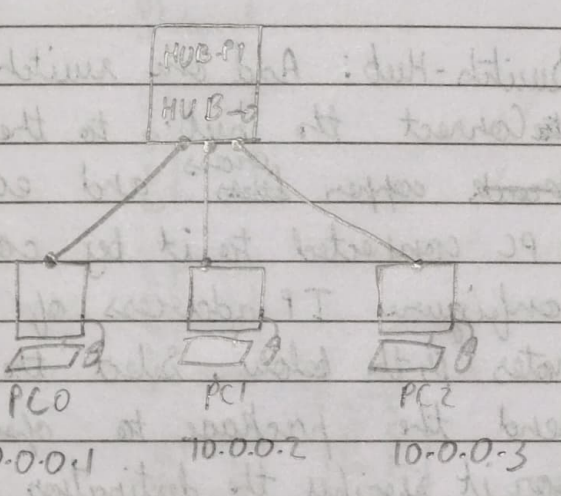
Lab-1

Date __/__/__
Page _____

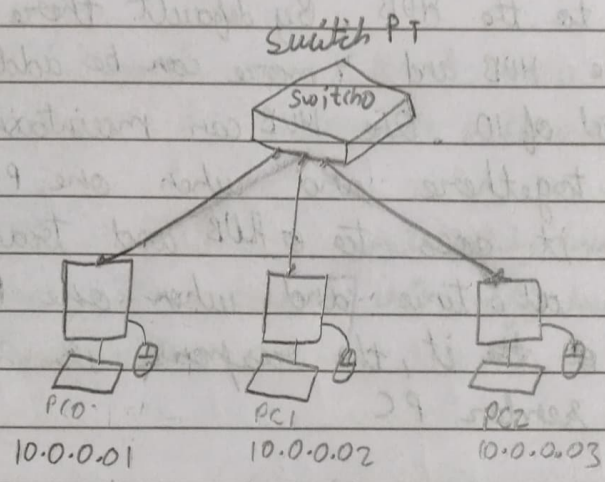
AIM: Creating a Topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices

Topology:

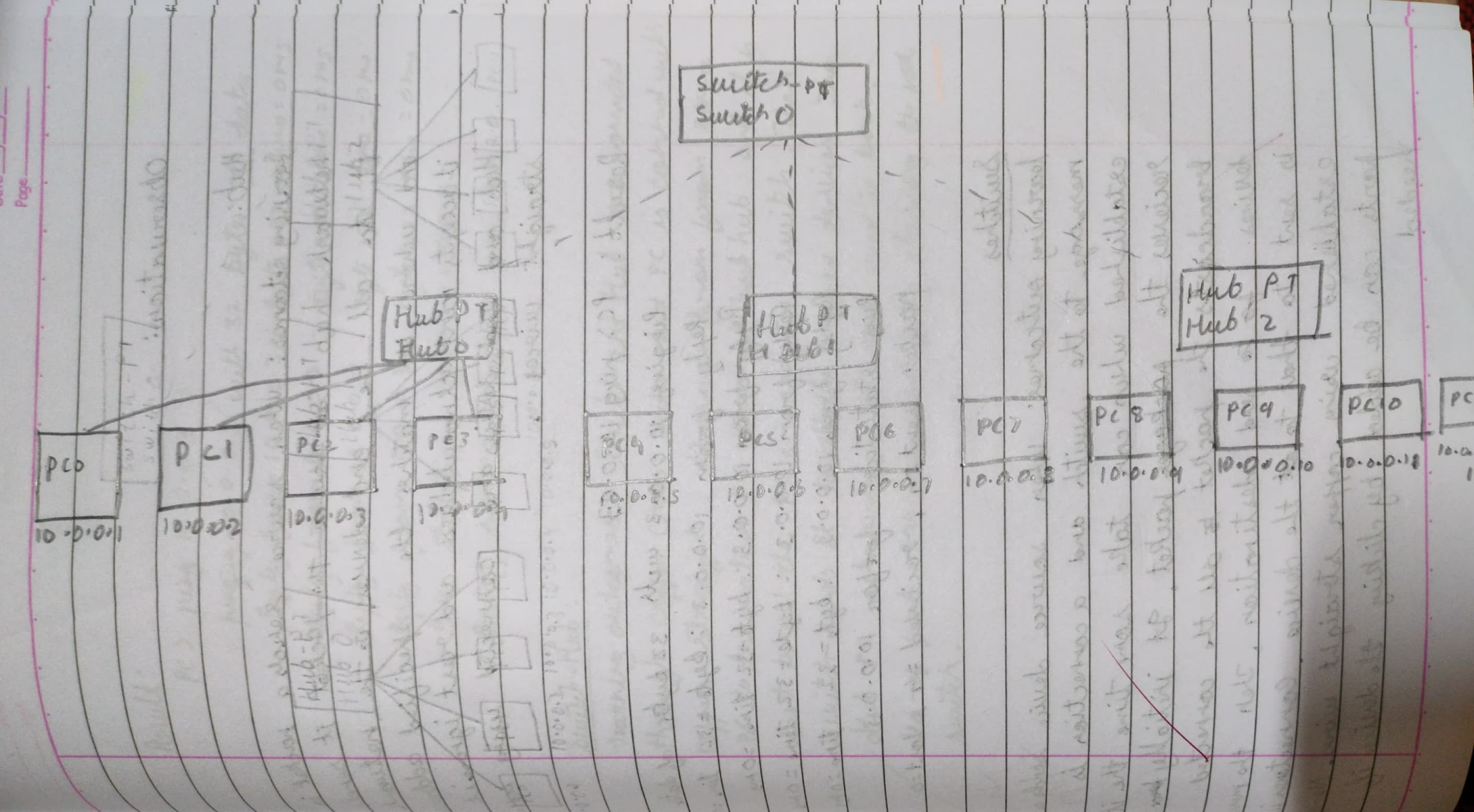
HUB: All the PC's are connected with copper wire to the HUB. By default there are 6 ports in the HUB and 4 more can be added making a total of 10. One HUB can maintain upto 10 PC's togethere and when one PC sends a PDU it goes to HUB and travel to other PC's at a time and when one PC acknowledges it, the response is sent back to the sender PC



Switch: It can also connect with multiple PC's and they are also connected with copper wire there connection here is between ethernet and fastethernet the PDU is sent from one PC to another it acknowledges it and then send it back. The the switch sends PDU to each of the system.



Switch-Hub: Add one switch, 3 hub's and 12 PC's. Connect the hub's to the switch with cross copper and each hub will have 4 PC connected to it by copper wires Configure IP address of each PC and add notes in the below. Select the PC you want to send the package to open the command prompt for it specify the destination PC and its IP address and Add a simple PDU by selecting the 2 PC's you want to send.


Switch-PT
Switch 0
Hub PT
Hub 0
Hub PT
Hub 1
Hub PT
Hub 2
PC0
10.0.0.1
PC1
10.0.0.2
PC2
10.0.0.3
PC3
PC4
10.0.0.5
PC5
10.0.0.6
PC6
10.0.0.7
PC7
10.0.0.8
PC8
10.0.0.9
PC9
10.0.0.10
PC10
10.0.0.18

Observation:

Hub:

Learning outcomes: when source sends a packet in network hub ~~so source the packet~~ it sends it to all the end devices in the network and where it matches the specified address it accepts and acknowledges and rest ignore it. Hub and end devices are connected using copper straight wires.

