

**SIES GST, NERUL, NAVI MUMBAI**

**DEPARTMENT OF COMPUTER ENGINEERING**

**LAB MANUAL**

**COMPUTER NETWORKS**

**(CSC503)**

**T.E. (COMPUTER ENGINEERING)**

**SEMESTER-V**

**(R-2016)-Ver1**

**COMPUTER ENGINEERING DEPARTMENT**

**DEPARTMENT’S VISION**

To be a centre of Excellence in Computer Engineering to fulfill the rapidly growing needs of the Society.

**DEPARTMENT’S MISSION**

M1: To Impart quality education to meet the professional challenges in the area of Computer Engineering.

M2: To create an environment for research, innovation, professional and social development.

M3: To nurture lifelong learning skills for achieving professional growth.

M4 To strengthen the alumni and industrial interaction for overall development of students.

**PROGRAMME EDUCATIONAL OBJECTIVES(PEOs)**

PEO1: Practice Computer engineering in core and muti-disciplinary domains.

PEO2: Exhibit leadership skills for professional growth.

PEO3: Pursue higher Studies for career advancement

**PROGRAMME OUTCOMES (POs)**

PO1: Engineering knowledge: Apply the knowledge of mathematics science engineering fundamentals and an mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual and as a member or leader in diverse teams and individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: long learning: Recognize the need for and have the Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

**PROGRAMME SPECIFIC OUTCOMS (PSOs )**

PSO1: To apply computational and logical skills to solve Computer engineering problems.

PSO2: To develop interdisciplinary skills and acquint with cutting edge technologies in software industries.

**General INSTRUCTIONS (Do's And Don'ts)**

1. Wearing ID-Card is compulsory.
2. Keep your bag at the specified place.
3. Shut down the system after use.
4. Place the chairs in proper position before leaving the laboratory.
5. Report failure/Non-working of equipment to Faculty In-charge / Technical Support staff immediately.
6. Know the location of the fire extinguisher and the FIRST-AID Box and how to use then in case of an emergency.
7. Do not eat or drink in the laboratory.
8. Do not litter in the laboratory.
9. Avoid stepping on electrical wires or any other computer cables.
10. Do not open the system unit casing or monitor casing particularly when the power in turn ON.
11. Do not insert metal objects such as clips, pins and needles into the computer casing. They may cause fire.
12. Do not remove anything from laboratory without permissions.
13. Do not touch, connect or disconnect any plug or cable without permission.

**LAB OUTCOMES (LO)**

|  |
| --- |
| LO1: Design and setup networking environment in Linux. |
| LO2: Simulate using network tools and simulators such as ns2, Wireshark etc. to explore networking algorithms and protocols. |
| LO3: Implement programs using core programming APIs for understanding networking concepts. |
| LO4: Design a network case study using CISCO packet tracer. |

**LAB ARTICULATION MATRIX (MAPPING WITH PO & PSO)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Lab Outcome | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 | PSO2 |
| LO1 | Design and setup networking environment in Linux. |  |  |  | **3** | 2 |  |  | 3 | 3 | 3 |  | 1 | 3 | 3 |
| LO2 | Simulate using network tools and simulators such as ns2, Wireshark etc. to explore networking algorithms and protocols. |  |  |  |  | **3** |  |  | 3 | 3 | 3 |  |  | 3 | 3 |
| LO3 | Implement programs using core programming APIs for understanding networking concepts. |  |  |  |  | 3 |  |  | 3 | 3 | 3 |  | **2** | 3 | 3 |
| LO4 | Design a network case study using CISCO packet tracer. |  |  |  |  | 3 |  |  | 3 | 3 | 3 |  | **1** | 3 | 3 |
| Average | |  |  |  | 3 | 2.75 |  |  | 3 | 3 | 3 |  | 1.33 | 3 | 3 |

**Laboratory Assessment**

**Academic Year**: \_\_\_\_\_\_\_\_\_\_ **Class/Sem:** **Div:** **Batch**: \_\_\_\_

**Student Name**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Roll No:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Course Name:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Course Code**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| **1. Preparedness and Efforts/ Preparation and Knowledge** | | |
| 3: Well prepared and puts efforts. | 2: Not prepared but puts efforts or prepared but doesn't put efforts | 1: Neither prepared nor puts efforts |
| **2. Presentation of output/ Accuracy and Neatness of Documentation** | | |
| 3: Uses all perfect Instructions / Interrupts/Presented well. | 2: Uses some perfect Instructions / Interrupts/ moderate presentation. | 1: Doesn't use any of the proper Instruction / Interrupt/ Not presented properly. |
| **3. Results/ Participation in Practical Performance** | | |
| 3: Participate and gets proper results | 2: Participate but doesn't get proper result or gets result but with the help of faculty in-charge | 1: Neither Participate nor gets the results |
| **4. Punctuality** | | |
| 3: Get the experiment checked in-time and is always in-time to the lab sessions | 2: Some time delays the experiment checking or is late to the lab sessions for few times | 1: Most of the time delays experiment checking and / or comes late for lab sessions |
| **5. Lab Ethics** | | |
| 3: Follows proper lab ethics by keeping the lab clean and placing things at their right place | 2: Sometimes doesn't follow the lab ethics | 1: Most of the times makes the lab untidy and keeps things at wrong place |

**EVALUATION:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Performance Indicator/ Expt. No** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Total |
| LO1 | LO1 | LO2 | LO2 | LO3 | LO4 | LO3 | LO3,LO4 | NA | NA |  |
| **1. Preparedness and Efforts/ Preparation and Knowledge** |  |  |  |  |  |  |  |  |  |  |  |
| **2. Presentation of output/ Accuracy and Neatness of Documentation** |  |  |  |  |  |  |  |  |  |  |  |
| **3. Debugging and results/ Participation in Practical Performance** |  |  |  |  |  |  |  |  |  |  |  |
| **4. Punctuality** |  |  |  |  |  |  |  |  |  |  |  |
| **5. Lab Ethics** |  |  |  |  |  |  |  |  |  |  |  |
| **Total** |  |  |  |  |  |  |  |  |  |  |  |
| **Average** |  |  |  |  |  |  |  |  |  |  |  |

Exceed Expectations (3), Meet Expectations (2). Below Expectations (1)

Faculty In-charge

**LIST OF EXPERIMENTS**

|  |  |
| --- | --- |
| **Sr No** | **EXPERIMENT TITLE** |
| 1 | Use basic networking commands in Linux (ping, tracert, nslookup, netstat, ARP, RARP, ip, ifconfig, dig, route ) |
| 2 | Implementation of Cyclic Redundancy Check (CRC) |
| 3 | Implementation of Hamming Code |
| 4 | Socket programming using TCP and UDP |
| 5 | Setup a network and configure IP addressing, subnetting, masking using CISCO packet tracer |
| 6 | Use simulator NS2 to understand functioning of TCP and UDP. |
| 7 | Implement Stop and wait protocol using NS2. |
| 8 | Implementation of leaky bucket algorithm for congestion control |
| 9 | Perform Remote login using Telnet server |
| 10 | Configuration of FTP server |

# PROGRAM NO. 1 : Use basic networking commands

**AIM : -** Use basic networking commands in Linux (ping, tracert, nslookup, netstat, ARP, RARP, ip, ifconfig.

**THEORY :-**

### 1. Ping

The PING utility tests connectivity between two hosts. PING uses a special protocol called the Internet Control Message Protocol (ICMP)to determine whether the remote machine (website, server, etc.) can receive the test packet and reply.

Also a great way to verify whether you have TCP/IP installed and your Network Card is working.

Type: **PING 127.0.0.1**

This tells that TCP/IP is working as well as Network Card.

To test out connectivity to a website all you have to do is type: **ping espn.com**

The results should tell you if the connection was successful or if you had any lost packets.

Packet loss describes a condition in which data packets appear to be transmitted correctly at one end of a connection, but never arrive at the other. Why? Well, there are a few possibilities.

