Hybrid Framework for Integrated Malware Analysis to Detect Advanced Persistent Threat

Thein Than Thwin, Mie Mie Su Thwin
University of Computer Studies, Yangon
theinthanthwin@gmail.com , miemiesuthwinster@gmail.com

Abstract

The term Advanced Persistent Threat (APT) is used as a replacement term for cyber warfare and malware has developed into the major vehicle for APT. Malware analysis and detection is a major resource in maintaining an organization's antivirus preparedness and responsiveness by contributing to the well-being of its IT health, and consequently to that of the economy as a whole. There is a need to develop an automatic malware detection and classification system to identify the variants of malware, in order to guide analysts in the selection of samples that require the most attention. In this paper, we introduce Hybrid Framework for our ongoing research, Integrated Malware Analysis to Detect Advanced Persistent Threat (APT). In our framework we integrated the static and dynamic malware analysis as well as K-mean clustering and Bayesian classification approaches.

Keywords: *Malware, Malware Analysis, Malware Classification, Malware Clustering.*

1. Introduction

The rise of Advanced Persistent Threats (APTs), a type of targeted attack, has continued to hit corporates and governments to infiltrating sensitive data. Organizations have found themselves the target of APTs. In the past, APTs were mostly directed at political and military targets, however, over the past few years, attackers increasingly use APTs to strike enterprise targets for financial gains. Most of the data breach investigation reports concludes that, in 86% of the cases, evidence about the data breach was recorded in the organization logs but the detection mechanisms failed to raise security alarms. This suggests that traditional defenses can become ineffective in detecting APTs and a new approach is required [13].

The definition given by US National Institute of Standards and Technology (NIST), states that an APT is: An adversary that possesses sophisticated levels of expertise and significant resources which allow it to create opportunities to achieve its objectives by using multiple attack vectors (e.g., cyber, physical, and deception). These objectives typically include establishing and extending footholds within the information technology infrastructure of the targeted organizations for purposes of exfiltrating information, undermining or impeding critical aspects of a mission, program, or organization; or positioning itself to carry out these objectives in the future. The advanced persistent threat: (i) pursues its objectives repeatedly over an extended period of time; (ii) adapts to defenders' efforts to resist it; and (iii) is determined to maintain the level of interaction needed to execute its objectives [13].

Analyses of specific APT instances concluded that each attack is unique and highly

customized for each target, however, across many attacks the stages of the APT are similar and they differentiate mostly in the specific methods used to pass each stage (Reconnaissance, Delivery, Exploitation, Operation, Data Collection and Exfiltration). APTs use a full spectrum of intrusion methods and technologies, therefore are very hard, if not impossible, to detect with conventional defense mechanisms but they can be detected at one of the operation stage.

So, we aims to perform malwares analysis on the APT related malwares. Based on the result of the analysis, the behaviors and the key functions of the malware will be extracted. And, a model that is accurate and robust for the detection of APT will be built. In this paper, we intended to support the weakness of dynamic analysis features with static analysis features and try to use supervised learning and unsupervised learning in hybrid.

2. Related Works

A brief review through some of the current research topics is provided below.

In (***K. Raman, "Selecting features to classify malware", In InfoSec Southwest, 2012.***), Raman used static PE file features for classification. Features are selected from metadata of PE file. He selected Debug Size, Image Version, IatRVA, Export Size, Resource Size, Virtual Size, and Number Of Sections features from PE metadata [9]. He tested IBk, J48, J48 Graft, PART, Random Forest, and Ridor classification algorithms on data set of 100000 malwares and 16000 clean files using the seven features as input to these algorithms. With J48 algorithm he achieved accuracy of 98.5 %.

In (***A. Lakhotia, A. Walenstein, C. Miles, A. Singh,"VILO: a rapid learning nearest neighbor classifier for malware triage", Journal of Computer Virology and Hacking Techniques, 2013***), Lakhotia et al. presented VILO, a malware classification system that make use of three components – N-perm feature vectors, Term Frequency × Inverse Document Frequency (TFIDF) weighting of features and the nearest neighbor search algorithm. N-perm is variation of N-gram. N-perms are extracted from disassembly of an executable. They evaluated their system on four malware family sets by choosing 100 samples from each set. They divided 100 samples into 40 and 60 for training and verification respectively. They achieved 86.44% accuracy for classification [1].

