

Ex.No: 10	ARP protocols using UDP
Date:	

Aim:

To write a java program for simulating ARP protocols using TCP

ALGORITHM:

Client

1. Start the program
2. Using socket connection is established between client and server.
3. Get the IP address to be converted into a MAC address.
4. Send this IP address to the server.
5. Server returns the MAC address to the client.

Server

1. Start the program
2. Accept the socket which is created by the client.
3. Server maintains the table in which IP and corresponding MAC addresses are stored.
4. Read the IP address which is sent by the client.
5. Map the IP address with its MAC address and return the MAC address to the client.

Program

Server Program

```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverarp12
{
    public static void main(String args[])
    {
        try
        {
            DatagramSocket server=new DatagramSocket(1309);
            while(true)
            {
                byte[] sendbyte=new byte[1024];
                byte[] receivebyte=new byte[1024];
                DatagramPacket receiver=new
DatagramPacket(receivebyte,receivebyte.length);
                server.receive(receiver);
                String str=new String(receiver.getData());
                String s=str.trim();
                //System.out.println(s);
                InetAddress addr=receiver.getAddress();
                int port=receiver.getPort();
                String ip[]{"165.165.80.80","165.165.79.1"};
                String mac[]{"6A:08:AA:C2","8A:BC:E3:FA"};
```

```

        for(int i=0;i<ip.length;i++)
        {
            if(s.equals(ip[i]))
            {
                sendbyte=mac[i].getBytes();
                DatagramPacket sender=new
DatagramPacket(sendbyte,sendbyte.length,addr,port);
                server.send(sender);
                break;
            }
        }
        break;
    }
}
catch(Exception e)
{
    System.out.println(e);
}
}
}

```

Client Program

```

import java.io.*;
import java.net.*;
import java.util.*;
class Clientarp12
{
    public static void main(String args[])
    {
        try
        {
            DatagramSocket client=new DatagramSocket();
            InetAddress addr=InetAddress.getByName("127.0.0.1");

            byte[] sendbyte=new byte[1024];
            byte[] receivebyte=new byte[1024];
            BufferedReader in=new BufferedReader(new
InputStreamReader(System.in));
            System.out.println("Enter the logical address (IP):");
            String str=in.readLine();
            sendbyte=str.getBytes();
            DatagramPacket sender=new
DatagramPacket(sendbyte,sendbyte.length,addr,1309);
            client.send(sender);
            DatagramPacket receiver=new
DatagramPacket(receivebyte,receivebyte.length);
            client.receive(receiver);
            String s=new String(receiver.getData());

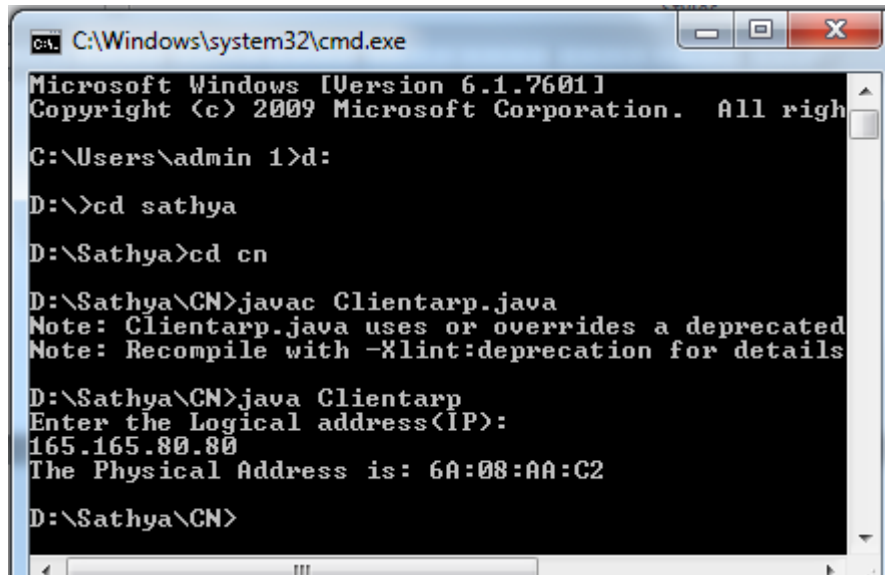
```

```

        System.out.println("The Physical Address is: "+s.trim());
        client.close();
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
}

