Journal of Computer Science

Original Research Paper

Concordance and Term Frequency in Analyzing API Calls for Malware Behavior Detection

[1]Nur Hilda Amira Abd Wahab, [1]Masnizah Mohd, [1]Ravie Chandren Muniyandi, [2]Balaji Rajendran and [2]Gopinath Palaniappan

[1]Center for Cyber Security, Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia
[2]Centre for Development of Advanced Computing, Bangalore, India

Article history
Received: 10-06-2019
Revised: 31-07-2019
Accepted: 24-09-2019

Corresponding Author:
Masnizah Mohd
Center for Cyber Security, Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia
Email: masnizah.mohd@ukm.edu.my

Abstract: Application Programming Interface (API) is used for the software to interact with an operating system to do certain task such as opening file, deleting file and many more. Programmers use this API to make it easier for their program to communicate with the operating system without having the knowledge of the hardware of the target system. Malware author is an attacker that may belong to an organization or work for themselves. Some malware author has the capabilities to write their own malware, uses the same kind of APIs that is used to create normal programs to create malware. There are many researches done in this field, however, most researchers used n-gram to detect the sequence of API calls and although it gave good results, it is time consuming to process through all the output. This is the reason why this paper proposed to use Concordance to search for the API call sequence of a malware because it uses KWIC (Key Word in Context), thus only displayed the output based on the queried keyword. After that, Term Frequency (TF) is used to search for the most commonly used APIs in the dataset. The results of the experiment show that concordance can be used to search for API call sequence as we manage to identify six malicious behaviors (Install Itself at Startup, Enumerate All Process, Privilege Escalation, Terminate Process, Process Hollowing and Ant debugging) using this method. And based on the TF score, the most commonly used API in the dataset is the Reg Close Key (TF: 1.388), which on its own is not a dangerous API, hence we can infer that most API is not malicious in nature, it is how they were implemented is making them dangerous.

Keywords: Concordance, KWIC, API Call Sequence, Malware Behaviors, Dynamic Analysis

Introduction

Nowadays, with a new variant of malware being discovered, we can see that malware is becoming more sophisticated in design. On top of that, malware is also becoming more notorious over time, with the rising amount of security breaches around the world as well as the severity of said breaches. According to Cisco (2018), security breaches can cause significant economic damages to an organization as it takes considerable time to fix the damages done. Moreover, more than half of the breaches cost more than $500,000 in financial damages. This shows how severe it is the effect of the malware attack on an organization. Take example the WannaCry ransomware outbreak in 2017 which shows how dangerous modern malware is. This ransomware affected more than 200 000 computers in over 150 countries worldwide and cause huge financial damages to its victims (Business Advantage, 2017).

Moreover, according to (Ghazvini and Shukur, 2017), security breaches that are caused by human mistakes may lead to costly fixed up as the company may lose the information or data stored in the system as well as their reputation which in turn may affect their market share prices. On the other hand, according to (Manap *et al*., 2015), malware may also be used to steal someone's identity by using keystroke logger and some form of spyware.

Malware is malicious software that can cause harm to our system or network. The consequence of malware attack is not limited to theft of data, destruction of data, system compromise and denial of service (DoS).

DOI: 10.3844/jcssp.2019.1307.1319

Table 1: Malware and their description

Category	Description
Trojan	Trojan is malware that masquerade as a legitimate software. Once runs, the Trojan can spy on the user, delete files, steals sensitive data and many more depending on what that Trojan is for. However, unlike virus and worms, Trojan cannot self-replicate itself.
Adware	Adware is a malware that shows ads on the computer. It is mostly annoying rather than malicious in nature.
Worm	Worm is a malware that can propagate itself within the system. Worm can self-replicate itself using the internet or Computer Network.
Backdoor	Backdoor is a type of access that will allow programmer to bypass the security of the system in order to access the system quickly. However, the attacker also uses the backdoor to gain unauthorized access to the system.
Virus	Virus is a malware that attached itself to a piece of software that will infect system that run that infected software. This malware will usually infect other system when user share that infected files

Table 2: Advantages and disadvantages of different malware detection technique

Technique	Advantage	Disadvantage
Specification	Can detect unknown attack	Too much time needed to write good specification
	Minimal false positive	False negative is increased due to missed attack
Signature	Recognized known attack	Cannot recognize unknown attack
	Require minimum system resources to detect attack mode	Not usable against invisible signature
	Concentrate on normal behaviors	
Anomaly	Able to detect new attack	Must renew user behaviors information periodically
	Focus on normal behaviors to detect unrecognized attacks	Increased false positive

There are many types of malware such as Virus, Trojan, Worm and Backdoor. Moreover, each of these malwares behaves differently from each other. Table 1 shows the different type of malware and their description.

There are 3 types of malware detection technique, namely Specification-based, Signature-based and Anomaly-based. There are advantages and disadvantages of these malware detection techniques which is shown in Table 2 (Mohaddes *et al*., 2016).

Malware analysis gives us an insight on how malware functions, so that we can ensure the safety of our system as well as how to eliminate the danger the malware poses. The analysis can be done with different goals such as understanding the magnitude of the malware infection, knowing the impact of malware attack as well as to analyze the behavior of the malware. There are three types of malware analysis, static analysis, dynamic analysis and hybrid analysis.

Static analysis is done by dissecting and studying the malware's code. It is done without executing the malware. The code is disassemble using a disassembler, or reverse engineered so that malware analyst can read the code and find out what the malware is supposed to do based on their code. Which is why, static analysis is also known as code analysis.

On the other hand, dynamic analysis is done by executing malware in a safe environment, maybe it virtual or sandbox environment. Dynamic analysis focusses on analyzing the behaviors of malware, hence it monitors the activity of registry, API calls, processes and network when running the analysis. Moreover, hybrid analysis uses both static and dynamic analysis to analyze the malware.

