**19CSE301 COMPUTER NETWORKS**

**CASE STUDY**

**COLLEGE NETWORK**

**GROUP DETAILS:**

|  |  |
| --- | --- |
| **Roll Number** | **Name of the Student** |
| CB.EN.U4CSE19305 | K ASHWANTH |
| CB.EN.U4CSE19306 | B VIGNESH |
| CB.EN.U4CSE19317 | DT SUVANESH |
| CB.EN.U4CSE19318 | DHARUN NARAYANAN L K |
| CB.EN.U4CSE19327 | K R RITHICK |

**Index:**

|  |  |
| --- | --- |
| 1. Description | 2 |
| 2. Why is Network needed? | 2 |
| 3. Type of Network | 2 |
| 4. Client Configuration | 2 |
| 5. Server Configuration | 3 |
| 6. Departments | 4 |
| 7. Architecture | 5 |
| 8. Network performance parameters | 6 |
| 9. File Handling Operations using Socket Programming   1. Description of programs 2. Server Program 3. Client Program 4. Outputs | 7  7 - 8  8  9 - 10 |

**Description:**

In this application, we are studying the structure of a College Network. This college network provides services to students as well as faculties in various blocks also, labs from Academic block1, Academic block2, Academic block3 connected through this network. As a result, students and faculty can seamlessly access the network and this is important for staff to conduct lab exams as well as publish marks in a more efficient manner by the exam department.

This aids the faculties in time management.The following thesis sheds light on some main parts which are improved during the practical work:

* Maintaining student information
* Supporting the campus management system
* Wi-Fi infrastructure for classes
* Security

**Why Networking is required For Our Application:**

The importance of effective communication within different blocks of a college can be overstated with the speed, efficiency, and integrity of the information being transferred receiving top emphasis. To enable this, a highly efficient networking solution becomes essential to the operation of such a large infrastructural system, where even minor errors can result in server crashes.

**Type of network:**

In College Network, we will be using Local Area Network (LAN). In this use case, it is a well-suited network type considering the scale of network required for this. A local area network (LAN) is a collection of devices connected in one physical location, such as a building, office, or home. A LAN can be small or large, ranging from a home network with one user to an enterprise network with thousands of users and devices in an office or school.

**Client Configuration:**

The clients in this network will be faculty or student with their username, password, and a compatible web browser. We can support an array of devices such as

* Computers in the Lab (Linux/Windows)
* Smartphone (Android/iOS/Harmony)
* Tablet/iPad
* Personal Laptop (Linux/Windows/Mac)

**Server Configuration:**

The server, forming the backbone of the entire college network system, will require the highest specs for efficient transmission, storage, and processing of college information. Hence, the following system configuration is recommended for, the server-side system:

* **Operating System:** Windows 10 and up
* **Processor:** Intel XEON Gold 6330 Processor and above
* **RAM:** 128GB
* **Memory:** 8TB SSD + Mass storage on a dedicated NAS server
* **Network Support:** Support for high speed 10 Gigabit Ethernet

**Servers Used in College Network:**

|  |  |
| --- | --- |
| **Server** | **Ip Address** |
| 1. INTRANET - AUMS Server | 1.0.0.2 |
| 1. WEB - SQL Server | 1.0.0.3 |
| 1. MAIL - FTP Server | 1.0.0.4 |
| 1. DNS - IoT Server | 1.0.0.5 |

**Network cable:**

* **Copper straight-through** cable is a **type of CAT5 with RJ-45** connectors at each end, and each has the same pinout. A Crossover cable is a type of CAT where one end is T568A configuration and the other end is T568B Configuration. It is one of the most used cable formats for network cables.
* **Serial DCE - DCE stands for Data Communications Equipment**. Therefore, a straight-through "one to one" cable is all that is necessary between a modem and a standard DTE serial port.

**Type of Topology**:

**Star Topology** is in which each network component is physically connected to a central node such as a router, hub, or switch.

**Different entities Within the Application:**

**1)Exam department:**

This department schedules and conducts exams for all students. They can communicate information about various upcoming exams to students and faculties

**2) Faculties:**

Faculties will be using the network to upload marks and attendance of students. They use the network to store and retrieve files from the FTP server. They also use the network to schedule quizzes and give question papers to students during lab exams.

**3) Students:**

The students use the network to view marks, grades, and attendance. They take up quizzes through the network and also use the network to download course resources and pay fees.

**4) Maintenance department:**

This department maintains the entire network, fixes occasionally occurring bugs and manages data by transferring the data to cold storage and maintaining size of caches. They do modifications to server-side resource files if necessary and take care of overall network security.

**Protocols used:**

**Application layer:**

* HTTP
* SMTP
* FTP
* DNS

**Transport layer:**

* TCP

**Network layer:**

* Ipv4 (Class A, B, C)

**Link layer:**

* Ethernet
* Wi-Fi

**Networks Concept Used:**

**FLSM:** is used to allocate IP addresses to different sub-networks in the company network.

**Static Routing:**

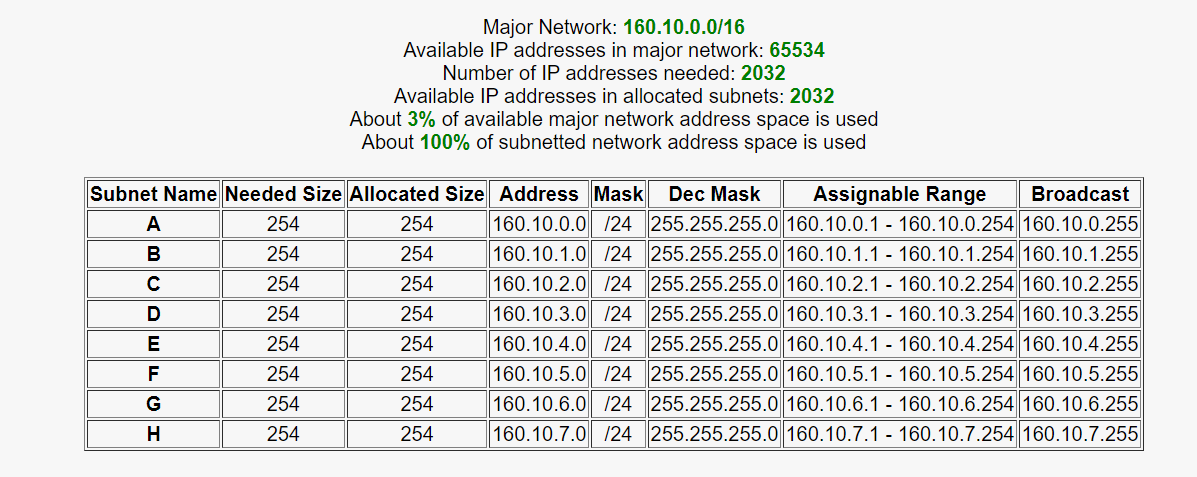
**Requirements:**

* 508 (254\*2) devices in Ab1 Lab.
* 508 (254\*2) devices in Ab2 Lab.
* 508 (254\*2) devices in Ab3 Lab.
* 508 (254\*2) devices in IT Lab.

Main Network IP address: 160.10.0.0/16

Server IP Address: 1.0.0.0/8

**Subnetting:**



**AB1 Lab –** Subnet A and B

**AB2 Lab –** Subnet C and D

**AB3 Lab –** Subnet E and F

**IT Lab –** Subnet G and H

**Architecture/Network Diagram:**

Diagram

Description automatically generatedDiagram

Description automatically generated

Diagram

Description automatically generatedDiagram

Description automatically generated

Chart

Description automatically generated with medium confidence

**Network Performance parameters with their purpose and formula:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Meaning** | | **Formula** |
| Bandwidth | | Bandwidth is the capacity of a wired or wireless network communications link to transmit the maximum amount of data from one point to another over a computer network or internet connection in each amount of time. | Expressed as [bits](https://web.archive.org/web/20190816003233/https:/whatis.techtarget.com/definition/bit-binary-digit) per second ([bps](https://web.archive.org/web/20190816003233/https:/searchnetworking.techtarget.com/definition/bits-per-second)), modern network links have greater capacity, which is typically measured in millions of bits per second ([megabits per second](https://web.archive.org/web/20190816003233/https:/searchnetworking.techtarget.com/definition/Mbps), or Mbps) or billions of bits per second ([gigabits per second](https://web.archive.org/web/20190816003233/https:/whatis.techtarget.com/definition/Gbps-billions-of-bits-per-second), or Gbps). |
| Throughput | | Throughput measures the percentage of data packets that are successfully being sent; a low throughput means there are a lot of failed or dropped packets that need to be sent again. |  |
| Packet Loss | | Packet loss occurs when one or more packets of data travelling across a computer network fail to reach their destination. Due to network congestion | Efficiency = 100% \* (transferred - retransmitted) / transferred  Network Loss = 100 - Efficiency |
| Transmission time | | The time required for the transmission of a message depends on the size of the message and the bandwidth of the channel. | Transmission time=Message size / Bandwidth |
| Propagation Time | | Propagation time measures the time required for a bit to travel from the source to the destination. The propagation time is calculated by dividing the distance by the propagation speed. | Propagation time = Distance /Propagation speed |
| Processing Delay | | Time taken by the processor to process the data packet is called processing delay. | Directly proportional to the processing speed of the routers. |
| Queuing Delay | | Time spent by the data packet waiting in the queue before it is taken for execution is called queuing delay. | Directly Proportional to the congestion in the network |

**Socket Programming**

The server has the following operations:

* View CGPA**(C)**
* View Attendance**(A)**
* View Fee payment status **(F)**
* Close Connection **(E)**

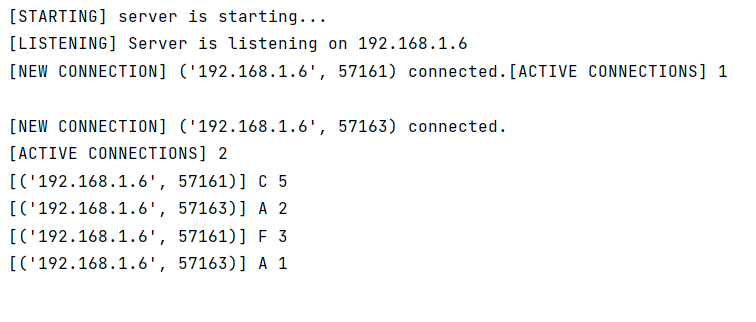
**Server code: (server.py)**

import socket  
import threading  
import pandas as pd  
  
HEADER = 64  
PORT = 5052  
SERVER = socket.gethostbyname(socket.gethostname())  
ADDR = (SERVER, PORT)  
FORMAT = 'utf-8'  
DISCONNECT\_MESSAGE = "!DISCONNECT"  
  
server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
server.bind(ADDR)  
df = pd.read\_csv('student.csv')  
  
  
def handle\_client(conn, addr):  
 print(f"[NEW CONNECTION] {addr} connected.")  
  
 def cgpa(df, id):  
 st = " "  
 for i in range(len(df)):  
 if df.loc[i, 'id'] == id:  
 st = 'Roll no: ' + str(df.loc[i, 'id']) + '| Name: ' + str(df.loc[i, 'name']) + '| CGPA: ' + str(  
 df.loc[i, 'marks'])  
 break  
 return st  
  
 def fees(df, id):  
  
 for i in range(len(df)):  
 if df.loc[i, 'id'] == id and df.loc[i, 'fees\_paid'] == True:  
 return 'Fees Paid'  
 return 'Fees due'  
  
 def attendance(df, id):  
 st = " "  
 for i in range(len(df)):  
 if df.loc[i, 'id'] == id:  
 st = ' Roll no: ' + str(df.loc[i, 'id']) + '| Name: ' + str(df.loc[i, 'name']) + '| Attendance: ' + str(  
 df.loc[i, 'attendance'])  
 break  
 return st  
  
 connected = True  
 while connected:  
 msg\_length = conn.recv(HEADER).decode(FORMAT)  
 if msg\_length:  
 msg\_length = int(msg\_length)  
 msg = conn.recv(msg\_length).decode(FORMAT)  
  
 if msg == DISCONNECT\_MESSAGE:  
 connected = False  
 elif msg[0] == 'C':  
 split\_data = msg.split()  
 pts = cgpa(df, int(split\_data[1]))  
 print(f"[{addr}] {msg}")  
 conn.send(pts.encode(FORMAT))  
  
 elif msg[0] == 'A':  
 split\_data = msg.split()  
 attnd = attendance(df, int(split\_data[1]))  
 print(f"[{addr}] {msg}")  
 conn.send(attnd.encode(FORMAT))  
 elif msg[0] == 'F':  
 split\_data = msg.split()  
 fee = fees(df, int(split\_data[1]))  
 print(f"[{addr}] {msg}")  
 conn.send(fee.encode(FORMAT))  
 else:  
 print(f"[{addr}] {msg}")  
 conn.send('Command incorrect'.encode(FORMAT))  
  
 conn.close()  
  
  
def start():  
 server.listen()  
 print(f"[LISTENING] Server is listening on {SERVER}")  
 while True:  
 conn, addr = server.accept()  
 thread = threading.Thread(target=handle\_client, args=(conn, addr))  
 thread.start()  
 print(f"[ACTIVE CONNECTIONS] {threading.activeCount() - 1}")  
  
  
print("[STARTING] server is starting...")  
start()

**client code: (client.py)**

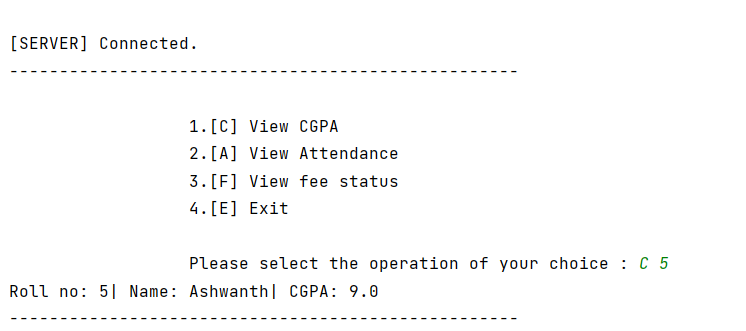
import socket  
  
HEADER = 64  
PORT = 5052  
FORMAT = 'utf-8'  
DISCONNECT\_MESSAGE = "!DISCONNECT"  
SERVER = "192.168.1.6"  
ADDR = (SERVER, PORT)  
  
client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
client.connect(ADDR)  
  
def send(msg):  
 message = msg.encode(FORMAT)  
 msg\_length = len(message)  
 send\_length = str(msg\_length).encode(FORMAT)  
 send\_length += b' ' \* (HEADER - len(send\_length))  
 client.send(send\_length)  
 client.send(message)  
 print(client.recv(2048).decode(FORMAT))  
  
def main():  
 print("\n[SERVER] Connected.")  
 while 1:  
  
 inp = input("""---------------------------------------------------  
  
 1.[C] View CGPA  
 2.[A] View Attendance  
 3.[F] View fee status  
 4.[E] Exit  
  
 Please select the operation of your choice : """)  
 if inp == 'E':  
 send(DISCONNECT\_MESSAGE)  
 break  
 else:  
 send(inp)  
  
 print("\n[SERVER] Disconnected.")  
  
  
main()  
  
  
**Outputs:**

**Server side:**

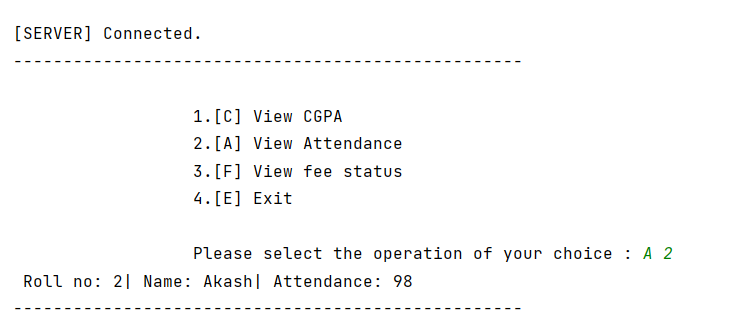


**Outputs on client side:**

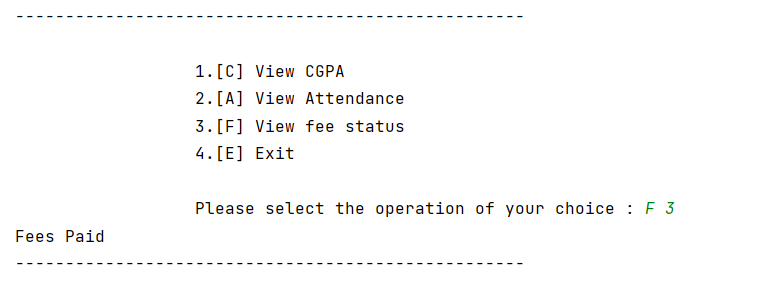
* We run client and give input as C 5
* Displays name, CGPA details of roll no 5.



* Displays name, attendance details of roll no 2.



* Shows whether roll no 2, has paid fees or not



* E to terminate/exit connection

