NS Project 2

309551064 張凱翔

1. What model or algorithm do you use?

I use rule-based models to finish this project.

2. What features/rules you used for your model?

Features:

destination port and destination ip

Rules:

I calculate four kinds of information for every log files:

1. destination port = 3389 count
2. destination port = 22 count
3. different destination port count
4. different destination ip count

After calculating the information mentioned above, I will divided them by the length of log file to get the proportion of each information which prevents the longer file have larger count.

Then, I recognize the five attack scenarios ‘RDP Brute-Force’, ‘DDos’, ‘Port Scan’, ‘IP Scan’ and ‘C&C’ in order. The ‘RDP Brute-Force’ is the one that information 1 is largest. The ‘DDos’ is the one that information 2 is largest. The ‘Port Scan’ is the one that information 3 is largest. The ‘IP Scan’ is the one that information 4 is largest. And the last one that has not been assigned is ‘C&C’. It is worth noting that I will delete one index after it has been assigned to one attack scenario to prevent re-assigned.

3. Why do you select them? Please describes as much detail as possible

The reason why I select these features is because after understanding each attack scenario, I think each attack can be recognized by one major features. ‘RDP Brute-Force’ focuses on the port 3389, so the count of ‘destination port = 3389’ would be the largest for sure. ‘DDos’ focuses on the port 22, so the same can be seen that the count of ‘destination port = 22’ would be the largest. ‘Port Scan’ and ‘IP Scan’ focus on sending requests to different ports and ips, so the numbers of different ports and ips can be used to recognize these two attack scenarios. However, there will be one situation that the ‘DDos’ attack also has large counts of different ips. Therefore, I first use the count of ‘destination port = 22’ to recognize ‘DDos’ to prevent it be recognized as ‘IP Scan’.

4. Anything interesting you find or problems you encounter in the whole process

The biggest problem is reading the large json files. The memory is not enough to load all the json files all together, and this problem costs me a lot of time. After trying many methods, I load one file at a time and flush the variable when I get all the information I want, then load the next file. By this method, I successfully read all the files when execute my code.