Journal of Theoretical and Applied Information Technology
30th September 2020. Vol.98. No 18

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MALWARE ANALYSIS BASED ON SMART AGENTS AND IMAGE CLASSIFICATION

[1]RODOLFO ROMERO-HERRERA, [2]JUAN ANTONIO JIMÉNEZ GARCÍA, [3]VICTOR MANUEL SILVA GARCÍA

[1]IPN-ESCOM, Department of CIC, México

[2,3]IPN-CIDETEC, Department of Posgraduate, México

E-mail: [1]rromeroh@ipn.mx

ABSTRACT

Windows-based systems and operating systems in general are significantly damaged, affecting infrastructures. At present, Malware analysis is performed in laboratories that use high costs and resources; so there are few methods of classification of Malware, based on artificial intelligence that consumes few resources. This article provides a system that was developed for the dynamic analysis of malware in Windows and classified using SIFT, SURF, and Bayesian networks. This involves the transformation of infected files into image files that allows the identification and classification of Malware. The samples of malicious software that allows generating a contingency plan were identified. The system was developed using intelligent agents. The analysis of Postal worm malware is presented as an example. When comparing with other malware detection and classification systems, it is observed that the multi-agent-based system is competitive.

Keywords: *Smart agent, classifier, malware, analysis, SIFT, SURF.*

1. INTRODUCTION

Cybersecurity is a discipline that has developed exponentially, due to the emergence of new technologies, some integrating into everyday things by connecting to the internet. This generates a large amount of information from individuals, institutions, companies, and even countries, which has attracted the attention of cybercriminals, compromising the pillars of security (confidentiality, integrity, and availability), exposing systems to various threats. Malware (Malicious Software) is a set of instructions that are processed by the computer equipment and make the system do what the attacker wants [1]. There are several classifications of malware, among the best known are the virus, Trojans, backdoors, worms, bots, spyware, and adware [2].

In terms of complexity, every day the evolution of malware makes it more difficult to recover information from infected systems. The diversity of malware has caused not only typical computer systems to be affected, but also Smartphones, tablets, and even new Smart TVs [3]. The excessive amount of malware per day complicates its analysis, demanding high computational and human resources.

It is clear that there is a constant and joint work of cybercriminals; therefore, intelligent systems capable of dealing with these threats must be created. Windows operating systems are the most used by users, which are more vulnerable to threats, which is why attackers focus on developing malware. The main objective of the project is a multi-agent system to perform dynamic analysis of malware in Windows operating systems, which provides us with information on the malware to classify, it using image processing techniques, and identification of malware samples to plan a containment plan. This was accomplished by creating smart agents for malware analysis and implementing an image-based threat classifier.

1.1 Research Question

Based on existing theories and knowledge, two research questions are posed:

- Is it feasible to use smart agents as an alternative to Malware detection?
- Is it possible to classify Malware by pattern recognition of infected files and converted into images?

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

1.2 Research Hypothesis

Malware detection can be done with low consumption of material resources by:

- Converting infected files into images where Malware can be seen.
- The recognition of patterns in images that allow a classification.
- The use of intelligent agents for analysis using Tropos methodology.

1.3 Justification

The amounts of Malware are enormous, which complicates their classification; however, it is important to continue with the study and analysis, since the economic impact, as well as the damage to the systems infrastructure, is in many cases irreversible. Existing systems turn out to be laboratories with complex real-time systems; For this reason, it is proposed to develop an intelligent environment that provides us with information on malware to classify it and identify new malware samples, which allows us to be prepared by having a plan to contain the threat once identified, following the stages of a response to the incident.

2. STATE OF THE ART

The complexity and risk of malware have sparked various investigations. Host-based antivirus systems have been proposed, with the ability to provide meaningful malware information; thus, tools used by attackers have also been used [4]. One of them is through images such as Nataraj, Yegneswaran that propose a third form of analysis different from static and dynamic analysis called "binary-texture", which is 4000 times faster than dynamic analysis to classify malware; but even with the disadvantage of not knowing the behavior of the malware, the sample binary becomes a grayscale image [5]. Zhang together with other researchers presents four research papers. The first is based on the extraction of the opcodes from the sample, forming images which are analyzed with grouping algorithms [6]. The second one proposes a new algorithm called Dual-Lane AdaBoost for malware detection which introduces semi-supervised learning. The third one obtains characteristics of the samples that allow them to be grouped by families employing alliance algorithms using a client-server architecture [7, 8, 9].

Malware classification becomes a difficult task as the emergence of new variants is accelerated. This is how Yusoff and Jantan propose a way to classify malware according to its objective and behavior called "Class Target Operation" (CTO) [10]. Thomas and Marinescu propose a classification of malware according to the interaction of the malware sample with other files using a graph [11]. In [12], random projections are used to have an easy but complex training in a neural network. Canzanese, Kam, and Mancoridis perform automatic online classification of new malware variants without previously knowing the family to which it belongs [13]. Systems like the so-called Malfinder present very good results in the classification of malware; although the number of samples in the tests is very small [14]. Lim, Yamaguchi, Shimada, and Takakura used network traffic flow using a clustering algorithm (K-means) to classify malware and detect new malware families [15]. González and Vázquez propose a feature vector based on dynamic link libraries that malware uses. Using a multilayer perceptron for the classification of malware (worms and Trojans) in [16], they developed a system to identify malicious files in the cloud when they are transmitted from the client to the server. Ma, Biao, Yang, and Jiang perform static and dynamic malware analysis to reduce false positives using three classifiers: Support Vector Machine (SVM), Trees, and Bays. Also using SVM together with Random Forest and Chia neural networks, Ordóñez and Cepeda use the Virus Total online API [17,18]. In [19] various classifiers are tested such as SVM, decision tree, random forest, Stochastic Gradient Descent (SGD), KNearest Neighbor (KNN), Bernoulli Naive Bayesian, and Multinomial Naive Bayesian. Aminu, Woodhead, and Gan detect the infection caused by worms through datagram analysis and create the containment plan [20]. Moore and Hahsler propose malware detection based on evasion techniques, with characteristics of polymorphic malware using sequence classification methods [21].