The network connection might be poor and packets get damaged in transit or the packet was dropped at a router because of internet congestion. Some Internet Web servers may be configured to disregard ping requests for security purposes.

Note the IP address of espn.com -- 199.181.132.250. You can also ping this address and get the same result.

However, Ping is not just used to test websites. It can also test connectivity to various servers: DNS, DHCP, your Print server, etc. As you get more into networking you'll realize just how handy the Ping utility can be.

**2. Tracert**

Tracert is very similar to Ping, except that Tracert identifies pathways taken along each hop, rather than the time it takes for each packet to return (ping).

If you know there are normally 4 routers but Tracert returns 8 responses, you know your packets are taking an indirect route due to a link being down.

### 3. ARP

The ARP utility helps diagnose problems associated with the Address Resolution Protocol (ARP).

TCP/IP hosts use ARP to determine the physical (MAC) address that corresponds with a Specific IP address.

Type **arp** with the **– a** option to display IP addresses that have been resolved to MAC addresses recently.

### 4. Netstat

Netstat (Network Statistics) displays network connections (both incoming and outgoing), routing tables, and a number of network interface statistics.

**Netstat –s** provides statistics about incoming and outgoing traffic.

### 5. Nbtstat

Nbtstat (NetBios over TCP/IP) enables you to check information about NetBios names.

It helps us view the NetBios name cache (nbtstat -c) which shows the NetBios names and the corresponding IP address that has been resolved (nbtstat -r) by a particular host as well as the names that have been registered by the local system (nbtstat –n).

### 6. NSLookup

NSLookup provides a command-line utility for diagnosing DNS problems. In its most basic usage, NSLookup returns the IP address with the matching host name.

### 7. IPConfig

Not part of the TCP/IP utilities but it is useful to show current TCP/IP settings.

The IPConfig command line utility will show detailed information about the network you are connected to. It also helps with reconfiguration of your IP address through release and renew.

Let's say you want to know what you're IP address is -- **ipconfig** is what you type in the command prompt.

**ipconfig** will give a quick view of you IP address, your subnet mask and default gateway.

**ipconfig /all** will give you more detailed information.

Through **ipconfig /all** we can find DNS severs, if we have DHCP enabled, MAC Address, along with other helpful information. All good things to know if we have trouble getting connected to the internet.

Other IPConfig tools that are helpful include **ipconfig /release** and **ipconfig /renew**. But before I get into this let's discuss how we actually get an IP Address.

Unless you know your static IP address you'll want to stick to the option for automatically obtaining the IP address. If you have it set to automatic your computer will be issued an IP through a DHCP server.

**Note:** ipconfig /release renew won't work if you manually assigned your IP addresses.

# CONCLUSION:

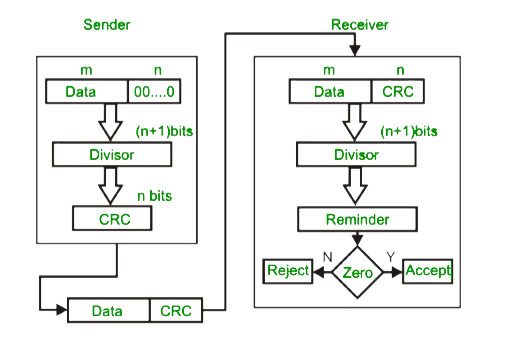
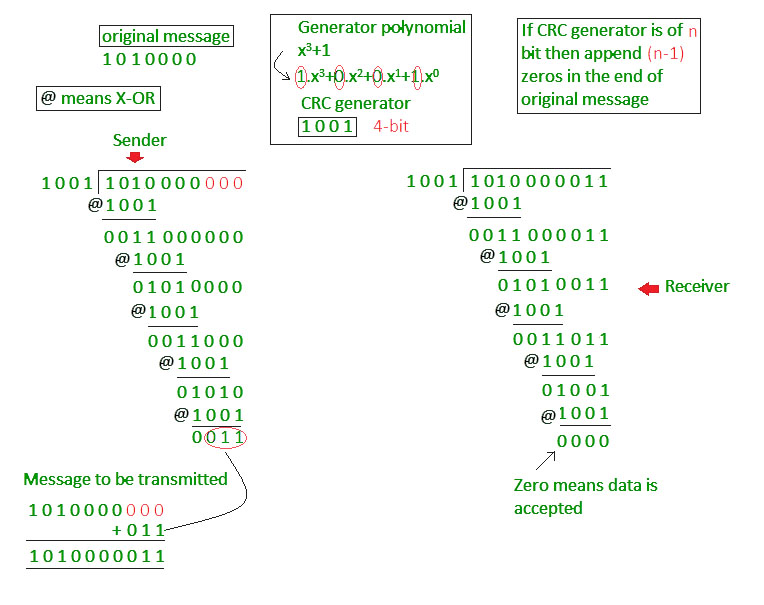
# PROGRAM NO. 2: Cyclic Redundancy Check (CRC)

# AIM: Implementation of Cyclic Redundancy Check (CRC)

# THEORY:

**Cyclic redundancy check (CRC)**

* Unlike checksum scheme, which is based on addition, CRC is based on binary division.
* In CRC, a sequence of redundant bits, called cyclic redundancy check bits, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
* At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
* A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.

[](https://cdncontribute.geeksforgeeks.org/wp-content/uploads/detect14.jpg)  
  
 **Example :**  
[](https://contribute.geeksforgeeks.org/wp-content/uploads/detect15.jpg)

**Test cases-**

**The program should be tested against the received message without error and with error.**

e.g. if the original message (1010000) and divisor is given as input, then program should show the sent message with checksum (1010000011).

Then take the message ( 1010000011 or 1010000010) received by the receiver and the program should check if the received message is accepted or discarded.

# CONCLUSION:

# PROGRAM NO. 3 : Hamming Code

# AIM:- Implementation of Hamming Code

**THEORY:-**

Hamming code is technique developed by R.W. Hamming for error correction. This method corrects the error by finding the state at which the error has occurred.

## Determining the positions of redundancy bits

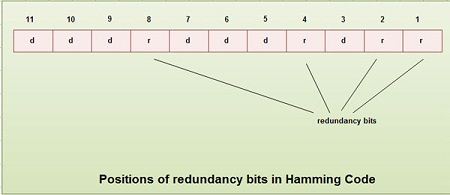
Till now, we know the exact number of redundancy bits required to be embedded with the particular data unit.

We know that to detect errors in a 7 bit code, 4 redundant bits are required.

Now, the next task is to determine the positions at which these redundancy bits will be placed within the data unit.

• These redundancy bits are placed at the positions which correspond to the power of2.

• For example in case of 7 bit data, 4 redundancy bits are required, so making total number of bits as 11. The redundancy bits are placed in position 1, 2, 4 and 8 as shown in fig.

[](http://ecomputernotes.com/images/Positions-of-redundancy-bits-in-Hamming-Code.jpg)

## Generating parity [information](http://ecomputernotes.com/fundamental/information-technology/what-do-you-mean-by-data-and-information)

• In Hamming code, each r bit is the VRC for one combination of data bits. rl is the VRC bit for one combination of data bits, r2 is the VRC for another combination of data bits and so on.

• Each data bit may be included in more than one VRC calculation.

• rl bit is calculated using all bits positions whose binary representation includes a 1 in the rightmost position.

• r2 bit calculated using all the bit positions with a 1 in the second position and so on.

• Therefore the various r bits are parity bits for different combination of bits.

