CYBER ATTACK ANALYSIS: SYN FLOOD ATTACK

Network Traffic Threat Intelligence Project

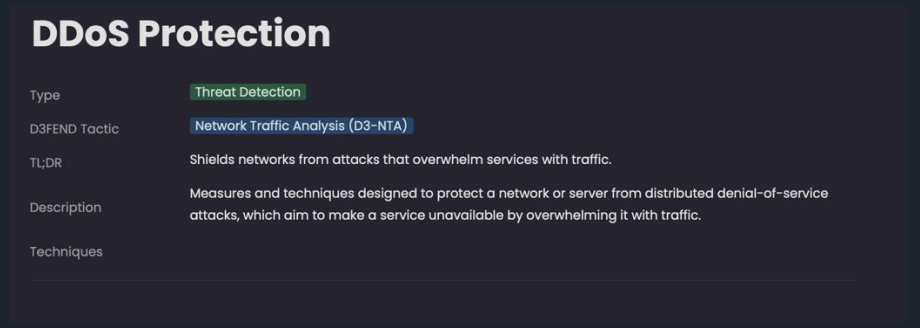
Yovel Hadari

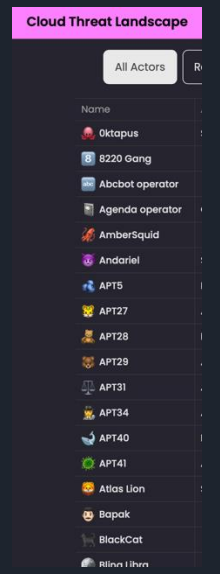
Hakeriot - Mentoriot

ABOUT

Analysis of a dataset comprising network traffic logs and packets to enhance hunting and detection capabilities using Python.

- Identifying anomalies and correlations between vectors and characteristics during an APT attack, specifically targeting DoS (Denial of Service) incidents.
- DoS/DDoS attacks can be integrated into APT (Advanced Persistent Threat) campaigns to: Distract security teams while executing a deeper intrusion.



