Design and Development of Network Monitoring and Controlling Tool for Domain Controller @Department of Computer Studies, CISBER using RMI Technology- A Case Study.

Dr. P.G. Naik [#1], Mr. M.B. Patil [*2]

[#] *Professor, Department of Computer Studies, CSIBER, Kolhapur (MS), India*
[*] *System Analyst, Department of Computer Studies, CSIBER, Kolhapur (MS), India*

Abstract**—** *A large and a moderate sized organization typically consists of a computer network comprising of nodes of different configuration. Further, based on the position of an employee in an organizational hierarchy the network user can be granted permission for accessing certain resources. This can be achieved at the application level by performing an employee role mapping to a pre-defined task list. In the case of shared resources disciplined approach is desirable for proper sharing of band width based on the factors such as a priority of a task, role of an employee in the organization etc., failing of which will result in a legitimate user being deprived of his fair share. In their earlier work, the authors have designed and developed a network monitoring and controlling tool using both single threading and multi threading models. The tool has been thoroughly tested in work group environment and works as per the expectations. However, the tool has not taken care of domain controller and as such has some limitations it its scope and usability. In order to overcome these limitations, the authors have re-designed the tool for domain controller using Java's RMI technology. A single site of control is achieved using FTP and Telnet technologies. As before, the task permissions and the duration for which the tasks are permissible on different machines are stored in XML configuration file which is parsed using JDOM (Java Document Object Model) parser. The configuration file contains the details such as the machine name and the list of tasks not permitted on that machine during a specified time interval. The list of machines and the list of tasks not permitted on that machine for the given time period are configurable by an end user. A user friendly interface is developed for this purpose.*

Keywords — *FTP, JDOM, Remote Method Invocation, Network, Tasklist, Telnet, Workgroup.*

I. Introduction

In today's digital age, the frequency of data transfer and storage has increased tremendously. There is an exponential increase in data transfer rate. To cater to the needs of an individual network user, monitoring and controlling of a network has gained a tremendous importance. Monitoring and diagnosis of network conditions is a central problem in networking. Monitoring refers to the meticulous examination of the entire network and system for changes in the status of individual components which primarily result in ill effects leading to sluggish and unresponsive systems and slowly migrate the network from satisfactory to a status requiring attention. Network monitoring is the information collection function of network management. Network monitoring applications are created to collect data for network management applications. The purpose of network monitoring is the collecting of useful information from various parts of the network so that the network can be managed and controlled by analyzing the collected information. Controlling is the task of taking specific actions against individual network and system components to change their status and make them available for monitoring, or to manipulate the use of the resources which guarantees the fair share of network resources. Network monitoring provides the information necessary for network management. It is important to find network trends, network bottlenecks and locate network problems quickly. At the lower layers of the OSI model there are three standards RMON, RMON2 and ATM-RMON in the single network monitoring where RMON is the remote monitoring standard created by IETF, Internet Engineering Task Force. SMON is an extension to RMON2 for switched networks. ATM-RMON is the ATM, Asynchronous Transfer Mode, equivalent for RMON. ATM is a newer technology and remote monitoring is newly added to it. In this paper, the authors focus mainly on monitoring and controlling network at application layer without focusing much on intricacies of lower layers. Network monitoring applications need to use effective ways of checking the status of their networks so that network management applications can fully control their network and provide economical, and high-quality networking services to the users. It is very important to know what are the goals to achieve in network monitoring. By knowing the goals of network monitoring, network monitoring application can choose among network monitoring techniques that will best help them monitor their networks.

There are generally three basic goals for network monitoring

- Performance monitoring
- Fault monitoring

- Account monitoring

Performance monitoring deals with measuring the performance of the network. There are three important issues in performance monitoring. First, performance monitoring information is usually used to plan future network expansion and locate current network usage problems. Second, the time frame of performance monitoring must be long enough to establish a network behaviour model. Third, choosing what to measure is important. There are too many measureable things in a network. But the list of items to be measured should be meaningful and cost effective. This list of items to be measured is called network indicators because they indicate attributes of the network.

In this paper the authors focus on proactive measure for network control by enabling only the legitimate users to use the network in their specified time frame. In their earlier work, authors have designed and implemented network monitoring and controlling tool for the workgroup using both a single threading and a multi threading model which poses some limitations in their scope and usability [1, 2]. In a workgroup, all computers are peers and no computer has control over another computer. Each computer has a set of user accounts and in order to use any computer in the workgroup, the user must have a local account on that computer. Also, the size of the network is limited as there are typically no more than ten to twenty computers in a workgroup. All computers must be on the same local network or subnet and security is an issue in such networks. Most of the organizations today host domain controllers for security concerns and for managing access control levels of different users. On a domain controller one or more computers are servers. Network administrators use servers to control the security and permissions for all computers on the domain. If a user has an account on the domain, he can log on to any computer on the domain without needing an account on that computer. There can be hundreds or thousands of computers and the computers can be on different local networks. Keeping in mind the scope and usability of such networks, the authors have re-implemented their tool for domain controller using RMI technology.

