Classifying Spyware Files Using Data Mining Algorithms and Hexadecimal Representation

P.Divya
PG Scholar, Dept Of CSE,
Velammal engineering college, Surapet,
Anna University of Technology
Chennai, India
divyapugazh87@gmail.com

Mrs.S.Rajalakshmi
Assistant Professor , Dept of CSE,
Velammal engineering college, Surapet,
Anna University of Technology
Chennai, India
raji780@yahoo.co.in

Abstract— Malicious program is a serious threat for the security components such as confidentiality, integrity and availability. These new malicious executables are created at the rate of thousands every year. There are several types of threat to violate these components; for example Viruses, Worms, Trojan horse and spyware. Spyware represents a serious threat to confidentiality since it may result in loss of control over private data for computer users. Unlike viruses and worms, spyware does not usually self-replicate. It is typically hidden from the user and difficult to detect since it can create significant unwanted CPU activity, disk usage and network traffic. In existing systems, new malicious programs can be detected by automatic signature generation called as F-Sign for automatic extraction of unique signatures from malware files. This is primarily intended for high-speed network traffic. The signature extraction process is based on a comparison with a common function repository. By eliminating functions appearing in the common function repository from the signature candidate list, F-Sign can minimize the risk of false-positive detection errors. To minimize false-positive rates even further, F-Sign proposes intelligent candidate selection using an entropy score to generate signatures. Evaluation of F-Sign was conducted under various conditions. The findings suggest that the existing method can be used for automatically generating signatures that are both specific and sensitive. In this proposed model, data mining techniques like association, classification, and regression are used to generate patterns to check the malicious code and training set of frequent malicious code patterns are used to identify two classes such as legitimate software and malicious code patterns. Bayesian classifier is integrated with SVM which is proven for accurate classification.

Index Terms—Automatic signature generation (ASG), malware, malware filtering.

I. INTRODUCTION

Modern computer and communication infrastructures are highly susceptible to various types of attack. A common way of launching these attacks is by means of malicious soft*ware* (malware), such as worms, viruses, and Trojan horses, which can cause severe damage to private users, commercial companies, and governments. The recent growth in high-speed Internet connections provides a platform for creating and rapidly spreading the new malware. Several analysis techniques for detecting malware have been proposed. They are classified as to whether they are static or dynamic. In *dynamic analysis* (also known as behavioral-based analysis), detection is based on information collected from the operating system at runtime (i.e., during the execution of the program), such as system calls, network access and files, and memory modifications. In *static analysis*, the detection is based on information extracted explicitly or implicitly from the executable binary/source code. The main advantage of static analysis is in providing rapid classification. Since antivirus Programs that have the potential to violate the privacy and security of a system. These programs include: spyware, adware, Trojans, freeware and backdoors. They may compromise integrity confidentiality, and availability of the system and may obtain sensitive information without informed user consent [2,3]. This information is valuable for marketing companies and also generates income for advertisers from online ads distribution through adware. This factor works as a catalyst for elevating the spyware industry [1].

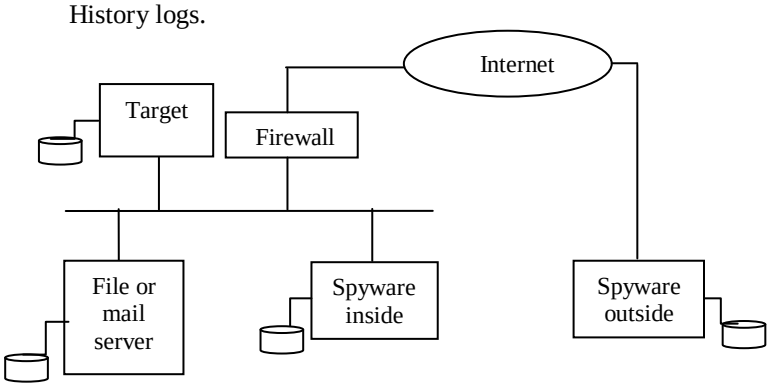
Static analysis solutions are primarily implemented using two methods: signature-based and heuristic-based. Signature-based methods rely on the identification of unique strings in the binary code. The heuristic methods are based on rules, which are either determined by experts or by machine-learning techniques that define a malicious or a benign behavior in order to detect unknown malware. The period of time from the release of an unknown malware until security software/hardware vendors update their clients with the proper malware signature is highly critical. During this time, the malware is undetectable by most signature-based solutions and is usually termed a zero-day attack (or zero-day threat). Since the new malware can easily spread and infect other machines, it is highly important to detect it as soon as possible and to rapidly generate a suitable signature so that signature-based solutions can be updated to block the new threat.