```

Output:

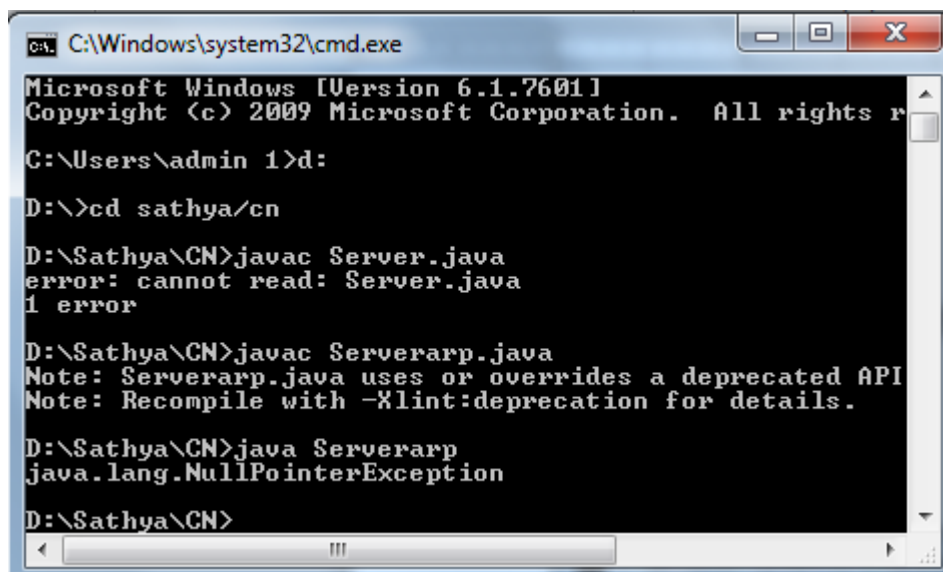


```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\admin 1>d:
D:\>cd sathya
D:\Sathya>cd cn
D:\Sathya\CN>javac Clientarp.java
Note: Clientarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\Sathya\CN>java Clientarp
Enter the Logical address(IP):
165.165.80.80
The Physical Address is: 6A:08:AA:C2
D:\Sathya\CN>

```



```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\admin 1>d:
D:\>cd sathya/cn
D:\Sathya\CN>javac Server.java
error: cannot read: Server.java
1 error
D:\Sathya\CN>javac Serverarp.java
Note: Serverarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\Sathya\CN>java Serverarp
java.lang.NullPointerException
D:\Sathya\CN>

```

RESULT: Thus ARP using UDP has been implemented using JAVA Socket Programming

Ex No: 11	STUDY OF IPV6 ADDRESSING AND SUBNETTING
Date:	

AIM:

To study IPV6 address terminology, IPV6 address format, Types of Addresses, assigning IPaddresses to devices and subnetting.

PROCEDURE:

IPv6 Address Terminology

Node

Any device that runs an implementation of IPv6. This includes routers and hosts.

Router

A node that can forward IPv6 packets not explicitly addressed to itself. On an IPv6 network, a router also typically advertises its presence and host configuration information.

Host

A node that cannot forward IPv6 packets not explicitly addressed to itself (a non-router). A host is typically the source and a destination of IPv6 traffic, and it silently discards traffic received that is not explicitly addressed to itself.

Upper-layer protocol

A protocol above IPv6 that uses IPv6 as its transport. Examples include Internet layer protocols such as ICMPv6 and Transport layer protocols such as TCP and UDP (but not Application layer protocols such as FTP and DNS, which use TCP and UDP as their transport).

Link

The set of network interfaces that are bounded by routers and that use the same 64-bit IPv6unicast address prefix. Other terms for “link” are subnet and network segment.

Network

Two or more subnets connected by routers. Another term for networks is internetworks.

Neighbors

Nodes connected to the same link. Neighbors in IPv6 have special significance because of IPv6 Neighbor Discovery, which has facilities to resolve neighbor link layer addresses and detect and monitor neighbor reach ability

Interface

The representation of a physical or logical attachment of a node to a link. An example of a physical interface is a network adapter. An example of a logical interface is a “tunnel” interface that is used to send IPv6 packets across an IPv4 network by encapsulating the IPv6 packet inside an IPv4 header.

