Improved Malware Clustering Using VirusTotal Meta-data

Carl Sabottke – Eddie Tanner – Richard Johnson

ABSTRACT

More than 1 million new malware samples are generated each day using advanced packing and obfuscation techniques, making it difficult for anti-virus products to detect new malware variants. Attempts to identify these malware have generally required large datasets not available to the public. We show that it is possible to automate the clustering of the Malicia dataset into malware families using the dynamic execution metadata from VirusTotal to train a machine learning classifier. In the process we are able improve on the original analysis by clustering approximately 400 previously unlabeled samples in the dataset, while also finding that previous measures of precision and recall on the Malicia dataset may be biased due to the dominance of 3 malware families. Our results can be generalized into a publically available methodology to cluster malware into families.

1. INTRODUCTION

Malware has always stayed one step ahead of the anti-virus industry largely due to automatic systems that pack and obfuscate malware. In fact Symantec estimates that a shocking 403 million new malware samples were created in 2011, a number that increases every year. This hides the fact that there in reality there is a much smaller number of malware strains being deployed at any one time. Malware authors automatically modify their samples with the hopes of delivering a unique binary to each host.

In this study our goal is to group malware into families using publically available meta-data. This is made difficult because of the sheer volume of malware samples and the rapid evolution of malware samples. Manual analysis through means such as reverse engineering are too labor-intensive to implement on such a large scale. Instead most of the malware classification must be done autonomously. This had been the focus of many research efforts on malware clustering [2][3][4], anti-virus signatures [2], and reputation-based security [5][6]. Likewise it is supported by sites that analyze submissions of unknown binary files.

While other studies have used VirusTotal anti-virus labels as a reference dataset, they later go on to gather features from alternate sources to perform the actual clustering [3][4]. VirusTotal has traditionally provided only anti-virus labels, but these recent additions make it more attractive for use in clustering. Their API provides a wealth of features which may uniquely identify malware families including code size, imports, networking behavior, and filesystem changes to name a few. While it may be possible for malware authors to hide code size, it would be much harder to hide features such as imports and connections behavior. We believe this makes VirusTotal submissions a powerful tool for identifying malware families.

In our study we first submit the Malicia dataset to retrieve the resulting VirusTotal meta-data. We then identify features that appear promising for clustering including code size, imports, and DNS requests. Unfortunately due the smaller scope of this study, we were only able to choose a few features for analysis. However further studies could improve by adding additional features. We also determine which of our chosen features performed best in clustering the Malicia dataset using SVMs. Finally we perform both K-mean and agglomerative clustering on the dataset and evaluate their performance.

We make the following contributions:

- Demonstrate an effective and simple methodology to automate the clustering of malware into families, by using VirusTotal dynamic execution metadata in a machine learning classifier.
- Expand upon the Malicia clustering analysis, finding approximately 400 unlabeled malware binaries that we conjecture belong to the ZeroAccess family. These binaries make a DNS request for j.maxmind.com which appears to uniquely identify this family.
- Use support vector machines (SVMs) to identify features useful for clustering.
- Reveal potential biases in the Malicia dataset, namely that 94% of the malware binaries belong to 3 families. One consequence is that previous measures of precision and recall are dominated by these 3 families and may not generalize to all malware families.

2. PROBLEM STATEMENT & GOALS

The following is the complete statement of our goals and resources used to achieve them.

Resources:

- Malicia dataset consisting of 11,362 malware binaries and ground truth labels. 1458 binaries do not contain family labels which can be treated either as noise or excluded in the analysis.
- Private key to the VirusTotal API. This service provides antivirus results and dynamic execution meta-data for each submitted file. Includes features such as AV labels, imports, exports, file size information, entry point, DNS requests, network requests, registry changes, and more.

Goals:

- Automate the clustering of malware into families using VirusTotal metadata in conjunction with machine learning algorithms.
- Assess the quality VirusTotal features for clustering purposes.
- Assess the validity of ground truth in Malicia dataset.
- Gain insights into differences between malware operations.

