C H A P T E R- 1

INTRODUCTION

Malware is a collective term for any malicious software which enters system without authorization of user of the system. The term is created from merging the words 'malicious' and 'software'. Malware is a very big threat in today's computing world. It continues to grow in volume and evolve in complexity. As more and more organizations try to address the problem, the number of websites distributing the malware is increasing at an alarming rate and is getting out of control. Most of the malware enters the system while downloading files over Internet. Once the malicious software finds its way into the system, it scans for vulnerabilities of operating system and perform unintended actions on the system finally slowing down the performance of the system.

Malware has ability to infect other executable code, data/system files, boot partitions of drives, and create excessive traffic on network leading to denial of service. When user executes the infected file; it becomes resident in memory and infect any other file executed afterwards. If operating system has a vulnerability, malware can also take control of system and infect other systems on network. Such malicious programs (virus is more popular term) are also known as parasites and adversely affect the performance of machine generally resulting in slow-down. Malware's includes computer viruses, spyware, dishonest ad-ware, rootkits, Trojans, dialers etc.
A computer virus is a self-replicating program that without the user's knowledge or intervention attaches itself to or replaces an executable file, a data file that can contain embedded executable code, or system areas or executable. At best, this type of malicious software (malware) simply utilizes computer or network resources; at worst it intentionally compromises the computer system's confidentiality, data integrity, or availability.

Security products such as virus scanners look for characteristics byte sequence (signature) to identify malicious code. The quality of the detector is determined by the techniques employed for detection. A good malware detection technique must be able to identify malicious code that is hidden or embedded in the original program and should have some capability for detection of yet unknown malware. Commercial virus scanners have very low resilience to new attacks because malware writers continuously make use of new obfuscation methods so that the malware could evade detections.

1.1 Problem definition

Regardless of manufacturer type or OS platform, Antivirus software works on a similar basis. Once installed, Antivirus software monitors all read and write attempts from any device, i.e. hard drive or network, calculating a file signature that it compares against a built-in database of known virus signatures. Most Antivirus software also has some heuristics built in that checks against possible virus activity for un-catalogued viruses.

The main weakness then of Antivirus software is the constant race to identify and protect against new viruses. The consumer must keep the Antivirus signatures up-to-date in order to stay optimally protected. Most modern scanners offer a subscription system and utilities to automatically check for and download new signatures. If a virus is detected, Antivirus software attempts to reverse the damage and eliminate it. This is not always possible, as is the case when files are replaced or overwritten.

As in all areas of computer security the objectives is to minimize risk. Our aim is to develop antivirus software which will be used by any user who has a computer with windows platform installed. We are now ready to try our best to fight and fix the problem created by these destructive codes (VIRUSES).

Malware Types

Malware can be roughly categorized into types according to the malware's method of operation. Anti-"virus" software, despite its name, is able to detect all of these types of malware. There are three characteristics associated with these malware types.

1. Self-replicating malware actively attempts to propagate by creating new copies, or instances, of itself. Malware may also be propagated passively, by a user copying it accidentally, but this isn't self-replication.
2. The population growth of malware describes the overall change in the number of malware instances due to self-replication. Malware that doesn't self-replicate will always have a zero population growth, but malware with a zero population growth may self-replicate.
3. Parasitic malware requires some other executable code in order to exist. "Executable" in this context should be taken very broadly to include any-thing that can be executed, such as boot block code on a disk, binary code in applications, and interpreted code. It also includes source code, like application scripting languages, and code that may require compilation before being executed.

- **Virus**

Self-replicating: yes
Population growth: positive
Parasitic: yes
A virus is malware that, when executed, tries to replicate itself into other executable code; when it succeeds, the code is said to be infected? The infected code, when run, can infect new code in turn. This self-replication into existing executable code is the key defining characteristic of a virus. When faced with more than one virus to describe, a rather silly problem arises. There's no agreement on the plural form of "virus." The two leading contenders are "viruses" and "virii;" the latter form is often used by virus writers themselves, but it's rare to see this used in the security community, who prefer "viruses. “If viruses sound like something straight out of science fiction, there's a reason for that. They are. The early history of viruses is admittedly fairly murky, but the first mention of a computer virus is in science fiction in the early 1970s. Traditionally, viruses can propagate within a single computer, or may travel from one computer to another using

human-transported media, like a floppy disk, CD-ROM, DVD-ROM, or USB flash drive. In other words, viruses don't propagate via computer networks; networks are the domain of worms instead.
However, the label "virus" has been applied to malware that would traditionally be considered a worm, and the term has been diluted in common usage to refer to any sort of self-replicating malware. Viruses can be caught in various stages of self-replication. A germ is the original form of a virus, prior to any replication. A virus which fails to replicate is called an intended. This may occur as a result of bugs in the virus, or encountering an unexpected version of an operating system. A virus can be dormant, where it is present but not yet infecting anything - for example, a Windows virus can reside on a Unix-based file server and have no effect there, but can be exported to Windows machines."

- **Ad ware**

Self-replicating: no
Population growth: zero
Parasitic: no
Adware has similarities to spyware in that both are gathering information about the user and their habits. Adware is more marketing-focused, and may pop up advertisements or redirect a user's web browser to certain web sites in the hopes of making a sale. Some adware will attempt to target the advertisement to fit the context of what the user is doing. For example, a search for "Calgary" may result in an unsolicited pop-up advertisement for "books about Calgary." Adware may also gather and transmit information about users which can be used for marketing purposes. As with spyware, adware does not self-replicate

- **Spy ware**

Self-replicating: no
Population growth: zero
Parasitic: no
Spyware is software which collects information from a computer and transmits it to someone else. Prior to its emergence in recent years as a threat, the term "spyware" was used in 1995 as part of a joke, and in a 1994 Usenet posting looking for "spy-ware" information. The exact information spyware gathers may vary, but can include anything which potentially has value:

1. Usernames and passwords. These might be harvested from files on the machine, or by recording what the user types using a key logger. A key logger differs from a Trojan horse in that a key logger passively captures keystrokes only; no active deception is involved.
2. Email addresses, which would have value to a spammer.
3. Bank account and credit card numbers.
4. Software license keys, to facilitate software pirating.

Viruses and worms may collect similar information, but are not considered spy ware, because spy ware doesn't self-replicate. Spy ware may arrive on a machine in a variety of ways, such as bundled with other software that the user installs, or exploiting technical flaws in web browsers.

The latter method causes the spyware to be installed simply by visiting a web page, and is sometimes called a drive-by download.

- **Worm**

Self-replicating: yes
Population growth: positive
Parasitic: no
A worm shares several characteristics with a virus. The most important characteristic is that worms are self-replicating too, but self-replication of a worm is distinct in two ways. First, worms are standalone, and do not rely on other executable code. Second, worms spread from machine to machine across net-works

- **Trojan horse**

Self-replicating: no
Population growth: zero
Parasitic: yes
There was no love lost between the Greeks and the Trojans. The Greeks had besieged the Trojans, holed up in the city of Troy, for ten years. They finally took the city by using a clever ploy: the Greeks built an enormous wooden horse, concealing soldiers inside, and tricked the Trojans into bringing the horse into Troy. When night fell, the soldiers exited the horse and much unpleasantness ensued.
In computing, a Trojan horse is a program which purports to do some benign task, but secretly performs some additional malicious task. A classic example is a password-grabbing login program which prints authentic-looking "username" and "password" prompts, and waits for a user to type in the information. When this happens, the password grabber stashes the information away for its creator, and then prints out an "invalid password" message before running the real login program. The unsuspecting user thinks they made a typing mistake and re-enters the information, none the wiser.

Strategies of Computer virus

A computer virus is a computer program that can copy itself and infect a computer without permission or knowledge of the user. In order to avoid detection by users, some viruses employ different kinds of deception such as the following Strategies

• **Overwriting Virus:** this type of virus overwrites files with their own copy. Of course, this is a very primitive technique, but it is certainly the easiest approach of all. Overwriting viruses cannot be disinfected from a system. Infected files must be deleted from the disk.

• **Companion Infection:** one approach to becoming a companion to an EXE file is to give the virus the same base name as the targeted program, but use a .COM extension instead of .EXE. This technique was employed by the Globe virus, first detected in 1992. When the victim

attempts to launch an EXE program, he or she usually types its name without the extension. In such cases, Windows gives priority to a file with the .COM extension over a file with the same base name but with the .EXE extension.

• **Appending Virus:** In this technique, a jump (JMP) instruction is inserted at the front of the host to point to the end of the original host. A typical example of this virus is Vienna. The appending technique can be implemented for any other type of executable file, such as EXE, PE, formats, and so on. Such files have a header section that stores the address of the main entry point, which, in most cases, will be replaced with a new entry point to the start of the virus code appended to the end of the file.

• **Prepending Virus:** This virus inserts its code at the front of host programs. This is a simple kind of infection, and it is often very successful. Virus writers have implemented it on various operating systems, causing major virus outbreaks in many. An example of a COM prep ender virus is the Hungarian virus Polimer.512.A, which prepends itself, 512 bytes long, at the front of the executable and shifts the original program content to follow itself.

• **Cavity or space filler Virus:** This virus attempts to install itself in this empty space while not damaging the actual program itself. An advantage of this is that the virus then does not increase the length of the program and can avoid the need for some stealth techniques. The Lehigh virus was an early example of a cavity virus. Because of the difficulty of writing this type of virus and the limited number of possible hosts, cavity viruses are rare.

• **Compressing Virus:** A special virus infection technique uses the approach of compressing the content of the host program. Sometimes this technique is used to hide the host program's size increase after the infection by packing the host program sufficiently with a binary packing algorithm.

• **Encrypted Virus:** consists of a constant descriptor followed by the encrypted virus body. It's relatively easy to detect because the descriptor is constant. The first known virus that implemented encryption was Cascade on DOS. Oligomorphic virus changes its descriptors in new generations. The simplest technique to change the descriptors is to use a set of descriptors instead of a single one. The first known virus to use this technique was Whale. Whale carried a few dozen different descriptors, and the virus picked one randomly.

• **Boot Sectors Virus:** this virus takes advantage of the executable nature of master boot record (MBR) and partition boot sector (PBS). A PC infected with a boot sector virus will execute the virus's code when the machine boots up. Michelangelo virus is an example of a Boot Sectors Virus.

• **Macro virus:** infects a Microsoft Word or similar application and causes a sequence of actions to be performed automatically when the application is started or something else triggers it. Macro viruses tend to be surprising but relatively harmless. A typical effect is the undesired insertion of some comic text at certain points when writing a line. A macro virus is often spread as an e-mail virus. A well-known example in March, 1999 was the Melissa virus

Some of the commonly available viruses are listed below one by one

- **regsvr.exe**

The file regsvr.exe is located in the folder C:\Windows\System32 or sometimes in a subfolder of "C:\Program Files". Known file sizes on Windows 7/XP are 617,343 bytes (21% of all occurrences), 807,388 bytes and 33 more variants.

There is no information about the author of the file. It is not a Windows system file. HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrenVersion\Run, HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\ShellHKEY_USERS\.DEFAULT\Software\Microsoft\Windows\ CurrentVersion\Run). The program has no visible window. The file is an unknown file in the Windows folder. Regsvr.exe is able to record inputs, monitor applications and manipulate other programs. Therefore the technical security rating is 77% dangerous.

If regsvr.exe is located in the folder C:\Windows, the security rating is 76% dangerous. The file size is 14,832 bytes (33% of all occurrences), 14,838 bytes and6 more variants. The process has no file description. The program has no visible window. The regsvr.exe file is located in the Windows folder, but it is not a Windows core file. The file is not a Windows system file. Regsvr.exe is able to hide itself, monitor applications, and manipulate other programs and record inputs.

If regsvr.exe is located in a subfolder of "C:\Documents and Settings", the security rating is 65% dangerous. The file size is 6,511,616 bytes (25% of all occurrences), 617,472 bytes, 635,259 bytes or 13,179,660 bytes. There is no information about the author of the file. The program has no visible window. The program starts upon Windows startup (see Registry key: HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run, HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell, HKEY_USERS\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\Run). The file is not a Windows core file. Regsvr.exe is able to record inputs, monitor applications and manipulate other programs.

- **new folder.exe**

New Folder.exe is an executable file (a program) for Windows. The filename extension .exe is the abbreviation for executable. Only run executable files from publishers you trust, because executable files can potentially change your computer settings or harm your computer. The free file information forum can help you determine if New Folder.exe is a virus, Trojan, spy ware, or ad ware that you can remove, or a file belonging to a Windows system or an application you can trust. The process known as New Folder.exe belongs to software New Folder by File Folder.

New Folder.exe is located in C:\. Known file sizes on Windows 7/XP are 431,908 bytes (25% of all occurrences), 290,419 bytes, 753,921 bytes or 266,734 bytes.
The program has no file description. The program is not visible. The New Folder.exe file is not a Windows system file. New Folder.exe is able to record inputs, monitor applications and manipulate other programs. Therefore the technical security rating is 56% dangerous.
If New Folder.exe is located in a subfolder of "C:\Program Files", the security rating is 74% dangerous. The file size is 455,939 bytes. There is no information about the author of the file. The program has no visible window. The New Folder.exe file is not a Windows core file. The process uses ports to connect to a LAN or the Internet. New Folder.exe is able to record inputs, monitor applications and manipulate other programs.

