**Introduction of IP address**

An IP (Internet Protocol) address is a numerical identifier assigned to every device connected to a network that uses the Internet Protocol for communication. It serves two main functions: identifying the host or network interface and providing the location of the host in the network. IP addresses are a fundamental part of the internet, enabling devices to send and receive data across vast and interconnected networks.

**Layer of Operation**

IP addresses function at the Network Layer (Layer 3) of the OSI (Open Systems Interconnection) model. At this layer, the primary responsibility is routing data packets from source to destination across different networks, ensuring they reach the correct recipient.

**Uses of IP address**

1. Device Identification: Every device connected to a network has a unique IP address, which distinguishes it from others.
2. Data Routing: IP addresses guide data packets to their intended destination, much like a postal address ensures mail delivery.
3. Communication Facilitation: They enable seamless communication between devices, whether within a local network or across the internet.
4. Network Management: They help network administrators monitor, troubleshoot, and optimize network performance.

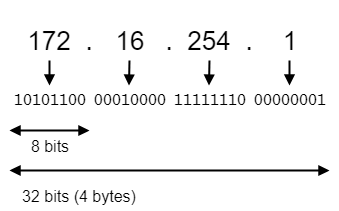
The invention and widespread adoption of IP addresses revolutionized global communication and connectivity. They enabled the internet to grow into a vast web of interconnected devices, fostering the development of e-commerce, social media, cloud computing, and countless other technological advancements. Without IP addresses, the seamless exchange of information across networks would not be possible, limiting technological progress and globalization.

**Classification of IP address**

IP addresses are categorized based on their structure and use:

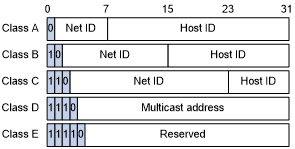
1. **IPv4 (Internet Protocol Version 4):**
   * Composed of 32 bits, typically represented in decimal format (e.g., 192.168.1.1).
   * Provides approximately 4.3 billion unique addresses.
   * Subdivided into classes (Class A, B, C, D, E) based on address range and usage.
2. **IPv6 (Internet Protocol Version 6):**
   * Composed of 128 bits, represented in hexadecimal format (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
   * Designed to address IPv4 exhaustion, offering an almost limitless number of unique addresses.
   * Provides better security, efficiency, and support for modern networks.
3. **Public vs. Private IP Addresses:**
   * **Public IPs** are accessible over the internet and assigned by ISPs.
   * **Private IPs** are used within private networks and cannot be accessed directly over the internet.
4. **Static vs. Dynamic IP Addresses:**
   * **Static IPs** are manually assigned and remain constant.
   * **Dynamic IPs** are assigned automatically and may change over time.

**Classification of IPv4 into Classes**

 IPv4 (Internet Protocol Version 4) is a widely used addressing system in computer networks. It uses a 32-bit address space, divided into four octets. IPv4 addresses are categorized into five classes: A, B, C, D, and E.

**IP address classes**

1. Class A:
   * Range: 0.0.0.0 to 127.255.255.255
   * Purpose: Suitable for very large networks, such as ISPs and large organizations.
   * Network-Host Distribution: 8 bits for the network and 24 bits for hosts (2³² addresses).
   * Default Subnet Mask: 255.0.0.0
2. Class B:
   * Range: 128.0.0.0 to 191.255.255.255
   * Purpose: Designed for medium-sized networks, such as universities and corporations.
   * Network-Host Distribution: 16 bits for the network and 16 bits for hosts (2¹⁶ addresses).
   * Default Subnet Mask: 255.255.0.0
3. Class C:
   * Range: 192.0.0.0 to 223.255.255.255
   * Purpose: Ideal for small networks, such as small businesses and private organizations.
   * Network-Host Distribution: 24 bits for the network and 8 bits for hosts (2⁸ addresses).
   * Default Subnet Mask: 255.255.255.0
4. Class D:
   * Range: 224.0.0.0 to 239.255.255.255
   * Purpose: Reserved for multicasting, enabling data distribution to multiple recipients.
   * Network-Host Distribution: Not applicable; used for group communication.
5. Class E:
   * Range: 240.0.0.0 to 255.255.255.255
   * Purpose: Reserved for experimental and future use; not commonly used in practice.