3. RELATED WORK

Our research follows upon the work of the original Malicia paper: *Driving in the Cloud: An*

Analysis of Drive-by Download Operations and Abuse Reporting [1]. In it the authors describe the creation of the Malicia dataset starting with the milking of exploit servers to gather malicious binaries. These binaries were then classified into generic labels by applying automatic clustering techniques to the malware icons, screenshots, and network traffic, which were then manually refined by an analyst. We use these classifications as the primary source ground truth in our study to group the samples into families. By contrast, the Malicia paper focused on clustering the exploit servers which distributed the malware. Our work with VirusTotal could be applied in a follow-up study to gain additional insight into to the differences between malware operations.

A subsequent study by the same authors [2] introduces FIRMA, a malware clustering technique that uses automated network signatures to achieve perfect precision and 97.7% recall on the Malicia dataset. We found this study useful for comparison of clustering quality, but we also question whether their clustering quality metrics are misleading. Since 93% of the Malicia dataset is comprised of 3 families, their metrics may essentially be evaluating the effectiveness of FIRMA against those 3 families.

Much previous work in automated clustering has made use of dynamic execution environments because it is an effective way to get around packing and obfuscation techniques [2][3][4]. For example, [4] extends the ANUBIS dynamic execution platform with taint tracking to build an accurate behavioral profile of each sample to be used in the clustering. We similarly make use of VirusTotal's dynamic execution metadata, but we also make use of AV-label information that is exclusive to VirusTotal. It is valuable to explore VirusTotal to gather clustering metadata due to its public availability, ease of use, and the recent introduction of dynamic execution features.

4. CLUSTERING & CLASSIFICATION PROCESS

We first collected over 11,000 malware samples from the Malicia project, and then we submitted these binaries to VirusTotal to obtain three types of data: antivirus (AV) labels, static binary analysis information, and dynamic analysis information. For the purposes of exploring clustering and classifiers, we separately considered the AV labels and the dynamic analysis information as feature groups. The main focus of our analysis was on exploring the potential of the dynamic analysis information, since this data can be obtained as soon as a new malware sample is obtained. Using AV labels as a feature, though, requires a time delay in the analysis pipeline, since the malware would have to already have been analyzed by one or more AV vendors.

4.1 Antivirus Label Clustering

The Malicia project data has malware family labels on 9904 of the 11362 samples that we analyzed. We used these labels as "ground truth" for our clustering comparisons. However, we also could potentially have used AV labels provided by VirusTotal as ground truth as was done by [3]. Furthermore, exploring the potential usefulness of this technique is one of the main reasons why we performed clustering in the AV label space.

Figure 1 panel A shows the malware samples projected into the first three principal components

(PCs) of the AV label space. We did not use principal component analysis for dimensionality reduction related to clustering. However, we did use principal components as a way of projecting the data into 3-dimensional spaces for visualization purposes. Instead of reducing the dimensionality of the AV label space via principal components, we constructed it as a 47 dimensional space, where each AV vendors labels were given their own feature and dimension of this space. The unique labels provided by each AV vendor were linearly mapped into one dimension with an alphabetical ordering. These dimensions were then normalized by dividing by the number of unique AV labels per vendor. This space has a non-optimal geometry because the notion of distance in this space is divorced from the ideal geometry which would optimally associate distance with semantic similarity. Nevertheless, treating each feature as a boolean variable would also be non-optimal. We can illustrate this problem with a fictitious example where there are four labels: FakeAVType1, FakeAVType2, FakeAVType3, and KeyloggerType1. In the mapping we used, prior to normalization these labels would have numerical values 1, 2, 3, and 4. Therefore, FakeAVType1 would be more dissimilar to FakeAVType3 than KeyloggerType1 and FakeAVType2. This is obviously non-optimal since with a semantically sensitive distance metric we would want FakeAVType1 and FakeAVType2 to be equidistant from FakeAVType3. We would then want KeyloggerType1 to be much farther away. Using boolean features for each unique label also does not solve this problem since then FakeAVType1, FakeAVType2 and KeyloggerType1 would be equidistant from FakeAVType3. One possibility for solving this issue is to use a string similarity metric like the Levenshtein distance to scale the distances in a one-dimensional space. However, we did not explore this possibility. The main focus of our AV label space clustering was simply to assess the similarity of these AV labels and to obtain a general estimate of how useful they might be to use as a substitution for the Malicia family labels.

