Product Offerings in Malicious Hacker Markets

Ericsson Marin, Ahmad Diab and Paulo Shakarian
Arizona State University
Tempe, Arizona
{ericsson.marin, ahmad.diab, shak}@asu.edu

Abstract—Marketplaces specializing in malicious hacking products - including malware and exploits - have recently become more prominent on the darkweb and deepweb. We scrape 17 such sites and collect information about such products in a unified database schema. Using a combination of manual labeling and unsupervised clustering, we examine a corpus of products in order to understand their various categories and how they become specialized with respect to vendor and marketplace. This initial study presents how we effectively employed unsupervised techniques to this data as well as the types of insights we gained on various categories of malicious hacking products.

I. INTRODUCTION

Websites on the deepweb and darkweb specializing in the sale of malicious hacking products - such as malware platforms, software exploits and botnet rental - have become the venue of choice for online purchase of these items by cyber criminals. In this paper, we leverage unsupervised learning to categorize and study the product offerings of 17 of these online markets. Specifically, we describe how we used manual labeling combined with clustering techniques to identify product categories (Section II), and then we analyze the results both quantitatively and qualitatively (Section III). We identify categories of products that are highly specialized with respect to particular vendors and markets. We also highlight other interesting facets of this ecosystem - for instance, vendors who habitually cross-list products on multiple sites and nearly identical products for sale by multiple vendors.

II. MALICIOUS HACKING PRODUCT CATEGORIZATION

In this section, we describe our malicious hacker product dataset and our use of clustering to identify malicious hacker product categories. We examined 17 malicious hacker marketplaces crawled over a 6 month period. The crawled information was then parsed and stored in a relational database. Relevant tables record the marketplaces themselves, the vendors of the various products and the items/products for sale. Each item is associated with a vendor and a marketplace, allowing for join queries. Some of the more relevant fields for marketplace items include the price, title, description, rating, posting date. In this work, we primarily extract features from the product title/name to generate features.

We note that many items are cross-posted and are nearly identical. We show the distribution of vendors who use the same screen-name across multiple marketplaces in Fig. 1(a). To clean our product data, we identify duplicate (cross-posted) products and report on the size of our dataset in Table I.

In addition, there is a clear absence of a standardized method for vendors to register their products. As a consequence, the great majority of products are unique when compared with simple matching or regular expression technique. It is valid even in the case where a pair of vendors with different screen names post what a human would determine to be the same product. Fig. 1(b) shows the distribution of products and number of vendors sharing each product. The distribution follows a power-law. Note that about 57% of products are unique by simple comparison methods.

Clustering approach. Using product names, we engineer features that represent each product as a vector. A set of pilot experiments suggests that word and character n-grams
would provide more pure clusters compared with other feature engineering methods, such as meta-data or domain-specific keywords. These features were valued using standard term frequency - inverse document frequency (TF-IDF), after the elimination of stopping words and the execution of stemming. We evaluated word n-gram feature vectors of length up to 1 and up to 2 words and many character n-gram features in the ranges from 3 to 7 and from 4 to 7. This gave us 10 different feature vectors in all. To verify which of these strategies could reach the best performance in our dataset, we evaluated the effect of the different types of feature vectors on the accuracy and purity of clusters produced by the K-means algorithm.

To determine the best feature vector, we manually labeled 500 samples using 34 labeled groups (listed in Table III). We used 400 of the samples to determine centroids for each of the 34 groups, and then we evaluated the resulting clustering on the remaining 100 samples. We examined the accuracy of the different approaches when compared to ground truth using the Rand-index method [13]. This method is defined as the number of pairs correctly considered in the same class or correctly considered in different classes divided by \( \binom{n}{2} \), where \( n \) is the number of samples. In addition, we used standard entropy measurements to examine the purity of the clusters. Entropy measures the amount of disorder in a cluster. A zero-value for this metric means the clusters are formed by just one class. The formal definition is as follows:

\[
\text{entropy}(D_i) = - \sum_{i=1}^{k} P_{r_i}(c_j) \log_2 P_{r_i}(c_j),
\]

where \( P_{r_i}(c_j) \) is the proportion of class \( c_j \) data points in cluster \( i \) or \( D_i \). The total entropy (considering all clusters) is:

\[
\text{entropy}_{\text{total}}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times \text{entropy}(D_i)
\]

