In [2]:

```
# K-Means Clustering

# Importing the libraries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [11]:

```
dataset = pd.read_csv('/content/Malware dataset.csv.zip')
dataset
```

Out[11]:

	hash	millisecond	classification	state	usage_counter	prio	sta
0	42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...	0	malware	0	0	3069378560	
1	42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...	1	malware	0	0	3069378560	
2	42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...	2	malware	0	0	3069378560	
3	42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...	3	malware	0	0	3069378560	
4	42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914...	4	malware	0	0	3069378560	
...	...	...	...	...	...	...	
99995	025c63d266e05d9e3bd57dd9ebd0abe904616f569fe4e2...	995	malware	4096	0	3070148608	
99996	025c63d266e05d9e3bd57dd9ebd0abe904616f569fe4e2...	996	malware	4096	0	3070148608	
99997	025c63d266e05d9e3bd57dd9ebd0abe904616f569fe4e2...	997	malware	4096	0	3070148608	
99998	025c63d266e05d9e3bd57dd9ebd0abe904616f569fe4e2...	998	malware	4096	0	3070148608	
99999	025c63d266e05d9e3bd57dd9ebd0abe904616f569fe4e2...	999	malware	4096	0	3070148608	

100000 rows × 35 columns

In [13]:

```
X = dataset.iloc[:,[3,10]].values
X
```

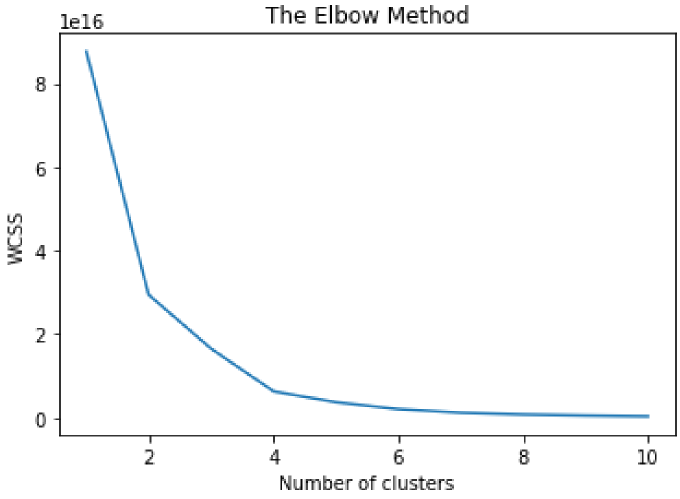
Out[13]:

```
array([[    0, 13173],
       [    0, 13173],
       [    0, 13173],
       ...,
       [ 4096, 10406],
       [ 4096, 10406],
       [ 4096, 10406]])
```

In [15]:

```
# Using the elbow method to find the optimal number of clusters
#The WCSS ( or Within Cluster Sum of Squares ) was caluated and plotted to find the optim
al number of clusters. The "Elbow Method" was used to find the optimal number of clusters
.

#Once the optimal number of clusters were found the model was reinitialised with the n_clu
ster arguments begin passed with the optimal number of clusters found using the "Elbow Me
thod".

from sklearn.cluster import KMeans
wcss =[]
```

```
for i in range (1,11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter =300, n_init = 10, random_state = 0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)

# Plot the graph to visualize the Elbow Method to find the optimal number of cluster
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

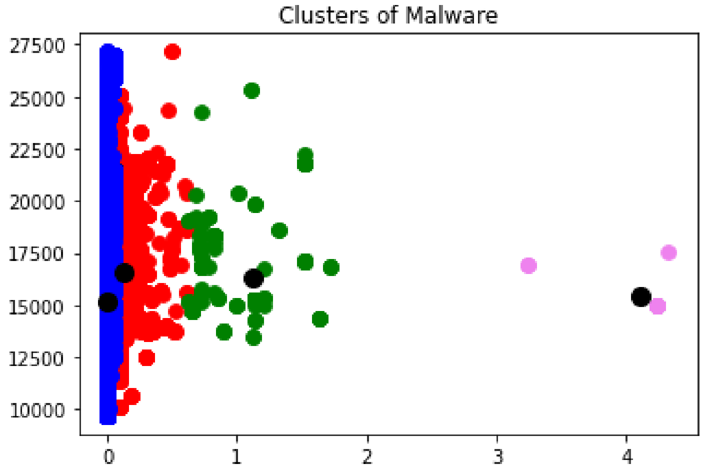


In [16]:

```
# Applying KMeans to the dataset with the optimal number of cluster

kmeans=KMeans(n_clusters= 4, init = 'k-means++', max_iter = 300, n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(X)
```

In [24]:

```
# Visualising the clusters

plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 60, c = 'red', label = 'Cluster1')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 60, c = 'blue', label = 'Cluster2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 60, c = 'green', label = 'Cluster3')
plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 60, c = 'violet', label = 'Cluster4')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 100, c = 'black', label = 'Centroids')
plt.title('Clusters of Malware')

plt.show()
```



1e7