Proposed Classification of Malware, Based on Obfuscation

Cristian Barría
Pontifícia Universidad Católica de Valparaíso
Valparaíso, Chile
cristian.barria@udp.cl

Claudio Cubillos
Pontifícia Universidad Católica de Valparaíso
Valparaíso, Chile
claudio.cubillos@ucv.cl

David Cordero
Universidad Andrés Bello
Santiago, Chile
d.cordero.v@gmail.com

Miguel Palma
Universidad Tecnológica de chile
Santiago, Chile
miguel.palma06@inacapmail.cl

***Abstract* —Malware are the big threat within the digital world as they have a highly complex technological structure that is capable of penetrating networks, obtaining confidential information from personal computers and corporate systems, and even of making systems of critical infrastructure vulnerable. However, in order to achieve their objectives, they need to remain updated, so that they will not be detected by the different protection systems which re primarily antivirus. This investigation proposes a certain malware classification based on their obfuscation capacity, and also considering the methods, techniques, procedures and tools that a malicious code requires and that whose result suggests a general vision of the malware and its effective evasion in cyber space.**

***Index Terms* - Malware, obfuscation techniques, cyber space, antivirus.**

I. INTRODUCTION

A malware is a malicious programme that gets into a system without the user's authorization, and that executes undesirable actions. This term and the term antivirus are usually used interchangeably, although they are in fact different [2]. Initially, the use of malware was strictly associated to the investigation and protection of software's developers' intellectual property (using cryptography), but it's objective changed through time, becoming an important thing to obtain recognition through their operation. Nowadays their im is mainly related to profit-making [3]-[4].

The evolution of the malware began in 1949 when Von Neumann stablished the idea of stored programmes, and exposed the Automata Theory, where the possibility of developing small replicants of programmes, capable of gaining control of other programmes of similar structure, was first introduced [5]. While the concept has loads of applications in science, it is quite easy to apply it to viruses, considering that in that time, both concepts, virus and malware, were considered equivalent.

In 1971, Bob Thomas created the first malicious code, named Creeper, capable of infecting IBM 360 machines of the ARPANET network, which delivered an on screen message that said “I'm the Creeper, catch me if you can!”. In order to find it and eliminate it, another code called Reaper was created, programmed for this matter [6]. This is the origin of the current antivirus.

However, malware are looking to primarily affect the Critical Information Infrastructure (CII), which is the main concern of both the attackers and defenders. It is important to highlight that a successful virus will leave the CII defenseless. This is why, from both sides, the defenders' and offenders', arises the necessity of implementing a SGI whose objective is preserving the reliability, integrity and availability of the information [28] so that they will act upon those threats.

Furthermore, for this investigation, obfuscation will be understood as: the application of transformations in the code (source or binary), which changes the appearance of the malware through the realization of a series of steps, maintaining its functionality, that allowed the malware to make the control systems vulnerable [7]. For this, it is fundamental to carry out an investigation that contributes to the analysis of obfuscation from the perspective of the update of a malware. The tool that should be used to do this is a crypter that will allow us to present a certain classification of malware based on obfuscation.

This investigation presents a revision of related work in Section II, while in Section III the classes, types and generation of malware are exposed, from the obfuscation's perspective. In Section IV, the use of crypter for the obfuscation of detected malware is presented, in Section V a flow chart of the update and classification of malware is proposed, and finally, in Section VI the conclusions and future work are exposed.

II. RELATED WORK

In this section, a brief general vision of the different malware classifications is offered. On the one hand, these are not mutually excluding, as malware can have characteristics of various classifications at the same time. On the other hand, different authors have different ways of classifying Malware, although each one of them coincide that they are general classifications as each one present common visions as classes always seem to overlap, and many times subclasses are strictly related [24].

Nowadays it results complex to establish a complete taxonomy, so that each category that composes it allows us to classify malware in an excluding way. That is to say, taking a specific malware specimen, it is possible that it has hybrid characteristics that belong to more than one categories in which it can be classified [9].

In this way we have a first classification that must be understood as a generic categorization in which a given malicious code can belong to more than one class in a simultaneous manner. There are 3 main categories to be considered as shown in Fig. 1: Propagation Malware, Hidden Malware and Lucrative Malware [9].

