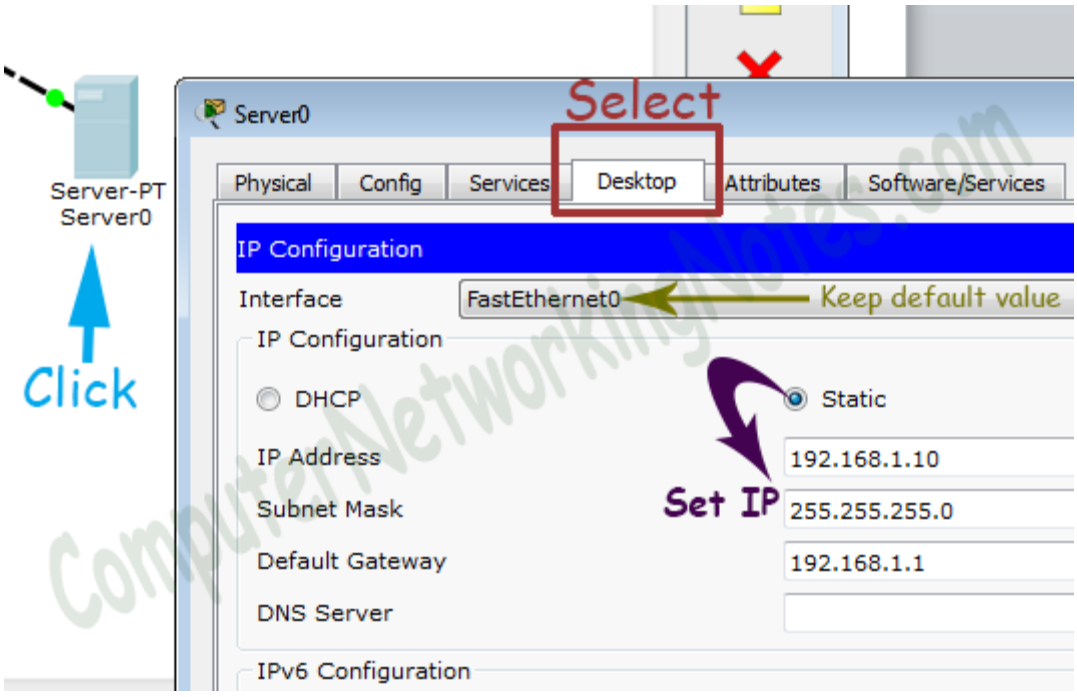
Device / Interface	IP Address	Connected With
Laotop0	10.0.0.10/8	Fa0/0 of R0
Laptop1	10.0.0.20/8	Fa0/0 of R0
Laptop2	10.0.0.30/8	Fa0/0 of R0
Server0	192.168.1.10/24	Fa0/0 of R1
Serial 0/0/0 of R1	100.0.0.1/8	Serial 0/0/0 of R2
Serial 0/0/0 of R2	100.0.0.2/8	Serial 0/0/0 of R2

If you are following this tutorial on my practice topology, skip this IP configuration section as that topology is already configured with this initial IP configuration.

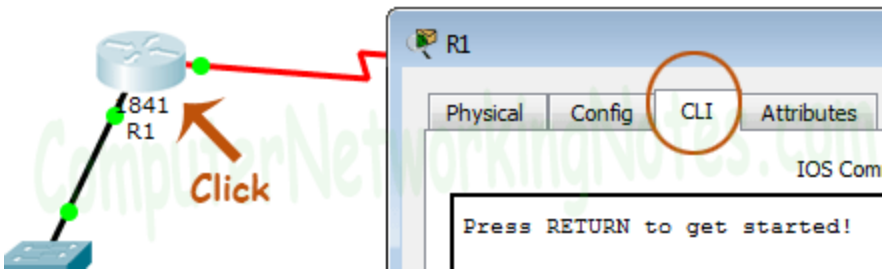
To assign IP address in Laptop click **Laptop** and click **Desktop** and click **IP configuration** and Select **Static** and set **IP address** as given in above table.



Following same way configure IP address in Server.



To configure IP address in Router1 click **Router1** and select **CLI** and press **Enter key**.



Run following commands to set IP address and hostname.

```
Router>enable
```

```
Router# configure terminal
Router(config)#
Router(config)#hostname R1
R1(config)#interface FastEthernet0/0
R1(config-if)#ip address 10.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface Serial0/0/0
R1(config-if)#ip address 100.0.0.1 255.0.0.0
R1(config-if)#clock rate 64000
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
```

Same way access the command prompt of R2 and run following commands to set IP address and hostname.

```
Router>enable
Router#configure terminal
Router(config)#hostname R2
R2(config)#interface FastEthernet0/0
R2(config-if)#ip address 192.168.1.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#interface Serial0/0/0
R2(config-if)#ip address 100.0.0.2 255.0.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
```

That's all initial IP configuration we need. Now this topology is ready for the practice of pat.

Configure PAT (NAT Overload)

PAT configuration requires four steps: -

1. Create an access list of IP addresses which need translation
2. Create a pool of all IP address which are available for translation
3. Map access list with pool
4. Define inside and outside interfaces

In first step we will create a standard access list which defines which inside local addresses are permitted to map with inside global address.

To create a standard numbered ACL following global configuration mode command is used:-

```
Router(config)# access-list ACL_Identifier_number permit/deny matching-parameters
```

Let's understand this command and its options in detail.

Router(config)#

This command prompt indicates that we are in global configuration mode.

access-list

Through this parameter we tell router that we are creating or accessing an access list.

ACL_Identifier_number

With this parameter we specify the type of access list. We have two types of access list; standard and extended. Both lists have their own unique identifier numbers. Standard ACL uses numbers range 1 to 99 and 1300 to 1999. We can pick any number from this range to tell the router that we are working with standard ACL. This number is used in groping the conditions under a single ACL. This number is also a unique identifier for this ACL in router.

permit/deny

An ACL condition has two actions; permit and deny. If we use permit keyword, ACL will allow all packets from the source address specified in next parameter. If we use deny keyword, ACL will drop all packets from the source address specified in next parameter.

matching-parameters

This parameter allows us to specify the contents of packet that we want to match. In a standard ACL condition it could be a single source address or a range of addresses. We have three options to specify the source address.

- Any
- host
- A.B.C.D

Any

Any keyword is used to match all sources. Every packet compared against this condition would be matched.

Host

Host keyword is used to match a specific host. To match a particular host, type the keyword host and then the IP address of host.

A.B.C.D

Through this option we can match a single address or a range of addresses. To match a single address, simply type its address. To match a range of addresses, we need to use wildcard mask.

Wildcard mask

Just like subnet mask, wildcard mask is also used to draw a boundary in IP address. Where subnet mask is used to separate network address from host address, wildcard mask is used to distinguish the matching portion from the rest. Wildcard mask is the invert of Subnet mask. Wildcard can be calculated in decimal or in binary from subnet mask.

We have three hosts in lab. Let's create a standard access list which allows two hosts and denies one host.

```
R1(config)#access-list 1 permit 10.0.0.10 0.0.0.0
R1(config)#access-list 1 permit 10.0.0.20 0.0.0.0
R1(config)#access-list 1 deny any
```

To learn standard ACL in detail you can use following tutorial.

Standard ACL Explained with Examples

In second step we define a pool of inside global addresses which are available for translation.

Following command is used to define the NAT pool.

```
Router(config)#ip nat pool [Pool Name] [Start IP address] [End IP address] netmask [Subnet mask]
```

This command accepts four options pool name, start IP address, end IP address and Subnet mask.

Pool Name: - This is the name of pool. We can choose any descriptive name here.

Start IP Address: - First IP address from the IP range which is available for translation.

End IP Address: - Last IP address from the IP range which is available for translation. There is no minimum or maximum criteria for IP range for example we can have a range of single IP address or we can have a range of all IP address from a subnet.

Subnet Mask: - Subnet mask of IP range.

Let's create a pool named ccna with a single IP address.

```
R1(config)#ip nat pool ccna 50.0.0.1 50.0.0.1 netmask 255.0.0.0
```

In third step we map access list with pool. Following command will map the access list with pool and configure the PAT.

```
Router(config)#ip nat inside source list [access list name or number] pool [pool name]overload
```

This command accepts two options.

Access list name or number: - Name or number the access list which we created in first step.

Pool Name: - Name of pool which we created in second step.

In first step we created a standard access list with number 1 and in second step we created a pool named ccna. To configure a PAT with these options we will use following command.

```
R1(config)#ip nat inside source list 1 pool ccna overload
```

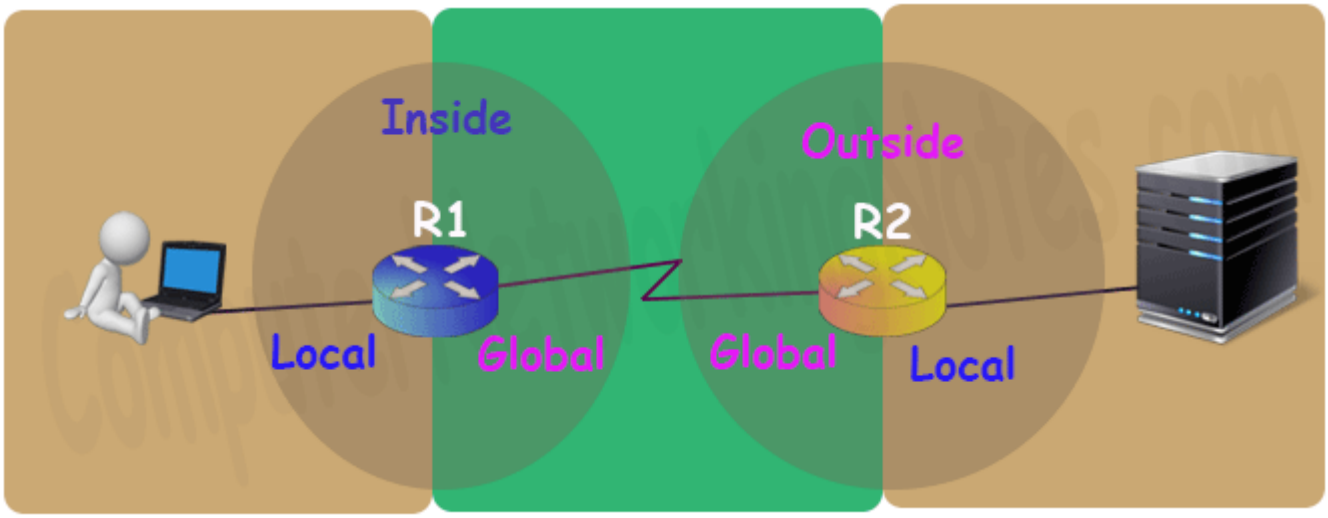
Finally we have to define which interface is connected with local network and which interface is connected with global network.

To define an inside local we use following command

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

Following command defines inside global

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



Let's implement all these commands together and configure the PAT.

R1 PAT (NAT Overload) Configuration

```
R1>enable
R1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#access-list 1 permit 10.0.0.10 0.0.0.0
R1(config)#access-list 1 permit 10.0.0.20 0.0.0.0
R1(config)#access-list 1 deny any
R1(config)#ip nat pool ccna 50.0.0.1 50.0.0.1 netmask 255.0.0.0
R1(config)#ip nat inside source list 1 pool ccna overload
R1(config)#interface FastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface Serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```

```
R1(config)#
```

For testing purpose I configured pat translations for two addresses only.

On R2 we can keep standard configuration or can configure dynamic NAT or can configure static NAT as we learnt in pervious parts of this article.

Let's do a quick recap of what we learnt in previous part and configure static NAT on R2.

```
R2>enable
R2#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#ip nat inside source static 192.168.1.10 200.0.0.10
R2(config)#interface Serial 0/0/0
R2(config-if)#ip nat outside
R2(config-if)#exit
R2(config)#interface FastEthernet 0/0
R2(config-if)#ip nat inside
R2(config-if)#exit
R2(config)#
```

To understand above commands in detail please see the second part of this tutorial.

Before we test this lab we need to configure the IP routing. IP routing is the process which allows router to route the packet between different networks. Following tutorial explain routing in detail with examples

Routing Protocol Explained

Configure static routing in R1

```
R1(config)#ip route 200.0.0.0 255.255.255.0 100.0.0.2
```

Configure static routing in R2

```
R2(config)#ip route 50.0.0.0 255.0.0.0 100.0.0.1
```

Testing PAT Configuration

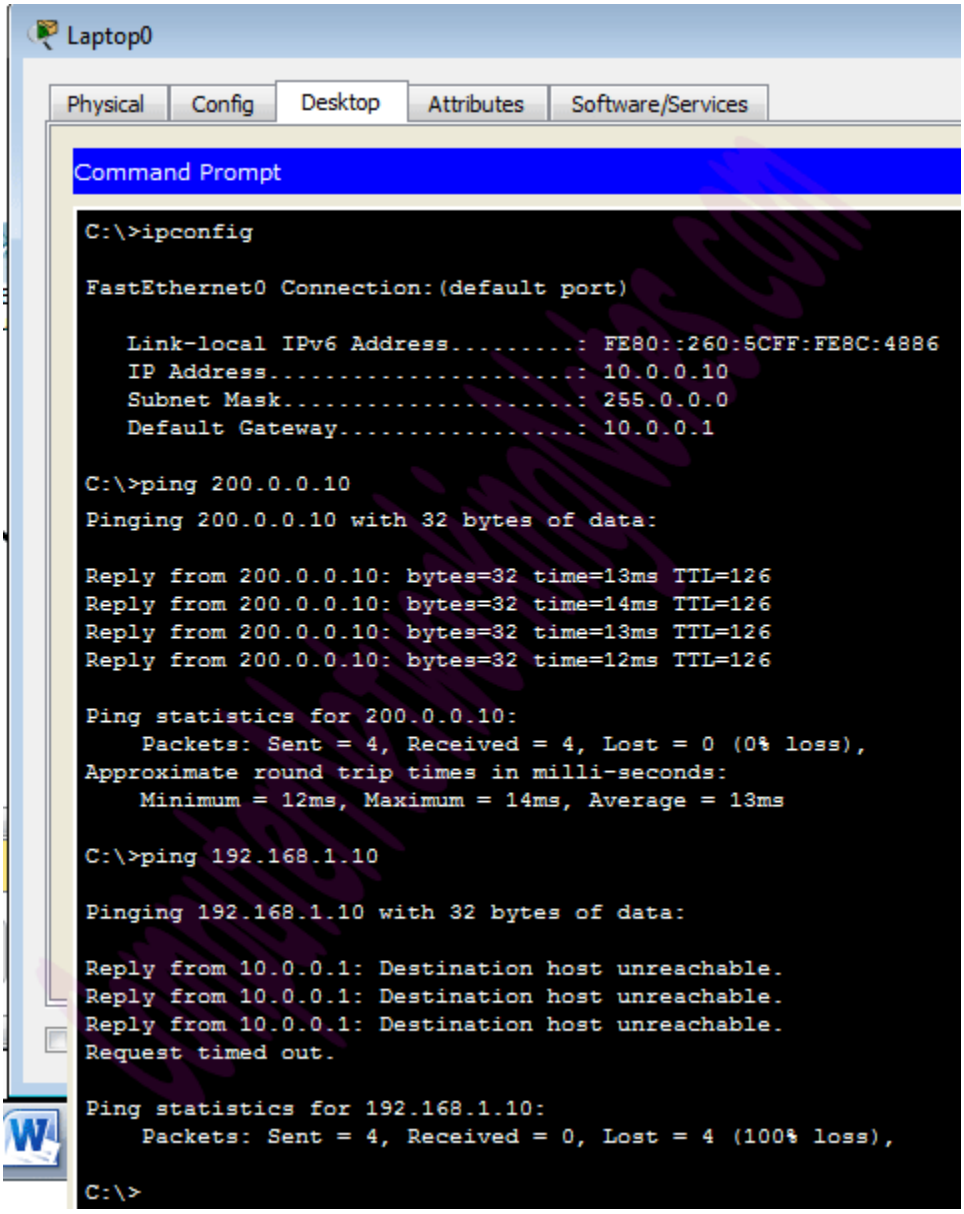
In this lab we configured PAT on R1for 10.0.0.10 and 10.0.0.20 and static NAT on R2 for 192.168.1.10.

Device	Inside Local IP Address	Inside
Laptop0	10.0.0.10	50.0.0
Laptop1	10.0.0.20	50.0.0
Server	192.168.1.10	200.0.

To test this setup click **Laptop0** and **Desktop** and click **Command Prompt**.

- Run **ipconfig** command.
- Run **ping 200.0.0.10** command.

- Run **ping 192.168.1.10** command.

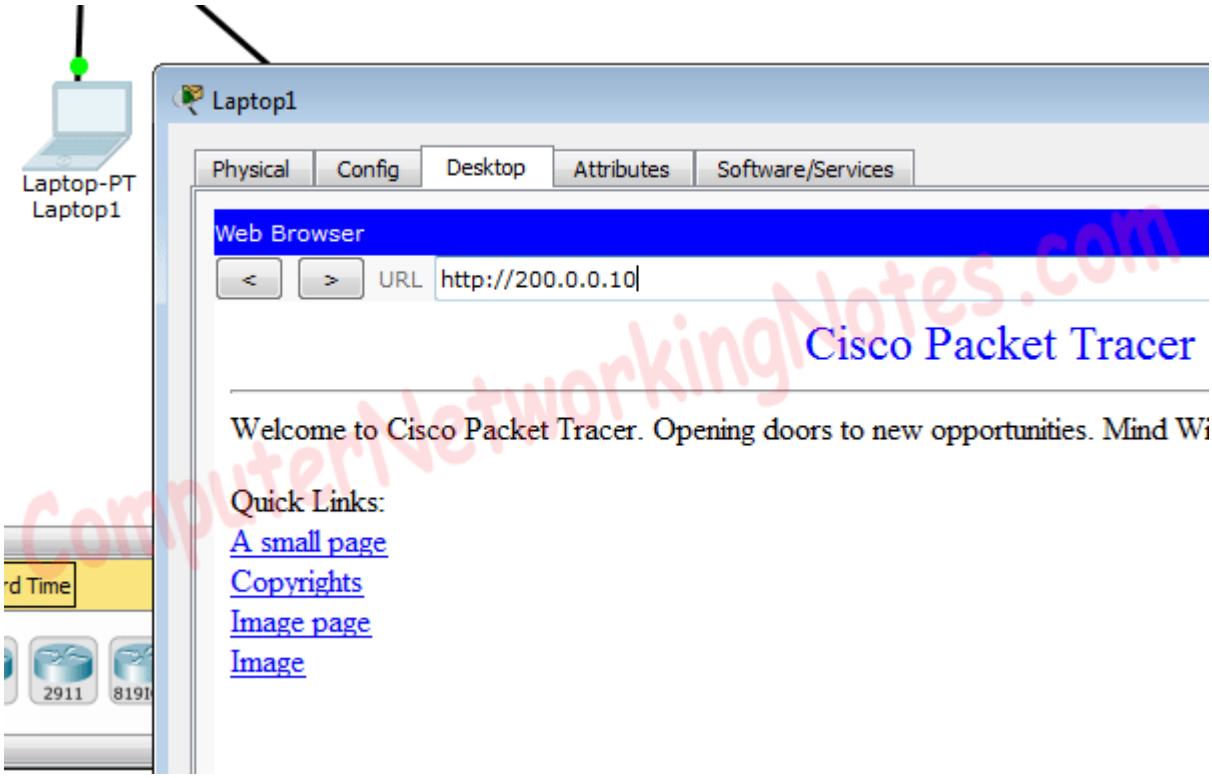


First command verifies that we are testing from correct NAT device.

Second command checks whether we are able to access the remote device or not. A ping reply confirms that we are able to connect with remote device on this IP address.

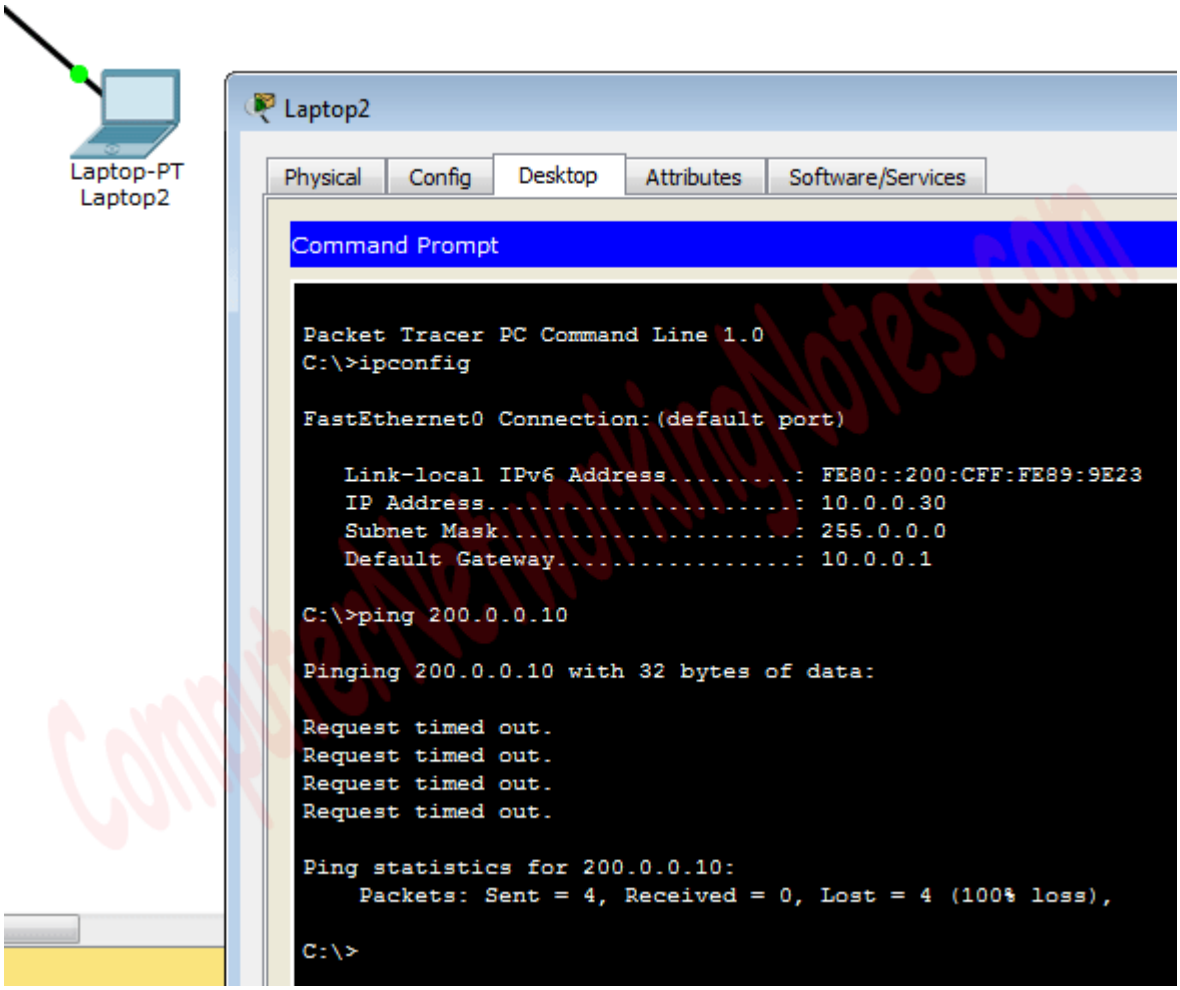
Third command checks whether we are able to access the remote device on its actual IP address or not. A ping error confirms that we are not able to connect with remote device on this IP address.

Let's do one more testing. Close the command prompt and click web server and access 200.0.0.10.

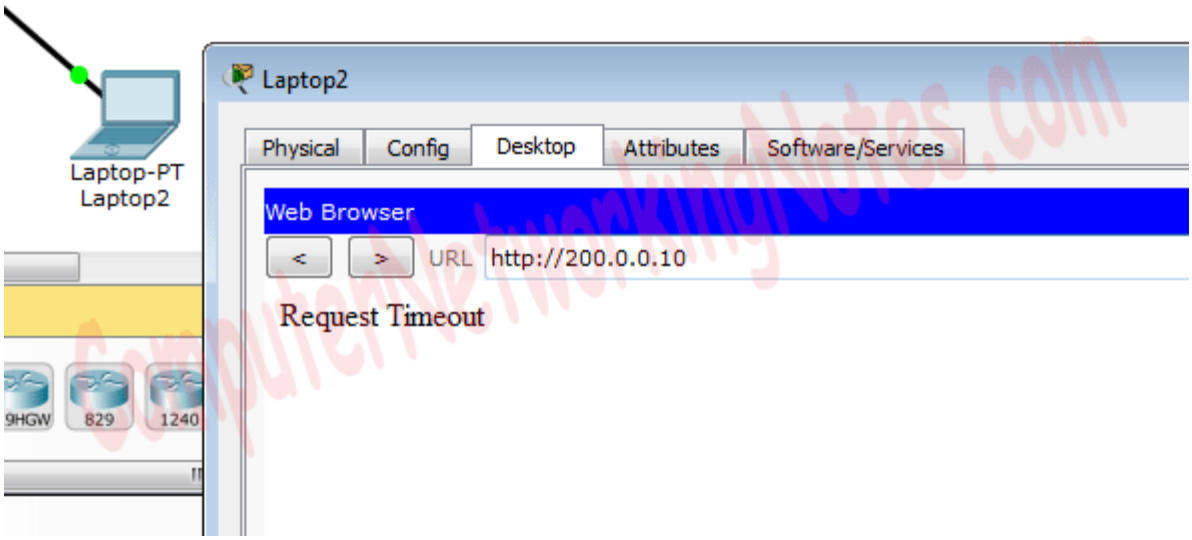


Above figure confirms that host 10.0.0.10 is able to access the 200.0.0.10. You can also do the same testing from Laptop1, result will be same.

Now run **ping 200.0.0.10** command from Laptop2.



Close the command prompt and access web server from this host.



Why we are not able to connect with the remote device from this host?

Because we configured PAT only for two hosts (Laptop0 and Laptop1) which IP addresses are 10.0.0.10 and 10.0.0.20. So only the host 10.0.0.10 and 10.0.0.20 will be able to access the remote device.

If you followed this tutorial step by step, you should get the same output of testing. Although it's very rare but some time you may get different output. To figure out what went wrong you can use my practice topology with all above configuration. Download my practice topology

Download NAT Practice LAB with PAT configuration

We can also verify this translation on router with ***show ip nat translation*** command.

Following figure illustrate this translation on router R1.

```
R1#show ip nat translation
Pro  Inside global     Inside local       Outside local      Outside global
icmp 50.0.0.1:1        10.0.0.20:1        200.0.0.10:1       200.0.0.10:1
icmp 50.0.0.1:2        10.0.0.20:2        200.0.0.10:2       200.0.0.10:2
icmp 50.0.0.1:3        10.0.0.20:3        200.0.0.10:3       200.0.0.10:3
icmp 50.0.0.1:4        10.0.0.20:4        200.0.0.10:4       200.0.0.10:4
tcp 50.0.0.1:1024      10.0.0.10:1025     200.0.0.10:80      200.0.0.10:80
tcp 50.0.0.1:1025      10.0.0.20:1025     200.0.0.10:80      200.0.0.10:80

R1#
```

As we can see in above output same inside global IP address is used to translate all the inside local IP addresses. For each inside local IP address a unique port number is used.

Following figure illustrate NAT translation on router R2

```
R2#show ip nat translation
Pro  Inside global     Inside local       Outside local      Outside global
icmp 200.0.0.10:1      192.168.1.10:1     50.0.0.1:1         50.0.0.1:1
icmp 200.0.0.10:2      192.168.1.10:2     50.0.0.1:2         50.0.0.1:2
icmp 200.0.0.10:3      192.168.1.10:3     50.0.0.1:3         50.0.0.1:3
icmp 200.0.0.10:4      192.168.1.10:4     50.0.0.1:4         50.0.0.1:4
---  200.0.0.10        192.168.1.10       ---                ---
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.1:1024      50.0.0.1:1024
tcp 200.0.0.10:80      192.168.1.10:80    50.0.0.1:1025      50.0.0.1:1025