If New Folder.exe is located in the folder C:\Windows\System32, the security rating is 56% dangerous. The file size is 40,960 bytes. There is no file information. The program is not visible. It is located in the Windows folder, but it is not a Windows core file. It is not a Windows system file.

- **update.exe**

Update.exe is an executable file (a program) for Windows. The filename extension .exe is the abbreviation for executable. Only run executable files from publishers you trust, because executable files can potentially change your computer settings or harm your computer. The free file information forum can help you determine if update.exe is a virus, Trojan, spy ware, or ad ware that you can remove, or a file belonging to a Windows system or an application you can trust.

The file update.exe is located in a subfolder of "C:\Program Files\Common Files". Known file sizes on Windows 7/XP are 14,336 bytes (32% of all occurrences), 131,072 bytes and 21 more variants.

There is no file information. The file is not a Windows core file. The program is not visible. Therefore the technical security rating is 62% dangerous.
If update.exe is located in a subfolder of C:\Windows, the security rating is 48% dangerous. The file size is 755,576 bytes (20% of all occurrences), 716,000 bytes and 50 more variants. The update.exe file is not a Windows core file. The program is not visible. Update.exe is able to manipulate other programs, monitor applications, record inputs and hide itself.

If update.exe is located in a subfolder of "C:\Program Files", the security rating is 35% dangerous. The file size is 303,104 bytes (23% of all occurrences), 404,737 bytes and 48 morevariants. Update.exe is not a Windows system file. The program has no visible window. Update.exe is able to monitor applications, record inputs, manipulate other programs and connect to the Internet.

If update.exe is located in a subfolder of "C:\Documents and Settings", the security rating is 54% dangerous. The file size is 397,824 bytes (21% of all occurrences), 1,470,530 bytes and 22 more variants.

If update.exe is located in the folder C:\Windows\System32, the security rating is 64% dangerous. The file size is 95,690 bytes (25% of all occurrences), 32,256 bytes and 11 more variants.

If update.exe is located in a subfolder of C:\Windows\System32, the security rating is 50% dangerous. The file size is 655,396 bytes (14% of all occurrences), 409,600 bytes and 10 more variants.

If update.exe is located in the folder C:\Windows, the security rating is 72% dangerous. The file size is 237,450 bytes (14% of all occurrences), 109,056 bytes and 5 more variants.

If update.exe is located in a subfolder of C:\, the security rating is 63% dangerous. The file size is 10,752 bytes (33% of all occurrences), 245,760 bytes, 87,040 bytes or 368,640 bytes.

If update.exe is located in C:\, the security rating is 63% dangerous. The file size is 51,712 bytes (50% of all occurrences) or 24,576 bytes.

If update.exe is located in the Windows Temp folder, the security rating is 74% dangerous. The file size is 238,080 bytes.

- **W32.Sality.L**

W32.Sality.L is a key logging; polymorphic, file-infecting virus that also has worm behavior. According to the Symantec report, Sality drops a malicious DLL in the c: nwindowsnsystem folder, calls it, and then injects the DLL into a running process. It then writes its configuration information to system.ini.

- **firefox.exe**

Firefox.exe is an executable file (a program) for Windows. The filename extension .exe is the abbreviation for executable. Only run executable files from publishers you trust, because executable files can potentially change your computer settings or harm your computer. The free file information forum can help you determine if firefox.exe is a virus, Trojan, spy ware, or ad ware that you can remove, or a file belonging to a Windows system or an application you can trust. The file firefox.exe is located in a subfolder of "C:\Program Files" or sometimes in the folder "C:\Program Files" or in a subfolder of the "My Files" folder (usually C:\Program Files\Mozilla Firefox\). Known file sizes on Windows 7/XP are 307,704 bytes (5% of all occurrences) 7,633,008 bytes and 97more variants the firefox.exe file is not a Windows core file. The application listens for or sends data on open ports to a LAN or the Internet. It is digitally signed. Firefox.exe is able to record inputs and monitor applications. Therefore the technical security rating is 23% dangerous.

If firefox.exe is located in a subfolder of C:\, the security rating is 27% dangerous. The file size is 6,637,161 bytes (10% of all occurrences), 7,667,312 bytes and 21 more variants. The firefox.exe file is not a Windows system file. The process uses ports to connect to a LAN or the Internet. The file has a digital signature. Firefox.exe is able to record inputs and monitor applications.

If firefox.exe is located in a subfolder of "C:\Documents and Settings", the security rating is 36% dangerous. The file size is 924,632 bytes (16% of all occurrences), 307,712 bytes and 9 more variants. The file is not a Windows core file. The program has no visible window. Firefox.exe is able to record inputs and monitor applications.

If firefox.exe is located in the folder C:\Windows\System32, the security rating is 69% dangerous. The file size is 544,768 bytes (50% of all occurrences), 64,524 bytes, 1,263,695 bytes or 7,168 bytes.

If firefox.exe is located in a subfolder of "C:\Program Files\Common Files", the security rating is 40% dangerous. The file size is 307,712 bytes (66% of all occurrences) or 307,704 bytes.
If firefox.exe is located in the "My Files" folder, the security rating is 48% dangerous. The file size is 7,190,637 bytes (50% of all occurrences) or 912,344 bytes.
If firefox.exe is located in the folder C:\Windows, the security rating is 86% dangerous. The file size is 76,800 bytes.

- **sqlserver.exe**

Sqlservr.exe is an executable file (a program) for Windows. The filename extension .exe is the abbreviation for executable. Only run executable files from publishers you trust, because executable files can potentially change your computer settings or harm your computer. The free file information forum can help you determine if sqlservr.exe is a virus, Trojan, spy ware, or ad ware that you can remove, or a file belonging to a Windows system or an application you can trust.

The file sqlservr.exe is located in a subfolder of "C:\Program Files" (usually C:\mssql7\Binn\ or C:\Program Files\Microsoft SQL Server\MSSQL\Binn\). Known file sizes on Windows 7/XP are 13,179,660 bytes (33% of all occurrences), 7,520,337 bytes and 19 more variants.
The program has no visible window. It is not a Windows core file. The application can be uninstalled in the Control Panel. Sqlservr.exe is able to monitor applications and connect to the Internet. Therefore the technical security rating is 37% dangerous; however also the users read reviews.

If sqlservr.exe is located in a subfolder of C:\Windows, the security rating is 22% dangerous. The file size is 13,179,660 bytes. The program is not visible. The program can be uninstalled in the Control Panel. It is certified by a trustworthy company. Sqlservr.exe is not a Windows system file. Sqlservr.exe is able to connect to the Internet and monitor applications.
If sqlservr.exe is located in a subfolder of "C:\Program Files\Common Files", the security rating is 41% dangerous. The file size is 7,520,337 bytes. The program is not visible. The process can be removed using the control panel Add\Remove programs applet. The sqlservr.exe file is not a Windows core file. The program listens for or sends data on open ports to a LAN or the Internet.

1.2 Existing System Description

There are so many antivirus software's in the world today, but particularly we will consider the following antivirus Smadav, Avira, Eset etc. Updating of an Antivirus system is very essential for computer security. Because if we do not update our Antivirus software it will be difficult for the system to detect new viruses as their signature is not added to the virus definition data base. In our country the main problem is that the Antivirus vendors do not have any idea of the viruses created locally. When they send updates of their antivirus it will only have the signatures (updates) of the newly founded viruses in their country. Even though the viruses created locally are not that much complicated to detect, they are yet creating problems because the existing systems do not have any signatures of these viruses. That is why they can easily bypass the popular antivirus available today.
Details for some of the existing antivirus software's are shown below.

- **AVG**

The brand AVG comes from Grisoft's first product, "Anti-Virus Guard", launched in 1992 in Czechoslovakia. In 1997, the first AVG licenses were sold in Germany and UK. AVG was introduced in the U.S. in 1998.

AVG Technologies provides a number of products from the AVG range, suitable for Windows 2000 onwards. In addition to this, AVG Technologies also provides Linux, FreeBSD, and most recently Mac OS X versions of the software. AVG Anti-Virus 9.0 is available in free and commercial editions. AVG 9.0 has identity theft protection through a partnership with Intersections Inc.; AVG 9.0 also adds white listing, behavioral protection and cloud operations to their signature-based blocking. The software adds the Resident Shield, firewall, and identity protection modules. The Link Scanner component has been improved to cut phishing threats further. Version 9 was the last version compatible with Windows 2000.

All versions of the AVG products, excluding AVG Anti-Rootkit Free Edition (now discontinued), are compatible with the 64-bit edition of Windows.
.AVG features most of the common functions available in modern anti-virus and Internet security programs, including periodic scans, scans of sent and received emails (including adding footers to the emails indicating this), the ability to "repair" some virus-infected files, and a quarantine area ("virus vault") in which infected files are held.

- **ESET**

It is IT Security Company headquartered inBratislava, Slovakia that was founded in 1992 by the merger of two private companies. ESET competes in the antivirus industry against Avira, Kaspersky, McAfee, Symantec, among others.

ESET's first product was NOD, an antivirus software for computers running MS-DOS. In 1998, ESET introduced NOD32 1.0 for Microsoft Windows, with version 2.0 following in 2003. The last release of the 2.x codebase was 2.70.39, which released in November 2006 and was discontinued in May 2012.

ESET NOD32 Antivirus and ESET Smart Security 4.2 both support Windows 2000, XP, Server 2003, Server 2003 R2, Vista, Server 2008, Server 2008 R2 and Windows 7; additionally, ESET NOD32 Antivirus 4.2 supports Microsoft Windows NT 4.0 with Service Pack 6a. Support for 64-bit (x86-64, not IA-64) versions of these operating systems is provided, although the program runs both 32-bit and 64-bit processes. Users of older versions of Microsoft Windows, such as Windows 95, 98, Me and NT 4.0, will need to install the older 2.70.39 release. In addition to Microsoft Windows, the company also supports the following operating systems: BSD, Linux, Mac OS X, Novell NetWare and Sun Solaris.

Improvements in ESET NOD32 and Smart Security 5 include an Enhanced Media Control that automatically scans all USB, external and CD/DVD drives for any threats and blocks removable media based on type and manufacturer of the media. Users can customize the behavior of the software by specifying rules for current running processes, programs, and registry and tweak the security posture of the software. Gamer Mode being introduced into the software conserves system resources and activities for gaming and other full-screen programs by halting all pop-up windows that require user intervention. ESET's use of assembly language in its products contributes to their low system requirements and disk space utilization.

ESET calls its scanning engine Threat Sense, and makes extensive use of generic signatures and heuristics. This should not be confused with ThreatSense.Net, which is ESET's system for submitting suspicious files and malware to their virus researchers. ESET's products are regularly tested by organizations such as AV-Comparatives, AV-TEST and Virus Bulletin.

- **Avira**

Avira Operations GmbH & Co. KG is a German multinational and family-owned antivirus software company. Avira competes in the antivirus industry against Kaspersky, McAfee, Symantec, among others.

Avira periodically "cleans out" the virus definition files, by replacing specific signatures with generic ones, resulting in a general increase in performance and scanning speed. A database clean-out with the size of 15 MB was made on 27 October 2008, causing problems to the users of the free edition because of its large size and slow servers of the free edition. To solve the problem, Avira improved the updating process by reducing the size of the individual updatable files, resulting in the delivery of less data in each update. Nowadays there are 32 smaller definition files that are updated regularly in order to avoid peaks in the download of the updates.
Avira Protection Cloud (APC) was first introduced in version 2013. The aim of APC is improving detection with less impact on system performance due to the use of cloud computing. APC was initially released as a sneak technical preview to selected testers.

- **Kaspersky**

Kaspersky is a Russianmulti-nationalcomputer security company, co-founded by Eugene Kaspersky and Natalia Kaspersky in 1997. Kaspersky Lab is a developer of secure content and threat management systems and the world's largest privately held vendor of software security products. Kaspersky Lab is headquartered in Moscow. The company currently works in almost 200 countries. Kaspersky Lab competes in the antivirus industry against Avira, Bit Defender, McAfee, Symantec, among others.

In addition to the company's consumer products, Kaspersky Lab offers a variety of security applications designed for small business, corporations and large enterprises. These offerings include security software to protect workstations, file servers, mail servers, mobile devices, and Internet gateways, all managed through a centralized Administration Kit. These applications are also available in bundled security suites, scaled to fit the requirements of organizations of varying sizes.
The latest 2012 line of Kaspersky products are certified for Windows 8 and are multi-core optimized.

- **McAfee**

McAfee, Inc**.**, formerly McAfee Security is an American global computer security software company headquartered in Santa Clara, California, and the world's largest dedicated security technology company. As of February 28, 2011, McAfee is a wholly owned subsidiary of Intel.
McAfee's competitors in the antivirus industry are Avira, Bit Defender, Kaspersky, Symantec, and to name a few.

On March 17, 2010, McAfee launched Cloud Secure program, a new service for Software-as-a-Service (SaaS) providers to add additional security to their cloud deployments. The new program includes cloud security certification services that are provided on an annual basis and will include existing security controls, processes and certifications, as well as future cloud security standards; and automatic and daily security audits, remediation of vulnerabilities and reporting of the security status of their service and network using the McAfee Cloud Secure service.

