WannaCry Ransomware Attack Hybrid Analysis

Nicholas S Milliken
University of Colorado
CYBR 5320
Boulder, CO
nick.milliken@colorado.edu

Xiaosong Fan
University of Colorado
CYBR 5320
Boulder, CO
xifa4716@colorado.edu

Kenneth Freeman
University of Colorado
CYBR 5320
Boulder, CO
kefr5264@colorado.edu

1. Business Understanding

This study is focused on one of the methods used in a large ransomware attack that occurred in 2017 which was the WannaCry ransomware. The team's goal is to develop a machine learning model that could be used to predict whether network traffic includes malware. To do this we must first gather a dataset and then use that dataset to train models for malware prediction. In order to begin analysis, we must first gather datasets that contain detailed information on network traffic which contain malware. The process begins with requesting full API keys from *hybrid-analysis.com* and downloading their VxAPI wrappers which will allow us to access the raw data feeds that are collected by the site.

The first step in our analysis is to understand how *hybrid-analysis* works and the structure of daily feed used for the machine learning prediction models. *Hybrid-analysis* is a free malware analysis service used to vet through the network traffic. Its advantage is that it can analyze URLs, PCAPS, binary samples, and integration with the Falcon Sandbox. *Hybrid-analysis* also has the capability to generate risk view summaries, incident reports in json files.

Hybrid-Analysis also combines the runtime data with memory dump to extract all historical reports and potential malicious incidents. The incident output report will greatly benefit us in our project to conduct qualitative and quantitative analysis and discover the features we want to predict. We will be using the daily feed and report to build our machine learning model. Our overall goal in this project is to discover the attack incidents that stand out and unique, and study how different factors contribute to the individuality of these incidents.

2. Data Understanding

The data used for the wanacry project was collected from the Hybrid-Analysis website (https://www.hybrid-analysis.com/). In order to gain access to an appropriate file for consideration we needed to apply for an api key from Hybrid-Analysis. Once access was granted we found a file that contained enough of the correct information for us to conduct our analysis. The format was in .json and needed to be converted into a pandas dataframe. Once the data was in proper format there were thirty-seven columns and two-hundred and four rows. It was then necessary to split the report by url to ascertain specific jobid's. Splitting the report would allow us to utilize the various column information. The columns consisted of hash values, threat scores, threatlevel_human, domains etc. There were also a lot of NA values that had to be addressed, most of which were accounted for once we finalized the columns to be used.

Upon closer look at the daily feed reports from *hybrid-analysis*, we discovered the most important variable is whether an incident is malicious or not. Though the data itself does not specify if an incident is

malicious or not, it does indicate the threat level, threat score, number of attacks, and if the incident is interesting. For the rest of the information in the data pull, most are related to the network information of each incident and are not suitable for building a prediction model. Though the threat score and level are given, we want to find the potential incidents that are irregular and provide possibility for further study.

Therefore, the target variable we want to predict is whether an incident is interesting. We will make such a prediction using the threat level, threat score, and number of mitre attacks. During the data cleaning process, we will remove other fields to streamline our analysis and improve the overall efficiency in model development.

3. Data Preparation

Once access to full API keys is granted after a few business days, they will be imputed into python files that are constructed to pull information via json feeds. Time will be spent analyzing the publicly available json feeds from *hybrid-analysis* in order to become familiar with the way the data is organized and will allow the team to find the factors we deem necessary to pull and would want to implement in our models.

After the key json target variables have been identified, the team will input them into the python files designated to pull the network traffic history. Ideally this will create a dataset that we can use to develop our prediction models. The team will create a variety of models in python based on the structure of the data gathered and use that information to see how accurately we can detect the WannaCray ransomware.

For our project, we conducted all our analysis and coding in the Python environment within Jupyter notebooks since it provides clear result output and markdowns for commenting.

Since the raw data was a json file, we will use the packages *requests* and *json* in Jupyter to pull the daily feed data. We first set the parameters of *api-key*, *accept-encoding* to gzip, and *user-agent* to Falcon Sandbox. We will then use the parameters to request the report from the *hybrid-analysis* website. We then used the json function to store the raw data from the website to a pandas data frame. After the data is stored in a panda, we will conduct further observation of the data and understand each column of the new data frame.

With the newly created data frame, we looked at all the columns and its data type. We also created a column checking the content of the response, and removed any record with no response content or information. This will eliminate any potential empty or null value in the dataset.

4. Modeling

The models created to evaluate all of the pulled and combined data included: Random Forest, Decision trees, SVM. The random forest model was run on a 70/30 split and used 100 trees with a max depth of 8 per tree. Before building the decision tree model, the data is split by 80/20 between train and test datasheets. The decision tree classification model was built to determine whether or not it can detect if a report contains a potential malicious attack. The model used threat level, threat score, and number of mitre attacks to detect if a report is interesting.