Result: PC> ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data
Reply from 10.0.0.3 : byte = 32 time = 0ms
Reply from 10.0.0.3 : byte = 32 time = 0ms
Reply from 10.0.0.3 : byte = 32 time = 0ms
Reply from 10.0.0.3 : byte = 32 time = 0ms
Ping statistics ~~for~~ for 10.0.0.3
packet sent = 4, recieved = 4, lost = 0

Switches

Learning outcomes: when source device sends a message to the switch and a connection is established which can take some time the it recieves the ~~package~~ packet. It initially ~~broa~~ broadcasts the packet to all the connected devices ~~to~~ to find destination. Then the message is sent to that to ~~it~~ the device. Connectivity is established using copper straight wires. No. of ports can be added by clicking the devices if needed

Result:

PC > ping 10.0.0.3

pinging 10.0.0.3 with 32 bytes of data

Reply from 10.0.0.3 : bytes = 32 time = 0ms

Reply from 10.0.0.3 : byte = 32 time = 0ms

Reply from 10.0.0.3 : byte = 32 time = 0ms

Reply from 10.0.0.3 : byte = 32 time = 0ms

Ping statistics for 10.0.0.3

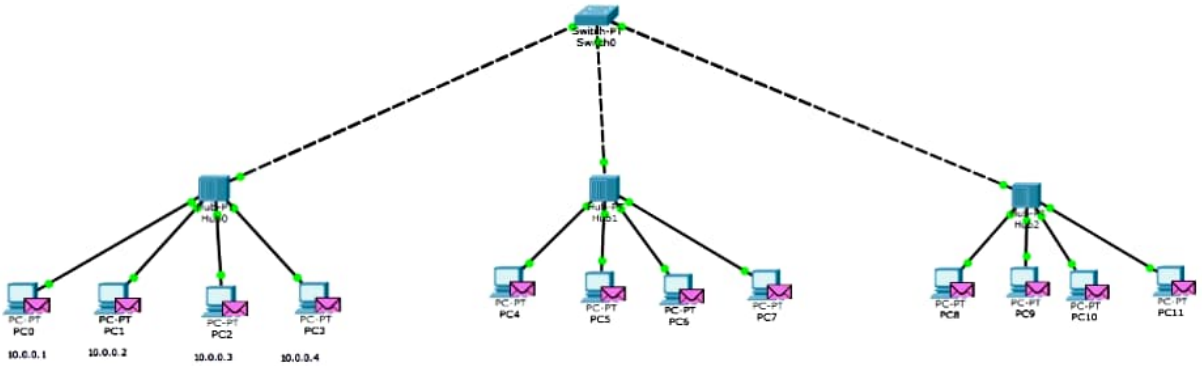
Packet sent = 4 received = 4, lost = 0

Switch-Hub :

Learning outcomes: Switch and Hub are connected through copper cross wire but PC is connected with copper straight. Message from ~~from~~ source PC to destination PC through ~~hub~~ hub and it is sent to all the PC's and the switch. Then switch sends it to all the connected devices. The dist PC acknowledges it and send it back. No of ports can be added ~~of~~ by clicking the ~~pods~~ switch.

17/11/22

SWITCH-HUB
1BM20CS064



Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type	Info
	4.105	Hub0	PC3	STP	
	4.105	Hub1	PC4	STP	
	4.105	Hub1	PC5	STP	
	4.105	Hub1	PC6	STP	
	4.105	Hub1	PC7	STP	
	4.105	Hub2	PC8	STP	
	4.105	Hub2	PC9	STP	
	4.105	Hub2	PC10	STP	
	4.105	Hub2	PC11	STP	

Reset Simulation | Constant Delay | Captured to: 4.105 s

Play Controls

Back | Auto Capture / Play | Capture / Forward

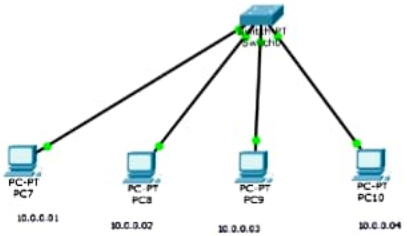
Event List Filters - Visible Events

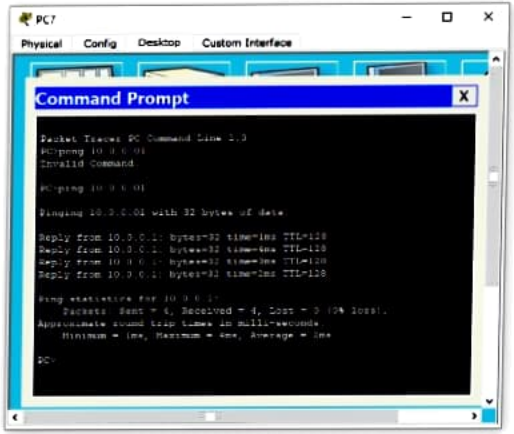
ACL Filter, ARP, BGP, CDP, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Edit Filters | Show All/None

10.0.0.2

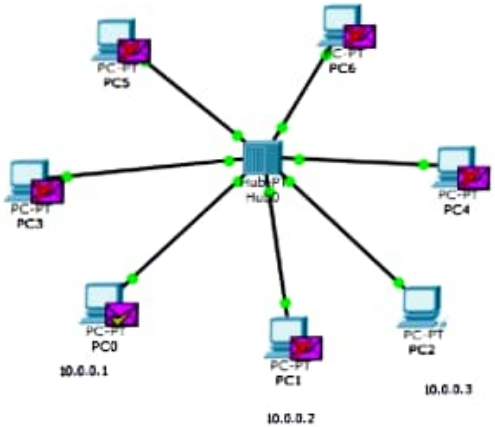
SWITCH REALTIME
1BM20CS064





HUB

1BM20CS064



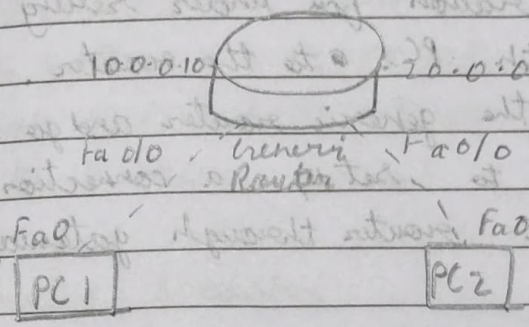
Lab - 2

Date 24/11/22
Page

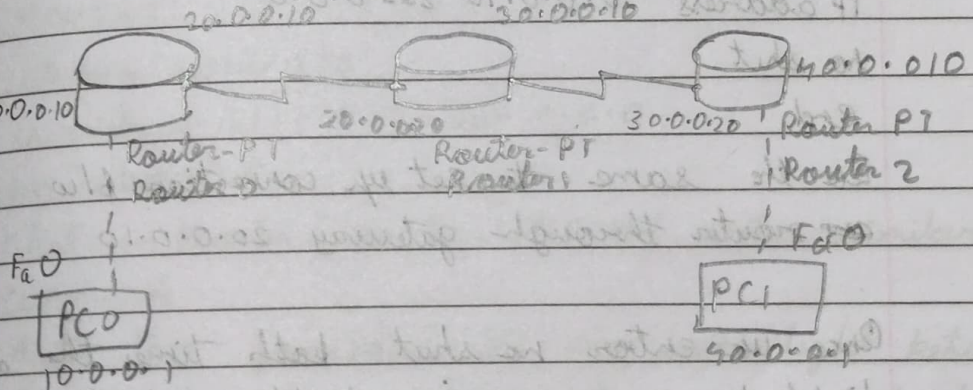
AIM: Configuring IP addresses to router in packet tracer to explore the following messages Ping responses : destination unreachable : Request timed out : reply

Topology

- Using single router : two PC's



- Using 3 router 2 PC's



Procedure

- Using single router, 2 PC's
1. Use generic router and add generic PC's in the workspace
2. Use copper cross over wires to connect the router and PC's.
3. Configure IP address for each PC and in configuration tab under setting set gateways for both PCs to the router.
4. Click on the generic router and go to the CLI. Enter commands to set up a connection between PC1 and generic router through gateway 10.0.0.10

```
No
#enable
#config
interface fastethernet 0/0
ip address 10.0.0.10 255.0.0.0
no shut
end
```

Do the same to set up connection b/w PC2 and router through gateway 20.0.0.10

Once we enter no shut both times the amber light turns green immediatly indicating that the 2 devices are ready for use.

- Simulation mode: Add a simple PDU by selecting the PCs and click on auto capture
- Real time mode: Select the PC you want to send the packet from which is PC0 and open command prompt from desktop tab. Specify the destination tab address. A response is sent from destination PC to source PC

Date __/__/__
Page _____

- Using 3 routers 2 PCs

① Place 3 gen routers and 2 gen PCs in the workspace
② Place notes for all devices and mention their IP address.
③ Use copper cross wires to connect PC and router.
④ Use serial DCE to connect routers.
⑤ Set IP address and subnet mask for PCs.
⑥ Set the gateway on the IP address of the next router
⑦ IP address of PC and its gateway address should belong to the same network.

For connecting 2 routers
Click on router 0 go to CLI and enter the commands
No
enable
#config t
interface serial 2/0
ip address 20.0.0.10 255.0.0.0
no shut
click on router 1 go to CLI and enter similar commands.
After this procedure, the red light between the two routers will now turn green indicating that they are ready for communication.