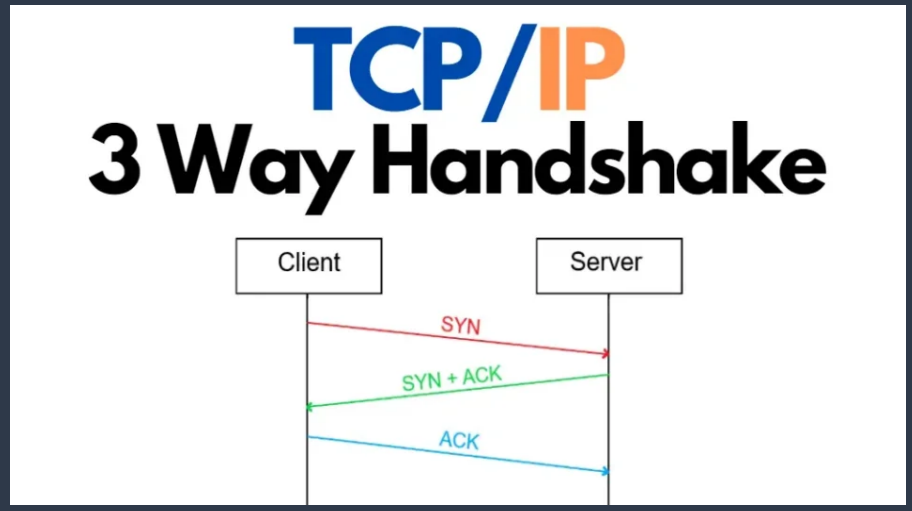


https://threats.wiz.io/defenses/ddos-protection

https://threats.wiz.io/all-actors

TCP Handshake Mechanism

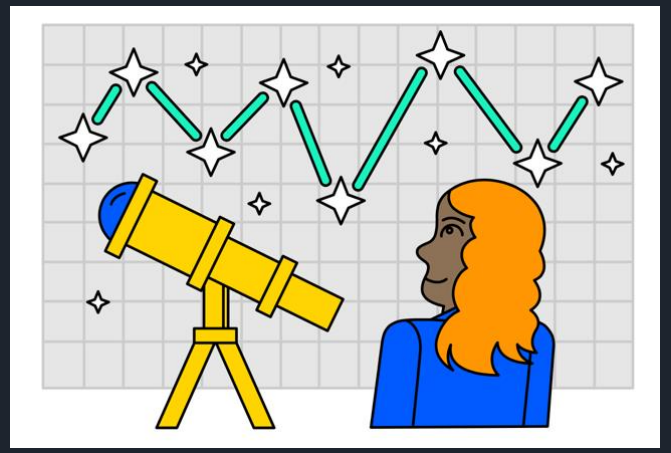
- **SYN Flood Attack:**
  - SYN Flood exploits the TCP handshake process
  - Attacker sends multiple SYN requests but never completes the handshake
  - Server resources remain allocated, leading to exhaustion



- **SYN Flood Attack Analysis-** Objective of this Report:

attack_type	Dst IP	CountSrc_uniq	DateTime
SYN_Flood	206.207.50.50	1	19/07/2019 15:15

  - Identify key attack characteristics
  - Propose possible mitigation strategies



ABOUT the CODE

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import scipy.stats as stats
```

```
df['group_DosDetect_id'] = df.groupby(['Dst IP', pd.Grouper(freq='5min')]).ngroup()

# - 'grouped_summary': Contains aggregate statistics (e.g., counts, averages) per Dst IP and time interval.
grouped_summary = df.groupby(['group_DosDetect_id', 'Dst IP', pd.Grouper(freq='5min')]).agg(
    # Aggregate summarized statistics per group
    CountRequests=('Src IP', 'count'),
    CountSrc_uniq=('Src IP', 'nunique'),  # Count unique source IPs
    Flow_Packets_s_avg=('Flow Packets/s', 'mean'),
    Flow_Bytes_s_avg=('Flow Bytes/s', 'mean'),
    SYN_count_sum=('SYN Flag Count', 'sum'),
    ACK_count_sum=('ACK Flag Count', 'sum'),
    Bwd_sum=('Total Bwd packets', 'sum'),
    Fwd_sum=('Total Fwd Packet', 'sum'),
).reset_index()

# Calculating ratios separately to avoid referencing within the aggregation function
grouped_summary['SYN_ACK_Ratio'] = grouped_summary['SYN_count_sum'] /((grouped_summary['ACK_count_sum']) + 1)
grouped_summary['ACK_SYN_Ratio'] = grouped_summary['ACK_count_sum'] / (grouped_summary['SYN_count_sum'] + 1)
grouped_summary['Bwd_Fwd_Ratio'] = grouped_summary['Bwd_sum'] / (grouped_summary['Fwd_sum'] + 1)
```

Data Processing & Aggregation

- **Data Grouping: DST and slats of 5 min**
- **Unique Key Assignment:** Added a key (group_DosDetect_id) for linking further aggregations of Src details.
- **Detected Attack:** Identified DoS attack - 3 if's
- FP/FN/TP/TN

Threshold

```python
def detect_outliers_iqr(df, column):
    # Normalized Outlier
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    iqr = Q3 - Q1
    lower_bound = Q1 - 1.5 * iqr
    upper_bound = Q3 + 1.5 * iqr

    outlier_dict = {
        "dict_name": "outlier_dict",
        # "df_pointer": point_to_df,
        "column": column,
        "lower_bound": lower_bound,
        "upper_bound": upper_bound,
        "iqr": iqr,
        "std": df[column].std()
    }
    return outlier_dict
```

Protocols

