Rootkits Detection Using Inline Hooking

GRADUATE PROJECT REPORT

Submitted to the Faculty of
School of Engineering and Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, TX

in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

By,

Srikanth Padakanti
Spring 2012

Committee Members

Dr. Mario Garcia  ______________________________
Committee Chairperson

Dr. Dulal Kar  ______________________________
Committee Member

Dr. David Thomas  ______________________________
Committee Member
ABSTRACT

A rootkit is simply defined as a software or piece of code, which is designed to gain the root level access over a system by hiding its footprint from the operating system and the end user by using stealth techniques. There are many techniques and tools currently available in the market to detect whether a computer system has a rootkit or not. The research is mainly focused on the rootkits that perform inline hooking.

All the techniques and tools give results based on the signature format which is already in existence. When coming to new rootkits with new behavior or different heuristics, the tool may not detect them. For this reason, research is mainly focused on the behavioral characteristics of the rootkits. The whole research is divided into two phases: In the first phase, a wide variety of rootkits and their characteristics are studied and then in the second phase, by the type and kind of hooks a rootkit does, the rootkit family is detected. Out of many hooks, only inline hooking is considered in this project and analyzes the hook place and does some statistical analysis to make a decision to which family it belongs to.
# TABLE OF CONTENTS

Abstract .......................................................................................................................................... ii

Table of Contents ............................................................................................................................ iii

List of Figures ................................................................................................................................... v

List of Tables ..................................................................................................................................... viii

1 Introduction ...................................................................................................................................... 1

1.1 Rootkit ...................................................................................................................................... 3

1.2 Exploiting Vulnerabilities by Rootkits .................................................................................... 4

1.3 Attacking Procedure of Rootkits .............................................................................................. 5

1.3.1 Inline Hooking ....................................................................................................................... 6

1.4 History ..................................................................................................................................... 6

1.4.1 Sony Digital Rights Management (DRM) Rootkit ............................................................... 8

2 Narrative ...................................................................................................................................... 10

2.1 Problem Statement .................................................................................................................. 10

2.2 Proposed System ...................................................................................................................... 11

2.2.1 Phase I ................................................................................................................................. 11

a) PWS: Win32/Zbot.gen! AF ........................................................................................................ 12

b) TrojanDownloader: Win32/Vundo.HIZ .............................................................................. 13
c) TrojanDropper: Win32/Rovnix.A ........................................................................................ 14
d) Trojan: Win32/Alureon.CT ..................................................................................................... 14
e) Worm: Win32/Cridex.B ........................................................................................................... 15

f) Virus: Win32/Virut.BN ............................................................................................................. 16
g) Trojan: Win32/Remhead .......................................................................................................... 18
h) Trojan: Win32/EyeStye.N ................................................................. 19
i) VirTool: Win32/Obfuscator.C ......................................................... 20
j) Worm: Win32/Feebs.EA ............................................................... 21
k) Trojan: Win32/Vundo.OD ............................................................ 22
l) PWS: Win32/Zbot.gen! Y .............................................................. 23

2.2.2 Phase II ................................................................................. 23

3 Proposed System Design ................................................................ 25

3.1 System Architecture ................................................................... 25

3.2 Framework ................................................................................. 27

3.2.1 System Requirements ............................................................ 31

4 Implementation and Testing .......................................................... 32

4.1 Infecting Fresh Windows XP Virtual Machine with PWS: Win32/Zbot.gen! AF ... 32

5 Results .......................................................................................... 41

6 Conclusion ..................................................................................... 51

Bibliography and References .......................................................... 52

Appendix ........................................................................................... 55
LIST OF FIGURES

Figure 1 Windows Operating System Architecture [Sparks 2010] .................................................1
Figure 2 Windows Operating System Memory Protection Rings [Russinovich 2009] .............2
Figure 3 Inline Function Hooking Procedure ..................................................................................7
Figure 4 Two Rootkits shows the Union and Intersection Properties ........................................24
Figure 5 Modification of the First Five Bytes of the Code ..............................................................26
Figure 6 Abstract View of the Architecture .....................................................................................28
Figure 7 Architecture of RootkitAnalyzer Tool .............................................................................29
Figure 8 a) Windows XP SP 3 X86 VM Desktop .......................................................................33
Figure 8 b) VirtualBox Machine Settings ....................................................................................33
Figure 8 c) Selecting the Shared Folders Option ..........................................................................34
Figure 8 d) Browse the Rootkit Dropper Location ......................................................................34
Figure 8 e) Browse the RootkitAnalyzer Tool Location ...............................................................35
Figure 8 f) Enabling the Auto-Mount Option ...............................................................................35
Figure 8 g) Displaying My Computer Window .............................................................................36
Figure 8 h) Copying RootkitAnalyzer Tool on to Desktop ..........................................................36
Figure 8 i) RootkitAnalyzer Tool Execution on Fresh VM ............................................................37
Figure 8 j) Rootkit Dropper Copied to Desktop ..........................................................................37
Figure 8 k) Executing Rootkit Dropper .......................................................................................38
Figure 8 l) Rootkit Dropper Vanishes from Desktop ....................................................................38
Figure 8 m) Execution of Rootkit Analyzer Tool after Infection ................................................39
Figure 8 n) Creating Two Log Files onto Desktop .......................................................................39
Figure 8  o) Showing the Contents of Two Log Files..............................40
Figure 9  Hooks Generated by PWS: Win32/Zbot.gen! AF Rootkit........41
Figure 10 Hooks Generated by PWS: Win32/Zbot.gen! Y Rootkit...........42
Figure 11 Hooks Generated by TrojanDownloader: Win32/Vundo.HIZ Rootkit......42
Figure 12 Hooks Generated by TrojanDropper: Win32/Rovnix.A Rootkit.........43
Figure 13 Hooks Generated by Trojan: Win32/Alureon.CT Rootkit...............44
Figure 14 Hooks Generated by Trojan: Win32/EyeStye.N Rootkit...............44
Figure 15 Hooks Generated by Trojan: Win32/Remhead Rootkit................45
Figure 16 Hooks Generated by Trojan: Win32/Vundo.OD Rootkit................46
Figure 17 Hooks Generated by VirTool: Win32/Obfuscator.C Rootkit.............46
Figure 18 Hooks Generated by VirTool: Win32/Obfuscator.WT Rootkit...........47
Figure 19 Hooks Generated by Virus: Win32/Virut.BN Rootkit..................48
Figure 20 Hooks Generated by Worm: Win32/Cridex.B Rootkit..................48
Figure 21 Hooks Generated by Worm: Win32/Feeb.EA Rootkit...................49
Figure 22 Final Results after Executing all the Rootkit Samples..............50
Figure 23 a) VirtualBox General-Advanced Settings..............................56
Figure 23 b) VirtualBox General-Basic Settings..................................56
Figure 23 c) VirtualBox General-Description Settings...........................57
Figure 23 d) VirtualBox System-Acceleration Settings...........................57
Figure 23 e) VirtualBox System-Motherboard Settings............................58
Figure 23 f) VirtualBox System-Processor Settings...............................58
Figure 23 g) VirtualBox Display-Remote Display Settings.......................59
Figure 23 h) VirtualBox Display-Video Settings...................................59