3. THEORETICAL FRAMEWORK

The dynamic analysis consists of running the malware sample in a controlled environment, observing its behavior, changes, and the interaction it has with the medium [22].

3.1 Malware Analysis

Malware analysis allows us to understand their behavior, the means of propagation, obfuscation techniques, and evolution, and mutation; thus, techniques can also be designed to

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

prevent and/or eradicate it. The analysis is performed on the malware code [23]. In the specific case of Windows operating systems, the file system, registry keys, processes, network connections, and data traffic are monitored.

The file system used by Windows is the NTFS which organizes the files into directories. The activity in the file system consists of identifying the files created, modified, changed of location, and deleted during and after the execution of the malware.

Records in Windows are used to:

1. User profiles.
2. Applications installed on the computer.
3. The types of documents that each application can create.
4. The settings of the property sheets for folders.
5. The application icons.
6. The hardware elements that are in the system.
7. The ports that are being used.

The database is divided into keys, which are listed in Table 1, where a brief description is presented in it [24].

Table 1: Windows registry keys (reproduced from [24]).

DEFAULT KEY	DESCRIPTION
HKEY_CURRENT_USER	It contains the information of the logged-in user. For example, control panel settings, display settings, etc.
HKEY_USERS	Contains all user profiles actively loaded on the computer.
HKEY_LOCAL_MACHINE	It contains equipment specific configuration information (for any user).
HKEY_CLASSES_ROOT	It is a subkey of HKEY_LOCAL_MACHINE \ Software. The information stored here ensures that when you open a file with Windows Explorer, the correct program will open. The HKEY_LOCAL_MACHINE \ Software \ Classes key contains the default settings that can be applied to all users on the local computer. The HKEY_CURRENT_USER \ Software \ Classes key contains the settings that override the default settings and apply only to the interactive user. The HKEY_CLASSES_ROOT key provides a view of the registry that combines the information from these two sources. HKEY_CLASSES_ROOT also provides a combined view for programs designed for earlier versions of Windows.
HKEY_CURRENT_CONFIG	It contains information about the hardware profile that the local computer uses when the system starts.

The auxiliary files for each section are located at the address: C: \ Windows \ sytem32 \ config. The database is organized in the form of a tree with a specific structure as we can see in Figure 1, as well as its data of types (see Table 2).

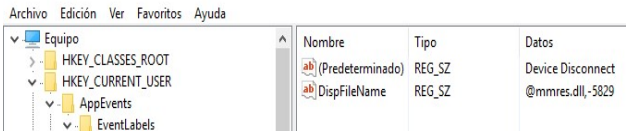

Figure 1. Structure of the Windows

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Tabla 2. Types of data in the Registry [24]

NAME	TYPE OF DATA	DESCRIPTION
Binary value	REG_BINARY	Raw binary data. Most information about hardware components is stored in binary data and is displayed in hexadecimal format in Registry Editor
DWORD value	REG_DWORD	Data represented by a number 4 bytes long (a 32-bit integer value). Many device drivers and service parameters are of this type and are displayed in the Registry Editor in binary, hexadecimal, or decimal formats. DWORD_LITTLE_ENDIAN (a least significant byte is in bottom address) and REG_DWORD_BIG_ENDIAN (a least significant byte is in top address) are related values.
Expandable alphanumeric value	REG_EXPAND_SZ	Variable-length data string. This type of data includes variables that are resolved when a program or service uses the data.
Multiple string value	REG_MULTI_SZ	Multiple chains. Values containing lists or multiple values; This is the format that is easier to read. Entries are separated by spaces, commas, or other punctuation marks.
String value	REG_SZ	Fixed-length text string.
Binary value	REG_RESOURCE_LIST	A series of nested arrays are designed to store a list of resources used by the controller of a hardware device or one of the physical devices it controls. The system detects and writes this data to the \ ResourceMap tree that is displayed in the Registry Editor in hexadecimal format as a binary value.
Binary value	REG_RESOURCE_REQUIREMENTS_LIST	A series of nested arrays designed to store a list of device drivers of possible hardware resources that the driver, or one of the physical devices it controls, can use. The system writes a subset of this list in the \ ResourceMap tree. The system detects this data and displays it in the Registry Editor in hexadecimal format as a binary value.
Binary value	REG_FULL_RESOURCE_DESCRIPTOR	A series of nested arrays are designed to store a list of resources used by a physical hardware device. The system detects and writes this data in the \ HardwareDescription tree that is displayed in the Registry Editor in hexadecimal format as a binary value.
None	REG_NONE	Data without any particular type. The system or an application writes this data to the Registry and displays it in the Registry Editor in hexadecimal format as a binary value.
Link	REG_LINK	The Unicode string that names a symbolic link.
QWORD value	REG_QWORD	Data represented by a 64-byte integer. This data is displayed in the Registry Editor as a binary value and was first entered in Windows 2000.

All those processes that have been modified, eliminated or impersonated by malicious software that has to be identified. The services that the malware tries to access must also be identified, as well as the exchange of data that occurs in the network flow; In this activity, it is important to recognize the IP addresses, ports and domains involved [25, 26].

3.2 SIFT (Scale Invariant Feature Transform)

Algorithm for extraction of characteristics by key points and calculation of descriptors:

Characteristics:

Extreme space detection at scale. The Gaussian Laplacian acts as a detector for regions in various sizes due to the change in the σ or scale parameter. Local maxima can be found through scale and space, with values (x, y, σ), which means that there are potential key points in (x, y) at σ scale. Due to the high computational cost, the SIFT algorithm uses the Gauss difference. Figure 2 shows this process which is performed for different octaves of the image in the Gaussian pyramid.

Once the Gaussian difference is obtained, look for the scale that is best represented at the key point, see Figure 3.

Location of key points. The Taylos scale is used to obtain a more precise location of the key points [27].

