Malware Detection Techniques: A Survey

Y. Supriya[1], Gautam Kumar[2], D. Sowjanya[3], Deepali Yadav[4], D. Lakshmi Kameshwari[5]
CMR Engineering College, Hyderabad, T.S.-501401, INDIA
[1]supriyaysp12@gmail.com, [2]gautam21ujrb@gmail.com, [3]dammusowjanya1256@gmail.com, [4]deepaliy101@gmail.com, [5]dlk58585@gmail.com

Abstract. Malware is derived from malicious software which mitigate to attacks on the computer systems and collecting private data. The survey is available in huge evidences to suggest its impact in global losses. Malware detectors are basic tools to protect from the same malware attacks. Therefore, it is important to require study on malware detection techniques, to avoid and identify the type of malware attacked on systems. In this manuscript, a survey report is available to defend against malware attacks and analysis techniques. There are many malware detection techniques, such as signature and anomaly detection techniques with an idea of comparison and decision making about its strengths. This provides as a user reference to the end user for likely detailed information.

Keywords: ***Malware, Anomaly Based Detection, Signature Based Detection, Hybrid Analysis, Dynamic Analysis***

1. Introduction

A malware detector is used to find the malicious program code present in the system by using some detection techniques. Malware detector protects the system from malicious activity. It's not necessary that malware detector sits on the same system on which it tries to detect the malware. Sometimes it resides on another system also and try to detect the abnormal behavior of the system. By using some detection techniques it checks for malware. Malware detectors take two inputs **[1], [2].** First one is the knowledge to recognize the maliciousness of the program and the other is program which has to be tested. It can use its techniques to identify whether the program is malicious or not. The used techniques for detecting malware can be categorized mainly in two parts such as anomaly-based detection and signature-based detection.

Anomaly detection is again divided into specification-based detection. A specification-based system uses some rule set to determine a valid behavior **[3]**. If behavior is valid then it is termed as un-malicious and normal. If not, then there is some anomaly and it turned as malicious. Signature-based systems used some characterization to determine the malicious of a program [4]. Fig. 1 shows all the possible types detection techniques. Each is further categorized as: static, dynamic, or hybrid. As anomaly and specification is a based malware detection check for malware by gathering information and then analyzing. The static approach uses the syntax and structural based programs to determine the malware. Like in signature-based detection, the static approach looks for the amount of bytes the program is taking, the syntax of the codes to determine the maliciousness in the program whereas the dynamic based approach looks for the run time environment **[5]**, **[6]**. It means if checks for the malicious before execution of the code or may be under execution of the code and the hybrid is the combination of the both static-based and dynamic-based.

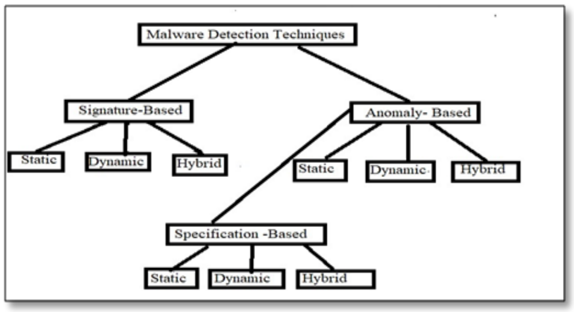

Fig. 1: Classification of malware detection

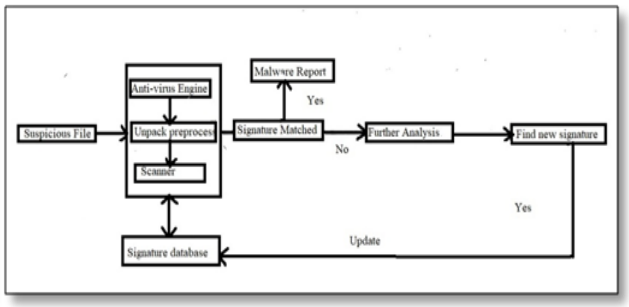

Fig. 2: Flow of signature based detection

The remaining part will describe about the anomaly-based detections with some examples and followed by specification-based and signature based.

