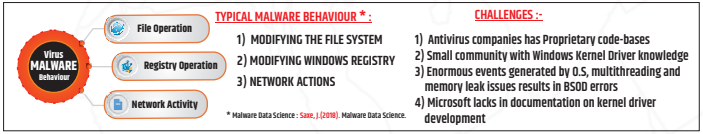
CAPTURING AND ANALYSING WINDOWS KERNEL EVENTS FOR ANOMALY DETECTION

Swapnil Bhosale | Anupam Joshi | Jeff Avery

EBIQUITY RESEARCH LAB, UNIVERSITY OF MARYLAND, BALTIMORE COUNTY

In this work, we have created a kernel resident system to tap into file, registry and network events in Windows O.S. We capture all events relating to files, registry and networks, including processes that are spawned by the O.S. In ongoing work, we have developed a unsupervised neural network LSTM based sequence anomaly detection model that uses this data.



TYPICAL MALWARE BEHAVIOUR * :

1) MODIFYING THE FILE SYSTEM
2) MODIFYING WINDOWS REGISTRY
3) NETWORK ACTIONS

* Malware Data Science : Saxe, J. (2018). Malware Data Science.

CHALLENGES :-

1) Antivirus companies has Proprietary code-bases
2) Small community with Windows Kernel Driver knowledge
3) Enormous events generated by O.S, multithreading and memory leak issues results in BSOD errors
4) Microsoft lacks in documentation on kernel driver development

Fig. : Architecture of Windows Kernel Driver

1) **File:** Use MiniFilter driver tehnique to tap into file events

2) **Network:** Using WFP framework, tap into required OSI network layers. We tap into flow_established and tcp_stream_layer

3) **Registry:** Hook registry calls by registering callback with "CmRegisterCallback" API.

We maintain a doubly linked-list data structure of the events in the kernel memory space. We employ synchronization technique, add records to one one end and remove records concurrently from other

MACHINE LEARNING MODEL IMPLEMENTATION

Data Collection

We capture the normal system usage behavior. For example, we assume all users would log in, check some internet sites, read some mail, use word processing, then log off. This type of session is assumed to be relatively typical of many computer users

The simulated home use of Windows generated a clean (attack-free) dataset for a 1 hour of activity has followin record to predict the timeseries and detect Anomalies.

Record Type	Total Records	Exepriment Run Time
File	2,427,119	~ 1 hours
Registry	539,588	~ 1 hours
Network	1,432,243	~ 1 hours

Sentence Embedding using FastText

fig: FastText text embedding data flow

First of all, we need a way to represent events to train a Machine Learning model using this data.We used the FastText sentence embedding NLP technique to represent each event as a vector. FastText has a "window size" hyperparameter, which allows the model to look back at many records to exploit spatial locality. We believe that this can help the model capture the process behavior with file, network, and registry events. Thus, the dataset is group by process names and then sorted by timestamp. Then the dataset is fed to FastText for training.We then use the t-Distributed Stochastic Neighbor Embedding (t-SNE) algorithm to reduce dimensions to 2D. The plot of the words shows that related events are appearing together.

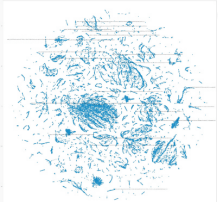


Fig: t-SNE visualization for the training dataset.

Anomaly detection using LSTM

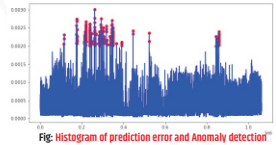


Fig: Histogram of prediction error and Anomaly detection

We experiment with the Autoencoder model and train it to reconstruct the event sequences of size n. To detect anomalies, the prediction model is supplied with 'X, 'Y' and the distance between its output 'y^' and 'y' is measured using the MSE loss function. The current event sequence window is label as anomalous if the distance is greater than the threshold. We also experiment by stacking multiple LSTM layers and trained to predict the next window sequence. To evaluate the model, we take over the running Java process using Meterpreter. The malicious activities are performed such as downloading files, keylogging the keyboard strokes, etc. The constructions errors are then plot and anomalies are detected

8TH ANNUAL HOT TOPICS in the SCIENCE OF SECURITY
APRIL 13-15, 2021 | VIRTUAL