**IMPLEMENT - PING COMMAND USING RAW SOCKETS**

**AIM:**

To write a java program to implement ping command in raw sockets.

**THEORY:**

## Syntax

**ping** [**-t**] [**-a**] [**-n** *Count*] [**-l** *Size*] [**-f**] [**-i** *TTL*] [**-v** *TOS*] [**-r** *Count*] [**-s** *Count*] [{**-j** *HostList* | **-k** *HostList*}] [**-w** *Timeout*] [*TargetName*]

## Parameters

**-t :** Specifies that ping continue sending Echo Request messages to the destination until interrupted. To interrupt and display statistics, press CTRL-BREAK. To interrupt and quit ping, press CTRL-C.

**-a :** Specifies that reverse name resolution is performed on the destination IP address. If this is successful, ping displays the corresponding host name.

**-n *Count* :** Specifies the number of Echo Request messages sent. The default is 4.

**-l *Size* :** Specifies the length, in bytes, of the Data field in the Echo Request messages sent. The default is 32. The maximum *size* is 65,527.

**-f :** Specifies that Echo Request messages are sent with the Don't Fragment flag in the IP header set to 1. The Echo Request message cannot be fragmented by routers in the path to the destination. This parameter is useful for troubleshooting path Maximum Transmission Unit (PMTU) problems.

**-i *TTL* :** Specifies the value of the TTL field in the IP header for Echo Request messages sent. The default is the default TTL value for the host. For Windows XP hosts, this is typically 128. The maximum *TTL* is 255.

**-v *TOS* :** Specifies the value of the Type of Service (TOS) field in the IP header for Echo Request messages sent. The default is 0. *TOS* is specified as a decimal value from 0 to 255.

**-r *Count* :** Specifies that the Record Route option in the IP header is used to record the path taken by the Echo Request message and corresponding Echo Reply message. Each hop in the path uses an entry in the Record Route option. If possible, specify a *Count* that is equal to or greater than the number of hops between the source and destination. The *Count* must be a minimum of 1 and a maximum of 9.

**-s *Count* :** Specifies that the Internet Timestamp option in the IP header is used to record the time of arrival for the Echo Request message and corresponding Echo Reply message for each hop. The *Count* must be a minimum of 1 and a maximum of 4.

**-j *HostList* :** Specifies that the Echo Request messages use the Loose Source Route option in the IP header with the set of intermediate destinations specified in *HostList*. With loose source routing, successive intermediate destinations can be separated by one or multiple routers. The maximum number of addresses or names in the host list is 9. The host list is a series of IP addresses (in dotted decimal notation) separated by spaces.

**-k *HostList* :** Specifies that the Echo Request messages use the Strict Source Route option in the IP header with the set of intermediate destinations specified in *HostList*. With strict source routing, the next intermediate destination must be directly reachable (it must be a neighbor on an interface of the router). The maximum number of addresses or names in the host list is 9. The host list is a series of IP addresses (in dotted decimal notation) separated by spaces.

**-w *Timeout* :** Specifies the amount of time, in milliseconds, to wait for the Echo Reply message that corresponds to a given Echo Request message to be received. If the Echo Reply message is not received within the time-out, the "Request timed out" error message is displayed. The default time-out is 4000 (4 seconds).

***TargetName* :** Specifies the destination, which is identified either by IP address or host name.

**ALGORITHM:**

1. Create a RAW socket in the client program.

2. Get the name of the host whose IP address is to resolve using ICMP.

3. Pass this name to the ICMP server through this socket.

4. The server will respond with the IP address of the host.

5. Receive the response and print it.

**PROGRAM:**

PING SERVER:

import java.io.\*;

import java.net.\*;

importjava.util.\*;

classPingServer

{

public static void main(String[] args)

{

try

{

ServerSocketss=new ServerSocket(2156);

Socket s=ss.accept();

if(s.isConnected())

System.out.println("Connected ...");

System.out.println("Listening ...");

DataInputStream dis=new DataInputStream(s.getInputStream());

DataOutputStream dos=new DataOutputStream(s.getOutputStream());

int no=0;

String ip="";

if((dis.readUTF()).equals("P"))

{

System.out.println("Getting No. Of Packets ...");

no=dis.readInt();

}

if((dis.readUTF()).equals("A"))

{

System.out.println("Getting the Address ...");

ip=dis.readUTF();

}

Process p=Runtime.getRuntime().exec("ping -c "+no+" "+ip);

System.out.println("Running ping -c "+no+" "+ip);

BufferedReaderbr=new BufferedReader(new InputStreamReader(p.getInputStream()));

String ipline=br.readLine();

while(ipline != null )

{

dos.writeUTF(ipline);

ipline=br.readLine();

}

dis.close();

dos.close();

}catch(Exception x)

{

x.printStackTrace();

}

}

}

PING CLIENT :

import java.io.\*;

import java.net.\*;

importjava.util.\*;

public class PingClient

{

public static void main(String[] args)

{

try

{

Socket s=new Socket("localhost",2156);

BufferedReaderbr=new BufferedReader(new InputStreamReader(System.in));

if(s.isConnected())

System.out.println("Connected !!");

Scanner in=new Scanner(System.in);

DataInputStream is=new DataInputStream(s.getInputStream());

DataOutputStreamos=new DataOutputStream(s.getOutputStream());

System.out.println("How many Packets You want to send ? ");

int no=in.nextInt();

System.out.println("Address to be pinged :");

String ip=br.readLine();

os.writeUTF("P");

os.writeInt(no);

os.writeUTF("A");

os.writeUTF(ip);

String pingline=is.readUTF();

while(pingline != null )

{

System.out.println(pingline);

pingline=is.readUTF();

}

os.flush();

os.close();

is.close();

}catch(Exception x)

{

}

}

}

**OUTPUT:**

**Pingserver**

Connected ...

Listening ...

Getting No. Of Packets ...

Getting the Address ...

Running ping -n 5 192.168.100.1

**pingclient**

Connected !!

How many Packets You want to send ?

5

Address to be pinged :

192.168.100.1

Pinging 192.168.100.1 with 32 bytes of data:

Reply from 192.168.100.1: bytes=32 time<1ms TTL=128

Reply from 192.168.100.1: bytes=32 time<1ms TTL=128

Reply from 192.168.100.1: bytes=32 time<1ms TTL=128

Reply from 192.168.100.1: bytes=32 time<1ms TTL=128

Reply from 192.168.100.1: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.100.1:

Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 0ms, Maximum = 0ms, Average = 0ms