P.T.O

Date __/__/__
Page

For connecting 2 devices

Give IP address of PC0 is already configured, go to router

no

enable

config t

interface fast ethernet 0/0

ip address 10.0.0.10 255.0.0.0

no shut

Red light turns green indicating the router is ready for communication

Teaching router 0 of network 30

no

enable

config t

interface serial 2/0

ip route 30.0.0.0 255.0.0.0 20.0.0.20

exit

show ip route

Repeat the same for network 40 as well as router 1 and router 2.

Observation:

1 router

When PC0 pings PC1 for the first time we get the request time out for 1st packet. If we ping PC1 from PC0 again we get all 4 packets without any loss. Reverse pinging will also not lead to any loss.

3 routers

Before training the routers we get the results as destination is unreachable. After training the routers we get clear statistics on the result

Results:

Using single router

Ping 20.0.0.1

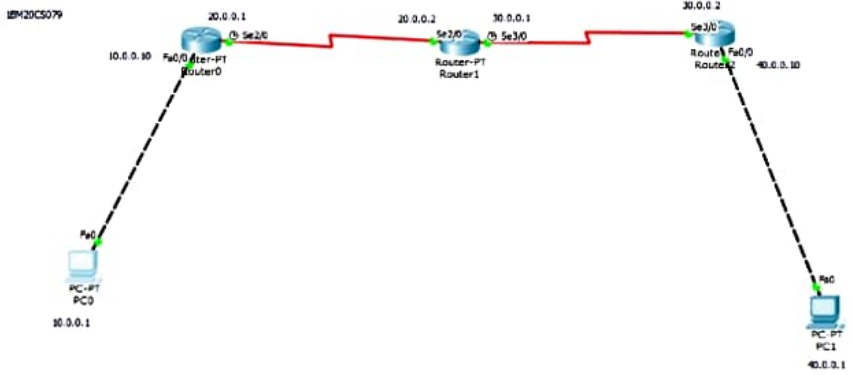
Pinging 20.0.0.1 with 32 bytes of data
Request timed out
Reply from 20.0.0.1 bytes = 32 time < 1ms
Reply from 20.0.0.1 bytes = 32 time < 1ms
Reply from 20.0.0.1 bytes = 32 time < 1ms
Reply from 20.0.0.1 bytes = 32 time < 1ms
Ping statistics from 20.0.0.1
packets = 4 recieved = 4 lost = 0

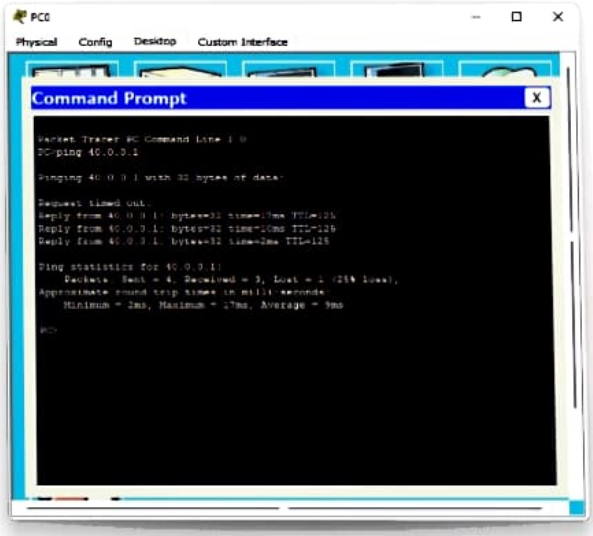
Using 3 routers, 2 PCs

Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data
Reply from 10.0.0.16 bytes = 32 time = 0ms
Reply from 10.0.0.10 bytes = 32 time = 0ms
Reply from 10.0.0.10 bytes = 32 time = 0ms
Reply from 10.0.0.10 bytes = 32 time = 0ms
Ping statistics for 40.0.0.1
packet sent = 4 recieved = 4 lost = 4

11/12/22


LBM20CS079
20.0.0.1
Se2/0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
30.0.0.2
Se3/0
10.0.0.10
Fa0/0
Router-PT
Router0
Router-PT
Router1
Router-PT
Router2
Fa0/0
40.0.0.10
Fa0
PC-PT
PC0
10.0.0.1
Fa0
PC-PT
PC1
40.0.0.1

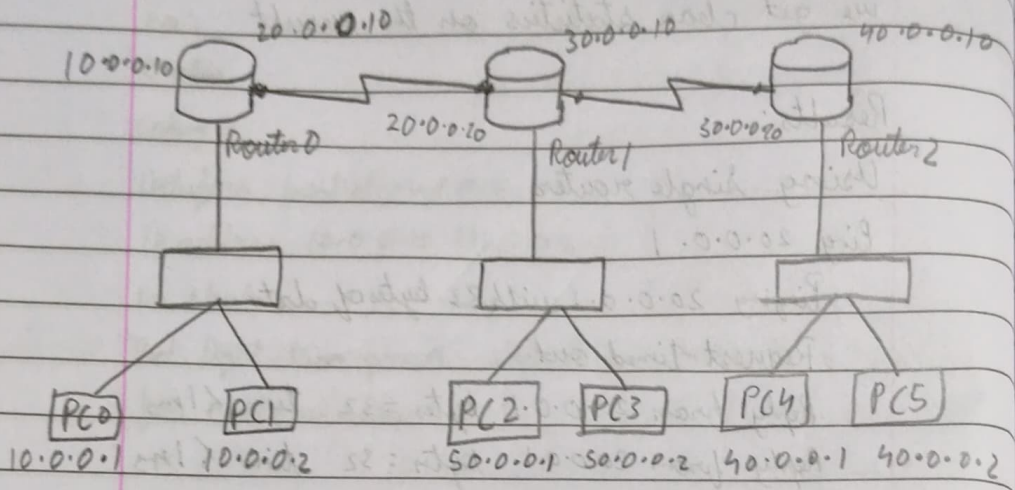

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 2ms, Maximum = 17ms, Average = 9ms
PC>

Lab-3

Date 29/11/22
Page

AIM: Configuring default route to the router

Topology



Procedure:

① Use 3 gen routers and 6 gen PCs in the workspace along with 3 switches.
② Place a nat for each device and specify the IP addresses
③ Use copper straight wires to connect router and switch
④ Use copper straight wires to connect switch and PC
⑤ Click on a PC to set attributes for a PC and each PC has 3 attributes subnet mask, IP addr and gateway. This needs to be done for all 6 PCs
⑥ For router 1 the config is done in the CLI. The IP address and subnet mask is set. Router2 is a default router for router 1 and this is done by command ip route 0.0.0.0 0.0.0.0 40.0.0.2
⑦ Router 2 IP address and subnet masks are set for all 3 interfaces. It has static routing done by commands

(5) Router 3 is config in both the interface with IP address and subnet mask. The default router for router 3 is router 2

(6) Ping command is executed from 10.0.0.1 to 20.0.0.1 and from 10.0.0.1 to 30.0.0.2

Observation

- 1 router can have 2 default router
- Default router for router 3 is also middle router
- Middle router gets no default router because if one router is made default then there ~~is a~~ are chances of packets going to switch are sent to the router.

Result

Ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data

Request timed out

Reply from 30.0.0.2 bytes = 32, time = 4ms TTL = 125

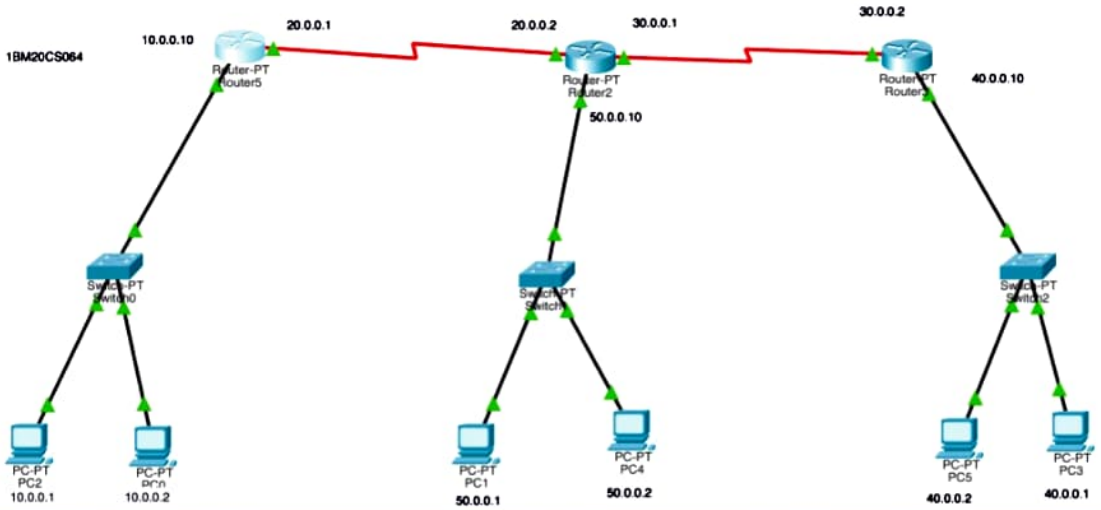
Reply from 30.0.0.2 bytes = 32, time = 4ms TTL = 125