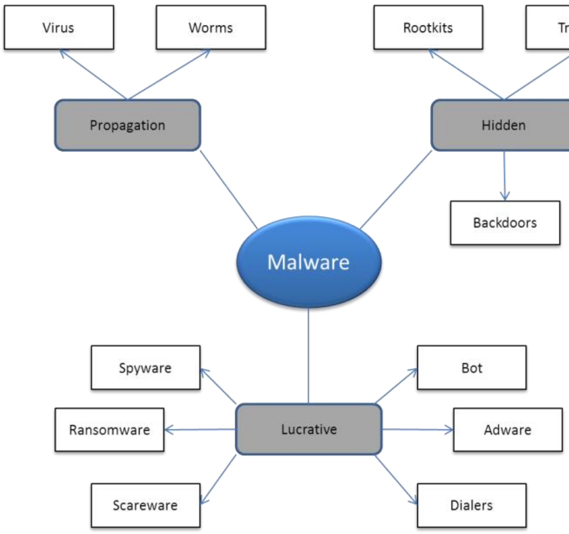


Figure 1 . Generic categorization of malware.

Another classification considers malware as a code or a portion of the code inserted in a document or programme, an whose objective is to harm the units, or take hold of them, classifying them into those which need a host programme (the code is embedded in the host) and the independents which are pure programmes that can be executed at any moment in an operating system [10]. This classification is clearly identified in Fig. 2.

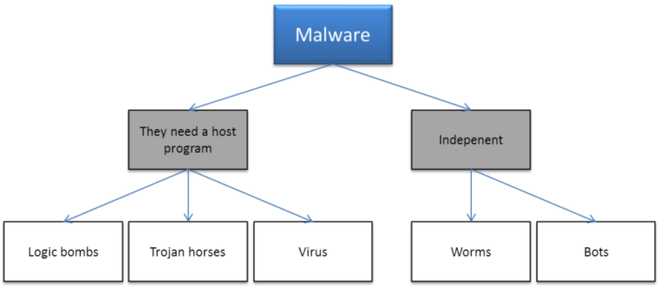


Figure 2 . Malware classification according to the use of a host.

Additionally, we need to consider other investigations such as the "Survey on Malware Detection Methods" by Vinod and Laxmi, in which a classification is exposed, but that even when it considers obfuscation, it only does this from the perspective of metamorphic and polymorphic [29].

Alternatively, in "A Survey on Automated Dynamic Malware Analysis Techniques and Tools" by Egele, a classification that considers the malware from its dynamic or static perspective, not considering obfuscation [8].

In "Introducing Stealth Malware Taxonomy" by Rutkowska, the malware classification is made according to the way in which it interacts with the operating system. It considers type 0 which does not affect the O.S., then the type II which does, and finally type III, which acts, at first instance, as type 0 but that it does harms the O.S without establishing a structured determination of malware [30].

Of the previous it becomes evident that the different work presented show different types of malware classification, but they lack a perspective that considers obfuscation.

This is why this investigation results fundamental as it delivers a classification of malware from the perspective of obfuscation and it incorporates through an empiric demonstration, that a malware which is considered obsolete, can be updated and still be in conditions to be used again.

III. MALWARE CLASSIFICATION BASED ON OBFUSCATION

The classification of malware according to its capacity of obfuscation, is the first step in order to establish a procedure of non-detection, applied to malicious programmes employing obfuscation techniques. For this, an inverted pyramid is developed going from the less relevant information to the most important. The peak is occupied but the species (basic unit) of the malicious code which we denominate malware [2].

A. Class

The different classes (groups of elements of a set, with common characteristics) are located in the second segment of the pyramid. There are four main malware classes: 1) virus, 2) worm, 3) botnet, y 4) trojan horse [11].

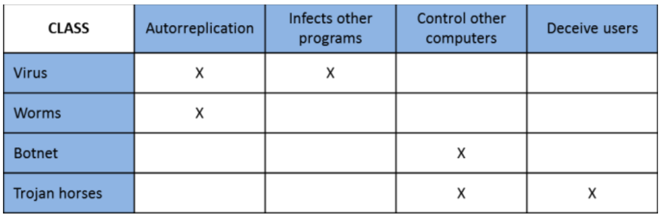
Virus: According to Doctor Fred Cohen, a virus is every programme capable of infecting others, drawn from their modification in order to enter them. They also possess two particular characteristics: the pretend to act in a transparent way and they have the quality of reproducing on their own. Generally speaking, assisted by the Techniques of Social Engineering. The damage that a virus can cause is extremely variable: from a simple message on the screen in order to bother the user, the deleting of system files, and to completely disable the access to the operating system. The previous are some known alternatives [11].

Worms: These are programmes that do copies of themselves, putting them in different locations within the computer. The objective of this malware is usually to colapse the computers and the information networks, preventing the users from working. The main objective of worms is to propagate and affect the greatest amount of computers with or without human help [12].