R2#
```

In above output the Outside global field also confirms that all packets are coming from single IP address.

That

ASSIGNMENT NO : 2

Using a Network Simulator (e.g. packet tracer) Configure Routing Protocols,

a) Configure EIGRP – Explore Neighbor-ship Requirements and Conditions, its K Values Metrics Assignment and Calculation.

b) OSPF – Explore Neighbor-ship Condition and Requirement, Neighbor-ship states, OSPF Metric Cost Calculation.

c) WLAN with static IP addressing and DHCP with MAC security and filters.

Features and characteristics of EIGRP

- It is a Cisco Proprietary routing protocol.
- It is based on IGRP Routing protocol.
- It is an enhanced version of IGRP (Interior Gateway Routing Protocol) protocol.
- In comparison of IGRP it provides faster convergence times, superior handling of routing loops and improved scalability.
- It was released in 1994.
- It is a hybrid routing protocol.
- It has characteristics of both distance vector and link state protocols.
- It uses DUAL (Diffusing Update Algorithm) algorithm to select the best path.
- It uses RTP (Reliable Transport Protocol) to communicate with neighbors.
- It uses multicast for routing updates.
- It supports IP [Both IPv4 and IPV6], Apple Talk and IPX routed protocols.
- It includes subnet mask information in routing updates.
- It supports route summarization and discontiguous networks.
- It supports VLSM/CIDR.
- It supports load balancing across the six routes for a single destination.
- It supports trigger updates.

From introduction to till the preparation of this tutorial, EIGRP is ruling the world of routing protocols. The only negative about EIGRP was Cisco kept this protocol as proprietary protocol. In order to run this protocol, we had to buy all routers from Cisco. This thing was changed a little in 2013 when partial functionality of EIGRP was converted in open standard. Now we can also buy routers from other vendors along with Cisco, still running EIGRP on all routers.

Since EIGRP is hybrid protocol, it has advantages of both link state and distance vector protocol. It uses composite metric calculation formula to select the best route for destination. It sends partial or full update only when something is change in network. It maintains three tables for ultra-fast convergence.

1. Neighbor Table
2. Topology Table
3. Routing Table

Neighbor Table iNFORMATION OF NEIGHBOURS

EIGRP shares routing information only with neighbors. To know who the neighbors are, it uses

neighbor table. When a new neighbor is discovered, EIGRP would add its address and interface on which neighbor is connected in neighbor table. EIGRP uses separate neighbor table for each routed protocol.

Topology Table

to store all routes which it learned from neighbors

EIGRP uses this table to store all routes which it learned from neighbors. It contains a list of all destinations and routes advertised by neighboring routers. EIGRP selects single best route for each destination from this list. That route goes in routing table. Remaining routes are marked

as backup routes. EIGRP refers selected route as Successor and backup route as Feasible Successor. EIGRP uses separate topology table for each routed protocol.

Routing Table

EIGRP stores single best (Successor) route for each destination in this table. Router uses this table to forward the packet. There is a separate routing table for each routed protocol.

Protocol Dependent Modules

PDMs are the special feature of EIGPR. Through these modules EIGRP supports multiple network layer protocols. It maintains separate tables for separate routed (Network Layer) protocols. For example if you are using both (IPv4 and IPv6) versions of IP protocol, it will maintain separate IPv4/EIGRP and IPv6/EIGRP tables.

Metric

EIGRP uses metric to select the best route from all available routes for destination. Metric has five components.

- Bandwidth
- Load
- Delay
- Reliability
- MTU

From these only bandwidth and delay are by default enabled.

RTP

EIGRP uses RTP to communicate with other EIGRP speaking routers. RTP (Reliable Transport Protocol) uses multicast and unicast to exchange the data with neighbors. It uses class D address 224.0.0.10 for multicast. It keeps track of each multicast it sends out. EIGRP maintains a list of the neighbors who have replied. If it doesn't receive a reply from any neighbor, RTP will resend the same data using unicast. It will make 16 unicast attempts before declaring neighbor is dead.

DUAL

EIGRP uses DUAL (Diffusing Update Algorithm) to provide the fastest route convergence among all protocols. Route convergence includes:-

- Selecting best route from all available routes
- Supporting VLSMs
- Dynamically recovering from route failure
- Finding an alternative route if primary route goes down

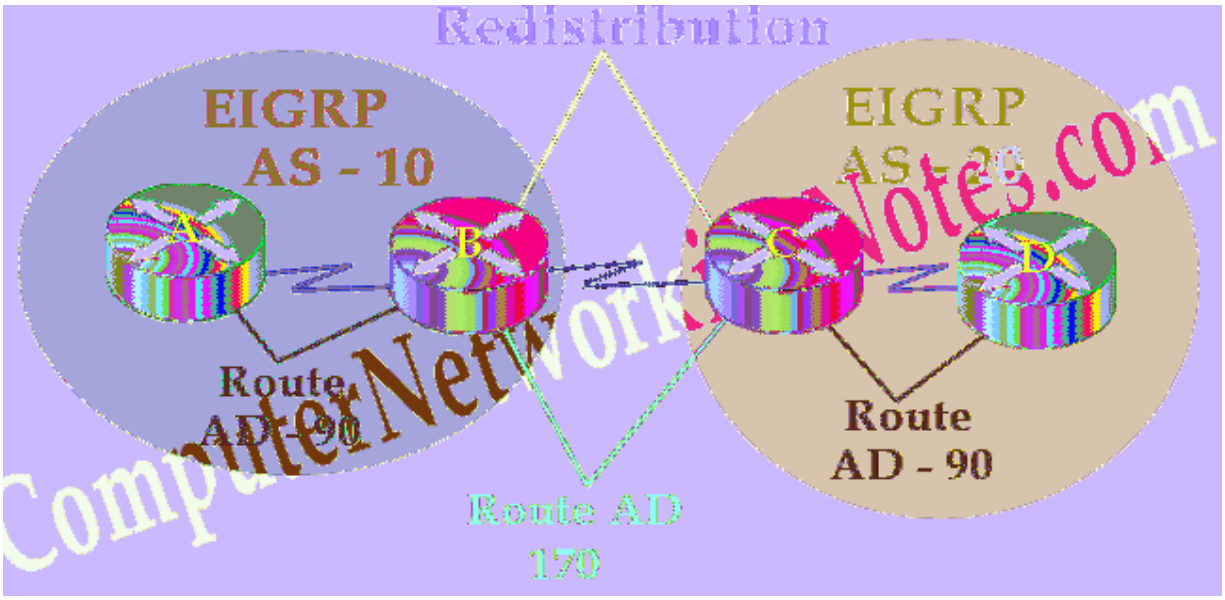
DUAL uses topology table along with RTP to accomplish above tasks in minimal time. As we know EIGRP maintain a copy of all routes including neighbors in topology table, so it would be the first place to look for an alternative route in a route failure situation. If EIGRP does not find an alternative here, it will ask neighbors for help. If neighbors have any updates about asked route, they will reply back with that information. This strong mechanism allows DUAL to find and maintain the best routes for destination speedily.

Autonomous System

EIGRP shares routing information only with neighbors. In order to become a neighbor AS number must be matched. AS create a logical boundary for route information. By default router will not propagate route information outside the AS. For example a router which belongs to AS number 10 will not share routing information with the router that belongs to AS number 20 or any other AS numbers except AS number 10. For easy administration a large network may have multiple ASes.

Not all routing protocols understand the concept of AS. Luckily EIGRP not only understand the concept of AS but also supports multiple ASes. We can easily configure multiple AS instance with EIGRP to divide a large network in smaller segments. By default EIGRP routers will not share routing information between different AS.

Redistribution is used to exchange the route information between different ASes. When a route is learned through the redistribution, it has higher AD value than its original source. For example EIGRP has two AD values 90 for interior EIGRP and 170 for exterior EIGRP. Exterior EIGRP means EIGRP instance which has different AS number.



Administrative Distance

In a complex network, we may have multiple routing protocols running simultaneously. Different routing protocols use different metrics to calculate the best path for destination. In

this situation router may receive different routes information for a single destination network. Routers use AD value to select the best path among these routes. Lower ad value has more trustworthiness.

AD value	Protocol / Source
0	Directly connected interface
0 or 1	Static route
90	EIGRP (Interior)
110	OSPF
120	RIP
170	EIGRP (Exterior)
255	Unknown source

Let's understand it with a simple example; a router learned two different paths for 20.0.0.0/8 network from EIGRP Interior and EIGRP Exterior. Which one should it select?

Answer of this question is hidden in above table. Check the AD value of both protocols. Administrative distance is the believability of routing protocols. Routers measure each route source on a scale of 0 to 255. 0 is the best route. 255 is the worst, router will never use the route learned by this source. In our question we have two protocols EIGRP Interior and EIGRP Exterior. EIGRP Interior has lower AD value than EIGRP Exterior. So its route will be selected for routing table.

That's all for this part. In this part we covered basic terminology used in EIGRP routing protocol.

Essential configuration values

EIGRP Router doesn't trust anyone blindly. It checks following configuration values to insure that requesting router is eligible to become his neighbor or not.

1. Active Hello packets
2. AS Number
3. K-Values

Active Hello packets

EIGRP uses hello packets to maintain the neighborship between routers. It uses them for neighbor discovery and recovery process. Hello packets are periodically sent from all active interfaces.

By default when we enable EIGRP routing, all interfaces (that meet network command criteria) become participate of it. EIGRP allows us to exclude any interface from it.

Passive interface

passive-interface command is used to exclude an interface from EIGRP. Passive interface command is a double edged sword. If used carelessly, it could bring entire network down. Once you marked an interface as passive, EIGRP will never send a hello packet from it. And we know that hello packet is first condition of EIGRP neighborship. In this situation EIGRP neighborship will not take place on this interface. This could be critical if this interface is the only way to connect with other routers. Making this interface as passive will close all possible doors to communicate with those networks.

So our first condition that needs to be fulfilled in order to become an EIGRP neighbor is an active interface generating hello packets. Two routers will become neighbors only when they see each other's hello packets on a common network.

EIGRP sends hello packets from all active interfaces in hello interval. Hello interval is a time duration that EIGRP takes between two hello packets. Default hello interval for high bandwidth link is 5 seconds. For low bandwidth links, hello interval is 60 seconds.

- Ethernet, Token Ring, Point to Point serial links, HDLC leased lines are the examples of high bandwidth link.
- Multipoint circuits, Multipoint ATM, Multipoint Frame Relay, ISDN and BRIs are the example of low bandwidth links.

An EIGRP router must receive hello packets continuously from its neighbors. If it does not receive hello packets from any neighbor in hold down time, it will mark that neighbor as dead.

Hold time is the time duration that an EIGRP router waits before marking a router dead without receiving a hello packet from it. Typically hold down time is three times of hello interval. So for high bandwidth link it would be 15 seconds and 180 seconds for slow bandwidth link. We can adjust hold down time with *ip hold-time eigrp* command.

EIGRP uses multicast and unicast for hello packets delivery. It uses 224.0.0.10 IP address for multicast. Since hello packets do not have any important routing information, they need not be acknowledged.

Basically Hello packets perform two essential functions of EIGRP.

- Find another EIGRP router in network and help in building neighborship.
- Once neighborship is built, check continuously whether neighbor is alive or not.

Adjacency

Neighborship is referred as adjacency in EIGRP. So when you see New Adjacency in log, take it for new neighborship. It indicates that a new neighbor is found and neighborship with it has been established.

AS Number

An AS is a group of networks running under a single administrative control. This could be our company or a branch of company. Just like Subnetting AS is also used to break a large network in smaller networks.

AS creates a boundary for routing protocol which allow us to control how far routing information should be propagated. Beside this we can also filter the routing information before sharing it with other AS systems. These features enhance security and scalability of overall network.

Basically AS concept was developed for large networks. Routing protocols which were developed for small networks such as RIP do not understand the concept of AS systems.

There are two types of routing protocols IGP and EGP.

- **IGP** (Interior Gateway Protocol) is a routing protocol that runs in a single AS such as RIP, IGRP, EIGRP, OSPF and IS-IS.
- **EGP** (Exterior Gateway Protocol) is a routing protocol that performs routing between different AS systems. Nowadays only BGP (Border Gateway Protocol) is an active EGP protocol.

To keep distinguish between different autonomous systems, AS numbers are used. An AS number start from 1 and goes up to 65535. Same as IP addresses, AS numbers are divided in two types; Private and public.

- **Public AS Numbers**: - We only need to use public numbers if we connect our AS with Internet backbone through the BGP routes. IANA (Numbers Authority) controls the public AS numbers.
- **Private AS Numbers**: - Private AS numbers are used to break our internal network into the smaller networks.

EIGRP routers that belong to different ASs don't become neighbors therefore they don't share any routing information.

So our second condition that needs to be fulfilled in order to become EIGRP neighbor is the same AS number. Two routers will become neighbors only when they see same AS number in each other's hello packets.

K Values

EIGRP may use five metric components to select the best route for routing table. These are Bandwidth, Load, Delay, Reliability and MTU. By default EIGRP uses only two components; Bandwidth and delay. With K-Values we can control which components should be used in route metric calculation. For five metric components we have five K values.

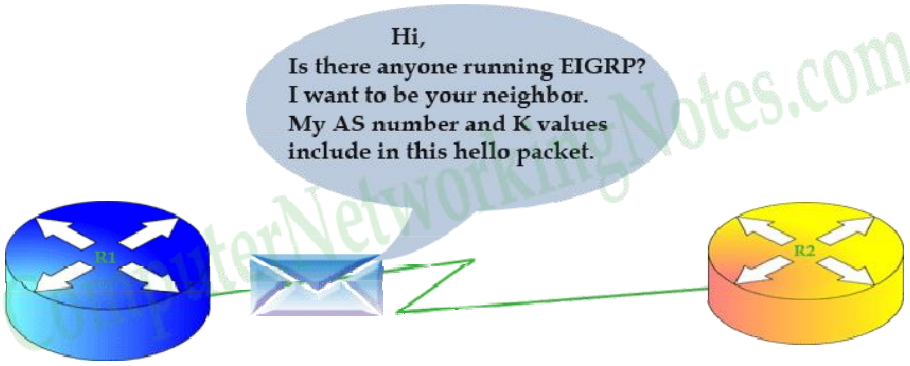
K Values	Metric components
K1	Bandwidth
K2	Load
K3	Delay
K4	Reliability
K5	MTU

Two routers must use same K Values in order to become the EIGPR neighbor. For example if one router is using three K- Values (K1, K2 and K3) while second router is using default K values (K1 and K3) then these two routers will never become neighbor.

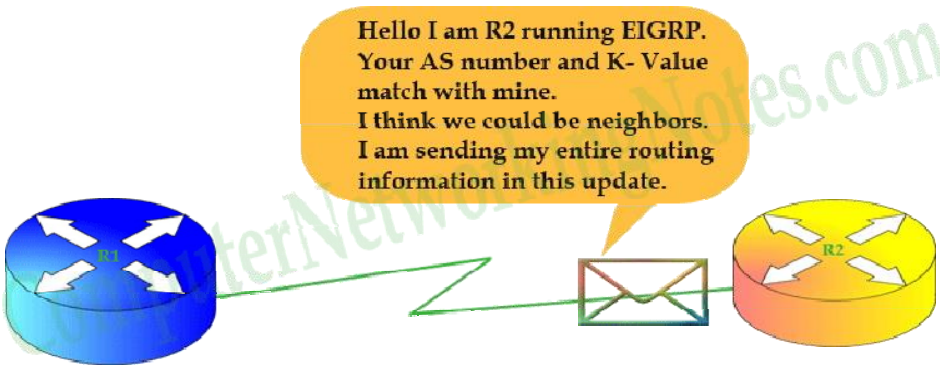
In order to become EIGRP neighbor two routers must use same K values.

EIGRP Neighbor Discovery process

Step 1:- First router R1 sends a hello packet from all active interfaces. This packet contains essential configuration values which are required to be a neighbor.



Step 2:- Receiving router R2 will compare these values with its own configuration values. If both necessary values match (AS number and K-values), it will reply with a routing update. This update includes all routes information from its routing table excluding one route. The route which it learned from the same interface that bring hello packet to it. This mechanism is known as split horizon. It states that if a router receives an update for route on any interface, it will not propagate same route information back to the sender router on same port. Split horizon is used to avoid routing loops.

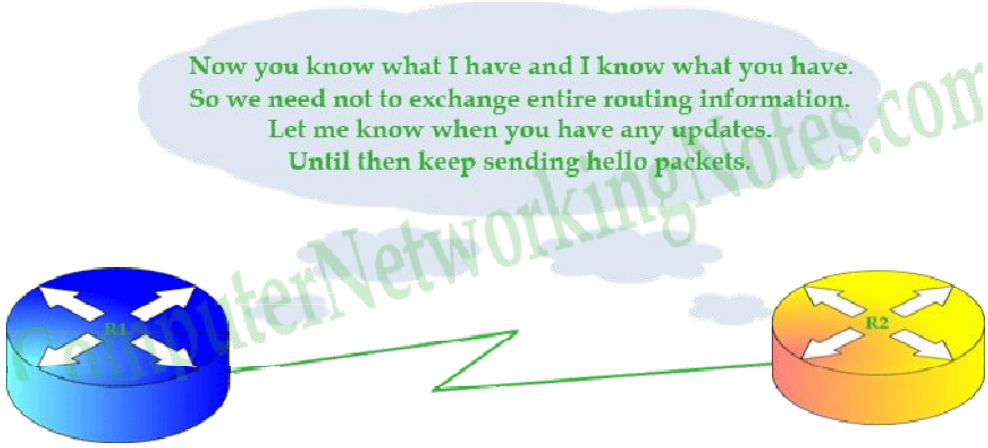


Step 3:- First router will receive R2's routing update and sends an acknowledgement message back to R2.

Step 4:- R1 will sync its EIGRP topology table with routing information that it received in routing update. It will also send a routing update containing all route information from its routing topology to R2.

Step 5:- R2 will respond with an acknowledgement message. It will also sync its EIGRP topology table with routing information that it received in routing update.

At this point, the two routers have becomes neighbor. Now they will maintain this neighborship with ongoing hello packets. If they see any change in network, they will update each other with partial updates.



Partial update contains information only about the recent change.

That's all for this part. In this part we explained how two routers become EIGRP neighbors.

K-Values and EIGRP Metrics

K-Values are the most confusing part of EIGRP. Usually newbies take K Values as EIGRP metric components. K Values are not the metric components in themself. They are only the place holder or influencer for actual metric components in metric calculation formula. So when we enable or disable a K value, actually we enable or disable its associate metric component.

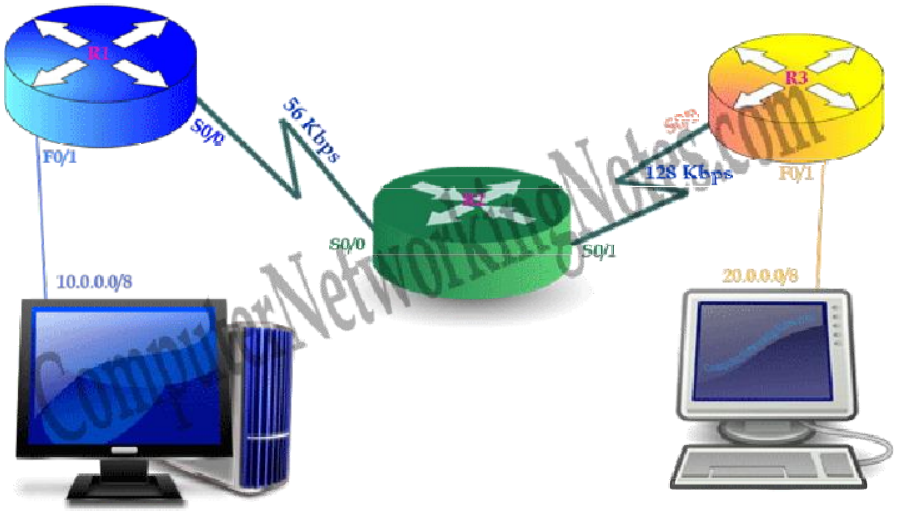
EIGRP uses four components out of five to calculate the routing metric.

K Value	Component	Description
K1	Bandwidth	Lowest bandwidth of route
K2	Load	Worst load on route based on packet rate
K3	Delay	Cumulative interface delay of route
K4	Reliability	Worst reliability of route based on keep alive
K5	MTU	Smallest MTU in path [Not used in route calculation]

Bandwidth (K1)

Bandwidth is a static value. It will change only when we make some physical (layer1) changes in route such as changing cable or upgrading link types. EIGPR picks lowest bandwidth from all outing going interfaces of route to the destination network.

For example have a look on following figure.



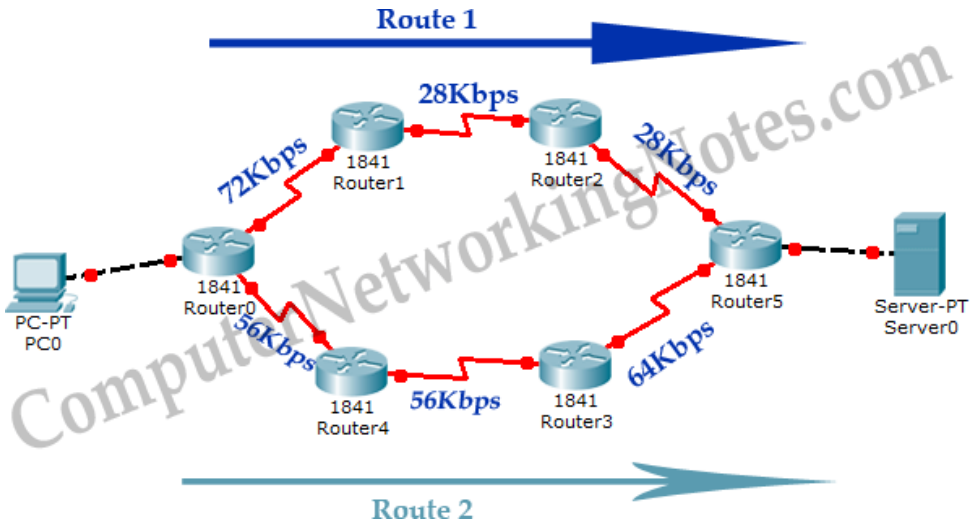
We have two serial links. One has 56Kbps bandwidth and other has 128Kbps. So which one will be selected?

Among these bandwidths EIGRP will pick 56Kbps for composite metric calculation formula.

You may surprise why it picks the lowest instead of the highest? Well picking the highest bandwidth doesn't give us a surety of equivalent bandwidth throughout the route. It's a maximum cap which means we will get its equivalent or lower bandwidth in this route.

While picking the lowest bandwidth gives us a guarantee of equivalent of higher bandwidth throughout the route. Since this is the bottleneck of route.

For example have a look on following network



With highest bandwidth comparison

Highest bandwidth of Route1 (72Kbps)

Highest bandwidth of Route2 (64Kbps)

Which route provides better bandwidth?

72Kbps (Route1) > 64Kbps (Route2)

With this comparison Route1 will be selected.

With lowest bandwidth comparison

Lowest bandwidth of Route1 (28Kbps)

Lowest bandwidth of Route2 (56Kbps)

Which route provides better bandwidth?

56Kbps (Route2) > 28Kbps (Route1)

With this comparison Route2 will be selected.

Looking at lowest bandwidth gives us the actual idea of route.

Next logical question is how EIGRP determine the bandwidth?

EIGRP first looks at ***bandwidth*** command. If bandwidth is set through this command, EIGRP will use it. If bandwidth is not set, it will use interface's default bandwidth.

When we enable an interface, router automatically assign a bandwidth value to it based on its type. For example serial interface has a default bandwidth value of 1544Kbps. Until we change this value with ***bandwidth*** command, it will be used where it is required.

Let me clear one more thing about bandwidth. Changing default bandwidth with ***bandwidth*** command does not change the actual bandwidth of interface. Neither default bandwidth nor bandwidth set by ***bandwidth*** command has anything to do with actual layer one link bandwidth.

Then what purpose does this command solve?

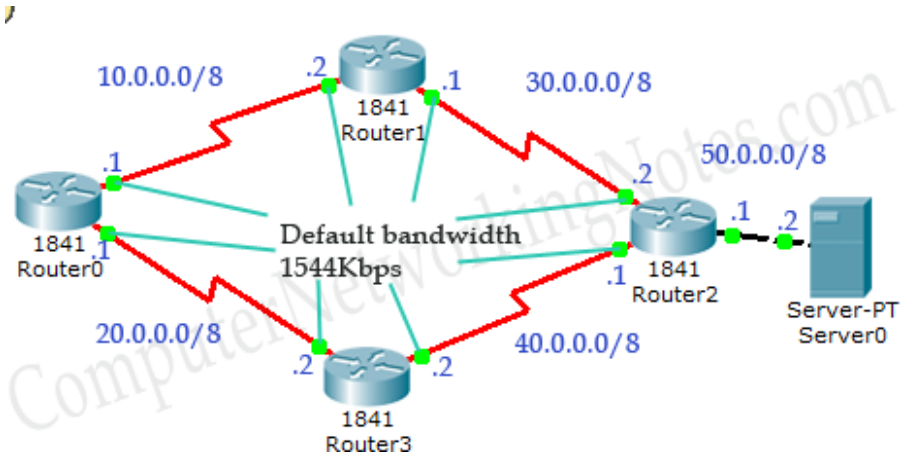
This command is only used to influence the routing protocol which uses bandwidth in route selection process such as EIGRP and OSPF.

Suppose we have two routes for single destination; Route1 and Route2. For some reason we want to take Route1 instead of Route2. How will we influence default metric calculation to select the Route1?

In starting of this article we talked about K-Values. K-Values allow us to influence the metric calculation. K1 is associated with bandwidth. K1 gets its weight from interface's default bandwidth or bandwidth set through the ***bandwidth*** command. Changing default bandwidth with ***bandwidth*** command will change the K1's value in metric calculation formula.

So to take Route1, we will have to make its lowest bandwidth higher than Route2. This can be done in two ways; either raise the lowest bandwidth of Route1 higher than Route2 or reduce the lowest bandwidth of Route2 lower than Route1. Both can be done easily with ***bandwidth*** command.

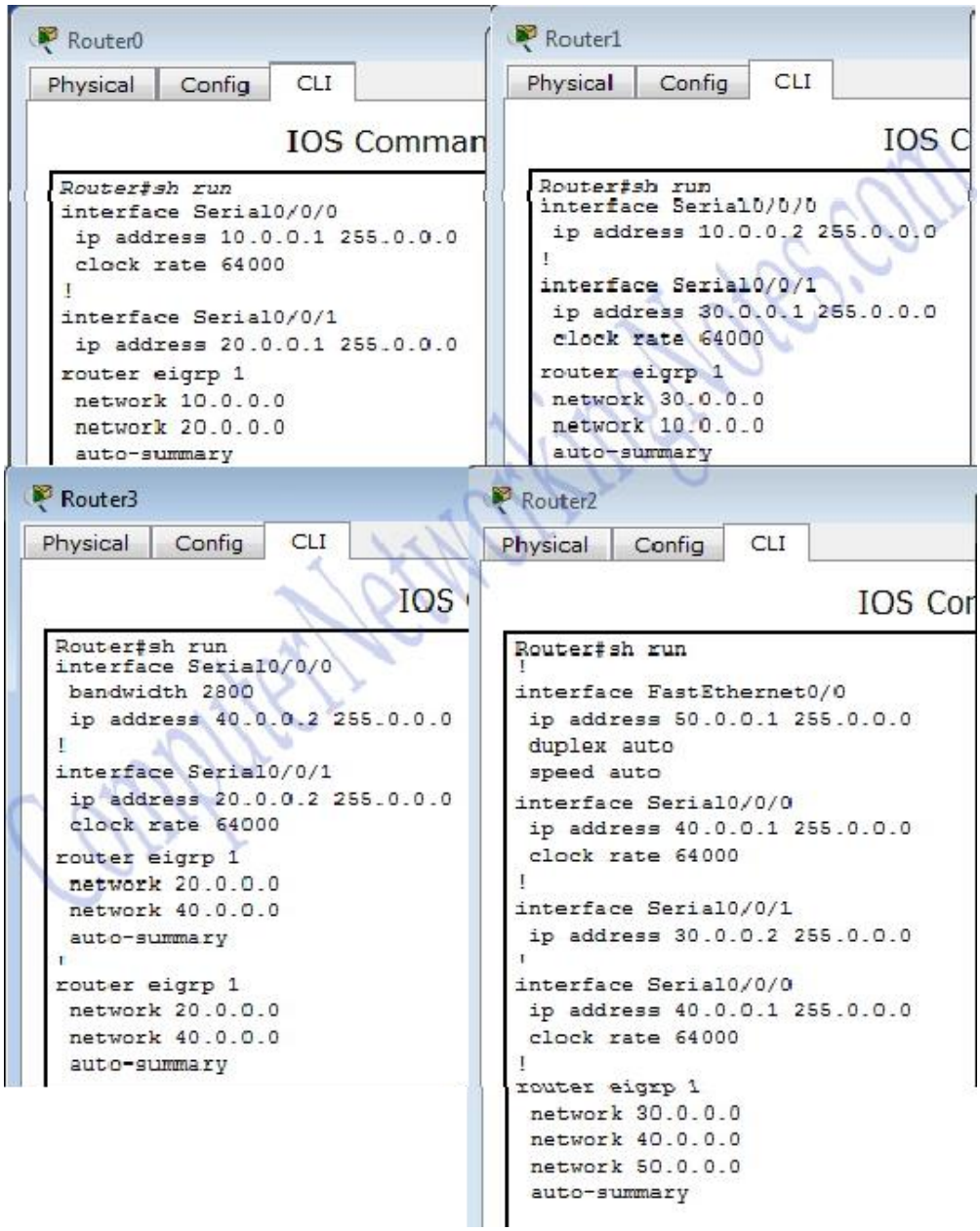
Let's understand this with a simple example. Following figure illustrate a simple EIGRP network.



In this network R0 has two routes to reach at 50.0.0.0/8 network.

1. Route1 (Via R0 – R1 – R2)
2. Route2 (Via R0 – R3 – R2)

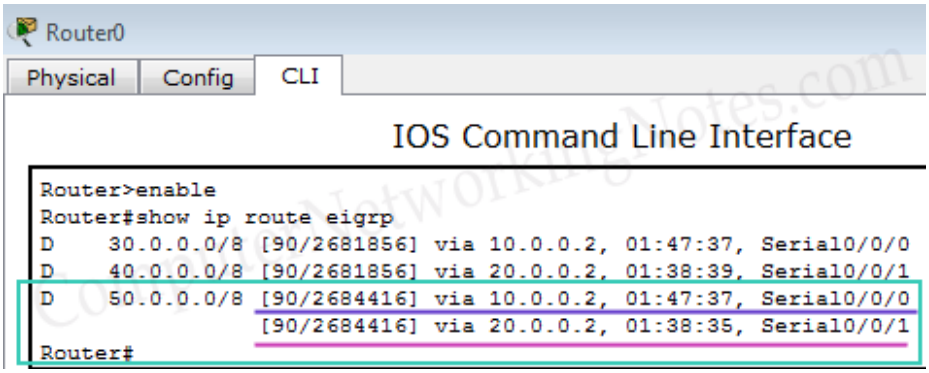
EIGRP is configured on all routers and all links have default bandwidth.



Serial link has default bandwidth of 1544Kbps. Until we change bandwidth of any route, both routes have equal lowest bandwidth.

Route1's lowest bandwidth (1544Kbps) = Route2's lowest bandwidth (1544Kbps)

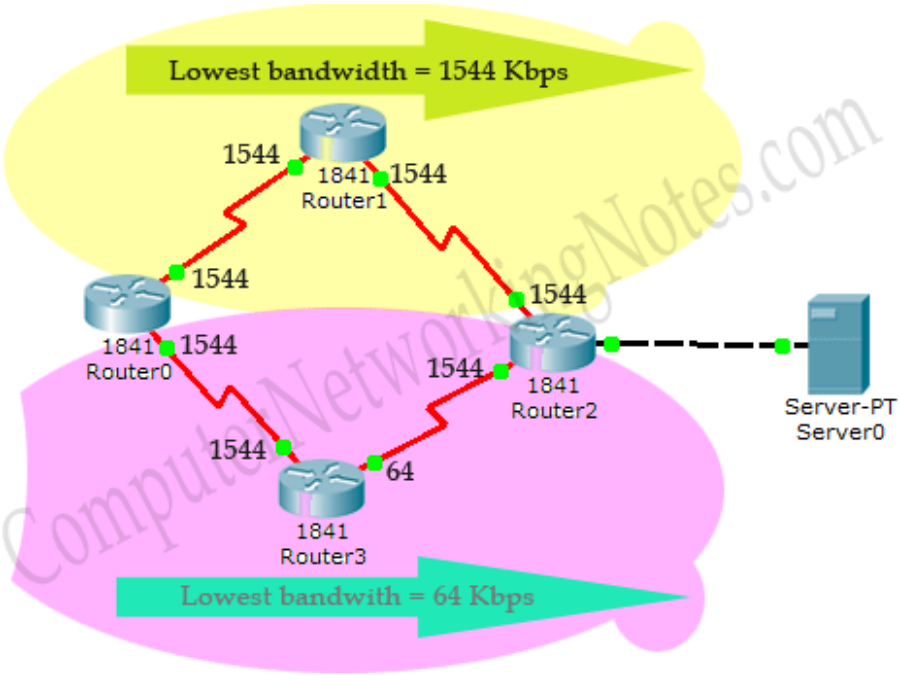
Both routes are load balanced with equal cost value 2684416.



Ok, let's change default bandwidth to see how bandwidth component influence the route metric.

Set bandwidth to 64Kbps (lower than default 1544Kbps) on R3's serial 0/0/0 interface.

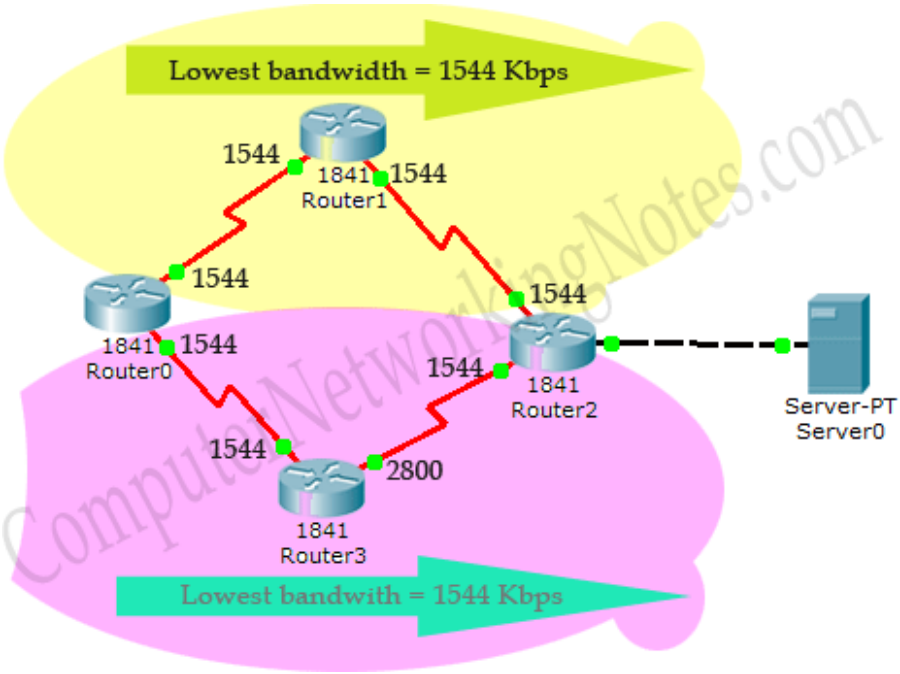
Route1's lowest bandwidth (1544Kbps) > Route2's lowest bandwidth (64Kbps)



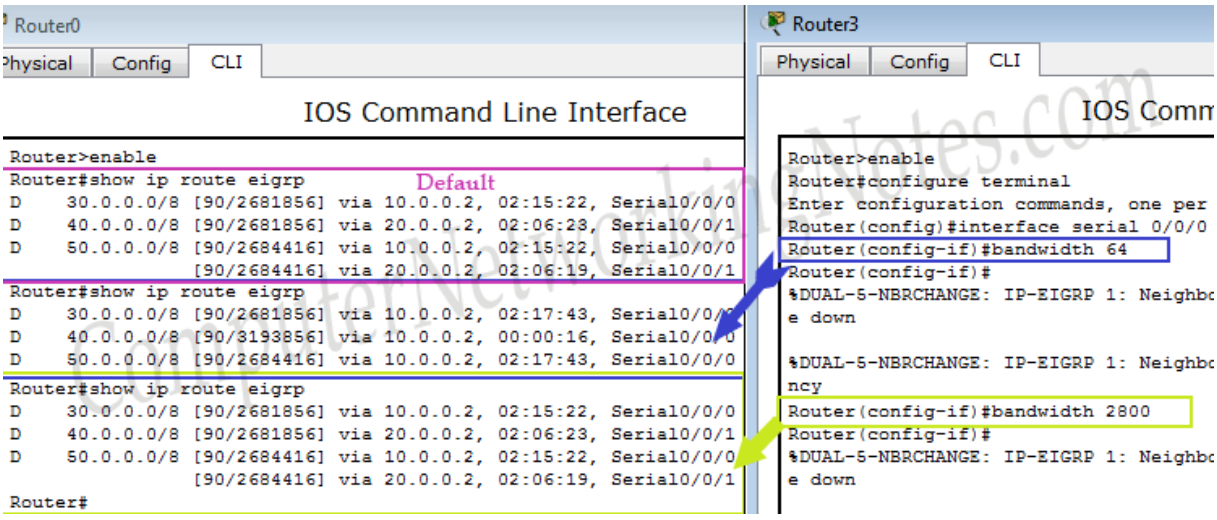
Now Route1 has the higher lowest bandwidth so it would be selected.

Ok let's change bandwidth at R3 again this time increase default bandwidth to 2800Kbps.

Route1's lowest bandwidth (1544Kbps) = Route2's lowest bandwidth (1544Kbps)



Both routes have equal lowest bandwidth. They will be load balanced.



Here I have question for you.

Why EIGRP load balanced between Route1 and Route2 while now Route2 has better bandwidth?

Because EIGRP uses the lowest bandwidth of route to calculate the path cost and that is still 1544Kbps.

Load (K2)

Load is a dynamic value that changes frequently. It is based on packet rate and bandwidth of interface. It calculates the volume of traffic passing through the interface in comparison of maximum capacity. It is expressed on a scale of 255 where 1 represent that an interface is empty and 255 represent that an interface is fully utilized.

Since data flows from both directions, router maintains two separate metric counters;

- **Txload** for outgoing traffic
- **Rxload** for incoming traffic

If K2 is enabled, maximum **Txload** value will be used in composite metric calculation formula.

Delay (k3)

Delay reflects the time taken by a packet in crossing the interface. It is measured in fractions of seconds. Like as bandwidth Cisco has implicit delay values for all interfaces based on the type of interface hardware. For example a FastEthernet has default delay of 100 microseconds. Since it is a static value, we can override it with **delay** command.

Delay can be set anywhere from 10 to 167,772,140 microseconds.

Default delay value or value set by **delay** command has nothing to do with the actual delay caused by interface. Just like bandwidth, this value is also an influencer.

It is expressed in terms of tens of microseconds. To define a delay of 1000 microseconds, we need to configure 100(1000/10) on interface. Output of **show interface** command will automatically multiply it with ten before displaying.

Total delay is used in metric calculation formula.

Total delay = delay received from neighboring router + its own interface delay

EIGRP is an enhanced distance vector routing protocol. It also uses route poisoning, withdrawing route, split horizon and poisoned reverse for loop free optimized network. For all these mentioned techniques EIGRP use the maximum delay as the indication of the unreachable route. To denote the unreachable route EIGRP uses the delay of 16,777,215 tens of the microseconds.

Reliability (K4)

Just like load, reliability is also a dynamic value. It compares all successfully received frames against all received frames. 100% reliability indicates that all the frames which we received were good. We don't have any issue with physical link. If we have any issue with physical link, this value will be decrease.

Reliability is expressed as the fraction of 255. 255 expresses 100% reliability while 0 represents 0% reliability. If K4 is enabled in metric calculation formula, it will use minimal reliability.

MTU (K5)

MTU stands for maximum transmission unit. It is advertised with routing update but it does not actively participate in metric calculation. EIGRP allows us to load balance between equal cost paths (6 maximum, default set to 4). It is used when equal cost paths for same destination exceed the number of allowed paths set from ***maximum-paths*** command. For example we set maximum allowed paths for load balancing to 5 and metric calculates 6 equal cost paths for a single destination. In this situation path with lowest MTU will be ignored.

EIGRP Metric Calculation Formula

EIGRP uses following formula to produce a single 32 bit metric:-

$$\left[\left(K_1 \cdot \text{Bandwidth}_A + \frac{K_2 \cdot \text{Bandwidth}_E}{256 - \text{Load}} - K_3 \cdot \text{Delay}_E\right) \cdot \frac{K_5}{K_4 + \text{Reliability}}\right] \cdot 256$$

At first glance this formula looks like a complicated equation. But it is not as difficult as it sound. Let's make it easier.

As we know MTU (K5) is not actively participate in formula. So set its value to Zero. When K5 is equal to 0 then [K_5/ (K_4 + reliability)] is defined to be 1.

By default EIGRP does not use dynamic values in metric. This will disable two more components; load (K_2) and reliability (K_4).

Now only two static values remain in formula.

Use of default constants [K1 (Enabled), K2 (Disabled), K3 (Enabled), K4 (Disabled), K5 (Disabled not used)] reduce our formula to:-

Metric = (Bandwidth_E + Delay_E)*256.

Cisco uses following configuration values for Bandwidth and delay

Bandwidth_E = 10^7/ least bandwidth of route [Lowest bandwidth from all interfaces between source and destination. Use interface default bandwidth wherever bandwidth is not set through the **bandwidth** command]

Value_E = cumulative delay of route [Sum of all outgoing interface's delay. Use interface default delay, if not set through the **delay** command]

Putting these configuration values will make formula to look like this

$$\text{Metric} = \left(\left(\frac{10^7}{\text{Least-bandwidth}} \right) + \text{Cumulative-delay} \right) * 256$$

Before we move further, let me explain why EIGRP keeps dynamic values disable by default.

Dynamic values change over the time. Enabling dynamic values will force EIGRP routers to calculate metric all the time and send updates each other just because the load or reliability of an interface has changed. This will create serious performance issue. To avoid such a situation EIGRP only enables static values for metric calculation.

If we only enable static values for metric calculation, EIGRP will not recalculate the metric unless it changed. Static values change only when a physical change occurred in network such as an interface is down or router is dead. This will keep EIGRP nice and clean.

Let's see this formula in action. Earlier in this tutorial we used an example topology to explain the bandwidth component. Load that topology in packet tracer and run ***show ip route eigrp*** command from privilege mode. We have four routes for three destination networks. One destination network has two routes.

30.0.0.0/8

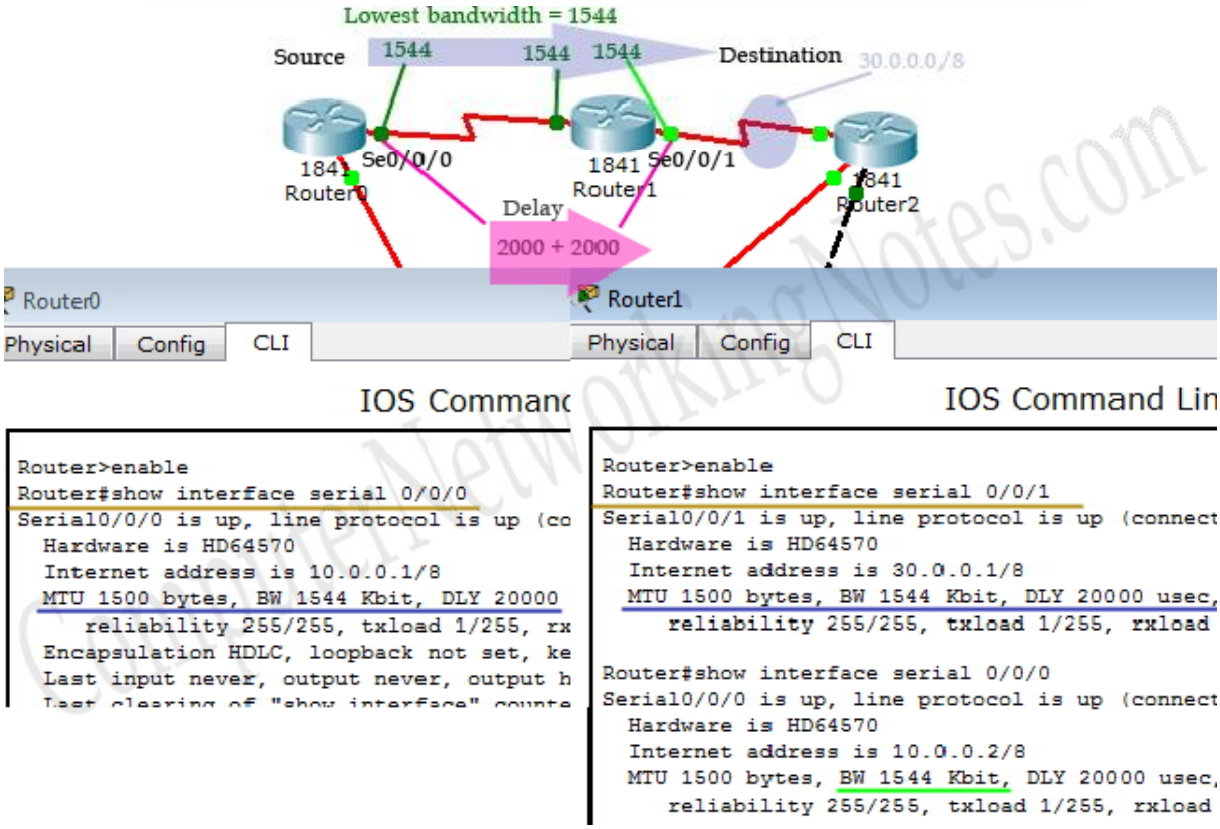
For this destination network metric cost is 2681856. Before we learn how this cost was calculated, we need to understand some key points associated with formula.

- EIGRP picks the lowest bandwidth from all interfaces in route.
- EIGRP picks delay from all outgoing interfaces in route.
- **show interface [*interface*]** command of privilege mode will display the configured value of metric components.
- While calculating the cost term least-bandwidth uses the unit of Kbps (Kilobits per second).
- **show interface [*interface*]** command list bandwidth in Kbps. So we can use listed bandwidth in formula as it is.
- While calculating the cost term cumulative-delay uses the unit of tens of microseconds.
- **show interface [*interface*]** command list delay in microseconds. So we need to divide it with 10 before using it in formula.

- Any decimal value will be rounded back to the nearest integer before performing the rest of the formula.

We have three serial interfaces between source and destination. So our first step is to find out the value of bandwidth and delay.

We can use show interface command to know the values.



All interfaces have equal bandwidth so our least bandwidth would be 1544Kbps.

We have two outgoing interfaces between source and destination. Both have a default delay of 20000 microseconds so total delay would be 40000 microseconds. As we know this delay is in microseconds and formula uses the unit of “tens of microseconds”. We need to divide 40000 with 10. So our cumulative delay would be 40000/10 = 4000.

Okay now we have least bandwidth (1544Kbps) and cumulative delay (4000) let’s put them in formula

$$\text{Metric} = \left(\left(\frac{10^7}{\text{Least-bandwidth}} \right) + \text{Cumulative-delay} \right) * 256$$

Metric = ((10000000/1544) +4000)*256

Metric = ((6476.6839) +4000)*256

As I said "*Any decimal value will be rounded back to the nearest integer before performing the rest of the formula*."

Before solving rest of the formula, convert decimal value back in positive integer.

Metric = ((6476) +4000)*256

Metric = (10476)*256

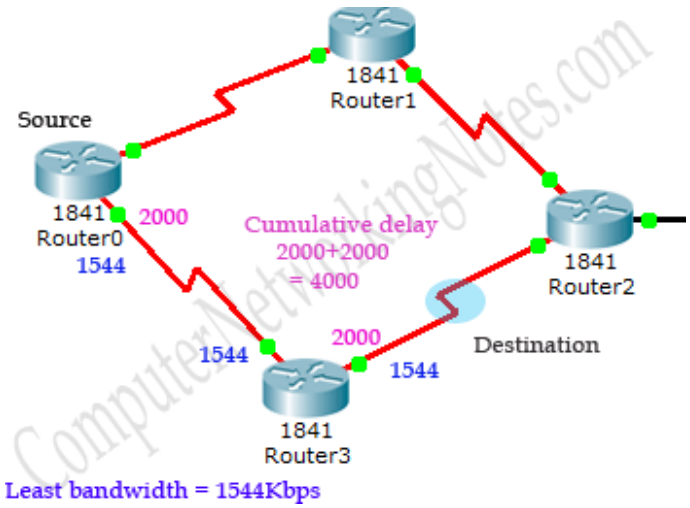
Metric = 10476*256

Metric = 2681856

Great! We have revealed the cost calculation method. Let's do this calculation again for next route.

40.0.0.0/8

For this route we have lowest bandwidth 1544Kbps and cumulative delay of 4000(ten of microseconds).



Let's put these values in our formula

Metric = ((10000000/1544)+4000)*256

Metric = (6476 + 4000) * 256

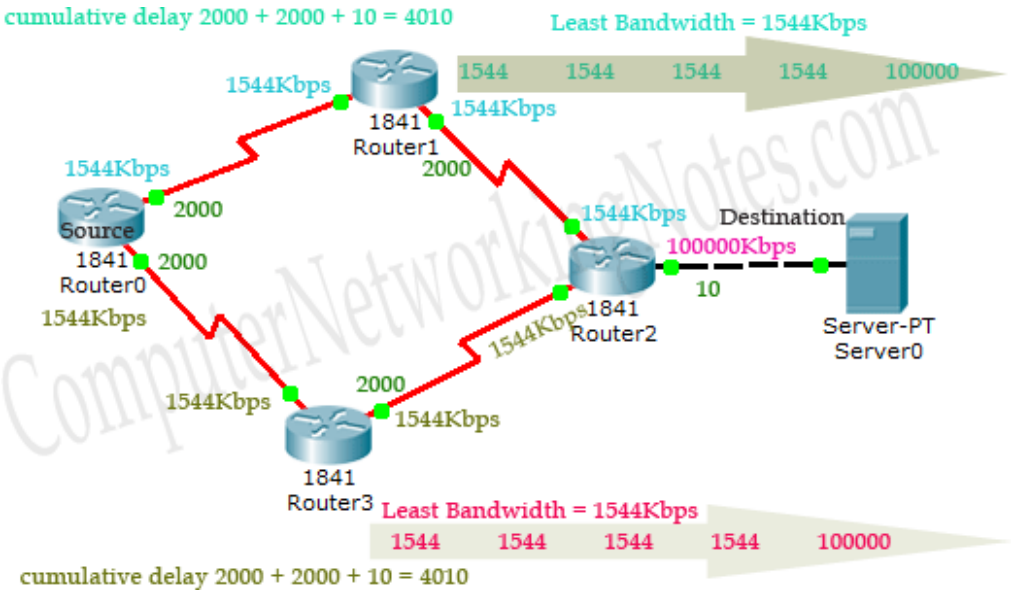
Metric = 10476 * 256

Metric = 2681856

Fine, now we have only route left. Let's figure out its cost also.

50.0.0.0/8

For this destination we have two routes. Both routes have equal least bandwidth and cumulative delay. So naturally their cost will also be same. As we know EIGRP automatically load balance equal cost routes and these routes have equal cost. So they both make their way to routing table.



Metric = ((10000000/1544) + 4010) * 256

Metric = (6476 + 4010) * 256

Metric = 10486 * 256

Metric = 2684416

This is how EIGRP calculates the route cost. In job life you will rarely need to calculate the route cost manually

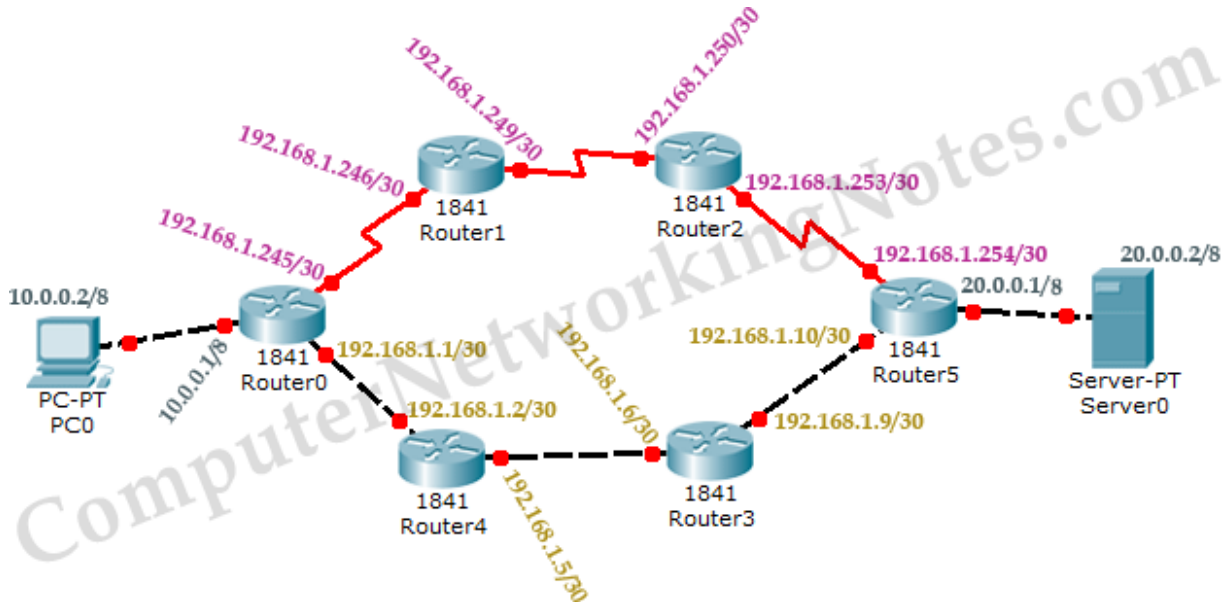
Configuration for EIGRP Routing Protocol

In this assignment we will see basic concepts of EIGRP such as Features and characteristics of EIGRP, Neighbor Table, Topology Table, Routing Table, Protocol Dependent Modules, Metric, RTP, DUAL, Autonomous System and Administrative Distance.

Also we we will see how two routers become EIGRP neighbor and maintain this neighborship. In order to become an EIGRP neighbor, three essential configuration values must be matched.

EIGRP uses composite metric calculation formula to calculate the best path. Bandwidth, reliability, delay, load and MTU are the components of formula. In this we explained these components with formula in easy language with examples.

Create a topology as illustrate in following figure or download this pre-created topology.

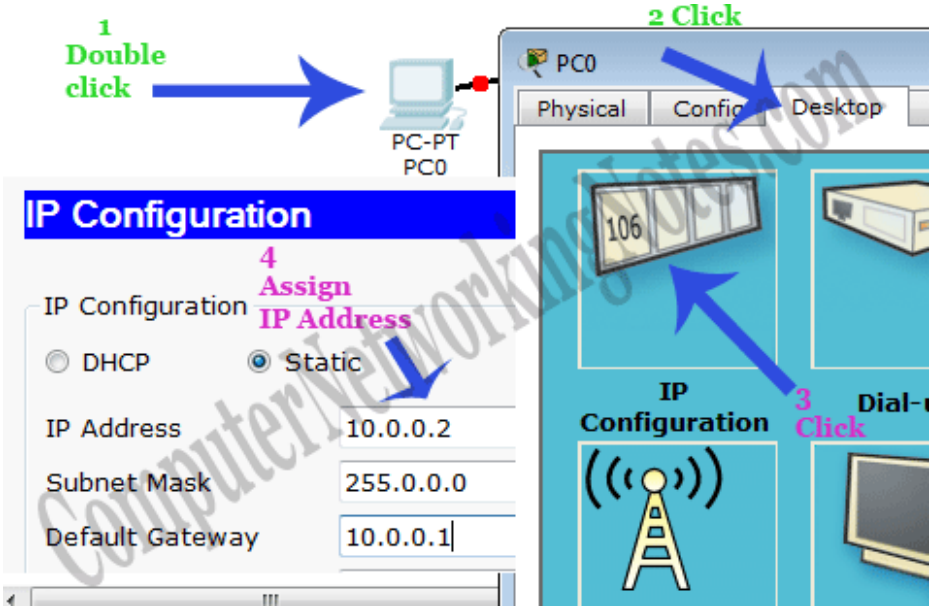


Device	Interface	IP Configuration	Connected with
PC0	Fa0/0	10.0.0.2/8	Router0's Fa0/0
Router0	Fa0/0	10.0.0.1/8	PC0's Fa0/0
Router0	Fa0/1	192.168.1.1/30	Router4's Fa0/1
Router4	Fa0/1	192.168.1.2/30	Router0's Fa0/1
Router4	Fa0/0	192.168.1.5/30	Router3's F0/0
Router3	Fa0/0	192.168.1.6/30	Router4's Fa0/0
Router3	Fa0/1	192.168.1.9/30	Router5's Fa0/1

Router5	Fa0/1	192.168.1.10/30	Router3's Fa0/1
Router5	Fa0/0	20.0.0.1 /8	Serve0's Fa0/0
Server	Fa0/0	20.0.0.2 /8	Router5's Fa0/0
Router5	Se0/0/0	192.168.1.254/30	Router2's Se0/0/0
Router2	Se0/0/0	192.168.1.253/30	Router5's Se0/0/0
Router2	Se0/0/1	192.168.1.250/30	Router1's Se0/0/1
Router1	Se0/0/1	192.168.1.249/30	Router2's Se0/0/1
Router1	Se0/0/0	192.168.1.246/30	Router0's Se0/0/0
Router0	Se0/0/0	192.168.1.245/30	Router1's Se0/0/0

Assign IP address to PCs

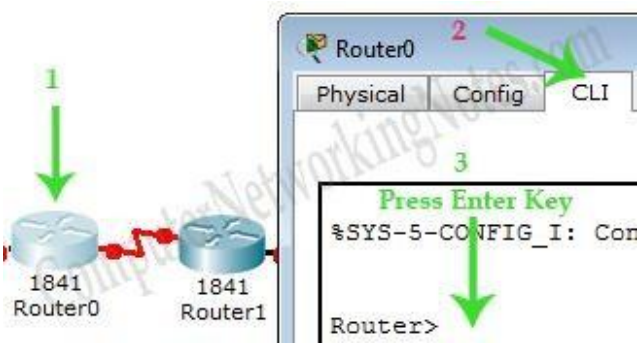
Double click **PC0** and click **Desktop** menu item and click **IP Configuration**. Assign IP address **10.0.0.2/8** to **PC0**.



Repeat same process for Server0 and assign IP address 20.0.0.2/8.

Assign IP address to interfaces of routers

Double click **Router0** and click **CLI** and press Enter key to access the command prompt of **Router0**.



Three interfaces **FastEthernet0/0, FastEthernet0/1** and **Serial0/0/0** of **Router0** are used in this topology. By default interfaces on router are remain administratively down during the start up.

We need to configure IP address and other parameters on interfaces before we could actually use them for routing. Interface mode is used to assign the IP address and other parameters. Interface mode can be accessed from global configuration mode. Following commands are used to access the global configuration mode.

```
Router>enable
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
```

From global configuration mode we can enter in interface mode. From there we can configure the interface. Following commands will assign IP address on FastEthernet0/0 and FastEthernet0/1.

```
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface fastEthernet 0/1
Router(config-if)#ip address 192.168.1.1 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

interface fastEthernet 0/0 command is used to enter in interface mode.

ip address 10.0.0.1 255.0.0.0 command will assign IP address to interface.

no shutdown command will bring the interface up.

exit command is used to return in global configuration mode.

Serial interface needs two additional parameters **clock rate** and **bandwidth**. Every serial cable has two ends DTE and DCE. These parameters are always configured at DCE end.

We can use **show controllers interface** command from privilege mode to check the cable's end.

```
Router#show controllers serial 0/0/0
Interface Serial0/0/0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 2000000
[Output omitted]
```

Fourth line of output confirms that DCE end of serial cable is attached. If you see DTE here instead of DCE skip these parameters.

Now we have necessary information let's assign IP address to serial interface.

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.245 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth  64
Router(config-if)#no   shutdown
Router(config-if)#exit
```

Router#configure terminal Command is used to enter in global configuration mode.

Router(config)#interface serial 0/0/0 Command is used to enter in interface mode.

Router(config-if)#ip address 192.168.1.245 255.255.255.252 Command assigns IP address to interface. For serial link we usually use IP address from /30 subnet.

Router(config-if)#clock rate 64000

In real life environment this parameter controls the data flow between serial links and need to be set at service provider's end. In lab environment we need not to worry about this value. We can use any valid rate here.

Router(config-if)#bandwidth 64

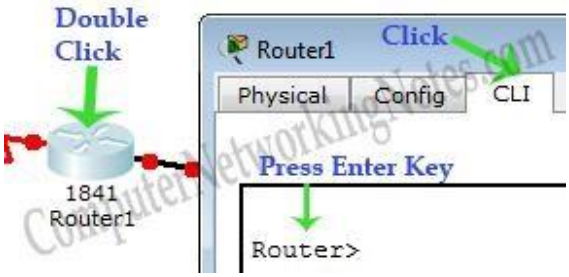
Bandwidth works as an influencer. It is used to influence the metric calculation of EIGRP or any other routing protocol which uses bandwidth parameter in route selection process.

Router(config-if)#no shutdown Command brings interface up.

Router(config-if)#exit Command is used to return in global configuration mode.

We will use same commands to assign IP addresses on interfaces of remaining routers. We need to provided clock rate and bandwidth only on DCE side of serial interface. Following command will assign IP addresses on interface of Router1.

Router1




```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.246 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.1.249 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth  64
Router(config-if)#no   shutdown
Router(config-if)#exit
```

We will use same commands to assign IP addresses on interfaces of remaining routers.

Router2

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.1.250 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.253 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth  64
Router(config-if)#no   shutdown
Router(config-if)#exit
```

Router5

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface fastEthernet 0/1
Router(config-if)#ip address 192.168.1.10 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.1.254 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
```

Router3

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 192.168.1.6 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface fastEthernet 0/1
Router(config-if)# ip address 192.168.1.9 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

Router4

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 192.168.1.5 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface fastEthernet 0/1
Router(config-if)# ip address 192.168.1.2 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

Great job we have finished our half journey. Now routers have information about the networks that they have on their own interfaces. Routers will not exchange this information between

them on their own. We need to implement EIGRP routing protocol that will insist them to share this information.

To be on same track I have uploaded my practice topology on our server. Use this if you want to skip the IP configuration part.

Configure EIGRP routing protocol

Enabling EIGRP is a two steps process:-

1. Enable EIGRP routing protocol from global configuration mode.
2. Tell EIGRP which interfaces we want to include.

For these steps following commands are used respectively.

Router(config)# router eigrp autonomous_system_#
Router(config-router)# network IP_network_# [subnet_mask]

Router(config)# router eigrp autonomous_system_#

This command will enable EIGRP routing protocol in router. We can use any ASN (Autonomous System Number) from 1 to 65,535. In order to become EIGRP neighbors this number must be same on all participates.

Router(config-router)# network IP_network_# [subnet_mask]

This command allows us to specify the local interfaces which we want to include in EIGRP. Basically we define a range of addresses and router search for these addresses in local interfaces. If match found EIGRP will be enabled on that interface. Once enabled, EIGRP will starts advertising about the connected subnets with that interface.

We have two options while defining the range of addresses with **network** command

1. Without wildcard mask
2. With wildcard

Without wildcard

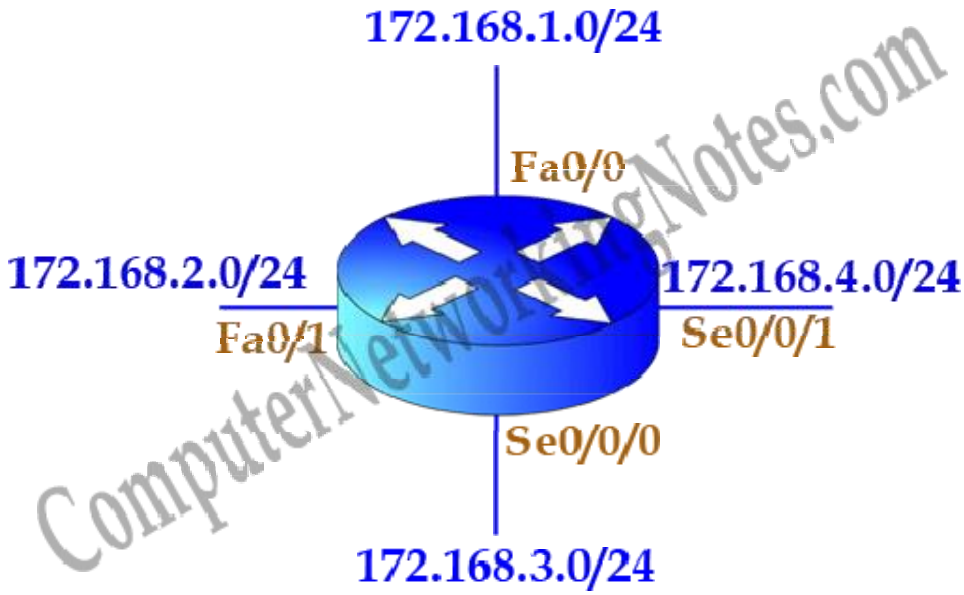
Choosing this option allows us to configure the classful network. This option is very straightforward. All we need to do is, type the network ID with network command. For example network 172.168.0.0 command will enable EIGRP on all interfaces which belong to network 172.168.0.0.

What if I type network number instead of network ID?

Well in this situation EIGRP will automatically convert it back to network ID in which this network number is resides. For example 172.168.1.1 will be converted back in 172.168.0.0.

This creates another query. Why it will be converted in 172.168.0.0 instead of 172.168.1.0?

Answer of this question is hidden in classful configuration. In classful configuration EIGRP will match network addresses with in default boundary. Consider following figure



We have four networks 172.168.1.0/24, 172.168.2.0/24, 172.168.3.0/24 and 172.168.4.0/24 Subnetted from single class B network 172.168.0.0/16. Classful configuration does not understand the concept of Subnetting. In classful configuration all these networks belong to a single network. Classful configuration works only with in default boundary of mask. Default boundary of this address is 16 bits. So it will match only first 16 bits (172.168.x.y) of network address.

If we want excludes serial interfaces from EIGRP, we need to configure network command with more specific information.

With wildcard

In this option we provide wildcard mask along with network ID. Wildcard mask allows us to match exact networks. With wildcard we are no longer limited with default boundaries. We can match Subnetted networks as well as default networks.

For example we were tasked to exclude serial interfaces in above configuration. We can use a wildcard mask of 0.0.0.255 to match the subnet mask of /24.

