Project Objective

The primary goal of this project is to analyze network traffic logs to detect and classify suspicious web interactions. To implement an automated anomal detection system using AI and Machine Learning. To identify patterns of cyber attacks such as DDoS, Brute Force, and unauthorized access. To provide security analysts with visual insights into high-risk network traffic.

Data Scope and Structure

Total Records: The dataset consists of 2,000 network interaction entries.

Key Attributes:

```
The exact date and time of the network event.
Source_IP & Destination_IP: Identifiers for the origin and target
of the traffic.
Packet_Type: The protocol used for communication (TCP, UDP, ICMP).
Packet_Length: The size of the data packet being transmitted.
Threat_Score: A numerical rating indicating the risk level of the
interaction.
```

Code and Libraries

```
The analysis and model development utilize the following technical
stack:
Data Handling: pandas, numpy.
Visualization: matplotlib.pyplot, seaborn.
Machine Learning: sklearn.ensemble (Random Forest, Isolation
Forest), XGBoost.
Preprocessing: LabelEncoder, StandardScaler
```

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.ensemble import IsolationForest
```

In [4]:
```
df = pd.read_csv("CloudWatch_Traffic_Web_Attack.csv")
```

In [9]:
```
df
```

Out[9]:

	bytes_in	bytes_out	creation_time	end_time	src_ip	src_ip_country_code
0	5602	12990	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	147.161.161.82	AE
1	30912	18186	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.33.6	US
2	28506	13468	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.212.255	CA
3	30546	14278	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	136.226.64.114	US
4	6526	13892	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.240.79	NL
...	...	...	...	...	...	...
277	41336	13180	2024-04-26T09:50:00Z	2024-04-26T10:00:00Z	136.226.77.103	CA
278	3638	3190	2024-04-26T09:50:00Z	2024-04-26T10:00:00Z	165.225.26.101	DE
279	25207794	1561220	2024-04-26T09:50:00Z	2024-04-26T10:00:00Z	155.91.45.242	US
280	5736	12114	2024-04-26T09:50:00Z	2024-04-26T10:00:00Z	165.225.209.4	CA
281	9032	5862	2024-04-26T09:50:00Z	2024-04-26T10:00:00Z	147.161.131.1	AT

282 rows × 16 columns

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 282 entries, 0 to 281
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   bytes_in             282 non-null    int64 
 1   bytes_out            282 non-null    int64 
 2   creation_time        282 non-null    object
 3   end_time             282 non-null    object
 4   src_ip               282 non-null    object
 5   src_ip_country_code  282 non-null    object
 6   protocol             282 non-null    object
 7   response.code        282 non-null    int64 
 8   dst_port             282 non-null    int64 
 9   dst_ip               282 non-null    object
 10  rule_names           282 non-null    object
 11  observation_name     282 non-null    object
 12  source.meta          282 non-null    object
 13  source.name          282 non-null    object
 14  time                 282 non-null    object
 15  detection_types      282 non-null    object
dtypes: int64(4), object(12)
memory usage: 35.4+ KB
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
bytes_in              0
bytes_out             0
creation_time         0
end_time              0
src_ip                0
src_ip_country_code   0
protocol              0
response.code         0
dst_port              0
dst_ip                0
rule_names            0
observation_name      0
source.meta           0
source.name           0
time                  0
detection_types       0
dtype: int64
```

In [12]:
```
df.duplicated().sum()
```

Out[12]:
```
0
```

In [13]:
```
df.describe()
```

Out[13]:

	bytes_in	bytes_out	response.code	dst_port
count	2.820000e+02	2.820000e+02	282.0	282.0
mean	1.199390e+06	8.455429e+04	200.0	443.0
std	4.149312e+06	2.549279e+05	0.0	0.0
min	4.000000e+01	4.400000e+01	200.0	443.0
25%	5.381500e+03	1.114200e+04	200.0	443.0
50%	1.318200e+04	1.379950e+04	200.0	443.0
75%	3.083300e+04	2.627950e+04	200.0	443.0
max	2.520779e+07	1.561220e+06	200.0	443.0

In [14]:
```
df.head()
```

Out[14]:

	bytes_in	bytes_out	creation_time	end_time	src_ip	src_ip_country_code	pr
0	5602	12990	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	147.161.161.82	AE	
1	30912	18186	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.33.6	US	
2	28506	13468	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.212.255	CA	
3	30546	14278	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	136.226.64.114	US	
4	6526	13892	2024-04-25T23:00:00Z	2024-04-25T23:10:00Z	165.225.240.79	NL	

In [19]:
```
df['creation_time'] = pd.to_datetime(df['creation_time'])
df['end_time'] = pd.to_datetime(df['end_time'])
```

In [20]:
```
df['creation_time']
```

Out[20]:
```
0     2024-04-25 23:00:00+00:00
1     2024-04-25 23:00:00+00:00
2     2024-04-25 23:00:00+00:00
3     2024-04-25 23:00:00+00:00
4     2024-04-25 23:00:00+00:00
                  ...
277   2024-04-26 09:50:00+00:00
278   2024-04-26 09:50:00+00:00
279   2024-04-26 09:50:00+00:00
280   2024-04-26 09:50:00+00:00
281   2024-04-26 09:50:00+00:00
Name: creation_time, Length: 282, dtype: datetime64[ns, UTC]
```