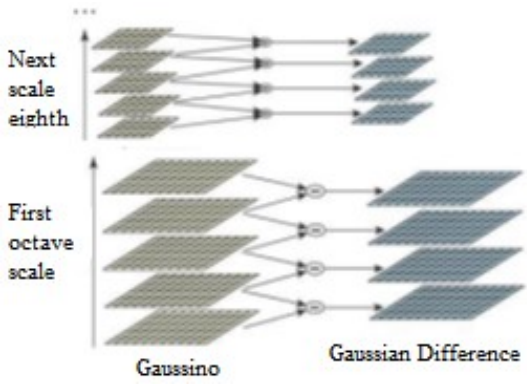

Figure 2: Gaussian difference [27]

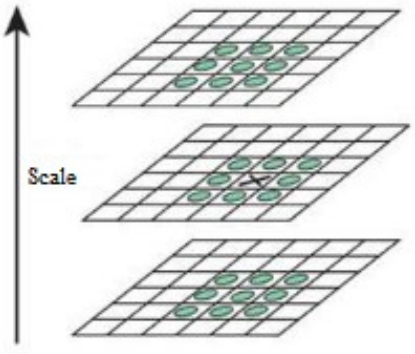

Figure 3. Scaled image [27]

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Orientation assignment. An orientation to achieve invariance is assigned to each key point in the image rotation.

Description of key points. A 16x16 neighborhood is taken around the key point, divided into 16 sub-blocks of 4x4. There are a total of 128 values available to represent a key point.

The coincidence of key points. The key points between two images coincide, identifying their closest neighbors [27].

3.3 SURF (Speeded-Up Robust Features)

It differs from SIFT in that it uses a box filter. See figure 4. A great advantage of this approach is that the convolution with the box filter can be calculated with integral parallel images for different scales.

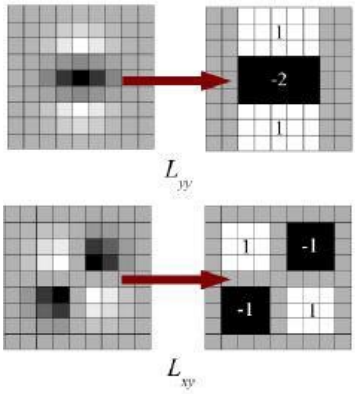

Figure 4. Box filter

SURF uses wavelets in the horizontal and vertical direction for assigning the orientation and traces a space as shown in Figure 5. It uses a 128-dimensional descriptor. Features are only compared if you have the same type of contrast as shown in Figure 6, allowing faster matching without reducing performance [28].

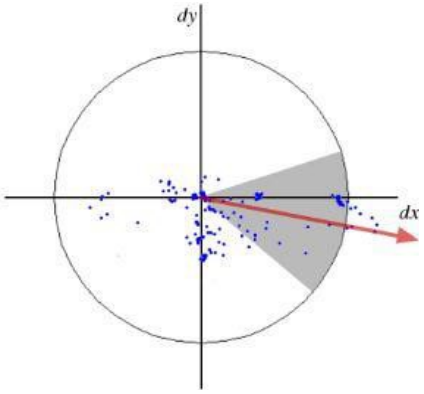

Figure 5. Sample space for orientation allocation [28].

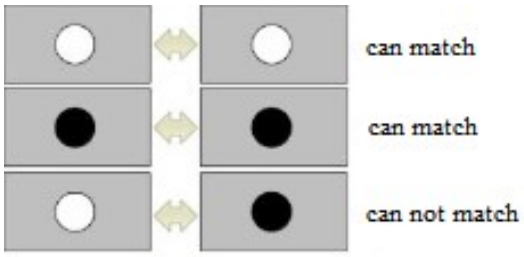

Figure 6. Contrast comparison (reproduced from [28]).

3.4 Network of Bayes

Bayesian networks model a phenomenon using a set of variables and the dependency relationships between them. The posterior probability of the unknown variables can be estimated based on the known variables. [29].

3.5 Binary texture

For malware detection, a dynamic and / or static analysis is used. However, in [5] a process was created through which the sample is transformed into an image having multiple advantages [5]:

1. Speed when classifying.
2. Small changes can be observed in the original sample.
3. The images show a similar structure between malware families.
4. Avoid disassembling the sample.

To obtain the image, the binary code of the malware is first obtained, then the binary code is ordered in 8-bit vectors and finally, the grayscale image is created. The width of the image should allow observing the various sections of the binary code. Measurements can be selected based on Table 3. Thus the height of the image depends on the size of the malware sample.

Tabla 3. Image widths according to the size of the malware.

SAMPLE SIZE	WIDTH OF THE IMAGE
< 10 KB	32
10 KB – 30 KB	64
30 KB – 60 KB	128
60 KB – 100 KB	256
100 KB – 200 KB	384
200 KB – 500 KB	512
500 KB – 1000 KB	768
> 1000 KB	1024

Figure 7 shows the image obtained at the end of the process. The segments into which it is divided coincide with the segments of an

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

executable file (exe). The segment ".text" corresponds to the code where the magic number and the program code are stored. The ".rdata" to the read-only data; ".Data" to the general content of the file and the initialized variables. The ".rsrc" corresponds to the resources of the program.

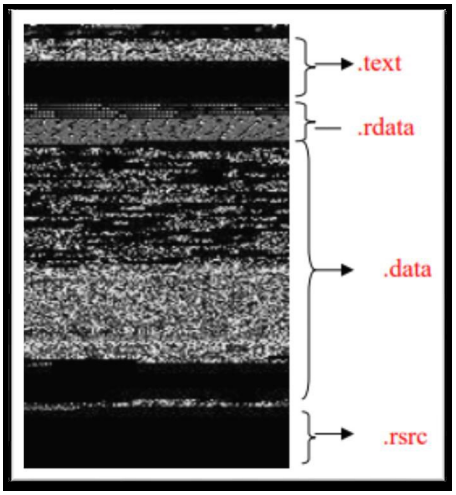

Figure 7. Segments of the image.

