

Linux networking Commands

1. IP

ip address show

[root@localhost student]# IP address show

1. Loc Lookback, uplower-up > 65536

qdisc no queue state unknown group default

qlen 10000

→ IP address, add

IP address add 192.168.1.254/24 dev

enp250

→ ip address delete

ip address del 192.168.1.254/24 dev enp250

→ IP link Setup

ip link set up

→ IP link Set down

IP link set to down

→ IP link set promisc on

IP link set to promisc on

→ IP route
#

enp200
→ Add d

IP

→ Adding
#

→ delete

IP

→ displ

#

172

→ mt

→ IP route and default

ip route add default via 192.168.1.254 dev

enp200

→ Add default through gateway

ip route add 192.168.1.0/24 via 192.168.1.254

→ Adding route to device

ip route add 192.168.1.0/24 dev enp2504

→ delete route through gateway

ip route delete 192.168.1.0/24 via 192.168.1.254

→ display route for ip

ip route get 10.10.1.4

10.10.1.4 via 172.16.8.1 dev enp2504

172.16.8.92 idcache

→ mtr

mtr google.com

1 172.16.8.1

statistik 41.229.49

142.251.227.127

→ mtr -b

mtr -b google.com

localhosts: local domain (0.0.0.0)

172.2.16.8.1

142.250.171.162

142.251.227.217

→ tcpdump -D

1. enp2s0 I up, running
2. epp3s0 I up, running
- 3- any Cpseudo device that capture Interception
4. lo I up, running

→ tcpdump -i

tcpdump -i enp2s0 src host 0.0.0.0

tcpdump -vv for full protocol

decide listening on enp2s0

Link type ETHER capture size

262144 bytes.

→ ping -c

ping -c 10 google.com

64 bytes from mac 05:51:01-f4-1a:et

C216.08. 200-192) 1comp seg - 2td - 120

→ nrmali - Connection show

new 802.3 - ethernet Connection 7d45 b/w -

1899-4241-8649-03 clofacetaba 802-3-3

ethernet cnp280

⇒ Annot: Connection add Con-name "new 802-3-

ethernet - connection" if name enr250 type ethernet

~~Successfully added~~

3.

→ #nmcli Connection modify "new 802 - 3 - ethernet
Connection" Connection.interface - name enp2s0

→ # nmcli connection modify "new 802 - 3 -
ethernet Connection Connection - autoconnect yes

#nmcli show "new 802 - 3 - ethernet Connection"
Connection.interface - name enp2s0

id: "new 802 - 3"

Uuid: 25ff 1097 - 370e - 4505 - 9040

Interface name: enp2s0

type: ethernet

autoconnect: yes

read only: no

nmcli show "new 802 - 3 ethernet Connection

Connection - autoconnect yes

id: "new 802"

Uuid: 25ff 1097 - 370e - 42c18

type: ethernet

nmcli show

5. nmcli

nmcli

nmcli

manual

9/10 1904

22/8/20

auto connect: yes

nmcli show "new" ipv4 - method auto

5. nmcli modify "new" ipv4 - method auto


nmcli modify "new" ipv6 - method auto

nmcli modify connection "new" ipv4 method

manual ipv4 address 192.0.2.1/24

~~9/10~~ 192.0.2.28 ipv4 Gateway

~~22/8/24~~ dns 192.0.2.200 ipv4 dns searchexample.us

Ex no: 

Installation of Cisco Packet Tracer

Aim








To study the Cisco Packet tracer tool installation and user interface simple network

