```python
import numpy as np
import pandas as pd

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px


import warnings
warnings.filterwarnings("ignore")
```

Loading the Data

```python
df = pd.read_csv('cybersecurity_attacks.csv')
df
```

```
                 Timestamp Source IP Address Destination IP Address  \
0      2023-05-30 06:33:58     103.216.15.12           84.9.164.252
1      2020-08-26 07:08:30    78.199.217.198         66.191.137.154
2      2022-11-13 08:23:25      63.79.210.48          198.219.82.17
3      2023-07-02 10:38:46     163.42.196.10        101.228.192.255
4      2023-07-16 13:11:07     71.166.185.76        189.243.174.238
...                    ...               ...                    ...
39995  2023-05-26 14:08:42      26.36.109.26         121.100.75.240
39996  2023-03-27 00:38:27      17.21.163.81         196.108.134.78
39997  2022-03-31 01:45:49     162.35.217.57            98.107.0.15
39998  2023-09-22 18:32:38    208.72.233.205         173.79.112.252
39999  2023-10-10 11:59:52     14.102.21.108           109.198.45.7

       Source Port  Destination Port Protocol  Packet Length Packet 
Type  \
0            31225             17616     ICMP            503        
Data
1            17245             48166     ICMP           1174        
Data
2            16811             53600      UDP            306     
Control
3            20018             32534      UDP            385        
Data
4             6131             26646      TCP           1462        
Data
...            ...               ...      ...            ...        .
..
39995        31005              6764      UDP           1428     
Control
39996         2553             28091      UDP           1184        
```

```
Control
39997        22505             25152      UDP          1043
Data
39998        20013              2703      UDP           483
Data
39999        50137             55575     ICMP          1175
Control

      Traffic Type                                       Payload Data
...  \
0             HTTP  Qui natus odio asperiores nam. Optio nobis ius...
...
1             HTTP  Aperiam quos modi officiis veritatis rem. Omni...
...
2             HTTP  Perferendis sapiente vitae soluta. Hic delectu...
...
3             HTTP  Totam maxime beatae expedita explicabo porro l...
...
4              DNS  Odit nesciunt dolorem nisi iste iusto. Animi v...
...
...            ...                                                ...
...
39995         HTTP  Quibusdam ullam consequatur consequuntur accus...
...
39996         HTTP  Quaerat neque esse. Animi expedita natus commo...
...
39997          DNS  Enim at aspernatur illum. Saepe numquam eligen...
...
39998          FTP  Officiis dolorem sed harum provident earum dis...
...
39999         HTTP  Eligendi omnis voluptate nihil voluptatibus do...
...

      Action Taken  Severity Level User Information  \
0           Logged             Low    Reyansh Dugal
1          Blocked             Low       Sumer Rana
2          Ignored             Low     Himmat Karpe
3          Blocked          Medium       Fateh Kibe
4          Blocked             Low     Dhanush Chad
...            ...             ...              ...
39995       Logged          Medium      Adira Madan
39996       Logged            High        Rati Dara
39997      Blocked             Low     Samiha Joshi
39998      Ignored             Low    Rasha Chauhan
39999       Logged          Medium      Zaina Kumar

                                      Device Information Network
Segment  \
0      Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...       Segment
A
```

```
1      Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...       Segment
B
2      Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...       Segment
C
3      Mozilla/5.0 (Macintosh; PPC Mac OS X 10_11_5; ...       Segment
B
4      Mozilla/5.0 (compatible; MSIE 5.0; Windows NT ...       Segment
C
...                                                  ...             .
..
39995  Mozilla/5.0 (iPad; CPU iPad OS 14_2_1 like Mac...       Segment
A
39996  Mozilla/5.0 (Windows; U; Windows 98; Win 9x 4....       Segment
C
39997  Mozilla/5.0 (Windows; U; Windows NT 4.0) Apple...       Segment
C
39998  Mozilla/5.0 (X11; Linux i686) AppleWebKit/536....       Segment
B
39999  Mozilla/5.0 (iPod; U; CPU iPhone OS 3_0 like M...       Segment
A

                  Geo-location Data Proxy Information Firewall
Logs  \
0                 Jamshedpur, Sikkim      150.9.97.135      Log Data

1                 Bilaspur, Nagaland               NaN      Log Data

2                  Bokaro, Rajasthan    114.133.48.179      Log Data

3                 Jaunpur, Rajasthan               NaN           NaN

4                 Anantapur, Tripura     149.6.110.119           NaN

...                              ...               ...           ...

39995                Nashik, Manipur               NaN      Log Data

39996              Vadodara, Mizoram       60.51.30.46      Log Data

39997  Mahbubnagar, Himachal Pradesh               NaN      Log Data

39998    Rourkela, Arunachal Pradesh      137.76.130.8      Log Data

39999       Pudukkottai, West Bengal   112.169.115.139      Log Data


      IDS/IPS Alerts Log Source
0                NaN     Server
1                NaN   Firewall
2         Alert Data   Firewall
```

```
3         Alert Data   Firewall
4         Alert Data   Firewall
...              ...        ...
39995     Alert Data   Firewall
39996            NaN   Firewall
39997     Alert Data     Server
39998            NaN     Server
39999     Alert Data   Firewall

[40000 rows x 25 columns]
```

Exploring the Dataset

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40000 entries, 0 to 39999
Data columns (total 25 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Timestamp               40000 non-null  object 
 1   Source IP Address       40000 non-null  object 
 2   Destination IP Address  40000 non-null  object 
 3   Source Port             40000 non-null  int64  
 4   Destination Port        40000 non-null  int64  
 5   Protocol                40000 non-null  object 
 6   Packet Length           40000 non-null  int64  
 7   Packet Type             40000 non-null  object 
 8   Traffic Type            40000 non-null  object 
 9   Payload Data            40000 non-null  object 
 10  Malware Indicators      20000 non-null  object 
 11  Anomaly Scores          40000 non-null  float64
 12  Alerts/Warnings         19933 non-null  object 
 13  Attack Type             40000 non-null  object 
 14  Attack Signature        40000 non-null  object 
 15  Action Taken            40000 non-null  object 
 16  Severity Level          40000 non-null  object 
 17  User Information        40000 non-null  object 
 18  Device Information      40000 non-null  object 
 19  Network Segment         40000 non-null  object 
 20  Geo-location Data       40000 non-null  object 
 21  Proxy Information       20149 non-null  object 
 22  Firewall Logs           20039 non-null  object 
 23  IDS/IPS Alerts          19950 non-null  object 
 24  Log Source              40000 non-null  object 
dtypes: float64(1), int64(3), object(21)
memory usage: 7.6+ MB
```

