United International University

School of Science & Engineering

Laboratory Manual

Computer Networks Laboratory

Version 4.0

First Edition: Summer 2018

Last Edition: June, 2022

Cisco Networking Academy®

Mind Wide Open™

Contents

Lab Materials: https://tinyurl.com/UIU-CN-Lab

Lab-1 - Introduction to Networks Understanding TCP/IP Properties

Objectives

- Understanding Networks Components
- Use of IP address, Subnet mask, Default gateway, MAC address.
- Identify tools used to discover a computer network configuration in Windows.
- Gather information including connection, host name, MAC address and IP address information.
- Compare network information to other PCs on the network.
- Learn to use the TCP/IP Packet Internet Groper (ping) command from a workstation/PC.

Part-1: Understanding Networks Components

1. End Devices

Server
A server is a computer that serves the data to other computers and users. The network components can be in the form of a computer, a hardware device, or a computer program that is loaded so that it can send data and any information to other computers. The term "server" usually refers to a computer system that receives a request for a web document and sends the request information to the client.

Client
The device that receives requests, and responses from the server, is called a client. When the server and its clients work together on the computer, we call it the client/server network.

2. Intermediate Devices

HUB
A hub is a device that splits a network connection among multiple computers. It works similarly to a distribution center. When a computer requests information from one network or from a specific computer, then it sends the request to the hub through a cable. The hub then receives that request and transmits it to the entire network.

After that, every computer checks whether that network then belongs to them or not. If belongs then it broadcasts if the request doesn't belong it will be dropped.

However, such network components nowadays are very less in circulation and being replaced by more advanced communication devices such as Routers and Switches. This hub is basically a multiport repeater.

This hub is used to connect multiple connections that come from different branches, For example, the connector in star topology is used to connect different stations for data access.

Switch
The switch is a component that helps devices to connect the networks so that they can transfer data to other connected devices. These network switches are identical to network hubs, but a switch has more advanced features than a hub. It doesn't broadcast entire data on the network like a hub.
The advanced features of the switch imply that the network switch first inspects the incoming packet and determines its source, destination address, and routes after that sends the data at the

correct destination accordingly to that packet. A network switch is also called the Switching hub, Bridging hub, and MAC Bridge.

Router
The router is a hardware network component. Routers operate at the network layer of the OSI (Open system interconnection) reference model, using them to send packets over the network using a logical address.

Any data which travels from one network to another network as a Packet. The Router receives such Packet data and forwards it to the Destination Device after analyzing hidden information in the Data Packet. This Networking Device is used to connect different networks either it is wired or wireless.

3. Media

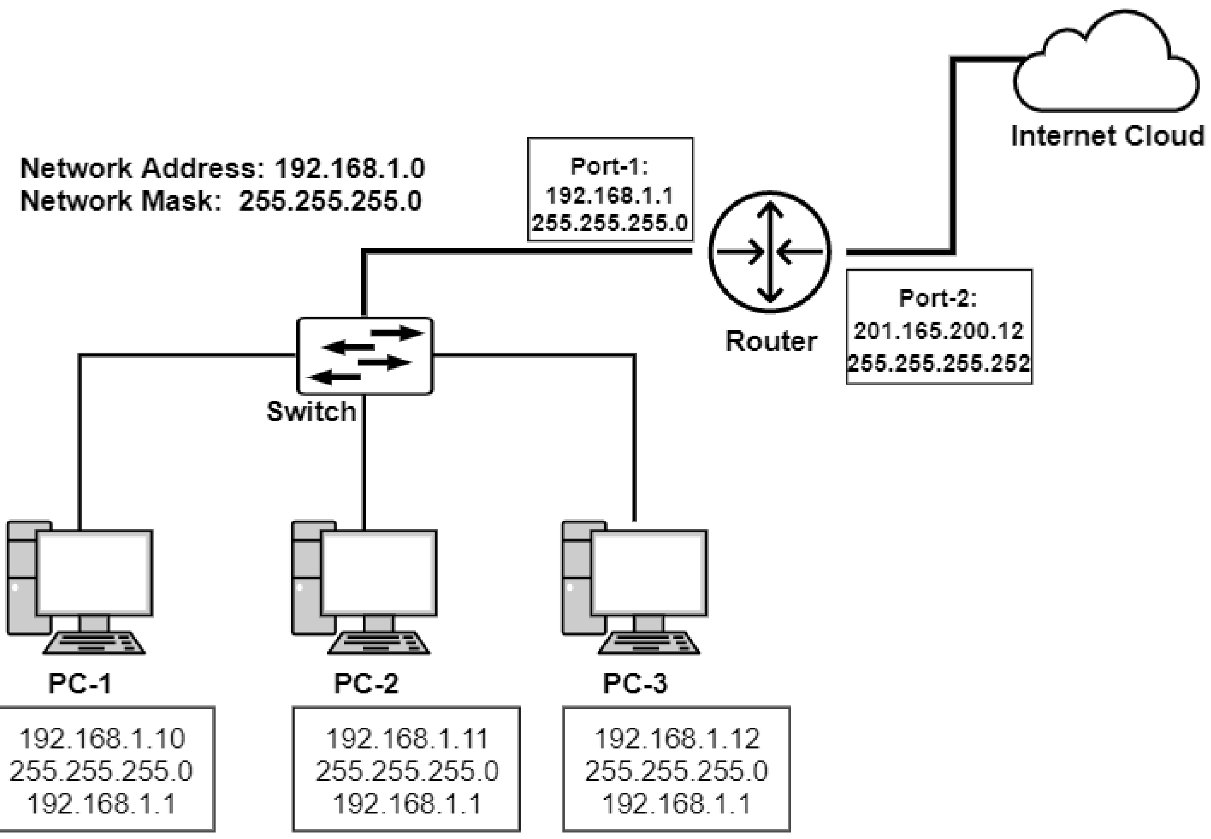
NIC (Network Interface cards)
Network Interface cards (NICs) are also called Network Interface Controller, Network adapter, LAN adapter, and Physical Network interface. NIC cards are hardware components used to connect computers with networks. Without NIC a computer cannot be connected to the network.

It is installed in a computer circuit board that provides a network connection to the computer. Due to the popularity and low cost of Ethernet standards, the network interface is built directly into the motherboard in almost all new computers.

Transmission media
Transmission media are the medium through which data is transferred from one device to another in a network. Transmission media can be used either in a physical transmission medium or wireless transmission medium.

Physical transmission medium includes the use of wires and cables like fiber optic cables, coaxial cable, etc.; and wireless transmission medium includes the use of unguided media like infra-red waves, electromagnetic, microwaves, etc.

Part-2: IP address, Subnet mask, Default gateway, MAC address



a. **Subnet Masks**: Subnet mask identifies the network portion of an IP address. Make sure every machine in the same network has the same subnet mask. The value of subnet mask is 255.255.255.0 in the example we will use.

b. **Network or Subnet Address:** Find the network portion of the IP address of the Gateway Machine. Fill in the host portion with 0s. Write that label above the network (in the upper left, in these diagrams). In the above example, the Gateway Machine has an IP address of 192.168.1.101 and since the subnet mask is 255.255.255.0, the network portion includes only the first 3 bytes. To find the subnet label, replace the last byte with zero: 192.168.1.0.

c. Check the IP Addresses:

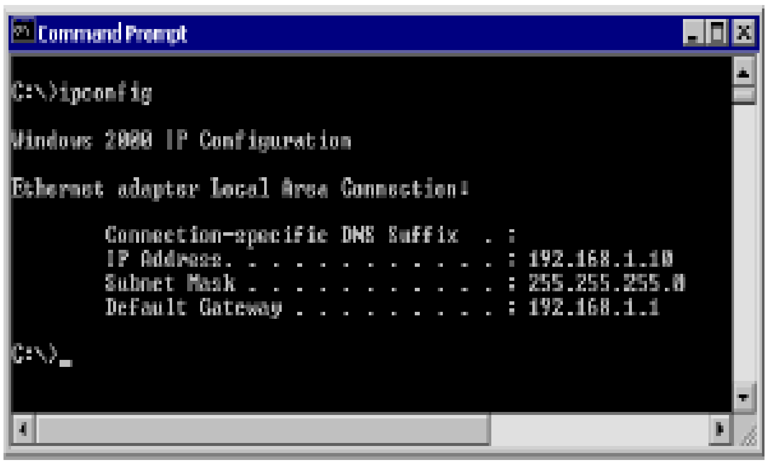
- **Network Portion:** Make sure that each NIC on a subnet has the same network address as the label you wrote at the top of the subnet. In the example, on the left subnet, that means every IP address must start with 192.168.1

- **Host Portion:** Make sure that each NIC on a subnet has a different host address, including the default gateway. In the example, the Gateway Machine has a host address of 1, and the others are 101, 102, and 103, so there are no duplicates.

d. **Default Gateway:** On each subnet, the default gateway is the Gateway Machine's IP address. It is the same for each NIC on the subnet, except the Gateway Machine itself, which has a default gateway of the network above it, usually an ISP. In the example, the Gateway Machine has an IP address of 192.168.1.1, so the default gateway must be 192.168.1.1 for all three PCs at the bottom of the chart.

Part-3: Gather TCP/IP configuration information

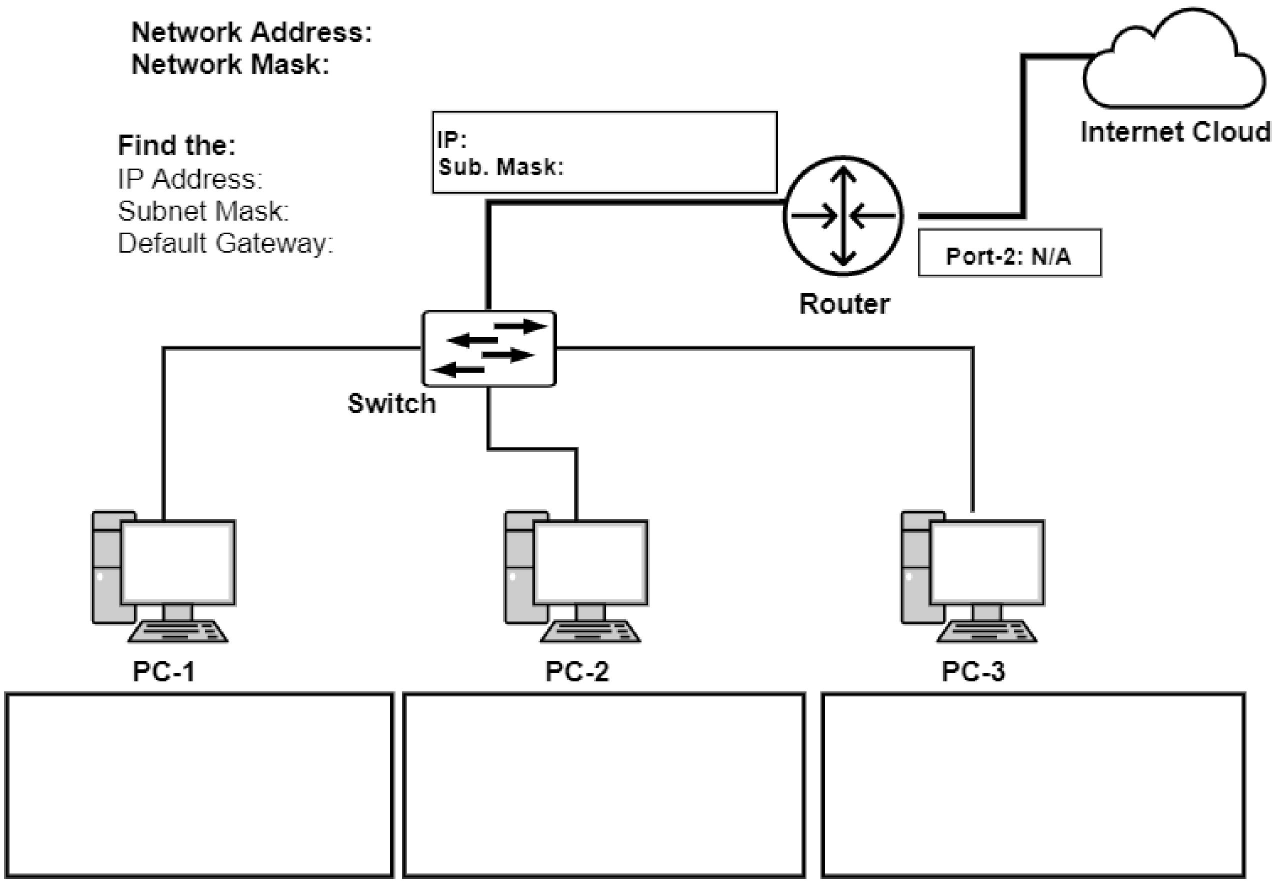
Use the Start menu to open the Command Prompt, an MS-DOS-like window. Type **ipconfig** and press the Enter key. The **ipconfig** is used for gathering the IP Configuration information. The following figure shows the Command screen.



```
C:\>ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.10
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\>_
```

Notes: This first screen shows the **IP address**, **subnet mask**, and **default gateway**. The **IP address** and the **default gateway** should be in the **same network** or **subnet**, otherwise this host would not be able to communicate outside the network. In the figure the **subnet mask** tells us that the **first three octets** must be the same to be in the same network.

Part-4: Experiment -1 - Gather information and



Part-5: Check additional TCP/IP configuration information

To see detailed information, type **ipconfig /all** and press Enter. The figure shows the detailed IP config-screen. The host name, including the computer name should be displayed. Notice the **Physical Address (MAC)** and the **NIC (Network Interface Card)** model (Description). All machines share the first three Hex pairs in the adapter address. These three pairs identify the manufacturer of the adapter.

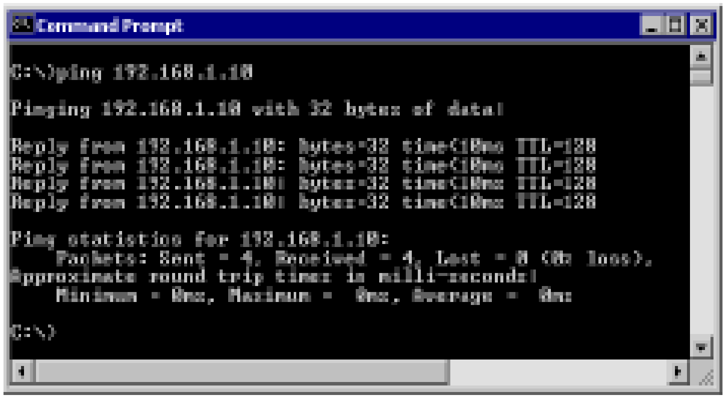
Write down the following from the output:

Host Name:
IP addresses of DNS server:
Now. Do your PC and DNS Server above share the same network portion? (Explain)

Answer:

Part-6: ping the IP address of another computer

In the window, type **ping**, a space, and the IP address of another computer. The following figure shows the successful results of **ping** to this IP address.



```
C:\>ping 192.168.1.10

Pinging 192.168.1.10 with 32 bytes of data:

Reply from 192.168.1.10: bytes=32 time<10ms TTL=128
Reply from 192.168.1.10: bytes=32 time<10ms TTL=128
Reply from 192.168.1.10: bytes=32 time<10ms TTL=128
Reply from 192.168.1.10: bytes=32 time<10ms TTL=128

Ping statistics for 192.168.1.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms

C:\>
```

Ping uses the ICMP echo request and echo reply feature to test physical connectivity. Since ping reports on four attempts, it gives an indication of the reliability of the connection. Now, ping the IP address of Computer B. Look over the results and verify that the ping was successful. Is the ping successful? (yes/no) .

Note the results: Packets: sent = , Received = , Lost =

Ping the IP address of the default gateway

Try to ping the IP address of the default gateway listed in the last exercise. If the ping is successful, it means there is physical connectivity to the router on the local network and probably the rest of the world. Was the ping successful? (yes/no)

ping the Loopback IP address of this computer

Type the following command: ping 127.0.0.1 The 127.0.0.0 network is reserved for loopback testing. If the ping is successful, then TCP/IP is properly installed and functioning on this computer. Was the ping successful? (yes/no)

Lab-2 - Introduction to Packet Tracer, Building a single-segment Network, Protocol Analysis using OSI and TCP/IP Model

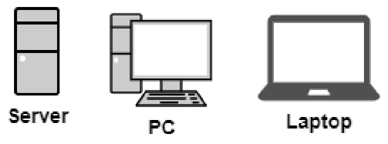
Objectives

- To introduce **Packet Tracer** (**PT**) and become familiar with its operations.
- Create a simple network using **Hub**, **Switch**.
- Learn about basic components of a router.
- Configure router interfaces and learn other basic router configuration settings.
- To connect hosts in different networks using a Router in Packet Tracer.