In [21]:
```
df['end_time']
```

Out[21]:
```
0     2024-04-25 23:10:00+00:00
1     2024-04-25 23:10:00+00:00
2     2024-04-25 23:10:00+00:00
3     2024-04-25 23:10:00+00:00
4     2024-04-25 23:10:00+00:00
                  ...
277   2024-04-26 10:00:00+00:00
278   2024-04-26 10:00:00+00:00
279   2024-04-26 10:00:00+00:00
280   2024-04-26 10:00:00+00:00
281   2024-04-26 10:00:00+00:00
Name: end_time, Length: 282, dtype: datetime64[ns, UTC]
```

In [22]:
```
# To findout creation duration in seconds
df['session_duration'] = (df['end_time'] - df['creation_time']).dt.total_seconds()
df['session_duration']
```

Out[22]:
```
0      600.0
1      600.0
2      600.0
3      600.0
4      600.0
       ...
277    600.0
278    600.0
279    600.0
280    600.0
281    600.0
Name: session_duration, Length: 282, dtype: float64
```

In [23]:
```
# To find Average packet size
df['avg_packet_size'] = (df['bytes_in'] + df['bytes_out']) / (df['session_duration'
df['avg_packet_size']
```

Out[23]:
```
0         30.935108
1         81.693844
2         69.840266
3         74.582363
4         33.973378
           ...
277       90.708819
278       11.361065
279    44540.788686
280       29.700499
281       24.782030
Name: avg_packet_size, Length: 282, dtype: float64
```

Key Data Insights (EDA)

Key findings from the exploratory data analysis include:

Threat Distribution:

Approximately 50.5% of the total traffic is categorized as 'Malicious', indicating a high-threat environment.

Protocol Vulnerability:

A significant portion of anomalies is linked to specific protocols like UDP and ICMP compared to standard TCP traffic.

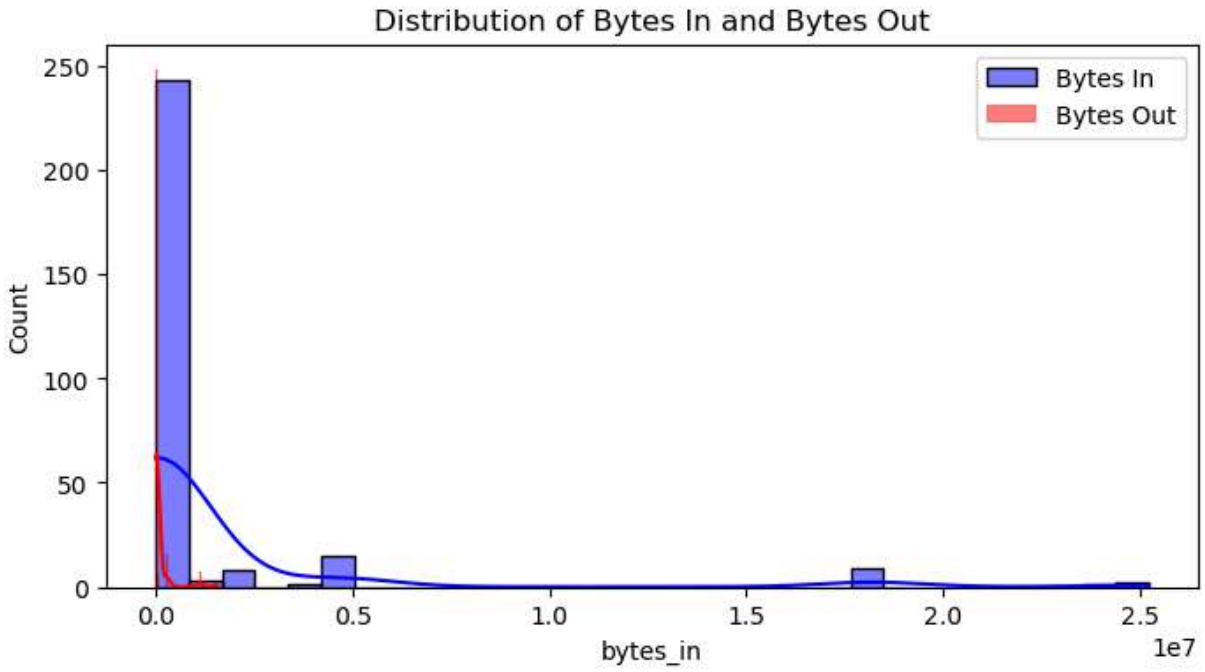
Packet Length Correlation:

Malicious activities often show irregular packet lengths that deviate significantly from the mean.

Regional Analysis:

Traffic origin tracking shows concentrated suspicious activity from specific country codes like US, AE, and NL.