```
df.columns
```

```
Index(['Timestamp', 'Source IP Address', 'Destination IP Address',
       'Source Port', 'Destination Port', 'Protocol', 'Packet Length',
       'Packet Type', 'Traffic Type', 'Payload Data', 'Malware
Indicators',
       'Anomaly Scores', 'Alerts/Warnings', 'Attack Type', 'Attack
Signature',
       'Action Taken', 'Severity Level', 'User Information',
       'Device Information', 'Network Segment', 'Geo-location Data',
       'Proxy Information', 'Firewall Logs', 'IDS/IPS Alerts', 'Log
Source'],
      dtype='object')
```

```
df.shape
```

```
(40000, 25)
```

Taking care of the Null Values

```
df.isnull().sum().sort_values(ascending=False)
```

```
Alerts/Warnings           20067
IDS/IPS Alerts            20050
Malware Indicators        20000
Firewall Logs             19961
Proxy Information         19851
Attack Type                   0
Geo-location Data             0
Network Segment               0
Device Information            0
User Information              0
Severity Level                0
Action Taken                  0
Attack Signature              0
Timestamp                     0
Source IP Address             0
Anomaly Scores                0
Payload Data                  0
Traffic Type                  0
Packet Type                   0
Packet Length                 0
Protocol                      0
Destination Port              0
Source Port                   0
Destination IP Address        0
Log Source                    0
dtype: int64
```

There are **5** columns with null values:

- **Alerts/Warnings** = 20067
- **IDS/IPS Alerts** = 20050
- **Malware Indicators** = 20000
- **Firewall Logs** = 19961
- **Proxy Information** = 19851

```
df['Alerts/Warnings'].unique()
```

```
array([nan, 'Alert Triggered'], dtype=object)
```

```
def categorize_alerts(row):
    if row['Alerts/Warnings'] =="Alert Triggered":
        return 'Alert Triggered'
    else:
        return 'None'

df['Alerts/Warnings'] = df.apply(categorize_alerts, axis=1)
df['Alerts/Warnings']
```

```
0                   None
1                   None
2        Alert Triggered
3        Alert Triggered
4        Alert Triggered
              ...
39995               None
39996               None
39997               None
39998    Alert Triggered
39999    Alert Triggered
Name: Alerts/Warnings, Length: 40000, dtype: object
```

```
df['IDS/IPS Alerts'].unique()
```

```
array([nan, 'Alert Data'], dtype=object)
```

```
df['IDS/IPS Alerts'] = df['IDS/IPS Alerts'].apply(lambda x: 'No Data'
                                                  if pd.isna(x)
                                                  else x)
df['IDS/IPS Alerts'].unique()
```

```
array(['No Data', 'Alert Data'], dtype=object)
```

```
df['Malware Indicators'].unique()
```

```
array(['IoC Detected', nan], dtype=object)
```

```
df['Malware Indicators'] = df['Malware Indicators'].apply(lambda x:
'No Detection'
                                                         if pd.isna(x)
                                                         else x)
df['Malware Indicators'].unique()
```

```
array(['IoC Detected', 'No Detection'], dtype=object)
```

```
df['Firewall Logs'] = df['Firewall Logs'].apply(lambda x: 'No Data'
                                               if pd.isna(x)
                                               else x)
df['Firewall Logs'].unique()
```

```
array(['Log Data', 'No Data'], dtype=object)
```

```
df['Proxy Information'].unique()
```

```
array(['150.9.97.135', nan, '114.133.48.179', ..., '60.51.30.46',
       '137.76.130.8', '112.169.115.139'], dtype=object)
```

```
df['Proxy Information'] = df['Proxy Information'].apply(lambda x: 'No
Proxy Data'
                                                       if pd.isna(x)
                                                       else x)
df['Proxy Information'].unique()
```

```
array(['150.9.97.135', 'No Proxy Data', '114.133.48.179', ...,
       '60.51.30.46', '137.76.130.8', '112.169.115.139'],
dtype=object)
```

```
df.isnull().sum().sort_values(ascending=False)
```

```
Timestamp                 0
Attack Type               0
IDS/IPS Alerts            0
Firewall Logs             0
Proxy Information         0
Geo-location Data         0
Network Segment           0
Device Information        0
User Information          0
Severity Level            0
Action Taken              0
Attack Signature          0
Alerts/Warnings           0
Source IP Address         0
Anomaly Scores            0
Malware Indicators        0
Payload Data              0
Traffic Type              0
Packet Type               0
Packet Length             0
Protocol                  0
Destination Port          0
```

```
Source Port               0
Destination IP Address    0
Log Source                0
dtype: int64
```

No more Null Values!!!

```
df.columns
```

```
Index(['Timestamp', 'Source IP Address', 'Destination IP Address',
       'Source Port', 'Destination Port', 'Protocol', 'Packet Length',
       'Packet Type', 'Traffic Type', 'Payload Data', 'Malware
Indicators',
       'Anomaly Scores', 'Alerts/Warnings', 'Attack Type', 'Attack
Signature',
       'Action Taken', 'Severity Level', 'User Information',
       'Device Information', 'Network Segment', 'Geo-location Data',
       'Proxy Information', 'Firewall Logs', 'IDS/IPS Alerts', 'Log
Source'],
      dtype='object')
```

Breaking Down Device Information

```
df['Device Information']
```

```
0        Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...
1        Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...
2        Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...
3        Mozilla/5.0 (Macintosh; PPC Mac OS X 10_11_5; ...
4        Mozilla/5.0 (compatible; MSIE 5.0; Windows NT ...
                               ...
39995    Mozilla/5.0 (iPad; CPU iPad OS 14_2_1 like Mac...
39996    Mozilla/5.0 (Windows; U; Windows 98; Win 9x 4....
39997    Mozilla/5.0 (Windows; U; Windows NT 4.0) Apple...
39998    Mozilla/5.0 (X11; Linux i686) AppleWebKit/536....
39999    Mozilla/5.0 (iPod; U; CPU iPhone OS 3_0 like M...
Name: Device Information, Length: 40000, dtype: object
```

** Creating a Device/OS Column**