One way to protect organizations from malware is to deploy High-speed network-based intrusion detection systems on the communication lines. Such appliances perform deep-packet inspection in real-time and use simple signatures for detecting and removing attacks such as malware, propagating worms, denial-of-service, or remote exploitation of vulnerabilities. In order to monitor traffic in real time without causing a major impact on performance (e.g., delay and latency), these devices

inspect the content of the packets without reassembly of the session. Malware is a collective term for any malicious software which enters system without authorization of user of the system. It is a very big thread in today's computing world. Most of the malware enters the system while downloading files over internet. Once the malicious software finds its way into the system, it scans for vulnerabilities of operating system and perform unintended actions on the system, finally slowing down the performance of the system. Malware has ability to infect other executable code, data and system files, and create excessive traffic on network leading to denial of service. Some malware are very easy to detect and remove through antivirus software. These antivirus software maintains a repository of virus signatures i.e., binary pattern characteristic of malicious code. Files suspected to be infected are checked for presence of any virus signatures. This method of detection worked well until the malware writer started writing polymorphic and metamorphic malware. Spyware is an one type of malware that can be installed on computer which collects some piece of information about users without their knowledge. Spyware suggests software that secretly monitors the user computing and gathers personal information about the user like the pages frequently visited, email address, credit card number, key pressed by user. It generally enters a system when free software is downloaded.

II. SPYWARE ARCHITECTURE

Spyware is related to other kinds of malicious software, including features built into the platform like Cookies and History logs.



Figure(a): Spyware architecture

A helpful way to organize the market space is first by whether the software just monitors or actually modifies the behavior of the system. Another axis is whether the resulting behavior is harmless to the use, or potentially very dangerous. In many cases, there is no need to purchase spyware, because the existing log and history files maintained by Windows. Spyware tools can capture very sensitive passwords and user information. In most spyware products, the log files are not encrypted and thus become a very sensitive target. Almost all encryption products rely on passwords to control access. Spyware tools easily capture passwords and thus completely destroy the protection provided by cryptography. For example an attacker can copy the S/MIME or PGP private key file and then capture the password to unseal it. Smart card offer limited help against spyware. The spyware can capture screen shots of decrypted file and keystrokes being entered. An attacker could also install a piece of software that uses the smart card to decrypt file keys while the user has it insert, though none of the programs we examined supported this feature.

Spyware can monitor a platform at many levels. The higher levels are easier for the spy to understand, but not as complete as the lower levels. For example, a log of a chat session is quite easy to understand, whereas a log of the keystroke would just show one side of the chat. The high level tools must be customized for each application, whereas the low level tools can capture information about any application. For example, we did not find a program that specifically handled SSH telnet sessions, but both the screen shot capture and the keystroke capture can get most of the information. The spyware can also capture voice and video using attached microphone and camera. It is always installed inside the target system in order to intercept low level system activity. The targets can be end-user machines or servers. The spy can be located inside or outside of the firewall. Network monitoring spyware can be located on any platform on a broadcast LAN or located on a choke point like a firewall.

III. AUTOMATIC MALWARE SIGNATURE GENERATION

In general, malware signatures can be classified as vulnerability-based, exploit-based, and payload-based. A **Vulnerability-based signature** describes the properties of a certain bug in the system that can be maliciously exploited by the malware. Vulnerability-based signatures do not attempt to detect every malicious code exploiting the vulnerability, and therefore, can be very effective when dealing with polymorphic malware. However, a vulnerability-based signature can be generated only when the vulnerability is discovered.

