Malware Visualization

Project Proposal

By

JAY PATEL (jmp840)
SHAILESH CHAUDHARY (ssc496)
RAHUL KALA (rsk430)

}

What is the problem you want to solve and who has this problem?

The reports generated during analysis of malware consist of large number of API and system process calls. If we manually analyze this report it's difficult to keep track of malware activities and finding patterns which makes it difficult to categorize the malware. These identifications is very much required by researches and the anti-virus manufacturer in-order to identify the newer version of them.

What questions do you want to be able to answer with your visualization?

Questions to be answered from Visualization:

- ➔ What is the behavior of malware in system?
- ➔ If this malware is a member of any existing family?

Both questions can be answered using the same kind of visualization. When malware infects any system it makes many standard system calls. These call remains almost same with new versions of the malware. Thus we need to visualize the system state after malware infection to see the pattern of malware activity and identify its family.

What is your data about? Where does it come from? What attributes are you going to use? What is their meaning? What are their attribute types (data abstraction)? Do you plan to generate derived attributes? If yes, which and why?

Our data is log entries of system calls. For each system call for any operation a log entry is generated. Each log entry has process id and name along with different types of attributes associated with it depending on type of system call.

e.g. :

```
pc: 2001563826
instr: 419428553
nt_create_file {
  proc {
    pid: 776
    name: "cmd.exe"
  }
  filename: "\\??\\D:\\bbc03a5638e801a09015eade0f496103.exe"
}
```

We took data from Prof. Brendon. The data is in plog file. Then we run a script to parse the data and converted the data in csv format which is easier to parse in d3. We will use process ids, type of system call, instruction id as timestamp, process name.

Attribute Name	Attribute Type	Description	Type / Value	Derived?
instr	Ordinal	An instruction number based on the time stamp.	Long integer E.g. 4850081937	No
call_name	Categorical	Type of system call	String E.g. nt_create_file	No
pid	Categorical	Unique identification number for a process	Integer E.g. 828	No
name	Categorical	Name of the process	String E.g. cmd.exe	No
new_pid	Categorical	Process id for newly created process	Integer E.g. 901	No
new_name	Categorical	Name of the newly created process	String E.g. paint.exe	No
file_name	Categorical	Name of the file to be created, read or written	String E.g. test.txt	No
keyname	Categorical	Name of the registry key to be read or written	String E.g. PCIIDE\IDECHANNE L\4&3084357F&0&1\ Device Parameters	No
target_pid	Categorical	The process id to which other process id will write data.	Integer E.g. 901	No
target_name	Categorical	Name of the above mentioned process	Integer E.g. 364	No
section_id	Categorical	Id for the section(a memory location shared by multiple processes)	String E.g. csrss.exe	No
section_name	Categorical	Name for the section	String E.g. \Windows\SharedSect ion	No
port_id	Categorical	Unique id created to be used as port	Integer E.g. 2212816928	No
port_name	Categorical	Name of the port	String	No

			E.g. \ThemeApiPort	
client_pid	Categorical	Client process id which initiates Inter Process Communication(IPS)	Integer E.g. 901	No
client_name	Categorical	Name of the client process	String E.g. cmd.exe	No
server_pid	Categorical	Server process id which acts as a server for IPC	Integer E.g. 1901	No
server_name	Categorical	Name of the server process	String E.g. conhost.exe	No

What have others done to solve this or related problems?

Some researchers have done static analysis in which they used decompiler to gain insight of malware infected code and predict the behaviour of the code. While many are doing dynamic analysis to see the behaviour of malware by running the code and plot the same using different visualization technique. The most popular techniques are Behavioural Image, API Calls Density cluster, Malware Tree map and Malware tree thread graph.