Address

An identifier that can be used as the source or destination of IPv6 packets that is assigned at the IPv6 layer to an interface or set of interfaces.

Packet

The protocol data unit (PDU) that exists at the IPv6 layer and is composed of an IPv6 header and payload.

Link

MTU The maximum transmission unit (MTU)—the number of bytes in the largest IPv6 packet—that can be sent on a link. Because the maximum frame size includes the link-layer medium headers and trailers, the link MTU is not the same as the maximum frame size of the link. The link MTU is the same as the maximum payload size of the link-layer technology. For example, for Ethernet Using Ethernet II encapsulation, the maximum Ethernet frame payload size is 1500 bytes. Therefore, the link MTU is 1500. For a link with multiple link-layer technologies (for example, a bridged link), the link MTU is the smallest link MTU of all the link-layer technologies present on the link

Path

MTU The maximum-sized IPv6 packet that can be sent without performing host fragmentation between a source and destination over a path in an IPv6 network. The path MTU is typically the smallest link MTU of all the links in the path.

IPv6 Address Format

Whereas IPv4 addresses use a dotted-decimal format, where each byte ranges from 0 to 255. IPv6 addresses use eight sets of four hexadecimal addresses (16 bits in each set), separated by a colon (:), like this: `xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx` (x would be a hexadecimal value). This notation is commonly called string notation.

Hexadecimal values can be displayed in either lower- or upper-case for the numbers A–F. A leading zero in a set of numbers can be omitted; for example, you could either enter 0012 or 12 in one of the eight fields—both are correct. If you have successive fields of zeroes in an IPv6 address, you can represent them as two colons (::). For example, 0:0:0:0:0:0:0:5 could be represented as ::5; and ABC:567:0:0:8888:9999:1111:0 could be represented as ABC:567::8888:9999:1111:0. However, you can only do this once in the address: ABC::567::891::00 would be invalid since :: appears more than once in the address. The reason for this limitation is that if you had two or more repetitions, you wouldn't know how many sets of zeros were being omitted from each part. An unspecified address is represented as ::, since it contains all zeros.

Types of IPv6 Addresses

Anycast

An anycast address identifies one or more interfaces. Notice that the term device isn't used since a device can have more than one interface. Sometimes people use the term node to designate an interface on a device. Basically, an anycast is a hybrid of a unicast and multicast address.

- With a unicast, one packet is sent to one destination;
- With a multicast, one packet is sent to all members of the multicast group;
- With an anycast, a packet is sent to any one member of a group of devices that are configured

with the anycast address. By default, packets sent to an anycast address are forwarded to the closest interface (node), which is based on the routing process employed to get the packet to the destination. Given this process, anycast addresses are commonly referred to as one-to-the-nearest addresses.

Multicast

Represent a group of interfaces interested in seeing the same traffic.

- The first 8 bits are set to FF.

- The next 4 bits are the lifetime of the address: 0 is permanent and 1 is temporary.
- The next 4 bits indicate the scope of the multicast address (how far the packet can travel):
1 is for a node, 2 is for a link, 5 is for the site, 8 is for the organization, and E is global(the Internet).

Unicast

The following types of addresses are unicast IPv6 addresses:

- Global unicast addresses
- Link-local addresses
- Site-local addresses
- Unique local addresses
- Special addresses
- Transition addresses

Global Unicast Addresses

IPv6 global addresses are equivalent to public IPv4 addresses. They are globally routable and reachable on the IPv6 Internet. Global unicast addresses are designed to be aggregated or summarized for an efficient routing infrastructure. Unlike the current IPv4-based Internet, which is a mixture of both flat and hierarchical routing, the IPv6-based Internet has been designed from its foundation to support efficient, hierarchical addressing and routing. The scope of a global address is the entire IPv6Internet. RFC 4291 defines global addresses as all addresses that are not the unspecified, loopback,link-local unicast, or multicast addresses. However, Figure shows the structure of global unicast addresses defined in RFC 3587 that are currently being used on the IPv6 Internet. The structure of global unicast addresses defined in RFC 3587. The fields in the global unicast address are described in the following list:

Global Routing Prefix Indicates the global routing prefix for a specific organization's site. The Combination of the three fixed bits and the 45-bit Global Routing Prefix is used to create a 48-bit site prefix, which is assigned to an individual site of an organization. A site is an autonomously operatingIP-based network that is connected to the IPv6 Internet. Network architects and administrators within the site determine the addressing plan and routing policy for the organization network. Once assigned,routers on the IPv6

Internet forward IPv6 traffic matching the 48-bit prefix to the routers of the organization's site.

