International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-6, April 2020

Proposed Model for Detecting Malware on Workstations

Cho Do Xuan, Tisenko Victor Nikolaevich, Do Hoang Long, Nguyen Vuong Tuan Hiep, Le Quang Sang

Abstract- The trend of network attacks through end-users are widely used by attackers today. One of them is the attack by distributing malware into users' computers to steal data or escalate to higher privileges. The technique of attack by distributing malware is a dangerous attack method that is difficult to detect and prevent. Therefore, the task of detecting the sign of malware and alerting it for the user or the system is very necessary today. Current studies and recommendations for detecting malware are often based on two main methods that are using a set of signs and analyzing abnormal behavior based on machine learning or deep learning. In this paper, we will propose a method to detect malware on users' computers using an Event ID profile analysis technique. Event IDs are signs and behaviors of malware that are tracked and collected on the operating system kernel of the workstation. The difference between our research and other published methods is the way to collect behaviors of the malware. We don't collect them through virtualization systems, but through direct processes in the operating system kernel. Therefore, even though malware uses hidden techniques, its actions are recorded by the operating system kernel and based on those processes, we use the Event ID analysis technique to conclude about the existence of malware in the system.

Keywords: MALWARE detection, WORKSTATIONS, Event ID

I. INTRODUCTION

The document [1] presented the concepts, characteristics, and classification of malware. In the study [2], the authors have listed and explained why the threat of malware distribution is becoming a trend of the current types of network attacks. Therefore, the issue of malware detection in the system has become a problem that needs attention today. In the study [1], there are two main methods used to detect current malware: signal-based detection technique and machine-based detection technique. However, one common feature of these methods is the use of methods of extracting signals and behavior of malware based on sample data sets.

These data sets are based on virtualization tools or network monitoring and static analysis tools. For virtualization tools, studies often use the Sandbox tool [3] to execute and extract malware. Sandbox tools have the disadvantage of only recognizing behaviors for a certain period of time, so they will not be able to fully report their behavior.

Revised Manuscript Received on April 30, 2020.
***** Correspondence Author
Cho Do Xuan, FPT University Hanoi, Vietnam chodx@fe.edu.vn
Tisenko Victor Nikolaevich , Peter the Great St. Petersburg Polytechnic University Russia, St.Petersburg, Polytechnicheskaya, 29v_tisenko@mail.ru
Do Hoang Long, FPT University Hanoi, Vietnamchodx@fe.edu.vn
Nguyen Vuong Tuan Hiep, FPT University Hanoi, Vietnam hiepnvtse05065@fpt.edu.vn
Le Quang Sang, FPT University Hanoi, Vietnam sanglqse04676@fpt.edu.vn

For data sets collected during static analysis, it is characterized by abnormalities only detected when malware has been spread and connected to steal data. Therefore, these traditional approaches will always be overcome by malware. In addition, the tendency of attackers to attack users to escalate privileges in the system. From the above reasons, in this paper, we propose a method to detect malware on Workstation. Our proposed method will directly monitor and detect malware on Workstation based on events collected on the operating system kernel using marker set and behavior analysis.

II. RELATED WORKS

Currently, the theoretical research related to detecting malware on Workstation machines is very limited. However, in regard to the issue of malware detection on Workstations, in addition to Anti Virus software products, there are some products that support Endpoint detection & Response (EDR). The EDR product has the function of detecting and tracking abnormal incidents on the end-user so that it can provide incident response scenarios. Accordingly, according to the announcement at [4], there are a number of EDR's solutions and products such as Apex One, which is Trend Micro's EDR, [5] has the ability to automatically detect and prevent as many endpoint threats as possible, without any manual intervention from the user. Apex One will also detect and prevent exploitation of vulnerabilities in the operating system before threats penetrate the endpoint with virtual patches constantly updated with artificial intelligence from Trend Micro's Zero Day Initiative. Similar to Trend Micro's Apex One, Palo Alto Networks Traps products from Palo Alto [6] prevents threats in endpoint, and combines with cloud and network security to prevent network attacks. Traps prevent the launch of malware, DLLs files, and Office files with various methods of prevention, reducing the attack surface and increasing the accuracy of malware prevention. This approach prevents both known and unknown malware from endpoint infection by combining components such as WildFire threat intelligence, Local analysis via machine learning, WildFire inspection, and analysis, Granular child process protection, Periodic scanning for dormant malware. Kaspersky EDR [7] can continuously monitor and analyze abnormal phenomena, suspicious processes on employee's workstations. It also responds to threats in manual mode, as well as in passive mode. In addition, Kaspersky EDR allows control of incidents at network endpoints, detects malware and unauthorized acts that aren't identifiable at the network protection level, and responds quickly to them. In addition, there are some other solutions such as VMware Carbon Black EDR [8], Falcon [9], Malwarebytes Endpoint Detection, and Response [10].

