Static Malware Analysis Using Machine Learning Methods

Hiran V. Nath[1,2] and Babu M. Mehtre[1]

[1] Center for Information Assurance & Management,
Institute for Development and Research in Banking Technology, India
{hvnath,bmmehtre}@idrbt.ac.in
[2] School of Computer and Information Sciences (SCIS),
University of Hyderabad, India

Abstract. Malware analysis forms a critical component of cyber defense mechanism. In the last decade, lot of research has been done, using machine learning methods on both static as well as dynamic analysis. Since the aim and objective of malware developers have changed from just for fame to political espionage or financial gain, the malware is also getting evolved in its form, and infection methods. One of the latest form of malware is known as targeted malware, on which not much research has happened. Targeted malware, which is a superset of Advanced Persistent Threat (APT), is growing in its volume and complexity in recent years. Targeted Cyber attack (through targeted malware) plays an increasingly malicious role in disrupting the online social and financial systems. APTs are designed to steal corporate / national secrets and/or harm national/corporate interests. It is difficult to recognize targeted malware by antivirus, IDS, IPS and custom malware detection tools. Attackers leverage compelling social engineering techniques along with one or more zero day vulnerabilities for deploying APTs. Along with these, the recent introduction of Crypto locker and Ransom ware pose serious threats to organizations/nations as well as individuals. In this paper, we compare various machine-learning techniques used for analyzing malwares, focusing on static analysis.

Keywords: Malware, Static Analysis, Machine Learning, Advanced Persistent Threat, Cyber Defence.

1 Introduction

In the current scenario, the communication network forms a backbone of any industry. Any security breach, of these systems or networks is of major concern to the organization as well as for the society. Compromising of one or more of these leads to violation of confidentiality, integrity or availability. With the introduction of "IoT" Internet of Things, everything starting from household equipments like refrigerator to a host in mission critical areas (like a nuclear power plant or a pacemaker) are connected to internet. These devices could be controlled or monitored from anywhere around the world. Here security of all these devices plays a vital role.

Initially all research were focused on the analysis after the incident has occurred. Recently, the trend has been changed almost to a proactive approach, which is commonly known as Cyber Defense. Currently researchers are focusing on finding out methods to prevent the incident from occurring rather than analyzing the incident once it occurred. Here, they are using a predictive analysis method, by finding what would be the actions or methods that would be used by an attacker, to compromise the system.

Malware analysis is generally classified into two categories, Static and dynamic analysis. In static analysis, we would be classifying the file based on various features extracted from the executable. Disassembly is one of the methods, which is used for extracting various features from the executables. Here certain features are OPCODES, byte sequence, PE Header [30], etc. Shafiq at el. in [37] have described about a PE-Probe method through which we could detect previously unknown or zero-day malwares. Here the detection method works after classifying whether the malware is packed or not. They have claimed that it could be implemented for real-time analysis.

In dynamic analysis, we would be generally executing the malware in isolated environment for analysis. This could be a debugger or a virtualized or sandbox environment. This method is very much handy once the code is obfuscated. Here the analysis is based on the sequence of system calls initiated by the program. However, the drawback of this method is that, here not all the execution paths can be traced. Certain malwares are coded to behave in specific manner when they detect a debugger in the execution environment, or when these are executed in a virtualized environment. In case of obfuscated malware, each module would be decrypted only if that module is executed. So it would be very difficult to predict the actual behavior of the file.

In [36], Shabtai at el. have done a state of art survey on statically malware analysis. They have come to a decision that an individual classifier used on a single feature will not be useful. They have predicted that a weighted average of multiple classifiers on different features could give better results. Lot of research has been done on this field after 2009, and the methods used by attackers have improved a lot. Our aim is to analyze the current scenario, with the introduction of targeted and persistent attacks.

In section 2, we describe latest trends employed by malware developers, which is followed by various methods used in statical malware analysis categorized based on the features in section 3. In section 4, we discuss the impact of these methods on latest malware trends. Finally, in section 5 we conclude the paper by providing various types of malware attacks that is yet to be studied.