Botnet: It is a network of computers that are compromised and controlled by an attacker. The botnets have been recently identified as one of the most important threats to the Internet’s security. Traditionally, botnets are organized in a hierarchical way with a central command and control site. This location can be statically defined in the bot or it can be defined in a dynamic way, based on a directory server [13].

Trojan Horses: The term “trojan” comes from the legend of The Trojan Horse, as its main objective is to trick the users so that they execute them simulating being legitimate files, or files that are apparently useful, allowing the attacker to emotely connect to the infected computer. Differently from worms and virus, they do not have the capacity to reproduce on their own [11].

TABLE I. SUMMARY OF CLASSES



CLASS	Autorreplication	Infects other programs	Control other computers	Deceive users
Virus	X	X		
Worms	X			
Botnet			X	
Trojan horses			X	X

B. *Type*

A third segment of this pyramid establishes the types of mlware (which correspong to their own and characteristic traits), among which the following are found: 1) Encrypted, 2) Olygomorphic, 3) Polymorphic y 4) Metamorphic [2].

Encrypted: They basically allow malware to change their appearance, and it is formed by two basic sections: a) an algorithm of decryption (stub), and b) the main body (encrypted malicious code), and also, there is a key between them that changes randomly, being the one in charge of activating the cryptographic algorithm. However, encryption routine is maintained, for which the antivirus can detect it through analyzing the stub [14], as exposed in Fig. 3.

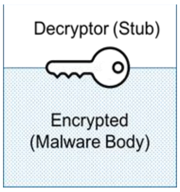


Figure 3 . Encrypted structure of the Malware.

Olygomorphic: It is comprised by two basic sections. a) a different set of decryption stubs, which randomly chooses to mutate in each execution instance and b) the principal body (encrypted malicious code), and also, there is a key between them that changes randomly, being the one in charge of activating the cryptographic algorithm. For antivirus, its detection does not imply any major problem, because it solely needs to analyze a finite quantity of possible stub [14], as shown in Fig. 4.

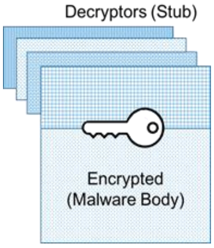


Figure 4 . Olygomorphic structure of Malware.

Polymorphic: In Fig. 5, this type of malware is exposed, which incorporates other obfuscation techniques (instruction replacement, with trash or dead code insertion) additionally from the classic encryption. It is comprised of three parts: a) a decryption loop (stub), b) mutation engine which can generate infinite new variants of stub for each execution instance. This connects to the body of the encrypted malware for the construction of a new malicious code and c) the body of the malware (encrypted malicious code). Additionally there is a key between parts a) and c) that changes randomly, being the one in charge of activating the cryptographic algorithm. For antivirus, its detection implies major problems because millions of decryptors can be generated due to the change of instructions of the following variable of the malware, to avoid the signature based detection [15].

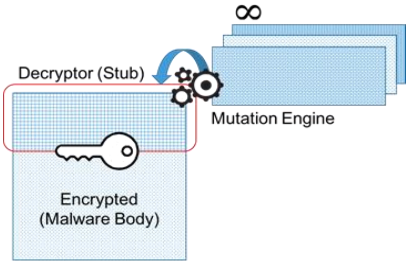


Figure 5 . Polymorphic Malware Structure.

Metamorphic: Contrary to the previous three types exposed, this malware does not count with an encrypted part, so it does not require a stub nor a key in charge to activate the cryptographic algorithm. It is comprised by two parts: a) body of the malware (encrypted malicious code) and b) a mutation engine that modifies al the body of the malware, as it is shown in Fig 6. It is almost impossible to detect using techniques that are signature based, so it is recommended that detection techniques based on behaviour should be rather used [1].

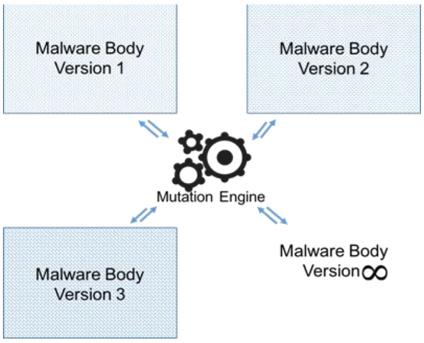


Figure 6 . Malware Metamorphic Structure.

TABLE II. TYPE SUMMARY

KIND	Stub	Malicious code encryption	Key	Mutation engine
Encryption	X	X	X	
Olygomorphic	X	X	X	
Polymorphic	X	X	X	X
Metamorphic		X		X

C. *Generation*

Currently, two generations are exposed, the first generation is where all those malware whose structure has not been modified (obsolescence), and the second generation is the one that frames those that have incorporated obfuscation methods in their codes, such as: Encryptor, Olygomorphic, Polymorphic and Metamorphic [2].

