Australian Journal of Basic and Applied Sciences, 5(9): 1482-1486, 2011
ISSN 1991-8178

Computer Virus Detection Using Features Ranking and Machine Learning

Altyeb Altaher, Sureswaran Ramadass and Ammar Ali.

National Advanced IPv6 Center Universiti Sains Malaysia. Penang, Malaysia.

Abstract: Computer virus is a computer program that can copy itself and infect a computer, computer virus have become an increasingly important problem that threatens the security of computer systems. Traditional signature-based antivirus systems fail to detect polymorphic and new, previously unseen malicious executables. This paper presents a new method for virus detection using a combination of the voted perceptron classification algorithm and the Information Gain method. In this approach, the voted perceptron classification algorithm is used to detect the virus based on the analysis of Windows APIs called by Portable Executable (PE) and the information gain method is used to reduce the number of input features to the voted perceptron classification algorithm, by selecting the most important API Calls which reflect the behavior of the (PE). An experimental result shows 99 % accuracy and detection rate which underlines the capability of the proposed method.

Key words: Computer virus detection, data mining, Information gain, voted perceptron classification algorithm, signature based detection.

INTRODUCTION

"Malware" is an abbreviation for 'malicious software' and is used to refer to any software designed to cause damage to a single computer, server, or computer network. Malicious programs, commonly termed as malwares, can be classified into virus, worms, Trojans, spywares, Spams and a variety of other classes and subclasses that sometimes overlap and blur the boundaries among these groups (Szor. P., 2005) According to Kaspersky labs in April 2009 alone, 45190 unique instances of malware where found on their customers computers (Zakorzhevsky, 2011) This worryingly high number is only likely to increase, especially as the malware author's incentives for writing such software is now mainly a financial one. According to its propagation methods, malicious code is usually classified into the following categories (Adleman, L., 1990; Filiol, E., 2005; McGraw, G. and G. Morrisett, 2002) viruses, worms, Trojan horses, backdoors and spyware . Due to the significant loss and damages induced by malicious executables, the malware detection becomes one of the most critical issues in the field of computer security. Currently, most widely-used malware detection software uses signature-based method to recognize threats (Filiol, E., 2006; Filiol, E., *et al*., 2007) Signatures are sequences of bytes in the machine code of the malware.

The inability of traditional signature based malware detection approaches to catch polymorphic and new, previously unseen malwares has shifted the focus of malware detection research to find more generalized and scalable features that can identify malicious behavior as a process instead of a single static signature.

This paper presents a new method for computer virus detection using a combination of the voted perceptron classification algorithm and the Information Gain method. The proposed method can detect both known and unknown worms with high detection rate and accuracy. The main objective of this method is to use data mining techniques to learn the behavior of known computer viruses and then detect known and unknown viruses based on their behavior.

Related Works:

Besides the traditional signature-based malware detection methods (Kephart, J. and W. Arnold, 1994; Lo, R., *et al*., 1995) which failed to detected polymorphic and unseen malware , there is some work to improve the signature-based detection (Christodorescu, M. and S. Jha, 2003; Lee, T. and J. Mody, 2006; Rabek, J., *et al*., 2003; Sung, A., *et al*., 2004) developed a signature based malware detection system called SAVE (Static Analyzer of Vicious Executables) which emphasized on detecting polymorphic malware by calculating a similarity measure between the known virus and the suspicious code. The basic idea of this approach is to extract the signatures from the original malware with the hypothesis that all versions of the same malware share a common core signature. Although this work improves the traditional signature-based detection in polymorphic malware detection, it fails against the unknown malware. Attempts to apply data mining and machine learning techniques include One of these works (Kolter J. *et al.*, 2004) have used n-grams of 1971 benign and 1651 malicious executables as features. But n-grams is not a suitable feature for detecting polymorphic and metamorphic malware, because of its sensitivity to order. The other two works try to use DLL file names as features (Schultz, M. *et al.*, 2001) the problem with these researches is a small number of data set they used.

Corresponding Author: Altyeb altaher National Advanced IPv6 Center Universiti Sains Malaysia Penang, Malaysia.
E.mail.altyeb @nav6.usm.my

Ye *et al.* (2008) developed a system named IMDS, is the first attempt to use API calls. They have used data mining techniques and features generated from API calls. Their data set consist of 12,214 benign executables and 17,366 malware. They used about 2,000 PE files, which is claimed to be randomly selected from training data. Although experiments presented good results, due to industrial nature of research there is no access to their data set to regenerate their results for further investigations.

The Proposed Virus Detection Method:

The proposed approach is a combination of the information gain method and the voted perceptron classification algorithm for detecting computer viruses. The information gain will be used for selecting the quality of attributes. The output of applying the information gain method is a set of features with high ranking values, the set of high ranked features will be the input for the voted perceptron classification algorithm. The selected features will be used by the voted perceptron classification algorithm to detect the malware.

