SRI RAMACHANDRA

INSTITUTE OF HIGHER EDUCATION AND RESEARCH

(Category - I Deemed to be University) Porur, Chennai

SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF CYBER SECURITY AND IOT

ACADEMIC YEAR: 2025-2026

ODD SEMESTER

LAB MANUAL

(REGULATION - 2023)

CSE23CL301 - Computer Networks Laboratory

FIFTH SEMESTER

B.TECH – CYBER SECURITY AND IOT

Prepared By

S.MANOJ KUMARAN / CYB & IoT

SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

UNIQUE ID

NAME

RECORD NOTE BOOK

Certified that this is a bonafide record of work done by ------------------------------------

Of --------------------------- class in the ---laboratory during the year ----------------------------.

Signature of the Subject Faculty

Evaluation Date:

Internal Examiner **External Examiner**

<table>
<tr><th colspan="4">INTERNAL ASSESSMENT FOR LABORATORY</th></tr>
<tr><td>S.NO</td><td>DESCRIPTION</td><td>MARKS</td><td>MARKS OBTAINED</td></tr>
<tr><td>1</td><td>CONDUCTION & EXECUTION OF EXPERIMENT</td><td>40</td><td></td></tr>
<tr><td>2</td><td>RECORD</td><td>10</td><td></td></tr>
<tr><td colspan="2">TOTAL</td><td>50</td><td></td></tr>
</table>

Index				
Exp No	Date	Experiment	Mark	Sign

Experiment 1: Study of Network Devices using Cisco Packet Tracer

Aim:
To study and understand the functionalities of various network devices such as Router, Switch, Hub, Repeater, Bridge, and Access Point using Cisco Packet Tracer.

Apparatus / Software Required:

- Cisco Packet Tracer software (version 7.x or later)
- PC or Laptop

Theory:
Computer networks consist of various hardware devices that work together to enable communication. Some of the common network devices include:

1. **Router**: Connects different networks and directs data packets between them.
2. **Switch**: Connects multiple devices within the same network and uses MAC addresses to forward data.
3. **Hub**: Basic networking device that transmits data to all connected devices.
4. **Repeater**: Regenerates and amplifies signals to extend the transmission distance.
5. **Bridge**: Connects two LANs and filters traffic based on MAC addresses.
6. **Access Point**: Allows wireless devices to connect to a wired network using Wi-Fi.

Each device plays a vital role in the overall network architecture and helps in efficient data communication.

Procedure:

1. **Open Cisco Packet Tracer**.
2. **Drag and Drop Devices**:
    - Go to the “Network Devices” section.
    - Select and drag the following to the workspace:
        - 1 Router (e.g., 2811)
        - 1 Switch (e.g., 2960)
        - 1 Hub
        - 1 Bridge (Generic or use another Switch)
        - 1 Repeater (Use another switch or emulate)

        - 1 Access Point (e.g., WAP)
3. **Add End Devices**:
    - Add 2-3 PCs or Laptops to show device connectivity.
4. **Connect Devices**:
    - Use appropriate cables (Copper Straight-Through or Cross-Over).
    - Connect PCs to the switch/hub.
    - Connect switch/hub to the router.
    - Connect wireless device to Access Point.
5. **Label Devices** for identification.
6. **Observe Device Behavior**:
    - Click on each device.
    - Study configuration options.
    - Note interfaces, ports, and LEDs.
    - Check the physical and logical topology.
7. **Optional**:
    - Assign IP addresses to PCs.
    - Use `ping` to test connectivity.

Sample Output / Observation Table:

Device	Function	Number of Ports	Layer
Router	Forwards data between networks	Multiple (FastEthernet, Serial)	Network (Layer 3)
Switch	Forwards data based on MAC address	24 or more Ethernet ports	Data Link (Layer 2)
Hub	Broadcasts data to all devices	4-8 ports	Physical (Layer 1)
Repeater	Regenerates signals	2 ports	Physical (Layer 1)
Bridge	Divides network to reduce traffic	2-4 ports	Data Link (Layer 2)
Access Point	Provides wireless access	1 Ethernet port + Wi-Fi	Data Link (Layer 2)

Result:

The functionalities and working of various network devices (Router, Switch, Hub, Bridge, Repeater, and Access Point) were studied successfully using Cisco Packet Tracer.