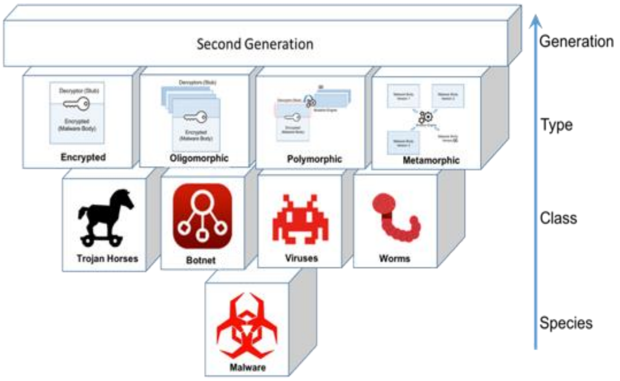


Figure 7 . Malware Classification.

In this way, and just as shown in Fig. 7, a structure that facilitate the classification of malware (species) could be established, in accordance to their class, type and generation. Allowing with this, in a succinct manner, to know part of the essential information about their non-detectable capacities.

IV. MALWARE UPDATE BASED ON OBFUSCATION

According to the classification of malware that the related literature exposes, first generation malware would not be considered part of the inverted pyramid shown in Fig. 6. First generation malware are those that have not had modifications over their code, and that are denominated No Stealth [14] because of their obsolescence or disuse.

Currently, the No Stealth, even when they are detected by antivirus, they are capable of being transformed into second generation malware using tools denominated crypter. These are easy to get in information security specialized websites. Crypter applies obfuscation techniques to any type of file, but in this particular case, it will be done on malware, in a deliberate act, still not altering their functionality in any of their states [16]-[7]-[17].

When we refer to the possible states in which we find malware regarded as a species, we find: a) the source code, consisting of the set of lines of text, written by a programmer in any programming language, but this state is not directly executable by a binary b) computer, when the code has been translated into machine language can be interpreted by the computer and thus can be executed. To perform this translation compilers, assemblers are used, among others [8]. As shown in Fig. 8.

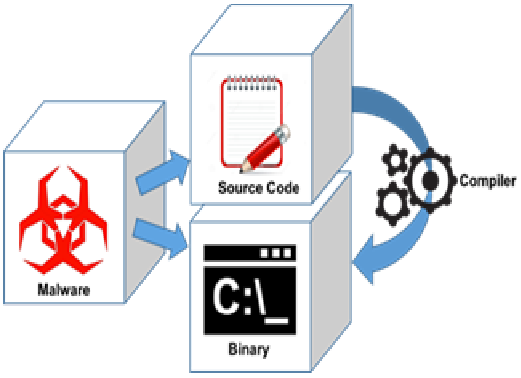


Figure 8 . Malware States.

A. *Update*

The update requires obfuscation in malware second generation can observe it from two perspectives: a) Manual, understood this as the operation to be performed by a person on the malware and applies to Encriptador types Oligomorfico b) automated, which refers to those who are able to make the process automatic procedures obfuscation, without requiring human operators, in this case we find the Polymorphic and Metamorphic [18]. As shown in Table 3.

TABLE III. OBFUSCATION UPDATE

	UPDATE	
MALWARE	Manual	Automated
Encryption	X	
Olygomorphic	X	
Polymorphic		X
Metamorfic		X

B. Methods

The changes that the involvement of the states of malware without altering its functionality, required to be carried out in an orderly and systematized to achieve the expected result, this is what is known as obfuscation method [3].

Thus, in Table 4, it can be stated that there are several methods that arise from different scientific sources consulted, among the most prominent:

TABLE IV. OBFUSCATION METHODS

Method	**Description**
Obfuscation of control transformations	Has the recovery capacity of opaque and variable predicates [3].
Encryption	It is based on the employment of an encryption tool of files (crypter) in such a wat that they become illegible. If the case is that it is a malware, what is looked for is the evasion of antivirus[3].
Obfuscation procedure abstraction	It looks to achieve obfuscation procedure abstraction, breaking those procedures defined by the user or introducing new false abstractions. The original structure of the code is destroyed [19].
Incorporates obfuscation on the data types	This obfuscation method affects the data of the application of origin, which results to be complex because it is part of the programming languages[20].

C. Techniques

Obfuscation techniques follow a procedure or set of rules, which require manual dexterity, intellectual or automated protocols, employing the use of specialized tools, which aims to achieve a certain result, which in this case is due to malicious intent all malware [14].

