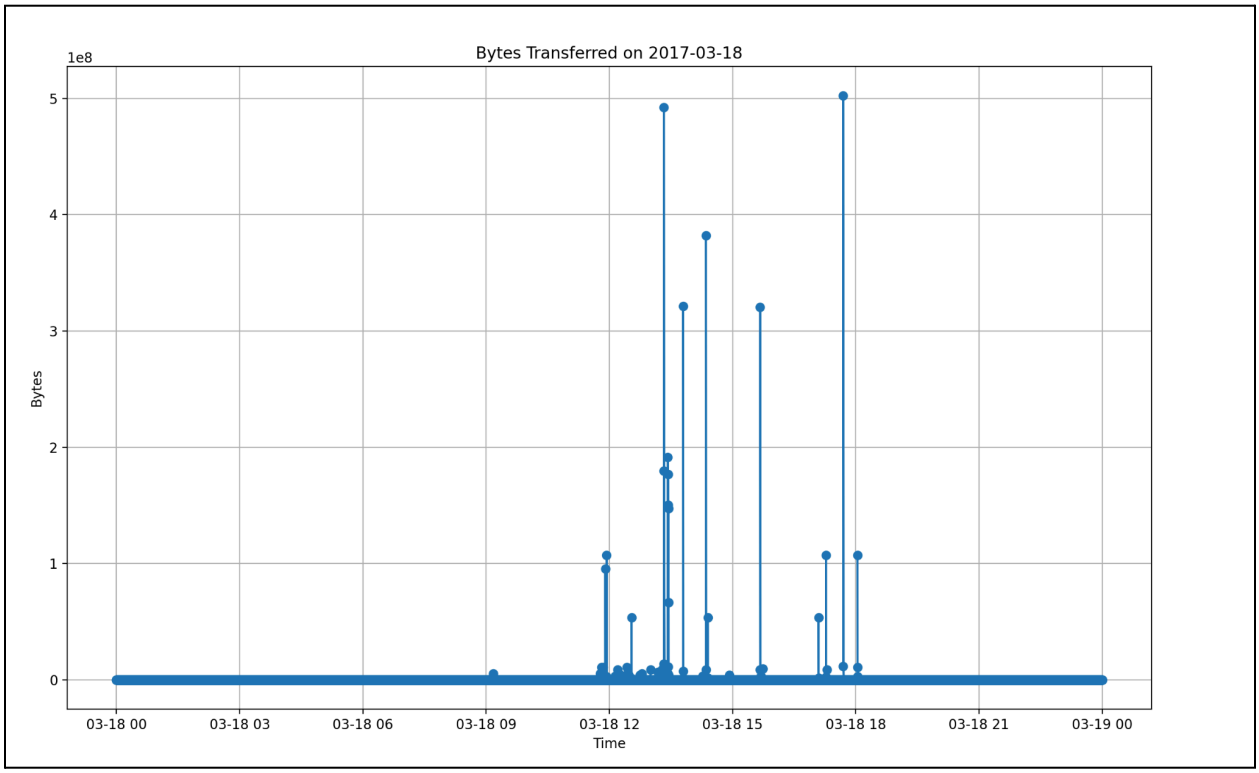
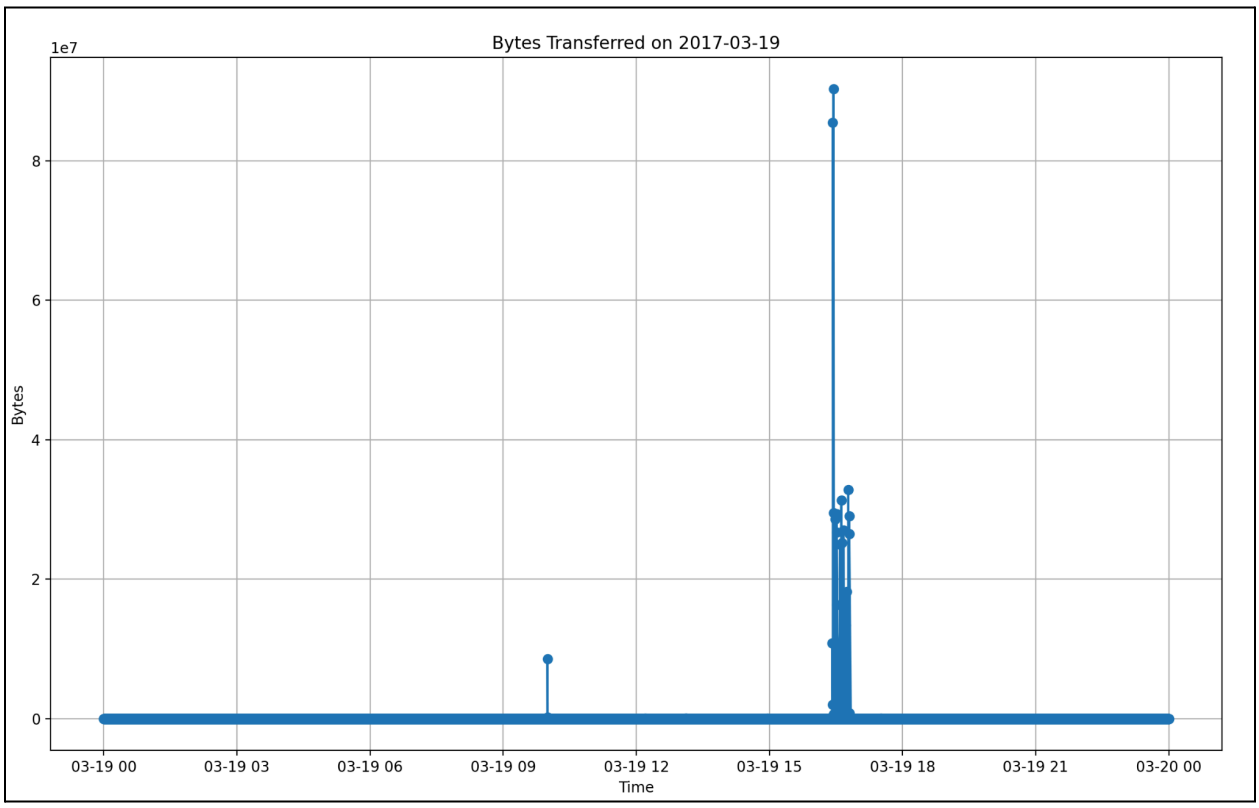
