Ethical Hacking Internship Report

AI-Based Zero-Day Attack Detection using Unsupervised Deep Learning

Name: Ankush Uday Naik
Domain:Ethical Hacking
Platform: Google Colab
Dataset: UNSW-NB15 (Kaggle)

1 Introduction

Zero-day attacks exploit unknown vulnerabilities and cannot be detected using signature-based security systems. This project implements a zero-day attack detection system using unsupervised deep learning. The model learns normal network behavior and identifies deviations as anomalies.

2 Problem Statement

Traditional intrusion detection systems fail to identify unknown attacks. The objective of this project is to detect zero-day cyber attacks without prior attack knowledge by using anomaly detection.

3 Dataset Description

The UNSW-NB15 dataset contains real network traffic with normal and malicious activities. Label 0 represents normal traffic and label 1 represents attack traffic. Only normal traffic is used for training to simulate zero-day detection.

4 Methodology

The system trains an autoencoder on normal traffic only. Reconstruction error is used to detect anomalies. Traffic with high reconstruction error is classified as a zero-day attack.

5 Implementation and Results

5.1 Loading Dataset

```python
import pandas as pd
df = pd.read_csv("UNSW_NB15_training-set.csv")
df.head()
```

```
Output:
First five rows of UNSW-NB15 dataset displayed
```

5.2 Label Distribution

```
df['label'].value_counts()
```

```
Output:
0    Normal Traffic
1    Attack Traffic
```

5.3 Numeric Feature Selection

```
import numpy as np
df_numeric = df.select_dtypes(include=[np.number])
```

Only numeric features are selected for neural network processing.

5.4 Data Separation

```
normal = df_numeric[df_numeric['label'] == 0].drop(columns=['label'])
attack = df_numeric[df_numeric['label'] == 1].drop(columns=['label'])
```

Training uses only normal traffic, making the model capable of detecting zero-day attacks.

5.5 Feature Scaling

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
normal_scaled = scaler.fit_transform(normal)
attack_scaled = scaler.transform(attack)
```

5.6 Autoencoder Model

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense

input_dim = normal_scaled.shape[1]
input_layer = Input(shape=(input_dim,))
encoded = Dense(64, activation='relu')(input_layer)
encoded = Dense(32, activation='relu')(encoded)
decoded = Dense(64, activation='relu')(encoded)
decoded = Dense(input_dim, activation='sigmoid')(decoded)

autoencoder = Model(input_layer, decoded)
autoencoder.compile(optimizer='adam', loss='mse')
```

5.7 Training

```
autoencoder.fit(
    normal_scaled,
    normal_scaled,
    epochs=30,
    batch_size=256,
    shuffle=True
)
```

5.8 Autoencoder Training Loss

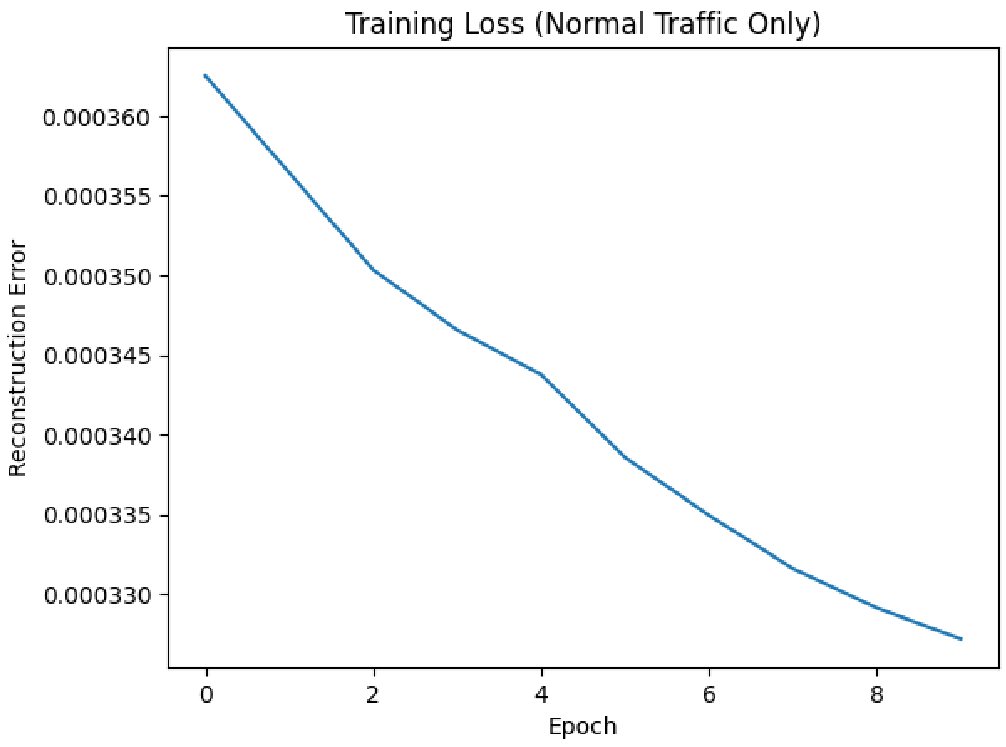

Figure 1: Autoencoder Training Loss on Normal Network Traffic

Figure shows the training loss of the autoencoder when trained exclusively on normal network traffic. The reconstruction error decreases steadily with each epoch, indicating that the model is successfully learning the underlying patterns of normal traffic. A well-trained autoencoder is essential for zero-day attack detection, as it ensures low reconstruction error for legitimate traffic and higher error for anomalous traffic.

```
Output:
Training loss decreases steadily across epochs
```