Steps

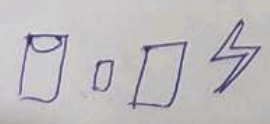
↳ Installation Cisco Packet tracer

↳ Setups the software for students login menu bar

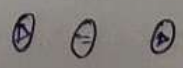
Cisco Packet tracer

File Edit option view tools Extension help
Tool bar       

Logical / physical workspace

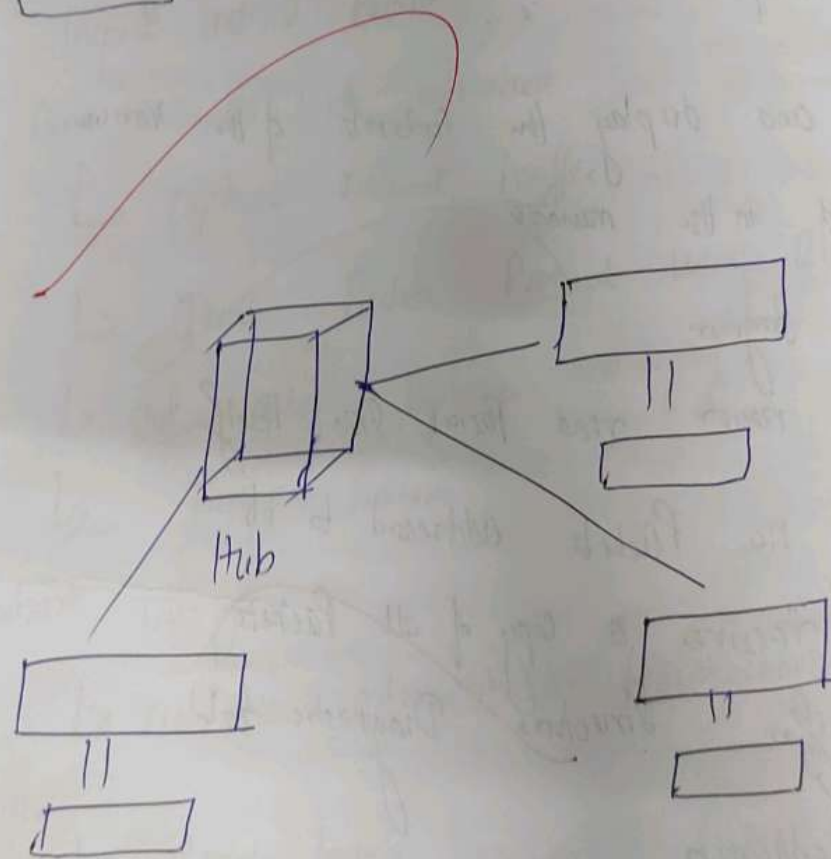
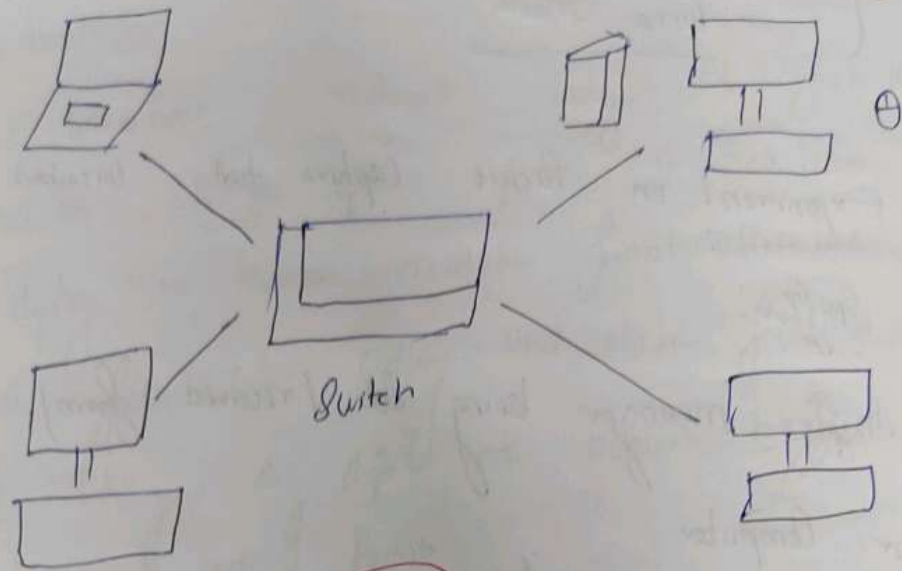


Scenario



network Connection

Switch vs Hub



Exp no: 5

wireshark

Aim:

Experiment on Packet Capture tool : Wireshark

Packet Sniffer:

* Sniffer messenger being sent / received from your Computer

* Store and display the content of the various potential fields in the message

* Passive Program

↳ never read packet size itself

↳ no packets addressed to it

↳ receive a copy of all packets

Packet Sniffer structure Diagnostic tool
tcpdump

↳ Eg: tcpdump -e -x -t

10.129.4.1.2 - e - pxe3 out

↳ wireshark

wireshark -r pxe3out

Description:

Wireshark:

Wireshark

known as

and deploy

includes file

that let

and inspect

What can

↳

↳

↳

↳

Wireshark

↳

Protocol

↳

↳

Description:

Wireshark:

Wireshark is a network analyzer tool formally known as Ethereal Capture Packets in real time and display in human readable format. Wireshark includes filter, color, coding and other features that let you dig deep into network traffic and inspect individual packets.

What can we do with Wireshark

- ↳ Capture network traffic
- ↳ Decode Packets Protocol using dissect
- ↳ Filter - capture and display
- ↳ Analyze Problems

Wireshark uses for

- ↳ Network administrators: troubleshoot network Problems
- ↳ People learn network Protocol internal
- ↳ Developer - debug Protocol / implementation

Getting Wireshark

Wireshark can be downloaded for Windows or macOS from its official website for Linux or another Unix-like system. Wireshark will be found in its Package repositories.

The Packet List Pane

The Packet List Pane displays all the packets in the current capture file. The Packet List Pane. Each line in the Packet List pane is compared to one packet in capture file.

The Packet Details Pane

The Packet Details Pane shows the current packet in a more detailed form. The Packet Details Pane shows the Protocol and Protocol fields of the packet selected in the Packet List.

The "Packet Bytes" Pane

The Packet Bytes pane shows the data of the current packet in a hex dump style.

Sample Ca

If

Own not

Covered.

Files that

Filtering Pac

Specific

when helps

using the
traffic

The

Time + Tot

+ x

x x

Sample Capture

If there's nothing interesting on your own network to inspect, Wireshark has you covered. The wiki contains a Page at Sample Capture files that you can load the inspect

Filtering Packets:

It's going to inspect something specific such as the traffic or Program Services when ~~helps~~ to close down all other application using the network so you can narrow down the traffic

The sum of Tim and Tom

$$\text{Tim} + \text{Tom} = 39$$

$$x + y = 39$$

$$x + 3 + x + 3 = 39$$

$$x + 3 + 2(x + 3) = 39$$

$$x + 3 + 2x + 6 = 39$$

$$3x + 9 = 39$$
$$3x = 39 - 9$$
$$3x = 30$$
$$x = \frac{30}{3}$$
$$x = 10$$

$$x - 2 + 2x + 19$$
$$4(7) + 1$$
$$28 + 6$$
$$34$$
$$7 + 3$$

from

Exercise - 6

Aim:

Write a Prog to implement error detection
the Conversion of Using Hamming Code

Create Sender Program with below features

- 1) Input to Sender file should text of any
Length Program should convert to text
- 2) Apply hamming code concept
- 3) Save the file in filled call channel

Create a Receiver

- 1) Input to Receiver from channel file
- 2) Apply hamming code on the binary
- 3) If this is an error on binary data
- 4) Else remove the result this and
convert the binary data to text

Sonder Py

string = input("Enter string: ")

s = ' '.join(format(ord(c), '09b') for c in string)

rb = 0

for i in range(len(s)):

if (2+i) >= len(s) + i + 1:

rb = i

break

m = len(s) + rb

l = []

pos = []

c = 0

s = s[::-1]

for i in range(rb):

pos.append(2+i)

for i in range(m):

if (i+1) in pos:

l.insert(i, 'p')

else:

l.insert(i, int(s[c]))

c += 1

print ("parity bit position:", pos)

print ("Initial encoded data with 'p' at parity position", l)

for Pin p:

Count=0

i=p-1

while i < m

Count += |E[i:i+p] count()

it = 2 * p

|E[p-1] = 1 if Count > 2 | = 0 else 0

print C "Final encoded data with Amtybib, '1E::-[]"

f = open("code.txt", "w")

f.write(Cstr[E::1])

f.close()

f.open("original.txt", "w")

f.write(Cstr[E::1])

f.close()

Receiver.py

f = open("code.txt", "r")

x = f.readline()

print C "E cd:" : x)

e = open("original.txt", "r")

y = e.readline()

print C "0

l = x.strip()

for i in l:
i.strip()

C = E

pos = E

rb = 0

for i in range

if C >

rb = i
break

for i in range

pos.append
print C "Pos: 10

q = E

for i in E::1:

if C i =

q.q

else
q.append

C.append

print C "Bi

change = 0
q = E::1

print C "

```
print C "0 code: ", x)
```

```
l = x.strip(" ").strip("\n").split(",")
```

```
for i in l:  
    i.strip(" ")
```

```
l = []
```

```
pos = []
```

```
rb = 0
```

```
for i in range(len(l)):
```

```
    if C[i] + i >= len(C) + i + 1:
```

```
        rb = i
```

```
        break
```

```
for i in range(rb):
```

```
    pos.append(i + 1)
```

```
print C "Position: ", pos)
```

```
q = []
```

```
for i in l[1:]:
```

```
    if C[i] == "1" or i == "1":
```

```
        q.append(i)
```

```
    else
```

```
        q.append(0)
```

```
C.append(0) if count + 1 == 0 else C.append(1)
```

```
print C "Binary: ", C[1:-1]
```

```
change = 0
```

```
q = l[1:]
```

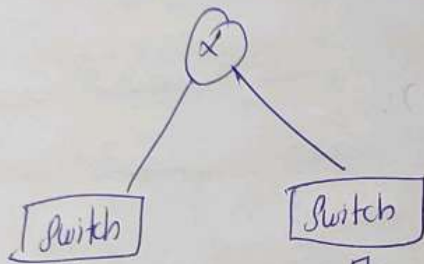
```
print C "Error: change)
```

```
print C "Corrected Code: ", q)
```


Ex-7

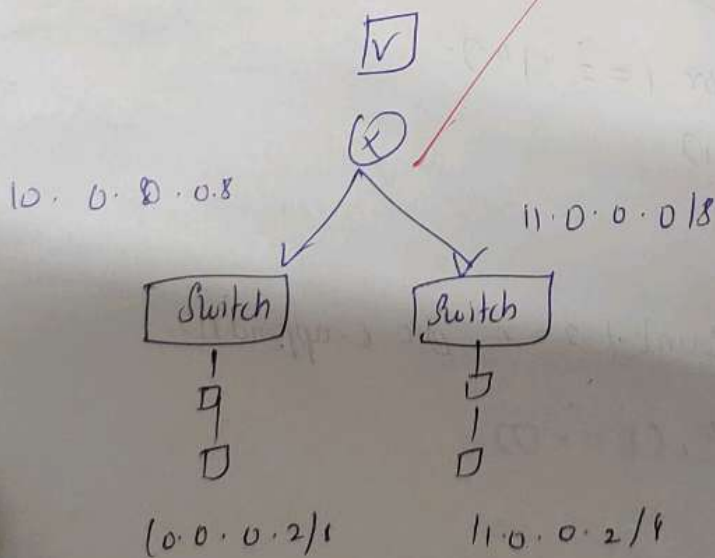
Aim:
Setup and connect a router to enable connection
b/w two networks

Connect every component as show below



3 → Configure IP to a component

we h/w id $10.0.0.0/8$ for a LAN and $11.0.0.0/8$
for another LAN



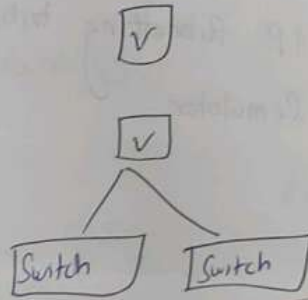
4 → give
1 → After
to check

Result:
Thus the

4 → give gateway address

1 → After Configure on all ports and one Command

to check Connection b/w the



Result:

This the experiment executed successfully

Ex-9

Aim

To implement a IP Subnetting techniques using Cisco packet tracer simulator

Steps:

Creating network topology:

Create a blank topology by clicking "new" button and then select "network" and

Adding Devices:

we will be adding router, switches and PCs to add select a device and drag in into the network topology

Subnetting:

To Subnet the network address of 192.168.0.124 to provide enough space by atleast 5 address for end devices, two switches and the router he can use a 12th Subnetmask

Pc0

Pc1

Pc2

Pc3

4) Config

5) Test

enc

conf

Inter

Swit

exit

interfa

Switc

exit

Result:

Ther

process in

Pc0

Pc1 Switch 0

Routers

routers

Pc2

Switch 1

Switch 2

Pc3 Switch 1

Pc4 Pc5 Pc6 Pc7

4) Configure the devices

5) Testing the network

enable

config terminal

Interface fast ethernet 0/1

Switchport mode access

exit

interface fast 0/1

Switchport access

exit

Result:

2/14 9/10

Thus the implementation of Subnetting
proven in vivo packet tracer is successfully executed

Ex-10

Aim:

Ethernetworking with wireless router http
Server and internet cloud

Steps:

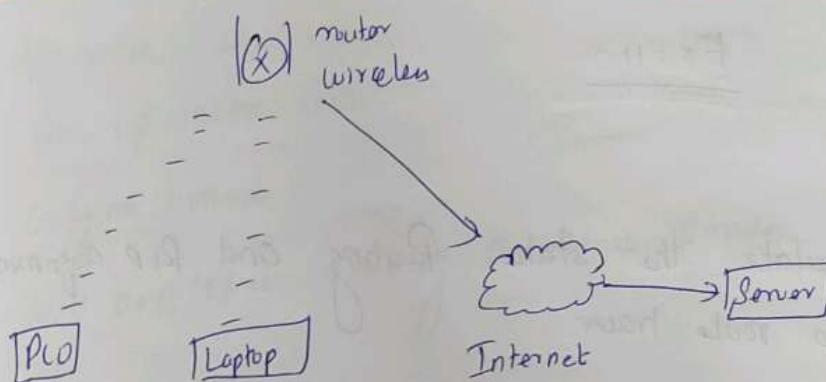
- * change display name of devices as wireless router o
- * change the Setup of the wireless router in ~~PHP~~ ^{PHP} enable
- * copy the ip address of the router and make the enable server
- * The Pcs go the Desktop go the Connect Router wireless mode into action
- * Go to the link formation, to see is five
- Connection got established.
- * go to physical and PHP button got enabled
- * Ping the router Pcs from any other Pcs through the wireless to get reply from PLS

Result:

Thus

wireless

IP assign



8/15/14 to

Result:

These all the PC get connected to the wireless router and hence through the PC the IP assignment automatically to all PC

Ex 11

Aim:

Simulate the Static Routing and RIP assignment
the Cisco route tracer

Steps:

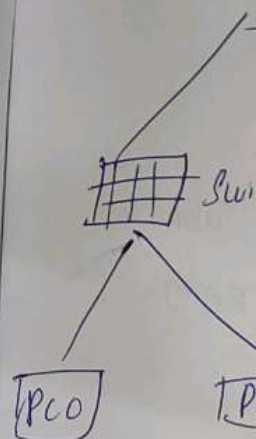
Assign the IP router address for each network
and classify the server router

For each router there is a collection of PC connected
to router

Assign the router IP address as gateway connected
each the PC as their IP address

Is a result of it for each PC the route get
updated and get reply from the within Protocol

After that statically assign the IP address
of the router network can get connected