2 Recent Malware Trends

Recently, hackers are concentrating on compromising systems and networks for financial or political gains, rather than just for fame, which was their initial motivation. Cyber-attacks are initiated using highly sophisticated techniques, which is difficult to detect and defeat using current available methods. They are depending on multi-stage attacks for achieving their goals. Malicious software, known as malware, is one of the major tools used by these cyber criminals at one or more of these stages. These malwares can be of any form, ranging from integrated chips to Office or PDF documents attached to mail. So, preventive defense must be augmented for timely detection of attacks and initiation of responses. In this type of attack, usually the first stage will be using browser level exploits, like when you visit some malicious website, it executes some script or flash that launches some code, which is executed as a separate process. The second stage then downloads and runs the final payload onto the computer. This could be also considered as similar to that of k-ary codes, which is discussed later in this article. This accomplishes two objectives:

1. The attack is more likely to bypass some security checks or inspections by appearing less harmful.
2. The source of the attack is not easily identified by forensic analysis.

This model could be extended to any stage, in which the final or intermediate payload could be malware, which belongs to any of these families, like Zeus, Torpig, Gozi, Shylock or even a new one formed by zero-day exploits. These type of exploits are used to infect targets globally. The current infection rate is very high which proves the effectiveness of this multi-stage method.

Becker at el. [3], have proved that extremely stealthy hardware trojans can be embedded into integrated circuits, even below the gate level. Here they have inserted trojan by changing the dopant polarity of the existing transistors. Similarly, Lin at el. [22] have proposed about a hardware trojan which will induce physical side channel to convey secret information. In case of pdf documents even though its known that exploits could be written in Visual Basic Script and JavaScript's. Recently Filiol at el. [6] have shown that adobe portable document format by itself is a true programming language. This increases the risk associated with the PDF language based malware. Along with these, once the attacker uses code obfuscation, it becomes very hard for the researchers to analyze the programs. These methods are generally used for creating targeted attacks at multiple stages. In case of targeted attacks, the attacks are focused to specific targets like, an industry, or a country, or to accomplish a specific purpose like political espionage[21].

From 2009 onwards, there were series of targeted attacks, victimizing either an entire country or some particular industries. In most of these, malicious code has played a significant role. The published well-known targeted attacks are given bellow.

1. Aurora attack have targeted Google and was discovered in 2009[43].
2. Stuxnet or Flame or Duqu or Gauss targeted Iran's nuclear program, discovered in 2010 [26,4].
3. HTran or HUC (Honker Union of China) Packet Transmit Tool used for RSA breach, Found in 2011[27].
4. Madi or Mahdi had almost 800 victims in Iran & Israel[29].
5. Red October or IXESHE was targeted for Diplomatic, Governmental and Scientific Research Organizations[2,31].
6. MANDIANT targeted for Hong Kong[21].
7. ICEFOG targeted for Japan and South Korea[1].

The oldest known targeted attack was against Iran's nuclear program with well-known malware named Stuxnet. As per, McDonald at el. the C&C server used was registered on November 2005 for version 0.5 and the latest known version 1.x, was programmed to stop infection on June 2012. "ICEFOG" [1] is one of the most recent one, which was discovered in 26 September 2013. Here, they have used various known exploits like CVE-2012-1856, CVE-2012-0158, CVE-2013-0422 and CVE-2012-1723. In addition, they have used exploits in HLP (Windows Helper) and HWP exploits (Hangul Word Processor). This belongs to the category of targeted attacks since Hangul Word Processor is mainly used in South Korea, especially in the government sector.

These targeted attacks, form a superset of Advanced Persistent Threats (APTs)[40]. These APTs are multi-stage attacks, where multiple methods from social engineering to zero-day [28] exploit are used for accomplishing the task. Recently Dube at el. [11,9,10,12,13] have introduced a novel architecture which is known as Malware Target Recognition (MaTR). They claimed that their system is capable of capturing latest threats. Liu at el. [23] came out with an n-Victim approach to find out victims of APTs. Here they assumed that all the machines would be having similar network traffic due to C&C channel and, used N-gram method to analyze the network traffic.

