Malware Detection, Supportive Software Agents and Its Classification Schemes

[1]Adebayo, Olawale Surajudeen (B.Tech, MSc, MNCS, MCPN, MIACSIT), [2]M.A. Mabayoje (B.Sc, MSc, MSAN, MCPN), [3]Amit Mishra, [4]Osho Oluwafemi (MTech.)

[1,4]Cyber Security Science Department, F.U.T. Minna, Niger State, Nigeria.

[1]waleadebayo@futminna.edu.ng

[2]Department of Computer Science, University of Ilorin, P.M.B 1515, Ilorin, Nigeria.

[2]mmabayoje@yahoo.com

[3]Mathematics & Computer Science Department, IBB University Lapai, Niger State, Nigeria.

[3]i.amitmishra@gmail.com

Abstract

Over time, the task of curbing the emergence of malware and its dastard activities has been identified in terms of analysis, detection and containment of malware. Malware is a general term that is used to describe the category of malicious software that is part of security threats to the computer and internet system. It is a malignant program designed to hamper the effectiveness of a computer and internet system. This paper aims at identifying the malware as one of the most dreaded threats to an emerging computer and communication technology. The paper identified the category of malware, malware classification algorithms, malwares activities and ways of preventing and removing malware if it eventually infects system.

The research also describes tools that classify malware dataset using a rule-based classification scheme and machine learning algorithms to detect the malicious program from normal program through pattern recognition.

Keywords: Malware, Malware Detection, Malware Classification, Malware Supportive Software Agents

1. Introduction

Protecting, securing and maintaining computer and internet system from all forms of security threats including malware, internet fraud, and phishing among others are the most curious task that are being battled by the contemporary computer professionals, users and stakeholder. Malware remains one of the big threats that are ravaging the contemporary computer emergence. The concern for the rate of spread of malware today is a global phenomenon, especially as it spreading double over the internet which is a means of global communication. Malware is malicious software that is included intentionally in a computing facility purposefully to harm a system. Malware can also be termed as all kind of intrusions that is disastrous to the computer software and hardware system. Malware writer creates malware for

different reasons and purposes ranging from challenges to economic gain, destruction to retaliation among others. Its growth is highly alarming in volume and its rate of expansion cannot be overlooked due to its damages. Once malware gets itself into the system by different media like copying of files from external devices onto the system and mostly by downloading files from the internet, it checks the vulnerabilities of the system and infects the system if the system is highly vulnerable.

Another emerging technology that is being threatening by malware activities is mobile communication. This technology is a very fast means of communication both over mobile and electronic networks. As the services of mobile devices are doubling daily which include email and messaging, multimedia and others, which adopt operating systems like Symbian and Linux, this has made the tools highly vulnerable to various form of attacks. F-Secure published more than 350 mobile malware including Cabir [1], Mabir [2], Skull [3] and others targeting at Symbian software platform.

Computer malwares include computer viruses, worms, Trojan, Malicious Mobile Codes (Botnets, Nitda worm), Tracking Cokies (spywares, adwares, crimewares), Attacker Tools (Backdoors, Keylogger, Rootkits, E-mail generator) and other harmful software.

A malware detector is a system that aims at analyzing and identifying malware while malware detection is a field of study that deals with the analysis, detection and containment of malware. Malware detector can be a commercial virus scanner which uses binaries signature and other heuristic rules and algorithm to identify malware.

A very common technique adopts by malware writer is code obfuscation [4] which prevent its detection by the detectors. Code obfuscation technique can be polymorphic or metamorphic. A metamorphic virus obfuscate by hiding itself completely to evade detection while a polymorphic virus obfuscate its decryption loops using code insertion and transposition [4]. Moreover, a metamorphic malware adopt methods like register renaming, dead code insertion, block reordering and command substitute in order to perform its dastardly acts.

Another technique adopted by malware writer is the modification and inclusion of new behavior in their malware so as to increase its strength and viability. Malware like beagle worms, Sorbig A. through Sorbig F [4]. worm variants were developed iteratively with inclusion of new features.

2. Literature Review

One of the greatest challenges in security tasks that are still battling the exploration of mobile communication devices, computer and network infrastructures, and web technology is Malware attacks, detection and its analyses. Several solutions that have been adopted in the past in the detection and containment of malware can be classified into static analysis, dynamic analysis techniques and combination of both static and dynamic methods. Static analysis is the process of analyzing a program's code statistically without actually executing the code. The static analysis approach has the advantage that an entire code can be covered and therefore, possibly a complete program behavior, independent of any single path executed during run-time, will be easily captured. However, the statics analysis is constrained with its inability to detect new malware or new variants of malware.

Dynamic analysis, on the other hand, is necessary to complement the lapses of static analysis due to various obfuscation mechanisms, which rendered static analysis an ineffective method. Dynamic analysis was based on some heuristics such as the monitoring of modifications to the

system registry and the hooks' insertion into system interface or library. Dynamic analysis, however also have shortcomings since the heuristics are not based on the fundamental attributes of malware, they can be subjected to high false positive and false negative rates.