```
import re

devices = [
    r'Windows',
    r'Linux',
    r'Android',
    r'iPad',
    r'iPod',
    r'iPhone',
    r'Macintosh']


def device_os_finder(user_agent):
    for device in devices:
        match_device = re.search(device, user_agent, re.I)  # re.I
makes the search case-insensitive
        if match_device:
            return match_device.group()
    return 'Unknown'

# Extract device or OS
df['Device/OS'] = df['Device Information'].apply(device_os_finder)
df['Device/OS'].head(10)
```

```
0      Windows
1      Windows
2      Windows
3    Macintosh
4      Windows
5        Linux
6        Linux
7    Macintosh
8    Macintosh
9      Windows
Name: Device/OS, dtype: object
```

```
df['Device/OS'].value_counts()
```

```
Device/OS
Windows      17953
Linux         8840
Macintosh     5813
iPod          2656
Android       1620
iPhone        1567
iPad          1551
Name: count, dtype: int64
```

** Creating a Browser Column**

```
df['Device Information']
```

```
0        Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...
1        Mozilla/5.0 (compatible; MSIE 8.0; Windows NT ...
2        Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...
3        Mozilla/5.0 (Macintosh; PPC Mac OS X 10_11_5; ...
4        Mozilla/5.0 (compatible; MSIE 5.0; Windows NT ...
                               ...
39995    Mozilla/5.0 (iPad; CPU iPad OS 14_2_1 like Mac...
39996    Mozilla/5.0 (Windows; U; Windows 98; Win 9x 4....
39997    Mozilla/5.0 (Windows; U; Windows NT 4.0) Apple...
39998    Mozilla/5.0 (X11; Linux i686) AppleWebKit/536....
39999    Mozilla/5.0 (iPod; U; CPU iPhone OS 3_0 like M...
Name: Device Information, Length: 40000, dtype: object
```

```
df['Browser'] = df['Device Information'].str.split('/').str[0]
df['Browser']
```

```
0        Mozilla
1        Mozilla
2        Mozilla
3        Mozilla
4        Mozilla
          ...
39995    Mozilla
39996    Mozilla
39997    Mozilla
39998    Mozilla
39999    Mozilla
Name: Browser, Length: 40000, dtype: object
```

```
df['Browser'].value_counts()
```

```
Browser
Mozilla    31951
Opera       8049
Name: count, dtype: int64
```

Creating Additional Time Columns

```
# Converting to datetime

df['Timestamp'] = pd.to_datetime(df['Timestamp'])
df['Timestamp'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 40000 entries, 0 to 39999
Series name: Timestamp
Non-Null Count  Dtype
--------------  -----
40000 non-null  datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 312.6 KB
```

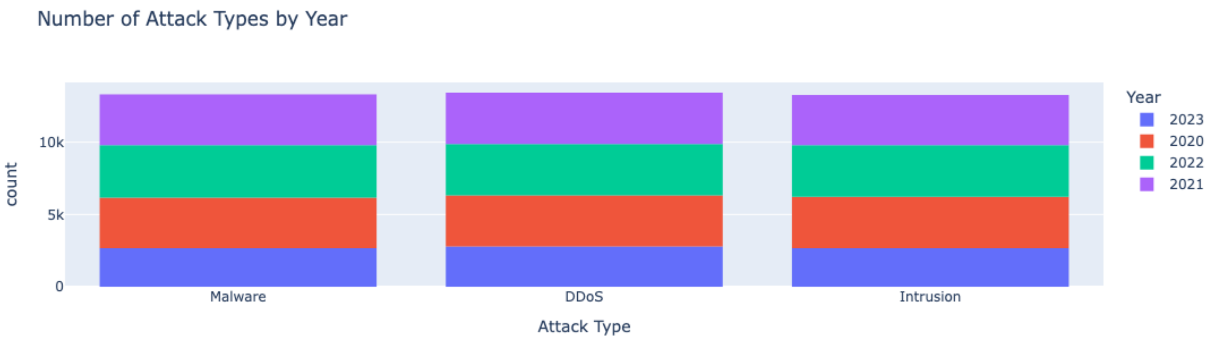
```
df['Year'] = df['Timestamp'].dt.year
df['Month'] = df['Timestamp'].dt.month
df['DayofWeek'] = df['Timestamp'].dt.dayofweek
df['Day'] = df['Timestamp'].dt.day
df['Hour'] = df['Timestamp'].dt.hour
df['Minute'] = df['Timestamp'].dt.minute
df['Second'] = df['Timestamp'].dt.second

df
```

```
                Timestamp Source IP Address Destination IP Address  \
0     2023-05-30 06:33:58     103.216.15.12           84.9.164.252
1     2020-08-26 07:08:30    78.199.217.198         66.191.137.154
2     2022-11-13 08:23:25      63.79.210.48          198.219.82.17
3     2023-07-02 10:38:46     163.42.196.10        101.228.192.255
4     2023-07-16 13:11:07     71.166.185.76        189.243.174.238
...                   ...               ...                    ...
39995 2023-05-26 14:08:42      26.36.109.26         121.100.75.240
39996 2023-03-27 00:38:27      17.21.163.81         196.108.134.78
39997 2022-03-31 01:45:49     162.35.217.57            98.107.0.15
39998 2023-09-22 18:32:38    208.72.233.205         173.79.112.252
39999 2023-10-10 11:59:52     14.102.21.108           109.198.45.7

       Source Port  Destination Port Protocol  Packet Length Packet 
Type  \
0            31225             17616     ICMP            503
Data
1            17245             48166     ICMP           1174
Data
2            16811             53600      UDP            306
Control
3            20018             32534      UDP            385
Data
4             6131             26646      TCP           1462
Data
...            ...               ...      ...            ...         .
..
39995        31005              6764      UDP           1428
Control
39996         2553             28091      UDP           1184
Control
39997        22505             25152      UDP           1043
Data
39998        20013              2703      UDP            483
Data
39999        50137             55575     ICMP           1175
Control

      Traffic Type                                       Payload Data 
...  \
0             HTTP  Qui natus odio asperiores nam. Optio nobis ius...
...
1             HTTP  Aperiam quos modi officiis veritatis rem. Omni...
...
2             HTTP  Perferendis sapiente vitae soluta. Hic delectu...
...
3             HTTP  Totam maxime beatae expedita explicabo porro l...
...
4              DNS  Odit nesciunt dolorem nisi iste iusto. Animi v...
...
...            ...                                                ...
...
39995         HTTP  Quibusdam ullam consequatur consequuntur accus...
...
39996         HTTP  Quaerat neque esse. Animi expedita natus commo...
...
39997          DNS  Enim at aspernatur illum. Saepe numquam eligen...
...
39998          FTP  Officiis dolorem sed harum provident earum dis...
...
39999         HTTP  Eligendi omnis voluptate nihil voluptatibus do...
...