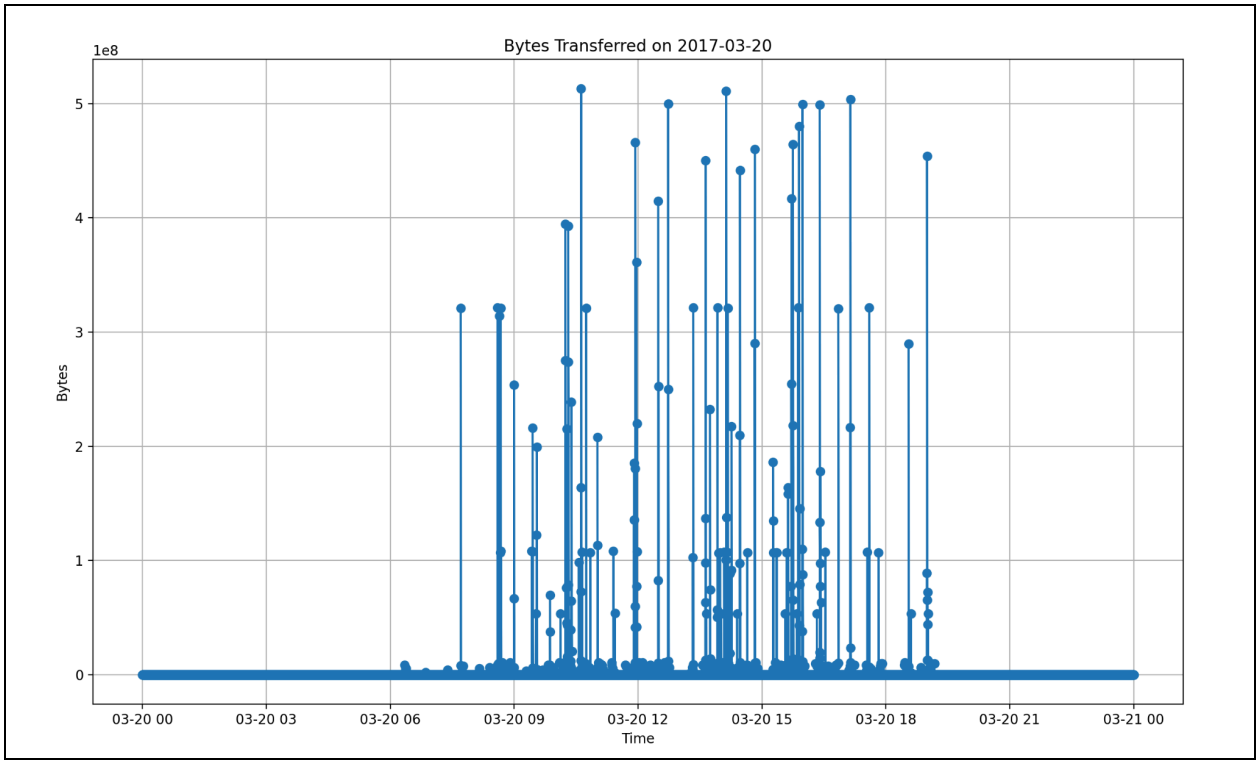