2. Literature Review

In literature, there exist variety of tools for monitoring and controlling network traffic for both wired and wireless networks and most of them aim at controlling of network traffic for small networks. Hwang et. al. [3]. have proposed the design and implementation of a remote monitoring and controlling system using ZigBee networks. ZigBee is a specification for a suite of networking, security and application software layers using small, low-power, low data rate communication technology based on IEEE 802.15.4 standard for personal area networks. They have employed web services and a smart phones for the client system to monitor and control the home. Over the past few decades, several approaches have been proposed for inferring network characteristics of interest such as topology, packet loss rate, delay, and failures using end-to-end measurements [4, 5]. This class of problems is commonly referred to as network tomography. Active measurement techniques have been proposed that send sequences of probe packets from a set of sources to a set of receivers, and infer link-level metrics of interest from the received packets. Some techniques send probes over unicast paths [6]. ANSWER [7] is the expert system responsible for monitoring AT&T's 4ESS switches. These switches are extremely important, since they handle virtually all of AT&T's long distance traffic. ANSWER is implemented in R++, a rule-based extension to the C++ object-oriented programming language, and is innovative because it employs both rule-based and object-oriented programming paradigms. The use of object technology in ANSWER has provided a principled way of modelling the 4ESS and of reasoning about failures within the 4ESS. This has resulted in an expert system that is more clearly organized, easily understood and maintainable than its predecessor, which was implemented using the rule-based paradigm alone. ANSWER has been deployed for more than a year and handles all 140 of AT&T's 4ESS switches and processes over 100,000 4ESS alarms per week[7]. There are several open-source and commercial network monitoring tools and architectures. The most popular systems that are related to ANEMOS. Among the open-source tools, the most closely related to ANEMOS are PingER [8], Surveyor, the National Inter- net Measurement Infrastructure (NIMI) [9-11], and the Network Weather Service [14]. PingER uses Ping to measure RTTs and loss rates to hundreds of hosts around the world. PingER provides performance information and long-term trends about many different geographical areas of the Internet. Surveyor is a measurement infrastructure that is being deployed at several sites around the world. Surveyor measures the performance of Internet paths using the IETF IPPM standardized metrics [12], [13]. Vaarandi [15] has developed a lightweight platform independent tool for rule-based event correlation called sec (Simple Event Correlator). The primary design goal of sec was to create an open source tool that could be used for both central and local event correlation, regardless of the underlying operating system, and that could be integrated into an arbitrary network management system. Sec is a rule-based event correlation tool that receives its input events from a file stream, and produces output events by executing user-specified shell commands. Yang et.al. [16] have presented Eden, an interactive, direct manipulation home network management system aimed at end users. Eden supports a range of common tasks, and provides a simple conceptual model that can help users understand key aspects of networking better. It provides a range of mechanisms for supporting end user management of home networks while retaining compatibility with existing IP based applications and devices. The system leverages a novel home network router that acts as a "dropin" replacement for users' current router. Authors demonstrated that Eden not only improves the user experience of networking, but also aids users in forming workable conceptual models of how the network works..

3. *DESIGN FRAMEWORK*

3.1 Application Architecture

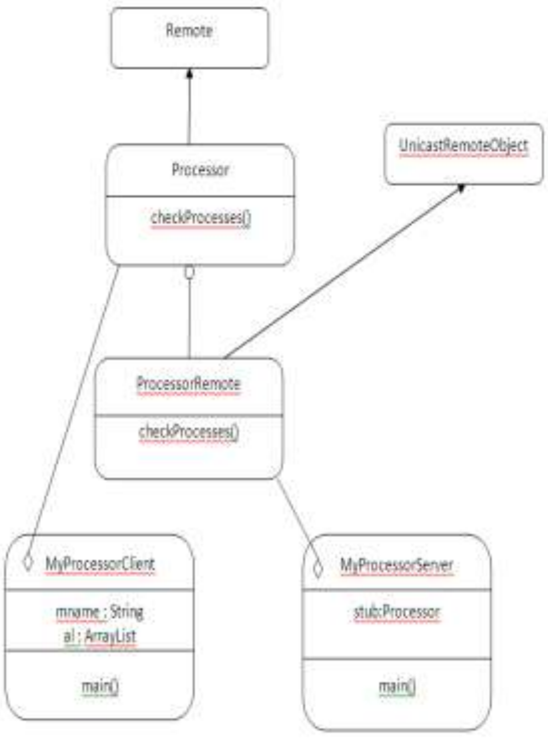


Fig. 1. Application Architecture for Network Monitoring and Controlling Tool

3.2 Control Flow Logic for Network Configuration and Monitoring

Control flow logic for configuring computers on domain controller and continuous monitoring the computers for execution of illegal tasks is depicted in Fig.2 a) and Fig. 2 b), respectively.