vi
Figure 23 i) VirtualBox Audio Settings

Figure 23 j) VirtualBox Network Settings

Figure 23 k) VirtualBox Serial Ports Settings

Figure 23 l) VirtualBox USB Settings
LIST OF TABLES

Table 1 Rootkits with its Behavioral Properties ..........................................................11
1. INTRODUCTION

Windows Operating System is designed as a layered architecture. Figure 1 shows the Windows operating system architecture. It tells how the users and applications are separated from the hardware details by using different layers in the windows operating system. Layering approach is used to maintain high level of extensibility and portability. But at the same time, it creates many chances for an intruder to exploit the system. If an intruder has control over any one of the communication paths between the layers, the intruder can then perform many malicious activities like denial of service, injecting spams, keystroke logging etc. Rootkits mainly focus on communication paths in the architecture of windows operating system [Sparks 2010].

![Fig 1](image1.png)

*Figure 1 Windows Operating System Architecture [Sparks 2010]*
Intel 32 bit and 64 bit chips provide different levels of the memory protection. It is shown in the form of rings and they are numbered from 0 to 3. Among all the rings, ring 0 is considered as high privilege level where all the kernel processes live and ring 3 is low privilege level where user applications are executed. The device drivers are executed in ring 1 and 2 with I/O access permissions. The architecture is divided into rings because the process which is running in the lowest privileged level cannot overwrite or modify the process in the high privileged level. Ring 3 users are limited to use API which works as interface to the operating system kernel and Ring 0 users can interface directly with the hardware and memory [Russinovich 2009]. Intruders always targets to gain the ring 0 privileges by executing the rootkit as Kernel Mode Driver (KMD).

Figure 2 Windows Operating System Memory Protection Rings [Russinovich 2009]
According to survey made on January 2010, approximately 7% of all the reported malicious infections are due to Kernel Mode Rootkits (KMD). Figure 2 explains the memory rings in the windows operating system and its privilege levels.

1.1 Rootkits

Rootkits are a kind of malware similar to viruses, worms, spyware etc. Rootkits are mainly designed to hide its activities. Besides that, it hides other programs and resources like files, drivers, processes, folders, threads, ports, and registry keys which are associated to them. Rootkits can also be used for some intentional purpose, but any way it is totally a potential risk and depends on the users who are using it. Many times, rootkits are used for malicious activities like helping the intruders to introduce some virus or worms or some malware through backdoor. These are always used to compromise the target machine and have a remote control over that machine for illegal or illegitimate activities like knowing important passwords, credit card details, accessing confidential documents, some criminal issues etc. A rootkit installation in a system creates a backdoor for an intruder to exploit the vulnerabilities of that particular system.

The installation of rootkit itself introduces the primary flaw into a system. This makes the user to mistrust the integrity of the operating system and lose faith on the results provided by the operating system. Due to presence of rootkits inside a system, the user cannot distinguish whether the system is malware free or not. There is a saying that, “Before we go against our enemies, we should know about them”. In the same way, if the user need to go against rootkits, first user need to have a clear picture that what a rootkit does and what are its consequences when it is installed in the target machine. The most
challenging part to know about the rootkits is they hide and locate at some places and it cannot be found by doing normal scan or regular diagnosis. One should take several measures to know whether any rootkit is installed or not [Corregedor 2011]. Mostly, these measures are useful in the centralized systems, because each and every user have the shared access for the data and some resources.

1.2 Exploiting Vulnerabilities by Rootkits

The actions of the rootkits are invisible to most of the traditional scanners. It usually hides its actions by lying to the operating system or report some falsified results. So, user cannot identify whether the result produced by the operating system can be trustable or not. Each time, when a program or an application sends request, the general assumption would be that the data or the information provided by the operating system is genuine and is a valid response. This way rootkit violates the fundamental and basic trust of the system [Riley 2009].

When the rootkits get administrative privileges, it starts changing and hiding the vital resources or components, like ports, files, directories, folders, registry keys, and even the code of the operating system. One of the ways in which a rootkit achieves hiding capability is by hooking the system calls and modifying the information or data, returned by the system utilities. Traditional scanners cannot trace the footprints of rootkits. So, the tampering of the system will not be evidenced or revealed.

Rootkits are considered as Trojan injected into our machines. Rootkits are divided into two categories known as user mode rootkits and kernel mode rootkits. The user mode rootkits are also called as application level rootkits as these rootkits generally
replace the operating system services with the modified versions and hides itself from the user. User mode rootkits are very simple to detect because they are not capable to modify or change the kernel functions or services. Kernel mode rootkits are the powerful rootkits when compared to user mode rootkits because these rootkits implant their functionality directly on the kernel and works at the same level where the operating system works i.e., in ring 0. They use the custom kernel module to modify the system function call of the kernel to hide files and processes [Jose 2007]. By introducing the kernel mode rootkits into the system, intruder can gain the root level access and normal user cannot find its trace even through the normal scanners. User should take enough measures to prevent or protect the system from rootkits.

1.3 Attacking Procedure of Rootkits

The attacking procedure of rootkits is divided into four stages.

i) **Get the information of the target:** The first stage of attacking is to get the information of the target. For that, intruder will scan the system completely and know the back doors and vulnerabilities of the system.

ii) **Gain Root level access:** Immediately after knowing the vulnerabilities and backdoors of the compromised system, intruder gets only temporary access to the target system and get the root level access.

iii) **Installation of rootkits:** After gaining the root level access, intruders installs the rootkits into the users’ machine and it hides its execution from the user and operating system.

iv) **Control over compromised system:** Once the intruder finishes all the
three steps, the next and final step is to control the compromised machine and try to hide itself from the user.