```python
# Global dictionary for protocol
mappings
PROTOCOL_MAPPING = {
    1: 'ICMP',  # Internet Control
Message Protocol
    6: 'TCP',  # Transmission Control
Protocol
    17: 'UDP',  # User Datagram
Protocol
    50: 'ESP',  # Encapsulating
Security Payload
    51: 'AH',  # Authentication Header
    8: 'EGP',  # Exterior Gateway
Protocol
    # Add other protocol mappings as
needed
}
```

Ports

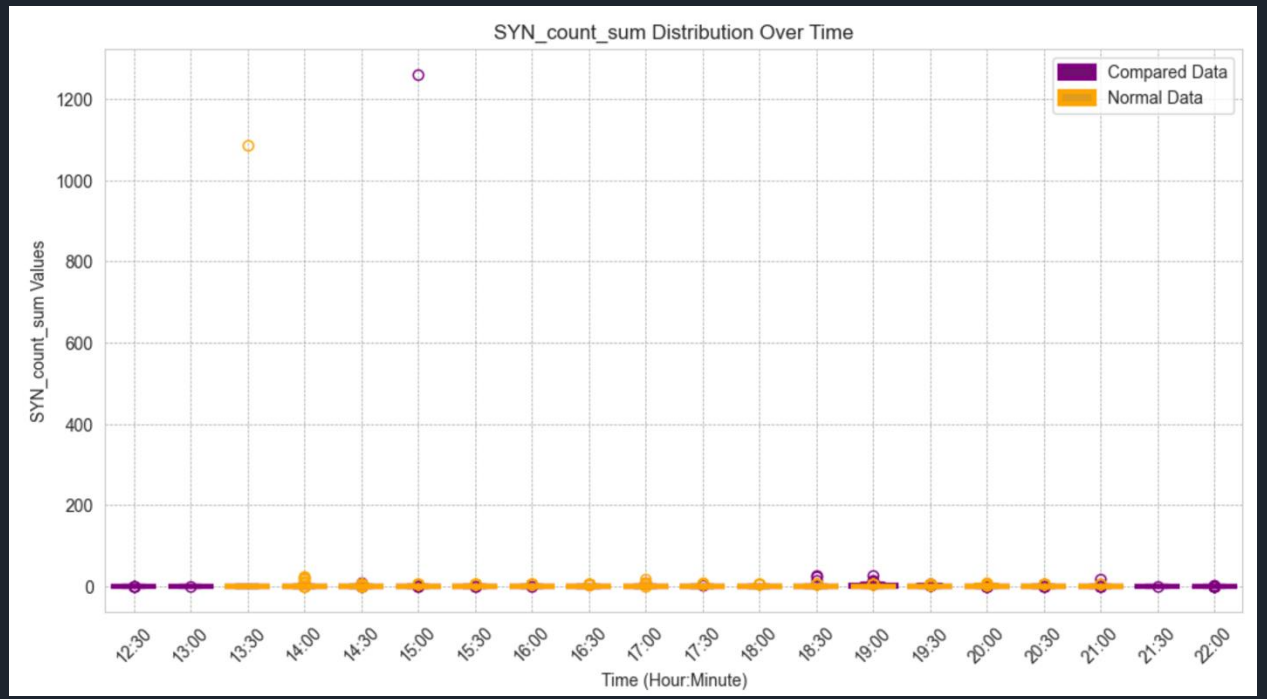
```python
def categorize_ports(port):
    """
    Categorize ports into Well-Known Ports,
Registered Ports, and Dynamic/Private Ports.

    - Well-Known Ports: 0-1023
    - Registered Ports: 1024-49151
    - Dynamic/Private Ports: 49152-65535
    """
    if port < 1024:
        return 'Well-Known Ports'
    elif 1024 <= port <= 49151:
        return 'Registered Ports'
    else:
        return 'Dynamic/Private Ports'



