package lte;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.InputStreamReader;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;

import java.net.URL;

import java.net.URLConnection;

import java.util.Arrays;

import java.util.HashMap;

import java.util.Iterator;

import java.util.Map;

import java.util.Properties;

import java.util.StringTokenizer;

import java.util.TreeMap;

import java.util.Vector;

public class Gatewaymultireceiver extends Thread

{

String sysno,portno,gatewayname,status,supersource,sportno,ssysno,sstatus,allvideo="";

public HashMap gatewayportno = new HashMap();

public HashMap gatewaysysno = new HashMap();

public HashMap nodeportno = new HashMap();

public HashMap nodesysno = new HashMap();

public Vector allgateway = new Vector();

public HashMap allstatus = new HashMap();

public HashMap alltraffic = new HashMap();

public HashMap allinsysno = new HashMap();

public HashMap allinport = new HashMap();

public HashMap allintraffic = new HashMap();

public Vector allinnodes = new Vector();

public TreeMap allvideocount = new TreeMap();

public HashMap<String, Vector> chmap = new HashMap<String, Vector>();

public Vector vvv = new Vector();

public Gatewaymultireceiver(String gatewayname,String portno,String sysno)

{

this.gatewayname = gatewayname;

this.sysno = sysno;

this.portno = portno;

start();

}

@Override

public void run()

{

while (true)

{

try

{

InetAddress in = InetAddress.getByName("225.89.67.45");

MulticastSocket ms = new MulticastSocket(4567);

ms.joinGroup(in);

byte[] b = new byte[1024];

DatagramPacket dp = new DatagramPacket(b, b.length);

ms.receive(dp);

String data1 = new String(dp.getData()).trim();

StringTokenizer str = new StringTokenizer(data1, "$");

String status = str.nextToken();

if(status.equalsIgnoreCase("LOCALGATEWAY"))

{

String gateway = str.nextToken();

String sys = str.nextToken();

String port = str.nextToken();

String sstatus = str.nextToken();

String traffic = str.nextToken();

gatewayportno.put(gateway,port);

gatewaysysno.put(gateway, sys);

if(!allgateway.contains(gateway))

allgateway.add(gateway);

allstatus.put(gateway,sstatus);

alltraffic.put(gateway, traffic);

}

else if(status.equalsIgnoreCase("NODEDETAILS"))

{

String node = str.nextToken();

String nsys = str.nextToken();

String nport = str.nextToken();

nodesysno.put(node,nsys);

nodeportno.put(node,nport);

}

else if(status.equalsIgnoreCase("TRAFFIC"))

{

String gateway = str.nextToken();

String traffic = str.nextToken();

alltraffic.put(gateway,traffic);

}

else if(status.equalsIgnoreCase("INTERMEDIATE"))

{

String node = str.nextToken();

String sys = str.nextToken();

String port = str.nextToken();

String traffic = str.nextToken();

allinport.put(node,port);

allinsysno.put(node,sys);

allintraffic.put(node, traffic);

if(!allinnodes.contains(node))

{

allinnodes.add(node);

}

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

public String urlcall()

{

try

{

File f = new File("");

String proper = f.getAbsolutePath();

String filepath = proper+"\\address.properties";

FileInputStream fs =new FileInputStream(filepath);

Properties pro = new Properties();

pro.load(fs);

String location = pro.getProperty("IP");

URL url = new URL("http://"+location+":9999/SAVEP2p//Sender?");

URLConnection uc = url.openConnection();

BufferedReader ins = new BufferedReader(new InputStreamReader(uc.getInputStream()));

String inputLine;

StringBuffer response = new StringBuffer();

while ((inputLine = ins.readLine()) != null)

{

response.append(inputLine);

}

ins.close();

String res = response.toString().trim().replace("{", "").replace("}", "");

String[] spl = res.toString().split("#");

for (int i = 0; i < spl.length; i++)

{

StringTokenizer stk = new StringTokenizer(spl[i], "=");

String cat = stk.nextToken().toString().trim();

String chListInCat = stk.nextToken().toString().trim().replace("[", "").replace("]", "");

Vector v = new Vector(Arrays.asList(chListInCat.split(",")));

chmap.put(cat, v);

}

Iterator it = chmap.entrySet().iterator();

while (it.hasNext())

{

Map.Entry pair = (Map.Entry) it.next();

Vector vv = (Vector) pair.getValue();

for (int j = 0; j < vv.size(); j++)

{

String videoname = vv.get(j).toString();

if(!vvv.contains(videoname))

{

vvv.add(videoname);

allvideo=allvideo+videoname+"\n";

}

}

}

} catch (Exception e)

{

e.printStackTrace();

}

return allvideo;

}

}