Conclusion:
This experiment helped in understanding the roles of different networking devices, their placement in a network, and their basic behaviour through simulation.

Experiment 2: Configure a Network Topology using Packet Tracer Software

Aim:
To design and configure a basic network topology using Cisco Packet Tracer, enabling communication between end devices.

Apparatus / Software Required:

- Cisco Packet Tracer software (version 7.x or later)
- PC or Laptop

Theoretical Background:

- **IP Addressing**: Every device in a network requires a unique IP address. An IP address consists of a network portion and a host portion, typically defined using a subnet mask.
- **Subnetting** *(optional but recommended)*: Subnetting divides a network into multiple logical sub-networks, helping in better organization and security.
- **Basic Device Configuration**:
  - Assigning IP addresses to end devices.
  - Configuring IP addresses and hostnames for switches and routers.
- **Connectivity Testing**:
  - **Ping**: Tests basic connectivity between devices.
  - **Traceroute**: Identifies the path data takes between devices.

Procedure:

1. **Launch Cisco Packet Tracer.**
2. **Design the Network Topology**:
   - Add **2 PCs**, **1 Switch**, and **1 Router** from the device menu.
   - Use **Copper Straight-Through** cables to connect:
     - PC0 ↔ Switch
     - PC1 ↔ Switch
     - Switch ↔ Router (Fa0/0 port)
3. **Assign IP Addresses**:
   - For this example:
     - PC0: IP – 192.168.1.10, Subnet Mask – 255.255.255.0, Default Gateway – 192.168.1.1

        - PC1: IP – 192.168.1.20, Subnet Mask – 255.255.255.0, Default Gateway – 192.168.1.1
        - Router (Fa0/0): IP – 192.168.1.1, Subnet Mask – 255.255.255.0

4. **Configure Router**:
    - Click on the Router → CLI tab
    - Enter the following commands:
    - `Router> enable`
    - `Router# configure terminal`
    - `Router(config)# interface fastethernet 0/0`
    - `Router(config-if)# ip address 192.168.1.1 255.255.255.0`
    - `Router(config-if)# no shutdown`
    - `Router(config-if)# exit`
    - `Router(config)# hostname R1`
    - `Router(config)# exit`
5. **Assign IP to PCs**:
    - Click on each PC → Desktop tab → IP Configuration.
    - Enter the IP address, subnet mask, and default gateway.
6. **Test Connectivity**:
    - Open the command prompt on PC0.
    - Type `ping 192.168.1.20` to test PC-to-PC connectivity.
    - Type `ping 192.168.1.1` to test connection to the router.

Sample Output:

```
Pinging 192.168.1.20 with 32 bytes of data:
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128
Reply from 192.168.1.20: bytes=32 time<1ms TTL=128

Pinging 192.168.1.1 with 32 bytes of data:
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
```

Result:
The basic network topology was successfully created and configured. Communication between end devices was established and verified using the `ping` command.

Conclusion:
This experiment demonstrated how to design and configure a simple network in Cisco Packet Tracer, assign IP addresses, and verify communication between devices using network utility commands.

Experiment 3: Simulating a Local Area Network using Cisco Packet Tracer

Aim:
To simulate a Local Area Network (LAN) using Cisco Packet Tracer and establish communication among multiple computers within the same network.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- PC or Laptop

Theoretical Background:

- **Local Area Network (LAN):** A LAN connects computers and devices within a limited area such as a building, school, or campus. It enables resource sharing like files, printers, and internet access.
- **Basic LAN Components**:
  - **End Devices**: PCs, Laptops
  - **Intermediary Devices**: Switches
  - **Media**: Copper Straight-Through cables
- **IP Addressing**: Each device in the LAN must be assigned a unique IP address within the same subnet to ensure proper communication.
- **Communication Protocols**:
  - **Ethernet**: Primary LAN protocol used for data transmission.
  - **ICMP (Ping)**: Used for testing connectivity between devices.