Table III shows the performance of each TF-IDF vectorization using Rand-index and entropy, when K-means starts with the 34 fixed centroids. For Rand-index, character n-grams in the range from 3 to 4, 3 to 5, and 3 to 6, when K-means used cosine similarity reached a high best performance (0.986). In addition, we also found the best entropy (0.067) when K-means uses the same specification. This way, K-means configuration with character n-grams in the range from 3 to 6 for vectorization, cosine similarity for distance function and the 34 points for the starting centroids was our natural choice to produce the clusters in the entire dataset.

We also examined the performance of our approach using random centroids. As expected, it performs worse than using the centroids derived from products. Additionally, we examined products (from the full dataset) with a cosine similarity of less than 0.1 from the calculated centroids. There were 410 such distinct products (4.51% of the dataset). These were then manually examined and most were found to be irrelevant to our target domain - and we did not consider them further.

III. ANALYST INTERPRETATION OF PRODUCT CLUSTERS

In this section, we examine the results of clustering based on character n-grams in the range 3 to 6 using initial centroids determined from the labeled data. In order to analyze the information of these clusters, we calculated their entropy with respect to two different criteria: marketplaces and vendors. We also checked in the database the number of distinct marketplaces and vendors inside each cluster. The idea was to understand the diversity of the clusters regarding these two facets. A low marketplace entropy for a given cluster would mean its products were mainly found in a particular marketplace. Similarly, low vendor entropy would mean the cluster’s products were mainly sold by a particular vendor. Table III presents the results.

As shown in Table III, Links holds the lowest entropy when we analyzed the marketplaces, suggesting the great majority of products come from the same market. In this cluster, 80% of products came from only 2 markets. However, when we check the vendor entropy for this same cluster, we can observe a higher value, suggesting that many vendors are actually offering products related to Links. It is possible that many markets discourage the re-selling of lists of links, as much of this information can be found on darkweb Wiki’s for free.

Similarly, Hacking Tools holds the lowest entropy for the vendor criteria. This suggest that only a few vendors are present in that cluster. Specifically, only 2 vendors authored 416 (50%) of this type of products. At first glance, this may be surprising as this appears to be a very general group. However, upon inspection of the contents, we find that many authors of these products are actually organizations. These organizations use similar language in their product description in an effort to brand their wares. This could indicate the presence of hacking-collectives that author products as well as the limitations of our text-based approach - which can potentially cluster products branded in a similar fashion. We also note one of the most
prominent vendor in this cluster was itself a marketplace - which is also reflected in the low marketplace entropy.

In our analysis of the Facebook and Keylogger clusters, we can see that they point to the other direction. They have high values for both entropy, a clear sign about the diversity with respect to both vendors and markets. For example, in cluster Facebook, there were 119 products and 67 vendors, and the most prolific vendor for this cluster authored only 8 products. In the same cluster, products were also spread across 15 markets - and the most well-represented market was associated with 30 products. This analysis also indicates widespread prevalence of keyloggers - which is not surprising as it is a well established hacking technique. However, observing the similar trend for the Facebook cluster could be indicative of an increase in demand for Facebook-directed social media hacking products and information.

**Conclusion and Future Work.** In this paper, we conducted an initial examination of malware products from 17 malicious hacker markets through unsupervised learning. Using manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features. We then analyzed manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features. We then analyzed an initial examination of malware products from 17 malicious hacker markets through unsupervised learning. Using manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features.

In this paper, we conducted an initial examination of malware products from 17 malicious hacker markets through unsupervised learning. Using manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features. We then analyzed manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features. We then analyzed an initial examination of malware products from 17 malicious hacker markets through unsupervised learning. Using manually-labeled data, we studied the effect of feature vector on cluster purity using text-based features.