The reason why this research experiment used dynamic analysis to extract the API calls of the malware is because static analysis that use only import table to extract the information of the malware will not be able to extract any resolved APIs of the malware. This is because, to extract resolved APIs, we need to run the malware since it can only be extracted during runtime and for static analysis do not execute the malware, they will miss this kind of information.

Moreover, this study proposed to use a concordance to map the API call sequence of a malware in the dataset. The reason for this is because the n - gram method which is used by most researchers displayed a lot of output, as the output is based on the n-value, hence it takes time to browse through it all. Meanwhile, concordance use Key Word In Context (KWIC) method to display its output, meaning the output displayed is only based on the keyed keywords and its close context, thus reducing the time it takes to process the displayed output.

The rest of the paper is organized as follows: Section II will discuss about the related works done in analyzing API calls. In section III, we will discuss the materials and methods used in the experiment. Section IV will discuss about the experimental setup for the research. Section V will discuss about the results and discussion of said results and finally, Section VI will discuss about the conclusion of the experiment and recommendation for future works. The purpose of this research is to identify malware behavior by using API call. This research will focus on:

1. Using dynamic analysis to extract API call and map the API call sequence using the concordance tool
2. To find out the most commonly used APIs in the dataset by using Term Frequency (TF)

DOI: 10.3844/jcssp.2019.1307.1319

On the other hand, the scope of this research is:

1. This research only focuses on Windows malware
2. The dataset used in this experiment is created using resolved APIs
3. There are criteria being imposed on the population and samples used

Related Work

Fujino *et al*. (2015) stated that API call has lots of useful information about the behavior of a malware. They use Bag-of-Words (BoW) models with TF-IDF word weighting to characterizing the API calls. They also use a soft clustering algorithm which is a non-Negative Matrix Factorization (NMF) to extract the API call topics which will be used to detect similar but unknown malware.

Sundarkumar *et al*. (2015) wrote that API level information inside the bytecode is beneficial to analyze software malevolence tendency since it shows the behavior of said executable which the API call sequence of. They also assert that the main problem in using Topic Model is the lots of choices in features, hence why, they propose to apply Latent Dirichl *et al* location (LDA) as a feature selection method in their research.

Ki *et al*. (2015) proposed a novel approach to use a sequence alignment algorithm in the API call sequence analysis. This is because sequence alignment algorithm will make it less confusing when detecting the malware. They also stress that the algorithm has been used in many other fields such as natural language processing and is known to yield excellent results. On top of that, LCS (Longest Common Subsequences) is used to extract the most common API call sequence between the malware which will then be used to generate malware signature.

Pektas and Acarman (2017) proposed classification program that uses Voting Expert algorithms to precisely recognize episode boundaries. Furthermore, the episode boundaries are then used to search the API call sequence. Moreover, they also track the changes made to the OS by malware. They wrote that the API call combines with OS state changes made by the malware will increase the accuracy of malware classification.

Lim (2016) proposed the extraction of malicious behaviors of malware as sets of k-grams, which is then used to compare the similarity between API calls to the k-grams to identify whether it is malicious behavior or not.

Salehi *et al*. (2014) proposed a new model in which API call features are extracted and selected based on their categories. The reason for this is to minimize the number of features of the malware and to reduce analysis time. After that, machine learning classification technique is used for each generated set of features. They wrote that their experiment has high detection rates in distinguishing malware and benign file using only small features.

The differences between (Ki *et al*., 2015) and (Lim, 2016) from this work is that one research uses a novel approach to analyze API call sequence which is to use sequence alignment algorithm and another one-use k-gram to analyze API call sequence to determine whether it is malicious or not. On the other hand, this research use concordance to identify API call sequence of malware's malicious behavior.

Arguably, most of the research done focus on detection of malware, however, research done by Alazab *et al*. (2010) focus on understanding the behavior of malware through statistical analysis of the API call. They use a static analysis to disassemble, analyze and extract the API call from the malware. They then proposed a novel approach to automate the process to extract the API call of the malware binary.

On the other hand, the researcher Belaoued and Mazouzi (2015) also doing a statistical analysis using Multiple Correspondence Analysis (MCA) to find out which APIs association that most likely used in malware. They use static analysis to import IAT so that they can extract the APIs.

This research is similar like research done by both Alazab *et al*. (2010) and Belaoued and Mazouzi (2015) albeit this research will use dynamic analysis instead of static analysis to extract the APIs of the malware. Moreover, this research will focus entirely on identifying malware behavior and their associated APIs.

All the reviews above shows the importance of API call in the understanding and detection of malware. Moreover, we can agree that API calls play a major role in understanding the malware itself, thus helps in identifying both existing and new variants of malware. Likewise, by observing the API calls of a malware, one can see how the malware work because API calls represents a specific operation that is used to run specific tasks. Prior researches have suggested that by observing the API calls of a malware, we can determine the behavior of said malware, as malware behave in specific behavior that differentiate it from benign program. Finding this malicious behavior and their APIs is one of the objectives of this research.

Materials and Methods

Dynamic Analysis

This study uses dynamic analysis to extract the API calls of a malware. Dynamic analysis can be done using two ways, manually or automatically. Manual dynamic analysis means the malware will be executed in a virtual environment and malware analyst will monitor the malware and take note of any changes in the registry or processes caused by the malware. Table 3 shows the list of tools usually used in doing manual dynamic analysis.

On the other hand, automated dynamic analysis is done in a sandbox environment. There are many types of sandbox available for free or commercially. One of the most used free sandboxes is Cuckoo sandbox. Using automated sandbox in analyzing malware is simple and easy as we only must submit the malware sample and wait for it to execute the malware. After that, we can download the reports generated by the sandbox and analyze the malware based on the reports. As mentioned before, there are many automated sandboxes available which is shown in Table 4.