We took reference from the following research papers:

Visual Analysis of Malware Behavior Using Treemaps and Thread Graphs
Philipp Trinius, Thorsten Holz, Jan Gobel and Felix C. Freiling

- This paper aims to help human analysts to quickly assess and classify the nature of a new malware sample.
- Here dynamic analysis is done on a malware sample by executing it in a sandbox environment to analyze the behaviour of malware.
- A sandbox executes a malware sample in a controlled environment and records all system-level behavior such as modifications of the filesystem or the registry. As a result, the sandbox generates an analysis report summarizing the observed behavior of the sample.
- The paper introduces two visualization techniques: treemap and thread graphs. Treemap displays the distribution of the individual operations performed by a sample. The resulting treemap presents this information as a set of nested rectangles and provides a quick overview of the main overall behavior of the sample, e.g., whether the main task of the sample lies in the area of network interaction, changes to the file system, or interaction with other processes. Since tree maps display nothing about the sequence of operations, we use Thread Graphs to visualize the temporal behavior of the individual

threads of a sample. A thread graph can be regarded as a behavioral fingerprint of the sample in which operations are recorded with time as log entries and plotted on graph. An analyst can then study this behavior graph to quickly learn more about the actions of each individual thread.

Visualization Techniques for Malware Behavior Analysis
André R. A. Grégio and Rafael D. C. Santosc

- In this paper, authors did analysis based on the behavior of malwares to understand them. They used dynamic analysis in order to record its behavior.
- To visualize the data obtained by analysing various malware, clustering technique was used where the vertices are the individual malware and the edges are based on the associated weight which is calculated based on the degree of agreement among antiviruses.
- They also used Thread Graph technique to represent the activities of malware and to track what new process do they run and create.
- Based on the information obtained from the graphs, analyst can understand what actually happened in the compromised system and can compare these studies with those of new malware to see whether they behave in a similar fashion or not.

Malware Images: Visualization and Automatic Classification
L. Nataraj, S. Karthikeyan, G. Jacob and B. S. Manjunath

- This paper uses the static analysis method i.e., without running the malware in virtual machine but studying malware binaries.
- They used image processing technique to visualize the malware binaries as gray-scale images.
- The gray-scale images produced for different malware remains almost similar if malwares are from same family.
- They tested about 9000 samples with 25 different malware families and had classification accuracy of 98%.

Malware analysis using visualized images and entropy graphs
Kyoung Soo Han, Jae Hyun Lim, Boojoong Kang and Eul Gyu Im

- Malware are created using some automated tools and methods may reuse some modules to develop malware variants which can be used to classify malware and identify malware families. This paper classifies malwares based on converting binary files into images and entropy graphs. They allow malware researchers to understand the structures of malware binary files without disassembling and to make a decision for applying of unpacking.
- They uses Bitmap Image Converter which receives Windows Portable Executable binary files as input and converts binary files into bitmap images. The Entropy Graph Generator calculates the entropy value of each line of bitmap image and generates entropy graph based on these values. These entropy values and graphs are stored in database and used to classify and detect malware based on similarities.

- They had a limitation when no particular pattern can be identified in gray-scale images and entropy value of binaries can be very high.