Rootkit is not a self propagating code. It requires typically three snippets of code known as dropper, loader and rootkit. Dropper code is responsible to downloading the rootkit and installing it. Loader code is responsible to load the code into the main memory and deleting itself. It causes a buffer overflow which in default, loads the rootkit into the memory. The last code is rootkit, which works on the host machine and hides from the user [Rowe 2003].

1.3.1 Inline Hooking

Inline function hooking is considered as an easy way to hook because it doesn’t need any drivers or other software to do so. It just overwrites the first five bytes of the code in the Application Programming Interface function with trampoline code. In that, first byte is replaced with JUMP instruction (E9 is its OPCODE) and remaining four bytes of the code is replaced with 32-bit address of the malicious code called trampoline. Figure 3 explains how the inline function hooking is done when the program calls another function and it shows how the code is executed [Lobo 2010].

1.4 History

The very first documented virus to target the personal computer platform is discovered in 1986 and it uses some different technique called cloaking technique to hide itself. This virus tries to read the boot sector of the system and redirect it to other memory location in the disk. Another virus called DOS virus uses the same cloaking techniques and these are evolved as a very complicated and sophisticated.
Rootkit got its name from the UNIX like operating systems, where the users with full administrative privileges are called as *root* users. Root is the user who has all the administrative privileges who can install any software or uninstall any software or can add a user or disable a user etc. Set of tools collectively termed as kit. By this way, it got the name rootkit. Suppose that an intruder replaced or modified the administrative tools on the compromised system with a rootkit, then that intruder can get the root access over that compromised system. The initial rootkits are obvious to detect by using some set of

**Figure 3 Inline Function Hooking Procedure**

[Diagram showing inline function hooking procedure]
tools like tripwire. Mr. Davis and Mr. Dake coded the earliest rootkit in the year 1990 for Sun Microsystems and SunOS companies and it is for UNIX operating system.

A famous scientist compromised the C compiler in the UNIX environment and exploited the lecture in 1983 when he received the Turing award. The altered compiler will produce different results when compared to the normal compiler. The ‘ls’ command in UNIX lists all the files in that particular directory but, rootkit alters the meaning of that command by displaying password file or showing hidden files etc. In the year 1999, NTRootkit is created for the first time for Windows NT environment and is followed by Hacker Defender. The currently existing rootkits are stealthy and there exist many tools and techniques to detect the presence of rootkits. Some techniques followed for detecting rootkits are behavior based detection, integrity monitoring, network based detection, signature based detection, and evolutionary based detection [Ring 2004].

1.4.1 Sony Digital Rights Management(DRM) Rootkit:

The most popular and famous windows platform rootkit is Sony Digital Rights Management (DRM) rootkit. The Sony BMG company released this rootkits with the audio disks in the year 2005. The audio disks include Extended Copy Protection (XCP) and MediaMax software on it. The former software was installed on fifty two titles and the latter software was installed on fifty titles. When the user tries to insert the disk and plays it, the next moment rootkit get installed into the system and the user who is using that disk has no clue of what has happened. The major intention of introducing this rootkit into the system is to take the copy protection measures. This created a very big vulnerability for those machines because other malicious software can exploit the system.
After this issue, Sony BMG released a software component to remove the rootkit from the affected systems. But, this software was studied by a scientist called Russinovich in his blog and posted the truth in the blog. The removal program will just unmask the hidden files which are installed by rootkit but, it actually did not remove the rootkit. But, after facing many issues Sony BMG released an improved and new version of the removal tool to remove that rootkit [Ring 2004].
2. NARRATIVE

The scope of the project is explained in this section. First phase is to collect the different samples of rootkits; then study their characteristics, behavior, rule set, and complexity. After that, using some statistical analysis, depending on the place of the hook, the rootkit family is analyzed and the result is saved onto a .csv file to have a saved documentation.

2.1 Problem Statement

Rootkits hide its presence and when it does so, it is very hard to detect them. The information got from the compromised system is used for selling the information to websites and also for criminal issues etc. When it comes to stand alone systems, it is very easy to detect and delete the rootkit. But, when the same concept is extended to network of computers then it is going to be a big challenge to security department. The very worrying feature of the rootkit is that it hides itself while it executes and it causes a big threat to one of the information security property called confidentiality. It is very clear from a survey that most of the malicious software are developed for the personal gains of intruders and out of all, 39% of them make use of rootkits to hide their malicious code execution on the compromised systems [Lobo 2010].

In recent years the complexity of rootkits are increased and in the same speed, the anti rootkit tools are being developed. So, every day a new rootkit is being evolved and it is very hard to detect each and every rootkit by all the tools. So, one of the possible solutions is to note down the heuristics of the rootkits and make a similar database to distinguish the new rootkits and classify their family.
2.2 Proposed System

The proposed system is purely based on the behavioral characteristics of the rootkits in the windows environment. By using these characteristic features, the proposed tool will find whether the system is affected with a rootkit via inline hooking or not. Then the output is send to a log for documentation purpose. After finding the hooking and the hook place, the tool will find the rootkit family. Knowing the characteristics of each and every rootkit is tedious task and it is almost impossible because those are not standard. So, to narrow the research, only the rootkits with inline hooking are considered. This is achieved in two phases.

2.2.1 Phase I

In the first phase, a wide variety of rootkit samples are studied, and its characteristics are used in the second phase to find whether the compromised system has a rootkit with inline hooking or not. Table 1 explains that n rootkits are examined and its behavioral properties are saved for the future purpose. The characteristics of some famous open source rootkits are shown below. These rootkits are found with the help of www.virustotal.com website.

<table>
<thead>
<tr>
<th>Rootkit</th>
<th>Process Name</th>
<th>Module Name</th>
<th>Function Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>[Empty]</td>
<td>[Empty]</td>
<td>[Empty]</td>
</tr>
<tr>
<td>R2</td>
<td>[Empty]</td>
<td>[Empty]</td>
<td>[Empty]</td>
</tr>
</tbody>
</table>
a) **PWS: Win32/Zbot.gen! AF:**

This is a password stealer and it makes use of rootkit to steal the passwords. It has many aliases like PWS-Zbot.gen.in (McAfee), Trojan.PES.Panda.818 (Dr. Web), Trojan.SpyEYE (Symantec) etc. The alert level is considered as severe. Win32/Zbot has backdoor functionality which allows an unauthorized access and control of a compromised system. This rootkit loads its code and installs it into all other computers when connected using Remote Desktop Services (RDS).

