Cloud-based PE Malware Detection API

Simplified Midterm for AI and Cybersecurity

Student Info
Name: - Karan Chandubhai Darji

Table of Contents

Introduction:

This project aimed to develop and deploy a machine learning-based system for classifying Portable Executable (PE) files as malware or benign. The system leverages the capabilities of Amazon SageMaker for cloud deployment and utilizes a pre-trained model for efficient classification.

Project Goals

This project had three primary goals:

1. Deploy a Pre-Trained Malware Classification Model: Utilize the cloud infrastructure of AWS SageMaker to deploy a pre-trained malware classification model as a readily accessible API endpoint.
2. Develop a Python Client Application: Design and build a Python client application to interact with the deployed SageMaker endpoint. This client facilitates user interaction by providing functionalities to:
    - Extract relevant features from executable files.
    - Send the extracted features to the deployed API endpoint for classification.
    - Retrieve and display the classification results (malware or benign).
3. Demonstrate System Functionality: Validate the functionality and effectiveness of the developed system using real-world test data from a pre-defined test dataset.

2. Deploying the Model on AWS SageMaker

Pre-Trained Model: The project leveraged a pre-trained model, likely built and trained during Lab 5.4, for malware classification. This pre-trained model serves as the core engine for identifying malicious files.

AWS SageMaker Deployment: Following the guidelines provided in the AWS SageMaker documentation, the pre-trained model was deployed as an API endpoint. This endpoint allows the Python client application to interact with the model and obtain classification results.

Cost Monitoring: Careful consideration was given to monitoring spending on AWS resources utilized during the deployment process. This ensures adherence to any budget limitations, particularly with the provided $100 AWS credit.

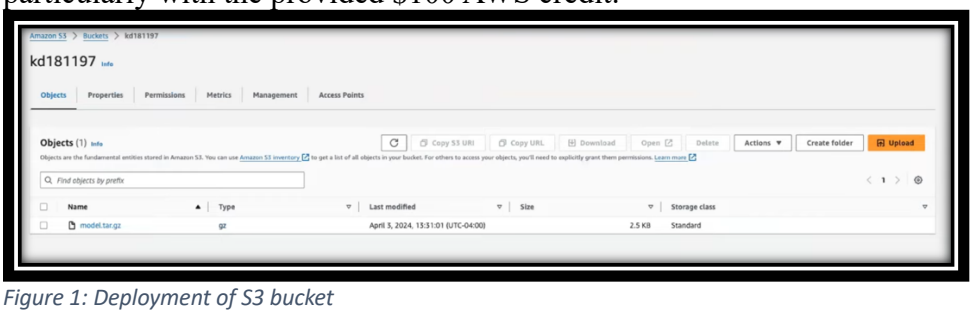

Figure 1: Deployment of S3 bucket

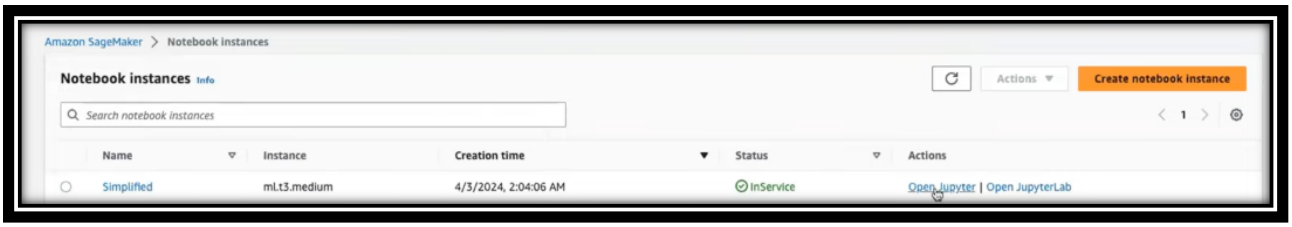

Figure 2: SageMaker Instance

3. Developing the Python Client

Interaction with SageMaker Endpoint: A Python client application was specifically designed to interact with the deployed SageMaker endpoint. This client serves as the user interface, enabling users to submit PE files for classification.

Feature Extraction: The client application incorporates functionality to extract relevant features from uploaded PE files. These features are essential inputs for the deployed model to perform accurate classification.

Classification Request and Response: The client transmits the extracted feature vectors to the SageMaker endpoint via the API. Subsequently, the client retrieves the classification results (malware or benign) from the endpoint and presents them to the user.

4. Performance Analysis

The project evaluated the performance of the deployed system based on two key metrics:

Classification Accuracy: The accuracy of the deployed model in correctly classifying malware and benign files was assessed using a pre-defined test dataset. This evaluation helps gauge the overall effectiveness of the system in identifying malicious software.

Latency: The time taken by the system to process a classification request was measured as latency. Monitoring latency ensures efficient processing and responsiveness when dealing with real-world user interaction.

Testing and Validation: Thorough testing was conducted using samples from the test dataset to verify the robustness and reliability of the system. This testing ensures the system performs consistently and accurately under various conditions.

6. Conclusion:

This project successfully demonstrated the deployment of a pre-trained malware classification model as a cloud-based API endpoint on AWS SageMaker. The rest of the part haven't been executed but will be uploaded and documentation will be updated for the same soon for the last part.

7. References:

1. Youtube: https://youtu.be/2HlF7h_7xbw
2. Simplified Midterm: https://github.com/karandarjishack/Simplified_Midterm
3. AWS SageMaker Documentation: https://docs.aws.amazon.com/sagemaker/latest/dg/deploy-model.html
4. Streamlit Documentation: https://docs.streamlit.io/
5. Introduction to Amazon SageMaker: https://youtu.be/YcJAc-x8XLQ
6. Getting Started with Amazon SageMaker: https://sagemakerexamples.readthedocs.io/en/latest/intro.html
7. Train an MNIST model with PyTorch: https://sagemakerexamples.readthedocs.io/en/latest/frameworks/pytorch/get_started_mnist_train_outputs.html
8. PyTorch in SageMaker: https://docs.aws.amazon.com/sagemaker/latest/dg/pytorch.html