Subnet ID The Subnet ID is used within an organization's site to identify subnets within its site. The Size of this field is 16 bits. The organization's site can use these 16 bits within its site to create 65,536 subnets or multiple levels of addressing hierarchy and an efficient routing infrastructure. With 16 bits in subnetting flexibility, a global unicast prefix assigned to an organization site is equivalent to a public IPv4 Class A address prefix (assuming that the last octet is used for identifying nodes on subnets). The routing structure of the organization's network is not visible to the ISP.

Interface ID Indicates the interface on a specific subnet within the site. The size of this field is 64 bits. The interface ID in IPv6 is equivalent to the node ID or host ID in IPv4.

Local-Use Unicast Addresses

Local-use unicast addresses do not have a global scope and can be reused. There are two types of local-use unicast addresses: Link-local addresses are used between on-link neighbors and for Neighbor Discovery processes. Site-local addresses are used between nodes communicating with other nodes in the same organization.

Link-Local Addresses FE8:: through FEB::

Link-local addresses are a new concept in IPv6. These kinds of addresses have a smaller scope as to how far they can travel: just the local link (the data link layer link). Routers will process packets destined to a link-local address, but they will not forward them to other links. Their most common uses for a device to acquire unicast site-local or global unicast addressing information, discovering the default gateway, and discovering other layer 2 neighbors on the segment. IPv6 link-local addresses identified by the initial 10 bits being set to 1111 1110 10 and the next 54 bits set to 0, are used by nodes when communicating with neighboring nodes on the same link. For example, on a single-link IPv6 network with no router, link-local addresses are used to communicate between hosts on the link. IPv6 link-local addresses are similar to IPv4 link-local addresses defined in RFC 3927 that use the 169.254.0.0/16 prefix. The use of IPv4 link-local addresses is known as Automatic Private IP Addressing (APIPA) in Windows Vista, Windows Server 2008, Windows Server 2003, and Windows XP. The scope of a link local address is the local link. A link-local address is required for some Neighbor Discovery processes and is always automatically configured, even in the absence of all other

unicast addresses. Link-local addresses always begin with FE80. With the 64-bit interface identifier, the prefix for link-local addresses is always FE80::/64.

Site-Local Addresses FEC:: through FFF::

It represents a particular site or company. These addresses can be used within a company without having to waste any public IP addresses—not that this is a concern, given the large number of addresses available in IPv6. However, by using private addresses, you can easily control who is allowed to leave your network and get returning traffic back by setting up address translation policies for IPv6.

Site-local addresses, identified by setting the first 10 bits to 1111 1110 11, are equivalent to the IPv4 private address space (10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16). For example, private intranets that do not have a direct, routed connection to the IPv6 Internet can use site local addresses conflicting with global addresses. Site-local addresses are not reachable from other sites, and routers must not forward site-local traffic outside the site. Site-local addresses can be used in addition to global addresses. The scope of a site-local address is the site. Unlike link-local addresses, site-local addresses are not automatically configured and must be assigned either through stateless or stateful address autoconfiguration. The first 10 bits are always fixed for site-local addresses, beginning with FEC0::/10. After the 10 fixed bits is a 54-bit Subnet ID field that provides 54 bits with which you can create subnets within your organization. You can have a flat subnet structure, or you can divide the high order bits of the Subnet ID field to create a hierarchical and summarized routing infrastructure. After the Subnet ID field is a 64-bit Interface ID field that identifies a specific interface on a subnet. Site-local addresses have been formally deprecated in RFC 3879 for future IPv6 implementations. However, existing implementations of IPv6 can continue to use site-local addresses.