Procedure:

1. **Open Cisco Packet Tracer.**
2. **Add Devices to the Workspace**:
   - 4 PCs (PC0, PC1, PC2, PC3)
   - 1 Switch (e.g., 2960)
3. **Connect Devices**:
   - Use **Copper Straight-Through** cables to connect each PC to the Switch.
4. **Assign IP Addresses to PCs**:
   - Use the same network (e.g., 192.168.10.0/24)
   - Example:
     - PC0: 192.168.10.1, Subnet Mask: 255.255.255.0

    - PC1: 192.168.10.2, Subnet Mask: 255.255.255.0
    - PC2: 192.168.10.3, Subnet Mask: 255.255.255.0
    - PC3: 192.168.10.4, Subnet Mask: 255.255.255.0

5. **Configure IP Addresses**:
   - Click on each PC → Desktop → IP Configuration → Enter IP and Subnet Mask
6. **Verify LAN Communication**:
   - Go to PC0 → Command Prompt
   - Type `ping 192.168.10.2` to test connectivity with PC1
   - Repeat pinging all other PCs

Sample Output:

```
C:\>ping 192.168.10.2

Pinging 192.168.10.2 with 32 bytes of data:
Reply from 192.168.10.2: bytes=32 time<1ms TTL=128
Reply from 192.168.10.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.10.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)
```

Result:
A Local Area Network was successfully simulated using Cisco Packet Tracer. All end devices were configured within the same network and verified to communicate with each other.

Conclusion:
This experiment demonstrated the creation of a simple LAN topology. It reinforced the concepts of IP addressing, switch-based connectivity, and intra-network communication testing using the `ping` command.

.

Experiment 4: Simulation of Stop-And-Wait Protocol using NS2

Aim:
To simulate the Stop-and-Wait ARQ (Automatic Repeat Request) protocol using NS2 and observe its behavior with respect to reliability and efficiency.

Apparatus / Software Required:

- NS2 (Network Simulator 2) installed on Linux OS (e.g., Ubuntu)
- Text Editor (e.g., gedit, nano) for writing TCL script
- XGraph / NAM (Network Animator) for visualization

Theoretical Background:

- **Stop-and-Wait ARQ Protocol**:
  A fundamental protocol used for reliable data transmission where:
  - Sender sends **one frame** and waits for an **ACK**.
  - If ACK is received correctly, the sender sends the next frame.
  - If not received within the **timeout period**, the frame is **retransmitted**.
- **Features**:
  - **Reliability**: Guarantees correct delivery through retransmissions.
  - **Timers**: Detect missing ACKs and trigger retransmission.
  - **Sequence Numbers**: Distinguish frames and their respective acknowledgments.
  - **Efficiency**: Decreases as propagation delay increases since sender remains idle while waiting for ACK.

Procedure (NS2 Simulation):

1. **Create a TCL Script** for Stop-and-Wait Simulation:
   - Define a simple two-node network (sender and receiver).
   - Implement UDP agent and error model.
   - Use timers or events to simulate Stop-and-Wait behavior manually (as NS2 doesn't have built-in Stop-and-Wait).
   - Example filename: `stopwait.tcl`

2. **Sample TCL Code Skeleton**:

```
# Create simulator
set ns [new Simulator]

# Open trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

# Create nodes
set n0 [$ns node]
set n1 [$ns node]

# Create link
$ns duplex-link $n0 $n1 1Mb 10ms DropTail

# Create UDP agent and attach to n0
set udp [new Agent/UDP]
$ns attach-agent $n0 $udp

# Create Null agent for n1
set null [new Agent/Null]
$ns attach-agent $n1 $null
$ns connect $udp $null
```

```
# Generate data using CBR

set cbr [new Application/Traffic/CBR]

$cbr set packetSize_ 1000

$cbr set interval_ 1.0

$cbr attach-agent $udp


# Schedule events

$ns at 0.5 "$cbr start"

$ns at 10.0 "$cbr stop"


# End simulation

$ns at 11.0 "finish"

proc finish {} {

   global ns tracefile

   $ns flush-trace

   close $tracefile

   exec nam out.nam &

   exit 0

}


$ns run
```

3. **Run the Simulation**:
   - Save the file as `stopwait.tcl`

   - In terminal:
   - `ns stopwait.tcl`

4. **View Output**:
   - Use `NAM` to visualize the animation.
   - Use `XGraph` if graph output is generated.
   - Observe packet sending, ACK reception, retransmissions due to timeout.