III. PROPOSING A DETECTION MODEL

3.1. Proposed model

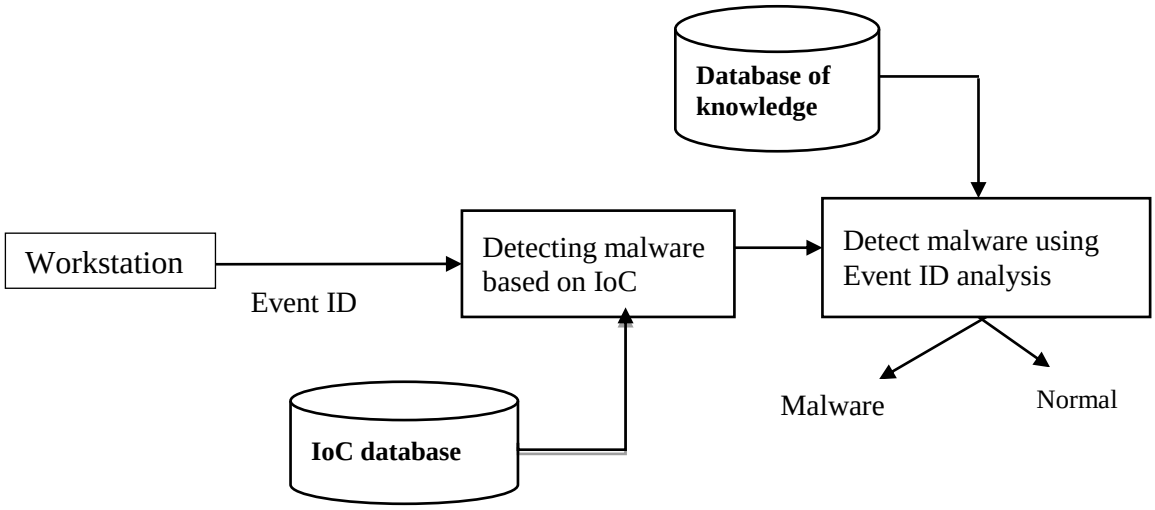

Figure 1. Proposing a malware detection model on user machines

Figure 1 shows the malware detection model on the user's computer which proposed by us. The main components in our proposed model include:

- **Workstation** is a users' machine that needs monitoring. Accordingly, in this paper, to collect the processes on the operating system kernel of a user's computer, it is necessary to have tools to collect, process, and transfer the process to the analysis center. Wherefore, each user using different operating systems will have different collection and processing tools. We will collect the process in the operating system kernel in the user's computer using Window and Ubuntu operating system by the Sysmon tool [11].
- **Event ID**: are processes collected by the Sysmon tool on the operating system kernel. According to [11], the Sysmon tool will collect a total of 21 processes from the operating system kernel. Table 1 below describes the 21 processes collected.

Table 1. List of processes collected on the operating system kernel with the Sysmon tool

No	Name and symbol of Event	Describe
1	Event ID 1: Process creation	The process creation event provides extended information about a newly created process. The full command line provides context on the process execution. The ProcessGUID field is a unique value for this process across a domain to make event correlation easier. The hash is a full hash of the file with the algorithms in the HashType field.
2	Event ID 2: A process changed a file creation time	The change file creation time event is registered when a file creation time is explicitly modified by a process. This event helps tracking the real creation time of a file. Attackers may change the file creation time of a backdoor to make it look like it was installed with the operating system. Note that many processes legitimately change the creation time of a file; it does not necessarily indicate malicious activity.
3	Event ID 3: Network connection	The network connection event logs TCP/UDP connections on the machine. It is disabled by default. Each connection is linked to a process through the ProcessId and ProcessGUID fields. The event also contains the source and destination host names IP addresses, port numbers and IPv6 status.
4	Event ID 4: Sysmon service state changed	The service state change event reports the state of the Sysmon service (started or stopped).
5	Event ID 5: Process terminated	The process terminate event reports when a process terminates. It provides the UtcTime, ProcessGuid and ProcessId of the process.
6	Event ID 6: Driver loaded	The driver loaded events provides information about a driver being loaded on the system. The configured hashes are provided as well as signature information. The signature is created asynchronously for performance reasons and indicates if the file was removed after loading.
7	Event ID 7: Image loaded	The image loaded event logs when a module is loaded in a specific process. This event is disabled by default and needs to be configured with the –l option. It indicates the process in which the module is loaded, hashes and signature information. The signature is created asynchronously for performance reasons and indicates if the file was removed after loading. This event should be configured carefully, as monitoring all image load events will generate a large number of events.

