Virus Detection Techniques and Their Limitations

Anita Thengade, Aishwarya Khaire, Devaj Mitra, Alok Goyal

Abstract— There are a variety of viruses out there infecting computers in newer ways that are difficult to trace. The most common way of detecting a virus is waiting for the virus to spread its reach on many computers, recognizing its mechanism and then designing a method to fight against that specific virus. However, this means that we can only devise a solution after a number of computers have been infected. Moreover, the metamorphic viruses are designed such that they are capable of changing their structure after every attack. Static signature detection can do little against viruses of these types. In this paper, we explore the different types of viruses so as to better equip ourselves to determine methods to detect them. We then study various virus detection techniques and shed light on the limitations of these techniques.

Index Terms— Virus, Virus Detection, Static Signatures, Metamorphic Viruses, Heuristic Analysis, Integrity Check, Dynamic Detection, Static Detection

1 INTRODUCTION

COMPUTER viruses are malicious bits of code that execute on the users system and result in leakage of confidential information or damage to the software running on the users machine. Viruses can also be used to create a network of botnets- malware infected computers that can then be used as tools to host a Denial of Service attack. All in all, viruses pose a huge threat to the functioning of our systems and to our privacy. The current most popular method of virus detection is the recognition of certain static signatures. The anti-virus software is equipped to recognize certain known viruses by checking a files code for these static signatures. If the file possesses such a signature that is known to the anti-virus software as being malicious then the user is warned that the file may pose a threat to the system.

However, not all viruses can be detected as easily by an anti-virus software. The makers of these viruses often update the code of the virus so as to prevent the detection of the virus through static signatures. Once the code is changed, the pre-defined static signatures in the anti-virus program will not match those of the virus and the virus will remain undetected. Updates to the anti-virus software that are capable of recognizing these new signatures are needed. Even then, some viruses are capable of modifying their code after each attack. The matching of static signatures requires that the anti-virus be familiar with the signature of the virus. The addition of new signature templates to the anti-virus software is a slow process. Hence, these self-modifying viruses cannot be detected by our traditional anti-virus software as they keep changing their signatures. This method of writing viruses that are difficult to detect or virus obfuscation can be achieved using a variety of methods. Some of them are encryption, polymorphism and the modern metamorphic techniques[3]. The current virus detection techniques prove to be inadequate to overcome this obfuscation. We need to study the various types of viruses and the existing detection methods if we hope to devise a method that can effectively recognize all types of viruses and face the challenge posed by metamorphic virus signatures.

- *Anita Thengade is currently a professor in Department of Computer Engineering in MIT College of Engineering, Pune, India, PH-091-9960639746 E-mail: anita.thengade@mitcoe.edu.in*
- *Aishwarya Khaire is currently pursuing bachelors degree program in computer engineering in MIT College of Engineering, Pune, India , PH-091-9921104669. E-mail: ashskhaire@gmail.com*
- *Devaj Mitra is currently pursuing bachelors degree program in computer engineering in MIT College of Engineering, Pune, India , PH-091-8983410463. E-mail: devajmitra@gmail.com*
- *Alok Goyal is currently pursuing bachelors degree program in computer engineering in MIT College of Engineering, Pune, India , PH-091-8149253471. E-mail: alokg1019@gmail.com*

2 TYPES OF VIRUSES

The development of various virus detection techniques has also led to the development of more robust types of viruses. A few of them have been listed below.

2.1 Overwriting Virus

These are the most primitive types of viruses that can simply overwrite the local files on the user systems with their own code. In case a file is infected with such a virus it will have to be deleted. By itself, the overwriting virus cannot cause much harm. However, when combined with network-propagation techniques, they tend to prove more lethal. An example could be the VBS/LoveLetter.A@mm virus which is known to mass mail itself to other systems[1]. The other category of overwriting viruses is the

tiny viruses such as the Trivial virus family of DOS. These can be as small as 22 bytes in size. These viruses are not capable of infecting files marked as read-only since that would require certain extra instructions to execute.

2.2 Appending Virus

In these viruses a JMP instruction is placed at the beginning of the host file which points to the end of the file where the virus code is appended. Most files consist of a header section with information about the main entry point of the file. The appending virus replaces this information with its own address. The first three overwritten bytes of the host program are saved in the virus code. When the virus file is executed, it loads itself along with the infected host and then it usually harms the system by replicating itself or executing an activator routine.

2.3 Prepending Virus

This is another simple technique in which the virus body is placed at the beginning of the code in the host file. This technique has proved successful in causing some major outbreaks in many operating systems. An example of this type of virus would be the Hungarian virus Polimer.512.A which is 512 bytes long and fixes itself to the beginning of the affected file and the original data follows[1].

2.4 Encryptor Decryptor Virus

These consist of an encrypted virus code and decryptor routine. The entry point of the host file is replaced with the decryptor's address. When the file executes, the decryptor decodes the encrypted virus code and passes control to this virus. The encryption technique could be as simple as xor of the key with the virus body[3]. The decryptor remains constant in this case.