API

Application Programming Interface (API) is a collection of rules that enable the developers to develop system or software for a specific operating system with little knowledge about that particular operating system (Merriam-Webster, 2018). There are two types of APIs that we will encounter during this experiment which is WinAPI and NTAPI.

WinAPI is a normal APIs that is normally used in developing a new Windows program. WinAPI uses suffix in their name such as A, W and Ex. APIs with A and W suffix have the same function but differ in their accepting parameters as one accepts ASCII string while the other accept a wide character string. Meanwhile, APIs with Ex suffix means that it is an API with an updated functionality.

On the other hand, NTAPI is not so commonly used in developing new programs and most of NTAPI is undocumented. NTAPI used prefix in their name such as Nt, Zw, Rtl and Ldr. According to Sikorski and Honig (2012), NTAPI is commonly used by malware author because it offers powerful and stealthier functionalities than it WinAPI counterpart. Since this research is about understanding malware behavior, we need to know which behaviors are considered malicious or suspicious which is shown in Table 5. On the other hand, Table 6 shows the list of malicious behaviors and their API call sequence.

Concordance

Concordance is a list of all examples of the search word or phrase found in a corpus. Concordance usually uses Key Word in Context (KWIC) to display their output. Concordance is usually used in corpus linguistics to analyze the concordance, word frequency, collocate and cluster of a corpus.

According to Bowker (2018) concordance is a corpus analysis method that fetches all occurrences of a queried search pattern in its immediate contexts. Moreover, according to her, it is also possible to sort the concordances, so that it can identify the patterns that might otherwise go undetected. Furthermore, according to Wynne (2008) concordance is a list of patterns or word in a text that is arranged with surrounding words so that the patterns surrounding the keywords can be visually identified. Other researchers that use concordance KWIC in their research are (Brown, 2017), (Jiang and Liu, 2017), (Yılmaz and Soruç, 2015) and (Rockwell, 2018).

Based on the reviewed articles, we can infer that concordance KWIC is mostly used in linguistic field only. Moreover, the consensus of the reviewed articles suggests that concordance with the use of KWIC is a powerful tool to analyze patterns. Hence why this paper proposed to use this method to map malicious API call sequence. There are many concordance tools available for free which is shown on Table 7.

Table 3: Manual dynamic analysis tools

Tool	Description
API monitor	API Monitor is a software that allow user to monitor and control API calls made by application or services in the system
Procmon	Procmon aka Process Monitor is a monitoring tools for Windows system that shows file system, Registry and process or thread activity in real time. It is a combination of a legacy Sysinternal utilities tool, Filemon and Regmon albeit with added enhancement
Process explorer	Process Explorer shows information about DLL processes and handles that have opened or loaded
Regshot	Regshot is an open-source tools that allows user to take snapshot of the system registry and compare it with second one

Table 4: List of automated sandbox

Sandbox	Description
Anlyz Sandbox	Online free sandbox. User upload the malware file and can download the reports in HTML or PDF format. However, this sandbox only allow 10 submission per account per day.
Cuckoo Sandbox	Free automated open-sourced sandbox. Since the sandbox will be set up on user computer, submission is limitless. However, the installation can be an arduous task.
Joe Sandbox	Joe Sandbox is commercial sandbox. However, user can use it for free albeit with limited functionalities and limited submission per account.

Table 5: List of malware malicious behaviors

Author (year)	Malicious/suspicious behavior
Sundarkumar *et al*. (2015)	• Obtain file directory
	• Search file to infect
	• File Write
	• Modify File Attributes
	• Modify Time of File
	• Distribute Global Memory
	• Distribute Virtual Memory
	• Load Register
Ki *et al*. (2015)	• DLL Injection Using CreateRemoteThread
	• IAT Hooking
	• Antidebugging
	• Screen Capture
Pektas and Acarman (2017)	• Process Hollowing
	• Create Remote Thread
	• Enumerating all Processes
	• Drop file from PE resource section
	• IAT Hooking
	• Delete itself
	• Download and Execute PE file
	• Bind TCP port
	• Capture Network Traffic
Lim (2016)	• Access to files
	• Write files
	• Modify the attributes of files
	• Modify the time of files
	• Move the location of files
Alazab *et al*. (2010)	• Search file to infect
	• Copy/Delete Files
	• Get File Information
	• Move Files
	• Read/Write Files
	• Change File Attributes

Table 6: Malicious behaviors and their API call sequence

Malicious behavior	API call sequence
Modify File Attribute	Get File Attribute, Set File Attribute
Modify Time of File	Get File Time, Set File Time
Load Register	Reg. Create Key, Reg. Set Value, Reg. Close Key
Enumerate all process	Create Tool help 32 Snapshot, Process 32 First, Process 32 Next
	WTS Enumerate Processes
Privilege Escalation	Open Process Token, Lookup Privilege Value A, Adjust Token Privileges
Terminate Process	Terminate Process
	NtT erminate Process

Table 7: Concordance tools

Tool	Description
Ant conc	This is a freeware tool for concordance and text analysis. It is very easy to use, and user have selection of option to choose from such as concordance, collocate, word list, keywords, cluster and n gram.
Word smith Windows user.	This is a tool that can be used to search concord, keyword and word list. However, this tool is only available for
Text STAT	This is a simple program for text analysis. It produces word frequency lists and concordances according to its corpus