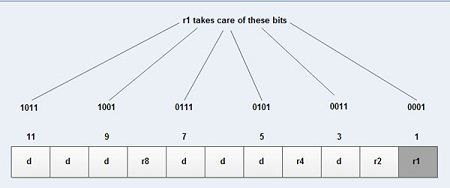
The various combinations are:

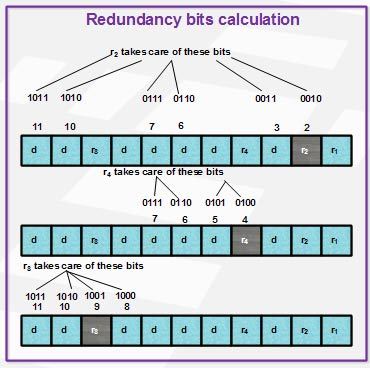
rl: bits 1,3,5, 7, 9, 11

r2 : bits 2, 3, 6, 7, 10, 11

r4 : bits 4, 5, 6, 7

r8 : bits 8, 9, 10, 11

[](http://ecomputernotes.com/images/Generating-parity-information.jpg)

[](http://ecomputernotes.com/images/Redundancy-bits-calculation.jpg)

## Example of Hamming Code Generation

Suppose a binary data 1001101 is to be transmitted. To implement hamming code for this, following steps are used:

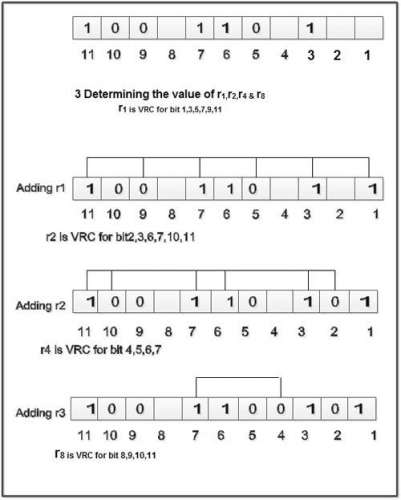
1. Calculating the number of redundancy bits required. Since number of data bits is 7, the value of r is calculated as

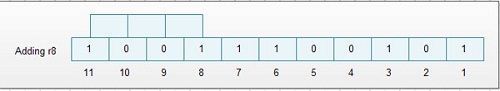
2r > m + r + 1

24 > 7 + 4 + 1

Therefore no. of redundancy bits = 4

2. Determining the positions of various data bits and redundancy bits. The various r bits are placed at the position that corresponds to the power of 2 *i.e.* 1, 2, 4, 8

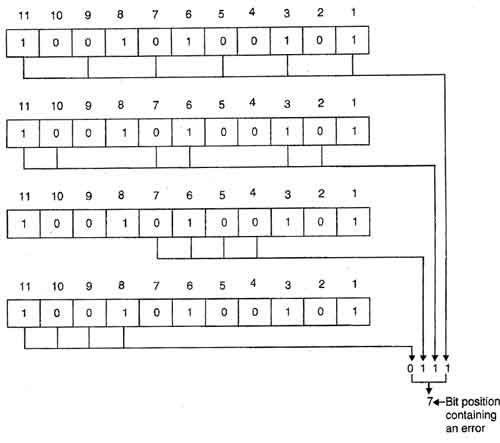
[](http://ecomputernotes.com/images/Example-of-Hamming-Code-Generation.jpg)



4. Thus data 1 0 0 1 1 1 0 0 1 0 1 with be transmitted.

## Error Detection & Correction

Considering a case of above discussed example, if bit number 7 has been changed from 1 to 0.The data will be erroneous.

[](http://ecomputernotes.com/images/Error-Detection--Correction.jpg)

Data sent: 1 0 0 1 1 1 0 0 1 0 1

Data received: 1 00 1 0 1 00 1 0 1 (seventh bit changed)

The receive takes the transmission and recalculates four new VRCs using the same set of bits used by sender plus the relevant parity (r) bit for each set as shown in fig.

Then it assembles the new parity values into a binary number in order of r position (r8, r4, *r2,*r1).

In this example, this step gives us the binary number 0111. This corresponds to decimal 7. Therefore bit number 7 contains an error. To correct this error, bit 7 is reversed from 0 to 1.

**CONCLUSION:-**

# PROGRAM NO. 4 : CLIENT/SERVER USING SOCKET

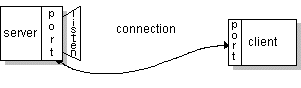
**AIM:-**Implementation of Client/Server using Socket.

**THEORY:-**

On the client-side: The client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system.

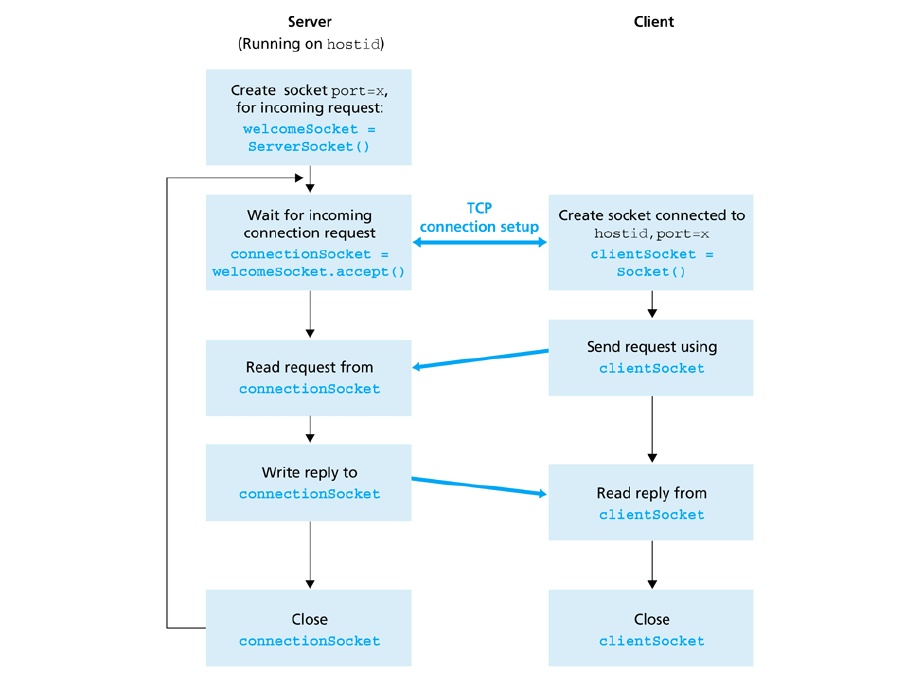


If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to the same local port and also has its remote endpoint set to the address and port of the client. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client.



On the client side, if the connection is accepted, a socket is successfully created and the client can use the socket to communicate with the server.

The client and server can now communicate by writing to or reading from their sockets.



**CONCLUSION:-**

# PROGRAM NO. 5 : CISCO PACKET TRACER

**AIM: -** Setup a network and configure IP addressing, subnetting, masking using CISCO packet tracer

**THEORY:-**

Download Cisco Packet Tracer. Once it is downloaded, install it using the default selections. After the installation, you can create a network topology to perform the hands-on lab exercises.

Create a network topology using Cisco Packet Tracer.

### Create a Network Topology

You can easily create a network topology using Cisco Packet Tracer. In the following sections, we are going to explain how to create a network topology that will contain four PCs, two switches, and two routers.

#### Adding PCs in Cisco Packet Tracer

To add PCs in Cisco Packet Tracer, you need to perform the following steps:

1. In the Cisco Packet Tracer console, click on the **PC** icon, click **Generic**, and then click in the logical view area to add a **Generic** PC.
2. Repeat the same step to add three more Generic PCs in the logical view area, as shown in the following figure.

Create a Network Topology in Cisco Packet Tracer

#### Adding Switches in Cisco Packet Tracer

1. To add a switch in Cisco Packet Tracer, click the **Switch** icon, select a switch type, such as **2960**, and then add the selected switch in the logical view area.
2. Repeat the same step to add one more switch.

#### Adding Routers in Cisco Packet Tracer

1. To add a router in Cisco Packet Tracer, click the **Router** icon, select a router type, such as **2811**, and then add the selected router in the logical view area.
2. Repeat the same step to add one more router.