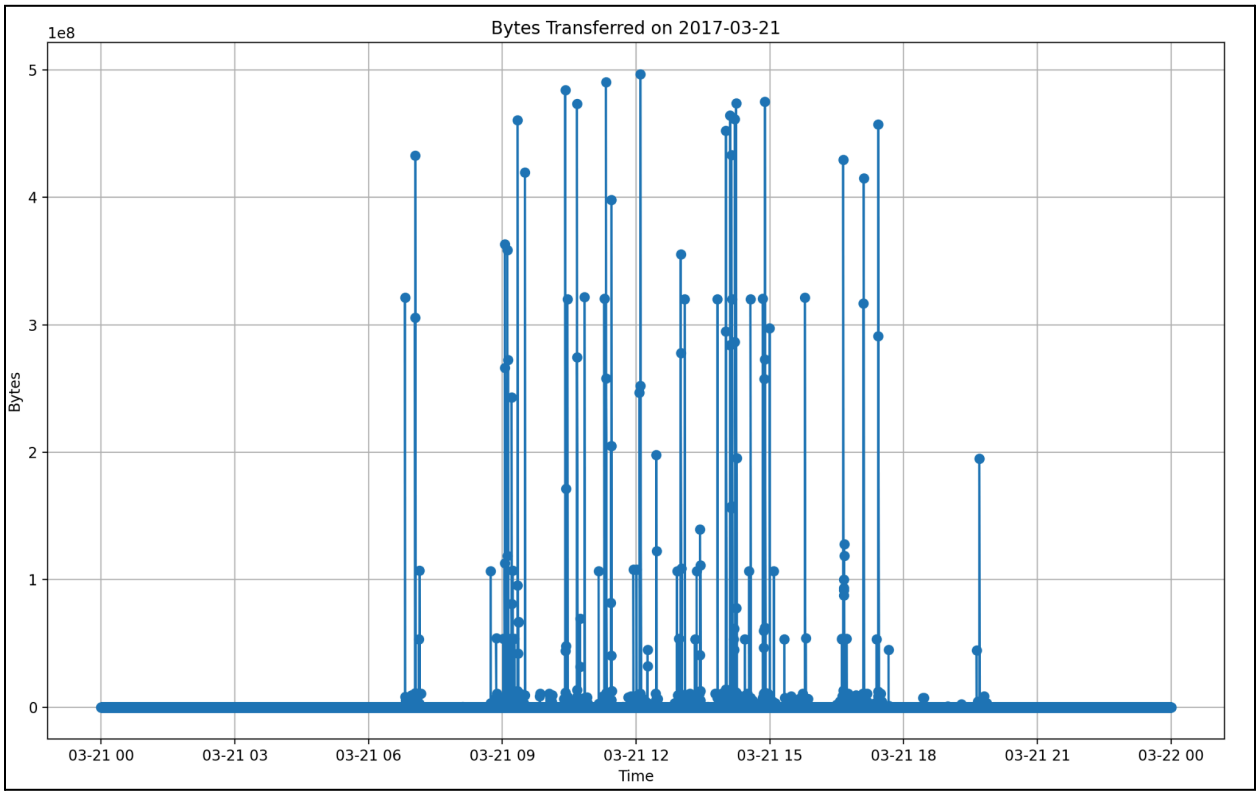
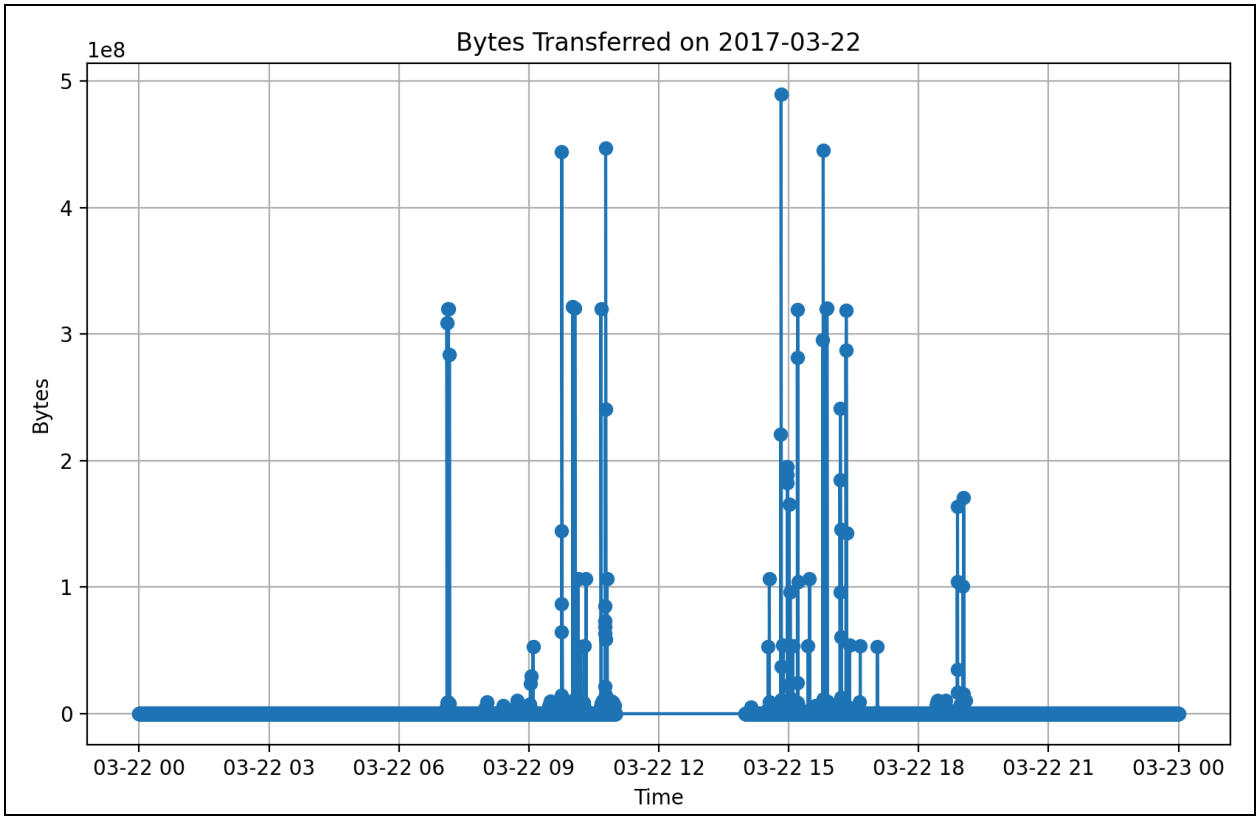
How does the distribution of Bytes/Time look per day.

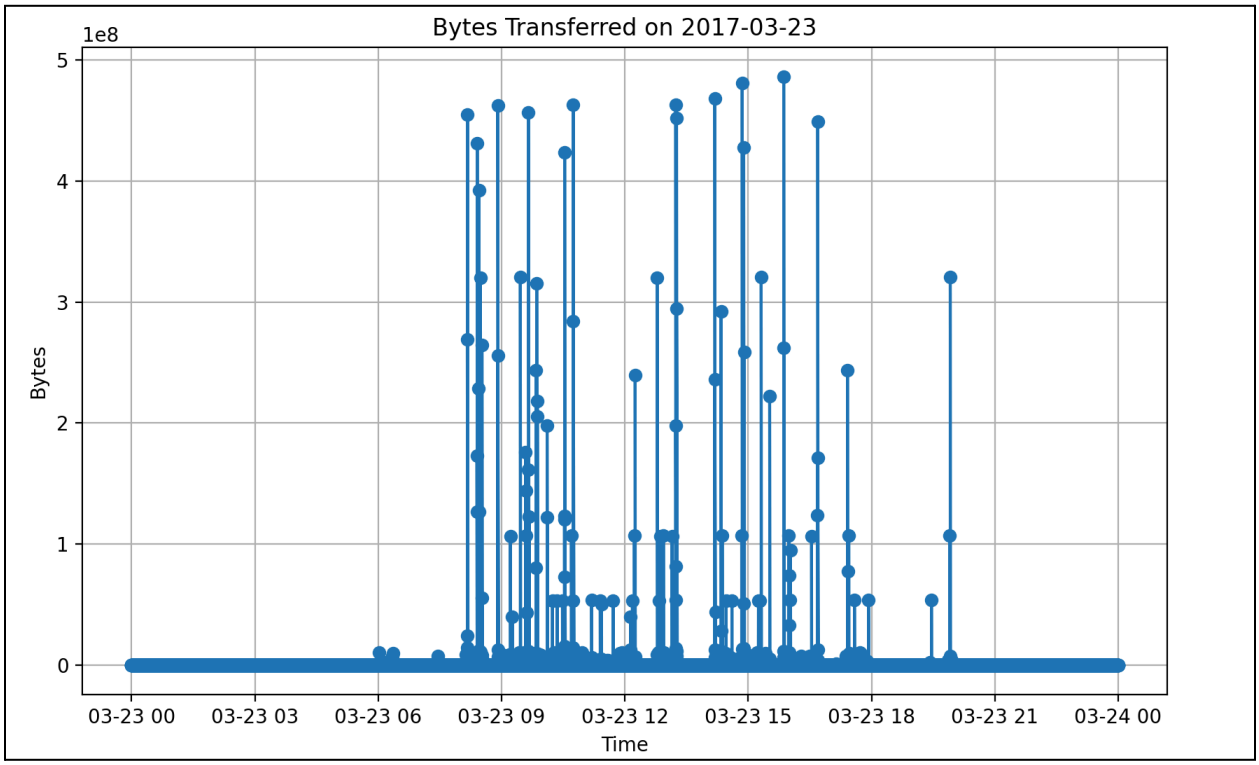


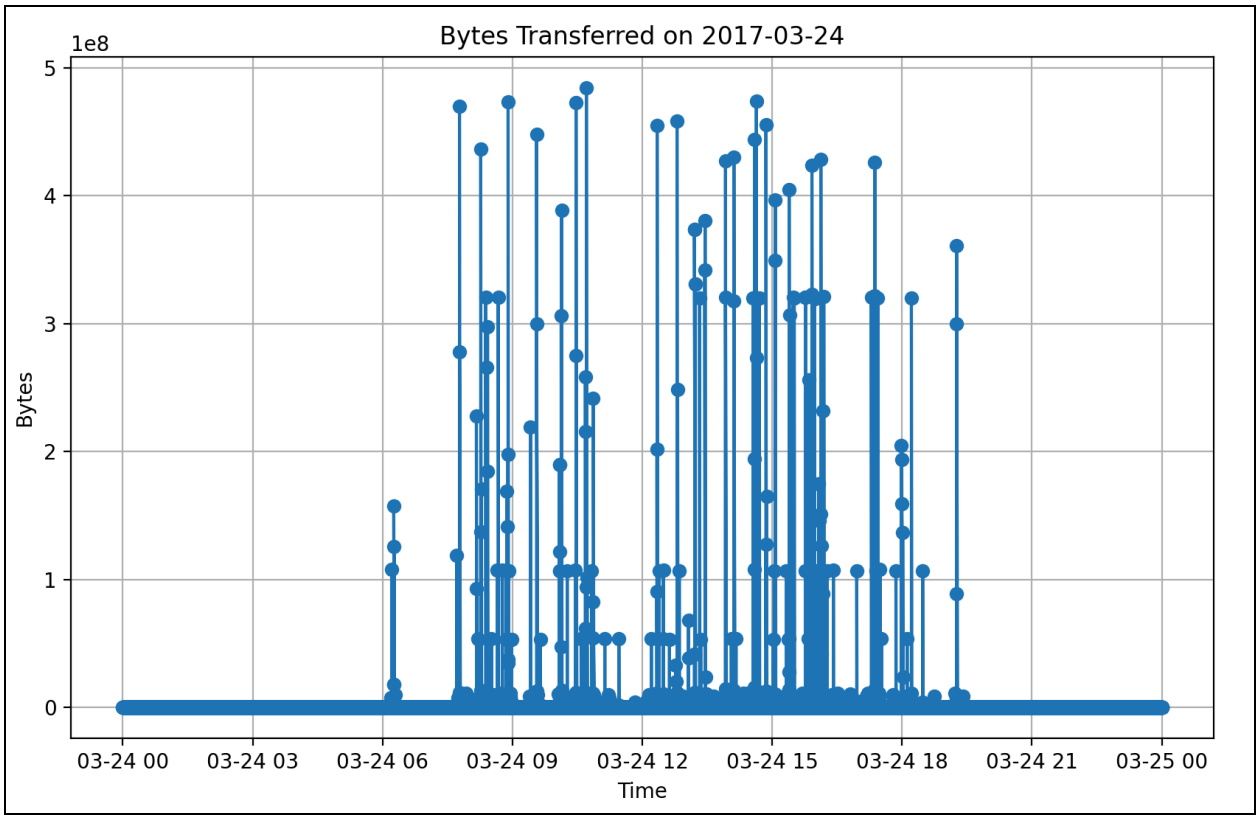


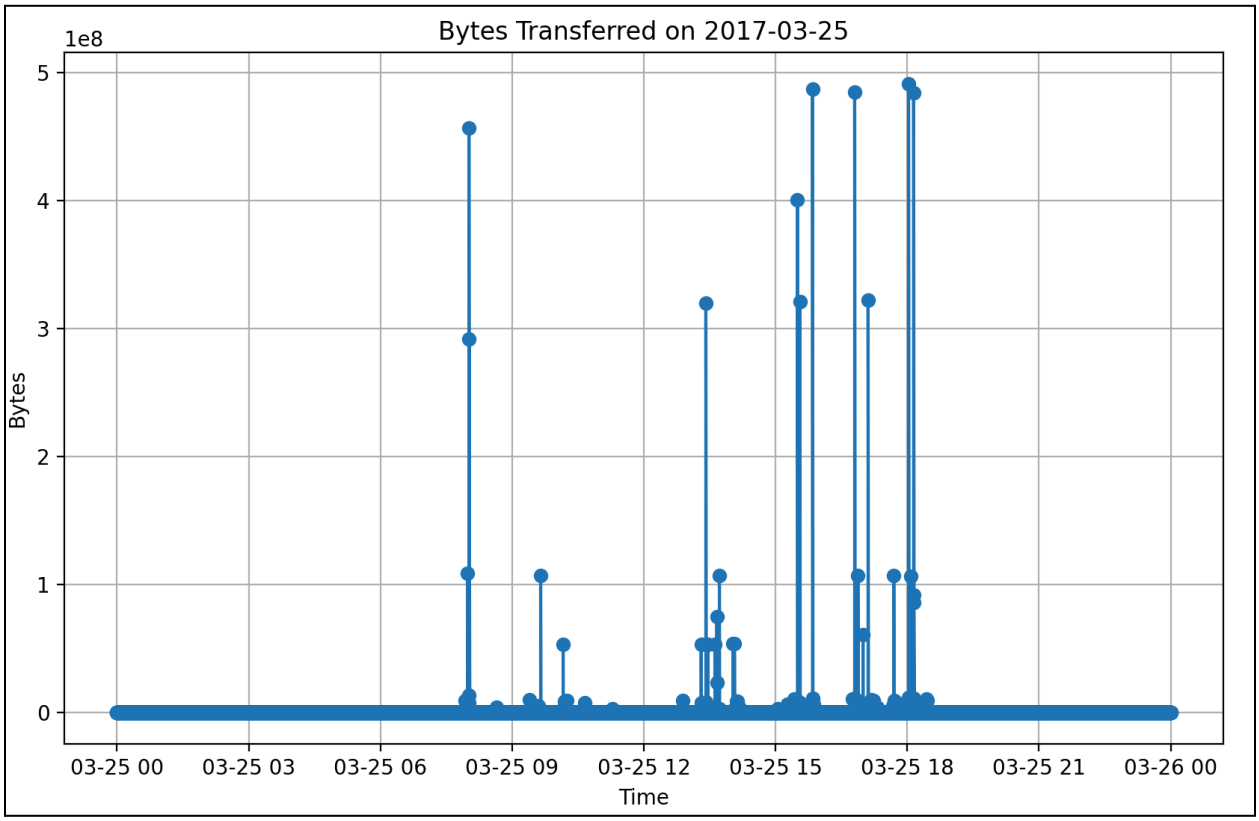

Bytes Transferred on 2017-03-20
1e8
5
4
3
2
1
0
Bytes
03-20 00
03-20 03
03-20 06
03-20 09
03-20 12
03-20 15
03-20 18
03-20 21
03-21 00
Time

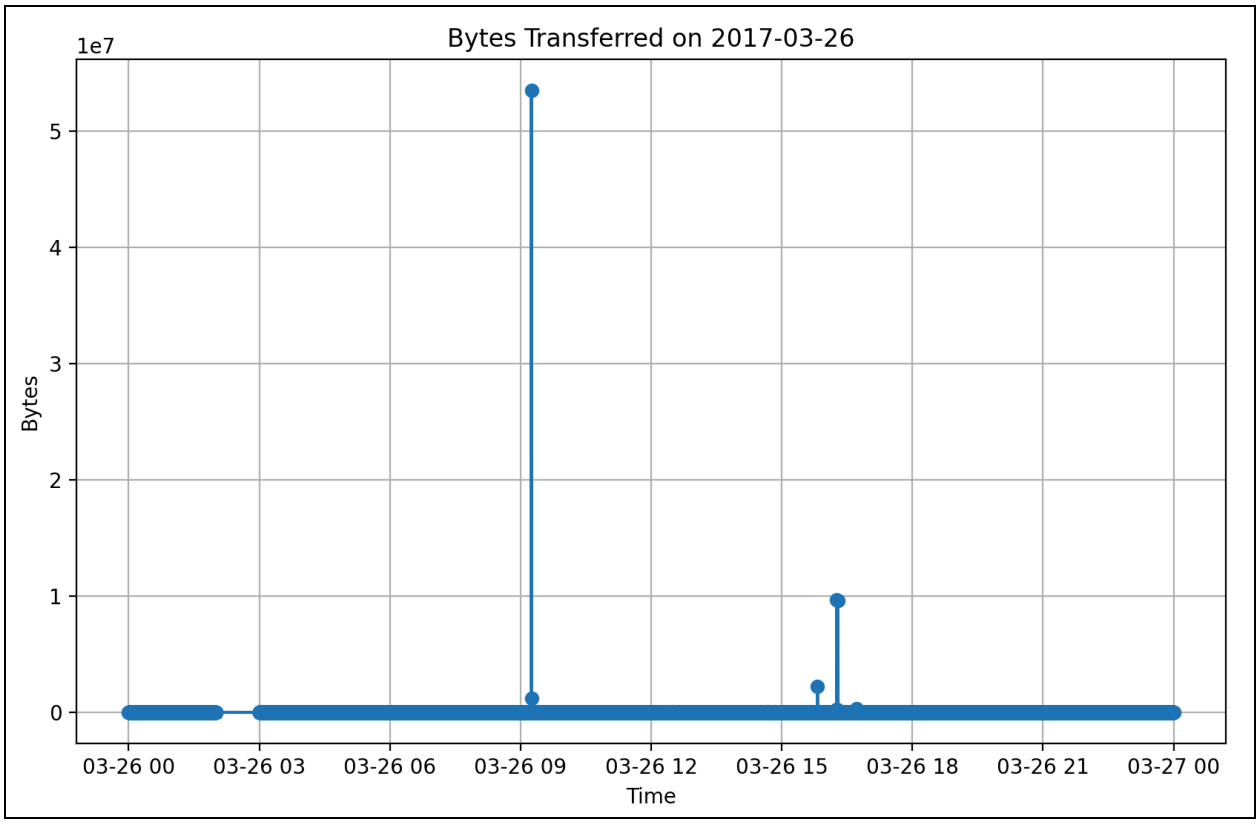

Bytes Transferred on 2017-03-21
1e8
5
4
3
2
1
0
Bytes
03-21 00
03-21 03
03-21 06
03-21 09
03-21 12
03-21 15
03-21 18
03-21 21
03-22 00
Time


Bytes Transferred on 2017-03-22
1e8
5
4
3
2
1
0
Bytes
03-22 00
03-22 03
03-22 06
03-22 09
03-22 12
03-22 15
03-22 18
03-22 21
03-23 00
Time


Bytes Transferred on 2017-03-23
1e8
5
4
3
2
1
0
Bytes
03-23 00
03-23 03
03-23 06
03-23 09
03-23 12
03-23 15
03-23 18
03-23 21
03-24 00
Time


Bytes Transferred on 2017-03-24
1e8
5
4
3
2
1
0
Bytes
03-24 00
03-24 03
03-24 06
03-24 09
03-24 12
03-24 15
03-24 18
03-24 21
03-25 00
Time


Bytes Transferred on 2017-03-25
1e8
5
4
3
2
1
0
Bytes
03-25 00
03-25 03
03-25 06
03-25 09
03-25 12
03-25 15
03-25 18
03-25 21
03-26 00
Time



Based on the above evidence, normalization is required.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv("C:\\Users\\Narayanakshay\\Downloads\\TotalDataset.csv")
pd.set_option('display.max_columns', None)

attacks_per_ip_type = df.groupby(['Src IP Addr', 'attackType'])['class'].count().reset_index()
attacks_per_ip_type[attacks_per_ip_type['Number of Attacks'] == 0]
attacks_per_ip_type = attacks_per_ip_type.rename(columns={'class': 'Number of Attacks'})
attacks_per_ip_type.drop_duplicates(subset='Src IP Addr', keep='first', inplace=True)

attacks_per_ip_type = attacks_per_ip_type.sort_values(by='Number of Attacks',
ascending=False)

plt.plot(attacks_per_ip_type['Src IP Addr'], attacks_per_ip_type['Number of Attacks'],
color='orange')
```