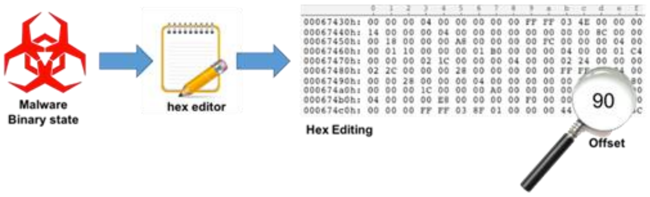
Of various scientific sources consulted, it was established that there are various techniques, which are presented in Table 5:

TABLE V. OBFUSCATION TECHNIQUES

Techniques	**Description**
Dead embed code (Non-ops)	It is the insertion of a binary code to modify the program sequence, without any effect on the functionality of the code and behavior. But if it helps, in evading antivirus scanner based firm, to modify the structure of Stub [7] - [21].
Code transposition	Order the original code sequence, without having any impact on their behavior [22].
Subroutine rearrangement	Obfuscate an original code by changing the order of its subroutines randomly [23]. This technique can generate different variants, depending on the amount of subroutines written in malware.
Replacement instruction	It is implemented using a library of equivalent instructions. Replaces an instruction in the body of code with one that is equivalent. This can greatly change the company code, and it is difficult to obfuscate [4].
Integration code	Sophisticated techniques used to generate new structure in the body of malware by each iteration. [22] - [24].
Encryption / Decryption	It corresponds to one of the first systematic techniques to hide information, consists of two parts: a) body encryption b) decryption code [25].

D. Procedures

If we consider the set of procedures and work instructions that allow you to perform a technique. We will require for implementation of the named: hexadecimal editing, which allows the user to view the specific position (offset) of intact and precise content of a file using a hex editor, which is a type of computer program, through which a user modifies binary [26]. As displayed in FIG. 9.



```
             0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00067430h: 00 00 00 04 00 00 00 00 00 FF FF 03 4E 00 00 00
00067440h: 14 00 00 00 04 00 00 00 00 00 00 00 00 8C 00 00
00067450h: 00 18 00 00 00 A8 00 00 00 00 FC 00 00 00 04 00
00067460h: 00 01 10 00 00 00 01 B0 00 00 00 04 00 00 01 C4
00067470h: 00 00 00 02 1C 00 00 00 04 00 00 02 24 00 00 00
00067480h: 02 2C 00 00 00 28 00 00 00 00 00 FF FF       00
00067490h: 00 00 28 00 00 00 04 00 00 00 00 00          80
000674a0h: 00 00 00 1C 00 00 00 A0 00 00 00 00          00
000674b0h: 04 00 00 00 E8 00 00 00 00 F0 00 00           0
000674c0h: 00 00 00 FF FF 03 8F 01 00 00 00 44           C
```

Figure 9 . Hexadecimal display Malware.

Therefore, to avoid signing the antivirus scanner, you must modify the hexadecimal value of an offset in particular functionality without altering malware, and for this, the implementation of specific procedures is required in the Modding computer field is called, being specialists in making such changes, known as Modders.

Among the procedures, which are highlighted in the various sources of information related to computer security, we can find: AvFucker, DSplit, RIT, Hexing and XOR, which aim to generate a binary than the original, the having been made modification relevant hexadecimal value, searching for antivirus evasion.

In short, FIG. 10 allows us to observe updating obfuscation, by applying a method, technique and procedure applied to the malware in any state.

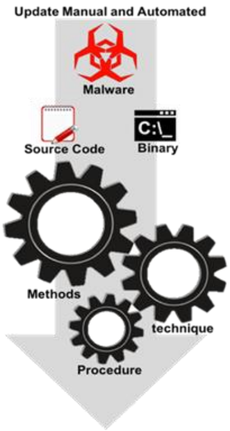


Figure 10 . Update malware obfuscation.

A crypter is composed of two parts: a) builder, it is in charge of encrypting the binary (input) and then attach to the stub, which generates malware encryption over the stub (output) b) stub, is the most important part of tool, since it performs the decrypt the encrypted binary (malware) by the builder using a key that activates the cryptographic algorithm and boot directly into memory, bypassing hard [27] drive, as shown in Fig. 11. Considering this as one of the core functions, because the virus usually perform their scans disk.