    # Define protocol categories
```

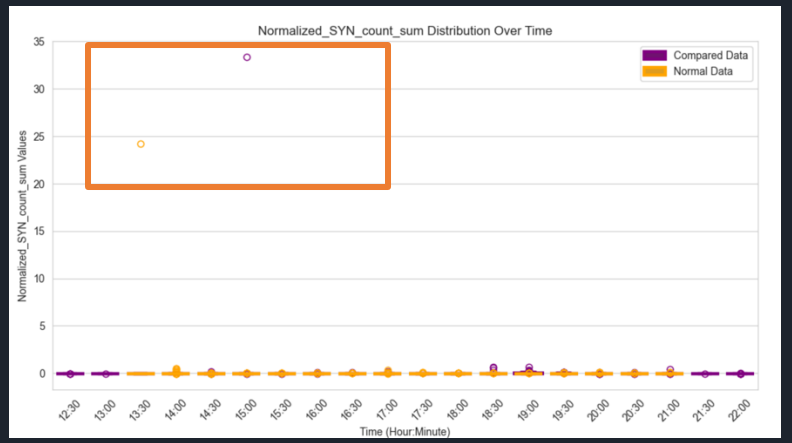
Dict of DoS Attacks

```
if attack_type != 'Normal':
    dict_dos_attacks[key] = {
        "Attack_id": f"{key}-{row.group_DosDetect_id}",
        "group_DosDetect_id": row.group_DosDetect_id,
        "Source_dfName": df_filename,
        "attack_type": attack_type,
        f"{column}": Ratio,
        "Dst IP": row['Dst IP'],
        "CountRequests": row['CountRequests'],
        "CountSrc_uniq": row['CountSrc_uniq'],  # Count unique source IPs
        "Datetime": row['Datetime'],
        "HourTime": row['HourTime'],
        # Calculate statistics for SYN Flag Count
        "SYN_count_sum": row.SYN_count_sum,
        # Calculate statistics for ACK Flag Count
        "ACK_count_sum": row.ACK_count_sum,
        # Calculate statistics for Fwd Packet Flag Count
        "Fwd_sum": row.Fwd_sum,
        # Calculate statistics for Bwd Packet Flag Count
        "Bwd_sum": row.Bwd_sum,
        # Avg Flow Packets/ Bytes
        "Flow_Packets_s_avg": row.Flow_Packets_s_avg,
        "Flow_Bytes_s_avg": row.Flow_Bytes_s_avg,
        # Source details
        "Src Details": row['Src Details'],
        "SrcIP_uniq": row['SrcIP_uniq'],
        "SrcPort_uniq": row.SrcPort_uniq,
        "SrcPort_categorical": row.SrcPort_categorical,
        "Protocol_uniq": row.Protocol_uniq,
        "Protocol_categorical": row.Protocol_categorical,