Term Frequency

Term Frequency (TF) is used to show how significant it is the terms in the whole documents or collections of counting their number of occurrences (Rajaraman and Ullman, 2011). Moreover, according to AbuHamad and Mohd (2019) TF-IDF can be used to determine the relevance of the retrieved information. TF is usually used alongside Inverse Document Frequency (IDF) which can further determine the significance of said terms. TF can be calculated using the equation below:

$$TF(t) = \frac{C(t)}{N} \quad (1)$$

Where:
T = API call
C = The frequency of API call
N = Total number of API call

Meanwhile, IDF is calculated using the equation below:

$$IDF(t) = \log\left(\frac{ND}{ND_t}\right) \quad (2)$$

Where:
T = API call
ND = Number of malware categories in the dataset
ND_t = Number of malware categories containing the API call

And finally, TF-IDF is calculated using the equation below:

$$TF - IDF(t) = TF(t) * IDF(t) \quad (3)$$

Where:
TF = Term frequency
IDF = Inverse document frequency

As mentioned, TF-IDF is popular among the researchers when doing API call analysis. Among the researchers that use TF-IDF in their research is Sundarkumar *et al*. (2015), Pektas and Acarman (2017), (Altawaier and Tiun, 2016) and (Bai *et al*., 2014).

Malware Samples

Malware samples are used by a malware researcher to research the malware behaviors and techniques use in their creation. The reason for this is so that they can understand how the malware work and come out with the solution on how to better protect the system from the threats. Samples can be collected using honeypots or downloaded from known malicious URL. However, with the blooming of malware research area, there are quite a lot of sites that offered downloadable malware samples for free which is shown in Table 8.

Some words of caution are in order, because the malware on these websites is live malware, it can infect the system or network if not handle carefully. Therefore, the user must ensure proper environment have been set up before beginning to analyze the malware and not to run the malware sample recklessly on their system.

This research is done using quantitative methods, as an experiment will be conducted. The methodology of the research can be seen in Fig. 1. All the steps involved have to be done in order since most of them are dependent on the results of the previous step. There are 4 main steps in the experiment which is:

a. Data Collection
b. Dataset
c. API Mapping
d. Data Analysis

Framework of Implementation

Based on the methodology above, Fig. 2 shows the implementation of the experiment.

Data Collection

The samples used in this experiment is downloaded from Virus Share. Virus Share provides a free live malware sample which can be downloaded in bulk using Torrent.

Dataset

The sample used in the creation of the dataset follow these following rules:

a. Must be a malware file
b. Must be detected by Kaspersky
c. Exclude benign file
d. Exclude malware that is not detected by Kaspersky

Dataset Preparation

During this phase, the malware samples will first submit to Virus Total to ensure that it is in fact a

malware sample and not benign one. The malware sample must also be detected by Kaspersky. This is to make sure the classification of malware is easier if they have same categorization format.

After that, the selected samples will then be submitted to Anlyz Sandbox to do dynamic analysis. Anlyz sandbox provides reports regarding static analysis, dynamic analysis and network analysis of the malware. Then, the generated reports will be downloaded to be used in the next step.

Dataset Creation

The dataset is created using LibreOffice Calc. The reason why this research use LibreOffice Calc is because we use Ubuntu OS during the experimentation. The information such as the malware category (name used by Kaspersky), SHA256 and the resolved API are extracted from the reports. The population of the sample is 524, however, only 182 samples match the criteria imposed on the samples. The categories of the malware in the dataset are shown in Table 9. The author decided to combine the malware categories which have less than 5 samples into one category. There are also few Trojan families such as Trojan-Dropper, Trojan-Banker and Trojan-Ransom which will be all categorize into one category which is a Trojan.

API Mapping

API mapping is the process of mapping known malicious or suspicious API to the API from the dataset. This step will map the dataset APIs to the most commonly used API calls which is gathered on Section III. The API mapping is done using Python codes. This step is done to dismiss any API that is not considered as malicious or suspicious.

The result of this step is that we get known malicious or suspicious APIs from the dataset which is 134 known malicious or suspicious APIs. Moreover, we will also map the APIs gathered from these steps to their functionalities on MSDN (2018). This will show us what the function of that APIs. After that, we will categorize the APIs based on their categories as shown in Table 10.

Table 8: Malware sample site

Malware sample	Description
Virus share	VirusShare provides free malware samples, however user must register as a member first before they can began downloading the sample. The registration is free of charge. The sample can be downloaded one file at a time (Direct Download) or bulk (Torrent)
The zoo	Provides live malware that for public. It is hosted on GitHub and doesn't require any registration to download the sample
Mal share	Provides free malware sample but need to register first. The registration is free
Virus sign	Provides malware sample but need to register to have access to the samples. They have free membership as well as premium one. Free membership has some limitation

Table 9: Malware categories and their number of samples

Category	Number of sample
Backdoor	14
Virus	19
Worm	8
Trojan	105
UDS: Dangerous Object	24
Others	12
Total	182

Table 10: APIs and their categories

Category	APIs
Hooking	SetWindowsHookEx, SetWindowsHookEx, UnhookWindowsHookEx
Services	ControlService, CreateService
Socket	ioctlsocket, socket, WSAAsyncSelect, WSACleanup, WSAGetLastError, WSAIoctl, WSASocket, WSAStartup, recvfrom, gethostbyname, gethostname, accept, bind, connect, free
Synchronization	CreateMutex, OpenMutex
Keyboard	GetKeyboardState, GetKeyState, SetKeyboardState, GetKeyboardState, GetAsyncKeyState, keybd_event, MapVirtualKey