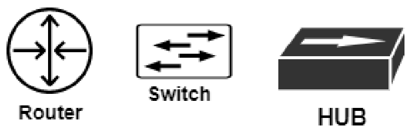
Part-A: Introduction to Packet Tracer

Part-1: Network Devices Symbol

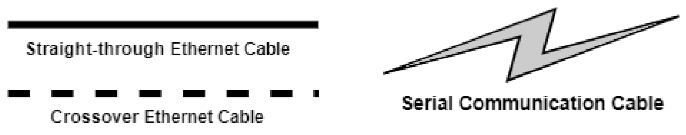
End Devices:



Intermediate Devices:

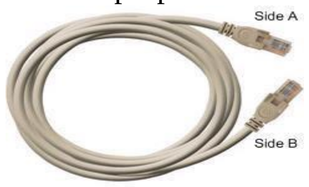


Media:



Part-2: Crossover and Straight-through Cables

Common Ethernet network cables are straight and crossover cable. This Ethernet network cable is made of 4 pair high performance cable that consists of twisted pair conductors that used for data transmission. Both end of cable is called RJ45 connector. Straight and crossover cable can be Cat3, Cat 5, Cat 5e or Cat 6 UTP cable, the only difference is each type will have different wire arrangement in the cable for serving different purposes.



Category 1 (Layer 2 or below): Hub Switch

Category 2 (Layer 3 or above): PC/Laptop/Server Router

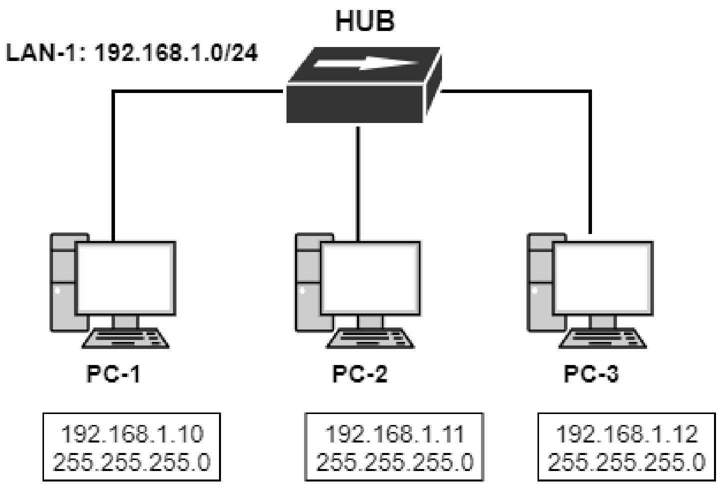
You usually use straight cable to connect different type of devices (different category). This type of cable will be used most of the time and can be used to:

- Connect a computer to a switch/hub's normal port.
- Connect a router's LAN port to a switch/hub's uplink port. (normally used for expanding network)

Sometimes, **you will use crossover cable, it's usually used to connect same type of devices or same category**. A crossover cable can be used to:

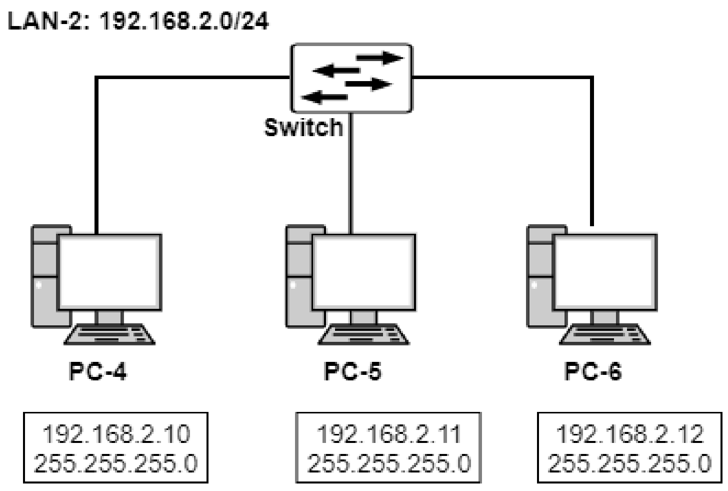
- Connect 2 computers directly.
- Connect a router's LAN port to a switch/hub's normal port. (normally used for expanding network)
- Connect 2 switches/hubs by using normal port in both switches/hubs.

Part-3: Create a single-segment network using a Hub



Test the network, e.g., ping PC2 from PC1 and ping PC2 from PC3.
Run the test in Simulation mode of Packet Tracer and Ping again

Part-4: Create a single-segment network using a Switch and Compare with the Part-3



Test the network, e.g., ping PC5 from PC4 and ping PC6 from PC5.
Run the test in Simulation mode of Packet Tracer and Ping again

Based on Part-3 and Part-4,

Which type of network is good for the enterprise solution? (Explain)

Part-B: Basic Router Configuration

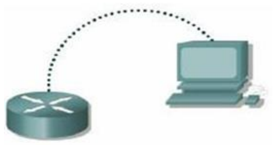
Part-1: Theoretical background

Cisco Internet Operating System (IOS) Command Interface User Levels The following table contains the different IOS command modes, their roles and the shape of the command prompt that illustrates the mode. Make sure to study this table carefully as it is essential for proper working with Cisco routers and switches.

IOS command mode	Functions/Roles of the mode	Command prompt
User EXEC mode	* Limited command set, e.g., ping, telnet * Change of system parameters	Router> Router>enable Router#
Privileged EXEC mode	* Manage configuration files * Examine state of router * Access control with password	Router# Router# configure terminal Router(config)#
Global Configuration Mode	* Change system parameters	Router(config)#interface fa0/0 Router(config-if)#
Interface Mode Or, Other Configuration Mode	* Interface configuration mode: Modify configuration of a specific interface * Router configuration mode: Modify configuration of specific routing protocol	Router(config-if)# Router(config-router)#
Use TAB and "?" mark frequently for command		

Part-2: Configure a Cisco IOS Router/Switch.

PT Terminal is a simple emulation program for serial communication that can be used to connect to the console port on Cisco IOS devices. A serial interface on a computer is connected to the Cisco device via a console cable. Using PT Terminal is the most basic way to access a router for checking or changing its configuration.



Step-1: Select a PC and a router from the Network Component Box.

Step-2: Connect the console (rollover) cable to the console port on the router. Connect the other cable end to the host computer with a **RS-232 port.**

Step-3: From the Windows taskbar, start the PT Terminal program by clicking

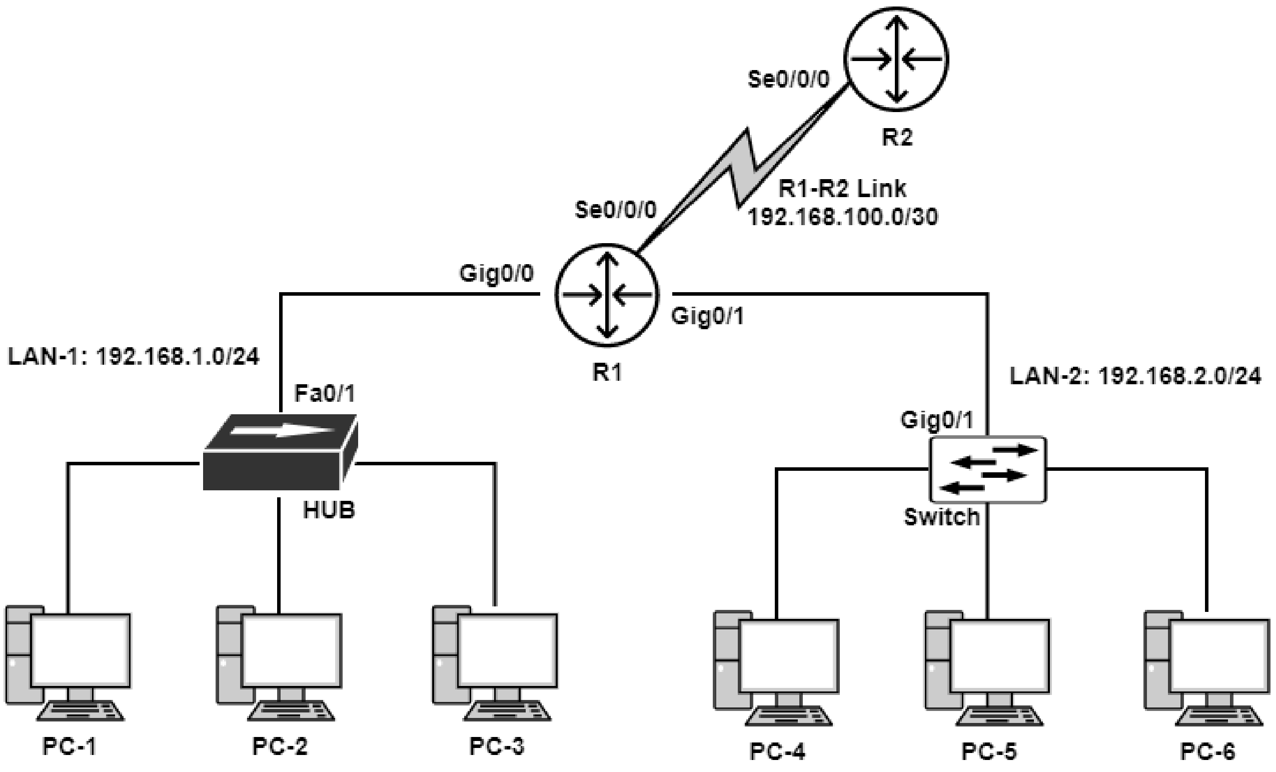
PC>Desktop Tab> Terminal

Step-4: **Click OK.**

5-You should see a response from the router on the screen (press enter several times).

Note: this process is the same for a Switch.

Part-3: Creating a Local Area Network (LAN) using Hub/Switch/Router



Part-3(A): Router console session and Global Configuration Mode
At First you need to create a console session for configure the Router. Then access via Console session and enter the specific command,

Router Console Session and User **Exec** mode

```
Router >
```

Router Exec mode to privileged Exec mode

```
Router >
Router >enable
Router #
```

Router privileged mode to configuration modes.

```
Router #
Router # configure terminal
Router (config) #
```

Part-3(B): Change Router Host Name as R1

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

Part-3(C): Assign R1 Gig 0/0 IP Address and state up the interface

G0/0 ip address is assign by LAN-1 Network and G0/0 is the default gateway of the LAN-1. For design purpose, we use 1st or Last IP address as a Default Gateway.

So, we use 1st IP Address of LAN-1. IP Address: 192.168.1.1 Subnet Mask: 255.255.255.0

```
R1 (config) # interface Gigabitethernet 0/0
R1 (config-if) # ip address 192.168.1.1 255.255.255.0
R1 (config-if) # no shutdown
R1 (config-if) # exit
R1 (config) #
```

Note: Now you configure the Gig0/1 interface.

Part-3(D): Configure R1-R2 Link Network

We take 1st IP address in R1 Side,

```
R1 (config) # interface Se0/0/0
R1 (config-if) # ip address 192.168.100.1 255.255.255.252
R1 (config-if) # no shutdown
R1 (config-if) # exit
R1 (config) #
```

Note: Now you configure the R2 Serial interface.

Part-3(E): Assign End Devices IP Address with the Default Gateway.

PC-1 → Desktop>IP Configuration

IP Address: 192.168.1.10
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.1.1

Close the PC-1 window

Note: Configure the other PC's

Part-3(F): Verify network connectivity.

Use the ping command to verify network connectivity with the router. If ping replies are not successful troubleshoot the connection:

Part-3(G): Verify the running configuration and save this configuration to NVRAM

```
R1#show startup configuration

--- more ---

R1#show running-config

--- more ---


R1# copy running-config startup-config

Destination filename [startup-config]? <ENTER>

Building configuration...

[OK]
```

Part-4: Write the Answer

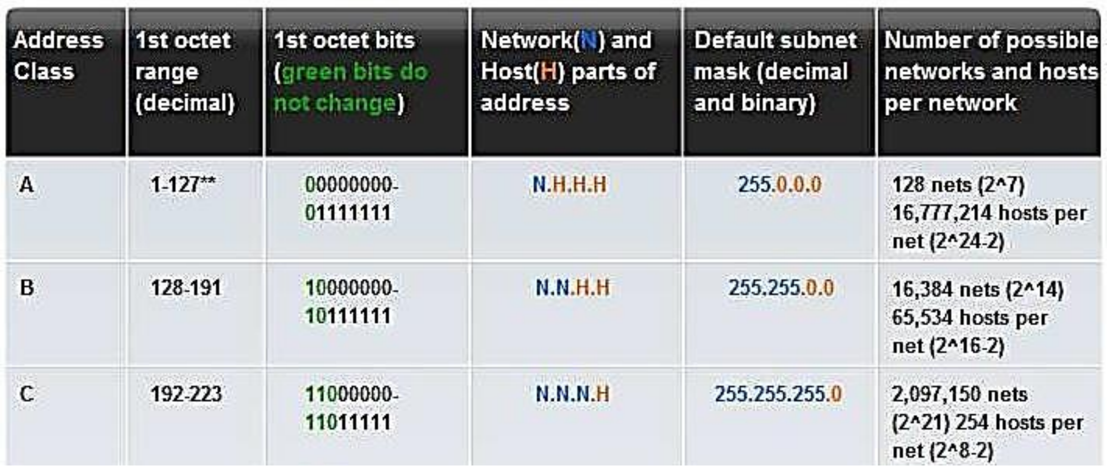
- What are the four primitive modes of Router IOS? What are the purposes of each mode?
- Why we use "TAB" and "?" symbol during router configuration?
- What is the basic difference between "startup configuration" file and "running configuration" file?
- What is the output of "show run" command?
- Why and in which mode "copy run start" command is used?
- Why "no shutdown" command is used?

Lab -3 - IP Addressing and Subnetting

Objectives

- To introduce IPv4 addressing and subnetting.
- To design an enterprise network IP addressing scheme.

Part-1: Introduction to IPv4 Address Class



Address Class	1st octet range (decimal)	1st octet bits (green bits do not change)	Network(N) and Host(H) parts of address	Default subnet mask (decimal and binary)	Number of possible networks and hosts per network
A	1-127**	00000000-01111111	N.H.H.H	255.0.0.0	128 nets (2^7) 16,777,214 hosts per net (2^24-2)
B	128-191	10000000-10111111	N.N.H.H	255.255.0.0	16,384 nets (2^14) 65,534 hosts per net (2^16-2)
C	192-223	11000000-11011111	N.N.N.H	255.255.255.0	2,097,150 nets (2^21) 254 hosts per net (2^8-2)

Part-2: Find the answers

Use the above IPv4 address table, determine the class, network address and broadcast address for the following IP addresses:

Part-2(A)

1. **Address: 207.21.54.240**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Number of usable Host:
Default Network Address: Default Broadcast Address:

2. **Address: 60.41.211.5**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Number of usable Host:
Default Network Address: Default Broadcast Address:

3. **Address: 190.101.2.199**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Number of usable Host:
Default Network Address: Default Broadcast Address:

Part-2(B)

1. **Address: 207.21.54.140** **Subnet Mask: 255.255.255.224**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Subnet Bit:
Network Address: Broadcast Address:
Total Number of Subnet: Total Usable IP Address per Subnet:

2. **Address: 60.41.211.5** **Subnet Mask: 255.255.255.0**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Subnet Bit:
Network Address: Broadcast Address:
Total Number of Subnet: Total Usable IP Address per Subnet:

3. **Address: 182.191.25.11** **Subnet Mask: 255.255.254.0**

Address Class: Class Host Bit: Class Network Bit:
Default Subnet Mask: Total Subnet Bit:
Network Address: Broadcast Address:
Total Number of Subnet: Total Usable IP Address per Subnet:

Part-3: Design an enterprise network IP addressing scheme

Part-3(A)

Suppose you are a network administrator and you will design your network with a new address scheme. For this design, you need **5 sub-networks** and **25 usable IP address** for each subnet network. You chose a Class-C address like **192.168.1.0/24**.

Now, calculate and Find the answers:

Address Class of the Address 192.168.1.0/24. Answers:
Host and Network bits in this given Network Answers:
How many bits must be borrowed from Host bit? Answers:
How many subnets does this create? Answers:
How many usable hosts does this create per subnet? Answers:

Total Network Bit:

Subnet Mask:

Part-3(B)

Subnet Number	Subnet Address	First Usable Host Address	Last Usable Host Address	Broadcast Address

You have the IP address 186.111.0.0, this network is subnetted by 10-bits.

Home Work

Part-3(C)

Find the following:
1. Find the Subnet Mask.
2. Determine the number of usable hosts per subnet.
3. To which subnet the following IP's belong to: 186.111.169.213
4. Determine the network address and broadcast address of the subnet to which this ip belongs to: 186.111.169.213.
5. Find Network Address, Broadcast Address and Host Range for the subnet # 121

Part-3(D)
Given a host with IP address 160.50.145.189/21:

1. Is a host with IP address 160.50.146.210/21 part of the same network? Show calculations.
2. Is the IP address 160.50.145.255 valid according to the given IP? Why or why not?
3. What is the first valid host on the sub-network that the node 172.18.142.179 255.255.254.0 belongs to?
4. Which subnet does host 192.168.11.198 255.255.255.240 belong to?
5. What is the last valid host on the sub-network 192.168.98.176 255.255.255.240?
6. What is the last valid host on the sub-network 172.25.13.112 255.255.255.240?
7. How many subnets and hosts per subnet can you get from the network 10.0.0.0 255.255.240.0?
8. What is the first valid host on the sub-network that the node 192.168.207.190/28 belongs to?