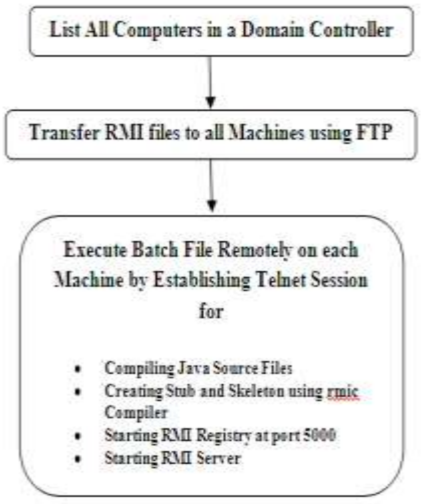

List All Computers in a Domain Controller
Transfer RMI files to all Machines using FTP
Execute Batch File Remotely on each Machine by Establishing Telnet Session for
Compiling Java Source Files
Creating Stub and Skeleton using rmic Compiler
Starting RMI Registry at port 5000
Starting RMI Server

Fig. 2 a). Control Flow Logic for Network Configuration

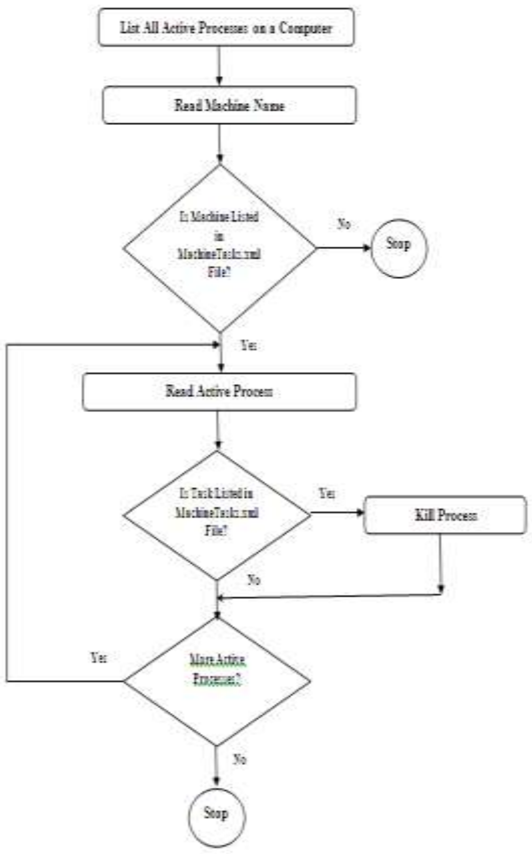

List All Active Processes on a Computer
Read Machine Name
Is Machine Listed in MachineTasks.xml File?
No
Stop
Yes
Read Active Process
Is Task Listed in MachineTasks.xml File?
Yes
Kill Process
No
Yes
More Active Processes?
No
Stop

Fig. 2 b) Control Flow Logic for Network Monitoring

3.3 Machine Configuration for transferring the files from centralized controller machine to a target machine.

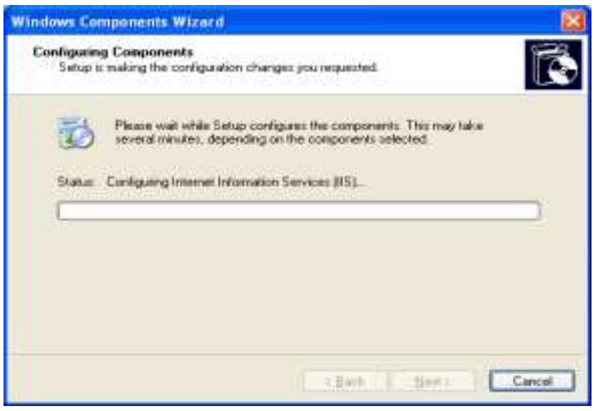
The pre-requisites on a target machine are specified below:

- The target machine should be configured for both FTP and Telnet.
- FTP user must be created on a target machine.
- Virtual Directory must be created on the target machine and must be mapped to a physical folder.

The detailed machine configuration for achieving each of these tasks is as follows:

3.3.1 Enabling FTP on the target machines.

- Select "Add or Remove Programs" applet in Control Panel.
- Select Add/Remove windows Components.
- Select Internet Information services and click on 'Details' button.
- Select File Transfer Protocol (FTP) service.
- Click on 'Next' button.
- Files needed are listed below.
    - ftpmib.dll
    - ftpctrs.h2_


Windows Components Wizard

Configuring Components
Setup is making the configuration changes you requested.

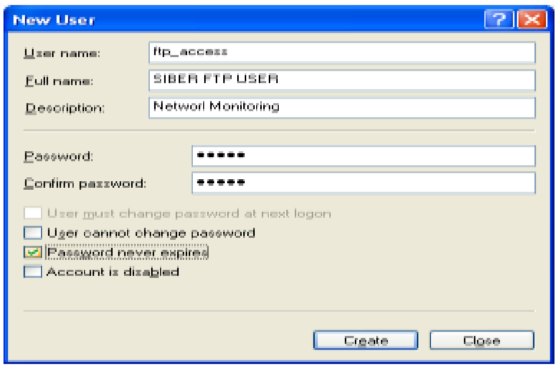
Please wait while Setup configures the components. This may take several minutes, depending on the components selected.

Status: Configuring Internet Information Services (IIS)...