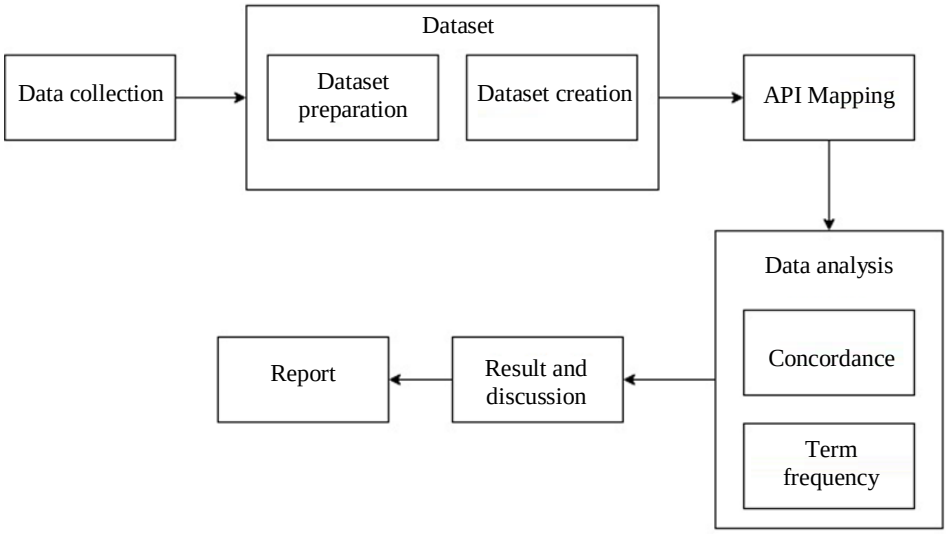

Fig. 1: Methodology of the research

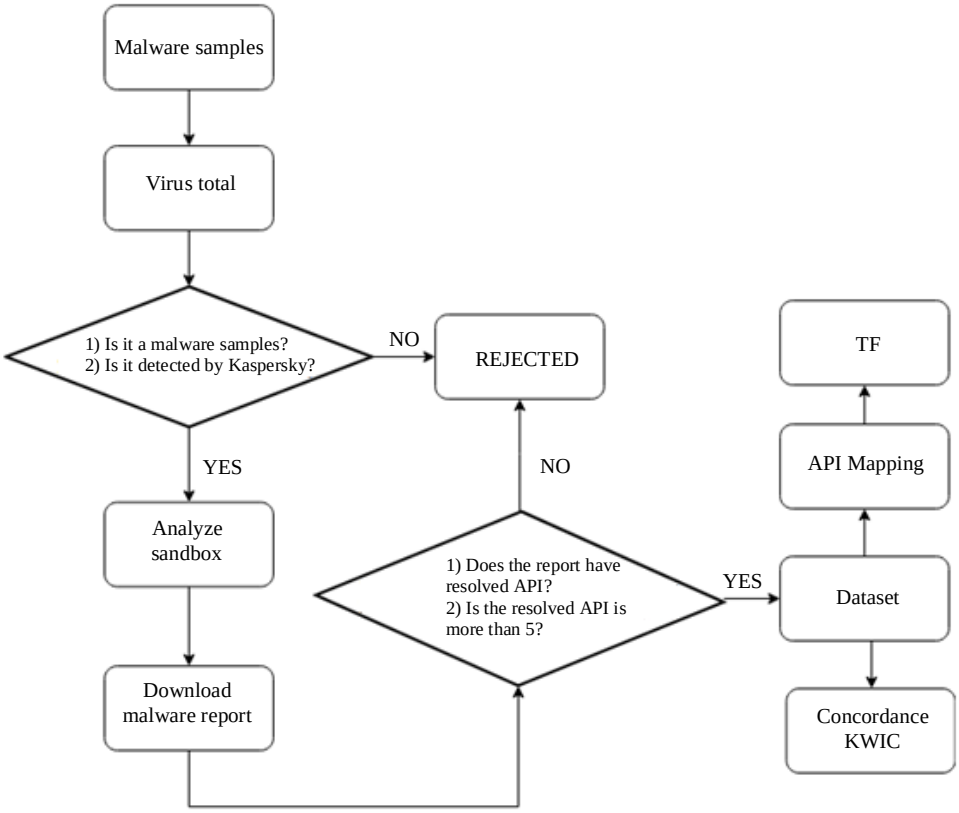

Fig. 2: The framework of implementation for the experiment

Data Analysis

Concordance

Anthony and Ant Conc (2018) is a concordance tool that is used in this experiment. The steps on how to use AntConc are as follows:

a. Compile a list of APIs from the dataset into a plain text file (.txt). This is because, concordance tool's corpus must be in plain text format
b. Then, compile a list of API call sequence gathered on section III into another text file
c. After that, key-in the API from previous step into the keyword box in the AntConc and click start

DOI: 10.3844/jcssp.2019.1307.1319

d. AntConc will display the results and if we need to further filter the results, just add other keywords to the previous one. Make sure to use spaces instead of commas when adding new keyword
e. Take note of the frequencies of the results as this will be used on section V

What makes this concordance easier than n-gram is that, this method only display output based on the keywords, hence making reading and filtering the results easier. As mentioned before, concordance uses KWIC method which will only display the output based on the keyed keywords and their close contexts.

Term Frequency

As mentioned before, TF is used to calculate the most commonly used APIs in the dataset. The API calls used in this step are chosen randomly and not based on the categories of the APIs. This is because the TF is used to show which of the malicious or suspicious APIs is favorable by the malware in the dataset. There are a few steps involved when calculating term frequency of the APIs. The steps are as follows:

a. The first thing to do is to create a list of APIs based on the malware categories in the dataset
b. Then, map the API to the result of the API mapping
c. After that, use freqKey.py to calculate the frequencies of the API
d. During the API Mapping phase, we got 134 known malicious or suspicious API in the dataset, which we will use as our N
e. Meanwhile, the C(t) is the result we get from freqKey.py. Then, use Eq. 1 to calculate the TF
f. After that, we will calculate IDF. Since there are 6 malware categories in the dataset, this is what we will use as our ND
g. To calculate ND_t we will calculate how many documents (categories) does the term appear. Then, use Eq. 2 to calculate IDF
h. And finally, to calculate TF-IDF, we will use Equation. 3
i. Take note of the results as this will be used on section V

Results and Discussion

Concordance

Based on the experiment done in section IV, the results for concordance can be seen in Fig. 3. The figure shows the lists of malware's malicious behaviors and their frequencies found in the dataset. The behaviors that we manage to identify using this method are; Antidebugging, Process Hollowing, Terminate Process, Privilege Escalation, Install itself at startup and Enumerate all process. On the other hand, the API call sequence of these behaviors can be seen on Table 11.