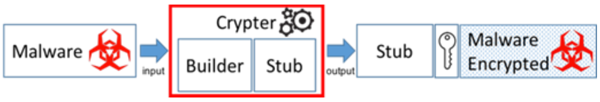


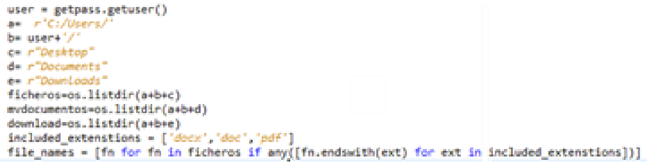
Figure 11 . Components of a Crypter.

It should be made clear that given the characteristics that has this tool to encrypt files and run in memory, the antivirus seek to analyze the output stub crypter, as the party that is not encrypted and therefore can be read, and and evaluate as malicious; that if the case; immediately considered to crypter within the same category. [8]

Therefore, in the event of detection stub is necessary to resort to the application of a method of obfuscation, allowing reapply crypter Malware, because for it to achieve its renovation and therefore its ability to evasion, reports directly to the stub that is engaged and therefore part of this updated malware. As it sets forth in the following tests:

- Construction of Malware (No Stealth)

He proceeds to build malicious code, whose objective is to obtain documents from a third party, of certain directories and various extensions from your computer. Part of the code is exposed (for reasons of confidentiality) in Fig. 12.

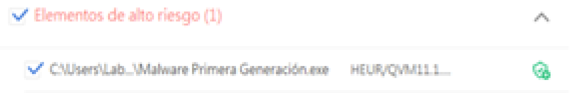


```
user = getpass.getuser()
a=  r'C:/Users/'
b= user+'/'
c= r"Desktop"
d= r"Documents"
e= r"Downloads"
ficheros=os.listdir(a+b+c)
mvdocumentos=os.listdir(a+b+d)
download=os.listdir(a+b+e)
included_extenstions = ['docx','doc','pdf']
file_names = [fn for fn in ficheros if any([fn.endswith(ext) for ext in included_extenstions])]
```

Figure 12 . Of the Code of Stealth Malware no.

- Antivirus tests

Analysis of malicious code with a specific antivirus (NOD 32) vasado signature, which has the ability to incorporate other antivirus engines is performed. As shown in Fig. 13, this code is detected and reported as an element of risk.



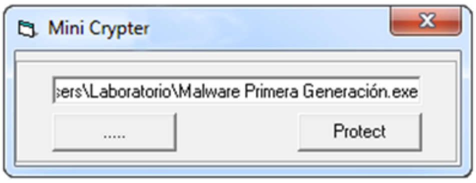
✓ Elementos de alto riesgo (1)

✓ C:\Users\Lab...\Malware Primera Generación.exe HEUR/QVM11.1...

Figure 13 . Antivirus test.

- Application of obfuscation

Upon detection of malicious code, it undergoes a crypter (Mini Crypter) to apply obfuscation techniques (encryption / decryption), the process once completed, becomes detectable one possibly undetectable malware.

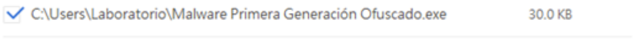


sers\Laboratorio\Malware Primera Generación.exe

Figure 14 . Application Crypter.

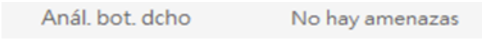
- Screening

Once malicious code obfuscation is achieved, the malware is subjected to antivirus (NOD 32) again under the same conditions as above (same house and built motors). Being possible to observe in Fig. 15, which was not detected, and reports a message alert. Considering further that reports of antivirus logs, catalog it as a file without threat, as shown in Fig. 16. In this way, the malware may fulfill its malicious intent.



✓ C:\Users\Laboratorio\Malware Primera Generación Ofuscado.exe 30.0 KB

Figure 15 . Antivirus test.



Anál. bot. dcho No hay amenazas

Figure 16 . Logs of antivirus.

In relation to the tests, and as is appreciated in Fig. 17, it is evident that not stheal malware considered first generation to apply obfuscation techniques, in this case with a crypter tool are possible to transform and therefore change of category, that is, one second generation, according to hierarchical criteria raised.

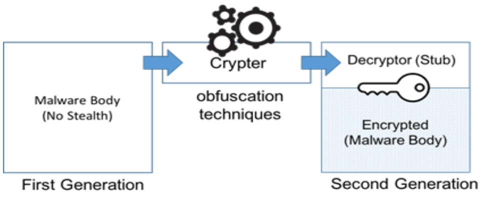


Figure 17 . Transforming a Malware.