On April 21, 2010, beginning at approximately 14:00 UTC, millions of computers worldwide running Windows XP Service Pack 3 were affected by an erroneous virus definition file update by McAfee, resulting in the removal of a Windows system file (svchost.exe) on those machines, causing machines to lose network access and, in some cases, enter a reboot loop. McAfee rectified this by removing and replacing the faulty DAT file, version 5958, with an emergency DAT file, version 5959 and has posted a fix for the affected machines in their consumer knowledge base. The University of Michigan's medical school reported that 8,000 of its 25,000 computers crashed. Police in Lexington, Ky., resorted to hand-writing reports and turned off their patrol car terminals as a precaution. Some jails canceled visitation, and Rhode Island hospitals turned away non-trauma patients at emergency rooms and postponed some elective surgeries. Australian supermarket Coles reported that 10 percent (1,100) of its point-of-sales

terminals were affected and was forced to shut down stores in both western and southern parts of the country.

- **SmadAV**

Most antivirus software cannot be installed with other antivirus; it is because the antivirus is designed for primary protection on your computer. In contrast to SmadAV, SmadAV is the type of antivirus are designed as additional protection, so 100% compatible and can work well although there has been another antivirus on your computer, in this case serves as a layer of defense SmadAVsecond. SmadAV has its own way (behavior, heuristic, and whitelisting) in detecting and cleaning viruses that will further enhance the security of the computer. Because the resource usage is very small SmadAV, SmadAV will not increase your computer's performance under heavy use. So, with a mix between SmadAV and antivirus protection that is installed on your computer will further strengthen the defense of your computer from virus infection.

USB stick is one of the biggest media spread of the virus in Indonesia. SmadAV have special technology for total prevention of virus that spreads via USB stick. SmadAV mission is no longer 100% viral infection of the flash. SmadAV have enough signatures of viruses that infect the flash, and has a special ability to detect new viruses in flash although not in the database SmadAV. Not only prevention, SmadAV also able to clean up a virus that infects and restore virus hidden files in the USB stick

SmadAV very well be used for computers that rarely or even not connected to the internet. SmadAV not need to update as often as other antivirus that usually do updates per week and even per day. SmadAV usually do updates only once a month (monthly).SmadAV not overly dependent on the signature / virus database, but much depends on the behavior detection techniques, heuristic, and whitelisting.

SmadAV also able to clean the virus that has infected your computer and fix registry altered by the virus. Other antivirus usually not did cleaning the registry so that the computer has not returned to normal after cleaning the antivirus. Many supporting tools that is included in SmadAV as a weapon for cleaning the virus. Note: Not all types of viruses can be cleaned SmadAV, SmadAV is still not able to clean the virus type or types of rootkits penginfeksiprogram (e.g. virus Ramnit, Sality, Alman, Virus, etc...) Because this strain has damaged most of your program files. For users SmadAV Free, you must re-download the file SmadAV 2012 Latest revisionSmadAV then open the program on your computer and do not need an internet connection again; there will be a confirmation that will be updated SmadAV. SmadAV Free version does not have an automatic update feature to get automatic updates via the Internet you have to use the Pro version SmadAV. If using SmadAV Pro, you no longer need to think about updating existing SmadAV because SmadAV on your computer will be automatically updated when connected to the internet. SmadAV will continue to be revised in a period usually once a month.

1.2.1 Drawbacks

1. They don't detect viruses that are created locally, since they do not have signatures for these viruses. This is because the Antivirus vendors don't know the viruses created in our country. So that they cannot provide signatures for the locally created viruses in the updates.

1.3 Proposed System Description

The main focus of the Antivirus system is to detect/delete the malwares that are mainly available locally (especially in our country). We are specially focusing on local malwares because

1. The internationally released antivirus system like Avira, Smadav, and Esset doesn't seem to find/detect the viruses that are widely distributed in this country's system because of their updates don't include these signatures.

2. Due to time constraints, we focus on detection of local viruses.

1.3.1 Advantages/Features

1. To detect and delete the viruses that is found.
2. It will not delete important user/system information.
3. It is easier to create signatures of locally available viruses and add their signatures to the anti-viruses definition data base.
4. Easy and fast updates and releases
5. Ease of use through language support

CHAPTER-2

SYSTEM STUDY

Introduction

The software that we are developing is designed to scan, detect and delete malware. It will also restore corrupted files to their original status as required.

As we know nowadays the main problem with using systems is the existence of various types of malwares that make it harder and difficult to operate properly. So the main goal of our system is to focus on the detection of malwares that are more common in our country. This is because in our country there are certain and specific types of malwares that are widely infecting the systems. So the software will be oriented to detecting these types and handling them accordingly. Another feature we are trying to incorporate is to make it very user friendly software by adding language support.

2.1 Product specification

Some other features and specifications that we are trying to incorporate are:-

- On-demand scan

  If user suspect that his/her computer is infected (it behaves abnormally), run an on-demand computer scan to examine your computer for infiltrations. From a security point of view, it is essential that computer scans are not just run when an infection is suspected, but regularly as part of routine security measures.

  This runs when explicitly started by the user. An on-demand scan may also be desirable when an infection is suspected, or when a questionable file is downloaded. It's also applicable for the different system disk drives.

- An on-access scan

  This runs continuously, scanning every file when it's accessed. As might be expected, the extra I/O overhead and resources consumed by the scanner impose a performance penalty.
- Targeted scan

  This will enable the user to scan a particular suspected file or directory or drive for malicious property.

- Full scan

  This will enable the user to scan the entire computer for viruses without any discrimination between the drives, folders, or files that reside on the system.

- Detect malwares from external devices

  These are devices like USB flash cards and external hard disks. This detection happens by scanning the device as soon as it is connected to the computer.

- Capability to quarantine files

  The quarantine folder is where infected or suspicious files are stored in a benign form. The resident protection module by default quarantines all newly created and modified suspicious files, but you can quarantine any file you wish. Quarantined files can be later restored to their original location, or to any location that you choose.

- Ability to do an offline virus signature update

  This is a very necessary feature because as we know in our country there is a weak internet connection. As such the users should be able to get the update signature manually and update the virus definition database offline. The updater file is uploaded in E.I.T server.
- Provide a log file of activities performed

  Threat log offers detailed information about infiltrations detected by the Antivirus modules. The information includes the time of detection, name of infiltration, location, the performed action and the name of the user logged in at the time the infiltration was detected. To copy or delete one or more lines from the log (or to delete the whole log), use the contextual menu.

  This would provide information about the different activities performed. Whether we have done an on demand access or other activities it will store it on the log file along with the different specifications necessary.
- Ability to open important system utilities

  These utilities are like task manager, command prompt, and registry editor that have been incapacitated due to malwares. As such this system will be providing an easy way or an alternative way of accessing these programs.

- Provide list of detected malwares

  After a full scan the system will give the user a list of all the detected viruses along with their location, size, and type of file. This information will help the user to be able to choose the appropriate action to take whether deleting, cleaning, quarantining the files.

- Provide a choice of actions to be performed on the malware

  The user will have the choice of either cleaning or deleting detected malicious files after a scanning task and before taking an action by the system itself. This is helpful because there is no such absolute security that means any antivirus software cannot provide absolute security that is why the antivirus software can not directly take action on the detected malware. Now days many problems related with antivirus systems are false positive and false negative problems

  - Restore

    This enables users to restore the file that are considered as malicious by the antivirus software which users consider that they are benign and they need to be retored.

  - Delete

    This action deletes detected files. If it is possible to clean an infected object, the "delete" action won't be displayed.

2.1.1 System objectives

To reiterate, the point of this entire system is to protect the confidentiality, integrity, and availability of our resources. We therefore, need to identify the assets, identify the threat, and then create proactive and reactive strategies to minimize risk and deal with incidents. Our proactive strategies aim to minimize risks, develop contingency plans, create and assign responsibilities to an incident response team. Reactive strategies assess damage, implement the contingency plan, repair the damage, and document the incident so that our policies can be re-examined and constantly improved. This is an ongoing and dynamic process.

We are approaching the creation or the development of the antivirus software with some clearly defined objectives. So we are listing below the main objectives included in our system:-

- To be able to identify a file whether it's malicious or benign

  This is necessary because our system needs to differentiate between harmful files and files that can cause damage on the overall system.
- To be able to define the type and location of the detected malware

  Once a malicious file is found its type and its location on the computer will be identified as part of the protection measure.
- To be able to clean, delete malwares

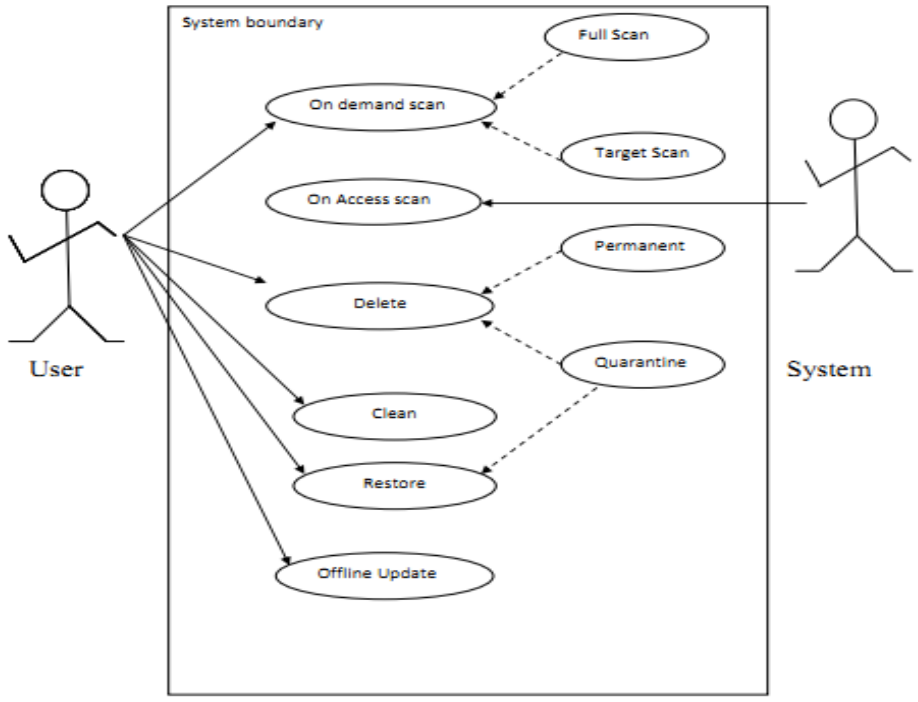
  To provide options for the user to select all or any detected viruses and take appropriate action on them. The actions to be taken are mainly clean, delete, quarantine.
- To be able to restore quarantined files

  These files will be restored to their original location and be able to run incases were the file is found to be benign.

- To block unauthorized, running malwares until the confirmation of deletion or cleaning or any other action by quarantining them.
- To be able to monitor any kind of file transfers to and from external devices.
- To be able to be used by the local people easily by being user friendly

  This especially focuses on the language support module. The users will be able to interact with the system by using the local language, which is Tigrigna. An antivirus cannot do anything alone user interaction and decision is needed whenever a malicious program is detected the antivirus software provides a list of options the user will take. So to make the system easily interactive with the user we added a language support for easier user interaction.

2.2-System use case diagram



2.2 System requirements

To develop this system we will need some important materials. Therefore the list below will provide the list of them:-

- Visual Studio 2010 by using Visual C# language
- WinRAR software for compressing and decompressing files(OPTIONAL)

- Bit assembler and de-assembler like IDA PRO and OllyDbg (OPTIONAL)
- Emulator like Sandbox (OPTIONAL)
- Windows operating System platform

2.3 Scope of the system

This software will be designed to operate in a computer with windows operating system. It will have the following constraints. The scope of the system is defined

- Detect, clean and delete locally available malware and any viruses that are found in any external device attached to the computer.
- Scan external and internal system devices that mean it able to detect external devices attached to the computer automatically and scan it for malicious code. It will also allow user to scan an external device on-Demand.
- At any time you can pause and resume scanning process.
- The update for the software is provided through offline update. So during updating user needs to get the update file and then update the antivirus manually.

2.4 Hardware and software requirements

- Minimum 256MB RAM
- Minimum 500MB free hard disk space
- Windows XP and above platforms

CHAPTER– 3

SYSTEM DESIGN

Introduction:

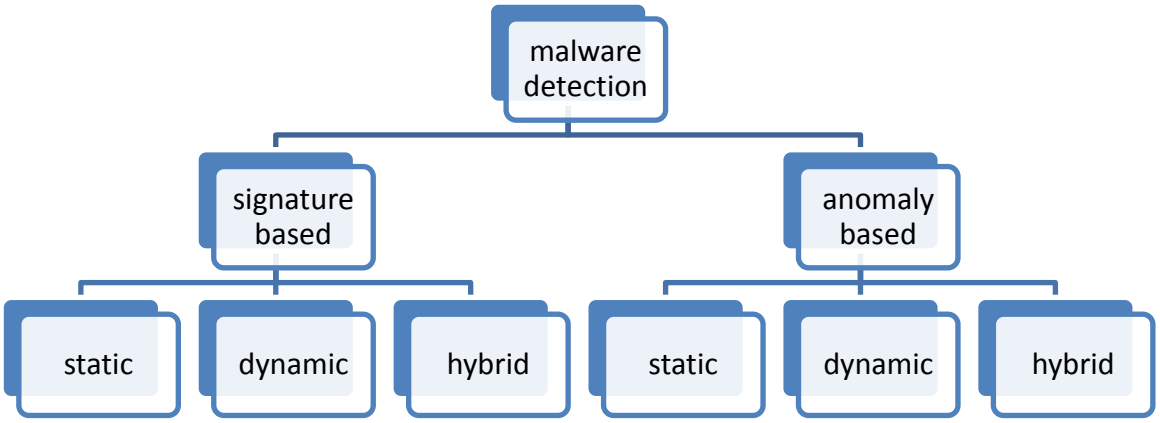
The purpose of system design is to specify how the program is designed. The system design will describe the required information for making the system, and also the life cycle. The design of the system is implemented into two methods.