The highest behavior frequency is to install itself at startup. And the API call sequence for this behavior is DelNodeRunDLL32. Upon further reading at Process Library (2018) about the API shows that the API are from advpack.dll which is not a malicious dll. It is a normal dll that helps with hardware and software installation. Moreover, further reading on DelNodeRunDLL32 shows that it is usually installed on the computer when user downloaded a free software (Abalmasov, 2010). When downloading free software, user will sometimes be asked to install another component, this technique is called bundled installation and when the user failed to reject the offer, DelNodeRunDLL32 will be installed on the computer.

On the other hand, the lowest behavior frequency is to enumerate all processes. This may be due to the only certain malware need to see all processes in the system to find their target.

Term Frequency

As mentioned before, TF is use in this experiment to show the most commonly used API calls in the dataset. Based on the experiment done in section IV, the result of TF can be seen on Fig. 4. The result shows that Reg Close Key (TF: 1.388) is the most commonly used APIs in the dataset. Meanwhile, the second most used API in the dataset is DelNodeRunDLL32 (TF: 0.806).

On the other hand, the TF-IDF (Fig. 5) result shows that Get Thread Context (TF-IDF: 0.097) has the highest TF-IDF score while the GetTickCount64 (TF-IDF: 0.074) shows that some of the malware is on 64-bit architecture since that API is only used on 64-bit Windows.

Table 11: Malicious behaviors and their API call sequence on the dataset

Behavior	API call sequence
Enumerate all process	CreateToolhelp32Snapshot, Process32First, Process32Next
Install itself at startup	DelNodeRunDLL32
Privilege Escalation	OpenProcessToken, LookupPrivilegeValueA, AdjustTokenPrivileges
Terminate Process	TerminateProcess OR NtTerminateProcess
Process Hollowing	CreateProcess, NtUnmapViewOfSection, VirtualAllocEx, WriteProcessMemory
Antidebugging	IsDebuggerPresent OR OutputDebugStringA OR OutputDebugStringW