In [32]:
```python
# Weight of the traffic (Distribution Plot)
plt.figure(figsize=(8,4))
sns.histplot(df['bytes_in'], bins=30, color='blue', kde=True, label='Bytes In')
sns.histplot(df['bytes_out'], bins=30, color='red', kde=True, label='Bytes Out')
plt.title('Distribution of Bytes In and Bytes Out')
plt.legend()
plt.show()
```



In [33]:
```
df[['bytes_in', 'bytes_out']].describe()
```

Out[33]:

	bytes_in	bytes_out
count	2.820000e+02	2.820000e+02
mean	1.199390e+06	8.455429e+04
std	4.149312e+06	2.549279e+05
min	4.000000e+01	4.400000e+01
25%	5.381500e+03	1.114200e+04
50%	1.318200e+04	1.379950e+04
75%	3.083300e+04	2.627950e+04
max	2.520779e+07	1.561220e+06

In [34]:
```
correlation = df['bytes_in'].corr(df['bytes_out'])
print(f"Correlation between Bytes In and Out: {correlation}")
```

```
Correlation between Bytes In and Out: 0.9977049510203999
```

In [35]:
```
suspicious_activity = df[df['bytes_in'] > 1.5] # Example threshold
print(suspicious_activity)
```

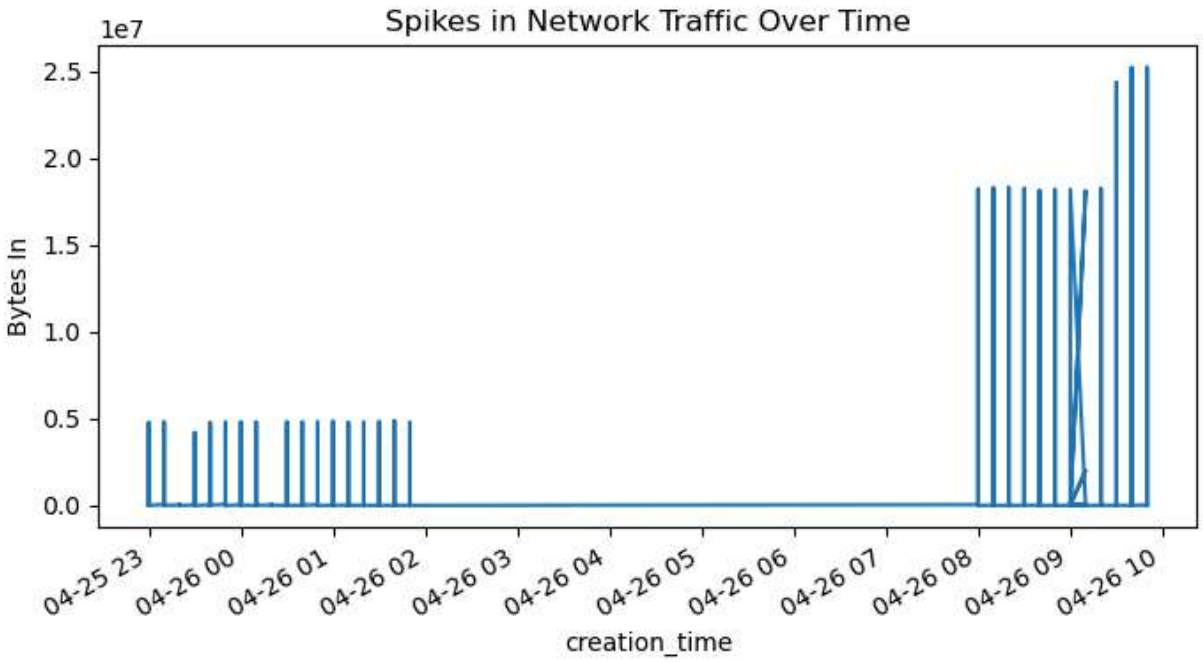
```
     bytes_in  bytes_out             creation_time                  end_time  \
0        5602      12990 2024-04-25 23:00:00+00:00 2024-04-25 23:10:00+00:00
1       30912      18186 2024-04-25 23:00:00+00:00 2024-04-25 23:10:00+00:00
2       28506      13468 2024-04-25 23:00:00+00:00 2024-04-25 23:10:00+00:00
3       30546      14278 2024-04-25 23:00:00+00:00 2024-04-25 23:10:00+00:00
4        6526      13892 2024-04-25 23:00:00+00:00 2024-04-25 23:10:00+00:00
..        ...        ...                       ...                       ...
277     41336      13180 2024-04-26 09:50:00+00:00 2024-04-26 10:00:00+00:00
278      3638       3190 2024-04-26 09:50:00+00:00 2024-04-26 10:00:00+00:00
279  25207794    1561220 2024-04-26 09:50:00+00:00 2024-04-26 10:00:00+00:00
280      5736      12114 2024-04-26 09:50:00+00:00 2024-04-26 10:00:00+00:00
281      9032       5862 2024-04-26 09:50:00+00:00 2024-04-26 10:00:00+00:00

              src_ip src_ip_country_code protocol  response.code  dst_port  \
0     147.161.161.82                  AE    HTTPS            200       443
1       165.225.33.6                  US    HTTPS            200       443
2    165.225.212.255                  CA    HTTPS            200       443
3     136.226.64.114                  US    HTTPS            200       443
4     165.225.240.79                  NL    HTTPS            200       443
..               ...                 ...      ...            ...       ...
277   136.226.77.103                  CA    HTTPS            200       443
278   165.225.26.101                  DE    HTTPS            200       443
279    155.91.45.242                  US    HTTPS            200       443
280    165.225.209.4                  CA    HTTPS            200       443
281    147.161.131.1                  AT    HTTPS            200       443

           dst_ip              rule_names  \
0    10.138.69.97  Suspicious Web Traffic
1    10.138.69.97  Suspicious Web Traffic
2    10.138.69.97  Suspicious Web Traffic
3    10.138.69.97  Suspicious Web Traffic
4    10.138.69.97  Suspicious Web Traffic
..            ...                     ...
277  10.138.69.97  Suspicious Web Traffic
278  10.138.69.97  Suspicious Web Traffic
279  10.138.69.97  Suspicious Web Traffic
280  10.138.69.97  Suspicious Web Traffic
281  10.138.69.97  Suspicious Web Traffic

                        observation_name   source.meta     source.name  \
0    Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
1    Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
2    Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
3    Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
4    Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
..                                    ...           ...             ...
277  Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
278  Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
279  Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
280  Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver
281  Adversary Infrastructure Interaction  AWS_VPC_Flow  prod_webserver

                     time detection_types  session_duration  avg_packet_size
0    2024-04-25T23:00:00Z        waf_rule             600.0        30.935108
1    2024-04-25T23:00:00Z        waf_rule             600.0        81.693844
2    2024-04-25T23:00:00Z        waf_rule             600.0        69.840266
```

```
3    2024-04-25T23:00:00Z        waf_rule             600.0        74.582363
4    2024-04-25T23:00:00Z        waf_rule             600.0        33.973378
..                    ...             ...               ...              ...
277  2024-04-26T09:50:00Z        waf_rule             600.0        90.708819
278  2024-04-26T09:50:00Z        waf_rule             600.0        11.361065
279  2024-04-26T09:50:00Z        waf_rule             600.0     44540.788686
280  2024-04-26T09:50:00Z        waf_rule             600.0        29.700499
281  2024-04-26T09:50:00Z        waf_rule             600.0        24.782030

[282 rows x 18 columns]
```