3.1Malware Detection Techniques

Techniques used for detecting malware can be categorized broadly into two categories: anomaly-based detection and signature-based detection. An anomaly-based detection technique uses its knowledge of what constitutes nor mal behavior to decide the maliciousness of a program under inspection. A special type of anomaly-based detection is referred to as specification-based detection. Specification-based techniques leverage some specification or rule set of what is valid behavior in order to decide the maliciousness of a program under inspection. Programs violating the specification are considered anomalous and usually, malicious.

Signature-based detection uses its characterization of what is known to be malicious to decide the maliciousness of a pro-gram under inspection. As one may imagine this characterization or signature of the malicious behavior is the key to a signature-based detection method's effectiveness. Each of the detection techniques can employ one of three different approaches: static, dynamic, or hybrid. The specific approach or analysis of an anomaly-based or signature-based technique is determined by how the technique gathers information to detect malware. Static analysis uses syntax or structural properties of the program (static)/process (dynamic) under inspection (PUI) to determine its maliciousness. For example, a static approach to signature-based detection would only leverage structural information (e.g. sequence of bytes) to deter mine the maliciousness, whereas a dynamic approach will leverage runtime information (e.g. systems seen on the runtime stack) of the PUI. In general, a static approach attempts to detect malware before the program under inspection executes. Conversely, a dynamic approach

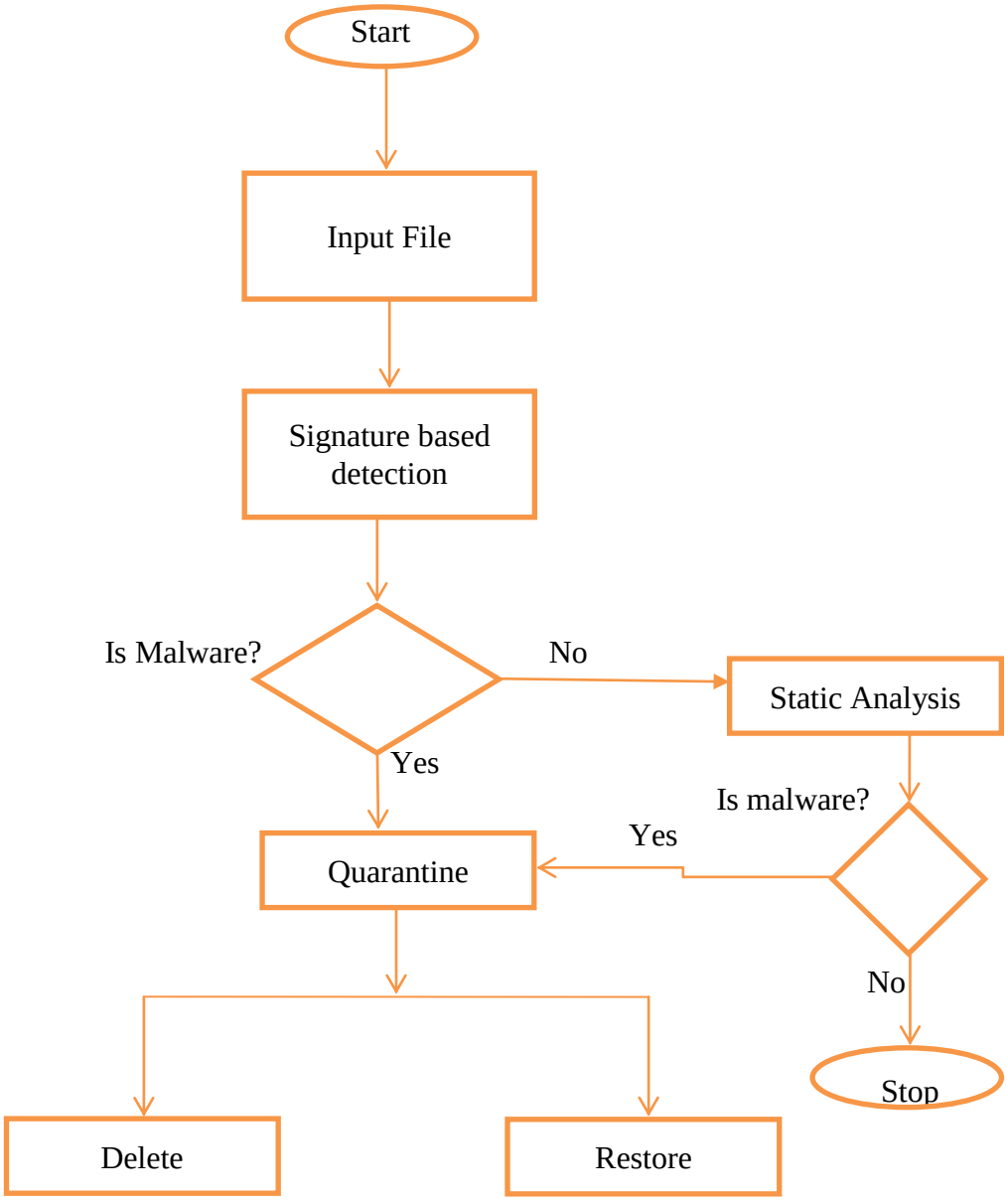
attempts to detect malicious behavior during program execution or after pro-gram execution. There are hybrid techniques that combine the two approaches. In this case, static and dynamic information is used to detect malware.



Classification of malware detection techniques

Flow diagram for malware detection of the system

The figure below shows the general flow of the entire system. First any file is input to the Antivirus then this file is sent to the signature based virus detector if the signature of the input file is found in the virus DB the file is then sent to the quarantine folder. Once the file is quarantined the user can either restore or delete the quarantined file from the quarantine folder. If the file signature is not found in the signature DB the file is sent to the malware detector based on static code analysis(i.e. analyzing the sequence of API calls in the virus code) then if the file contains a malicious behavior it is moved to the quarantine otherwise the file is benign those the process terminates.



3.1.1 Signature-based detection method

Signature-based detection attempts to model the malicious behavior of malware and uses this model in the detection of malware. The collections of all of these models represent signature-based detection's knowledge. This model of malicious behavior is often referred to as the signature. Ideally, a signature should be able to identify any malware exhibiting the malicious behavior specified by the signature. Like any data that exists in large quantities which requires storage, signatures require a repository. This repository represents all of the knowledge the signature-based method has, as it pertains to malware detection. The repository is searched when the method attempts to assess whether the PUI contains known signature. The most common signatures are hashes and byte-signature. We are choosing the hash signature.

Malware Detector

A Malware detector 'D' is defined as a function whose domain and range are the set of executable program 'P' and the set {malicious, benign}. In other words malware detector can be defined as shown below.

$$D(p) = \begin{cases} \text{malicious} & \text{malicious if p contains malicious code} \\ \text{Benign} & \text{benign otherwise.} \end{cases}$$

The detector scans the program 'p' ε P to test whether a program is benign program or malicious program. The goal of testing is to find out false positive, false negative, hit ratio. The malware detector detects the malware based on signatures of malware.The binary pattern of the machine code of a particular virus is called as signature. Antivirus programs compare their database of virus signatures with the files on the hard disk and removable media (including the boot sectors of the disks) as well as within RAM. The antivirus vendor updates the signatures frequently and makes them available to customers via the Web.

Hash Signatures

The most basic and easiest type of signature is a hash value. A hash value is created by a hash function that is a procedure or mathematical function which converts a large amount of data into a single value. The most commonly used hash function is MD5 and SHA-1. These hashes Functions are extremely accurate. The MD5 hashing algorithm produces a fixed 16 byte fixed output from a variable file size that is being inspected for malicious behavior.

Naming malware

Typically the primary, human-readable name of a piece of malware is decided by the anti-virus researcher who first analyzes the malware. Names are often based on unique characteristics that malware has, either some feature of its code or some effect that it has.

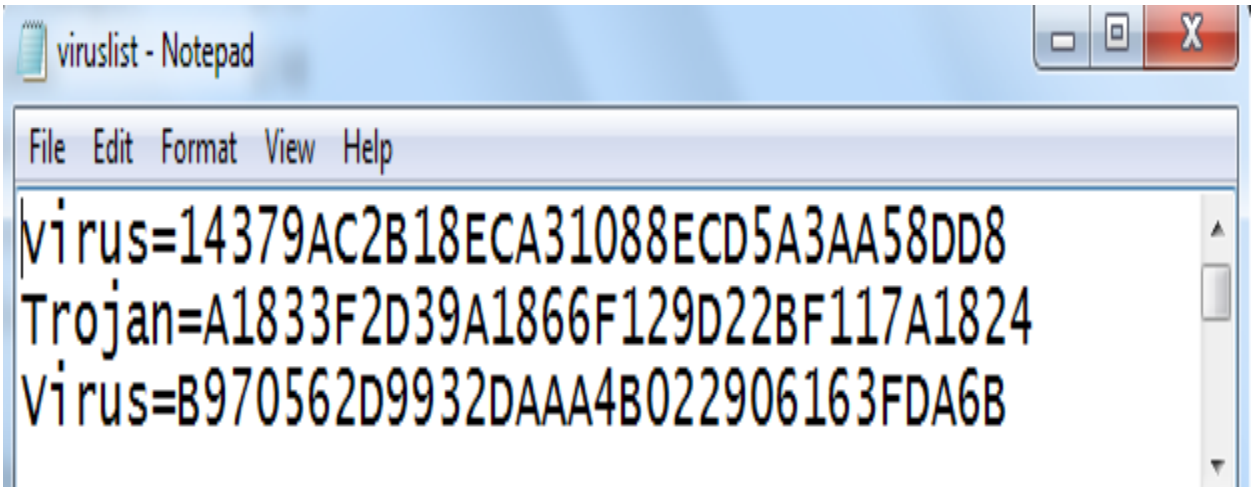
There is no central naming authority for malware, and the result is that a piece of malware will often have several different names. Needless to say, this is confusing for users of anti-virus software, trying to reconcile names heard in alerts and media reports with the names used by

their own anti-virus software. To compound the problem, some sites use anti-virus software from multiple different vendors, each of whom may have different names for the same, piece of malware.

Virus Databases

Conceptually, a virus database is a database containing records, one for every known virus. When a virus is detected using a known-virus detection method, one side effect is to produce virus identifier. This virus identifier may not be the virus' name, or even be human-readable, but can be used to index into the virus database and find the record corresponding to the found virus. A virus record will contain all the information that the anti-virus software requires to handle the virus. This may include:

• A printable name for the virus, to display for the user.
• A unique virus hash signature

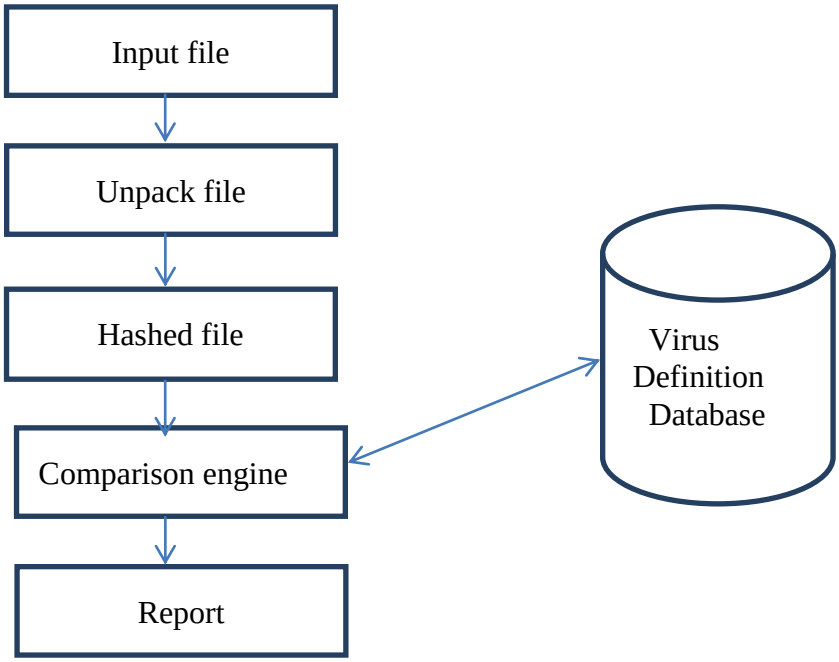


This is the virus signature database, as we can see from the above figure this database consists of two entries the text before the equal character is the malware name and the part after the equal is the MD5 based hashed signature of the corresponding virus.
Malware Name=virus
MD5 Hash Signature=14379AC2B18ECA31088ECD5A3AA58DD8

Any virus signatures stored in the database must be carefully handled. It illustrates a potential problem with virus databases, when more than one anti-virus program is present on a system. If virus signatures are stored in an unencrypted form, then one anti-virus program may declare another vendor's virus database to be infected, because it can find a wealth of virus signatures in the database file! The safest strategy is to encrypt stored virus signatures, and never to decrypt them. Instead, the input data being checked for a signature can be similarly encrypted, and the signature check can compare the encrypted forms.