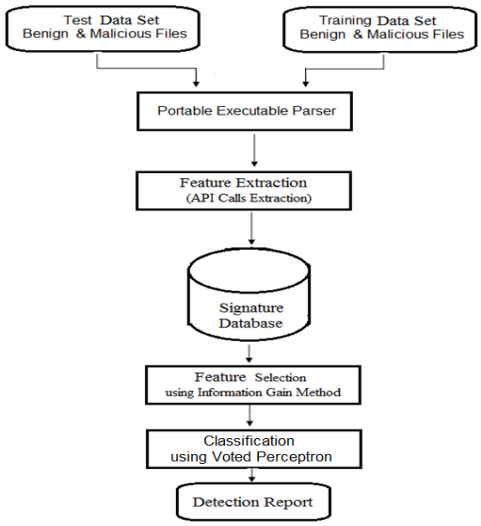

Fig. 1: Architecture of the proposed malware detection method.

Fig. 1. Shows the malware detection procedure of the proposed method:

First step: the proposed method uses the portable executable parser to extract information from the PE executables that would dictate its behavior.

Second step: After the portable executable parsing, the proposed method uses the feature extractor to extract the Windows API calls from the collected PE files, converts them to a group of 32-bit global IDs as the features of the training data, and stores these features in the signature database. A sample signature database is shown in Fig. 2, in which there are 5 fields: record ID, PE file name, PE file type ("1" represents benign file while "0" is for malicious file), called APIs name and the called API information gain.

Third step: In the feature selection stage, the proposed method adopts correlation measures based on the information theoretical concept of entropy, a measure of the uncertainty of a random variable. The distinguishing power of each feature is derived by computing its information gain (IG) based on its frequencies of appearances in the malicious class and benign class. Features with negligible information gains can then be removed to reduce the number of features and speed the classification process.

The entropy of a variable X is defined as:

$$H(X) = -\sum_i P(X_i) \log_2 P(X_i) \quad (1)$$

Where in H(P), the P(X) is as follows:

$$P(X_i) = \frac{Number\cdot of\cdot PE\cdot with\cdot x_i\cdot as\cdot certain\cdot API}{Total\cdot number\cdot of\cdot PE} \quad (2)$$

And the entropy of X after observing values of another variable Y is defined as:

$$H(X|Y) = -\sum_j P(Y_j) \sum_i P(X_i|Y_j) \log_2 (P(X_i|Y_j)) \quad (3)$$

The amount by which the entropy of X decreases reflects additional information about X provided by Y is called information gain, given by

$$IG(X \mid Y) = H(X) - H(X \mid Y) . \quad (4)$$

Table1

ID	PE File Name	PE File Type	Called APIs Name	Called Api Information Gain
1	INTERNET EXPLORER	1	CloseHandleCreateFileMappingWCreateFileWCrea	0.1970.3850.3850.1940.4140.279
2	GAMES FREECELL	1	GetCurrentProcessGetCurrentProcessIdGetCurrent	0.3650.3850.5250.2240.590.5250
3	GAME HEART	1	CreateFileACreateFileWCreateMutexWCreateThre	0.5250.2650.7580.5250.590.3650
4	TELPORT	1	DeleteCriticalSectionEnterCriticalSectionExitProces	0.3850.5250.22400000.33500.525
5	WINDOWS MEDIA PLAYER	1	CreateMutexWExitProcessExpandEnvironmentStri	0.5250.22400.52500.3970.33500(
6	WINRAR	1	BackupReadBackupSeekCloseHandleCompareStrin	0.590.52500.3970.33500.5250.75
7	GAMES MINESWEEPER	1	CloseHandleCreateDirectoryWCreateEventWCreat	0.1940.3970.33500.5250.7580.22
8	Helplibagent	1	DeleteCriticalSectionDisableThreadLibraryCallsEnt	0.590.52500.3970.3350.1970.525
9	Sidebar	1	CancelWaitableTimerCloseHandleCompareFileTim	00.590.5250.1940.3970.3350.197
10	MSASCUI	1	CloseHandleCompareFileTimeConvertDefaultLoca	0.3850.5250.22400.5250.1940.39
11	Virus.win32.arcer	0	CloseHandleCreateFileACreateThreadDeleteCritic	0.41400.145000.265000000000(
12	virus.win32.arch.a	0	DeleteFileAEnterCriticalSectionExitProcessFreeLib	0.41400.145000000000000000.33
13	virus.win32,aris	0	CopyFileACreateFileACreateProcessACreateThrea	000.145000.2650000000.2240
14	Joke.win32.Zapa	0	CreateFileADeleteCriticalSectionEnterCriticalSectic	00000.145000000000000000.335
15	win32.funlove	0	ExitProcessFlushFileBuffersFreeEnvironmentString	0.145000.2650000000.224000000(
16	virus.win32.bee	0	CopyFileACreateFileADeleteFileAExitProcessFind(	0.2240.617000000000000.1450.2
17	virus.win32.bonding	0	CreateDirectoryACreateFileACreateThreadCreateT	00.6170000.145000.6170000.145
18	virus.win32.elly	0	CreateThreadGetCommandLineWGetModuleHand	00.610.145000.6170000.1450000
19	virus.win32.glyn	0	CreateFileACreateFileMappingADeleteFileAExitPr	0.385000000000.385000000000(
20	virus.win32.Zorg	0	CreateFileADeleteCriticalSectionEnterCriticalSectic	0.41400000.41400.145000.26500