Lab-4 – Internetworking: Static and Default Routing Concept

Objectives

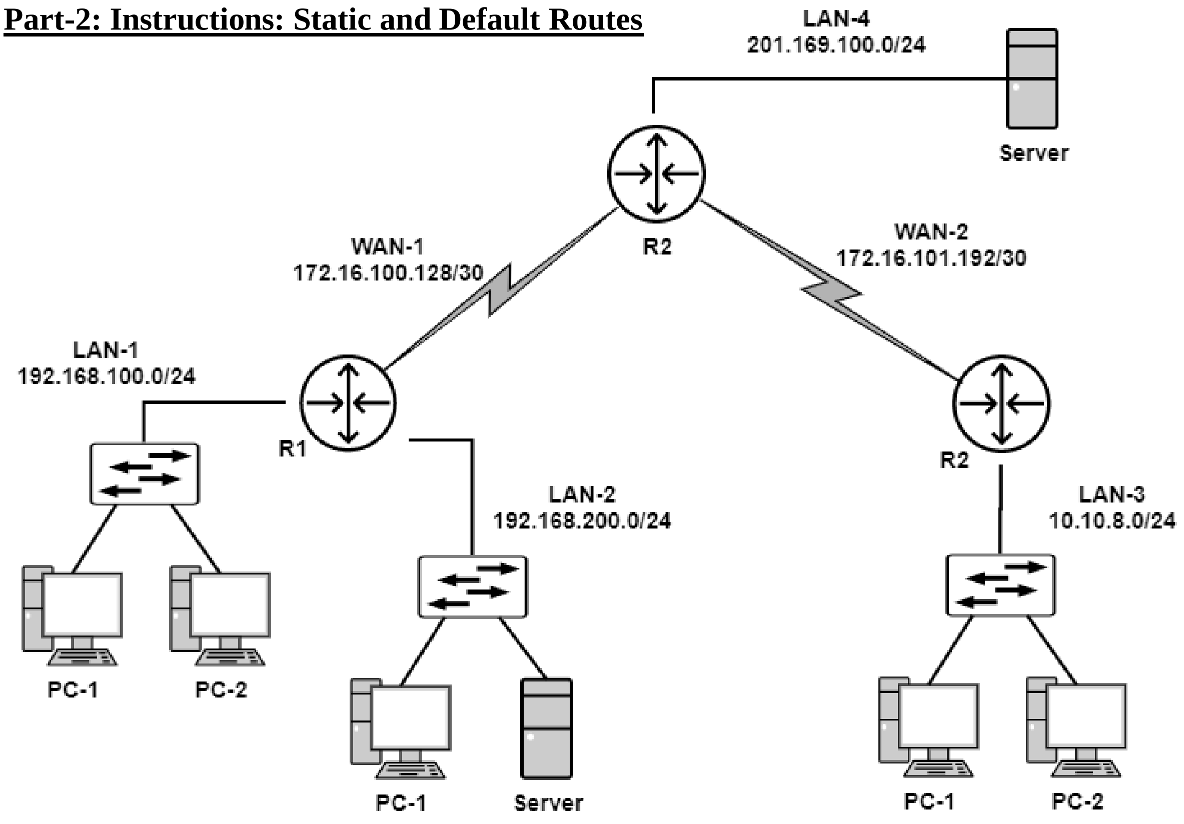
- To learn how to connect and configure more than one Router
- Using static and default routes
- Using a dynamic routing protocol (RIP)
- To build and configure an internetwork using Packet Tracer

Part-1: Background: Static and Default Routes

In this activity, you will configure static and default routes. A static route is a route that is entered manually by the network administrator to create a reliable and safe route. A directly attached static route relies on its exit interface in order for packets to be sent to its destination, while a recursive static route uses the IP address of the next hop router.

A default route, also known as the gateway of last resort, is the network route used by a router when no other known route exists for a destination network. A static route is used to route traffic to a specific network, while a default route is used when destination network is unknown.

Part-2: Instructions: Static and Default Routes



Question: How many Networks In this topology? Answers:

Part-2(A): Routing Table

```
R1 # show ip route
```

Part-2(B): Write the unknown network of Router R1, R2 and R3

Router R1 unknown network	Next_Hop_IP_Address to go this Network
201.169.100.0/24	172.16.100.130 (WAN-1 R2 Side IP Address :)
10.10.8.0/24	172.16.100.130
172.16.101.192/30	172.16.100.130

Router R2 unknown network	Next_Hop_IP_Address to go this Network

Router R3 unknown network	Next_Hop_IP_Address to go this Network

Part-2(C): Configure Static & Default Routes Command Syntax

Static Route: Syntax:

```
Router (config) # ip route [Dest_Net_Address] [Net_Mask] [Next_Hop_IP_Address]
```

Default Static Route: Syntax:

```
Router (config) # ip route 0.0.0.0 0.0.0.0 [Next_Hop_IP_Address]
```

Part-2(D): Static & Default Routes Command

```
R1(config)#ip route 201.169.100.0 255.255.255.0 172.16.100.130
R1(config)#ip route 172.16.101.192 255.255.255.252 172.16.100.130
R1(config)#ip route 10.10.8.0 255.255.255.0 172.16.100.130


R2(config)#ip route 10.10.8.0 255.255.255.0 172.16.101.194
R2(config)#ip route 192.168.100.0 255.255.255.0  172.16.100.129
R2(config)#ip route 192.168.200.0 255.255.255.0  172.16.100.129


R3(config)#ip route 0.0.0.0 0.0.0.0 172.16.101.193
```

Part-2(E): Verify Routing table and Test the network and Save to the NVRAM

```
R1 # show ip route
R2 # show ip route
R3 # show ip route
```

Check your routing table for entries that are preceded by a capital letter **"S"** for static Route and **"S * "** for Default Static Route.

When you are finished with the routing configuration, return to privileged EXEC mode and save the current configuration to NVRAM.

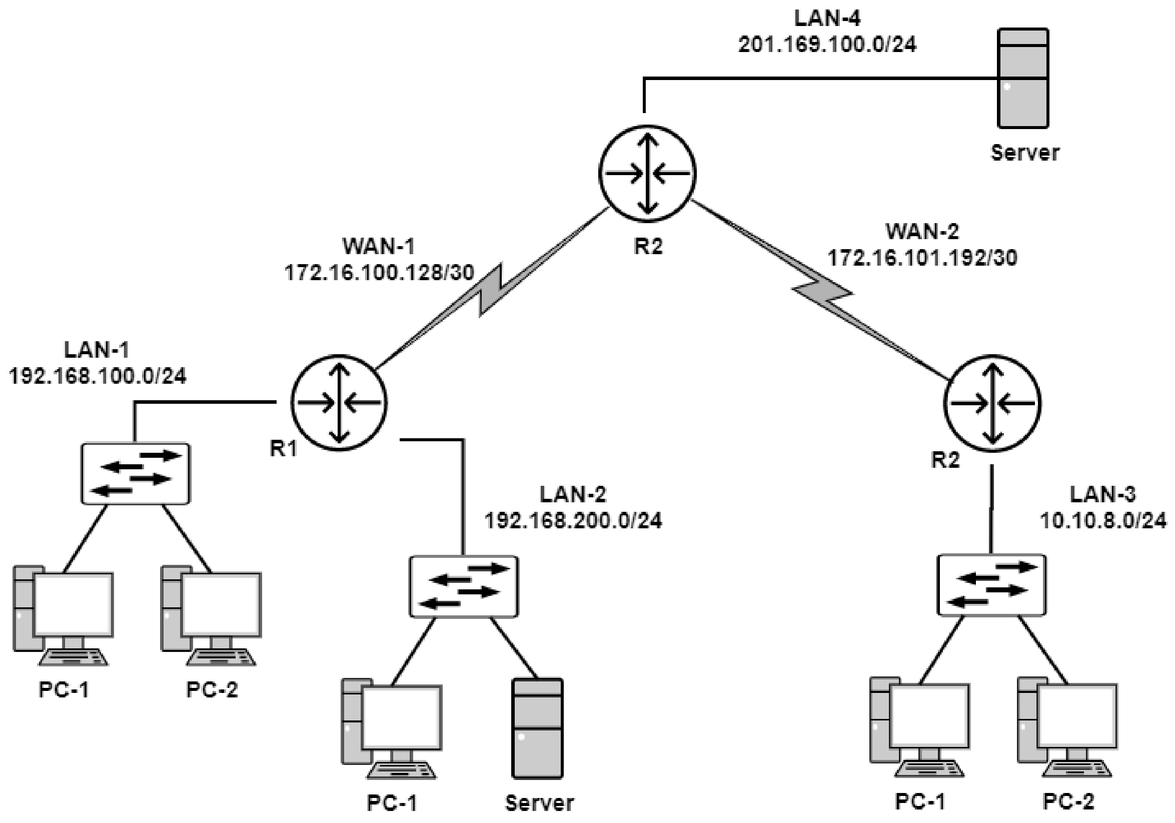
```
R1#copy run start
```

Lab-5 – Internetworking: Dynamic Routing (RIP and OSPF) Concept

Objectives

- Types of Dynamic Routing Protocols
- Difference between RIP and OSPF
- Configure and verify Dynamic Routing (i.e. RIP and OSPF)
- Verify the Routing Table and analysis.

Routing Topology



Part-A: Dynamic Routing: RIP

Part-1: Background: RIP (Routing Information Protocol)

In this activity, you will configure a simple dynamic routing protocol named **Routing Information Protocol (RIP).** RIP is a relatively old but still commonly used interior gateway protocol created for use in small, homogeneous networks. It is a classical **distance-vector** routing protocol.

Part-2: Configuring Routing Information Protocol (RIP)

For this router to participate in a dynamic routing using a dynamic routing protocol like RIP, you'll need to enable a routing protocol and advertise **the directly connected networks** that want advertised.

Part-3: Routing Table of the routers

```
R1 # show ip route
```

<u>Part-4: Write the Connect network of Router R1, R2 and R3</u>

Router R1 Directly Connected Networks

192.168.100.0/24

192.168.200.0/24

172.16.100.128/30

Router R2 Directly Connected Networks

Router R3 Directly Connected Networks

<u>Part-5: Enable RIP and Create RIP Table</u>

To enable a dynamic routing protocol, enter global configuration mode and use the router command. Enter "**router ?"** at the global configuration prompt to a see a list of available routing protocols on your router. To enable RIP, enter the command router rip in global configuration mode. Once you are in routing configuration mode, enter the network address for **each directly connected network**,

Command Syntax,

```
Router (config) # router rip
Router (config-router) # network <Connected Network>
Router (config-router) # exit
```

Command:

```
R1 (config) # router rip
R1 (config-router) # network 192.168.100.0
R1 (config-router) # network 192.168.200.0
R1 (config-router) # network 172.16.100.128
R1 (config-router) # exit
R1 (config) #
```

<u>Part-6: Enable RIP version 2 for Classless Network Address</u>

```
        R1 (config) # router rip
        R1 (config-router) # rip version 2
        R1 (config-router) # exit
```

Note: Now you configure the R2 and R3 Router.

Part-7: Verify Routing table and Test the network and Save to the NVRAM

```
        R1 # show ip route
        R2 # show ip route
        R3 # show ip route
```

Check your routing table for entries that are preceded by a capital letter **"R"** to ensure that you are receiving routing updates using RIP. Use show ip route to see the routing table. Ensure that both routers configured so that you can receive his updates. No updates, no ping

When you are finished with the routing configuration, return to privileged EXEC mode and save the current configuration to NVRAM.

```
            R1#copy run start
```

Part-B: Dynamic Routing: OSPF

Part-1: Background: OSPF (Open Shortest Path First)

Open Shortest Path First (OSPF) is a routing protocol for Internet Protocol (IP) networks. It uses a link state routing (LSR) algorithm and falls into the group of interior gateway protocols (IGPs), operating within a single autonomous system (AS).

Part-2: Configuring OSPF (Open Shortest Path First)

For OSPF, you'll need to enable a routing protocol and advertise **the directly connected networks** that want advertised. We also need an **Area number**, **OSPF Process ID** and **Router ID**.

Part-3: OSPF Process ID, Router ID and Area

The Process ID is locally significant and is needed to identify a unique instance of an OSPF database. Process ID number can be from 0 to 65535.

The router ID is a 32-bit number that identifies the OSPF router. Just like IPv4 addresses, it's an 8-bit decimal number separated by ". (dot)" and then write four of them in a row. **The router ID is the name of the OSPF router**, so to speak, and a unique router ID is always required for OSPF processing. The router ID recognizes the neighbor. The LSA is also marked with the router ID of the generated router.

OSPF has two types, one is single area OSPF and Multi area OSPF. In this course we cover the single area OSPF. In single area OSPF we use **area 0**.

Part-4: Enable OSPF and Create OSPF Table

Based on **Part-4 of Part-A**, to enable OSPF, enter the command “Router OSPF” with OSPF process ID in global configuration mode. Once you are in routing configuration mode, enter the network address for **each directly connected network**,

Command Syntax,

```
Router (config) # router ospf <Process ID>
Router (config-router) # router-id <Router ID>
Router (config-router) # network <Connected Network> <Wildcard Mask> <Area>
Router (config-router) # exit
```

Command:

```
R1 (config) # router ospf 100
R1 (config-router) # router-id 100.100.100.1
R1 (config-router) # network 192.168.100.0 0.0.0.255 area 0
R1 (config-router) # network 192.168.200.0 0.0.0.255 area 0
R1 (config-router) # network 172.16.100.128 0.0.0.3 area 0
R1 (config-router) # exit
R1 (config) #
```

Note: Now you configure the R2 and R3 Router.

Part-5: Verify Routing table and Test the network

Follow the **Part-7 of Part-A**

Lab-6 (Part-1) Dynamic Host Configuration Protocol (DHCP)

Objectives

- To learn about Dynamic Host Configuration Protocol (DHCP): why and how used?
- To build an internetwork and configure DHCP using Packet Tracer

Part-1: Background

Dynamic Host Control Protocol (DHCP) enables you to automatically assign reusable IP addresses to **DHCP clients**. The **DHCP Server** feature is a full **DHCP** server implementation that assigns and manages IP addresses from specified address pools within the router to DHCP clients. **Figure 1** shows the basic steps that occur when a DHCP client requests an IP address from a DHCP server. The client, Host A, sends a **DHCPDISCOVER** broadcast message to locate a DHCP Server. A DHCP server offers configuration parameters (such as an IP address, a MAC address, a domain name, and a lease for the IP address) to the client in a **DHCPOFFER** unicast message.

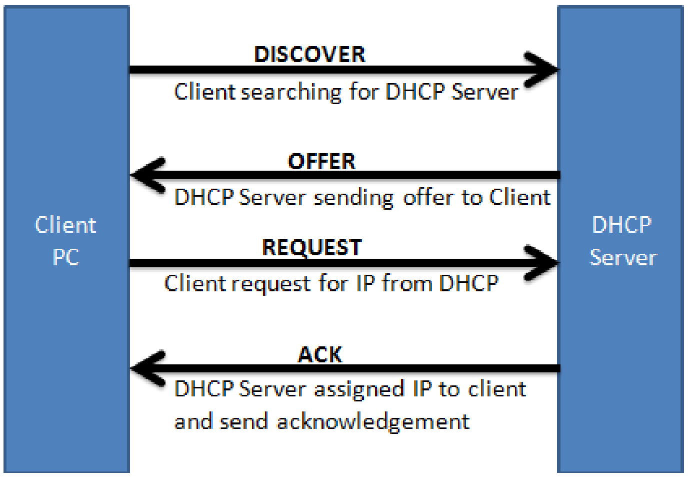

Client PC
DISCOVER
Client searching for DHCP Server
OFFER
DHCP Server sending offer to Client
REQUEST
Client request for IP from DHCP
ACK
DHCP Server assigned IP to client and send acknowledgement
DHCP Server

Figure 1: DHCP client/server mode

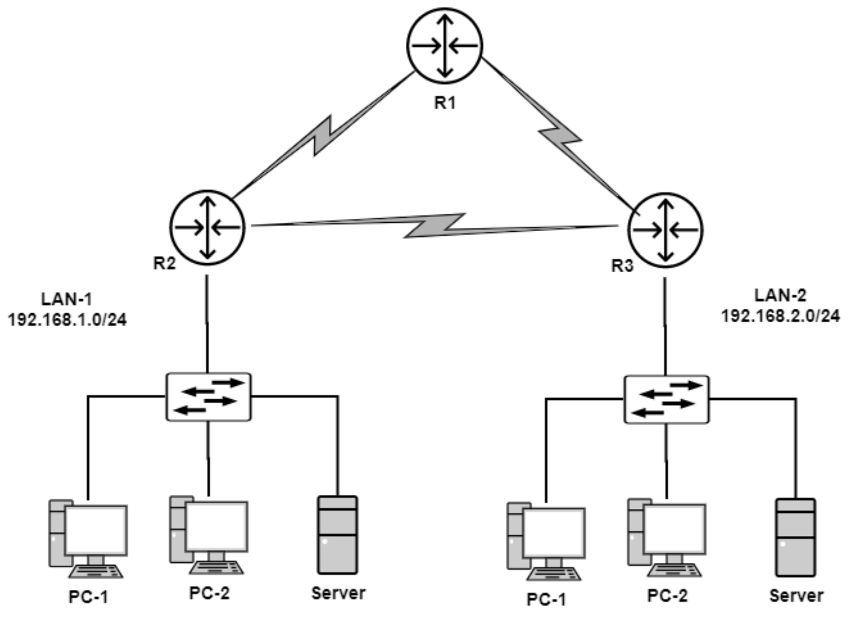
Benefits of use DHCP

Reduced Internet access costs: Using automatic IP address assignment at each remote site substantially reduces Internet access costs. Static IP addresses are considerably more expensive to purchase than are automatically allocated IP addresses.