V. OBFUSCATION AND CLASSIFICATION OF MALWARE FLOW CHART PROPOSAL

In accordance with the malware classification that was previously exposed, the first generation has no place on it as it lacks obfuscation techniques, but, it is possible to evince that the No Stealth are a reality in cyber space that needs to be considered, (despite of their obsolescence/disuse) as they are need little work on them to get them to become part of the malware that are not detectable for antivirus. This is why, certain knowledge about them should exist, as they are a latent risk. Taking into account the points previously exposed, a flow chart is proposed for the update of malware, as shown in Fig. 18

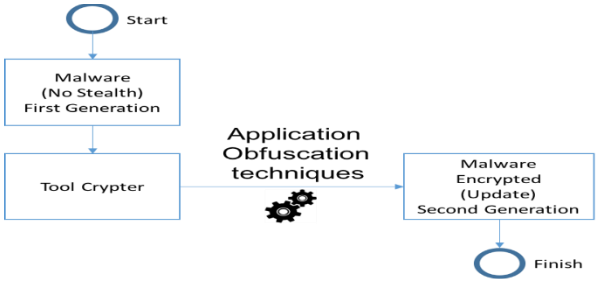

Start
Malware (No Stealth) First Generation
Tool Crypter
Application Obfuscation techniques
Malware Encrypted (Update) Second Generation
Finish

Figure 18 . Flow chart proposal for the application of an obfuscation procedure.

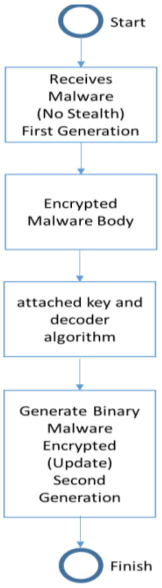

Start
Receives Malware (No Stealth) First Generation
Encrypted Malware Body
attached key and decoder algorithm
Generate Binary Malware Encrypted (Update) Second Generation
Finish

Figure 19 . Flow chart proposal for the update of malware.

In this way, it is estimated that the No Stealth are considered part of the types of malware, as they can be related to any of those that belong to that class (virus, worms, botnet, and trojan horse) and because they belong to those that are defined as the first generation. As it can be seen in Fig. 20, a new inverted pyramid is proposed so that there is a complete structure that classifies malware.

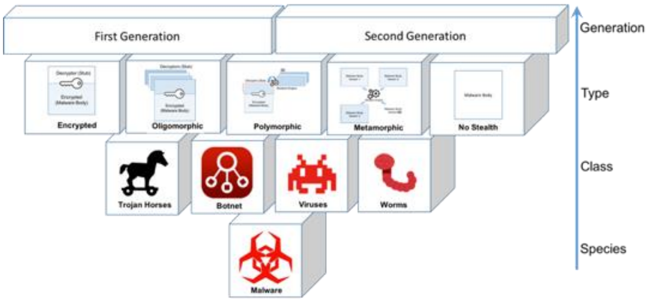


Figure 20 . Proposed classification of malware.

I. CONCLUSIONS

Literature offers malware classification from multiple perspectives, but there is no obfuscation classification that allows us to fit a malware with the given capacities, which is why it results imperative to structure it hierarchically so that the capacities pertinent to each one of them are known. Considering that the scenario in which they develop, denominated cyber space, is where practically all the information that is employed for different purposes at user or corporate level, circulates and is stored.

At a first glance it is possible to observe because of the rapid evolution of malware, they were framed in a new generation denominated as the second one, leaving out all those in which there have been none modification over its code, and are thus considered obsolete in cyber space. Becoming possible to demonstrate that those called No Stealth are a real threat because they can be obfuscated and thus, become second generation malware. The manual update that malware requiere in order to achieve the evasion necessary and be able to face antivirus, is one of the most complex problems to overcome and to do it, it will be necessary to resort to the application of methods, techniques, procedures and tools in order to achieve the obfuscation.

This is how in the present investigation a flow chart has been proposed, that allows us to demonstrate that when the encryption method is employed, and also by applying a tool named crypter, it is possible to incorporate procedures that are specific to obfuscation if the stub's signature is detected. It is possible to obtain as a result, an updated malware that has the same functionality as the base malware or the entry malware, and that needs to be considered in the classification of malware due to the hierarchical structure proposed through the inverted pyramid.

The previous signifies a real challenge for malware developers, who day to day need to elaborate complex updates, but this means an even bigger challenge to the antivirus community. As a future work, the development of a system which automates the manual update that requieres a certain type of malware.