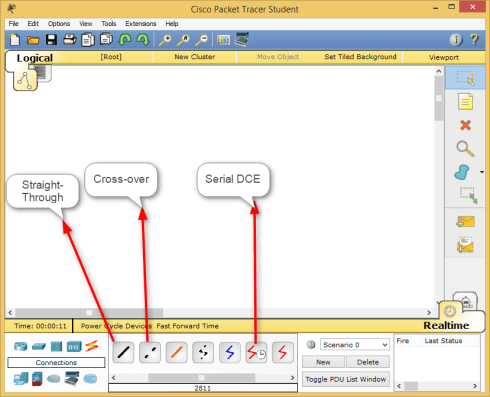
Note: Different types of router series provide different types of [features](https://protechgurus.com/new-features-of-windows-server-2016/) and limitations.

#### Understanding Connection Types in Cisco Packet Tracer

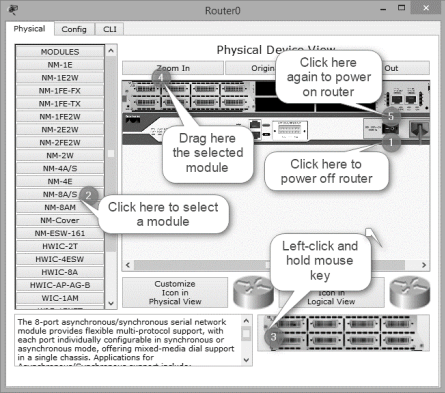
To connect devices in Cisco Packet Tracer, first, you need to understand the various types of cables (connections) used to connect network devices. Some of the common types of cables are:

1. **Straight-through**: Used to connect different types of devices (devices that use different wiring standards), such as Router-to-Switch and Switch-to-PC.
2. **Cross-over**: Used to connect same types of devices, such as router-to-router, PC-to-PC, and switch-to-switch.
3. **Serial DCE**: Used to connect router-to-router in a WAN network.
4. **Console**: Used to take console (using hyper terminal) of a router on a PC.

To see the various types of connections, click the **Connection** icon. Spend some time to understand the connections. Once you are familiar with the types of connections, connect the devices to create the network topology.

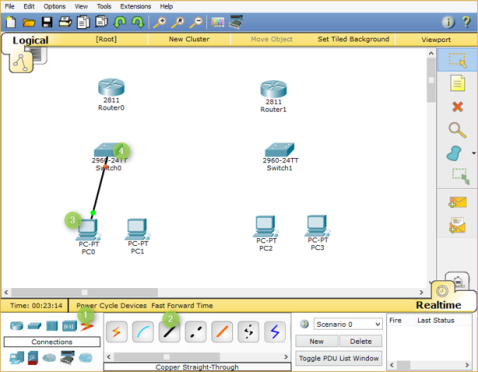
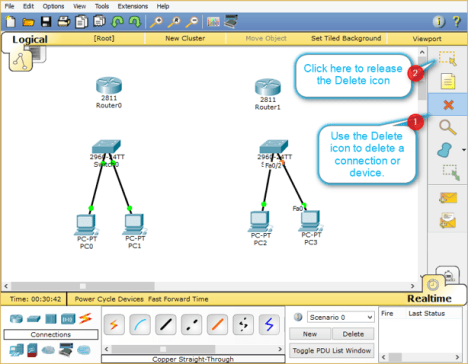
The following figure displays the various types of connections used to connect devices.

Since we have chosen the modular router (that allows you to modify the number of interfaces), you may need to customize the interfaces before it can be used to connect other network devices. To do this, double-click **Router0**, on the **Router0** properties dialog box, click the **Power** button to power off **Router0**.

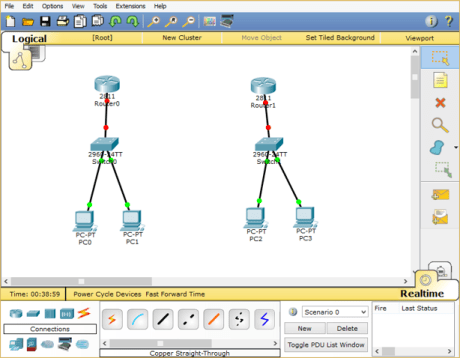
The following figure displays how to add a module in a router using Cisco Packet Tracer.

Now, open the **Router1** properties dialog box, add the same module to **Router1** also, and then close the **Router1** properties dialog box.

#### Connecting Devices in Cisco Packet Tracer

1. To connect devices in Cisco Packet Tracer, click the connection type icon, and selectan appropriate cable. For example, to connect **PC0** to **Switch0**, select the straight-through cable, click on **PC0**, select the **FastEthernet0** interface.
2. Next, click on **Switch0**, and then select the **FastEthernet0/1** interface. The following figure displays how to connect a PC to a switch in Cisco Packet Tracer.
3. Now, add **PC1** to **Switch0** using the **FastEthernet0/2** interface. Also, add **PC2** and **PC3** to the **FastEthernet0/1** and **FastEthernet0/2** interfaces of **Switch1,**respectively.
4. If you have connected a wrong device to a wrong interface, you can use the **Delete**option to delete a connection or device. The following figure displays how to use the **Delete** option to delete a device or connection in Cisco Packet Tracer.
5. Once, you have connected all the PCs to switches, now, connect **Switch0** to **Router0,** and **Switch1** to **Router1** using the straight-through cables.
6. Select the straight-through cable, click on **Switch0**, and then select**FastEthernet0/3** interface.
7. Click **Router0**and select the **FastEthernet0/0** interface.
8. Select again the straight-through cable, click on **Switch1**, and select**FastEthernet0/3** interface.
9. Next, click **Router1** and then select the **FastEthernet0/0** interface.

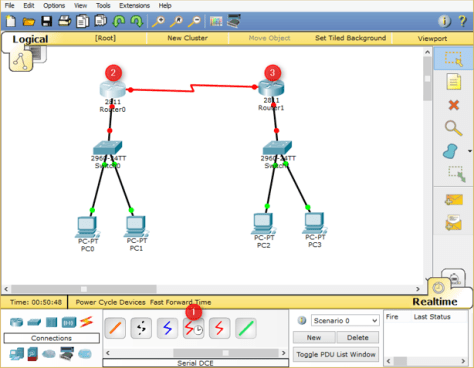
The following figure displays how to connect routers to switches to create a network topology.



### Interconnecting Routers in Cisco Packet Tracer

Now, connect **Router0** to **Router1** using the serial connection. To do this, you need to perform the following steps:

1. Select the **Serial DCE** cable, click on **Router0,** and select the **Serial1/0** interface.
2. Click on **Router1** and select the **Serial1/0** interface, as shown in the following figure.



In this post, you have learned how to [create a network topology in Cisco Packet Tracer.](https://protechgurus.com/create-a-network-topology-in-cisco-packet-tracer/) If you wish, you can save the created network topology for the later use. To do this, you need to perform the following steps:

1. In Cisco Packet Tracer, click File, and select **Save** **As**.
2. In the **File name** text box, type a name of the topology, and then click **Save**.

**CONCLUSION :**

# PROGRAM NO. 6 : Basic Wired topology In NS2.

**AIM: -** Use simulator NS2 to understand functioning of TCP and UDP.

**THEORY:-**

NS-2 stands for Network Simulator version 2.

IT is a discrete event simulator for networking research. It work at packet level, provide substantial support to simulate bunch of protocols like TCP, UDP, FTP, HTTP and DSR. It simulate wired and wireless network. It is primarily UNIX based.Use TCL as its scripting language.