Reduced client configuration tasks and costs: Because DHCP is easy to configure, it minimizes operational overhead and costs associated with device configuration tasks and eases deployment by nontechnical users.

Centralized management: Because the DHCP server maintains configurations for several subnets, an administrator only needs to update a single, central server when configuration parameters change.

Part-2: DHCP Server Configuration



Part-2(A): DHCP IP POOL

We configure DHCP in LAN-1 Network and choice any name for IP POOL. But as a network professional we maintain some DHCP Pool create rules. We take the 1st the Router name and then the interface name. Example, LAN-1 network DHCP POOL name: **R2G0/0**

```
R2 (config)# ip dhcp pool R2G0/0
```

Specify the subnet to use when assigning IP addresses. DHCP pools automatically associate with an interface based on the network statement.

```
R2 (config-dhcp)# network 192.168.1.0 255.255.255.0
```

Configure the **default router** and **domain name** server for the network. Clients receive these settings via DHCP, along with an IP address.

```
R2 (config-dhcp)# default-router 192.168.1.1
R2 (config-dhcp)# dns-server 8.8.8.8
R2 (config-dhcp)#exit
```

Part-2(B): Exclude statically assigned addresses

The DHCP server assumes that all IP addresses in a DHCP address pool subnet are available for assigning to DHCP clients. You must specify the IP addresses that the DHCP server should not assign to clients. These IP addresses are usually static addresses reserved for the router interface, switch management IP address, servers, and local network printer. The ip dhcp excluded-address command prevents the router from assigning IP addresses within the configured range. The following commands exclude the first 10 IP addresses from each pool for the LANs attached to R1. These addresses will not be assigned to any DHCP clients.

```
R2(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.10
```

Part-2(C): Test the DHCP configuration

- Select **DHCP** option to set PC-1 and PC-2 to get dynamic IP from the **DHCP Server.**
- Configure the **Server** with the **static** address **192.168.1.5**.

It is assumed that the server requires a **fixed IP address** in order to allow access from any PC in the LAN. Configure the server address manually. Ping from the PC2 and PC3 to router ports and server host.

Part-3: Configure other Network

Configure LAN-2 Network DHCP

Lab-6 (Part-2): Introduction to Wireshark

Objectives

- To learn about **Wireshark.**
- Packet Capture & Traffic Analysis
- Analyze the Packets using Wireshark

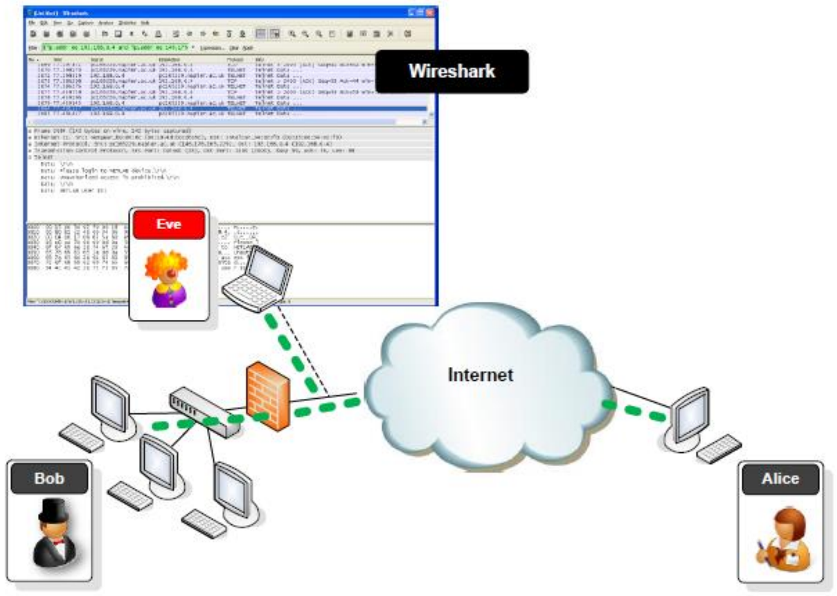
Part-1: Wireshark

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Develop by The Wireshark team. Website, www.wireshark.org

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License.

Part-2: Packet Capture/Packet Sniffing

A packet sniffer is an application which can capture and analyses network traffic which is passing through a system's Network Interface Card (NIC). The sniffer sets the card to promiscuous mode which means all traffic is read, whether it is addressed to that machine or not. The figure below shows an attacker sniffing packets from the network, and the Wireshark packet sniffer/analyzer (formerly known as ethereal).



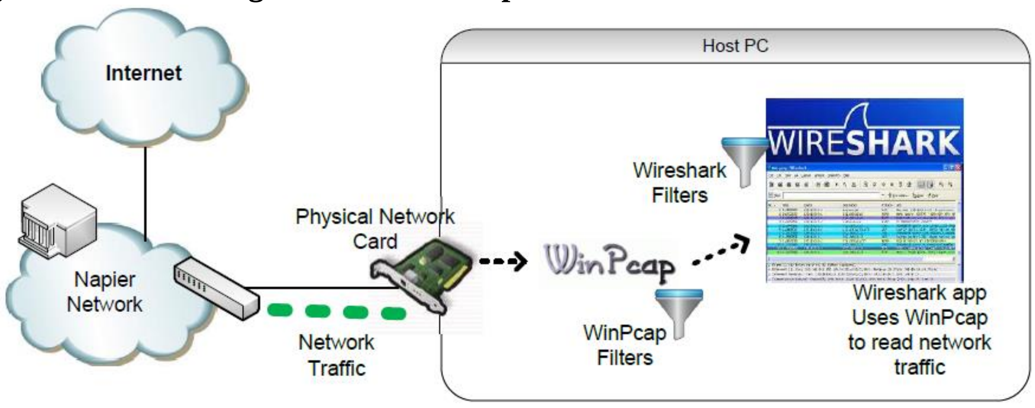
Part-3: Packet Analysis

Wireshark is an open source cross-platform packet capture and analysis tool, with versions for Windows and Linux. The GUI window gives a detailed breakdown of the network protocol stack for each packet, colorizing packet details based on protocol, as well as having functionality to filter and search the traffic, and pick out TCP streams. Wireshark can also save packet data to files for offline analysis and export/import packet captures to/from other tools. Statistics can also be generated for packet capture files.

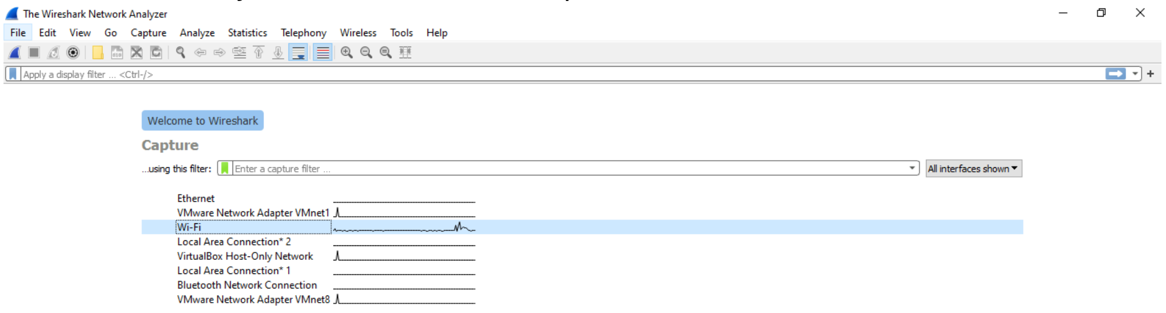
Wireshark can be used for network troubleshooting, to investigate security issues, and to analyses and understand network protocols. The packet sniffer can exploit information passed in plaintext, i.e. not encrypted. Examples of protocols which pass information in plaintext are Telnet, FTP, SNMP, POP, and HTTP.

Wireshark is a GUI based network capture tool. There is a command line based version of the packet capture utility, called TShark. TShark provides many of the same features as it's big brother, but is console-based. It can be a good alternative if only command line access is available, and also uses less resources as it has no GUI to generate.

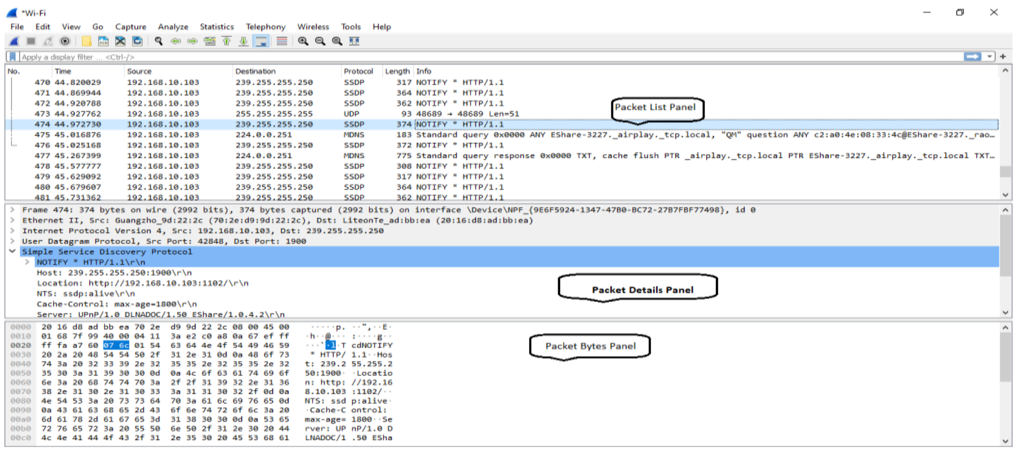
Logical View for Using Wireshark to Capture Traffic



Start the Wireshark application. When Wireshark is run, all network interface are shown in screen. Select any network interface for capture traffic i.e. Wi-Fi


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
All interfaces shown
Ethernet
VMware Network Adapter VMnet1
Wi-Fi
Local Area Connection* 2
VirtualBox Host-Only Network
Local Area Connection* 1
Bluetooth Network Connection
VMware Network Adapter VMnet8

Generate some network traffic with a Web Browser, such as Internet Explorer or Chrome. Your Wireshark window should show the packets, and now look something like.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Packet List Panel
Packet Details Panel
Packet Bytes Panel

```
No.   Time        Source          Destination      Protocol Length Info
  470 44.820029   192.168.10.103  239.255.255.250  SSDP     317 NOTIFY * HTTP/1.1
  471 44.869944   192.168.10.103  239.255.255.250  SSDP     364 NOTIFY * HTTP/1.1
  472 44.920788   192.168.10.103  239.255.255.250  SSDP     362 NOTIFY * HTTP/1.1
  473 44.927762   192.168.10.103  255.255.255.255  UDP       93 48689 → 48689 Len=51
  474 44.972730   192.168.10.103  239.255.255.250  SSDP     374 NOTIFY * HTTP/1.1
  475 45.016876   192.168.10.103  224.0.0.251      MDNS     183 Standard query 0x0000 ANY EShare-3227._airplay._tcp.local, "QM" question ANY c2:a0:4e:08:33:4c@EShare-3227._rao…
  476 45.025168   192.168.10.103  239.255.255.250  SSDP     372 NOTIFY * HTTP/1.1
  477 45.267399   192.168.10.103  224.0.0.251      MDNS     775 Standard query response 0x0000 TXT, cache flush PTR _airplay._tcp.local PTR EShare-3227._airplay._tcp.local TXT…
  478 45.577777   192.168.10.103  239.255.255.250  SSDP     308 NOTIFY * HTTP/1.1
  479 45.629092   192.168.10.103  239.255.255.250  SSDP     317 NOTIFY * HTTP/1.1
  480 45.679607   192.168.10.103  239.255.255.250  SSDP     364 NOTIFY * HTTP/1.1
  481 45.731362   192.168.10.103  239.255.255.250  SSDP     362 NOTIFY * HTTP/1.1
```

```
> Frame 474: 374 bytes on wire (2992 bits), 374 bytes captured (2992 bits) on interface \Device\NPF_{9E6F5924-1347-47B0-BC72-27B7FBF77498}, id 0
> Ethernet II, Src: Guangzho_9d:22:2c (70:2e:d9:9d:22:2c), Dst: LiteonTe_ad:bb:ea (20:16:d8:ad:bb:ea)
> Internet Protocol Version 4, Src: 192.168.10.103, Dst: 239.255.255.250
> User Datagram Protocol, Src Port: 42848, Dst Port: 1900
v Simple Service Discovery Protocol
  > NOTIFY * HTTP/1.1\r\n
    Host: 239.255.255.250:1900\r\n
    Location: http://192.168.10.103:1102/\r\n
    NTS: ssdp:alive\r\n
    Cache-Control: max-age=1800\r\n
    Server: UPnP/1.0 DLNADOC/1.50 EShare/1.0.4.2\r\n
```

```
0000  20 16 d8 ad bb ea 70 2e  d9 9d 22 2c 08 00 45 00   ·····p. ··",··E·
0010  01 68 7f 99 40 00 04 11  3a e2 c0 a8 0a 67 ef ff   ·h··@··· :····g··
0020  ff fa a7 60 07 6c 01 54  63 64 4e 4f 54 49 46 59   ···`·l·T cdNOTIFY
0030  20 2a 20 48 54 54 50 2f  31 2e 31 0d 0a 48 6f 73    * HTTP/ 1.1··Hos
0040  74 3a 20 32 33 39 2e 32  35 35 2e 32 35 35 2e 32   t: 239.2 55.255.2
0050  35 30 3a 31 39 30 30 0d  0a 4c 6f 63 61 74 69 6f   50:1900· ·Locatio
0060  6e 3a 20 68 74 74 70 3a  2f 2f 31 39 32 2e 31 36   n: http: //192.16
0070  38 2e 31 30 2e 31 30 33  3a 31 31 30 32 2f 0d 0a   8.10.103 :1102/··
0080  4e 54 53 3a 20 73 73 64  70 3a 61 6c 69 76 65 0d   NTS: ssd p:alive·
0090  0a 43 61 63 68 65 2d 43  6f 6e 74 72 6f 6c 3a 20   ·Cache-C ontrol:
00a0  6d 61 78 2d 61 67 65 3d  31 38 30 30 0d 0a 53 65   max-age= 1800··Se
00b0  72 76 65 72 3a 20 55 50  6e 50 2f 31 2e 30 20 44   rver: UP nP/1.0 D
00c0  4c 4e 41 44 4f 43 2f 31  2e 35 30 20 45 53 68 61   LNADOC/1 .50 ESha
```

To stop the capture, select the **Capture->Stop** menu option, Ctrl+E, or the Stop toolbar button. What you have created is a Packet Capture or ***'pcap'***, which you can now view and analyses using the Wireshark interface, or save to disk to analyses later.

The capture is split into 3 parts:
Packet List Panel – this is a list of packets in the current capture. It colours the packets based on the protocol type. When a packet is selected, the details are shown in the two panels below.

Packet Details Panel – this shows the details of the selected packet. It shows the different protocols making up the layers of data for this packet. Layers include Frame, Ethernet, IP, TCP/UDP/ICMP, and application protocols such as HTTP.

Packet Bytes Panel – shows the packet bytes in Hex and ASCII encodings.