      Log Source  Device/OS  Browser  Year Month DayofWeek Day Hour
Minute  \
0         Server    Windows  Mozilla  2023     5         1  30    6
33
1       Firewall    Windows  Mozilla  2020     8         2  26    7
8
2       Firewall    Windows  Mozilla  2022    11         6  13    8
23
3       Firewall  Macintosh  Mozilla  2023     7         6   2   10
38
4       Firewall    Windows  Mozilla  2023     7         6  16   13
11
...          ...        ...      ...   ...   ...       ...  ..  ...
...
39995   Firewall       iPad  Mozilla  2023     5         4  26   14
8
39996   Firewall    Windows  Mozilla  2023     3         0  27    0
38
39997     Server    Windows  Mozilla  2022     3         3  31    1
45
39998     Server      Linux  Mozilla  2023     9         4  22   18
32
39999   Firewall       iPod  Mozilla  2023    10         1  10   11
59

      Second
```

```
0         58
1         30
2         25
3         46
4          7
...      ...
39995     42
39996     27
39997     49
39998     38
39999     52

[40000 rows x 34 columns]
```

Insights

```
df.columns

Index(['Timestamp', 'Source IP Address', 'Destination IP Address',
       'Source Port', 'Destination Port', 'Protocol', 'Packet Length',
       'Packet Type', 'Traffic Type', 'Payload Data', 'Malware
Indicators',
       'Anomaly Scores', 'Alerts/Warnings', 'Attack Type', 'Attack
Signature',
       'Action Taken', 'Severity Level', 'User Information',
       'Device Information', 'Network Segment', 'Geo-location Data',
       'Proxy Information', 'Firewall Logs', 'IDS/IPS Alerts', 'Log
Source',
       'Device/OS', 'Browser', 'Year', 'Month', 'DayofWeek', 'Day',
'Hour',
       'Minute', 'Second'],
      dtype='object')

plt = px.histogram(df, x='Attack Type', color='Year', title='Number of
Attack Types by Year')
plt.show()
```



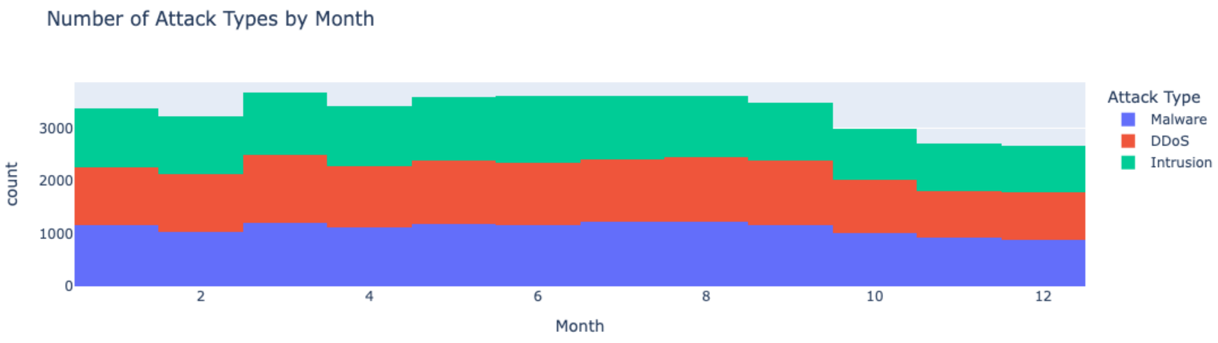
```
df.groupby(['Year'])['Attack Type'].value_counts()

Year  Attack Type
2020  Intrusion      3551
      DDoS           3533
      Malware        3489
2021  DDoS           3545
      Malware        3518
      Intrusion      3475
2022  Malware        3629
      Intrusion      3563
      DDoS           3558
2023  DDoS           2792
      Intrusion      2676
      Malware        2671
Name: count, dtype: int64
```

- In **2020**, **Intrusion** attacks were the most frequent
- In **2021** and **2023**, **DDoS** attacks were the most frequent
- In **2022**, **Malware** attacks were the most frequent

```
df.groupby(['Year'])['Attack Type'].count()

Year
2020    10573
2021    10538
2022    10750
2023     8139
Name: Attack Type, dtype: int64
```

Least amount of attack types in **2023**!

```
plt = px.histogram(df, x='Month', color='Attack Type', title='Number
of Attack Types by Month')
plt.show()
```



```
month_attacks = df.groupby(['Month'])['Attack Type'].count()
month_attacks.sort_values(ascending=False)

Month
3     3678
7     3623
8     3615
6     3609
5     3595
9     3482
4     3421
1     3378
2     3232
10    2989
11    2703
12    2675
Name: Attack Type, dtype: int64
```

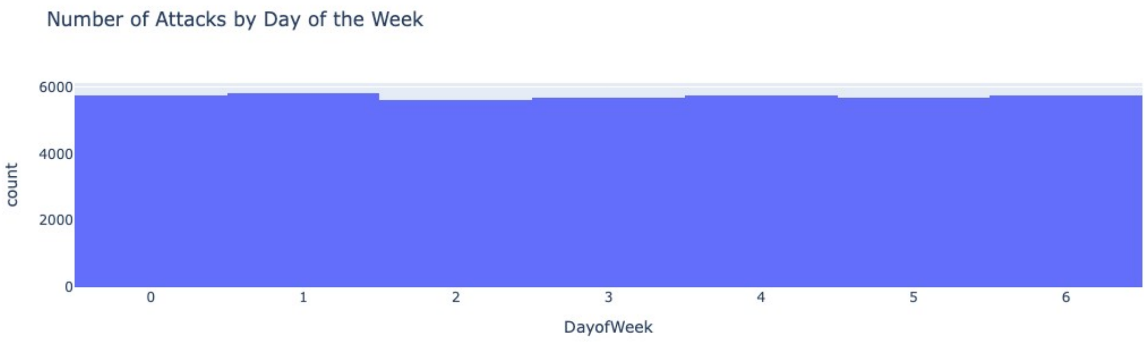
Most attacks happened in March and the least amount happened in December!