Recently, there is a tremendous rise in the number of ransomware threats in internet. Many different variants of these threat are there, generally known as Cryptolocker and Ransomware. These threats hijack computer or its data and demand that a payment is made in order to unlock or decrypt them. The authors of these malicious threats have a strong financial motive for infecting as many computers as possible, and have put substantial resources in making these threats prevalent. New variants are being released frequently. Till date, there are no specific antivirus, capable of capturing Cryptolocker and Ransomware or their variants. If system gets infected and encrypted, they have to pay and wait for the mercy of the attacker/hacker. Symantec have come out with a report illustrating the percentage of infection, based on region. As per their document, greater than 10% of Indian community is infected with this malware.

Antivirus generally employee signature based methods for capturing malware. By performance analysis, it has been found that signature based system is very much inefficient compared to other static analysis methods. Current antivirus tools use sandbox or cloud environment for analyzing malware. In [14], Filiol has described k-ary malicious codes. Detection of k-ary codes is proved to be NP-Complete problem. Here k-ary means that instead of malware being delivered as a single file, it would be delivered as k files, which are mutually, disjoint set. It would be malicious only if all the k files are executed in a sequential or a parallel order. In [15], Filiol have came out with a method of combining k-ary codes with malicious cryptography techniques to amour code, which fails both static and dynamic reverse engineering techniques. A known example of this type of virus is 2-ary virus, which work in combination of W32.Qaz virus and W32.Funlove virus as referred in [16].

From long before itself, software or platform dependent malwares are there. Recently Vasiliadis in 2010 [41] have proposed about identifying and infecting the system based on the GPGPU card in that machine. More recently Denos at el. in [8] have proposed about a hardware depended malware, where it infects the system based on the onboard processor chips. Here the system works by identifying the family of processors' based on the Floating Point Arithmetic (FPA). These techniques are major stepping stones for hackers who are engaged in targeted attacks or APTs.

3 Methods for Static Analysis of Executables

Various data mining or machine learning methods are used for static analysis of malwares. The main bottleneck faced by researchers is the lack of enough number of training samples. The initial process for almost all of these methods is to obtain the disassembled code of the malicious program. Here the selection of feature and the machine-learning algorithm forms a unique method.

3.1 *n*-Gram

n-gram analysis is the method of analyzing byte sequences in a file. Here n stands for the number of bytes taken to form a single sample. n could take any values like 1,2,3,4... depending upon the designer. In [19,20,17,18], Kolter and maloof have came up with n-gram analysis. Here, they have fixed the value of n as 4 with a 1 byte sliding window after many trial and error. That is, instead of taking disjoint n-grams, they have taken over lapping n-gram patterns. From almost 1,651 malware samples, they able to create 255,904,403 distinct n-grams. Top 500 n-grams were selected as training set for analyzing the performance of various machine learning algorithms. Comparative study of naive Bayes, Support Vector Machine, Decision Tree (C 4.5) and its boosted version, by combining multiple classifiers were used. Here AdaBoost algorithm was used. The executables will be divided into ten disjoint sets, the first nine set were considered as training samples, and the final set was used as testing sample, for conducting ten-fold cross-validation. Here the best performance was given by boosted decision tree. They have obtained a true-positive rate of 0.98 for a desired false-positive rate of 0.05. This result was out-performed by MaTR architecture proposed by Dube at el.. For lower number of samples n-gram approach works fine, but as the number of sample increases the performance of the system decreases, but it stays better than the performance obtained by well-known antivirus.

In [44], Zhang at el. have also used n-gram as features based on the entropy (information gain). They have proposed a multi-classifiers system, which was built using probabilistic neural network. Here, they are using individual classifier to get the evidence, which is combined by Dempster-Shafer combination rules.

3.2 Byte Sequence

Information density or entropy is a method for measuring uncertainty in a series of numbers or bytes. Entropy measures the level of difficulty or the probability of independently predicting each numbers in a series. Here the basic assumption is that, if a file is encrypted, then its byte sequence would be very much scrambled, which in turn increases the information content of the file. This method is used mainly to identify the packed and encrypted malware. It was proposed by Lyda at el. in [25], where statically variation of malware executable is examined. Here a general assumption is taken like packing and encryption would be only used to conceal the code in a malicious executable. Lyda at el. have initially developed a bintropy, a binary file entropy analysis tool. Their experiments have shown that the entropy value would be high if the file is encrypted. Here they have analyzed 21,576 PE formatted tool. Here they have developed entropy metric, which could be generalized for any packed executables. This method is good for small size file whose size is less than 500KB.