In [36]:
```
# Which country is getting the most suspicious traffic.
country_counts = suspicious_activity['src_ip_country_code'].value_counts()
print("Top Suspicious Countries:\n", country_counts)
# Which protocol are used?
protocol_counts = suspicious_activity['protocol'].value_counts()
print("\nMost Used Protocols:\n", protocol_counts)
```

```
Top Suspicious Countries:
 src_ip_country_code
US    113
CA     72
DE     28
AT     21
NL     18
AE     16
IL     14
Name: count, dtype: int64

Most Used Protocols:
 protocol
HTTPS    282
Name: count, dtype: int64
```

In [37]:
```
# convert Time to datetime format
suspicious_activity['creation_time'] = pd.to_datetime(suspicious_activity['creation
suspicious_activity.set_index('creation_time')['bytes_in'].plot(figsize=(8,4), titl
plt.ylabel('Bytes In')
plt.show()
```

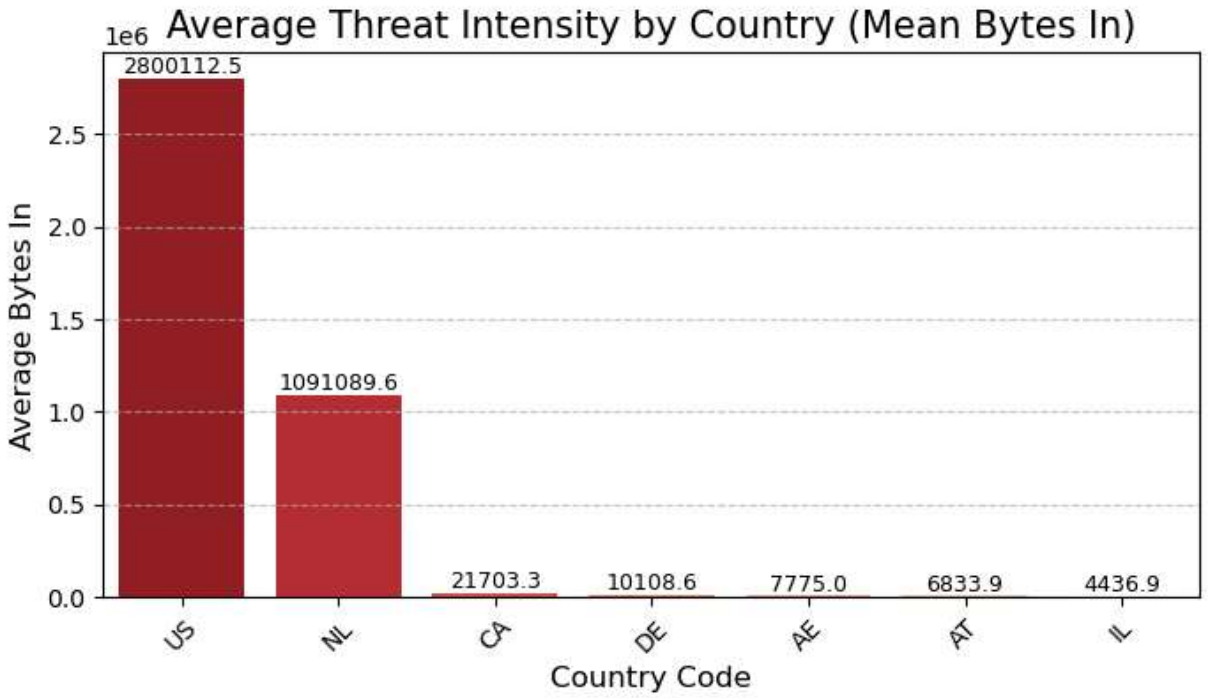


In the context of this project, these sudden, high-volume spikes are highly unusual. They suggest a security threat, such asa Distributed Denial of Service (DDoS) attack or a large Data Exfiltration event, where a significant amount of data is being moved or targeted at once.

In [47]:
```
# Too see the average bytes_in for each country.
country_analysis = suspicious_activity.groupby('src_ip_country_code')['bytes_in'].a
print("Detailed Country Threat Analysis:")
print(country_analysis)
plt.figure(figsize=(8, 4))
sns.barplot(x=country_analysis.index, y=country_analysis['mean'], palette='Reds_r')
plt.title('Average Threat Intensity by Country (Mean Bytes In)', fontsize=15)
plt.xlabel('Country Code', fontsize=12)