8	Event ID 8: CreateRemoteThread	The CreateRemoteThread event detects when a process creates a thread in another process. This technique is used by malware to inject code and hide in other processes. The event indicates the source and target process. It gives information on the code that will be run in the new thread: StartAddress, StartModule and StartFunction. Note that StartModule and StartFunction fields are inferred, they might be empty if the starting address is outside loaded modules or known exported functions.
9	Event ID 9: RawAccessRead	The RawAccessRead event detects when a process conducts reading operations from the drive using the \\.\ denotation. This technique is often used by malware for data exfiltration of files that are locked for reading, as well as to avoid file access auditing tools. The event indicates the source process and target device.
10	Event ID 10: ProcessAccess	The process accessed event reports when a process opens another process, an operation that's often followed by information queries or reading and writing the address space of the target process. This enables detection of hacking tools that read the memory contents of processes like Local Security Authority (Lsass.exe) in order to steal credentials for use in Pass-the-Hash attacks. Enabling it can generate significant amounts of logging if there are diagnostic utilities active that repeatedly open processes to query their state, so it generally should only be done so with filters that remove expected accesses.
11	Event ID 11: FileCreate	File create operations are logged when a file is created or overwritten. This event is useful for monitoring autostart locations, like the Startup folder, as well as temporary and download directories, which are common places malware drops during initial infection.
12	Event ID 12: RegistryEvent (Object create and delete)	Registry key and value create and delete operations map to this event type, which can be useful for monitoring for changes to Registry autostart locations, or specific malware registry modifications.
13	Event ID 13: RegistryEvent (Value Set)	This Registry event type identifies Registry value modifications. The event records the value written for Registry values of type DWORD and QWORD.
14	Event ID 14: RegistryEvent (Key and Value Rename)	Registry key and value rename operations map to this event type, recording the new name of the key or value that was renamed.
15	Event ID 15: FileCreateStreamHash	This event logs when a named file stream is created, and it generates events that log the hash of the contents of the file to which the stream is assigned (the unnamed stream), as well as the contents of the named stream. There are malware variants that drop their executables or configuration settings via browser downloads, and this event is aimed at capturing that based on the browser attaching a Zone.Identifier "mark of the web" stream.
16	Event ID 17: PipeEvent (Pipe Created)	This event generates when a named pipe is created. Malware often uses named pipes for interprocess communication.
17	Event ID 18: PipeEvent (Pipe Connected)	This event logs when a named pipe connection is made between a client and a server.
18	Event ID 19: WmiEvent (WmiEventFilter activity detected)	When a WMI event filter is registered, which is a method used by malware to execute, this event logs the WMI namespace, filter name and filter expression.
19	Event ID 20: WmiEvent (WmiEventConsumer activity detected)	This event logs the registration of WMI consumers, recording the consumer name, log, and destination.
20	Event ID 21: WmiEvent(WmiEventConsumerToFilter activity detected)	When a consumer binds to a filter, this event logs the consumer name and filter path.
21	Event ID 22: DNSEvent (DNS query)	This event generates when a process executes a DNS query, whether the result is successful or fails, cached or not. The telemetry for this event was added for Windows 8.1 so it is not available on Windows 7 and earlier.
22	Event ID 255: Error	This event is generated when an error occurred within Sysmon. They can happen if the system is under heavy load and certain tasked could not be performed or a bug exists in the Sysmon service.

- **Database of IoC:** is a database of Indicators of compromise (IoC) of known malware, including IP blacklist, malicious URL, C&C server, Virus signatures, MD5 hashes, botnet command, etc. Figure 2 shows the components of IoC in malware:

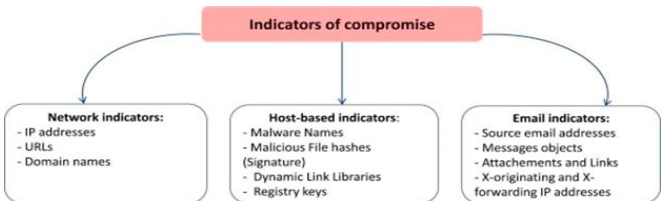

Figure 2. Indicators of compromise

In this paper, we build the IoC database of malware, including the fields as shown in Table 2.

Table 2. Malware fields described in IoC Database

No	Attribute	Describe
1	Connection	The addresses that malware connects to.
2	Malware_Behavior	Type of malware behavior
3	Malware_Type	Malware Type
4	Malware_name	Malware name
5	Malware_path	Path of malware
6	Other_path	Other related paths of the malware
7	Patch	Version
8	Process	Information of Process that malware uses and creates
9	Service	Information of Server that malware creates
10	VulnerabilityInfo	Information of Vulnerability

- **Detects malware based on IoC**: This is the block whose function is to compare each Event ID collected with the IoC database of malware. The result of the Event ID comparison will know which Event is malicious.
- **Knowledge database:** this is the database that we have built based on collected malware samples. Accordingly, we will give the experimental malware samples in a virtualized environment to obtain Event ID groups as well as the behaviors of each Event ID group. We then conducted a weight rating for each Event ID as well as other malware behavior. The evaluation of this weight is crucial to assessing process records to conclude about malware signs.
- **Detecting malware using Event ID analysis:** based on the built Event profiles as well as the weight of each Event in the Event profile, we will evaluate these Event profiles by Event ID analysis technique. Accordingly, for every new Event added in the Event profile, the system will immediately evaluate the weight of the Event ID profile based on the weight of the Event in the profile.