```
Router(config-router)# network  172.168.1.0 0.0.0.255
Router(config-router)# network  172.168.2.0 0.0.0.255
```

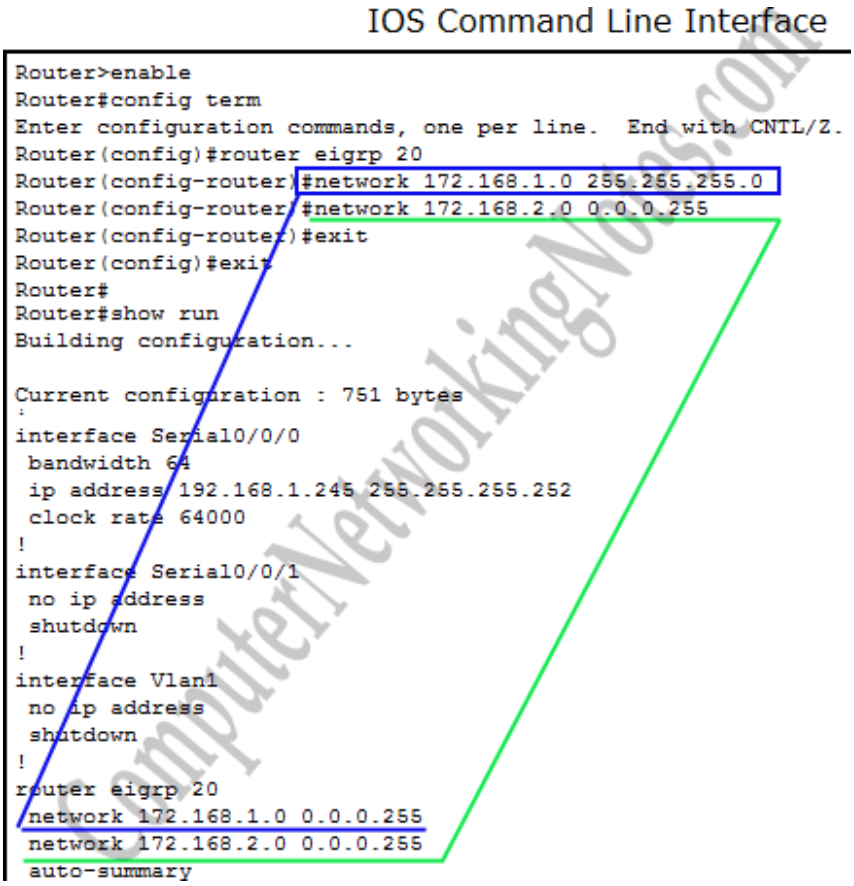
Above commands will ask router to match /24 bits of address instead of default /16 bits. Now router will look for 172.168.1.x and 172.168.2.x network. Our serial interfaces have 172.168.3.0/24 and 172.168.4.0/24 networks which do not fall in these search criteria.

If you are unfamiliar with wildcard mask, I suggest you to read our tutorials on ACL where we explained wildcard mask in detail with examples.

Until you learn wildcard mask, use subnet mask in the place of wildcard mask. Following commands are also valid and do the same job by matching /24 bits of address.

Router(config-router)# network 172.168.1.0 255.255.255.0
Router(config-router)# network 172.168.2.0 255.255.255.0

Subnet mask is a substitute, not a replacement of wildcard mask. When we use Subnet mask, router converts them in wildcard mask before searching for associated interfaces. We can look in running configuration to know what exactly being used by router.



EIGRP configuration

Now we know the essential commands for configuration. Let's implement them in our network.

Router0

Router(config)#router eigrp 20
Router(config-router)#network 10.0.0.0 0.0.0.255
Router(config-router)#network 192.168.1.244 0.0.0.3

```
Router(config-router)#network 192.168.1.0 0.0.0.3
Router(config-router)#
```

Router1

```
Router(config)#router eigrp 20
Router(config-router)#network 192.168.1.244 0.0.0.3
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.245 (Serial0/0/0) is up: new adjacency
Router(config-router)#network 192.168.1.248 0.0.0.3
Router(config-router)#
```

Router2

```
Router(config)#router eigrp 20
Router(config-router)#network 192.168.1.248 0.0.0.3
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.249 (Serial0/0/1) is up: new adjacency
Router(config-router)#network 192.168.1.252 0.0.0.3
Router(config-router)#
```

As I mentioned earlier, we can use both wildcard mask and subnet mask with network command. We have used wildcard mask for above routers. In remaining routers we will use subnet mask.

Router5

```
Router(config)#router eigrp 20
Router(config-router)#network 20.0.0.0 255.0.0.0
Router(config-router)#network 192.168.1.252 255.255.255.252
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.253 (Serial0/0/0) is up: new adjacency
Router(config-router)#network 192.168.1.8 255.255.255.252
Router(config-router)#
```

Router3

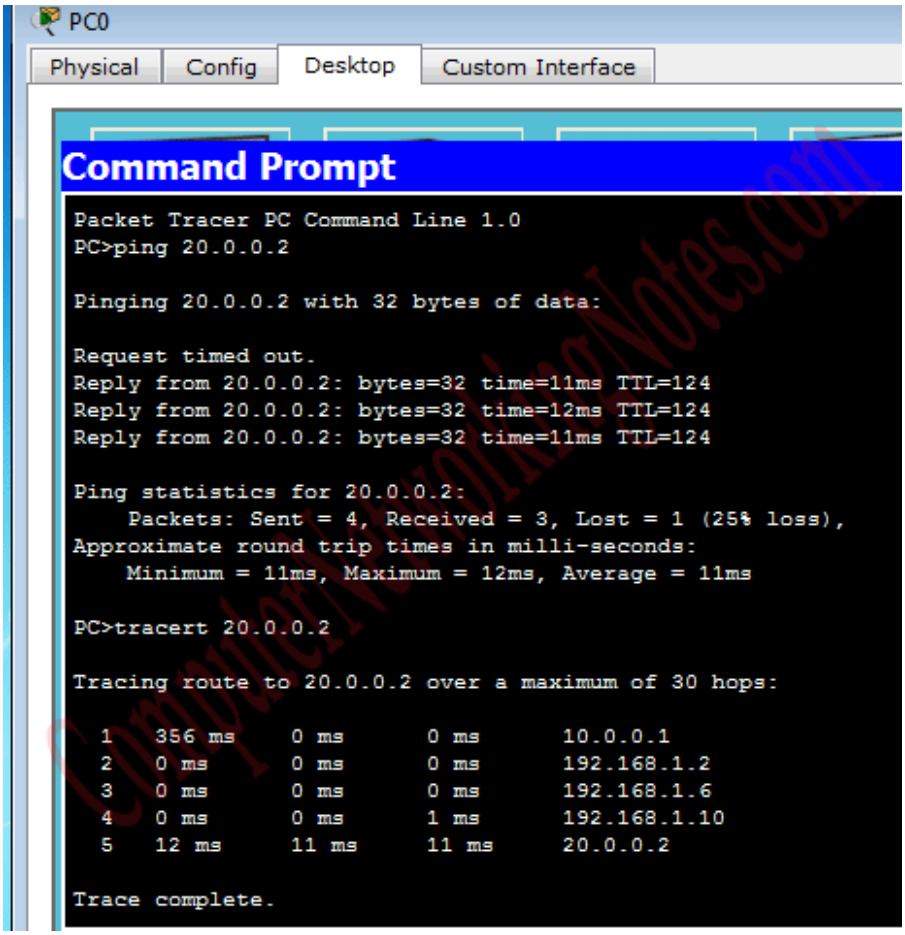
```
Router(config)#router eigrp 20
Router(config-router)#network 192.168.1.8 255.255.255.252
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.10 (FastEthernet0/1) is up: new
adjacency
Router(config-router)#network 192.168.1.4 255.255.255.252
Router(config-router)#
```

Router4

```
Router(config)#router eigrp 20
Router(config-router)#network 192.168.1.4 255.255.255.252
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.6 (FastEthernet0/0) is up: new
adjacency
Router(config-router)#network 192.168.1.0 255.255.255.252
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 20: Neighbor 192.168.1.1 (FastEthernet0/1) is up: new
adjacency
Router(config-router)#
```

That's it. Our network is ready to take the advantage of EIGRP routing. To verify the setup we will use ping command. **ping** command is used to test the connectivity between two devices. We have two routes between source and destination. **tracert** command is used to know the route which is used to get the destination.

Access the command prompt of PC1 and use ping command to test the connectivity from Server0. After that use **tracert** command to print the taken path.



Good going we have successfully implemented EIGRP routing protocol in our network. For cross check we have uploaded a configured topology on our server. You can use this if not getting same output.

EIGRP protocol automatically manages all routes for us. If one route goes down, it will automatically switch to another available route. To explain this process more clearly we have added one additional route in our network.

Currently there are two routes between PC0 and Server.

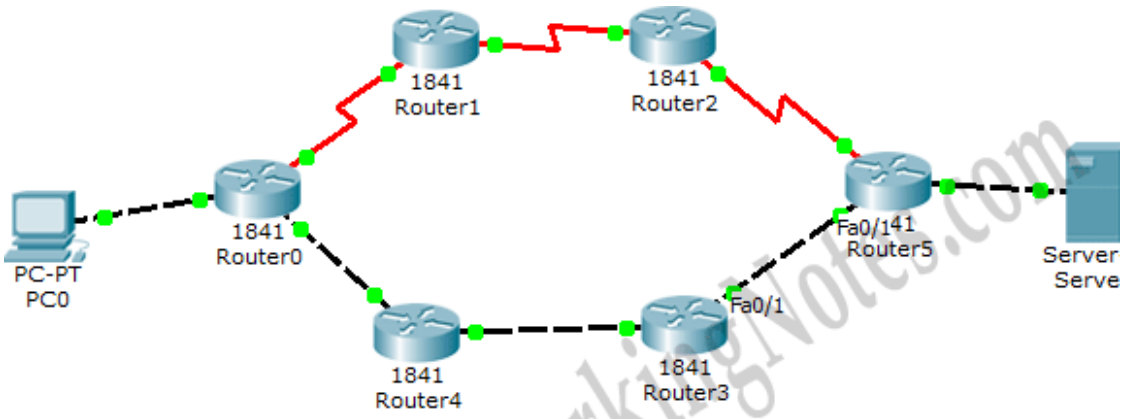
Route 1

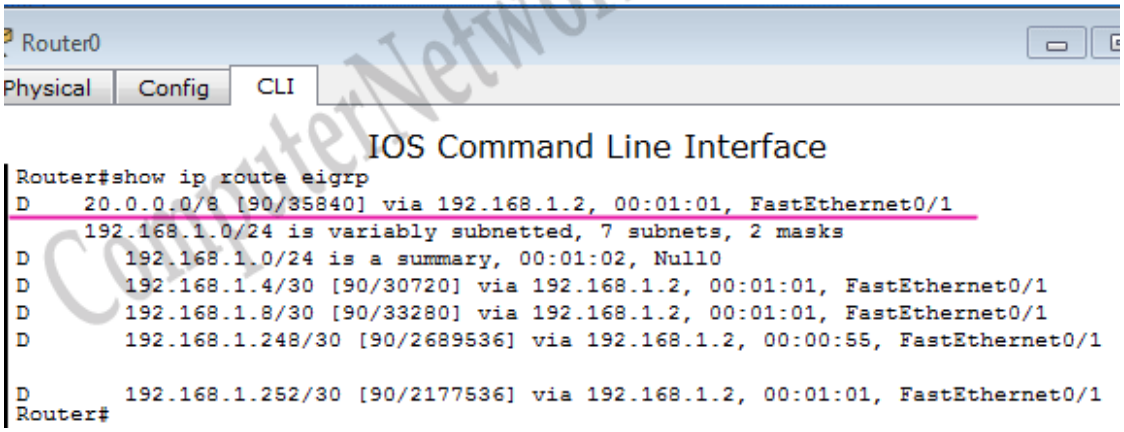
PC0 <==> Router0 <==> Router4 <==> Router3 <==> Router5 <==> Server0

Route 2

PC0 <==> Router0 <==> Router1 <==> Router2 <==> Router5 <==> Server0

By default EIGRP uses the route that has low metric value. Our path separates from Router0, so let's see which route it takes to deliver the packet of 20.0.0.0 network. **show ip route eigrp** command will list all available routes.





Output of show ip route eigrp Explained

D: - It indicates that route is learned by EIGRP. Cisco chose letter D for EIGRP, because letter E was already taken by Exterior Gateway Protocol (EGP).

20.0.0.0 /8: - It is our destination network.

90: - Administrative distance of EIGRP.

35840: - Is the metric value of this route calculated by EIGRP

Via 192.168.1.2: - IP address of the next hop.

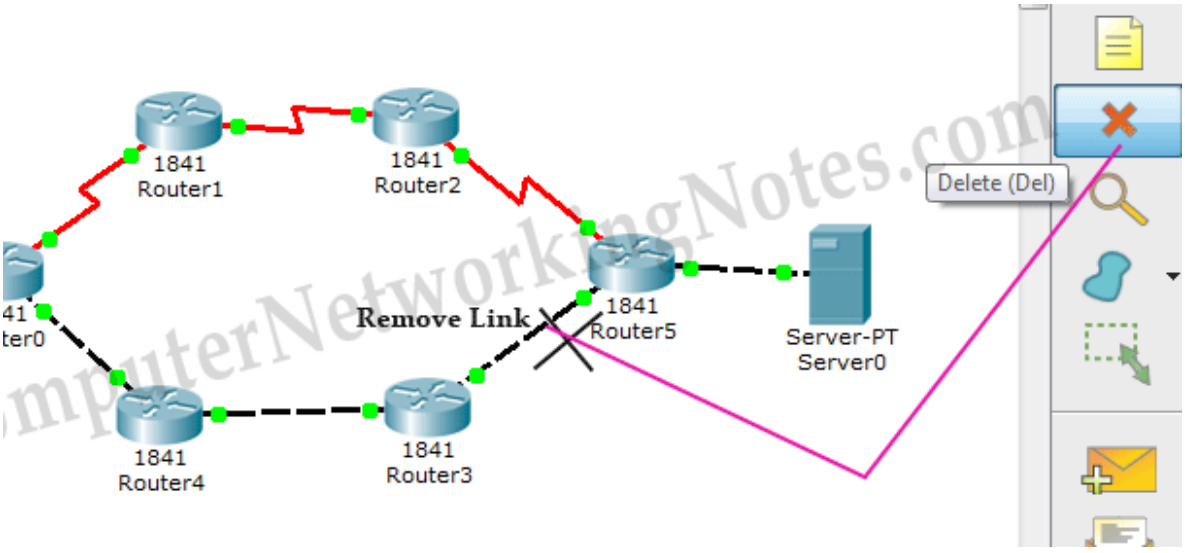
00:01:01: - How long this route was learned (Age of route)

FastEthernet1: - Exit interface of this router to get the next hop.

You may wonder where Route2 is in this output. Well EIGRP puts only the best route in routing table. Route2's metric value is higher than Route1. Till route1 is available, it will not insert route2 in routing table. When route1 is down, it will look for next possible route. If other routes are available, it will replace current route with new route which has the lowest metric value. We can watch this process live with **debug eigrp fsm** command. On debug process on Router0.

Router# debug eigrp fsm

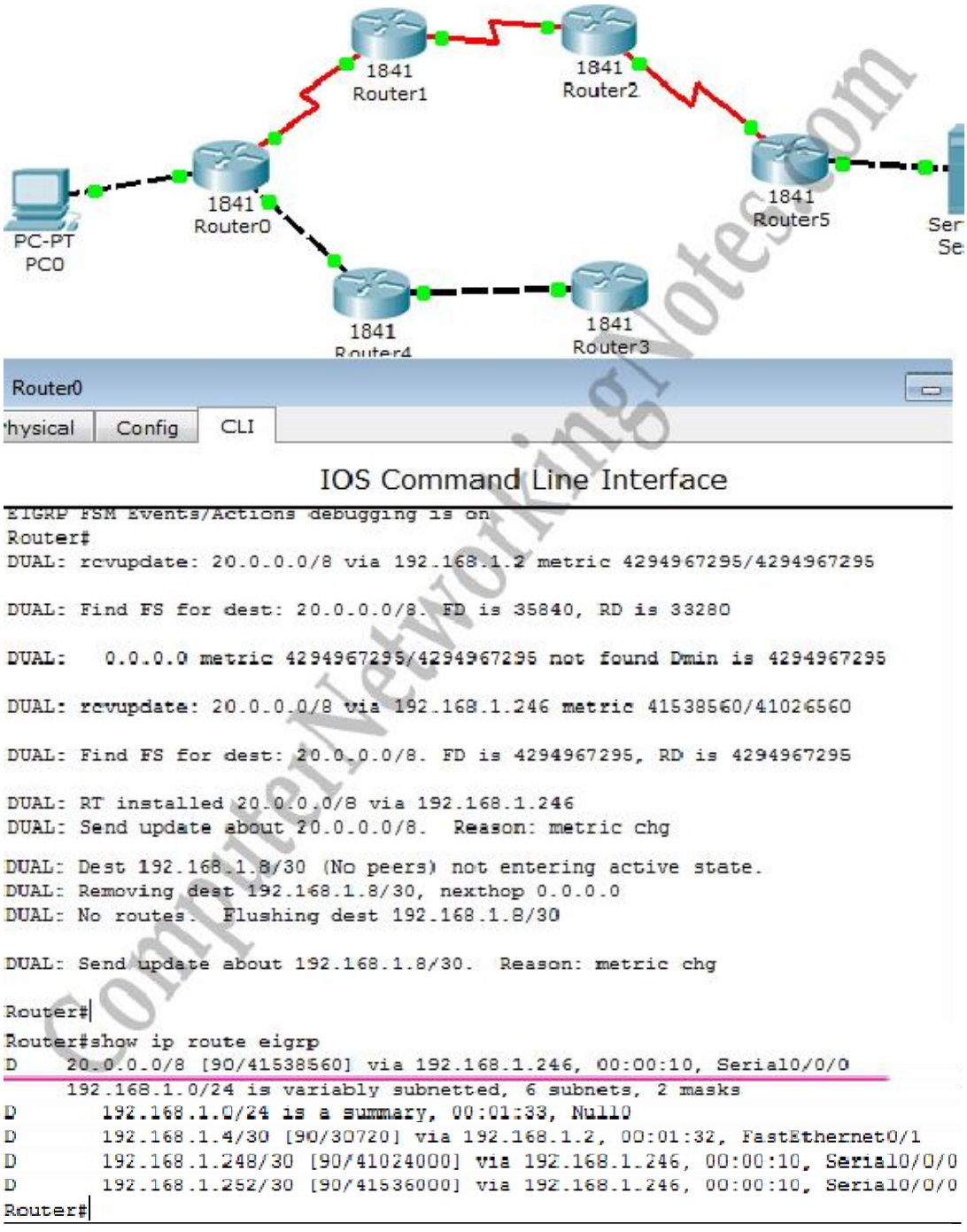
Now suppose route1 is down. We can simulate this situation by removing the cable attached between Router3 [Fa0/1] and Router5 [Fa0/1].



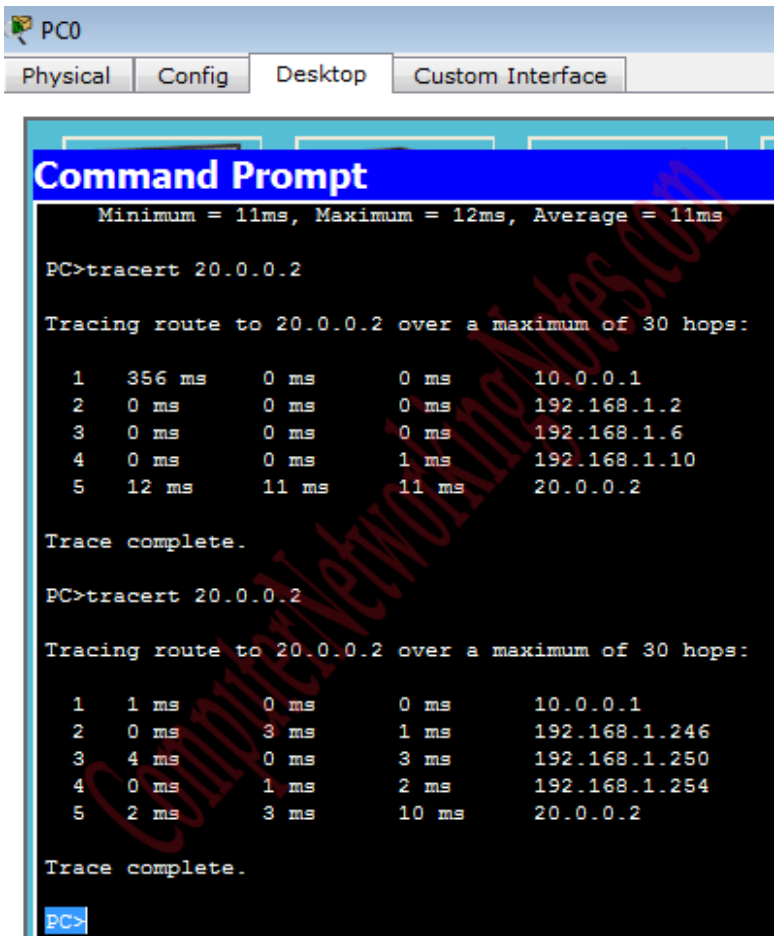
Okay our primary route went down. What will be happen now?

EIGRP will look in topology table for next available routes. If single alternative is available, it will be selected. If multiple routes are available, it will select the route with the lowest metric value.

We can use **show ip route eigrp** command again to see the selected route.



Run **tracert** command again from PC0 to verify the change.



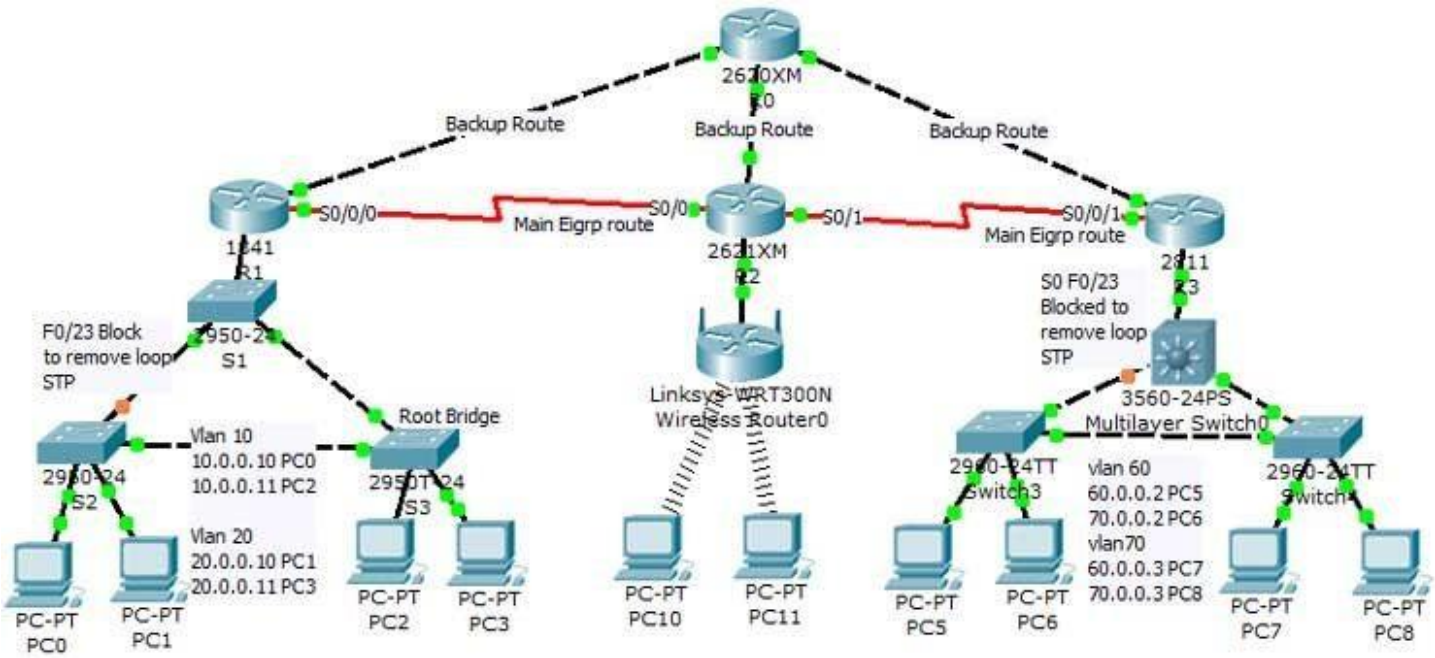
That's all for this article. Before closing just do a quick recap of important commands.

EIGRP configuration commands cheat sheet

Command	Description
Router(config)#router eigrp 20	Enable EIGRP with AS number 20. AS number must be same on all routers to become EIGRP neighbor.
Router(config-router)#network 10.10.0.0	Enable EIGRP on interfaces which belongs to network 10.0.0.0/8. [Classful implementation].
Router(config-router)#network 10.10.0.0 0.0.255.255	Enable EIGRP on interfaces which belongs to network 10.10.0.0/16. [Classless implementation – Wildcard mask method].
Router(config-router)#network 10.10.0.0 255.255.0.0	Enable EIGRP on interfaces which belongs to network 10.10.0.0/16. [Classless implementation – Subnet mask method].
Router(config-router)#no network 10.10.0.0	Disable EIGRP on interfaces which belongs to network 10.0.0.0/8.

Router(config-router)#no network 10.10.0.0 0.0.255.255	Disable EIGRP on interfaces which belongs to network 10.10.0.0/16.
Router(config-router)#no network 10.10.0.0 255.255.0.0	Disable EIGRP on interfaces which belongs to network 10.10.0.0/16.
Router(config-router) #metric weights tos k1 k2 k3 k4 k5	Enable/Disable K values used in metric calculation formula. Default values are tos=0, k1=1, k2=0, k3=1, k4=0, k5=0 Tos(type of service), K1(bandwidth), K2(load), K3(delay), K4(reliability), K5(MTU). By default only K1 and K3 are enabled.
Router(config-router)#auto-summary	Enable auto summarization feature of EIGRP. (Default – disable)
Router(config-router)#no auto-summary	Disable auto summarization feature of EIGRP.
Router(config)#no router eigrp 20	Disable EIGRP routing process 20.
Router(config-if)#bandwidth 64	Set bandwidth to 64Kbps. Used to influence the metric calculation.
Router#show ip eigrp neighbors	Display the neighbor table in brief.
Router#show ip eigrp neighbors detail	Display the neighbor table in detail. Used to verify whether a neighbor is configured as stub router or not.
Router#show ip eigrp interfaces	Display information about all EIGRP interfaces.
Router#show ip eigrp interfaces serial 0/0	Display information about a particular EIGRP interface.
Router#show ip eigrp interfaces 20	Display information about EIGRP interfaces running AS process 20.
Router#show ip eigrp topology	Displays the topology table.
Router#show ip eigrp traffic	Displays the number and type of packets sent and received.
Router#show ip route eigrp	Display EIGRP route from routing table.
Router#debug eigrp fsm	Displays the events or actions related to feasible successor metrics (FSM).
Router#debug eigrp packet	Displays the events or actions related to EIGRP packets.
Router#no debug eigrp fsm	Turn off debug message related to feasible successor metrics (FSM).
Router#no debug eigrp packet	Turn off debug message related to EIGRP packets.

Configure EIGRP with RIP on same network



R0		
Port	IP address	Connected to
F0/0	80.0.0.1	R1 F0/1
F1/0	90.0.0.1	R2 F0/1
F1/1	100.0.0.1	R3 F0/1
R1		
Port	IP address	Connected to
F0/0.10	10.0.0.1	S1 F0/24
F0/0.20	20.0.0.1	S1 F0/24
F0/1	80.0.0.2	R0 F0/0
S0/0/0	30.0.0.1	R2 S0/0
R2		
Port	IP address	Connected to
F0/1	90.0.0.2	R0 F1/0
S0/0	30.0.0.2	R1 S0/0/0
F0/0	40.0.0.1	WR1 0/1
S0/1	50.0.0.1	R3 S0/0/1
R3		
Port	IP address	Connected to
F0/1	100.0.0.2	R0 F1/1
S0/0/1	50.0.0.2	R2 S0/1
F0/0.60	60.0.0.1	S1 G0/1
F0/0.70	70.0.0.1	S1 G0/1

Configuration of R0

First we will configure R0. To configure double click on R0 select CLI and configure it as given below

To configure and enable RIP as backup routing on R0 follow these commands exactly.

```
R0>enable
R0#sh ip interface brief
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        80.0.0.1        YES manual up                    up

FastEthernet1/0        90.0.0.1        YES manual up                    up

FastEthernet1/1        100.0.0.1       YES manual up                    up
R0#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R0(config)#router rip
R0(config-router)#network  80.0.0.0
R0(config-router)#network  90.0.0.0
R0(config-router)#network 100.0.0.0
R0(config-router)#exit
R0(config)#exit
%SYS-5-CONFIG_I: Configured from console by console
R0#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R0#
```

We need not to configure EIGRP on it as its only going to be a backup route

Configuration of R1

Now configure R1. On R1 we need to configure both RIP and EIGRP. RIP for backup and EIGRP for main route.

```
R1>enable
R1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
```

```
FastEthernet0/0        unassigned      YES manual up                    up

FastEthernet0/0.10     10.0.0.1        YES manual up                    up

FastEthernet0/0.20     20.0.0.1        YES manual up                    up

FastEthernet0/1        80.0.0.2        YES manual up                    up

Serial0/0/0            30.0.0.1        YES manual up                    up

Serial0/0/1            unassigned      YES manual administratively down down

Vlan1                  unassigned      YES manual administratively down down

R1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#router rip
R1(config-router)#network  10.0.0.0
R1(config-router)#network  20.0.0.0
R1(config-router)#network  30.0.0.0
R1(config-router)#network 80.0.0.0
R1(config-router)#exit
R1(config)#router eigrp 1
R1(config-router)#network  10.0.0.0
R1(config-router)#network  20.0.0.0
R1(config-router)#network 30.0.0.0
R1(config-router)#exit
R1(config)#exit
%SYS-5-CONFIG_I: Configured from console by console
R1#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

R1#

Why did we not include network 80.0.0.0 in EIGRP routing?

Network 80.0.0.0 is connecting R1 to R0 which is backup route. And only going to be used In the failure of main EIGRP route.

If we include network 80.0.0.0 in EIGRP routing, still R1 will take main EIGRP (via R2) route to reach R3 why?

Because by default Matrix of EIGRP is bandwidth and delay, and in this scenario backup route (R1 – R0 – R3) is using Ethernet cable link and main EIGRP route (R1 – R2 – R3) is using serial cable link. Serial link have better matrix then Ethernet link that's why R1 will take main EIGRP route to reach R3.

Configuration of R2

To configure and enable eigrp with rip routing on R2 follow these commands exactly.

```
Router>enable
R2#show ip interface brief
Interface          IP-Address      OK? Method Status          Protocol

FastEthernet0/0    40.0.0.1        YES manual up              up

FastEthernet0/1    90.0.0.2        YES manual up              up

Serial0/0          30.0.0.2        YES manual up              up

Serial0/1          50.0.0.1        YES manual up              up
R2#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#router rip
R2(config-router)#network  30.0.0.0
R2(config-router)#network  40.0.0.0
R2(config-router)#network  50.0.0.0
R2(config-router)#network 90.0.0.0
R2(config-router)#exit
R2(config)#router eigrp 1
R2(config-router)#network 30.0.0.0
```

```
R2(config-router)#
R2(config-router)#network 40.0.0.0
R2(config-router)#network 50.0.0.0
R2(config-router)#exit
R2(config)#exit
%SYS-5-CONFIG_I: Configured from console by console
R2#
```

Configuration of R3

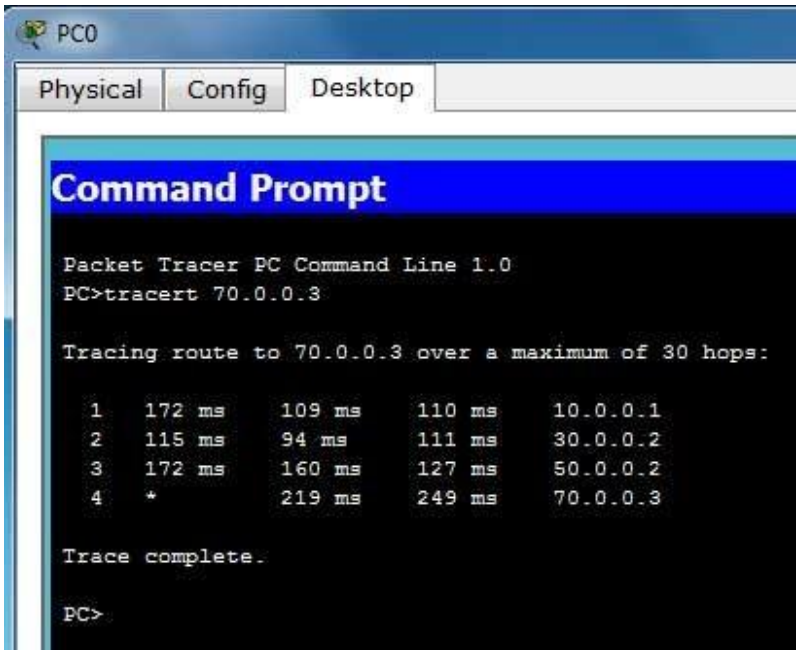
To configure and enable eigrp with rip routing on R3 follow these commands exactly.

```
Router>enable
R3#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol

FastEthernet0/0        unassigned      YES manual up                    up

FastEthernet0/0.60     60.0.0.1        YES manual up                    up

FastEthernet0/0.70     70.0.0.1        YES manual up                    up

FastEthernet0/1        100.0.0.2       YES manual up                    up

Serial0/0/0            unassigned      YES manual administratively down down

Serial0/0/1            50.0.0.2        YES manual up                    up

Vlan1                  unassigned      YES manual administratively down down

R3#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R3(config)#router rip
R3(config-router)#network  50.0.0.0
R3(config-router)#network  60.0.0.0
R3(config-router)#network  70.0.0.0
R3(config-router)#network 100.0.0.0
R3(config-router)#exit
```

```
R3(config)#router eigrp 1
R3(config-router)#network 50.0.0.0
R3(config-router)#network
R3(config-router)#network 60.0.0.0
R3(config-router)#network 70.0.0.0
R3(config-router)#exit
R3(config)#exit
%SYS-5-CONFIG_I: Configured from console by console
R3#
```

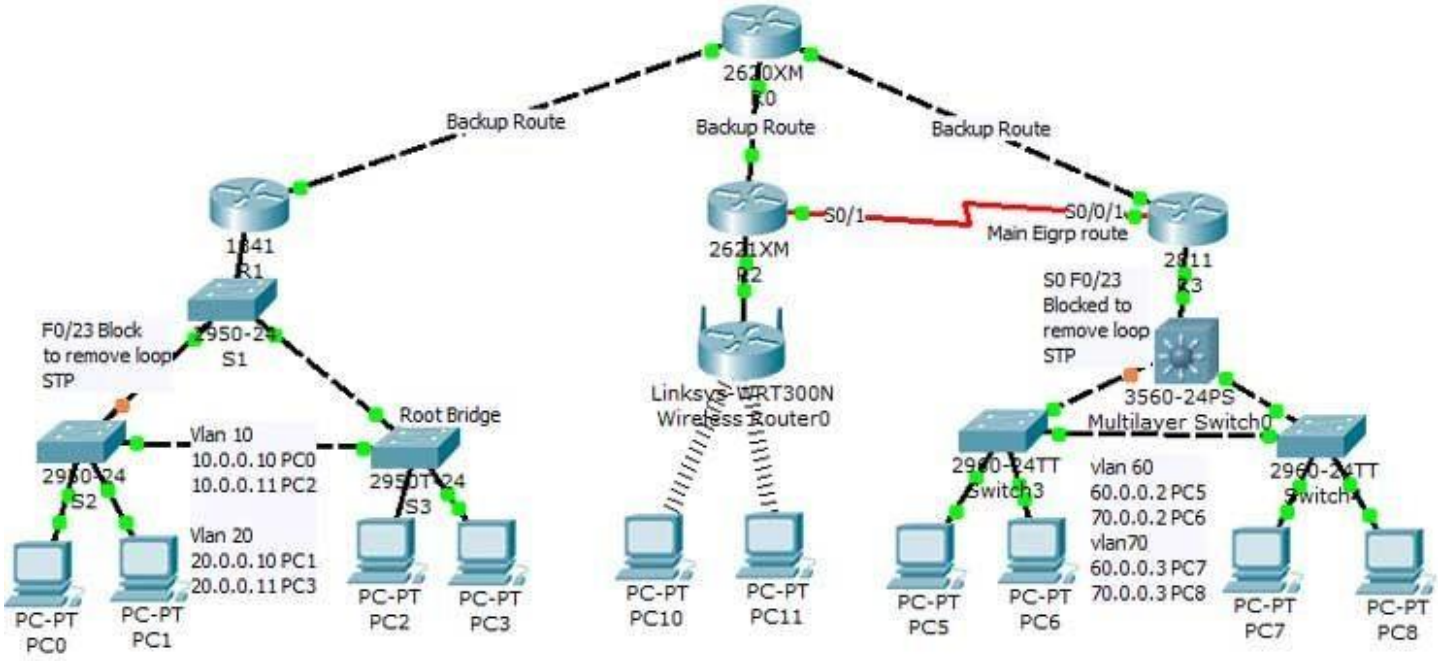
Testing of EIGRP with RIP

Now we have configured both RIP and EIGRP in this network. To test this network double click on PC-PT PC0 and select command prompt tracert 70.0.0.3



As you can see image R1 is taking Main EIGRP route to rich at R3

Now remove the serial cable by using Red Cross sign shown in right control panel



Now again tracert 70.0.0.3 from PC0

```
PC>tracert 70.0.0.3

Tracing route to 70.0.0.3 over a maximum of 30 hops:

  1   109 ms    64 ms     124 ms    10.0.0.1
  2   124 ms    109 ms    140 ms    30.0.0.2
  3   140 ms    187 ms    187 ms    50.0.0.2
  4   220 ms    265 ms    203 ms    70.0.0.3

Trace complete.

PC>tracert 70.0.0.3

Tracing route to 70.0.0.3 over a maximum of 30 hops:

  1   125 ms    94 ms     78 ms     10.0.0.1
  2   125 ms    *         156 ms    80.0.0.1
  3   156 ms    156 ms    187 ms    100.0.0.2
  4   234 ms    155 ms    265 ms    70.0.0.3

Trace complete.

PC>
```

As you can see in image this time R1 takes R0 to reach R3 because Main EIGRP route is down.

OSPF Fundamental Terminology Explained

OSPF is a complex routing protocol. It uses many terms to define its functions and operations. This tutorial explains the meaning of the terms OSPF routing protocol uses.

Link

A link is a router's interface connected to an IP subnet. When we add an interface to the OSPF process, OSPF considers the interface as a link.

State

Since a link is an interface, it has two states: up and down. The up state shows the link (interface) is operational and OSPF can reach the IP subnet connected to the link. The down state shows the link is not operational and OSPF cannot reach the IP subnet connected to the link.

Link state protocol

OSPF is a link-state protocol. Link state protocols use the Shortest Path First (SPF) algorithm to calculate the best path to a destination. To run this algorithm, link-state protocols learn the complete topology of the network. In a big size network, this feature creates scalability problems. To solve this problem, OSPF uses two concepts: autonomous systems and areas.

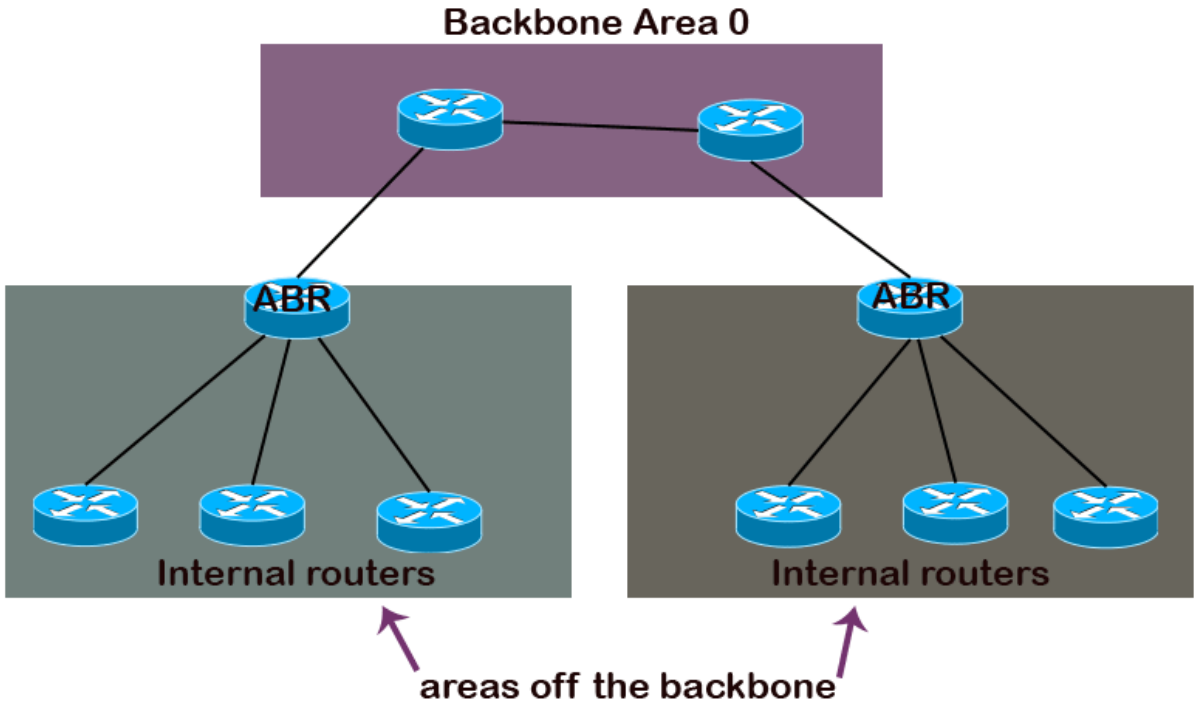
Autonomous System

An autonomous system is a group of networks under a single administrative control which can be a group of companies, a company, or a division within the company. There are two types of routing protocols: Interior Gateway Protocol (IGP) and Border Gateway Protocol (BGP). IGP routing protocols provide routing within a single AS. BGP routing protocols provide routing between different AS. OSPF is a IGP routing protocol. OSPF provides routing within a single AS.

Hierarchical Design and Areas

Within the AS, OSPF uses areas and hierarchical design. OSPF implements a two-layer hierarchy: backbone area and areas off the backbone. OSPF uses the backbone area to provide routing between areas off the backbone and areas off the backbone to control when and how much routing information is shared between routers.

An area is a group of contiguous networks. Each area uses a unique area ID. All routers in the same area use the same area ID. The following image shows how OSPF uses areas for hierarchical routing.



The following table lists some common terms used in the hierarchical design and the area concept.

Term	Description
Backbone area	A special area to which all other areas must connect.
area	A set of contiguous routers that share the same routing information
Backbone routers	Routers in the backbone area
Internal routers	Routers in areas off the backbone
ABR	A router that connects the area to the backbone area
Intra-area route	A route within the same area
Interarea route	A route between the areas

Router ID (RID)

RID is the name of the router OSPF uses to identify the router. OSPF uses the highest configured IP address as RID. If loopback interfaces are configured, OSPF uses them to choose RID. If loopback interfaces are not configured, OSPF uses all active physical interfaces to choose RID.

Neighbors

OSPF neighbors are two or more routers that have an interface in the same network and have certain configuration values same. These configure values are called neighborship requirements.

Adjacency

OSPF does not share routing updates with all neighbors. An OSPF router shares routing updates with adjacent neighbors only. An adjacency is a relationship between two adjacent routers that permits them to directly exchange routing updates.

Designated router (DR)

OSPF uses the concept of DR and BDR in the broadcast network to minimize the number of adjacencies formed. In a broadcast network, one router is selected as DR. A designated router shares routing updates with all routers.

Backup Designated Router (BDR)

A BDR is a hot standby router for the DR. The BDR keeps the backup copy of all databases running on DR. If DR fails, BDR immediately takes over the position of DR.

Hello protocol

OSPF uses the hello protocol to discover OSPF routers in the network and maintain the relationship with neighbors. Hello packets are sent to multicast address 224.0.0.5.

OSPF database

OSPF maintains two types of databases: neighborship database and topological database. It uses the neighborship database to store a list of all OSPF routers for which hello packets have been seen. It uses the topological database to store information about all LSA packets.

Link State Advertisement (LSA)

LSA is an OSPF data packet containing link-state and routing information. OSPF routers share LSA packets only with established adjacent routers.

LSDB

An LSDB is a collection of all LSAs received by the OSPF router. Each LSA has a unique sequence number. OSPF stores an LSA in LADB with its sequence number. Adjacent routers maintain the same LSDB.

OSPF Neighborship Condition and Requirement

This explains OSPF Neighborship Requirements Area ID, Authentication, Hello and Dead intervals, Stub Flag and MTU size in detail with examples.

OSPF routers share routing information only with neighbors. OSPF uses hello packets to discover neighbors in segments. A hello packet contains some essential configuration values that must be same on both routers who want to build an OSPF neighborship. In this tutorial we will explain these configuration values in detail with example.

OSPF Neighborship Requirement

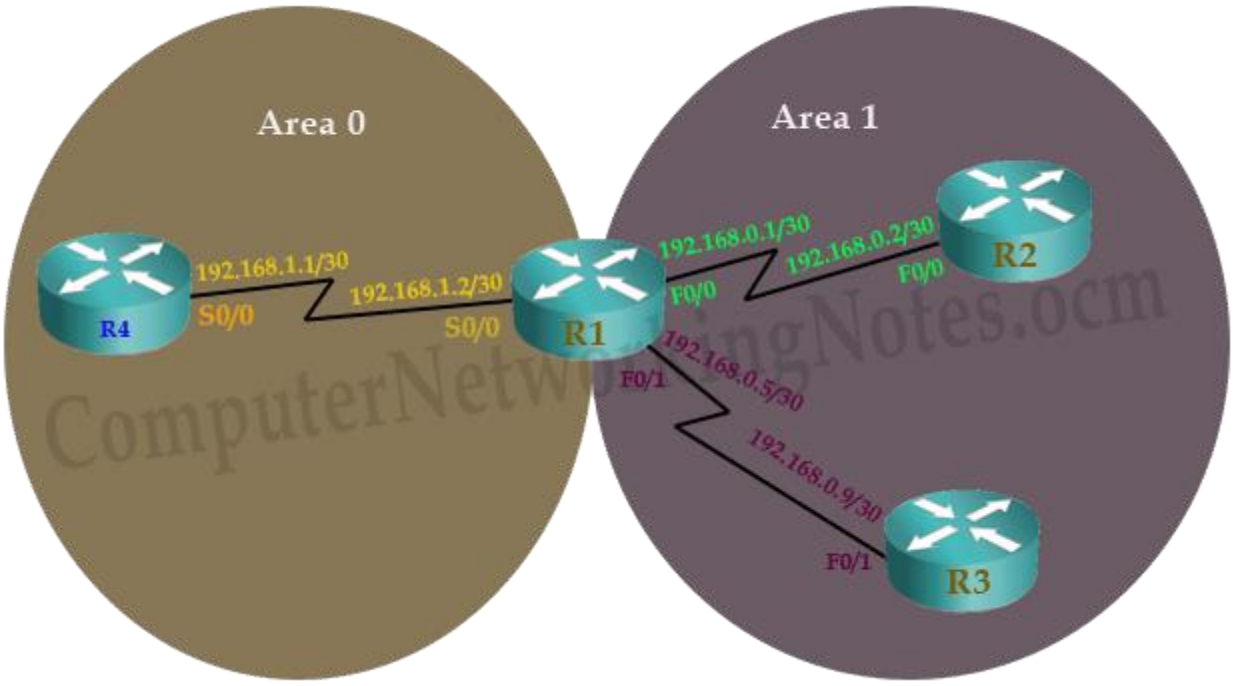
In order to become OSPF neighbor following values must be match on both routers.

- Area ID
- Authentication
- Hello and Dead Intervals
- Stub Flag
- MTU Size

Area ID

OSPF uses area concept to scale an enterprise size network. I have explained OSPF Areas in first part of this article. Just for reference, OSPF areas create a logical boundary for routing information. By default routers do not share routing information beyond the area. So in order to become neighbor, two routers must belong to same area. Here one confusing fact needs to clear. Area is associated with specific interface, not with entire router. This allows us to configure the router in multiple areas. For example a router that has two interfaces; Serial interface and FastEthernet interface, can run Serial interface in one area and FastEthernet in another area. It means link which connects two routers need be in same area including its both ends interface. Beside this interfaces should have same network ID and subnet mask.

Following figure illustrate a simple OSPF network. In this network **R1** is eligible to form neighborship with **R4** and **R2** respectively on **S0/0** and **F0/0**.



I have question for you. Why neighborship cannot be built between **R1** and **R3**?

Let's find out the answer step by step.

Both interfaces should be in same area.

Yes both interfaces (R1's Fo/1 and R3's F0/1) are in same area.

Both interfaces should be in same segment.

Yes both interfaces (R1's Fo/1 and R3's F0/1) are connected with direct link.

Both interfaces should have same subnet mask.

Yes both interfaces have same subnet mask /30.

Both interfaces should have same network ID.

No both interfaces have different network ID. **R1's F0/1** has network ID *192.168.0.4/30* while **R3's F0/1** has network ID *192.168.0.8/30*. This condition does not match. Thus these two routers on these interfaces cannot build neighborship.

Authentication

To enhance the security of network, OSPF allows us to configure the password for specific areas. Routers who have same password will be eligible for neighborship. If you want to use this facility, you need to configure password on all routers which you want to include in network. If you skip any router, that will not be able to form an OSPF neighborship.

Suppose that our network has two routers R1 and R2. Both routers are connected with direct link and meet all criteria mentioned in first requirement. What if I configure password in R1 and leave R2 as it is? Will it form neighborship with R2?

Well in this situation neighborship will not take place. Because when both routers see each other's hello packet in segment, they try to match all configure values including password field. One packet has a value in password filed while other has nothing in it. In this case routers will simply ignore each other's packet.

Hello packets and hello interval

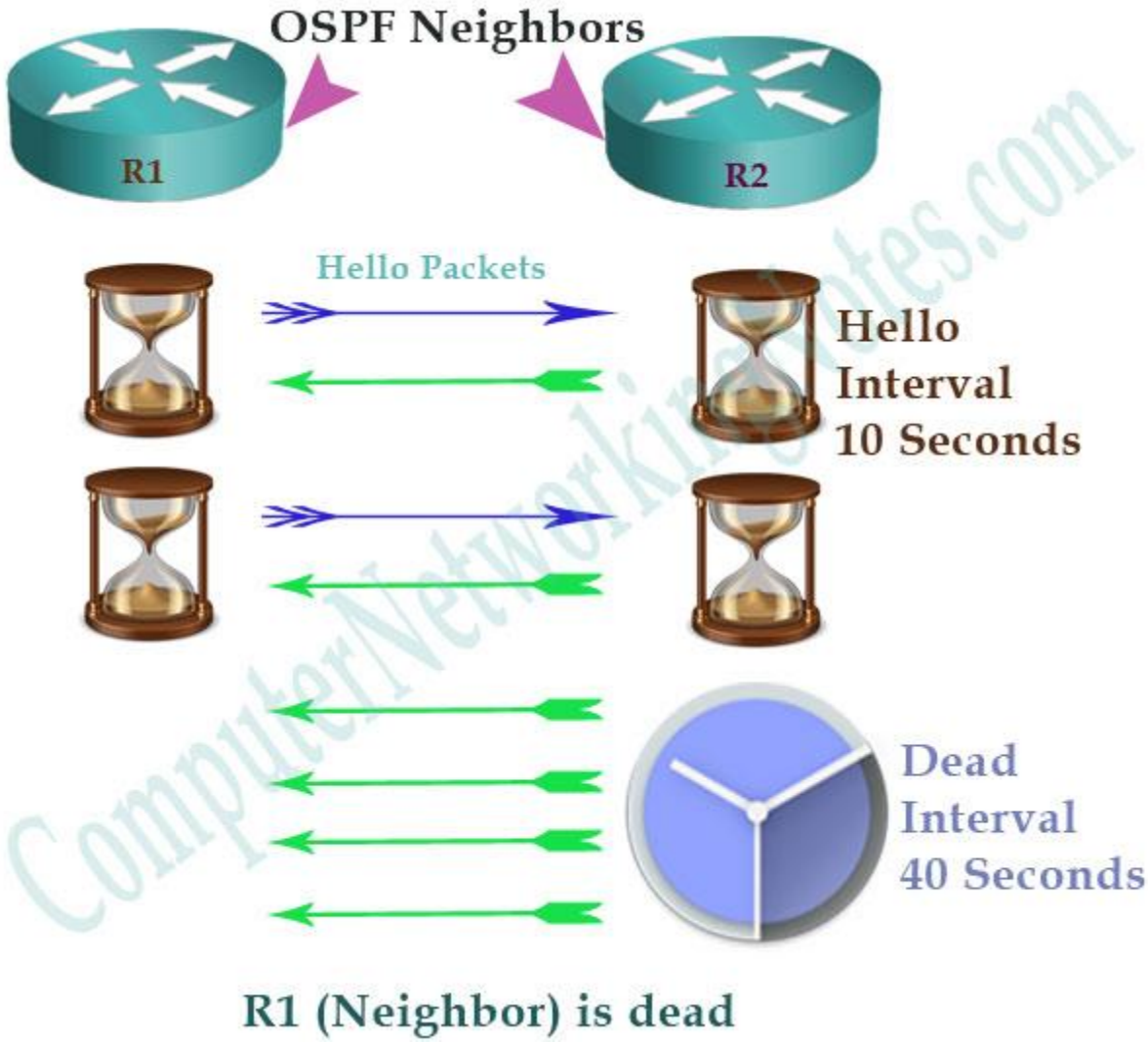
Hello packets are the special type of LSAs (Link State Advertisements) which are used to discover the neighbors in same segment. And once neighborship is built same hello packets are used to maintain the neighborship. Hello packets contain all necessary information that is required to form a neighborship. Hello packets are generated and distributed in hello interval via multicast. Hello interval is the length of time in seconds between the hello packets. Default hello interval is **10** seconds.