* Create simulator instance

**set ns [new Simulator]**

- Usually the first non-comment statement in ns-2 script

- Initialize the packet format

- Create a scheduler (default is a calendar scheduler)

- Create a “null agent”

* Open file for NS tracing

**set tracefile [open out.tr w]**

**$ns trace-all $tracefile**

* Open file for nam tracing

**set nf [open out.nam w]**

**$ns namtrace-all $nf**

* Open your own trace file

**set my\_f [open my\_out.tr w]**

**puts $my\_f “[$ns now] [expr $x(1) + $y(1)]”**

* Creating nodes

**set node\_(h1) [$ns node]**

* Creating Link and Queue

**$ns duplex-link $node\_(h1) $node\_(r1) 10Mb 2ms DropTail**

* Start ns

**$ns run**

Last statement

* Stop ns

**exit 0**

**ALGORITHM:-**

1. Implement protocol models
2. Setup simulation scenario, i.e. create tcl file describing type of scenario, e.g. number of nodes, kind of agent working on nodes etc.
3. Run simulation, i.e. Run the tcl file
4. Analyze simulation results, i.e. by GNU Awk and gnuplot

**EXECUTION:-**

* Open the notepad
* Write the tcl script
* Save as “example-1.tcl” select “All files” in the **save as type** box and save into the bin directory of NSi.e. c:\NS2\bin\
* Open the command prompt at c:\NS2\bin\
* Type the following at the command prompt:

C:\NS2\bin> ns example-1.tcl

**TCP/UDP :**

* Create simulator instance
* Setup a TCP Agent connection

**set tcp [new Agent/TCP]**

**$tcp set class\_ 2**

**$ns attach-agent $n0 $tcp**

* Setup a TCP sink

**set sink [new Agent/TCPSink]**

**$ns attach-agent $n3 $sink**

* Connect the TCP Agent and Sink

**$ns connect $tcp $sink**

**$tcp set fid\_ 1**

* Setup a FTP over TCP connection

**set ftp [new Application/FTP]**

**$ftp attach-agent $tcp**

**$ftp set type\_ FTP**

* Setup a UDP Agent connection

**set udp [new Agent/UDP]**

**$ns attach-agent $n1 $udp**

**set null [new Agent/Null]**

**$ns attach-agent $n3 $null**

**$ns connect $udp $null**

**$udp set fid\_ 2**

* Setup a CBR over UDP connection

**set cbr [new Application/Traffic/CBR]**

**$cbr attach-agent $udp**

**$cbr set type\_ CBR**

**$cbr set packet\_size\_ 1000**

**$cbr set rate\_ 1mb**

**$cbr set random\_ false**

* Schedule events for the CBR and FTP agents

**$ns at 0.1 "$cbr start"**

**$ns at 1.0 "$ftp start"**

**$ns at 4.0 "$ftp stop"**

**$ns at 4.5 "$cbr stop"**

**CONCLUSION:-**

# PROGRAM NO. 7 : STOP AND WAIT ARQ PROTOCOL USING NS2

**AIM:-** Implement Stop and wait protocol using NS2.

**THEORY:-**

**Stop-and-wait** is a method used in telecommunications to send information between two connected devices. It ensures that information is not lost due to dropped packets and that packets are received in the correct order. It is the simplest kind of automatic repeat-request (ARQ) method. A stop-and-wait ARQ sender sends one frame at a time; it is a special case of the general sliding window protocol with both transmit and receive window sizes equal to 1. After sending each frame, the sender doesn't send any further frames until it receives an acknowledgement (ACK) signal. After receiving a good frame, the receiver sends an ACK. If the ACK does not reach the sender before a certain time, known as the timeout, the sender sends the same frame again.

**NS2 Code:-**

|  |
| --- |
|  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# stop and wait protocol in normal situation   
# features : labeling, annotation, nam-graph, and window size monitoring

set ns [new Simulator]

set n0 [$ns node]   
set n1 [$ns node]

$ns at 0.0 "$n0 label Sender"   
$ns at 0.0 "$n1 label Receiver"

set nf [open A1-stop-n-wait.nam w]   
$ns namtrace-all $nf   
set f [open A1-stop-n-wait.tr w]   
$ns trace-all $f

$ns duplex-link $n0 $n1 0.2Mb 200ms DropTail   
$ns duplex-link-op $n0 $n1 orient right   
$ns queue-limit $n0 $n1 10

Agent/TCP set nam\_tracevar\_ true

set tcp [new Agent/TCP]   
$tcp set window\_ 1   
$tcp set maxcwnd\_ 1   
$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]   
$ns attach-agent $n1 $sink

$ns connect $tcp $sink

set ftp [new Application/FTP]   
$ftp attach-agent $tcp

$ns add-agent-trace $tcp tcp   
$ns monitor-agent-trace $tcp   
$tcp tracevar cwnd\_

$ns at 0.1 "$ftp start"   
$ns at 3.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n1 $sink"   
$ns at 3.5 "finish"

$ns at 0.0 "$ns trace-annotate \"Stop and Wait with normal operation\""

$ns at 0.05 "$ns trace-annotate \"FTP starts at 0.1\""

$ns at 0.11 "$ns trace-annotate \"Send Packet\_0\""   
$ns at 0.35 "$ns trace-annotate \"Receive Ack\_0\""   
$ns at 0.56 "$ns trace-annotate \"Send Packet\_1\""   
$ns at 0.79 "$ns trace-annotate \"Receive Ack\_1\""   
$ns at 0.99 "$ns trace-annotate \"Send Packet\_2\""   
$ns at 1.23 "$ns trace-annotate \"Receive Ack\_2  \""   
$ns at 1.43 "$ns trace-annotate \"Send Packet\_3\""   
$ns at 1.67 "$ns trace-annotate \"Receive Ack\_3\""   
$ns at 1.88 "$ns trace-annotate \"Send Packet\_4\""   
$ns at 2.11 "$ns trace-annotate \"Receive Ack\_4\""   
$ns at 2.32 "$ns trace-annotate \"Send Packet\_5\""   
$ns at 2.55 "$ns trace-annotate \"Receive Ack\_5    \""   
$ns at 2.75 "$ns trace-annotate \"Send Packet\_6\""   
$ns at 2.99 "$ns trace-annotate \"Receive Ack\_6\""

$ns at 3.1 "$ns trace-annotate \"FTP stops\""

proc finish {} {   
 global ns nf   
 $ns flush-trace   
 close $nf

 puts "filtering..."   
 exec tclsh ../ns-allinone-2.1b5/nam-1.0a7/bin/namfilter.tcl A1-stop-n-wait.nam   
        puts "running nam..."   
        exec nam A1-stop-n-wait.nam &   
 exit 0   
}

$ns run 

**CONCLUSION:-**

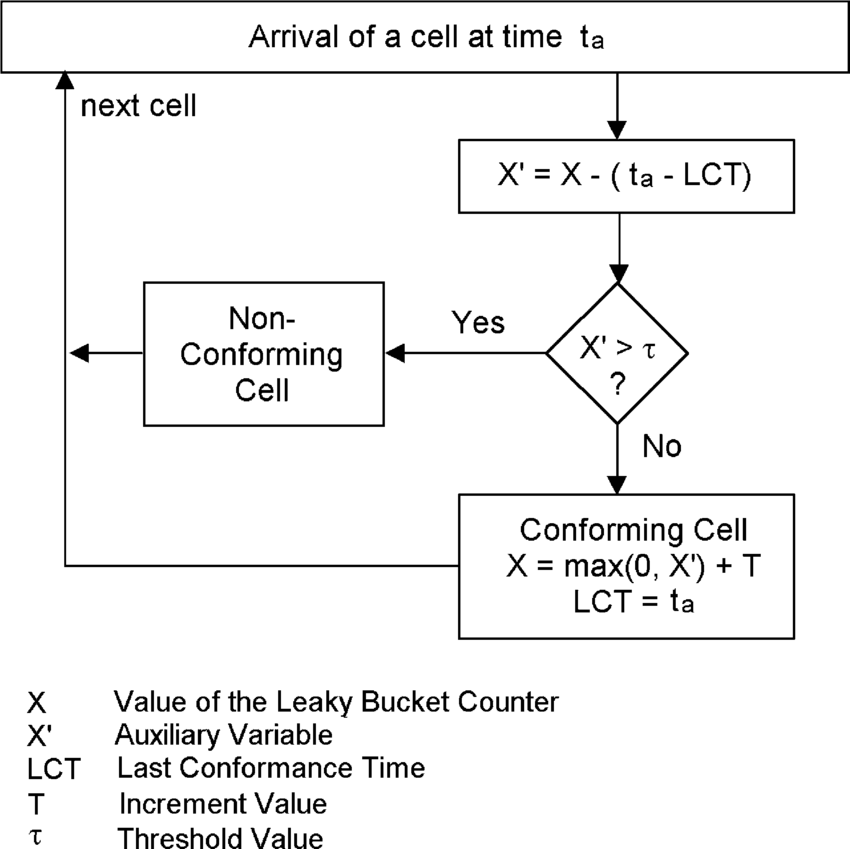
# PROGRAM NO. 8 : bucket algorithm for congestion control