3.6 Design

To use the system, a host computer is required, which will host 2 virtual machines, the repository with the malware samples and the knowledge base. The malware samples are run on the virtual machine 1 for analysis; virtual machine 2 is responsible for capturing the network traffic generated by the malware. The main purpose of the architecture is to capture all the network traffic generated by the sample. The scanning machine is configured in such a way that all traffic is redirected to the traffic capture machine. Figure 8 shows the configuration of the properties of IPv4 (Internet Protocol version 4), the gateway contains the IP address of the analysis machine. A transparent proxy is used in Debian 8 to have more control over network traffic; Squid 3 is used, so 2 network adapters are required. The eth0 network adapter is configured with the IP address of the internal network and the eth1 network adapter is configured with the IP address that has access to the internet; both IPs are configured statically. Through IPTables, the traffic received by eth0 is redirected to port 3128, which Squid 3 uses by default.

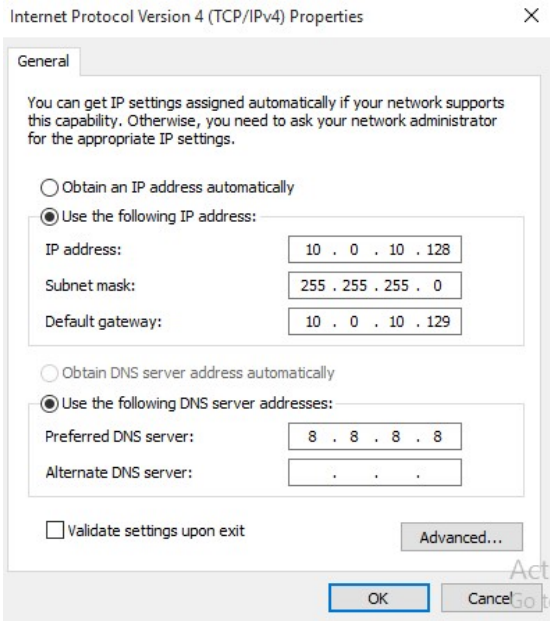

Figure 8. IPv4 properties.

For the analysis and development of the agents, the Tropos methodology was used [30]. The technique is oriented to the development of intelligent agent software. It is based on two ideas: the first, from the initial stage to implementation, mainly considers the agents, as well as their goals and plans. The second covers the project analysis stage, which allows understanding the agents' operating environment and their interaction with it [31]. Tropes consist of six phases:

Analysis of early requirements. This stage identifies and analyzes the parties involved and their intentions. Figure 9 shows the actors in the system and the relationship that exists between them to meet their goals respectively. The goals of the actor named Malware were defined according to the Instituto de Seguridad de España [32].

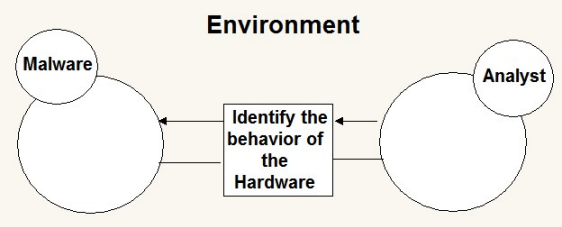

Figure 9. System actors.

The goals of a malware analyst are: to understand how the sample works to prevent the spread, create a response plan for infection, identify vulnerabilities, become familiar with malware development methods and techniques.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Analysis of late requirements. At this stage a new actor is included which is the multi-agent system; The dependencies that are created in the environment when it is added are also inserted (see Figure 10).

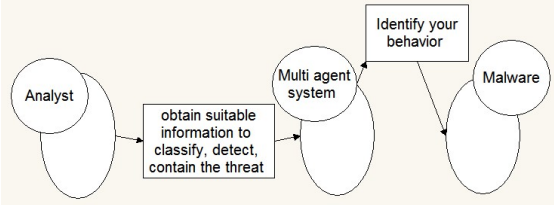

Figure 10. Diagram of late requirements.

Architecture.

Agents involved in the system are displayed. Each agent performs a series of goals, socializes with other agents, and manages the resources to achieve these goals. There are six agents involved (see Figure 11).

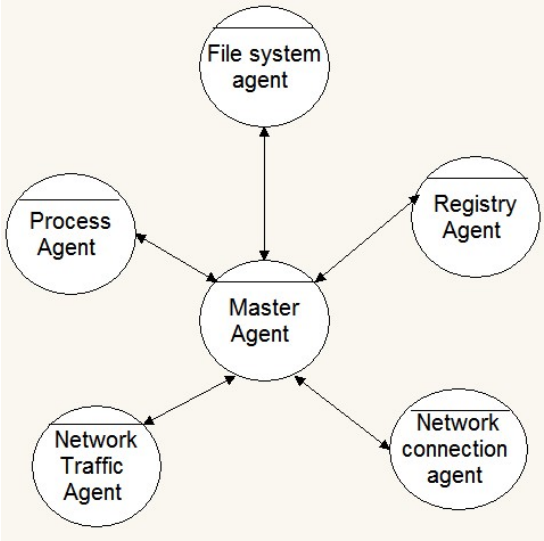

Figure 11. Smart agents.

The master agent is reactive, it acts according to the information provided by the other agents and its main goals are to start the execution and analysis of malware, obtain characteristics, collect information, unify the information and coordinate the activity. from the other agents in a controlled environment.

The file system agent is of the reactive type; it acts according to the activity it detects in NTFS. Its goals are to detect the files created, deleted, and modified, while the malware is running.

The registry agent is a filter-type agent, which makes a copy of the key registry before and after the execution of the malware. With both copies, it makes a comparison between files and reports the changes. So I could detect the keys of new and deleted records; as well as name and value changes in the keys.

The network connection agent is reactive and acts according to the activity of the network connections. If the connection is considered malicious, the destination IP address and the ports involved are stored.

The network traffic agent is a reactive agent that acts according to the information captured by the network connection agent, based on this information, it analyzes the captured traffic to obtain suspicious domains, that the malware consults, either to obtain resources or establish a communication channel.

The Process Agent acts depending on the sample and what is derived; stores the name and identifier to have control over it, and determine the resources created or the connections created by the main process.

