MADANAPALLEINSTITUTEOF TECHNOLOGY &SCIENCE
(UGC-AUTONOMOUS INSTITUTION)
Affiliated to JNTUA, Ananthapuramu &Approved by AICTE, New Delhi
NAAC Accredited with A+ Grade,
NBA Accredited -B.Tech. (CIVIL, CSE, CST, ECE, EEE, MECH), MBA & MCA

DEARTMENT OF COMPUTER SCIENCE & ENGINEERING-DATA SCIENCE

Assignment Submission Details

AY-2024- 25

Subject: COMPUTER NETWORKS

Subject Code: 20CSD115

Name of the Student	D.Dinesh Kumar Reddy
Roll. No.	22691A3215
Year /Sec	III – CSE (DS) – A Sec
Assignment No.	I // II
Marks (Max 5 Marks)	
Assignment Moodle uploaded Date	Uploaded // Drafted - _____
Faculty Sign with Name & Date	Mrs. S. Manjula Prabakaran

Q1 : Illustrate open-loop congestion control with examples

Q2: Examine how the File Transfer Protocol (FTP) works and its key features.

Q3:

Q4:

Assignment – 2

1. Illustrate open-loop congestion control with examples.

A.

Open-Loop Congestion Control

Open-loop congestion control is a method in network traffic management where the sender regulates the data transmission rate to prevent congestion without getting any feedback or knowledge about the network condition. This is being done ahead of time using a defined set of rules or assumptions, with no response to the real network situations or congestion signals like packet loss, delay, or explicit receiver's feedback.

Major Features of Open-Loop Congestion Control:

No Feedback Mechanism: The sender does not depend on network feedback to regulate the rate of data transmission. It anticipates network conditions beforehand.

Preemptive Traffic Control: It attempts to prevent congestion beforehand by restricting the data rate.

Static Control: The sending rate is fixed or adjusted periodically irrespective of the dynamic state of the network.

Examples of Open-Loop Congestion Control:

1. Traffic Shaping

Traffic shaping is an open-loop control in which the sender controls the flow of data to equal the target traffic rate, with the goal of preventing congestion within the network.

Example: Within an enterprise network of a company, a server can restrict the amount at which large files are transmitted so that the remaining network resources are not saturated with excessive data. The server is given a fixed rate for sending files irrespective of the network's condition at the time.

How It Works: The server can be set up to send not more than 10 Mbps of traffic at any moment. Even when the network is not saturated, this constant rate of transmission is sustained to ensure that data does not swamp the network during unforeseen congestion.

2. Token Bucket Algorithm

Token Bucket algorithm is a common method applied in traffic shaping. It manages the transmission rate by capping the amount of data to be sent at a particular instant.

Example: In an Internet Service Provider (ISP) context, the token bucket algorithm can be applied to cap the amount of data a customer sends into the network at any instant. The client is provided tokens at a fixed rate, and data can only be transmitted when tokens are present within the bucket. When the client does not consume all the tokens, they are stored, but if the data transmission rate is more than the tokens present, the data is queued.

How It Works: The algorithm controls the rate of data transmission through tokens that denote "permission" to transmit data. This does not allow excessive data to pour into the network and thus prevent congestion.

3. Leaky Bucket Algorithm

The Leaky Bucket algorithm is another traffic shaping method, in which incoming data is placed into a "bucket," and the data is transmitted out at a fixed rate. If the rate of incoming data is greater than the rate of outgoing data, the surplus data is dropped, smoothing traffic and preventing congestion.

Example: An organization can employ the leaky bucket method to control the rate of flow of data from employees to the internet. When lots of employees start sending large files at the same time, the leaky bucket manages to keep the output data rate steady, and the surplus traffic is dropped.

How It Works: The system takes incoming data into the bucket, and at a steady rate, data is transmitted out. In case of excessive incoming traffic (grater than the bucket capacity), some packets will get dropped to prevent the network from being overwhelmed.

4. Fixed Rate Transmission

In certain situations, a sender predefines a constant data rate at which it sends data, irrespective of network conditions. This approach avoids congestion through placing a limit on the rate at which data is sent, independent of whether the network is congested or not.

Example: A video streaming app may be set to stream at a fixed bit rate (say, 2

Mbps), independent of available network bandwidth. This fixed rate will ensure not to clog the network, though the sender does not possess any real-time network congestion information.

How It Works: The program sends data at a fixed rate (2 Mbps here), so that the network does not get burdened. Yet, if the network was able to support higher traffic, then this approach will not make full use of the bandwidth.

5. Slow Start in TCP (Early Mechanism)

In certain initial implementations of TCP, open-loop methods such as "slow start" were used to slowly raise the sending rate without dynamic adaptation against realtime congestion feedback. Even though slow start developed into a feedbackoriented mechanism subsequently, it operated somewhat as an

open-loop in the past by beginning transmission at a low level and raising it based on a preprogrammed algorithm.

Example: TCP could start with sending a single segment of data and gradually enlarge the window to send more data. It would not respond quickly to congestion but would instead follow a preconfigured rule to slowly increase the transmission rate.

How It Works: First, a small number of data is transmitted to the receiver. The sender, with each received acknowledgment packet, boosts its sending rate based on a predetermined algorithm, without taking real-time network conditions into account.

Advantages of Open-Loop Congestion Control:

Simplicity: Open-loop methods are simpler to implement since they don't ask the sender to observe the state of the network at all times or get feedback.

Predictability: By precontrolling the rate of data, the sender negates the unpredictability of congestion.

Stability: As transmission rate is being controlled beforehand, there are fewer chances of explosive bursts that consume the network.