Reply from 30.0.0.2 bytes = 32, time = 4ms TTL = 125

Cisco Packet Tracer File Edit Options View Tools Extensions Window Help Fri 9 Dec 18:32:49

Cisco Packet Tracer - /Users/joeanimegod/Downloads/3 routers default.pkt

Logical Physical x: 517, y: 379 Root 00:51:00



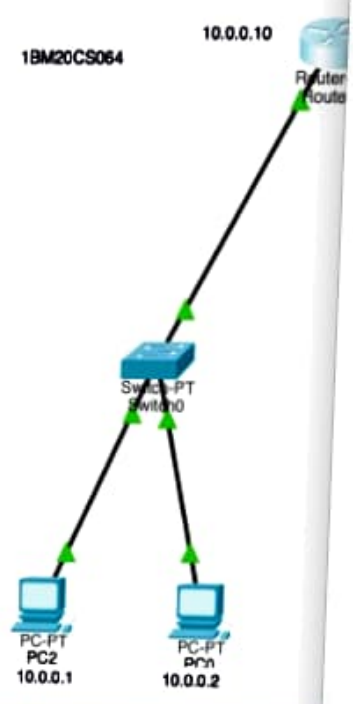
Time: 00:01:39 Realtime Simulation

4331 4321 1941 2901 2911 819IOX 819HGW

(Select a Device to Drag and Drop to the Workspace)

Scenario 0 New Delete Toggle PDU List Window

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	--	PC2	PC1	IC...		0.000	N	0	(...	(delete)
	--	PC0	PC4	IC...		0.006	N	1	(...	(delete)




```
Readonly ROMMON initialized

Self decompressing the image :
######################################## [OK]

                Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco Internetwork Operating System Software
IOS (tm) PT1000 Software (PT1000-I-M), Version 12.2(28), RELEASE SOFTWARE (fc5)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2005 by cisco Systems, Inc.
Compiled Wed 27-Apr-04 19:01 by miwang

PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory.

Processor board ID FTX0123 (0123)
PT2005 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)

Press RETURN to get started!


%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
```

Copy Paste

Top

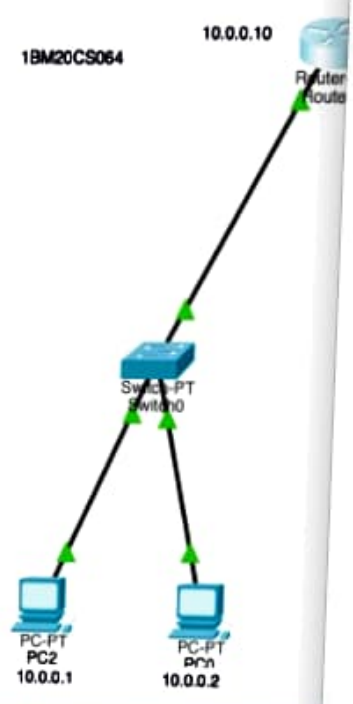
Time: 00:01:26

4331

(Select a Device to Drag and Drop to the Workspace)

New Delete

Toggle PDU List Window




```
Readonly ROMMON initialized

Self decompressing the image :
######################################## [OK]

                Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco Internetwork Operating System Software
IOS (tm) PT1000 Software (PT1000-I-M), Version 12.2(28), RELEASE SOFTWARE (fc5)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2005 by cisco Systems, Inc.
Compiled Wed 27-Apr-04 19:01 by miwang

PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory.

Processor board ID FTX0123 (0123)
PT2005 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)

Press RETURN to get started!


%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
```

Copy Paste

Top

Time: 00:01:26

4331

(Select a Device to Drag and Drop to the Workspace)

New Delete

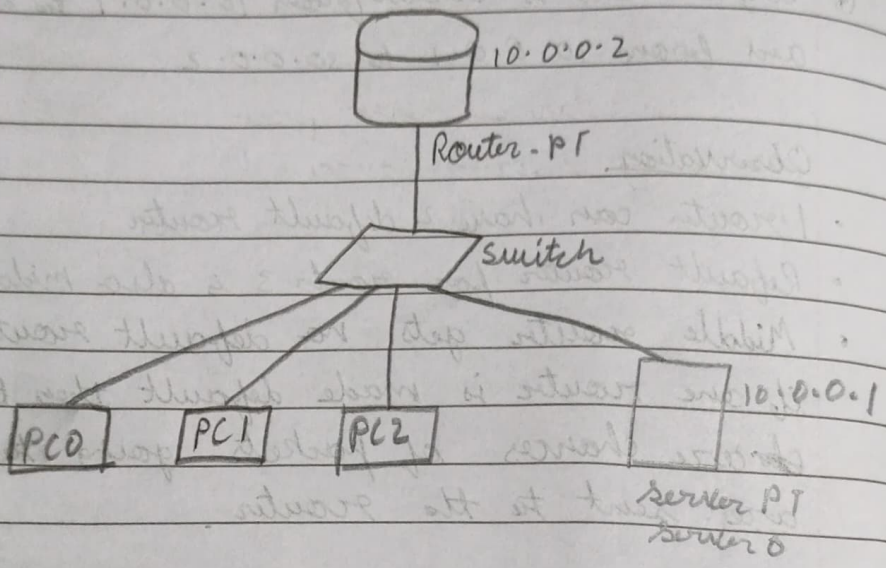
Toggle PDU List Window

Lab-4

Date 01/12/22
Page

Aim: Configuring DHCP within a LAN in a packet tracer

Topology



Procedure

① Use one gen router one gen switch one gen server and 3 gen PCs in the workspace
② Use copper straight wire to connect PCs and server.
③ Use fibre to connect switch and router.
④ Configure the server by adding IP address, subnet mask and gateway.
⑤ Config the router by setting IP address, subnet masks and executing this command in CLI
ip route 10.0.0.2 255.0.0.0
⑥ Click on server and go to services, Select DHCP ~~network~~ Switch it on and add the gateway servers, ip address ~~and~~ and subnet mask.

Desktop tab.

⑦ Click on the 1st PC and go to IP config. Select DHCP
Repeat this for all.

⑧ Ping command is executed from 10.0.0.3 to 10.0.0.5

Observation

Pool of ip addresses exist from which IP addresses can be dynamically allocated.
This is called Dynamic Host Configuration Protocol (DHCP).