4. DEVELOPING

Tropos methodology was used for the design of agents, and analyzes were carried out mainly with malware such as Postal worm, Cryptolocker, Capture, and death of Osama Bin Laden, etc. Take the Worm Postcard as an example because it is a sample that shows activity in each of the variables to be analyzed (processes, network connections, file systems, and the Windows registry), although the procedure is the same for the other cases. Worm Postcard is a malware that is distributed by email.

Table 4 shows the information to identify the malware sample. It indicates the name of the file that is being analyzed, its size, the format it is in, and if you use any Hash SHA1 (Secure Hash Algorithm 1) function to identify it.

Table 4. Information about the worm postal malware

Details	
Name	Postcard.exe
Size	247 KB
Format	EXE
SHA1	5a923137a7fcbe5b0e5b80bd9ca9cfacba6dd0c5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Dynamic analysis results were obtained with tools such as RegShot for monitoring registry keys and the file system, Process Explorer to see system processes, TCPView to monitor network connections, and TCPdump for packet capture.

The process called gui.exe is created and from this, the process for the browser installed on the computer is derived. See Figure 12.

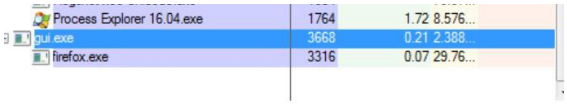

Figure 12. Processes created

Of the keys obtained as a result with RegShot, the registry key at the address HKLM \ SOFTWARE \ Microsoft \ Windows \ CurrentVersion \ Run is important, since with it the software guarantees persistence on the computer causing the gui.exe file to run every time the computer starts.

In the file system, there are JavaScript, HTML, and gif types among others. Figure 13 shows the new files on the system that RegShot has identified.

```
C:\Windows\System32\404.html
C:\Windows\System32\bcp\index.html
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\0.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\1.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\2.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\3.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\4.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\5.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\6.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\7.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\8.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\9.gif
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\comunes.js
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\login.js
C:\Windows\System32\bcp\OperacionesEnLinea\archivos\zona_segura.css
C:\Windows\System32\bcp\OperacionesEnLinea\clave.html
C:\Windows\System32\bcp\OperacionesEnLinea\clave_digital.html
C:\Windows\System32\bcp\OperacionesEnLinea\clave_e.html
```

Figure 13. Files created.

In the traffic capture, there is a domain name "www.claro.com.pe"; which probably serves to complete the malware objective or to send information collected from the victim (see Figure 14).

```
17:51:45.307092 IP 8.8.8.8.53 > 172.16.16.151.52290: 24654 4/0/0 CNAME www.cl
aro.com.pe.edgesuite.net., CNAME a908.b.akamai.net., A 201.149.60.90, A 201.1
49.60.105 (138)
        0x0000:  000c 29ca 40d4 0050 56e1 66dd 0800 4500  ..).@..PV.f...E.
        0x0010:  00a6 3fdd 0000 8011 2db3 0808 0808 ac10  ..?.....-.......
        0x0020:  1097 0035 cc42 0092 01f6 604e 8180 0001  ...5.B....`N....
        0x0030:  0004 0000 0000 0377 7777 0563 6c61 726f  .......www.claro
        0x0040:  0363 6f6d 0270 6500 0001 0001 c00c 0005  .com.pe.........
        0x0050:  0001 0000 036b 0020 0377 7777 0563 6c61  .....k...www.cla
        0x0060:  726f 0363 6f6d 0270 6509 6564 6765 7375  ro.com.pe.edgesu
        0x0070:  6974 6503 6e65 7400 c02e 0005 0001 0000  ite.net.........
```

Figure 14. Captured packet.

From figure 14 the analysis shows the characteristics of malware that can be detected dynamically in a controlled environment.

To obtain a classification of the malware, a data set consisting of 860 images is obtained, divided as shown in Table 5. Families are determined using Microsoft Security Essentials. Figure 15 shows a grayscale image of the malware [10].

Table 5. Malware families.

Families	Quantity
Autorun	90
Dialplatform.B	177
Dontovo.A	162
Instantaccess	431
Total:	860

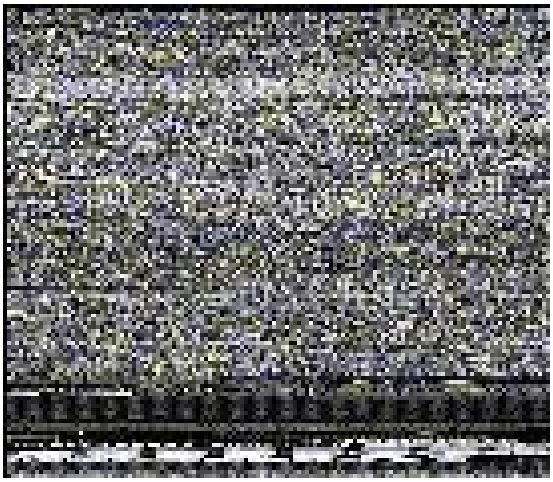

Figure 15. Grayscale malware.

Two SIFT and SURF algorithms were used for the classification; the results obtained are different as can be seen in Figures 16 and 17; the image is of malware of the Autorun family, the colored circles indicate each of the characteristics found with SIFT and its concentration in the ".rsrc" segment can be appreciated; however, in Figure 17 when applying SURF more circles are seen in the figure.

Journal of Theoretical and Applied Information Technology
30th September 2020. Vol.98. No 18

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 16. Autorun SIFT

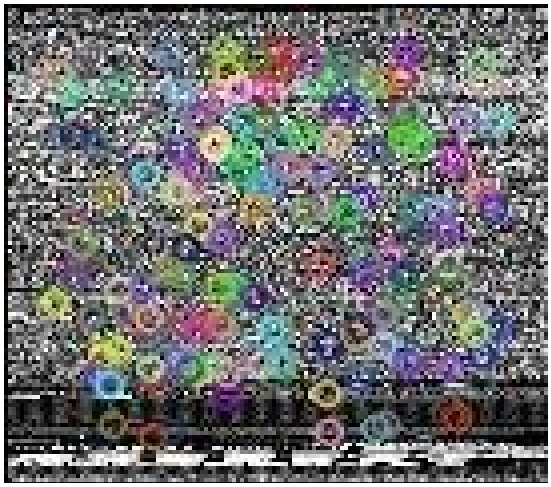

Figure 17. Autorun SURF.