Dead Intervals

As we already know once neighborship is built, hello packets are used to maintain the neighborship.

So a router must see hello packets from neighbor in particular time interval. This time interval is known as dead interval. Dead interval is the number of seconds that a router waits for hello packet from neighbor, before declaring it as dead.

Default dead interval is **40** seconds. If a router does not receive hello packet in **40** seconds from neighbor it will declare that as dead. When this happens, router will propagate this information to other OSPF neighboring router via LSA message.



Hello and dead interval must be same between two neighbors. If any of these intervals are different, neighborship will not form.

Stub Area Flag

This value indicates that whether sending router belong to stub area or not. Routers who want to build OPSF neighborship must have same stub area flag.

For example we have two routers R1 and R2:-

- Both routers belong to same stub area, neighborship can be built
- Both routers belong to different stub area, neighborship cannot be built
- Both routers do not belong to any stub area, neighborship can be built
- Only one router belongs to a stub area, neighborship cannot be built

Just like another areas, Stub area also has some specific meanings in OSPF hierarchal design.

A stub area has following requirements:-

- A stub area can have only single exit point from that area.
- Stub area cannot be used as a transit area for virtual links.
- Routing from stub area to outside of the area should not have to take an optimal path.
- Any external networks (redistributed from other protocols into OSPF) should not be flooded in stub area.

Configuring a stub area reduces the size of topology table inside that area. Thus routers running in this area require less memory.

MTU

Technically MTU (Maximum Transmission Unit) is not a part of compulsory matching conditions. Still we should match this value. If this value does not match routers may stuck in Exstart/Exchange exchange stage.

Consider a situation where MTU setting between two OSPF routers does not match. If the router with the higher MTU sends a packet larger than the MTU set on the neighboring router, the neighboring router will ignores this packet. This function creates serious problem for database updates. Database updates are heavier in nature. Once an update becomes larger than the configured MTU setting, it needs to be spilt. In a case of miss match MTU, database update may lost few bytes. Due to this, OSPF will ignore that update and cannot sync with database. It will be stuck in Exstart/Exchange stage.

It is always worth to spend a little extra time in matching optional values along with compulsory values. Matching configuration values will make troubleshooting easier.

OSPF Neighbor States Explained With Example

OSPF routers go through the seven states, called Down, Attempt/Init, Two ways, Exstart, Exchange, Loading and full while building adjacency with other OSPF speaking routers. In this we will see these states in easy language with examples. Along with these states, also explanation of few other terminologies used in this process.

OSPF Neighborship states

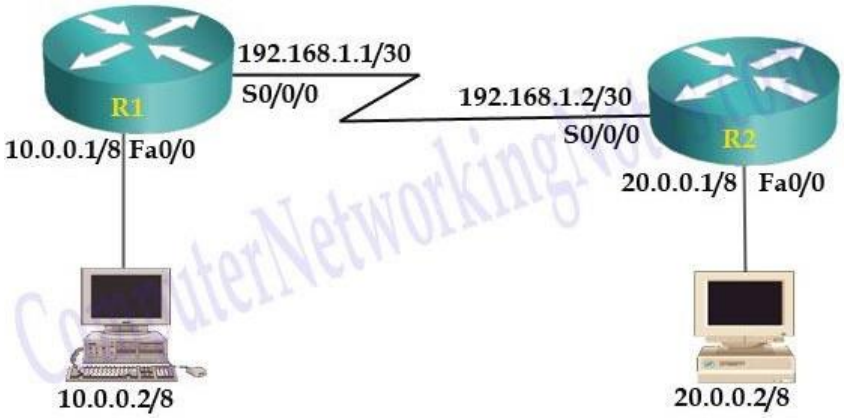
OSPF routers go through the seven states while building neighborship with other routers.

- Down state
- Attempt/Init state
- Two ways state
- Exstart state
- Exchange state
- Loading state
- Full state

Let's understand these states with a simple example. Assume that our network has two routers running OSPF routing protocol. Routers are connected with each other via serial link. We just turned on both routers simultaneously.

Down state

At this point both routers have no information about each other. R1 does not know which protocol is running on R2. Vice versa R2 have no clue about R1. In this stage OSPF learns about the local interfaces which are configured to run the OSPF instance.



In down state routers prepares themselves for neighborship process. In this state routers choose RID (Router ID). RID plays a big role in OSPF process. Before we move in next state let's understand what is RID.

RID

RID is a unique identifier of Router in OSPF network. It must be unique within the autonomous system. Routers identify each other through the RID in AS.

How do routers choose RID?

An OSPF router looks in three places for RID:-

1. Manual configuration
2. Loopback interface IP configuration
3. Active interfaces IP configuration

Manual configuration

Because RID plays a significant role in network, OSPF allows us to configure it manually. RID is 32 bit long. IP address is also 32 bit in length. We can use IP address as a RID. This gives us more flexibility over RID. For example we can use a simple and sequential IP scheme such as 1.1.1.1 for R1, 1.1.1.2 for R2, 1.1.1.3 for R3, 1.1.1.4 for R4, 1.1.1.5 for R5 and so on.

We can assign RID from OSPF sub command mode.

```
Router(config)#router ospf 1
Router(config-router)#router-id ip_address
```

If we have assigned RID manually, OSPF will not look in next two options. Suppose we did not assign it through the command. In this situation OSPF will look in next option to find the RID.

Loopback interface IP configuration

If loopback interface is configured, OSPF will choose its IP address as RID. If multiple loopback interfaces are configured, highest IP address will be chosen from all loopback interfaces configuration.

If loopback interface is not configured, OSPF will look in next and last possible place to choose the RID.

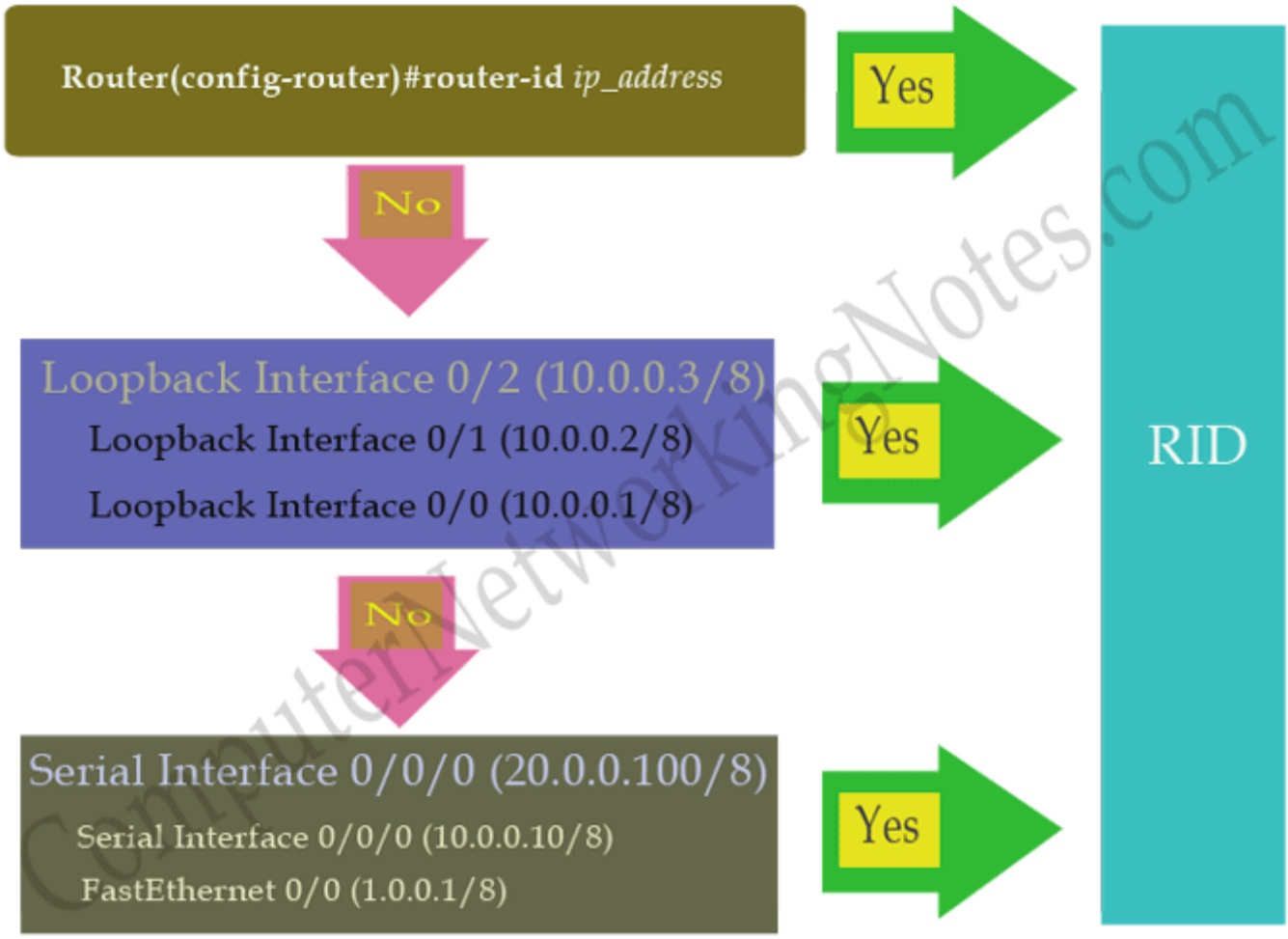
Active interface IP configuration

OSPF will choose the highest IP address from all operational IP interfaces. We should not let the OSPF to use this option. This option does not provide a fix RID which is very necessary for network stability.

This option has several reasons which may force OSPF to recalculate the RID such as Interface which IP address is chosen may go down or for troubleshooting we may enable / disable the interfaces.

Key points

- OSPF will follow the sequence (Manual configuration => Loopback interface => Active interface) of options while selecting RID. If RID is found, it will not look in next option.
- OSPF will choose IP address only from operational IP interface. Operational means interface should be listed as line is up and line protocol is up in the output of show ip interface brief command.
- When multiple IP addresses are available, OSPF will always pick highest IP address for RID.
- For network stability we should always set RID from either **router-id** command or by using loopback interfaces.
- By default Router chooses OSPF RID when it initialized. Once RID is selected it will use that RID until next reboot.
- OSPF will not consider any change in RID which we make after initialization. We have two options to implement new RID. Either reboots the router or clear the OPSF process with clear ip ospf process command.
- If OSPF fails to select the RID, it will halt the OSPF process. We cannot use OSPF process without RID.



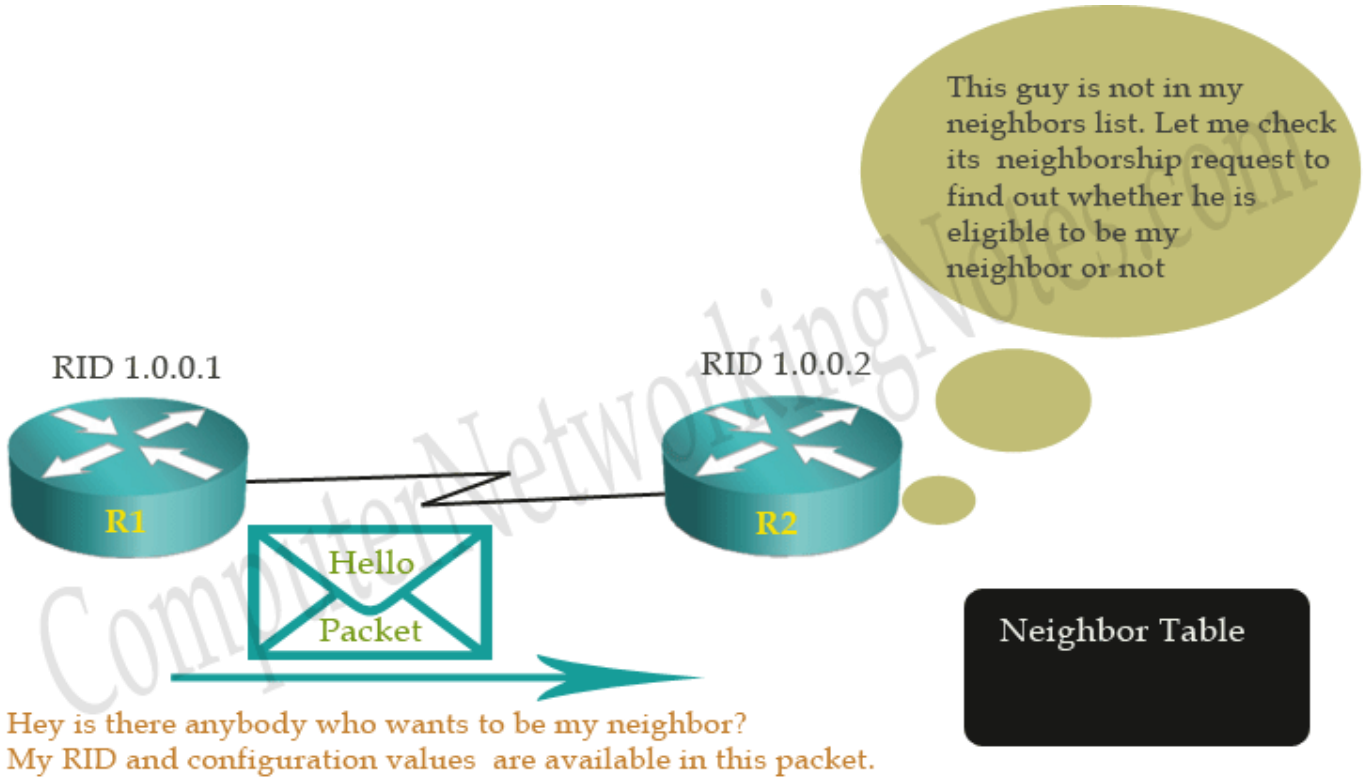
In down state router do following

- Choose RID and initialize the OSPF process
- Run OSPF instance on local interfaces which are configured through the network command such as **R1(config-router)#network *10.0.0.0 0.0.255.255* area *0***.
- Collect necessary information for Hello packet such RID and configuration values which are required to build the neighborship.

Attempt/Init state

Neighborship building process starts from this state. R1 multicasts first hello packet so other routers in network can learn about the existence of R1 as an OSPF router. This hello packet contains Router ID and some essential configuration values such as area ID, hello interval, hold down timer, stub flag and MTU. Essential configuration values must be same on routers who want to build an OSPF neighborship.

In previous part of this article I explained essential configuration values in detail with example. For this tutorial I assume that these values match on both routers. If essential configuration values match, R2 will add R1 in his neighbor Table.



In Init state routers do following

- R1 will generate a hello packet with RID and essential configuration values and send it out from all active interfaces.
- The hello packets are sent to the **multicast address *224.0.0.5***.
- R2 will receive this packet.
- R2 will read RID from packet and look in neighbor table for existing entry.
- If match found, R2 would skip neighborship building process and reset the dead interval timer for that entry.
- If OSPF does not find a match in neighbor table, it will consider R1 (sender router) as a possible OSPF neighbor and start neighborship building process.
- R2 will match its essential configuration values with values listed in packet.
- If all necessary configuration values match, R2 will add R1 in its neighbor table.

At this moment R1 has no idea about R2. R1 will learn about R2 when it will respond.

Before we enter in third state, let's have a quick look on attempt state.

Attempt

In Non-broadcast multi-access environment such as Frame Relay and X.25, OSPF uses Attempt state instead of Init state. OSPF uses this state only if neighbors are statically configured with **neighbor** command. In this situation, it does not have to discover them dynamically. As it already knows the neighbors, it will use unicast instead of multicast in this state.

Once neighborship is built, OSPF uses hello packets as keep alive. If a router does not receive a hello packet from any particular neighbor in dead interval, it will change its state to down from full. After

changing the state it will make an effort to contact the neighbor by sending Hello packets. This effort is made in Attempt state.

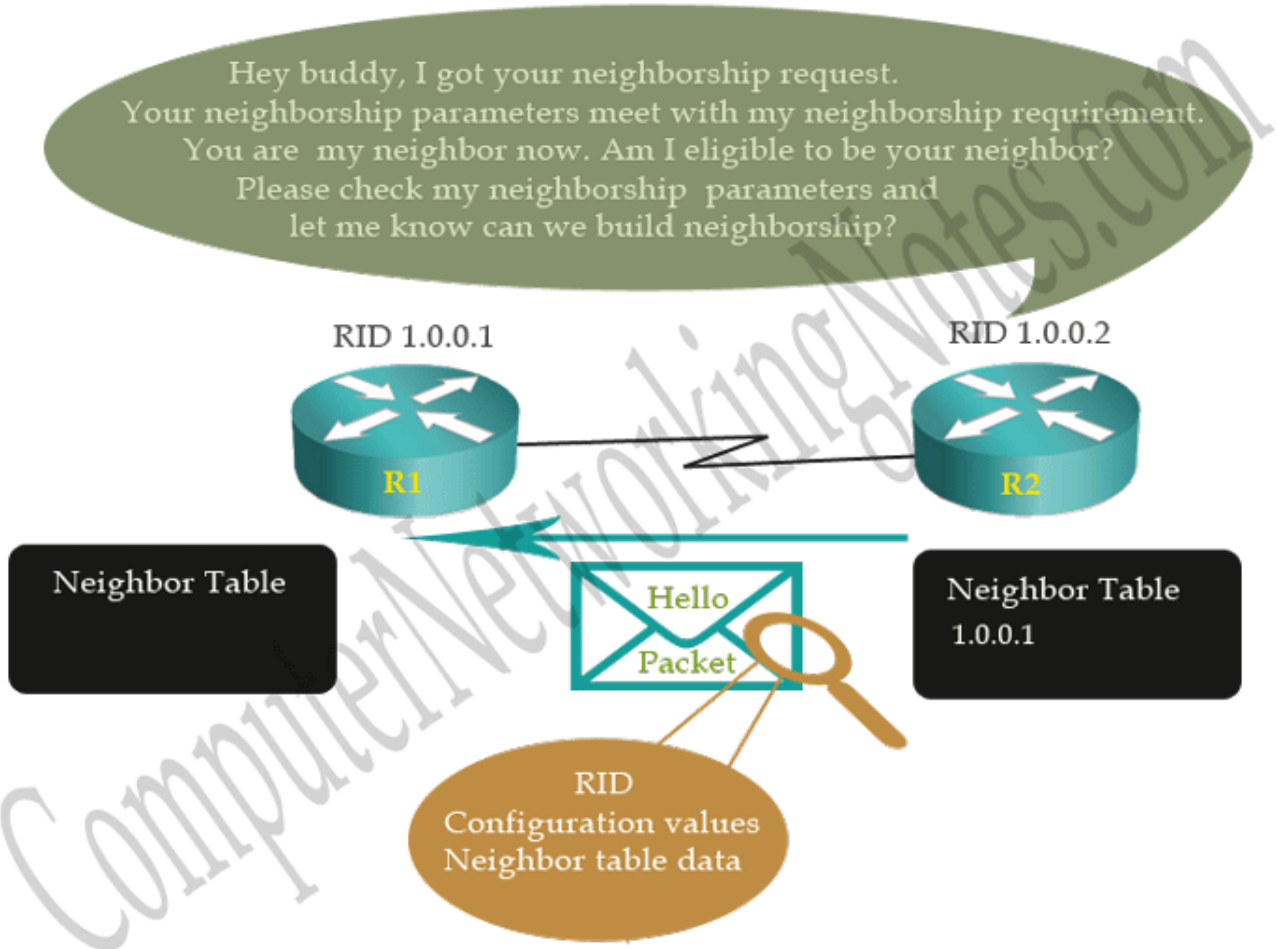
Basically Both Init and Attempt states describe similar situation where one router has sent a hello packet and waiting for response.

Two ways state

If essential configuration values match, R2 will add R1 in neighbor table and reply with its hello packet. As R2 knows the exact address of R1, it will use unicast for reply. Beside RID and configuration values, this packet also contains the R2's neighbor table data. As we know R2 has already added R1 in its neighbor table. So when R1 will see R2's neighbor table data, R1 would also see its name in this data. This will assure R1 that R2 has accepted its neighborship request.

At this point:-

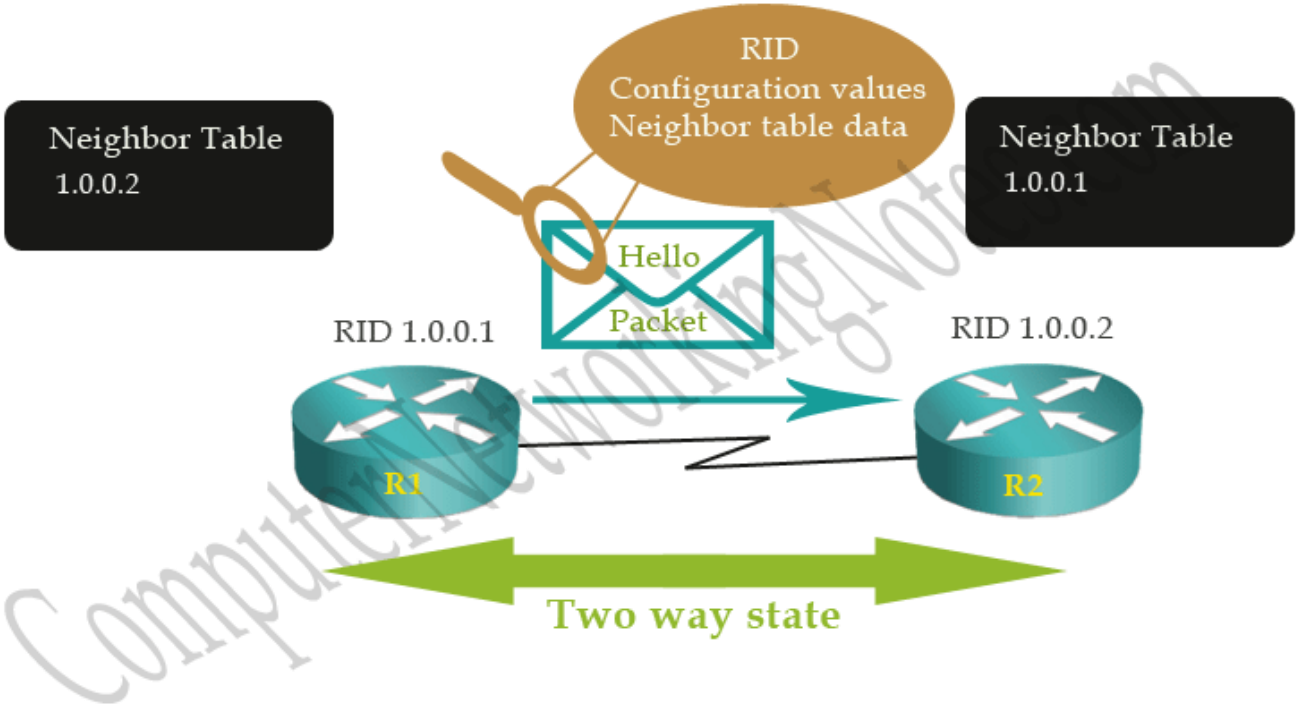
- R2 has checked all essential configuration values listed in hello packet which it received from R1.
- R2 is ready to build neighborship with these parameters.
- R2 has added R1 in its neighbor table.
- To continue the neighborship process, R2 has replied with its hello packet.
- R1 has received a reply from neighbor, with its own RID listed in R2's neighbor table.



Now it is R1's turn to take action on R2's reply. This reply would be based on hello packet which it received from R2. As we know that this hello packet contains one additional field; Neighbor table data field which indicates that this is not a regular neighbor discovery hello packet. This packet is a reply of its own request.

R1 will take following actions:-

- It will read RID from hello packet and look in its neighbor table for existing entry.
- If a match for RID found in neighbor table, it would reset the dead interval timer for that entry.
- If a match is not found in neighbor table, it would read the essential configuration values from packet.
- It will match configuration values with its own values. If values match, it will add R2's RID in neighbor table.
- If packet contains neighbor table data with its own RID, it will consider that as request to enter in two way state.
- R1 will reply with a hello packet which contains its neighbor table data.
- This packet is a confirmation of two ways state.



Fine, our routers are neighbor now. They are ready to exchange the routing information.

Before we understand how routers will exchange routing information, we need to understand the types of network. OSPF uses different types of exchange process for different types of network.

Point to point network

It is a Cisco specific network type. It connects a single pair of routers. HDLC and PPP are example of point to point network type. In this type of network:-

- All routers form full adjacencies with each other.
- Hello packets are sent using a multicast address 224.0.0.5
- No DR and BDR are required.
- All routers are considered as **AllSPFRouters**.

I will explain the terms adjacencies, DR, BDR and AllSPFRouters shortly.

Broadcast Networks

Broadcast networks are capable in connecting more than two devices. Ethernet and FDDI are the example of broadcast type network. In this type of network:-

- A single transmitted packet can be received by all attached devices.
- DR and BDR are required.
- All routers form full adjacencies only with DR and BDR.
- Routers use a multicast address 224.0.0.6 to update the DR.
- DR uses a multicast address 224.0.0.5 to update the all routers.

NMBA

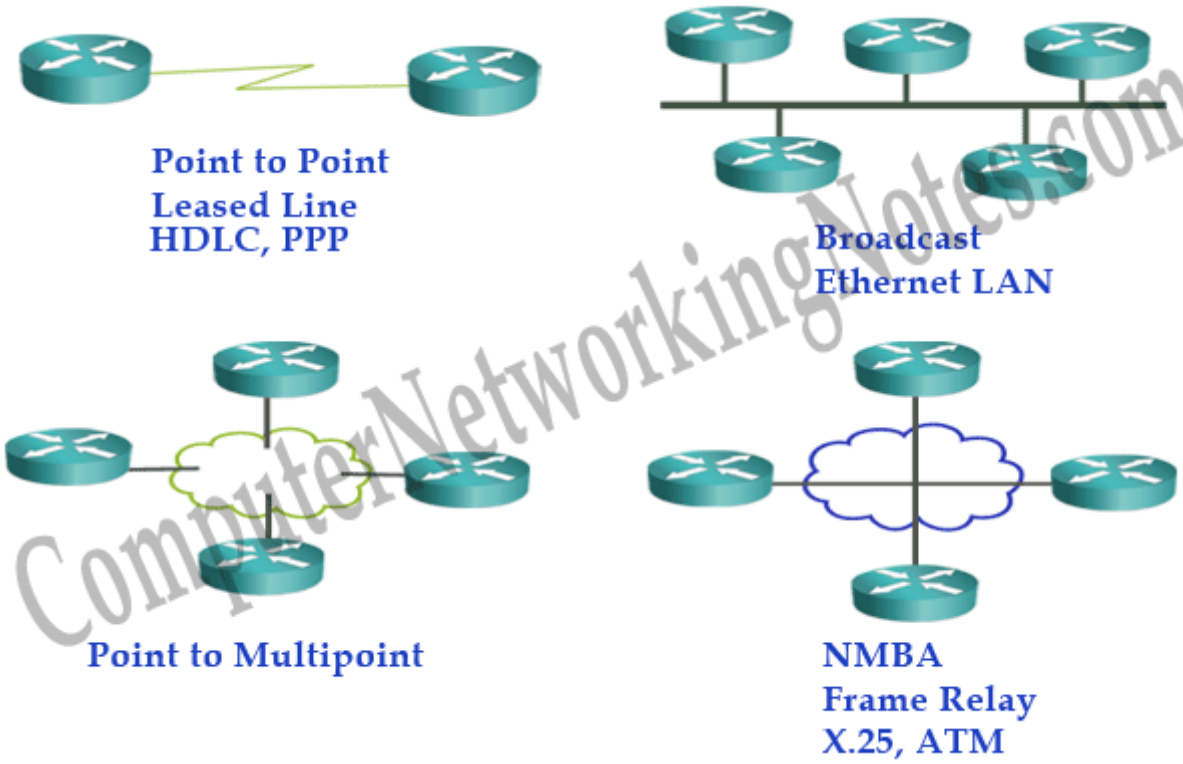
Non-broadcast Multi-access networks are also capable in connecting more than two devices. But they do not have broadcast capability. X.25 and Frame Relay are the example of NMBA type network. In this type of network:-

- As network does not have broadcast capability, dynamic network discovery will not be possible.
- OSPF neighbors must have to define statically.
- All OSFP packets are unicast.
- DR and BDR are required.

Point to multipoint

Point to multipoint is a special implementation of NMBA network where networks are configured as a collection of point to point links. In this type of network:-

- Network must be configured statically.
- No DR and BRD are selected in this type of network.
- OSPF packets are multicast.



We can divide these networks in two types;

1. Networks which need DR and BDR such as broadcast and NBMA
2. Networks which do not need DR and BDR such as point to point and point to multipoint

So what does DR and BDR actually do? Why do we need them in our network?

DR and BDR

OSPF routers in a network which need DR (Designated router) and BDR (Backup designated router) do not share routing information directly with all each other's. To minimize the routing information exchange, they select one router as designated router (DR) and one other router as backup designated router (BDR). Remaining routers are known as DROTHERs.

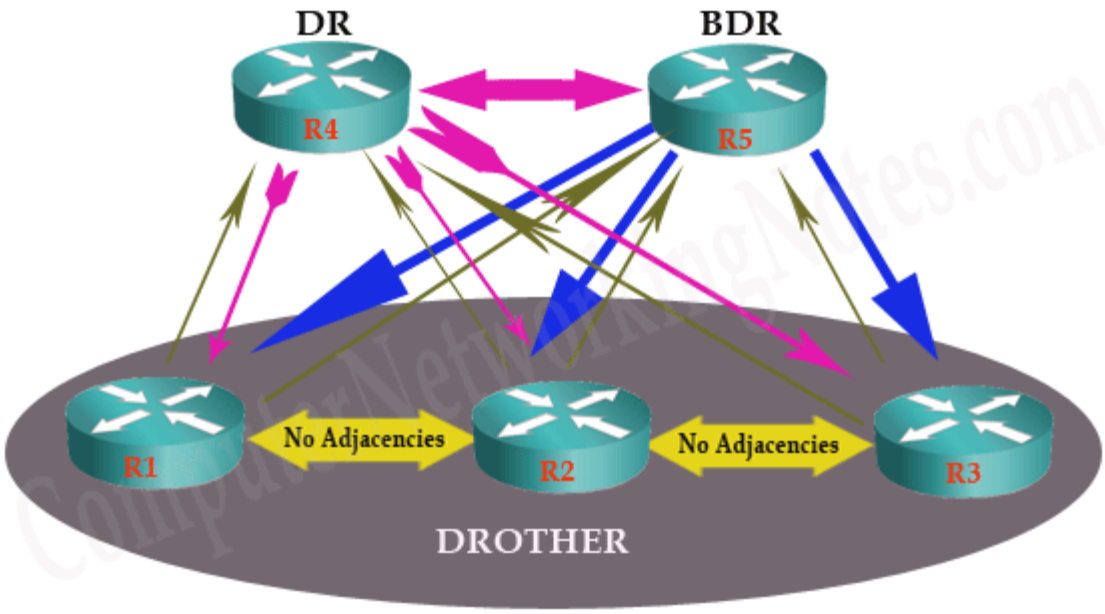
All DROTHERs share routing information with DR. DR will share this information back to all DROTHERs. BDR is a backup router. In case DR is down, BDR will immediately take place the DR and would elect new BDR for itself.

Main reason behind this mechanism is that routers have a central point for routing information exchange. Thus they need not to update each other's. A DROTHER only need to update the central point (DR) and other DROTHERs will receive this update from DR.

Practically this will cut the numbers of routing information exchange from **O(n*n)** to **O(n)** where n is the number of routers in a multi-access segment.

For example following figure illustrates a simple OSPF network. In this network R4 is selected as DR and R5 is selected as BDR. DROTHERs (R1, R2 and R3) will share routing information with R4 (DR) and

R5 (BDR), but they will not share routing information with each other. Later DR will share this information back to all DROTHERs.



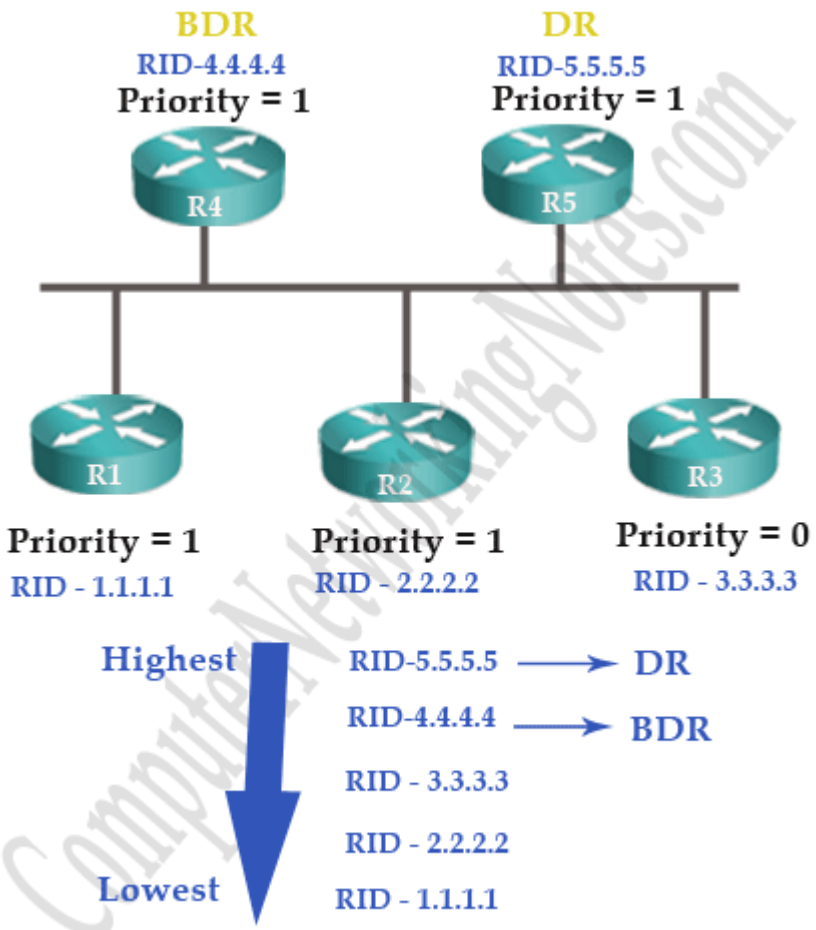
DR and BDR Election process

OSPF uses priority value to select DR and BDR. OSPF router with the highest priority becomes DR. Router with second highest priority becomes BDR. If there is a tie, router with the highest RID will be chosen.

Priority value is 8 bit in length. Default priority value is 1. We can set any value from range 0 to 255. We can change it from Interface Sub-configuration mode with *ip ospf priority* command.

We can force any router to become DR (Highest) or BDR (Second highest) by changing its priority value. If we set priority value to 0, it will never become DR or BDR.

For example following figure illustrates a simple OPSF network. In this network we have five routers. We do not want that R3 becomes DR or BDR. So we changed its default priority value to 0. Now let's see how these routers select DR and BDR.



Condition 1:- ***Use the highest priority value***

This condition says "Arrange all routes in high to low order and pick the highest for DR and second highest for BDR". If we arrange our routers in high to lower order, R3 will stand at last. Remaining routers have equal priority value. So at the end of this condition we have a tie between four routers.

Condition 2:- ***If there is a tie use the highest RID***

This condition says "If there is a tie, use RID value to choose". In our network we have a tie between four routers, so our routers will use RID to elect the DR and BDR. Arranging routers in high to low order will give us the DR and BDR.

As we know that there are two types of network; networks which do not require DR and BDR for exchange process and networks which require DR and BDR for exchange process.

In first type all routers will exchange routing information with each other's. In second type DROTHERs will exchange routing information with DR and BDR.

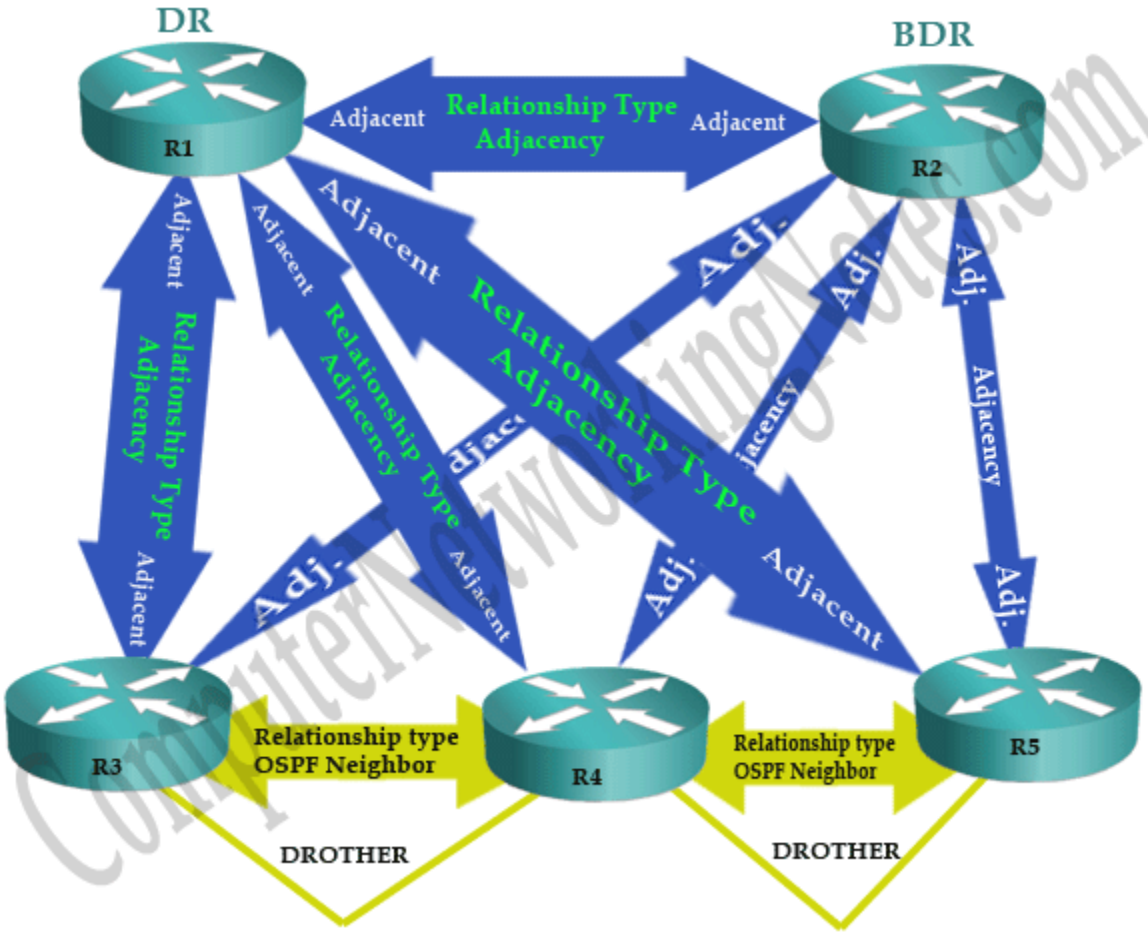
Routers which will exchange routing information are known as adjacent. Relationship between two adjacent is known as adjacency. This terminology is associated with interfaces.

A router which has two interfaces can be **adjacent** in one interface and **DROTHER** in other interface.

For example following figure illustrates an OSPF running NBMA network. In this network;

R3 will build adjacency with R1, so in this relationship they will be considered as **Adjacent**.

R3 will not build adjacency with R4, so in this relationship they will be considered only **DROTHER**.

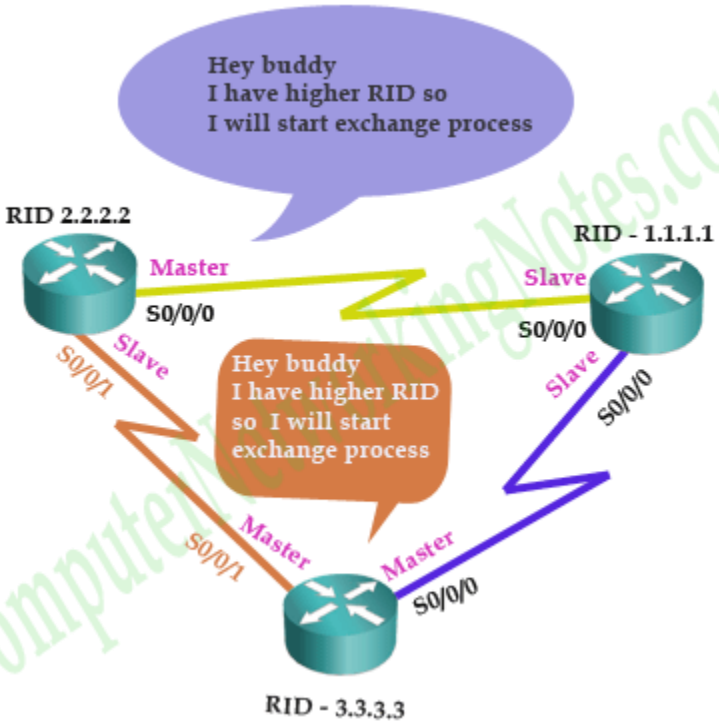


In a network which doesn't require DR and BDR, all routers will be considered as **Adjacent** and relationship between them will be considered as **Adjacency**.

Only adjacent routers will enter in next states to build the adjacency.

Exstart state

Routers who decided to build adjacency will form a master / slave relationship. In each adjacency router who has higher RID will become master and other will become slave. Do not mix Master /Slave relationship with DR/ BDR/ DROTHER relationship. Both terms look similar but have different meaning. DR/ BDR/ DROTHER relationship is built in a segment and have a wider meaning while Master / Slave relationship is built between two interfaces which need to exchange routing information. Master / Slave relationship has limited purpose. It is used to decide the Router who will start exchange process. Always Master starts exchange process.



Once routers settle down on Master/Slave, they will establish the initial sequence numbers which will be used in routing information exchange process. Sequence numbers insure that routers get most accurate information.

Exchange state

In exchange state, Master and slave decide how much information needs to be exchange. A router that has more than one interface may learn same network information from different sources. An OSFP router is smart enough to filter the updates before receiving it. It will ask only for the updates which it does not have. In this state, routers will filter the updates which need be to exchange.

Before we learn how routes will filter this information, let's understand few relative terms.

LSA and LSDB are explained in the first part of this tutorial. To maintain the flow of this article I am including the summary of these terms here again.

LSA

Link state advertisement (LSA) is a data packet which contains link-state and routing information. OSPF uses it share and learn network information.

LSDB

Every OSPF router maintains a Link state database (LSDB). LSDB is collection of all LSAs received by a router. Every LSA has a sequence number. OSPF stores LSA in LADB with this sequence number.

DBDs

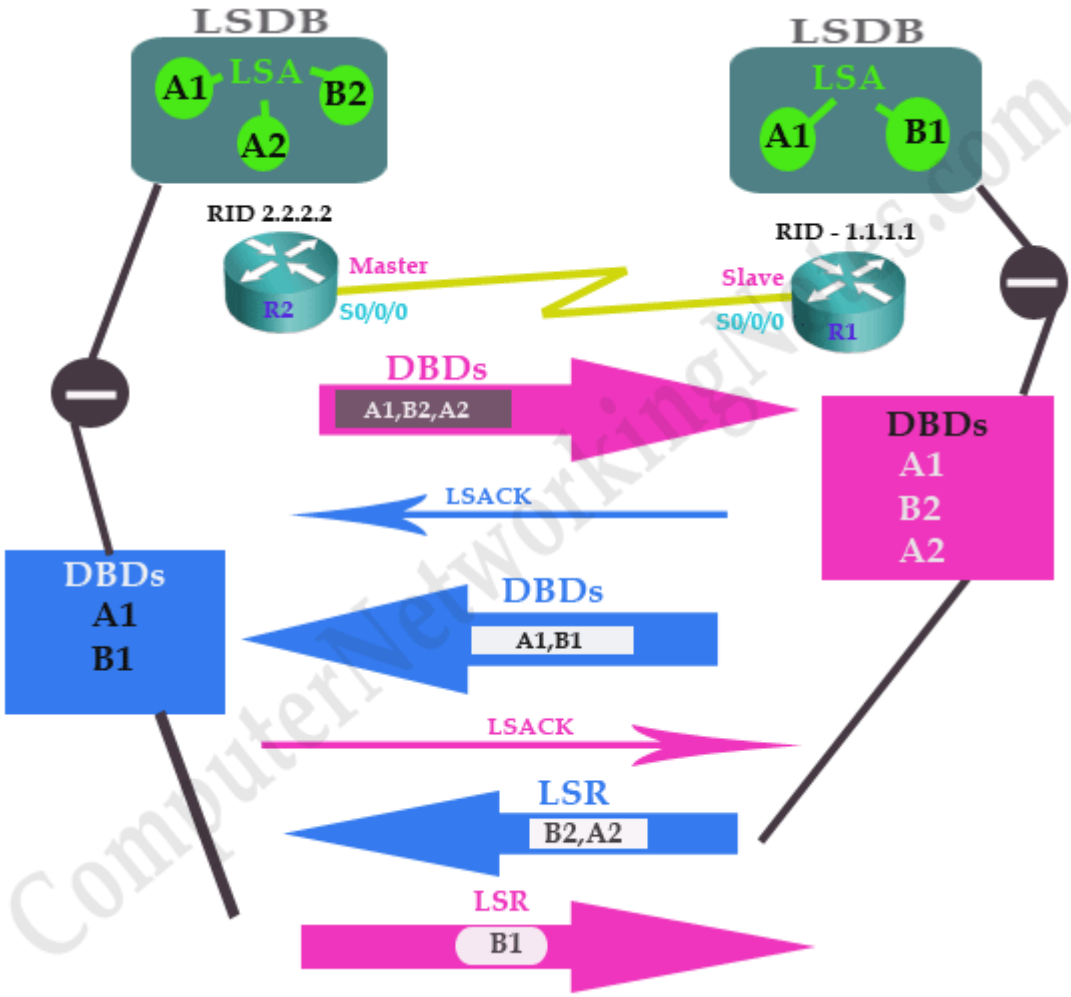
Database description packets (also referred as DDPs) contain the list of LSA. This list includes link state type, cost of link, ID of advertising router and sequence number of link. Make sure you understand this term correctly. It is only a list of all LSAs from its respective database. It does not include full LSAs.

In this state, routers exchanges DBDs. Through DBDs routers can learn which LSAs they already have. For example in following network R1 has A1, A2 and B2 LSAs in its LADB. So it will send a list of these LSAs to R2. This list is a DBDs. R2 will send an acknowledgment of receiving the list with LSACK signal. Same as R2 will send its DBDs to R1 and R1 would acknowledge that with its LSACK single.

LSR

Upon receiving DBDs, routers will compare it with their own LADB. Thus they will learn what they need to order. For example R1 received a check list (DBDs) of A1 and B1. When it will compare this list with its own LSA database (LADB), it will learn that it already has A1. So it does not need to order this LSA again. But it does not have B1, so it needs to order for this LSA. After a complete comparison, both routers will prepare a list of LSAs which they do not have in their own LADB. This list is known as LSR (Link State Request).