The techniques of malware detection can also be classified as Signature-based malware detection, Specification-based malware detection and Behaviour-based detection. To overcome the limitations of signature-based detection, behavior-based malware analysis and detection techniques was proposed by Forrest *et al.*[5], who designed host-based anomaly detection. The behaviour-based approaches observe the application behaviour of a code in the form of system routine calls and create a database of all the fixed-length consecutive system calls from normal applications. Possible intrusions are discovered by looking for call sequences that do not appear in the database. The techniques of Advanced mining was later consolidated on the behavioural detection technique deviations on the call sequences by using heuristic algorithms [6], Hidden Markov Model [7], and finite-state automata [8]. All the aforementioned detection methods share the same concept of representing programs' normal behavior with system calls and performing anomaly detection by measuring the deviation from normal profiles. However, the above approaches were characterized with shortcoming of ignoring the semantics of system call sequences and thus, could be evaded by simple obfuscation or mimicry attacks [10]. In other to address this deficiency, [9] proposed semantics-aware malware detection that attempts to detect polymorphic malware by identifying semantically-equivalent instruction sequences in the malware categories. In this work, he described the malicious behavior e.g, decryption loop with a template of instruction sequences, where a matching algorithm is applied on the disassembled binaries to find the instruction sequences that match the predefined malicious template. The authors discovered that it is resilient to several code obfuscation techniques by abstracting away the name of register and symbolic constants. However, attacks using the equivalent instruction replacement and reordering are still possible because it is still requires exact matching between the template node and application instructions.

Salvatore J. et al [11] presented File analysis for malware detection. They use statistical content analysis of files in order to detect anomalous file segments that may suggest infection by malcode. Their goal is to develop an efficient means of detecting suspect infected files for application to online network communication such as file sharing or media streaming, or scanning a large store of collected information, such as a data warehouse of acquired content. The problems with signature-based AV systems failing to detect new zero-day exploits are well known; a new generation of anomaly detection systems aimed at detecting zeroday exploits are beginning to appear in commercial products.

The behavioural malware detection on mobile handset in order to curb the casualty in the mobile community is another detection technique by [12]. Their approach is unique in the definition of application behavior. Their approach observes the programs' *run-time* behavior at a higher level (i.e., system events or resource-access) than system calls of [7] and machine instructions of [9]. This higher-level abstraction improves resilience to polymorphism and facilitates detection of malware variants, as it abstracts away more low-level implementation details. Also, the approach employs a runtime analysis, effectively bypassing the need to deal with code/data obfuscation [13]. Runtime analysis also avoids the possible loss of information of the static approach, since a static analysis often fails to reveal inter-compnet/system interaction information [13] and disassembly is not always possible for all binaries (Linn and Debray, 2001) showed that dissemblers can be thwarted with simple obfuscations. Moreover, in contrast to Forrest's anomaly detection [5] which learns only normal applications' behaviour or Christodorescu's misuse detection [9], which matches against only malicious templates, this

approach exploits information on *both* normal programs' and malware's behaviours, and employs a machine learning (instead of exact matching) algorithm to improve the detection accuracy. Since the learning and classification are based on two opposite-side data sets, this approach conceptually combines the anomaly detection with misuse detection and therefore, could strike a balance between false positives and false negatives.

Christopher Kruegel, et al [15] in another work propose binary analysis to detect the kernel rootkits by statically analyzing kernel modules and looking for suspicious instruction sequences. [16] is another work that determines a spyware component by statically extract a list for Windows API calls invoked in response to browser events, and combines it with dynamic analysis to identify the interactions between the component and the OS. A spyware-like behavior is detected if the component monitors user behavior and leaks this information by invoking some API calls. Static analysis is also widely used to collect the structural information of an executable file (e.g., control and data flow) and detect various malware [17].

Newsome and Song (2002) proposed a dynamic taint analysis to detect the buffer overflow exploits on commodity software. Their approach is to perform binary rewriting at run-time to track the propagation and improper use of unsafe or tainted data. [13] collected a sequence of application events at run-time and constructed an opaque object to represent the behaviour in rich syntax. Their work also applies a machine learning algorithm on high-level behaviour representations. However, their work focuses on clustering malware into different families using nearest-neighbour algorithms based on the edit distance between data samples, while they only interested in distinguishing normal from malicious programs. They also used a supervised learning procedure to make best of existing normal and malicious program information while clustering is a common unsupervised learning procedure.

Ellis *et al.* in [18] present a novel approach for automatic detection of Internet worms using their behavioural signatures. These signatures were generated from worm behaviours manifested in network traffic; the behaviour includes tree-like propagation and changing a server into a client. In the same vein, NetSpy [19] performs behaviour characterization and differential analysis on the network traffic to help automatically generate network-level signatures of new spyware. Their approach is fundamentally different from that of [18] in that they focused on characterization of host-based malware behaviour, incorporating a wide range of system events into behaviour signatures. The Primary Response from Sana Security [20] is another host-based behavioural approach that monitors desktop applications and employs multiple behavioural heuristics and correlations (e.g., Registry modification, key logging procedures, process hijacking, etc.) to identify a malicious application. In their BackTracker, King, S. T. and Chen, P. M. [21] aims to automatically identify potential sequences of activities that occurred in an intrusion.