Drawbacks of Open-Loop Congestion Control:

Inefficiency: The system can fail to capitalize on available bandwidth if the sender is overly conservative in its scheme.

No Adaptability: When network conditions vary (e.g., more congestion or available capacity), open-loop control does not vary with the changes.

Potential Waste: Where the network is not fully utilized, the rate set through a fixed rate can result in inefficient use of available resources.

2. Examine how the File Transfer Protocol (FTP) works and its key features.

A.

File Transfer Protocol (FTP) Overview

File Transfer Protocol (FTP) is a network protocol that is used to transfer files from one computer to another over a TCP/IP network, for example, the internet or a local area network (LAN). FTP is based on a client-server model, where the client makes requests to transfer files and the server replies by sending the requested files or receiving files for storage. FTP was invented in the early 1970s and is among the oldest protocols used on the internet today.

How FTP Works

FTP uses a client-server model, where:

The FTP Client is the program or application that issues requests for files or uploads files to a server.

The FTP Server is the computer that stores files and replies to client requests.

Key Steps in FTP Operation:

Connection Establishment:

FTP requires a control connection and a data connection.

The client connects to the FTP server on port 21 (the default FTP port) for control communication.

Authentication:

After establishing the control connection, the client sends a username and password (authentication credentials) to the FTP server for logging in. A few FTP servers permit anonymous access without authentication, often applied to public file sharing.

Command Exchange:

Once logged in, the client and server send commands across the control connection (port 21). The commands are requests to list directories, retrieve files, upload files, etc.

FTP commands examples:

USER – Sends the user name to the server.

PASS – Sends the password for verification.

LIST – Requests a list of the files in the present directory.

RETR – Requests the retrieval (download) of a file.

STOR – Transfers a file to the server (upload).

Data Transfer:

After a file transfer command is sent (e.g., RETR or STOR), a data connection is created between the client and server.

FTP employs two data transfer modes: Active Mode and Passive Mode.

Active Mode:

The client establishes a random port and notifies the server about this port.

The server establishes a connection with the client on the given port to transfer data.

Passive Mode:

The client establishes both the control and data connection.

The server allocates a random port and notifies the client of this port.

The client establishes a connection to the specified port of the server for data transfer.

File Transfer:

Data is sent across the data connection. This is done in binary mode (for non-text files) or ASCII mode (for text files), depending on the file type being transferred.

Termination:

Upon completion of the file transfer, the data connection is terminated.

The control connection stays open until the user ends the session (using the QUIT command) or times out.

Important Features of FTP Two Connection
Channels:

Control Connection: FTP maintains an ongoing connection (usually on port 21) for the sending of commands and receiving replies.

Data Connection: FTP has a distinct connection for file transfers, which may either be active or passive mode based on network settings.

User Authentication:

FTP normally needs to be accessed via a username and password, but anonymous FTP exists for users who wish to have access to publicly accessible files without the need for an account.

File Operations:

FTP supports different operations on files and directories as follows:

Download (RETR): Download files from the server.

Upload (STOR): Put files on the server.

List (LIST): List directories' contents.

Rename (RNFR, RNTO): Change file or directory names.

Delete (DELE): Remove files from the server.

Change Directory (CWD): Alter the present working directory on the server.

Binary and ASCII Modes:

Binary Mode: Employed for non-text files such as images, videos, or executables. It guarantees the file is transmitted without alteration.

ASCII Mode: Employed for text files, enabling automatic conversion of newline characters between operating systems (e.g., from Windows-style line endings to Unix-style).

Support for Large Files:

FTP facilitates the transfer of large files by dividing the transfer into smaller chunks over the data connection.

Support for Directory Navigation:

FTP enables users to move around directories on the server and manipulate file structures (e.g., creating, removing directories).

Security:

Regular FTP is insecure since data, such as usernames, passwords, and file contents, is transmitted in plain text.

Secure variants of FTP are:

FTPS (FTP Secure): Adds SSL/TLS encryption to FTP, ensuring both the control and data channels are secure.

SFTP (SSH File Transfer Protocol): employs SSH for secure file transfer using a single encrypted connection, offering confidentiality and integrity.

File Transfer Resumption:

A few FTP servers facilitate resuming halted file transfers, by enabling the client to ask the server to continue the transfer from the point where it was halted.

FTP Modes: Active and Passive Active Mode:

The client establishes a random port and instructs the server to use it to connect to transfer the data. The server makes a connection back to the client.

Active mode creates issues when the client is behind a NAT (Network Address Translation) or a firewall since the server will not be able to connect directly to the client.

Passive Mode:

The client opens both the control and data connection. The server opens a random port for data transfer and notifies the client. The client then connects to the server's data port.

Passive mode is generally employed when the client is behind NAT or a firewall since it prevents the server from having to open connections to the client.

Key Advantages of FTP

Simple and Widespread Support: FTP is simple to use and has been extensively used in both personal and commercial settings.

File Integrity: FTP accommodates various modes of transfer (binary and ASCII) to preserve the integrity of the file during transfer.

Automation: FTP allows scripts and batch processing, enabling automatic file transfers.

Key Disadvantages of FTP

Security: Vanilla FTP transmits data, even confidential information, in plain text. This exposed mode makes it prone to man-in-the-middle and eavesdropping attacks.

Firewall Issues: Behind NAT devices and firewalls, FTP may suffer issues, particularly active mode.

No Built-in Encryption: As a matter of default, FTP does not come with inherent encryption, thereby it is unfit for secure transmission unless upgraded modes like FTPS or SFTP are employed.