DOI: 10.3844/jcssp.2019.1307.1319

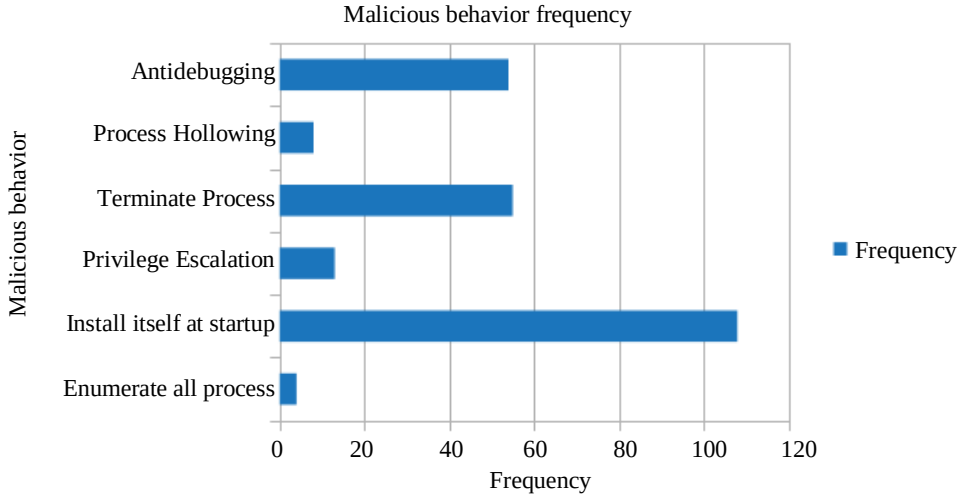

Fig. 3: Concordance Results

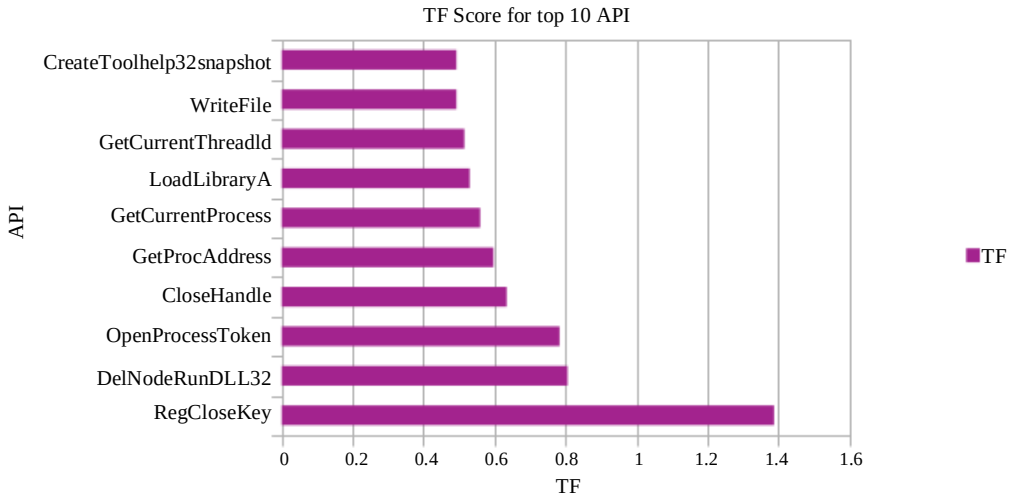

Fig. 4: The TF score

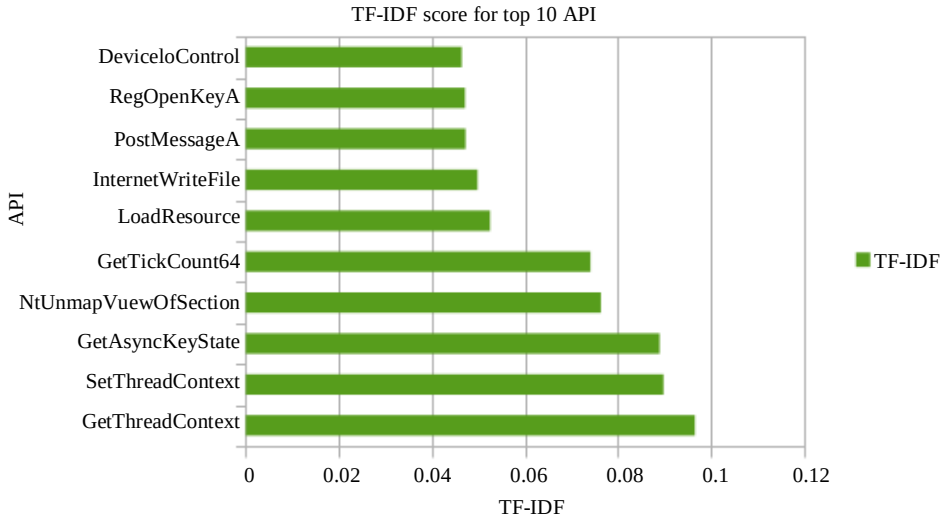

Fig. 5: The TF-IDF Score

DOI: 10.3844/jcssp.2019.1307.1319

Discussion

As mentioned before, concordance is used to identify the API call sequence from the dataset. The KWIC concordance method is easier to use than n-gram because n-gram listed all possible outcomes based on n value, meaning there will be a lot of output being displayed as compared to this method who will only display results based on the queried keywords. For example, we want to search whether the malware has Terminate Process behavior, we just must query the TerminateProcess as the keyword and it will display the context of the TerminateProcess in the dataset.

Moreover, we can also query multiple keywords separated by a space, in the search box. However, by using this method means that we must know the API call sequence for the malicious behavior beforehand as if we do not know the API call sequence for the malicious behavior, then we have nothing for the keyword. This is one of the disadvantages of this method, though we should never let it cover our judgment regarding this method.

On the other hand, based on the results from the experiment, it does provide evidence that this method can be used to search for API call sequence patterns in a dataset. The reason for this is because the researcher manages to identify six malicious behaviors from the dataset using this method. Table 12 shows the comparison results for malicious behaviors detected among different methods.

Based on the Table 12 above, we can infer that Concordance KWIC and N-gram both have great capabilities in detecting malware behaviors as compared to other methods. However, as mentioned before, concordance will display less results as compared to the n-gram output, making reading the results easier.

The TF score from the experimental result shows that the most used APIs is the RegCloseKey (1.388). While the TF-IDF score shows that the GetThreadContext (0.097) has the highest TF-IDF score. Both of these APIs is not dangerous in nature, which give revelation that not all APIs are malicious in nature, it will only be dangerous if it is programmed to do malicious tasks like what malware authors do. They use normal APIs and programmed it to do malicious task. Take examples DeleteFile API which function is to delete existing files. The normal user may use the API to delete unwanted file, however, malware author may use it to compromise the system by deleting important files.

To say that this research, manages to cover all malicious APIs is an overstatement of the year since there are so many APIs out there and there is no way this research will cover it all. That being said, at least we can state that this research, manages to cover a small portion of APIs that is considered malicious and should we encounter those APIs, we have an inkling of an idea that the program that use those APIs may be a malware program instead of benign.

Table 12: Comparison results between malware behavior detection methods

Researcher	Method	Results (malicious behavior detected)
Alazab *et al*. (2010)	N-gram	6
Ki *et al*. (2015)	Sequence alignment algorithm	4
This study	Concordance KWIC	6

This further solidified the view that by viewing the APIs of a malware, we can identify whether that program is a malware or benign. This is because some APIs are dangerous in nature, such as TerminateProcess. If TerminateProcess are used recklessly such as terminating the critical process of a program or system, it may cause blue screen of death, which consequently would cause loss of data or worse, system failure. This shows how dangerous some APIs are as compared to the others.

Conclusion

In this experiment, we use concordance tool to identify malicious API call sequence from the dataset. The results are that we manage to identify six behaviors using this method which is antidebugging, install itself during startup, privilege escalation, terminate processes, process hollowing and enumerate all process. This shows that we can use a concordance to identify API call sequence of a malware.

Moreover, we also use TF to statistically identify which of the malicious or suspicious APIs is favorable by the malware. This is because, the results of the experiment show that RegCloseKey (TF: 1.388) as the most commonly used APIs in the dataset. However, its functionality is not dangerous in nature. Hence, we can infer that some API are not dangerous and will only be dangerous if programmed to do so.

All in all, this study has proven that we can use a concordance to identify malware behavior. And based on the results of the experiment, we manage to achieve the purpose of this research which is to use a concordance to detect malware behavior based on their API call and to find out the API that is frequently used by the malware in the dataset.

Hence, the next logical step is to develop a concordance tool that can automatically identify malware behaviors from the database. This is what we recommend for future research. Furthermore, since this research only uses a small sample dataset, the future research may use a bigger dataset to further test the concordance KWIC ability in identifying the API call sequence.