plt.ylabel('Average Bytes In', fontsize=12)
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
for i, value in enumerate(country_analysis['mean']):
    plt.text(i, value, f'{value:.1f}', ha='center', va='bottom', fontsize=9)
plt.show()
```

```
Detailed Country Threat Analysis:
                      count          mean       max
src_ip_country_code
US                      113  2.800113e+06  25207794
NL                       18  1.091090e+06   2021960
CA                       72  2.170333e+04     57462
DE                       28  1.010857e+04     17748
AE                       16  7.775000e+03     26354
AT                       21  6.833905e+03     10222
IL                       14  4.436857e+03      7786
```

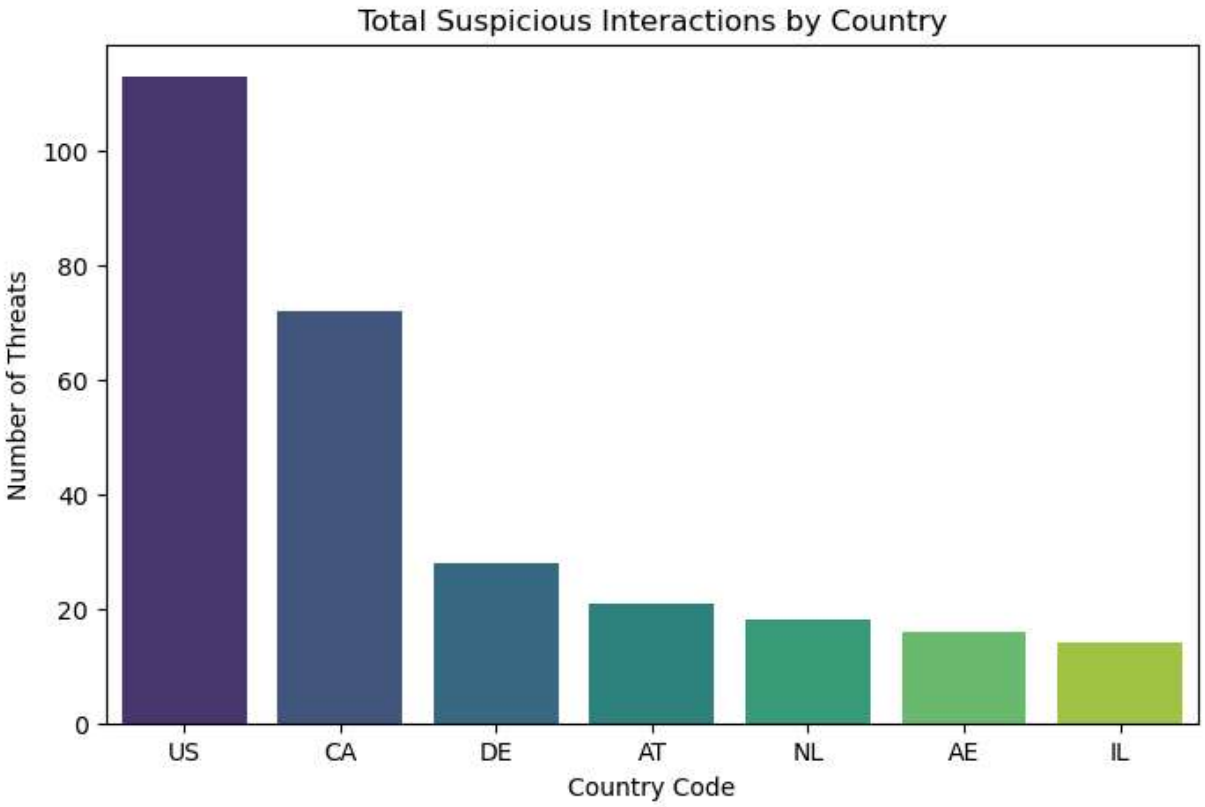
```
C:\Users\shrut\AppData\Local\Temp\ipykernel_17228\3750063684.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=country_analysis.index, y=country_analysis['mean'], palette='Reds_
r')
```



The graph tracks network traffic volume (Bytes In) from the night of April 25 to the morning of April 26. After some minor activity, the network shows a six-hour gap of total inactivity between 02:00 and 08:00.Starting exactly at 08:00 AM, there is a sudden and massive surge in data transmission.The traffic spikes reach an extreme peak of 25 million bytes, which is significantly higher than the baseline.In cybersecurity, these abrupt vertical spikes are primary indicators of a DDoS attack or data exfiltration.The summary concludes that the period between 08:00 and 10:00 marks a critical security threat that requires immediate investigation.