Sample Output (Observations):

- Data packet sent → ACK received → next packet sent.
- If no ACK received within time → **Retransmit** same packet.
- If packet drops are introduced via error model:
  - Sender waits → timeout → retransmit.
- Only **one packet** is in transit at any time.

Result:
The Stop-and-Wait ARQ protocol was successfully simulated using NS2. The protocol behavior with respect to reliability (acknowledgments and retransmissions) and efficiency (waiting time) was observed.

Conclusion:
The Stop-and-Wait protocol is simple and reliable but can be inefficient in high-delay networks due to idle waiting time. NS2 simulation demonstrated the protocol's timing and retransmission behavior clearly.

Experiment 5: Examining Network Address Translation (NAT) using Cisco Packet Tracer

Aim:
To configure and examine Network Address Translation (NAT) in Cisco Packet Tracer, allowing private IP addresses to communicate with public networks.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- PC or Laptop

Theoretical Background:

- **Network Address Translation (NAT)** allows private IP addresses to access public networks like the internet by translating private IPs to a public IP.
- **Types of NAT**:
  - **Static NAT**: One-to-one mapping between private and public IP.
  - **Dynamic NAT**: Many-to-many mapping using a pool of public IPs.
  - **PAT (Port Address Translation)** or Overloading: Many-to-one mapping where multiple private IPs share one public IP using different ports.
- **Benefits of NAT**:
  - Conserves public IP addresses.
  - Adds a layer of security by hiding internal IPs.
  - Enables devices in private networks to access external networks.

Procedure (Using Dynamic NAT):

1. **Design the Topology**:
   - Add 3 PCs (e.g., PC0, PC1 in private LAN and PC2 in external network).
   - Add 1 Router and 1 Switch.
2. **Connect the Devices**:
   - PC0 and PC1 ↔ Switch ↔ Router (interface G0/0)
   - Router (interface G0/1) ↔ PC2 (simulating external/public network)
3. **Assign IP Addresses**:
   - **Inside Network**:
     - PC0: 192.168.10.10 /24, Gateway: 192.168.10.1
     - PC1: 192.168.10.11 /24, Gateway: 192.168.10.1
     - Router G0/0: 192.168.10.1 /24
   - **Outside Network**:
     - Router G0/1: 200.0.0.1 /24

        - PC2: 200.0.0.2 /24, Gateway: 200.0.0.1

4. **Configure Router NAT Settings**:
    - Go to Router CLI:

    ```
    Router> enable
    Router# configure terminal
    Router(config)# interface g0/0
    Router(config-if)# ip address 192.168.10.1 255.255.255.0
    Router(config-if)# ip nat inside
    Router(config-if)# no shutdown
    Router(config-if)# exit

    Router(config)# interface g0/1
    Router(config-if)# ip address 200.0.0.1 255.255.255.0
    Router(config-if)# ip nat outside
    Router(config-if)# no shutdown
    Router(config-if)# exit

    Router(config)# access-list 1 permit 192.168.10.0 0.0.0.255
    Router(config)# ip nat pool MYPOOL 200.0.0.100 200.0.0.110
    netmask 255.255.255.0
    Router(config)# ip nat inside source list 1 pool MYPOOL
    Router(config)# exit
    ```

5. **Set IP Configurations on PCs**:
    - Set IPs and default gateways as described above.
6. **Test NAT Functionality**:
    - Open **Command Prompt** on PC0 or PC1 and type:
    - `ping 200.0.0.2`
    - You should receive replies, indicating NAT translation is working.
7. **Verify NAT Table**:
    - On router CLI:
    - `Router# show ip nat translations`

Sample Output:

```
Pro  Inside global        Inside local       Outside local       Outside global
icmp 200.0.0.100          192.168.10.10      200.0.0.2           200.0.0.2
```

Result:
NAT was successfully configured on the router, and private IP addresses were able to communicate with external public IPs.