What other have (DBDs) – What I have (LADB) = What I need to order (LSR)



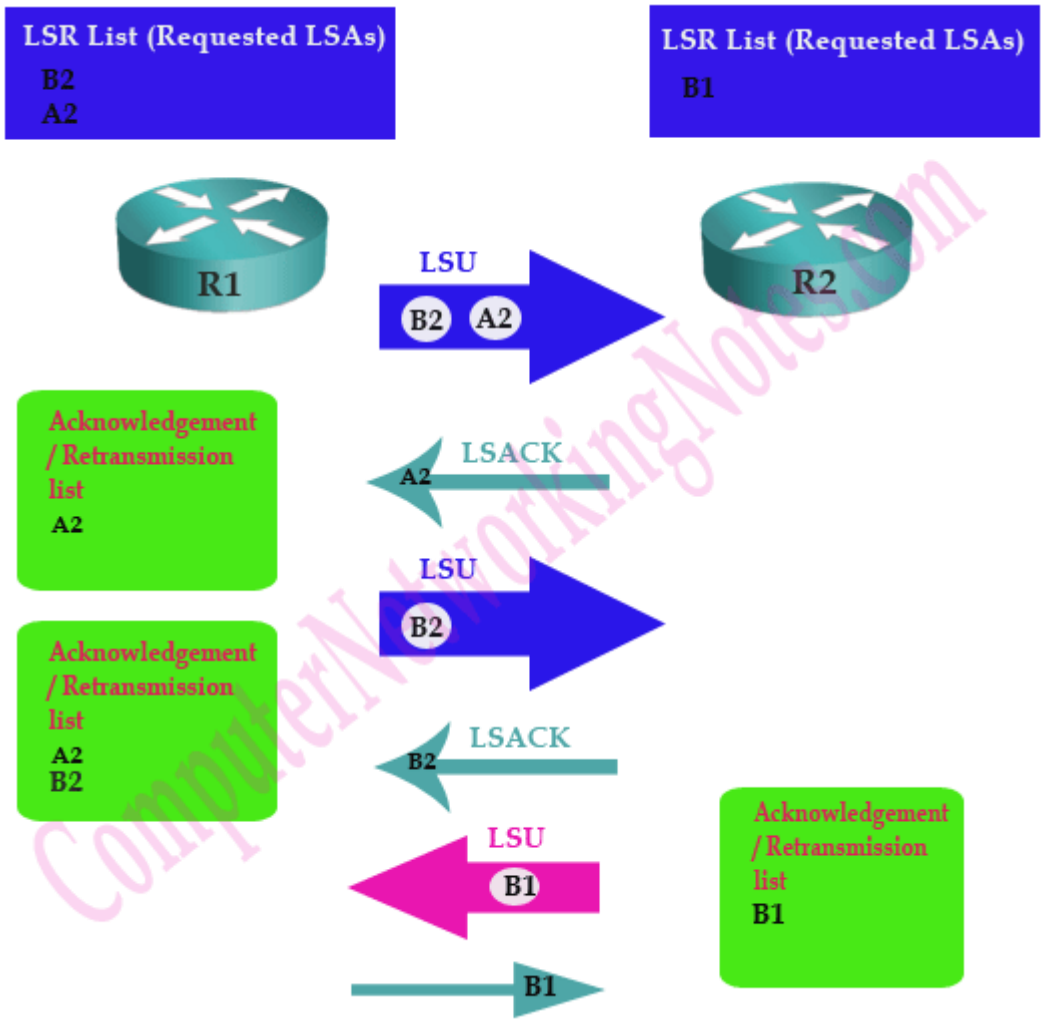
At the end of this state both routers have a list of LSAs which need to be exchanged.

Loading state

In this state actual routing information is exchanged. Routers exchange LSAs from LSR list.

Routers will use LSU (Link state update) to exchange the LSAs. Each LSA contains routing information about a particular link. Routers also maintain a retransmission list to make sure that every sent LSA is acknowledged.

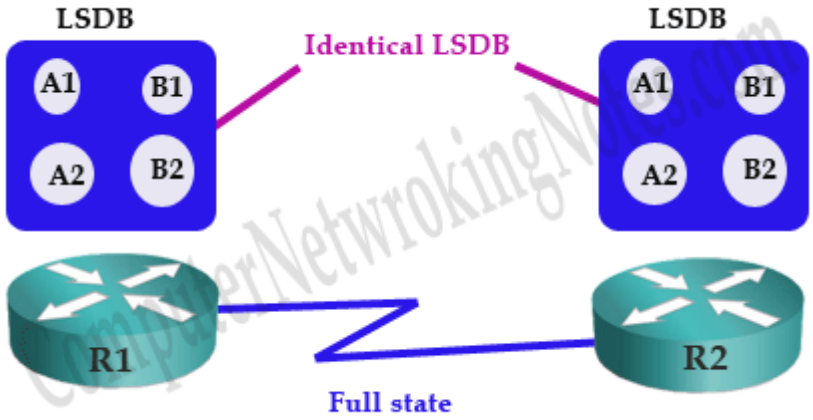
For example following figure illustrates loading state of above example. R1 sent a LSU which contain two LSAs but it received acknowledgement of only one, so it had to resend lost LSA again.



This exchange process will continue till router has any unsent LSA in LSR list.

Full state

Full state indicates that both routers has been exchanged all LSAs from LSR list. Now they have identical LSDB.



Adjacent routers remain in this state for life time. This state also referred as adjacency. If any change occurs in network, routers will go through this process again.

Maintaining adjacency

- Routers will send hello messages in hello interval.
- If a router does not receive hello message from neighbor in dead interval, it will declare that neighbor as dead.
- Once a neighbor is dead, router will flood this change to other connected neighbors.
- Beside this if router detect any change in network or receive any update, it will flood that change.
- A LSA has a default lifetime of 30 minutes. Any unchanged LSAs must be reflooded in every 30 minutes.

That's all for this part. In next part, I will explain configuration part of OSPF.

To keep this tutorial simple, I used terms *neighbor* and *adjacencies* synonymously. Technically both terms are related but have different meanings especially in OSPF. Neighboring routers are defined in RFC 2328.

Neighboring routers are the routers that have interfaces in common network.

Adjacency is a relationship formed between neighboring routers for the purpose of exchanging routing information. Not every pair of neighboring routers becomes adjacent.

OSPF Metric Cost Calculation Formula Explained

OSPF uses SPF (Shortest Path First) algorithm to select the best route for routing table. SPF algorithm was invented in 1956 by Edsger W. Dijkstra. It is also referred as Dijkstra algorithm. SPF is a quite complex algorithm. In this tutorial we will explain a simplified overview of this algorithm.

Shortest Path First (SPF) Algorithm

As we know upon initialization or due to any change in routing information an OSPF router generates a LSA. This LSA (Link State Advertisement) contains the collection of all link-states on that router. Router propagates this LSA in network. Each router that receives this LSA would store a copy of it in its LSA database then flood this LSA to other routers. After database is updated, router selects a single best route for each destination from all available routes. Router uses SPF algorithm to select the best route.

Just like other routing algorithm SPF also uses a metric component called cost to select the best route for routing table.

OSPF Metric cost

Logically a packet will face more overhead in crossing a 56Kbps serial link than crossing a 100Mbps Ethernet link. Respectively it will take less time in crossing a higher bandwidth link than a lower bandwidth link. OSPF uses this logic to calculate the cost. Cost is the inverse proportional of bandwidth. Higher bandwidth has a lower cost. Lower bandwidth has a higher cost.

OSPF uses following formula to calculate the cost

Cost = Reference bandwidth / Interface bandwidth in bps.

Reference bandwidth was defined as arbitrary value in OSPF documentation (RFC 2338). Vendors need to use their own reference bandwidth. Cisco uses 100Mbps (10^8) bandwidth as reference bandwidth. With this bandwidth, our equation would be

Cost = 10^8/interface bandwidth in bps

Key points

- Cost is a positive integer value.
- Any decimal value would be rounded back in nearest positive integer.
- Any value below 1 would be considered as 1.

Now we know the equation, let's do some math and figure out the default cost of some essential interfaces.

Default cost of essential interfaces.

Interface Type	bandwidth	Metric Calculation	Cost
Ethernet Link	10Mbps	100000000/10000000 = 10	10
FastEthernet Link	100Mbps	100000000/100000000 = 1	1
Serial Link	1544Kbps(default)	100000000/1544000 = 64.76	64

Cost of common lines

Line	Bandwidth	Metric calculation	Cost
56 Kbps line	56Kbps	100000000/56000 = 1785.71	1785
64 Kbps line	64Kbps	100000000/64000 = 1562.5	1562
128 Kbps line	128Kbps	100000000/128000 = 781.25	781
512 Kbps line	512 Kbps	100000000/512000 = 195.31	195
1 Mbps line	1Mbps	100000000/1000000 = 100	100
10 Mbps line	10Mbps	100000000/10000000 = 10	10
100 Mbps line	100Mbps	100000000/100000000 = 1	1
1 Gbps line	1Gbps	100000000/100000000 0= 0.1	1
10 Gbps line	10Gbps	100000000/10000000000 = 0.01	1

SPT (Shortest Path Tree)

OSPF router builds a Shortest Path Tree. SPT is just like a family tree where router is the root and destination networks are the leaves. SPF algorithm calculates the branch cost between leaves and root. Branch with lowest cost will be used to reach at leaf. In technical language route that has lowest cumulative cost value between source and destination will be selected for routing table.

Cumulative cost = ***Sum of all outgoing interfaces cost in route***

Best route for routing table = ***Route which has the lowest cumulative cost***

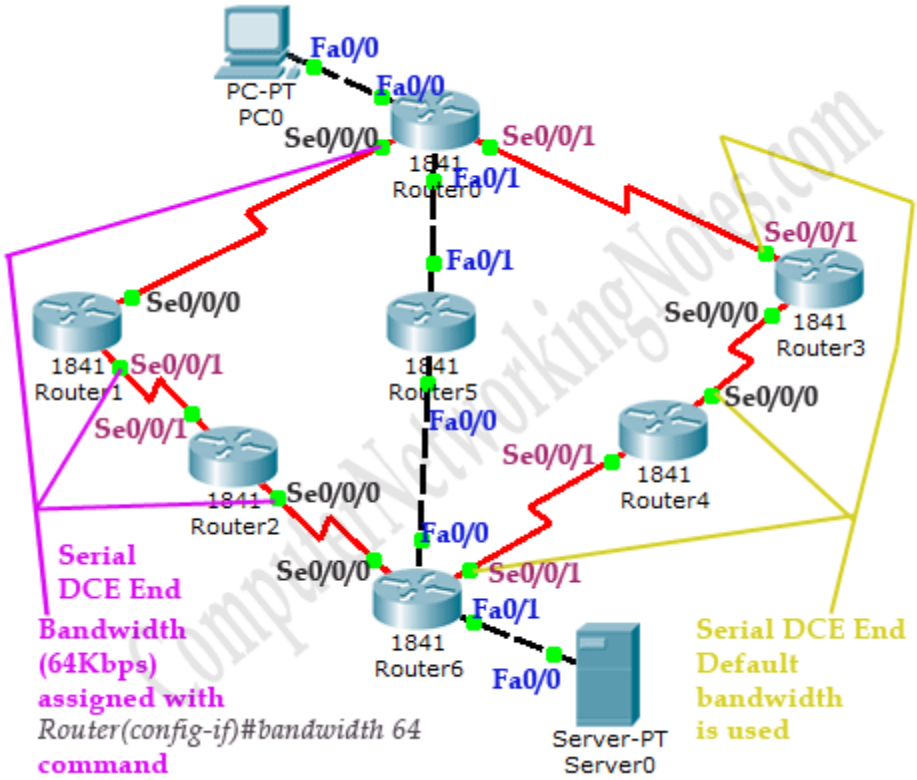
ummary

- OSPF uses SPT tree to calculate the best route for routing table.
- A SPT tree cannot grow beyond the area. So if a router has interfaces in multiple areas, it needs to build separate tree for each area.
- SPF algorithm calculates all possible routes from source router to destination network.
- Cumulative cost is the sum of the all costs of the outgoing OSPF interfaces in the path.
- While calculating cumulative cost, OSPF consider only outgoing interfaces in path. It does not add the cost of incoming interfaces in cumulative cost.
- If multiple routes exist, SPF compares the cumulative costs. Route which has the lowest cumulative cost will be chosen for routing table.

Now we have a basic understanding of SPF algorithm. In remaining part this tutorial we will explain how SPF algorithm selects the best route from available routes.

For demonstration purpose we will use same network topology which we have created in previous part of this article.

Following figure illustrates that network topology.



Open this topology in packet tracer and access CLI prompt of Router0.

Run **show ip route ospf** command from privilege mode to view all learned routes through the OSPF protocol.

As output shows, Router0 has six routes from OSPF in routing table. We will go through the each route and find out why it was chosen as the best route for routing table by OSPF.

Route 20.0.0.0

We have three routes to get 20.0.0.0/8 network. Let's calculate the cumulative cost of each route.

Via Route R0-R1-R2-R6

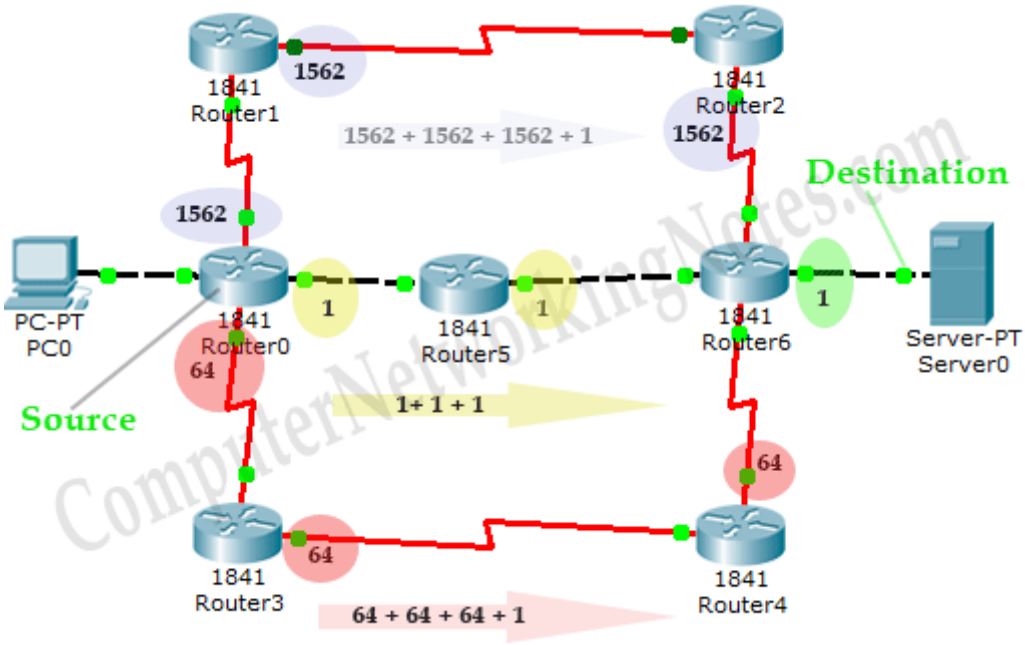
Router	**Exit Interface**	**Bandwidth**	**Metric Calculation**	**Cost**
R0	Se0/0/0	64Kbps (Manually Assigned)	100000000/64000 = 1562.5	1562
R1	Se0/0/1	64Kbps (Manually Assigned)	100000000/64000 = 1562.5	1562
R2	Se0/0/0	64Kbps (Manually Assigned)	100000000/64000 = 1562.5	1562
R6	Fa0/1	100Mbps	100000000/100000000 = 1	1
Cumulative cost of route (1562 + 1562 + 1562 + 1) = 4687				

Via route R0 – R3 – R4 – R6

Router	Exit Interface	Bandwidth	Metric Calculation	Cost
R0	Se0/0/1	1544Kbps (Default)	100000000/1544000 = 64.76	64
R3	Se0/0/0	1544Kbps (Default)	100000000/1544000 = 64.76	64
R2	Se0/0/1	1544Kbps (Default)	100000000/1544000 = 64.76	64
R6	Fa0/1	100Mbps	100000000/100000000 = 1	1
Cumulative cost of route (64 + 64 + 64 + 1) = 193				

Via route R0 – R5 – R6

Router	Exit Interface	Bandwidth	Metric Calculation	Cost
R0	Fa0/1	100Mbps	100000000/100000000 = 1	1
R5	Fa0/0	100Mbps	100000000/100000000 = 1	1
R0	Fa0/1	100Mbps	100000000/100000000 = 1	1
Cumulative cost of route (1+ 1 + 1) = 3				



Among these routes, route R0-R5-R6 has the lowest cumulative cost. So it was selected as the best route for routing table.

Route 192.168.0.4

Via Route R0 – R1

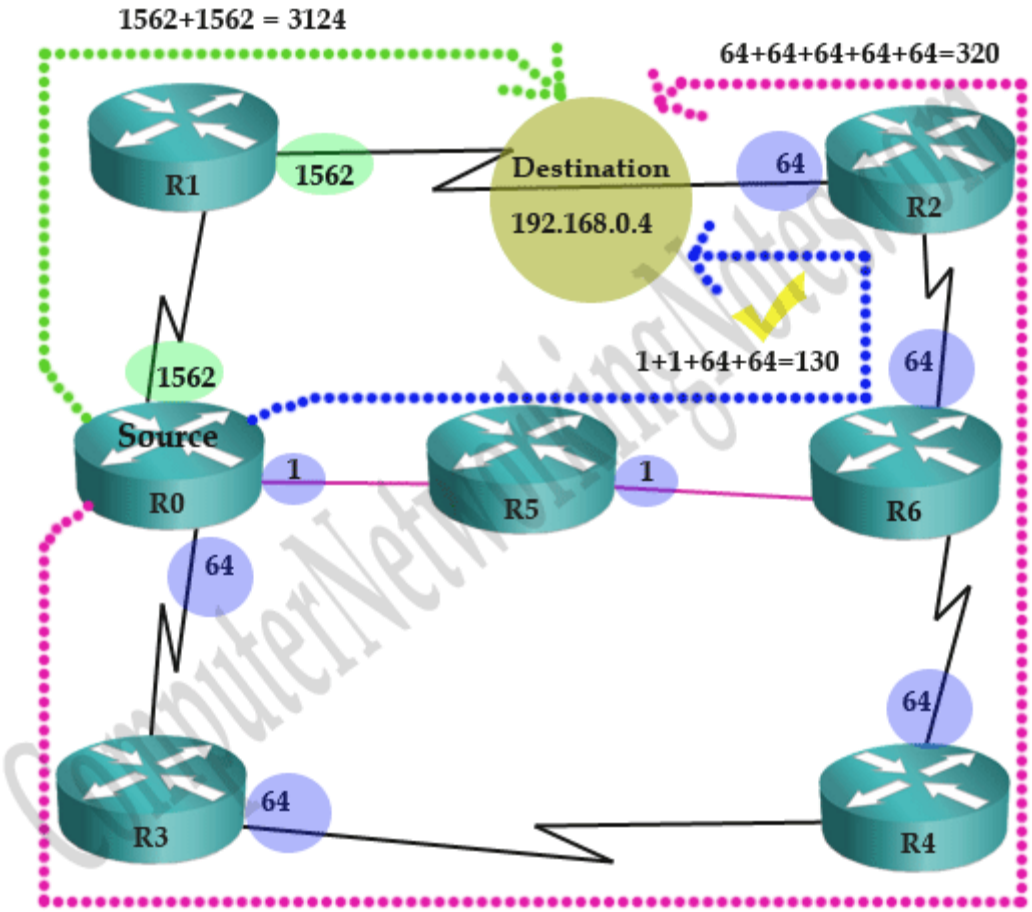
R0's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 cost (1562) = 3124 (Cumulative cost)

Via Route R0 – R3 – R4 – R6 – R2

R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) + R6's Serial 0/0/0 cost (64) + R2's Serial 0/0/1 cost (64) = 320 (Cumulative cost)

Via Route R0 – R5 – R6 – R2

Ro's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/0 cost (64) +R2's Serial 0/0/1 cost (64) = 130 (Cumulative cost)



Among these routes, Route R0 – R5 – R6 – R2 has the lowest cost so it was picked for routing table.

Route 192.168.0.8

Via Route R0 – R1

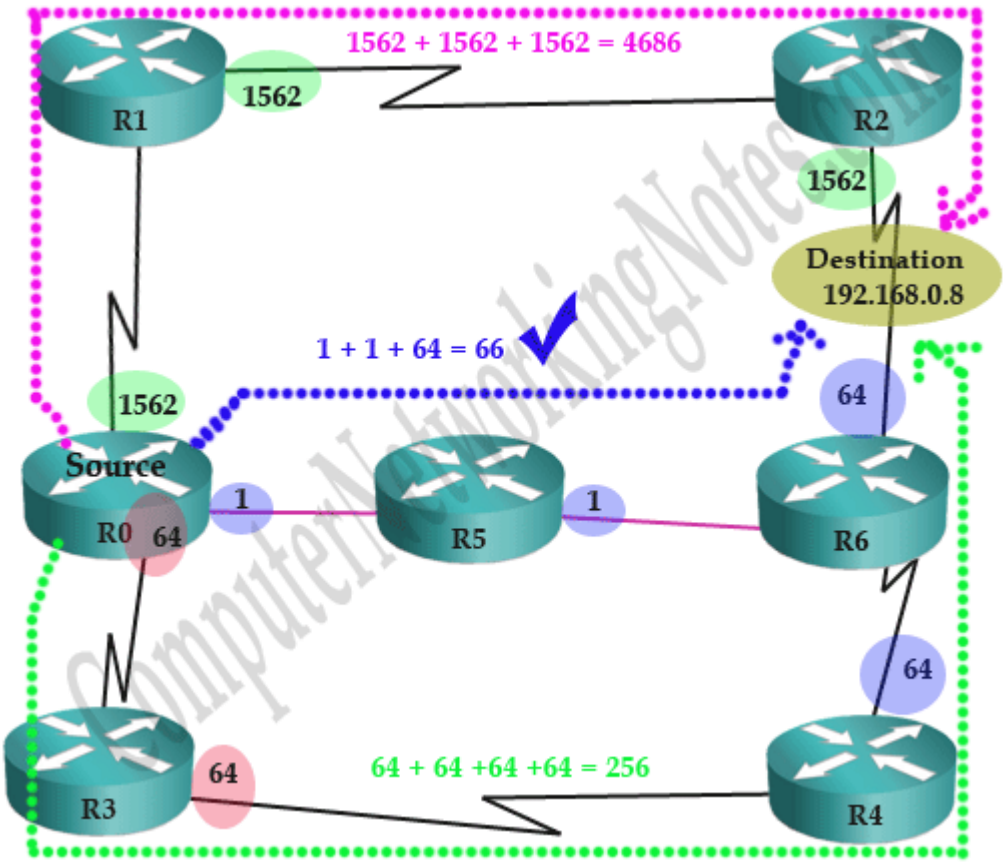
R0's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 cost (1562) + R2's Serial 0/0/0 (1562) = 4686 (Cumulative cost)

Via Route R0 – R3 – R4 – R6

R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) + R6's Serial 0/0/0 cost (64) = 256 (Cumulative cost)

Via Route R0 – R5 – R6

Ro's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/0 cost (64) = 66 (Cumulative cost)



Among these routes, Route R0 – R5 – R6 has the lowest cost so it was picked for routing table.

Route 192.168.1.4

Via Route R0 – R1 – R2 – R6

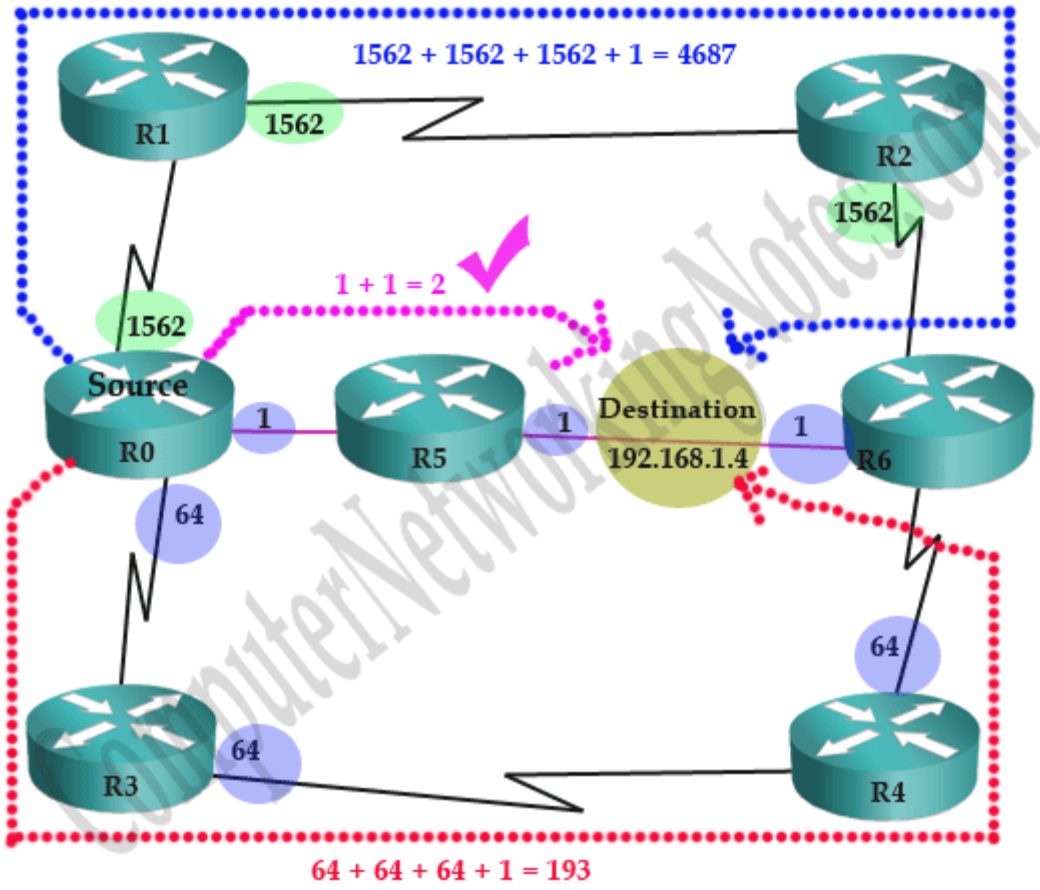
R0's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 (1562) + R2's Serial 0/0/0 (1562) + R6's FastEthernet 0/0 (1) = 4687 (Cumulative cost)

Via R0 – R3 – R4 – R6

R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) + R6's FastEthernet 0/0 (1) = 193

Via R0 – R5

R0's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) = 2



Among these routes, Route R0 – R5 has the lowest cost so it was selected as the best route.

Route 192.168.2.4

Via Route R0 – R1 – R2 – R6 – R4

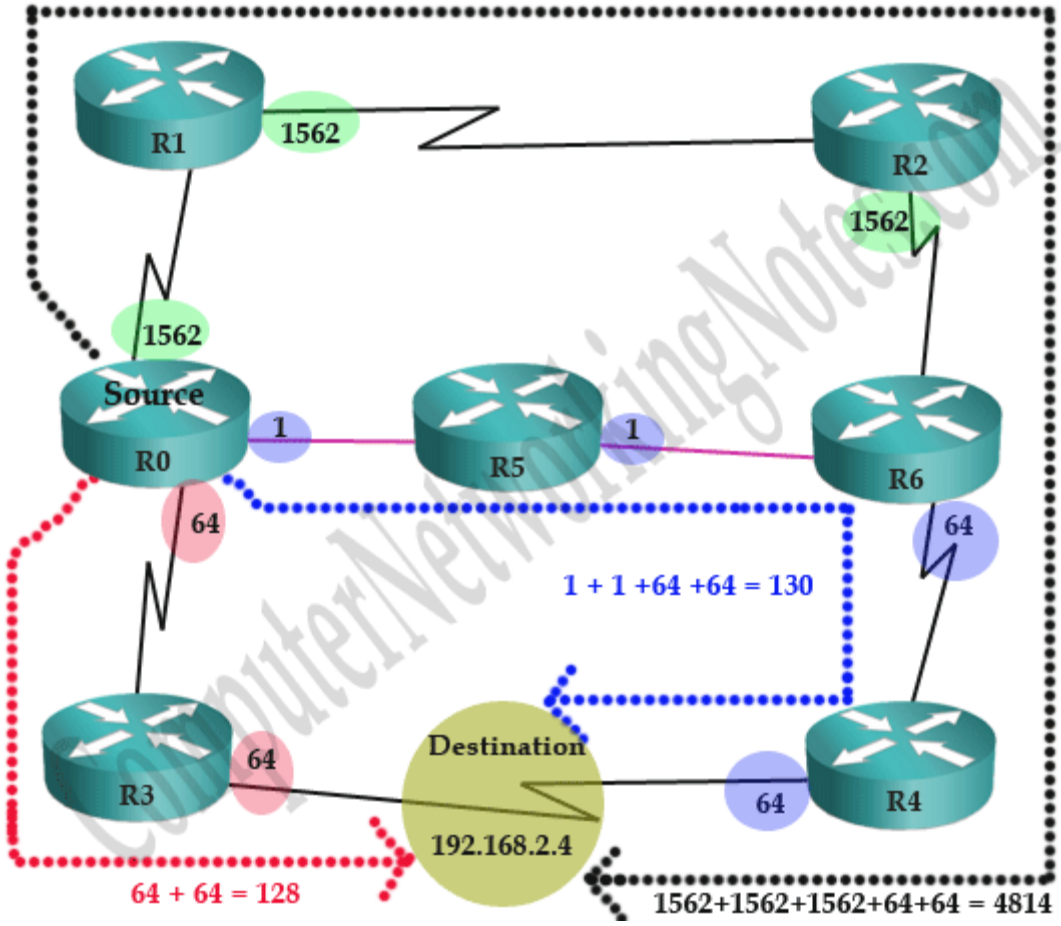
R0's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 cost (1562) + R2's Serial 0/0/0 cost (1562) + R6's Serial 0/0/1 cost (64) + R4's Serial 0/0/0 cost (64) = 4814

Via Route R0 – R5 – R6 – R4

R0's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/1 (64) + R4's Serial 0/0/0 cost (64) = 130

Via Route R0 – R3

R0's Serial 0/0/1 cost (64) + R3's serial 0/0/0 cost (64) = 128



Among these routes, Route R0 - R3 has the lowest cost for destination 192.168.2.4.

Route 192.168.2.8

Via Route R0 – R3 – R4

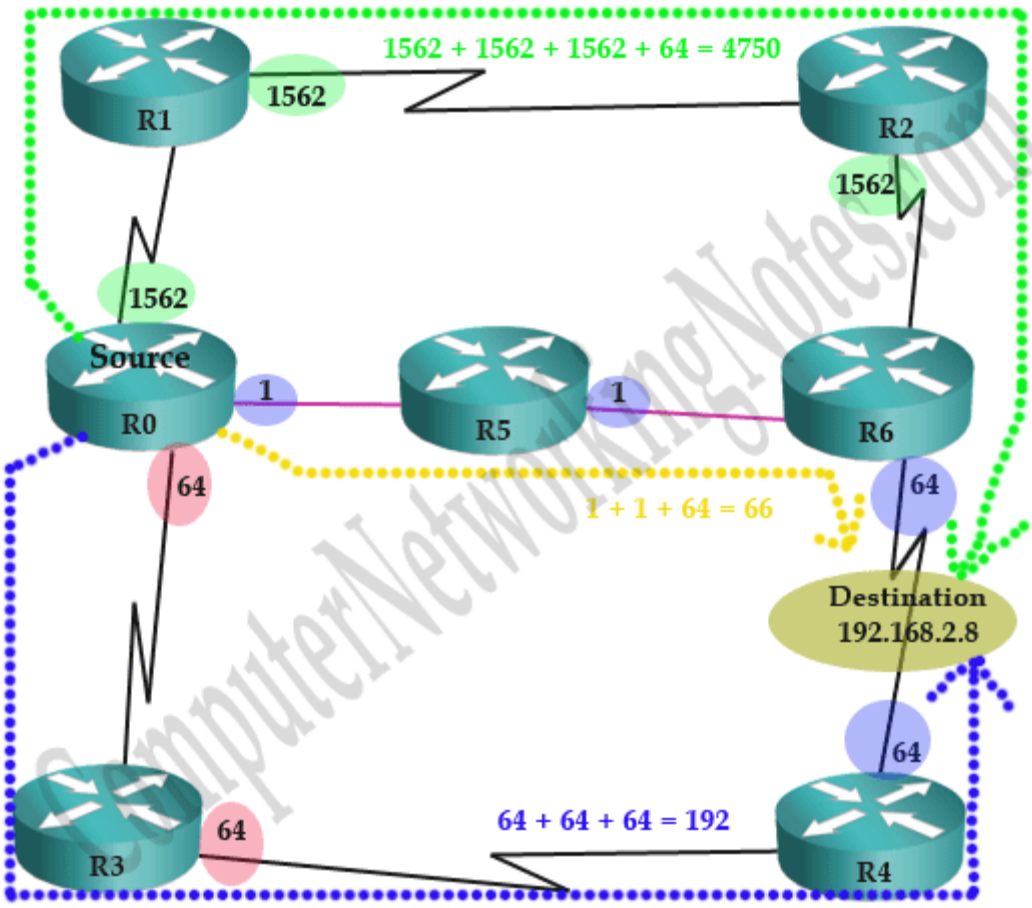
R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) = 192

Via Route R0 – R1 – R2 – R6

Ro's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 cost (1562) + R2's Serial 0/0/0 cost (1562) + R6's Serial 0/0/1 cost (64) = 4750

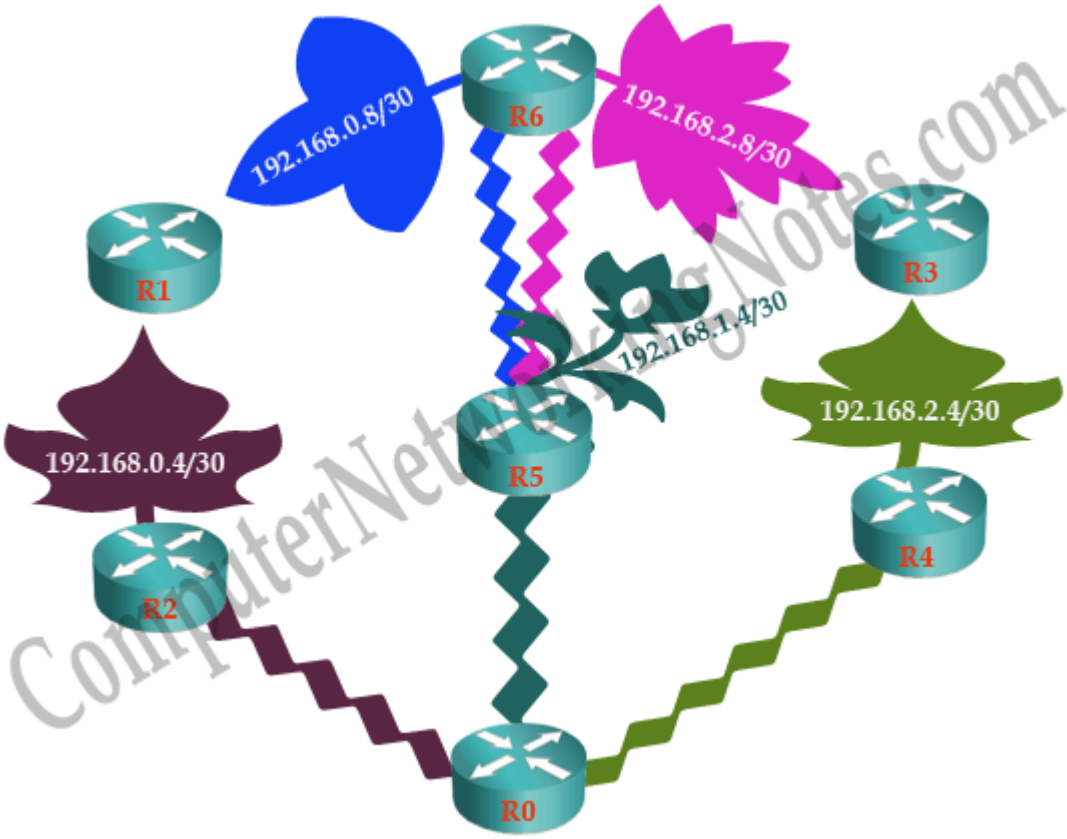
Via Route R0 – R5 – R6

R0's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/1 cost (64) = 66



Route R0 – R5 – R6 has the lowest cost value.

After selecting best route for each destination OSPF network look likes following figure.



OSPF Route cost Manipulation

We can manipulate OSPF route cost in two ways.

1. By changing bandwidth of interface
2. By changing reference bandwidth value

By changing bandwidth of interface

Sub interface mode command ***Bandwidth*** is used to set the bandwidth of supported interface.

If bandwidth is set through this command, OSPF will use it. If bandwidth is not set, it will use interface's default bandwidth.

When we enable an interface, router automatically assign a bandwidth value to it based on its type. For example serial interface has a default bandwidth value of 1544k. Until we change this value with bandwidth command, it will be used where it is required.

Let me clear one more thing about bandwidth. Changing default bandwidth with bandwidth command does not change actual bandwidth of interface. Neither default bandwidth nor bandwidth set by bandwidth command has anything to do with actual layer one link bandwidth.

Then what purpose does this command solve?

This command is only used to influence the routing protocol which uses bandwidth in route selection process such as OSPF and EIGRP.

We have already seen an example of this method in our example. We changed default bandwidth (1544Kbps) to custom (64kbps) bandwidth on R0's serial 0/0/0, R1's serial 0/0/1 and R2's serial 0/0/0. Due to this change R0 took another router for 192.168.0.4 network.

Let's understand this in more detail.

Current cost for destination 192.168.0.4 from R0

Via Route R0 – R1

R0's Serial 0/0/0 cost (1562) + R1's Serial 0/0/1 cost (1562) = 3124 (Cumulative cost)

Via Route R0 – R5 – R6 – R2

Ro's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/0 cost (64) +R2's Serial 0/0/1 cost (64) = 130 (Cumulative cost)

Via Route R0 – R3 – R4 – R6 – R2

R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) + R6's Serial 0/0/0 cost (64) + R2's Serial 0/0/1 cost (64) = 320 (Cumulative cost)

Among these routes, Route R0 – R5 – R6 – R2 has the lowest cost so it was picked for routing table.

Well … Which route would have selected, if we had used default bandwidth?

Cost for destination 192.168.0.4 from R0 with default bandwidth.

Via Route R0 – R1

R0's Serial 0/0/0 cost (64) + R1's Serial 0/0/1 cost (64) = 128 (Cumulative cost)

Via Route R0 – R5 – R6 – R2

Ro's FastEthernet 0/1 cost (1) + R5's FastEthernet 0/0 cost (1) + R6's Serial 0/0/0 cost (64) +R2's Serial 0/0/1 cost (64) = 130 (Cumulative cost)

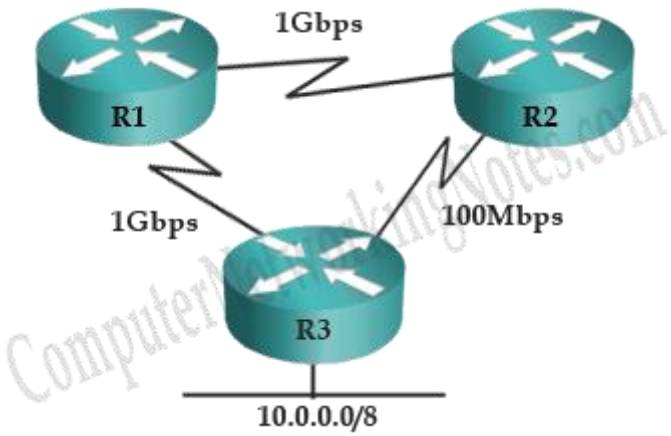
Via Route R0 – R3 – R4 – R6 – R2

R0's Serial 0/0/1 cost (64) + R3's Serial 0/0/0 cost (64) + R4's Serial 0/0/1 cost (64) + R6's Serial 0/0/0 cost (64) + R2's Serial 0/0/1 cost (64) = 320 (Cumulative cost)

Among these routes, Route R0 – R1 has the lowest cost value so it would be selected for routing table. Thus by changing interface bandwidth we actually influenced route selection process.

By changing reference bandwidth value

As I mention earlier, by default OSPF uses 100Mbps bandwidth as a reference bandwidth. Changing this value would also change the cost of route. If we use 1000Mbps as a reference bandwidth, cost of 100Mbps link would become 10. This sounds great, especially if we have higher bandwidth links in our network. For example have a look on following figure.



Which route will R2 take to get the network of 10.0.0.0/8?

Route R2 – R3

In this route we have two exit points. Both points have default 1oo Mbps speed.

R2's FastEthernet cost (100000000/100000000) = 1

R3's FastEthernet cost (100000000/100000000) = 1

Cost of this route 1 + 1 = 2

Route R2 – R1 – R3

In this route we have three exit points. Two exit points (R2 and R1) have 1 Gbps link.

R2's FastEthernet cost (100000000/1000000000) = .1 (Anything below 1 would be considered as 1)

R3's FastEthernet (100000000/1000000000) = .1 (Anything below 1 would be considered as 1)

R3's FastEthernet cost (100000000/100000000) = 1

Cost of this route 1 + 1 + 1 = 3

With default reference bandwidth R2 will choose Route R2 – R3, which is not good.

We can adjust reference bandwidth with **auto-cost reference-bandwidth *ref-band*** command.

We need to adjust reference bandwidth on all routers of network. Mismatched reference bandwidth can cause routers to run the SPF algorithm continually, which could create a serious performance issue.

Reference bandwidth is assigned in Mbps. Valid range is 1 to 4294967. Default reference bandwidth is 100Mbps.

Sadly packet tracer does not include this command. For the practice of this command please use other simulator software which support this command or use real router.

Let's change reference bandwidth to 1000Mbps on all three routers using following commands

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router (config)#router ospf 1
Router (config-router)#auto-cost reference-bandwidth 1000
% OSPF: Reference bandwidth is changed.
      Please ensure reference bandwidth is consistent across all routers.
Router (config-router)#exit
Router #
```

Route cost with new reference bandwidth

{module in_art_slot_10}

Route R2 – R3

R2's FastEthernet cost (1000000000/100000000) = 10

R3's FastEthernet cost (1000000000/100000000) = 10

Cost of this route 10 + 10 = 20

Route R2 – R1 – R3

R2's FastEthernet cost (1000000000/1000000000) = 1

R3's FastEthernet (1000000000/1000000000) = 1

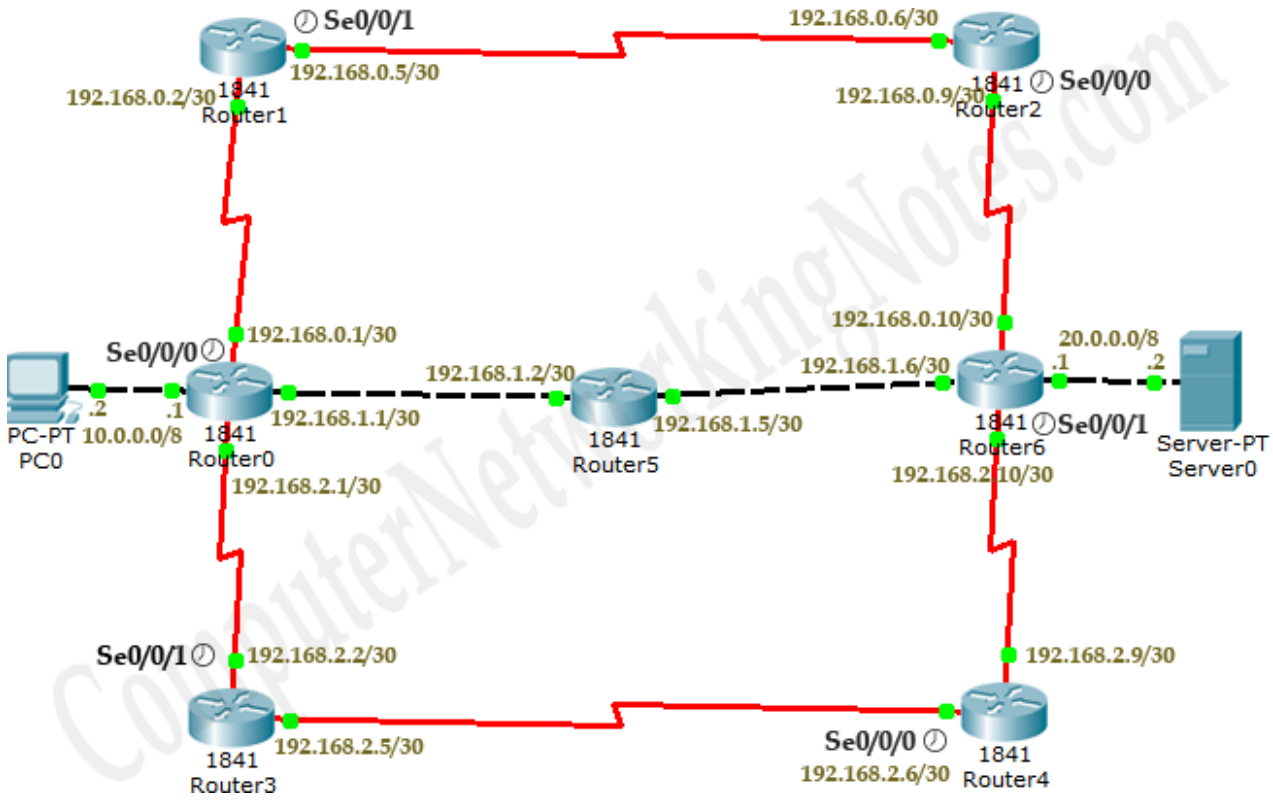
R3's FastEthernet cost (1000000000/100000000) = 10

Cost of this route 1 + 1 + 10 = 12

In this case Route R2-R1-R3 will be selected, which is the shortest route for destination.

Configuration of OSPF Routing Protocol

Create a topology as illustrate in following figure.



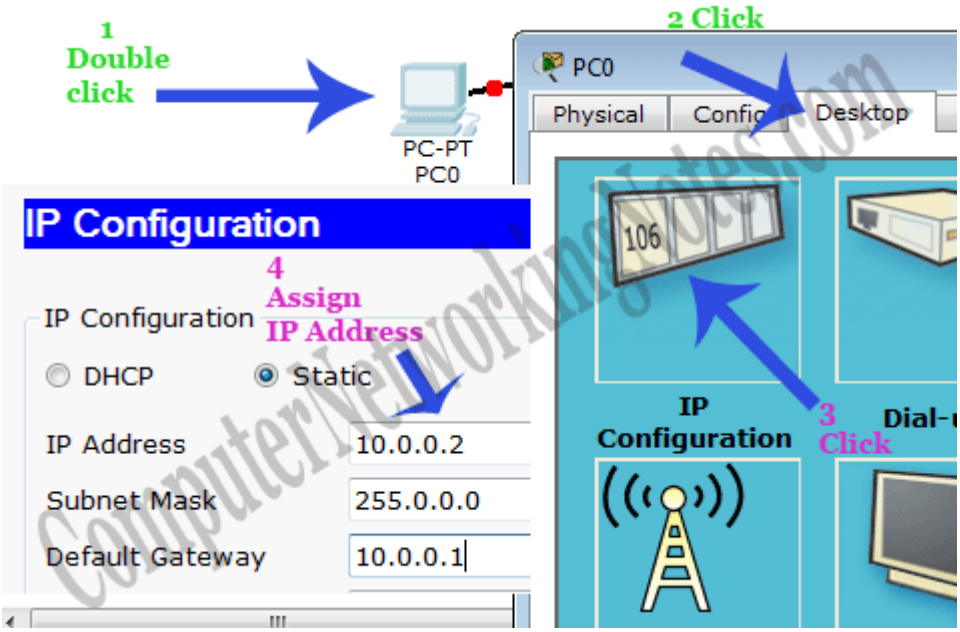
Initial IP Configuration

Device	Interface	IP Configuration
PC0	Fa0/0	10.0.0.2/8
Router0	Fa0/0	10.0.0.1/8
Router0	Fa0/1	192.168.1.1/30
Router5	Fa0/1	192.168.1.2/30
Router5	Fa0/0	192.168.1.5/30
Router6	Fa0/0	192.168.1.6/30
Router6	Fa0/1	20.0.0.1/8
Server0	Fa0/0	20.0.0.2/8
Router0	Serial 0/0/0 (DCE)	192.168.0.1/30
Router1	Serial 0/0/0	192.168.0.2/30

Device	Interface	IP Configuration
Router1	Serial 0/0/1 (DCE)	192.168.0.5/30
Router2	Serial0/0/1	192.168.0.6/30
Router2	Serial 0/0/0 (DCE)	192.168.0.9/30
Router6	Serial 0/0/0	192.168.0.10/30
Router0	Serial 0/0/1	192.168.2.1/30
Router3	Serial 0/0/1 (DCE)	192.168.2.2/30
Router3	Serial 0/0/0	192.168.2.5/30
Router4	Serial 0/0/0 (DCE)	192.68.2.6/30
Router4	Serial 0/0/1	192.168.2.9/30
Router6	Serial0/0/1 (DCE)	192.168.2.10/30

Assign IP address to PC

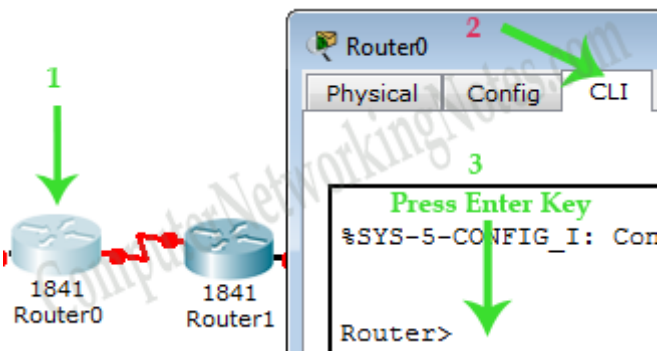
Double click **PC0** and click **Desktop** menu item and click **IP Configuration**. Assign IP address *10.0.0.2/8* to **PC0**.



Repeat same process for Server0 and assign IP address 20.0.0.2/8.

Assign IP address to interfaces of routers

Double click **Router0** and click **CLI** and press Enter key to access the command prompt of **Router0**.



Four interfaces FastEthernet0/0, FastEthernet0/1, Serial 0/0/0 and Serial0/0/1 of Router0 are used in this topology. By default interfaces on router are remain administratively down during the start up.

We need to configure IP address and other parameters on interfaces before we could actually use them for routing. Interface mode is used to assign the IP address and other parameters. Interface mode can be accessed from global configuration mode. Following commands are used to access the global configuration mode.