Xuxian Jiang, et al [22] proposed Stealthy Malware Detection Through VMM-Based "Out-of-the-Box" Semantic View Reconstruction where they presented VMwatcher, a novel VMM-based approach that enables out-of-the-box malware detection by addressing the semantic gap challenge. More specifically, VMwatcher achieves stronger tamper-resistance by moving anti-malware facilities out of the monitored VM while maintaining the native semantic view of the VM via external semantic view reconstruction. Their evaluation of the VMwatcher prototype on both Linux and Windows platforms demonstrates its practicality and effectiveness. In particular, the experiments with real-world self-hiding rootkits further demonstrate the power of the new malware detection capabilities enabled by VMwatcher.

In order to curb the disastrous effects of a class of code injection attacks called SQLIAs that take the advantages of lack of validation of user input, William G. et al [23] propose AMNESIA (Analysis and Monitoring for Neutralizing SQL Injection Attacks), a fully automated technique and tool for detection and prevention of SQLIAs. The vulnerabilities occur when a developer combine hard-coded strings with user input to create dynamic queries. Their approach combine the static analysis and routine monitoring, where the program analysis is used, in static part, to build automatically a model of legitimate queries that could be generated by the application.

3. Malware

Malware is the term that is used to describe computer software or hardware that is harmful to the emergence of other computing system. The classification of malware is based on their attributes like replication tendency and strategy, purposes of creation, method of propagation and containment methodology. Malware may be created in order to destroy system of for various challenges (stuxnet, viruses), it may be created for financial gain (i.e. backdoors, botnets), or to gain unauthorized access to the system by compromise system effectiveness (adware, spyware, worms).

A malware can also be seen as a computer program that has various kinds of malicious intents [4]. Some commonly known Malware categorizations are viruses, Trojans, worms, Attacker toolkits, Malicious Mobile Code and Tracking Cokies. Malicious programs present an incessant threat to the privacy and security of sensitive data and the availability of critical services at crucial point in time.

3.1 Categories of Malware

There are various types of malware. Malware can be classified according to the purpose and method of propagation [25]. This paper therefore categorized malware according to the following group:

Computer worms, Computer Viruses, Trojan, Tracking Cookies, Attacker tools, Malicious Mobile Code.

3.1.1 Computer Viruses

This is a computer program that is designed to replicate itself and distribute the copied data to other computer thereby causing disinfection to other programs and files. Virus has payloads that contain codes for executing virus activities, which can either be benign or malicious in nature. A benign program may either irritate or consumes memory space unnecessarily while a malicious program causes several damages to the system. A virus may be compiled or interpreted. The source code of a compiled virus is converted by a compiler program for proper execution on the operating system while interpreted viruses codes can only be executed by some applications.

The obfuscation of virus has made it difficult for its detection. Malware writer used obfuscation like polymorphism, metamorphism, stealth, self-encryption and decryption, armoring among others to efface detection by the detectors. The basic purpose of creating virus is for system destruction by attackers.

3.1.2 Computer Worms

Computer worms are malicious programs that are both self-replicating and self-contained in nature i.e. they require no host program unlike virus in order to carry out their dastardly acts. Worms are also self-propagating malware and capable of creating and executing potential copies without user intervention. Many worms are purposely created to waste system resources and gain unauthorized access to the system.

Worms can be network service or mass mailing worms. The network service worm exploiting the network service vulnerabilities to gain access to the system while the mass wailing worms send a bulk of unwanted messages which may often reduce the effective performance of a system in question.

3.1.3 Trojan horse

This is a malware that pretends to perform a desirable function to the user prior to the installation but later facilitates and exposes user to unauthorized access of user's system. It emulates the attributes of original program such as login shell and hijack user password to gain access to the system remotely. It damages vital resources of the system like files and data and retards system from performing important functions. Some of most toughest Trojans malware includes:
Trojan.Wn32.Generic!SB.0,Trojan.Wn32.Malware,Trojan.ASF.Wimad,Trojan.HTML.FakeAlert.a

3.1.4 Tracking Cookies

These are small data that are capable of retaining information about the activities of a particular website.

3.1.5 Adware

Adware is otherwise called advertising-supported software that plays, displays, or download advertisement automatically to a computer after the installation of harmful software by the computer users. The advertising module is generally embedded into the malicious software, specially designed to detect the sites visited by the internet users in other to gain access to user's vital information for majorly financial purposes.

Some adware can also be referred to as shareware (also known as triaware or demoware) which is computer software that is licensed under the exclusive right of the owner. The licensee is given the right to use the software under certain conditions, but restricted from other uses, such as modifications, further distributions, or reverse engineering. Shareware often offer free download from the internet website. It gives buyer opportunities to assessing the software before it can be purchased. Some common adware software are free games on the internet, free software packages, Kazaa and Bearshare.

3.1.6 Crimeware

This is a kind of malware that is designed specifically to perpetrate and facilitate all forms of cybercrime. It is designed to access user's account online by presenting itself using the identity of another computer user in other to gain access to another person's account and other valuable information.