An **Exploit-based signature** describes a piece of code (sequence of commands or inputs) triggered by the malware, which actually exploits a vulnerability in the system. Exploit-based methods include Autograph, PAYL sensor, Net spy, and Early Bird, which focus on analyzing similarities in packet payloads belonging to suspicious network traffic. These systems first identify anomalous traffic originating from suspicious IP addresses, and then, generate a signature by identifying most frequently occurring byte sequences. This signatures can be generated rapidly to detect zero-day exploits of uncovered vulnerabilities. They are, however, less effective on polymorphic malware. In addition, the signatures generated by the above techniques were extracted and tested for short, worm-related, malware, ignoring the fact that malware, such as viruses and Trojan horses, can appear as large executable

files, carrying full-fledged applications. These files usually contain a significant portion of invariant code segments that are planted by the software development platform spawning the malware. For these large malware files, selecting a signature that will be both sensitive and specific is a challenging task. Another limitation of these techniques is that they focus on detecting malware after it has been unleashed and try to generate a signature from the traffic it creates while the attack is being launched. **Position-Aware Distribution Signatures (PADS)** that are computed from polymorphic worm samples and are composed of a byte frequency distribution (instead of a fixed value) for each position in the signature "string."

A **Payload-based Signature** indentifies the actual malware code or body. The approach proposed in this paper falls into the payload-based signature category. Payload-based signature generation methods extracting good, "near optimal" signatures from the code of a virus. In the first step, decoy programs on isolated machines are deliberately infected with the virus. Then, the infected regions of the decoys are compared with one another to establish that regions of the virus are constant from one instance to another. These regions are considered as signature candidates. The second phase estimates the probability that each of the candidate signatures will match a randomly chosen block of bytes in the code of a randomly chosen program. The candidate with the lowest estimated false-positive probabilities is chosen as the signature. The Hancock system was proposed for automatically extracting signatures for antivirus software. Based on several heuristics, the Hancock system generates a set of signature candidates, selecting the candidates that are not likely to be found in benign code. Similar to our approach, Hancock relies on modeling benign code in order to minimize false-alarm risks.

IV. MALWARE DETECTION METHOD

Techniques used for malware detection can be broadly classified into two categories: anomaly-based detection and signature-based detection. An anomaly based detection techniques uses the knowledge of what is considered as normal to find out what actually is malicious .A special type of anomaly based detection is specification-based detection. Specification based detection makes use of certain rule set of what is considered as normal in order to decide the maliciousness of the program violating the predefined rule set. Thus programs violating the rule set are considered as malicious program. Signature based detection uses the knowledge of what is considered as malicious to find out the Maliciousness of the program under inspection.

A. The Malware Detector

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

$$D(p) = \begin{cases} \text{Malicious if p contains, Malicious code} \\ \text{Benign, Otherwise.} \end{cases}$$

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

1) False positive:

A false positive occurs when a virus scanner erroneously detects a 'virus' in a non-infected file. False positives result when the signature used to detect a particular virus is not unique to the virus - i.e. the same signature appears in legitimate, non-infected software.

2) False negative:

A false negative occurs when a virus scanner fails to detect a virus in an infected file. The antivirus scanner may fail to detect the virus because the virus is new and no signature is yet available, or it may fail to detect because of configuration settings or even faulty signatures.

3) Hit ratio:

A hit ratio occurs when a malware detector detects the malware. This happen because the signature of malware matches with the signatures stored in the signature databases.

B. Data mining process

Data mining is the process of generating patterns and comparing the patterns with target resource and identifies their characteristics. In this spyware detection process, we make use of classification, association and regression techniques to mine the files and WebPages. The notion of using data mining for this purpose is that, data mining is capable of identifying the features of a data that is completely new to the system. This detection is performed on the basis of similar data set that is present in the system in the form of training data. When a collection of data with certain characteristics is provided, the system will be able to classify the new data or predict the nature of the new data entering the system based on the features of the training data set. In this case, the classification and feature detection is to identify whether the data is spyware or legitimate software. The resources that are vulnerable to spyware threat are identified and the resource is discarded by the system. This process requires a basic training data that is used to generate the patterns of legitimate software and spyware [2].