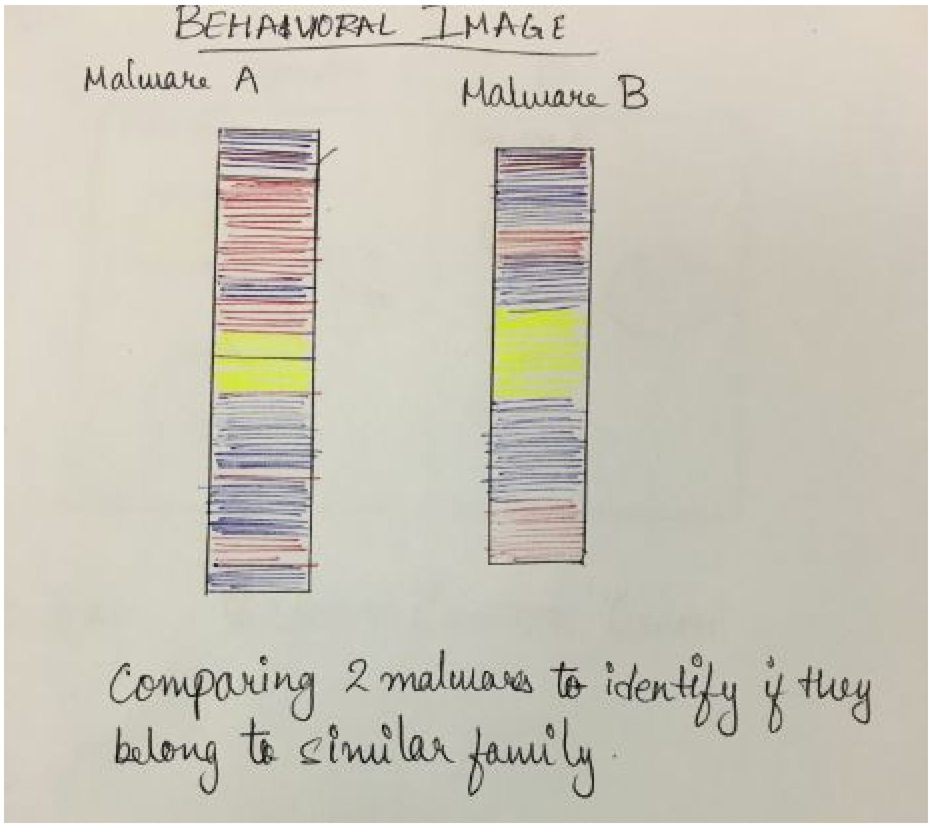
Malware Behavior Image for Malware Variant Identification
Syed Zainudeen Mohd Shaid, Mohd Aizaini Maarof

- This paper talks about mapping malware behaviour using color in behaviour image.
- A color map is a map which ranges from red(hot) to blue(cold). Hot colors are used to represent malicious API calls while cold colors are used to represent non malicious API calls. Calls are grouped together as malicious, less malicious , non malicious, etc. Each API call is represented by 64*4 pixel colored rectangle and the color is determined by the type of call.
- Finally we get a Behaviour Image for the malware. When we plot these images for malwares of the same family, we can see that they are nearly similar.

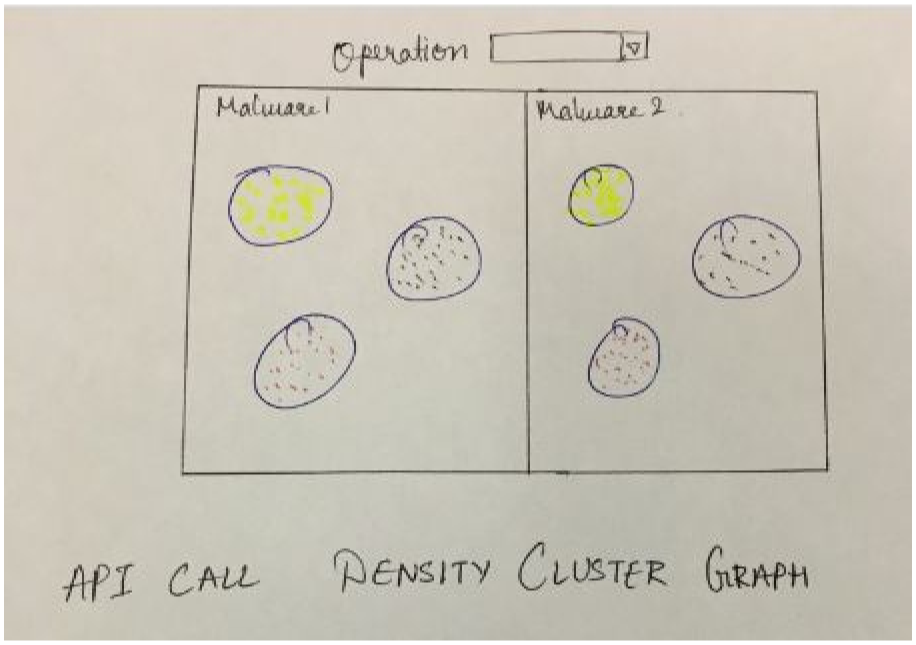
What solution do you propose? How does the solution help you answer the questions stated above?