In (***R. Canzanese, M. Kam, S. Mancoridis, "Toward an automatic, online behavioral malware classification system," in IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems (SASO),*** 2013), they presented an automatic classification system based on behavior features extracted from sensors. They selected three distinct types of features - performance monitor, system call and system call sequence - for classification. They used decision trees and random forest algorithms for classification. They evaluated classification system on set of 800 malware and achieved 99% accuracy.

Mohaisen et al. introduced AMAL, an Automated behavior based Malware Analysis, classification and clustering system in (***A. Mohaisen, O. Alrawi, M. Larson, "AMAL: Highfidelity, behavior-based automated malware analysis and classification", 2013***). AMAL contains two subsystems, AutoMal and MaLabel. AutoMal collect malware behavior that contains file system, memory, network and registry modifications. MaLabel uses these artifacts to create features for classifications. It also enables unsupervised learning using multiple clustering algorithms. They evaluated

on two sample sets of 4000 and 115000. They achieved 98% accuracy in classification [2].

In (***Rafiqul Islam, Ronghua Tian, Lynn M.Batten, SteveVersteeg, Classification of malware based on integrated static and dynamic features, Journal of Network and Computer Applications, 2013***), Ronghua Tian et al. analysis malwares in two group (2003 – 2008) and (2009-2010). The first group contains 881 of malware and second group contains 1517.They firstly unpacked each of the 2398 malware files and extract static features (FLF, PSI) and extract API feature, comprising API function names and parameters by running all the executable files. They integrated the static and dynamic feature and classified. They got the accuracy of 99.8 in first group and 94.4 in second group [15].

3. Malware Analysis

Malware analysis is the process of determining the purpose and characteristics of a given malware sample such as a virus, worm, or Trojan horse [4,8]. There are basically three malware analysis techniques exits:

1. Static malware analysis
2. Dynamic malware analysis
3. Hybrid malware analysis

3.1. Static Analysis

Static analysis technique analyzes executable code without actually executing the file. It extract the low level information from codes which gathered by decompiling or disassembling the codes with the use of any disassembler tools [5,7]. It can cover all possible execution paths of a malware sample and is faster and safer than dynamic analysis. But it may be difficult in analyzing unknown malware and requires good knowledge of assembly language and the working operating system [17].

3.2. Dynamic Analysis

Dynamic analysis technique involves executing the malware and monitoring its behavior. It analyses infected files in simulated environment like a virtual machine, simulator, emulator, sandbox etc. It is needs to make the working environment invisible to the malware as the malware writers use anti-emulation and anti-virtual machine tools for hiding the malware functions [12]. It is easy to detect unknown malware but difficult to analyze multipath malware. It may fails to detect an activity which shows the behavioral changes in codes by different trigger conditions during the course of its execution.

3.3. Hybrid analysis

This technique includes the combinatorial approach of both dynamic and static analysis. It analyses about the signature specification of any malware code then combine it with the other behavioral parameters for enhancement of complete analysis. It can overcome the limitations of both static and dynamic analysis [6,11].

4. Our Approach

This section describes architecture and our approach towards the design of the system. The system architecture is given in figure 1.