Conclusion:
This experiment demonstrated the working of Network Address Translation (NAT) using Cisco Packet Tracer. NAT effectively allowed internal hosts to communicate with an external network using public IPs, ensuring both connectivity and security.

Experiment 6: Deploying Routing Information Protocol (RIP) to Route Packets using Cisco Packet Tracer

Aim:

To configure and deploy the Routing Information Protocol (RIP) on multiple routers using Cisco Packet Tracer to enable dynamic routing between different networks.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- PC or Laptop

Theoretical Background:

- **Routing Information Protocol (RIP):**
  RIP is a **distance-vector routing protocol** that uses **hop count** as a routing metric. It updates routing tables by broadcasting them every 30 seconds.
- **Key Characteristics of RIP**:
  - Maximum hop count is **15** (16 is unreachable).
  - Supports **equal-cost load balancing**.
  - Ideal for small networks due to its simplicity.
- **Dynamic Routing vs Static Routing**:
  - In **dynamic routing**, routers exchange routing information automatically.
  - RIP allows routers to adapt to network changes without manual intervention.

Procedure:

1. **Design the Network Topology**:
   - Use **3 Routers (R1, R2, R3)** connected in series.
   - Add **1 PC** to each router to represent different networks.
   - Connect routers and PCs using switches (optional for clarity).
2. **Assign IP Addresses**:

Device	Interface	IP Address	Subnet Mask
R1	G0/0	192.168.1.1	255.255.255.0

Device	Interface	IP Address	Subnet Mask
R1	G0/1	10.0.0.1	255.0.0.0
R2	G0/0	10.0.0.2	255.0.0.0
R2	G0/1	172.16.0.1	255.255.0.0
R3	G0/0	172.16.0.2	255.255.0.0
R3	G0/1	192.168.3.1	255.255.255.0
PC0	—	192.168.1.2	255.255.255.0
PC1	—	192.168.3.2	255.255.255.0

3. **Basic Configuration on Routers**:
   Example for **Router R1**:

```
Router> enable
Router# configure terminal
Router(config)# interface g0/0
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit

Router(config)# interface g0/1
Router(config-if)# ip address 10.0.0.1 255.0.0.0
Router(config-if)# no shutdown
Router(config-if)# exit

Router(config)# router rip
Router(config-router)# version 2
Router(config-router)# network 192.168.1.0
Router(config-router)# network 10.0.0.0
Router(config-router)# exit
```

Repeat similar configurations on **R2** and **R3** using appropriate IPs and networks.

4. **Set IP Configuration on PCs**:

   - PC0: IP – 192.168.1.2, Gateway – 192.168.1.1
   - PC1: IP – 192.168.3.2, Gateway – 192.168.3.1

5. **Verify RIP Configuration**:
   - On any router:
   - `Router# show ip route`
   - You should see **RIP routes (R)** listed.
6. **Test End-to-End Communication**:
   - From PC0, open the command prompt and:
   - `ping 192.168.3.2`
   - Observe successful replies if routing is properly configured.

Sample Output:

```
Pinging 192.168.3.2 with 32 bytes of data:
Reply from 192.168.3.2: bytes=32 time<1ms TTL=128
...
```

Routing Table on R1:

```
R    172.16.0.0 [120/1] via 10.0.0.2, 00:00:10, GigabitEthernet0/1
R    192.168.3.0 [120/2] via 10.0.0.2, 00:00:10, GigabitEthernet0/1
```

Result:
RIP was successfully configured, allowing routers to exchange routing information dynamically and enabling end-to-end communication across different networks.