```
attack_types_month = df.groupby(['Month'])['Attack
Type'].value_counts()
attack_types_month.sort_values(ascending=False)

Month  Attack Type
3      DDoS           1299
6      Intrusion      1268
7      Malware        1236
8      DDoS           1226
7      Intrusion      1224
9      DDoS           1221
8      Malware        1216
5      Intrusion      1212
       DDoS           1200
3      Malware        1197
6      DDoS           1190
5      Malware        1183
```

```
3      Intrusion      1182
8      Intrusion      1173
9      Malware        1172
4      DDoS           1166
7      DDoS           1163
1      Malware        1163
6      Malware        1151
4      Intrusion      1140
1      Intrusion      1116
4      Malware        1115
2      Intrusion      1107
1      DDoS           1099
9      Intrusion      1089
2      DDoS           1085
       Malware        1040
10     DDoS           1015
       Malware        1013
       Intrusion       961
11     Malware         935
       Intrusion       899
12     DDoS            895
       Intrusion       894
       Malware         886
11     DDoS            869
Name: count, dtype: int64
```

- **March: Most DDoS Attacks**
- **June: Most Intrusion Attacks**
- **July: Most Malware Attacks**

```
plt = px.histogram(df, x='DayofWeek', title='Number of Attacks by Day
of the Week')
plt.show()
```

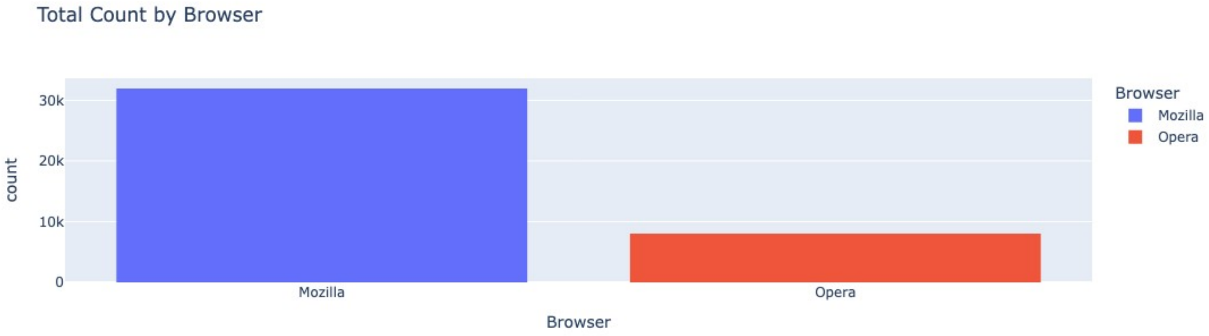


Monday is the most popular day for an attack!

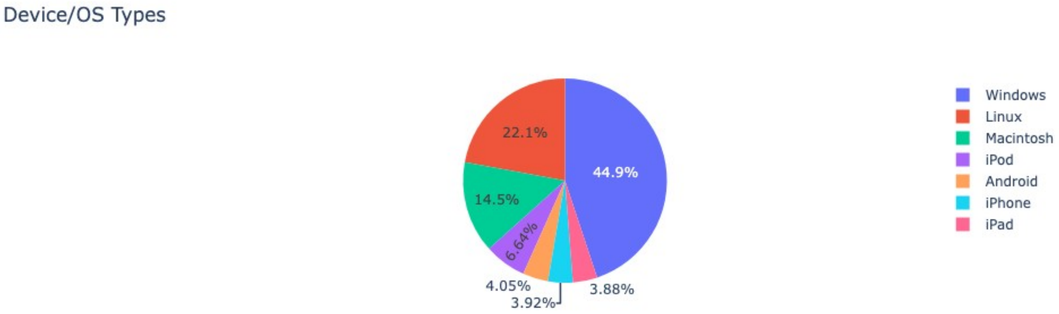
```
df['DayofWeek'].value_counts()
```

```
DayofWeek
1    5813
4    5753
0    5752
6    5744
3    5676
5    5663
2    5599
Name: count, dtype: int64
```

```
plt = px.histogram(df ,x='Browser', color = 'Browser', title = 'Total
Count by Browser')
plt.show()
```



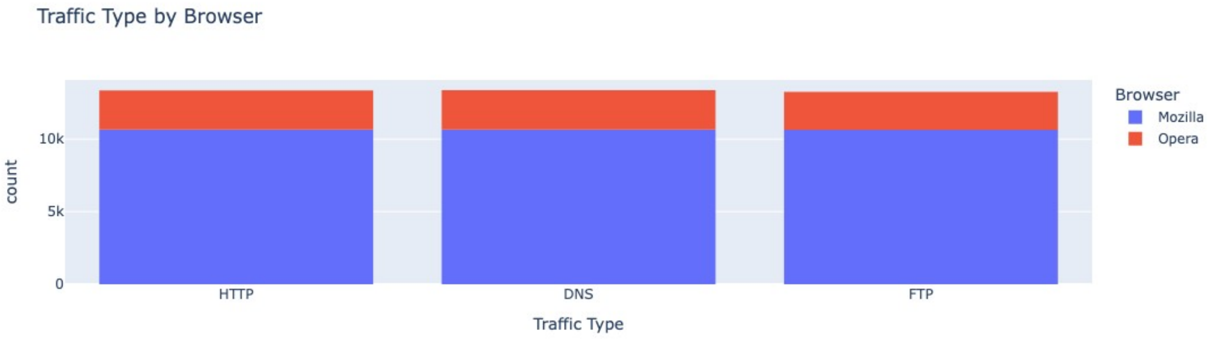
```
plt = px.pie(df ,names='Device/OS', title = 'Device/OS Types')
plt.show()
```



Top 3 Device/OS Types:

1. **Windows**
2. **Linux**
3. **Macintosh**