The Malicia labels identify 53 distinct malware families and 54 if the null labels are counted as a separate family. Deciding how to handle these null samples is a much more relevant aspect of our analysis than the specific geometry of the AV label space, even though this geometry issue is generally very import in regards to clustering analysis. Figure 1 panel B shows the results of k-means clustering in the AV label space. As a comparison metric, we have plotted the Dunn index which we calculated by finding the minimum of the set of values that consists of all the intercluster distances divided by the maximum intracluster distance value. The Dunn index is a useful clustering metric because it is an unsupervised metric for assessing cluster quality which does not require any cluster labels in order to have meaning. A standard way of determining the number of clusters to use in a k-means analysis when the number of clusters is not known a priori is to find the value of k, the number of clusters, that maximizes the Dunn index. Clearly, a k of 53 or 54 does not maximize the Dunn index for this dataset. However,the purity achieved for 54 clusters is still 89%. Figure 1 panels C and D show the 3-D PC space representations of the AV label space that have been colorized based on the Malicia labels and based on the k-means clustering. Initially, the visual distinction between these two projections might seem discordant with a purity level of 89% which is quite high for this dataset when such a relatively low number of clusters is considered. However, this is because multiple clusters in the k-means clustering are assigned to be representative of the same class when calculating the purity value.

4.2 Purity and Normalized Mutual Information as Metrics of Clustering Quality

Purity is a quantity which can only be calculated when class labels for data are known or can be estimated from another source. Purity assigns a class label to each cluster by simple majority vote process and is then computed by counting the number of correct assignments for each cluster and dividing by the total number of samples. Purity is a very simple measure to compute, but it is far from optimal, since it does not penalizing clusterings that have an exceedingly high number of clusterings. Therefore, a "clustering" which assigns each data point to a separate cluster always has a purity of 100% [7].

In order to deal with the issues inherent in the purity measure, we also computed normalized mutual information. Normalized mutual information does tradeoff cluster similarity with number of clusters, so it is a much better metric to use than purity when generally assessing cluster quality. Normalized mutual information is calculated by first computing the mutual information between the clusters and the class labels and then dividing by the average entropy of the classes and the clusters [7].

4.3 Support Vector Machines for Feature Space Analysis

Our main focus for this project was on using non-AV label malware features as a means of clustering the malware into families. For these clusterings, we computed both purity and normalized mutual information. However, prior to clustering the malware, we used support vector machines (SVMs) as a means of determining which features might have utility in relation to malware clustering and classification. Support vector machines are used to solve 2 class problems when labeled data is available. We used the Malicia family labels as class labels for the training and testing of support vector machines. We considered training and testing scenarios which both included and excluded the null labeled data. However, SVMs are formulated for two class problems and malware family identification is inevitably a multi-class problem. Therefore, we chose to implement one vs. all SVMs in which each class is trained against all others in order to formulate the problem as a two class scenario. Tables 1-4 show the results of our SVM analysis when considering imports, exports, DNS request, and size features for training and testing the SVMs. For each SVM, we calculated precision and recall values. Precision is the number of true positives divided by the sum of the numbers of true positives and false positives. Recall is the number of true positives divided by the sum of the numbers of true positives and false negatives.

Nevertheless, when inspecting the results in tables 1-4 it is important to understand the composition of the Malicia dataset. Achieving high recall and precision on the Winwebsec family is not necessarily impressive, because 59% of the labeled data belongs to this family. 59% precision and 100% recall on the entire dataset can be achieved by simply labeling all samples as Winwebsec. 22% of the labeled data belongs to the zbot family and 13% belongs to ZeroAccess. 94% of the labeled malware belongs to only three families. Furthermore, 23 of the 53 families only have a single sample in

the dataset. The fourth most prominent family, SecurityShield, only has 150 samples labeled in the 9904 samples. Therefore, we only constructed one vs. all SVMs for these four most prominent families. If we were using these SVMs for true classification, then we would need to construct a decision rule for deciding which class a sample belongs to when multiple SVM classifiers claim the same sample. One way to do this would be based on prior probability (prevalence in the training data). However, we are not using these SVMs to construct a true classification scheme, so we only looked at each SVM individually for each family as a means of assessing feature utility.