Conclusion:
This experiment demonstrated the use of Routing Information Protocol (RIP) in Cisco Packet Tracer. It confirmed that dynamic routing can simplify network administration and provide fault tolerance in multi-router environments.

Experiment 7: TCP and Three-Way Handshake using Cisco Packet Tracer

Aim:
To observe the functioning of the **TCP three-way handshake** mechanism in Cisco Packet Tracer during the establishment of a reliable connection between two hosts.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- PC or Laptop

Theoretical Background:

- **Transmission Control Protocol (TCP):**
  TCP is a connection-oriented protocol that ensures **reliable and ordered delivery** of data between applications. Before any data is transmitted, a **three-way handshake** is used to establish the connection.
- **Three-Way Handshake Process**:
  1. **SYN**: The client sends a synchronize (SYN) packet to the server to initiate a connection.
  2. **SYN-ACK**: The server acknowledges the SYN and responds with a SYN-ACK.
  3. **ACK**: The client sends an ACK back, completing the handshake.
- This mechanism is essential for:
  - Ensuring both parties are ready for communication.
  - Establishing initial sequence numbers for tracking data segments.

Procedure:

1. **Open Cisco Packet Tracer.**
2. **Add Devices**:
   - Add 2 PCs: **PC0 (Client)** and **PC1 (Server)**.
   - Add 1 Switch to connect both PCs.
3. **Connect Devices**:
   - Use **Copper Straight-Through** cables to connect PC0 and PC1 to the switch.
4. **Assign IP Addresses**:
   - PC0: 192.168.1.10, Subnet Mask: 255.255.255.0
   - PC1: 192.168.1.20, Subnet Mask: 255.255.255.0
5. **Enable Simulation Mode**:

- Switch to **Simulation Mode** in Packet Tracer.

6. **Initiate TCP Communication**:
    - On **PC0**, open the **Web Browser** (or use Command Prompt and type `telnet 192.168.1.20` if Telnet is configured).
    - Enter the IP address of **PC1** in the browser or use `ping` to trigger communication.
7. **Observe TCP Handshake**:
    - In the **Event List Filters**, check only **TCP** and **ARP**.
    - Click **"Capture / Forward"** to step through the simulation.
    - Watch for:
        - **SYN** packet from PC0 to PC1
        - **SYN-ACK** packet from PC1 to PC0
        - **ACK** packet from PC0 to PC1

Sample Output (Simulation Events):

Time	Event	Info
0.0s	TCP	PC0 → PC1: SYN
0.1s	TCP	PC1 → PC0: SYN, ACK
0.2s	TCP	PC0 → PC1: ACK

Once the three-way handshake completes, a connection is established and data transfer begins.

Result:
The **TCP three-way handshake** was successfully observed in Cisco Packet Tracer. The client initiated the connection, the server responded, and the client acknowledged to complete the setup.

Conclusion:
This experiment demonstrated how TCP establishes a reliable connection using a three-step handshake mechanism. It highlighted the roles of SYN, SYN-ACK, and ACK packets in initiating reliable communication between hosts.

Experiment 8: Examine HTTP GET Response Interactions with PCAP Files using Cisco Packet Tracer

Aim:
To simulate an HTTP session in Cisco Packet Tracer and analyze the GET and response interactions by exporting and examining the generated PCAP file.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- Wireshark (for PCAP file analysis – optional)
- PC or Laptop

Theoretical Background:

- **HTTP (Hypertext Transfer Protocol)** is an application-layer protocol used for communication between web clients (browsers) and servers.
- **HTTP GET**: A request method used by clients to retrieve resources (e.g., web pages) from a server.
- **HTTP Response**: The server responds with data and status codes (e.g., 200 OK).
- **PCAP (Packet Capture)**: A file format used to capture and analyze packet-level network data using tools like Wireshark.