Result

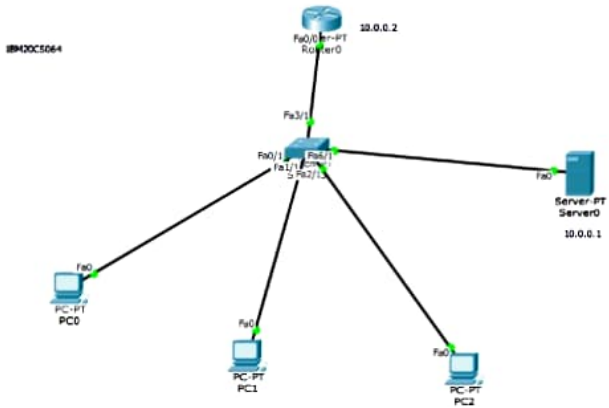
ping 10.0.0.5

Pinging 10.0.0.5 with 32 bytes of data

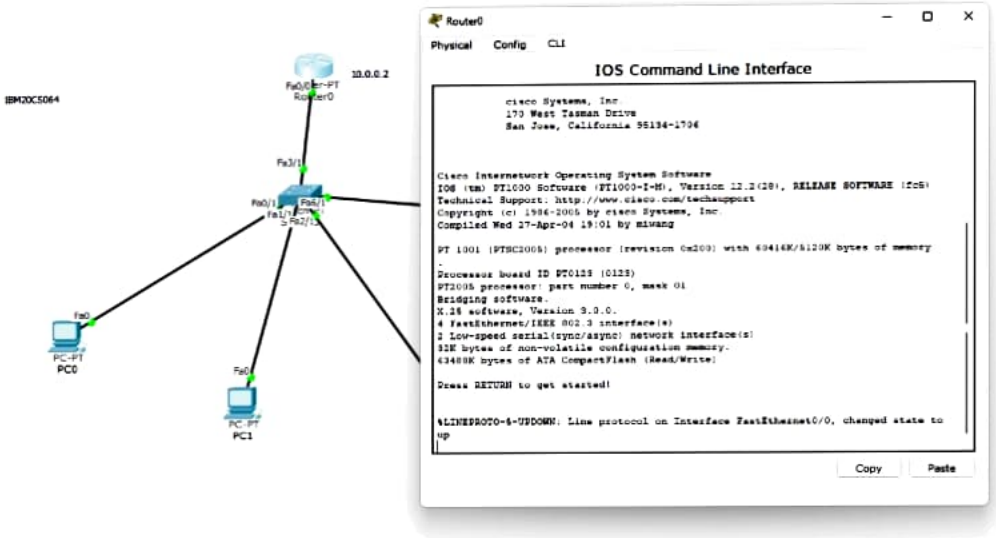
Reply from 10.0.0.5 bytes = 32 time = 2ms TTL = 125
Reply from 10.0.0.5 bytes = 32 time = 2ms TTL = 125
Reply from 10.0.0.5 bytes = 32 time = 2ms TTL = 125
Reply from 10.0.0.5 bytes = 32 time = 2ms TTL = 125

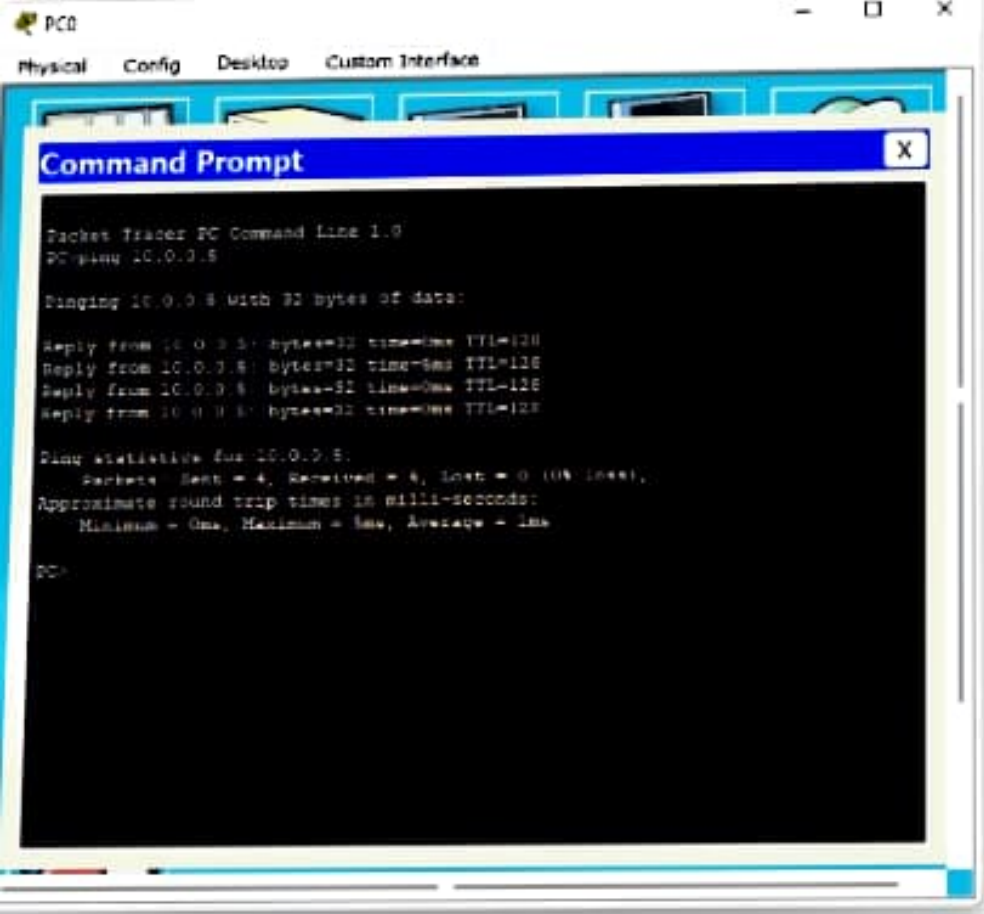
Ping statistics for 10.0.0.5
Packets sent = 4, recieved = 4, lost = 0

8/12/22


10.0.0.2
Fa0/0
Fa3/1
Fa0/1
Fa1/1
Fa2/1
Fa0
Server-PT
Server0
10.0.0.1
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
Fa0
PC-PT
PC2

Logical


IBM20C5064
10.0.0.2
Router0
Fa3/1
Fa0/1
Fa6/1
Fa1/1
Fa2/1
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
Router0
Physical
Config
CLI
IOS Command Line Interface
cisco Systems, Inc
170 West Tasman Drive
San Jose, California 95134-1706
Cisco Internetwork Operating System Software
IOS (tm) PT1000 Software (PT1000-I-M), Version 12.2(28), RELEASE SOFTWARE (fc5)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2005 by cisco Systems, Inc.
Compiled Wed 27-Apr-04 19:01 by miwang
PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory.
Processor board ID PT0123 (0123)
PT2005 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)
Press RETURN to get started!
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
Copy
Paste

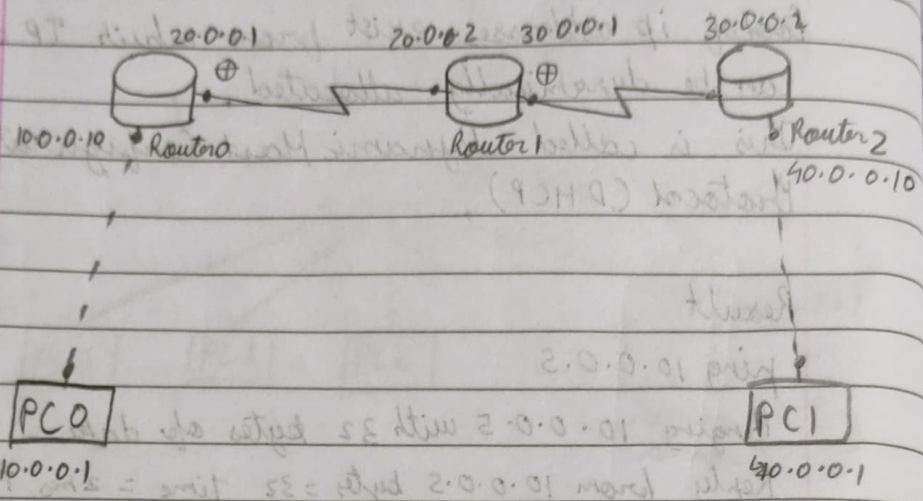

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.5
Pinging 10.0.0.5 with 32 bytes of data:
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Ping statistics for 10.0.0.5:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
PC>

Lab-5

Date 8/12/22
Page

Aim: Configuring RIP Routing Protocol in Routers

Topology:



Procedure:-

- Use 3 generic routers, 2 generic PC and place notes to indicate respective IP addresses
- Use serial DCE cable to connect routers and use copper cross cable to connect PC with router1 and router3
- Set IP addresses, gateway and subnet mask as 10.0.0.10, 255.0.0.0 for PC0 and set 40.0.0.1, 40.0.0.10, 255.0.0.0 for PC1
- interface PC0 and router1
  → interface fastethernet 0/0
  → ip address 20.0.0.10 255.0.0.0
  → no shut

- for interfacing serial 2/0 of router 1 use.
  → interface serial 2/0
  → ip address 20.0.0.1 255.0.0.0
  → encapsulation ppp
  → clock rate 64000
  → no shut
- Use above commands for interfacing router which has clock symbol in cable near to it and for other interfaces of routers use same above command except "clock rate 64000"
- once all the lights are turned green follow the commands below each router
  → router rip
  → network 10.0.0.0
  → network 20.0.0.0
  → exit
- ~~respo~~ Repeat the same command for router 2 and router 3

Observation:
Use RIP routing becomes easy when large number of routers are present