We can identify the symptoms by observing the system behavior. But, with this rootkit there are no common symptoms. Alert notifications from antivirus are considered as one of the symptoms. This rootkit has a unique installation procedure. When Win32/Zbot.gen! AF loads into the system, it copies into the system with a randomly named file in the following format

```
%APPDATA%\<random name>\<random name>.exe
```

For ex: C:\Documents and Settings\Admin\App Data\sygnr\geouft.exe

This Trojan modifies the registry in a way it starts when the windows starts booting. It injects its code into the address table of all the processes which are running and have some security privileges. If not, it directly injects its code into all user level
applications like Iexplorer.exe, Explorer.exe, Mozilla.exe, etc. The code injection is usually done to hide it from the security services. This rootkit hooks the following application programs to hack the sensitive data from the compromised system. It reduces the Internet Explorer, Firefox, security by modifying the registry files in the System32 folder. It captures the most important and sensitive information such as Cache passwords, digital certificates, Internet Explorer cookies [Encyclopedia entry 2011].

b) TrojanDownloader: Win32/Vundo.HIZ:

This Rootkit is a Trojan downloader and it accesses the remote websites to download and install unwanted and malicious software. Some rootkits focus on some of the specific files on the remote website; and other rootkits focus on the specific URL of some websites which potentially downloads and installs the malicious software. This specific threat caused by the Win32/Vundo.HIZ rootkit is detected by the Microsoft antivirus engine. Win32/Vundo is a family with multiple components. It delivers the ‘out of context’ pop-up advertisements. Apart from the pop up advertisements, it also downloads and installs some arbitrary files. It is distributed in compromised system as a DLL file and installs as a Browser Helper Object (BHO) without user’s awareness. This Vundo family uses advanced stealth techniques and uses advanced techniques to hide itself from the regular scanners.

Some artifacts of Win32/Vundo spread via network drives. And each variant that belongs to these family compromise systems in different ways and uses some unique methods to install the malicious software. Vundo used a dropper component as Win32/Vundo.A or Win32/Vundo.B or Win32/Vundo [Encyclopedia entry 2011].
c) **TrojanDropper: Win32/Rovnix.A:**

Win32/Rovnix.A is a Trojan Dropper. It has many aliases such as Win32/Lageliz.B, Win32/PSW.Papras.BZ trojan, Trojan.Win32.Agent.nkwx, Trojan.Cidox, TROJ_VUND-O.SMC and so on. The alert level is considered as severe. Win32/Rovnix.A is a Trojan dropper, which modifies the New Technology File System boot sector to run the other malicious software. This Trojan also installs another variant which is used to restart the computer, so that the rootkit starts its execution after restarting the windows.

It follows a unique style in installing the malicious code to the system’s local hard disk and it also changes the New Technology File System (NTFS) boot sector to run the installed code. This rootkit is detected in different ways based on the operating system. On windows 32 bit operating system this malicious code is found by name VirTool: Win32/Rovnix.A but, same code when found on the windows 64 bit operating system it is detected as VirTool: Win64/Rovnix.A. This Trojan dropper installs a piece of code called as Trojan: Win32/Rovnix.A and reboots the machine to change or modify the file system during the booting process and runs this malicious code after windows restart [Encyclopedia entry 2011].

d) **Trojan: Win32/Alureon.CT:**

Trojan: Win32/Alureon.CT is a trojan which works as obeying servant. It sends the information of the compromised system to remote address. It belongs to Win32/Alureon family. This rootkit has different aliases like Win32/Tidserv.H, DNSCharget!bc, Trojan.TDSS.HWD, W32/DNSCharger.FUBV, etc. This
Win32/Alureon family is considered as data stealing Trojan. Alureon Trojan is used to sense the incoming and outgoing data packets in order to get the confidential information, like credit card details, passwords, sensitive data etc from the compromised system. Besides sensing the data, this trojan can inject malware or malicious data in to the compromised system. All these things can happen only if the security settings are changed. So, here the Domain Name Service (DNS) settings are changed by the intruder to enter into the network and compromise the system. It may infect or modify the device drivers to make them unusable.

The Alureon family rootkit has different symptoms based on the variant of the rootkit. For this variant Win32/Alureon.CT, the keyboard may be disabled or drivers are infected and user will see a pop up message that requests for the activation or the reinstallation of the drivers. This Alureon has different variants and each variant is used for different activities like changing the search results of the user produced by the server, redirecting the affective users browsing site to other malicious sites called phishing sites, changing the DNS settings to redirect the users to different websites without the knowledge of the user, etc. Win32/Alureon uses most complex stealth mechanisms to hide its detection from various scanners and other software. This rootkit is usually dropped and installed in `<system folder>\spool\PRTPROCS\W32X86\<random name or number>.tmp and code is introduced into spoolsv.exe file [Encyclopedia entry 2011].

e) **Worm: Win32/Cridex.B:**

Worm: Win32/Cridex.B is a worm and it is a self propagating program which usually spreads from one machine to another machine. Worms use different strategies and variety techniques to spread from one machine to another machine. The most
traditional way used by worms are, they copy themselves to the local hard drives or the network drives or explore the vulnerabilities of the systems which are going to be compromised. Sometimes worms spread through spam emails such that, user will get an email with some attachment and fake email description. If user downloads it or clicks it, then the worm installs its code on to the local machine hard drive and spreads itself to all the machines in the network. The messages sent by the intruders don’t look like spam but, it is embedded with some data which looks like useful information to user but in real it is a fake data.

There are some files in the system by which a user can understand that a rootkit is present in the system. This is one of the symptoms. If this rootkits is installed into the machine following files are shown in the local machine.

C:\Documents and Settings\Admin\App Data\mslivesvc.exe
C:\Documents and Settings\Admin\Local Settings\Temp\ 

After adding those files to local machine, it adds a sub key to the existing machine HKCU\Software\Microsoft\Windows\CurrentVersion\Run.

This rootkit makes use of injection technique inorder to hide itself from scanners and other software from removal. When Worm: Win32/Cridex.B runs, it starts to inject its code into some of the running processes like rfwmain.exe, cmd.exe, explorer.exe. So, before searching for this kind of rootkit, user should make sure that there is no hooking in these processes [Encyclopedia entry 2011].

f) **Virus: Win32/Virut.BN:**

Virus: Win32/Virut.BN is also known as polymorphic file detector which always targets .scr and .exe files. It opens a loop hole for intruders by connecting to the IRC
server. This creates a way for intruder or a remote attacker to download and run some arbitrary files on the compromised system. There are no specific symptoms with this rootkit. The notifications produced by the scanners or antivirus engines will be the general symptoms. This rootkit mainly targets the file protection ability. It disables the windows system file protection by embedding the code into a running process winlogon.exe. The code injection patches the sfc_os.dll file into the winlogon.exe in the memory to infect the resources like files, folders, processes, etc.