Zone IDs for Local-Use Addresses

Unlike global addresses, local-use addresses (link-local and site-local addresses) can be reused. Link-local addresses are reused on each link. Site-local addresses can be reused within each site of an organization. Because of this address reuse capability, link-local and site-local addresses are ambiguous. To specify the link on which the destination is located or the site within which the destination is located, an additional identifier is needed. This additional identifier is a zone identifier (ID), also known as a scope ID, which identifies a connected portion of a network that has a specified scope. The

syntax specified in RFC 4007 for identifying the zone associated with a local-use address is Address%zone ID, in which Address is a local-use unicast IPv6 address and zone ID is an integer value representing the zone. The values of the zone ID are defined relative to the sending host.

Therefore, different hosts might determine different zone ID values for the same physical zone. For Example, Host A might choose 3 to represent the zone of an attached link and Host B might choose 4 to represent the same link.

Unique Local Addresses

Site-local addresses provide a private addressing alternative to global addresses for internet traffic. However, because the site-local address prefix can be reused to address multiple sites within an organization, a site-local address prefix can be duplicated. The ambiguity of site local addresses in an organization adds complexity and difficulty for applications, routers, and network managers

To replace site-local addresses with a new type of address that is private to an organization yet unique across all the sites of the organization, RFC 4193 defines unique local IPv6 unicast addresses.

The first 7 bits have the fixed binary value of 1111110. All local addresses have the address prefix FC00::/7. The Local (L) flag is set 1 to indicate that the prefix is locally assigned. The L flag value set to 0 is not defined in RFC 3879. Therefore, unique local addresses within an organization with the L flag set to 1 have the address prefix of FD00::/8. The Global ID identifies a specific site within an organization and is set to a randomly derived 40-bit value. By deriving a random value for the Global ID, an organization can have statistically unique 48-bit prefixes assigned to their sites. Additionally, two organizations that use unique local addresses that merge have a low probability of duplicating a 48-bit unique local address prefix, minimizing site renumbering. Unlike the Global Routing Prefix in global addresses, the Global IDs in unique local address prefixes are not designed to be summarized.

The following are the special IPv6 addresses:

Unspecified address

The unspecified address (0:0:0:0:0:0:0:0 or ::) is used only to indicate the absence of an address. It is equivalent to the IPv4 unspecified address of

0.0.0.0. The unspecified address is typically used as a source address when a unique address has not yet been determined. The unspecified address is never assigned to an interface or used as a destination address.

Loopback address

The loopback address (0:0:0:0:0:0:0:1 or ::1) is assigned to a loopback interface, enabling a node to send packets to itself. It is equivalent to the IPv4 loopback address of 127.0.0.1. Packets addressed to the loopback address must never be sent on a link or forwarded by an IPv6 router.

Transition Addresses

To aid in the transition from IPv4 to IPv6 and the coexistence of both types of hosts, the following addresses are defined:

IPv4-compatible address

The IPv4-compatible address, 0:0:0:0:0:0:w.x.y.z or ::w.x.y.z (where w.x.y.z is the dotted decimal representation of a public IPv4 address), is used by IPv6/IPv4 nodes that are communicating with IPv6 over an IPv4 infrastructure that uses public IPv4 addresses, such as the Internet. IPv4-compatible addresses are deprecated in RFC 4291 and are not supported in IPv6 for Windows Vista and Windows Server 2008.

IPv4-mapped address

The IPv4-mapped address, 0:0:0:0:0:FFFF:w.x.y.z or ::FFFF: w.x.y.z, is used to represent an IPv4 address as a 128-bit IPv6 address.

ISATAP address

An address of the type 64-bit prefix:0:5EFE:w.x.y.z, where w.x.y.z is a private IPv4 address, is assigned to a node for the Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) IPv6 transition technology.

Teredo address

A global address that uses the prefix 2001::/32 and is assigned to a node for the Teredo IPv6 transition technology. Beyond the first 32 bits, Teredo addresses are used to encode the IPv4 address of a Teredo Server, flags, and an obscured version of a Teredo client's external address and UDP port number.