Result:
Pinging 10.0.0.1 with 32 bytes of data

reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32

ping statistics for 10.0.0.1
packets: sent = 4, recieved = 4, lost = 0

Time: 00:45:49 | Power Cycle Devices | Fast Forward Time

Connections

Scenario 0

New | Delete

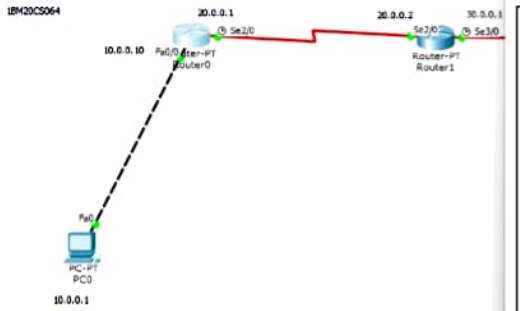
Toggle PDU List Window

Fire | Last Status | Source | Destination

Cisco Packet Tracer Student - C:\Users\BMSCE\Downloads\RIP.pkt

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport



Router0

Physical Config CLI

IOS Command Line Interface

```
Cisco Internetwork Operating System Software
IOS (tm) PT1000 Software (PT1000-I-M), Version 12.2(28), RELEASE SOFTWARE (fc5)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2006 by cisco Systems, Inc.
Compiled Wed 27-Apr-04 19:01 by mnwang

PT 1000 (PTSC2906) processor (revision 0x200) with 4041K/512K bytes of memory

Processor board ID FT0123 (0123)
PT2006 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)

Press RETURN to get started!


%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to
up

%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
```

Copy Paste

Time: 00:26:38 | Power Cycle Devices | Fast Forward Time | Realtime

Routers

(Select a Device to Drag and Drop to the Workspace)

Scenario 0 | New | Delete | Toggle PDU List Window

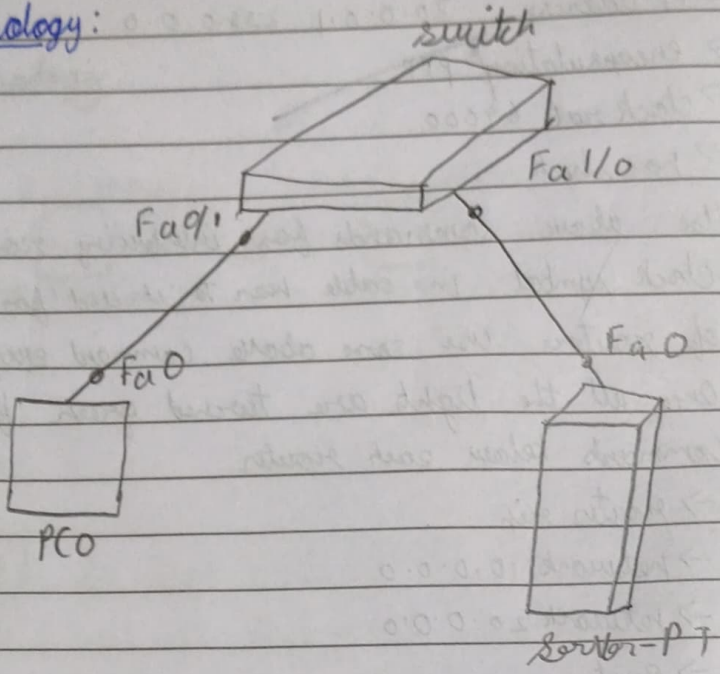
Fire | Last Status | Source | Destination | Type | Color | Time(sec) | Periodic | Num | Edit | Delete

LNG INTL 17:48 08-12-2022

Lab-6

Aim: Demonstration of Web servers and DNS

Topology:



Procedure:

→ Place an PC, Server and switch and then set the IP addresses of PC and server as 10.0.0.1 and 10.0.0.2.
→ Open web browser from dekstop tab of the PC and type "http://10.0.0.2"
→ It will display a default page
→ Open server and in services enable HTTP and change or add the contents in the HTTP.
→ After save refresh the browser of PC0 to the updated changes.

Activate DNS:-

→ enable DNS on the server to activate it
→ Enter the name and the IP address needed to be mapped
→ Click on add to add the new mapping.
→ now give the name in the web browser you saved to check if its working.

Custom page:-

→ Create a new page test.html and save it in http services
→ change hyperlink in index.html to link the the created file.
→ check the output in the webbrowser of the PC0 by clicking on hyper link

Observations:-

→ We can view the web page when we type 'csebmsce' in browser because 10.0.0.2 address is mapped to the name 'csebmsce'.
Mapping is required because its difficult for users to remember IP's. Hence it is mapped.

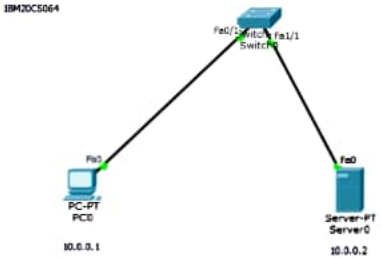
Result:

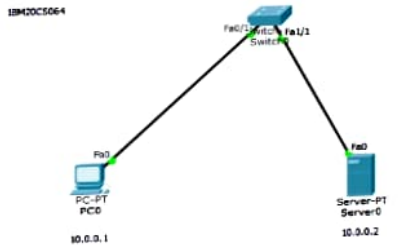
Web Browser

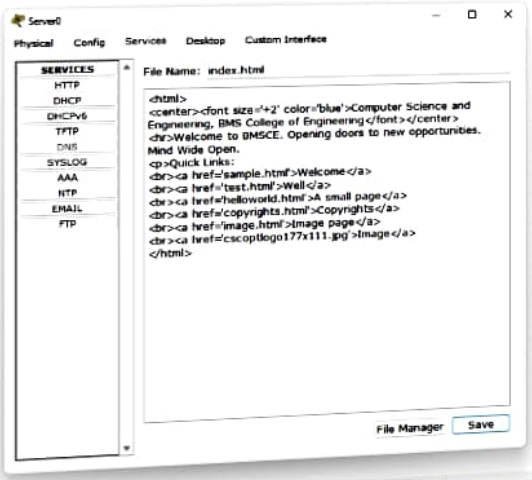
<URL> http://cn/resume.html

Name	Jyotiraditya
age	20
College	BMSCE
Experience	2 years

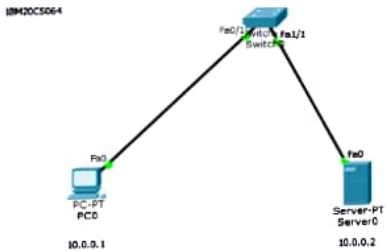
29/12/22

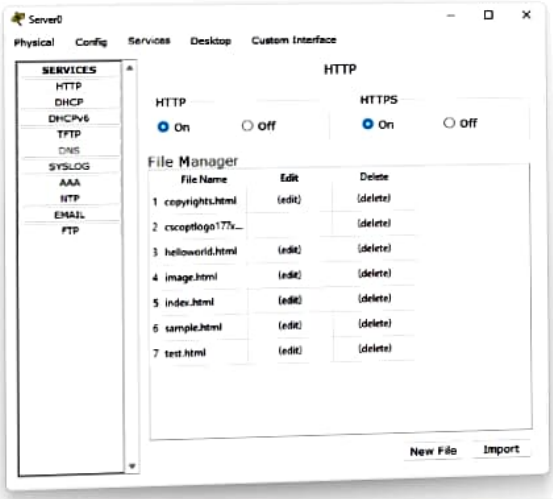

IBM20C5064
Fa0/1
Switch0
fa1/1
Fa0
PC-PT
PC0
10.0.0.1
fa0
Server-PT
Server0
10.0.0.2



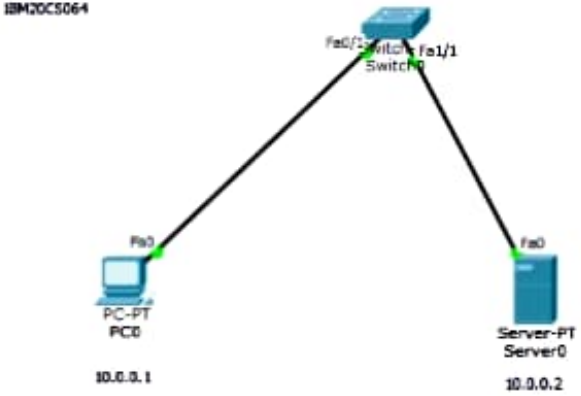


Source Destination Type Color Time(sec) Periodic Num Edit Delete


IBM20C5064
Fa0/1
Switch
Fa1/1
Fa0
PC-PT
PC0
10.0.0.1
Fa0
Server-PT
Server0
10.0.0.2


Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
HTTP
HTTP
On
Off
HTTPS
On
Off
File Manager
File Name
Edit
Delete
1 copyrights.html (edit) (delete)
2 cscoptlogo177x... (delete)
3 helloworld.html (edit) (delete)
4 image.html (edit) (delete)
5 index.html (edit) (delete)
6 sample.html (edit) (delete)
7 test.html (edit) (delete)
New File
Import

IBM20CS064


Fa0/1
Switch
Fa1/1
Switch0
Fa0
PC-PT
PC0
10.0.0.1
Fa0
Server-PT
Server0
10.0.0.2

Lab-7

AIM: Write a program for error detection for CRC

```
#include <stdio.h>
#include <string.h>
#define N strlen(gen_poly)
char data[28];
char check_value[28];
char gen_poly[10];
int data_length, i, j;
void XOR(){
  for(j=1; j<N; j++)
  check_value[j] = ((check_value[j] == gen_poly[j])?
                    '0':'1');
}
void receiver(){
  printf("Enter the recieved data: ");
  scanf("%s", data);
  printf("\n----------\n");
  printf("Data recieved: %s", data);
  crc();
  for(i=0; (i<N-1) && (check_value[i]!='1'); i++);
  if(i<N-1)
    printf("\nError Detected\n\n");
  else
    printf("\nNo Error Detected\n\n");
}
```

```
void crc(){
    for(i=0; i<N; i++)
        check_value[i]=data[i];
    do{
        if(check value[0]=='1')
            xor();
        for(j=0; j<N; j++)
            check_value[j]=check_value[j+1];
        check_value[j]=data[i++];
    }while(i<=data_length+N-1);
}
int main()
{
    printf("\nEnter data to be transmitted:");
    scanf("%s", data);
    printf("\n Enter the Crcn polynomial:");
    scanf("%s", gen_poly);
    data_length=strlen(data);
    for(i=data_length; i<data_length+N-1; i++)
        data[i]='0';
    printf("\n---------------");
    printf("\n Data padded with n-1 zeros: %s", data);
    printf("\n---------------");
    crc();
    printf("\n CRC or check value is: %s", check_value);
    for(i=data_length; i<data_length+N-1; i++)
        data[i]=check_value[i-data_length];
    printf("\n---------------");
    printf("\nfinal data to be sent: %s", data);
    printf("\n---------------\n");
    receiver();
}
return 0;
```

Output

Enter the data : 1000100000010000 1

Enter poly : 101101

Data padded with n-1 : 1000100000010000100000

CRC or check value : 010011

Final data sent : 1000100000010000101011

Enter the received data : 1000100000010000101011

Data received : 1000100000010000101011

No error Detected

```
Enter data to be transmitted: 1000100000100001

 Enter the Generating polynomial: 1011101

-----------------------------------
 Data padded with n-1 zeros : 10001000000100001000000
-----------------------------------
CRC or Check value is : 010011
-----------------------------------
 Final data to be sent : 10001000000100001010011
-----------------------------------
Enter the received data: 10001000000100001010011

-----------------------------
Data received: 10001000000100001010011
No error detected
```

Leaky bucket

```
#include<iostream>
using namespace std;
int main()
{ cout<< "Enter bucket size" <<endl
  int bucketsize;
  int filled =0;
  int outputrate;
  int inputpacket;
  int choice;
  cin>> bucketsize;
  cout << "Enter output rate"<<endl;
  cin >> outputrate;
do {
  cout<< "Enter packet size"<<endl;
  cin >> inputpacket;
  if (inputpacket<= bucketsize)
{ if (filled + inputpacket > bucketsize)
  { cout <<" Packets too big for bucket"<<endl;
  }
  else {
      filled = filled + inputpacket;
  }
}

else {
  cout<< "Packets to big for bucket"<<endl;
}

if (filled <= outputrate)
 { filled =0;
 }
 else {
     filled = filled - outputrate;
 }
```

```
cout <<"Amount of bucket filled" << filled <<endl;
cout <<" Do you want to enter another packet (1 for yes, 2 for no)"
        <<endl;
cin >>choice;
} while(choice==1);
}
```

Output

```
Enter bucket size : 500
Enter output rate : 100
Enter packet size : 700
Packets too big for bucket
Amount of bucket filled 0
Do you want to enter another packet (1 for yes, 2 for no)
1
Enter packet size
200
Amount of bucket filled 100
Do you want to enter another packet (1 for yes, 2 for no)
1
Enter packet size 300
Amount of bucket filled 300
Do you wants to enter another packet (1 for yes, 2 for no)
1
Enter packet size 100
Amount of bucket filled 300
Do you want to enter another packet (1 for yes, 2 for no)
2
```

```
Enter bucket size
500
Enter  output rate
100
Enter packet size
700
Packets too big for bucket
Amount of bucket filled 0
Do you want to enter another packet(1 for yes, 2 for no)
1
Enter packet size
200
Amount of bucket filled 100
Do you want to enter another packet(1 for yes, 2 for no)
1
Enter packet size
300
Amount of bucket filled 300
Do you want to enter another packet(1 for yes, 2 for no)
1
Enter packet size
100
Amount of bucket filled 300
Do you want to enter another packet(1 for yes, 2 for no)
2
```

Lab 9

Bellman

AIM: write a C prog for Bellman ford algorithm

```
#include<stdio.h>
#include<stdlib.h>
int Bellman_Ford(int G[20][20], int V, int E, int edge[20][3])
{ int i,u,v,k,distance[20],parent[20],s,flag=1;
  for(i=0; i<V; i++)
    distance[i]=1000, parent[i]=-1;
  printf("Enter source:");
  scanf("%d",&s);
  distance[s-1]=0;
for(i=0; i<V-1; i++)
{  for(k=0; k<E; k++)
   {  u=edge[k][0], v=edge[k][1];
      if(distance[u]+G[u][v]<distance[v])
      distance[v]=distance[u]+G[u][v], parent[v]=u;
   }
}

for(k=0; k<E; k++)
 { u=edge[k][0], v=edge[k][1];
   if(distance[u]+G[u][v]<distance[v])
    flag=0;
 }
 if(flag)
  for(i=0; i<V; i++)
   printf("Vertex %d -> cost = %d parent = %d\n", i+1, distance
          [i], parent[i]+1);
  return flag;
}
int main()
{ int v, edge[20][3], G[20][20], i, j, k=0;
  printf("BELLMAN FORD\n");
  printf("Enter no. of vertices:");
  scanf("%d",&v);
```

```
printf("Enter graph in matrix form:\n");
for(i=0; i<V; i++)
  for(j=0; j<V; j++)
  {
   scanf("%d", &G[i][j]);
   if (G[i][j] != 0)
     edge[k][0] = i, edge[k++][1] = j;
  }
if (Bellmann_Ford(G, V, K, edge))
   printf("\n No negative weight cycle exists\n");
  return 0;
}
```

Output

```
BELLMAN FORD
Enter the no. of vertices : 4
Enter the graph in matrix form:
0   5   4   999
5   0   6   3
999 3   1   6
2   0   1   4
Enter source : 1
Vertex 1 -> cost = 0 parent = 0
Vertex 2 -> cost = 5 parent = 1
Vertex 3 -> cost = 4 parent = 1
Vertex 4 -> cost = 8 parent = 2

No negative weight cycle
```

```
BELLMAN FORD
Enter no. of vertices: 4
Enter graph in matrix form:
0 5 4 999
5 0 6 3
999 3 1 6
2 0 1 4
Enter source: 1
Vertex 1 -> cost = 0 parent = 0
Vertex 2 -> cost = 5 parent = 1
Vertex 3 -> cost = 4 parent = 1
Vertex 4 -> cost = 8 parent = 2

No negative weight cycle


...Program finished with exit code 0
```