```
Router>enable
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

From global configuration mode we can enter in interface mode. From there we can configure the interface. Following commands will assign IP address on FastEthernet0/0 and FastEthernet0/1.

```
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface fastEthernet 0/1
Router(config-if)#ip address 192.168.1.1 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

interface fastEthernet 0/0 command is used to enter in interface mode.

ip address 10.0.0.1 255.0.0.0 command would assign IP address to interface.

no shutdown command would bring the interface up.

exit command is used to return in global configuration mode.

Serial interface needs two additional parameters **clock rate** and **bandwidth**. Every serial cable has two ends DTE and DCE. These parameters are always configured at DCE end.

We can use **show controllers *interface*** command from privilege mode to check the cable's end.

```
Router#show controllers serial 0/0/0
Interface Serial0/0/0
Hardware is PowerQUICC MPC860
DCE V.35, clock rate 2000000
[Output omitted]
```

Fourth line of output confirms that DCE end of serial cable is attached. If you see DTE here instead of DCE skip these parameters.

Now we have necessary information let's assign IP address to serial interfaces.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.0.1  255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth 64
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.2.1  255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
```

Router#configure terminal Command is used to enter in global configuration mode.

Router(config)#interface serial 0/0/0 Command is used to enter in interface mode.

Router(config-if)#ip address 192.168.0.1 255.255.255.252 Command assigns IP address to interface. For serial link we usually use IP address from /30 subnet.

Router(config-if)#clock rate 64000 In real life environment this parameter controls the data flow between serial links and need to be set at service provider's end. In lab environment we need not to worry about this value. We can use any valid clock rate here.

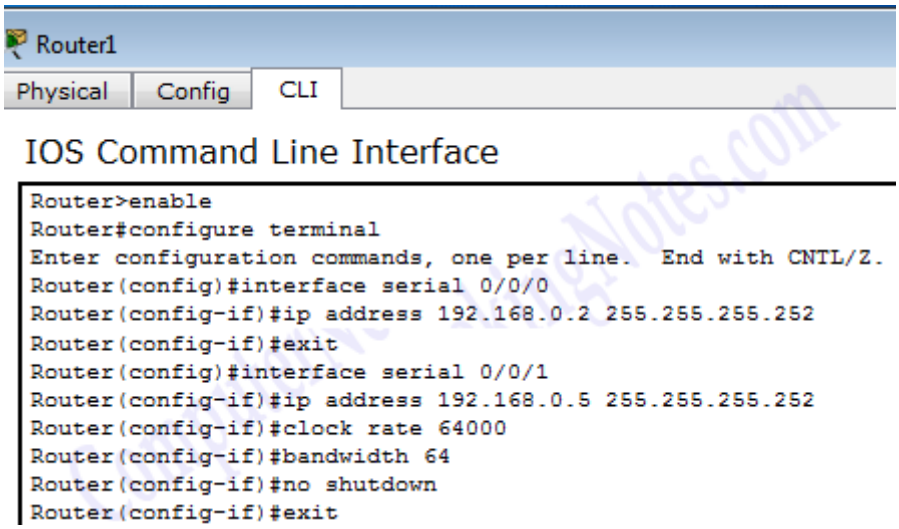
Router(config-if)#bandwidth 64 Bandwidth works as an influencer. It is used to influence the metric calculation of OSPF or any other routing protocol which uses bandwidth parameter in route selection process. Serial interface has default bandwidth of 1544Kbps. To explain, how bandwidth influence route selection process we will configure (64Kbps) bandwidth on three serial DCE interfaces of our network; R0's Se0/0/0, R1's Se0/0/1 and R2's Se0/0/0.

Router(config-if)#no shutdown Command brings interface up.

Router(config-if)#exit Command is used to return in global configuration mode.

We will use same commands to assign IP addresses on interfaces of remaining routers.

Router1

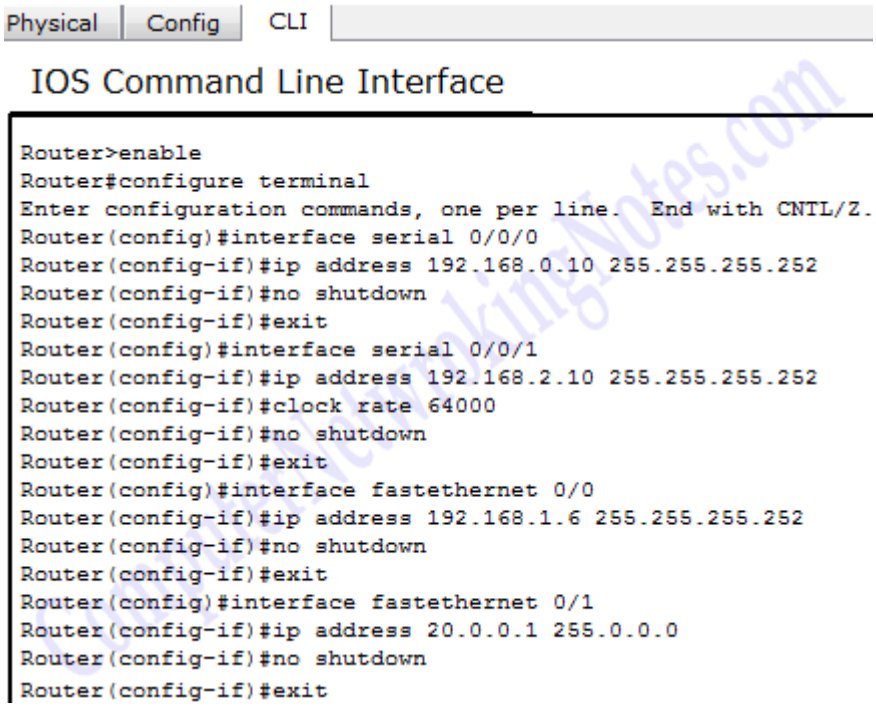


Router2

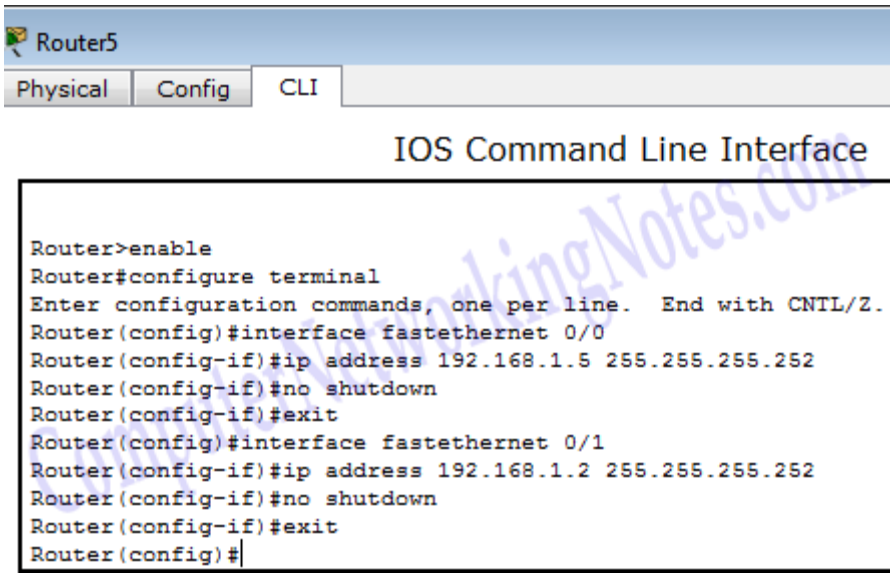
```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.0.9 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#bandwidth 64
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config-if)#
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.0.6 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

As I mention earlier, serial interface has a default bandwidth of 1544Kbps. If we don't assign any custom bandwidth, router would use default bandwidth. To see this feature in action we will not assign bandwidth on remaining routers.

Router6



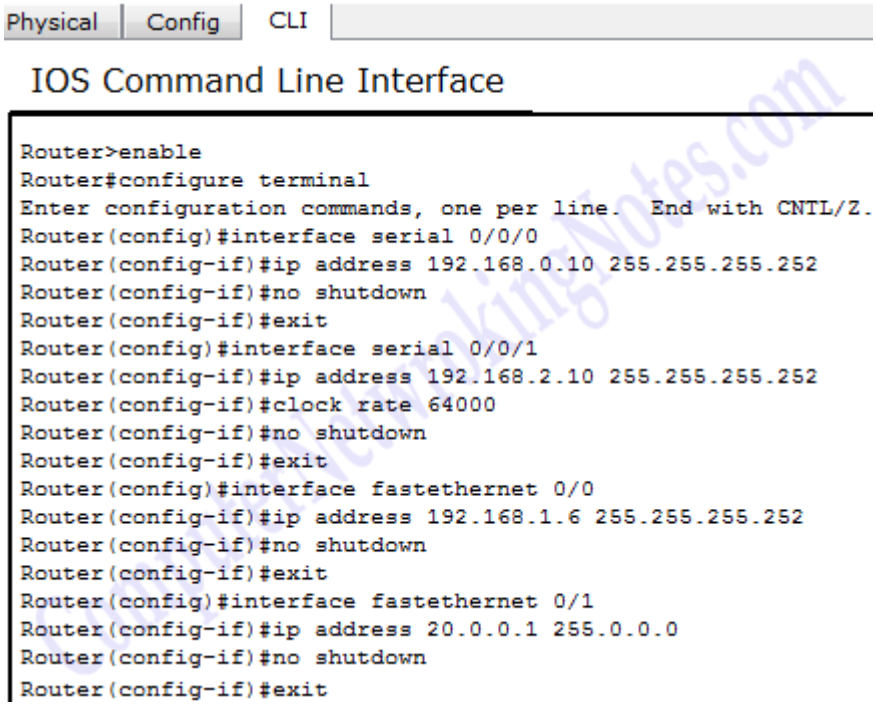
Router5



Router3

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface serial 0/0/0
Router(config-if)#ip address 192.168.2.5 255.255.255.252
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface serial 0/0/1
Router(config-if)#ip address 192.168.2.2 255.255.255.252
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
```

Router4



Great job we have finished our half journey. Now routers have information about the networks that they have on their own interfaces. Routers will not exchange this information between them on their own. We need to implement OSPF routing protocol that will insist them to share this information.

To be on same track I have uploaded my practice topology. Use this if you want to skip above IP configuration part.

Download OSPF Practice Topology with IP configuration

Configure OSPF routing protocol

Enabling OSPF is a two steps process:-

- Enable OSPF routing protocol from global configuration mode.
- Tell OSPF which interfaces we want to include.

For these steps following commands are used respectively.

```
Router(config)# router ospf process_ID
Router(config-router)# network IP_network_# [wild card mask] Area Number area number
```

Router(config)# router ospf process ID

This command will enable OSPF routing protocol in router. Process ID is a positive integer. We can use any number from 1 to 65,535. Process ID is locally significant. We can run multiple OSPF process on same router. Process ID is used to differentiate between them. Process ID need not to match on all routers.

Router(config-router)# network IP_network_# [wildcard_mask] area [area number]

Network command allows us to specify the interfaces which we want to include in OSPF process. This command accepts three arguments network number, wildcard mask and area number.

Network number

Network number is network ID. We can use any particular host IP address or network IP address. For example we can use 192.168.1.1 (host IP address) or we can use 192.168.1.0 (Network IP address). While targeting a specific interface usually we use host IP address (configured on that interface).

While targeting multiple interfaces, we use network IP address. So any interface that belongs to specified network ID will be selected.

Wildcard mask

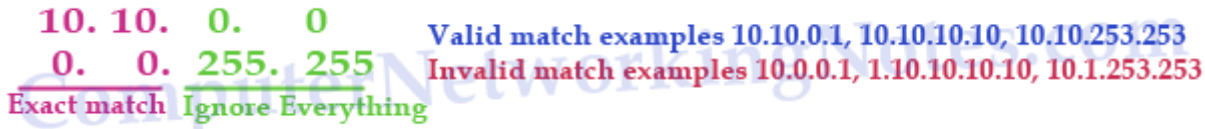
Wildcard mask are used with network ID to filter the interfaces. Wildcard mask is different from subnet mask. Subnet mask is used to separate the network portion and host portion in IP address. While wildcard mask is used to match corresponding octet in network portion. Wildcard mask tells OSPF the part of network address that must be matched. Wildcard masks are explained with examples in access list tutorials of this category.

Key points

0 (*Decimal – octet format*) Wildcard mask indicates that corresponding octet in network address must be matched exactly.

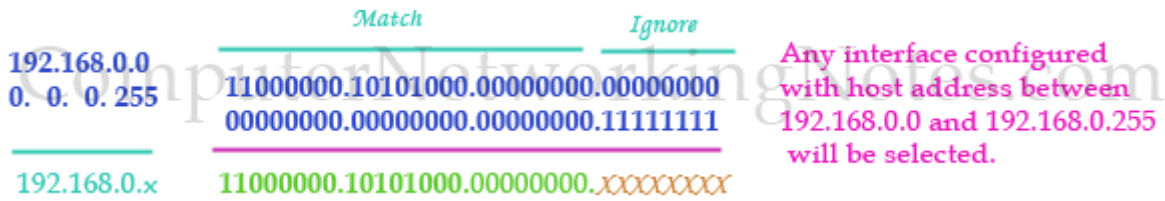
255 (*Decimal – octet format*) Wildcard mask indicates that we don't care about corresponding octet in network address.

For example

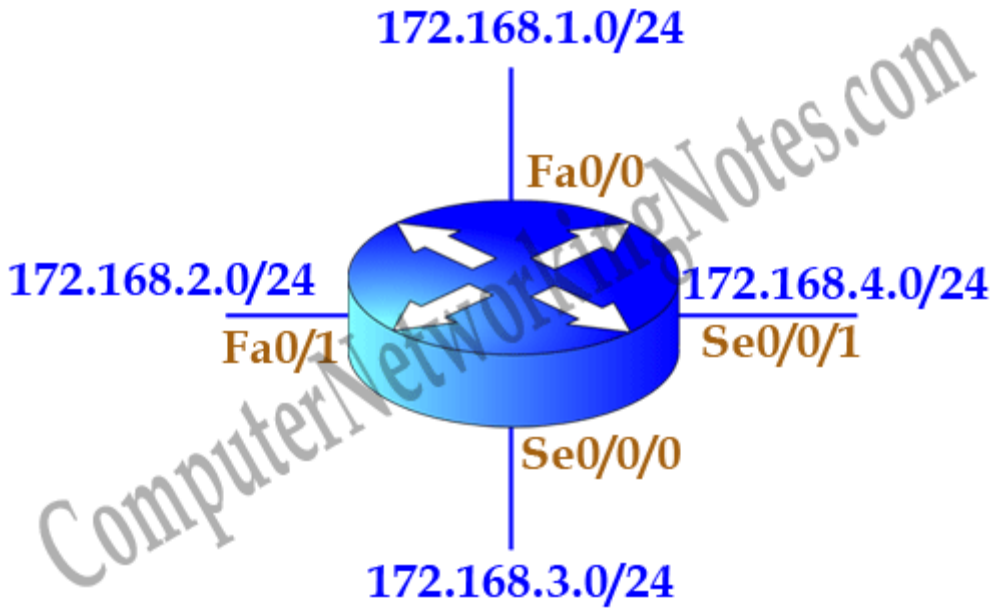


0 (*Binary – bit format*) Wildcard mask indicates that corresponding bit in network address must be matched exactly.

255 (*Binary – bit format*) Wildcard mask indicates that we don't care about corresponding bit in network address.



OSPF is a classless protocol. With wildcard we can also filter Subnetted networks. In classes implementation usually we use Subnetted networks. For example consider following figure



We have four networks 172.168.1.0/24, 172.168.2.0/24, 172.168.3.0/24 and 172.168.4.0/24 subnetted from single class B network 172.168.0.0/16. Classful configuration does not understand the concept of subnetting. In classful configuration all these networks belong to a single network. Classful configuration works only with in default boundary of mask. Default boundary of this address is 16 bits. So a classful routing protocol will match only first 16 bits (172.168.x.y) of network address. A classful routing protocol such as RIP cannot distinguish between different Subnetted networks.

A classless routing protocol such as OSPF goes beyond the default boundary of mask and work well with Subnetted networks. With wildcard mask we can easily filter Subnetted networks.

With wildcard we are no longer limited with default boundaries. We can match Subnetted networks as well as default networks.

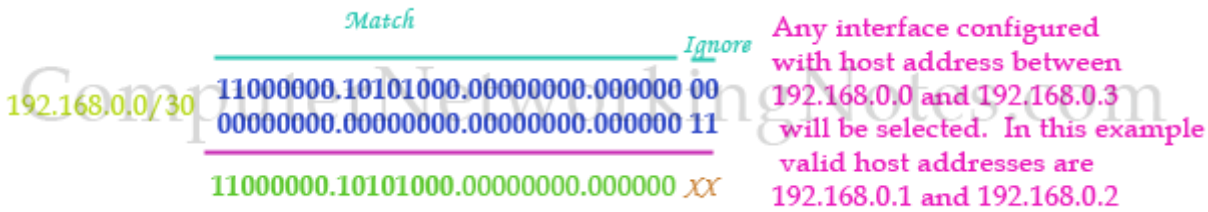
For example we want to exclude serial interfaces in above configuration. We can use a wildcard mask of 0.0.0.255 to match the subnet mask of /24.

```
Router(config-router)# network 172.168.1.0 0.0.0.255
Router(config-router)# network 172.168.2.0 0.0.0.255
```

Above commands will ask router to match /24 bits of address instead of default /16 bits. Now router will look for 172.168.1.x and 172.168.2.x network. Our serial interfaces have 172.168.3.0/24 and 172.168.4.0/24 networks which do not fall in these search criteria.

Let's take one more example, if we use following network command, which interfaces would be selected.

```
Router(config-router)# network 192.168.0.0 0.0.0.3
```



In this case valid host IP addresses are 192.168.0.1 and 192.168.0.2. So any interface that has these IP address would be selected. /30 network is usually used for serial link connection which need only two valid host IP addresses; one for each end.

If you are unfamiliar with wildcard mask, I suggest you to check our tutorials on access lists configuration in this category. In those tutorials wildcard masks are explained in detail with examples.

For this tutorial let's move on third argument. Third argument which network command accept is area number. This parameter say router to put matched interface in specified area. OSPF areas are explained in second part this article.

Now we know the essential commands for configuration. Let's implement them in our network.

OSPF configuration

Router0

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 10
Router(config-router)#network 10.0.0.0 0.255.255.255 area 0
Router(config-router)#network 192.168.0.0 0.0.0.3 area 0
Router(config-router)#network 192.168.1.0 0.0.0.3 area 0
Router(config-router)#network 192.168.2.0 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

Router1

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 10
Router(config-router)#network 192.168.0.0 0.0.0.3 area 0
Router(config-router)#network 192.168.0.4 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

Router2

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 20
Router(config-router)#network 192.168.0.4 0.0.0.3 area 0
Router(config-router)#network 192.168.0.8 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

Router6

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 60
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#network 192.168.0.8 0.0.0.3 area 0
Router(config-router)#network 192.168.2.8 0.0.0.3 area 0
Router(config-router)#network 192.168.1.4 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

Router5

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 50
Router(config-router)#network 192.168.1.0 0.0.0.3 area 0
Router(config-router)#network 192.168.1.4 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

Router4

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 40
Router(config-router)#network 192.168.2.8 0.0.0.3 area 0
Router(config-router)#network 192.168.2.4 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```

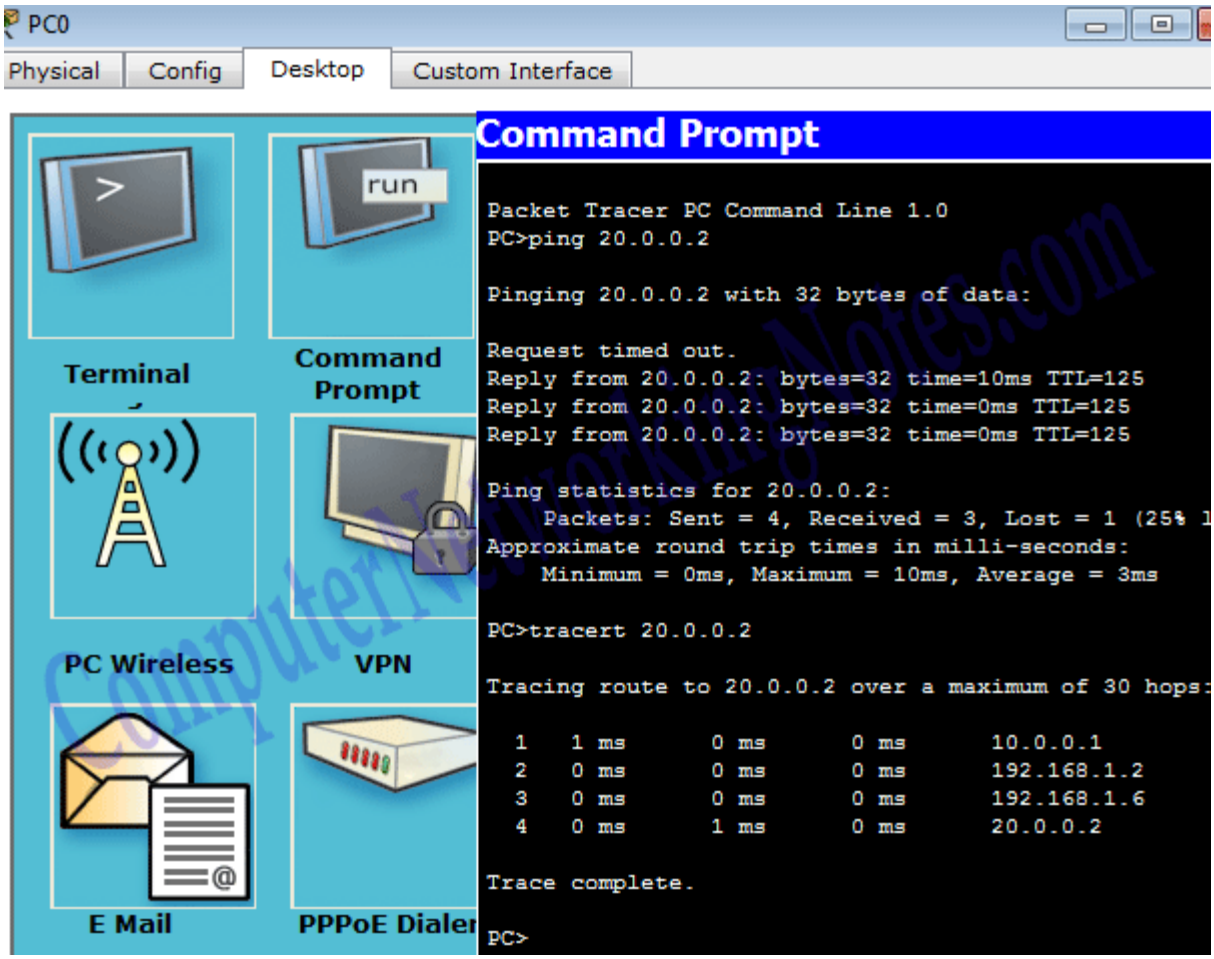
Router3

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 30
Router(config-router)#network 192.168.2.0 0.0.0.3 area 0
Router(config-router)#network 192.168.2.4 0.0.0.3 area 0
Router(config-router)#exit
Router(config)#
```


That's it. Our network is ready to take the advantage of OSPF routing. To verify the setup we will use **ping** command. ping command is used to test the connectivity between two devices.

We have two routes between source and destination. tracert command is used to know the route which is used to get the destination.

Access the command prompt of PC1 and use ping command to test the connectivity from Server0. After that use **tracert** command to print the taken path.



Great! We have successfully implemented OSFP routing in our network.

If you did not get the same out as explained in this tutorial, use this configured topology to cross check your topology and find out the reason. [Download OSPF Practice Topology with OSPF configuration]

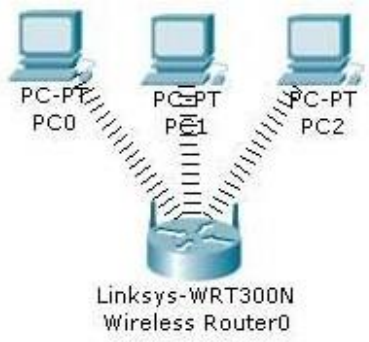
Summary

Command	Description
outer(config)#router opsf 10	Enable OSPF routing protocol under process ID 10.
Router(config-router)#network 10.10.0.0 0.0.255.255 area 0	Enable OSPF with area 0 on matching interface.
Router(config)#interface loopback 0	Create a Loopback interface and move in sub interface co
Router(config-if)#ip address 192.168.250.250 255.255.255.0	Assign IP address to loopback interface.
Router(config-router)#router-id 1.1.1.1	Set 1.1.1.1 as router ID
Router(config)#interface serial 0/0	Inter in sub interface configuration mode
Router(config-if)#ip ospf priority 100	Used to influence DR/BDR selection process. Valid range i ineligible for DR/BDR while 255 makes router guaranteed means higher chance of becoming DR/BDR.
Router(config-if)#bandwidth 256	Used to influence route metric cost. Cost is the inverse of has lower cost. Bandwidth is defined in Kbps. 256 means
Router(config-if)#ip ospf hello-interval timer 15	Set hello interval timer to 15 seconds. Hello timer must be become neighbors.
Router(config-if)#ip ospf dead-interval 60	Set dead interval timer to 60 seconds. Dead interval timer in order to become neighbor
Router#show ip route	Display all routes from routing table
Router#show ip route ospf	Display all routers learned through OSPF from routing tab
Router#show ip ospf	Display basic information about OSPF
Router#show ip ospf interface	Display information about all OSPF active interfaces
Router#show ip ospf interface serial 0/0/0	Display OSPF information about serial 0/0/0 interface
Router#show ip ospf neighbor List all	OSPF neighbors with basic info
Router#show ip ospf neighbor detail	List OSPF neighbors with detail info
Router#show ip ospf database	Display data for OSPF database
Router#clear ip route *	Clear all routes from routing table.
Router#clear ip route 10.0.0.0/8	Clear particular route from routing table
Router#clear ip ospf counters	Clear OSPF counters
Router#debug ip ospf events	Display all ospf events
Router#debug ip ospf packets	Display exchanged OSPF packets
Router#debug ip ospf adjacency	Display DR/BDR election process state

How to Configure Wireless Network

This assignment explains how to configure wireless network in packet tracer step by step including how to enable static routing in Linksys router.

We have covered all CCNA exam topics in our wireless networking section. As its new topic in CCNA exam so you have very little chance to get simulated base question on wireless networking. Still we recommended you to get familiar with some basic wireless configuration.



In this topology we have three pc connected with Linksys Wireless routers.

- DHCP is configured and enabled on Wireless router
- IP pool for DHCP is 192.168.0.100 to 192.168.0.150
- PC are configured to receive IP from DHCP Server
- No security is configured
- Default SSID is configured to Default
- Topology is working on infrastructure mode
- Default user name and password is admin
- IP of wireless is set to 192.168.0.1

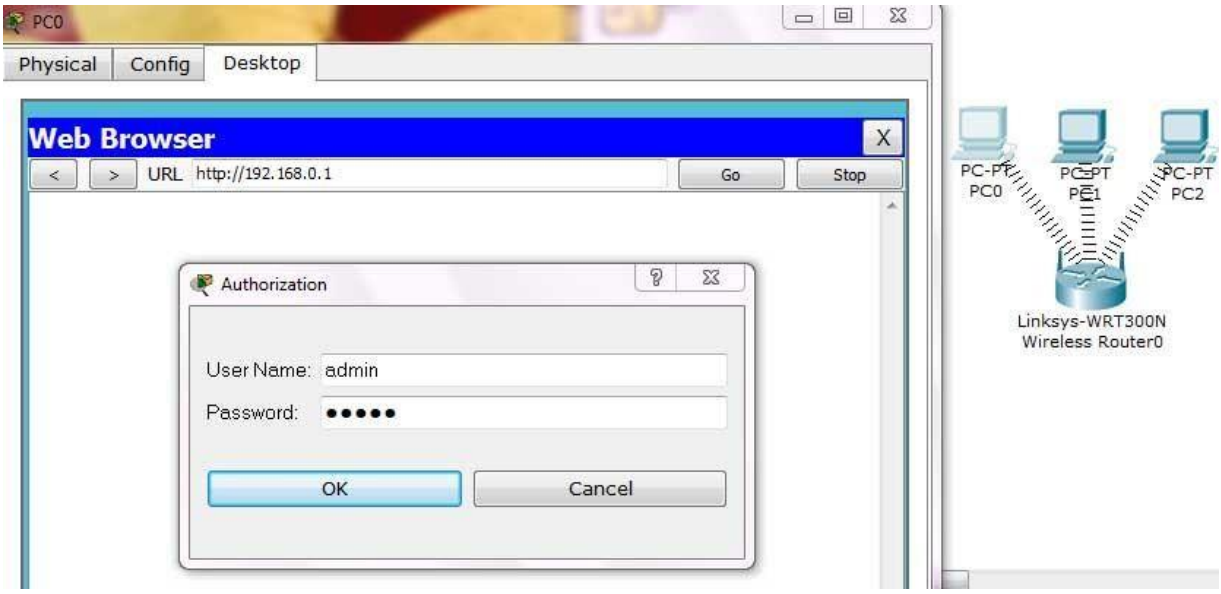
Now your task is to:-

- Configure Static IP on PC and Wireless Router
- Change SSID to MotherNetwork
- Change IP address of router to 10. 0.0.1 and 10. 0.0.2 of PC0 10. 0.0.3 of PC1 10.0.0. 4 of PC2
- Secure your network by configuring WAP key on Router
- Connect PC by using WAP key

To complete these tasks follow this step by step guide of how to configure wireless network

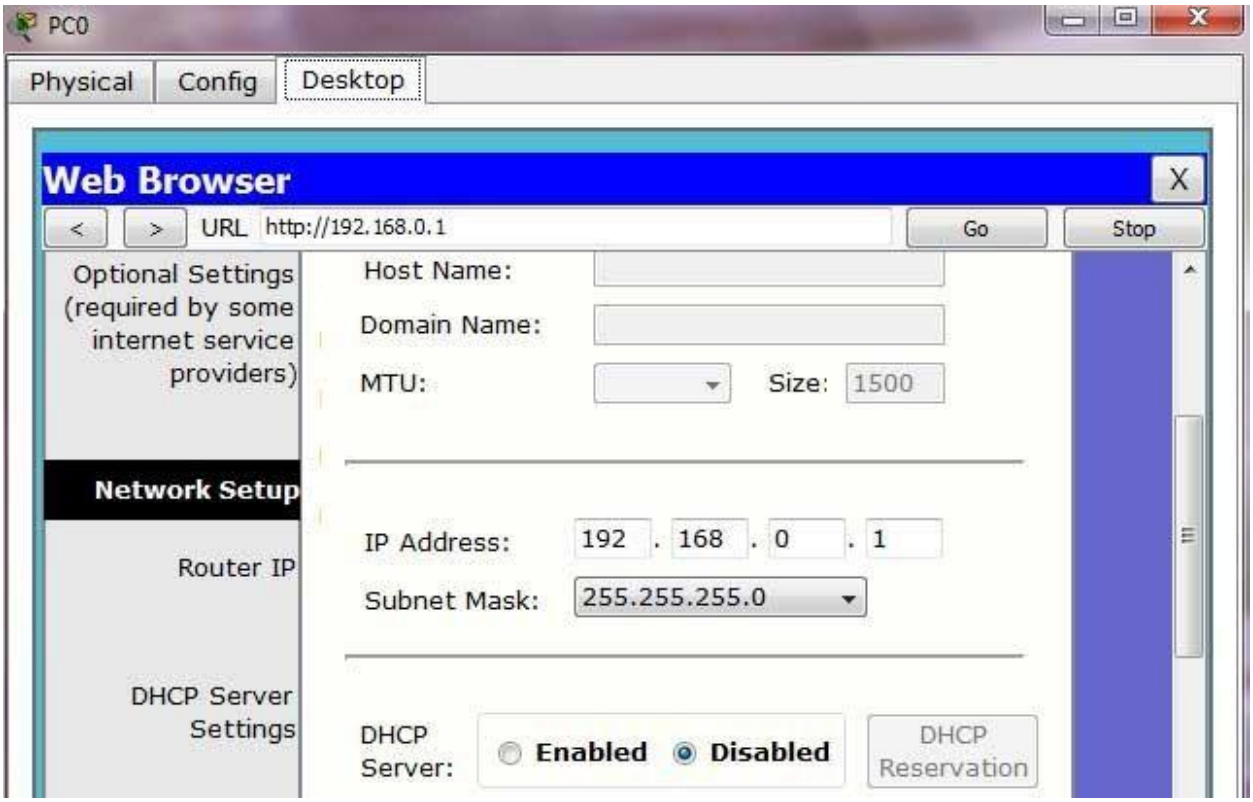
As given in question our network is running on 192.168.0.0 network and all PC's are DHCP clients and functioning properly. So we will first connect to Wireless router to off DHCP.

Double click on PC and select Web Browser. As given in question IP of Wireless router is 192.168.0.1 so give it in Web browser and press enter, now it will ask for authentication which is also given in question. Give user name admin and Password to admin



This will bring GUI mode of Wireless router. Scroll down screen to Network Step and Select Disable

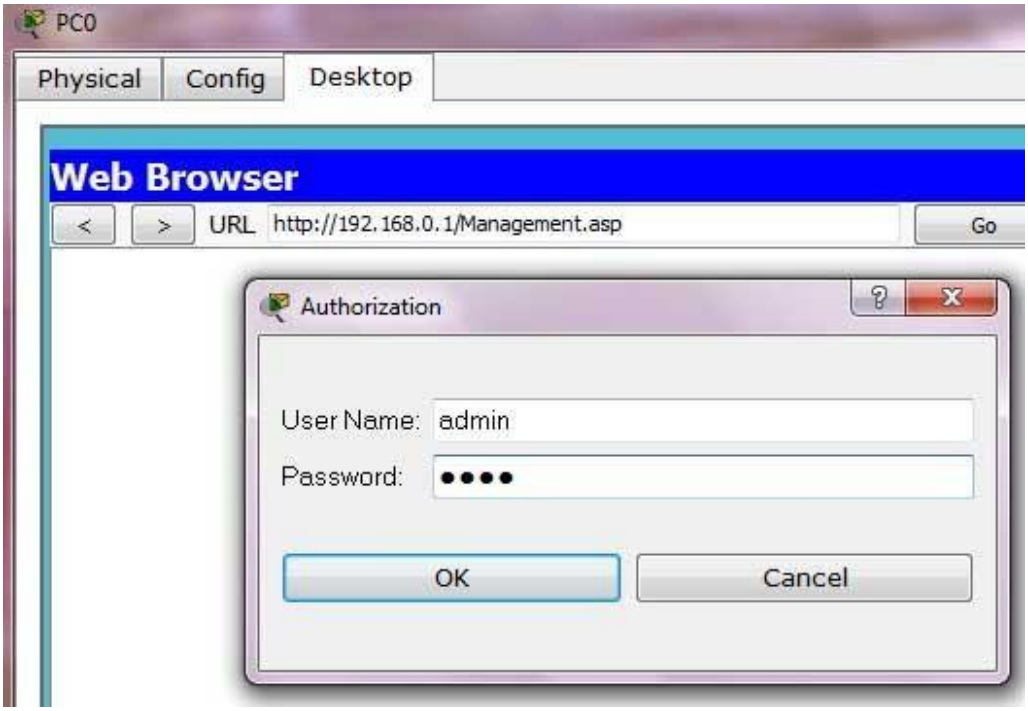
DHCP



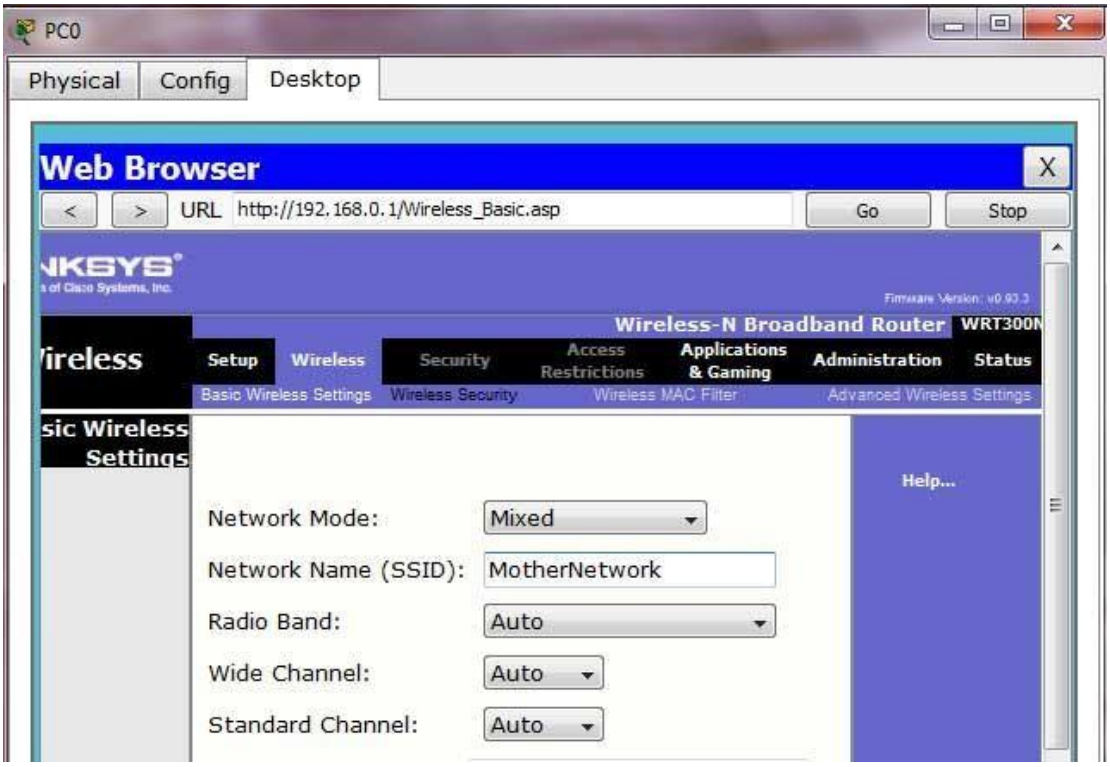
Go in end of page and click on Save setting this will save setting click on continue for further setting

Now select Administration from top Manu and change password to test and go in the end of page and Click on Save Setting

Click on continue for further setting. This time it will ask you to authenticate again give new password test this time



Now click on wireless tab and set default SSID to MotherNetwork



Now Select wireless security and change Security Mode to WEP

Set Key1 to 0123456789

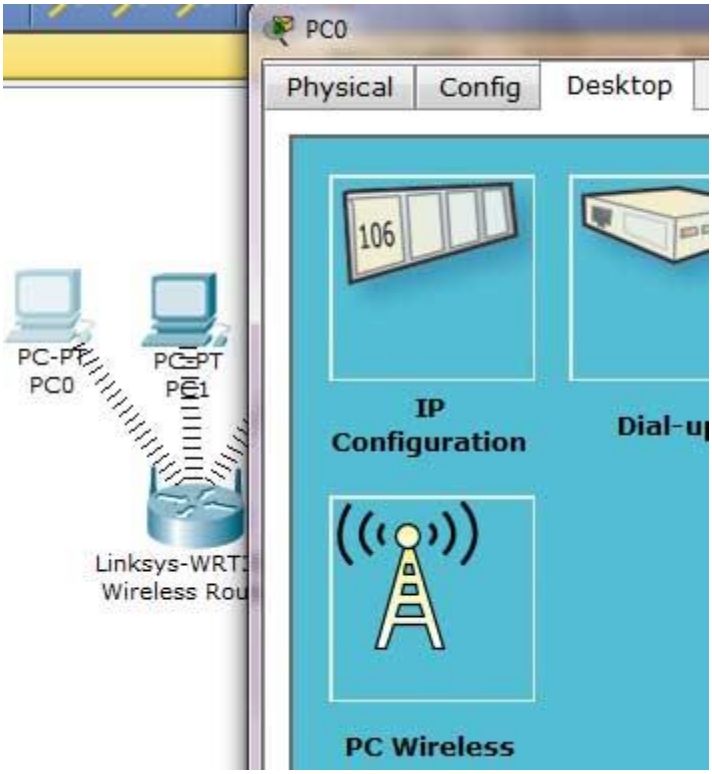
Again go in the end of page and Click on Save Setting

Now we have completed all given task on Wireless router. Now configure the static IP on all three PC's

Double click on pc select Desktop tab click on IP configuration select Static IP and set IP as given below

PC	IP	Subnet Mask	Default Gateway
PC0	192.168.0.2	255.255.255.0	192.168.0.1
PC1	192.168.0.3	255.255.255.0	192.168.0.1
PC2	192.168.0.4	255.255.255.0	192.168.0.1

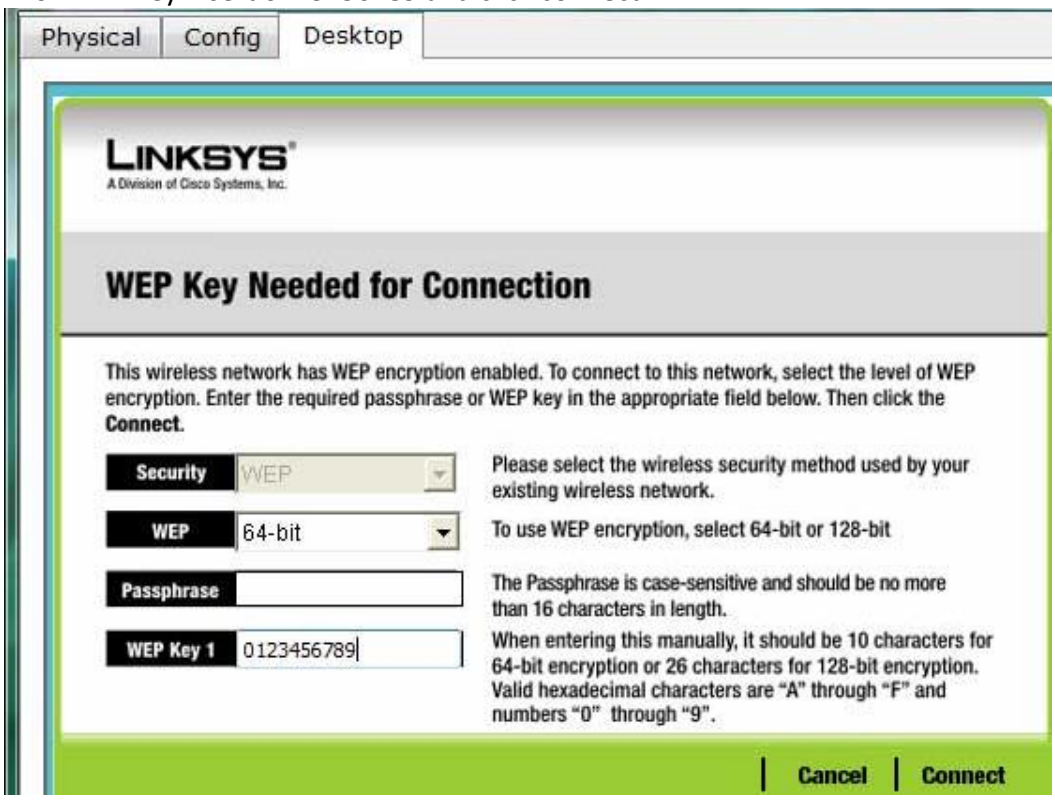
Now it's time to connect PC's from Wireless router. To do so click PC select Desktop click on PC Wireless



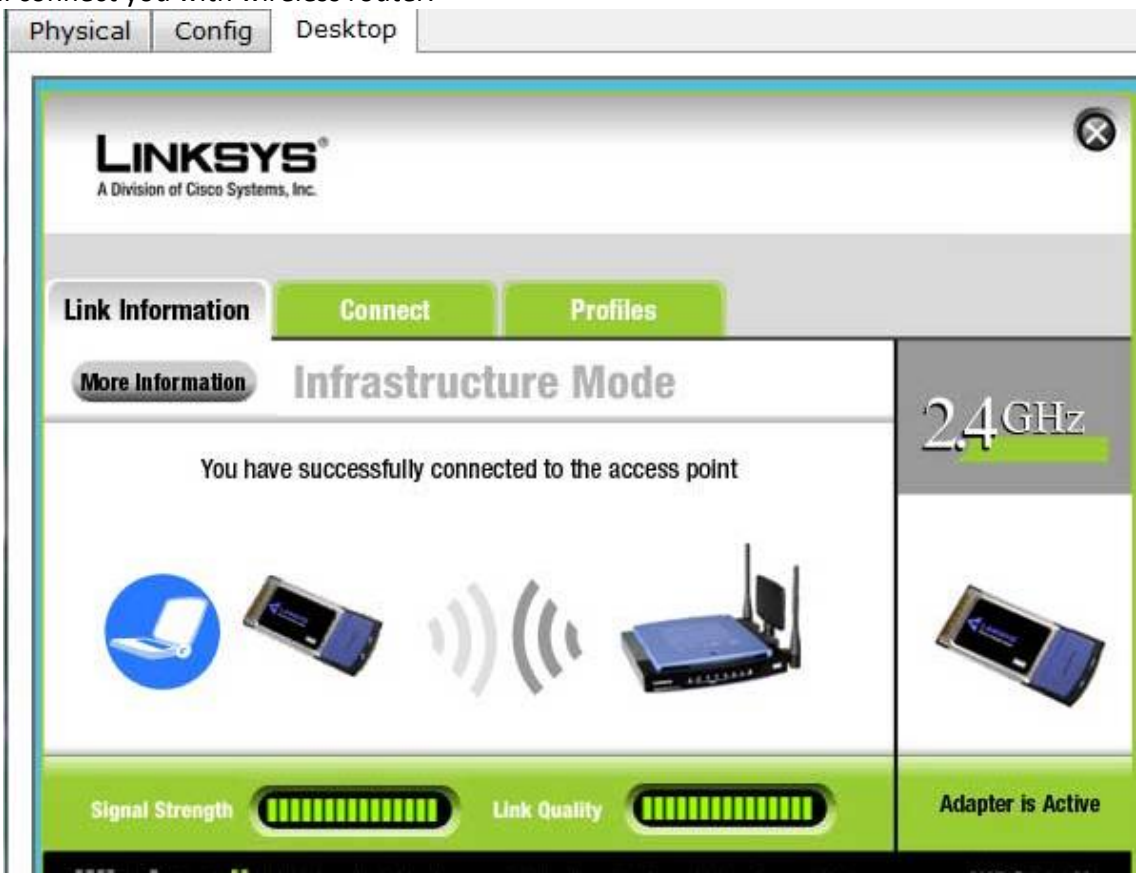
Click on connect tab and click on Refresh button

As you can see in image that Wireless device is accessing Mother Network on CH 6 and signal strength is 100%. In left side you can see that WEP security is configured in network. Click on connect button to connect Mother Network

It will ask for WAP key insert 0123456789 and click connect



It will connect you with wireless router.



As you can see in image below that system is connected. And PCI card is active.
Repeat same process on PC1 and PC2.

ASSIGNMENT NO : 3

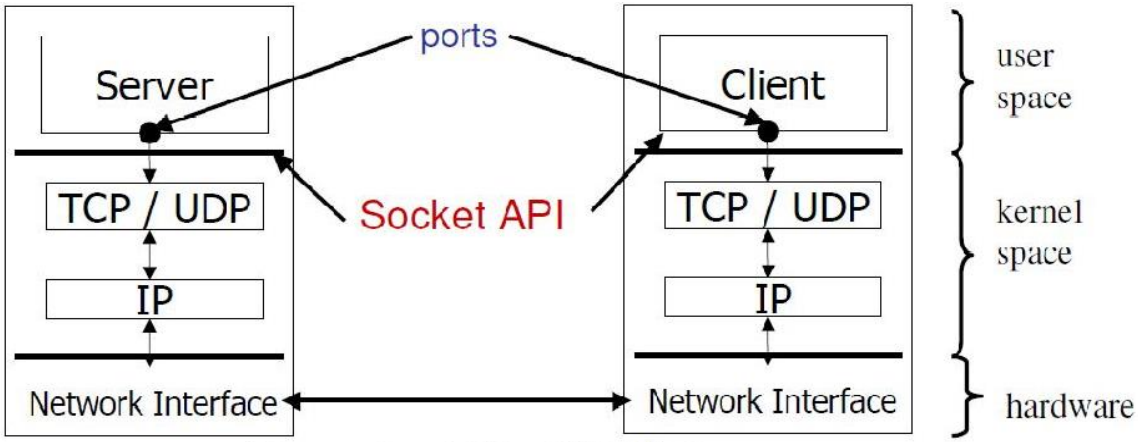
Socket Programming in C/C++ on Linux.

a) TCP Client, TCP Server

b) UDP Client, UDP Server

Theory:

- Sockets are used for interprocess communication.
- Most of the interprocess communication follow a Client-Server
- Model, where client and server are two separate processes in itself.
- Server and Client exchange messages over the network through a common Socket API



* Port numbers: 1,024 -- 65,535 (2^{16} - 1)

Server Examples

- Web server (port 80)
- FTP server (20, 21)
- Telnet server (23)
- Mail server (25)

Client Examples

- Examples of client programs

– Web browsers, ftp, telnet, ssh

How does a client find the server?

- The IP address in the server socket address identifies the host
- The (well-known) port in the server socket address identifies the service, and thus implicitly identifies the server process that performs that service.

Examples of well know ports

- Port 7: Echo server
- Port 23: Telnet server
- Port 25: Mail server
- Port 80: Web server

What is an API ?

API expands as Application Programming Interface.

A set of routines that an application uses to request and carry out lower-level services performed by a computer's operating system.

What is a socket?

- An interface between application and network which is used for communication between processes
- Once configured the application can
  - pass data to the socket for network transmission
  - receive data from the socket (transmitted through the network by some other host)
- To the kernel, a socket is an endpoint of communication.
- To an application, a socket is a file descriptor that lets the application read/write from/to the network.
- Clients and servers communicate with each by reading from and writing to socket descriptors.
- Remember: All Unix I/O devices, including networks, are modeled as files.

Two essential types of sockets SOCK_STREAM

- TCP
- connection-oriented
- reliable delivery
- in-order guaranteed
- bidirectional

SOCK_DGRAM

- UDP

- no notion of “connection” – app indicates dest. for each packet
- unreliable delivery
- no order guarantees
- can send or receive

Socket Primitives

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

socket()

The function *socket()* creates an endpoint for communication and returns a file descriptor for the socket. *socket()* takes three arguments:

- *domain*, which specifies the protocol family of the created socket. For example:
  - *AF_INET* for network protocol IPv4 or
  - *AF_INET6* for IPv6.
  - *AF_UNIX* for local socket (using a file).
- *type*, one of:
  - *SOCK_STREAM* (reliable stream-oriented service or Stream Sockets)
  - *SOCK_DGRAM* (datagram service or Datagram Sockets)
  - *SOCK_SEQPACKET* (reliable sequenced packet service), or
  - *SOCK_RAW* (raw protocols atop the network layer).
- *protocol* specifying the actual transport protocol to use. The most common are IPPROTO_TCP, IPPROTO_SCTP, IPPROTO_UDP, IPPROTO_DCCP. These protocols are specified in file *netinet/in.h*. The value *0* may be used to select a default protocol from the selected domain and type.

The function returns -1 if an error occurred. Otherwise, it returns an integer representing the newly assigned descriptor.

Prototype:

- int socket(int domain, int type, int protocol)

bind()

bind() assigns a socket to an address. When a socket is created using *socket()*, it is only given a protocol family, but not assigned an address. This association with an address must be performed with the bind() system call before the socket can accept connections to other hosts. *bind()* takes three arguments:

- *sockfd*, a descriptor representing the socket to perform the bind on.
- *my_addr*, a pointer to a *sockaddr* structure representing the address to bind to.
- *addrlen*, a *socklen_t* field specifying the size of the *sockaddr* structure.

Bind() returns 0 on success and -1 if an error occurs. Prototype:

- int bind(int sockfd, const struct sockaddr *my_addr, socklen_t addrlen);

listen()

After a socket has been associated with an address, *listen()* prepares it for incoming connections. However, this is only necessary for the stream-oriented (connection-oriented) data modes, i.e., for socket types (*SOCK_STREAM*, *SOCK_SEQPACKET*). listen() requires two arguments:

- *sockfd*, a valid socket descriptor.
- *backlog*, an integer representing the number of pending connections that can be queued up at any one time. The operating system usually places a cap on this value.

Once a connection is accepted, it is dequeued. On success, 0 is returned. If an error occurs, -1 is returned.

Prototype:

- int listen(int sockfd, int backlog);

accept()

When an application is listening for stream-oriented connections from other hosts, it is notified of such events (cf. select() function) and must initialize the connection using the *accept()* function. The accept() function creates a new socket for each connection and removes the connection from the listen queue. It takes the following arguments:

- *sockfd*, the descriptor of the listening socket that has the connection queued.
- *cliaddr*, a pointer to a sockaddr structure to receive the client's address information.
- *addrlen*, a pointer to a *socklen_t* location that specifies the size of the client address structure passed to accept(). When *accept()* returns, this location indicates how many bytes of the structure were actually used.

The accept() function returns the new socket descriptor for the accepted connection, or -1 if an error occurs. All further communication with the remote host now occurs via this new socket.

Datagram sockets do not require processing by accept() since the receiver may immediately respond to the request using the listening socket.

Prototype:

- int accept(int sockfd, struct sockaddr *cliaddr, socklen_t *addrlen)

connect()

The *connect()* system call *connects* a socket, identified by its file descriptor, to a remote host specified by that host's address in the argument list.

Certain types of sockets are *connectionless*, most commonly user datagram protocol sockets. For these sockets, connect takes on a special meaning: the default target for sending and receiving data gets set to the given address, allowing the use of functions such as send() and recv() on connectionless sockets.

connect() returns an integer representing the error code: 0 represents success, while -1 represents an error. Historically, in the BSD-derived systems, the state of a socket descriptor is undefined if the call to connect() fails (as it is specified in the Single Unix Specification), thus, portable applications should close the socket descriptor immediately and obtain a new descriptor with socket(), in the case the call to connect() fails.[3]

Prototype:

- int connect(int sockfd, const struct sockaddr *serv_addr, socklen_t addrlen)

gethostbyname() and gethostbyaddr()

The *gethostbyname()* and *gethostbyaddr()* functions are used to resolve host names and addresses in the domain name system or the local host's other resolver mechanisms (e.g., /etc/hosts lookup). They return a pointer to an object of type *struct hostent*, which describes an Internet Protocol host. The functions take the following arguments:

- *name* specifies the name of the host. For example: www.wikipedia.org
- *addr* specifies a pointer to a *struct in_addr* containing the address of the host.
- *len* specifies the length, in bytes, of *addr*.
- *type* specifies the address family type (e.g., AF_INET) of the host address.

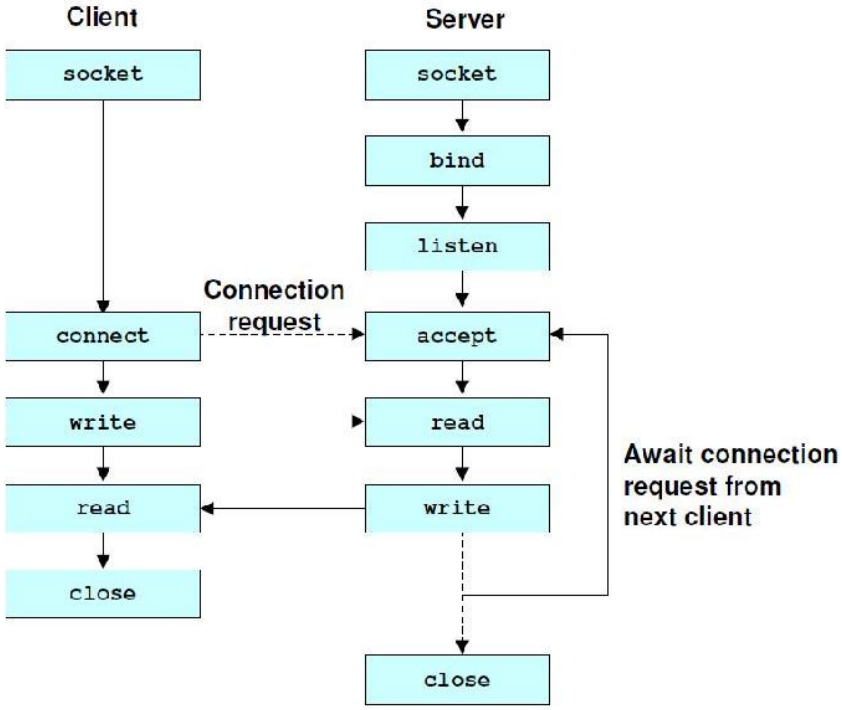
The functions return a NULL pointer in case of error, in which case the external integer *h_errno* may be checked to see whether this is a temporary failure or an invalid or unknown host. Otherwise a valid *struct hostent** is returned.

These functions are not strictly a component of the BSD socket API, but are often used in conjunction with the API functions. Furthermore, these functions are now considered legacy interfaces for querying the domain name system. New functions that are completely protocol- agnostic (supporting IPv6) have been defined. These new function are getaddrinfo() and getnameinfo(), and are based on a new *addrinfo* data structure.

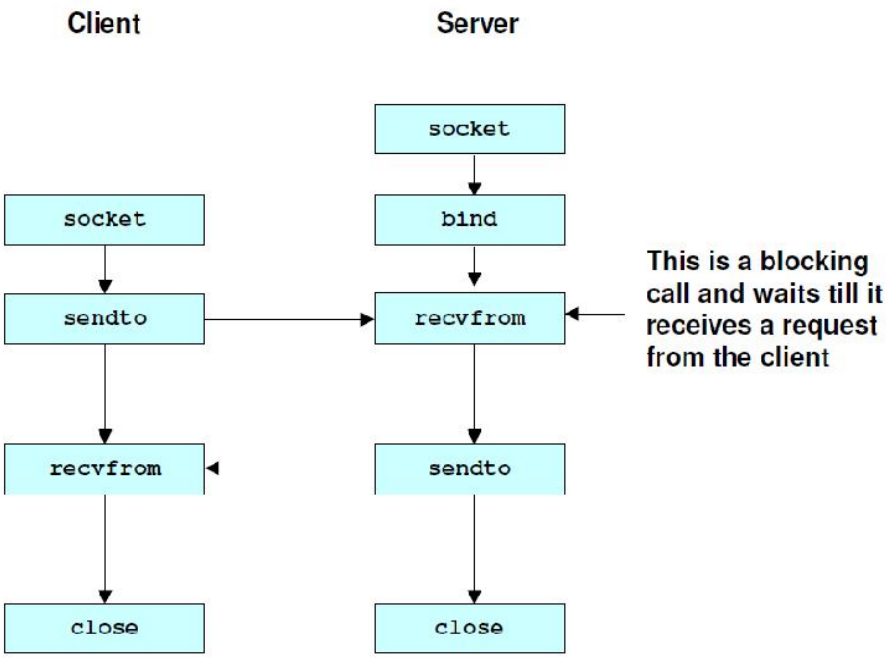
Prototypes:

- struct hostent *gethostbyname(const char *name)
- struct hostent *gethostbyaddr(const void *addr, int len, int type)

Socket programming *with TCP*



Socket programming *with UDP*



Conclusion: TCP & UDP socket programs are studied and executed.

Server Program