        # Port categorial count
        'Well_Known_Port_Count': row.Well_Known_Port_Count,
        'Registered_Port_Count': row.Registered_Port_Count,
        'Dynamic_Private_Port_Count': row.Dynamic_Private_Port_Count,
        # Protocol categorial count
        'Protocol_TCP_Count': row.Protocol_TCP_Count,
        'Protocol_UDP_Count': row.Protocol_UDP_Count,
        'Protocol_ICMP_Count': row.Protocol_TCP_Count,
        'Other_Protocol_Count': row.Other_Protocol_Count
                                                        }
```

SYN Count Over Time

- Key Observations:
  - Unusual spike detected in SYN packet count at 15:15
  - Compared with normal traffic data
  - SYN count above 1200 sum, significantly deviating from normal levels

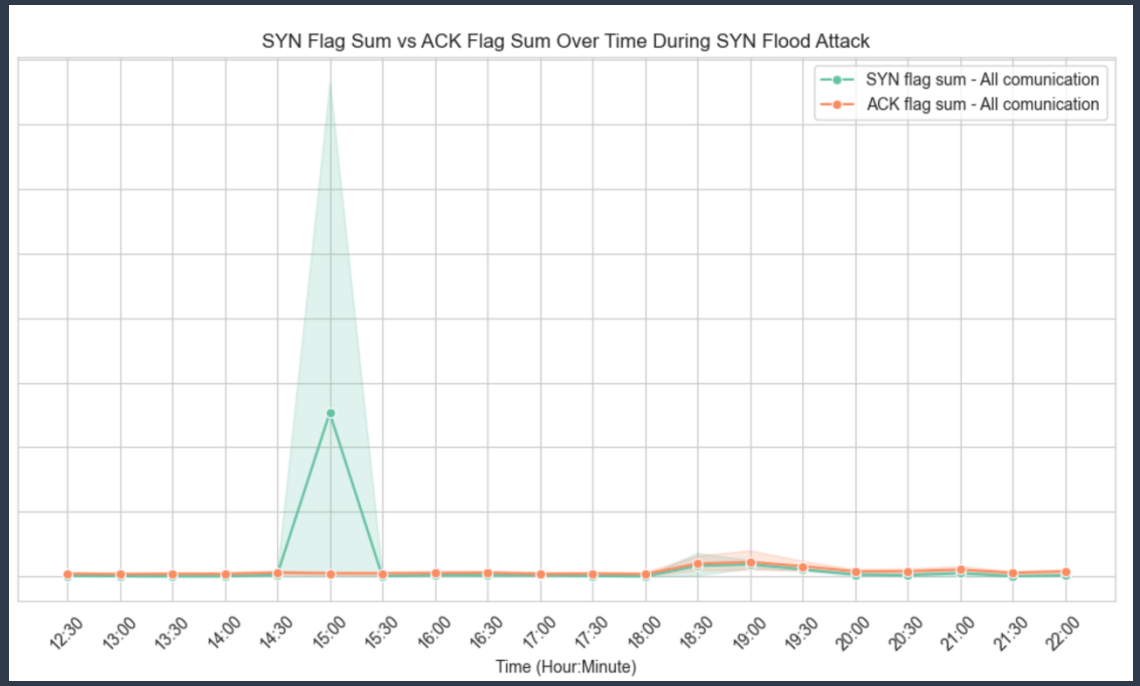


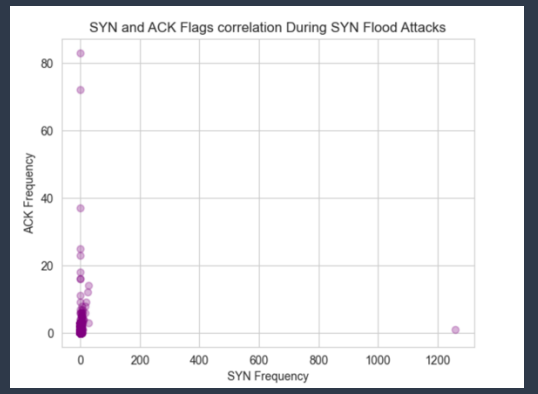


Anomalous SYN/ACK Ratio

- **Findings:**
  - High SYN to ACK ratio, indicating an incomplete handshake pattern
  - Normal traffic exhibits balanced SYN-ACK pairs
  - Attack traffic shows unmatched SYN packets

CountRequests	Syn_count_sum	ACK_count_sum	SYN_ACK_Ratio
1261	1260	1	630





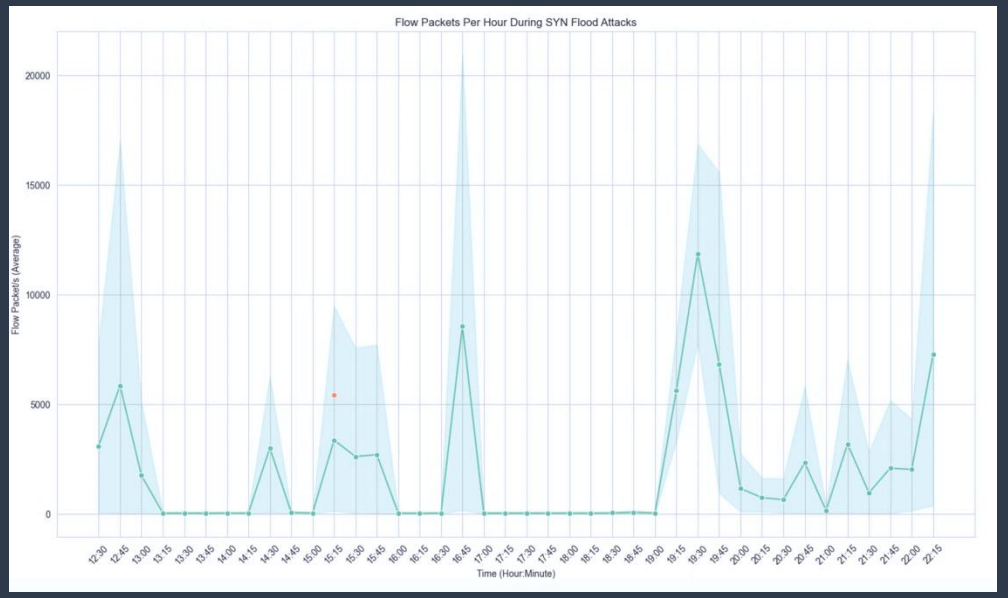
Anomalous Packets/Bytes (Payload)

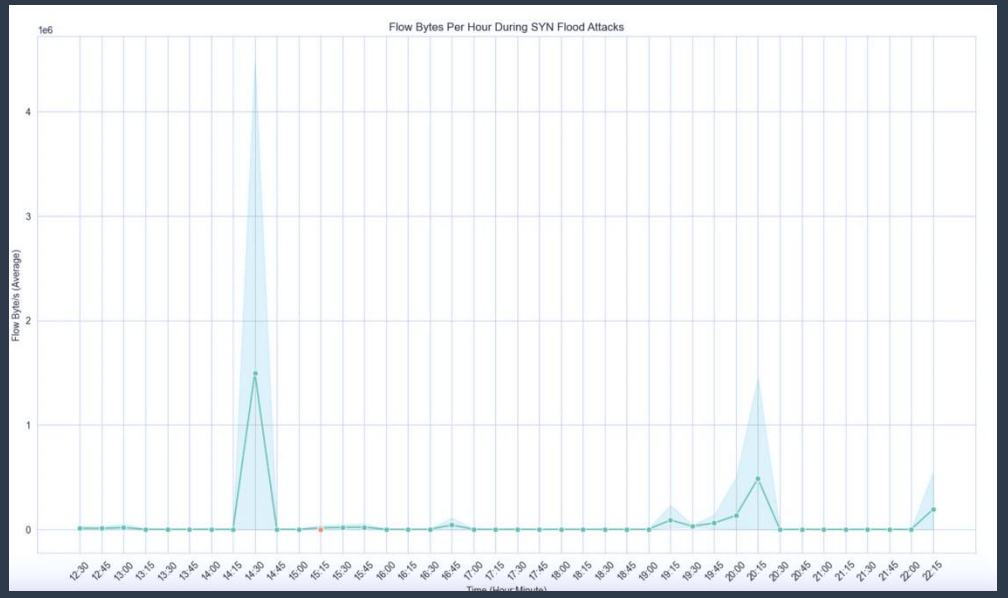
- **Findings:**
  - High Packet to Byte flow, indicating a lot of Syn Packets but no Payloa

Flow Packets	Flow Bytes
5411	0