We will create the below visualizations to answer the questions.

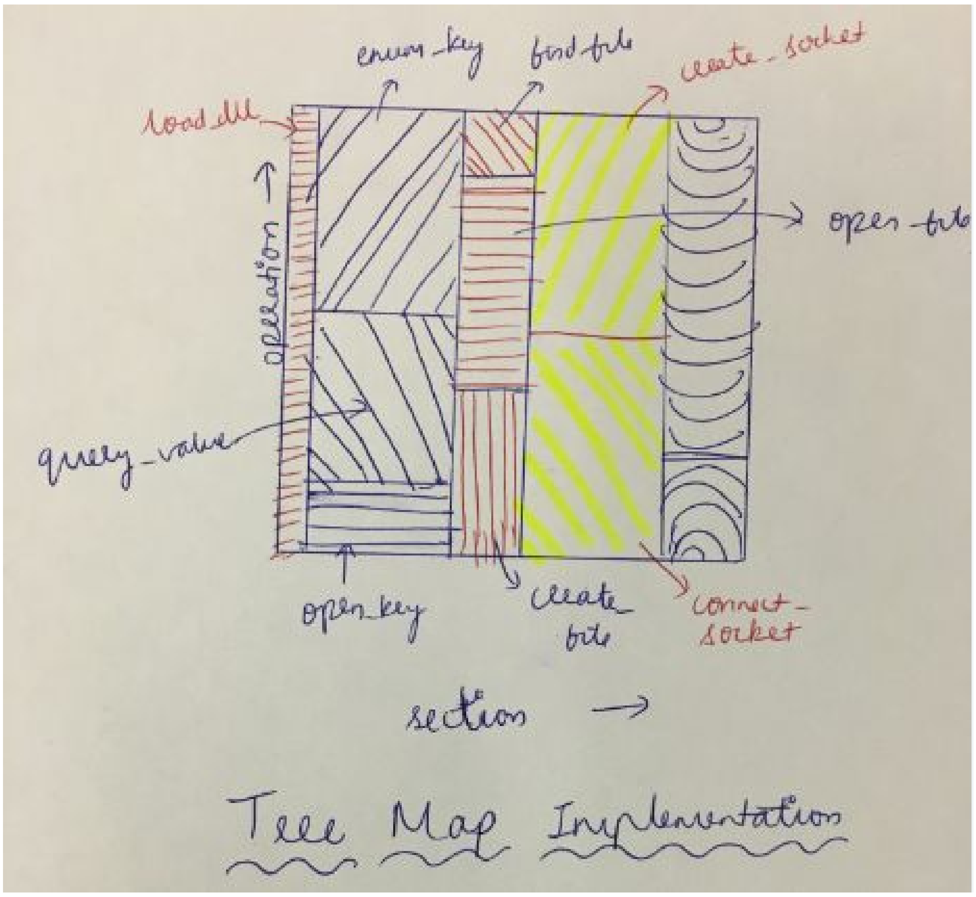
Behavioural Image: This visualization shows the sequence of API calls in the system and based on the maliciousness of the call they are represented by different color with the help of color map. Now if we look at the diagram we see that the sequence of calls are not similar. That means, the two malware are not from the same family.


BEHAVIORAL IMAGE
Malware A
Malware B
Comparing 2 malwares to identify if they belong to similar family.