3.3 OPCODE

An opcode stands for ‘Operation Code’. An opcode is a single instruction that can be executed by the CPU. In machine language, it is a binary or hexadecimal value, which is loaded into the instruction register. In assembly language mnemonic, an opcode is a command such as MOV or ADD or JMP. Santos at el. in [33,32,34], have designed an machine learning based classifier which works based on frequency of appearance of opcode sequence. Here, they have used opcode sequence of length 2 and have discarded the parameters used by opcode. The assumption taken is that malicious operations take more than one machine code. Here they are using semi-supervised machine learning technique. It is very much useful when only small number of labeled data exit for each class. Roc-SVM method is used for partially labeled data. In their analysis, they have studies various cases like whether it is better to label malicious or being software, the optimal number of labeled instance and its impact on final accuracy and about how to reduce the labeling efforts. They have compared their results with simple Euclidean distance with malware labeled and with legitimate software labeled. They have also done 10-fold cross validation on 900 instances for each fold.

In [5], Bilar have used statistical data for detecting malicious executables. It was also based on the frequency distribution of opcode over malicious as well non-malicious samples. He has indicated that frequency of top five opcodes is same for malware as well as good-ware. They came out with 14 rare opcodes, like bt, fdvip, fild, fstcw, imul, int, nop, pushf, rdtsc, sbb, setb, setle, shld, std which could be considered as best feature for classification. These opcodes are more in malicious executables.

3.4 Portable Executable Header

Many researchers have used header information's extracted from portable executables as their feature set for classifiers. Majority of them works with assumption that the programs behavior could be predicted by analyzing PE Header details. This will store the information like the DLL's that the files is linked to. In 2001, Schultz at el. have came up with data mining approach for detecting new malicious executables [35]. The dataset he has analyzed was 3,265 malicious programs and 1,001 clean programs. Out of this data set, 38 malicious and 206 clean PE files were also analyzed. They have used list of DLLs used, list of DLL function calls, and number of different function calls per DLL as their feature set. They have used "Repeated Incremental Pruning to Produce Error Reduction" (RIPPER), which was proposed by Cohen in [7] as the classifier. The other classifiers they have used are Naive Bayes and Multi-Naive Bayes Classifier. This could be termed as the starting point for using header details for file classification.

PE-Probe. In [37] Shafiq at el., have came up with a system know as PE-Probe. Here the system works in two levels. The first level is a packer detector, which classifies between packed and non-packed executables and will be given as input to second level. Similar to other systems, for detecting packers, they uses features like number of standard, non-standard, executable section, read or write executable section, entries in import address table (IAT), along with the entropy of PE header, code section, data section, and entire PE File. This classification is done using multi-layer perceptron (MLP). Here, they are using PE-Miner from [39,38]. The PE-Miner was trained and tested on a large set of malicious executables files taken from VX-Heaven. In this, various preprocessing techniques like Redundant Feature Removal (RFR), Principal Component Analysis (PCA), Haar Wavelet Transform (HWT). The classification was done using five classifiers: instance based learner (IBk), decision tree (J48), Naive Bayes (NB), inductive rule learner (RIPPER), and support vector machines using sequential minimal optimization (SMO).Decision tree (J48) gave better detection accuracy compared to other classifiers. Here they have claimed to have better accuracy and performance than n-gram approach. The claim is that, *n-gram based techniques are more suitable for classification of loosely structured data; therefore, they fail to exploit format specific structural information of a PE file.*

Malware Target Recognition (MaTR). In this method, Dube at el. have claimed to have 98.5% efficiency. In [11,9,10,12,13], they have used bagged decision tree classifier. The classification was done based on the structural anomalies like, section names, section characteristics, entry point, imports, exports and alignment in PE Header. Here the prototype implementation was done using

TreeBagger in MATLAB. Here they have claimed to outdate Kolter and Maloof's *n*-gram approach. Their training set was 32-bit malwares obtained from VXHeaven until 2010. They have used 25,195 clean and 31,147 malicious samples. They have also claimed that they have tested the files, which belong to APT class, and the system is capable of detecting APT. However, they have not mentioned it clearly, whether their training set contains any APT or not. In addition, they have not mentioned what are the specific features with which they could capture targeted attacks.