3.1.7 Spyware

Spyware as the name refers is software that is designed specifically to monitor the activities of users on the internet with a view to collect information about users without their knowledge. Spyware obtains information like credit card number, frequently visited sites, e-mail address among others. It also not only interferes with control of the computer but also changes the computer setting, which results in slow computer connection settings.

3.1.8 Attacker Tools

These are combinations of tools being used by malware writer and attacker to launch an attack on a system and network. They include tools like backdoors, keylogger, rootkits, e-mail generator and dialer

3.1.8.1 Rootkits

A Rootkit is a malicious software or hardware device designed to acquire administrator-level control over a computer system without being detected. Their activities include appropriating computing resources illegally without the knowledge of system administrator. The targeted computer resources include BIOS, Kernel, Boot loader among others.

3.1.8.2 Backdoor

This is a tool that is designed to listen for commands on a network protocol purposefully to gain access to a system at some other time.

3.1.8.3 Keylogger

This harmful code monitors and records the activities of keyboard and probably information on it.

3.1.8.4 e-mail generator

This is a malware tool that can be used to generate large number of email purposefully for undesired message distribution to other system.

3.9 Malicious Mobile Code

These are harmful codes that are designed purposefully to hamper the facilities on a mobile communication. The code is transmitted on a remote system and executed on a local system. They are capable of transmitting viruses and worms. They include nimda worms and botnets.

4. System Vulnerabilities

The cybercriminals overtime have developed several malware tools, which allow them to have unauthorized access and launch their attacks on the host system. Once a system has been compromised by a malware, an attacker can then launch their attack through several tools like packet sniffer, port scanner, vulnerability scanner, password crackers among others.

Malware like worms either send copies of its body via email or exploit the vulnerabilities that affect large number of hosts. Some of these vulnerabilities are source code revelation ($DATA) which is a bug in a poor programming, information exposures via scripts, buffer overflow syndrome where user submits more data than the script expects to store, which allows the script to break and thereby malware writer can gain access and install malware on the system.

On a mobile application, the use of out-band authentication via SMS and phone as an additional layer of security has added to the vulnerabilities in the mobile channel (RSA 2011). Another vulnerability on a mobile application is the rate at which people download information from internet to their phone. Beyond these, mobile phone is largely being used for mobile banking and payments, online transactions and personal data storage. All these have make malware proliferation on the mobile application remain expanding.

5. Malware Attack Techniques

The infection strategies of malware include entry point obfuscation, code integration, code insertion, register renaming, memory access reordering and session hijacking. In entry point obfuscation, the virus hijacks the control of the program after the program has been launched, overwrite program import table addresses and function call instructions. During the code integration, a virus merges its code with legitimate program that requires disassembly of target which is a very difficult operation (W95/Zmist).

Malware can also either append virus code and thereby modify the entry point of a legitimate program or inject its code into unused sections of a program code.

On the other hand, malware has two basic strategies adopted on a cell phone viz;

1) By creating a new process to launch its attack
2) By redirecting the program flow of a legitimate application in order to execute its malicious code within a legitimate security context (e.g messaging process) [12].

5.1 Malware First Attack Technique on mobile phone

Malware, in this case created a new process to execute its malicious code and compromise the cell phone. This is a case where user operations are required, for example when a user downloads software on an internet or opens a received message from another user. The newly created process contains a program descriptor, which describes the address content, execution state and security context, which is different from that of the invoked parent process. This technique is widely adopted by the most existing malware one to its simplicity.

In this technique, the cell phone malware launch an attack through legally installed application, having realized that the symbian and windows programs register themselves within a platform and use their system services within their API framework.

A good example is a cardblock Trojan, which is a cracked version of a legitimate symbian application called instansis. It allows a user to create SIS archive. Cardblock blocks the MMC memory card and detect the subdirectories under \system (SDI attack)

5.2 Malware Second Attack Technique on mobile phone

Malware, in this case redirects the program flow of a legitimate application (e.g. messaging activities) to execute its malicious code within a legitimate security context [12]. Open Source-based OS and application a framework is the major target of this kind of malware attack. i.e Linux-based smart phones. This type of attack is possible for malware by exploiting the stack buffer overflow [24] in a Linux-based cell phone to “hijack “the normal program flow and launch its attacks.

6. Malware method of Propagation

The basic method of propagation of malware is either self-propagation or user interaction. A malware like worm does not require any user intervention before its execution occur. It is capable of copying itself and causing occasional execution without the intervention of host program or its user. Virus on the other hand is a user-interaction oriented malware that always looks for a host program for its execution and consequent infection. Other malware might not require any of these methods for its propagation, but may adopt internet medium for their spreading.

Mobile malware on the other hand, adopt mobile phone network on the internet in order to propagate itself, but this action is usually curtailed by the internally built defense mechanism in the network mobile phone. Another opportunity for mobile malware to propagate is through the direct pair-wise communication resources i.e. Bluetooth, Wi-Fi, Infrared [26].