This rootkit targets to infect the .scr and .exe files upon execution. Therefore opening files with the explorer or sharing the files in the network will automatically result in corruption of the files and spreading the same virus from one system to another system. This rootkit introduces its piece of code into the explorer.exe or winlogon.exe to hook the low level NTDL windows application program interface and this is done in order to stay in the memory. It hooks some functions like createFile, openFile, createProcess, and createProcessEx in every running process NTDLL.dll. So, whenever user calls these processes, the handler will automatically redirects its execution to the virus which is handled by the intruder and this is possible only through the backdoor created by the rootkit.

This rootkits infects the HTML files, such that it injects the code into HTML files which indirectly points to an unknown domain zief.pl by adding a hidden IFrame. So, when the infected HTML file is opened in a browser it immediately redirects to the zief.pl server without user awareness. After affecting all the files, this rootkit modifies the local system’s host file such that the remote machines will run the copy of the virus.
These modified or changed HTML files are detected as Virus:HTML/Virut.BH. It connects to IRC (Internet Relay Chat) server called as irc.zief.pl using port 80 and if this connection fails or if it encounters any problems, it searches for alternative server called proxim.ircgalaxy.pl via port 80. Even this rootkit has aliases like W32/Virut.AIGeneric, W32/Virut.Gen, Win32/Virut.17408, and Win32/Virut.NBP. The alert level for this rootkit is considered as severe [Encyclopedia entry 2011].

g) Trojan: Win32/Remhead:

Trojan: Win32/Remhead is a complex and dangerous trojan which is capable of doing some disasters in the compromised systems. This rootkit is mainly used when intruder wants to do some online crime by gaining the administrative control and access onto a particular machine. This rootkit dramatically degrades the performance of the compromised system and reduces the speed of the machine to one half of the original speed. This trojan tries to change the normal behavior of the system by creating some unnecessary files and deleting the existing files. The effect of the Trojan: Win32/Remhead is very complex and dangerous. So, a user must take some efficient and effective measures to eliminate this trojan completely from the compromised machine. To be specific, there are no particular symptoms to notice the threat caused by the Trojan: Win32/Remhead and the alert notifications generated by the antivirus engines will be the common symptoms.

These rootkits have a certain behavior which is common to one or more malicious software. It runs the applications without the user awareness. It modifies the performance of the system by patching the running processes into the memory. It modifies the boot sector files such that for every restart, this rootkit automatically starts
running after windows logon. It always tries to download some malicious content from the remote websites. The alert level is considered as severe for this rootkit. Minimum precautions should be planned by the user to detect these rootkits in the system like enabling the firewall, download the latest updates for the installed software, use most recently released version of antivirus, avoid downloading the content from insecure websites and cracked software, use strong and unique password patterns [Encyclopedia entry 2011].

**h) Trojan: Win32/EyeStye.N:**

Trojan: Win32/EyeStye.N is a Trojan which records the keystrokes from keyboard, monitors the internet traffic, and steals certain credentials which belongs to user or administrator. After that, transfers the captured data to intruder for personal gain. After installing this rootkit, many unknown files such as c:\recycle.bin\recycle.bin.exe, c:\rcss.bin\rcss.bin.exe, c:\po000oasi\<random characters>.exe, c:\montes\montes.exe, and c:\system.bin\config.bin etc. are found in the compromised machines. Trojan: Win32/EyeStye.N is installed with the help of other malicious software like TrojanDropper: Win32/EyeStye, TrojanDownloader: Win32/Waledac and Backdoor:- Win32/Kelihos. When this rootkit starts its execution, it creates a mutex which is unique and ensures that at any given time; only one instance of the Trojan will be executed. Some sample mutexes are SystemBoot, SystemSrv, Global\LocksNA, and Global\Systemmo. The default storage location for this rootkit is C:\Winnt\system32 for windows NT and C:\Windows\System32 for Windows XP service pack 3, Windows Vista, and Windows 7.
Similar to other rootkits, it also uses code injection technique to hide itself from the user or administrator. This rootkit injects its malicious code into running processes like DRWEB32.exe, svchost.exe, wmiprsve.exe, and winlogon.exe. This trojan installs a user mode rootkit which hooks some low level application programming interfaces like NtQueryDirFile, NtEnumValKey, NtSetInfoFile, and NtVdmCtrl to hinder its malicious and corrupted processes, directories, registry data and files. In addition to this hooking, it may download some malicious content from the remote websites and reduce the browser security [Encyclopedia entry 2011].

i) VirTool: Win32/Obfuscator.C:

VirTool: Win32/Obfuscator is also known as trojan horse because it exploits the security vulnerabilities and execute malicious files as legitimate files in the compromised system. The VirTool: Win32/Obfuscator.C trojan is used for different purposes like monitoring the local system characteristics; collecting the confidential information like passwords, credit card details etc. from key strokes; modifying the web browser settings; downloading and executing harmful malware; deleting necessary files and installing unwanted files; and creating security loop holes or back doors. There are many trojans in present cyber generation like backdoor trojans, password stealer trojans, downloader trojans, dropper trojans, dialer trojans, and rootkit trojans.

VirTool: Win32/Obfuscator rootkit is mainly used for hiding its activity from user, antivirus scanners, and other software. These rootkits generally employ hybrid methods which include compression, encryption, anti debugging and anti emulation methods. This obfuscation rootkit is used on different kinds of malicious software. The alert levels for this rootkit are considered as severe. This rootkit belongs to VirTool:

j) Worm: Win32/Feebs.EA:

This rootkit belongs to Worm: Win32/Feebs family. This rootkit is one of the variant of Internet Worms. The worms from this family spreads as an attachment to some spam emails or malicious messages and also spreads through sharing of documents, files and resources in a particular network. These worms are very dangerous and its alert level is considered as severe. The worms from Feebs family have the capability to turn off the antivirus and also firewall of a system. Another variant of this family is Worm: Win32/Feebs.gen and it is used to detect the scripting part of the webpage called as JavaScript because this code is responsible to spread the infective emails with the attachments. This piece of code downloads the worm from the respective servers, and it stores on to the compromised system and immediately starts its execution.