5. TEST AND RESULTS

Malware connectivity tests were carried out, obtaining results through intelligent agents when analyzing three malware samples and it was compared with other automatic analysis tools such as Cuckoo through page malwr.com and page virustotal.com .

5.1 Malware connectivity test

It is extremely important to verify that the communication in the system to study malware follows the proper flow, that is, to verify that the communication from virtual machine 2 to the virtual machine is null; therefore, it is checked using the ping command (see Figure 18); and all sent packages are lost. In Figure 21 the communication test is from virtual machine 1 to virtual machine 2 to verify that all packets arrive at virtual machine 2 and then analyze the captured traffic.

```
C:\Users\Juan\Desktop\Agentes\Agentes>ping 10.0.10.129

Pinging 10.0.10.129 with 32 bytes of data:
Reply from 10.0.10.129: bytes=32 time<1ms TTL=64
Reply from 10.0.10.129: bytes=32 time<1ms TTL=64
Reply from 10.0.10.129: bytes=32 time=1ms TTL=64
Reply from 10.0.10.129: bytes=32 time=2ms TTL=64

Ping statistics for 10.0.10.129:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms
```

Figure 18. Ping from MV1 to MV2.

5.2 Worm postcard

Figure 19 shows the report generated by the malware analysis system for the worm Zip file, the identifier SHA1 is indicated in light green, the deleted and created registry keys are shown in yellow, the connections are shown in pink The domains identified have been established, in strong green, in blue the files with which there was interaction and in red the processes created during the execution of the malware.


```
{"5a923137a7fcbe5b0e5b80bd9ca9cfacba6dd0c5":[
{"RegNuevas":[{"llave":"4,f1,56,94,65,d2,01,00,00,00,00"},
{"llave":"HRZR_PGYFRFFVBA=hex:00,00,00,00,49,01,00,00,18,08,00,0
0,29,7d,bc,05,4d,00,00,"},
{"llave":"00,a4,01,00,00,e2,0f,d7,00,7b,00,31,00,41,00,43,00,31,
00,34,00,45,00,37,00,"},
{"llave":"Zvpebfbsg.NhgbTrarengrq.{Q8Q180NR-44Q4-1854-6S4P-55S25
66QR2N7}=hex:00,00,00,"},
{"llave":"00,14,00,00,00,75,01,00,00,b1,da,d9,00,00,00,80,bf,00,
00,80,bf,00,00,80,bf,"},
{"llave":"00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,
80,bf,00,00,80,bf,00,"},
{"llave":"C@@Program Files@WinRAR=C:\\Program Files\\WinRAR"},
{"llave":"C@@WINDOWS@system32@=C:\\WINDOWS\\system32\\"},
{"llave":"14,f1,56,94,65,d2,01,00,00,00,00"},
{"llave":"HRZR_PGYFRFFVBA=hex:00,00,00,00,49,01,00,00,18,08,00,0
0,29,7d,bc,05,4d,00,00"},
{"llave":"00,a4,01,00,00,e2,0f,d7,00,7b,00,31,00,41,00,43,00,31,
00,34,00,45,00,37,00"},
{"llave":"Zvpebfbsg.NhgbTrarengrq.{Q8Q180NR-44Q4-1854-6S4P-55S25
66QR2N7}=hex:00,00,00"},
{"llave":"00,14,00,00,00,75,01,00,00,b1,da,d9,00,00,00,80,bf,00,
00,80,bf,00,00,80,bf,"},
{"llave":"00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,
80,bf,00,00,80,bf,00,"},
{"llave":"C@@Program Files@WinRAR=C:\\Program Files\\WinRAR"},
{"llave":"C@@WINDOWS@system32@=C:\\WINDOWS\\system32\\"}]},
{"RegEliminadas":[{"llave":"14,f1,56,94,65,d2,01,00,00,00,00"},
{"llave":"HRZR_PGYFRFFVBA=hex:00,00,00,00,49,01,00,00,17,08,00,0
0,29,7d,bc,05,4d,00,00,"},
{"llave":"00,a4,01,00,00,e2,0f,d7,00,7b,00,31,00,41,00,43,00,31,
00,34,00,45,00,37,00,"},
{"llave":"Zvpebfbsg.NhgbTrarengrq.{Q8Q180NR-44Q4-1854-6S4P-55S25
66QR2N7}=hex:00,00,00"},
{"llave":"00,14,00,00,00,74,01,00,00,b1,da,d9,00,00,00,80,bf,00,
00,80,bf,00,00,80,bf,"},
{"llave":"00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,
80,bf,00,00,80,bf,00,"},
{"llave":"C@@Program Files@WinRAR=C:\\Program Files\\WinRAR"},
{"llave":"14,f1,56,94,65,d2,01,00,00,00,00"},
{"llave":"HRZR_PGYFRFFVBA=hex:00,00,00,00,49,01,00,00,17,08,00,0
0,29,7d,bc,05,4d,00,00,"},
{"llave":"00,a4,01,00,00,e2,0f,d7,00,7b,00,31,00,41,00,43,00,31,
00,34,00,45,00,37,00,"},
{"llave":"Zvpebfbsg.NhgbTrarengrq.{Q8Q180NR-44Q4-1854-6S4P-55S25
66QR2N7}=hex:00,00,00"},
{"llave":"00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,80,bf,00,00,
80,bf,00,00,80,bf,00,"},
{"llave":"C@@Program Files@WinRAR=C:\\Program Files\\WinRAR"}]},
{"Archivos":[{"nombre":"C:\\Windows\\SysWOW64\\riched20.dll"},
{"nombre":"C:\\Windows\\Registration\\R000000000006.clb"},
{"nombre":"C:\\Windows\\SysWOW64\\en-US\\user32.dll.mui"}]},
{"Procesos":[{"nombre":"Postal_Gusanito.exe"},{"nombre":"gui.exe
"}]},
{"Conexiones":[{"ip":"0.0.0.0","puerto":80}]},
{"Trafico":[{"dominio":"www.claro.com.pe"},{"dominio":"www.claro
.com.pe.edgesuite.net"}]}]}
```


Figure 19. Analysis of the postcard worm.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 6 shows a comparison of the results of the analysis provided by page malwr, VirusTotaly, and the system proposed and developed by agents. VirusTotal limits the information it provides regarding the behavior of the sample but stands out in the static analysis. Regarding malwr in the files and registry keys part, it presents several false positives, the analysis by intelligent agents detected activity in both, although not totally. As for the processes, malwr fully developed the tree, while in the analysis by agents it reached half of the detections. In the malwr network connections, I present false positives while the analysis by intelligent agents does register most of it. The analysis time with smart agents is less compared to malwr.