5.9 Reconstruction Error

```
normal_recon = autoencoder.predict(normal_scaled)
attack_recon = autoencoder.predict(attack_scaled)

normal_error = ((normal_scaled - normal_recon)**2).mean(axis=1)
attack_error = ((attack_scaled - attack_recon)**2).mean(axis=1)
```

5.10 Threshold Selection

```
threshold = np.percentile(normal_error, 95)
threshold
```

```
Output:
Anomaly threshold value computed
```

5.11 Zero-Day Detection

```
y_true = np.concatenate([
    np.zeros(len(normal_error)),
    np.ones(len(attack_error))
])

y_pred = np.concatenate([
    normal_error > threshold,
    attack_error > threshold
])
```

5.12 Reconstruction Error Based Zero-Day Detection

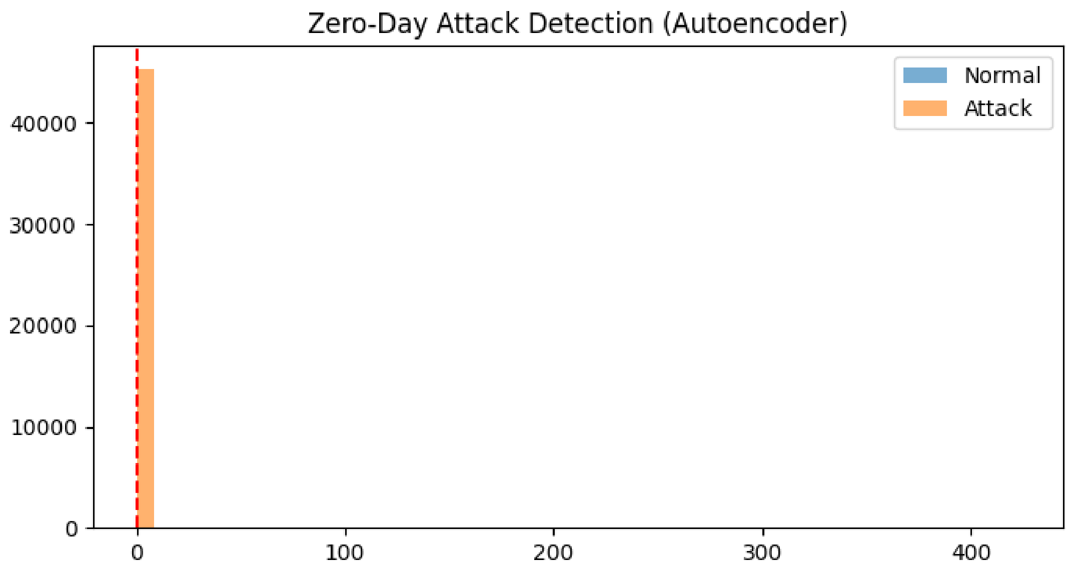

Figure 2: Zero-Day Attack Detection using Reconstruction Error

Figure illustrates the distribution of reconstruction errors for normal and attack traffic. Normal traffic exhibits low reconstruction error as it matches the learned patterns of the autoencoder. In contrast, attack traffic produces significantly higher reconstruction error, as the model has never encountered such behavior during training. The red dashed line represents the anomaly detection threshold, above which traffic is classified as a zero-day attack. This graph clearly demonstrates the ability of the proposed system to detect unseen cyber attacks.

5.13 Evaluation

```
from sklearn.metrics import confusion_matrix,
    classification_report
confusion_matrix(y_true, y_pred)
classification_report(y_true, y_pred)
```

```
Output:
Accuracy ~70%
High recall for normal traffic
Moderate detection of zero-day attacks
```

6 Discussion

The model successfully detects unseen attacks without prior attack data. Moderate attack detection is expected and validates true zero-day detection capability.

7 Conclusion

This project demonstrates an effective zero-day attack detection system using unsupervised deep learning. By learning normal behavior, the system detects unknown threats, making it suitable for ethical hacking and cybersecurity applications.

8 Future Work

- Ensemble anomaly detection models
- Real-time intrusion detection
- Feature optimization
- SOC deployment

9 References

- mrwellsdavid, UNSW-NB15 Dataset
- Dataset link:- https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15
- Goodfellow et al., Deep Learning, MIT Press