Malware attacks are captured in the form of trace data which is a sequence of action. An example of a trace is read File A, read file B, write on file A …..The goal is to put the traces in groups (malware families) based on similarity.

For our work and making simple to understand, let say we have 3 traces, T1,T2,T3

T1 = a1,a2,a3,a1,a3,a2.

T2= a1,a4,a3,a1,a2,a1,a2.

T3= a1,a4,a2,a3,a2,a1,a2,a3.

The idea is to compare each traces with other. The comparison is performed in this way.

Say, we want to compare T1 and T2.

1. We take the union of trace actions which is a set of a1,a2,a3,a4.

2. Now we construct a directed complete graph nodes – a1,a2,a3 and a4 and each edge is labeled with 1 as in the figure below. From the directed graph, we need to build representation of T1 and T2. The representation of T1 is constructed in the following way- Compute the occurrences of each pair of trace action, for example, (a1,a2),(a2,a3),(a3,a1),(a1,a3),(a3,a2) occurs once. On the edges of the graph, add the occurrences of each pair to 1. Call it GT1. Similarly, construct GT2 (the representative graph for T2)- Compute the occurrences of the pairs which are (a1,a4) occurs 1, (a4,a3) occurs 1,(a1,a2) occurs 2 and (a2,a1) occurs 1. Add the occurrences to 1 on the edges of the graph below. The new graph will be GT2.

3. Now weight the values of each node in each of the graphs, GT1 and GT2 in the following way. For example, for a node a1, if the edge weights to a2,a3 and a4 are 1,2,1,1. The new weights will be 1/4, 2/4,1/4 and 1/4. Similarly do it for all the nodes of GT1 and GT2, Once you have computed the new weights and updated,

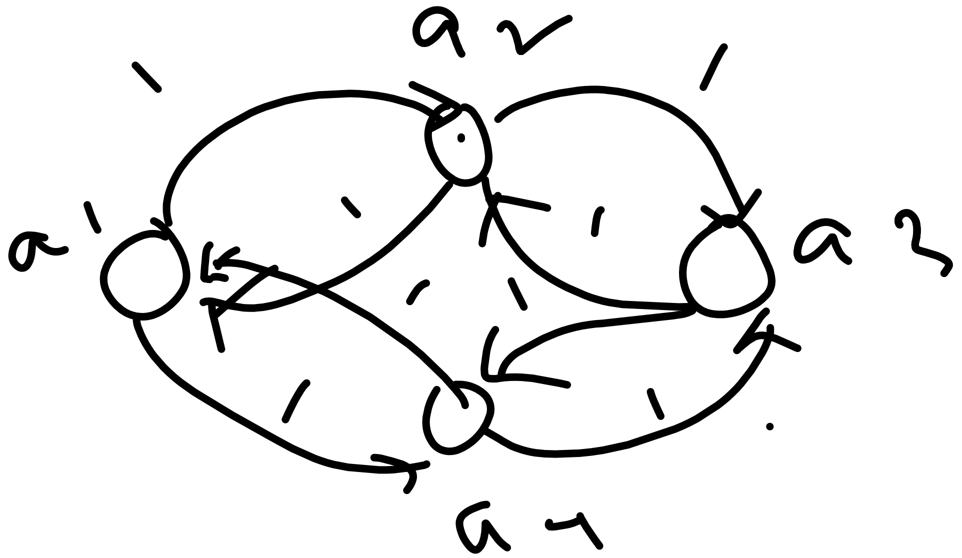
4. Compute the similarity metric… Kullback-Leibler divergence, Jansen-Shannon, Wasserstein metric between GT1 and GT2. It is a formula to compute how dissimilar are the two graphical structures.

https://towardsdatascience.com/using-statistical-distance-metrics-for-machine-learning-observability-4c874cded78.

For T3 and T2, Repeat the above from step 1.

5. The output should be a directed completed graph with nodes as trace, T1,T2 and T3 with edge values as similarity metric..(pick any one of them for the time being)

Eventually, we want to compute similarities for hundreds of traces and create a (similarity) directed graph where each node become a trace and edge weights are the similarity distance KLD value or JSD or Wasserstein metric. You can store the graph in matrix form or dictionary etc..however you will comfortable.


a2
1
1
a1
1
1
1
a3
1
1
1
1
a4