3.2. Workflow of approach

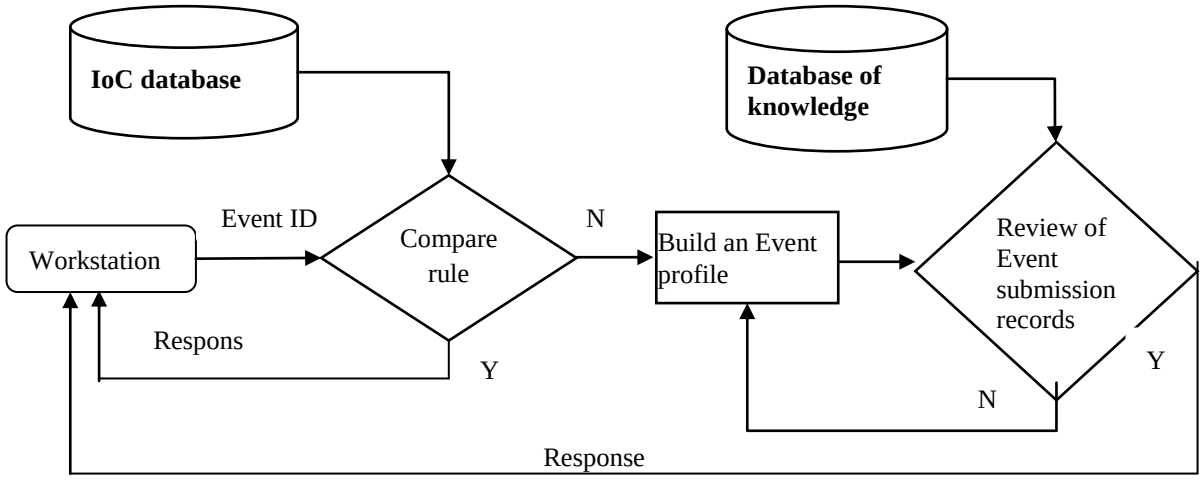

Figure 3. Activity flow of multilayer analysis model for APT attack detection based on Network traffic

Figure 3 shows the flow of the malware detection model on a Workstation. Accordingly, the process of detecting malware of the system is as follows:

- **Step 1:** Collect and process Event ID on Workstation: As mentioned in section 3.1, the malware detection system has the function of detecting and monitoring the signs and behavior of malware based on the processes which are recorded on the operating system. To accomplish the task of collecting and extracting these processes, we will install and configure two main tools that are Sysmon and Linux Auditing System [12] These tools correspond to 2 current popular operating systems that are Windows and Linux. These tools collect the processes recorded by the operating system and transfer them to the processing and monitoring center.
- **Step 2:** Detect malware based on Event ID using the database about IoC. After the processes are collected and transported to the processing and monitoring center,

the system will check the Event ID based on the IoC database of malware collected earlier. If the system detects that the Event ID is in the IoC database, it immediately alerts the user about the existing malware. If there is no abnormality in the Event IDs, the system will move the monitoring of this Event ID to the next stage.

- **Step 3:** Detect malware based on analyzing Event profile. At step 3, to check whether the newly created Event ID is an abnormal Event, we perform two main phases:

  - **Phase 1:** Develop an Event behavior profile. Accordingly, events born from the operating system will be checked and evaluated to know whether this event is related to any previously collected processes. If true, the event will continue to be attached to the behavior profile of previous Events. If false, the event will be built into a behavior profile of the new event.

  - **Phase 2:** Detecting malware based on Fuzzy logic: Accordingly, based on the Event profiles collected in phase 1, we will conduct a review of each event profile to conclude about the signs of the malware in the system. To accomplish this task, in phase 2 we propose two tasks as follows:

    - **Task 1**: Assess the Event weights in the Event profile. Based on the role and importance of each Event, we will assign weights to the process. The weight assignment for each Event will be done automatically based on the weight that was built for each Event included in the Knowledge Database. In this paper, to evaluate the weight of each Event ID we use the matrix MITRE's ATT & CK [13]. MITRE's ATT & CK is an adversary's technical, tactical, and strategical framework based on real-world situations. The MITRE matrix is a globally accessible knowledge base of opposite tactics and techniques based on real-world observations. The ATT & CK Knowledge Base is used as a platform to develop specific models and threat methods in cybersecurity products. In this system, we divided into 5 levels of attack techniques that show the danger level increases gradually according to color

      + Level 1 (blue):
        - Not exploitable.
        - Additional techniques are required to be feasible.
        - No effect.
      + Level 2 (green):
        - Can exploit moderately.
        - No need for POC malware, scripts or tools.
        - Do not affect or little effect.
      + Level 3 (yellow):
        - Need some tools like Metasploit or POC scripts.
        - Stronger than level 2.
        - Can affect many.
      + Level 4 (orange):
        - Need more conditions to be able to exploit.
        - Can be exploited, stolen information, and remotely controlled.
      + Level 5 (red):
        - Need to execute DLL / EXE.
        - Need deep understanding of the operating system.
        - Can hijack.
        - Serious impact.

Task 2: Evaluation of Event Profiles: Based on the Event profiles and the weights of each Event, we will use a technique to accrue the factors of the Event ID in the Event ID profile to conclude about the signs of malware in the system.