Unlike the resources used like KNN [17] [18], and neural networks in [12], etc; here SIFT or SURFT was used for classification, supporting Zhang's proposal [8], but using intelligent agents within the analysis; Similar results are obtained to platforms such as those used in [20], but with less infrastructure. Two Core i7 2.2 GHz computers are used. See table 6.

Table 6. Comparison between worm postcard analyzers.

	INTELLIGENT AGENTS	VIRUSTOTAL	MALWR
Files	3	1	50+
Registry keys	29	-	150+
Processes	2	-	4
IP addresses	1	-	10
Domains	2	-	10
Analysis time (seconds)	95.65	-	135

5.3 Malware classification

Figure 20 and 22 show the two-dimensional graph where the "X" axis corresponds to the malware families and the "Y" axis corresponds to the SIFT or SURF characteristics, the blue points correspond to the "autorun" family, the red dots "dialplatform.b", the green dots to "dontovo.a" and the light blue dots correspond to "Instantaccess", you can see the distinctive grouping in various places on the graph of each family.

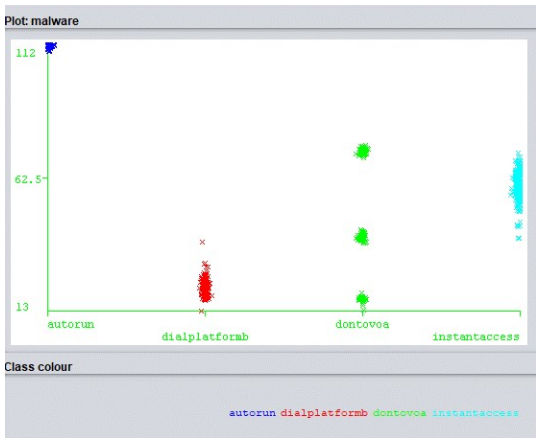

Figure 20. Graph. VS SIFT family.

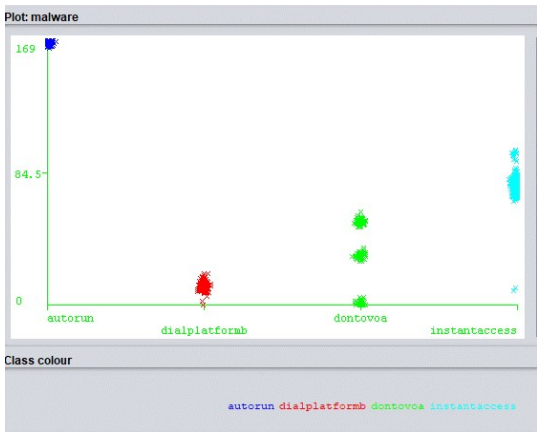

Figure 21. Graph. VS SURF family.

Figure 25 shows the graph in two dimensions where the “X” axis corresponds to the SIFT characteristics and the “Y” axis corresponds to the SURF characteristics.

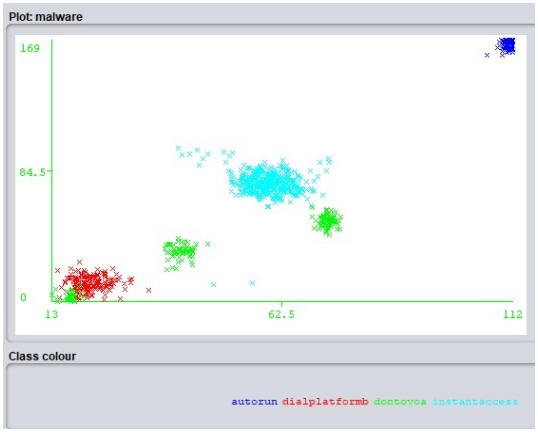

Figure 22. Graph. SIFT VS SURF.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 7 shows the accuracy obtained by each classifier; in general, all the classifiers present good results, varying among them by tenths of a percentage. The vector support machines obtained the worst result, the Euclidean classifier and the closest neighboring k classifier, configured with a value of k = 3, obtained the same percentage of accuracy, since both classifiers are based on the calculation of distances. The multilayer perceptron neural network and the LMT decision tree (Logistic Model Trees) obtain similar results; however, if we compare complexity, it is easier to implement a decision tree than a neural network. The Bayesian network classifier achieved the best result, using 40% of the database for training and 60% for validation; a confusion matrix is obtained, which is shown in Table 8.

Table 7. Evaluation of classifiers.

Classificatory	Exactitude %
Euclidian	99.41
K nearest neighbors	99.41
Bayesian network	99.61
Decision trees	99.53
Neural Networks	99.50
Vector support machines	99.12

Table 8. Confusion matrix.

	Autorun	Dialplatform.B	Dontovo.A	Instantaccess
Autorun	55	0	0	0
Dialplatfrom.B	0	103	2	0
Dontovo. A	0	0	97	0
Instantaccess	0	0	0	259

6. CONCLUSION

The purpose and objective of the research were fulfilled with the development of a multi-agent system capable of examining malware samples; upon completion, obtains the information necessary to implement a containment plan. Smart agents are capable of analyzing samples from three malware families; however, it is still necessary to test more existing malware families and adjust the agents according to the needs of the new families.

There are various methods to achieve interaction between agents such as creating protocols or sending requests through ports at the network level, however, if one of these robust communication methods is implemented, it would generate more activity on the equipment that could interfere in the behavior of malware.