Assigning IPv6 address to Devices

IPv6 Addresses for a Host

An IPv4 host with a single network adapter typically has a single IPv4 address assigned to that adapter. An IPv6 host, however, usually has multiple IPv6 addresses assigned to each adapter. The interfaces on a typical IPv6 host are assigned the following unicast addresses:

A link-local address for each interface

Additional unicast addresses for each interface (which could be one or multiple unique local or global addresses)

- The loopback address (::1) for the loopback interface: Typical IPv6 hosts are always logically multi homed because they always have at least two addresses with which they can receive packets—a link-local address for local link traffic and a routable unique local or global address. Additionally, each interface on an IPv6 host is listening for traffic on the following multicast addresses:

- The interface-local scope all-nodes multicast address (FF01::1)
- The link-local scope all-nodes multicast address (FF02::1)
- The solicited-node address for each unicast address assigned
- The multicast addresses of joined groups

SUBNETTING

A subnetwork or subnet is a logical subdivision of an IP network. The practice of dividing a network into two or more networks is called subnetting. Computers that belong to a subnet are addressed with an identical most-significant bit-group in their IP addresses.

Advantage of Subnetting

- Subnetting allows us to break a single large network in smaller networks. Small Networks are easy to manage.
- Subnetting reduces network traffic by allowing only the broadcast traffic which is relevant to the subnet.
- By reducing unnecessary traffic, Subnetting improves overall performance of the network.
- By blocking a subnet' traffic in the subnet, Subnetting increases security of the network.
- Subnetting reduces the requirement of IP range.

Disadvantage of Subnetting

- Different subnets need an intermediate device known as router to communicate with each other. Since each subnet uses its own network address and broadcast address, more subnets mean more wastage of IP addresses.
- Subnetting adds complexity in the network. An experienced network administrator is required to manage the subnetted network.

Class A Subnets

In Class A, only the first octet is used as Network identifier and the rest of three octets are used to be assigned to Hosts (i.e. $2^{24} - 2$ Hosts per Network). To make more subnet in Class A, bits from Host part are borrowed and the subnet mask is changed accordingly. For example, if one MSB (Most Significant Bit) is borrowed from host bits of second octet and added to Network address, it creates two Subnets ($2^1 = 2$) with $(2^{23} - 2)$ 8388606 Hosts per Subnet.

Class B Subnets

By default, using Classful Networking, 14 bits are used as Network bits providing (2^{14}) 16384 Networks and $(2^{16} - 2)$ 65534 Hosts. Class B IP Addresses can be subnetted the same way as Class A addresses, by borrowing bits from Host bits.

Class C Subnets

Class C IP addresses are normally assigned to a very small size network because it can only have 254 hosts in a network.

RESULT:

Thus the IPV6 address terminology, IPV6 address format, Types of Addresses, assigning IP addresses to devices and subnetting were studied successfully.

Ex No: 12	Implementation of NAT
Date:	

AIM:

To implement Network Address Translation (NAT) Protocol using Java program

ALGORITHM

- A new datagram packet is made consisting of the data from the client and is sent to the server with the device's assigned addresses - hence performing NAT.
- The server on receiving the packet will display the message and then ask for a response. This response is then sent to the device i.e. the address from where the packet came.
- The device will then print the System IP Address
- If An error "Cannot Execute Properly" message is sent either by client or server, the entire established connection will be terminated.

PROGRAM:

```
import java.net.*;
import java.io.*;
import java.util.*;
import java.net.InetAddress;

public class JavaProgram
{
    public static void main(String args[]) throws Exception
    {
        // Returns the instance of InetAddress containing
        // local host name and address
        InetAddress localhost = InetAddress.getLocalHost();
        System.out.println("System IP Address : " +
            (localhost.getHostAddress()).trim());

        // Find public IP address
        String systemipaddress = "";
        try
        {
```

```

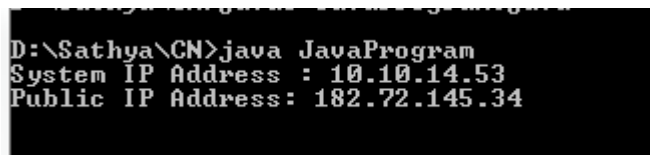
URL url_name = new URL("http://bot.whatismyipaddress.com");

BufferedReader sc =
    new                                BufferedReader(new
InputStreamReader(url_name.openStream()));

    // reads system IPAddress
    systemipaddress = sc.readLine().trim();
}
catch (Exception e)
{
    systemipaddress = "Cannot Execute Properly";
}
System.out.println("Public IP Address: " + systemipaddress + "\n");
}
}

```

Output:



```

D:\Sathya\CN>java JavaProgram
System IP Address : 10.10.14.53
Public IP Address: 182.72.145.34

```

RESULT:

Thus NAT has been implemented