IV. SOME EMPIRICAL RESULTS DETECT MALWARE ON WORKSTATIONS

4.1. Detecting malware based on IoC

Experiment with a sample of malware already stored in the IOC, the sample Boy.exe has a hash of:
MD5: 79292189FB211042653601CC8E32AEA2
SHA-1:
71A9B6E8E136F2BCDA3B1E11C27FC0CFB48F8492


```
"Connection": [
        "ms.fq520000.com:443",
        "nicehash.hashnice.org:8080",
        "note.hashnice.org:443",
        "note1.hashnice.org:5555",
        "ad0.fq520000.com:8088",
        "ad1.fq520000.com:4433",
        "ad2.fq520000.com:5555"
    ],
    "Malware_Behavior": "Coin Miner",
    "Malware_Type": "Trojan",
    "Malware_name": "CPUInfo.exe",
    "Malware_path": [
       "%windir%\\IIS\\CPUInfo.exe"
    ],
    "Other_path": [
       "C:\\Windows\\IIS",
       "C:\\Windows\\boy.exe",
       "C:\\Windows\\System32\\boy.exe",
       "C:\\Windows\\System32\\las.exe",
       "C:\\Windows\\System32\\sys.exe"
    ],
    "Patch": [
       ""
    ],
    "Process": [
       {
          "Name": "srvany.exe",
          "Path": "C:\\Windows\\IIS\\srvany.exe",
          "Reg": "",
          "cmd": "C:\\Windows\\IIS\\srvany.exe"
```

```
        },
        {
          "Name": "CPUInfo.exe",
          "Path": "%windir%\\IIS\\CPUInfo.exe",
          "Reg": "",
          "cmd": "C:\\Windows\\IIS\\CPUInfo.exe "
        },
        {
          "Name": "svsohst.exe",
          "Path": "C:\\Windows\\IIS\\svsohst.exe",
          "Reg": "",
          "cmd": "C:\\Windows\\IIS\\svsohst.exe"
        },
        {
          "Name": "cmd.exe",
          "Path": "%sys%\\cmd.exe",
          "Reg": "",
          "cmd": "cmd /c \"\"C:\\Windows\\IIS\\qdx.bat\" \""
        },
        {
          "Name": "cmd.exe",
          "Path": "%sys%\\cmd.exe",
          "Reg": "",
          "cmd": "cmd /c \"\"C:\\Windows\\IIS\\s.bat\" \""
        }
      ],
      "Service": [
        {
          "Name": "Application Layre Gateway Service",
          "Path": "C:\\Windows\\boy.exe",
          "Reg":
"HKEY_LOCAL_MACHINE\\SYSTE
M\\CurrentControlSet\\services\\Appli
cation Layre Gateway Service"
        },
        {
          "Name": "iis",
          "Path": "C:\\Windows\\IIS\\srvany.exe",
          "Reg":
"HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\services\\ii
s"
        }
      ],
      "VulnerabilityInfo": [
        "MS 17-010"]
```


Figure 4. List of signs of Boy.exe

The process of analyzing and detecting the code has been recorded by the system:


```
{
  "DETECTED": [
    {
      "Malware_Behavior": "Coin Miner",
      "Malware_Type": "Trojan",
      "Malware_name": "CPUInfo.exe",
      "Malware_path": [
        "%windir%\\IIS\\CPUInfo.exe"
      ]
    }
  ],
  "KILL PROCESS":  [
    {
      "Name": "srvany.exe",
      "Path": "C:\\Windows\\IIS\\srvany.exe",
      "Reg": "",
      "cmd": "C:\\Windows\\IIS\\srvany.exe"
    },
    {
      "Name": "CPUInfo.exe",
      "Path": "%windir%\\IIS\\CPUInfo.exe",
      "Reg": "",
      "cmd": "C:\\Windows\\IIS\\CPUInfo.exe "
    },
    {
      "Name": "svsohst.exe",
      "Path": "C:\\Windows\\IIS\\svsohst.exe",
      "Reg": "",
      "cmd": "C:\\Windows\\IIS\\svsohst.exe"
    },
```

```
        {
          "Name": "cmd.exe",
          "Path": "%sys%\\cmd.exe",
          "Reg": "",
          "cmd": "cmd /c \"\"C:\\Windows\\IIS\\qdx.bat\" \""
        },
        {
          "Name": "cmd.exe",
          "Path": "%sys%\\cmd.exe",
          "Reg": "",
          "cmd": "cmd /c \"\"C:\\Windows\\IIS\\s.bat\" \""
        }
    ],
  "DELETE SERVICE": [
        {
          "Name": "Application Layre Gateway Service",
          "Path": "C:\\Windows\\boy.exe",
          "Reg":
"HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\services\\
Application Layre Gateway Service"
        },
        {
          "Name": "iis",
          "Path": "C:\\Windows\\IIS\\srvany.exe",
          "Reg":
"HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\services\\
iis"
        }
    ],
  "DELETED": [
        "C:\\Windows\\IIS",
        "C:\\Windows\\boy.exe",
        "C:\\Windows\\System32\\boy.exe",
        "C:\\Windows\\System32\\las.exe",
        "C:\\Windows\\System32\\sys.exe"
    ]
}
```


Figure 5. Malware detection results using IoC database

4.2. Detect malware using Event ID profile analysis

Experiment with a sample of malware called test 1.EXE. This malware is not yet in IoC. The malware information collected by the system is as follows:
MD5: 20CCBEE3F7DE86559FFB85FAA3C8DB8E
SHA-1: 7CFD78B7D0259C9C35123A050CCA072FB232773D
Figure 6 below shows the result of checking malware on VirusTotal.