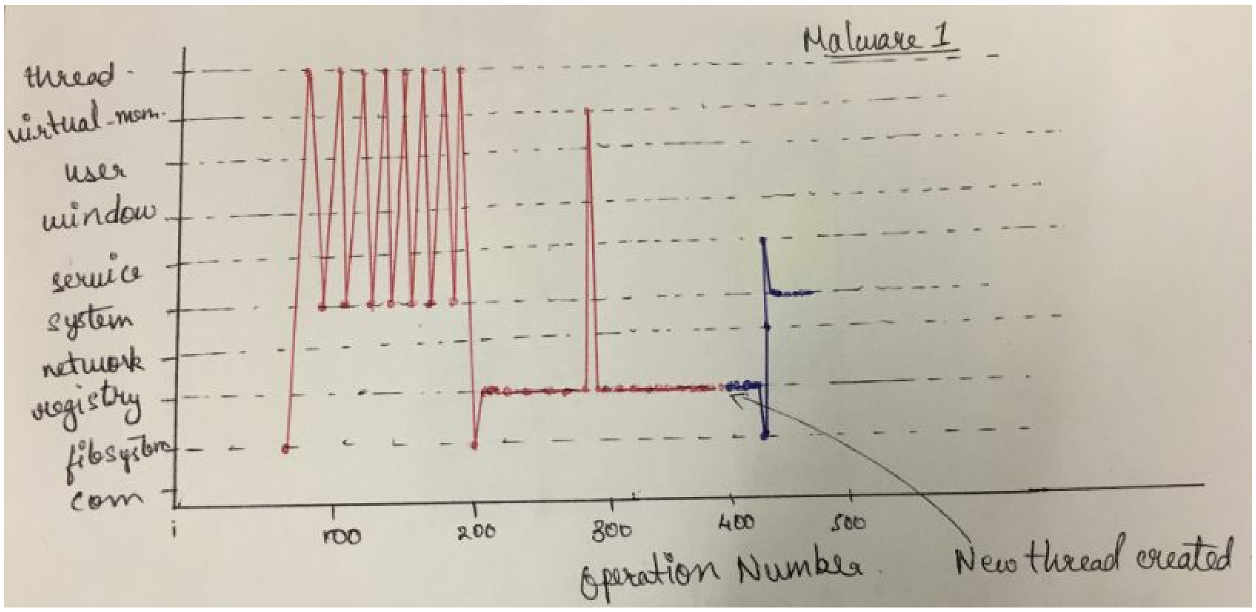
<u>Density</u> <u>Cluster:</u> API Call Density cluster used to represent the overall behaviour of the malware and we can compare two malware in it. The malwares can be compared at different levels of operations or can be compared using all the operations. If the clusters formed are exactly similar implies that the two malwares belong to the same family or with little variations, a new version of the existing malware is found.



Tree Map: In this visualization, a map is divided into horizontal sections. Each section represent a specific group of API call such as file, socket or key operation calls. Inside each section there are vertical sections created which represents specific call for that group like file_create, file_delete or file_read. If dynamic analysis of two malware produces same kind of tree map then they belong to same family or one malware can be the new version of other malware.



Thread Graph: The Threaded graph represents all the activities done by the malware during its execution in the virtual system. The graph also displays the new processes created by the one we are monitoring.



How do you plan to verify whether you have met your goals with this project?

We will consult Prof. Brendan every week to check whether we are moving in right direction to meet our goal. We will ask for some more data of same family malwares to confirm whether our visualization approach gives correct results.

How is your project team going to work on the project? Who is going to do what?

<u>Combined Work:</u>
We collected raw data and some scripts from Professor Brendan to understand them completely. We will implement malware threaded graph. Most of the work will be done under the supervision of Prof. Enrico Bertini and Prof. Brendan.

<u>Jay:</u>
He wrote a python script to extract meaningful data from raw data and save into a csv file. He will guide the team as a team leader.

<u>Shailesh:</u>
He along with Rahul read research papers and surfed internet to find all the visualization techniques that have been implemented for our problem and decide what our team should use.
He is creating the report for the project. He is planning to implement Behavioural image and clustered image.

<u>Rahul:</u>
He read research papers and surfed internet to find visualization techniques. Also, he made the final diagrams along with other team mates for our proposal. He is planning to create malware Tree map.