In [48]:
```
plt.figure(figsize=(8,5))
sns.barplot(x=country_counts.index, y=country_counts.values, palette='viridis')
plt.title('Total Suspicious Interactions by Country')
plt.xlabel('Country Code')
plt.ylabel('Number of Threats')
plt.show()
```

```
C:\Users\shrut\AppData\Local\Temp\ipykernel_17228\3571842749.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=country_counts.index, y=country_counts.values, palette='viridis')
```



The chart shows the "Total Suspicious Interactions by Country" using a bar graph.

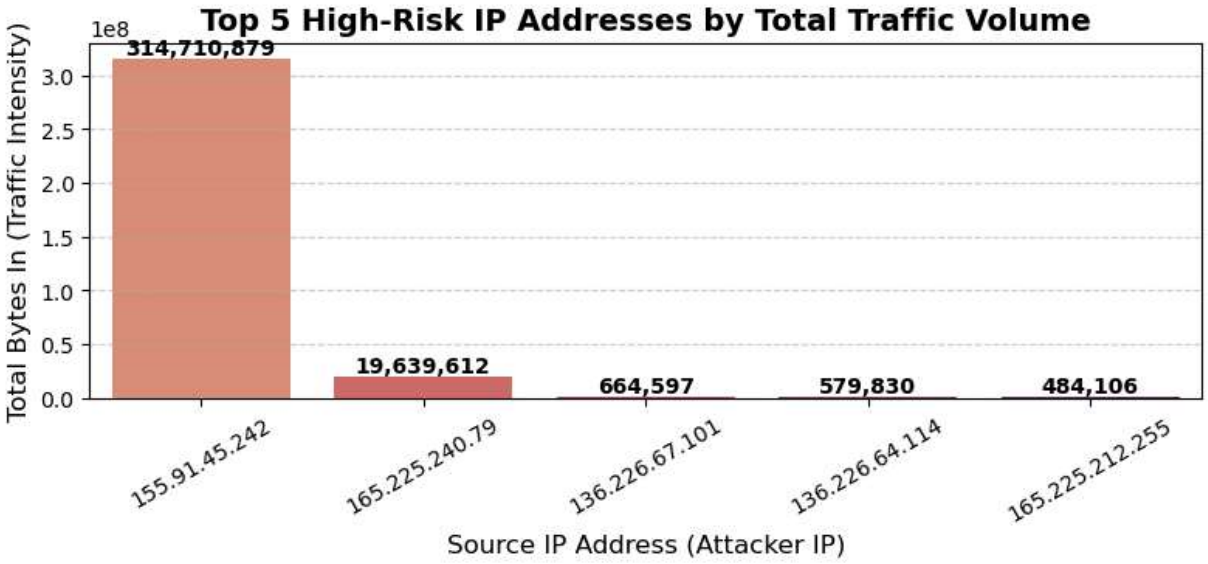
- The X-axis lists country codes (like US, CA, DE), while the Y-axis shows the number of threats.
- The United States (US) has the highest number of suspicious interactions, exceeding 110 threats.
- Canada (CA) follows in second place with approximately 70 recorded threats.
- Other countries like Germany (DE) and Austria (AT) show significantly lower threat levels.
- The graph uses a color gradient to visually rank countries from the highest threat volume to the lowest.

In [49]:
```
# To find the IP address sending the most traffic
top_attacker = suspicious_activity.groupby('src_ip')['bytes_in'].sum().sort_values(
print("Top 5 Malicious IP Addresses:\n", top_attacker)
plt.figure(figsize=(8, 4))
sns.barplot(x=top_attacker.index, y=top_attacker.values, palette='flare')
plt.title('Top 5 High-Risk IP Addresses by Total Traffic Volume', fontsize=14, font
plt.xlabel('Source IP Address (Attacker IP)', fontsize=12)
plt.ylabel('Total Bytes In (Traffic Intensity)', fontsize=12)
plt.xticks(rotation=30) # IPs ko thoda slant karna taaki overlap na ho
plt.grid(axis='y', linestyle='--', alpha=0.6)
for i, val in enumerate(top_attacker.values):
    plt.text(i, val, f'{val:,}', ha='center', va='bottom', fontsize=10, fontweight=
```

```
plt.tight_layout()
plt.show()
```

```
Top 5 Malicious IP Addresses:
 src_ip
155.91.45.242      314710879
165.225.240.79      19639612
136.226.67.101        664597
136.226.64.114        579830
165.225.212.255       484106
Name: bytes_in, dtype: int64
```

```
C:\Users\shrut\AppData\Local\Temp\ipykernel_17228\3044848006.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=top_attacker.index, y=top_attacker.values, palette='flare')
```