Procedure:

1. **Launch Cisco Packet Tracer.**
2. **Add and Connect Devices**:
   - **PC0 (Client)**
   - **Server0 (HTTP Server)**
   - **Switch**
   - Use **Copper Straight-Through** cables to connect both devices to the switch.
3. **Configure IP Addresses**:
   - PC0: 192.168.1.10, Subnet Mask: 255.255.255.0, Gateway: 192.168.1.1
   - Server0: 192.168.1.20, Subnet Mask: 255.255.255.0
4. **Configure HTTP Service on Server0**:
   - Click on Server0 → Services → HTTP → Ensure HTTP is **ON**.
5. **Configure PC0 to Use Browser**:
   - Click on PC0 → Desktop → Web Browser

- Enter the Server IP: `http://192.168.1.20` and click Go

6. **Switch to Simulation Mode**:
    - Change mode to **Simulation**
    - Use **Capture/Forward** to observe packet flow
7. **Filter for HTTP Only**:
    - In the Event List → Edit Filters → Enable only **HTTP** and **TCP**
8. **Observe the HTTP Exchange**:
    - PC0 sends an HTTP **GET** request to Server0
    - Server0 responds with HTTP **OK (200)** and web content
9. **Export the Simulation to PCAP File**:
    - Click **File** → **Export Packet Trace** → **PCAP file**
    - Save the file for external analysis in **Wireshark**
10. **(Optional)** Open the PCAP file in **Wireshark**:

- Observe the full TCP 3-way handshake
- Locate the **HTTP GET** and **HTTP/1.1 200 OK** response

Sample Output (HTTP Exchange in Packet Tracer):

Time	Source	Destination	Protocol	Info
0.0s	PC0	Server0	TCP	SYN (3-way handshake)
0.1s	PC0	Server0	HTTP	GET /index.html HTTP/1.1
0.2s	Server0	PC0	HTTP	HTTP/1.1 200 OK

Result:
An HTTP GET and response exchange was successfully simulated between a client and a server. The interaction was captured and saved as a PCAP file for analysis.

Conclusion:
This experiment demonstrated how HTTP works over TCP and how packet exchanges can be monitored using Cisco Packet Tracer and exported as PCAP files. This enables deeper analysis of application-layer protocols and network traffic.

Experiment 9: Domain Name System (DNS) Client Interaction with PCAP Files using Cisco Packet Tracer

---

Aim:

To simulate a DNS client-server interaction using Cisco Packet Tracer and examine the DNS request and response behavior by capturing and analyzing the packet exchange using PCAP files.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- Wireshark (optional, for deeper PCAP analysis)
- PC or Laptop

Theoretical Background:

- **Domain Name System (DNS)** translates domain names (like `www.example.com`) into IP addresses.
- When a client types a URL into a browser, a **DNS request (query)** is sent to a DNS server.
- The DNS server responds with a **DNS response**, which includes the corresponding IP address.
- This interaction typically uses **UDP on port 53**.
- **PCAP (Packet Capture)** files are used to analyze such communication for educational or troubleshooting purposes.

Procedure:

1. **Open Cisco Packet Tracer.**
2. **Add Devices**:
   - **PC0 (Client)**
   - **Server0 (DNS Server)**
   - **Switch**
3. **Connect Devices**:
   - Use **Copper Straight-Through** cables to connect both devices to the switch.
4. **Configure IP Addresses**:
   - PC0: IP – 192.168.1.10, Subnet Mask – 255.255.255.0, DNS – 192.168.1.20
   - Server0: IP – 192.168.1.20, Subnet Mask – 255.255.255.0
5. **Configure DNS Service on Server0**:

- Click on Server0 → **Services** → **DNS**
- Add a DNS record:
- `Name: www.example.com`
- `Address: 192.168.1.100`
- Ensure DNS service is turned **ON**

6. **Switch to Simulation Mode**:
   - Change to **Simulation Mode** in Cisco Packet Tracer.
7. **Initiate DNS Query**:
   - Click PC0 → Desktop → Web Browser
   - Enter `http://www.example.com` in the address bar
8. **Capture DNS Interaction**:
   - Use **Capture/Forward** to observe DNS packets
   - In the Event Filters, enable only **DNS** and **UDP**
9. **Export the Packet Capture**:
   - Click **File** → **Export Packet Trace** → **PCAP file**
   - Save it for further analysis
10. **(Optional)** Open the PCAP file in **Wireshark**:
    - Filter using `dns`
    - Analyze the **Standard Query** and **Standard Response**

Sample Output (Packet Tracer Event List):

Time	Source	Destination	Protocol	Info
0.0s	PC0	Server0	DNS	Standard query A www.example.com
0.1s	Server0	PC0	DNS	Standard query response A 192.168.1.100

Result:

A DNS request from the client and its response from the server were successfully simulated. The packet exchange was exported as a PCAP file for further inspection.