Baidu	Undetected	BitDefender	Undetected
BitDefenderTheta	Undetected	Bkav	Undetected
CAT-QuickHeal	Undetected	ClamAV	Undetected
CMC	Undetected	Comodo	Undetected
CrowdStrike Falcon	Undetected	Cylance	Undetected
Cyren	Undetected	DrWeb	Undetected
Emsisoft	Undetected	Endgame	Undetected
eScan	Undetected	ESET-NOD32	Undetected
F-Prot	Undetected	F-Secure	Undetected
FireEye	Undetected	Fortinet	Undetected
GData	Undetected	Ikarus	Undetected
Jiangmin	Undetected	K7AntiVirus	Undetected
K7GW	Undetected	Kaspersky	Undetected

Figure 6. Malware test results on VirusTotal l

From figure 6 shows that reputable malware inspection software such as BitDefender, BKAV, Kaspersky cannot detect this is malware.

The process of building the Event ID profile when the malware is

executed is shown in Figures 7 and 8. Accordingly:

Event ID 1: We receive information about the malware's Hash values that are MD5 and SHA 256. These values coincide with the values obtained when we conduct manual analysis. At the same time, the system records information:

- ParentProcessID: 9156. This indicates that this is a new process that is not included in the Event ID record.
- Next noticed the malware: Load some DLLs, notably the Python-related DLLs such as python36.dll, pythoncom36.dll, etc. This information is sent by Event ID 11.

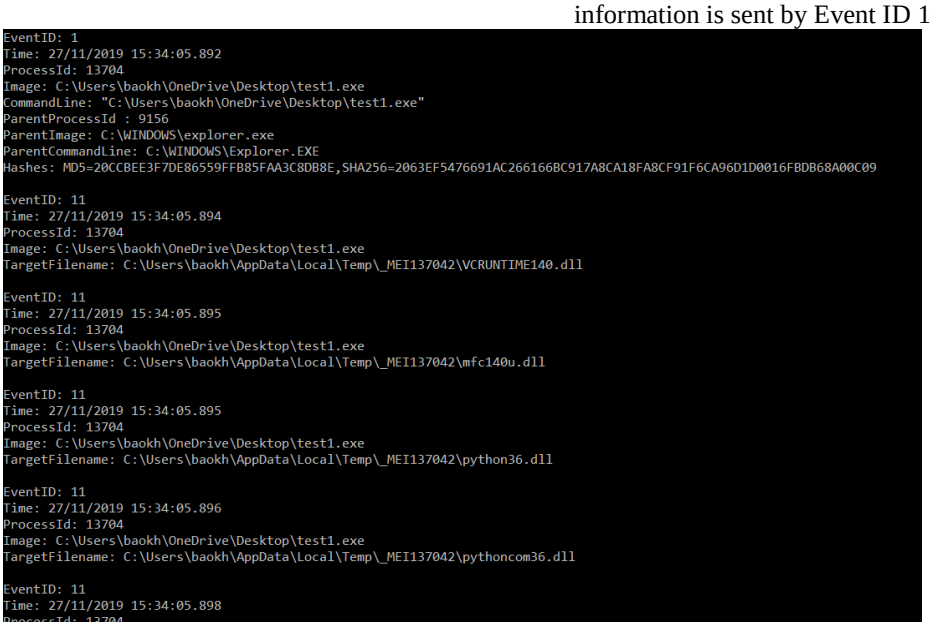

Figure 7. The process of building the Event ID profile from the data sent by Sysmon

- From Event ID 3, we see some actions to drop and execute _test1.exe file and also detect an external connection:

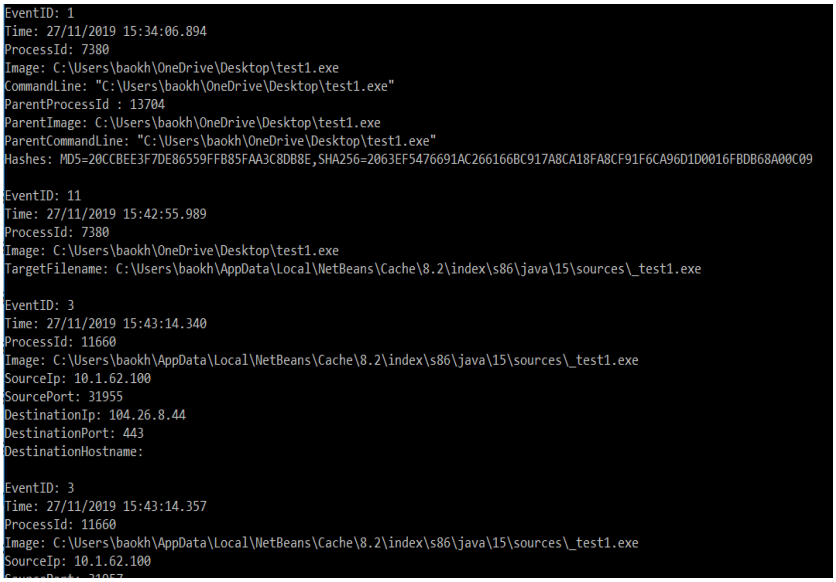

Figure 8. Event ID 3 sends back the connection of the executable file with the external server