Routing Info

IP address

no Shutdown

exit

IP address

no Shutdown

IP address 10

check value 64

exit

network 192.16

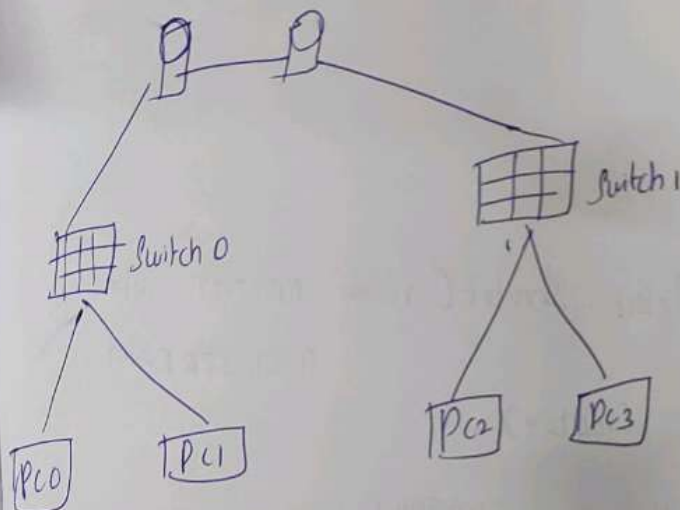
Result: 11

IP route static

Show ip route static

Configure terminal

20.0.0.0/8 is Subnetting 2, Subnet, 2 master



Routing Information Protocol:

IP address 10.0.0.1 255.0.0.0

no Shutdown

exit

IP address 10.0.0.1 255.0.0.0

no Shutdown

IP address 192.168.1.246 255.255.0.0

check value 64006

exit

network 192.168.1.282

Result: Thus the RIP configuration and static IP config has done.

Ex - 12

Aim:

Implement chat client server using TCP/UDP

Sketch

Program:

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

print("Socket created")

s.bind(("localhost", 5555))

s.listen(5)

while True:
    conn, addr = s.accept()
    print("Connection from", addr)
    data = conn.recv(1024)
    if not data:
        break
    print("Received from client:", data)

    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client.connect(("localhost", 5555))
    name = input("Enter your name: ")
    print("Message received from server: ")
```

UDP Program

import

from

socket

UDP

host =

server

while 1:

data =

print

socket

Output:

Message

udp client

Message

Connect

Do you

Then the

Result

Then the

UDP Program.

import sys

from socket import

Server Port = 55555

Buffsize = 1024

host = "127.0.0.1"

s = socket ("Proto - PORT")

while 1:

data, addr = s.recvfrom (1024)

print ("Server received data from %s" % addr)

s.sendto (data, addr)

Output:

Message sent to client

udp client to reply

Message received from client
addr to server

Connect with 127.0.0.1

Do you want to get continue in break

Then the connection get terminated

Result:

This is client client using TCP/IP server has been done

Ex-13

Aim:

To implement your own Ping Program

Steps:

1. import sys

from socket import

ECHO - PORT = 55555

BUFSIZE = 1024

s = socket(AF_INET, SOCK_DGRAM)

s.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)

print('udp server echo ready')

while 1:

data, client = s.recvfrom(BUFSIZE)

s.sendto(data, client)

import sys

from socket import

import sys

s.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)

ECHO - PORT = 55555

host = "129.169.8.10"

s.sendto(data, client)

data, client = s.recvfrom(BUFSIZE)

print()

Onel

mn

OUTPUT

udp

client

udp

server

Result:

Then the

printf("client received %s from %s\n", data, from)

Onel = time.time()

printf("R+2")

OUTPUT

udp echo client ready, reading stdin

client received puma 124.0.0.0

udp Server is ready

Server received from 172.16.8.0

Result:

2/19/14 9/10
Then the Ping Program has been executed

Ex-14

Aim

implement your packet sniffing

CODE:

```
from Scapy import all
from Scapy.layers.net import IP, TCP, UDP

pcap = ...