Fig. 3. Configuring FTP on a Target Machine

3.3.2 Creating FTP Users.

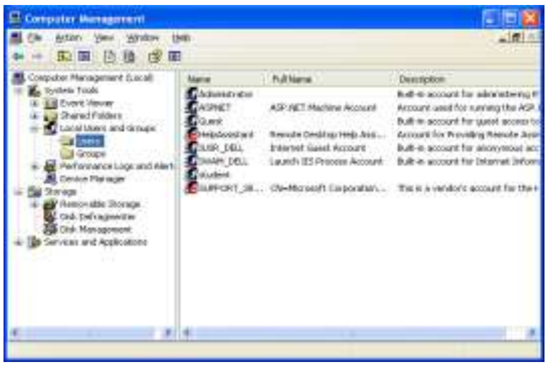
- Right-click on 'My Computer' and select 'Manage' from shortcut menu.
- Expand 'Local Users and Groups' and select Users.
- Right-click and select 'New User' from the shortcut menu. Enter the username ftp_access and password siber as shown below.
- Select 'Password Never Expires' option as shown.


New User

User name: ftp_access
Full name: SIBER FTP USER
Description: Networl Monitoring

Password: •••••
Confirm password: •••••

[ ] User must change password at next logon
[ ] User cannot change password
[✓] Password never expires
[ ] Account is disabled

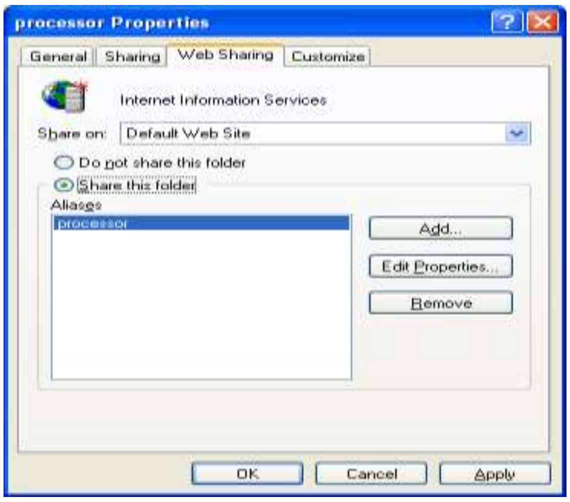

Computer Management

Name	Full Name	Description
Administrator		Built-in account for administering
ASPNET	ASP.NET Machine Account	Account used for running the ASP.
Guest		Built-in account for guest access to
HelpAssistant	Remote Desktop Help Ass...	Account for Providing Remote Ass
IUSR_DELL	Internet Guest Account	Built-in account for anonymous acc
IWAM_DELL	Launch IIS Process Account	Built-in account for Internet Inform
student		
SUPPORT_38...	CN=Microsoft Corporation,...	This is a vendor's account for the

Fig. 4. Creating FTP Users

3.3.3 Creating Virtual Directory and mapping it to a physical folder on a target machine.

- Create a folder with the name 'processor' on C:\ drive.
- Right-click on the folder and select 'Web Sharing' from the shortcut menu.
- Share the folder with the alias name 'processor' as shown below:
- Click on 'Edit Properties' button and assign all access permissions as shown in Fig. 5.

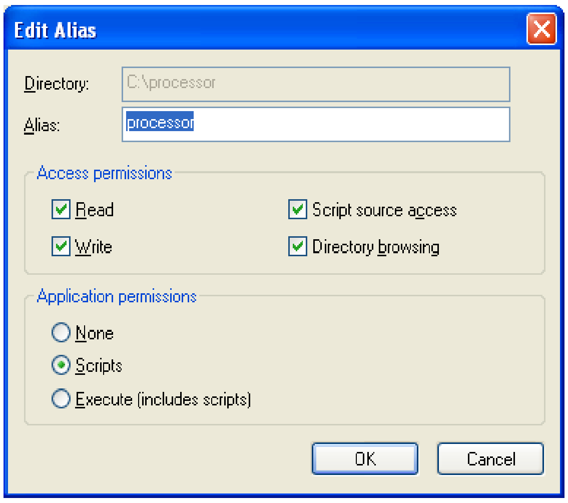

processor Properties

Internet Information Services

Share on: Default Web Site

( ) Do not share this folder
(●) Share this folder

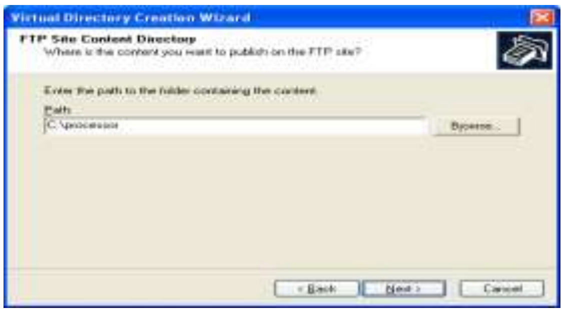
Aliases
processor



Directory: C:\processor
Alias: processor
[✓] Read
[✓] Write
[✓] Script source access
[✓] Directory browsing
None [✓ Scripts] Execute (includes scripts)

Fig. 5. Creating Virtual Directory

- In control panel, select 'Administrative Tools' and then 'Internet Information Services'. After successful FTP installation, FTP site will be visible in Internet Information Services panel.
- Expand FTP site and select 'Default FTP site'.
- Right-click and select 'New Virtual Directory' from the shortcut menu.
- Create an alias 'processor' and select processor folder created earlier on C:\ drive containing the required content. Now, the physical path c:\processor is mapped to the virtual directory processor.
- Allow both read and write access permissions to the folder. The virtual directory processor is successfully created.