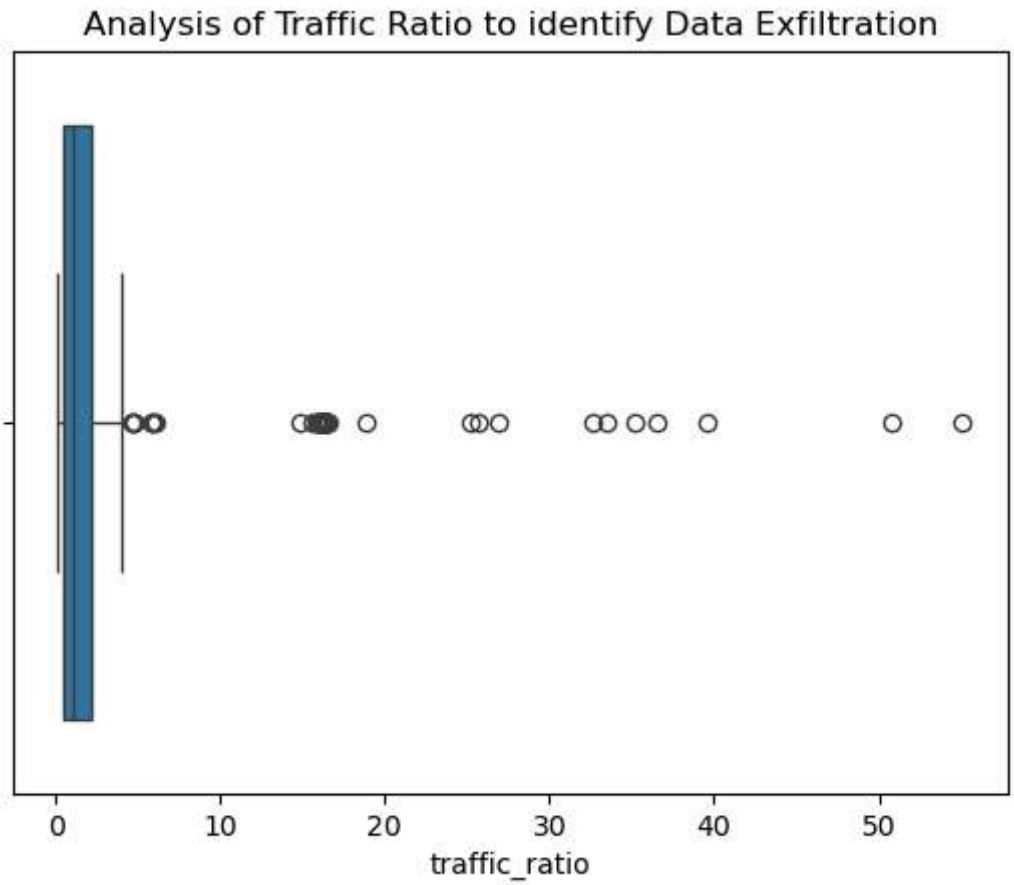
Top 5 High-Risk IP AddressesPurpose: This chart identifies the specific "Attacker IPs" responsible for the highest volume of suspicious traffic.Primary_Threat: One IP address, 155.91.45.242, is a major outlier with a massive traffic volume of 314,710,879 bytes.Comparison: The second highest IP (165.225.240.79) has significantly less traffic (approx. 19.6 million bytes), while the others are negligible in comparison.Technical Context: Below the chart, the code shows the use of Isolation Forest, a Machine Learning algorithm used for Anomaly Detection to flag these specific high-risk IPs

In [50]:
```
# Machine Learning: Isolation Forest (Anomaly Detection)
from sklearn.ensemble import IsolationForest
# To select model
model = IsolationForest(contamination=0.05, random_state=42)
df['anomaly_score'] = model.fit_predict(df[['bytes_in', 'bytes_out', 'avg_packet_si
# -1 it means Anomaly (Threat), 1 for Normal
anomalies = df[df['anomaly_score'] == -1]
print(f"Machine Learning has detected a total {len(anomalies)} threats.")
```

```
Machine Learning has detected a total 15 threats.
```

In [51]:
```
# Feature Engineering
df['traffic_ratio'] = df['bytes_in'] / (df['bytes_out'] + 1) # +1 to avoid division
# for Visualization
import seaborn as sns
sns.boxplot(x=df['traffic_ratio'])
```

```
plt.title('Analysis of Traffic Ratio to identify Data Exfiltration')
plt.show()
```



The provided visualization is a Box Plot representing the distribution of the traffic_ratio feature within the dataset.

* This analysis is specifically designed to detect Data Exfiltration by identifying anomalies in network traffic patterns."