Our system found that this malware sample dropped _test1.exe file and created a process to call a program written in Python to make a botnet connection. Based on the collected events as above, comparing with the rating scale of the Miter technique [13], we calculated a total score of 130. The system concludes that these are the malware's events. In conclusion, the test file 1.EXE is malware.

```
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe bot.contact_server 50
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe Pylock 40
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe CreateProcessW 40
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe Malicious score:  130 >>> Malware
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe Pylock 40
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe bot.contact_server 50
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe CreateProcessW 40
C:\Users\baokh\OneDrive\Desktop\test1\test1.exe Malicious score:  130 >>> Malware
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe Pylock 40
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe CreateProcessW 40
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe bot.contact_server 50
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe Malicious score:  130 >>> Malware
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe Pylock 40
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe bot.contact_server 50
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe CreateProcessW 40
C:\Users\baokh\AppData\Local\NetBeans\Cache\8.2\index\s123\java\15\refs\_test1.exe Malicious score:  130 >>> Malware
```

Figure 9. Malware detection results based on Event ID profile analysis

Testing systems with malware from the emerging Mustang Panda hacker group [14]. The purpose of the malware is to connect to the c & c server, download the malware to steal user information and provide remote control functionality.

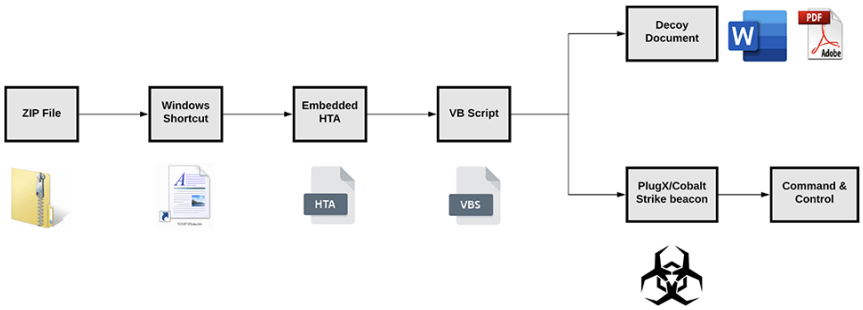

Figure 10. Operation process of Mustang Panda malware

Hash:
MD5: 26B444E01A080E9DBAFA9387929BF9DE
SHA-1:
B1650BEA5E02310AC5401A70D74D29E64A85B418

The analysis results are compared with the MITRE matrix as shown in Figure 11:

Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery
34 items	62 items	32 items	69 items	21 items	23 items
AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery
CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window Discovery
Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force	Browser Bookmark Discovery
Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery
Component Object Model and Distributed COM	AppInit DLLs	Application Shimming	Clear Command History	Credentials from Web Browsers	File and Directory Discovery
Control Panel Items	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Files	Network Service Scanning
Dynamic Data Exchange	Authentication Package	DLL Search Order Hijacking	Code Signing	Credentials in Registry	Network Share Discovery
Execution through API	BITS Jobs	Dylib Hijacking	Compile After Delivery	Exploitation for Credential Access	Network Sniffing
Execution through Module Load	Bootkit	Elevated Execution with Prompt	Compiled HTML File	Forced Authentication	Password Policy Discovery
Exploitation for Client Execution	Browser Extensions	Emond	Component Firmware	Hooking	Peripheral Device Discovery
Graphical User Interface	Change Default File Association	Exploitation for Privilege Escalation	Component Object Model Hijacking	Input Capture	Permission Groups Discovery
InstallUtil	Component Firmware	Extra Window Memory Injection	Connection Proxy	Input Prompt	Process Discovery
Launchctl	Component Object Model Hijacking	File System Permissions Weakness	Control Panel Items	Kerberoasting	Query Registry
Local Job Scheduling	Create Account	Hooking	DCShadow	Keychain	Remote System Discovery
LSASS Driver	DLL Search Order Hijacking	Image File Execution Options Injection	Deobfuscate/Decode Files or Information	LLMNR/NBT-NS Poisoning and Relay	Security Software Discovery
Mshta	Dylib Hijacking	Launch Daemon	Disabling Security Tools	Network Sniffing	Software Discovery
PowerShell	Emond	New Service	DLL Search Order Hijacking	Password Filter DLL	System Information Discovery
Regsvcs/Regasm	External Remote Services	Parent PID Spoofing	DLL Side-Loading	Private Keys	System Network Configuration Discovery
Regsvr32	File System Permissions Weakness	Path Interception	Execution Guardrails	Securityd Memory	System Network Connections Discovery
Rundll32	Hidden Files and Directories	Plist Modification	Exploitation for Defense Evasion	Steal Web Session Cookie	System Owner/User Discovery
Scheduled Task	Hooking	Port Monitors	Extra Window Memory Injection	Two-Factor Authentication Interception	System Service Discovery
Scripting	Hypervisor		File and Directory Permissions Modification		System Time Discovery
Service Execution			File Deletion		Virtualization/Sandbox Evasion

Figure 11. Malware detection result using our system versus MITRE matrix

The result shown in Figure 11, gives the following behavioral signs of events:

- Hooking (Level 5 – Dangerous): A single function mechanism can block events (message, keyboard, mouse, etc.)

before they are sent to the application's queue.
- File and Directory Discovery (Level 1 – Normal): Lists files and directories or can search in specific locations of servers or networks for certain information in the file system.
- System Time Discovery (Level 1 – Normal): Collect system time and / or time zone from local or remote system.
- Detecting more InstallUtil behavior.