1.1 Signature Based Detection

It maintains the database of signature (signature is nothing but a series of bits that make a unique number which is used to detect fingerprints and identify specific viruses) detects malware by comparing pattern with the database. Mostly Antivirus software is designed by using this technique. It is also called as Misuse detection. All files are translated to high level language code to find the signatures and these signatures are stored in signature database. Then this is code is examined and some of the characteristics are taken out. Then these characteristics are used to construct the signature of a malware. When the antivirus software is installed in a computer then signature of a familiar code is updated and refreshed continuously so that this it can detect the known occurrences of malware correctly. Its advantage can identify the known occurrences of malware correctly and it takes the less number of resources to identify the malware. The disadvantage is that it cannot detect the new and unknown malware instances because we cannot find the signatures for those types of malware.

The suspicious file is sent to the Anti-virus Engine as shown in Fig. 2, then it checks whether the file is packed or unpacked, if the file is packed then it unpacks the file and send that file to scanner otherwise if the file is unpacked then it directly sends that file to scanner without making any changes. This unpacking is a built in function in Antivirus engine. Antivirus engine asses the code which is in assembler language (human readable form) of a file with the malware signature database. If the file signature or a pattern is matches with the database then that file is considered as harmful, then a report is generated [7]. So this is the process of information flow of signature based detection. It is having the following types of Signature based detection:

- Static Analysis of Signature based detection
- Dynamic Analysis of Signature based detection
- Hybrid Analysis of Signature based detection

1.1.1 Static Analysis of Signature based malware detection

In this the program is examined for the some sequence of code so that it would give information about the main aim of malicious program. It is used to give the approximate run time behavior of a program. It uses its knowledge and compares the program with the known signatures to find whether the program is malicious or not. Mostly the malware is spread in the form of binary code which is used to provide the information about a program. This is having the following a**dvantages:**

- More Time and Resource Consumption
- Global View
- Easily Accessible Form
- Stable and Repeatable
- Safety and Data Independent
- Limitation of Software Reverse-engineering (copying) Techniques
- Susceptible to Inaccuracies due to obfuscation and polymorphic Techniques
- Conservative Approximation

1.1.2 Dynamic Analysis of Signature based detection

In this detection process, the malware is detected by using specialized monitoring mechanisms. The malware is identified by API calls which are made by executable. Application Programming Interface is a communicating layer between Windows environment and executable. Because of some unclear techniques which are used in malware we cannot separate the part of a program but there is a chance that we can separate a part of program by using API calls in following ways they are:

- **Data pre-processing:** In this method, the programs are executing use the API calls. To trace the API calls the software was created which monitors and displays API calls and their services were used. The output of this method is given as file which contains the list of API calls.
- **Signature generation by API call tracing:** In this method, the Signature for a malware is generated by using critical API call. The signature is based on API program call that can be represented as a components, such as critical API calls and frequency call.
- **Determination of the degree of membership of malicious program to a malware class:** It represents the relationship between the malware program and one of the malware classes in form of the number of critical API calls. It will allow separating the malware programs within the class.
- **Detection of a malicious program based on API call tracing:** To detect the malicious program, we need to find the difference between the frequencies of the critical API calls of a program and the frequency of critical API calls for each class.
- **Rule based IDS Approach:** In this we use the state transition diagrams to know the attacks that are done by malicious code. The data is send to the pre-processor which changes the data format so that it can be analyzed by transition diagrams and this data is compared with state diagrams.
- **Behavioural Approach to Worm Detection:** In this we use different kind of behavioural signatures. One is base signature and this base signature can be identified by observing the data flow from inside and outside of a node.

The advantages of the same are:

- **Effectiveness and precision:** It uses the real values during run time so it gives the effective results and it gives exact runtime behavior of code.
- **Simplicity**: Only single path execution is considered in this dynamic analysis so it is simpler than static analysis.
- **Runtime Behavioral Information:** The malware is assessed by using the runtime information given by code during dynamic analysis.
- **No Unpacking:** In this unpacked code is executed automatically and runs the code .so no unpacking is done in this method it directly executes the code.
- **Robust to Code Obfuscation:** In this the obfuscated (unclear) code does not make any changes in the behavioral information which is collected during the execution of a code.