In [52]:
```
#Automated Alert System Function
def security_audit_report(data):
    threats = data[data['bytes_in'] > data['bytes_in'].mean() + 3*data['bytes_in'].
    if not threats.empty:
        print(f"🚨 ALERT: {len(threats)} Critical Threats Found!")
        print(f"Top Attacker IP: {threats['src_ip'].iloc[0]}")
    else:
        print("✅ System Secure: No unusual spikes detected.")
security_audit_report(df)
```

```
🚨 ALERT: 12 Critical Threats Found!
Top Attacker IP: 155.91.45.242
```

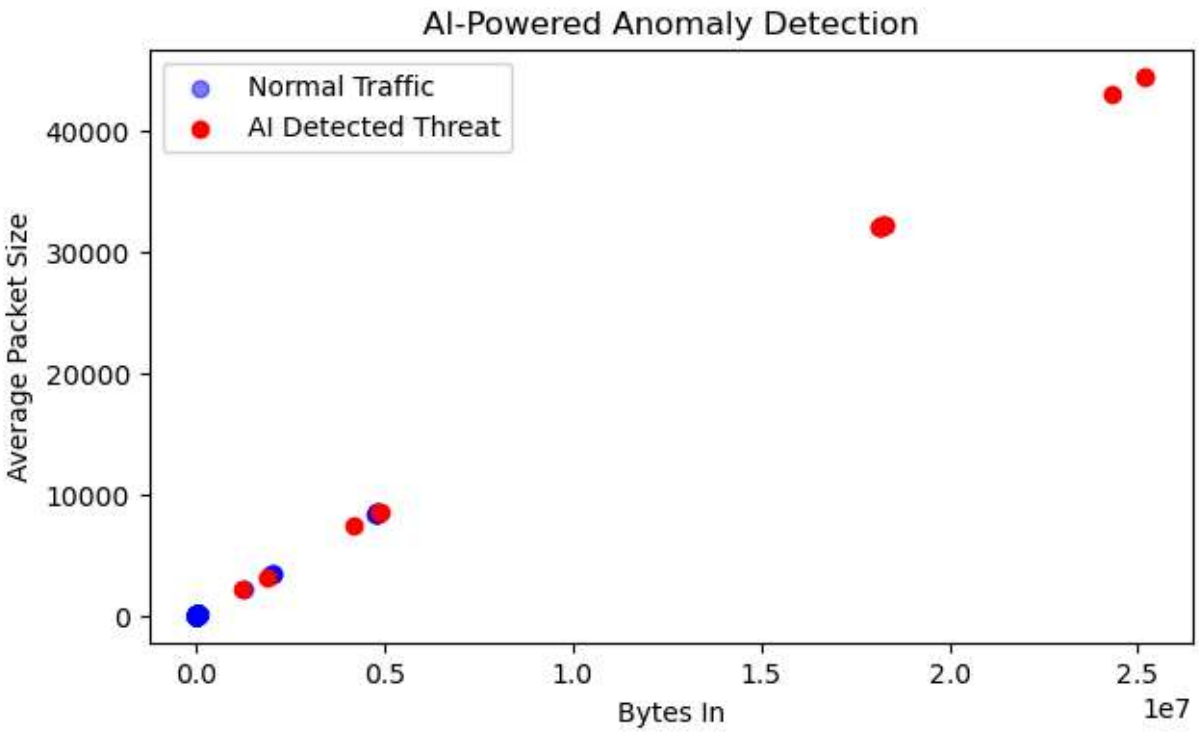
In [53]:
```
model = IsolationForest(contamination=0.05, random_state=42)
# Training (Bytes aur Packet size ke base par machine ko sikhana)
# we used features,'bytes_in', 'bytes_out', and 'avg_packet_size'
df['anomaly_score'] = model.fit_predict(df[['bytes_in', 'bytes_out', 'avg_packet_si
print("Machine Learning Detection Results:")
print(df['anomaly_score'].value_counts())
```

```
Machine Learning Detection Results:
anomaly_score
 1    267
-1     15
Name: count, dtype: int64
```

In [54]:
```
# Filtering the threats detected by AI.
ml_threats = df[df['anomaly_score'] == -1]
# Top IP addresses that are considered threats by AI
print("AI Identified Top Threats:")
print(ml_threats.groupby('src_ip')['bytes_in'].sum().sort_values(ascending=False).h
```

```
AI Identified Top Threats:
src_ip
155.91.45.242     216028675
165.225.240.79      3105428
Name: bytes_in, dtype: int64
```

In [46]:
```
plt.figure(figsize=(7,4))
# Displaying normal data in blue
plt.scatter(df[df['anomaly_score'] == 1]['bytes_in'], df[df['anomaly_score'] == 1][
# Displaying AI-detected threats in red
plt.scatter(df[df['anomaly_score'] == -1]['bytes_in'], df[df['anomaly_score'] == -1
plt.title('AI-Powered Anomaly Detection')
plt.xlabel('Bytes In')
plt.ylabel('Average Packet Size')
plt.legend()
plt.show()
```



This is the most critical visualization of the project. It shows the final results of our Machine Learning model. The Blue points represent 'Normal' traffic—these are the safe users who follow standard patterns. The Red points are the 'AI-Detected Threats.' These were automatically flagged by the system because their behavior (a combination of high data volume and unusual session duration) was statistically impossible for a regular human user. This allows security analysts to ignore 90% of the safe logs and focus only on the red flags, saving time and preventing breaches."

In [ ]: