Project Report

On

# IMPLEMENT IP SUBNETTING

Submitted in partial fulfilment of the requirements for the award of

## BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

(Artificial Intelligence & Machine Learning)

by

**Ms. V. KEAVALYA(22WH1A6618)**

**Ms. K. LAXMI PRASANNA (22WH1A6620)**

**Ms. E. LAHARI(22WH1A6637)**

**Ms. A. GAYATHRI (22WH1A6656)**

**Under the esteemed guidance of**

**Ms. P Anusha**

**Assistant Professor, CSE(AI&ML)**



## BVRIT HYDERABAD College of Engineering for Women

**(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)**

**(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)**

**Bachupally, Hyderabad – 500090**

2024-25

## ABSTRACT

This program implements a foundational IP Subnetting Calculator designed to assist in dividing a network into smaller, manageable subnetworks. Its primary goal is to provide a simple yet effective method of calculating key subnetting parameters such as the network address, broadcast address, and the range of usable IP addresses.

The program is built around a straightforward command-line interface (CLI) that allows users to input an IP address and subnet mask. By leveraging efficient binary arithmetic, the system ensures accurate calculations and provides essential information for network configuration and management..

1. Initialization

* The system begins by accepting user input for an IP address and subnet mask in a standard dotted-decimal format. These inputs are then parsed and converted into integer arrays, enabling further bitwise computations.
* This initialization phase eliminates the need for external libraries or databases, ensuring a lightweight and portable implementation suitable for local network setups.

1. Address calculation

* The network address is determined using a logical AND operation between the IP address and the subnet mask, which identifies the starting point of the subnet.
* The broadcast address is computed using a logical OR operation with the complement of the subnet mask, marking the endpoint of the subnet.
* The program also includes error handling to validate that the input is within acceptable ranges, avoiding calculation errors due to invalid inputs.

1. Usable IP range

* After determining the network and broadcast addresses, the system calculates the range of usable IP addresses by incrementing the network address and decrementing the broadcast address.
  + This range provides all valid host addresses within the subnet, excluding the network and broadcast addresses themselves. This information is critical for configuring devices and assigning IPs in a network.

**PROBLEM STATEMENT**

This project involves the creation of a comprehensive C program that functions as an **IP Subnetting Calculator**, designed to aid users in understanding and solving subnetting problems in computer networks. The program facilitates user interaction by prompting them to input an IP address and a subnet mask in the standard dotted-decimal notation. It validates the correctness of these inputs to ensure they adhere to the formatting and range constraints of IP addressing.

Upon receiving valid inputs, the program performs the necessary binary arithmetic operations to compute essential subnetting parameters, including the **network address**, **broadcast address**, and the **range of usable IP addresses**. These results are then presented to the user with clear and concise messages, such as "Calculation successful!" for valid computations or "Invalid IP address or subnet mask. Please try again." for erroneous entries, ensuring a seamless user experience.

The program employs bitwise operations to derive accurate results, ensuring that the calculations adhere to networking standards. Beyond computations, the program incorporates a robust logging mechanism to enhance traceability and accountability. Each successful subnetting calculation is logged into a file named **subnet\_calculations.log**, which records the provided IP address, subnet mask, calculated results, and a timestamp. This log serves as a vital tool for auditing and reviewing subnetting activities. To ensure reliability, the program gracefully handles potential file operation errors, such as the inability to open or write to the log file, and informs users of such issues through appropriate error messages.

Moreover, the program emphasizes secure and efficient resource management. It ensures proper handling of resources, such as closing files and avoiding memory leaks, to maintain the system's integrity. The program terminates only after successfully completing the computation and logging processes, showcasing its commitment to accuracy and reliability. By focusing on user-friendly interaction, accurate subnetting calculations, and secure logging practices, this program not only demonstrates the practical application of IP subnetting principles but also serves as an educational and functional tool for network engineers, IT professionals, and students.

It highlights foundational concepts in networking while offering a reliable solution to real-world subnetting challenges, making it an invaluable resource for network planning, configuration, and management.

## FUNCTIONAL REQUIREMENTS

**User Input Validation**:  
The system must validate the IP address and subnet mask entered by the user to ensure they are in the correct dotted-decimal format and within valid ranges.

1. **Subnet Calculation**:  
   The system should compute and display the following based on the valid IP address and subnet mask:
   * **Network Address**
   * **Broadcast Address**
   * **Usable IP Range**
2. **Error Handling for Invalid Input**:  
   If the user provides an invalid IP address or subnet mask, the system must deny further calculations and prompt the user to re-enter valid inputs.
3. **Logging Results**:  
   After successful computation, the program must log the following details into a file named subnet\_calculations.log:
   * User-provided IP address and subnet mask.
   * Calculated network address, broadcast address, and usable IP range.
4. . **File Operation Handling:**

Handle file-related errors gracefully with appropriate messages.

1. **Program Termination**:

Exit only after successful logging and computation.

**Non-Functional Requirements:**

1. **Usability**:

The system should provide a simple and intuitive command-line interface with clear prompts for user input and concise error messages for invalid entries.

1. **Performance**:

The system should perform calculations and respond to user inputs promptly, ensuring efficient handling of subnetting tasks.

1. **Reliability**:

The program should operate consistently without crashes or unexpected behavior even when processing invalid inputs or encountering file operation errors.

1. **Accuracy**:

The system must perform precise binary arithmetic to ensure that all calculated results, such as the network address, broadcast address, and usable IP range, are correct.

1. **Security**:

The program should handle sensitive operations securely, such as preventing unintended

1. **Maintainability**:

The code should be clean, modular, and well-documented, making it easy to update or extend functionality, such as supporting additional subnetting features.

1. **Portability**:

The system should be portable and run seamlessly on standard C compilers across different operating systems with minimal configuration.

## SOURCE CODE

## 

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

// Function to convert an IP string to an integer array

void ipToArray(const char \*ip, int arr[4]) {

sscanf(ip, "%d.%d.%d.%d", &arr[0], &arr[1], &arr[2], &arr[3]);

}

// Function to calculate the network address

void calculateNetworkAddress(int ip[4], int mask[4], int network[4]) {

for (int i = 0; i < 4; i++) {

network[i] = ip[i] & mask[i];

}

}

// Function to calculate the broadcast address

void calculateBroadcastAddress(int ip[4], int mask[4], int broadcast[4]) {

for (int i = 0; i < 4; i++) {

broadcast[i] = ip[i] | (~mask[i] & 255);

}

}

// Function to print an IP address

void printIP(const char \*label, int ip[4]) {

printf("%s: %d.%d.%d.%d\n", label, ip[0], ip[1], ip[2], ip[3]);

}

int main() {

char ipStr[16], maskStr[16];

int ip[4], mask[4], network[4], broadcast[4];

// Input IP address and subnet mask

printf("Enter IP address (e.g., 192.168.1.10): ");

scanf("%15s", ipStr);

printf("Enter Subnet Mask (e.g., 255.255.255.0): ");

scanf("%15s", maskStr);

// Convert strings to integer arrays

ipToArray(ipStr, ip);

ipToArray(maskStr, mask);

// Calculate network address and broadcast address

calculateNetworkAddress(ip, mask, network);

calculateBroadcastAddress(ip, mask, broadcast);

// Print results

printIP("IP Address", ip);

printIP("Subnet Mask", mask);

printIP("Network Address", network);

printIP("Broadcast Address", broadcast);

// Calculate the range of usable IP addresses

int firstUsable[4], lastUsable[4];

for (int i = 0; i < 4; i++) {

firstUsable[i] = network[i];

lastUsable[i] = broadcast[i];

}

firstUsable[3] += 1; // First usable IP

lastUsable[3] -= 1; // Last usable IP

printIP("First Usable IP", firstUsable);

printIP("Last Usable IP", lastUsable);

return 0;

}

**OUTPUT**