In the similar, this one is also having disadvantages, which are as:

- **Limited View of Malware:** It is not possible to check all execution paths and variable values during dynamic analysis so because of this reason it might not allow to view of the total malware
- **Trace Dependence and Execution Time Period:** The main disadvantage of dynamic analysis is that it is trace dependent. The malicious behavior‘s cannot be collected in dynamic analysis within the given period of time.
- **Lack of Interactive Behavior Information:** malware runs automatically without any human acknowledgement or interaction, so it is lack of interaction with human beings.
- **Time and Resource Consumption**: In Dynamic analysis when we want to analyze a huge amount malware files it takes a lot of time and many resources are required for it.

1.1.3 Hybrid Analysis of Signature based detection: The detection in Hybrid Analysis is a mixed observation in dynamic and static behavior. It will combine run time data which is taken from dynamic analysis and static analysis to identify the malicious functionality in the applications. In general, the hybrid analysis tool are as:

- **Mobile Sandbox:** This uses analysis of APK file with the static analysis, then with the user permission the anti viruses takes the user control and send manifest.xml file in recognition to the suspicious code under the scanned. On the other hand, the dynamic analysis uses an emulator to run and check with the behavior of code and work with the suspicious code. This sandbox is used as a security mechanism in order to separate, of running programs, in untrusted programs execution and/or untrusted programs in no cause to any harm in the system..
- **Andrubis:** The framework for Andrubis is to perform the dynamic analysis and in the same the static analysis results are used too. As this one is representing static is used, therefore it is mandatory to perform static analysis first and then dynamic analysis applied. The framework is focused on android manifest.xml file and byte code.

1.2 ANOMALY - BASED DETECTION

Anomaly detection process mainly consists of two phases - a learning (training) phase and a monitor (detection) phase. In the learning phase the detector tries to learn the normal behaviour of the system when encounters by a operation. The detector can be trained by the behaviour of the host or the PUI (Potentially Unwanted Installations (spyware)) or both. An anomaly based detection advantage in is in the possession to find detect the zero-day attacks.

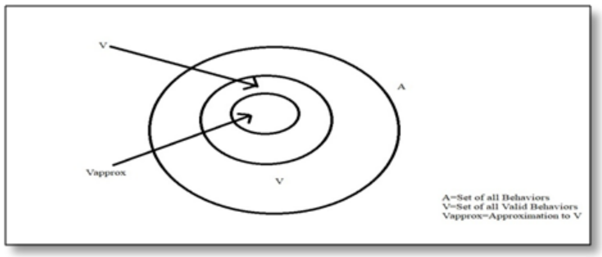

Fig. 3: Behavior characterization of anomaly based detection

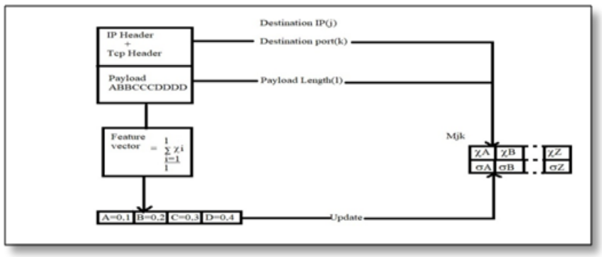

Fig. 4: PAYL Architecture

The term zero days refer to the frangibility itself, or an attack that has zero days between the time the frangibility is happened and the first attack. The two fundamentals drawback are the complexion involved in the training phase to distinguish normal behavior & abnormal, and having of the high false rate. Fig. 3 will tell us that why anomalies system is alone not good for malware detection. As shown in fig. 3, V is the set of valid behaviors of a system, and V’ is the set of invalid behaviours. Anomaly-based detection attempts to approximate the execution. The execution to valid behaviours is set as V approx which is shown in the above figure. Valid behaviour can be changed as malicious because V approx is a conjecture. For example, if estimation is never encountered in learning phase, an estimation seen during the monitoring phase gives a false alarm. It increases the high false rate with this detection technique. The count of unseen behaviour in anomaly is not zero. Therefore, the probability of giving a *false positive* is not zero.