We considered four feature classes: imports, exports, DNS requests, and size information. We treated each unique import, export, and DNS request as a separate boolean feature. For size information, we considered code size, file size, and initialized data size each as separate features. In our dataset we found 244 unique imports, 1854 unique exports, and 1304 unique DNS requests. We initially suspected that it might be necessary to perform dimensionality reduction rather than use 3402 boolean features for SVM training, testing and clustering. However, this was unnecessary and we were able to train SVMs and perform clustering in a reasonable amount of time despite the large number of features.

There are several interesting results to highlight in tables 1-4. Namely, imports do well for classifying Winwebsec, but not the other families. Exports performs rather poorly for most families, but it performs excellently for the ZeroAccess family. DNS request also are not highly effective except for the ZeroAccess family. In fact, this family shows a marked performance improvement after the null labels are excluded. We suspect that this is because about 400 of the null label malware samples actually belong to the ZeroAccess family. These null samples all make a DNS request to j.maxmind.com, which is only requested by this subset of null samples and a subset of the ZeroAccess family. However, the low recall for this family also leads to an important caveat related to using DNS requests as a feature. Observation of this feature class could potentially be quite variable. Ideally, we would have requested for all of the malware samples to be run by VirusTotal at multiple points in time to help partially control for some of this variability.

Surprisingly the size features proved to be a very useful feature for clustering this dataset. However, the SVMs show that this feature is mostly only highly successful at classifying the Winwebsec family. This highlights how important it is to remember that this dataset is not necessarily representative of most malware and that looking only at performance on the dataset as a whole does not necessarily give a good indication of how particular features will generally be successful in relation to malware classification and clustering.

4.4 Agglomerative Clustering based on Dynamic Analysis Features

Rather than using k-means clustering,we focused on agglomerative linkage-based clustering for the bulk of our dynamic feature clustering analysis. We chose this approach since we had no prior assumptions about the number of clusters for this problem. We also chose to use linkage-based clustering, because the malware similarity information contained in a linkage tree or dendrogram is

essentially what we are most interested in.

When forming linkages, we used a Euclidean distance metric and for multi-point clusters we computed the distance between clusters as the distance between the farthest apart datapoints for most cases. All figures shown used these criteria. However, we also performed clusterings in which we used the shortest distance between cluster datapoints for linkage formation.

Figure 2 shows the results when only using the import data for clustering. Panel A shows purity as a function of the linkage distance used to separate clusters. Purity decreases as this distance increases, because increasing this distance results in fewer clusters which contain more dissimilar samples. Panel B shows cluster purity as a function of the number of clusters. Panel C shows the normalized mutual information as a function of the number of clusters, and panel D shows the import data projected into a 3-dimensional principal component space with colorization based on the 200 cluster results. The peak for normalized mutual information occurs around 200 clusters, and this value almost reaches 0.4.

Figure 3 shows the results for clustering based only on export data. Panel A shows purity as function of cutoff distance. Panel B shows purity as a function of number of clusters. Panel C shows normalized mutual information as a function of cluster number, and panel D shows a projection of the data into the 3D principal components space with colorization based on the clustering results for 200 clusters. About 70% purity is achieved using the export data compared to nearly 90% for the import data. Normalized mutual information also only achieves a value of about 0.3, and the peak occurs for a much higher number of clusters.

Figure 4 shows the clustering results based only on DNS request data. Panel A shows purity as a function of cutoff distance. Panel B shows purity compared to the number of clusters. Panel C shows normalized mutual information, and panel D shows the result of division into 200 clusters that has been projected into a lower dimensional principal components space. DNS requests performs the most poorly of the features considered, and only achieves a normalized mutual information value of 0.2 and a purity value of about 60%.

