November 7, 2019

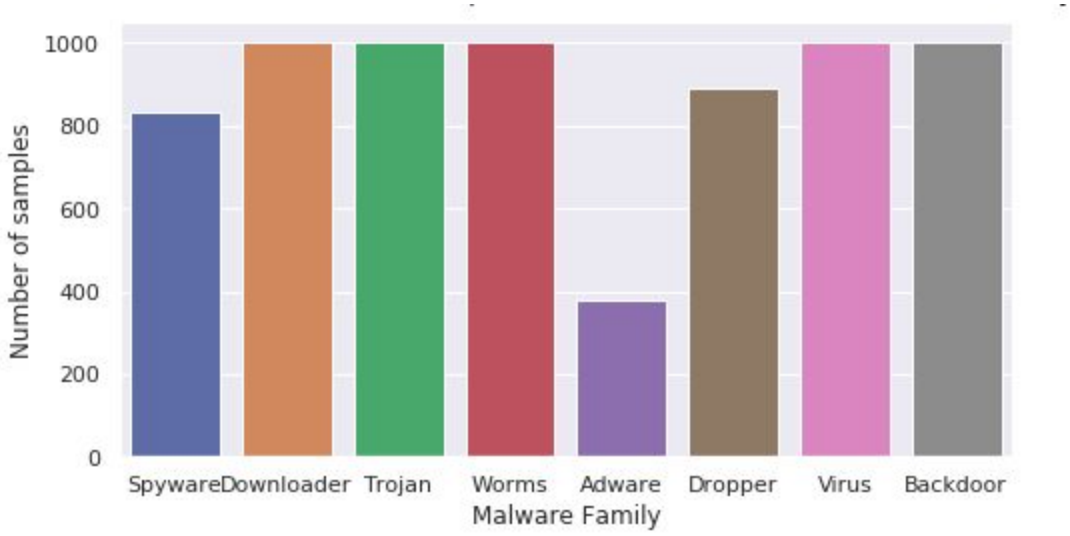
BotBucket: Adversary Resistant Malware Classification Using RNN

Progress Report - CSE 518

Group: Sayli Karnik, Shubhangi Kishore, Binayak Ranjan Das, Atharva Urdhwareshe

- **Progress**

The use of operating system API calls is a promising task in the detection of PE-type malware in the Windows operating system. We plan to leverage **The Mal-API-2019 Dataset** [1] published by Catak and Yazi in 2019. We have have performed exploratory analysis of the dataset and created a baseline model for the dataset. This labeled dataset consists of Windows operating system API calls of various malware. We're using using PyTorch neural network framework for training the many to one RNN model to classify the malwares into 8 families (Trojan, Backdoor, Downloader, Worms, Spyware Adware, Dropper, Virus).



The dataset contains about 7107 rows, each row consists of many binaries separated by space. We first converted the space separated file to a csv file. Corresponding to each row a label is associated which gives the malware.

Our approach classified each row into a vector and inside each vector the binaries have been classified with the help of CALCS method. After classifying each vector, we classified each vector which consists of the comma separated binaries. We used the LSTM algorithm as a RNN approach to classify and train the dataset and then test it to the labels associated with it.

- **Schedule**

We have succeeded in making a baseline model which is a shallow layered RNN. The progress seems to be a week behind what we estimated. Nevertheless, we will continue to improvise the model and make it resistant to adversarial malware samples by 25th November.

- **Obstacles/Workarounds:**

Initially we planned to use the Microsoft dataset on kaggle [2] that consisted of .asm, .byte files of malwares. Further research led us to a newer dataset with API sequences.
Instead of spending effort in reverse engineering these and using LCS, we used the LSTM algorithm as an RNN approach along with the CALCS approach to classify and train the dataset and then test it to the labels associated with it.

- **Preliminary results**

Preliminary analysis of the dataset suggests that each class of malware manifests differently i.e executes different sequence of system API calls on different victim machines. In such a scenario, classifying a malware software by comparing the longest common subsequence of System API calls may not help in accurately classifying the binary. We are utilizing the approach suggested by Patrick et al for text summarization, to train our RNN model on a differentiable loss metric where the longest common subsequence metric is relaxed. The accuracy of our shallow RNN model is 67%. This occurred for the train and test splits of 33% with batch size set to 64 and epochs 3. We will now follow our schedule in order to try to increase the accuracy of our model by experimenting with these hyperparameter values, the included features and other latent features as we deem necessary.

- **Future Milestones**

Deep neural architectures, like all other machine learning approaches, are vulnerable to what is known as adversarial samples. We want to improve on malware code signature based attempts in classifying malware which have used similar Deep Learning Networks. We need to generate windows system calls (API features) for known adversarial malware samples using known adversarial windows malware samples using cuckoo sandbox.[4] We would augment our dataset with these samples along with adding meaningless opcodes to some of the samples. Training on this dataset would enable the classifier to become resistant to adversarial malware samples. We also need to improve on our classification model by making our network deeper and including suitable optimisations. We would also be doing a comparative study of similar models and report our findings.

- **References:**

1. Catak, FÖ., Yazi, AF., *A Benchmark API Call Dataset for Windows PE Malware Classification*, arXiv:1905.01999, 2019.
2. Microsoft Malware Classification, 2015
   https://www.kaggMal-API-2019le.com/c/malware-classification/data

3. Mal-API-2019 Dataset Description
   https://github.com/ocatak/malware_api_class/blob/master/README.md