2.5 Cavity Virus

These viruses avoid increasing the size of host file, thus making it difficult to detect the attack. The virus code overwrites the empty spaces in the host program instead of appending or prepending itself. These viruses are difficult to write and are hence very rare. They typically fill in the zeroes in a binary file but can also overwrite other areas like the instruction alignment code used by C compilers[1].

2.6 Compressing Virus

This technique is used to disguise the host programs increase in size after it has been affected by a virus. The host program is compressed and the virus code is made to fit in maintaining the same size of the host program. This is achieved using a binary packing algorithm. The benefit of using this technique is that it saves disk space.

2.7 Metamorphic Virus

These are the advanced forms of viruses that can modify their decryptor thus making it harder to detect. Not only the decryptor, but the virus body may also change following every attack. However, the behavior of the virus remains the same in spite of changes in its code. The virus achieves this by changing its code to a temporary representation, then editing this representation and then translating back to original format[2]. It can change inlining and outlining, register names, reorder instructions and data, etc. This ability of the virus makes it impossible to detect it using static detection methods.

3 VIRUS DETECTION TECHNIQUES

Various virus detection techniques have been developed over the years, in order to keep up with the different types of known and unknown viruses. Most of these techniques fall under one of the four generalized techniques outlined below.

3.1 Static Signature-based Method(String Scanning)

This is the most effective and common way to identify known viruses. The virus is simply a piece of unwanted code which gets attached to a file. This code will consist of series of instructions for the computer to execute in order for the virus to function. These instructions must be present in exactly the same order in each of the infected file for the virus to do its job. These specific set of instructions are unique to a specific virus and are unlikely to be found anywhere else in a normal program. These instructions or strings of bytes are referred to as "virus signatures"[3]. Most of the antivirus programs look for these virus signatures or strings of bytes in the file they are analyzing for threat. If the virus signature is found in a particular file or target program being analyzed, it is marked as infected. For the antivirus program to be able to recognize a virus, its virus signature must be present in its signature database. The efficiency and effectiveness of an antivirus program depends on its ability to search and match the presence of virus signature in the file from the thousands of virus signatures in its database. Increase in the number of virus signatures in the antivirus database increases the ability of the antivirus to detect different types of viruses. If a particular virus signature is not present in the antivirus database, then that virus cannot be detected by the antivirus program.

For example, if a file contains the string 0400 B801 020E 07BB 0002 33C9 8BD1 419C then the file is infected with Stoned virus [3].

3.1.1 Limitations of Static Signature-based Method

Static signature scanning method cannot work for all kinds of viruses. This method cannot detect unknown viruses whose signature is not present in the signature database of the antivirus software. Further the virus signature databases have to be constantly kept updated in order for the antivirus software to be able to detect new viruses. Extracting a virus signature from the virus requires through analysis and understanding of the virus specimen by the researcher and is a highly skillful job. The researchers must have knowledge about various programming environments and languages for them to be able to understand the

complexities of the modern viruses. Since the virus signature cannot be obtained without analyzing the virus code, the time gap between the virus creation and detection is substantial. By the time the antivirus researchers obtain a virus specimen and roll out an update to the antivirus software, the new virus easily spreads and causes considerable damage without being detected.

Besides, there are certain viruses which are capable of modifying their code in the virus body after each infection. As their code is not static they cannot be detected by signature scanning method.

The virus developers may also update their virus code by reshuffling certain instructions or inserting NOP(no operation) instructions to evade detection from the antivirus software. This mutation of the code invalidates the signatures used by the antivirus software and new signatures have to be added to the antivirus database to detect these modified versions of the same virus, which is again a very slow process[3].

3.2 Generic Signature Scanning/Wildcards

Since static signature scanning method can easily be evaded by modifying certain instructions or rearranging the virus code, therefore generic signature or scanning using wildcards is also used by antivirus scanners to detect viruses. In this method the virus signatures are defined using wildcards which allows the scanner to skip certain bytes and detect all variants of the same virus family[2]. This method works in the same way as regular expressions are used to define and recognize a pattern from the long string. This method is effective in against viruses which may mutate their code slightly or rearrange certain instructions to evade detection. Also since many new viruses are also created by modifying code of previous existing viruses, using wildcard scanning may also help in detecting these new variants.

For example,

0400 B801 020E 07BB ??02 %3 33C9 8BD1 419C

The ' ?? ' denotes the wildcard characters which can be skipped by the scanner while scanning the input file[1].

3.2.1 Limitations of Generic Signature Scanning

Generic methods may detect new viruses but are unable to disinfect the affected files. Again, if the virus does not lie within the scope of the signature database of generic scanning, then it cannot be detected. This happens because wildcards are used to scan signatures of viruses belonging to the same family. A new virus that belongs to the same family may not lie within the scope of the wildcard.[4].

3.3 Heuristic Analysis

Several types of viruses such as encryptor decryptor and metamorphic viruses cannot be detected using signature matching techniques, as the virus signature can keep changing, or become unreadable due to encryption. In such cases, the heuristic analysis technique can be used, which works by observing the behavior of a binary file, either through static analysis of the binary or dynamic analysis, in a virtual environment, and then determining how likely the file is to be a threat.