Our system consists of four main processes

1. Data Collection
2. Preprocessing
3. Feature Extraction
4. Classification

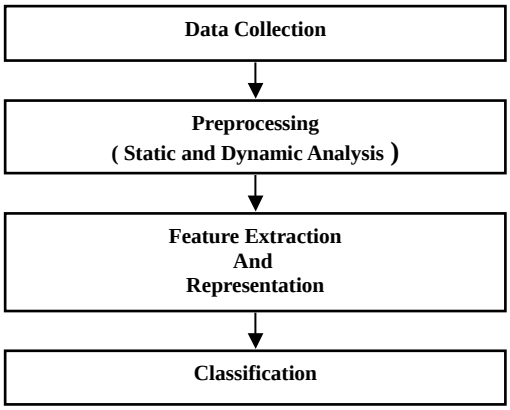

Figure 1: Architecture of the System

4.1. The Purposes of the Proposed Approach

The majority of anti-virus detection systems are signature-based which was effective in the early stages of malware. To deal with the rapid development of malware, it is needed to shift from a signature-based method to new approaches based on either a static or dynamic analysis to detect and classify malware. It is needed to focus on finding out methods to prevent the incident from occurring rather than analyzing the incident once it occurred. The predictive analysis method, finding what would be the actions or methods that would be used by an attacker, to compromise the system should be used. Most of the existing malware analysis systems performed analysis on a single event or single malware and cannot detect the APT.

4.2. Data Collection

In this process we collect the APT cases and extract the malwares that are used. The published well-known targeted attacks can be searched and collected online. We also collect test samples, including other malicious software and benign software.

4.3. Preprocessing and Feature Extraction

In this process perform the two sub processes, static analysis and dynamic analysis to extract features.

In our static Preprocess, we unpacked the malware before passing them to IDA Pro, the reverse-engineering software. Malware makers have always strived to evade detection from anti-virus software [16]. Code obfuscation is one of the methods they use to achieve this. In our unpacking pre-processing, we used VMUnpacker-1.6. After we obtained the unpacked executable files, we used reverse-engineering techniques to perform our static analysis. We chose IDA Pro as our main reverse-engineering analysis tool because IDA Pro is a Windows or Linux hosted, programmable, interactive, multi-processor disassembler and debugger that offer many features. Then, we extract the static features from the sample.

In the dynamic analysis, we execute each file under a controlled environment which is based on Virtual Machine Technology. Before starting dynamic analysis, a snapshot of the virtual machine was taken and we needed to revert to the snapshot for every execution [14]. In this way, we were assured that the virtual machine was rehabilitated every time. After the execution, we obtained a log file which reflected the behaviors of the malware in terms of API function calls. Then, we extract the dynamic features.

4.4. Classification

In this process Clustering of Database is done through K-Means Clustering and Naive Bayes Classifier prediction technique is applied to

determine the probability of the binary is as APT Malware or benign software. K-Means Clustering is applied on dynamic features and Validity is checked if still we are not able to determine the Validity of binary then Naive Bayes Classifier is applied onto static as well as dynamic features of binary and probability is evaluated based on training model.

Following are the steps that are followed during the execution of the system:

Step 1: Given the binary X.
Step 2: Extract the dynamic features from X.
Step 3: Apply K-Means Clustering on dataset of X and predict the cluster in which the X is nearer to centroid (-1, 0, +1). // *-1: benign software, 0: Suspicious, +1: APT malware*
Step 4: If output is -1 or +1, predict the result. Else if output is 0 then go to step 5.
Step 5: Perform static analysis and extract the static features and enter into X.
Step 6: Classify X using Naive Bayes Classifier and predict the output -1 or +1.

4.4.1. K-Means Clustering

K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). The main idea is to define k centers, one for each cluster. Thus we can create the database with application of the clustering [10]. The database can be divided into three clusters on the characteristics:

1. APT Malware: This cluster contains the feature sets having higher values of features. These features show the properties of an APT Malware.
2. Suspicious: This cluster contains the feature sets that are some feature shows binary is benign and some feature shows the binary is APT.
3. Benign: This cluster contains the feature sets whose values are relatively very less. These features indicate properties of the benign software.

Now that we know how the database is to be clustered in horizontal partitions, we employ K-Means Clustering algorithm. K-Means Clustering denotes the clustering of the database of n feature sets using k partitions, for our clustering k=3. For the initial clustering there is need of providing initial values for clusters [3].
For the given set of feature sets (fs_1, fs_2, fs_3 … fs_n), where each feature set is of *n* dimensional vectors, K-Means clustering with the help of following formula:

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{xj \in Si} \|x_j - \mu_i\|^2 \quad (1)$$

We measure the distance between each centroid and feature set after every iteration and updating of centroid values.