1.2.1 Dynamic Anomaly -Based Detection Process

In this detection process, data collected from the execution of program is used to perceive malicious code. Firstly the Inspection phase scans the program under inspection during its implementation, checking for maliciousness with what the detector has understood in the training phase of PAYL, Fig. 4. In reference **[8]** presented PAYL which is the most prominent payload based anomaly based network intrusion detection system (**NIDS).** NIDS is a detection in hacking activities with the system, denial-of-service (DoS) attacks, and detection in port scans in computer networking, which calculates the expected payload on each service port of a system. PAYL works in 2- step, first packets are classified according to the payload length then an n-gram analysis is applied to payload. The working of PAYL is frequency on byte distribution, that works for centroid model, and calculates the *Mahalanobis* distance between them. The Mahalanobis distance [9] **[**https://www.sciencedirect.com/topics/engineering/mahalanobis-distance**]** recognizes mean values of a feature vector, variance and co-variance resulting a strong statistical measure of similarity. So, if we have large Mahalanobis distance value i.e. the incoming payload is having a large distance from centroid model then the incoming payload is malicious.

1.2.1.1 Data Mining Approaches for Intrusion Detectors: *Intrusion detection*: An **intrusion detection** system (IDS) is software application that keep track on a network for malicious activity or policy violations. Lee and Stolfo [**10]** suggested the use of intrusion detection systems with frequent event and association rules. Meta detection used base detection and output of intrusions. On this audit data detection system execute the behavior. All authors use tcp dump data to learn the normal behavior. Through manual assessment of normal and abnormal data, they can identify that it can detect the possibility of abnormal activity.

1.2.1.2 Short Sequences of System Calls: Hofmeyr et. al in **[**11**]** advance a technique that inspects system call sequences to recognize maliciousness in the system. We need to make profiles to show the normal behavior. Normal is defined as short sequences of system calls. We need to calculate Hamming distance to tell how a system call sequence chose another. We need to set the threshold to determine whether its anomalous or not. If we get a large hamming value then its malicious. Authors can properly check encroachment that tries to stop many UNIX programs.

1.2.1.3 Forensic with Computing methods in Privacy-invasive: Boldt and Carlson [12] It gives privacy-invasive software (PIS). Spyware and Adware are basic types of PIS. Boldt and team used the Forensic Tool Kit (FTK) to recognize PIS. Initially authors create a system free of PIS, clean the system and this is considered the baseline of the system. And it is main target. If the baseline is taped, task is performed to PIS on the target. Like, surfing on the browser. And snapshot recorded at definite intervals and Ad-Aware is most popular PIS removal tool, and so we assess Ad-Aware by forensic and static analysis methods. In this technique, Boldt and his team noticed that Ad-Aware generates false positives as well as false negatives.

1.2.1.4 Finite State Automata in Detecting Anomalous programs: Sekar et al., [13] made a Finite State Automata (FSA) method to classify anomaly inspection. Every node in the same constitutes a state in which the algorithm takes as input to learn valid behavior faster and good detection. The FSA transitions are to be done in response to the *system calls.* The working of FSA based approach is to detect anomaly in to construct.

1.2.1.5 NATE: Taylor and Alves-Foss [14] present cost efficient approach for detecting anomaly. NATE stands for Network Analysis of Anomalous Traffic Events. Their method focus on attack which ruined network protocol destructibility. The synchronization, finish, and reset are supposition packets in anomalous in large. The information flows from source IP to port is called as session.

1.2.2 Static Anomaly based detection

In these detection techniques, we can possibly get the maliciousness of the program without even executing it. This one is having the following analysis categories:

1.2.2.1 File print Analysis: This analysis is also called as n-gram analysis for detecting the malware in files. The author tests this by considering a benign file that has regular byte composition for their respective type. For instance if benign file is having unique byte distribution then from .doc and .exe files. Then this can be marked as suspicious and it has to undergo several other methods to detect the malware presence. **Li et al., [15]** experienced that requesting 1-gram analysis to PDF files inserted with malware productively to COTS AV scanner. If the malware is present then we get alarms before downloading it. Since it is possible to keep the malware in between the .pdf. At that time the user must open the .pdf to view content at that point also this method works.