The advanced mechanism of the above mentioned mechanism is termed as phishing. Phishing means to redirect the legitimate user to a fake website, which normally looks like a genuine page and user cannot differentiate on the first look. So, when user enters the credentials, the information is directly passed to the intruder in
remote location. This scripting code is responsible to delete some files in the system registry like RapDrv, PciPim, FirePM, and KmxFile [Encyclopedia entry 2011].

k) Trojan: Win32/Vundo.OD:

Trojan: Win32/Vundo.OD is a Trojan that belongs to Trojan: Win32/Vundo family. This Trojan usually sends a pop up advertisements on the installed machine and it also has the capability to drop some files onto the victim machine and these files are responsible to run and install some malicious software on the compromised machine. This Trojan generally drops a dll file into the compromised machine and it is saved using a random name. The dropped dll file is detected as TrojanDownloader: Win32/Vundo.HIY. If the security services of the compromised machine is greater and if it won’t allow the Trojan to drop the above mentioned dll component, then, it copies itself as flash player update in the programming files folder with administrative privileges. Similar to other trojans, it starts its execution parallel to windows boot up.

This trojan creates some additional registry entries like HKLM\Microsoft\Software\Windows\CurrVer\Win and sets it value to ApplicationInit.dll and with data on to randomly named folder denoted as <random folder>/<dll file>. Here, random folder refers to any variable location which is chosen by the malicious software by querying the operating system. The default installation folder would be System32 folder. This Trojan always drops a file which is detected as TrojanDownloader: Win32/Vundo.OD and it is saved in <startup folder>/Microsoft_update.exe. Startup folder changes as we change the OS version. It is different for Win NT, Win XP, Win Vista, and Win 7 etc [Encyclopedia entry 2011].
1) **PWS: Win32/Zbot.gen! Y:**

This rootkit belongs to password stealer category and it is usually accessed remotely. This Trojan is installed with the help of other malicious software. The other variant of this rootkit is Exploit: Win32/Cplnk.B and it is transferred to the victim machine as an attachment via spam emails. For example, consider an email received from DHL courier service saying that a shipment is received and the attached file contains a tracking number and shipment product information. But, in real that attachment consists of hidden rootkit and is installed as soon as it is opened. When this rootkit is executed, it drops a copy with the following name \%APPDATA\%\<random name>\<random letters>.exe. Once the rootkit is installed, the rootkit is stored in one of the following location <drive:><documents and settings\default user or <drive:><users\default or <drive:><users><user name>.

PWS: Win32/Zbot.gen! Y rootkit injects its code into some running processes like explorer.exe, taskhost.exe, tasking.exe, and wscntfy.exe. For example, if the user has administrative privileges and if the user runs the Trojan then it injects its code into admin level processes like winlogon.exe. If this has not happened, it injects its code into the processes which runs in level 3 (upper layer) processes like explorer.exe, smss.exe etc [Encyclopedia entry 2011].

### 2.2.2 Phase II

The identification of the rootkits is done using some statistical analysis and the place of the hooks it made. Before that, some knowledge of set theory is required to
classify the rootkits into its categories. This is done by analyzing the inline hooks made by the rootkits. Before analyzing, some basic properties of set theory must be known.

Let A and B be two sets and the total elements of A and B is calculated by union operation and it is denoted as AUB. The common elements of A and B is calculated by intersection operation and it is denoted by A∩B. These operations can be explained by using Venn diagrams. In Figure 4 the shaded part is represented as intersection of f1 and f2. The whole f2 is represented as union of f1 and f2 because f1 is subset of f2.

Figure 4 Two Rootkits shows the Union and Intersection Properties

After collecting the characteristic features and working environment of the rootkit samples, the tool is developed such that it should detect whether the system has any inline hooks in its running processes.
3. PROPOSED SYSTEM DESIGN

3.1 System Architecture

Inline hooking is widely used by the intruders because of several features

- By using inline hooking, almost any function is modified.
- No need to install a special driver to affect the system. It only need administrative privileges (It is for ring 3)
- This inline hooking works in both ring 0 and ring 3.
- It is very easy to run and no special skills are required.

Therefore detection of inline hooking is important in system analysis. RootkitAnalyzer tool is designed to detect this kind of anomalies. Starting from Microsoft Windows XP Service Pack 2 almost every windows application function begins with the same piece of code as shown below

```assembly
mov edi,edi
push ebp
mov ebp, esp
```

For Example, Figure 5. shows the affect of inline hooking. The first five bytes of the program is modified such that first byte is replaced with the JUMP instruction and its_OPCODE is E9. The next four bytes is replaced with the modified address (which redirects the process control to trampoline).
Figure 5 Modifications of the First Five Bytes of the Code

It allows hot-patching without the need to reboot the machine and it allows the malicious users to impact the machine. The main task of the RootkitAnalyzer tool is to analyze all the WinAPI functions. It is done in several steps:

1. RootkitAnalyzer enables `SeDebugPrivilege` (adding debugging privileges) in order to access the remote processes and it is performed by using `SetPrivilegeProcess` function.

2. RootkitAnalyzer creates process snapshot with `CreateToolhelp32Snapshot` function. For each process in the snapshot function `AnalyzeProcess` is called.

3. Internally `AnalyzeProcess` function creates the snapshot of the modules (with the help of `CreateToolhelp32Snapshot` function) and calls `AnalyzeModule` routine for each and every process.

4. `AnalyzeModule` function loads corresponding .dll files from the disk (functions `CreateFile/ReadFile`), then it starts iteration through module export table and compares the beginning of each and every functions in memory and on disk. If the beginning of the function is different and first byte is E9 (OPCODE for the absolute JUMP instruction), then the hook is logged with `LogHook` function.

5. `LogHook` function creates two comma separated log files (.csv).
First file is in the following format:

\textbf{ProcessName, ModuleName, FunctionName}

Second file contains unique ModuleName and FunctionName pairs for further analysis. Based on the second log file, a unique rootkit stamp is created and it is identified through further modifications.

Figure 6 explains the abstract view of the architecture where input is the windows XP operating system with clean installation. Using any one of the dropper from the available samples, run the rootkit on that fresh XP machine, then, execute the RootkitAnalyzer tool. Now, the tool takes a snapshot of each and every running process and finds whether any inline hooking is made or not. The snapshot is copied to log file for document purpose and the ProcessName, ModuleName, and FunctionName of the hooked process is copied to another log file.