Fig, 2: A sample signature database.

Representative feature selection is important for many pattern classification systems (Jain, A., *et al*., 2000) Identifying the most representative features is critical to minimize the classification error, as not all of the API calls are contributing to malware detection (Sung, A., *et al*., 2004).

Fourth step, the proposed method uses the voted perceptron classifier (Freund, Y. and R.E. Schapire, 1999) to construct the malware detection classifier. The voted perceptron has advantages in ease of implementation, and efficiency.

To conduct the experiment, malicious executables were downloaded from the website VX Heavens (Accessed 2 Jan *et al.*, 2011) The used data set composed of benign programs and malicious executable codes. All the executables were in Windows PE format. The clean programs were gathered from a freshly installed Windows 7 operating system. All the experiments are conducted under the environment of Windows 7 operating system plus Intel Quad CPU 2.66 Ghz and 4 Gb of RAM.

To evaluate the efficiency of the proposed virus detection method, the proposed detection method is compared with other detection methods like Naïve Bayes, and Decision Tree methods. The executables are randomly selected from the data set, in which 50 % files are benign and 50 % executables are malicious. The API calls extracted and stored in the signature database then the information gain method has been used to rank each API call, the top 50 API calls are selected as features for later classification.

Table 1 and Fig.3. show sample of the ranking of features after applying the information gain algorithm on the API calls.

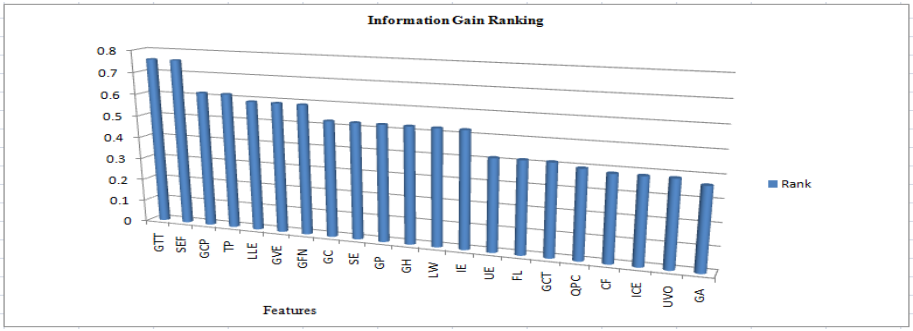

Fig. 3: Ranking of attributes after applying the information gain algorithm.

In the experiments, the voted perceptron classifier and J4.8 version of Decision Tree implemented in WEKA were used. The following estimates were used to evaluate the performance of the proposed approach:
True Positive (TP):Number of correctly detected malicious files.
False Positive (FP): Number of wrongly detected benign files.

True Negative (TN): Number of correctly detected benign files.
False Negative (FN): Number of wrongly detected malicious files.
Detection Rate (DR): Percentage of correctly detected malicious files

$$\text{Detection Rate} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Table 1: Information Gain Ranking.

Feature	Rank
GetSystemTimeAsFileTime (GTT)	0.758
SetUnhandledExceptionFilter(SEF)	0.758
GetCurrentProcess(GCP)	0.617
TerminateProcess(TP)	0.617
LoadLibraryExW(LLE)	0.59
GetVersionExW(GVE)	0.59
GetModuleFileNameW(GFN)	0.59
GetTickCount(GC)	0.525
SetLastError(SE)	0.525
GetCurrentProcessId(GP)	0.525
GetModuleHandleW(GH)	0.525
LoadLibraryW(LW)	0.525
InterlockedExchange(IE)	0.525
UnhandledExceptionFilter(UE)	0.414
FreeLibrary(FL)	0.414
GetCurrentThreadId(GCT)	0.414
QueryPerformanceCounter(QPC)	0.397
CreateFileW(CF)	0.385
InterlockedCompareExchange(ICE)	0.385
UnmapViewOfFile(UVO)	0.385
GetProcAddress(GA)	0.365

Table 2: Results by using different classifiers. TP, FP, DR, and PRE refer to True Positive, False Positive, Detection Rate, and precision, respectively.