7. Characteristics of Malware

Malware is a self-replicating program, which discreetly installs itself in a data processing system, without user's knowledge or consent, with a view to endangering data confidentiality, data integrity and system availability or making sure that the authentic user is being framed for computer crime. Among the attributes of a malware are Self-replication, Self-propagation, user interaction, Self- contained and the purpose to which a malware is created.

8. Malware Detection Techniques

The task of detecting malware can be categorized into analysis, classification, detection and eventual containment of malware. Several classification techniques have been used in order to classify malware according to their instances and this has made it possible to recognize the type and activities of a malware and new variant. Analysis of malware has to do with identifying the instances of malware by different classification schemes using the attributes of known malware characteristics. Malware detection has to do with the quick detection and validation of any instance of malware in order to prevent further damage to the system. The last part of the job is containment of the malware, which involves effort at stopping escalation and preventing further damages to the system. A commercial antivirus uses signature based technique where the database must be regularly updated in order to possess the latest virus data detection mechanisms. However, the zero-day malicious exploit malware cannot be detected by antivirus, based on signature-based scanner, but the use of statistical binary content analysis of file to detect anomalous file segments [11]. Toward this end, malware detection technique has been classified according to the following:

8.1 Signature-based malware detection

A pattern-marching approach by [9] such as commercial antivirus is an example of signature-based malware detection where the scanner scans for a sequence of byte within a program code to identify and report a malicious code. This approach to malware detection adopts a syntactic-level of code instructions in order to detect malware by analyzing the code during program compilation. This technique usually covers complete program code and within a short period of time. However, this method has limitation by ignoring the semantics of instructions, which allows malware obfuscation during the program's run-time.

8.2 Specification-based malware detection

[9] is a special case of specification-based malware detection, where a detection algorithm that addresses the deficiency of pattern-matching was developed. This algorithm incorporates instruction semantics to detect malware instances. The approach is highly resilience to common obfuscation techniques. It used template T to describe the malicious behaviours of a malware, which are sequence of instructions represented by variables and symbolic constants. The limitation of this approach is that the attribute of a program cannot be accurately specified.

8.3 Behavioural-based detection

This approach does not only perform surface scanning but also identify the malware's action. The approach generates database of a malicious behaviours by studying a distinct number of families of malware on a target operating system. [12] develops a two stage mapping technique that constructs signatures at run-time from the monitored system event and API calls. The system trains a classifier using a support vector machines (SVMs) to distinguish a malicious program from normal application behaviours. This detection system is capable of detecting metamorphic malware which keep reproducing.

8.4 Data mining technique of detecting malware

In their paper titled data mining methods for detecting malicious executables, [39] defined a malicious executable as a program that performs function, such as compromising a system's security, damaging a system or obtaining sensitive information without the user's permission. Their data mining methods detect patterns in large amounts of data, such as byte code, and use these patterns to detect future instances in similar data. Their framework used *classifiers* to detect new malicious executables. According to [39], classifier is a rule set, or detection model, generated by the data mining algorithm that was trained over a given set of training data. They designed a framework that used data mining algorithms to train multiple classifiers on a set of malicious and benign executables to detect new examples. The binaries were first statically analysed to extract properties of the binary, and then the classifiers trained over a subset of the data. Their large sets of programs from public sources were separated into two classes: malicious and benign executables. Example of this data set is a Windows or MS-DOS format executable, which is also applicable to other formats. Since the virus scanner was updated and the viruses were obtained from public sources, it was assumed that the virus scanner has a signature for each malicious virus. They then split the dataset into two subsets: the *training set* and the *test set.* The data mining algorithms used the training set while generating the rule sets. The test set was then used to check the accuracy of the classifiers over unseen examples.

This data mining method was able to detect previously undetectable malicious executables by comparing the results with traditional signature-based methods and with other learning algorithms. According to [39], the Multi-Naive Bayes method had the highest accuracy and detection rate of any algorithm over unknown programs, 97.76%, over double the detection rates of signature-based methods. Its rule set was also more difficult to defeat than other methods because all lines of machine instructions would have to be changed to avoid detection.

9. Malware Classification Schemes

Classification scheme is a technique used to identify the pattern of a data by classifying data according to their attribute. Among the data mining classification techniques for malware detection are

9.1 Ruled-Based Classifier

This classification rule is used to classify malware database based on their attribute using a collection of "If …then…" rules. For example, consider the following malware database

CHARACTERISTICS	Self Contained	Self Replicating	Propagation Method	Purpose	Malicious	Class
W32/Mabezat.B	**Yes**	Yes	Self Propagation	Destruction	yes	Worm
TR/ATraps.Gen	Yes	No	N/A	Financial gain	yes	Trojan
Dark Avenger	No	YES	User Interaction	Destruction	yes	Virus
Nimda worm	No	No	N/A	Financial gain	yes	Malicious Mobile Code
Trackware.Gemius	Yes	No	N/A	Financial gain	no	Tracking Cookies
Rook.4517	Yes	No	N/A	Financial gain	yes	Attacker tools

Table 1. Malware classification

A classification rule can be given such that:

1. R1: (Self Contained = yes) ∧ (Self Replicating = yes) → Worm
2. R2: (Self Contained = no) ∧ (Self Replicating = yes) → Virus
3. R3: (Self Contained = no) ∧ (Self Replicating = no) → Malicious Mobile Code
4. R4: (Self Contained = yes) ∧ (Malicious = no) → Tracking Cookies
5. R5: (Self Contained = yes) ∧ (Propagation Method = N/A) → Trojan

A rule classifier may be mutually exclusive or exhaustive. A classifier is mutually exclusive if the rules define variable are independent of each other (i.e. every record is covered by at most one rule). A rule on the other hand is exhaustive if the rule accounts for every possible combination of attribute value (i.e. every record is covered by at least one rule). A rule-based classification can be easily interpreted, highly expressive, easy to generate and new instances of malware can be easily and rapidly classified.

