POSTER: Blind Separation of Benign and Malicious Events to Enable Accurate Malware Family Classification

Hesham Mekky[1] Aziz Mohaisen[2] Zhi-Li Zhang[1]
[1]University of Minnesota [2]VeriSign Labs
{hesham, zhzhang}@cs.umn.edu amohaisen@verisign.com

ABSTRACT

Malware families classification has been studied extensively in the literature. Machine learning based identification techniques rely on building a classification model for the malware traffic, and then the model is used for labeling unseen observations. In practice, malware traffic (malware signal) is mixed with other legitimate traffic (background signal). Consequently, the classifier's effectiveness may be hindered, since the observed traffic is mixed. We propose to apply signal decomposition in order to decompose the observed traffic into two components, malware traffic and background traffic, and then classification techniques are applied effectively on the malware traffic after removing the background attributes. Our preliminary results show the effectiveness of the proposed approach.

Categories and Subject Descriptors

C.2 [**Computer Communication Networks**]: General—*data communications, security and protection*

General Terms

Measurement, Security

Keywords

ICA; Malware; Classification; Background noise elimination

1. INTRODUCTION

Malware analysis, classification and labeling is a well-investigated problem in the research community and industry. Techniques used for malware analysis fall into two categories: static and dynamic. Static techniques for malware analysis utilizes meta-data associated with binaries, including certain patterns in the binary itself, whereas dynamic techniques utilize artifacts generated by the malware at the run time, including memory, network, file system and registry usage. While static analysis is fast and scalable, it requires costly reverse engineering efforts for obtaining patterns and roles from binaries, and are defendable by obfuscation. On the other hand, the dynamic analysis techniques are highly accurate, capable of detecting previously unseen malicious behavior, and address code obfuscation, although cost more time to run [7]. In addition, one circumvention mechanism utilized by malware authors is "behavior-poisoning", in which random noise is generated as part of the execution of malware to disguise its real behavior.

In this paper, we focus on malware analysis based on artifacts collected from network traces i.e., we do not perform binary analysis. For example, hosts infected with malware produce network artifacts that contain both malware-related and background-related network traffic components. Thus, approaches that rely on machine learning classification [5, 9] can be hindered by this background traffic since the features extracted from the network traffic will contain both malware and background traffic attributes.

Our proposed approach aims at cleaning those features from the background traffic attributes, which significantly improves the effectiveness of machine learning classification in detecting the malware traffic and assigning the appropriate malware family label. Towards this goal, we use Independent Component Analysis (ICA) [6] to remove the background traffic attributes from the mixed traffic. Then, we use classification algorithms to predict the malware label. Our preliminary results show a significant success in isolating background traffic, classifying malware samples to their proper family, and promise further related applications.

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.
ACM 978-1-4503-2957-6/14/11.
http://dx.doi.org/10.1145/2660267.2662365.

2. PROPOSED APPROACH

The main technique used for separating malware traffic from the background traffic is the ICA, which we review in §2.1, followed by an overview of our system for malware classification in §2.2.

2.1 ICA Primer

The ICA is a method for decomposing a multivariate signal into additive components assuming statistical independence. Assuming m independent source signals $S = [S_1, ..., S_m]^T$. We observe the mixture $X = [X_1, ..., X_m]^T$ given by $X = A \cdot S$, where A is named the mixing matrix. The goal of ICA is to find an unmixing matrix $W(\approx A^{-1})$ such that $Y = W \cdot X \approx S$ will be the best approximation for S.

ICA algorithms rely on independence to recover the original signals from the mixture. For instance, given two signals X and Y: (i) Entropy $H(X)$ is a measure of uncertainty in X i.e. the lower the value the more information we have about X, (ii) Conditional entropy $H(X|Y)$ is the amount of uncertainty in X after the observation of Y, (iii) Mutual information, $I(X;Y) = H(X) - H(X|Y)$, is the reduction of uncertainty in X after the observation of Y.

Therefore by having an algorithm that minimizes the mutual information between the estimated components [1], we are looking for latent variables that are maximally independent i.e., hidden distributions for components that are independent.

2.2 System Overview

Figure 1 shows the high level design of our proposed method. As show in the figure, for each malware family, we have an ICA decomposer and a classifier. Each ICA decomposer, recovers the malware signal from the mixed traffic, then we apply that signal to the classifier to obtain a label and score. Finally, we label the given sample based on scores given by all classifiers. While we demonstrate the system for two families, it is clear that the system can easily scale to a large number of families.

We leverage ICA to remove the background traffic from the features extracted from the network traffic. Then, we apply classification on the resulting signal to predict the malware family label. We assume that the behavior of our features is different and independent from the background traffic, which we validate to be correct. For instance, Figure 2 illustrates an example of one of the features used in our analysis, detailed feature extraction is discussed in §3.2. As shown in the figure, the distribution for that specific feature in the family "Darkness" is completely different from the background "Noise" traffic. Consequently, the distribution of the same exact feature in the mixed traffic will be different from the malware feature distribution, which limits the effectiveness of classifiers. Our method uses ICA to recover the malware distribution for all extracted features, and then uses classification to predict the label.

