Progress Report

Introduction

Cyber Attacks and Security was first found to be necessary in 1970 when network connectivity was established, and that was due to new advancements in establishing a network that showed a computer program can move through the network. Now, we have more security concerns to be looking out for in the modern advancements of Machine Learning and generative ai models as upcoming advancements as in the same way Bob showed a computer program creeper and traverse the network, it is found that you can interfere and disrupt the Machine model's learning phase to take control and access it for malicious intentions. This calls for a better defense, and just as the Machine learning model can be used for cyber attacks, we can use them for better defenses as well. Intrusion Detection Systems (IDSs) are essential for monitoring and identifying suspicious behaviors in digital environments. There are three main approaches to intrusion detection: misuse-based, anomaly-based, and hybrid techniques. Misuse-based detection identifies attacks by comparing system activity to a database of known attack signatures. This works well for recognizing documented threats, but it struggles with new or evolving attacks. In contrast, this detection creates models of normal system behavior and flags any deviations as potential attacks. This method is more flexible and can detect zero-day attacks, but it often leads to high false positive rates. To address these issues, hybrid detection methods combine misuse and anomaly-based strategies, aiming to improve detection accuracy while reducing false alarms.

Problem Definition/Research progress

The main method that will be implemented will be a hybrid detection method on a UNSW_NB15 dataset since they reflect real world host

based attacks as samples focusing on worms because of the complexities of Malware being developed and executed, so there is a demand to advance better defenses against these Malwares, especially with Machine Learning.

Research Comparisons
To utilize these techniques effectively, machine learning (ML) provides powerful tools to automate the detection process. In this project, I'll use three specific ML models suited for each detection type: a Decision Tree classifier for misuse-based detection, K-Means clustering for anomaly-based detection, and an ensemble Random Forest model for hybrid detection.

So far, the only codes I have implemented are setting everything up to prepare for a machine learning model such as downloading pytorch, anaconda, all the libraries for each model, docker, tensorflow and setting the environment up for virtual simulation.
Research resources include many papers and datasets from this github:https://github.com/jivoi/awesome-ml-for-cybersecurity
To prepare for the application implementation. Because this is an application flavored topic, I will not be following word for word as described on a research paper, therefore, most of my current resources include youtube, articles, and sample codes. Here I provided all the references so far:
https://www.geeksforgeeks.org/machine-learning/intrusion-detection-system-using-machine-learning-algorithms/

I was able to train/split and clean the dataset due to my previous experiences working with machine learning.

Code so far:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
df = pd.read_csv("UNSW_NB15_training-set.csv")
df.drop(['id'], axis=1, inplace=True)
df.replace([np.inf, -np.inf], np.nan, inplace=True)
df.dropna(inplace=True)
cat_cols = df.select_dtypes(include='object').columns
label_encoders = {}
for col in cat_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le
X = df.drop(['label'], axis=1)
y = df['label']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.3, random_state=42)
```

Milestones:

7/11- 7/22 = finish code implementation

- 7/11-7/15= finish the detection implementation
- 7/15- 7/19= finish classification
- 7/19-7/22 = finish performance evaluation calculations and implementation

7/22-7/26=finish slide presentation+demo

7/27-7/31=Finish project Report

Division of workload: Me