This virus definition only holds the signature of known malwares only. Whenever new viruses are known then their signatures must be added to this file so that these new viruses will be

detected by the antivirus. As new viruses are discovered, an anti-virus vendor will update their virus database, and all their users will require an updated copy of the virus database in order to be properly protected against the latest threats.



Signature based flow diagram

The above diagram is the block diagram of signature based virus scanner. For every file to be scanned MD5 hash value is computed and then it is compared with the already existing virus signatures. If the signature is found in the database the virus is malicious otherwise the file is sent for further inspection using other malware detection technique that is similarity measure based on the API sequence.

Advantages Of signature Based Virus Detection

- No false positive
  A false positive occurs when a virus scanner erroneously detects a 'virus' in a no infected file. False positives result when the signature used to detect a particular virus is not unique to the virus - i.e. the same signature appears in legitimate, non-infected software.
- No false positive
  A false negative occurs when a virus scanner fails to detect a virus in an infected file. The antivirus scanner may fail to detect the virus because the virus is new and no signature is Yet available, or it may fail to detect because of configuration settings or even faulty signatures.
- Hit Ratio
  A hit ratio occurs when a malware detector detects the malware. This happen because the signature of malware matches with the signatures stored in the signature databases.

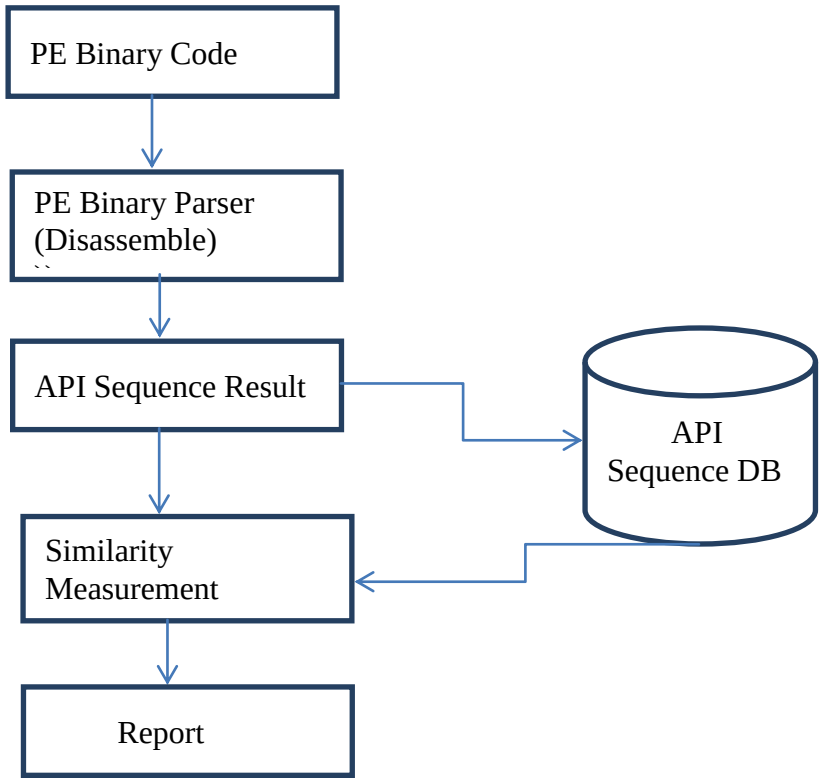
Disadvantages Of signature Based Virus Detection

- Signature extraction and distribution is a complex task.
- The signature generation involves manual intervention and requires strict code analysis.
- The signatures can be easily bypassed as and when new signatures are created.
- The size of signature repository keeps on growing at an alarming rate.

3.1.2 Static anomaly based detection method (Based on API Call sequence)

Anomaly-based detection usually occurs in two phases–a training (learning) phase and detection (monitoring) phase. During the training phase the detector attempts to learn the normal behavior. The detector could be learning the behavior of the host or the PUI or a combination of both during the training phase. A key advantage of anomaly-based detection is its ability to detect zero-day attacks. Similar to zero-day exploits, zero-day attacks are attacks that are previously unknown to the malware detector. The two fundamental limitations of this technique is its high false alarm rate and the complexity involved in deter mining what features should be learned in the training phase.

In static anomaly-based detection, characteristics about the file structure of the program under inspection are used to detect malicious code. A key advantage of static anomaly-based detection is that its use may make it possible to detect malware without having to allow the malware carrying program execute on the host system.



Static anomaly based flow diagram

Following steps are adopted to detect malicious nature of program.

Step1: Program executable is decompressed (optional) if the program is compressed.
Step2: Decompressed program is disassembled using the disassembler module.
Step3: Each disassembled program is represented as a vector of functions. Each function is represented as array of equal length.
Step 4: The similarities between the functions of program P' and P'' is computed using cosine similarity measure.
Step 5: The value of the similarity is compared with the threshold value; if the value is very less than the threshold value then the program under inspection is benign otherwise malicious.
Note the choice of similarity is crucial. A high value of threshold increases the risk of false negative and low value increases the risk of false positiveness.

Similarity Analysis

Malware signatures are long and similarity analysis between signatures of different samples takes more number of comparisons. Thus there exists need of shorter signatures. Similarity analysis based on API sequence is used to detect polymorphic variants .Similarities between the files are checked to find whether the variant is the child of the sample under inspection. Similarity analysis using Euclidean normal form can be used to find the distance between some vectors x and y as:

Cosine similarity is literally the angular difference between two vectors. Cosine Similarity is expressed by the formula:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

For those of you allergic that doesn't speak "mad mathematician" (like me), to calculate the cosine similarity, we need to:

1. Take the dot product of vectors A and B.
2. Calculate the magnitude of Vector A.
3. Calculate the magnitude of Vector B.
4. Multiple the magnitudes of A and B.
5. Divide the dot product of A and B by the product of the magnitudes of A and B.

The result of this calculation will always be a value between 0 and 1, where 0 means 0% similar, and the 1 means 100% similar.

A program is represented as some number of functions f, and each function contains some number of statements which are termed as vectors x and y. The total number of vectors for the

same program P and for all function f is kept same. Similarity analysis can be performed by using cosine similarity measure

3.2.User Interface Design

User interface part is the interactive part of the antivirus and is the front end communicator with end users.

Our software has a series of form interfaces for the user to interact with the system and they are given below:

1. Main form
2. Scanner tab
   2.1 Scanner
   2.2 Current report
   2.3 Result
3. Tools tab
4. Quarantine tab
5. Setting tab
6. Update tab
7. About tab

3.2.1 Main form

This is the main form of the antivirus software whenever the user opens the software this is the first page displayed. It contains various tabs links to all functionalities of the antivirus such as scanner tab, current report tab,and resulttab, settingtab, toolstab, quarantine tab and update tab. This form is the parent of all forms of the antivirus which enables user just to open any form with in the same window.

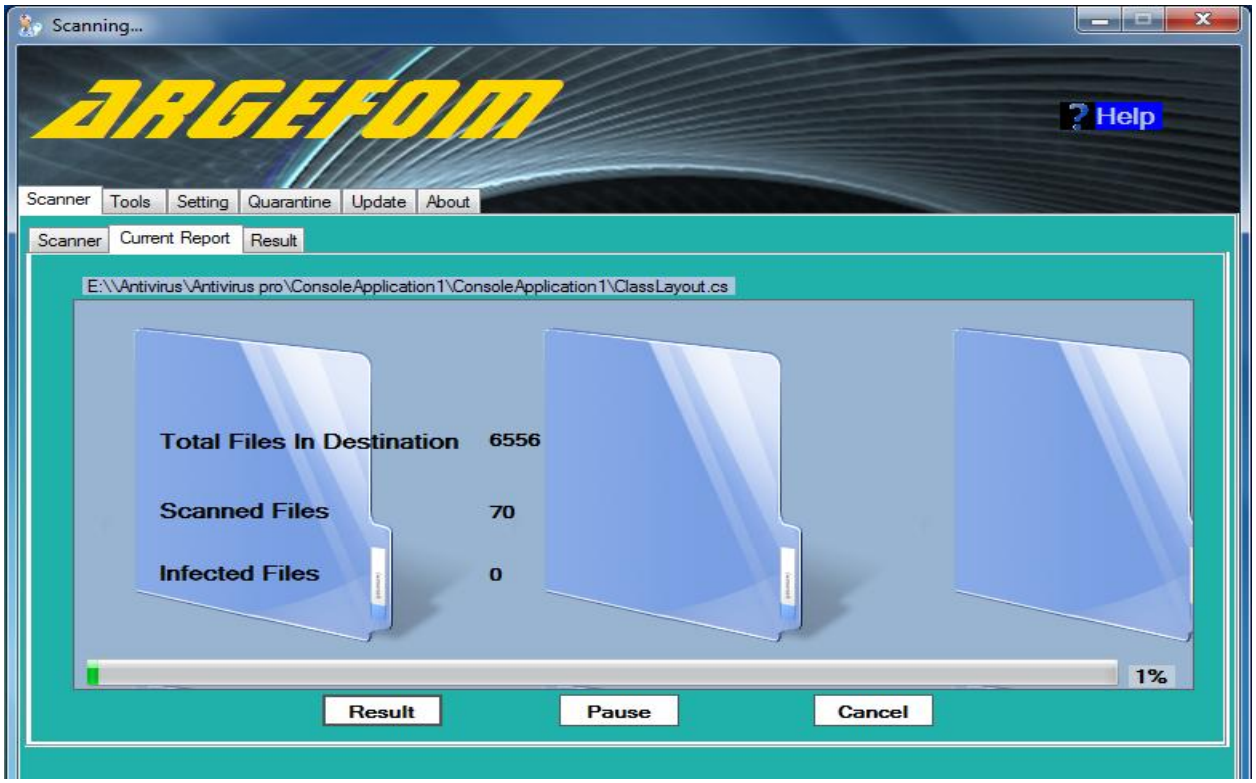
3.2.2 Scanner Tab

This tab displays a page which contains all the system drives and their sub directories as well as any removable disks attached to the user computer. The scanner tab enables user to select the directories or files to be scanned. After selecting user can users can decide whether they want custom scan for a specific directories or full scan for the entire computer drives. These choices are made easy through radio buttons for full and custom scan. Whenever users open the scanner tab they can see the icons of newly inserted removable disks during their arrival. After pressing the scan button users are forwarded to the current report tab which is described below.

3.2.3 Current Report Tab

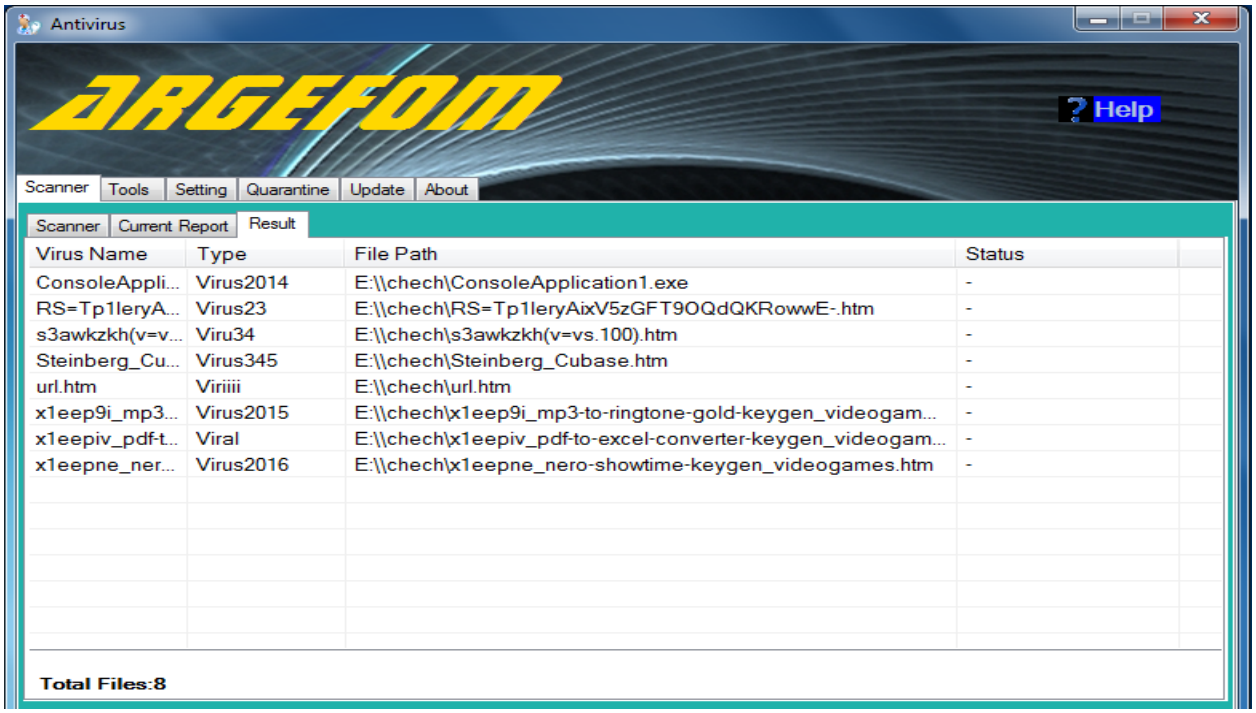
The current report tab monitors what is going on during scanning process. These contains the number of files which are at the destination file being scanned , the number files scanned ,number of files detected and the current percentage of files scanned yet. The file name of the currently scanned file is also shown to the user. Scanning progress is shown to users using progress bar and the percentage of files scanned. The report tab also enables users to pause the scanning process so that they can resume the scanning later. It also enables users to cancel the

scanning process as well as to see the current results of the detected files. After finishing scanning when viruses are found the control goes to the results tab otherwise the control goes to the scanner tab.