```
plt = px.histogram(df, x='Traffic Type', color='Browser',
title='Traffic Type by Browser')
plt.show()
```

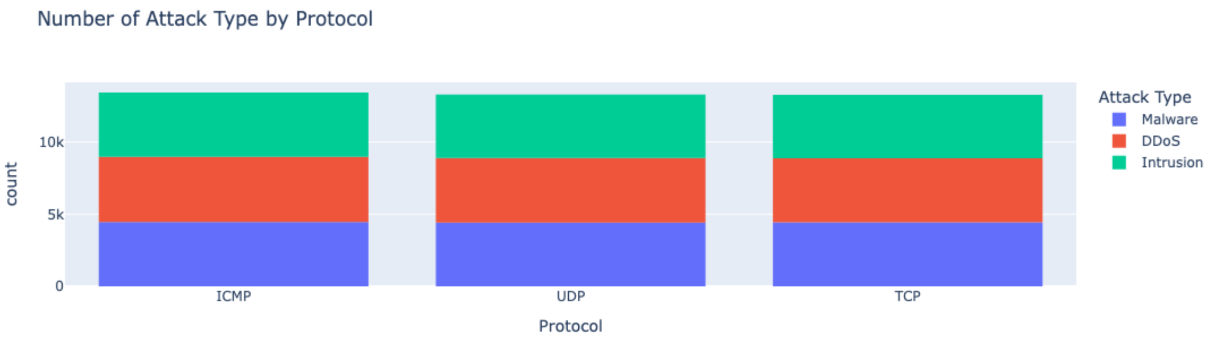


Average Packet Length by Device/OS Type for each Browser

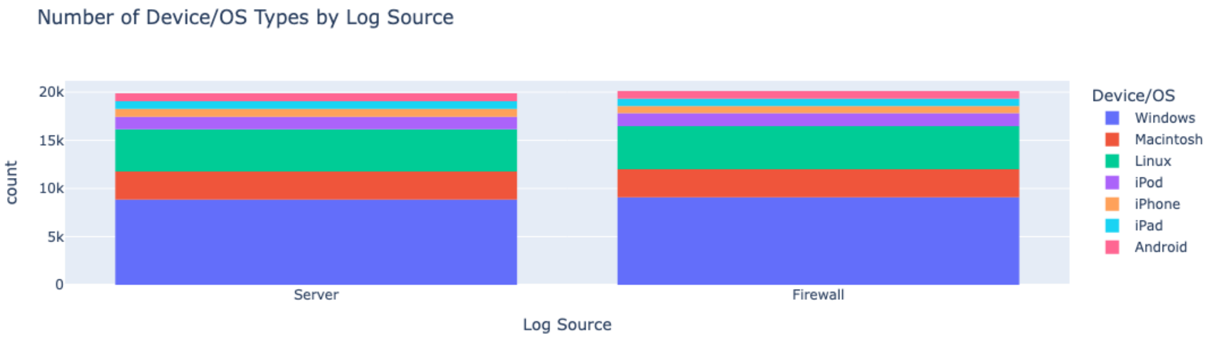
```
packet_length = df.groupby(['Browser', 'Device/OS']).agg({'Packet
Length':'mean'})
packet_length.sort_values('Packet Length',ascending=False)
```

```
                   Packet Length
Browser Device/OS
Mozilla iPad          800.304320
Opera   Windows       788.932635
Mozilla Android       786.717284
        iPod          784.408886
        Macintosh     782.677963
        Windows       779.130513
Opera   Linux         778.100223
Mozilla iPhone        777.880664
        Linux         774.952907
```

```
plt = px.histogram(df ,x='Protocol', color = 'Attack Type', title =
'Number of Attack Type by Protocol')
plt.show()
```

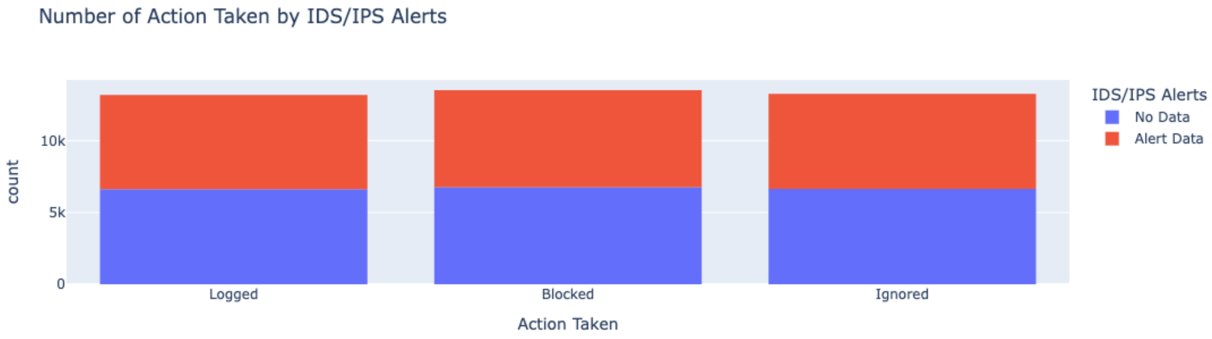


```
plt = px.histogram(df, x='Log Source', color='Device/OS',
title='Number of Device/OS Types by Log Source')
plt.show()
```

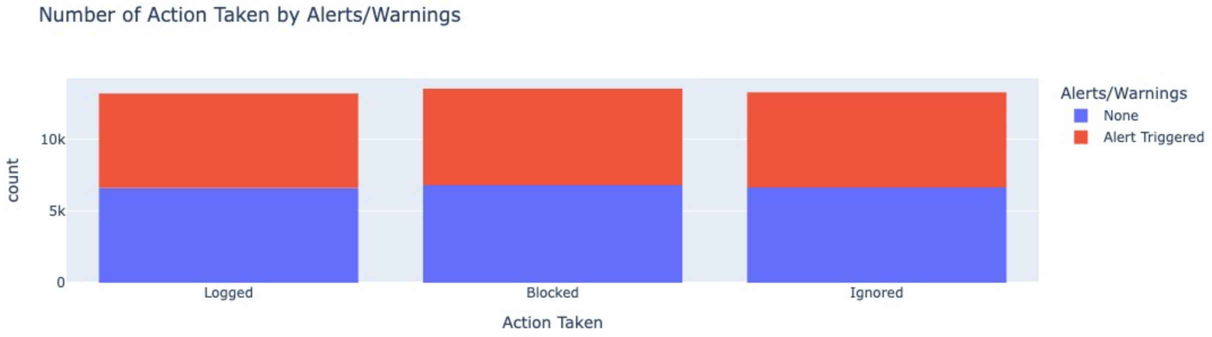