C. *The Naive Bayes (NB) Algorithm*

The Naive Bayes algorithm is one classification method based on conditional probabilities that uses a statistical approach to the problem of pattern recognition. Literature reports that it is the most successful known algorithms for learning to classify text documents, and further it is fast and highly scalable for model building and scoring reference. The idea behind a Naive Bayes algorithm is the Bayes Theorem and the maximum posteriori hypothesis. Bayes Theorem finds the probability of an event occurring given the probability of another event that has occurred already. Among data mining methods, Naive Bayes algorithm is easy to implement and is an efficient and effective inductive learning algorithm for machine learning.

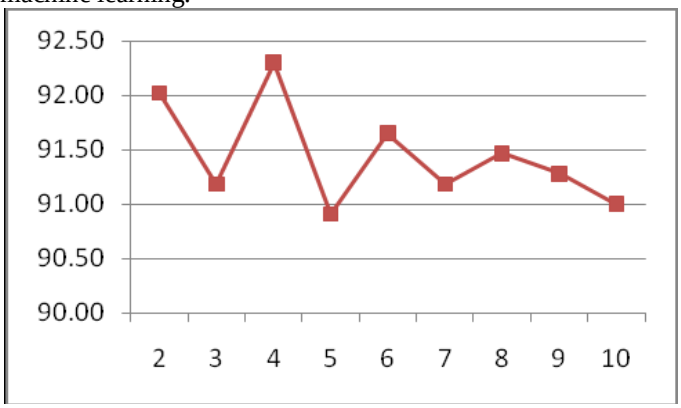


Figure 1 Performance of Naive Bayes (NB) with k cross validations (k=2 to 10)

Figure 1 provides the overall accuracy rate for malware detection achieved through our experiments using Naive Bayes with k cross validations, k= {2,3,4,5,6,7,8,9,10}. An advantage of the naive Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

V. PROPOSED WORK

Programs that have the potential to invade privacy and security of system are given a term Potentially Unwanted Programs (PUP). These programs include virus, Spyware, adware, Trojan, worms. These programs may compromise confidentiality, integrity, and availability of the system or may obtain sensitive information without the user's consent. In start, virus was the only malicious threat and since then much research has been done in this area. A more recent type of malicious threat is Spyware. According to the University of Washington's department of computer science and Engineering, Spyware is defined as "software that gathers information about use of a computer, usually without the knowledge of the owner of the computer, and relays the information across the Internet to a third party location". Another definition of Spy ware is given as "Any software that monitors user behavior, or gathers information about the user without adequate notice, consent, or control from the user".

Spyware may be capable of capturing keystrokes, taking screenshots, saving authentication credentials, storing personal email addresses and web form data, and thus may obtain behavioral and personal information about users. This can lead to financial loss, as in identity theft and credit card fraud .The knowledge about Spy ware is generally perceived as low among the common users and the process of Spyware identification or removal is generally considered as outside of their competence. It may show characteristics like nonstop appearance of advertisement pop-ups. It may open a website or force the user to open a website which has not been visited before, install browser toolbars without seeking acceptance from the user, change search engine results, make unexpected changes in the browser, and display error messages. Furthermore, indications of Spyware include a noticeable change in computer speed after installation of new software, auto opening of software or browser, a changed behavior of already installed software, network traffic without request, and increased disk utilization even in idle situations.

Some researchers have doubtingly predicted that advanced Spyware can possibly take control of complete systems in the near future. There is no single anti-Spyware tool that can prevent all existing Spyware because without vigilant examination of a software package, the process of Spyware detection has become almost impossible.

Spyware can be a part of freeware, plug-in, shareware, or illegal software. Normally, one would need a diverse set of anti-Spyware software to be fully protected. Anti-virus program may not be capable of detecting the Spyware until it has been designed for this purpose. Current anti-virus systems use signature-based methods or heuristic-based approaches against different malware. Signature-based Anti-virus systems use specific features or unique strings extracted from binary code. This method demonstrates good results for known viruses but lacks the capability of identifying new and unseen malicious code.

Heuristic-based systems try to detect known and unknown Malware on the basis of rules defined by experts who define behavior patterns for malicious and benign software. The heuristic method is considered costly and often ineffective against new Spyware. A heuristic approach, on the other hand, may detect novel threats with a reasonable accuracy. Anti-virus software is normally not designed with the focus on spyware but some experiments are done to prove that they can be used for Spyware detection. Consequently, we cannot be sure that they are capable of detecting new types of Spyware. So it may be possible to apply some other existing technologies that can help in finding new Spyware.