Part-4: Analyze the Packets using Wireshark

Run Wireshark and Go to www.facebook.com

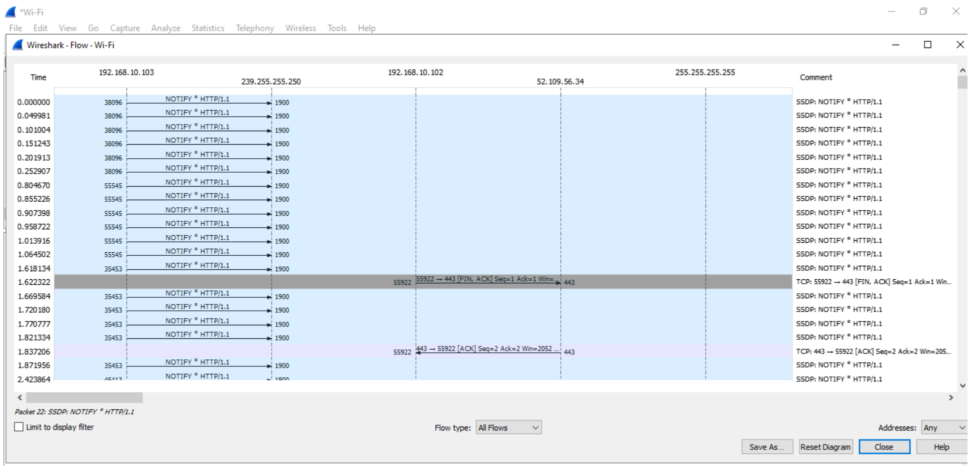
Write the filter option,

ip.addr == [your Src ip] and ip.addr == [your des. IP]

tcp.port eq 80 and no dns

tcp.port == 443 and no arp

Check the packets and analysis,
Now open **Statistics->Flow Graph** menu option



Time	Comment
0.000000	SSDP: NOTIFY * HTTP/1.1
0.049981	SSDP: NOTIFY * HTTP/1.1
0.101004	SSDP: NOTIFY * HTTP/1.1
0.151243	SSDP: NOTIFY * HTTP/1.1
0.201913	SSDP: NOTIFY * HTTP/1.1
0.252907	SSDP: NOTIFY * HTTP/1.1
0.804670	SSDP: NOTIFY * HTTP/1.1
0.855226	SSDP: NOTIFY * HTTP/1.1
0.907398	SSDP: NOTIFY * HTTP/1.1
0.958722	SSDP: NOTIFY * HTTP/1.1
1.013916	SSDP: NOTIFY * HTTP/1.1
1.064502	SSDP: NOTIFY * HTTP/1.1
1.618134	SSDP: NOTIFY * HTTP/1.1
1.622322	TCP: 55922 → 443 [FIN, ACK] Seq=1 Ack=1 Win...
1.669584	SSDP: NOTIFY * HTTP/1.1
1.720180	SSDP: NOTIFY * HTTP/1.1
1.770777	SSDP: NOTIFY * HTTP/1.1
1.821334	SSDP: NOTIFY * HTTP/1.1
1.837206	TCP: 443 → 55922 [ACK] Seq=2 Ack=2 Win=205...
1.871956	SSDP: NOTIFY * HTTP/1.1
2.423864	SSDP: NOTIFY * HTTP/1.1

Packet 22: SSDP: NOTIFY * HTTP/1.1

Lab-6 (Part-3): Wireshark: Examine HTTP and HTTPS Traffic

Objectives

- Capture and view HTTP traffic and HTTPS traffic

Background / Scenario

Hypertext Transfer Protocol (HTTP) is an application layer protocol that presents data via a web browser. With HTTP, there is no safeguard for the exchanged data between two communicating devices.

With HTTPS, encryption is used via a mathematical algorithm. This algorithm hides the true meaning of the data that is being exchanged. This is done through the use of certificates that can be viewed later in this lab.

Regardless of HTTP or HTTPS, it is only recommended to exchange data with websites that you trust. Just because a site uses HTTPS does not mean it is a trustworthy site. Threat actors commonly use HTTPS to hide their activities.

In this lab, you will explore and capture HTTP and HTTPS traffic using Wireshark.

Part-1: Examine HTTP Traffic

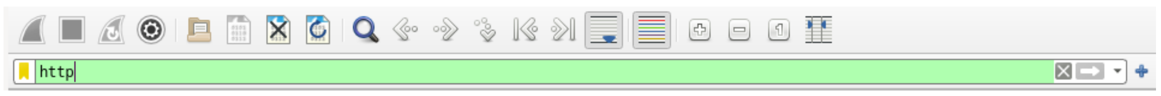
Run Wireshark and also web browser.

a. Open a web browser and go to this site **http://www.altoromutual.com/login.jsp**

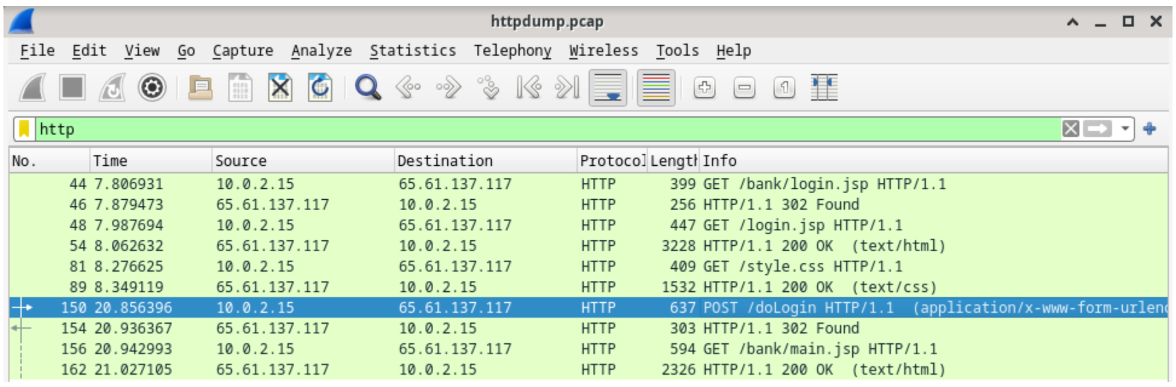
b. Enter a username of **Admin** with a password of “**UIU@123” or “Admin”** and click **Login**.

c. Close the web browser.

d. In the Wireshark application, filter for **http** and click **Apply**.

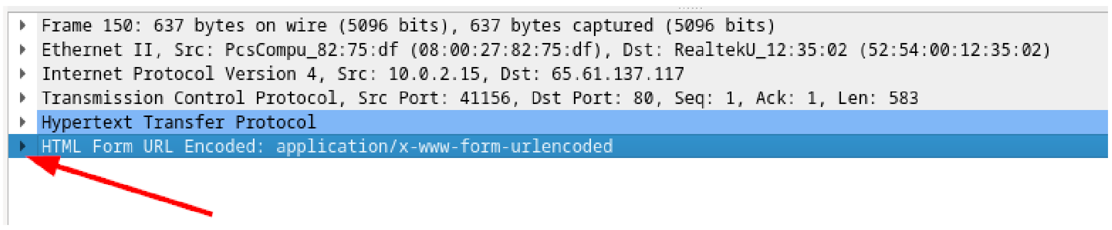


e. Browse through the different HTTP messages and select the **POST** message.


httpdump.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http

```
No.    Time       Source         Destination    Protocol Length Info
 44 7.806931   10.0.2.15      65.61.137.117  HTTP    399 GET /bank/login.jsp HTTP/1.1
 46 7.879473   65.61.137.117  10.0.2.15      HTTP    256 HTTP/1.1 302 Found
 48 7.987694   10.0.2.15      65.61.137.117  HTTP    447 GET /login.jsp HTTP/1.1
 54 8.062632   65.61.137.117  10.0.2.15      HTTP   3228 HTTP/1.1 200 OK  (text/html)
 81 8.276625   10.0.2.15      65.61.137.117  HTTP    409 GET /style.css HTTP/1.1
 89 8.349119   65.61.137.117  10.0.2.15      HTTP   1532 HTTP/1.1 200 OK  (text/css)
150 20.856396  10.0.2.15      65.61.137.117  HTTP    637 POST /doLogin HTTP/1.1  (application/x-www-form-urlen
154 20.936367  65.61.137.117  10.0.2.15      HTTP    303 HTTP/1.1 302 Found
156 20.942993  10.0.2.15      65.61.137.117  HTTP    594 GET /bank/main.jsp HTTP/1.1
162 21.027105  65.61.137.117  10.0.2.15      HTTP   2326 HTTP/1.1 200 OK  (text/html)
```

f. In the lower window, the message is displayed. Expand the **HTML Form URL Encoded: application/x-www-form-urlencoded** section.



```
Frame 150: 637 bytes on wire (5096 bits), 637 bytes captured (5096 bits)
Ethernet II, Src: PcsCompu_82:75:df (08:00:27:82:75:df), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 65.61.137.117
Transmission Control Protocol, Src Port: 41156, Dst Port: 80, Seq: 1, Ack: 1, Len: 583
Hypertext Transfer Protocol
HTML Form URL Encoded: application/x-www-form-urlencoded
```

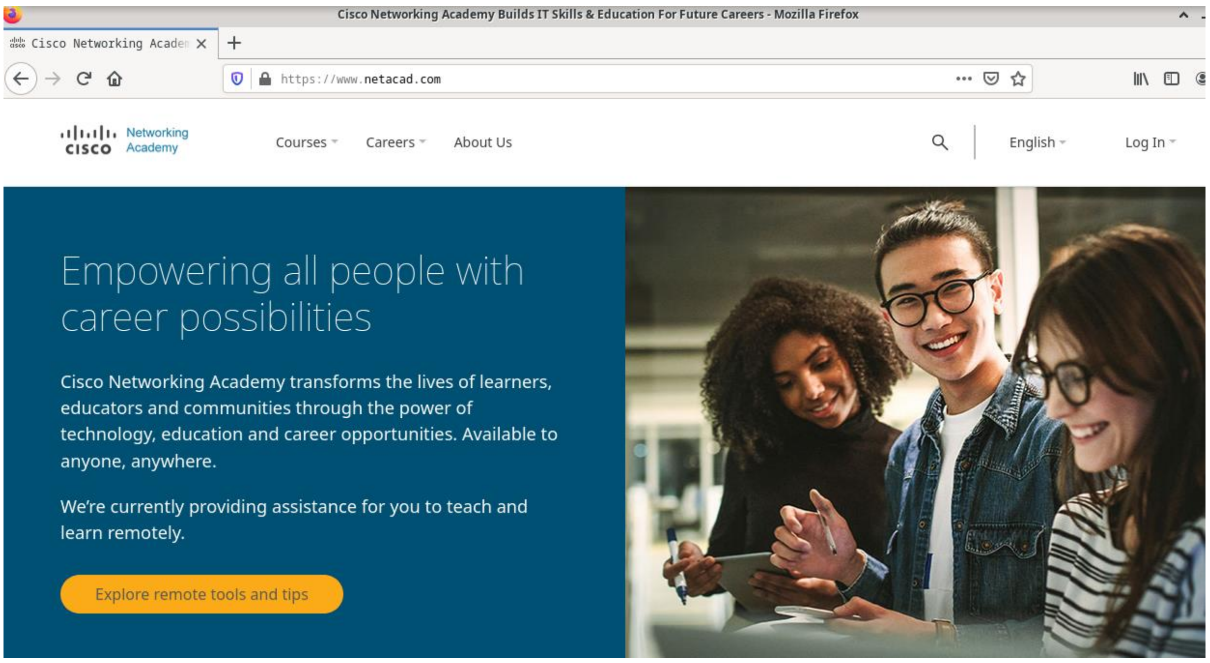
Question: What two pieces of information are displayed?

g. Close the Wireshark application.

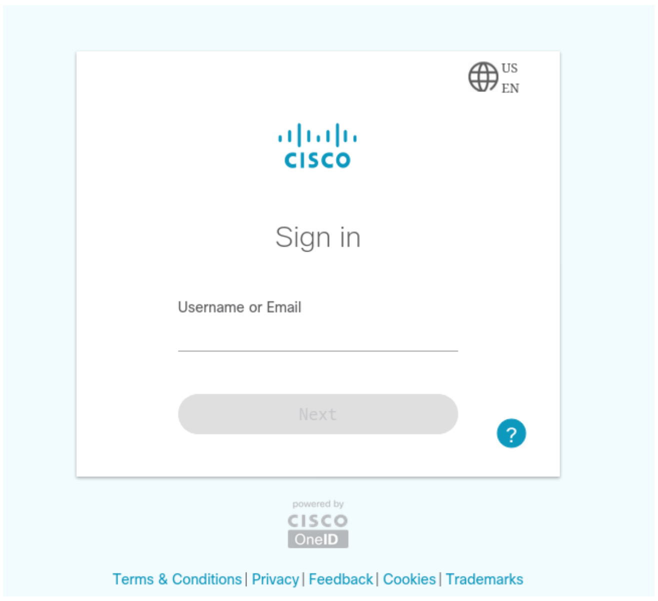
Part-2: Examine HTTPs Traffic

Open a web browser and www.netacad.com

a. Click **Log in**.


Cisco Networking Academy Builds IT Skills & Education For Future Careers - Mozilla Firefox
Cisco Networking Acader
https://www.netacad.com
Networking Academy
Courses
Careers
About Us
English
Log In
Empowering all people with career possibilities
Cisco Networking Academy transforms the lives of learners, educators and communities through the power of technology, education and career opportunities. Available to anyone, anywhere.
We're currently providing assistance for you to teach and learn remotely.
Explore remote tools and tips

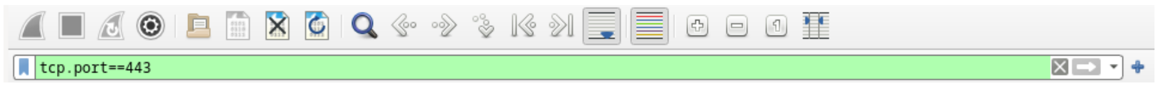
b. Enter in your NetAcad username and password. Click **Next**.


US EN
cisco
Sign in
Username or Email
Next
?
powered by
cisco
OneID
Terms & Conditions | Privacy | Feedback | Cookies | Trademarks

c. Close the web browser.

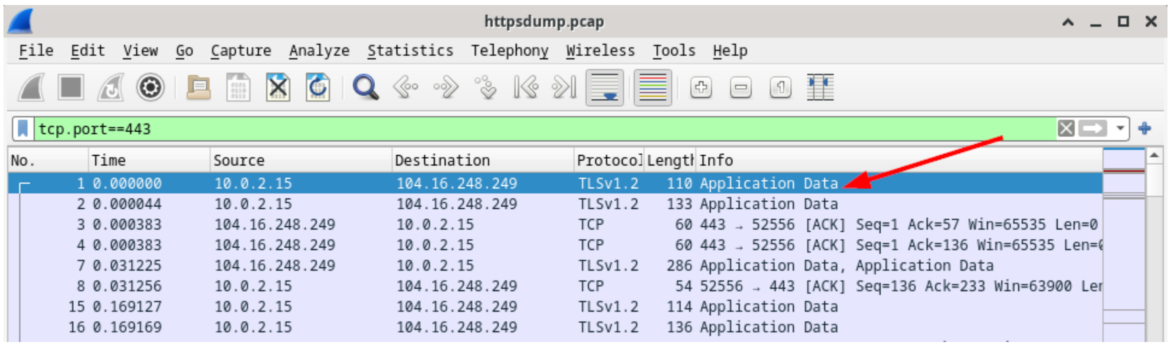
h. In the Wireshark application, expand the capture window vertically and then filter by HTTPS traffic via port 443.

Enter **tcp.port==443** as a filter, and click **Apply**.



```
tcp.port==443
```

i. Browse through the different HTTPS messages and select an **Application Data** message.


httpsdump.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

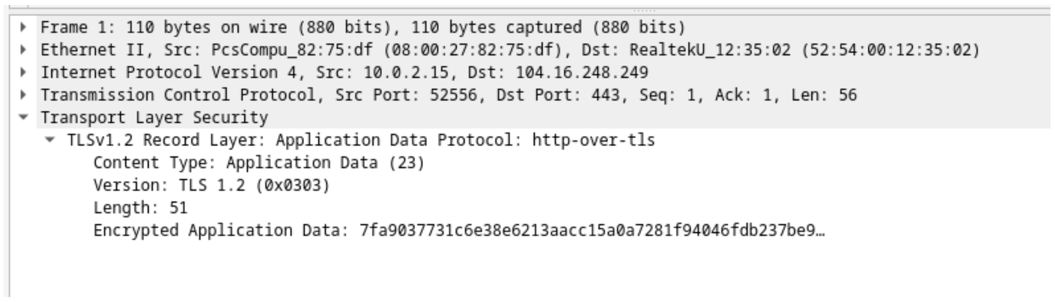
```
tcp.port==443
```

```
No.   Time      Source          Destination     Protocol Length Info
    1 0.000000  10.0.2.15       104.16.248.249  TLSv1.2   110 Application Data
    2 0.000044  10.0.2.15       104.16.248.249  TLSv1.2   133 Application Data
    3 0.000383  104.16.248.249  10.0.2.15       TCP        60 443 → 52556 [ACK] Seq=1 Ack=57 Win=65535 Len=0
    4 0.000383  104.16.248.249  10.0.2.15       TCP        60 443 → 52556 [ACK] Seq=1 Ack=136 Win=65535 Len=(
    7 0.031225  104.16.248.249  10.0.2.15       TLSv1.2   286 Application Data, Application Data
    8 0.031256  10.0.2.15       104.16.248.249  TCP        54 52556 → 443 [ACK] Seq=136 Ack=233 Win=63900 Ler
   15 0.169127  10.0.2.15       104.16.248.249  TLSv1.2   114 Application Data
   16 0.169169  10.0.2.15       104.16.248.249  TLSv1.2   136 Application Data
```

j. In the lower window, the message is displayed.