FTP Site Content Directory
Where is the content you want to publish on the FTP site?
Enter the path to the folder containing the content
Path: C:\processor

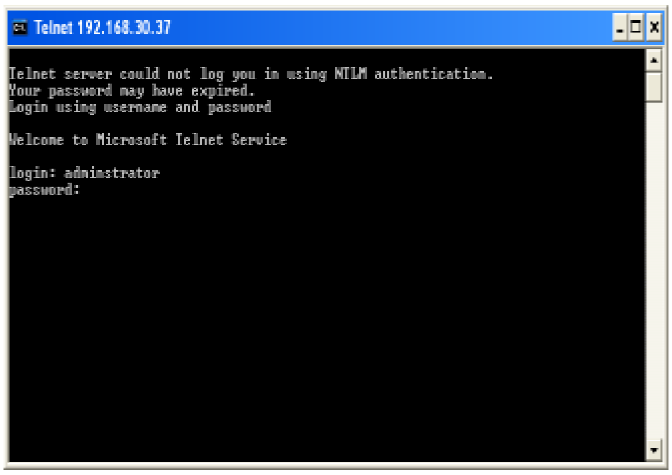
Fig. 6. Mapping Virtual Directory to a Physical Directory in IIS

3.4 Starting Telnet and executing the commands remotely on Windows XP.

- Start the Telnet service if it is not already started and set it to automatic.
- For connecting to dell37 machine issue the following command.

  telnet 192.168.30.37

  telnet.exe is located in the in C:\Windows\system32 folder. Set path to this folder for accessing telnet from any location.
- Enter the username and password for accessing the machine.
- In order to terminate the telnet session type exit.



```
Telnet server could not log you in using NTLM authentication.
Your password may have expired.
Login using username and password

Welcome to Microsoft Telnet Service

login: adminstrator
password:
```

Fig. 7. Using Telnet for Remote Execution

3.5 Setting up XML Configuration File for storing Task Permissions.

For the smooth operation of the network and controlling the execution of certain tasks only on certain machines and only for certain users based on their position in the organization/institution, the list of tasks which are not permissible on certain machines is documented in an XML file with the structure and the corresponding DTD is shown in Fig. 8.

3.5.1 Structure of XML File

```
<machines>
  <machine name="dell43">
    <deny>
      <task>firefox.exe</task>
      <from>2PM</from>
      <to>4PM</to>
    </deny>
  </machine>
  <machine name="dell44">
    <deny>
      <task>iexplore.exe</task>
      <from>*</from>
      <to>*</to>
    </deny>
  </machine>
</machines>
```

* indicates unrestricted access to the resource on that particular machine.

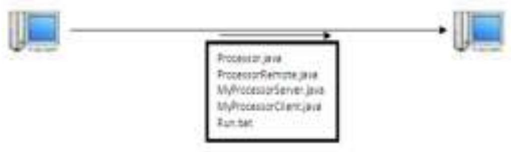
3.5.2 Document Type Definition for XML File

```
<?XML version=”1.0” ?>
<!ELEMENT machines (machine+)>
<!ELEMENT machine (deny)>
<!ATTLIST name CDATA #REQUIRED>
<!ELEMENT deny (task+)>
<!ELEMENT task (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT to (#PCDATA)>
```

Fig. 8. Structure of XML File with Document Type Definition

4. Results and Analysis

The application model presented above is implemented in Java using RMI technology. DRPGN is the machine in the network which acts a central controlling point for file transfer and execution.



DRPGN

DELL37

Fig. 9. Transferring Files using FTP

To enable the file transfer, a batch file runf.bat is created containing the following content.

```
open 192.168.30.37
ftp_access
siber
hash
asc
cd processor
lcd c:\processor
put Processor.java
put ProcessorRemote.java
put MyProcessorServer.java
put MyProcessorClient.java
Run.bat
Bye
```

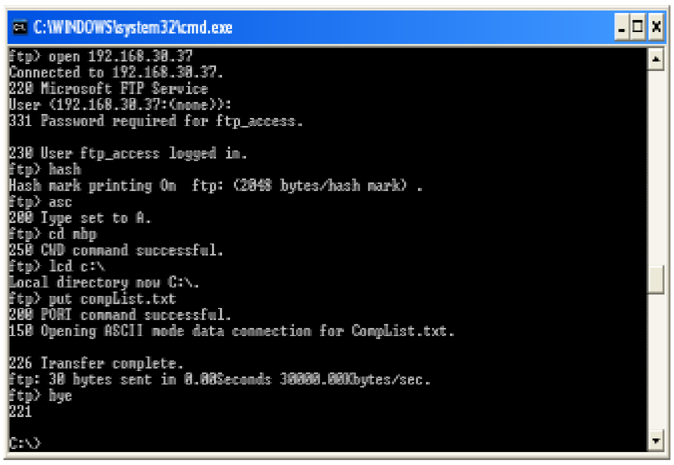
cd -> Refers to Remote Machine 192.168.30.37 and refers to virtual directory.

