**Title: Malware Memory Analysis for Intrusion Detection**

1. **INTRODUCTION:**

The dataset used is Malware Memory Analysis for Intrusion Detection. Malware Memory Analysis is the method which is used to determine whether the memory is corrupted by malware or not. It provides us a way to detect and prevent potential cyber-attacks.

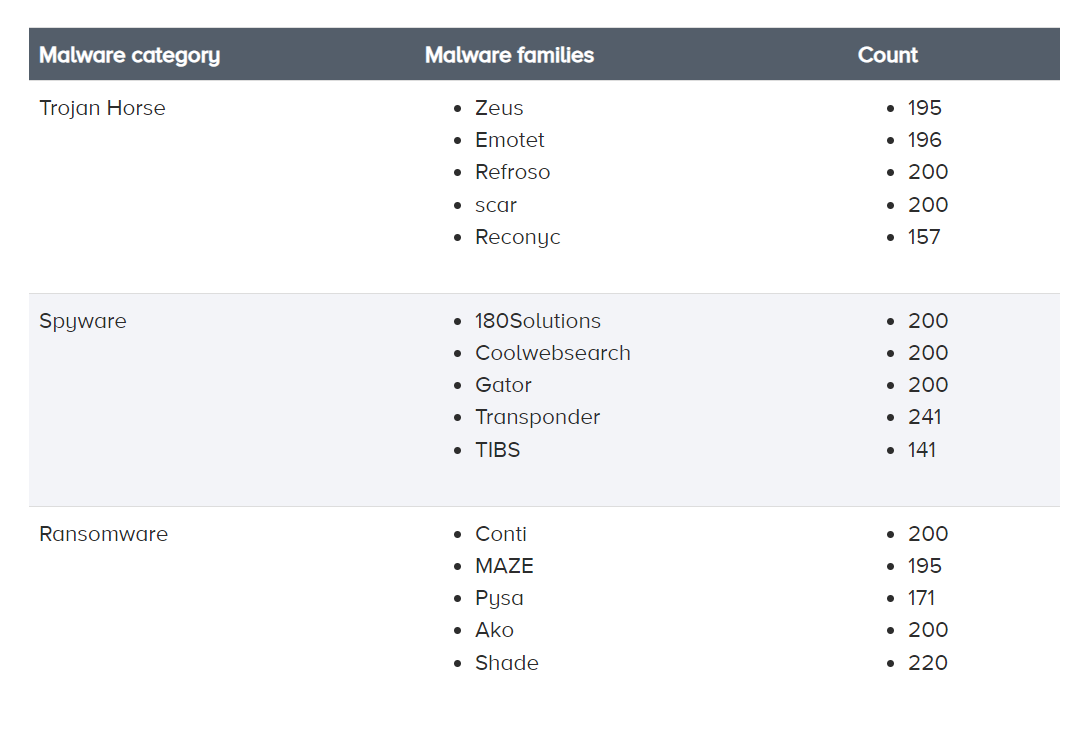
Obfuscated malware is malware that hides to avoid detection and extermination. The obfuscated malware dataset is designed to test obfuscated malware detection methods through memory. The dataset was created to represent as close to a real-world situation as possible using malware that is prevalent in the real world. Made up of Spyware, Ransomware and Trojan Horse malware, it provides a balanced dataset that can be used to test obfuscated malware detection systems.