**AIM:-** Implementation of leaky bucket algorithm for congestion control

**THEORY:-**

When a packet violates the contract, network can discard or tag the packet giving it• The process of monitoring and enforcing the traffic flow is called policing. •Network monitors traffic flows continuously to ensure they meet their traffic contract. lower priority Leaky Bucket Algorithm is the most commonly used policing mechanism• If congestion occurs, tagged packets are discarded first • o Bucket has specified leak rate for average contracted rate o Bucket has specified depth to accommodate variations in arrival rate o Arriving packet is conforming if it does not result in overflow Leaky Bucket algorithm can be used to police arrival rate of a packet stream.

The above figure shows the leaky bucket algorithm that can be used to police the At the arrival of the first packet, the content of the bucket is set to zero and the last•traffic flow. The depth of the bucket is L+I, where l depends on the traffic burstiness.•conforming time (LCT) is set to the arrival time of the first packet.



At the arrival of the kth packet, the auxiliary variable X’ records the difference between the bucket content at the arrival of the last conforming packet and the If the auxiliary variable is greater than L, the packet is considered as nonconforming,•interarrival time between the last conforming packet and the kth packet. otherwise the packet is conforming. The bucket content and the arrival time of the packet are then updated.

**CONCLUSION:-**

# PROGRAM NO. 9 : Perform Remote login using Telnet server

**AIM:-** Perform Remote login using Telnet server

**THEORY:-**

Telnet, developed in 1969, is a protocol that provides a command line interface for communication with a remote device or server, sometimes employed for remote management but also for initial device setup like network hardware. Telnet stands for Teletype Network, but it can also be used as a verb; 'to telnet' is to establish a connection using the telnet protocol.

Because it was developed before the mainstream adaptation of the internet, telnet does not employ any form of encryption, making it outdated in terms of modern security. It has largely been overlapped by Secure Shell (SSH) protocol, at least on the public internet.

#### How does Telnet work?

Telnet provides users with a bidirectional interactive text-oriented communication system utilizing a virtual terminal connection over 8 byte. User data is interspersed in-band with telnet control information over the transmission control protocol (TCP). Often, Telnet was used on a terminal to execute functions remotely.

The user connects to the server by using the Telnet protocol, which means entering Telnet into a command prompt by following this syntax: telnet hostname port. The user then executes commands on the server by using specific Telnet commands into the Telnet prompt. To end a session and log off, the user ends a Telnet command with Telnet.

#### What are common uses for Telnet?

Telnet can be used to test or troubleshoot remote web or mail servers, as well as for remote access to MUDs (multi-user dungeon games) and trusted internal networks.

**To use telnet, follow the steps below:**

1. First, find out the ip address of the server/main computer. For this you need to access the server and use the ipconfig command in MS-DOS. See Additional Information section for more details about this command.
2. Select the **Windows** key and the **R** key.
3. In the Run box type **CMD.**
4. Select **OK.**
5. Type **Telnet <IP Address> 13531.**

**Note:** Do not include the <> when entering the IP Address.

1. If you see a blank cursor then the connection is fine. You can close the command prompt window.
2. If you get the message that 'telnet' is not recognized as an internal or external command, operable program or batch file. you will want to enable Telnet. See Additional Information on how to Enable telnet.
3. If you get an error or are unable to telnet to the server please contact your Network Administrator.

**Example:**  
If the server's ip address is 192.168.0.100, then:

1. Open command prompt on the workstation computer
2. Type: telnet 192.168.0.100 13531

**CONCLUSION:-**

**PROGRAM NO. 10 : Configuration of FTP server**

**AIM:** Configuration of FTP server

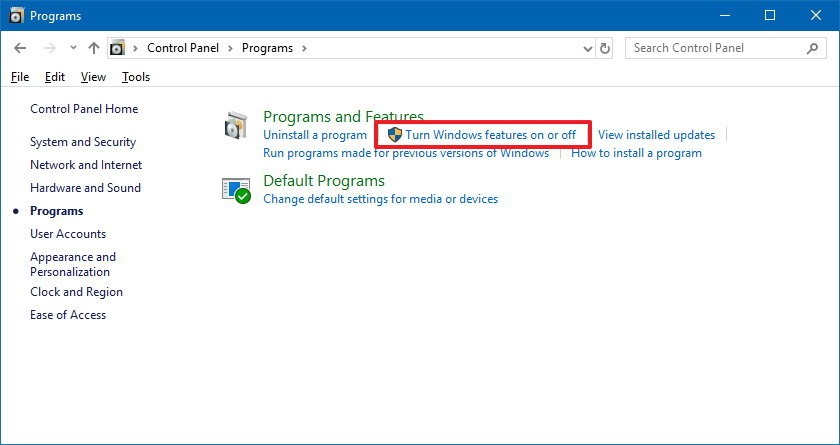
**THEORY:-**

## How to install the FTP server components on Windows 10

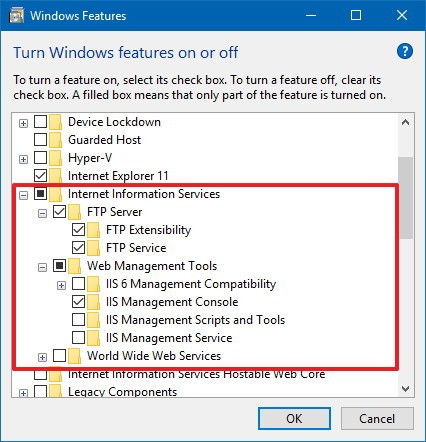
Although Windows 10 includes support to set up an FTP server, you need to add the required components manually.

To install the FTP server components, do the following:

1. Open **Control Panel**.
2. Click on **Programs**.
3. Under "Programs and Features," click the **Turn Windows features on or off** link.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/control-panel-programs-settings.jpg?itok=RA0fX9zH)

1. Expand the "Internet Information Services" feature, and expand the **FTP server**option.
2. Check the **FTP Extensibility** and **FTP Service** options.
3. Check the **Web Management Tools** option with the default selections, but making sure that the **IIS Management Console** option is checked.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/windows-features-ftp-components.jpg?itok=_Xo4kQvW)

1. Click the **OK** button.
2. Click the **Close** button.

Once you've completed the steps, the components to set up an FTP server will be installed on your device.

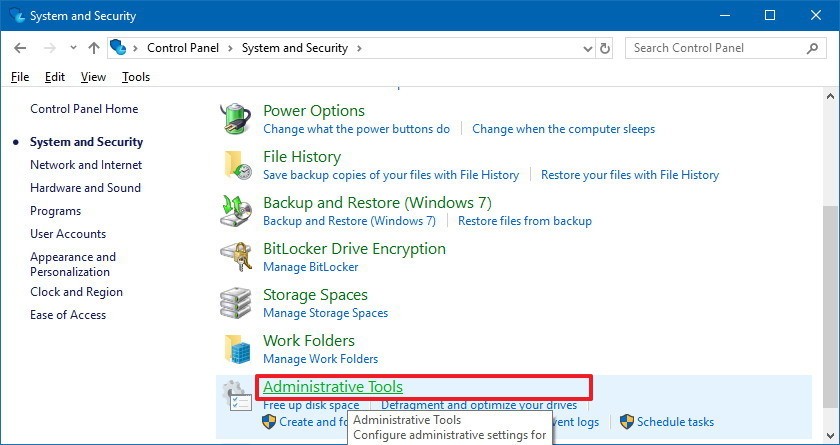
## How to configure an FTP server site on Windows 10

After installing the required components, you can proceed to configure an FTP server on the computer, which involves creating a new FTP site, setting up firewall rules, and allowing external connections.