lcd -> Refers to Local Machine and refers to physical directory.

lcd refers to ‘local change directory’ and is by default set to a folder from where ftp command is executed. cd refers to remote change directory. lcd command uses as its argument a physical directory while cd command uses virtual directory name, an alias created for the mapped physical directory.
The batch file is executed from the Java program using the following command-line option.

ftp -s:runf.bat

Local Directory is set to c:\ processor which contains all the required files and are copied from local directory on DRPGN to the directory c:\processor on remote machine with IP address, 192.168.30.37.
The following output is generated:



```
ftp> open 192.168.30.37
Connected to 192.168.30.37.
220 Microsoft FTP Service
User (192.168.30.37:(none)):
331 Password required for ftp_access.

230 User ftp_access logged in.
ftp> hash
Hash mark printing On  ftp: (2048 bytes/hash mark) .
ftp> asc
200 Type set to A.
ftp> cd mbp
250 CWD command successful.
ftp> lcd c:\
Local directory now C:\.
ftp> put compList.txt
200 PORT command successful.
150 Opening ASCII mode data connection for CompList.txt.

226 Transfer complete.
ftp: 30 bytes sent in 0.00Seconds 30000.00Kbytes/sec.
ftp> bye
221

C:\>
```

Fig. 10. Executing FTP Commands stored in a Batch File from a Java Program

The parameters employed are listed in the following table along with their meaning.

ftp_access	FTP Username
Siber	FTP Password
processor	Virtual Directory on 192.168.30.37 mapped to physical folder c:\siber
processor	Virtual Directory on local machine mapped to physical folder c:\uploads

Using Telnet to execute commands stored in run.bat.

For executing the batch file remotely, two alternating methods have been used and tested.

i) Executing the Telnet program from Java by spawning a separate command-line process.
The partial code for achieving this is depicted below:

```
ProcessBuilder builder = new
ProcessBuilder("cmd.exe", "/c", "
start
telnet.exe 192.168.30.37");
builder.redirectErrorStream(true);
Process p = builder.start();
BufferedReader r = new
BufferedReader(new
InputStreamReader(p.getInputStrea
m()));
String line;
while (true) {
       line = r.readLine();
 if (line == null) { break; }
System.out.println(line);
```

Similary. *SendKeys.vbs* command is executed from the command line.

ii) Using the class TelnetClient located in java package org.apache.commons.net.telnet.

The telnet session is established using the following commands

```
start telnet.exe 192.168.30.37
SendKeys.vbs
```

Run.bat file with the following content is remotely executed

Contents of run.bat are as follows:

```
cd\
cd c:\processor
set classpath=c:\jdom.jar;.
javac Processor.java
javac ProcessorRemote.java
javac MyProcessorServer.java
javac MyProcessorClient.java
rmic ProcessorRemote
start rmiregistry 5000
start java MyProcessorServer
pause
```

Contents of SendKeys.vbs is as follows

```
set OBJECT=WScript.CreateObject("WScript.Shell")
WScript.sleep 50
OBJECT.SendKeys "y{ENTER}"
WScript.sleep 50
OBJECT.SendKeys "administrator{ENTER}"
WScript.sleep 50
OBJECT.SendKeys "password{ENTER}"
WScript.sleep 50
OBJECT.SendKeys "cd{\} {ENTER}"
WScript.sleep 50
OBJECT.SendKeys "cd{\}siber {ENTER}"
WScript.sleep 50
OBJECT.SendKeys "run {ENTER}"
```

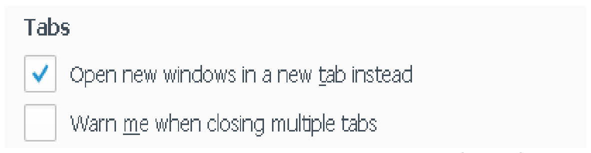
Complete Java Code for parsing XML file using JDOM Parser and Killing Illegal Processes is furnished in Appendix A and the partial code for connecting to Telnet server using TelnetClient class is furnished in Appendix B.

Termination of Firefox with multiple tabs open.

During the termination of illegal tasks another challenge faced was terminating the Firefox process when multiple tabs are open. The solution to this problem is proposed as follows.
The Firefox options settings are located in prefs.js file located in the folder,

C:\Documents and Settings\<windows user name>\Application Data\Mozilla\Firefox\Profiles\0in3o8j4.default

Start Mozilla Firefox browser and enter 'about:config' in address bar. The current Firefox configurations are displayed. Browser.tabs.warnOnclose option is set to true by default. On the General tab of options dialog of Mozilla Firefox clear the 'Warn me when closing multiple tabs as shown below.