In the static heuristic analysis, the binary file is reverse engineered to obtain the code. The code is then analyzed and fragments of code that are likely to cause suspicious behavior are identified, by comparing with a database of code fragments that cause harmful effects[4]. Static heuristic analysis is different from signature matching. In signature matching we match the code of the executable with a database of known viruses, and a perfect match gives us the exact name and family of the virus with complete certainty. In static heuristic analysis, we do not test against known virus signatures, but against code fragments which are likely to cause virus like behavior such as replication, metamorphosis, deleting files etc. If the file is likely enough to exhibit virus like behavior, it is flagged and reported.

In dynamic heuristic analysis, a virtual environment is created and the file is executed inside this virtual environment. It is allowed to function as it would normally, and it's behavior is observed in this isolated environment. If the file performs any suspicious actions, such as adding questionable registry entries, replication to other files, formatting the hard disk and such, the file is flagged as a virus and reported. This method is slower than the static analysis, but more likely to detect an unknown virus and less susceptible to false positives, as the file is allowed to run its course of execution and it's exact behavior is observed.

Thus, heuristic analysis makes it possible to probabilistically detect new and unknown viruses, whose exact signatures are not present in the signatures database. Heuristic analysis is required in addition to signature matching not only to detect new virus families, but also to detect self mutating and encrypted viruses.

3.3.1 Limitations of Heuristic Analysis

Although heuristic analysis is useful in detecting viruses which cannot be detected using signature matching, it has some limitations which may limit its usefulness in several practical scenarios.

Static heuristic analysis involves mapping of code of fragments to their runtime behavior. This mapping can become difficult, as there can be large number of ways a particular behavior can be implemented. For example, a program can be terminated using several different ways. In addition, code obfuscation can make a particular implementation of a behavior difficult to detect.

Dynamic heuristic analysis, although accurate and superior to static heuristic analysis in detecting suspicious behavior,

can be very slow. The virus may be activated very late in the execution of the program being observed, which cause detection to be delayed. In addition, the virus may only be activated when in response to a certain external interrupt, such as a user pressing a submitting a form. In such cases, the virus may not be detected at all. Some viruses are activated only in certain external environments, such as on a particular date or time. These viruses too will not be detected by dynamic heuristic analysis.

3.4 Integrity Checking

Viruses use several evasive mechanisms such as obfuscation, mutation, and periodic activation which can make it very difficult to identify a virus signature or detect suspicious behavior by analyzing or emulating a binary executable. Integrity checking detects viruses by checking integrity of a file, by comparing its present fingerprint with its past uninfected fingerprint[5].

Initially, a system is assumed to be uninfected. At this stage the fingerprint of each file on the system is calculated and stored in a secure location on the disk. The fingerprint can be calculated using several different algorithms such as an MD4, MD5, or even a simple CRC32. Just before any binary file is executed on this system, its current fingerprint is calculated using the same algorithm and compared to its initial fingerprint. If the fingerprints match then the file is still uninfected and it is executed normally. If the fingerprints don't match, it denotes that the file has been modified in some way. The file is then flagged and it can then be reported or subjected to further analysis.

Thus, integrity checking does not look for presence of malicious code. Instead, it checks whether the code is uninfected as compared to its initial state.

3.4.1 Limitations of Integrity Checking

Although integrity checking can detect viruses with certainty, it lacks accuracy. Integrity checking technique reports a large number of false positives. A lot of programs change themselves, such as by storing configuration and user data in the same binary file. In such cases, the fingerprint of the binary file will change, and it will be flagged as a virus.

Another major limitation to integrity checking is that the initial state of the files on the system is assumed as uninfected. This may not be true. If a file is already infected with a virus, it will be reported as safe in all further checks[4].

Speed is also a limiting factor in integrity checking. Binaries can be several megabytes big. Computing checksums every time before these files execute can cause delay in execution.

4 CONCLUSION

The advantages and limitations of various virus detection techniques were studied. As no one technique is perfect, more research in the field of virus detection needs to be performed.

REFERENCES

[1] Peter Szor, "The Art of Computer Virus Research and Defense", http://computervirus.uw.hu/ch04lev1sec2.html

[2] Essam Al Daoud, Iqbal H. Jebril and Belal Zaqaibeh, "Computer Virus Strategies and Detection Methods",Int. J. Open Problems Compt. Math., Vol. 1, No. 2, September 2008

[3] Nitesh Kumar Dixit, Lokesh mishra, Mahendra Singh Charan and Bhabesh Kumar Dey, "THE NEW AGE OF COMPUTER VIRUS AND THEIR DETECTION", International Journal of Network Security & Its Applications (IJNSA), Vol.4, No.3, May 2012

[4] Umakant Mishra, "Methods of Virus Detection and their Limitations", http://www.trizsite.com

[5] Peter Szor, "The Art of Computer Virus Research and Defense", http://computervirus.uw.hu/ch11lev1sec11.html