3.2.4 Result Tab

The result tab have a list view which holds the information of the currently detected viruses this information contains full path of the virus , virus type virus name and the virus status which indicates weather the virus deleted or cleaned. The user can select any number of detected viruses from the list and can delete or clean the virus files.

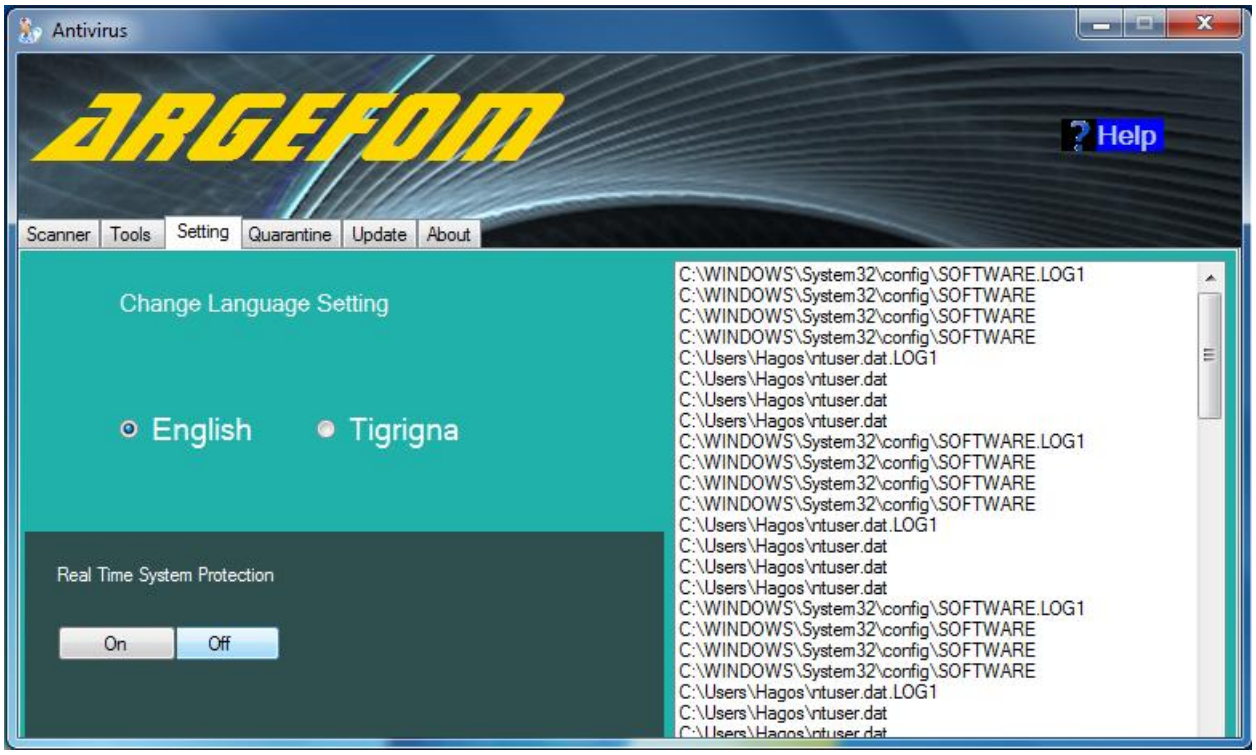


3.2.5 Tools Tab

This tab enables users to open various tools easily without to navigate somewhere. Some of the utilities are like task manager and registry editor.

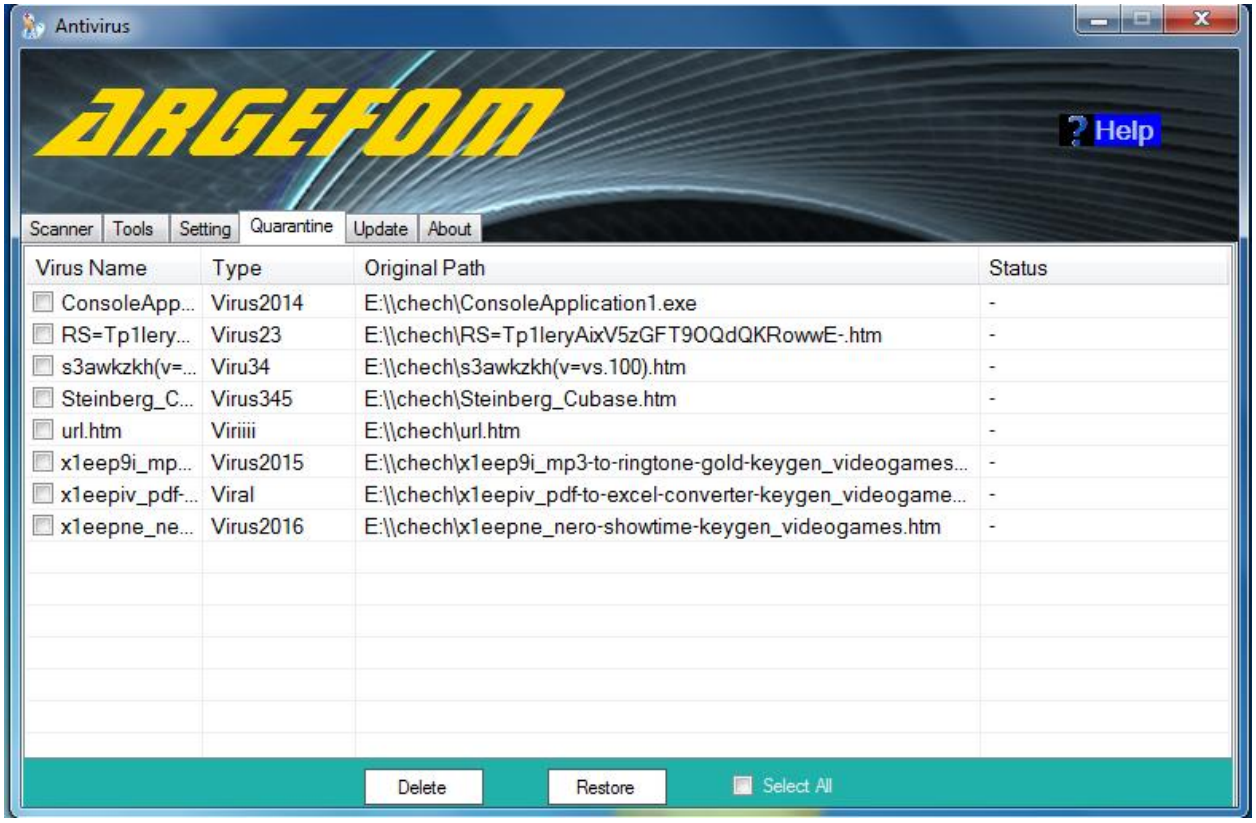
3.2.6 Setting Tab

Setting tab provides different setting for the antivirus software such as language setting which has two choices English and Tigrigna. In addition to these settings there is also setting for real time scanning which makes the real time on or off. The real time scanning when enabled whenever files are modified or created or opened will be automatically scanned for malicious activity.



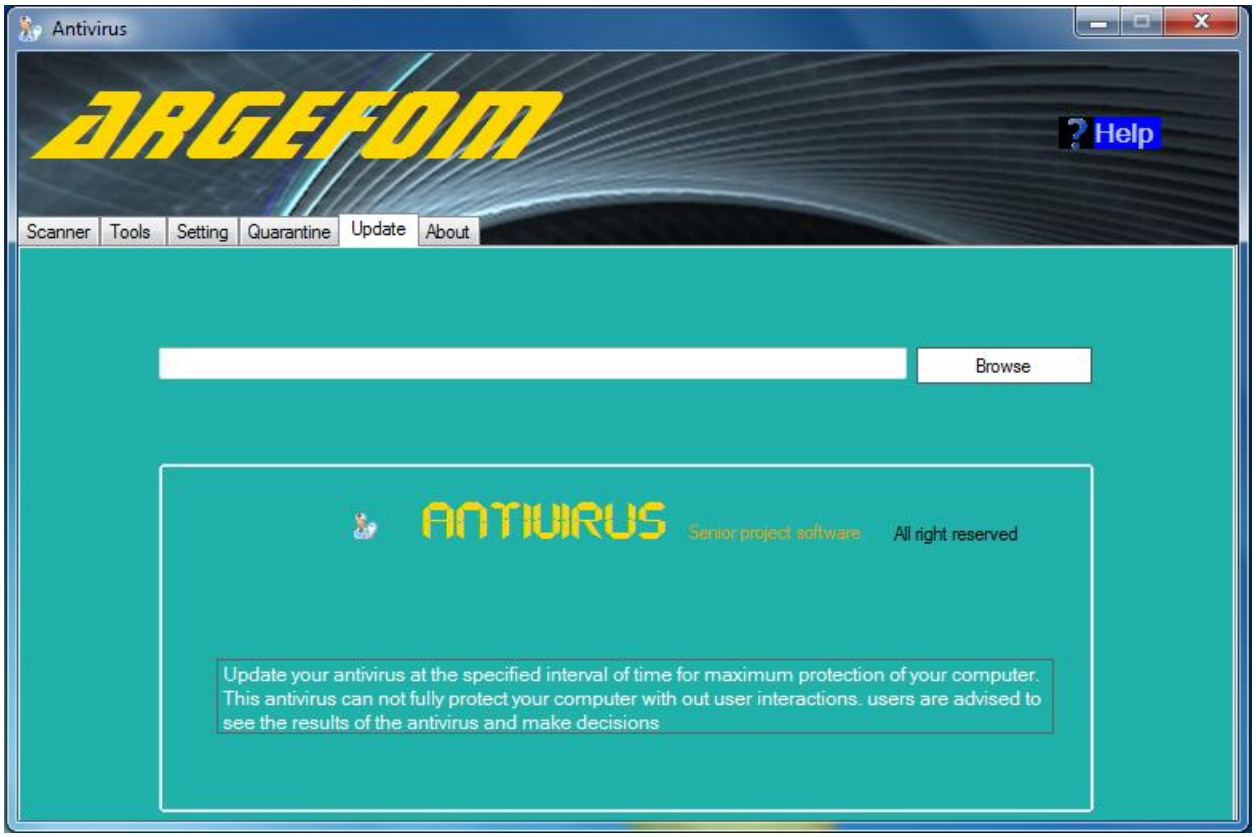
3.2.7 Quarantine Tab

The quarantine tab holds the details of quarantined files which contains virus name, virus type, virus original path and virus status that indicates whether the virus is deleted or restored from the quarantine folder. Users can select files and choose to delete the file permanently from the quarantine folder or to restore the quarantined file to its original location. During scanning all files which are detected as virus are moved to the quarantine folder so that users can decide what to do with them.



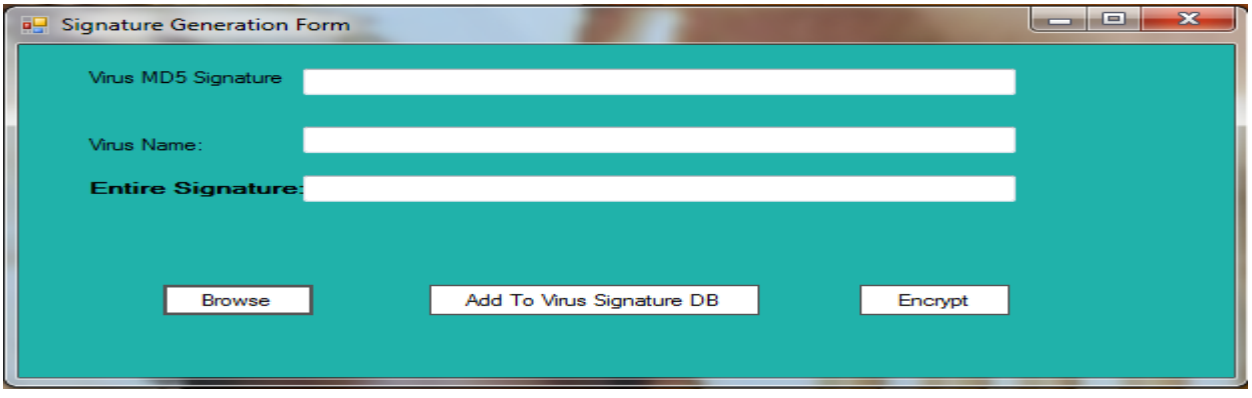
3.2.8 Update Tab

This tab displays a page which enables users to update the antivirus signature database. Users first need to get the update and then locate the file then just click update button. Updating keeps the antivirus database up to date by including the signatures of the newly known viruses.



Generate Signature Form

Actually this form is not part the antivirus software released to users. It is used by the antivirus developers to generate a signature for known viruses so that an update of these viruses can be provided to the users for maximum protection of their computers. This form is completely handled by the developers.



CHAPTER-4

SYSTEM DEVELOPMENT

4.1. Descriptions and Definition of System Components:

This defines all the components that make up the entire system this includes defining all the user interfaces and the algorithm to interact with them. At this phase the algorithm of all components is specified.

4.2. Algorithm and their functions

4.2.1 Scanning Algorithm

4.2.1.1 Custom Scan

1. If drive/folder/file is selected
2. Click Scan button go to current report tab
3. If a virus is detected

Then move it to the quarantine.