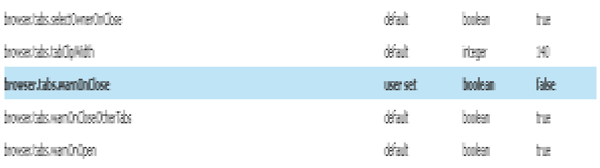


Tabs

[✓] Open new windows in a new tab instead

[ ] Warn me when closing multiple tabs

The options will be written to the prefs.js file on terminating Firefox. The content in prefs.js file is changed as shown in Fig. 11.



browser.tabs.selectOwnerOnClose	default	boolean	true
browser.tabs.tabClipWidth	default	integer	140
browser.tabs.warnOnClose	user set	boolean	false
browser.tabs.warnOnCloseOtherTabs	default	boolean	true
browser.tabs.warnOnOpen	default	boolean	true

Fig. 11. Firefox Options Stored in prefs.js File

5. CONCLUSION AND SCOPE FOR FUTURE WORK

In this paper the authors have presented a generic network management automation tool for monitoring the illegal tasks executed on different machines of the domain controller. Java RMI technology is employed for executing processes remotely in a distributed environment. A single site of control is established using FTP for transferring the required files to all machines which are part of a domain controller. Telnet is used for remote execution of the batch file. Two different approaches are specified for establishing the Telnet session, by spawning a separate processes and using TelnetClient class located in a package org.apache.commons.net.telnet. Machine-task mapping information is stored in an XML file which is well formed and valid. The Document Type Definition (DTD) for the XML document is presented. The Graphical User Interface (GUI) is designed in VB for editing XML configuration file and other functions such as listing all machines in the current network, identifying active processes on each machine in the network, validating these processing against data stored in XML file by parsing XML file using JDOM parser etc. are carried out in Java. The communication between VB and Java is established using intermediate RDBMS system designed in Ms-Access. Our future focus is on upgrading the tool for hybrid network.

REFERENCES

[1] Design and Development of Network Monitoring and Controlling Tool for Department of Computer Studies CSIBER, International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Volume: 3 Issue: 3 1237 – 1243

[2] Design and Development of Multithreaded Network Monitoring and Controlling Tool for Department of Computer Studies CSIBER, Proceedings of UGC Sponsored National Seminar on Recent Trends in Information Technology, organized by Department of Computer Studies CSIBER, Kolhapur, 6th and 7th March, 2015.

[3]. Soyoung Hwang and Donghui Yu, Remote Monitoring and Controlling System Based on ZigBee Networks, International Journal of Software Engineering and Its Applications Vol. 6,No. 3, July, 2012

[4] R. Caceres, N. G. Du±eld, J. Horowitz and D. Towsley, \Multicast-based inference of network-internal loss characteristics", IEEE Trans. in Inf. Theory, vol. 45, pp. 2462-2480, 1999

[5] M. Rabbat, R. Nowak and M. Coates, \Multiple source, multiple destination network tomography",in Proc.of IEEE Infocom 2004

[6] Y. Chen, D. Bindel, H.Song and R.Katz, \An algebraic approach to practical and scalable overlay network monitoring,"in Proc. ACM SIGCOMM 2004

[7] ANSWER: Network Monitoring Using Object-Oriented Rules Gary M. Weiss, Johannes P. Ros and Anoop Singhal Proceedings of the Tenth Conference on Innovative Applications of Artificial Intelligence

[8] W. Mathews and L. Cottrell, “The PingER project: Active internet performance monitoring for the HENP community,” IEEE Commun. Mag., vol. 38, no. 5, pp. 130–136, May 2000.

[9]S. Kalidindi and M. Zekauskas, “Surveyor: An infrastructure for internet performance measurements,” in Proc. INET’99, 1999.

[10]R. Wolski, N. Spring, and C. Peterson, “Implementing a performance forecasting system for metacomputing: the network weather service,” in Proc. of Supercomputing, 1997.,AAAI Press, 1998, 1087-1093

[11] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis, “An architecture for large-scale internet measurement,” IEEE Commun. Mag., vol. 36, no. 8, pp. 48–54, Aug. 1998.

[12]G. Almes, S. Kalidindi, and M. Zekauskas, “A one-way delay metric for IPPM,” RFC2679, 1999.

[13]G.Almes, S. Kalidindi, and M. Zekauskas, “A one-way loss metric for IPPM,” RFC2680, 1999.

[14] T. Oetiker and D. Rand. Multi router traffic grapher. [Online]. Available: http://www.mrtg.org/

[15] Risto Vaarandi, “Platform Independent Event Correlation Tool forNetwork Management”, Proceedings of the 2002 IEEE/IFIP NetworkOperations and Management Symposium.

[16] Jeonghwa Yang, W. Keith Edwards, David Haslem, “Eden: SupportingHome Network Management Through Interactive Visual Tools”, UIST’10 October 3-6, 2010, New York.