Table 1. Comparison of various Static Malware Analysis Methods

Researcher	#Malware	#benign	Algorithm	Feature	True Positive	False Positive
Kolter & Maloof	1651	1971	Boosted Decision Tree	n-gram	0.98	0.05
Lyda & Hamrock	21567	0	Bintropy	Byte Sequence	—	0.005
Santos at el.	17000	—	Roc-SVM	Opcode-Sequence	0.96	0.05
Bilar	67	—	Frequency	Opcode	—	—
Shafiq at el.	—	—	Decision Tree	PE Header	0.996	0.003
Dube at el.	31147	25195	Boosted Decision Tree	PE Header	0996	—

4 Discussion

Initially in 1998, White have came out with various open problems in computer virus[42].Here he have considered computer virus as a similar one to that of biological virus. In 2006 Filiol at el. have came out with an updated version of various open problem in computer[16]. Here, they have indicated that, there is a lack of deep research in computer virus. Lot of the researchers think like research in this area is not of great use, since it is mostly a cycle of getting new sample, creating signature, updating the antivirus rather than having a general system, which is capable of capturing even new malware variants. Here, they have indicated various theoretical problems, which have yet to be addressed by research community.

The major bottleneck faced by researchers in using machine learning for malware analysis is the lack of training datasets. Each day new variant or encrypted version is released. Generally, it is not possible for the researchers to decrypt the encrypted code. Even if he were doing dynamic analysis for decrypting, only a certain subroutine would be decrypted and executed. Using that information, it is not possible to predict whether it is malware or not. So general scenario which happens, is to train the system to detect and check for obfuscation or encryption. If so, it is probably considered as a malware. So almost all the encrypted files generated by known packers, are also considered as a malware. This problem is equally applicable in Entropy Analysis for bytes sequence and MaTR Architecture. In entropy analysis, the bytes would be totally scrambled if the code is obfuscated. Obfuscation is used not only just for creating malicious code, but for hiding the logic or proprietary information in a program.

In case of code obfuscation, none of the opcode analysis methods will work. Since, it is only possible for the program to extract opcode from none obfuscated code blocks. That is, It is only possible for the system to extract opcodes from code block of the loader, which loads the encrypted file to memory. If we were training the system with this frequency of the loader, then all the files, which use this loader, would be considered as malicious code, which increase false positive. It leads to the same scenario given above.

In MaTR architecture, they have claimed that the system is capable of even detecting advanced threats. However, in their paper, they are not having enough proof to justify their claim. We have tried to obtain their code, so to test for the claimed efficiency, but we were not successful in getting the implementation details or the code. Since they have used decision tree approach, the efficiency depends on the selection of parameters, which is being considered in each level/node of the tree. Here, they are considering the header details, which mainly focus on Obfuscation. Therefore, there is a probability for the system to have more false positives even though their results shows promising.

Many researchers have claimed to use ensemble or boosted versions or multiple classifier one after another to detect malicious executables. However, none of them has done a comparative analysis on which sequence to select so that they would able to get a precise result. They have to also decide precisely on what are the features that should be considered on each level for each data-mining algorithm. The above results, shows that decision tree give good result for features like n-gram and PE header. It is necessary to have a good study of various features and algorithm combination to predict the sequence.

5 Conclusion

From the above study, we could conclude that none of the current research focus on a solution for multi-stage delivery of malware or *k-ary* codes or APTs. The trend of infection and exploiting has totally shifted from having a single malicious file to multi-stage or multiple files executed in parallel or sequential. Therefore, it would be better if we could have some machine learning system, which could correlate the possible activities of different files in parallel or sequential mode before classifying it as malicious or benign. None of above research deals with Crypto lockers and Ransom ware, which is one of the serious threats, now days. Therefore, there is a lot to be done in malware research to address these problems. Current solutions are not adequate for solving latest threats.