A new approach that can be used for the detection of Spyware is data mining. Data mining is widely adopted in various fields such as weather forecasting, marketing campaigns, discovering patterns from the financial data for fraud detection, etc. Data mining uses historical data for the prediction of a possible outcome in future. Data mining is an application of machine learning that is a subarea of Artificial Intelligence (AI). Machine learning is a study of making a system intelligent that learns automatically to make correct predications or to act intelligently without human assistance. Machine learning encompasses with different fields especially statistics but mathematics and computer science as well. Reference [38] has applied data mining approach for the detection of worms and built a classification model which secured 94.0% of overall accuracy with random forest classifier. Many Spyware are considered legal but yet could be dangerous to the computer systems.

In 2005, the US Federal Trade Commission (FTC) prosecuted Seismic Entertainment Productions and stopped them infecting consumer PCs with Spyware. According to the commission they had developed a method that detained control of computers nationwide by spreading Spyware and other malicious software and by flooding advertisements to their clients, this breach had made computers work slowly or stopped them from working. In the end Seismic released their anti-Spyware software to counter all problems that they themselves had created and earned more money than what had been earned previously by spreading the Spyware.

VI. EXPERIMENTAL WORK

Implementation is the carrying out, execution, or practice of a plan, a method, or any design for doing something. In an information technology context, implementation encompasses all the processes involved in getting new software or hardware operating properly in its environment, including installation, configuration, running, testing, and making necessary changes.

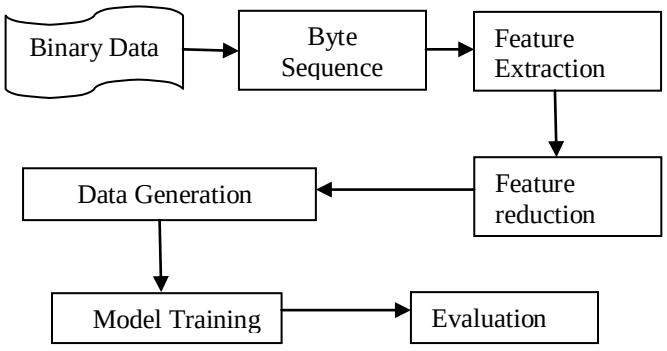


Figure 2: Proposed model

A. Data Collection

Data set consists of 100 binaries out of which 90 are benign and 10 are spyware binaries. The benign files were collected from Download.com, which certifies the files to be free from spyware. The spyware files were downloaded from the links provided by SpywareGuide.com. This hosts information about different types of spyware and other types of malicious software.

B. Byte Sequence Generation

We have opted to use byte sequences as data set features in our experiment. These byte sequences represent fragments of machine code from an executable file. We use xxd, which is a UNIX-based utility for generating hexadecimal dumps of the binary files. From these hexadecimal dumps we may then extract byte sequences, in terms of *n*-grams of different sizes.

C. Feature Extraction

The output from the parsing is further subjected to feature extraction. We extract the features by using following approaches, the Common Feature-based Extraction (CFBE) and Frequency-based Feature Extraction. The occurrence of a feature and the frequency of a feature. Both methods are used to obtain Reduced Feature Sets (RFSs) which are then used to generate the ARFF files.

D. Dataset Generation

Two ARFF databases based on frequency and common features were generated. All input attributes in the data set are represented by Booleans. These ranges are represented by either 1 or 0.

E. Classification

A Naive Bayes classifier is a probabilistic classifier based on Bayes theorem with independence assumptions, i.e., the different features in the data set are assumed not to be dependent of each other. This of course, is seldom true for real-life applications. Nevertheless, the algorithm has shown good performance for a wide variety of complex problems. J48 is a decision tree-based learning algorithm. During classification, it adopts a top-down approach and traverses a tree for classification of any instance. Moreover, Random Forest is an ensemble learner. In this ensemble, a collection of decision trees are generated to obtain a model that may give better predictions than a single decision tree.

F. Bayes classifier

A Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model". In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be

considered to be an apple if it is red, round, and about 4" in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple. Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting.