9.2 Bayes Classifier

Bayes is a probalistic technique for classifying malware instances.

Consider A as the attributes of malware and C as the class a particular malware is belong to, then Bayes theorem is given as:

$$P(A / C) = \frac{P(A / C) \times P(C)}{P(A)}$$

Given a conditional probability $P(C / A) = \frac{P(A, C)}{P(A)}$

and

$$P(A / C) = \frac{P(A, C)}{P(C)}$$

Consider the following attributes A_1, A_2 … A_n of malware M; the problem is to predict the class C to which each malware belong i.e. we want to find the class C that maximize

P (C | A1 A2…An).

Then $P(C/A_1, A_2, \ldots A_n) = \frac{P(A_1, A_2, \ldots A_n/C) \times P(C)}{P(A_1, A_2, \ldots A_n)}$

But in order to choose the value of C that maximizes
P (A1, A2, …, An | C) P(C), then we adopt naïve bayes theorem which is describe below.

9.3 Naïve Bayes Classifier

A naïve bayes classifier associate independency to various attribute of malware when the class of malware is known. For example, if A1, A2 … An is a set of attribute of a malware and $C_k \in$ C is a class, then

P(A1, A2, …, An |C) = P(A1| C_k) P(A2| C_k)… P(An| C_k) and this probability can estimate P(Ai| C_k) for all Ai and C_k. Therefore, the new point for the database is classified to C_k if $P(C_k) \Pi P(Ai| C_k)$ which is maximal.

From table 1, consider a test data X = (Self Contained = no, Self Replicating = no, Propagation Method = User Interaction)

P (Self Contained = yes/no) = 2/3

P (Self Contained = yes/yes) = 2/5

P (Self Contained = no/yes) = 2/5

P (Self Contained = no/no) = 0

P (Self Replicating = yes/no) = 0

P (Self Replicating = yes/yes) = 2/5

P (Self Replicating = no/yes) = 3/5

P (Self Replicating = no/no) = 1

P (Propagation Method = Self Propagation/yes) = 1/5

P (Propagation Method = Self Propagation/no) = 0

P (Propagation Method = User Interaction/yes) = 1/5

P (Propagation Method = User Interaction/no) = 0

P (Propagation Method = N/A/yes) = 3/5

P (Propagation Method = N/A/no) = 1

But P (X/Class = no) = P (Self Contained = no/Class = no) x P (Self Replicating= no/Class = no) x P (Propagation Method = User Interaction/Class = no)

i.e. P (X/Class = no) = 0

P (X / Class = yes) =

P (Self Contained = no/Class = yes) x P (Self Replicating= no/Class = yes) x P (Propagation Method = User Interaction/Class = yes)

= 2/5 x 3/5 x 1/5 = 6/125 = 0.048

Since P (X/yes) P (yes) > P (X/no) P (no), therefore P(yes/X) > P (no/X) which implies that

Class = yes

The robustness of Naïve Bayes classification in the isolation of noise points and irrelevant attributes has made it unique in the classification of malware. It is also effective in handling missing values by ignoring the instances during the estimation.

9.4 Artificial Neural Networks (ANN) Classifier

This network model is an assembly of interconnected nodes and weighted links that output nodes that is total of its input value according to its weight links. The model train data using a feed forward back propagation algorithm with the aid of scaled conjugated gradient for effective learning. The model algorithm initializes its weights WI and adjusts the weight so that the output of the network is parallel with the class of training data (Tan, Steinbach and Kumar (2004)).

9.5 Instance-based Classifier

This is a classifier that stores training data and uses it to predict the class label of unseen data cases. The stored database is compared with the attribute of an unknown database. This classifier can be rote-learner or nearest neighbour. The rote-learner memorizes the entire training and classifies the malware only if attribute of new database match that of the training dataset. On the other hand, the nearest neigbour classify malware base on the closeness of test data to the trained data.

9.6 Support Vector Machines (algorithm for learning)

These classifiers are learning machine that separate the data points on a hyper plane and solving the problems of over fitting. It is used to train the dataset in order to classify malware into either malign or malicious code. Support vector Machines have been largely and successfully applied to solve a number of classification problems. This algorithm is capable of separating both linear and non-linear data on a hyper plane. Over a given training set, the learning machine according to Vapnik and Joachim [41] on their statistical learning theory must be able to predict accurately the relationship between the training set and the general algorithm in order to determine its capacity to predict the future unknown variables.