The SVM model is a supervised model that is designed to find the best decision boundary based on classification algorithms. SVM's work best with data that has two groups (features) to separate. After training, in this case a 70/30 split was used, the model is then able to predict which group new data will be placed into, either wanacry or not wanacry. The data used in this project was simple with few features

making it easy to see the boundaries created by the model, as one downside to SVM is an overlapping of features, fortunately we did not face this issue.

5. Evaluation

In the next section we will address the findings of our research. There will be a characterization of the data, the scalability of our approach, the model's ability to identify malicious traffic and a comparison of approaches. To best understand the topic and build an approach a characterization of the data must be completed.

Random Forest:

The accuracy of the model was 94% at predicting whether or not malware was present and the precision score was 83%. In addition the recall score was .71 and the model had an F1 score of .765. The model also resulted in a false negative rate of 4.5% which is substantial. The results prove to be on the right track but with the small data set more training and tuning is needed before this model can be implemented.

Decision Trees:

The overall accuracy of the model is 91.67%, F1 score of 0.80. and a probability of 83.33%. The prediction results are also plotted on a ROC curve to demonstrate the model's precision and accuracy.

SVM:

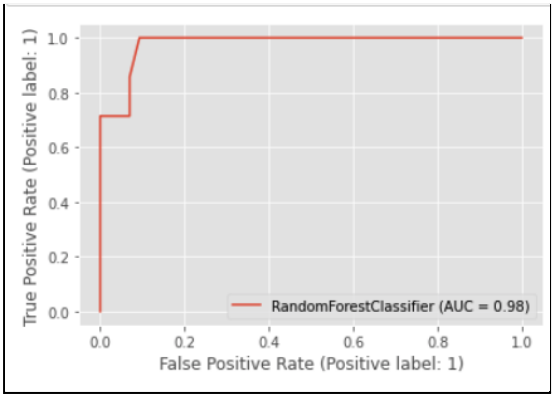
The model demonstrates the ability to identify the wanaCry malware with a precision of 87.5% and has an accuracy of 96.6%. The key features to detecting wanaCry were the number of mitre attacks along with the threat score and threat level of the incoming traffic. A train test split of 70% train data and 30% test data. Our ability to cluster malicious traffic and verify on a test set gives us confidence that our model is doing its job at identifying wanaCry malware.

The approach to the problem is a simple one, collect the data through *hybrid-anlysis*, place it into a dataframe and run it through a support vector machine. The support vector machine model was chosen because it can handle multiple dimensions of data and split them into various groupings. Given that we used functions as a means to process the data, create column information and run that through our model it is reasonable to say that if we scaled up production the model would hold integrity. The model would hold integrity if and only if the data being brought in could easily be put into the categories defined by the dataframe. It should be possible to apply this model horizontally across many servers to look for the wanacry malware.

5.1 Select a model to recommend for deployment

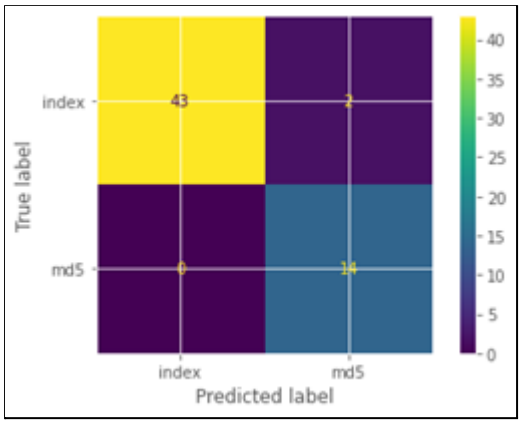
After developing three different machine learning models and comparing between the results ,the model with the most optimal result is the SVM model with 96.6% of accuracy. The SVM model is also capable to be scaled for future data and reports.

a) ROC curve for Random Forest Regression



b) ROC curve for Decision Tree

c) Confusion Matrix from the Support Vector Machine



6. Deployment

When the models are adopted in the future, the new users should first apply for a separate API to be approved by *Hybrid-Analysis* before pulling the daily feed data from the reports. Once the API pull is scalable and thousands of records are available to be collected then we can have a database that is large enough to test on with more accurate results. In addition if we were able to parse out other fields besides the number of mitre attacks then the models should increase in accuracy as well. After these two steps have been implemented and work consistently then we should have the confidence to deploy this model onto a server to screen all incoming files and classify them correctly as benign or malicious.

To ensure this model is being deployed correctly we recommend deploying this on a test server in a safe and contained environment. This allows us to test the model without risking any malware spreading on the actual network until we feel as if the accuracy is high enough and we are not getting any more false negatives. This project allowed us to practice interacting with API's for the first time along with using json files which we had not previously done. In addition it allowed us to practice parsing out information from json formatted data and advance out python data cleaning skills. The biggest drawbacks from this project were only being able to pull one new column from the API summary and ran into a lot of issues reaching the daily maximum requests. If we did not