\textbf{3.2 Framework}

Based on second log, unique rootkit stamp is created. And further modifications are identified based on those values that are populated in the log file. The important contribution in this tool is to manually analyze the Export Table (PE) format for finding the hooks. As shown in Figure 6, first clean windows XP is installed in Virtual Box and rootkit droppers are shared into the Virtual Machine (VM). All rootkits are installed in separate VM’s and dropper is executed on each of them. Now, the dropper infects the system with given rootkit and after reboot, the rootkit starts its execution.
The framework for RootkitAnalyzer consists of six modules. They are:

1. Process Enumeration Module
2. DLL Enumeration Module
3. Debug Privilege Enable Module
4. Export Function Enumeration Module
5. Function Code Compare Module
6. Log Module

Each module has some unique functionality. Figure 7 explains how the communication is done between all the modules.
1. Process Enumeration Module

Process Enumeration Module is a simple module needed to create a snapshot of all running processes to find hooks in them. It uses `CreateToolhelp32Snapshot` function to create a process snapshot and pair of `Process32First/Process32Next` functions to walk through the process snapshot.

2. DLL Enumeration Module

DLL Enumeration Module is similar to process enumeration module, needed to create a DLL snapshot in each running process. It also uses the same function `CreateToolhelp32Snapshot` function as Process Enumeration Module and a pair of `Module32First/Module32Next` function to walk through the dll snapshot.
3. **Debug Privilege Module**

Debug Privilege Module is an auxiliary module and is not directly involved in the hook detection but is needed to access remote process virtual memory. Other modules will not work without it. It opens current process token with `OpenProcessToken` function and it is used to perform privilege lookup with `LookupPrivilegeValue` function, which sets debug privilege in token and adjusts process token with `AdjustTokenPrivileges` function.

4. **Export Enumeration Module**

Export Enumeration Module is intend to enumerate all of dll exported functions. It checks MZ-PE header, locates export table and walks through the export table.

The output of this module is function name and its address (relative offset).

5. **Log Module**

Log Module stores all found hooks in 2 log files in csv format. In the first one `ProcessName`, `ModuleName`, `FunctionName` are stored; and the second one contains only unique `ModuleName`, `FunctionName` pairs for further analysis. This module uses c standard functions to access the files and some standard library functionality to exclude repeated hooks.

6. **Function Code Compare Module**
Function Compare Module loads corresponding dll files from disk and compares function code in memory and on disk. If any discrepancy is found, then, the log module is called.

3.2.1 System Requirements:

The basic system requirements that are required to be met in developing this project are:

- **Hardware requirements:**
  - SYSTEM: Pentium IV Dual core
  - HARD DISK: 40 GB +
  - MONITOR: 15 VGA color
  - RAM: 1 GB +

- **Software requirements:**
  - Operating system: Windows XP Service Pack 3
  - Coding Language: CPP Language
  - Virtual Machine: Virtual Box
  - Rootkit Samples: Open Source Rootkits
4. IMPLEMENTATION AND TESTING

VirtualBox settings are followed according to Figure 8. Now, open VirtualBox application and create a new Windows XP VM and share the RootkitAnalyzer tool and the rootkit dropper from the physical hard drive to VM. It is a clean and fresh installation of Windows XP. So, the operating system will perform as mentioned in the manufacturer manual. Next step is to execute the shared rootkit dropper in the VM (double click the .exe file) and reboot the machine (not mandatory). The available VM is infected with rootkit and it is no more genuine. But, in reality these rootkits (executable files) are attached to an attachment in the email or downloaded through browser without the knowledge of the user etc.

After infecting the fresh VM, open the command prompt in the VM and change the present directory to the place where RootkitAnalyzer is saved. Type

*C:/Administrator and Settings/Desktop>*RootkitAnalyzer.exe*

RootkitAnalyzer tool is used to find the hooks and categorize the rootkits into families. So, user can detect whether any hooks are made or not by executing the RootkitAnalyzer tool which generates one text file and two Comma Separated Values files. Executed result is saved into one log file (.txt) and the ProcessName, ModuleName, FunctionName is saved in another log file (.csv). But, the ModuleName and FunctionName is saved in another log file (.csv) for future purpose (to detect the family). The entire procedure is shown in the Figure 8.

4.1 Infecting Fresh Windows XP Virtual Machine with PWS: Win32\Zbot.gen! AF:

VirtualBox is loaded with Windows XP Service Pack 3 X86 VM and boot up the system. The home screen of the VM is shown in Figure 8 a).
Next, click on the VirtualBox Machine tab and go to settings tab. It is shown in Figure 8 b).
After clicking the Settings tab go to Shared Folders option. It is shown in Figure 8 c).

![Figure 8 c) Selecting the Shared Folders Option](image1)

Browse the path for the storage location of the rootkit dropper. It is shown in Figure 8 d).

![Figure 8 d) Browse the Rootkit Dropper Location](image2)
Next step is to share the RootkitAnalyzer tool. It is shown in the Figure 9 e).

**Figure 8 e) Browse the RootkitAnalyzer Tool Location**

Auto-mount option is selected and the folder is shared. It is shown in Figure 8 f).

**Figure 8 f) Enabling the Auto-Mount Option**
Now restart the VM and open My computer. This is shown in Figure 8 g).

**Figure 8 g) Displaying My Computer Window**

Copy the RootkitAnalyzer tool from shared folder to desktop. It is shown in Figure 8 h).

**Figure 8 h) Copying RootkitAnalyzer Tool on to Desktop**
Now open command prompt and execute the RootkitAnalyzer tool on the fresh and clean Windows XP. It executes perfectly and no hooks are found. It is shown in the Figure 8 i).

Figure 8 i) RootkitAnalyzer Tool Execution on Fresh VM

For safety purpose, rootkit dropper is saved with .ex# extension. Because .exe file is executed by double clicking it. So, there are many chances that user accidentally clicks it. It is shown in Figure 8 j).

Figure 8 j) Rootkit Dropper Copied to Desktop
Execute the rootkit dropper by just double clicking it or right click on the icon and click open. It is shown in Figure 8 k).

Figure 8 k) Executing Rootkit Dropper

Now, dropper injects its code and .exe file is vanished. It is shown in Figure 8 l).

Figure 8 l) Rootkit Dropper Vanishes from Desktop
Executing the RootkitAnalyzer tool after the machine is infected with the rootkit dropper. It is shown in Figure 8 m).

![Figure 8 m) Execution of Rootkit Analyzer Tool after Infection](image)

After finishing the execution of RootkitAnalyzer tool. It creates two .csv files on the desktop. It is shown in Figure 8 n).

![Figure 8 n) Creating Two Log Files onto Desktop](image)
One of the log file contains the ModuleName, MethodName and FunctionName. Another log file contains MethodName and FunctionName. It is shown in Figure 8 o). Similar procedure is followed for remaining rootkits and results are saved into log files.