Appendix A

Complete Java Code for parsing XML file using JDOM Parser and Killing Illegal Processes

```
   import java.rmi.*;
   import java.rmi.server.*;
   import java.util.*;
   import java.io.*;

   public class ProcessorRemote extends
UnicastRemoteObject implements Processor{
   ProcessorRemote()throws
RemoteException{
   super();
   }
   public void checkProcesses(ArrayList al)
    {
      while (true)
      {
      String
uname=System.getProperty("user.name").toL
owerCase();
      System.out.println("user:"+uname);
      if (uname.equals("administrator"))
        return;
      try
      {
        System.out.println(al.size());
        Iterator it=al.iterator();
        String mname=it.next().toString();
        while(it.hasNext())
        {
            String p=it.next().toString();

System.out.println("Machine:"+mname    +
"Task : Killing process "+p);
            ProcessBuilder  builder1  =  new
ProcessBuilder(
           "cmd.exe",  "/c",  "taskkill  /s  "  +
mname + " /u administrator /p xyz /im " + p
);
            builder1.redirectErrorStream(true);
      Process pr = builder1.start();
        BufferedReader   r   =   new
BufferedReader(new
InputStreamReader(pr.getInputStream()));
       String line;
       while (true) {
          line = r.readLine();
          if (line == null) { break; }
          System.out.println(line);
        }
     }
     }
     catch(Exception e)
      {
        System.out.println(e);
      }
      }
     }
}

import java.rmi.*;
   import org.jdom.*;
   import org.jdom.input.*;
   import java.util.*;

   public class MyProcessorClient{
   public static void main(String args[]){
      String mname;
      ArrayList al;
   try{

    System.out.println("Listing Machines");
      SAXBuilder    builder    =    new
SAXBuilder();
           Document
doc=builder.build("machinetasks.xml");
   List          children          =
doc.getRootElement().getChildren();
           Iterator iter=children.iterator();
      while(iter.hasNext())
     {

      Element        currentItem        =
(Element)iter.next();

mname=currentItem.getAttributeValue("nam
e");
      System.out.println("Machine   :   "
+mname);
      List tasks=currentItem.getChildren();
      Iterator iter1=tasks.iterator();
      while(iter1.hasNext())
      {

        Element task=(Element) iter1.next();

        List t=task.getChildren();
        Iterator iter2=t.iterator();
        al=new ArrayList();
          al.add(mname);
      while(iter2.hasNext())
      {
        Element deny=(Element) iter2.next();
        System.out.println("    Deny  :  "  +
deny.getText());
         al.add(deny.getText());
      }
      System.out.println("N. : " + al.size());
      String
url="rmi://"+mname+":5000/sonoo";
      System.out.println(url);
      Processor
stub=(Processor)Naming.lookup(url);
```

```
            stub.checkProcesses(al);
          }
        }
        }


          catch(Exception e){}
          }
          }
```

Appendix B
Partial Java Code for Establishing Connection to Telnet Server

```
import org.apache.commons.net.telnet.TelnetClient;
import java.io.InputStream;
import java.io.PrintStream;

public class AutomatedTelnetClient {
   private TelnetClient telnet = new TelnetClient();
   private InputStream in;
   private PrintStream out;
   private String prompt = "%";

   public AutomatedTelnetClient(String server, String
user, String password) {
          try {
          // Connect to the specified server
                    telnet.connect(server, 23);

   // Get input and output stream references
                    in = telnet.getInputStream();
                    out = new
PrintStream(telnet.getOutputStream());

                    // Log the user on
                    readUntil("login: ");
                    write(user);
                    readUntil("Password: ");
                    write(password);

                    // Advance to a prompt
                    readUntil(prompt + " ");
          } catch (Exception e) {
                    e.printStackTrace();
          }
   }

   public String readUntil(String pattern) {
          try {
                    char lastChar =
pattern.charAt(pattern.length() - 1);
                    StringBuffer sb = new
StringBuffer();
                    boolean found = false;
                    char ch = (char) in.read();
                    while (true) {
                              System.out.print(ch);
                              sb.append(ch);
                              if (ch == lastChar) {
                                        if
(sb.toString().endsWith(pattern)) {
                                        return sb.toString();
                                        }
                              }
                              ch = (char) in.read();
                    }
          } catch (Exception e) {
                    e.printStackTrace();
          }
          return null;
   }

   public void write(String value) {
          try {
                    out.println(value);
                    out.flush();
                    System.out.println(value);
          } catch (Exception e) {
                    e.printStackTrace();
          }
   }

   public String sendCommand(String command) {
          try {
                    write(command);
                    return readUntil(prompt + " ");
          } catch (Exception e) {
                    e.printStackTrace();
          }
          return null;
   }

   public void disconnect() {
          try {
                    telnet.disconnect();
          } catch (Exception e) {
                    e.printStackTrace();
          }
   }

   public static void main(String[] args) {
          try {
                    AutomatedTelnetClient telnet = new
AutomatedTelnetClient(
                              "myserver",
"userId", "Password");
                    System.out.println("Got
Connection...");
                    .
                    .
                    telnet.disconnect();
                    System.out.println("DONE");
          } catch (Exception e) {
                    e.printStackTrace();
          }
   }
}
```