9.7 Malware Prevention Tools

These are supportive software agents that are used to detect and curtail the dastard effects of malware. Among several tools adopted in order to prevent the emergence of malware are commercial anti-virus scanner, firewalls and routers, Intrusion prevention system among others.

10. Conclusion

This survey has presented a number of malware detections, malware classification schemes and associated problems with various detection techniques. The benefits of each malware classification scheme are also highlighted. The task of curtailing the dastard effects of malware cannot be overemphasized as it constitutes global threat to our online resources and financial activities. As malware writer change their techniques by adding new behaviours and modifying existing ones, the task of defending vita facilities against malware lies on the appropriate consideration for security control while developing software. The research identified some best practices for an organization to prevent the effects of malware activities.

11. Recommendation

For an organization to protect it system from all forms of the activities of malware, the following precautions are to be routinely undertaken.

11.1 An organization must develop and routinely implement an approach to malware incident prevention based on the vectors of attack. The stakeholders or personnel in charge of information technology must be as regard the malware awareness programs related to programs vulnerabilities and threats.

11.2 Every organization must be security conscious right from the development of software regarding the incorporation of incidents of malware detection and prevention.

11.3 An organization must ensure that the firewall of the system is appropriately turned on always. The firewall is the computer software that screens out hackers, viruses, worms and all other malware that may try to penetrate the system.

11.4 The security policies of an organization must effectively and efficiently support the detection and prevention of malware incidents.

11.5 The threats reduction capabilities of an organization's system should effectively contain malware emergence.

11.6 The antivirus software should be kept updated. The antivirus software scans and removes viruses from the system if detected.

11.7 The antispyware technology should be kept updated. This is the software that protects the system from the activities of spyware malware.

11.8 All software and operating system on the system should be kept updated

11.9 Organization should train her personnel as to the discouragement of unknown file downloading from the internet or unreliable sources.

12. References

[1] F-secure. Cabir. (2006). Access from http://www.f-secure.com/v-descs/cabir.shtml, 29-10-2011.

[2] F-secure. Lasco.a. (2006). Access from http://www.f-secure.com/v-descs/lasco a.shtml, 29-10-2011.

[3] F-Secure. SymbOS (2006) "Acallno Trojan description", Access from http://www.f-secure.com/swdesc/acallno a.shtml, August 2006, 29-10-2011.

[4] Mihai Christodorescu, Somesh Jha, Douglas Maughan, Dawn Song, Cliff Wang (2007) "Malware Detection": Advance Information Security; ISBN-10: 0-387-32720-7, ISBN-13: 978-0-387-32720-4,e-I SBN-10: 0-387-44599-4, e-ISBN-13: 978-0-387-44599-1

[5] Stephanie, F., Steven, A., Hofmeyr, A. S. and Thomas, A. L. (1996) "A sense of self for Unix Processes", In *Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy*, pages 120–128. IEEE Computer Society Press.

[6] William, W. C. (1995) "Fast effective rule induction", In Armand Prieditis and Stuart Russell, editors, *Proc. of the 12th International Conference on Machine Learning*, pages 115–123, Tahoe City, CA, July 9–12, 1995. Morgan Kaufmann.

[7] Christina Warrender, Forrest, S. & Barak, A. (1999) "Pearlmutter Detecting intrusions using system calls : Alternative data models", In *IEEE Symposium on Security and Privacy*, pages 133–145.

[8] Andrew, P. K. and Steven, A. H. (1997) "Intrusion detection via system call traces", *IEEE Softw.*, 14(5):35–42.

[9] Christodorescu, M., Jha, S., Seshia, S.A., Song, D. and R.E.Bryant. (2005) "Semantics-aware malware detection", In *Proceedings of the IEEE Symposium on Security and Privacy*.

[10] Wagner, D. and Soto, P. (2002), " Mimicry attacks on host based intrusion detection systems".

[11] Salvatore J. Stolfo, Ke Wang, Wei-Jen Li. (2005) "File analysis for malware detection", HSARPA #0421001/H-SB04.2-002.WORMS 2005 Columbia IDS Lab June 19, 2005 2

[12] Abhijit, B., Xin, H., Kang G. S. and Taejoon, P. (2008) " Behavioral detection of Malware on Mobile Handsets", June 17–20, 2008, Breckenridge, Colorado, USA. ACM 978-1- 60558-139-2/08/06

[13] Lee, T. and Mody, J.J. (2006), "Behavioural classification". Access from http://www.microsoft.com/downloads/details.aspx?FamilyID=7b5d8cc8-b336-4091-abb5-2cc500a6c41a&displaylang=en, 2006.

[15] Christopher Kruegel, William Robertson, and Giovanni Vigna. (2004) " Detecting kernellevel rootkits through binary analysis", In *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04)*, pages 91–100, Washington, DC, USA. IEEE Computer Society.

[16] Kirda, E., Kruegel, C., Banks, G., Vigna, G. and Kemmerer, R. (2006) " Behavior-based spyware Detection", In *Proceedings of the 15th USENIX Security Symposium*.