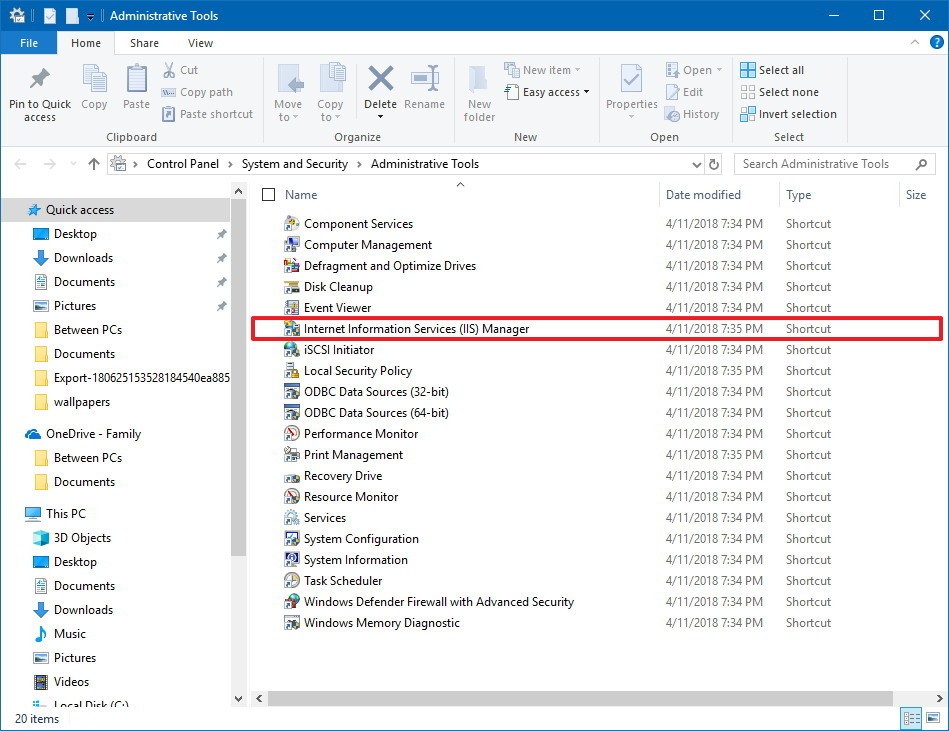
### Setting up an FTP site

To set up an FTP site, do the following:

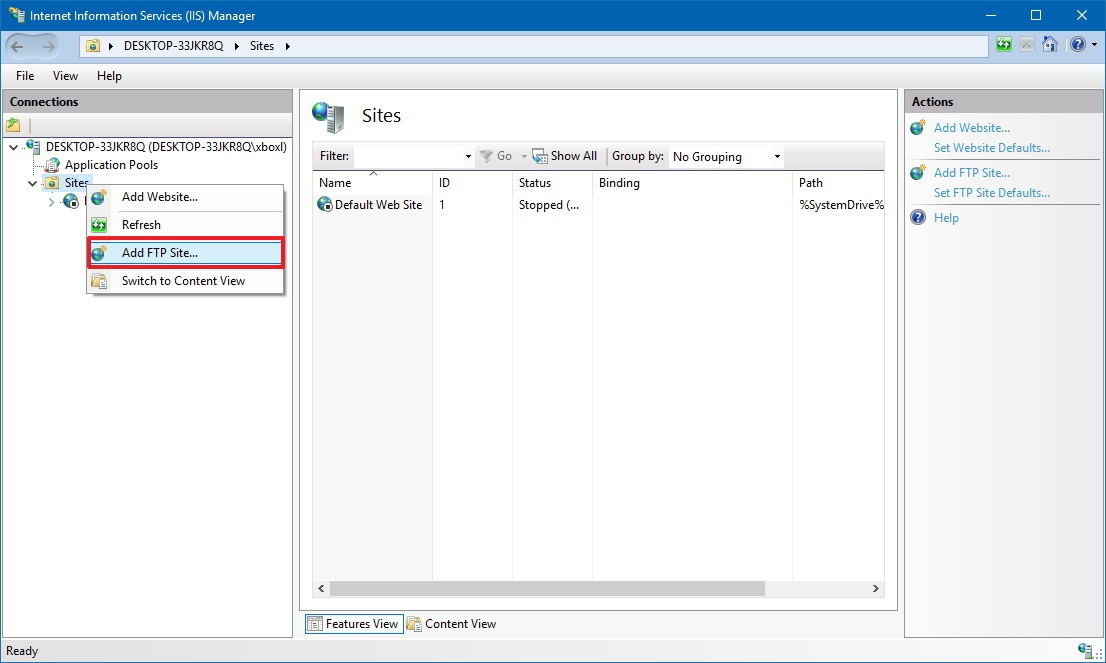
1. Open **Control Panel**.
2. Click on **System and Security**.
3. Click on **Administrative Tools**.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/control-panel-administrative-tools.jpg?itok=gZ5bSDe0)

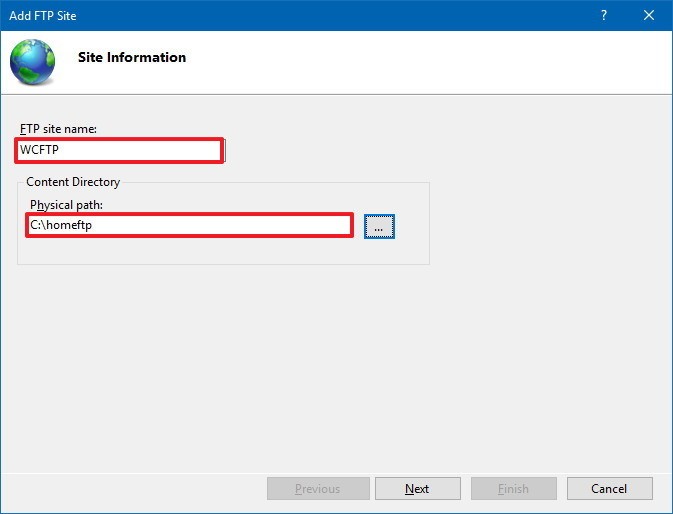
1. Double-click the **Internet Information Services (IIS) Manager** shortcut.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/control-panel-iis-manager.jpg?itok=HV7aEQ_N)

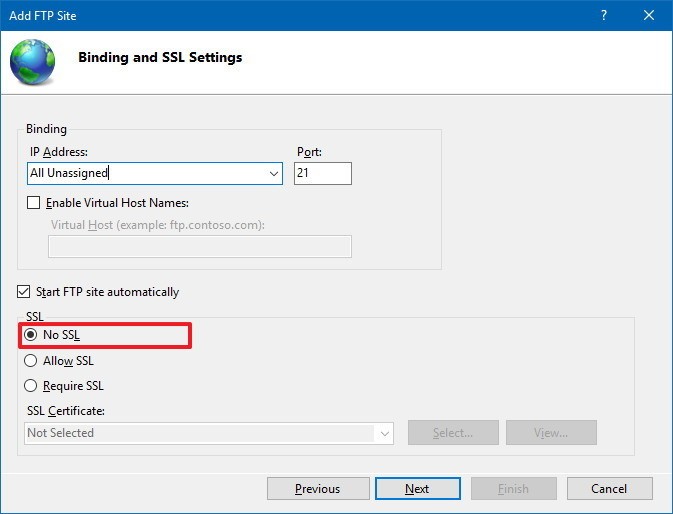
1. On the "Connections" pane, right-click **Sites**, and select the **Add FTP Site** option.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/iis-add-ftp-site-windows10.jpg?itok=TuPow9TF)

1. In the FTP site name, type a short descriptive name for the server.
2. In the "Content Directory" section, under "Physical path," click the button on the right to locate the folder you want to use to store your FTP files.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/ftp-name-directory-setup-windows-10.jpg?itok=bNnQr_FT)

1. Click the **Next** button.
2. Use the default **Binding** settings selections.
3. Check the **Start FTP site automatically** option.
4. In the "SSL" section, check the **No SSL** option.