Figure 5 shows the clustering results for the size information. Out of all the feature groups, this group performs the best. Panel A shows the achievement of a purity value above 95%. Panel B shows a peak at .5 in the normalized mutual information, and interestingly this peak occurs for when there are a relatively small number of clusters (~10). Panel C then shows the data projected into a principal components space where colorization has been done based on the results for 200 clusters.

Figure 6 shows the results of clustering done on all of the boolean features. Surprisingly, all of the boolean features combined do not perform significantly better than the imports alone. Figure 7 then shows the combination of all of the dynamic analysis features which we considered. Normalized mutual information almost achieves a value of 0.6, so we do have an improvement over individual feature groups considered on their own. However, this improvement is not very large considering the increase in feature size. It is also important to note here that after normalizing the size feature data we increased these values by a factor of 200 to weight these features more evenly with the high dimensional boolean features. Yet, this scaling factor is a tunable parameter, which ideally would have been iterated over to

find a value that maximizes the maximum normalized mutual information.

One important caveat in regards to our analysis is that some features, most notably DNS requests, might not be consistently measurable. A more idealized analysis would have submitted each binary to VirusTotal multiple times to compare the retrieved results across time and check for consistent behavior from the binary. Having one time sample per binary does not invalidate our results, but it does leave some questions open about how reliable some dynamic features can be for malware clustering and classification.

5. CONCLUSION

In this paper, we expand on the original Malicia analysis and present a repeatable methodology to cluster malware into families. By gathering VirusTotal meta-data for use in a classifier we achieve a high overall precision and recall. It is notable that our process is able to achieve comparable clustering performance to the original Malicia clustering because it is fully automated and less labor-intensive.

Moreover we evaluate the value of VirusTotal features for clustering, finding that imports and simple file size are surprisingly effective. Adding exports and DNS requests does not greatly improve clustering performance, however SVM's show that for individual families exports and DNS can provide useful classification information.

Finally, we question whether the dominance of 3 malware families in the Malicia dataset biases previous measurements of precision and recall. Winwebsec, zbot, and zeroaccess comprise 94% of all malware binaries. It is then intuitive that any such measure is influenced almost entirely by the specific traits of these three families. For example, we found that code size, file size, and initialized data size are all very useful as features for obtaining clustering performance boosts on this dataset. However, our work with SVMs showed that this feature group was very successful at identifying the Winwebsec family but not at identifying the other families. Therefore, it should not be assumed that this size feature information should necessarily be useful as a feature group in analyzing other malware datasets which would not be dominated by the Winwebsec family.

REFERENCES

1. [DIMVA 2013] Driving in the Cloud:An Analysis of Drive-by Download Operations and Abuse Reporting
PDF Bibtex Publisher

Antonio Nappa, M. Zubair Rafique, Juan Caballero.

Proceedings of the 10th Conference on Detection of Intrusions and Malware & Vulnerability Assessment Berlin DE, July 2013.

2. [RAID 2013] FIRMA: Malware Clustering and Network Signature Generation with Mixed Network Behaviors.
PDF Bibtex Publisher

M. Zubair Rafique and Juan Caballero.

16th International Symposium on Research in Attacks, Intrusions and Defenses, St. Lucia, October, 2013

3. BitShred: Feature Hashing Malware for Scalable Triage and Semantic Analysis
http://users.ece.cmu.edu/~jiyongj/papers/ccs11.pdf

4. Scalable, Behavior-Based Malware Clustering
http://www.cs.ucsb.edu/~chris/research/doc/ndss09_cluster.pdf

5. Polonium: Tera-Scale Graph Mining and Inference for Malware Detection
http://www.umiacs.umd.edu/~tdumitra/courses/ENEE759D/Fall13/papers//Chau11.pdf

6. CAMP: Content-Agnostic Malware Protection
http://www.umiacs.umd.edu/~tdumitra/courses/ENEE759D/Fall13/papers//AbuRajab13.pdf

7. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press. 2008.
http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html