



University of
New Haven

Midterm Project
AI and CyberSecurity
DSCI6015
Simplified Midterm

Roshan Mummadi

00887116

University of New Haven

Dr. Vahid Behzadan

April 01, 2024

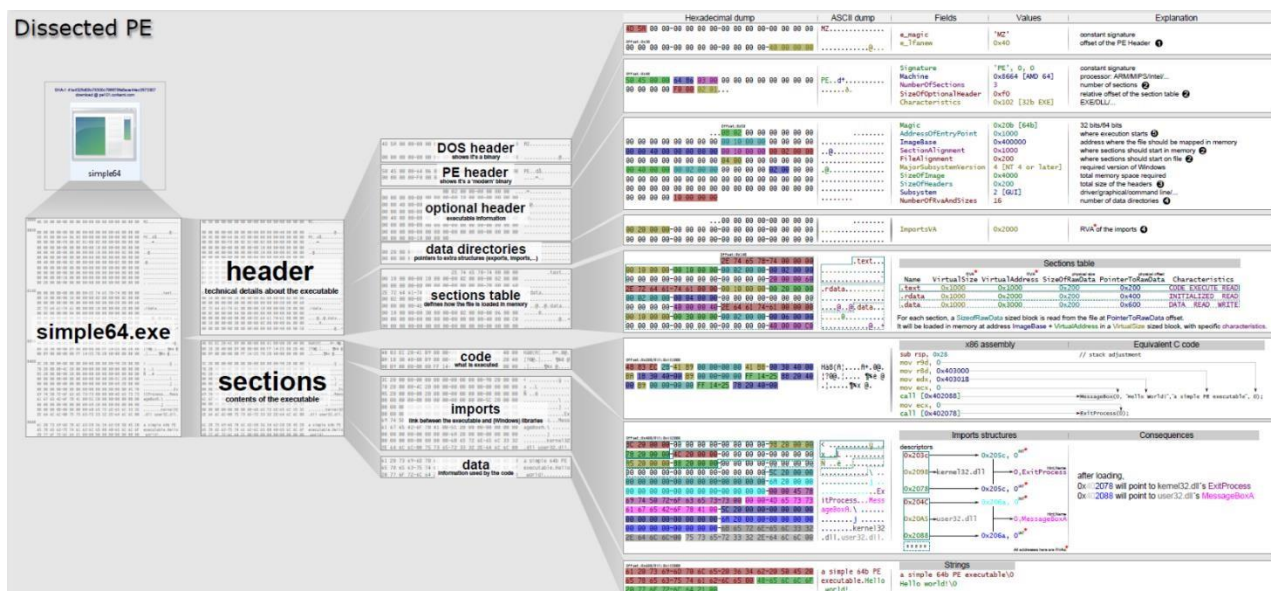
Summary

This paper describes how AWS Sagemaker was used to successfully create a cloud-based Portable Executable (PE) static malware detection API. The API employs a Random Forest binary classifier, trained on a labeled dataset of binary feature vectors, to categorize PE files as benign or harmful. Throughout the project, AWS Sagemaker was utilized for model building, training, and deployment. Furthermore, an easy-to-use web application was created to facilitate remote file uploads for threat detection. Python was the primary programming language, while machine learning tools such as sklearn, pefile, and nltk were used for model construction and implementation.

Introduction

PE Files

PE files are commonly used by Windows operating systems to hold executable code and associated data. These files contain metadata, imported libraries, machine instructions, and other vital information required to run the program. They are often used for drivers, applications, and dynamic link libraries (DLLs). They have a methodical layout with headers that provide information about the file's attributes, such as its architecture, entry point, and section order. The PE file format is crucial for tasks like malware detection, reverse engineering, and software analysis because it allows the examination and modification of executable code.



Random Forest Classifier

The Random Forest classifier is a flexible and dependable machine learning method used for both regression and classification tasks. This method is classified as ensemble learning since it creates predictions by merging several different models. Within the framework of Random Forests, individual models are decision trees. The term "forest" in Random Forest refers to a grouping of decision trees, each of which was constructed independently and makes decisions based on a specific set of input attributes.

The fundamental principle of Random Forests is to nurture a diverse range of decision trees by introducing randomness into both the prediction and training phases. During training, each tree is constructed using a random subset of features and a random subset of training data.

Random forests offer substantial advantages, particularly when working with high-dimensional data that has a large number of attributes. They exhibit tolerance to noise and outliers within the dataset and require significantly less hyperparameter tuning than more sophisticated models. Moreover, Random Forests offer valuable insights into feature relevance, enabling users to pinpoint the features that most

influence the model's predictions. As a result, Random Forests are extensively utilized across numerous industries.