def packet_callback(packet):
    if IP in packet:
        src_ip = packet[IP].src
        dst_ip = packet[IP].dst
        protocol = packet[IP].proto

        if protocol == 6:
            protocol_name = "TCP"
        elif protocol == 17:
            protocol_name = "UDP"
        else:
            protocol_name = "Unknown"

        print(f"Protocol: {protocol_name}")
        print(f"Source IP: {src_ip}")
        print(f"Destination IP: {dst_ip}")
        print(f"Length: {len(packet.payload)}")
```

def main():

sniff (ip addr = "wif: " opm = packet - callback, function
"ip", store = 0)

if name == "__main__":

main()

OUTPUT:

protocol : TCP

Source IP: 20.24.84.143

Destination IP: 172.20.10.2

Print:

19/11/2019
Thus the Packet sniffing has been implemented
Successfully and verified

Ex 15

Aim

To analyze the different type of web log using webanalyzer

Procedure:

Step 1: Run webanalyzer window version

* input web log file & download from web

* Press run

* Input Directory is given at the forget Directory

OUTPUT:

Normal Statistic

Summary Subnet max

192.168.1.2 192.168.1.29

87 89.36 101.24 62

HOSTS

Address	Destination	IP
100	8.14.1.	88.1.
72	8.94.1.	43.1.
39	94.2.1.	41.1.

unrecognized

Address

Execute

Veracore

110.46

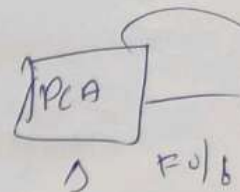
100.00.1.

224.00.13

Result: Thus the different type of weblogs using the analyzer has been Identified and verified.

Ex

Virtual lan



Addressing Table

Device Interface

S1 VLA

S2 VLA

PC-A N

PC-B M

Part -1

objective

To build

device setting

Step

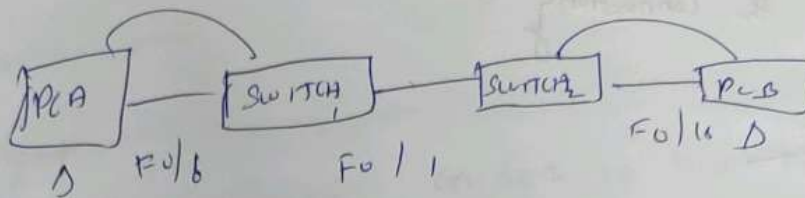
1) Build the

2) open

basic set

Ex - 8

Virtual lan Configuration using Cisco Packet Tracer



Addressing Table:

Device	Interface	IP Address	Subnetmask	Default Gateway
S1	VLAN 1	192.168.1.1	255.255.255.0	N/A
S2	VLAN 2	192.168.1.12	255.255.255.0	N/A
PC-A	NIC	192.168.10.3	255.255.255.0	192.168.1
PC-B	NIC	192.168.10.4	255.255.255.0	192.168.1

Part - 1

Objective

To build the network and Configure hosts

device settings

Step

- 1) Build the network as shown in topology
- 2) open the configuration window and Configure basic settings for each switch

3) From the Desktop the configuration pc keyboard
enter IP addressing Information

4) Test the Connectivity

Part - 2

objective

To create Virtual LAN's and assign Switch ports

Steps:

Create Virtual LAN's on the Switches

1) open Configuration window

a) Create LAN's on S1

b) Create Same on S2

c) Issue show vlan

3) Assign VLAN to connect Switch

Part - 3

objective

To maintain VLAN Port Assignment and
the VLAN database

Steps:

Design a LAN to multiple interface

a) Remove v

3) Remove a

Part 4:

Objective:

To

the Switches

Steps:

1) Use 2

2) Manu

a) on Int

made to force
on both Switch

b) Design
given Scenario

Result:

Therefore

Configuration

is executed

pc hdb and

- 2) Remove VLAN assignment from an Interface
- 3) Remove a VLAN ID and the VLAN Database

Part 4:

Objective:

To Configure on 80.2.10 trunk betw
the Switches

Steps:

- 1) Use DTP to initiate trunking on Fa/1
- 2) Manually configure the trunk Interface Fa/1
- 3) on Interface Fa/1, change the Switch port mode to force trunking. make sure to do this on both Switches

4) Design and configure a VLAN for the below given Scenario

Result:

Therefore the Simulation of Virtual LAN Configuration Cisco Packet Tracer simulation is executed Successfully