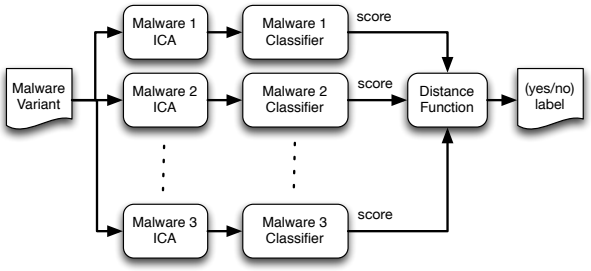

Figure 1: Malware Labeling Process

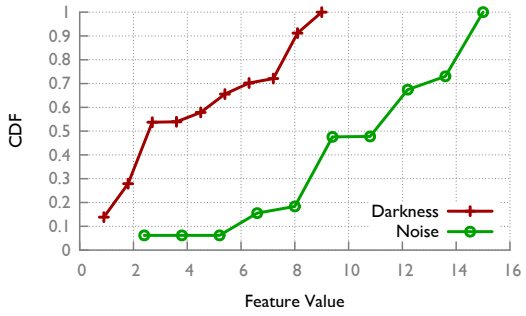

Figure 2: Sample Feature: Malware vs. Background

3. PRELIMINARY RESULTS

In order to understand the power of the proposed system in eliminating background traffic, we experimentally test it on two datasets. In §3.1 we outline the datasets and their context, followed by the features extraction in §3.2, and the results in §3.3.

3.1 Datasets

Our datasets are composed of two malware families: (a) Darkness malware family, which infects machines to carry out DDoS attacks, and (b) Shady RAT (SRAT) malware family, which infects machines to target high profile organizations (credentials stealing, DDoS attacks, etc). These families cover a wide range of network behavior for our evaluation purposes. Each malware sample is labeled manually using an operational product at VeriSign [7].

Each sample is executed in a controlled environment for a predefined amount of time to collect network artifacts generated solely by each malware family in the form of PCAP traces. Details of each dataset is shown in Table 1. In addition to the malware families, we have another dataset that resembles the background traffic behavior generated from regular hosts such as web browsing.

Table 1: Datasets

Family	Samples
Darkness	534
Shady RAT (SRAT)	1096

We use the PCAP trace generated by each sample to create a profile for each malware sample based on the ordering of events in the packet trace [8]. Profiles are based on network events seen in the packets e.g., an outbound packet using UDP on port 53 is mapped to "A0A2A5", where A0 refers to the outbound occurrence, A2 refers to the UDP protocol, and A5 refers to the port number 53. Consequently the "word" A0A2A5 in a malware sample profile indicates an outbound DNS query over UDP. We apply the same mapping to all packets in the trace. Table 2 summarizes all network events we consider in our analysis.

3.2 Features

Based on the discussion in §3.1, each malware or background traffic sample is represented as a sequence of words, where each word indicates the occurrence of a particular event. Consequently, the words in a malware sample correspond to the two-way communication of the malware with the remote C&C server. We use n-gram analysis on that final representation to capture relevant network events in both malware and background traffic. We use the generated n-grams as our feature set, where n ranges from 1 to 5.

We perform the n-grams analysis for constructing the feature vectors for each malware family, and the background traffic. Then, we select the top discriminating features for the malware family with the background traffic using recursive feature elimination [3]. For instance, malware family 1 (MF1) along with the background traffic are used to select the most discriminant features between them. After the end of this process, we have the top features that can differentiate between the malware family and the background traffic without being mixed.

Next, we build the ICA model for decomposing the malware from the background traffic. For each malware family, we mix the selected top n-gram features for the malware and the background traffic linearly. For instance, feature 1 distribution in MF1 is mixed with its corresponding distribution in the background traffic. Then,

Table 2: Network Events

Network Event	Description
Connection	TCP, UDP, RAW
Size	request quartiles, response quartiles
DNS	A, NS, MX, CNAME, SOA, PTR
Request type	GET, POST, HEAD
Response type	200's, ..., 500's
Ports	80, 8080, 53, and others

the generated distribution is fed as input to ICA to build a model for MF1. The generated model in used in the first stage in Figure 1.

Finally, the two output components from the ICA model, malware component and background component, are used to train the classifier, which is the second stage in Figure 1. Consequently, we have an ICA model and a classifier model for each malware family in the system. The ICA model removes the background traffic attributes from the features, and then the classifier is used to label the corresponding malware sample.

3.3 Classification Results

In the following, we demonstrate the effectiveness of our system in separating mixed signals and classifying malware samples based on the cleaned features.

