**Network Security Data Analysis Project**

**Problem Statement**

**Overview**

In this project, you will apply your knowledge of Python basics, NumPy, and Pandas to analyze a dataset related to network security and monitoring. The dataset simulates real-world network traffic in a corporate environment, where devices communicate over the network, and security events are logged. This is common in IT infrastructure, where analyzing traffic helps detect anomalies like potential cyber threats (e.g., data exfiltration or brute-force attacks), optimize performance, and ensure compliance.

You have been provided with three related datasets:

1. **Network Traffic Log** (traffic.csv): Hourly records of network flows between devices.
2. **Security Events** (security.csv): Logs of security incidents, such as login failures or malware detections.
3. **Device Info** (device.csv): Static information about devices and their IPs.

Your goal is to perform Exploratory Data Analysis (EDA) on these datasets, clean them based on your findings, join them into a unified DataFrame, and then conduct further analysis and visualization to derive insights. This project will help you practice data loading, manipulation, cleaning, merging, statistical analysis, and visualization—key skills for data science in domains like networking and cybersecurity.

**Objectives**

By the end of this project, you should be able to:

* Load and explore datasets using Pandas and NumPy.
* Identify and clean anomalies (e.g., missing values, outliers, duplicates).
* Join multiple datasets to create a richer view of the data.
* Perform univariate and bivariate analysis to uncover patterns.
* Use groupby operations to aggregate data.
* Visualize findings to communicate insights effectively.

**Steps to Follow**

Follow these steps in sequence. Use Jupyter Notebook or a Python script to document your code, explanations, and outputs. Include comments in your code for clarity.

1. **Data Loading**:
   * Load the three CSV files into Pandas DataFrames: traffic\_df, security\_df, and device\_df.
   * Convert the timestamp columns in traffic\_df and security\_df to datetime format using pd.to\_datetime().
   * Display the first few rows, shape, and data types of each DataFrame using .head(), .shape, and .info().
2. **Exploratory Data Analysis (EDA)**:
   * **Univariate Analysis**: For each numerical column (e.g., bytes\_sent, duration\_sec), compute summary statistics using .describe(). Calculate skewness using NumPy or SciPy. Plot histograms or boxplots to visualize distributions (use Matplotlib or Seaborn).
   * **Bivariate Analysis**: Explore relationships between pairs of variables, e.g., scatter plot of bytes\_sent vs. duration\_sec, or correlation matrix using .corr(). Group by categorical variables like protocol and compute means.
   * **Groupby Operations**: Use .groupby() to aggregate data, e.g., total bytes\_sent by protocol or average duration\_sec by source\_ip.
   * **Identify Anomalies**: Look for missing values (.isnull().sum()), duplicates (.duplicated()), negative values, zeros in inappropriate places, and outliers (e.g., using IQR method).
   * Document insights, e.g., "The bytes\_sent distribution is right-skewed due to a few high values, suggesting potential anomalies."
3. **Data Cleaning**:
   * Based on your EDA, clean the data. For example:
     + Remove duplicates using .drop\_duplicates().
     + Impute missing values with median (use .fillna()) for numerical columns.
     + Fix negative duration\_sec by taking absolute values or dropping rows.
     + Handle outliers by capping them at the 99th percentile using NumPy's np.clip().
     + Filter invalid zeros (e.g., in bytes\_sent for active connections).
   * Re-run summary statistics and plots after cleaning to verify improvements.
   * (Optional Challenge: Create a custom function to encapsulate the cleaning steps for reusability.)
4. **Joining the Datasets**:
   * Merge traffic\_df with device\_df on source\_ip and dest\_ip (use .merge() with how='left' to add device details).
   * Merge the result with security\_df on timestamp and IP (match source\_ip or dest\_ip to ip in security logs). Handle multiple matches if needed.
   * The final merged DataFrame should include columns from all three datasets, enabling analysis like "traffic volume during security events."
   * Check for any data loss post-merge using .shape and handle NaNs if introduced.
5. **Further Analysis and Visualization**:
   * With the merged DataFrame, perform advanced EDA:
     + Groupby: e.g., average bytes\_sent by device\_type or event\_type.
     + Bivariate: Scatter plots colored by severity or protocol.
     + Time-series analysis: Resample by day (resample('D')) and plot total bytes\_sent over time.
   * Visualize key insights:
     + Bar charts for categorical summaries (e.g., bytes by protocol).
     + Heatmaps for correlations.
     + Line plots for trends over timestamps.
   * Derive 3-5 key insights, e.g., "TCP traffic from IoT devices shows higher anomalies during 'high' severity events."
6. **Conclusion and Recommendations**:
   * Summarize your findings in a short report section.
   * Suggest real-world applications, e.g., "This analysis could help in building an anomaly detection model for cybersecurity."

**Deliverables**

* A Jupyter Notebook or Python script with all code, outputs, and explanations.
* At least 5 visualizations.
* A list of insights derived from the analysis.

**Tips**

* Import necessary libraries: import pandas as pd, import numpy as np, import matplotlib.pyplot as plt, import seaborn as sns.
* Handle errors gracefully (e.g., check for non-numeric columns in correlations).
* For ML preparation (future extension): Add a column like is\_anomalous based on thresholds (e.g., high bytes\_sent or specific event\_type).
* If stuck, refer to Pandas documentation or your class notes on NumPy arrays for statistical computations.

**Data Dictionary**

**1. Network Traffic Log (traffic.csv)**

This table contains 205 rows of simulated hourly network flow data.

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| timestamp | datetime | Date and time of the network flow (e.g., '2025-01-01 00:00:00'). |
| source\_ip | string | IP address of the source device. |
| dest\_ip | string | IP address of the destination device. |
| protocol | string | Network protocol used (e.g., 'TCP', 'UDP', 'ICMP'). |
| bytes\_sent | float | Number of bytes sent from source to destination (may contain NaNs, zeros, or outliers). |
| bytes\_received | float | Number of bytes received by source from destination (may contain zeros or outliers). |
| duration\_sec | float | Duration of the flow in seconds (may contain NaNs, negatives, or outliers). |
| packet\_count | integer | Number of packets exchanged in the flow. |

**2. Security Events (security.csv)**

This table contains 53 rows of security incident logs, with some duplicates.

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| timestamp | datetime | Date and time of the security event. |
| ip | string | IP address associated with the event (matches source\_ip or dest\_ip in traffic). |
| event\_type | string | Type of event (e.g., 'login\_fail', 'malware\_detect', 'firewall\_block', 'login\_success'). |
| severity | string | Severity level (e.g., 'low', 'medium', 'high'). |

**3. Device Info (device.csv)**

This table contains 9 rows of static device information.

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| ip | string | IP address of the device (joins with source\_ip, dest\_ip, or ip in other tables). |
| device\_type | string | Type of device (e.g., 'workstation', 'router', 'IoT', 'server'). |
| location | string | Location of the device (e.g., 'data\_center', 'remote'). |