REFERENCES

[1] Tarifa, E. Teoría de Modelos y Simulación, Universidad Nacional de Jujuy, Buenos Aires. 2001

[2] Carr, J. Inside Ciber Warfare, O'Reilly Media Inc, 1ra ed. 2010

[3] You, I. and Yim, K. Malware Obfuscation Techniques: A Brief Survey, IEEE Computer Society Washington, DC, USA. 2010

[4] Konstantinou, E. Metamorphic Virus: Analysis and Detection, RHUL-MA-2008-02, Technical Report of University of London. 2008

[5] Von Neumann, J. Theory of sel- reproducing automata, University of Illinois Press. London.1966

[6] Trend Micro Incorporated. Annual Security Roundup. TrendLabs. 2014

[7] Balakrishnan, A. and Schulze, C. Code ofuscation literature survey, University of Wisconsi, USA. 2005

[8] Egele, M.; Scholte, T.; Kirda, E. and Kruegel, C. A Survey on Automated Dynamic Malware Analysis Techniques and Tools, ACM Computing Surveys, Vienna University of Technology, Austria. 2014

[9] Castillo, S.; Vulnerabilidades y Software Malicioso, II Congreso sobre las Nuevas Tecnologías y sus repercusiones en el seguro: Internet, Biotecnología y Nanotecnología,pp. 97-113, Madrid. 2012

[10] Erquiaga, M., Botnets: Mecanismos de Control y de propagación, CACIC XVII Congreso argentino de ciencias de la computación. 2011

[11] Wang, P.; Aslam, B and Zou, C. Peer-to-Peer Botnets, University of Central Florida, Orlando, Florida, Springer, 2010

[12] Bashari, B.; Masrom, M. and Ibrahim, S. Evolution of Computer Virus Concealment and Anti-Virus Techniques: A Short Survey, International Journal of Computer Issues, Vol. 8, Issue 1, January 2011

[13] Vasileios, T. Bypassing Antivirus Detection with Encryption, University of Piraeus, Pireo, Grecia, 2014

[14] Fredes, C.; Hernández, J. and Díaz, D. Potencial y Problemas de la Simulación en Ambientes Virtuales para el Aprendizaje, Formación Universitaria , pp 45-56. 2012

[15] Livingston, W. COTS: Commercial Off-The-Shelf or Custom Off-The-Shelf?, CrossTalk , pp 31. 2007

[16] Chiang, L. Simuladores semi-inmersivos para la educación técnico-profesional. Hacia un modelo educacional sustentado con herramientas TIC, Comisión Nacional de Investigación Científica y Tecnológica - CONICYT, Santiago. 2008

[17] Zico Kolter, J. and Maloof, M. Learning to Detect and Classify Malicious Executables in the Wild, Journal of Machine Learning Research 7 2721-2744, USA. 2006

[18] Zhang, Q. MetaAware: Identifying Metamorphic Malware, Publisher: IEEE North Carolina State Univ., Raleigh ; Reeves, D.S., 10-14. 2007

[19] International Standard ISO 9241. Ergonomic requirements for office work with visual display terminals, Ginebra, ISO/IEC. 1998

[20] Solano, A.; Rusu, C.; Collazos, C. and Arciniegas, J. Evaluating interactive digital television applications through usability heuristics, Revista Chilena de Ingeniería, vol 21, n° 1 , pp. 16-29. 2013

[21] Christodorescu M. and Jha, S. Static analysis of executables to detect malicious pat- terns. In 12th USENIX Security Symposium, pages 169–186. 2003

[22] Kumar Agarwal, S. and Shrivastava, V. BASIC: Brief Analytical Survey on Metamorphic Code. International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 9. 2013

[23] Wong, C. and Stamp, M. Hunting for metamorphic engines, Springer, Volume 2, Issue 3, pp 211-229. 2006

[24] Szor, P. The Art of Computer Virus Research and Defense, Addison Wesley Professional. 2005

[25] Bashari, B.; Masrow, M. and Ibrahim, S. Camouflage in malware: from ecryption to metamorphism. International Journal of Computer Science and Network Security, Vol. 12 Nº8, Malasya. 2012

[26] Murad, K.; Noor-ul-Hassan, S.; Bin Zikria, Y. and Ikram, N. Evading Virus Detection Using Code Obfuscation. National University of Science and Technology (NUST), Islamabad, Lecture Notes in Computer Science Volume 6485, pp 394-401, Springer. 2010

[27] Ammann, C. Implementation of a PE-Crypter, Nullsecurity. 2012

[28] International Standard ISO 27001, Information security management, Ginebra, ISO/IEC, 2014.

[29] Vinod P., Laxmi V. and Gaur M. Survey on Malware Detection Methods The 3rd Hackers Workshop, Hack.in, 2009

[30] Rutkowska J. Introducing Stealth Malware Taxonomy. COSEINC Advanced Malware Labs, 2006