```
df.groupby(['Log Source'])['Device/OS'].value_counts()

Log Source  Device/OS
Firewall    Windows      9092
            Linux        4449
            Macintosh    2920
            iPod         1347
            Android       792
            iPad          765
            iPhone        751
Server      Windows      8861
            Linux        4391
            Macintosh    2893
            iPod         1309
            Android       828
            iPhone        816
            iPad          786
Name: count, dtype: int64
```

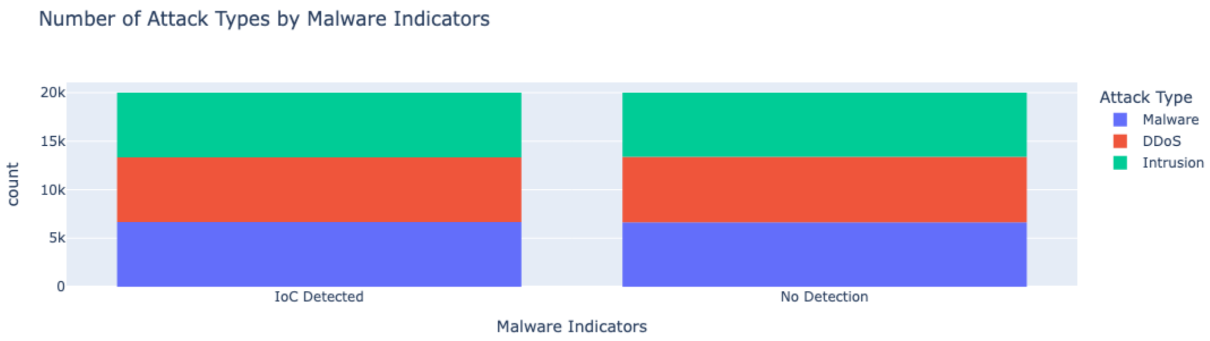
```
plt = px.histogram(df, x='Action Taken', color='IDS/IPS Alerts',
title= 'Number of Action Taken by IDS/IPS Alerts')
plt.show()
```



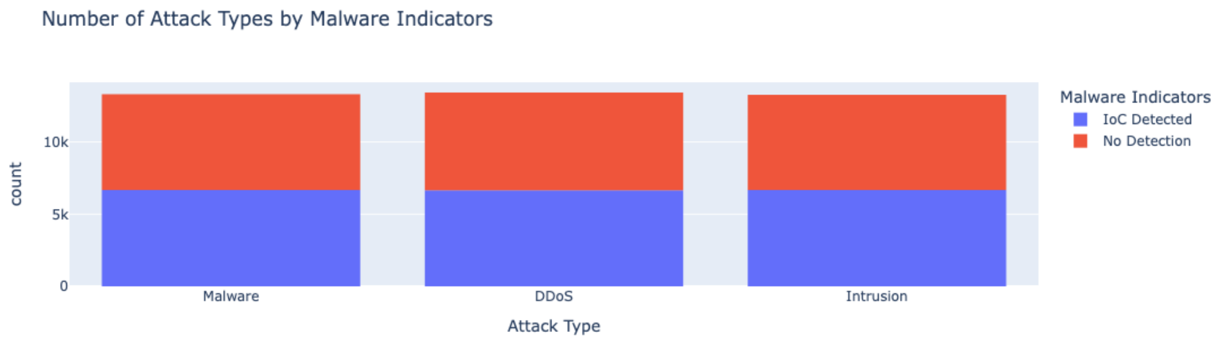
```
plt = px.histogram(df, x='Action Taken', color='Alerts/Warnings',
title= 'Number of Action Taken by Alerts/Warnings')
plt.show()
```



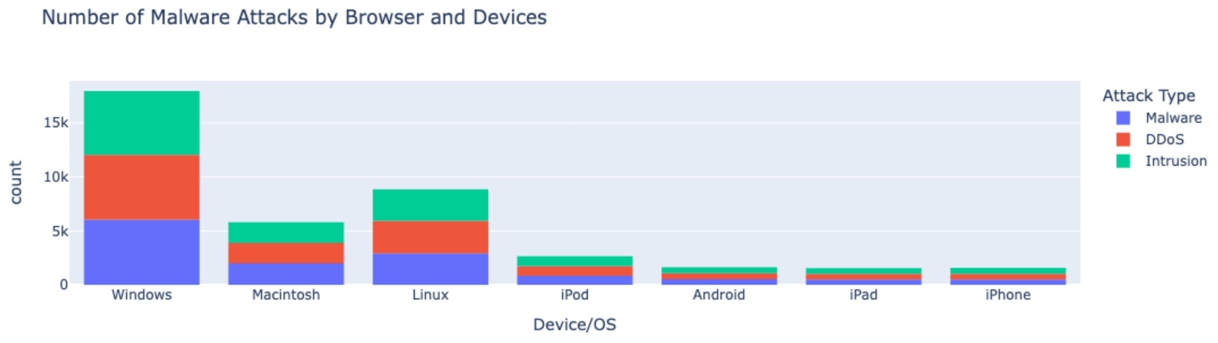
```
df['Action Taken'].value_counts()