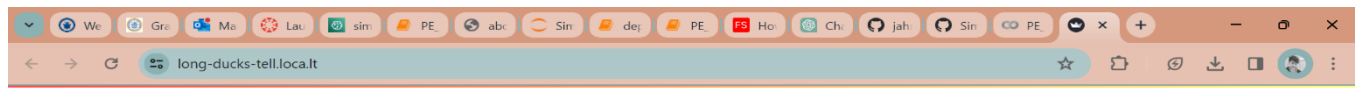
Task Approach:

Building and Training the Model: Scikit-learn 1.2.1 is the version used for development and training in AWS Sagemaker. The Random Forest binary classifier was trained using a dataset of binary feature vectors that had been accurately classified. After that, the trained model is dropped into a joblib file.

Deploying the Model as a Cloud API: The trained model was used with Amazon Sagemaker to establish an endpoint for a cloud-based real-time prediction API. The model is loaded with the help of the stored joblib file, and it is then ready for deployment using Sagemaker.Module SKLearn. The model gets deployed to an Endpoint that has been set up and configured.

Creating a Client Application: An easy-to-use user interface was built for a well-lit web application. Users have the ability to upload executable (.exe) files to the webclient. The webclient uses pefile lib and additional pretrained data to extract the necessary features from the.exe file. The features are then delivered to the deployed API after being converted to JSON. The classification results (Malware - Danger or Benign - Safe) are subsequently shown by the program. I launched the streamlit application using Google Colab and utilized it for the client deployment.


The screenshot shows a web browser window with the address bar displaying 'long-ducks-tell.loca.lt'. The page content includes a warning message: 'You are about to visit: long-ducks-tell.loca.lt'. Below this, it states 'This website is served for free via a localtunnel.' and 'You should only visit this website if you trust whoever sent this link to you.' A warning about phishing is also present: 'Be careful about giving up personal or financial details such as passwords, credit cards, phone numbers, emails, etc. Phishing pages often look similar to pages of known banks, social networks, email portals or other trusted institutions in order to acquire personal information such as usernames, passwords or credit card details.' A section titled 'Please proceed with caution.' follows. Then, it says 'To access the website, please enter the tunnel password below.' and 'If you don't know what it is, please ask whoever you got this link from.' There is a text input field labeled 'Tunnel Password:' with a masked password '*****'. Below the field is a blue button labeled 'Click to Submit'. At the bottom, it asks 'Are you the developer?' and provides instructions for developers: 'If you're the developer of this website, please read this:' followed by two bullet points: '• We display this page to prevent abuse.' and '• You and other visitors will only see this page from a standard web browser once per public IP every 7 days.'



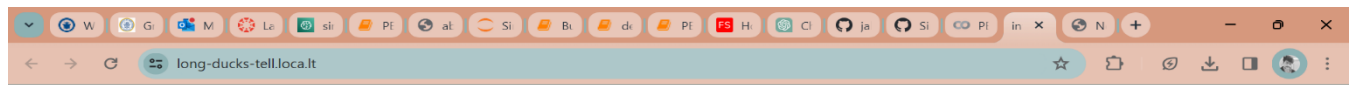
⋮

SageMaker Inference with Streamlit

Upload .exe file

 Drag and drop file here
Limit 200MB per file • EXE


Browse files




⋮

SageMaker Inference with Streamlit

Upload .exe file

 Drag and drop file here
Limit 200MB per file • EXE

Browse files


 VirusShare_00b06a1fedb082b09b04c6995290097d.exe 204.0KB ×

Benign - Safe



SageMaker Inference with Streamlit

Upload .exe file

 Drag and drop file here
Limit 200MB per file • EXE

Browse files

 ActivateApplication.exe 23.4KB ×

Benign - Safe

Project Results

By creating a malware detection model that can differentiate between safe and malicious PE files, the research was able to achieve its goals. After a successful training process, this model was made available as a real-time prediction API on Amazon Sagemaker. Furthermore, an intuitive web-based interface was developed, enabling users to input information and evaluate the possible degree of risk connected to those kinds of files.

Result and Conclusion

The information above shows how precise my models are. The model's accuracy in predicting the type of exe file is 97.47% of the time, with an accuracy of 0.9979035639412998.

Conclusion

Developing and deploying a cloud-based PE static malware detection API was the project's successful completion. The study demonstrates how machine learning can be used to identify malware effectively and how scalable and user-friendly applications can be made using cloud platforms like Google Colab and Amazon Sagemaker.

Resources:

- <https://github.com/endgameinc/ember>
- <https://github.com/endgameinc/ember/tree/master/malconv>
- <https://github.com/UNHSAIILab/S24-AISec/tree/main/Midterm%20Tutorial>
- <https://sagemaker-examples.readthedocs.io/en/latest/intro.html>
- https://sagemakerexamples.readthedocs.io/en/latest/frameworks/pytorch/get_started_mnist_train_outputs.html
- <https://docs.aws.amazon.com/sagemaker/latest/dg/deploy-model.html>
- <https://github.com/RamVegiraju/Pre-Trained-Sklearn-SageMaker>
- <https://youtu.be/ueI9Vn747x4?si=KSFTvR9hBnU0u0DO>
- <https://youtu.be/oOqqwYI60FI?si=3WKd-iDz93mm1Vbe>
- https://youtu.be/g6kQl_EFn84?si=9MHbO9I52AS2pPjx