1.2.2.2 Hybrid Anomaly-Based Detection Strider Ghost Buster: Wang et el., [16] introduced a malware detector which is also called as ghost ware. This malware secretes itself in the Operating systems .It hides himself by ambush the queries of them and alter so that it can’t be traced. For example, if user asks to list the files in the current directory, then there is chance that ghost ware delete any of its assets from results given by dir command. Author tells about *"cross-view diff-based*" method to inspecting these malware. In this the authors' compares the results in comparison to high-level to a low-level accesses on system call (*Master File Table* (MFT) directly) of the same data without using a system call. This process is known the "cross-view diff-based" approach.

1.2.3 Specification Based Detection

This process is derived from anomaly based detection. Specification based detection approximates the requirements of application or system where as other detection techniques approaches the implementation of files or system. There is a tool called panorama which captures the system wide information flow of the program under inspection over a system and checks behaviour against a valid set of rule to detect malicious activity. Specification based Detection is based on the analysis of the behaviour that are described in the system specification. Known, unknown and new malware can be detected by the specification detection technique but it is a tedious process. Specification based Detection is further classified into four types they are:

- Dynamic Specification based Detection
- Static Specification based Detection
- Hybrid Specification based Detection

1.2.3.1 Dynamic specification Based detection process

This method which comes under dynamic specification based their behaviour is observed during runtime to find the harmful programs. This detection process is available in various forms:

1.2.3.2 Minority security-critical programs: There is a monitor called as Distributed program Execution monitor. In this monitor, trace policies are generated for 15 unix programs and one trace policy was constructed for the remote file distribution client program.

1.2.3.3 Attachment Chain Tracing (ACT): After that system admin will clear the number of layers and that should be recognized from any host i before manufacturing the decision of how many hosts are contaminated.

1.2.3.4 Automated detection of vulnerabilities in privilege programs: In this program, it is derived from a program specification its runtime behaviour concluded to whether it is hostile or not. Programs specification is interpreted into System a call that is differentiated to the system calls of the PUI. System call of the PUI is seized by the operating system.

1.2.3.5 Process Behavior Monitoring: In this program is physically interpreted into an Auditing Specification Language(ASL) and it is assembled into a C++ class and it links with an framework that represents system calls and it generates the system call detection engine. Every system call implored at run time is represented and it will send to the system call detection engine. Detection engine compares with the system calls that are being made at run time against the ASL specifications.

1.2.3.6 Enlisting Hardware in Malicious code injection: In this method the processor has its own stack known as Secure Return Address Stack (SRAS). With the help of SRAS the processor identifies attacks based upon whether the stack in the memory is correspondent with SRAS.

1.2.3.7 Mitigating XSS Attacks from the Client-side: The easiest and the most effective **client-side** solution to the **XSS** problem for users are to deactivate JavaScript in their browsers. Unfortunately, this solution is often not feasible because a large number of web sites uses JavaScript for navigation and enhanced presentation of information.

1.2.3.8 Dynamic information Flow Tracking: In this data is being specified as either safe or unsafe or genuine or fake. The privacy policy is marked by the operating system it recognizes the fake data. Operating system has done this by keeping a bit of already recognized data. The processor verifies that the control in the program is not unexpected. If it identifies the control is based upon fake data by verifying the bit given by the os with the recognized data, then processor will throws an exception that is seized by the operating system.

1.2.3.9 Using Instruction Block signatures: In this some of the resources are allocated to processors to check that the most secure instructions are only being executed. In this method instruction block signatures are confirmed at runtime. And these signatures are coded by a secret processor key.

1.2.3.10 Fast detection of Scanning worms: In this, entries are used to represent whether a host is clean or contagious or uncontagious. A host is considered closer to contagious entries for unsuccessful connection and moved further from the clean entry.

1.2.3.11 Protecting Against unexpected system calls: This method holds Interrupt Address table(IAT). The IAT carries the system call number with respect to the system call that is seem in the executable and also it carries the address after the system call.

1.2.3.12 Preventing SQL injection Attacks: To prevent the SQL injection attack, first we need to establish in co-relation to which applications are vulnerable in the same. But its SQL language is a complex in trivial to construct a code snippets, is to attempt a query injection to compromise in the database.