4.4.2. Naive Bayes Classifier

Bayesian classifiers find the distribution of attribute values for each class in the training data. To find the probability $p(C_j|d)$ of the instance d being in class C_j, Bayesian classifiers use Bayes theorem which says:

$$¿ \frac{p(¿ C_j}{p(d|C_j)p(C_j)} ¿ \quad (2)$$

In other words the formula for Naive Bayes can be given as follows:

$$V_{nb} = \text{argmax}_{fsi \in V} P(fs_i) \prod P(fs_{i\,j})$$

(3)

We are using the Naive Bayes Classifier which estimate p(fs|C) using m-estimates as follows:

$$p(fs_{i\,j}) = \frac{n_c + mp}{n + m}$$

(4)

n = no. of training examples for which $fs_i = fs$.
m = arbitrary value, equivalent sample size
n_c = no. of examples for which $f_s = fs$ and C = C_j.
$p = 1/$ no. of values attribute can have (2)

The advantages of using the NB classifier over the decision tree classifiers is they can classify the unknown or null attribute values by omitting from probability computation. Hence the results will be more accurate than that of the decision tree classifiers as they cannot handle null or unknown attributes meaningfully.

5. Estimated Results

Following example will illustrate how the system will predict the results by fusion of the clustering and NB classifier.

Following is the illustration for the K-Means Clustering for our model:

BIN	D1	D2	D3	D4	CLUSTER
A	0	2	2	1	**1**
B	0	1	1	2	**1**
C	0	2	3	4	**1**
D	0	5	4	9	**2**
E	1	3	5	10	**2**
F	1	4	7	20	**2**
G	1	5	9	4	**2**
H	1	8	13	15	**3**
I	1	9	9	16	**3**

INITIAL CLUSTERS CENTROID				
C1	1	1	1	5
C2	1	5	5	10
C3	2	10	10	20

Following is the illustration of NB Classifier:

TRAINING DATA SET FOR NAÏVE BAYES								
BIN	**D1**	**D2**	**D3**	**D4**	**S1**	**S2**	**S3**	**CLUSTER**
A	0	2	2	1	0	1	0	**1**
B	0	1	1	2	0	2	1	**1**
C	0	2	3	4	0	1	0	**1**
D	0	5	4	9	1	5	2	**1**
E	1	3	5	10	1	7	0	**3**
F	1	4	7	20	0	1	5	**1**
G	1	5	9	4	1	7	4	**3**
H	1	8	13	15	1	5	7	**3**
I	1	9	9	16	1	9	8	**3**
J	1	5	9	10	1	7	2	?

CALCULATION FOR (CLUSTER=3)					
	N	**NC**	**M**	**P**	**PROB**
D1 = 1	5	4	7	0.5	0.625
D2 = 5	2	1	7	0.5	0.500
D3 = 9	2	2	7	0.5	0.611
D4 = 10	1	1	7	0.5	0.563
S1 = 1	5	4	7	0.5	0.625
S2 = 7	2	2	7	0.5	0.611
S3 = 2	1	0	7	0.5	0.438
					0.017950

CALCULATION FOR (CLUSTER=1)					
	N	**NC**	**M**	**P**	**PROB**
D1 = 1	5	1	7	0.5	0.375
D2 = 5	2	1	7	0.5	0.500
D3 = 9	2	0	7	0.5	0.389
D4 = 10	1	0	7	0.5	0.438
S1 = 1	5	1	7	0.5	0.375

S2 = 7	2	0	7	0.5	0.389
S3 = 2	1	1	7	0.5	0.563
					0.002617

The probability of feature set belonging to Cluster 3 is more than Cluster 1 (0.017950>0.002617) hence this feature set is classified as APT Malware.

6. Contributions of the Proposed System

Strictly speaking, malware samples are now able to mutate their syntactical representation after each infection, thus making signature-based strategies completely ineffective. While advances can be made in research on how malware code or behavior can be more accurately represented by signatures or models, there are opportunities for improving how accurately information is extracted from malware to generate signatures and how the signatures are utilized by programs to detect malware. By improving the effectiveness of these aspects, the overall effectiveness of malware detection can be improved in general. Our research intends to explore such opportunities by integrating the static and dynamic malware analysis to promote their effectiveness. In addition, the hybrid usage of supervised learning and unsupervised learning can lead to not only high efficiency but also more reliability.