Action Taken
Blocked    13529
Ignored    13276
Logged     13195
Name: count, dtype: int64

plt = px.histogram(df, x='Malware Indicators',color='Attack Type',
title='Number of Attack Types by Malware Indicators')
plt.show()
```



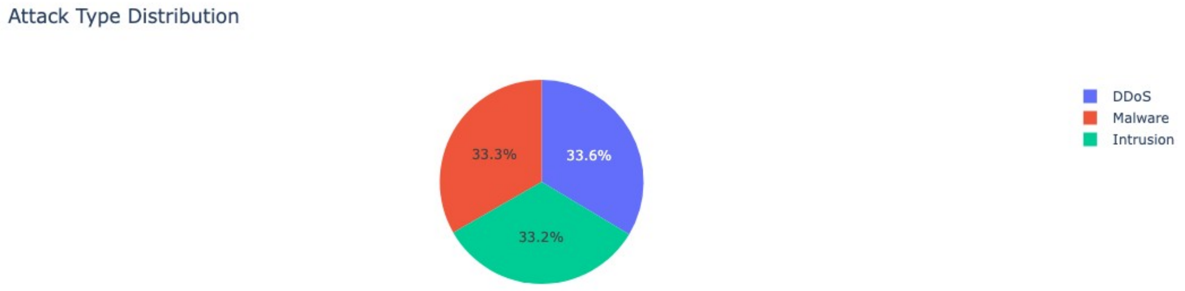
```
df['Malware Indicators'].value_counts()

Malware Indicators
IoC Detected    20000
No Detection    20000
Name: count, dtype: int64

df.groupby(['Malware Indicators', 'Attack Type']).agg({'Packet
Length':'mean'})

                                Packet Length
Malware Indicators Attack Type
IoC Detected       DDoS            780.819958
                   Intrusion       787.093473
                   Malware         777.033678
No Detection       DDoS            789.797313
                   Intrusion       774.694545
                   Malware         779.070631

df.groupby(['Malware Indicators'])['Attack Type'].value_counts()

Malware Indicators  Attack Type
IoC Detected        Malware        6681
                    Intrusion      6665
                    DDoS           6654
No Detection        DDoS           6774
                    Malware        6626
                    Intrusion      6600
Name: count, dtype: int64

plt = px.histogram(df, x='Attack Type', color='Malware Indicators',
title='Number of Attack Types by Malware Indicators')
plt.show()
```



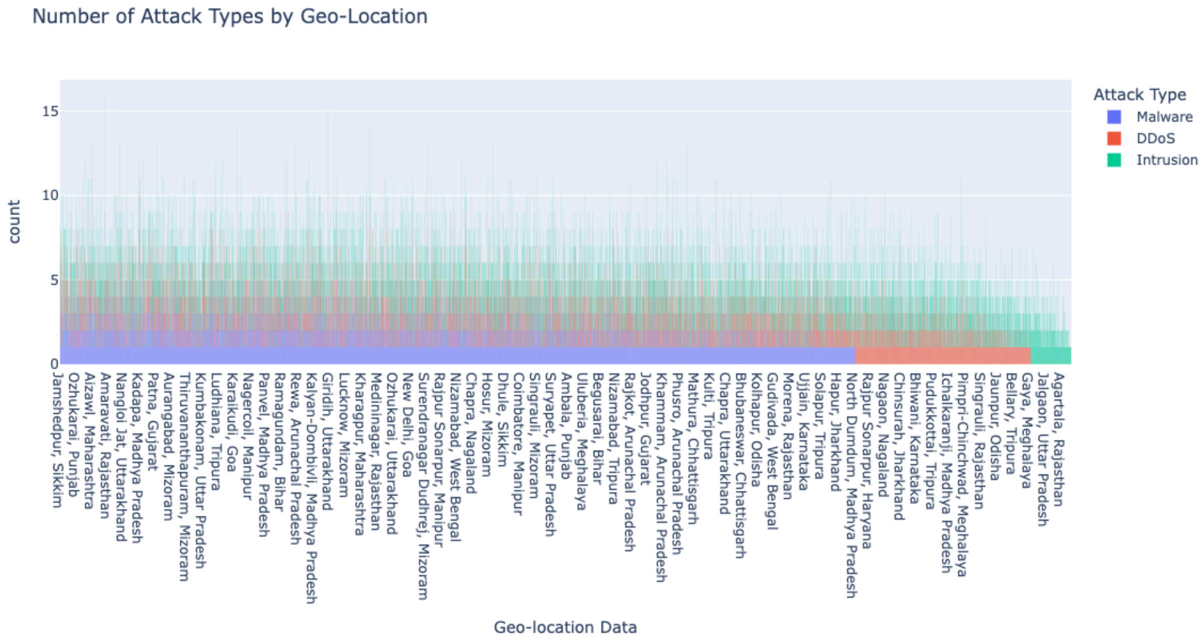
```
plt = px.histogram(df, x= 'Device/OS', color = 'Attack Type', title =
'Number of Malware Attacks by Browser and Devices')
plt.show()
```



```
plt = px.pie(df, names = 'Attack Type', title = 'Attack Type
Distribution')
plt.show()
```



```
plt = px.histogram(df ,x='Geo-location Data', color = 'Attack Type',
title = 'Number of Attack Types by Geo-Location',width=1000,
                        height=600)
plt.show()
```



```
geo = df.groupby(['Geo-location Data'])['Attack Type'].value_counts()
geo.sort_values(ascending=False).head(20)

Geo-location Data                Attack Type
Ghaziabad, Jharkhand             Intrusion      10
Aligarh, Chhattisgarh            Malware         9
Aurangabad, Nagaland             Malware         9
Srikakulam, Uttarakhand          Intrusion       8
Yamunanagar, Arunachal Pradesh   Malware         8
Rampur, Gujarat                  Intrusion       8
Jalgaon, Mizoram                 Malware         8
Amroha, Sikkim                   Intrusion       8
Panvel, Jharkhand                Intrusion       8
Kochi, Tamil Nadu                DDoS            7
Ghaziabad, Meghalaya             Intrusion       7
Tenali, Madhya Pradesh           Malware         7
Ghaziabad, Nagaland              Malware         7
Pimpri-Chinchwad, Manipur        DDoS            7
Karnal, Tamil Nadu               Intrusion       7
Kottayam, Nagaland               Malware         7
Junagadh, Telangana              DDoS            7
Kadapa, Mizoram                  Intrusion       7
Fatehpur, Gujarat                DDoS            7
Hospet, Gujarat                  Malware         7
Name: count, dtype: int64
```