**Subnetting**

Subnetting is the process of dividing a larger network into smaller, more manageable sub-networks, called subnets. It is a fundamental technique in networking that improves address allocation efficiency, enhances security, and optimizes network performance. Subnetting involves borrowing bits from the host portion of an IP address to create additional network addresses.

**Why Subnetting is Important**

1. **Efficient Address Utilization:**
   * Prevents wastage of IP addresses by allowing network administrators to allocate only the required number of IPs for a specific subnet.
2. **Improved Network Management:**
   * Smaller subnets are easier to manage and troubleshoot compared to one large network.
3. **Enhanced Security:**
   * Subnets can isolate network segments, reducing the spread of threats like malware and unauthorized access.
4. **Reduced Network Traffic:**
   * Local traffic remains within the subnet, reducing unnecessary congestion in other parts of the network.
5. **Better Scalability:**
   * Networks can be expanded more effectively by adding subnets rather than extending the entire network.

**How IPv4 are Subnetted**

Subnetting involves manipulating the subnet mask, which defines how an IP address is divided into its network and host components.

1. **Subnet Mask:**
   * A subnet mask specifies which portion of an IP address belongs to the network and which part is for hosts.
   * Example:
     + Default Class C Subnet Mask: **255.255.255.0**
       - Network: First 24 bits (192.168.1.0)
       - Host: Last 8 bits (0 to 255)
2. **Borrowing Bits:**
   * Bits from the host portion are borrowed to create additional subnet bits. The more bits borrowed, the more subnets created, but the fewer hosts are available per subnet.
3. **Formula for Subnetting:**
   * **Number of Subnets:** 2^*n*, where n = number of bits borrowed.
   * **Hosts per Subnet:** 2^h − 2, where h = remaining host bits (subtract 2 for the network and broadcast addresses).

**Classful and Classless addressing**

IP addressing plays a crucial role in networking, enabling devices to communicate effectively across local and global networks. Over time, different methods have been developed to allocate and organize IP addresses, primarily categorized into classful addressing and classless addressing.

**Classful addressing**

Classful addressing is an IP addressing system that strictly adheres to predefined address classes (Class A, B, C, D, and E). It was the original addressing scheme used in the early days of the internet.

**Key Characteristics:**

* **Fixed Network and Host Portions:** The division of bits into network and host portions is fixed based on the class.
* **No Subnet Mask in Routing:** Subnet masks are implied by the address class and are not explicitly used in routing.
* **Efficient for Simplicity:** Easy to implement but can lead to inefficient use of IP address space.

**Limitations:**

* **Wastage of IP Addresses:** Often allocates more IPs than needed. For example, a Class A network provides over 16 million addresses, even if fewer are required.
* **No Support for VLSM (Variable-Length Subnet Masking):** Cannot create subnets of variable sizes.

**Example of Classful Addressing:**

* A network with an IP address of 192.168.1.0 is automatically assumed to be a Class C network with a default subnet mask of 255.255.255.0.

**Classless addressing**

Classless addressing, introduced with CIDR (Classless Inter-Domain Routing), removes the rigid boundaries of classful addressing. It allows flexible division of an IP address into network and host portions, enabling better address space utilization.

**Key Characteristics:**

* **Custom Subnet Masks:** The network and host portions can be defined using a subnet mask, regardless of the original class.
* **CIDR Notation:** Uses a suffix (e.g., /24) to specify the number of network bits.
* **Supports VLSM:** Allows networks to be divided into subnets of variable sizes based on need.
* **Efficient Utilization:** Minimizes wastage of IP addresses, especially important given IPv4 address exhaustion.