7. Conclusions

In this paper, we evaluated two malware detection system mechanisms out of which one is dependent on dynamic features and second is based on combination of dynamic and static features. We have created a trail of combination of these two systems and using base techniques given by them. Application of clustering on this system efficiently generates the output but by compromising with the accuracy of results. Although Bayesian approach requires analyzing the training data set provided and takes a very long time of execution, it generates more accurate results. A mechanism resulted by the combination of these two algorithms is more efficient and reliable than these two separate techniques. To generate output at higher throughput, our mechanism uses K-Means Clustering which with lack of efficiency and this lack of efficiency is recovered with the Naive Bayes Classifier. Moreover, the weakness of dynamic malware analysis features can be fulfilled by static analysis features.

8. Future Work

In this article, we have described how our classification step work and others are in general, as our system is under development. There are some practical and theoretical issues that need to be addressed, however. On the practical side, the features we used are presented by the assumptions and we have to analyze these features and describe the features in detail for static and dynamic analysis. On the theoretical side, the analysis of the accuracy of the present approach is rather informal. Much remains to be done in this regard, especially when comparing to the accuracy of existing models. For this comparison, additional work will be required. So, we intend to publish the articles to show the rest of our steps; malware analysis, feature extraction, feature selection and evaluating the detection accuracy.

References

[1] A. Lakhotia, A. Walenstein, C. Miles, A. Singh, “VILO: a rapid learning nearest neighbor classifier for malware triage”, Journal of Computer Virology and Hacking Techniques, 2013.

[2] A. Mohaisen, O. Alrawi, M. Larson, “AMAL: Highfidelity, behavior-based automated malware analysis and classification”, 2013.
[3] A. Sillberschatz, H. Korth, and S. Sudarshan, “Database System Concepts”, 5th Edition, pp. 895-903.
[4] B. M. Eshete, "Effective Analysis, Characterization, and Detection of Malicious Activities on the Web", University of Trento, December 2013
[5] E. R. Weippl, "Malware Analysis, Software Obfuscation and the Ethics of Large Scale Malware Evaluation", SBA Research & Vienna University of Technology, 2012
[6] K. A. Roundy, "Hybrid Ananlysis and Control of malicious Code", University of Wisconsin-Madison,2012
[7] K. M. Abdelrahman Y Alzarooni, "Malware Variant Detection",University College London, March 2012
[8] K. Mathur and S. Hiranwal, "A Survey on Techniques in Detection and Analyzing Malware Executables", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013
[9] K. Raman, “Selecting features to classify malware”, In InfoSec Southwest, 2012.
[10] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Analysis of Malware Behavior using Machine Learning", Journal of Computer Security, IOS Press, http://www.iospress.nl, 2011
[11] M. I. Sharif, "Robust and Efficient Malware Analysis and Host-based Monitoring", Georgia Institute of Technology, December 2010
[12] N. Singh, S. S. Khurmi, "Malware Analysis, Clustering and Classification: A Literature Review", IJCST Vol. 6, Issue 1 Spl- 1 Jan-March 2015
[13] P. Chen, L. Desmet, C. Huygens, “A study on Advanced Persistent Threats”, 2014
[14] R. Canzanese, M. Kam, S. Mancoridis, “Toward an automatic, online behavioral malware classification system,” in IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems (SASO), 2013
[15] R. Islam, R. Tian, S. Versteeg, “Classification of malware based on integrated static and dynamic features”, Journal of Network andComputerApplications36, 2013
[16] S. Cesare, Y. Xiang, W. Zhou, “Malwise – An Effective and Efficient Classification System for Packed and Polymorphic Malware”, IEEE Transactions on Computers, Vol. 62, 2013
[17] U. Bayer, I. Habibi, D. Balzarotti, E. Kirda, and C. Kruegel, "A View on Current Malware Behaviors", 2008