**SCHOOL OF COMPUTER SCIENCE**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**DEHRADUN, UTTARAKHAND**



**DIGITAL FORENSICS**

**ASSIGNMENT FILE - 1**

**(2024-2025)**

**For**

**Vth Semester**

**Submitted To: Submitted By:**

Prof. Subhranil Das Mr. Akshat Negi

Assistant Professor S.S. 500106533(SAP ID)

[Vth Semester] R2142220414(Roll No.)

School of Computer Sciences B.Tech. CSF (Batch-1)

**Case Study - 1**

information about any device. Using the Nmap, we can gather information about any client

**Data Preprocessing:**

**Cleans the network traffic data and extracts relevant features, scaling them for uniformity.**

import pandas as pd  
from sklearn.preprocessing import StandardScaler  
  
data = pd.read\_csv('network\_traffic.csv')  
data = data.dropna()  
  
features = data[['src\_ip', 'dst\_ip', 'src\_port', 'dst\_port', 'timestamp', 'packet\_size', 'protocol']]  
  
scaler = StandardScaler()  
normalized\_features = scaler.fit\_transform(features)

**Explanation**:

* The script first loads network traffic data using pandas.
* Missing values in the dataset are removed with dropna().
* Specific columns, such as source IP, destination IP, ports, timestamps, packet sizes, and protocols, are extracted to form the relevant feature set.
* The extracted features are then scaled using StandardScaler to ensure they are on the same scale, which is important for applying machine learning models.

**Anomaly Detection Using DBSCAN:**

**Detects potential network intrusions by finding anomalous data points that deviate from typical traffic behaviour.**

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.5, min\_samples=5)

dbscan\_anomalies = dbscan.fit\_predict(normalized\_features)

**Explanation**:

* The DBSCAN clustering algorithm is applied to the normalized network data.
* eps=0.5 defines the maximum distance between two samples for them to be considered in the same neighborhood.
* min\_samples=5 defines the minimum number of points required to form a cluster.
* Points classified as -1 are considered noise (anomalies), while other values represent different clusters.

**Pattern Analysis:**

**Analyzes the anomalies to uncover attack patterns, such as which IP addresses and protocols were most involved, helping to infer how the breach occurred.**

anomalous\_data = data[anomalies == -1]  
print(anomalous\_data['src\_ip'].value\_counts())  
print(anomalous\_data['protocol'].value\_counts())

**Explanation**:

* This section filters the data to include only the anomalous entries (where anomalies == -1).
* The script then prints the most frequent source IP addresses and protocols found in the anomalies.
* This helps identify which IPs were most involved in suspicious activities and the protocols potentially exploited.

**Case Study - 2**

**Data Preprocessing**

**The network traffic dataset is loaded, cleaned, and key features are extracted for analysis.**

import pandas as pd

from sklearn.preprocessing import StandardScaler

data = pd.read\_csv('network\_traffic.csv')

data = data.dropna()

features = data[['src\_ip', 'dst\_ip', 'src\_port', 'dst\_port', 'timestamp', 'packet\_size', 'protocol']]

scaler = StandardScaler()

normalized\_features = scaler.fit\_transform(features)

**Explanation**:

* The dataset is loaded using pandas and cleaned by removing any missing values.
* Relevant features (source/destination IPs, ports, timestamps, packet sizes, and protocols) are extracted from the dataset.
* The StandardScaler scales the extracted features, ensuring uniform scaling across all features, which is necessary for applying machine learning algorithm

**Anomaly Detection (DBSCAN)**

**The Isolation Forest algorithm isolates anomalies by randomly partitioning thedataset. Alternatively, DBSCAN detects anomalies by identifying outliers that don't belong to any dense cluster. Both approaches help identify unusual network traffic that might indicate a breach.**

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.5, min\_samples=5)

dbscan\_anomalies = dbscan.fit\_predict(normalized\_features)

**Explanation**:

* The DBSCAN algorithm is applied to detect clusters in the normalized data, where points that don’t belong to any cluster are marked as anomalies.
* eps=0.5 specifies the maximum distance between two samples to consider them in the same neighborhood, while min\_samples=5 indicates the minimum number of points required to form a cluster.
* Anomalous points (noise) are labeled as -1.

**Pattern Analysis**

**The anomalous data is examined to identify frequent IP addresses and commonly used protocols, which can reveal the methods and tools used by attackers, providing insight into the nature of the attack and network vulnerabilities.**

anomalous\_data = data[anomalies == -1]  
print(anomalous\_data['src\_ip'].value\_counts())  
print(anomalous\_data['protocol'].value\_counts())

**Explanation**:

* This filters out the anomalous data points (where anomalies == -1).
* The script then prints the most frequent source IP addresses and the most commonly used protocols in the anomalies, providing insight into which IPs or protocols are involved in suspicious activities.