**Benefits:**

* Enables route summarization, reducing the size of routing tables.
* Provides flexibility to allocate IPs based on specific requirements.
* Solves the inefficiencies of classful addressing.

**Example of Classless Addressing:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subnet | CIDR | Subnet mask | IP range | Usable host |
| Subnet 1 | /26 | 255.255.255.192 | 192.168.1.0 - 192.168.1.63 | 62 |
| Subnet 2 | /27 | 255.255.255.224 | 192.168.1.64 - 192.168.1.95 | 30 |
| Subnet 3 | /28 | 255.255.255.240 | 192.168.1.96 - 192.168.1.111 | 14 |

**Identification of network id and host id from the given IP address**

Here is the algorithm to determine the class of a given IP address and identify its Network ID and Host ID:

**Algorithm**

**Input**

A string representing the IP address in the format x.x.x.x.

**Output:**

The class of the IP address.

The Network ID part.

The Host ID part.

**Steps:**

1. **Input Validation**:
   * Read the input IP address as a string.
   * Split the IP address into four parts (octets) using the dot (.) as a delimiter.
   * Check if the number of parts is exactly 4; if not, output "Invalid IP address format" and terminate.
   * Convert the first octet to an integer. If it is not a valid number between 0 and 255, output "Invalid IP address format" and terminate.
2. **Class Determination**:
   * Use the value of the first octet to determine the class:
     + If 1 ≤ first\_octet ≤ 126:
       - Class is A.
       - Network ID = first octet.
       - Host ID = remaining three octets (x2.x3.x4).
     + If 128 ≤ first\_octet ≤ 191:
       - Class is B.
       - Network ID = first two octets (x1.x2).
       - Host ID = remaining two octets (x3.x4).
     + If 192 ≤ first\_octet ≤ 223:
       - Class is C.
       - Network ID = first three octets (x1.x2.x3).
       - Host ID = remaining one octet (x4).
     + If 224 ≤ first\_octet ≤ 239:
       - Class is D.
       - Used for multicast. Network ID and Host ID are N/A.
     + If 240 ≤ first\_octet ≤ 255:
       - Class is E.
       - Reserved for future use. Network ID and Host ID are N/A.
3. **Output Results**:
   * Print the determined class.
   * Print the Network ID and Host ID.
4. **End**.

**Program**

import java.util.Scanner;

public class IPAddressAnalyzer {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input the IP address

System.out.print("Enter an IP address: ");

String ipAddress = scanner.nextLine();

// Split the IP address into octets

String[] octets = ipAddress.split("\\.");

if (octets.length != 4) {

System.out.println("Invalid IP address format.");

return;

}

try {

// Parse the first octet to determine the class

int firstOctet = Integer.parseInt(octets[0]);

String ipClass;

String networkId;

String hostId;

if (firstOctet >= 1 && firstOctet <= 126) {

ipClass = "A";

networkId = octets[0];

hostId = octets[1] + "." + octets[2] + "." + octets[3];

} else if (firstOctet >= 128 && firstOctet <= 191) {

ipClass = "B";

networkId = octets[0] + "." + octets[1];

hostId = octets[2] + "." + octets[3];

} else if (firstOctet >= 192 && firstOctet <= 223) {

ipClass = "C";

networkId = octets[0] + "." + octets[1] + "." + octets[2];

hostId = octets[3];

} else if (firstOctet >= 224 && firstOctet <= 239) {

ipClass = "D (Multicast)";

networkId = "N/A";

hostId = "N/A";

} else if (firstOctet >= 240 && firstOctet <= 255) {

ipClass = "E (Reserved)";

networkId = "N/A";

hostId = "N/A";

} else {

ipClass = "Invalid";

networkId = "N/A";

hostId = "N/A";

}

// Display results

System.out.println("IP Class: " + ipClass);

System.out.println("Network ID: " + networkId);

System.out.println("Host ID: " + hostId);

} catch (NumberFormatException e) {

System.out.println("Invalid IP address format.");

}

scanner.close();

}

}