References

1. The 'ICEFOG' APT: A tale of cloak and three daggers. Kaspersky Lab Global Research And Analysis Team(GREAT) (2013)
2. Balduzzi, M., Ciangaglini, V., McArdle, R.: Targeted attacks detection with spunge. Trend Micro Research, EMEA (2013)

3. Becker, G.T., Regazzoni, F., Paar, C., Burleson, W.P.: Stealthy dopant-level hardware trojans (2013)
4. Bencsáth, B., Pék, G., Buttyán, L., Félegyházi, M.: The cousins of stuxnet: Duqu, flame, and gauss. Future Internet 4(4), 971–1003 (2012)
5. Bilar, D.: Opcodes as predictor for malware. International Journal of Electronic Security and Digital Forensics 1(2), 156–168 (2007)
6. Blonce, A., Filiol, E., Frayssignes, L.: Portable document format (pdf) security analysis and malware threats. Tech. rep., Virology and Cryptology Laboratory, French Army Signals Academy (2008)
7. Cohen, W.W.: Fast effective rule induction. ICML 95, 115–123 (1995)
8. Desnos, A., Erra, R., Filiol, E.: Processor-dependent malware... and codes. arXiv preprint arXiv:1011.1638 (2010)
9. Dube, T., Raines, R., Peterson, G., Bauer, K., Grimaila, M., Rogers, S.: Malware type recognition and cyber situational awareness. In: Second International Conference on Social Computing (SocialCom), pp. 938–943. IEEE (2010)
10. Dube, T., Raines, R., Peterson, G., Bauer, K., Grimaila, M., Rogers, S.: Malware target recognition via static heuristics. Computers & Security 31(1), 137–147 (2012)
11. Dube, T.E.: A Novel Malware Target Recognition Architecture for Enhanced Cyberspace Situation Awareness. Ph.D Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio (September 2011)
12. Dube, T.E., Raines, R.A., Grimaila, M.R., Bauer, K., Rogers, S.: Malware target recognition of unknown threats. IEEE Systems Journal 7(3) (September 2013)
13. Dube, T.E., Raines, R.A., Rogers, S.K.: Malware target recognition. US Patent 20, 120, 260, 342 (October 11, 2012)
14. Filiol, E.: Formalisation and implementation aspects of k-ary (malicious) codes. Journal in Computer Virology 3(2), 75–86 (2007)
15. Filiol, E.: Malicious cryptography techniques for unreversable (malicious or not) binaries. arXiv preprint arXiv:1009.4000 (2010)
16. Filiol, E., Helenius, M., Zanero, S.: Open problems in computer virology. Journal in Computer Virology 1(3-4), 55–66 (2006)
17. Kolter, J.Z., Maloof, M.A.: Learning to detect and classify malicious executables in the wild. The Journal of Machine Learning Research 7, 2721–2744 (2006)
18. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: An ensemble method for drifting concepts. The Journal of Machine Learning Research 8, 2755–2790 (2007)
19. Kolter, J.Z., Maloof, M.A.: Learning to detect malicious executables in the wild. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 470–478. ACM (2004)
20. Kolter, J.Z., Maloof, M.A.: Using additive expert ensembles to cope with concept drift. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 449–456. ACM (2005)
21. Li, F., Lai, A., Ddl, D.: Evidence of advanced persistent threat: A case study of malware for political espionage. In: 6th International Conference on Malicious and Unwanted Software (Malware), pp. 102–109. IEEE (2011)
22. Lin, L., Kasper, M., Güneysu, T., Paar, C., Burleson, W.: Trojan side-channels: Lightweight hardware trojans through side-channel engineering. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 382–395. Springer, Heidelberg (2009)
23. Liu, S.-T., Chen, Y.-M., Hung, H.-C.: N-victims: An approach to determine n-victims for apt investigations. In: Lee, D.H., Yung, M. (eds.) WISA 2012. LNCS, vol. 7690, pp. 226–240. Springer, Heidelberg (2012)
24. Lu, Y., Din, S., Zheng, C., Gao, B.: Using multi-feature and classifier ensembles to improve malware detection. Journal of CCIT 39(2), 57–72 (2010)