4. If result button is pressed display the results tab
5. Else if pause button is pressed

Stop scanning for time being.

6. Else if cancel button is clicked

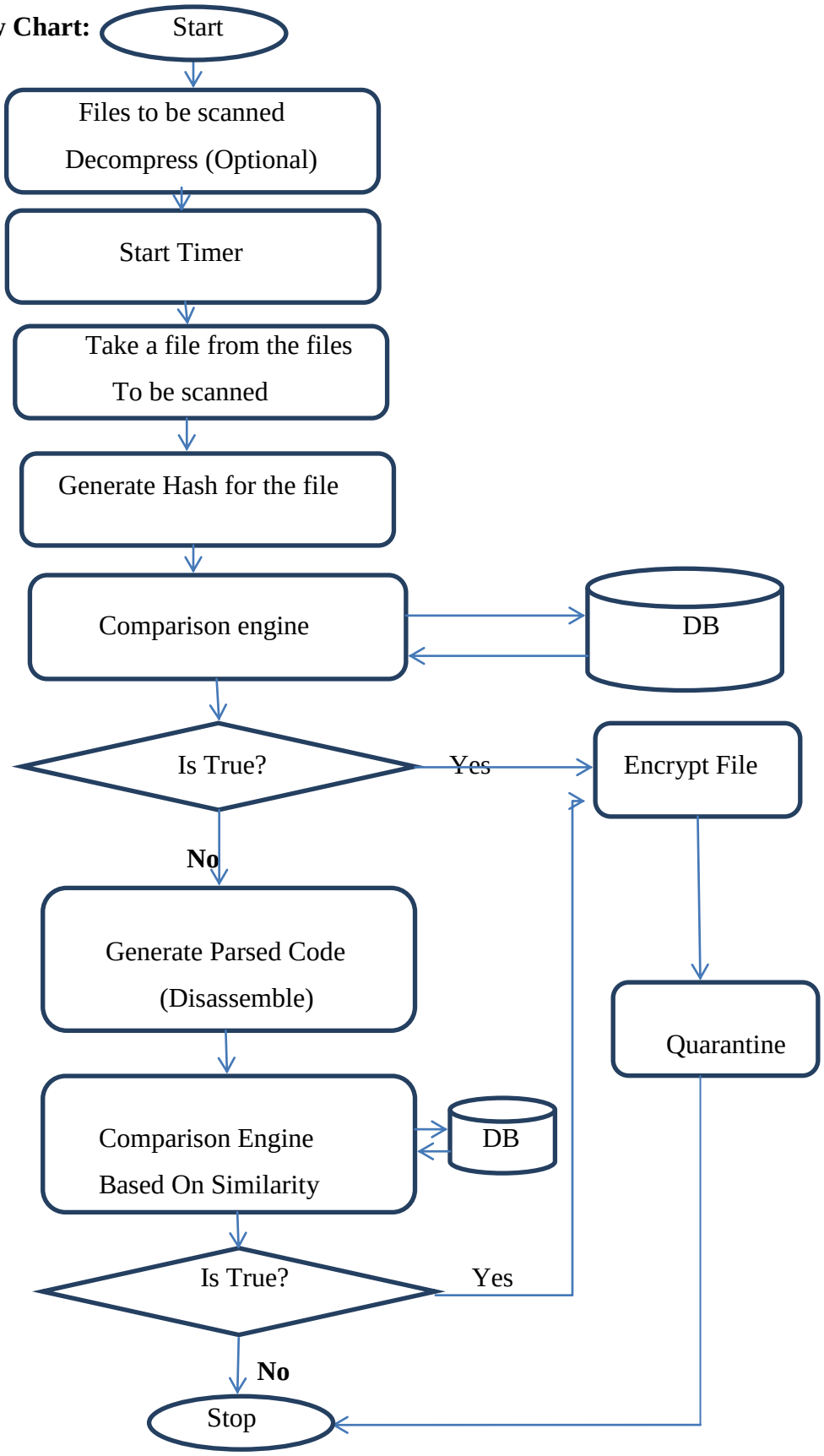
Stop scanning process and display the scanner tab.

7. Else if no drive is selected

Display error message “select a drive to be scanned”

8. Stop.

Scanning Flow Chart:



4.2.1.2 Full Scan

1. Click Scan button then Got current report tab
2. if result button is pressed Got results tab
3. Else if pause button is pressed stop scanning for time being.
4. Else if cancel button is clicked Stop scanning process and Got the scanner tab.
5. Stop

4.2.1.3 on Access Scan

1. If a flash drive/cd disk/removable hard disk is attached to the computer
2. Automatically start scanning and direct the user to the scanning progress.
3. If result button is pressed go to the results tab
4. Else if pause button is pressed

   Stop scanning for time being.
5. Else if cancel button is clicked

   Stop scanning process and go to the scanner tab.
6. Stop.

4.2.2 Results Tab

1. Select a file from the list to be cleaned or deleted
2. If clean button is clicked

Move the selected file to the quarantine folder

3. Else if Delete button is clicked

   Delete the selected file permanently.
4. Stop.

4.2.3QuarantineAlgorithm

1. Select a file from the list to be restored or deleted
2. If delete button is clicked

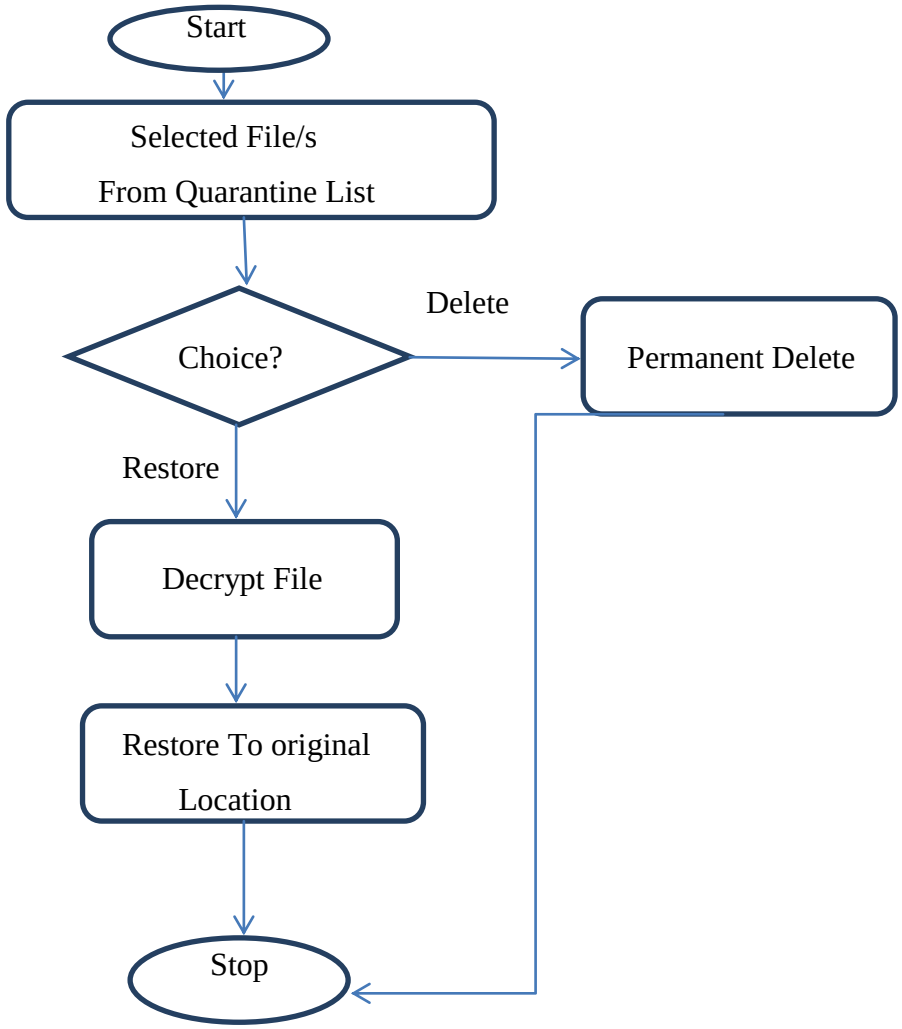
Deletes the selected file from the quarantine folder

3. Else if restore button is clicked

Restore the selected file to its original path

4. Stop.

Flow Chart for Quarantine



4.2.4 Setting Tab

1. If English Radio Button is checked

   Makes all text/labels of the system in English language

2. Else if Tigrigna Radio Button is checked

   Makes all text/labels of the system in English language

3. Stop.

4.2.5 Update Tab

1. Browse and locate the updater file

   Then click the open button of the file browser Dialog Box.

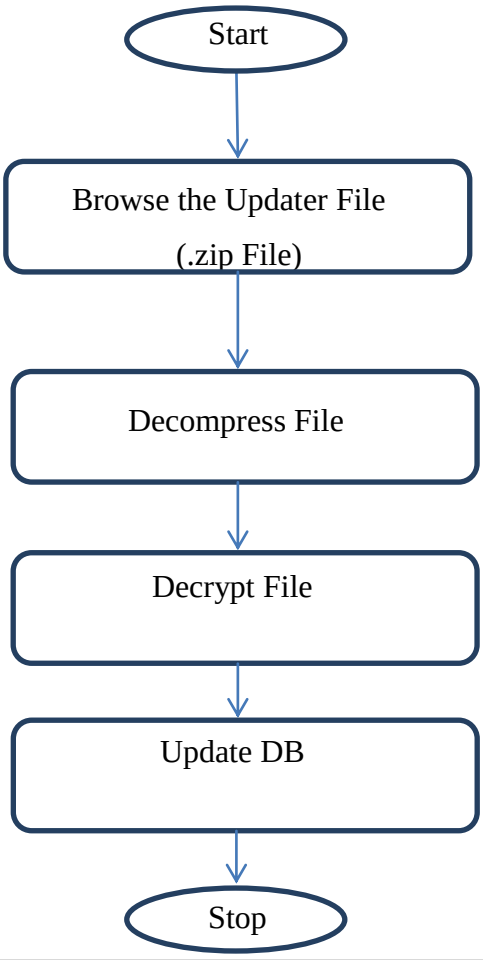
2. If the updater file is corrupted or not found

   Display error message

3. Else update the contents of the file to the signature database.

4. Stop.

Flow Chart for Update



4.2.6 Tools Tab

1. If Task manager button is clicked

   Open task manager.

2. Else if registry editor Button is clicked

   Open registry editor

3. Stop.

4.2.7 Help Tab

1. When help tab is opened

   Display a help file

2. Stop.

4.2.8About Tab

1. When help tab is opened

   Display information about the antivirus software.

2. Stop.

CHAPTER-5

IMPLEMENTATIONAND TESTING

5.1 Testing the System

Software testing is the execution of the software with actual test data. The main objective of software testing is to prove that the software product as a minimum meets a set of established acceptance criteria under a prescribed set of environmental circumstances. There are two components to this objective. The first component is to prove that the requirements specification From which the software was designed is correct. The second component is to prove that the design and coding correctly respond to the requirements. The system is tested using two testing techniques: Black Box testing and White Box Testing.

5.1.1 Black Box Testing

The functionalities of the system components are tested in this testing technique. The testing is Performed by using simple test cases applicable to the individual components.

5.1.2 White Box Testing

This testing is focused on the inner structure of the system. White box testing is a test case design that uses the control structure of the procedural design to derive test case. The tester has knowledge of the back-end, structure and language of the system. The following tests were made on the system:

- All independent paths within the system were exercised at least once. Each possible path was selected and executed. All case structures were checked. The bugs that were prevailing in some part of the code where fixed.
- All logical decisions were checked for both the truth and false values.

- Method coverage is applied to ensure that the defined methods have been triggered. All defined methods in the system are triggered by different events.

Execution-based software testing, especially for large systems, is usually carried out at different levels. In most cases there will be 3–4 levels, or major phases of testing: unit test, integration test, system test, and some type of acceptance test.

A test case in a practical sense is a test-related item which contains the following information:

1. A set of test inputs. These are data items received from an external source by the code under test. The external source can be hardware, software, or human.

2. Execution conditions. These are conditions required for running the test, for example, a certain state of a database, or a configuration of a hardware device.

3. Expected outputs. These are the specified results to be produced by the code under

5.2 Unit Testing

In unit test a single component is tested. A principal goal is to detect functional and structural defects in the unit. Test cases should be developed, using both white and black box test design strategies. The unit should be tested by an independent tester (someone other than the developer) and the test.

5.2.1Generate Hash Component

Test case1:

Input: C:\Downloads\odbg110\ollydbg.exe (not a Virus)

Execution conditions: The state of the device must be ready.

Expected outputs: Hash Output=BD3ABB4AC01DA6EDB30006CC55953BE8

Test case2:

Input: E:\Downloads\Newfolder.exe (Virus)

Expected outputs: Hash Output= A52801B0B77077D321DB749D04D9353D

5.2.2 Signature Based detector

Test Case1:

Input: The Above Hash Value of the file C:\Downloads\odbg110\ollydbg.exe

Output: Not malicious

Test Case2:

Input: Hash Value of the file autorun.ini

Output: Malicious

5.2.3 Similarity Based detector

Test Case1:

Input: any PE file (.exe or .dll) C:\Downloads\odbg110\ollydbg.exe

Output: not malicious

Test Case2:

Input: any PE files (.exe or .dll) H:\updater.exe

Output: malicious

5.3 Integration Testing

Integration test for procedural code has two major goals:

(i) To detect defects that occurs on the interfaces of units;

(ii) To assemble the individual units into working subsystems and finally a complete system that is ready for system test.