A pre-attack spike
may indicate reconnaissance or probing activity.

Decay Pattern
Network Anomaly Detection by Using a Time-Decay Closed Frequent Patterns

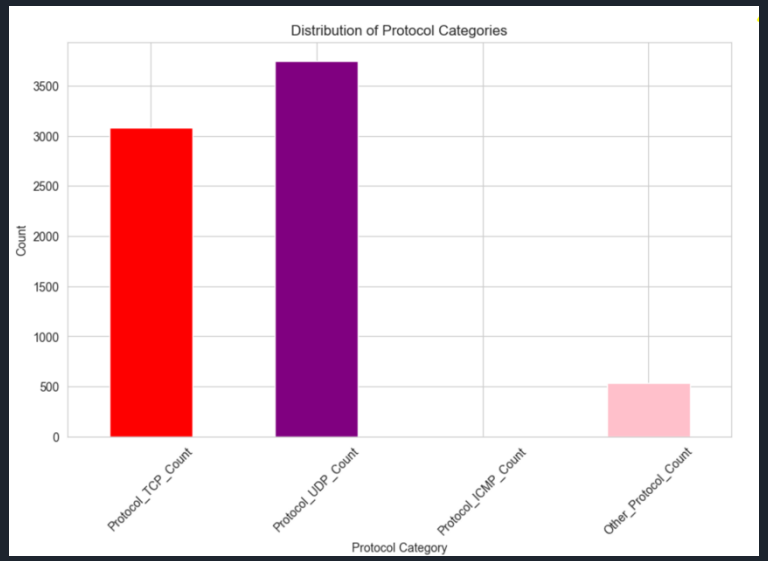


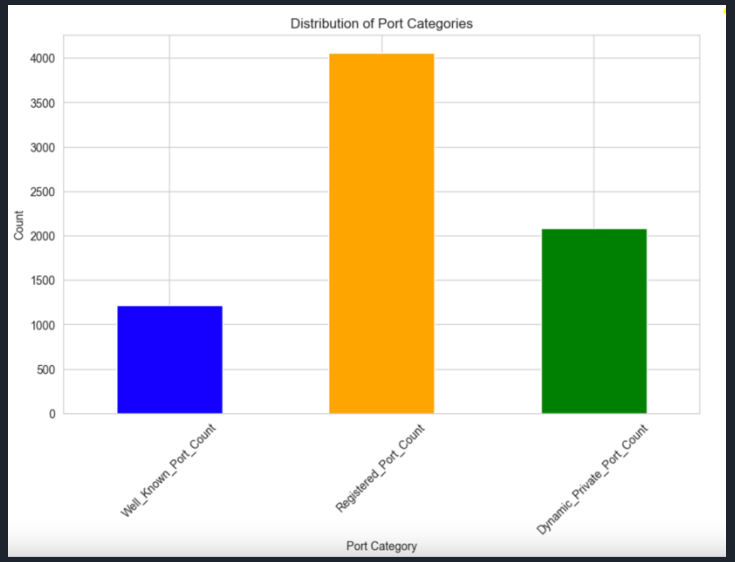


Comparison with Normal Traffic:

- Mean, std, Quarters, and max: Bytes > Packets
- Attack scenario shows sudden bursts followed by system slowdowns
- **Ports and Protocols**

Normal Data





Normal Data

	Flow_Packets_s_avg	Flow_Bytes_s_avg
count	589	589
mean	1,625.344004	45,330.44
std	6,368.88894	742,403.20
min	0.025	0.00
25%	0.198407	0.00
50%	0.747536	36.68
75%	2.207692	187.92
max	131,166.0729	14,484,450.00

Compared Data

Protocol_TCP Count	Protocol_UDP Count	Protocol_ICMP Count	Other_Protocol Count
1261	0	1261	0

Compared Data

Well_Known_Port Count	Registered_Port Count	Dynamic_Private_Port Count
1	1261	0

Compared Data

CountRequests	Syn_count_sum	ACK_count_sum	SYN_ACK_Ratio	Flow_Packets_s_avg	Flow_Bytes_s_avg	Protocol	Src Port
1261	**1260**	**1**	**630**	**5411.156741**	**0**	**TCP, Other**	**[Registered Ports, Well-Known Ports]**

Conclusions and what's next

Project Objectives:

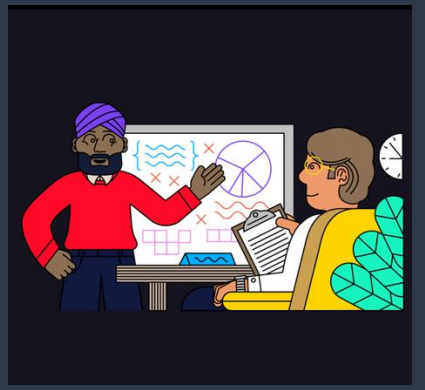
- Detect network anomalies and potential threats.
- Identify cyberattack behavioral patterns.
- Extract insights and correlations for threat intelligence.
- Improve detection accuracy using :
  - customer