Figure 8 o) Showing the Contents of Two Log Files
5. RESULTS

a) PWS: Win32/Zbot.gen! AF

RootkitAnalyzer tool is executed on the Windows XP machine with PWSWin32Zbot.gen! AF rootkit and the hooks generated are sent to log.csv file. It is shown in Figure 9.

Figure 9 Hooks Generated by PWS: Win32/Zbot.gen! AF Rootkit

b) PWS: Win32/Zbot.gen! Y

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by PWS: Win32/Zbot.gen! Y rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 10.
RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanDownloaderWin32Vundo.HIZ rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 11.
d) TrojanDropper: Win32/Rovnix.A

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanDropperWin32Rovnix.A rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 12.

![Figure 12 Hooks Generated by TrojanDropper: Win32/Rovnix.A Rootkit](image)

e) Trojan: Win32/Alureon.CT

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanWin32Alureon.CT rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 13.
f) Trojan: Win32/EyeStye.N

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanWin32EyeStye.N rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 14.
g) **Trojan: Win32/Remhead**

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanWin32Remhead rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 15.

![Figure 15 Hooks Generated by Trojan: Win32/Remhead Rootkit](image)

h) **Trojan: Win32/Vundo.OD**

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by TrojanWin32Vundo.OD rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 16.
i) VirTool: Win32/Obfuscator.C

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by VirToolWin32Obfuscator.C rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 17.
j) **VirTool: Win32/Obfuscator.WT**

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by VirToolWin32Obfuscator.WT rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 18.

![Figure 18 Hooks Generated by VirToolWin32Obfuscator.WT Rootkit](image)

k) **Virus: Win32/Virut.BN**

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by VirusWin32Virut.BN rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 19.
Figure 19 Hooks Generated by Virus: Win32/Virut.BN Rootkit

1) Worm: Win32/Cridex.B

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by WormWin32Cridex.B rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 20.

Figure 20 Hooks Generated by Worm: Win32/Cridex.B Rootkit
m) Worm: Win32/Feebs.EA

RootkitAnalyzer tool is executed on the Windows XP machine, which is infected by WormWin32Feebs.EA rootkit. The hooks generated are sent to log.csv file and it is shown in Figure 21.

![Figure 21 Hooks Generated by Worm: Win32/Feebs.EA Rootkit](image)

After running the RootkitAnalyzer tool with all the droppers. Cumulatively one file (.csv) is created with ‘+’ and ‘-‘. Here ‘+’ indicates that function is hooked and ‘-‘ indicates that function is not hooked. It is shown in Figure 22.
Figure 22 Final Results after Executing all the Rootkit Samples
6. CONCLUSION

The project delivers information of some windows internals, such as hot-patching mechanism and inline hooking, demonstrates how it can be used in malicious applications and describes the methods how to defeat this kind of vulnerability. Based on this knowledge, proposed and tested special tool named RootkitAnalyzer. It consist of 6 different independent modules (process enumeration module, dll enumeration module, debug privilege enable module, export function enumeration module, function code compare module and log module) which allows to detect an active infection.

In the document were explored 12 different "in the wild" malware samples and RootkitAnalyzer was not only able to detect the presence of infection, but infection could be also classified based on RootkitAnalyzer output. Proposed tool does not generate "false positives" on clean system and thus can be widely used. However, RootkitAnalyzer can be improved - further work may consist of automatic malware classification module and cure module.


[Encyclopedia entry 2011]


[Encyclopedia entry 2011]


[Encyclopedia entry 2011]


[Encyclopedia entry 2011]

APPENDIX: VIRTUAL BOX

VirtualBox is used to test the RootkitAnalyzer tool for finding the hooks. VirtualBox is a powerful and great virtualization product. Besides its high performance and great features, the most important feature is that it is open source software. VirtualBox is available for almost all famous operating systems like Windows, Mac, Linux, and Solaris. First, this VirtualBox is developed by a small software company and from then Sun Microsystems purchased it. Now Oracle Corporation is developing the VirtualBox and releasing it with updates. The older name for this VirtualBox is Sun VirtualBox but, now it is known to be Oracle VirtualBox.

This VirtualBox software is installed on the existing operating system similar to normal applications. This application works as home, which accepts the other guest operating systems and run each operating system as a different system with its own properties and environment. The settings for Virtual box are shown in Figure 23.

VirtualBox Settings:

After opening the VirtualBox application, click on the settings tab. In General settings select the advanced tab and enable the first two options and choose shared clipboard as Bidirectional. It is shown in Figure 23 a).
Figure 23 a) VirtualBox General-Advanced Settings

Next, click on the basic tab and chose the operating system as Microsoft Windows and version as Windows XP. It is shown in Figure 23 b).

Figure 23 b) VirtualBox General-Basic Settings
Next, click the Description tab and leave it as it is. It is shown in Figure 23 c).

![Figure 23 c) VirtualBox General-Description Settings](image)

Now go to system settings and click the acceleration tab. Check the both options and click ok. It is shown in Figure 23 d).

![Figure 23 d) VirtualBox System-Acceleration Settings](image)
Now, in system settings go to Motherboard tab and enable CD/DVD-ROM and Hard Disk. It is shown in Figure 23 e).

![Figure 23 e) VirtualBox System-Motherboard Settings](image)

Next, click the processor tab and select 1 CPU and execution cap as 100%.

It is shown in Figure 23 f).

![Figure 23 f) VirtualBox System-Processor Settings](image)
Now, click the display settings and go to Remote display tab and uncheck the Enable server option. It is shown in Figure 23 g).

![Figure 23 g) VirtualBox Display-Remote Display Settings](image)

Now click the video tab and select video memory as 128MB and monitor count to 1. It is shown in figure 23 h).

![Figure 23 h) VirtualBox Display-Video Settings](image)
Now click the audio settings and check the enable audio option. It is shown in Figure 23 i).

![VirtualBox Audio Settings](image1)

**Figure 23 i) VirtualBox Audio Settings**

In network settings, uncheck the enable network adapter option. It is shown in Figure 23 j).

![VirtualBox Network Settings](image2)

**Figure 23 j) VirtualBox Network Settings**
In serial port settings uncheck the enable serial port option. It is shown in Figure 23 k).

![VirtualBox Serial Ports Settings](image)

**Figure 23 k) VirtualBox Serial Ports Settings**

Enable the USB controller option in the USB settings. It is shown in Figure 23 l).

![VirtualBox USB Settings](image)

**Figure 23 l) VirtualBox USB Settings**