Algorithm	TP	FP	DR	PRE
Decision Tree	0.90	0.10	90 %	90 %
Voted Perceptron	0.95	0.05	95 %	95 %
Our proposed method	0.99	0.01	99 %	99 %

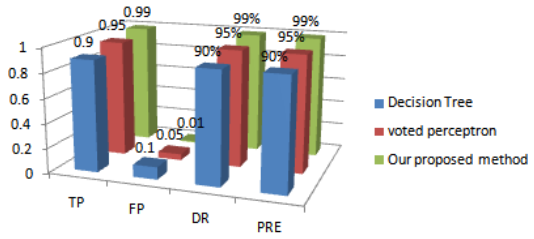

Fig. 4: Results by using different classifiers. TP, FP, DR, and PRE refer to True Positive, False Positive, Detection Rate, and Precision, respectively.

From the comparison in table 2 and Fig. 4, it's observed that the proposed virus detection method outperforms other classification methods in both detection rate and precision. This is because the use of information gain result in selection of significant features which make the classification is accurate.

Conclusion:

This paper proposes a computer virus detection method based on data mining technique, the proposed detection method combined the information gain algorithm and the voted perceptron classifier.

The proposed method is evaluated using a collection of executables including benign files and malicious files, a comprehensive experimental study has been provided on various data mining techniques for malware detection using the collected data set. The proposed malware detection method outperforms other classification methods in both detection rate and accuracy.

REFERENCES

Adleman, L., 1990. An abstract theory of computer viruses (invitedtalk). In: CRYPTO ’88: Proceedings on Advances in Cryptology, pp: 354-374.

Christodorescu, M., S. Jha, 2003. Static analysis of executables to detect malicious patterns. In: Proceedings of the 12th USENIX Security Symposium.

Filiol, E., 2005. Computer Viruses: from Theory to Applications.Springer, Heidelberg.

Filiol, E., 2006. Malware pattern scanning schemes secure against blackbox analysis. J. Comput. Virol., 2(1): 35-50.

Filiol, E., G. Jacob, M.L. Liard, 2007. Evaluation methodology and theoretical model for antiviral behavioral detection strategies. J. Comput. Virol., 3(1): 27-37

Freund, Y. and R.E. Schapire, 1999. Large margin classification using the perceptron algorithm. In Machine Learning, 37(3): 277-296.

Hashemi, S., Y. Yang, D. Zabihzadeh and M. Kangavari, 2008. Detecting intrusion transactions in databases using data item dependencies and anomaly analysis. Expert Systems, 25(5).

Jain, A., R. Duin, J. Mao, 2000. Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Mach. Intell., 22:. 4-37.

Kephart, J., W. Arnold, 1994. Automatic extraction of computer virus signatures. In: Proceedings of 4th Virus Bulletin International Conference, pp: 178-184.

Kolter, J. and M. Maloof, 2004. Learning to detect malicious executables in the wild. In Proceedings of KDD'04.

Lee, T., J. Mody, 2006. Behavioral classification. In: Proceedings of EICAR Conference.

Lo, R., K. Levitt, R. Olsson, 1995. Mcf A malicious code filter. Comput. Secur, 14: 541-566

McGraw, G., G. Morrisett, 2002. Attacking malicious code: report to the infosec research council. IEEE Softw., 17(5): 33-41.

Rabek, J., R. Khazan, S. Lewandowski, R. Cunningham, 2003. Detection of injected, dynamically generated and obfuscated malicious code. In: Proceedings of the ACM Workshop on Rapid Malcode, pp: 76-82.

Schultz, M., E. Eskin and E. Zadok, 2001. Data mining methods for detection of new malicious executables. In Security and Privacy Proceedings IEEE Symposium, pp: 38-49.

Sung, A., J. Xu, P. Chavez, S. Mukkamala, 2004. Static analyzer of vicious executables (save). In: Proceedings of the 20th Annual Computer Security Applications Conference.

Szor, P., 2005. The Art of Computer Virus Research and Defense. Addison Wesley for Symantec Press, New Jersey,

Vx virus dataset, Available from: http://vx.netlux.org [Accessed 2 Jan 2011].

Wang, J., P. Deng, Y. Fan, L. Jaw and Y. Liu, 2003. Virus detection using data mining techniques. In Proceedings of IEEE International Conference on Data Mining.

Witten, H. and E. Frank, 2000. Data mining: Practical machine learning tools with Java implementations. Morgan Kaufmann.

Ye, Y., D. Wang, T. Li and D. Ye, 2008. An intelligent pe-malware detection system based on association mining. In Journal in Computer Virology,

Zakorzhevsky, 2011. Monthly Malware Statistics. Available from: http://www.securelist.com/en/analysis/204792182/Monthly_Malware_Statistics_June_ [Accessed 2 July.