Question: What has replaced the HTTP section that was in the previous capture file?

k. Completely expand the **Secure Sockets Layer** section.



```
▸ Frame 1: 110 bytes on wire (880 bits), 110 bytes captured (880 bits)
▸ Ethernet II, Src: PcsCompu_82:75:df (08:00:27:82:75:df), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 104.16.248.249
▸ Transmission Control Protocol, Src Port: 52556, Dst Port: 443, Seq: 1, Ack: 1, Len: 56
▾ Transport Layer Security
  ▾ TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 51
      Encrypted Application Data: 7fa9037731c6e38e6213aacc15a0a7281f94046fdb237be9…
```

l. Click the **Encrypted Application Data**.

Question: Is the application data in a plaintext or readable format?

Reflection Questions

1. What are the advantages of using HTTPS instead of HTTP?

Ans:

2. Are all websites that use HTTPS considered trustworthy?

Ans:

Lab-7: Packet-level Firewalls – Access Control Lists (ACLs)

Objectives

- Use of Access Control Lists (ACLs) defined in Routers to control access in a network.
- Difference between Standard and Extended ACLs.
- Configuring and applying Standard and Extended ACLs in Cisco routers.

Part-1: Access Control Lists (ACLs)

The **Access Control List (ACL)** is a collection of security rules or policies that allows or denies packets after looking at the packet headers and other attributes. Each permit or deny statement in the ACL is referred to as an **access control entry (ACE).** These **ACE**s can classify packets by inspecting Layer 2 through Layer 4 headers for a number of parameters, including the following:

- Layer 2 header information such as Ether Types
- Layer 3 header information such as ICMP, TCP, or UDP
- Layer 3 header information such as source and destination IP addresses
- Layer 4 header information such as source and destination TCP or UDP ports

After an ACL has been properly configured, you can apply it to an interface to filter traffic. The security appliance can filter packets in both the inbound and outbound direction on an interface. When an inbound ACL is applied to an interface, the security appliance analyzes packets against the ACEs after receiving them. If a packet is permitted by the ACL, the firewall continues to process the packet and eventually passes the packet to the defined interface.

Two types of Access Control Lists are,

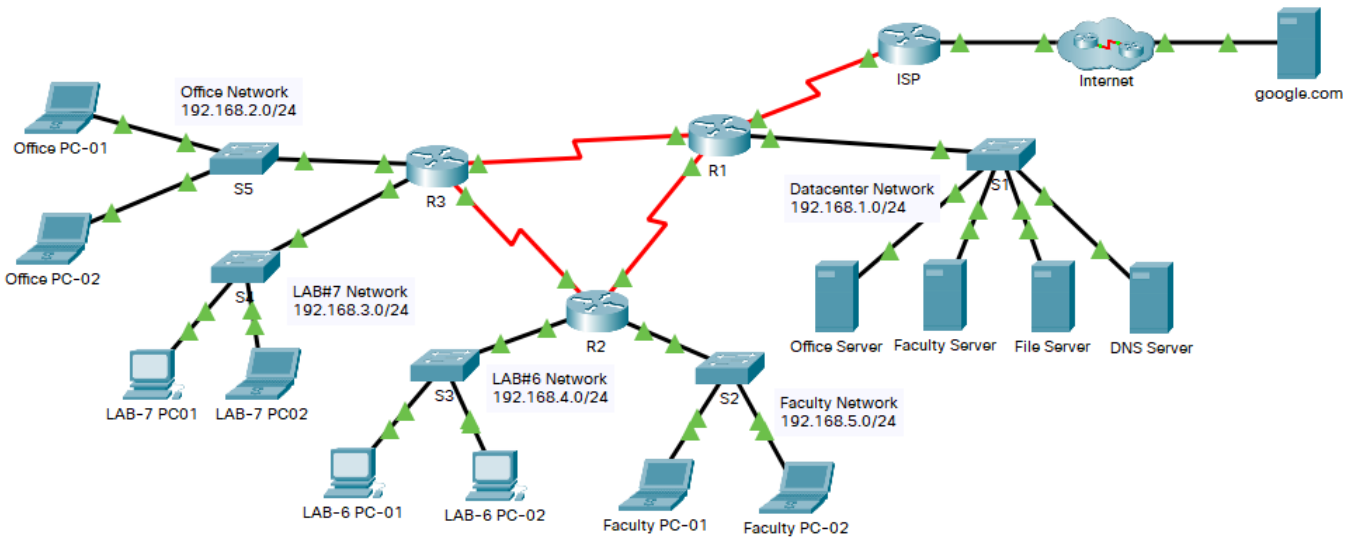
1. Standard Access Control Lists (ACLs):

Standard access control lists (ACLs) are router configuration scripts that control whether a **router permits or denies packets based on the source address only**.
Standard ACL defined using any number from 1 to 99.

2. Extended Access Control Lists (ACLs):

Extended access control lists (ACLs) are extremely powerful. They offer a much greater degree of control than standard ACLs as to the types of traffic that can be filtered, as well as where the traffic originated and where it is going. Extended ACLs can filter traffic in many different ways. **Extended ACLs can filter on source IP addresses, source ports, destination IP addresses, destination ports, as well as various protocols and services**.
Extended ACL defined using any number from 100 to 199.

Part-2: Access Control Lists (ACLs) Implementation



Part-2(A): Standard Type Access Control Lists (ACLs)

Create an ACL using the number on R1 (Why not on R2 or R3?) with a statement that denies access to the Datacenter Network (192.168.1.0/24) from the LAB#7 Network (192.168.3.0/24).

```
R1 (config)# access-list 1 deny 192.168.3.0 0.0.0.255
```

By default, an access list denies all traffic that does not match a rule. To permit all other traffic, configure the following statement:

```
R1 (config)# access-list 1 permit any
```

For the ACL to actually filter traffic, it must be applied to some router operation. Apply the ACL by placing it for outbound traffic on the Gigabit Ethernet 0/0 interface.

```
R1 (config)# interface GigabitEthernet0/0
R1 (config-if)# ip access-group 1 out
R1 (config-if)# exit
```

Prevent (Deny) LAB#7 PC01 to Access Datacenter Network.

```
R1(config)#access-list 2 deny host 192.168.3.10
R1(config)#access-list 2 permit any

R1(config)#int gigabitEthernet 0/0
R1(config-if)#ip access-group 2 out
R1(config-if)#exit
```

Prevent all Private IP Address to Access Internet (Please follow the order)

```
ISP(config)#access-list 10 deny 10.0.0.0 0.255.255.255
ISP(config)#access-list 10 deny 172.16.0.0 0.15.255.255
ISP(config)#access-list 10 deny 192.168.0.0 0.0.255.255
ISP(config)#access-list 10 permit any

ISP(config)#int serial 0/0/0
ISP(config-if)#ip access-group 10 in
ISP(config-if)#exit
```

Part-2(A): Extended Type Access Control Lists (ACLs)

LAB#6 network is not allowed to access Office Server using http/https/browsing, but All other Traffic is Permitted for Lab#6 Network.

```
R2(config)#access-list 100 deny tcp 192.168.4.0 0.0.0.255 host 192.168.1.11 eq www
R2(config)#access-list 100 permit ip any any

R2(config)#int gigabitEthernet 0/1
R2(config-if)#ip access-group 100 in
R2(config-if)#exit
```

Office Server will be allowed to access only by Office Network, but all other networks will be denied. All networks will be allowed to access any other Datacenter Servers.

```
R1(config)#access-list 101 permit ip 192.168.2.0 0.0.0.255 host 192.168.1.11
R1(config)#access-list 101 deny ip any host 192.168.1.11
R1(config)#access-list 101 permit ip any any

R1(config)#int gigabitEthernet 0/0
R1(config-if)#ip access-group 101 out
R1(config-if)#exit
```

LAB-7 PC01 cannot ping File server but only acccess by http, All other Traffic is Permitted for Data Center Network.

Answer-01: (in Router R1)

```
R1(config)#access-list 102 deny icmp host 192.168.3.10 host 192.168.1.14 echo-reply
R1(config)#access-list 102 permit tcp host 192.168.3.10 host 192.168.1.14 eq www
R1(config)#access-list 102 deny ip host 192.168.3.10 host 192.168.1.14
R1(config)#access-list 102 permit ip any any



R1(config)#int gigabitEthernet 0/0
R1(config-if)#ip access-group 102 out
R1(config-if)#
```

Answer-02: (in Router R3)

```
R1(config)#access-list 103 deny icmp host 192.168.3.10 host 192.168.1.14 echo-reply
R1(config)#access-list 103 permit tcp host 192.168.3.10 host 192.168.1.14 eq www
R1(config)#access-list 103 deny ip host 192.168.3.10 host 192.168.1.14
R1(config)#access-list 103 permit ip any any

R1(config)#int gigabitEthernet 0/0
R1(config-if)#ip access-group 103 in
R1(config-if)#
```

Lab-8: Introduction VLAN and Inter-VLAN Routing

Objectives

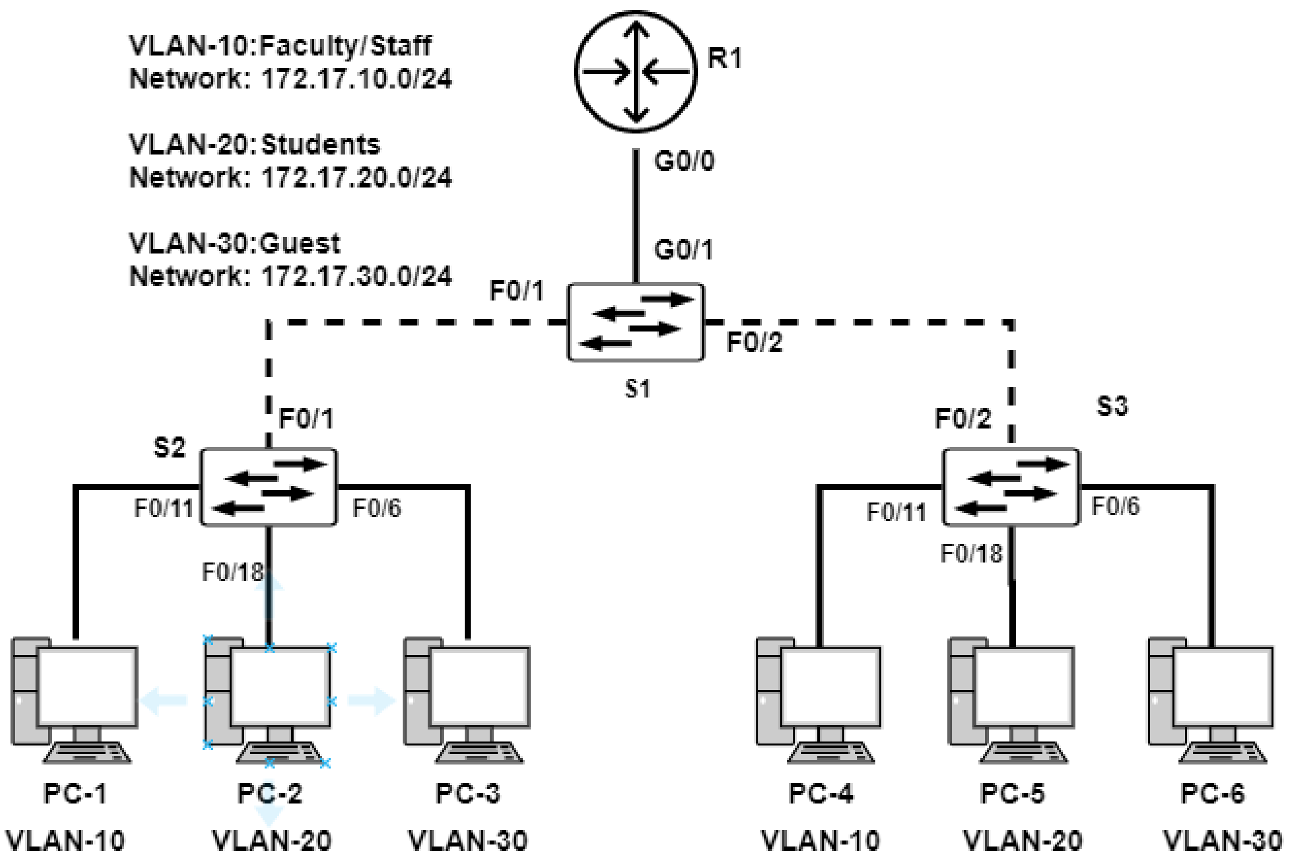
- To learn about **Virtual LAN (VLAN)**: why and how used?
- To implement **Inter-VLAN Routing** using **Packet Tracer**

Part-1: Background: VLAN

Modern switches use virtual local-area networks (VLANs) to improve network performance by separating large Layer 2 broadcast domains into smaller ones. VLANs can also be used as a security measure by controlling which hosts can communicate. In general, VLANs make it easier to design a network to support the goals of an organization. VLAN trunks are used to span VLANs across multiple devices. Trunks allow the traffic from multiple VLANS to travel over a single link, while keeping the VLAN identification and segmentation intact.

In this lab, you will create VLANs on both switches in the topology, assign VLANs to switch access ports, verify that VLANs are working as expected, and then create a VLAN trunk between the two switches to allow hosts in the same VLAN to communicate through the trunk, regardless of which switch the host is actually attached to.

Part-2: VLAN and Inter-VLAN Routing Configuration



Addressing Table

Device (Hostname)	Interface	IP Address	Subnet Mask
PC1	NIC	172.17.10.21	255.255.255.0
PC2	NIC	172.17.20.22	255.255.255.0
PC3	NIC	172.17.30.23	255.255.255.0
PC4	NIC	172.17.10.24	255.255.255.0
PC5	NIC	172.17.20.25	255.255.255.0
PC6	NIC	172.17.30.26	255.255.255.0

Initial Port Assignments (Switches 2 and 3)

Ports	Assignment	Network
Fa0/1 – 0/5	802.1q Trunks (Native VLAN 99)	172.17.99.0 /24
Fa0/6 – 0/10	VLAN 30 – Guest (Default)	172.17.30.0 /24
Fa0/11 – 0/17	VLAN 10 – Faculty/Staff	172.17.10.0 /24
Fa0/18 – 0/24	VLAN 20 – Students	172.17.20.0 /24

Part-2(A): Prepare the Network

It is a good practice to disable any unused ports on the switches by using shutdown. Disable all ports on the switches:

```
Switch(config)#interface range fa0/1-24
Switch(config-if-range)#shutdown
Switch(config-if-range)#exit

Switch(config-if-range)#interface range gi0/1-2
Switch(config-if-range)#shutdown
Switch(config-if-range)#exit
```

Part-2(B): Create VLAN in all Switch

Configure the switch hostname and Create VLANs on S1, S2 and S3.

```
S1(config)#vlan 10
S1(config-vlan)#name faculty/staff
S1(config-vlan)#vlan 20
S1(config-vlan)#name students
S1(config-vlan)#vlan 30
S1(config-vlan)#name guest
S1(config-vlan)#end
```

Note: Configure VLANs' in S2 and S3

Part-2(C): Verify that the VLANs have been created on S1, S2 and S3

```
S1#show vlan brief
```

```
VLAN Name Status Ports
---- ------------------------------ --------- -------------------------
1 default active        Fa0/1, Fa0/2, Fa0/4, Fa0/5, Fa0/6, Fa0/7, Fa0/8,
                        Fa0/9, Fa0/10, Fa0/11, Fa0/12, Fa0/13, Fa0/14,
                        Fa0/15, Fa0/16,Fa0/17, Fa0/18, Fa0/19, Fa0/20,
                        Fa0/21, Fa0/22, Fa0/23,Fa0/24,Gi0/1,Gi0/2
10 faculty/staff active
20 students active
30 guest active
```

Note: See also S2 and S3 VLAN Database.

Part-2(D): Assign switch ports to VLANs on S2 and S3.

```
S3(config)#interface range fa0/6-10
S3(config-if-range)#switchport mode access
S3(config-if-range)#switchport access vlan 30
S3(config-if-range)#no shutdown
```

Note: Configure others VLAN same as this configuration and also Configure S2

Part-2(E): Configure the Trunking ports on all switches.

```
S1(config)#interface range fa0/1-2
S1(config-if-range)#switchport mode trunk
S1(config-if-range)#no shutdown
S1(config-if-range)#end
```

Part-2(F): Verify the VLAN

Ping the same VLAN PCs'

Part-3: Configure Inter-VLAN Routing

Add a Router R1 with the switch s1 and configure sub interfaces on R1 using the 802.1Q encapsulation.

State-up the router Gig0/0 interface

```
R1(config)# int gig0/0
R1(config-if)# no shutdown
R1(config-if)# exit
```

<u>Create the sub interface G0/0.10 sub interface</u>

Set the encapsulation type to **802.1Q** and assign VLAN 10 to the sub interface.