Conclusion:

This experiment demonstrated the process of domain resolution using DNS. It highlighted how client devices communicate with DNS servers, and how these interactions can be captured and analyzed using Cisco Packet Tracer and tools like Wireshark.

Experiment 10: Measuring Quality of Service (QoS) in Ad Hoc Networks using Cisco Packet Tracer

---

Aim:

To configure a simple ad hoc wireless network in Cisco Packet Tracer and analyze Quality of Service (QoS) parameters such as **latency**, **throughput**, and **packet loss** under different traffic conditions.

Apparatus / Software Required:

- Cisco Packet Tracer (Version 7.x or later)
- PC or Laptop

Theoretical Background:

- **Quality of Service (QoS):**
  Refers to the performance level of service experienced by users in a network. Key parameters include:
  - **Latency**: Delay in packet delivery.
  - **Throughput**: Amount of data successfully delivered over time.
  - **Packet Loss**: Percentage of lost data packets in transit.
- **Ad Hoc Wireless Network**:
  A decentralized wireless network where devices communicate directly with each other without fixed infrastructure like access points or routers.
- **Traffic Types**:
  - **Voice**: Low latency, low jitter requirements.
  - **Video**: High bandwidth, low packet loss required.
  - **Data**: Tolerates latency, less sensitive to jitter.

Procedure:

1. **Open Cisco Packet Tracer.**
2. **Add Wireless Devices**:
   - Add 3 **laptops**: Laptop0, Laptop1, Laptop2.
   - Add a **Wireless Router (e.g., Linksys)** for basic control.
   - Place laptops within the signal range of the router.
3. **Configure Wireless Router**:
   - Set **SSID**: `AdHocNet`

   - Enable **DHCP** or assign static IPs.
4. **Configure Wireless Interfaces on Laptops**:
   - Laptop0: Static IP – 192.168.0.2
   - Laptop1: Static IP – 192.168.0.3
   - Laptop2: Static IP – 192.168.0.4
   - All Subnet Masks: 255.255.255.0
   - Gateway: 192.168.0.1 (Router IP)
5. **Set Up Different Traffic Types**:
   - Configure **Laptop0 → Laptop1**: Use a **voice or video** application (VoIP or streaming).
   - Configure **Laptop2** to send file transfers (data traffic) to Laptop1.
   - You can use **Simulation Mode** to insert packets (PDU) and simulate traffic manually.
6. **Enable Simulation Mode and Capture Traffic**:
   - Open **Simulation Mode**.
   - Use **Capture/Forward** to observe traffic flow.
   - Enable **TCP, UDP, ICMP, DNS, HTTP, VoIP** in Event List Filters.
7. **Monitor QoS Metrics**:
   - **Packet Delivery Time**: Observe time from source to destination.
   - **Packet Loss**: Drop some packets by overloading traffic.
   - **Throughput**: Observe number of packets successfully transmitted over time.

Sample Output (Simulation Events):

Time	Source	Destination	Protocol	Info
0.0s	Laptop0	Laptop1	VoIP	Call setup, RTP packet sent
0.1s	Laptop2	Laptop1	FTP	Data transfer packet
0.2s	Laptop1	Laptop0	VoIP	RTP packet received

Result:
QoS was observed and measured in an ad hoc wireless network with mixed traffic. Delay, packet drops, and delivery success rates were identified under simulated loads.

Conclusion:
This experiment demonstrated the practical observation of QoS in an ad hoc wireless network. It emphasized how different traffic types (voice, video, data) are affected by network load, showcasing the importance of QoS-aware network design.