1.2.4 Static specification based detection

The specifications is in static detection mechanisms determines maliciousness in the program code and it uses the structural properties for the same. These are having the various forms:

- **1.2.4.1** Static Detection of Malicious code in Executables
- **1.2.4.2** Static Analysis of Binaries
- **1.2.4.3** Compiler Approach to Mal code Detection
- **1.2.4.4** Detection of malicious executables (DOME)
- **1.2.4.5** Intrusion detection via static analysis
- **1.2.4.6** Stack Guard
- **1.2.4.7** Spike

CONCLUSION

In this paper we discussed about what actually malware is followed by the categories of malware. We identified that though malware is troublesome there is a way to transpire and to make system benevolent. There is vast history of malware as firstly it created its impact from1970s and continues till now. Now we have methods to recognize malware. We saw several ways to identify the malware present in the system and define some techniques to remove them. We even disclose the target of various malware and wrote about the prevention from them. There are several techniques to detect malware and abandon it from causing harm to honest user's system (can be mobile).Among the categories the anomaly based detection is having too less techniques to identify malware. Among each malware detection technique there is some pros and cons. The cons are giving false alarms of malware and considering a malicious behavior as normal. But these techniques won't stop here. With growing world the researchers encounters more new ways to detect malware with cost-efficiently.

References

[1] C. a Castillo, "Android Malware Past, Present, and Future," McAfee White Pap. Mob. Secur. Work. Gr., pp. 1–28, 2011

[2] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," 2012 IEEE Symp. Secur. Priv., no. 4, pp. 95–109, 2012

[3] Mobile Malware evolution report, 2016. https://securelist.com/mobile-malware-evolution-2016/77681/

[4] Layers of Cybersecurity: Signature Detection vs. Network Behavioral Analysis. Accessed: 16th July 2020. https://bricata.com/blog/signature-detection-vs-network-behavior/

[5] Kumar, P., Nema, A., and Kumar, R., 2009. Hybrid analysis of executables to detect security vulnerabilities. In Proceedings of the 2nd Annual Conference on India Software Engineering. 141--142.

[6] https://malware.wikia.org/wiki/Melissa. Malissa Accessed dated 16th July 2020.

[7] http://virus.wikidot.com/michelangelo. Michelangelo. Accessed dated 16th July 2020.

[8] Wang, K., and Stolfo, S.J. Anomalous payload-based network intrusion detection. In Proceedings of the 7th International Symposium on (RAID), pages 201–222,September 2004.

[9] [Online] Available: Accessed: 16th July 2020. https://www.sciencedirect.com/topics/engineering/mahalanobis-distance

[10] Lee, W. and Stolfo, S. 1988. Data mining approaches for intrusion detection. In Proceedings of the 7th USENIX Security Symposium. https://www.usenix.org/legacy/publications/library/proceedings/sec98/full_papers/lee/lee.pdf

[11] Hofmeyr, S., Forrest, S., and Somayaji, A. 1988. Intrusion detection using sequences of system calls. Journal of Computer Security, pp. 151 – 180.

[12] Boldt, M. and Carlsson, B. 2006. Analysing privacy-invasive software using computer forensic methods. https://dl.acm.org/doi/10.5555/1267549.1267555

[13] Sekar, R. Bendre, M., Bollineni, P., and Dhurjati, D. 2001. A fast automaton-based approach for detecting anomalous program behaviors. In IEEE Symposium on Security and Privacy.

[14] Taylor, C. and Alves-Foss J., 2001. Nate – network analysis of anomalous traffic events,a low-cost approach. New Security Paradigms Workshop.

[15] Li, W., Wang, K., Stolfo, S., and Herzog, B. 2005. Fileprints: Identifying file types by n-gram analysis. 6th IEEE Information Assurance Workshop.

[16] Wang, Y. M., Beck, D., Vo, B., Roussev, R., and Verbowski, C. 2005. Detecting stealth software with strider ghostbuster. In Proceedings of the 2005 International Conference on Dependable Systems and Networks, pages 368–377.