```
#include <stdio.h> #include <stdlib.h> #include <string.h> #include <sys/types.h> #include
<sys/socket.h> #include <netinet/in.h> #include <arpa/inet.h> #include <unistd.h> #include <errno.h>

#define PORT 5561
#define BUF_SIZE 2000
#define CLADDR_LEN 100
char *itoaa(int val, int base); int main()
{
struct sockaddr_in addr, cl_addr; int sockfd, len, ret, newsockfd; char buffer[BUF_SIZE];
pid_t childpid;
char clientAddr[CLADDR_LEN]; int num, rem, sum;
char *str;
sockfd = socket(AF_INET, SOCK_STREAM, 0); if (sockfd < 0)
{
printf("Error creating socket!\n"); exit(1);
}

printf("Socket created...\n");

memset(&addr, 0, sizeof(addr)); addr.sin_family = AF_INET; addr.sin_addr.s_addr = INADDR_ANY;
addr.sin_port = PORT;

ret = bind(sockfd, (struct sockaddr *) &addr, sizeof(addr)); if (ret < 0)
{
                        printf("Error binding!\n"); exit(1);

}
else                    printf("Binding done...\n");



printf("Waiting for a connection...@ port no : %d\n",PORT );
```

```
listen(sockfd, 5);

for (;;)                        //infinite loop
{
len = sizeof(struct sockaddr_in);
newsockfd = accept(sockfd, (struct sockaddr
*)&cl_addr,(socklen_t  *)&len);
if (newsockfd <  0)
{
                                printf("Error accepting connection!\n"); exit(1);

}
else                            printf("Connection accepted from ");


inet_ntop(AF_INET, &(cl_addr.sin_addr), clientAddr, CLADDR_LEN);


printf("Port %d of %s Client\n",ntohs(cl_addr.sin_port),inet_ntoa(cl_addr.sin_addr));

if ((childpid = fork())  == 0)                                  //creating a child process
{
                        close(sockfd);
                        //stop listening for new connections by the main
process.
                        //the child will continue to listen.
                        //the main process now handles the connected client.

                        for (;;)
                        {
memset(buffer, 0, BUF_SIZE);

ret = recvfrom(newsockfd, buffer, BUF_SIZE, 0, (struct sockaddr *) &cl_addr, (socklen_t *)&len);

if(ret < 0)
{
                                                printf("Error receiving data!\n"); exit(1);

}
else                                            printf("Received data from Port No %d of
```

```
Client %s : %s\n ", ntohs(cl_addr.sin_port),clientAddr, buffer);


num=atoi(buffer); sum=0; while(num>0)
{
sum = sum + (num % 10); num = num / 10;
}
```

```
strcat(buffer,"                                                    = sum of digits = ");
str=itoaa(sum,10);
strcat(buffer,str);

ret = sendto(newsockfd, buffer, BUF_SIZE, 0, (struct sockaddr *) &cl_addr, len);

if (ret < 0)
{
                                                printf("Error sending data!\n"); exit(1);

}
else                                            printf("\tSent data to %s on Port No %d :

%s\n", clientAddr,ntohs(cl_addr.sin_port), buffer);

printf("                                        ----------
    ----------
\n");
}
}
close(newsockfd);
}
return(0);
}


char *itoaa(int val, int base)
{
static char buf[32] = {0}; int i = 30;
for( ; val && i ; --i, val /= base) buf[i]="0123456789abcdef"[val %  base];

return &buf[i+1];
}
```

Client Program

```
#include"stdio.h" #include"stdlib.h" #include"sys/types.h" #include"sys/socket.h" #include"string.h"
#include"netinet/in.h" #include"netdb.h"

#define PORT 5561
#define BUF_SIZE 2000

int main(int argc, char**argv) { struct sockaddr_in addr, cl_addr; int sockfd, ret;
char buffer[BUF_SIZE]; struct hostent * server; char * serverAddr;

if (argc < 2) {
printf("usage: client < ip address >\n"); exit(1);
}

serverAddr = argv[1];

sockfd = socket(AF_INET, SOCK_STREAM, 0); if (sockfd < 0) {
printf("Error creating socket!\n"); exit(1);
}
printf("Socket created...\n");

memset(&addr, 0, sizeof(addr)); addr.sin_family = AF_INET; addr.sin_addr.s_addr =
inet_addr(serverAddr); addr.sin_port = PORT;

ret = connect(sockfd, (struct sockaddr *) &addr, sizeof(addr)); if (ret < 0) {
printf("Error connecting to the server! : : %d\n",ret); exit(1);
}
printf("Connected to the server @ %s\n",serverAddr);

memset(buffer, 0, BUF_SIZE); printf("Enter your message(s): ");

while (fgets(buffer, BUF_SIZE, stdin) != NULL) {
ret = sendto(sockfd, buffer, BUF_SIZE, 0, (struct sockaddr *) &addr, sizeof(addr));
if (ret < 0) {
printf("Error sending data!\n\t-%s", buffer);
```

```
}
ret = recvfrom(sockfd, buffer, BUF_SIZE, 0, NULL, NULL); if (ret < 0) {
printf("Error receiving data!\n");
} else { printf("Received: "); fputs(buffer, stdout); printf("\n");
}
}