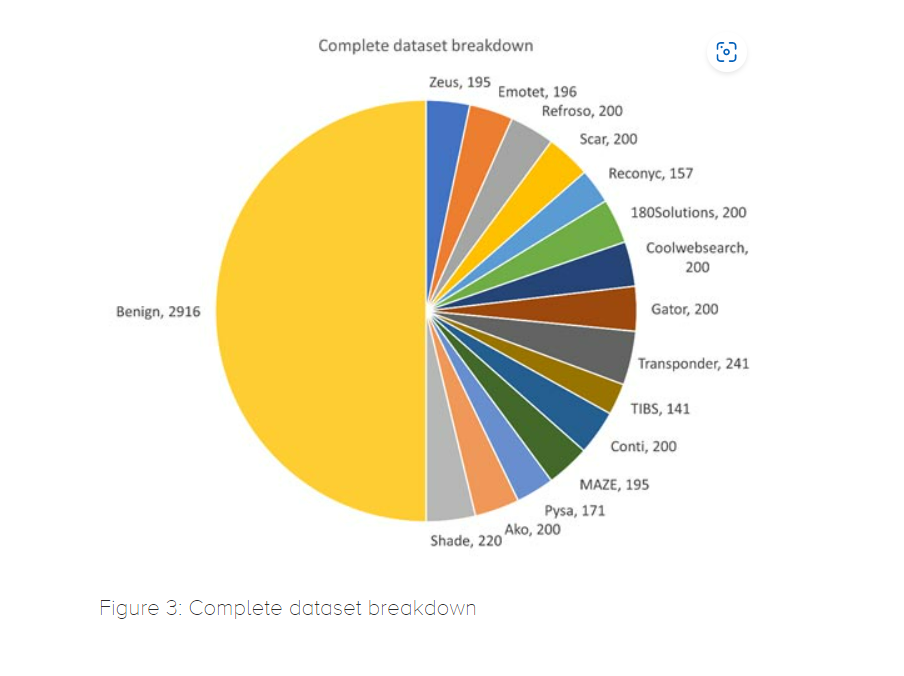
This dataset uses debug mode for the memory dump process to avoid the dumping process to show up in the memory dumps. This works to represent a more accurate example of what an average user would have running at the time of a malware attack.

**2. Dataset details**

The dataset is balanced with it being made up by 50% malicious memory dumps and 50% benign memory dumps. The break down for malware families is shown in the table below. The dataset contains a total of 58,596 records with 29,298 benign and 29,298 malicious. Figure 4 shows the total count of each malware family from each malware category.

Link: http://insideairbnb.com/get-the-data.html





Problem Statement:

1. Malicious and obfuscated malware are specifically designed to evade traditional detection methods and security measures. They employ sophisticated techniques to disguise their code, behavior, or appearance, making it difficult for antivirus software and security systems to identify and block them effectively.
2. Malicious malware aims to remain undetected and persist on compromised systems for as long as possible. It employs techniques such as rootkit installation, process injection, or hiding in system processes to avoid detection and removal by security software.
3. Malware, especially when obfuscated, can facilitate data breaches and unauthorized access to sensitive information. It can steal personal data, financial credentials, intellectual property, or compromise confidential corporate data, leading to financial losses, reputation damage, and legal implications.

Data Analysis:

Here is an analysis of some of the features present in the dataset:

1. Process-related Features:
   * pslist.nproc: The number of processes in the system.
   * pslist.nppid: The number of parent processes for each process.
   * pslist.avg\_threads: The average number of threads per process.
   * pslist.nprocs64bit: The number of 64-bit processes in the system.
   * pslist.avg\_handlers: The average number of handlers per process.
2. DLL-related Features:
   * dlllist.ndlls: The number of DLLs (Dynamic Link Libraries) loaded in the processes.
   * dlllist.avg\_dlls\_per\_proc: The average number of DLLs loaded per process.
3. Handles-related Features:
   * handles.nhandles: The total number of handles in the system.
   * handles.avg\_handles\_per\_proc: The average number of handles per process.
   * handles.nport: The number of port handles.
   * handles.nfile: The number of file handles.
   * handles.nevent: The number of event handles.
   * handles.ndesktop: The number of desktop handles.
   * handles.nkey: The number of registry key handles.
   * handles.nthread: The number of thread handles.
   * handles.ndirectory: The number of directory handles.
   * handles.nsemaphore: The number of semaphore handles.
   * handles.ntimer: The number of timer handles.
   * handles.nsection: The number of section handles.
   * handles.nmutant: The number of mutant handles.
4. LdrModules-related Features:
   * ldrmodules.not\_in\_load: The number of modules not in the load state.
   * ldrmodules.not\_in\_init: The number of modules not in the init state.
   * ldrmodules.not\_in\_mem: The number of modules not in the memory state.
   * ldrmodules.not\_in\_load\_avg: The average number of modules not in the load state per process.
   * ldrmodules.not\_in\_init\_avg: The average number of modules not in the init state per process.
   * ldrmodules.not\_in\_mem\_avg: The average number of modules not in the memory state per process.
5. Malfind-related Features:
   * malfind.ninjections: The number of memory injections detected.
   * malfind.commitCharge: The commit charge of the injected memory.
   * malfind.protection: The protection level of the injected memory.
   * malfind.uniqueInjections: The number of unique injections detected.
6. Psxview-related Features:
   * psxview.not\_in\_pslist: The number of processes not found in the pslist.
   * psxview.not\_in\_eprocess\_pool: The number of processes not found in the eprocess pool.
   * psxview.not\_in\_ethread\_pool: The number of processes not found in the ethread pool.
   * psxview.not\_in\_pspcid\_list: The number of processes not found in the pspcid list.
   * psxview.not\_in\_csrss\_handles: The number of processes not found in the csrss handles.
   * psxview.not\_in\_session: The number of processes not found in the session.
   * psxview.not\_in\_deskthrd: The number of processes not found in the deskthrd.
   * psxview.not\_in\_pslist\_false\_avg: The average number of processes not found in the pslist per process.
   * psxview.not\_in\_ep

After removing the outliers:

Accuracy of logistic regression model is: 0.999877541023757

Data set is balanced so we use accuracy matrix.

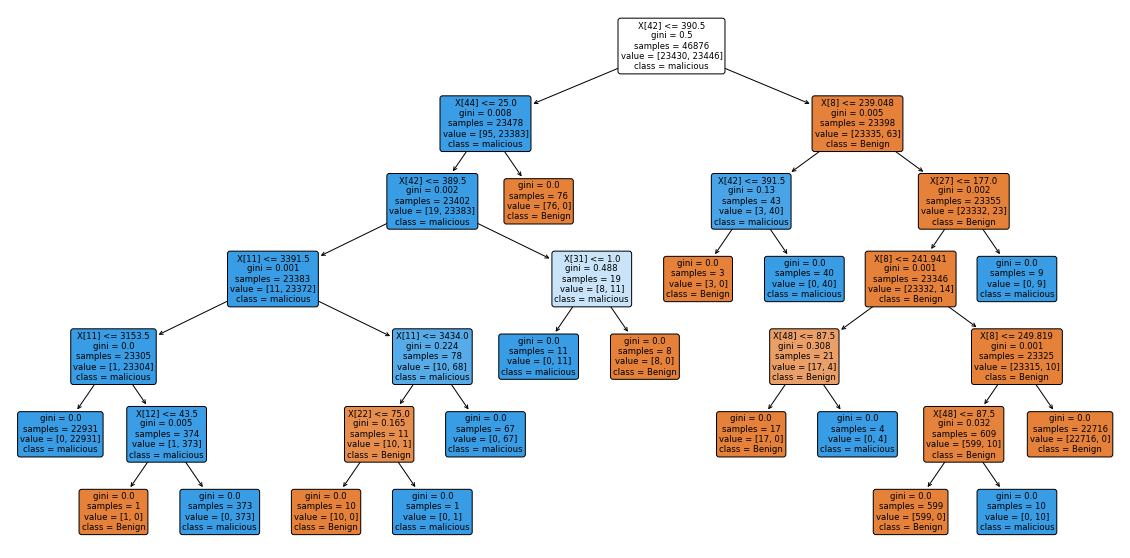
Accuracy of decision tree model is: 0.9997550820475141

No of decision tree node = 1

Accuracy of SVM is: 0.999877541023757

Without removing outlier:

No of decision tree node



Accuracy of decision tree = 0.9997440273037542