- Refer to the Address Table and assign the correct IP address to the sub interface.

```
R1(config)# int g0/0.10
R1(config-subif)# encapsulation dot1Q 10
R1(config-subif)# ip address 172.17.10.1 255.255.255.0
R1(config-subif)# exit
```

<u>Repeat for the G0/0.20 sub interface.</u>

```
R1(config-subif)# int g0/0.20
R1(config-subif)# encapsulation dot1Q 30
R1(config-subif)# ip address 172.17.20.1 255.255.255.0
R1(config-subif)# exit
```

<u>Repeat for the G0/0.30 sub interface.</u>

```
R1(config-subif)# int g0/0.30
R1(config-subif)# encapsulation dot1Q 30
R1(config-subif)# ip address 172.17.30.1 255.255.255.0
R1(config-subif)# exit
```

<u>Lab-9: Network Address Translation (NAT) – Connecting to an ISP</u>

Objectives

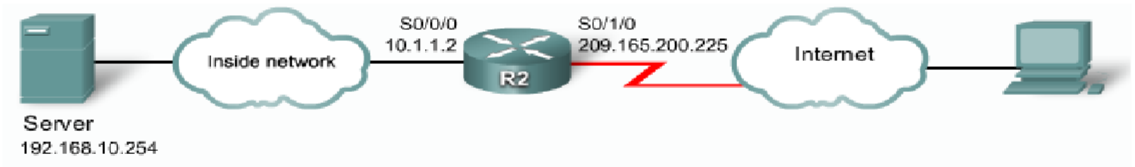
- To learn about Network Address Translation (NAT): why and how used?
- Types of NAT: Static and Dynamic NAT, Port Address Translation (PAT)
- To build an internetwork using NAT using Packet Tracer

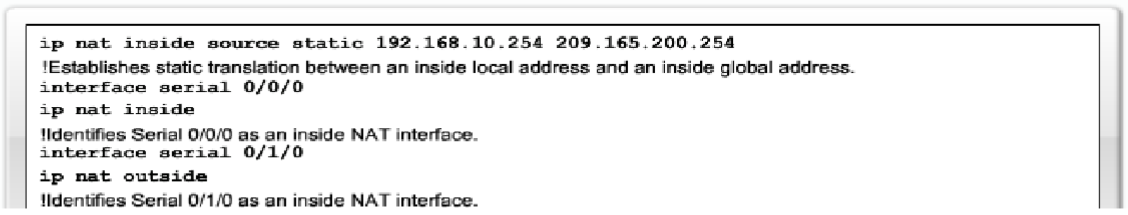
<u>Part-1: Background: Network Address Translation (NAT)</u>

NAT (Network Address Translation) is a technique for preserving scarce Internet IP addresses. It converts private IP addresses (not routable to Internet) to public IP addresses so that Internet can be accessed from private network.

Types of NAT
Developed by Cisco, Network Address Translation is used by a device (firewall, router or computer) that sits between an internal network and the rest of the world. NAT has many forms and can work in several ways:

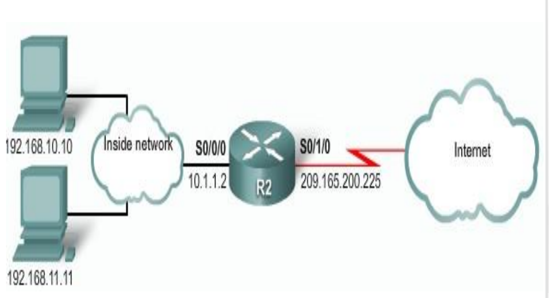
<u>Static NAT</u> - Mapping an unregistered (private) IP address to a registered (public) IP address on a one-to-one basis. Particularly useful when a local host with private IP address needs to be accessible from outside the network (from Internet).

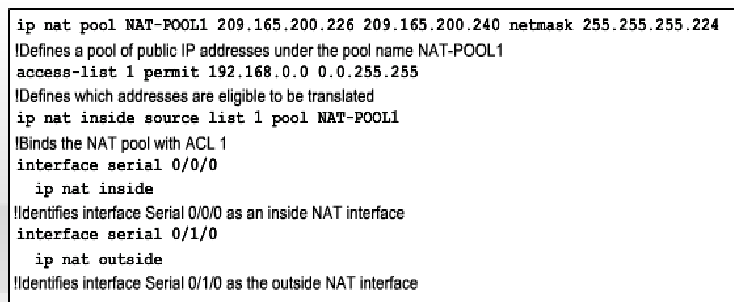




```
ip nat inside source static 192.168.10.254 209.165.200.254
!Establishes static translation between an inside local address and an inside global address.
interface serial 0/0/0
ip nat inside
!Identifies Serial 0/0/0 as an inside NAT interface.
interface serial 0/1/0
ip nat outside
!Identifies Serial 0/1/0 as an inside NAT interface.
```

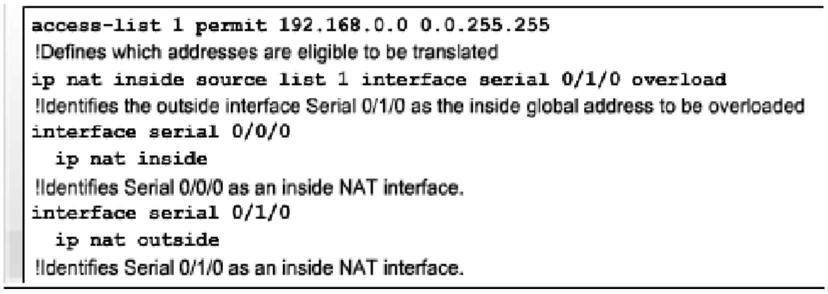
<u>Dynamic NAT</u> - Maps an unregistered IP address to a registered IP address from a group of registered IP addresses.





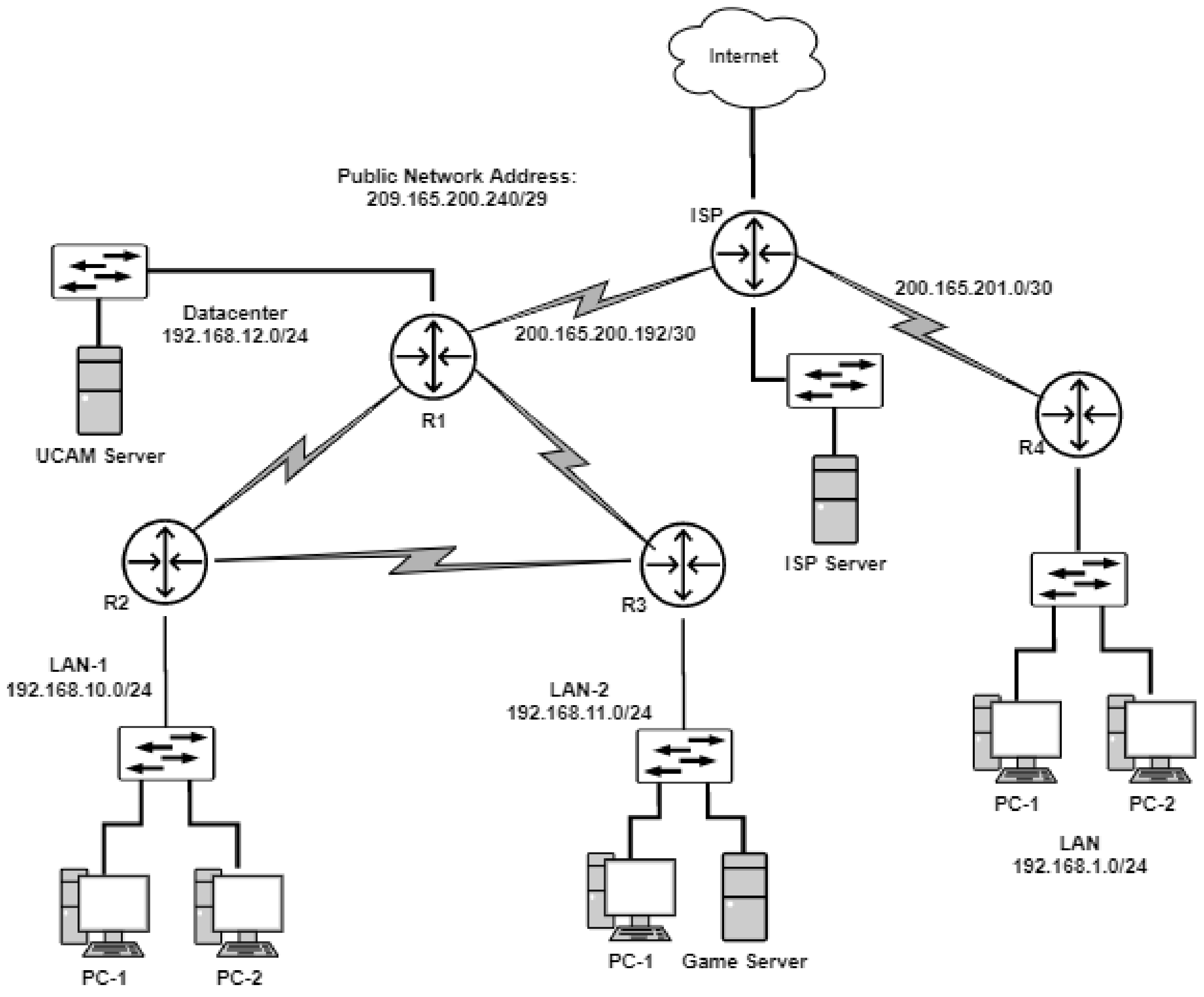
```
ip nat pool NAT-POOL1 209.165.200.226 209.165.200.240 netmask 255.255.255.224
!Defines a pool of public IP addresses under the pool name NAT-POOL1
access-list 1 permit 192.168.0.0 0.0.255.255
!Defines which addresses are eligible to be translated
ip nat inside source list 1 pool NAT-POOL1
!Binds the NAT pool with ACL 1
interface serial 0/0/0
  ip nat inside
!!Identifies interface Serial 0/0/0 as an inside NAT interface
interface serial 0/1/0
  ip nat outside
!!Identifies interface Serial 0/1/0 as the outside NAT interface
```

PAT (Port Address Translation)- A form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address by using different ports. This is known also as PAT (Port Address Translation), single address NAT or port-level multiplexed NAT.



```
access-list 1 permit 192.168.0.0 0.0.255.255
!Defines which addresses are eligible to be translated
ip nat inside source list 1 interface serial 0/1/0 overload
!Identifies the outside interface Serial 0/1/0 as the inside global address to be overloaded
interface serial 0/0/0
  ip nat inside
!Identifies Serial 0/0/0 as an inside NAT interface.
interface serial 0/1/0
  ip nat outside
!Identifies Serial 0/1/0 as an inside NAT interface.
```

Part-2: Configure Network Address Translation (NAT)



Part-2(A): Configure Routing

ISP (Internet Service Provider) uses static routing to reach all networks beyond R1 (Border routers). However, R1 translate private addresses into public addresses before sending traffic to ISP (Internet). Therefore, ISP must be configured with the public addresses (209.165.200.240/29) for R1. But R4 no need to ISP routing because it's use the link public address for PAT.

We will show the configurations on R1, Enter the following static route on ISP. This static route includes all addresses assigned to R1 for public use.

```
ISP (config) #ip route 209.165.200.240 255.255.255.248 200.165.200.194
```

Configure RIP in all LAN routers,

```
R1(config)#router rip
R1(config-router)#net 192.168.12.0
R1(config-router)#net 192.168.220.0
R1(config-router)#net 192.168.221.0
R1(config-router)#exit
```

Note: Configure others LAN router.

Configure a default route on R1 and propagate the route in RIP.

```
R1(config)#ip route 0.0.0.0 0.0.0.0 209.165.200.193

R1(config)#router rip
R1(config-router)#default-information originate
R1(config-router)#exit
```

Allow a few seconds for R2 & R3 to learn the default route from R1 and then check the R2 & R3 routing table using show ip route command.

Alternatively, you can clear the routing table with the clear ip route "*" command. A default route pointing to R1 should appear in the R2 & R3 routing table. From R2, ping the serial interface (connected to ISP) on R1. The pings should be successful. Troubleshoot if the pings fail.

Part-2(B): Configure Static NAT

Statically map a public IP address to a private IP address.

The **UCAM Server** attached to R1 is accessible by outside hosts beyond ISP. Statically assign the public IP address 209.165.200.241 as the address for NAT to use to map packets to the private IP address of the **UCAM Server** at 192.168.12.12. Similarly, assign the public IP address 209.165.200.242 to the private IP address of the **Game Server** at 192.168.11.11

```
R1 (config)#ip nat inside source static 192.168.12.12 209.165.200.241
R1 (config)#ip nat inside source static 192.168.11.11 209.165.200.242
```

Part-2(C): Configure Dynamic NAT

While static NAT provides a permanent mapping between an internal address and a specific public address, dynamic NAT maps private IP addresses to public addresses. These public IP addresses come from a NAT pool.

Define a pool of global addresses.
Create a pool of addresses to which matched source addresses are translated. The following command creates a pool named **NAT-POOL** that translates matched addresses to an available IP address in the 209.165.200.243 - 209.165.200.245 range.

```
R1 (config)#ip nat pool NAT-POOL 209.165.200.243 209.165.200.245 netmask 255.255.255.248
```

Create a standard access control list to identify which inside addresses are translated.

```
R1 (config)#access-list 1 permit 192.168.10.0 0.0.0.255
R1 (config)#access-list 1 permit 192.168.11.0 0.0.0.255
R1 (config)#access-list 1 permit 192.168.12.0 0.0.0.255
```

Establish dynamic source translation by binding the pool with the access control list.
A router can have more than one NAT pool and more than one ACL. The following command tells the router which address pool to use to translate hosts that are allowed by the ACL.

```
R1 (config)#ip nat inside source list 1 NAT pool NAT-POOL
```

Part-2(D): Specify inside and outside NAT interfaces
Before NAT can work, you must specify which interfaces are inside and which interfaces are outside.

```
R1(config)#interface serial 0/1/1
R1(config-if)#ip nat outside
R1(config-if)#exit

R1(config-if)#interface G0/0
R1(config-if)#ip nat inside
R1(config-if)#exit

R1(config)#interface serial 0/0/0
R1(config-if)#ip nat inside
R1(config-if)#exit

R1(config)#interface serial 0/0/1
R1(config-if)#ip nat inside
R1(config-if)#exit
```

Part-2(E): Verify the configuration.

```
R1#show ip nat translations
Pro   Inside global      Inside local   Outside local      Outside global
---   209.165.200.241    192.168.12.2   ---                ---
---   209.165.200.242    192.168.11.3   ---                ---
icmp  209.165.200.243    192.168.10.2   200.165.200.193    200.165.200.193
icmp  209.165.200.244    192.168.11.2   200.165.200.193    200.165.200.193
```

Part-3: Configure PAT (Port Address translation)

In the previous example, what would happen if you needed more than the four public IP addresses that the pool allows? By tracking port numbers, NAT overloading allows multiple inside users to reuse a public IP address.

In this task, you will configure **PAT** (or NAT overload) on R4 so that all internal IP addresses are translated to the R4 S0/0/0 IP address when connecting to any outside device.

The **configuration is similar to dynamic NAT**, except that instead of a pool of addresses, the interface keyword is used to identify the outside IP address. Therefore, no NAT pool is defined. The overload keyword enables the addition of the port number to the translation.

Create a standard access control list to identify which inside addresses are translated.

```
R4 (config)#access-list 10 permit 192.168.1.0 0.0.0.255
```

Establish dynamic source translation by binding the pool with the access control list.

```
R4(config)#ip nat inside source list 10 interface S0/0/0 overload
```

Specify inside and outside NAT interfaces

```
R4(config)#int serial 0/0/0
R4(config-if)#ip nat outside
R4(config-if)#exit

R4(config)#int gig0/0
R4(config-if)#ip nat inside
R4(config-if)#exit
```

Verify the configuration

```
R4#show ip nat translations
Pro     Inside global           Inside local        Outside local         Outside global
icmp    209.165.200.194:3       192.168.10.11:3     209.165.200.193:3     209.165.200.193:3
icmp    209.165.200.194:1024    192.168.11.11:3     209.165.200.193:3     209.165.200.193:1024
---     209.165.200.241         192.168.12.2  ---       ---
---     209.165.200.242         192.168.11.3  ---       ---
```

Lab-10: Introduction to Socket Programming

Objectives

- Introduction to Socket Programming concepts.
- Introduction to Python API for Socket Programming.
- To create a Client-Server application using Socket Programming in Python.

Part-1: Background- Socket Basics

A network socket is an endpoint of an inter-process communication flow across a computer network. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents. Today, most communication between computers is based on the internet protocol; therefore most network sockets are internet sockets. To create a connection between machines, Python programs import the socket module, create a socket object, and call the object's methods to establish connections and send and receive data. Sockets are the endpoints of a bidirectional communications channel.

In this lab, you will learn the basics of socket programming for TCP connections in Python: how to create a socket, bind it to a specific address and port, as well as send and receive a HTTP packet. You will also learn some basics of HTTP header format.

Hints

- Here are a few hints that may help you as you write the program.
- You have to choose a server port to connect to. Ports from 1-1023 are mostly used for certain services and require administrative privileges. Use port numbers greater than at least 1023.
- Close your sockets cleanly before exit. If you abort the program, the port may not be freed.
- You can run all of the processes on the same machine. For your machine, just use localhost. You can use ifconfig (unix) or ipconfig (windows) to know IP address for testing multiple machines.
- Be wary of overzealous firewalls stopping your connections - try temporarily disabling firewalls if you find your connections timeout or are denied.