return 0;
}
```

ASSIGNMENT NO : 4

Introduction to server administration (server administration commands and their applications) and configuration of below Server: (Study/Demonstration Only)

a) FTP
b) Web Server

A. FTP Server

Introduction

The File Transfer Protocol (FTP) is used as one of the most common means of copying files between servers over the Internet. Most web based download sites use the built in FTP capabilities of web browsers and therefore most server oriented operating systems usually include an FTP server application as part of the software suite. Linux is no exception.

FTP Overview

FTP relies on a pair of TCP ports to get the job done. It operates in two connection channels as

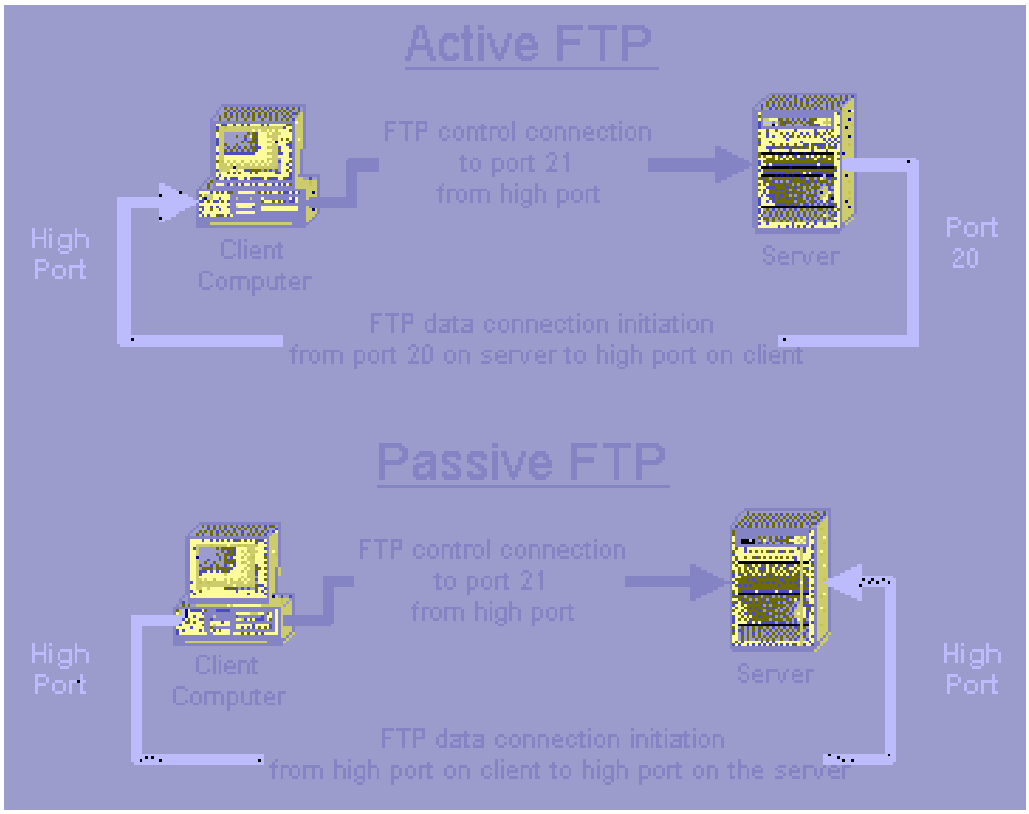
FTP Control Channel, TCP Port 21: All commands you send and the ftp server's responses to those commands will go over the control connection, but any data sent back (such as "ls" directory lists or actual file data in either direction) will go over the data connection.

FTP Data Channel, TCP Port 20: This port is used for all subsequent data transfers between the client and server.

In addition to these channels, there are several varieties of FTP.

Types of FTP

From a networking perspective, the two main types of FTP are active and passive. In active FTP, the FTP server initiates a data transfer connection back to the client. For passive FTP, the connection is initiated from the FTP client. These are illustrated in Figure



From a user management perspective there are also two types of FTP: regular FTP in which files are transferred using the username and password of a regular user FTP server, and anonymous FTP in which general access is provided to the FTP server using a well known universal login method.

Take a closer look at each type.

Active FTP

The sequence of events for active FTP is:

1. Your client connects to the FTP server by establishing an FTP control connection to port 21 of the server. Your commands such as 'ls' and 'get' are sent over this connection.
2. Whenever the client requests data over the control connection, the server initiates data transfer connections back to the client. The source port of these data transfer connections is always port 20 on the server, and the destination port is a high port (greater than 1024) on the client.
3. Thus the ls listing that you asked for comes back over the port 20 to high port connection, not the port 21 control connection.

FTP active mode therefore transfers data in a counter intuitive way to the TCP standard, as it selects port 20 as it's source port (not a random high port that's greater than 1024) and connects back to the client on a random high port that has been pre-negotiated on the port 21 control connection.

Active FTP may fail in cases where the client is protected from the Internet via many to one NAT (masquerading). This is because the firewall will not know which of the many servers behind it should receive the return connection.

Passive FTP

Passive FTP works differently:

1. Your client connects to the FTP server by establishing an FTP control connection to port 21 of the server. Your commands such as ls and get are sent over that connection.
2. Whenever the client requests data over the control connection, the client initiates the data transfer connections to the server. The source port of these data transfer connections is always a high port on the client with a destination port of a high port on the server.

Passive FTP should be viewed as the server never making an active attempt to connect to the client for FTP data transfers. Because client always initiates the required connections, passive FTP works better for clients protected by a firewall.

As Windows defaults to active FTP, and Linux defaults to passive, you'll probably have to accommodate both forms when deciding upon a security policy for your FTP server.

Regular FTP

By default, the VSFTPD package allows regular Linux users to copy files to and from their home directories with an FTP client using their Linux usernames and passwords as their login credentials.

VSFTPD also has the option of allowing this type of access to only a group of Linux users, enabling you to restrict the addition of new files to your system to authorized personnel.

The disadvantage of regular FTP is that it isn't suitable for general download distribution of software as everyone either has to get a unique Linux user account or has to use a shared username and password. Anonymous FTP allows you to avoid this difficulty.

Anonymous FTP

Anonymous FTP is the choice of Web sites that need to exchange files with numerous unknown remote users. Common uses include downloading software updates and MP3s and uploading diagnostic information for a technical support engineers' attention. Unlike regular FTP where you login with a preconfigured Linux username and password, anonymous FTP requires only a username of anonymous and your email address for the password. Once logged in to a VSFTPD server, you automatically have access to only the default anonymous FTP directory (/var/ftp in the case of VSFTPD) and all its subdirectories.

Configuration of FTP Server in Linux using VSFTPD

INSTALLING VSFTPD

While there are a variety of FTP server tools available for Linux, one of the most popular and mature options is vsftpd.

Begin by SSHing into your server as root and use the apt-get command to install *vsftpd*:

$ apt-get update

$ apt-get install vsftpd Reading package lists... Done Building dependency tree
Reading state information... Done [...]
The following NEW packages will be installed:

vsftpd

0 upgraded, 1 newly installed, 0 to remove and 18 not upgraded. Need to get 111 kB of archives.
After this operation, 361 kB of additional disk space will be used.

Get:1 http://mirrors.digitalocean.com/ubuntu/ trusty-updates/main vsftpd amd64 3.0.2-

1ubuntu2.14.04.1 [111 kB] Fetched 111 kB in 0s (231 kB/s) Preconfiguring packages ...
Selecting previously unselected package vsftpd.

(Reading database ... 175600 files and directories currently installed.) Preparing to unpack .../vsftpd_3.0.2-1ubuntu2.14.04.1_amd64.deb ... Unpacking vsftpd (3.0.2-1ubuntu2.14.04.1) ...
Processing triggers for man-db (2.6.7.1-1) ... Processing triggers for ureadahead (0.100.0-16) ... Setting up vsftpd (3.0.2-1ubuntu2.14.04.1) ... vsftpd start/running, process 18690
Processing triggers for ureadahead (0.100.0-16) ...

CONFIGURATION

The next step is to change any configuration settings for vsftpd. Open the /etc/vsftpd.conf file in your preferred text editor:

nano /etc/vsftpd.conf

# Example config file /etc/vsftpd.conf # ...
# Run standalone? vsftpd can run either from an inetd or as a standalone # daemon started from an initscript.
listen=YES #
# Allow anonymous FTP? (Disabled by default) anonymous_enable=NO
#
Uncomment this to allow local users to log in. local_enable=YES
#
Uncomment this to enable any form of FTP write command. #write_enable=YES
You may restrict local users to their home directories. See the FAQ for # the possible risks in this before using chroot_local_user or
chroot_list_enable below. #chroot_local_user=YES

The critical settings seen above are outlined below:

listen=YES tells vsftpd to run as a standalone daemon (the simplest method for getting up and running).anonymous_enable=NO disallows anonymous FTP users, which is generally preferred for security reasons but can be enabled for testing purposes.

local_enable=YES allows any user account defined in the /etc/passwd file access to the FTP server and is generally how most FTP users will connect.

write_enable=YES is commented out by default, but removing the hash (#) allows files to be uploaded to the FTP server.chroot_local_user=YES restricts users to their home directory and is also commented out by default.

To begin your testing and make sure everything is working, start with the following settings for the above parameters:

listen=YES anonymous_enable=YES local_enable=YES write_enable=YES chroot_local_user=YES

Save the vsftpd.conf file then restart the vsftpd service for the changes to take effect:

sudo service vsftpd restart vsftpd stop/waiting
vsftpd start/running, process 18954

TESTING YOUR FTP SERVER

To quickly determine if your server was installed properly and is up and running, try to connect to the FTP server from your active shell, using the name anonymous and a blank password:

ftp localhost Connected to localhost. 220 (vsFTPd 3.0.2)
Name (localhost:root): anonymous 331 Please specify the password. Password:
230 Login successful. Remote system type is UNIX.
Using binary mode to transfer files. ftp>

With both anonymous_enable and local_enable set to "YES" in the configuration, you should be able to successfully login to your local FTP server as seen above!

With that out of the way, simply enter quit at the ftp> prompt to cancel out:

ftp> quit

221 Goodbye.

With the test complete, you may wish to disable anonymous access once again by settinganonymous_enable=NO in the /etc/vsftpd.conf file and restarting the service:

nano /etc/vsftpd.conf

Set to NO to disable anonymous access

anonymous_enable=NO sudo service vsftpd restart vsftpd stop/waiting
vsftpd start/running, process 18996

ADDING AN FTP USER

If this is a new server it may be advisable to add a specific user for FTP access. Doing so is a fairly simple process but begin by creating a new user:

sudo adduser foobarAdding user `foobar' ...Adding new group `foobar' (1000) ...Adding new user `foobar' (1000) with group `foobar' ...Creating home directory `/home/foobar' ...Copying files from `/etc/skel' ...Enter new UNIX password:Retype new UNIX password:passwd: password updated successfullyChanging the user information for foobarEnter the new value, or press ENTER for the default Full Name []: Room Number []: Work Phone []: Home Phone []: Other []:Is the information correct? [Y/n] Y

With a new user added you can now connect to your server remotely with an FTP client such as FileZilla, but you will immediately run into an error:

Status: Connecting to 104.131.170.253:21...Status: Connection established, waiting for welcome message...Response: 220 (vsFTPd 3.0.2)Command: USER foobarResponse: 331 Please specify the password.Command: PASS **********Response: 500 OOPS: vsftpd: refusing to run with writable root inside chroot()

The "500 OOPS" error vsftpd returns is a security measure designed to prevent *writable* root access for FTP users by default.To resolve this issue there are two main options available.

Allowing Writable User-root Access

The simplest method is to alter the /etc/vsftpd.conf file once again and enable one particular setting:

nano /etc/vsftpd.conf

Allow users to write to their root directory allow_writeable_chroot=YES

With allow_writeable_chroot enabled following a service vsftpd restart, you can now successfully FTP into your server remotely as your newly created user:

Status: Connecting to 104.131.170.253:21...Status: Connection established, waiting for welcome message...Response: 220 (vsFTPd 3.0.2)Command: USER foobarResponse: 331 Please specify the password.Command: PASS **********Response: 230 Login successful.

Using Writeable Subdirectories

The other option to maintain slightly stronger security is not to enable allow_writeable_chroot as outlined above, but instead to create a new subdirectory in the user's root directory with write access:

sudo chown root:root /home/foobar sudo mkdir /home/foobar/uploads
sudo chown foobar:foobar /home/foobar/uploads sudo service vsftpd restart

Now when you connect remotely to your FTP server as the new user, that user will not have write access to the root directory, but will instead have full write access to upload files into the newly created uploads directory instead.

SECURING YOUR FTP WITH SSL

While standard unencrypted FTP access as outlined so far is sufficient in many cases, when transferring sensitive information over FTP it is useful to utilize a more secure connection using SSL.

To begin you'll likely need to generate a new SSL certificate with the following command, following the prompts as appropriate to complete the process:

sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout /etc/ssl/private/vsftpd.pem -out /etc/ssl/private/vsftpd.pem

Now you must ensure that vsftpd is aware of the SSL certificate. Open the /etc/vsftpd.conf file once again:

sudo nano /etc/vsftpd.conf

Look near the bottom of the file for two rsa_ settings like this, indicating the location of the SSL certificate that was just created:

rsa_cert_file=/etc/ssl/private/vsftpd.pem rsa_private_key_file=/etc/ssl/private/vsftpd.pem

If those lines don't exist or match the appropriate path to the SSL certificate created, update them accordingly.

Additionally, there are a number of configuration settings to handle SSL connections, particularly forcing use of the TLS protocol which is ideal:

ssl_enable=YES allow_anon_ssl=NO force_local_data_ssl=YES force_local_logins_ssl=YES ssl_tlsv1=YES ssl_sslv2=NO ssl_sslv3=NO require_ssl_reuse=NO ssl_ciphers=HIGH

Some of the settings are self-explanatory, but the key components are the overall enabling of SSL, the restriction to use only TLS, and disallowing anonymous access.

With the settings added and the file saved, once again restart the vsftpd service:

sudo service vsftpd restart

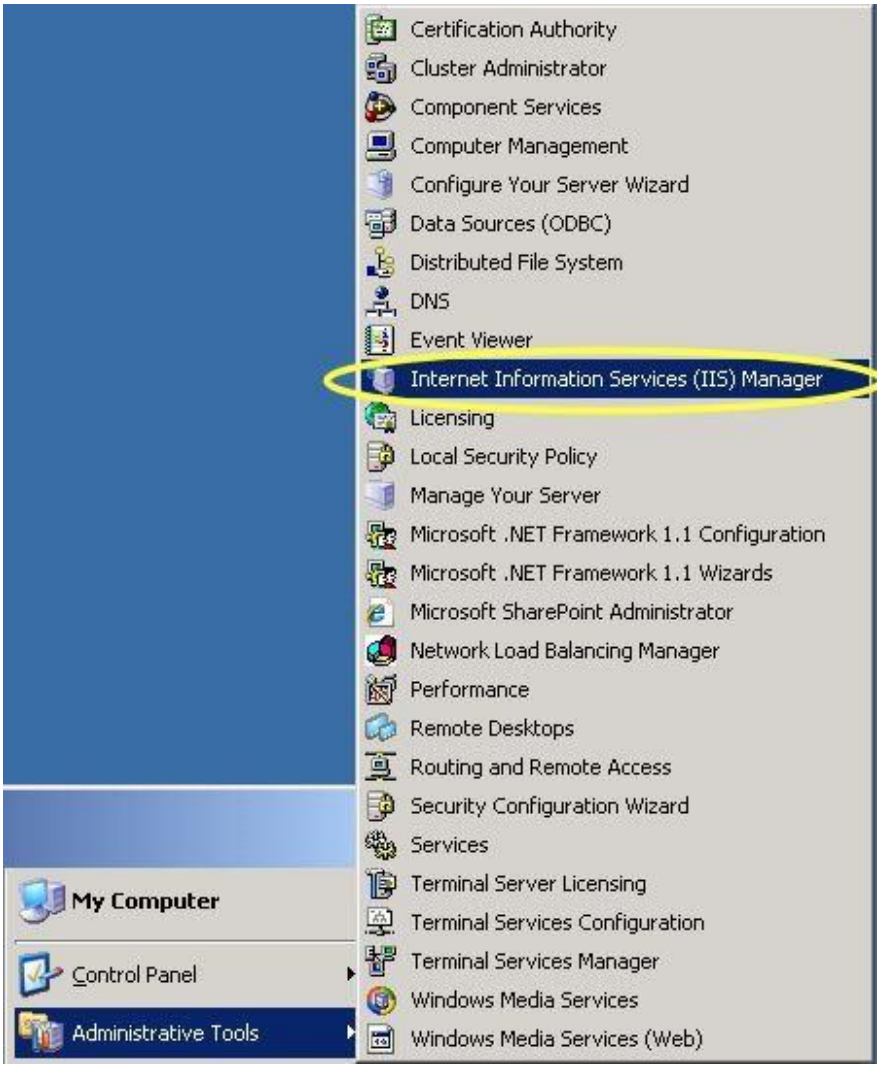
Now your FTP server is ready to accept secure connections using "FTP over TLS" encryption. Using a client such as FileZilla, you will be presented with a certificate popup asking to verify the newly created SSL certification.

Upon accepting you will now be securely connected and transfers will be encrypted via SSL:

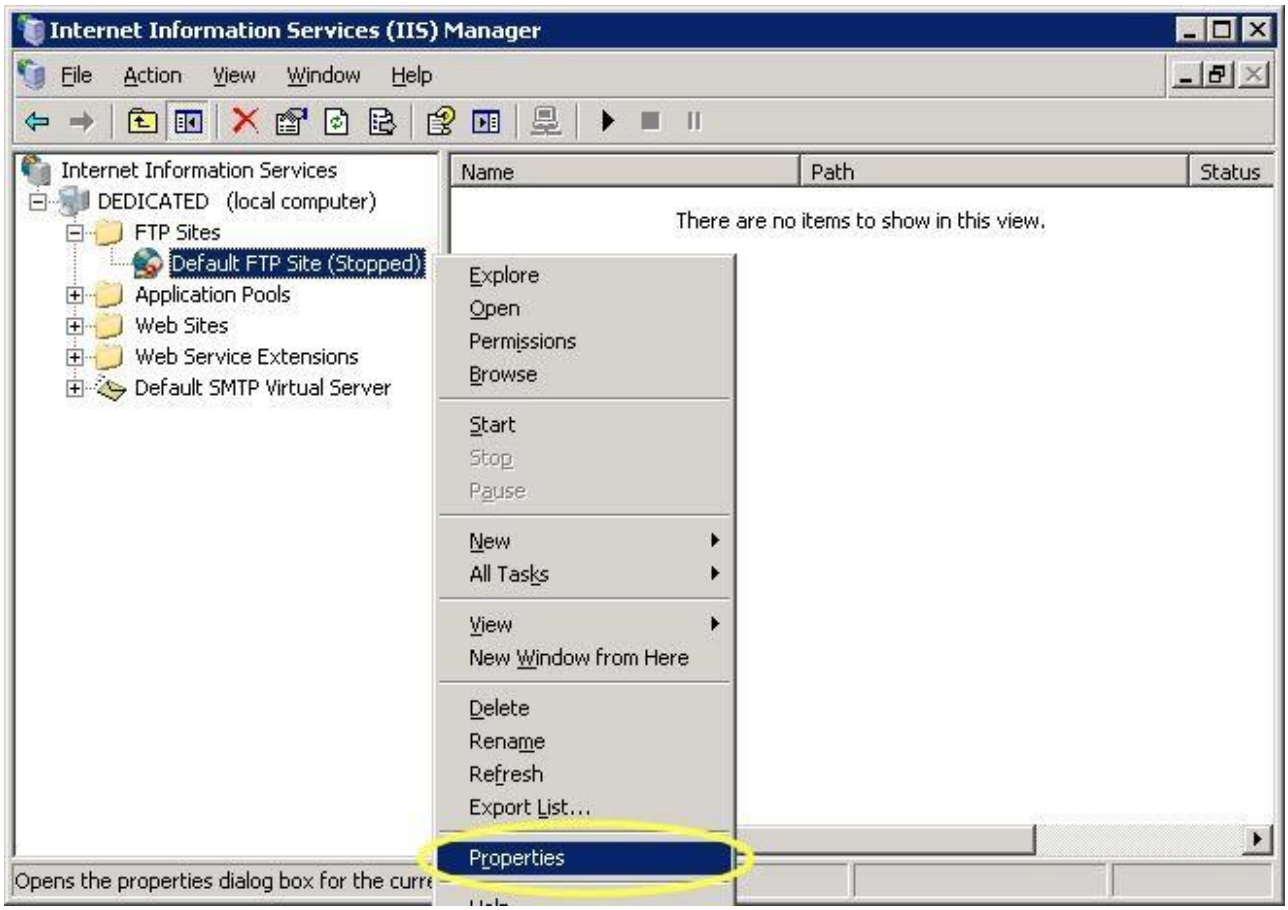
Status: Connecting to 104.131.170.253:21...Status: Connection established, waiting for welcome message...Response: 220 (vsFTPd 3.0.2)Command: AUTH TLSResponse: 234 Proceed with negotiation.Status: Initializing TLS...Status: Verifying certificate...Command: USER foobarStatus: TLS/SSL connection established.Response: 331 Please specify the password.Command: PASS **********Response: 230 Login successful.

Creating and Configuring FTP Server in Windows Server 2003

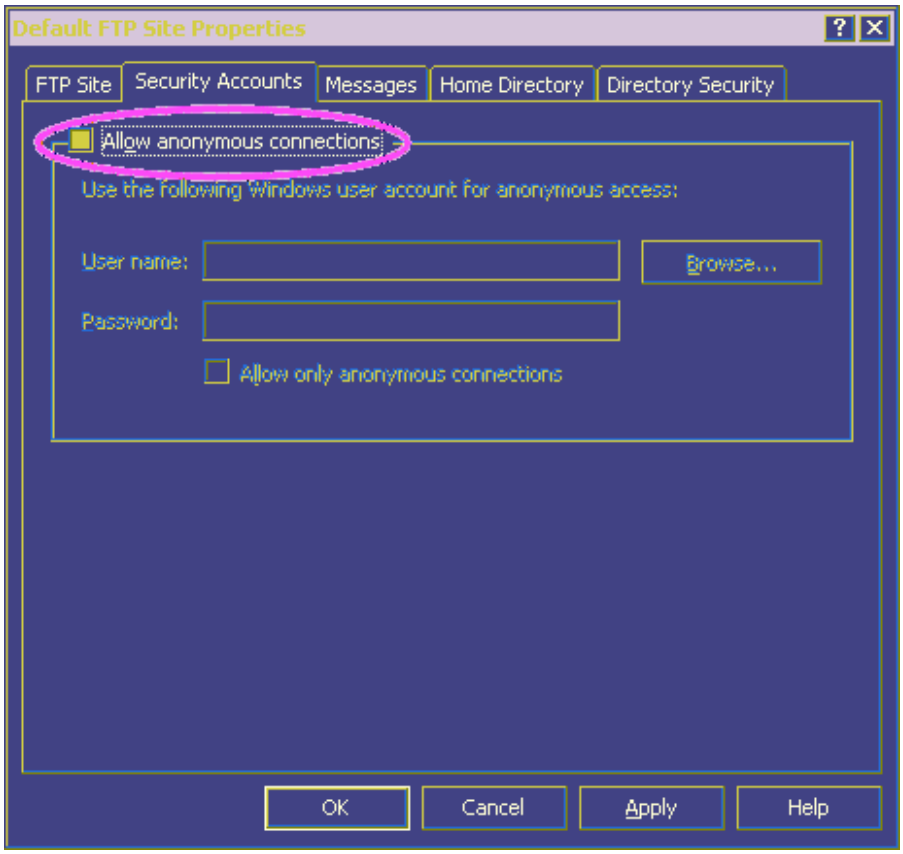
To enable the FTP service, go to the start menu -> Administrative Tools -> Internet Information Services (IIS) Manager.



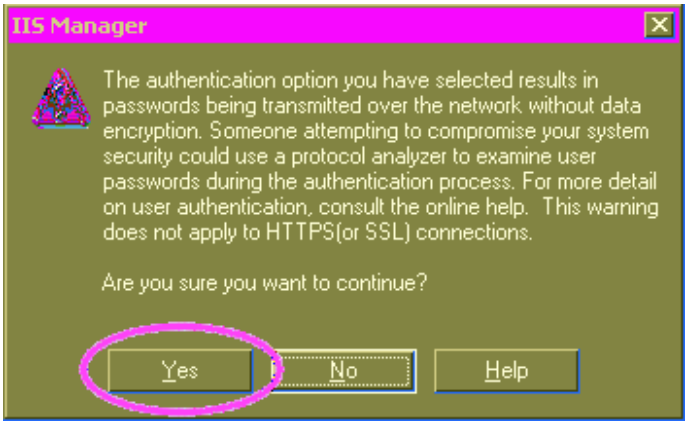
Tree open your server in the Internet Information Services list. Here my server is called "DEDICATED". Then tree open FTP Sites and right-click Default FTP Site (Stopped). Click Properties from the context menu.



On the dialog, choose the Security Accounts tab. Make sure to uncheck Allow anonymous connections. We don't want to allow anonymous access to the FTP server or we will have spammers, porn-servers, and who knows what else on here in a matter of days. We only want to allow authenticated user accounts to connect.



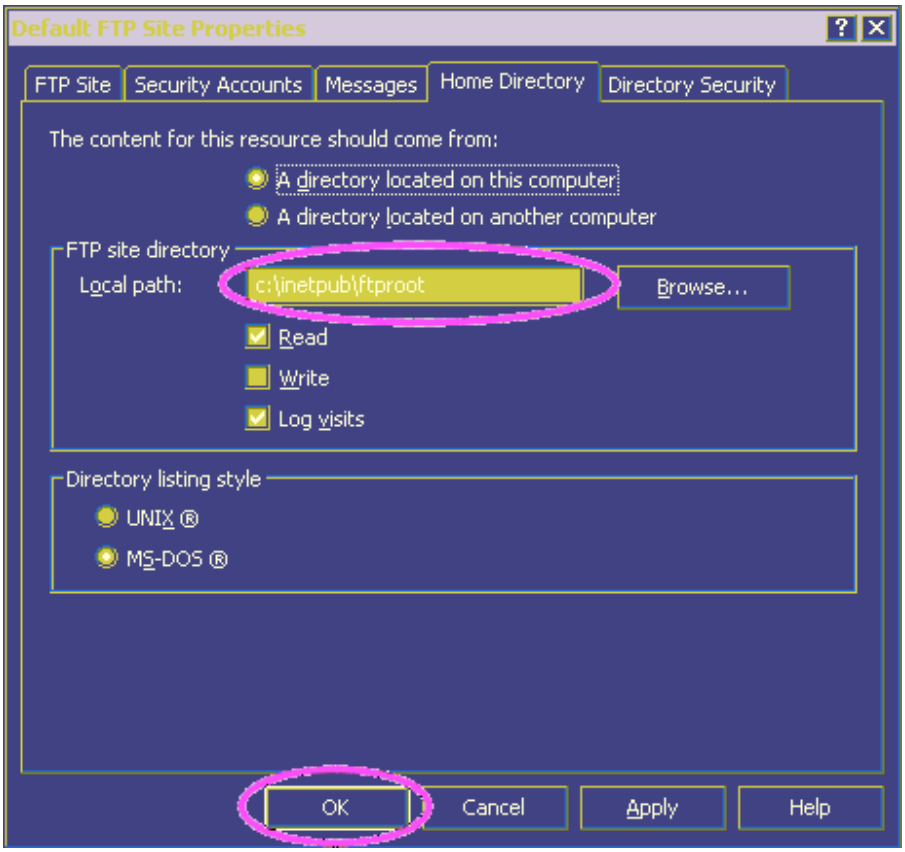
When you uncheck this box, you will see a dialog like this. Basically, this is just telling you that since you don't want to allow anonymous access, you will have to login with a UserName and Password. Since FTP is not a secure protocol, these credentials will be passed in clear text and there is a remote possibility that someone could see the credentials. In other words, this is saying, make sure that you don't use base windows accounts that you want to be secure. I recommend using a dummy ftp account that you change on a regular basis instead. Just click Yes on this dialog.



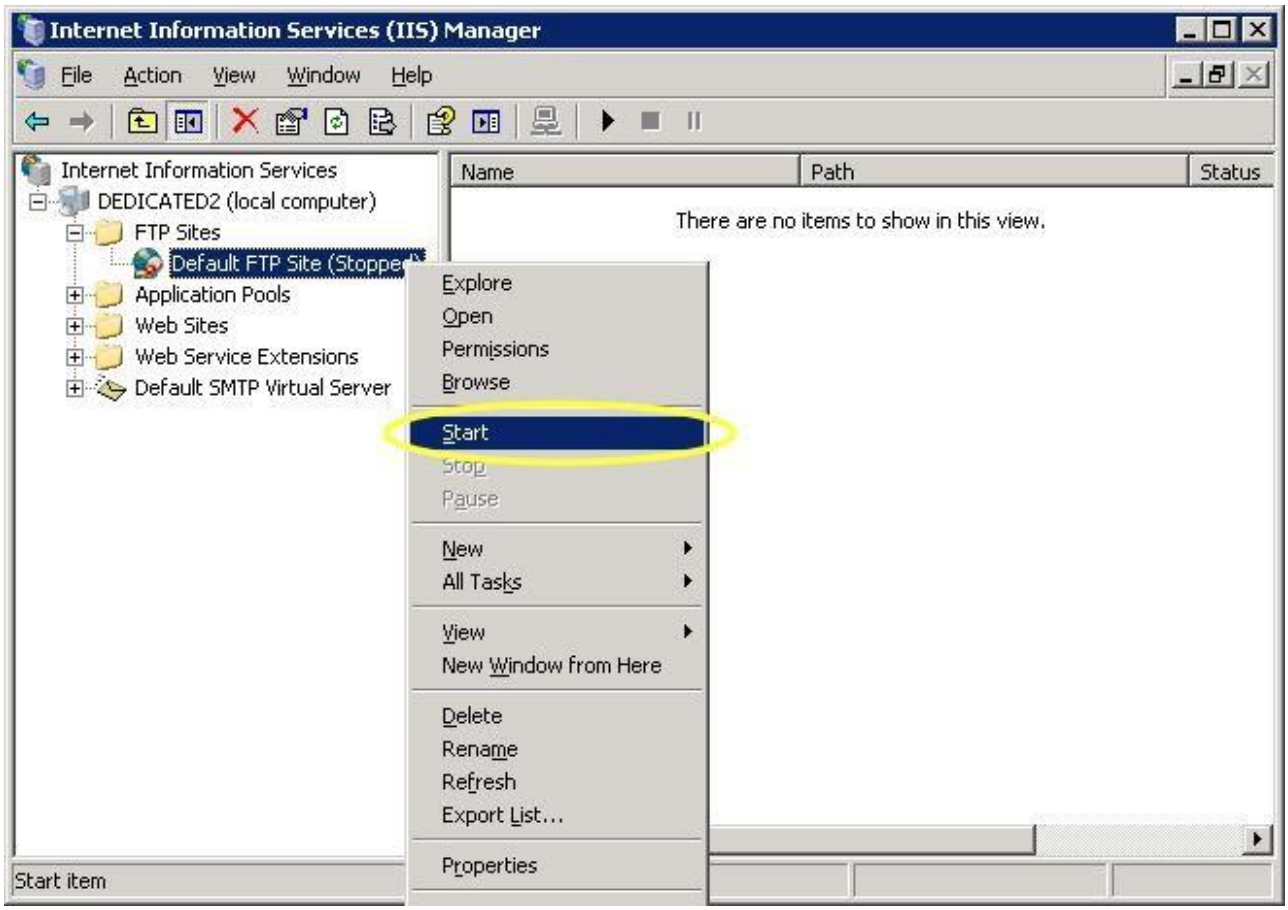
On the Home Directory tab, set the path to where you want your FTP files to be placed. NOTE: By default the path is set to inetpub\ftproot. If you want to allow users to create directories and add files

instead of just downloading, make sure the Write box is checked. Then click OK to

apply all these changes.

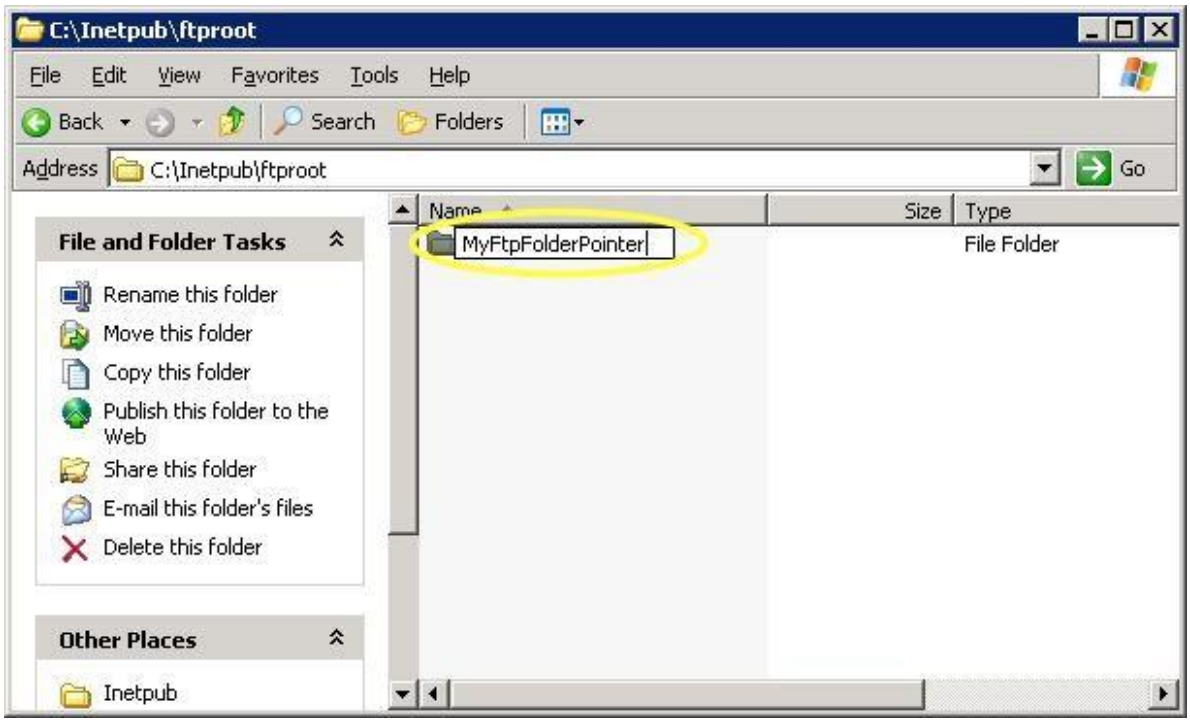


Now we want to start our FTP service. Right-click the Default FTP Site (Stopped) in the tree view and select Start to run the FTP server.

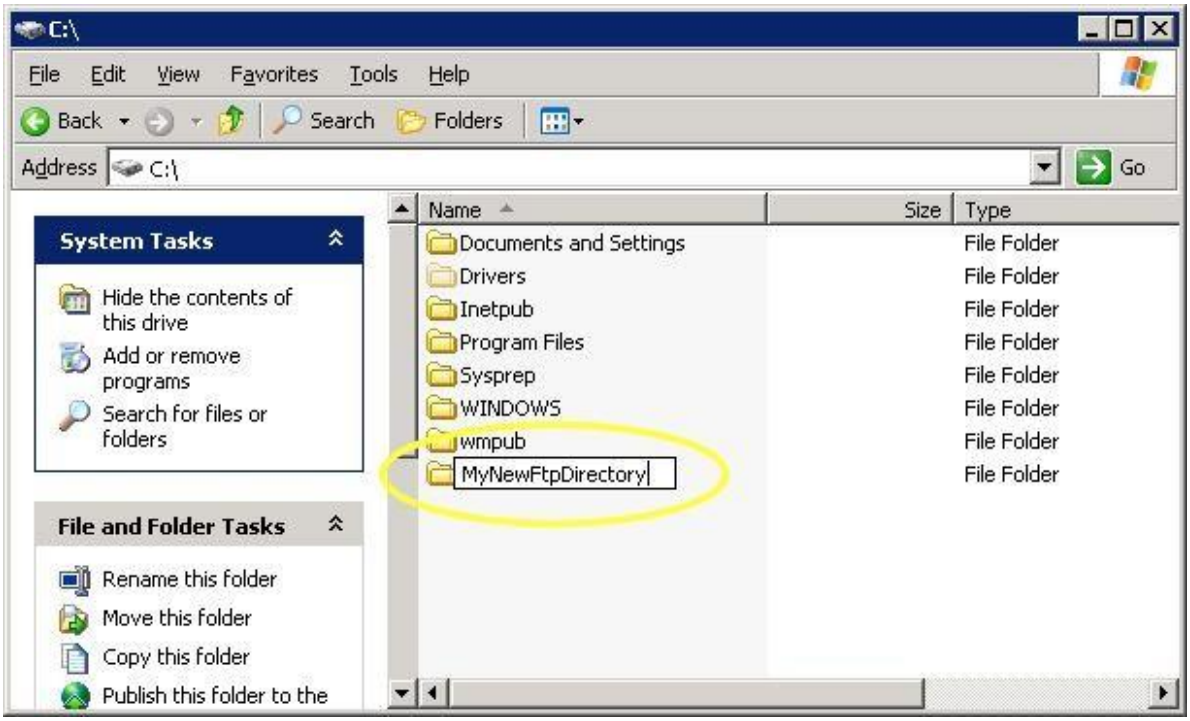


Setting Up FTP Directories & Permissions

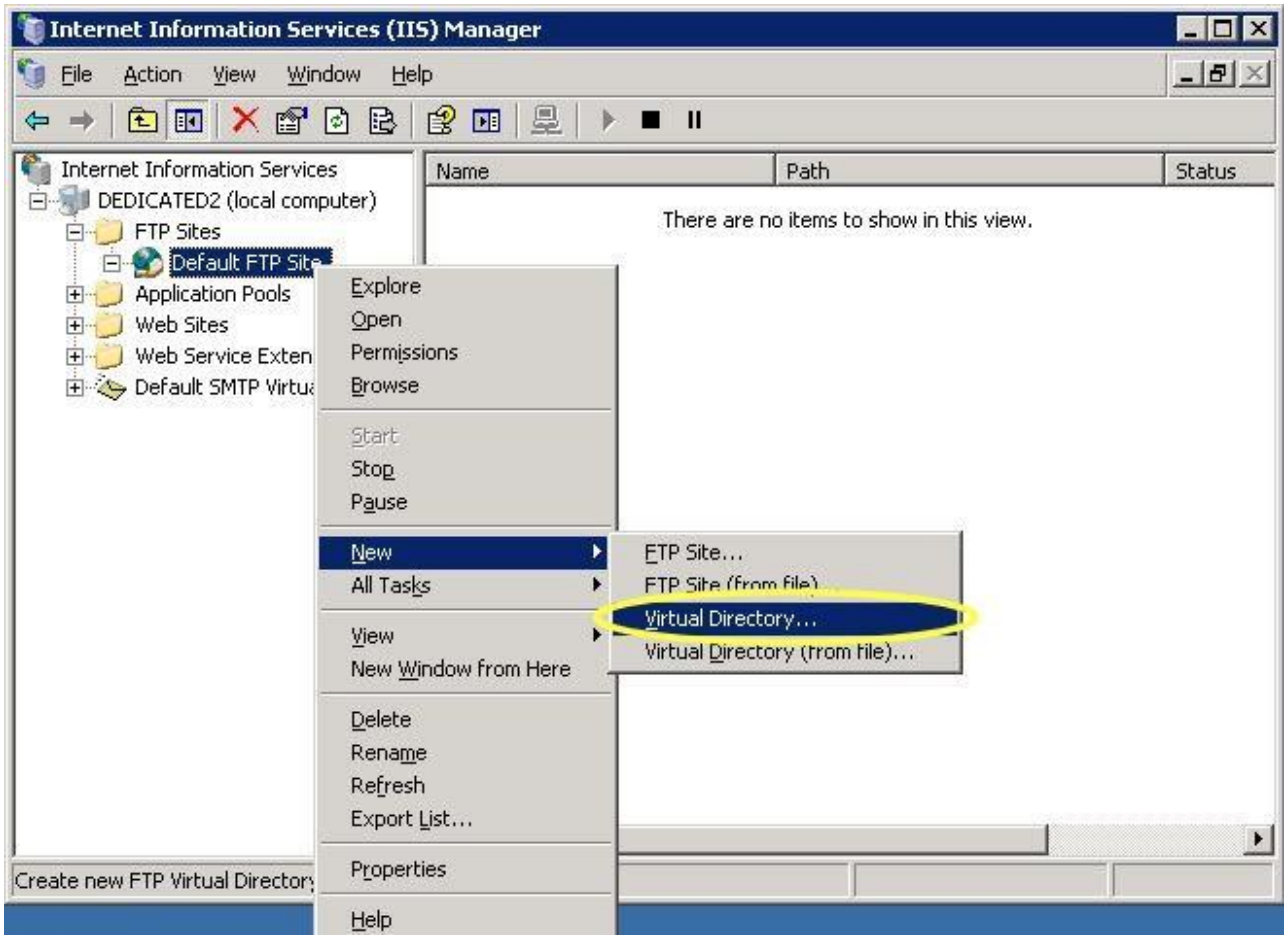
The previous tasks are all you need to do if you want to just put files in the mail FTP directory. But sometimes you want to set up specific directories for users that actually put the files in different directories than the default directory. The way you do this is to set up a "pointer" directory in your default inetpub\ftproot that will just be an empty folder (FTP Service requires this for a virtual directory). Here, I created a new folder in my default FTP root folder called "MyFtpFolderPointer".



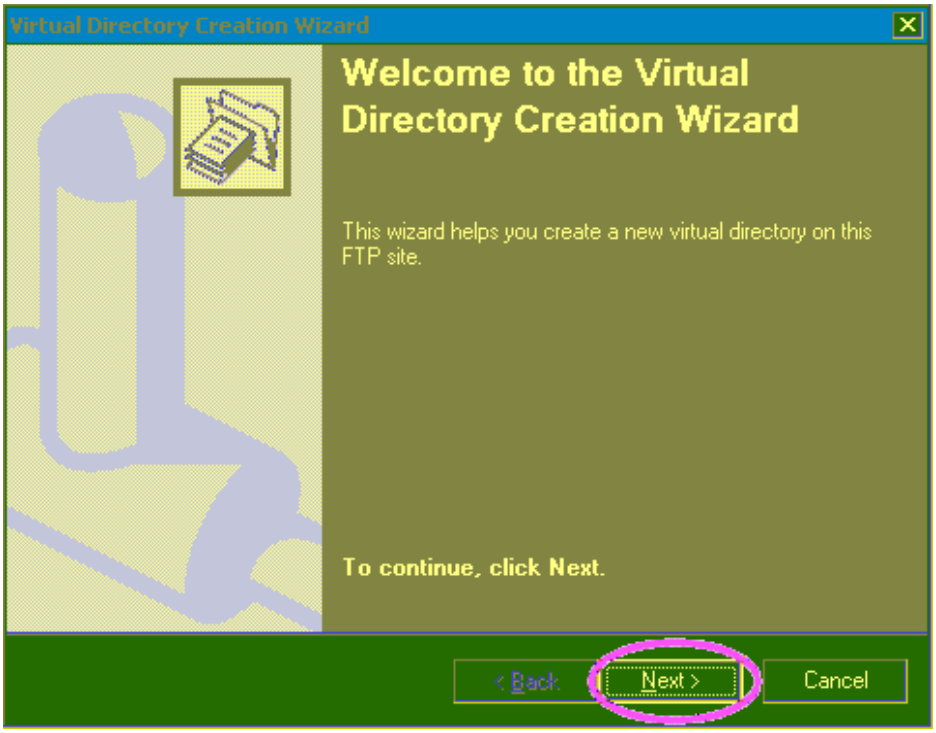
Now, we create a folder where we actually want our files to be placed when they are uploaded/downloaded. So I put a folder in the C:\ drive and called it "MyNewFtpDirectory". This is the place where the FTP files will actually go and the folder we created in the previous step will point to this folder.



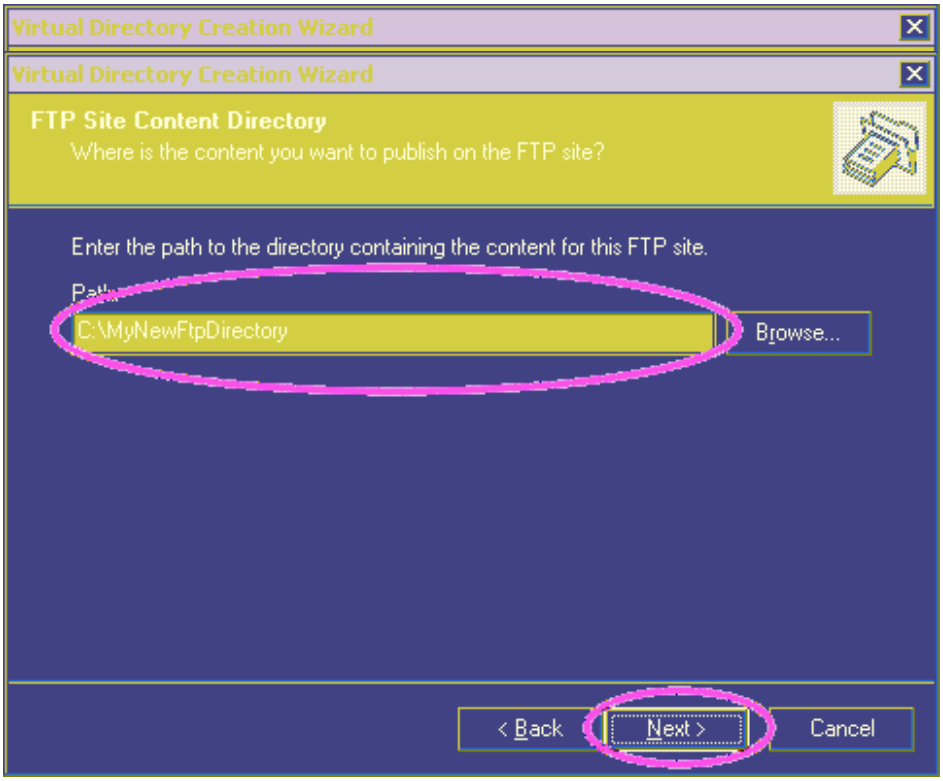
Now go back to the Internet Information Services (IIS) Manager and right-click the Default FTP Site. Choose New -> Virtual Directory... to start the virtual directory wizard.



Click Next to start the wizard.

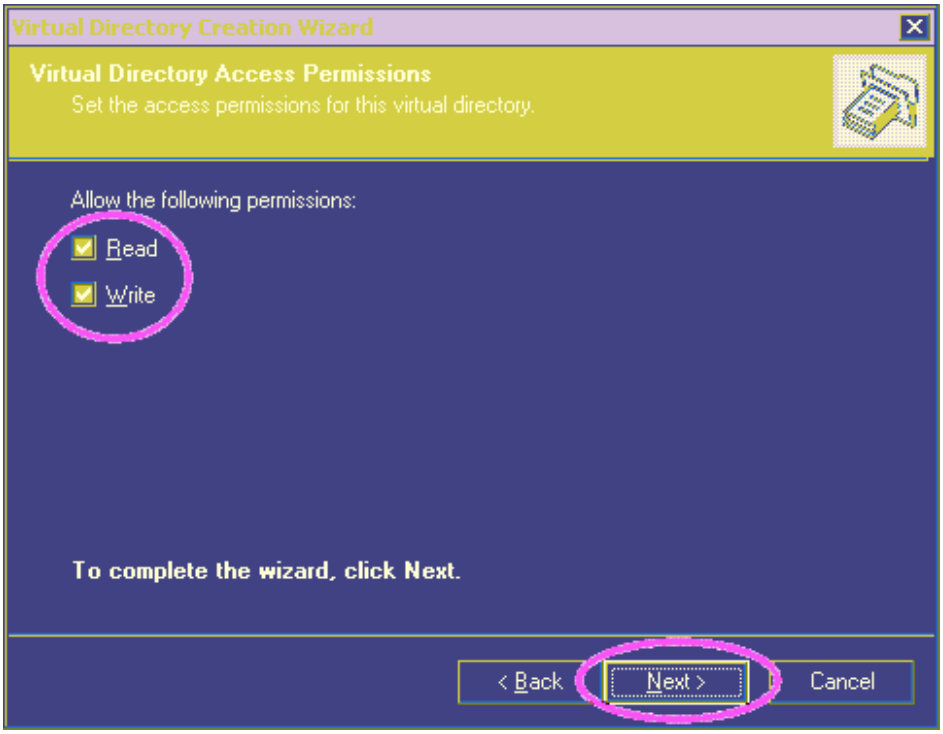

Virtual Directory Creation Wizard
Welcome to the Virtual Directory Creation Wizard
This wizard helps you create a new virtual directory on this FTP site.
To continue, click Next.
< Back
Next >
Cancel

Name your alias for this virtual directory the EXACT same name you named the Virtual Directory folder we created in the FTP root since this is the one we want to point to the C:\ drive folder. So here, we name our Virtual Directory "MyFtpFolderPointer". Click Next.

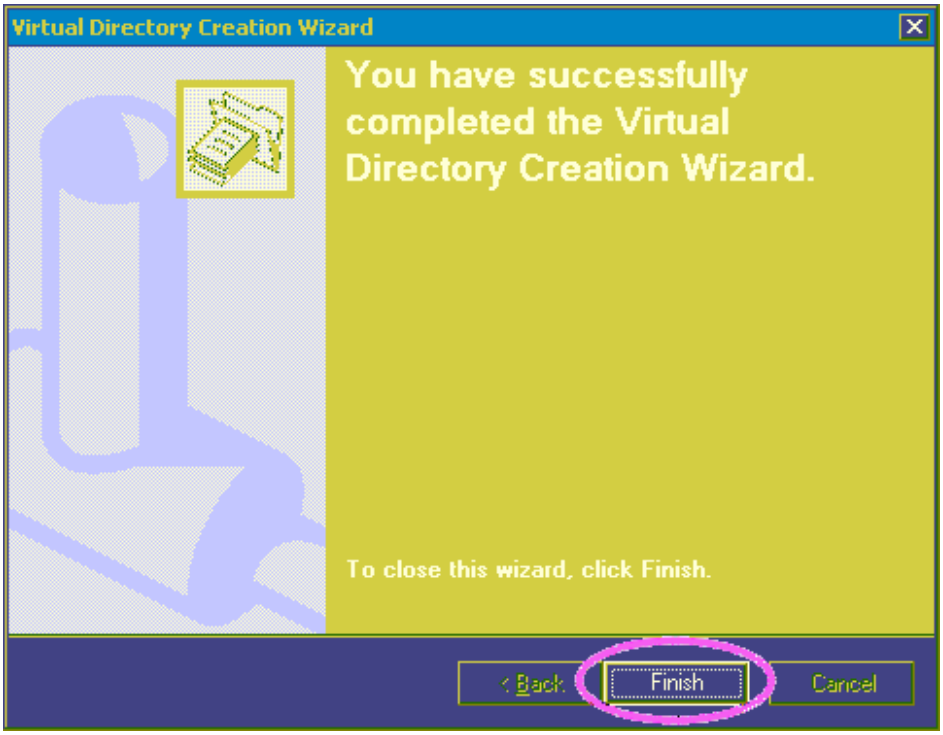


Now we tell this virtual directory what it's actual path should be. So point this path to the folder where you actually want the FTP files to go. It can be any path you want. Here we put the path to our folder on C:\MyNewFtpDirectory. Click Next.

If you want users to be able to both download and upload files to this FTP directory, check both the Read and Write boxes. Click Next.



Click Finish to complete the wizard and apply the virtual directory settings.



By default, FTP used Port 21 so you will need make sure that your Windows Firewall (if that is what you are using) is configured to allow Port 21 for FTP.

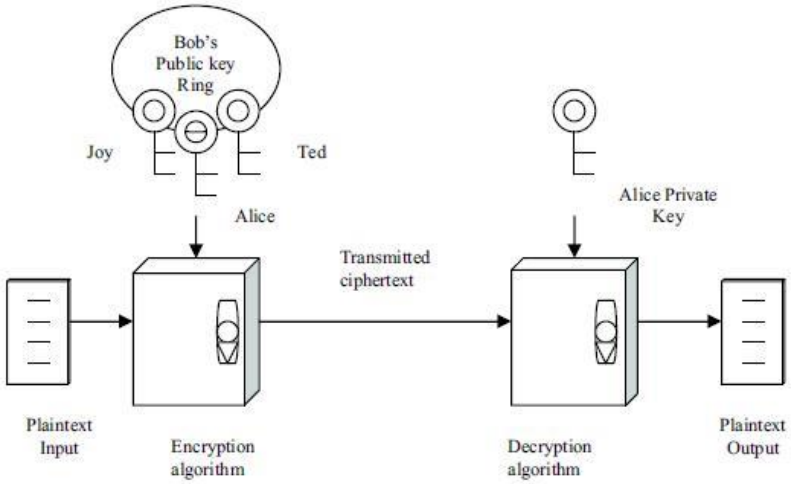
Group B: Network Security

ASSIGNMENT NO : 1

Aim: Implement a client and a server on different computers using python. Perform the communication
between these two entities by using RSA cryptosystem.

Theory:
Asymmetric/Public key Algorithm
This type of algorithm rely on one key for encryption and a different but related key for decryption. These
algorithms have the following important characteristic:
It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.
In addition, some algorithms, such as RSA, also exhibit the following characteristic:
Either of the two related keys can be used for encryption, with the other used for decryption.
A public-key encryption scheme has six ingredients:
Plaintext: This is the readable message or data that is fed into the algorithm as input.
Encryption algorithm: The encryption algorithm performs various transformations on the plaintext.
Public and private keys: This is a pair of keys that have been selected so that if one is used for encryption,
the other is used for decryption. The exact transformations performed by the algorithm depend on the
public or private key that is provided as input.

Ciphertext: This is the scrambled message produced as output. It depends on the plaintext and the key.
For a given message, two different keys will produce two different ciphertexts. Decryption algorithm:
This algorithm accepts the ciphertext and the matching key and produces the original plaintext.



The essential steps are the following:

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or other accessible file. This is the public key.

The companion key is kept private. Each user maintains a collection of public keys obtained from others.

3. If B wishes to send a confidential message to A, B encrypts the message using A's public key.
4. When A receives the message, she decrypts it using her private key. No other recipient can decrypt the

message because only A knows A's private key.

RSA Algorithm

RSA(which stands for Ron Rivest, Adi Shamir, and Len Adelman),an algorithm for public- key cryptography involves three steps key generation,encryption and decryption.The RSA scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and n − 1. A typical size of n is 1024 bits, or 309 decimal digits. Thus, the plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to log2(n). Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C:

$C = Me \bmod n$

$M = Cd \bmod n$

Both sender and receiver must know the value of n. The sender knows the value of e, and only the receiver knows the value of d. Thus, this is a public-key encryption algorithm with a public key of PU = {e, n} and a PR = {d, n}.

Algorithm

1. Key Generation

1. Select p, q p and q both prime, p ! q
2. Calculate n = p *q
3. Calculate $\Phi(n) = (p - 1)(q - 1)$
4. Select integer e ,such that $\gcd(\Phi(n), e) = 1; 1 < e < \Phi(n)$

5. Calculate d = e-1 mod Φ(n)
6. Public key PU = {e, n}
7. Private key KR = {d, n}
2. Encryption
Plaintext: M < n
Ciphertext: C = Me (mod n)
3. Decryption
Ciphertext: C
Plaintext: M = Cd(mod n)

Example
Select primes p=11, q=3.
n=pq=11.3=33
Φ(n) = (p-1)(q-1) = 10.2 = 20
Choose e=3
Check gcd(e, p-1) = gcd(3, 10) = 1 (i.e. 3 and 10 have no common factors except 1),
and check gcd(e,q-1)=gcd(3,2)=1
therefore gcd(e, Φ(n)) = gcd(e, (p-1)(q-1)) = gcd(3, 20) = 1
compute d = e-1 mod Φ(n) = 3-1 mod 20
i.e. find a value for d such that Φ(n) divides (ed-1)
i.e. find d such that 20 divides 3d-1.
Simple testing (d = 1, 2,...) gives d = 7
Check: ed-1 = 3.7 - 1 = 20, which is divisible by Φ(n).
Public key = (n, e) = (33, 3)
Private key = (n, d) = (33, 7).
Now say we want to encrypt the message M = 7
C = Me mod n = 73 mod 33 = 343 mod 33 = 13.
Hence the ciphertext C= 13.
To check decryption we compute
M'= C d mod n = 137 mod 33 = 7.
Input: Select p & q large prime nos.
Information, which is to be, encrypt.
Output: Sending values of N and E.
Information encryption and decryption.
Conclusion: Thus RSA algorithm is used to implement asymmetric key cryptography.

Assignment 5
FAQ:
1. What are possible attacks on RSA.
2. List applications of Public Key cryptography.
3. State Fermat's little theorem with an example

ASSIGNMENT NO : 2

Implement a client and a server on different computers using python. Perform the authentication of sender between these two entities by using RSA digital signature cryptosystem.

FAQ:

1. List security objectives satisfied by Digital Signature
2. What are various ways of user authentication.
3. Distinguish between authentication and Authorization.

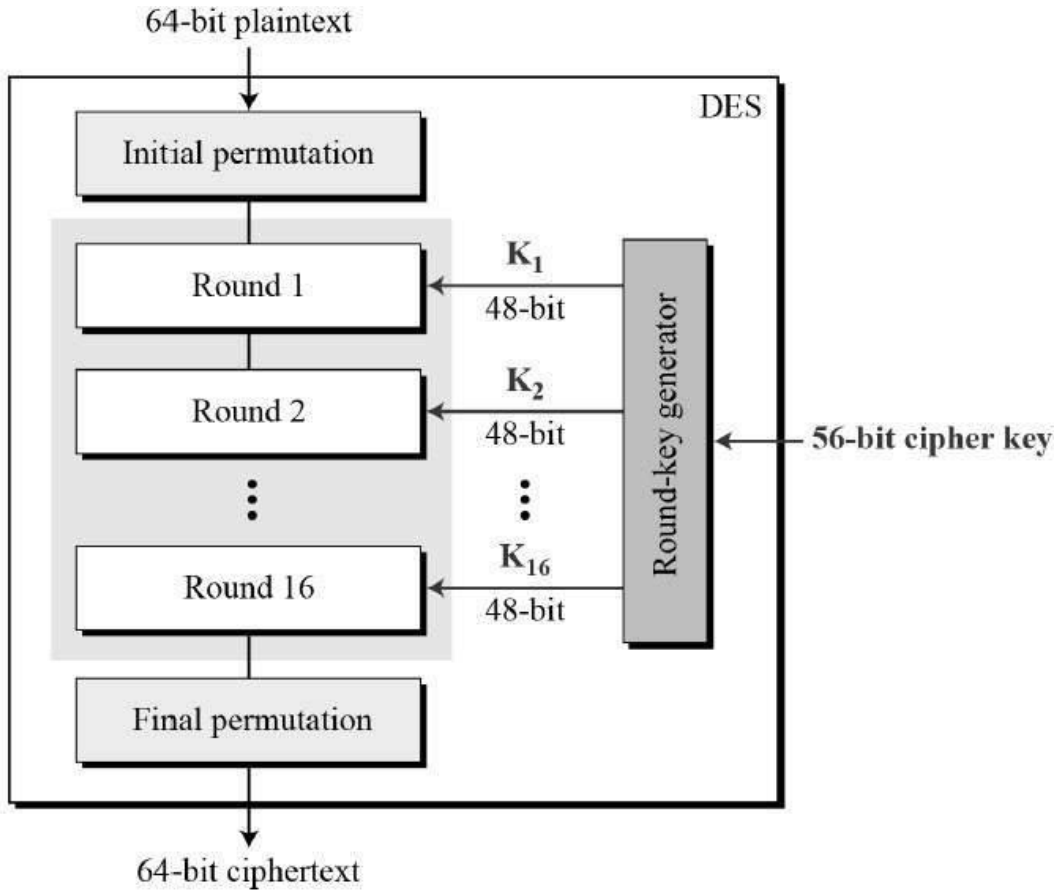
ASSIGNMENT NO : 3

Aim : Implement a client and a server on different computers using python. Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys

Theory:

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration −

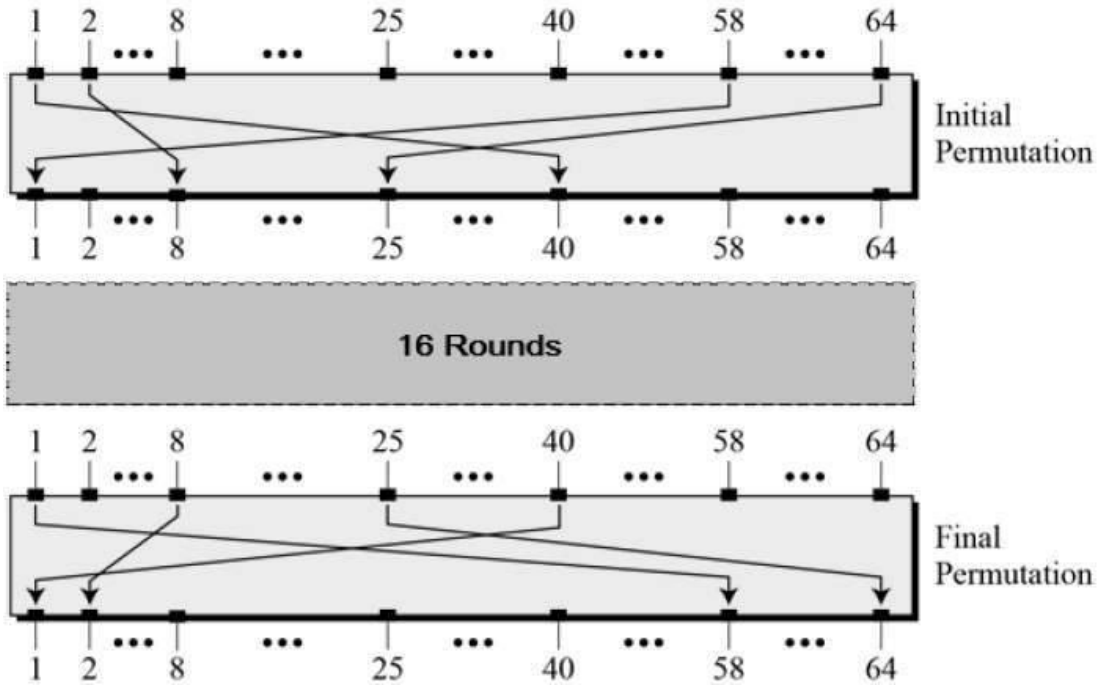


Since DES is based on the Feistel Cipher, all that is required to specify DES is −

- R
ound function
- K
ey schedule
- A
ny additional processing − Initial and final permutation

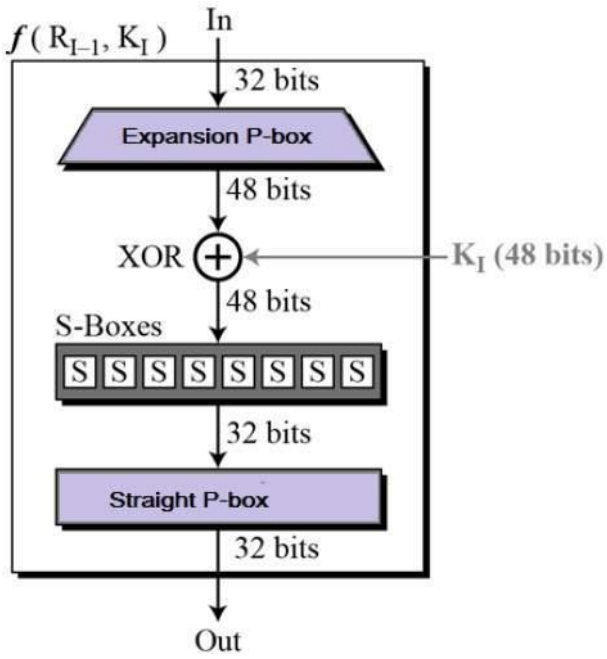
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows −

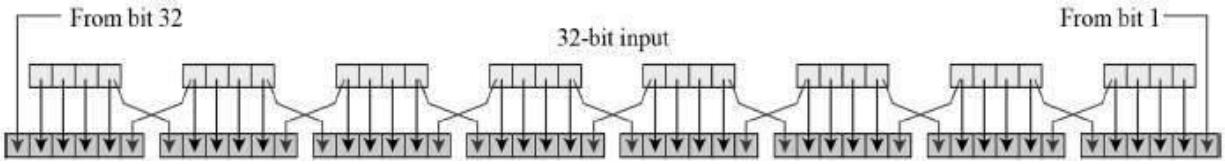


Round Function

The heart of this cipher is the DES function, *f*. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



Expansion Permutation Box − Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration −



XOR (Whitener). − After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.

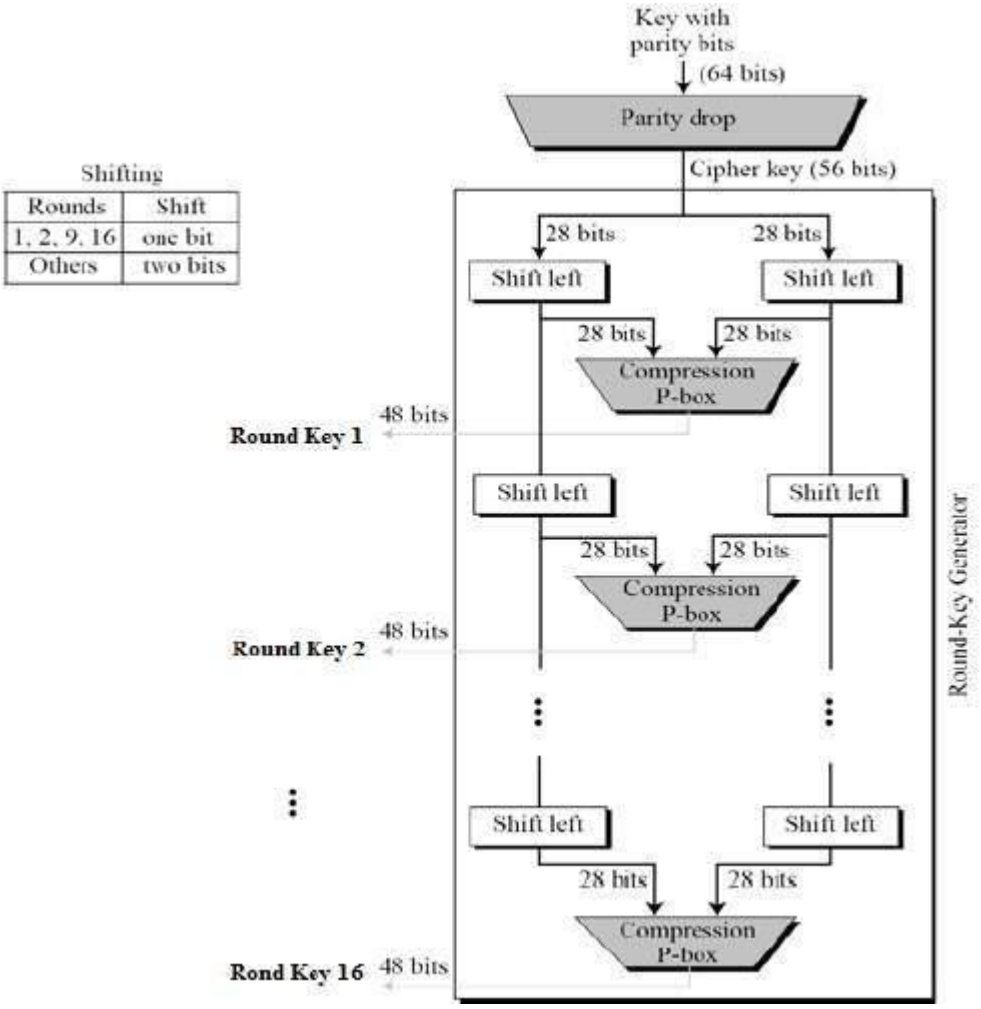
Substitution Boxes. − The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration −

There are a total of eight S-box tables. The output of all eight s-boxes is then combined in to 32 bit section.

Straight Permutation − The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

 **Key Generation**

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration −



Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

DES AnalysisThe DES satisfies both the desired properties of block cipher. These two properties make cipher very strong.

Avalanche effect − A small change in plaintext results in the very great change in the ciphertext.

Completeness − Each bit of ciphertext depends on many bits of plaintext.

During the last few years, cryptanalysis have found some weaknesses in DES when key selected are weak keys. These keys shall be avoided.

DES has proved to be a very well designed block cipher. There have been no significant cryptanalytic attacks on DES other than exhaustive key search.

Diffie Hellman Key Exchange-

- This algorithm is used to exchange the secret key between the sender and the receiver.
- This algorithm facilitates the exchange of secret key without actually transmitting it.

Diffie Hellman Key Exchange Algorithm-

Let-

- Private key of the sender = X_s
- Public key of the sender = Y_s
- Private key of the receiver = X_r
- Public key of the receiver = Y_r

Using Diffie Hellman Algorithm, the key is exchanged in the following steps-

Step-01:

One of the parties choose two numbers 'a' and 'n' and exchange with the other party.

- 'a' is the primitive root of prime number 'n'.
- After this exchange, both the parties know the value of 'a' and 'n'.

Step-02:

Both the parties already know their own private key.

- Both the parties calculate the value of their public key and exchange with each other.

Sender calculate its public key as-

$$\mathbf{Y_s = a^{X_s} \bmod n}$$

Receiver calculate its public key as-

$$\mathbf{Y_r = a^{X_r} \bmod n}$$

Step-03:

Both the parties receive public key of each other.

- Now, both the parties calculate the value of secret key.

Sender calculates secret key as-

Secret key = $(Y_r)^{X_s}$ mod n

Receiver calculates secret key as-

Secret key = $(Y_s)^{X_r}$ mod n

DES Diffie hellman

FAQ:

1. What are possible attacks on DES
2. What are attacks on Diffie Hellman key exchange algorithms
3. What are the strength of DES

1. Possible Attacks on DES
Brute Force Attack: DES uses a 56-bit key, and there are only 2^56 possible keys. With modern computing power, it's possible to try all possible keys (brute force) in a reasonable amount of time to decrypt the data.

Differential Cryptanalysis: This attack analyzes how differences in the plaintext can affect the ciphertext.

Linear Cryptanalysis: This attack attempts to find a linear approximation between the plaintext, ciphertext

Meet-in-the-Middle Attack: This attack works when two different encryption keys are used in a two-step process.

3. Strength of DES
Key Size: DES uses a 56-bit key, and this small key size makes it vulnerable to brute-force attacks. With today's computational power, attackers can break DES encryption in a matter of hours or even minutes.

No Long-Term Security: DES was designed in the 1970s, and its security was considered good at the time. However, now it's easy to crack, and it's considered obsolete for most purposes.

Limited Block Size: DES uses a 64-bit block size, which means that it can encrypt 64 bits of data at a time. This makes it susceptible to "birthday attacks" and also increases the risk of data repetition when encrypting large amounts of data.

2.-->Man-in-the-Middle Attack (MITM): If an attacker intercepts the messages between the two parties, they can change the key exchanges and create a false shared key. The two parties may believe they have a secure connection when the attacker actually has access to the communication.

Logjam Attack: This is a specific attack on Diffie-Hellman using weak primes. If weak primes are used during the key exchange, an attacker can break the security by precomputing certain values, reducing the difficulty of the key exchange.

Small Subgroup Attack: This attack takes advantage of weak values used in the Diffie-Hellman exchange. If the parameters are not chosen carefully (e.g., small prime numbers), the key exchange could be broken.

ASSIGNMENT NO : 4

Aim: Use the snort intrusion detection package to analyze traffic and create a signature to identify problem traffic.

Theory:

Intrusion Detection System Defined as the tools, methods, and resources to help identify, assess, and report unauthorized or unapproved network activity.An IDS detects activity in traffic that may or may not be an intrusion. IDSes can detect and deal with insider attacks, as well as, external attacks, and are often very useful in detecting violations of corporate security policy and other internal threats.

Components of Intrusion Detection System:

An Intrusion Detection system comprises of Management console and sensors. Management console is the management and reporting console. Sensors are agents that monitor hosts or networks on a real time basis. An Intrusion Detection System has a database of attack signatures. The attack signatures are patterns of different types of previously detected attacks. If the sensors detect any malicious activity, it matches the malicious packet against the attack signature database. In case it finds a match, the sensor reports the malicious activity to the management console. The sensor can take different actions based on how thy are configured. For example, the sensor can reset the TCP connection by sending a TCP FIN, modify the access control list on the gateway router or the firewall or send an email notification to the administrator for appropriate action

Types of Intrusion-Detection systems

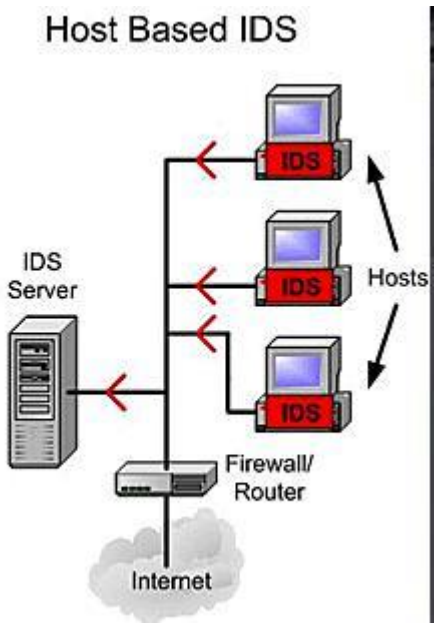
There are broadly two types of Intrusion Detection systems. These are host based Intrusion Detection System and network based Intrusion Detection System. A Host based Intrusion Detection system has only host based sensors and a network based Intrusion detection system has networkbased sensor

Host based Intrusion Detection System:

A host-based intrusion detection system (HIDS) is an intrusion detection system that monitors and analyzes the internals of a computing system as well as (in some cases) the network packets on its network interfaces

A host-based IDS monitors all or parts of the dynamic behavior and the state of a computer system. Besides such activities as dynamically inspecting network packets targeted at this specific host (optional component with most software solutions commercially available), a HIDS might detect which program accesses what resources and discover that, for example, a word-processor has suddenly and inexplicably started modifying the system password database. Similarly a HIDS might look at the state of a system, its stored information, whether in RAM, in the file system, log files or elsewhere; and check that the contents

of these appear as expected, e.g. have not been changed by intruders.



Monitoring dynamic behavior

Many computer users have encountered tools that monitor dynamic system behaviour in the form of anti-virus (AV) packages. While AV programs often also monitor system state, they do spend a lot of their time looking at who is doing what inside a computer – and whether a given program should or should not have access to particular system resources. The lines become blurred here, as many of the tools overlap in functionality.

Intrusion prevention systems are a type of HIDS software that protects against buffer overflow attacks on

system memory and can enforce security policy.

Monitoring state

The principle operation of a HIDS depends on the fact that successful intruders (hackers) will generally leave a trace of their activities. In fact, such intruders often want to own the computer they have attacked, and will establish their "ownership" by installing software that will grant the intruders future access to carry out whatever activity (keystroke logging, identity theft, spamming, botnet activity, spyware-usage etc.) they envisage.In theory, a computer user has the ability to detect any such modifications, and the HIDS attempts to do just that and reports its findings.

Ideally a HIDS works in conjunction with a NIDS, such that a HIDS finds anything that slips past the NIDS. Commercially available software solutions often do correlate the findings from NIDS and HIDS in order to find out about whether a network intruder has been successful or not at the targeted host. Most successful intruders, on entering a target machine, immediately apply best-practice security techniques to secure the system which they have infiltrated, leaving only their own backdoor open, so that other intruders can not take over their computers.

Technique

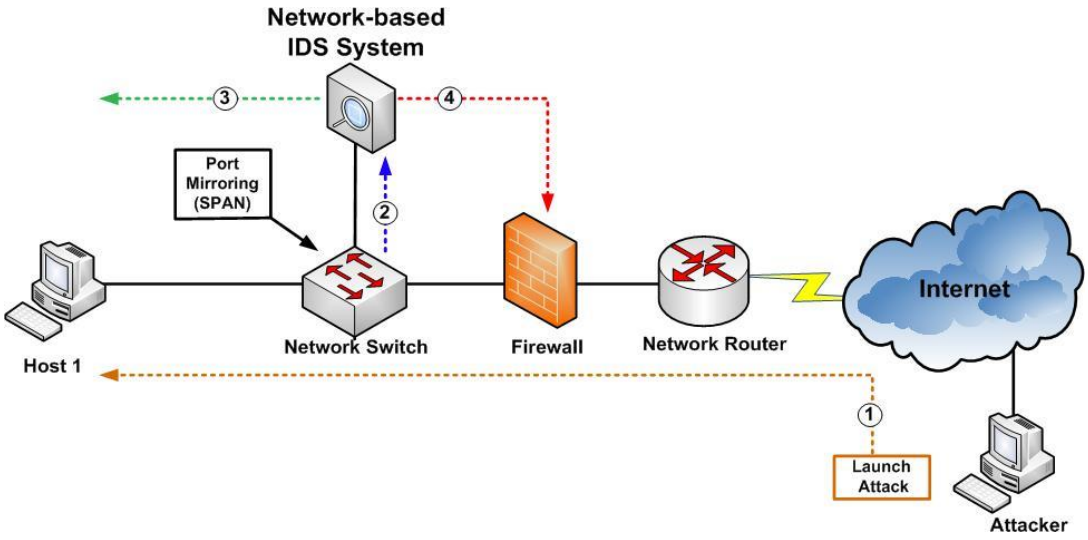
In general a HIDS uses a database (object-database) of system objects it should monitor – usually (but not necessarily) file system objects. A HIDS could also check that appropriate regions of memory have

not been modified – for example, the system call table for Linux, and various vtable structures in Microsoft Windows. During the communication establishment phase and while transferring the data requested by the client, the host's server and the client exchanges a passphrase to verify their identity. The server uses the same passphrase all the time for this purpose. Based upon that an object is created. For each object in question a HIDS will usually remember its attributes (permissions, size, modifications dates) and create a checksum of some kind (an MD5, SHA1 hash or similar) for the contents, if any. This information gets stored in a secure database for later comparison (checksum database). An alternate method to HIDS would be to provide NIDS type functionality at the network interface (NIC) level of an end-point (either server, workstation or other end device). Providing HIDS at the network layer has the advantage of providing more detailed logging of the source (IP address) of the attack and attack details, such as packet data, neither of which a dynamic behavioral monitoring approach could see.

Network Intrusion Detection System: -

identifies intrusions by examining network traffic and monitors multiple hosts. Network Intrusion Detection Systems gain access to network traffic by connecting to a hub, network switch configured for port mirroring, or network tap. An example

of a NIDS is Snort. Host-based Intrusion Detection System: -consists of an agent on a host which identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability/acl databases) and other host activities and state



Hybrid Intrusion Detection System: -

combines one or more approaches. Host agent data is combined with network information to form a comprehensive view of the network. An example of a Hybrid IDS is Prelude.

Passive system v/s reactive system

In a passive system, the IDS sensor detects a potential security breach, logs the information and signals an alert on the console. In a reactive system, which is known as an Intrusion Prevention System (IPS) the IDS responds to the suspicious activity by resetting the connection it believes to be suspicious or by reprogramming the firewall to block network traffic from the suspected malicious source, either autonomously or at the command of an operator. Though they both relate to network security, an IDS dif

fers from a firewall in that a firewall looks outwardly for intrusions in order to stop them from happening. The firewall limits the access between networks in order to prevent intrusion and does

not signal an attack from inside the network. An IDS evaluates a suspected intrusion once it has taken place and signals an alarm. An IDS also watches for attacks that originate from within a system

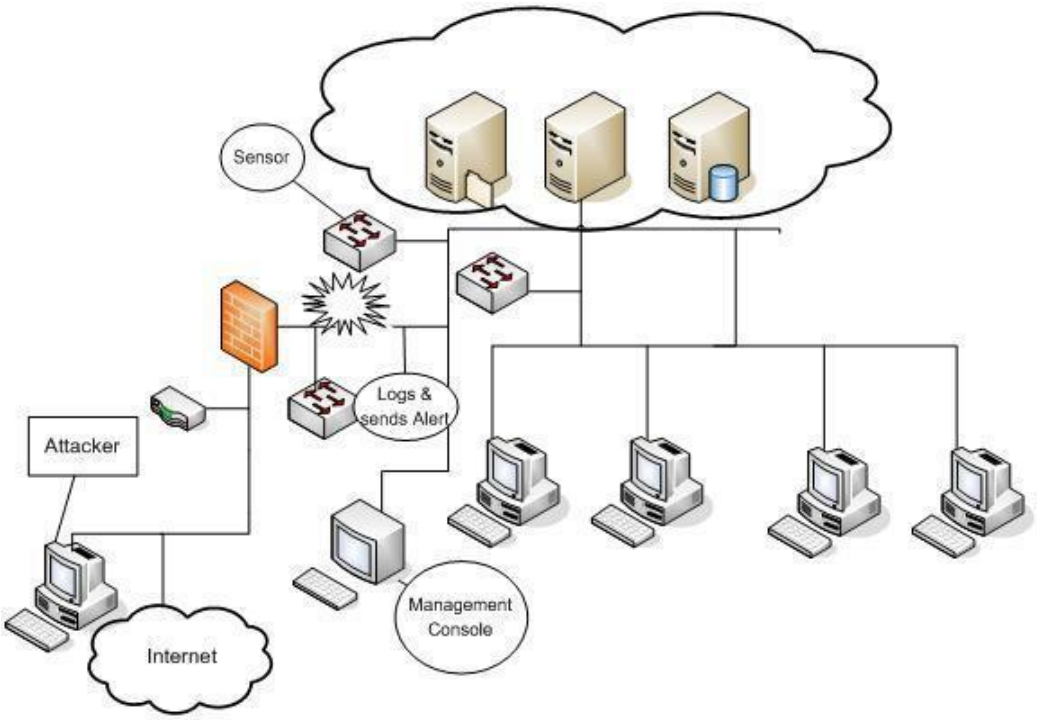


Fig: Passive system

Signature Based Detection v/s Anomaly Based Detection

Signature based detection:-

This detection technique uses specifically known patterns to detect malicious code. These specific patterns are called signatures. Identifying the worms in the network is an example of signature based detection.

Anomaly Detection:-

These techniques are designed to detect abnormal behavior in the system. The normal usage pattern is baselined and alerts are generated when usage deviates from the normal behavior.

Example if a user logs on and off 20 times a day while thenormal behavior is 1-2 times.

Conclusion: An Intrusion Detection System (IDS) is a tool that watches over a computer network or a computer to detect suspicious activities(like hacking attempts, viruses, or unauthorized access).

Thus we have studied Intrusion Detection system and its types.

FAQs:

1.What is Intrusion Detection System?

2.What are the different types of intrusion Detection System?

3.What is a Host Intrusion Detection System?

4.Why is intrusion detection required in today's computing environment?

5. Are there tools to visualize the data from an intrusion detection system?

4.Cyberattacks are more common than ever.
Systems are connected to the internet, so hackers have more ways to attack.
Data breaches are expensive — financially and in terms of trust.

2-->two ways IDS can detect threats:

Signature-based Detection:
Looks for known attack patterns (like antivirus does).

Anomaly-based Detection:
Looks for unusual behavior (even if it's a new, unknown attack).

Network-based IDS (NIDS):
It monitors the entire network's traffic. It's like putting a security guard at the entrance of your building to watch who comes and goes.

Host-based IDS (HIDS):
It monitors activities on a specific computer (host). It watches things like system files and application logs on that machine.

3.-->A Host Intrusion Detection System (HIDS) is an IDS that is installed on an individual computer (host).
It checks things like that: 1) Changes to important system files
Strange activity by users
Suspicious logins

5--> Yes! 1) Kibana (works with Elasticsearch; great for graphs and dashboards) , Splunk (collects, analyzes, and visualizes security events) 3)Grafana (can show IDS data in real-time dashboards)