25. Lyda, R., Hamrock, J.: Using entropy analysis to find encrypted and packed malware. IEEE Security & Privacy 5(2), 40–45 (2007)
26. McDonald, G., Murchu, L.O., Doherty, S., Chien, E.: Stuxnet 0.5: The missing link. Symantec Security Response (online) 26 (2013)
27. Menn, J.: Key internet operator verisign hit by hackers. Reuters (February 2, 2012)
28. Muttik, I.: Zero-day malware. In: Virus Bulletin Conference (2010)
29. Prosecutors, Public: Messiah spyware infects middle east targets
30. Rafiq, N., Mao, Y.: Improving heuristics. In: Virus Bulletin Conference, pp. 9–12 (2008)
31. Raymond, D., Conti, G., Cross, T., Fanelli, R.: A control measure framework to limit collateral damage and propagation of cyber weapons. In: Fifth International Conference on Cyber Conflict (CyCon), pp. 1–16. IEEE (2013)
32. Santos, I., Brezo, F., Sanz, B., Laorden, C., Bringas, P.G.: Using opcode sequences in single-class learning to detect unknown malware. IET Information Security 5(4), 220–227 (2011)
33. Santos, I., Brezo, F., Ugarte-Pedrero, X., Bringas, P.G.: Opcode sequences as representation of executables for data-mining-based unknown malware detection. Information Sciences (2011)
34. Santos, I., Nieves, J., Bringas, P.G.: Semi-supervised learning for unknown malware detection. In: Abraham, A., Corchado, J.M., González, S.R., De Paz Santana, J.F. (eds.) International Symposium on DCAI. AISC, vol. 91, pp. 415–422. Springer, Heidelberg (2011)
35. Schultz, M.G., Eskin, E., Zadok, F., Stolfo, S.J.: Data mining methods for detection of new malicious executables. In: Proceedings of the 2001 IEEE Symposium on Security and Privacy, S&P 2001, pp. 38–49. IEEE (2001)
36. Shabtai, A., Moskovitch, R., Elovici, Y., Glezer, C.: Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey. Information Security Technical Report 14(1), 16–29 (2009)
37. Shafiq, M., Tabish, S., Farooq, M.: Pe-probe: leveraging packer detection and structural information to detect malicious portable executables. In: Proceedings of the Virus Bulletin Conference (VB), pp. 29–33 (2009)
38. Shafiq, M.Z., Tabish, S.M., Mirza, F., Farooq, M.: A framework for efficient mining of structural information to detect zero-day malicious portable executables. Tech. rep., TR-nexGINRC-2009-21 (January 2009), `http://www.nexginrc.org/papers/tr21-zubair.pdf`
39. Shafiq, M.Z., Tabish, S.M., Mirza, F., Farooq, M.: Pe-miner: mining structural information to detect malicious executables in realtime. In: Kirda, E., Jha, S., Balzarotti, D. (eds.) RAID 2009. LNCS, vol. 5758, pp. 121–141. Springer, Heidelberg (2009)
40. Sood, A., Enbody, R.: Targeted cyber attacks-a superset of advanced persistent threats. In: IEEE Computer and Reliability Societies, Michigan State University (2013)
41. Vasiliadis, G., Polychronakis, M., Ioannidis, S.: Gpu-assisted malware. In: 2010 5th International Conference on Malicious and Unwanted Software (MALWARE), pp. 1–6. IEEE (2010)
42. White, S.R.: Open problems in computer virus research. In: Virus Bulletin Conference (1998)
43. Zetter, K.: Google hack attack was ultra sophisticated, new details show. Wired Magazine 14 (2010)
44. Zhang, B., Yin, J., Hao, J., Zhang, D., Wang, S.: Malicious codes detection based on ensemble learning. In: Xiao, B., Yang, L.T., Ma, J., Muller-Schloer, C., Hua, Y. (eds.) ATC 2007. LNCS, vol. 4610, pp. 468–477. Springer, Heidelberg (2007)