In integration testing the interfaces between units are more adequately tested during integration test when each unit is finally connected to a full and working implementation of those units it calls, and those that call it. As a consequence of this assembly or integration process, software subsystems and finally a completed system is put together during the integration test. The completed system is then ready for system testing.

5.3.1 Signature Based and Similarity Based Detector

Test Case1:

Input: C:\Downloads\odbg110\ollydbg.exe

Output: not malicious

Test Case2:

Input: H:/updater.exe

Output: Malicious (by similarity based detector)

Test Case3:

Input: H:/Newfolder.exe

Output: Malicious (by signature based detector)

5.4 System Testing

When integration tests are completed, a software system has been assembled and its major subsystems have been tested. At this point the developers/testers begin to test it as a whole. The goal is to ensure that the system performs according to its requirements. System test evaluates both functional behavior and quality requirements such as reliability, usability, performance and security. System test planning should begin at the requirements phase with the development of a master test plan and requirements-based (black box) tests.

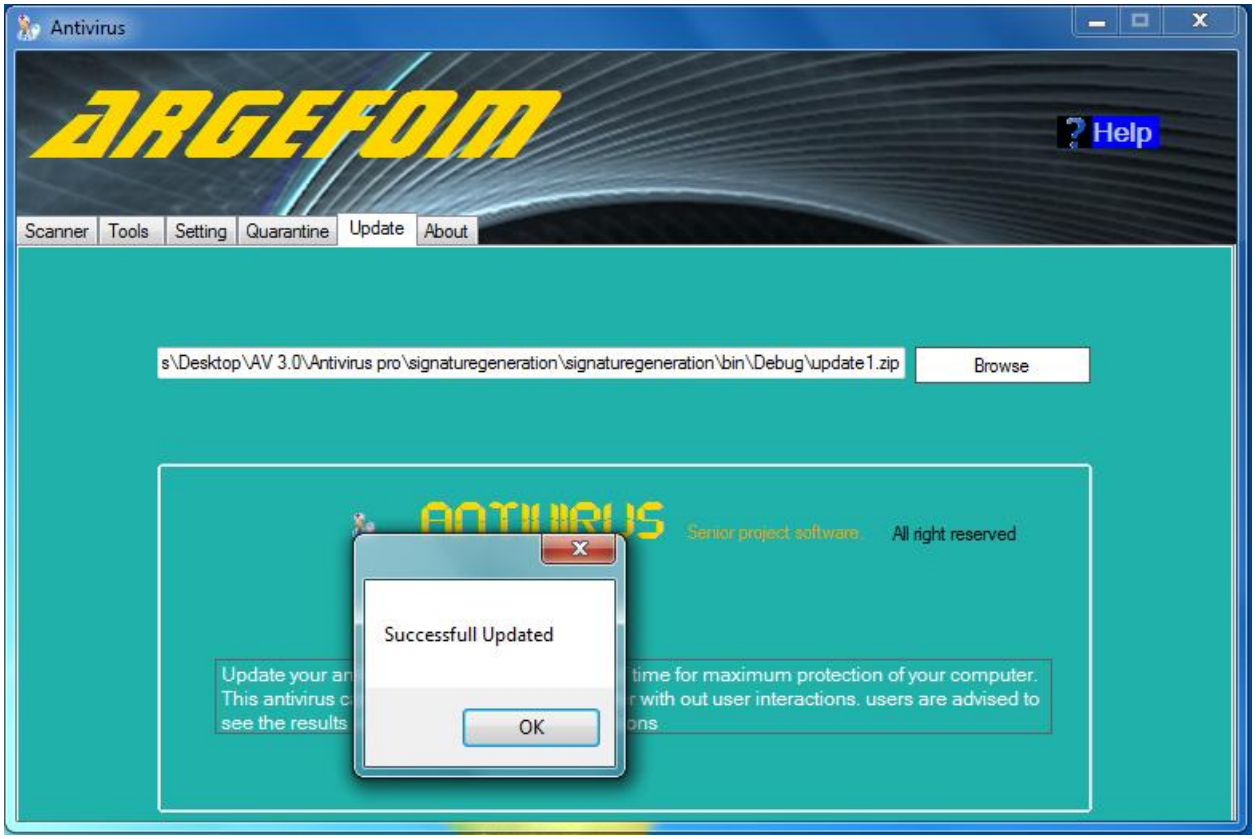
5.4.1Updater component

Test Case 1

Input: updater File (E:\updater1.zip)

Execution Condition:

Output: Update Successful.

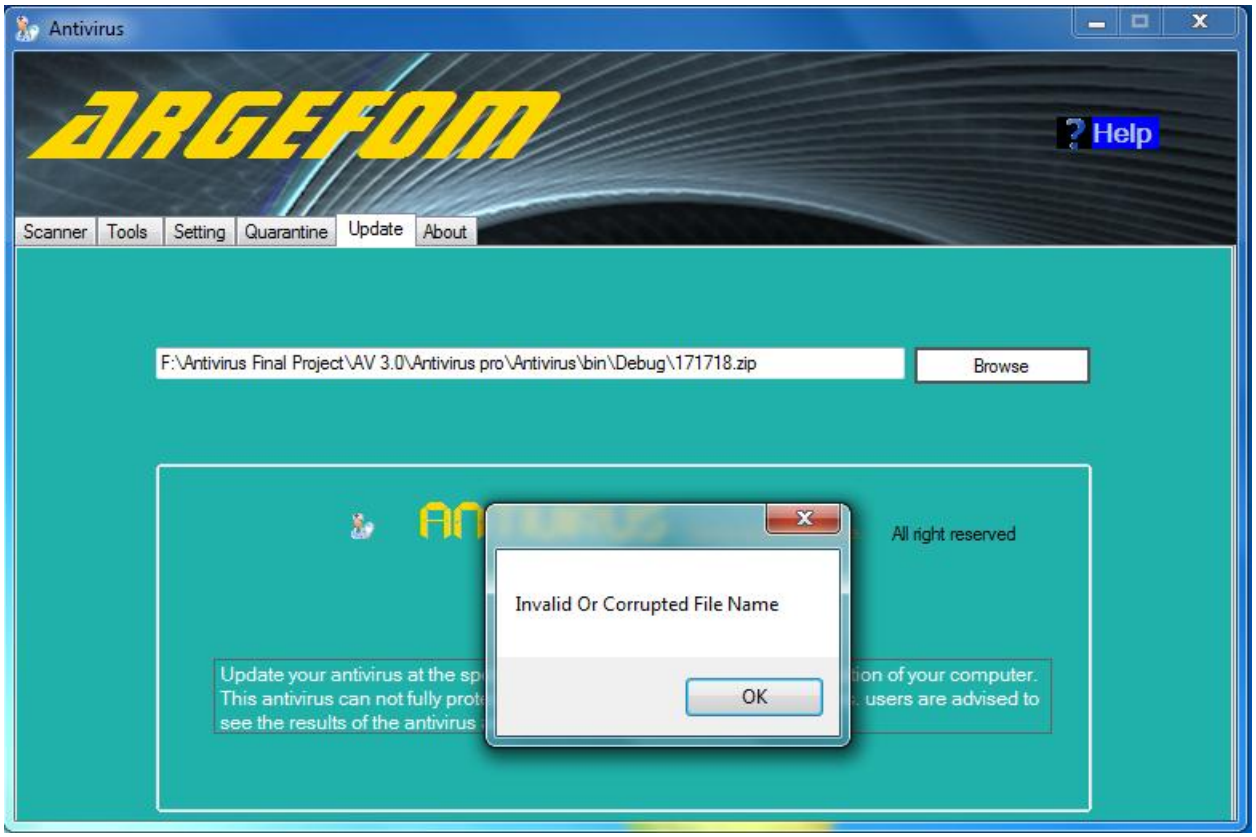


Test Case 2

Input: Updater File (E:\test.zip)

Execution Condition;

Output: Invalid or Corrupted File



5.4.2 While selecting Drive to Scan

Test Case 1

Input: No Selected Drive

Execution Condition:

Output: Message=>please select a drive to be scanned

5.5 System Implementation

Implementation is the stage in the project where the theoretical design is turned into a working System. The implementation phase involves the construction, installation and operation of the new system. The most crucial stage in achieving a new successful system is that it will work efficiently and effectively. There are several activities involved while implementing a new project, they are,

- Educate the end users
- Training on the Application Software
- Post implementation Review

Educate the end users:

The education of the end user starts after the implementation and testing is over. If the system is found to be more difficult to understand than expected, more effort is put to educate the end users to make them aware of the system, giving them lectures about the new system and providing them with necessary documents and materials about how the system works.

Training on the application software

After providing the basic training on computer awareness, the users will have to thoroughly train on the new system. They should understand the event flows, the data to be entered, and the type of errors that occur while performing invalid operation, the corresponding validation checks at each entry and the ways to overcome the problems. It should then cover information needed by the specific user or group to use the system. The users will be provided with the necessary training on the new technology so as to familiarize them with all the functions that the system provides.

.

5.6 Evaluation of System Efficiency:

When evaluating a developed software, the software should pass through a series of evaluation processes with an approach of best quality in mind, moreover it should describe the mechanisms/procedures for software evaluation e.g. establish client/user needs, establish software capabilities and match. In other direction when evaluating we had to understand the need for establishing evaluation criteria, to include;

i. Agreed problem specification: in our case it is already set when we took the project and it is completed as proposed.
ii. Functionality: the software it's required operation and this has been verified.

iii. Usability and human –machine: we have made it as much as user friendly and easy to use.

iv. Compatibility with existing software base: it is compatible with the so proposed requirements and systems.

v. User support: considering this we have put the documentation of the software in the software's help.

5.7 Conclusion

Malware is posing a threat to user's computer systems in terms of stealing personal and private information, corrupting or disabling our security systems. The system uses static signature based and similarity based virus detection. Signature based detection compares the MD5 hash value of a virus with a database of hash signatures of known viruses. This is the best technique that ensures the hit ratio. The similarity based on PE parsed code enables to detect the variants of a polymorphic virus. This technique ensures from dramatically increase the size of database. However, it doesn't ensure the hit ratio as the signature based technique.

This software is designed to perform elementary functions for detecting malware. It is user friendly and easily understandable but very limited at the moment. Its design needs some improvements but also with the developer's programming experience.

5.8 Future Enhancement

This antivirus software can be enhanced with additional features mentioned below. Those by adding these functionalities the software will be more efficient and effective.

1. Making Online Update
2. Automatic Update
3. Adding highly adequate techniques that minimize false positive and false negative problems. These techniques however take more time to implement.
4. Increasing the performance of the scanner
5. Restoring Deleted or modified registry files

Terms or Keywords

PE- Portable Executable File
DLL- dynamic Link Library
DB- Database
API- Application Programming Interface
COM- Common Object Model

References

Aycock - Computer Viruses and Malware (Springer, 2006)

Szor - The Art of Computer Virus Research and Defense

Mark Ludwig-THE LITTLE BLACK BOOK OF COMPUTER VIRUS

http://hooked-on-mnemonics.blogspot.com/2011/01/intro-to-creating-anti-virus-signatures.html

http://security.stackexchange.com/questions/30362/how-do-antiviruses-scan-for-thousands-of-malware-signatures-in-a-short-time

http://sourceforge.net/projects/clamav

http://wikipedia.org

http://www.gettingcirrius.com/

Table of Contents

C H A P T E R- 1 1

INTRODUCTION 1

1.1 Problem definition 1

1.2 Existing System Description 10

1.2.1 Drawbacks 14

1.3 Proposed System Description 14

1.3.1 Advantages/Features 14

CHAPTER-2 15

SYSTEM STUDY 15

2.1 Product specification 15

2.1.1 System objectives 17

2.2-System use case diagram 18

2.2 System requirements 18

2.3 Scope of the system 19

2.4 Hardware and software requirements 19

CHAPTER– 3 20

SYSTEM DESIGN 20

3.1Malware Detection Techniques 20

3.1.1 Signature-based detection method 23

3.1.2 Static anomaly based detection method (Based on API Call sequence) 26

3.2.User Interface Design 28

3.2.1 Main form 28

3.2.2 Scanner Tab 29

3.2.3 Current Report Tab 29

3.2.4 Result Tab 30

3.2.5 Tools Tab 31

3.2.6 Setting Tab 32

3.2.7 Quarantine Tab 32

3.2.8 Update Tab 33

CHAPTER-4 35

SYSTEM DEVELOPMENT 35

4.1. Descriptions and Definition of System Components: 35

4.2. Algorithm and their functions 35

4.2.1 Scanning Algorithm 35

4.2.2 Results Tab 37

4.2.3QuarantineAlgorithm 38

4.2.4 Setting Tab 39

4.2.5 Update Tab 39

4.2.6 Tools Tab 40

4.2.7 Help Tab 40

4.2.8About Tab 40

CHAPTER-5 41

IMPLEMENTATIONAND TESTING 41

5.1 Testing the System 41

5.1.1 Black Box Testing 41

5.1.2 White Box Testing 41

5.2 Unit Testing 42

5.3 Integration Testing 43
5.4 System Testing 44
5.5 System Implementation 46
5.6 Evaluation of System Efficiency: 47
5.7 Conclusion 48
5.8 Future Enhancement 48
Terms or Keywords 49