[17] Johannes, K., Stefan, K., Christian, S., and Helmut V. (2005) "Detecting malicious code by model checking", In *GI SIG SIDAR Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA'05)*, volume 3548 of *Lecture Notes in Computer Science*, pages 174–187.

[18] Daniel, R. E., John G. A., Kira S. A. and Scott D. Tenaglia. (2004) "A behavioral approach to worm detection", In *WORM '04: Proceedings of the 2004 ACM workshop on Rapid malcode*, pages 43–53.

[19] Wang, H., Jha, S., and Ganapathy, V. (2006) "Automatic generation of spyware signatures for NIDS", In *Proceedings of Annual Computer Security Applications Conference*.

[20] Hofmeyr, S. and Williamson, M. (2005) "Primary response technical white paper", In *Sana Security*.

[21] King, S. T. and Chen, P. M. (2005), "Backtracking intrusions", ACM *Transactions on Computer Systems (TOCS)*, 23(1):51–76, 2005.

[22] Xuxian, J., Xinyuan, W., and Dongyan Xu. (2005) Department of Information and Software Engineering Department of Computer Science, George Mason University Purdue University

[23] William G.J. Halfond and Alessandro Orso, (2005) "Detection and Prevention of SQL Injection Attacks" Georgia Institute of Technology.

[24] Somayaji A. and Forrest, S. (2006) "Automated response using system-call delays", In *Proceedingsof the USENIX Security Symposium*

[25] Peter, M., Karent, K. and Joseph, N, (2005) "Guild to Malware Incident Prevention and Handling", NIST special publication 800-83

[26] Gjergji, Z., Goefrey M., Michael, L., and Per, J. (2005) "Defending Mobile Phones from Proximity Malware"

[27] Sekar, R., Bendre, M., Dhurjati, D. and Bollineni, P. (2001) " A fast automaton-based method for detecting anomalous program behaviours", In *SP '01: Proceedings of the 2001 IEEE Symposium on Security and Privacy*, page 144, Washington, DC, USA. IEEE Computer Society.

[28] Tan, Steinbach and Kumar (2004) " Introduction to data mining ".

[29] Moore, D., Shannon, C. and Brown, J. (2002) " Code-Red: a case study on the spread and victims of an internet worm", In *Proceedings ot the Internet Measurement Workshop 2002, Marseille, France*, November 6-8 2002.

[30] Albert-Laszlo Barabasi and Reka Albert. (1999), "Emergence of scaling in random networks *Science"*, 286:509–512.

[31] John Bellardo and Stefan Savage. (2003) "802.11 denial-of-service attacks: Real vulnerabilities and practical solutions", In *Proceedings of the USENIX Security Symposium*, August 2003.

[32] Gerard Berry and Georges Gonthier. The esterel synchronous programming language: Design, semantics, implementation. *Science of Computer Programming*,

[33] Kaspersky Corporation. Kaspersky Anti-Virus Mobile. Access from http://usa.kaspersky.com/products services/antivirus-mobile.php, 2006.

[34] Yariv Kaplan. (2000) "API spying techniques for Windows 9x, NT and 2000", Access from http://www.internals.com/articles/apispy/apispy.htm.

[35] Karagiannis, T., Molle, M., Faloutsos, M. and A. Broido. (2004) "A non-stationary Poisson view of internet traffic", In *IEEE INFOCOM*, March 2004.

[36] Nazario, J. (2003) *"Defense and Detection Strategies against Internet Worms"*, Artech House.

[37] Symantec. Palm.phage (2000) "dropper virus description", Access from http://securityresponse.symantec.com/avcenter/venc/data/palm.phage.dropper.html, September 2000.

[38] F-secure. Mobile detection descriptions. Access from http://www.f-secure.com

[39] Matthew G. S, Eleazar Eskin, Erez Zadok & Salvatore J. S (2001) "Data Mining Methods for Detection of New Malicious Executables" 1081-601 1/01 $10.00 *0* 2001 IEEE

Authors

Olawale Surajudeen Adebayo (MCPN, MNCS, MIAENG, MGDN, MIACSIT) is a lecturer in the department of Cyber security science department, Federal University of Technology, Minna, Niger State Nigeria. He bagged B.Tech. in Mathematics and Computer science from Federal University of Technology, Minna and the MSc. in Computer science from University of Ilorin, Kwara state, Nigeria. He is presently a PhD student in the department of cyber Security science, Federal University of Technology, Minna.

His current research interests include: Network security, Cryptology, Machine learning, Data mining and computational intelligent. He has published some papers in the above-mentioned research areas.

He is a member of Computer Professional Registration Council of Nigeria (CPN), Nigeria Computer Society (NCS), Global Development Network, International Association of Engineer (IAENG) and many others.

M. A. Mabayoje is a lecturer in the department of Computer Science, University of Ilorin, Nigeria. She bagged Bachelor of Science and Master of Science in Computer Science in the University of Ilorin, Nigeria. She is currently a PhD student in the same university. She is a member of Nigeria Computer Society, Science Association of Nigeria among others. Her research interests include ontology, Artificial Intelligence, Software Engineering. She is married with children.