In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods. In spite of their naive design and apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification methods in 2006 showed that Bayes classification is outperformed by more current approaches, such as random forests. An advantage of the naive Bayes classifier is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

VII. FUTURE ENHANCEMENT

For future work we can collect large collection of binary files and we can evaluate our approach when the dataset features represent opcode instead of bytes. Additionally, we aim to develop a hybrid spyware identification method that is based on the combination of EULA-based and executable based detection techniques.

VIII. CONCLUSION

Data mining base malicious approach have been proven to eb successful in detecting viruses and worms. Data mining techniques perform better than traditional techniques such as signature-base detection and Heuristic-based detection Since no suitable dataset was available we collected spyware and byte sequences and generated a data set.

IX. REFERENCE

[1] Asaf Shabtai, Eitan Menahem, and Yuval Elovici (2011), "F-Sign: Automatic, Function-Based Signature Generation for Malware", IEEE Transaction on System, Man, and Cybernatics.

[2] Yamini.K, Sivapriyadarshini.S (2010), "A Data Mining Approach For On Time Detection Of Spyware Threat", IEEE International Conference on Computational Intelligence and Computing Research.

[3] Sufal Das, Banani Saha, "Data Quality Mining using Genetic Algorithm", International Journal of Computer Science and Security, (IJCSS) Volume (3) : Issue (2)

[4] McAfee(2005), "Potentially Unwanted Programs: Spyware and Adware", http://www.mcafee.com/us/local_content/white_papers/wp_antiSpyware_shadesofgray.pdf

[5] Arastouie.N, Razzazi.M.R (2008), " Hunter: An Anti Spyware for windows Operating System", Information and Communication technologies, ICTTA, 3rd International Conference on Digital Object Identifier:10.1109/ICTTA.2008.4530281, Page(s): 1 – 5.

[6] The silent epidemic of 2005: 84% of Malware on computers worldwide is Spyware, http://www.pandasecurity.com/

[7] Henchiri.O, Japkowicz.N (2006), "A Feature Selection and Evaluation Scheme for Computer Virus Detection", Data Mining, ICDM '06. Sixth International Conference on Digital Object Identifier: 10.1109/ICDM.2006.4 Publication Year: 2006 , Page(s): 891 – 895

[8] Moskovitch.R, Feher.C, Tzachar.N, Berger.E, Gitelman.M, Dolev.S, and Elovici.Y (2008) "Unknown Malcode Detection Using OPCODE Representation", ISI 2008, June 17-20, Taipei, Taiwan.

[9] Bozagac.C.D, "Application of Data Mining based Malicious Code Detection Techniques for Detecting new Spyware", White paper, Bilkent University 2005.

[10] Ming-Wei Wu, Yi-Min Wang, Sy-Yen Kuo, Yennun Huang (2007), "Self-Healing Spyware: Detection, and Remediation", Reliability, IEEE Transactions on Volume: 56,Issue:4, Page(s): 588 - 596 Cited by: 2

[11] Yanfang Ye, Tao Li, Qingshan Jiang, Youyu Wang (2010), " CIMDS: Adapting Post processing Techniques of Associative Classification for Malware Detection", Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on , Issue:3, Volume: 40, Digital Object Identifier: 10.1109/TSMCC.2009.2037978 Page(s):298-307.

[12] Jiong Zhang, Zulkernine, M., Haque, A, "Random-Forests-Based Network Intrusion Detection Systems", Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transaction, Volume: 38 , Digital Object Identifier: 10.1109/TSMCC.2008.923876, isuues:3, Page(s): 649 - 659 Cited by: 1.

[13] Wenke Lee, Sal Stolfo, and Kui Mok (1999), "A Data Mining Framework for Building Intrusion Detection Models". IEEE Symposium on Security and Privacy.

[14] Arnold.W, Tesauro. G (2000), "Automatically Generated Win32 Heuristic Virus Detection", Proceedings of an International Virus Bulletin Conference.

[15] Schultz.M. G, Eskin.E, Zadok.E, and Stolfo.S. J (2007), "Data Mining Methods for Detection of New Malicious Executables", IEEE Symposium on Security and Privacy.