Figure 3 shows the effectiveness of ICA in recovering the distribution a feature from the mixed traffic, where "Original" is the distribution of a malware n-gram feature without being mixed, "Mixed" is the distribution of the feature that we see in the mixed traffic, and "ICA" is the distribution of the feature recovered by ICA from the mixed distribution. The ICA distribution closely follows the same trend of the original malware feature.

We use FastICA [4] for the ICA stage, and Random Forests [2] for the classification stage. We use 10-fold cross validation to evaluate the ICA and the classifier. Each dataset is split into 10-folds, where 9-folds are used for training the ICA and the classifier model, and the 10th fold is used for testing. For the testing fold, we use the mixed feature distributions to evaluate the effectiveness of the ICA model. The classification results are shown in Table 3. We use standard metrics, including the accuracy, precision, recall, and F1 score [8], to evaluate the classifier as illustrated by Table 3. From the table, it is clear that ICA can achieve high accuracy in labeling the malware families, even outperforming the related literature utilizing the same dataset [8].

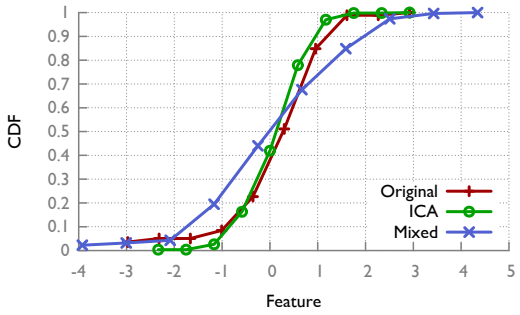

Figure 3: ICA Recovered Feature

Table 3: Results

Malware	Accuracy	Precision	Recall	F1
Darkness	0.943	0.971	0.932	0.924
SRAT	0.981	0.983	0.981	0.976

4. CHALLENGES & LIMITATIONS

Mixing Challenges Our system mixes the plain malware feature distributions with their corresponding features in the background traffic to train the ICA model to split them. In practice, some n-grams in the mixed traffic may be missing in the plain malware or background traffic. Therefore, we are considering other techniques to capture those missing n-grams. For instance, we can use a skipping factor λ to skip over words in the profiles e.g., considering *"w1 w2 w3 w4"* as a sentence with $\lambda = 1$, we generate the following bi-grams *{(w1 w2), (w1 w3), (w2 w3), (w2 w4), (w3 w4)}*.

Feature Interpretation We are exploring the top discriminant features in order to gain insight into the main differences in behavior between the malware traffic and the background traffic. The n-gram analysis makes it a little bit challenging to interpret the features into meaningful behavior, however, we are analyzing them to learn the different malware patterns that enable ICA algorithms to decompose the mixed traffic.

ICA Limitations The two assumptions that ICA relies on are independence and non-normality for the malware traffic. We assume that the background traffic such as web browsing, OS updates, or peer-to-peer traffic are independent from the malware traffic running on the same machine. We used normal distribution tests to confirm that malware feature distributions are not normal. Consequently, ICA can only work on malware families that do not exhibit any Gaussianity in their features. Our work cannot be applied to families that have similar properties of the background traffic (e.g., both are following normal distribution or dependent). Exploring the extent to which our system is affected by violation of those properties is left as a future work.

5. CONCLUSION

In this paper, we proposed using ICA to decompose a machine network traffic into malware and background traffic to improve the performance of classification based methods. Our preliminary results improved over previous work in the literature, which shows the effectiveness of ICA in decomposing the traffic. The system that we have introduced in this paper is generic in purpose, and it can meet needs in multiple applications. For example, traffic analysis and identification is a generic problem related to services profiling, and our system can perhaps used for that.

6. ACKNOWLEDGMENTS

We would like to thank Allison Mankin for her involvement in earlier stages of this work. The work of Hesham Mekky was mainly done while at VeriSign. This research was supported in part by US NSF grants CNS-10171647, CNS-1017092, CNS-1117536 and CRI-1305237.

7. REFERENCES

[1] A. J. Bell and T. J. Sejnowski. An Information-Maximization Approach to Blind Separation and Blind Deconvolution. *Neural computation*, 1995.

[2] L. Breiman. Random Forests. *Machine learning*, 2001.

[3] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine learning*, 2002.

[4] A. Hyvarinen. Fast and Robust Fixed-point Algorithms for Independent Component Analysis. *Neural Networks*, 1999.

[5] D. Kong and G. Yan. Discriminant Malware Distance Learning on Structural Information for Automated Malware Classification. In *ACM KDD*, 2013.

[6] T.-W. Lee. *Independent Component Analysis*. Springer, 1998.

[7] A. Mohaisen and O. Alrawi. Av-meter: An evaluation of antivirus scans and labels. In *DIMVA*, 2014.

[8] A. Mohaisen, A. G. West, A. Mankin, and O. Alrawi. Chatter: Exploring classification of malware based on the order of events. In *CNS*, 2014.

[9] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and Classification of Malware Behavior. In *DIMVA*. 2008.