Dijkstra

AIM: Write a C/C++ program for Dijkstra algorithm

```
#include <stdio.h>
#include <conio.h>
#define INFINITY 9999
#define MAX 10
void dijkstra (int G[MAX][MAX], int n, int startnode);
int main (){
    int G[MAX][MAX], i, j, n, u;
    printf ("Enter no. of vertices: ");
    scanf ("%d", &n); printf ("\n Enter the adjency matrix:\n");
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            scanf ("%d", &G[i][j]);
    printf ("\n Enter the starting node:");
    scanf ("%d", &u); dijkstra (G, n, u); return 0;
}
void dijkstra (int G[MAX][MAX], int n, int startnode){
    int cost[MAX][MAX], distance[MAX], pred[MAX];
    int visited[MAX], count, mindistance, nextnode, i, j;
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            if (G[i][j]==0)
                cost[i][j] = INFINITY;
            else
                cost[i][j] = G[i][j];
    for (i=0; i<n; i++){
        distance[i] = cost[startnode][i];
        pred[i] = startnode; visited[i]=0;
    }
    distance[startnode] = 0;
    visited[startnode] = 1;
    count = 1;
```

```
while (count<n-1){
    mindstance = INFINITY;
    for(i=0; i<n; i++)
      if (distance[i]<mindstance && !visited[i]){
          mindstance = distance[i]; nextnode = i;
      }
      visited[nextnode] = 1;
      for(i=0; i<n; i++)
        if (!visited[i])
          if (mindstance + cost[nextcode][i] < distance[i])
            { distance[i] = mindstance + cost[nextnode][i];
              prod[i] = nextnode;
            }
          count++;
}
for(i=0; i<n; i++)
  if (i! = startnode){
    printf("\n@Distance of node %d = %d", i, distance[i]);
    printf("\nPath = %d", i); j = i;
    do {
      j = prod[j];
      printf(" <- %d", j);
    } while(j! = startnode);
  }
}
```

Output

```
Enter no of vertices : 4
Enter the adjency matrix:
0    5    4    qqq
5    0    6    3
qqq  3    1    6
2    0    1    4
```

Enter the starting node: 1
Distance of node 0 = 5
path = 0 ← 1
Distance of node 2 = 4
path = 2 ← 3 ← 1
Distance of node 3 = 3
Path = 3 ← 1

```
Enter no. of vertices:4

Enter the adjacency matrix:
0 5 4 999
5 0 6 3
999 3 1 6
2 0 1 4

Enter the starting node:1

Distance of node0=5
Path=0<-1
Distance of node2=4
Path=2<-3<-1
Distance of node3=3
Path=3<-1

...Program finished with exit code 0
Press ENTER to exit console.
```

Lab 10

For Client

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
```

AIM: Using TCP/IP write client server program

2

```
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("\nEnter file name: ")
clientSocket.send(sentence.encode())
fileContents = clientSocket.recv(1024).decode()
print('\nFrom Server: \n')
print(fileContents)
clientSocket.close()
```

Output

```
Enter file name: server.py
From Server:
from socket import*
serverName = "127.0.0.1"
serverport = 12000
serverSocket = socket AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print('\n sent contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

Server

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to recieve")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print('\n Sent contents of' + sentence)
    file.close()
    connectionSocket.close()
```

Output

```
The server is ready to recieve

Sent contents of server.py
The server is ready to recieve
```

File Edit Shell Debug Options Window Help

```
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/mdsur/Desktop/server.py ==========
The server is ready to receive
```

File Edit Shell Debug Options Window Help

```
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/mdsur/Desktop/client.py ==========

Enter file name: server.py

From Server:

from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    print ('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()

>>>
```

File Edit Shell Debug Options Window Help

```
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/mdsur/Desktop/server.py ==========
The server is ready to receive

Sent contents of server.py
The server is ready to receive
```

Lab11

Date __/__/____
Page _____

Server UDP

AIM: Using UDP sockets. write a client server prog

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)
    serverSocket.sendto(bytes(l, "utf-8"), clientAddress)
    print('\nSent contents of ', end = ' ')
    print(sentence)
    file.close
```

Output

The server is ready to recieve
Sent contents of server UDP.py

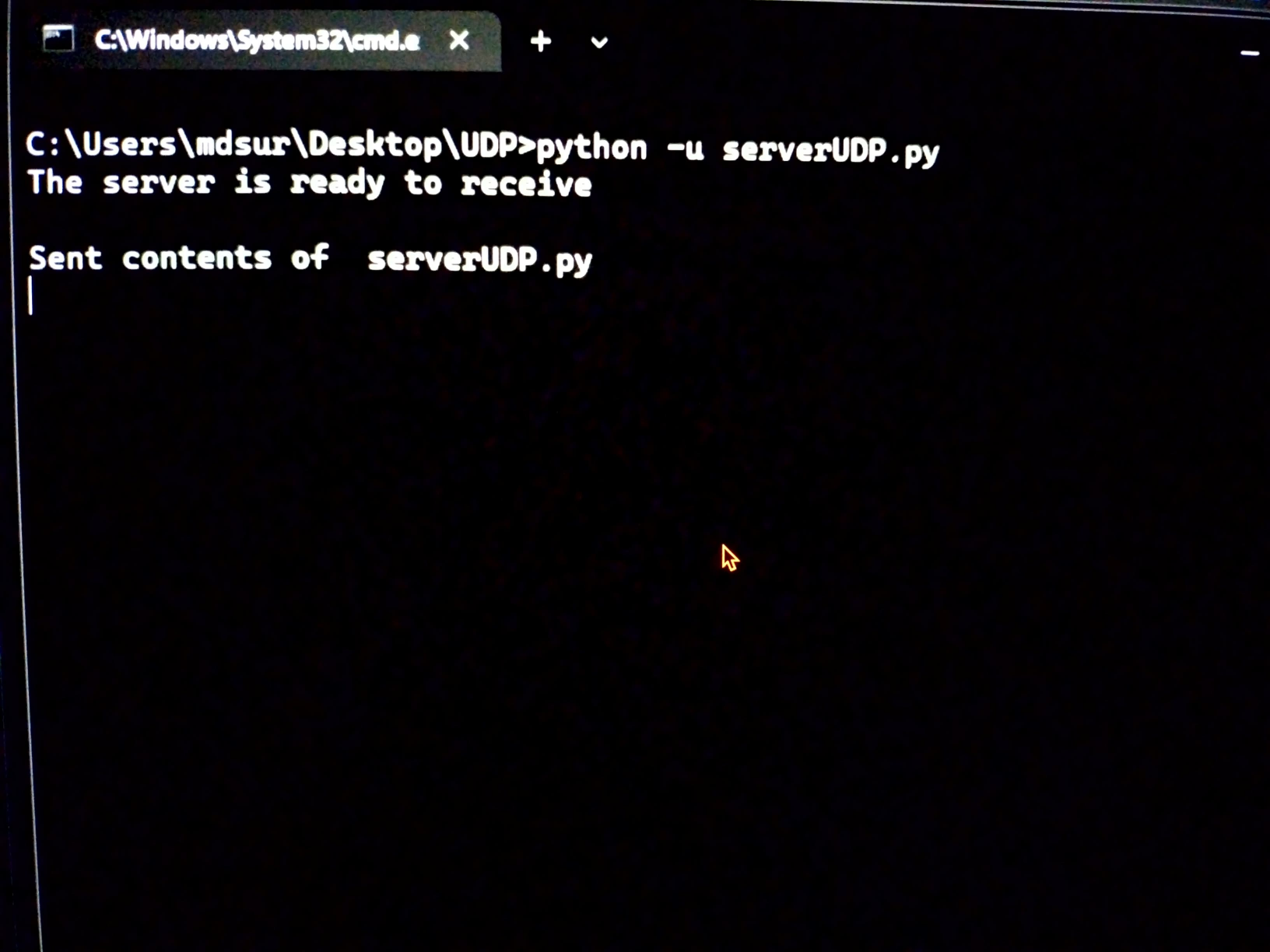
Client UDP

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\nEnter file name: ")
clientSocket.sendto(bytes(sentence, "utf-8"), (serverName, serverPort))
fileContents, serverAddress = clientSocket.recvfrom(2048)
print('\nReply from server: \n')
```

print (filecontents.decode("utf-8"))
clientSocket.close()
clientSocket.close()

Output

Enter file name: ServerUDP.py
Reply from server:
from socket import*
serverport = 12000
serverSocket = socket (AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode ("utf-8")
    file = open (sentence, "r")
    l = file.read(2048)
    serverSocket.sendto (bytes(l, "utf-8"), clientAddress)
    print ('\n Sent contents of ', end = ' ')
    print (sentence)
# for i in ~~in~~ sentence:
# print (str(i), end = '')
    file.close()


C:\Windows\System32\cmd.e
C:\Users\mdsur\Desktop\UDP>python -u serverUDP.py
The server is ready to receive
Sent contents of serverUDP.py

```
C:\Users\mdsur\Desktop\UDP>python -u clientUDP.py

Enter file name: serverUDP.py

Reply from Server:

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    l=file.read(2048)
    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print ('\nSent contents of ', end = ' ')
    print (sentence)
# for i in sentence:
# print (str(i), end = '')
    file.close()


C:\Users\mdsur\Desktop\UDP>
```