From Figure 13 we conclude that the executable file which we tested is malware. When comparing this result with the analysis at [15] we can see our results in more detail when we discover the InstallUtil behavior of the malware. This is a technique that malware uses to execute and can bypass existing anti-virus tools on the market. Hackers can use the InstallUtil technique to execute executable code through a trusted utility of the Windows operating system.

V. CONCLUSIONS

In this paper, we have proposed a malware detection model on Workstation. The science and novelty of our paper are expressed in data collection and processing. Unlike traditional studies, in this article, all the behavior and signs of malware are collected on the operating system kernel and then analyzed and evaluated so that even though the malware is very sophisticated, their behaviors are recorded and evaluated. This research has great significance in anomaly detection systems in the network because the current monitoring and analysis systems are in data overload which leads to insufficient memory to build the Event behavior. The empirical results in our paper are only at a simple level. In subsequent studies, to improve the effectiveness of the process of monitoring and detecting malware, we will improve the Event ID weighting method and apply the fuzzy theory to the Event ID record evaluation.

REFERENCES

1. Alireza Souri, Rahil Hosseini. A state-of-the-art survey of malware detection approaches using data mining techniques (2018) 8:3. pp 1-22. https://doi.org/10.1186/s13673-018-0125-x.
2. YANFANG YE, TAO LI, DONALD ADJEROH, S. SITHARAMA IYENGAR. 2017. A survey on malware detection using data mining techniques. ACM Comput. Surv. 50, 3, Article 41 (June 2017), 40 pages. DOI: http://dx.doi.org/10.1145/3073559.
3. IMPORTANT INFORMATION REGARDING SANDBOXIE VERSIONS. https://www.sandboxie.com/. [Last accessed 26 February 2020]
4. Endpoint Detection and Response Solutions Market- https://www.gartner.com/reviews/market/endpoint-detection-and-response-solutions. [Last accessed 26 February 2020]
5. Endpoint Security with Apex One Endpoint security redefined. https://www.trendmicro.com/en_us/business/products/user-protection/sps/endpoint.html [Last accessed 26 February 2020]
6. Palo Alto Networks Traps Endpoint (EDR). https://paloaltofirewalls.co.uk/palo-alto-traps-endpoint/ [Last accessed 26 February 2020]
7. Kaspersky Endpoint Detection and Response- https://www.kaspersky.com/enterprise-security/endpoint-detection-response-edr [Last accessed 26 February 2020]
8. VMware Carbon Black EDR - https://www.carbonblack.com/products/edr/ [Last accessed 26 February 2020]
9. Falcon Insight: EDR -https: //www.crowdstrike .com / endpoint-security-products / falcon-insight-endpoint-detection-response /[Last accessed 26 February 2020]
10. https://www.malwarebytes.com/business/endpointdetectionresponse/ [Last accessed 26 February 2020]
11. Sysmon v10.42. https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon [Last accessed 26 February 2020]
12. Auditd Linux Tutorial. https://linuxhint.com/auditd_linux_tutorial/. [Last accessed 26 February 2020]
13. ATT&CK for Industrial Control Systems. https://attack.mitre.org/.[Last accessed 26 February 2020]
14. Adam Meyers, "Meet CrowdStrike's Adversary of the Month for June: MUSTANG PANDA," CrowdStrike Blog, accessed September 17, 2019, published June 15, 2018, https://www.crowdstrike.com/blog/meet-crowdstrikes-adversary-of-the-month-for-june-mustang-panda/. [Last accessed 26 February 2020]
15. China-Based APT Mustang Panda Targets Minority Groups, Public and Private Sector Organizations. https://www.anomali.com/blog/china-based-apt-mustang-panda-targets-minority-groups-public-and-private-sector-organizations [Last accessed 26 February 2020]

[1]

AUTHOR PROFILE

Dr. Do Xuan Cho is currently a lecturer at the Faculty of Information Technology at FPT University in Vietnam. In 2008, received a bachelor's degree in the Saint Petersburg Electrotechnical University "LETI" on a specialty "Computer science and computer facilities", Russia. In 2010, graduated a masters from the Saint Petersburg Electrotechnical University "LETI" on a specialty "Computer science and computer facilities", Russia. In 2013, received a PhD in the Saint Petersburg Electrotechnical University "LETI", on a specialty CAD. Russia. Area of scientific interests - modeling, control systems, algorithmization.Email: **chodx@fe.edu.vn**

Do Hoang Long, Nguyen Vuong Tuan Hiep, Le Quang Sang are fourth-year students majoring in information security at FPT University. These students have over 2 years of experience working with malicious attack detection issues.

Email: ***, longdhse05220@fpt.edu.vn, hiepnvtse05065@fpt.edu.vn, sanglqse04676@fpt.edu.vn***

Second Author: My position is the professor of Institute of computer sciences and technologies in Peter the Great Saint-Petersburg Polytechnic Ubiversity. I have received the degree Doctor of Technical Sciences in 1998 in accordance of scientific speciality "Systems of automatic Desing" in SPbPY. The area of scintific interest is use of new type of fuzzy logics in different applications. I think that we could cooperate intensively in future.

Email: ***v_tisenko@mail.ru***