Create a Socket

This first thing to do is create a socket. The socket.socket function does this. Run socketExample1.py. The code will create a socket with Address Family : AF_INET (this is IP version 4 or IPv4), Type : SOCK_STREAM (this means connection oriented TCP protocol).

Connect to a Server

We connect to a remote server on a certain port number. So we need 2 things , IP address and port number to connect to. So you need to know the IP address of the remote server you are connecting to. Here we used the ip address of google.com as a sample. Run socketExample2.py. It creates a socket and then connects. Try connecting to a port different from port 80 and you should not be able to connect which indicates that the port is not open for connection. This logic can be used to build a port scanner.

Sending data to a Server

Function sendall will simply send data. Let us send some data to google.com. Run socketExample3.py. In the above example , we first connect to an ip address and then send the string message "GET / HTTP/1.1\r\n\r\n" to it. The message is actually an "http command" to fetch the mainpage of a website. Now that we have send some data, it’s time to receive a reply from the server. So let us do it.

Sending data to a Server and receiving data from the Server
Function recv is used to receive data on a socket. In the following example we shall send the same message as the last example and receive a reply from the server. Run socketExample4.py. Google.com replied with the content of the page we requested. Quite simple! Finally, we close the socket.

Part-2: Sample Server and Client Code

A Simple Server
To write Internet servers, we use the socket function available in socket module to create a socket object. A socket object is then used to call other functions to setup a socket server. Now call **bind** (hostname, port) function to specify a port for your service on the given host. Next, call the accept method of the returned object. This method waits until a client connects to the port you specified, and then returns a connection object that represents the connection to that client.

```
#server, (TCP Server Code)
from socket import *             # Imports socket module

host="127.0.0.1"                 # Set the server address to variable host

port=4446                        # Sets the variable port to 4444

s = socket(AF_INET, SOCK_STREAM

s.bind((host,port))        # Binds the socket. Note that the input to the bind function is a tuple

s.listen(1)              # Sets socket to listening state with a  queue of 1 connection

print ("Listening for connections.. ")

# Accepts incoming request from client and returns socket & address to variables q and addr
q,addr=s.accept()

print("Connected with "+addr[0]+ ':' + str(addr[1]))

# Data to be sent is stored in variable data from user
data=input("Enter data to be send:  ")

# Sends data to client
q.send(data.encode())
s.close()
# End of code
```

A Simple Client
Now we will write a very simple client program which will open a connection to a given port 12345 and given host. This is very simple to create a socket client using Python's *socket* module function. The **socket.connect (hosname, port)** opens a TCP connection to *hostname* on the *port*. Once you have a socket open, you can read from it like any IO object. When done, remember to close it, as you would close a file.

The following code is a very simple client that connects to a given host and port, reads any available data from the socket, and then exits:

```
#Client
# TCP Client Code

#host="127.0.0.1"          # Set the server address to variable host

port=4446                 # Sets the variable port to 4444

from socket import *       # Imports socket module

s=socket(AF_INET, SOCK_STREAM)  # Creates a socket
s.connect((host,port))          # Connect to server address

msg = s.recv(1024)           # Receives data upto 1024 bytes and stores in variables msg

print ("Message from server : " + msg.decode())

s.close()                # Closes the socket
# End of code
```

Now run this **server.py** in background and then run above client.py to see the result

<u>Part-3: A web Server</u>

You will develop a **web server** that handles **one HTTP request** at a time. Your web server should accept and parse the HTTP request, get the requested file from the server’s file system, create an **HTTP response message** consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present in the server, the server should send an HTTP “**404 Not Found**” message back to the client.

Skeleton Code
Below find the skeleton code for the Web server. You are to **<u>complete the skeleton code</u>**. The places where you need to fill in code are marked with **<u>#Fill in start</u>** and **<u>#Fill in end</u>**. Each place may require one or more lines of code.

Running the Server
Put an **HTML file** (e.g., HelloWorld.html) in the **same directory** that the server is in. Run the server program. Determine the IP address of the host that is running the server (e.g., 128.238.251.26). From another host, open a browser and provide the corresponding URL. For example:
http://128.238.251.26:6789/HelloWorld.html

‘HelloWorld.html’ is the name of the file you placed in the server directory. Note also the use of the port number after the colon. You need to replace this port number with whatever port you have used in the server code. In the above example, we have used the port number 6789. The browser should then display the contents of HelloWorld.html. If you omit ":6789", the browser will assume port 80 and you will get the web page from the server only if your server is listening at port 80. Then try to get a file that is not present at the server. You should get a “404 Not Found” message.

Submit the files

You will hand in the complete server code along with the screen shots of your client browser, verifying that you actually receive the contents of the HTML file from the server.

Skeleton Python Code for the Web Server

```
#import socket module
from socket import *
serverSocket = socket(AF_INET, SOCK_STREAM)

#Prepare a sever socket
#Fill in start
#Fill in end while True:
#Establish the connection print 'Ready to serve...'
connectionSocket, addr = #Fill in start          #Fill in end

try:
message = #Fill in start          #Fill in end
filename = message.split()[1] f = open(filename[1:])
outputdata = #Fill in start        #Fill in end

#Send one HTTP header line into socket
#Fill in start
#Fill in end
#Send the content of the requested file to the client

for i in range(0, len(outputdata)): connectionSocket.send(outputdata[i])
connectionSocket.close()
except IOError:

#Send response message for file not found
#Fill in start
#Fill in end
#Close client socket
#Fill in start
#Fill in end
serverSocket.close()
```

Lab-11: Basic Security/Cryptography (Compare Data with a Hash)

Objectives

Use a hashing program to verify the integrity of data.

Background / Scenario

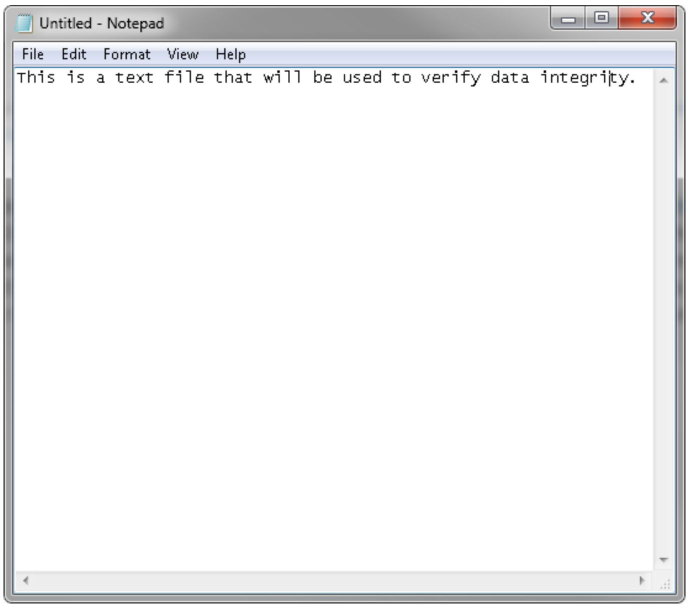
It is important to understand when data has been corrupted or it has been tampered with. A hashing program can be used to verify if data has changed, or if it has remained the same. A hashing program performs a hash function on data or a file, which returns a (usually much shorter) value. There are many different hash functions, some very simple and some very complex. When the same hash is performed on the same data, the value that is returned is always the same. If any change is performed on the data, the hash value returned will be different.

Note: You will need installation privileges and some knowledge of the process to install Windows programs.

Part-1: Create a Text file

a. Search your computer for the Notepad program and open it.

b. Type some text in the program.



```
This is a text file that will be used to verify data integrity.
```

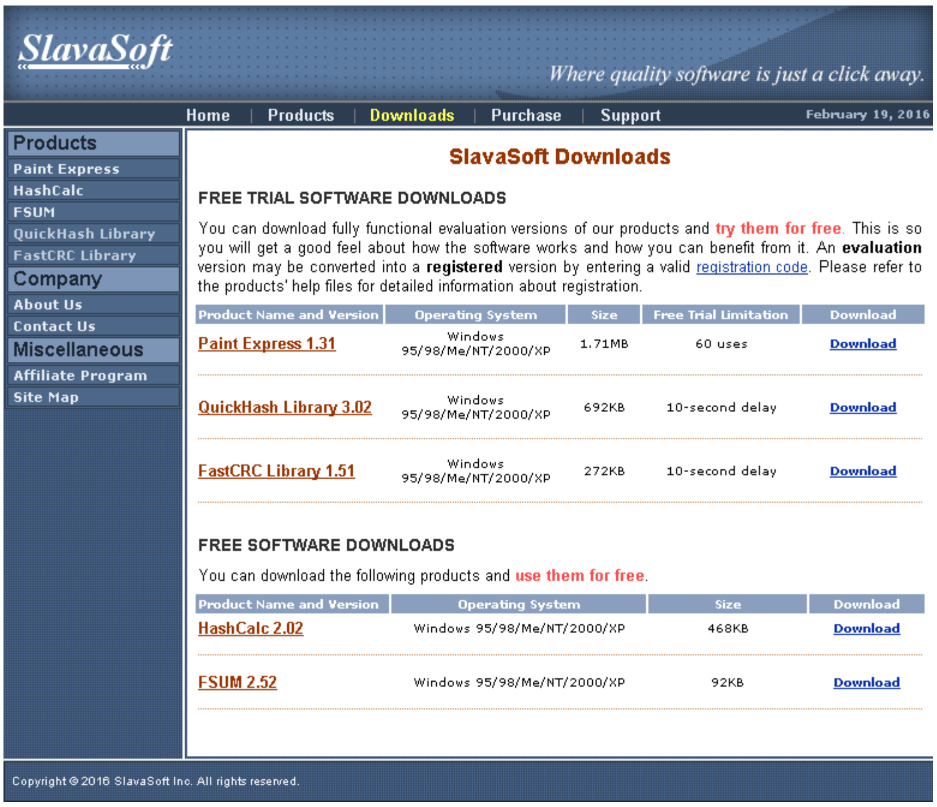
c. Choose **File > Save**.

d. Navigate to **Desktop**.

e. Type **Hash** in the **File name:** field, and click **Save**.

Part-2: Install HashCalc

f. Open a web browser and navigate to **http://www.slavasoft.com/download.htm**



SlavaSoft Downloads

FREE TRIAL SOFTWARE DOWNLOADS

You can download fully functional evaluation versions of our products and try them for free. This is so you will get a good feel about how the software works and how you can benefit from it. An **evaluation** version may be converted into a **registered** version by entering a valid registration code. Please refer to the products' help files for detailed information about registration.

Product Name and Version	Operating System	Size	Free Trial Limitation	Download
Paint Express 1.31	Windows 95/98/Me/NT/2000/XP	1.71MB	60 uses	Download
QuickHash Library 3.02	Windows 95/98/Me/NT/2000/XP	692KB	10-second delay	Download
FastCRC Library 1.51	Windows 95/98/Me/NT/2000/XP	272KB	10-second delay	Download

FREE SOFTWARE DOWNLOADS

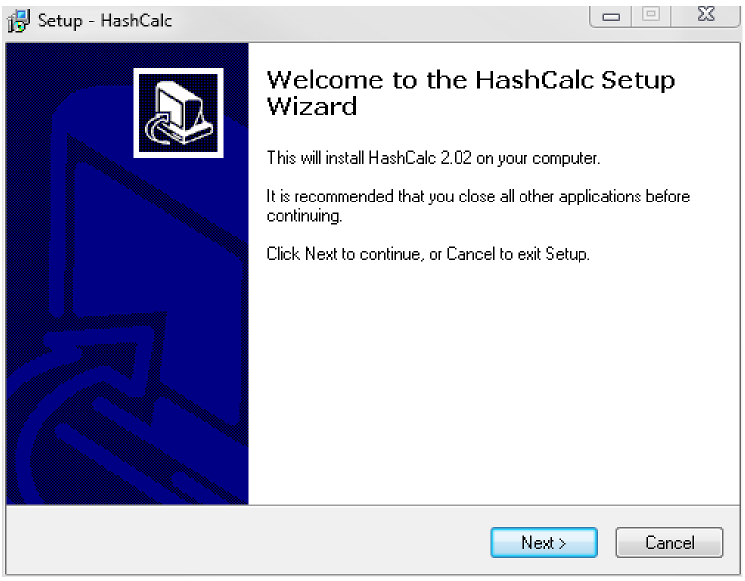
You can download the following products and use them for free.

Product Name and Version	Operating System	Size	Download
HashCalc 2.02	Windows 95/98/Me/NT/2000/XP	468KB	Download
FSUM 2.52	Windows 95/98/Me/NT/2000/XP	92KB	Download

Copyright © 2016 SlavaSoft Inc. All rights reserved.

g. Click **Download** in the **HashCalc 2.02** row.

h. Open the **hashcalc.zip** file and run the **setup.exe** file inside.



Welcome to the HashCalc Setup Wizard

This will install HashCalc 2.02 on your computer.

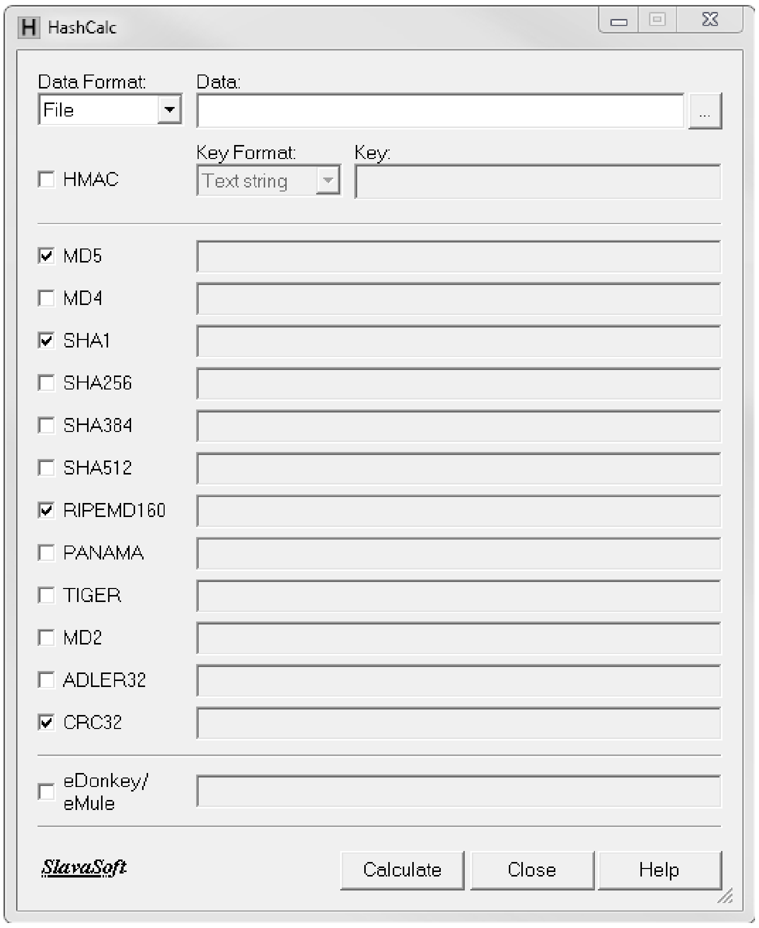
It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

i. Follow the installation wizard to install HashCalc.

j. Click **Finish** on the last screen, and close the **README** file if it opened. You may read the file if you wish.

k. **HashCalc** is now installed and running.


HashCalc

Data Format: File
Data:

☐ HMAC
Key Format: Text string
Key:

☑ MD5
☐ MD4
☑ SHA1
☐ SHA256
☐ SHA384
☐ SHA512
☑ RIPEMD160
☐ PANAMA
☐ TIGER
☐ MD2
☐ ADLER32
☑ CRC32
☐ eDonkey/ eMule

Part-3: Calculate a hash of the Hash.txt file

a. Set the following items in HashCalc:

1) Data Format: **File**.

2) Data: Click the **…** button next to the Data field, navigate to the **Desktop** and choose the **Hash.txt** file.

3) Uncheck **HMAC**.

4) Uncheck all hash types except **MD5**.

b. Click the **Calculate** button.

What is the value next to **MD5**?

Answer:

Part-4: Make a change to the Hash.txt file

a. Navigate to the **Desktop** and open the **Hash.txt** file.

c. Make a minor change to the text, such as deleting a letter, or adding a space or period.

d. Click **File > Save**, and close **Notepad**.

Part-5: Calculate a new hash of the Hash.txt file

a. Click the **Calculate** button in HashCalc again.

What is the value next to **MD5**?

Ans:

b. Note: Is the value different from the value recorded in Step 3?

Ans:

c. Place a check mark next to all of the hash types.

d. Click **Calculate**.

e. Notice that many of the hash types create a hash of a different length. Why?

Ans:

======== End ======

Azim Uddin Chowdhury

azim@admin.uiu.ac.bd