[](https://www.windowscentral.com/sites/wpcentral.com/files/styles/xlarge/public/field/image/2018/06/ftp-ssl-setup-windows-10.jpg?itok=buhmdqPx)

**Important:** In a business environment or on an FTP server that will host sensitive data, it's best practice to configure the site to require SSL to prevent transmitting data in clear text.

1. Click the **Next** button.
2. In the "Authentication" section, check the **Basic** option.
3. In the "Authorization" section, use the drop-down menu, and select **Specified users** option.

**CONCLUSION :**

**PROGRAM NO. 10 : WIRESHARK**

**AIM:** Use of Wireshark to understand the operation of TCP/IP layers

**THEORY:-**

Wireshark can be downloaded at no cost from the Wireshark Foundation website for both macOS and Windows operating systems. Unless you are an advanced user, it is recommended that you only download the latest stable release. During the Windows setup process, you should choose to install WinPcap if prompted, as it includes a library required for live data capture.

The application is also available for Linux and most other UNIX-like platforms including Red Hat, Solaris, and FreeBSD. The binaries required for these operating systems can be found toward the bottom of the download page in the Third-Party Packages section. You can also download Wireshark's source code from this page.

### How to Capture Data Packets

When you first launch Wireshark, a welcome screen appears containing a list of available network connections on your current device. In this example, you'll notice that the following connection types are shown: Bluetooth Network Connection, Ethernet, VirtualBox Host-Only Network, and Wi-Fi. Displayed to the right of each is an EKG-style line graph that represents live traffic on that respective network.

To begin capturing packets, select one or more of the networks by clicking on your choice and using the **Shift** or **Ctrl** keys if you want to record data from multiple networks simultaneously. After a connection type is selected for capturing purposes, its background is shaded in either blue or gray. Click on **Capture** in the main menu located toward the top of the Wireshark interface. When the drop-down menu appears, select the **Start**option.

You can also initiate packet capturing via one of the following shortcuts.

* **Keyboard:** Press ​**Ctrl**+**E.**
* **Mouse:** To begin capturing packets from one particular network, double-click on its name.
* **Toolbar:** Click on the blue shark fin button located on the far left side of the Wireshark toolbar.

The live capture process begins, and Wireshark displays the packet details as they are recorded. To Stop capturing:

* **Keyboard:** Press **Ctrl**+**E**
* **Toolbar:** Click on the red **Stop** button located next to the shark fin on the Wireshark toolbar.

### Viewing and Analyzing Packet Contents

After you record some network data, it's time to take a look at the captured packets. The captured data interface contains three main sections: the packet list pane, the packet details pane, and the packet bytes pane.

### Packet List

The packet list pane, located at the top of the window, shows all packets found in the active capture file. Each packet has its own row and corresponding number assigned to it, along with each of these data points.

* **Time:** The timestamp of when the packet was captured is displayed in this column. The default format is the number of seconds or partial seconds since this specific capture file was first created. To modify this format to something that may be a bit more useful, such as the actual time of day, select the **Time Display Format**option from Wireshark's View menu located at the top of the main interface.
* **Source:** This column contains the address (IP or other) where the packet originated.
* **Destination:** This column contains the address that the packet is being sent to.
* **Protocol:** The packet's protocol name, such as TCP, can be found in this column.
* **Length:** The packet length, in bytes, is displayed in this column.
* **Info:** Additional details about the packet are presented here. The contents of this column can vary greatly depending on packet contents.

When a packet is selected in the top pane, you may notice one or more symbols appear in the first column. Open or closed brackets and a straight horizontal line indicate whether a packet or group of packets are all part of the same back-and-forth conversation on the network. A broken horizontal line signifies that a packet is not part of said conversation.

### Packet Details

The details pane, found in the middle, presents the protocols and protocol fields of the selected packet in a collapsible format. In addition to expanding each selection, you can apply individual Wireshark filters based on specific details and follow streams of data based on protocol type via the details context menu, which is accessible by right-clicking your mouse on the desired item in this pane.

### Packet Bytes

At the bottom is the packet bytes pane, which displays the raw data of the selected packet in a hexadecimal view. This [hex dump](https://www.lifewire.com/xxd-linux-command-unix-command-4097149) contains 16 hexadecimal bytes and 16 ASCII bytes alongside the data offset.

Selecting a specific portion of this data automatically highlights its corresponding section in the packet details pane and vice versa. Any bytes that cannot be printed are instead represented by a period.

You can choose to show this data in bit format as opposed to hexadecimal by right-clicking anywhere within the pane and selecting the appropriate option from the context menu.

### Using Wireshark Filters

One of the most important feature sets in Wireshark is its filter capability, especially when you're dealing with files that are significant in size. Capture filters can be set before the fact, instructing Wireshark to only record those packets that meet your specified criteria.

Filters can also be applied to a capture file that has already been created so that only certain packets are shown. These are referred to as display filters.

Wireshark provides a large number of predefined filters by default, letting you narrow down the number of visible packets with just a few keystrokes or mouse clicks. To use one of these existing filters, place its name in the **Apply a display filter**entry field located directly below the Wireshark toolbar or in the **Enter a capture filter** entry field located in the center of the welcome screen.

There are multiple ways to achieve this. If you already know the name of your filter, type it into the appropriate field. For example, if you only want to display TCP packets, you type **tcp**. Wireshark's autocompleting feature shows suggested names as you begin typing, making it easier to find the correct moniker for the filter you're seeking.

Another way to choose a filter is to click on the bookmark-like icon positioned on the left side of the entry field. This presents a menu containing some of the most commonly used filters as well as an option to **Manage Capture Filters** or **Manage Display Filters**. If you choose to manage either type, an interface appears allowing you to add, remove, or edit filters.

You can also access previously used filters by selecting the down arrow on the right side of the entry field to display a history drop-down list.

Once set, capture filters are applied as soon as you begin recording network traffic. To apply a display filter, you click on the right arrow button found on the far right side of the entry field.

**Color Rules**

While Wireshark's capture and display filters allow you to limit which packets are recorded or shown on the screen, its colorization functionality takes things a step further by making it easy to distinguish between different packet types based on their individual hue. This handy feature lets you quickly locate certain packets within a saved set by their row color in the packet list pane.

Wireshark comes with about 20 default coloring rules built in, each of which can be edited, disabled, or deleted if you wish. You can also add new shade-based filters through the coloring-rules interface, accessible from the **View** menu. In addition to defining a name and filter criteria for each rule, you are also asked to associate both a background color and a text color.

Packet colorization can be toggled off and on via the **Colorize Packet List** option, also found in the **View** menu.

**Statistics**

In addition to the detailed information about your network's data shown in Wireshark's main window, several other useful metrics are available via the **Statistics**drop-down menu found toward the top of the screen. These include size and timing information about the capture file itself, along with dozens of charts and graphs ranging in topic from packet conversation breakdowns to load distribution of HTTP requests.

Display filters can be applied to many of these statistics via their interfaces, and the results can be exported to several common file formats including CSV, XML, and TXT.

### Advanced Features

In addition to Wireshark's main functionality, there is also a collection of additional features available in this powerful tool typically reserved for advanced users. This includes the ability to write your own protocol dissectors in the Lua programming language.

**CONCLUSION :**