  - AI services.

Key Takeaways:

- SYN Flood exploits the TCP handshake process.
- Analysis detected significant anomalies in SYN packet behavior.
- Proactive security measures are essential for network resilience.

Future Work:

- Improving response strategies for customer service and incident handling.
- Enhancing detection using AI-based anomaly detection techniques and unsupervized Data Science methods.



Bibliografy

1. https://www.splunk.com/en_us/blog/learn/ttp-tactics-techniques-procedures.html
2. https://purplesec.us/learn/prevent-syn-flood-attack/
3. https://www.paloaltonetworks.com/blog/security-operations/6-questions-you-must-ask-for-a-successful-incident-response/
4. . Microsoft SecurityDocumentation
5. Documents from BSc Information Systems and Cybersecurity
6. https://www.codecademy.com/article/visualizing-time-series-data-with-python
7. https://www.netscout.com/what-is-ddos/ssl-tls-exhaustion
8. https://www.mdpi.com/2078-2489/10/8/262?utm_source=chatgpt.com
9. https://www.konfidas.com/cybersecurity-briefs/cybersecuritybrief05012023
10. https://developers.cloudflare.com/fundamentals/basic-tasks/protect-your-origin-server/
11. OWASP - DDoS Prevention Cheat Sheet OWASP
12. Palo Alto Networks - Understanding DDoS and Proxy Attacks Palo Alto Networks
13. https://purplesec.us/learn/prevent-syn-flood-attack/
14. https://www.kaggle.com/datasets/sowmyamyneni/dapt2020