The feasibility of the use of intelligent agents and the classification of Malware by image processing as a tool for the analysis of malicious software was verified.

The Bayesian network is recommended, although if a quick and simple implementation is sought, the Euclidean classifier could be implemented since only a few tenths of accuracy would be sacrificed, which may be imperceptible in the implementation.

The methodology and System developed used little infrastructure compared to Workstation; since as minimum resources to use, it is feasible to use two Intel Core i7 computers with a 2.20 GHz CPU, a Debian 8 operating system, and analysis for Windows 10. The second team has VMWare Workstation PRO as a guest and for the analysis is used Python 2.7; which results in a feasible implementation with fewer resources than those used in public laboratories.

7. FUTURE WORK

We are going to analyze more types of malware, and create more gray-scale images with other malware families to strengthen the learning of classifiers.

ACKNOWLEDGMENT

The support received to carry out the project to the Instituto Politecnico Nacional (IPN) and the Consejo Nacional de ciencia y Tecnología of Mexico (CONACYT) is acknowledged.

REFERENCES:

[1] Skoudis, E. & Zeltser, L. (2003). Malware: Fighting Malicious Code. Estados Unidos de América: Prentice Hall.

[2] Jiménez, J. R. & Soto, R. (2009). ¿Qué es malware? [Entrada en blog]. Usuario Casero. Recuperado de: http://www.seguridad.unam.mx/usuario-casero/eduteca/main.dsc?id=193. Consulta: 23 de julio del 2018.

[3] Eset. Guía de Respuesta a una Infección por Malware. Eset, 2015.

[4] Bailey, M. et al. (2007). Automated Classification and Analysis of Internet Malware. RAID'07.4637, 178-197.

[5] Nataraj, L., Yegneswaran, V., Porras, P. & Zhang, J. (2011). A Comparative Assessment

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

of Malware Classification using Binary Texture Analysis and Dynamic Analysis. AISec´11, 21-30. doi: 10.1145/2046684.2046689.

[6] Fraley, J. B. & Figueroa, M. (2016). Polymorphic Malware Detection Using Topological Feature Extraction with Data Mining. 2016 SoutheastCon. doi: 10.1109/SECON.2016.7506685.

[7] Zhang, X., Hou, Z., Zhu, X., Wu, G. & Wang, S. (2016). Robust Malware Detection with DualLane AdaBoost. 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). doi: 10.1109/INFCOMW.2016.7562248.

[8] Zhang, J. et al. (2016). Malware Variant Detection Using Opcode Image Recognition with Small Training Sets. 2016 25th International Conference on Computer Communication and Networks (ICCCN). doi: 10.1109/ICCCN.2016.7568542.

[9] Das, S., Xiao, H., Liu, Y. & Zhang, W. (2016). Online Malware Defense Using Attack Behavior Model. 2016 IEEE International Symposium on Circuits and Systems (ISCAS). doi: 10.1109/ISCAS.2016.7527492.

[10] Yusoff, M. N. & Jantan, A. (2011). Optimizing Decision Tree in Malware Classification System by Using Genetic Algorithm. International Journal of New Computer Architectures and their Applications (IJNCAA). 3 (1), 694-713.

[11] Karampatziakis, N., Stokes, J. W., Thomas, A. & Marinescu, M. (2012). Using File Relationships in Malware Classification. DIMVA´12, 1-20. doi: 10.1007/978-3-642-37300-8_1.

[12] Dahl, G. E., Stokes, J. W., Deng, L. & Yu, D. (2013). Large-Scale Malware Classification Using Random Projections and Neural Networks. International Conference on Acoustics Speech and Signal Processing (ICASSP).

[13] Canzanese, R., Kam, M. & Mancoridis, S. (2013). Toward an Automatic, Online Behavioral Malware Classification System. Seventh IEEE International Conference on Self-Adaptative and SelfOrganizing System (SASO). doi: 10.1109/SASO.2013.8

[14] Kim, T. et al. (2015). Malfinder: Accelerated Malware Classification System Through Filtering on Manycore System. 2015 International Conference on Information Systems Security and Privacy (ICISSP).

[15] Lim, H., Yamaguchi, Y., Shimada, H. & Takakura, H. (2015). Malware Classification Method Based on Sequence of Traffic Flow. 2015 International Conference on Information Systems Security and Privacy (ICISSP).

[16] González, L. E. & Vázquez, R. A. (2015). Clasificación de Malware Mediante Redes Neuronales Artificiales. Revista del Centro de Investigación. Universidad La Salle. 11(44), 69-102.

[17] [Ma, X., Biao, Q., Yang, W. & Jiang, J. (2016). Using Multi-Features to Reduce False Positive in Malware Classification. 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference. doi: 10.1109/ITNEC.2016.7560382.

[18] Chia, D. L., Cepeda, C. & Ordóñez, P. (2016). Feature Selection and Improving Classification Performance for Malware Detection. 2016 IEEE International Conferences on Big Data and Clod Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom). doi: 10.1109/BDCloud-SocialCom-SustainCom.2016.87.

[19] Mosli, R., Li, R., Yuan, B. & Pan, Y. (2016). Automated Malware Detection Using Artifacts in Forensic Memory Images. 2016 IEEE Symposium on Technologies for Homeland Security (HTS). doi: 10.1109/THS.2016.7568881.

[20] Aminu, M., Woodhead, S. & Gan, D. (2016). Early Containment of Fast Network Worm Malware. 2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS). doi: 10.1109/NICS.2016.7725649.

[21] Drew, J., Moore, T. & Hahsler, M. (2016). Polymorphic Malware Detection Using Sequence Classification Methods. 2016 IEEE Security and Privacy Workshops (SPW). doi: 10.1109/SPW.2016.30.

[22] Hale, M., Adair, S., Hartstein, B. & Richard, M. (2010). Malware Analyst's Cookbook and DVD: Tools and Techniques for Fighting Malicious Code. Estados Unidos de América: John Wiley & Sons Inc.

[23] Distler, D. ”Malware Analysis: An Introduction”. Sans Institute, diciembre 2017.