Acknowledgment

This study was supported by the Universiti Kebangsaan Malaysia grant: DCP-2017-007/4

Author's Contributions

Nur Hilda Amira Abd Wahab: Researching and conducting the experiment as well as writing the manuscript.

Masnizah Mohd: Provide publication recommendations, reviewing the manuscript as well as supporting the publication of this manuscript.

Ravie Chandren Muniyandi: Reviewing the manuscript and provided guidance during project experimentation phase.

Balaji Rajendran, Gopinath Palaniappan: Contributing on reviewing the manuscript.

Ethics

This is an original manuscript and contains unpublished material. All authors have read, reviewed and approved the manuscript and there are no ethical issues involved.

References

Abalmasov, A., 2010. Delnoderundll32 Removal-Remove Delnoderundll32 Easily.

AbuHamad, M. and M. Mohd, 2019. Data categorization and model weighting approach for language model adaptation in statistical machine translation. Int. J. Advanced Comput. Sci. Appli., 10: 135-141.

Alazab, M., S. Venkataraman and P. Watters, 2010. Towards understanding malware behaviour by the extraction of API calls. Proceedings of the 2nd Cybercrime Trustworthy Computing Workshop, Jul. 19-20, IEEE Xplore press, Australia.
DOI: 10.1109/CTC.2010.8

Altawaier, M.M. and S. Tiun, 2016. Comparison of machine learning approaches on Arabic Twitter sentiment analysis. Int. J. Advanced Sci. Eng. Inform. Technol., 6: 1067-1073.
DOI: 10.18517/ijaseit.6.6.1456

Anthony, L. and Ant Conc, 2018. ttp://www.laurenceanthony.net/software

Bai, J., Z. Zhao and J. Wang, 2014. Malware detection method based on the control-flow construct feature of software. IET Inform. Secur. 8: 18.24.

Belaoued, M. and S. Mazouzi, 2015. Towards an automatic method for API Association extraction for PE-Malware categorization. Proceedings of the International Conference Intelligent Information Processing, Security Advanced Communication, Nov. 23-25, ACM New York, USA.
DOI: 10.1145/2816839.2816921

Bowker, L., 2018. Corpus linguistics is not just for linguists: Considering the potential of computer-based corpus methods for library and information science research. Library Hi Tech.

Brown, M.H., 2017. Using the sentence corpus of remedial English to introduce Data-Driven Learning tasks. Kanda Acad. Rev., 1: 1-14.

Business Advantage, 2017. The State of Industrial Cybersecurity.

Cisco, 2018. Annual Cybersecurity Report.

Fujino, A., J. Murakami and T. Mori, 2015. Discovering similar malware samples using API call topics. Proceedings of the 12th Annual IEEE Consumer Communications Networking Conference, Jan. 9-12, IEEE Xplore perss, USA,
DOI: 10.1109/CCNC.2015.7157960

Ghazvini, A. and Z. Shukur, 2017. Review of information security guidelines for awareness training program in healthcare industry. Proceedings of the 6th International Conference Electrical Engineering and Informatics, Nov. 25-27, IEEE Xplore perss, Malaysia,
DOI: 10.1109/ICEEI.2017.8312399

Jiang, Y. and H. Liu, 2017. Research on the Construction of Parallel Corpus for the Specific Field of Machine Translation. 55: 77-82.

Ki, Y., E. Kim and H.K. Kim, 2015. A novel approach to detect malware based on API call sequence analysis. Int. J. Distrib Sens Networks.

Lim, H., 2016. Detecting malicious behaviors of software through analysis of API sequence k-grams. Comput. Sci. Inf. Technol., 4: 85-91.

Manap, N.A., A.A. Rahim and H. Taji, 2015. Cyberspace Identity Theft: The Conceptual Framework Nazura Abdul Manap Anita Abdul Rahim. M.J.S.S., 6: 595-605.

Merriam-Webster, 2018. Application Programming Interface.

Mohaddes, H., R.C. Muniyandi, I.T. Ardekani and A. Sarrafzadeh, 2016. Taxonomy of malware detection techniques: A Systematic Literature Review. IEEE.

Pektas, A. and T. Acarman, 2017. Malware classification based on API calls and behaviour analysis. IET. Inform. Secur.

Process Library. 2018. Advpack.dll.

Rajaraman, A. and J.D. Ullman, 2011. Mining of Massive Datasets: Data Mining (Ch01). Min Massive Datasets. 18: 114-42.

Rockwell, G., 2018. Too Much Information and the KWIC.

Salehi, Z., A. Sami and M. Ghiasi, 2014. Using feature generation from API calls for malware detection. Comput. Fraud Secur., 2014: 9-18.
DOI: 10.1016/S1361-3723(14)70531-7

Sikorski, M. and A. Honig, 2012. Practical malware analysis the hands-on guide to dissecting malicious software. No Starch Press.

Sundarkumar, G.G., V. Ravi, I. Nwogu and V. Govindaraju, 2015. Malware Detection via API calls, Topic Models and Machine. Proceedings of the International Conference Automation Science Engineering, Aug. 24-28, IEEE Xplore press, Sweden, pp: 1212-7.
DOI: 10.1109/CoASE.2015.7294263

Wynne, M., 2008. Searching and Concordancing. Pre-publication Draft a chapter which Appear Handb Corpus Linguist Ed by Merja Kytö Anke Lüdeling, Mout Gruyter.

Yılmaz, E. and A. Soruç, 2015. The use of concordance for teaching vocabulary: A data-driven Learning Approach. Proc. Soc. Behav. Sci., 191: 2626-2630. DOI: 10.1016/j.sbspro.2015.04.400 Get