Ransomware Anti-Analysis and Evasion Techniques: A Survey and Research Directions

Mohammad N. Olaimat
School of Computing,
Faculty of Engineering
Universiti Teknologi Malaysia
Johor Bahru, Johor 81310, Malaysia
mnOlaimat@gmail.com

Mohd Aizaini Maarof
School of Computing,
Faculty of Engineering
Universiti Teknologi Malaysia
Johor Bahru, Johor 81310, Malaysia
aizaini@utm.my

Bander Ali S. Al-rimy
School of Computing,
Faculty of Engineering
Universiti Teknologi Malaysia
Johor Bahru, Johor 81310, Malaysia
bnder321@gmail.com

2021 3rd International Cyber Resilience Conference (CRC) | 978-1-6654-1844-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/CRC50527.2021.9392529

***Abstract*— Ransomware has been proven to constitute a severe threat to the world's digital assets. Resources or devices' recovery from a Crypto-Ransomware infection is practically infeasible unless an error in the malicious cryptographic implementation has been made, as robust encryption is irreversible. This paper attempts to justify as to why designing and deploying an effective and efficient detective solution against this particular malware category represents a formidable technical challenge. The paper starts with a recent presentation of the Ransomware's epidemic, as reported by the security industry. Subsequently, a taxonomy of Ransomware is presented. The anatomy of the malware's invariant intrusions and infection vectors are illustrated. In addition, the paper navigates and analyzes the various anti-analysis and evasive techniques that are deployable by Ransomware. In every context enumerated in the narrative, the technical difficulty being posed by this malware is illuminated. If a computer security researcher intends to devise a Crypto-Ransomware's preventive solution or a predictive or proactive one, then it is imperative to have a sound perception of the technical challenges that will manifest prior to launching the proposed research project-- so as to be equipped to tackle the anticipated problems. This paper concludes with an advance notice underscoring the resilience of Ransomware intrusions and highlighting research open-problems.**

Keywords—Crypto-Ransomware,Locker-Ransomware, Ransomware, Crypto-Miners.

I. INTRODUCTION

Conspicuous breaches by Ransomware have been manifested by the notorious Ransomware families or strains. During March of 2019 the LockerGoga Crypto-Ransomware crippled operations of enterprises [20] followed by another new Ransomware attack by Sodin (a.k.a Sodinokibi and REvil) in July, 2019 [1]. The year 2019 was plagued with the emergence of more notorious Ransomware families such as JSWorm (01/2019), CLOP (02/2019), Maze (05/2019), DoppelPaymer (07/2019), Nemty (08/2019), NetWalker (08/2019), LockBit (11/2019), and RagnarLocker (12/2019); the operators of Maze, LockBit, and RagnarLocker formed a Ransomware cartel [2]. During the year 2020, new Ransomware families emerged such as: Nephilim or Nefilim (03/2020), VHD (05/2020), Tycoon (06/2020), Sekhmet (06/2020), WastedLocker (08/2020), Egregor (09/2020), and RansomEXX (10/2020) [2, 3].

Ransomware agents have exacerbated the problem of combatting crypto-malware by having steadily been generating new polymorphic variants, adapting their engines to serve as crypto-miners in Cryptojacking attacks, and coupling data exfiltration along with file-system cryptography [3, 10, 14, 20, 36]. Resorting to a remote digital style of performing work and attending e-classrooms, during the COVID-19 pandemic, has only exposed more vectors of infection [1, 4, 5].

If a computer security researcher intends to devise a Crypto-Ransomware's preventive solution or rather a predictive or proactive detection one, then it is imperative to have a sound perception of the technical and procedural challenges that would potentially manifest prior to launching the research project. Hence, this paper surveys some of the impactful technologies and techniques that facilitate infections by Crypto-Ransomware or making them more robust.

The remaining part of the paper is structured as follows. In section II, a taxonomy of Ransomware is presented. Section III analyzes the variant anatomies of Crypto-Ransomware intrusions. Section IV illuminates the exploitable anti-analysis and evasive techniques deployable by Ransomware. Section V discusses any other problem(s) that may remain intractable. Section VI concludes the paper.

II. TAXONOMY OF RANSOMWARE

One of the first challenges that Ransomware represents is that it is a versatile category of malware [6]. Figure 1 depicts a taxonomy of Ransomware based on the malware's objective of the intrusion: Locker-Ransomware, Crypto-Ransomware, and Crypto-Miners. Locker-Ransomware's

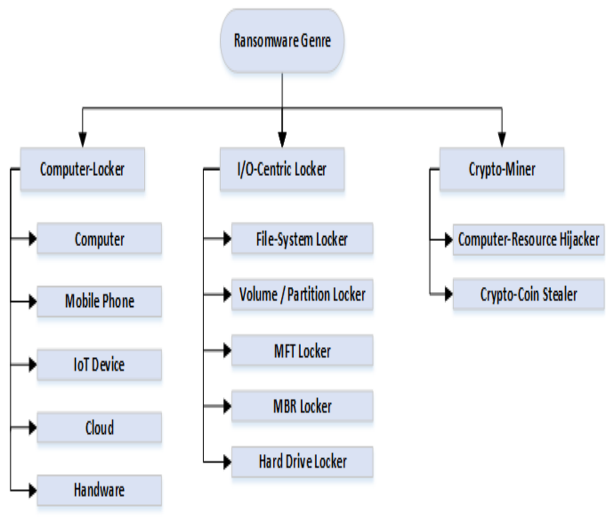

Fig. 1: Ransomware categories & infection vectors

objective is locking a computing device (PC, mobile, cloud, handware, and IoT) to deny access to its operating-system and resources [6-8] whereas Crypto-Ransomware (CRW) leverages legitimate cryptographic infrastructures (AES, RSA, OS Crypto-API) to encrypt users' files, volumes or partitions, hard-drives, or network-mapped virtual drives on the targeted hosts [6, 9]. As for the category Crypto-Miners, it infects computers for the purpose of either hijacking their computing resources to mine crypto-coins, or alternatively, stealing the crypto-coins themselves by the cybercriminals [3, 4, 10].

Due to variances of the infection logic of those Ransomware categories, a detection solution that should account for the entire scope of their intrusion would be complex and it would intuitively suffer from high resource-complexity; consequently, leading to detection inefficacy or inefficiency or perhaps both.

III. Anatomy of the Crypto-Ransomware Intrusions

Since Ransomware is a subclass of malware, it is intuitive to conceive that it would leverage inherited intrusion dimensions that its ancestor utilizes [11-13]. Therefore, Ransomware would implement fundamental malware techniques of intrusion with varying degrees of adaption contingent upon the Ransomware family or strain [4, 6, 7, 12]. However, a generic Crypto-Ransomware (CRW) attack could be outlined by the following phases [14, 15]:

- Delivery of Payload: During this phase, the payload is delivered to the targeted host via any method of exploitation, such as executing a Trojan or leveraging an exploit-kit;
- Reconnaissance and Anti-Analysis: During this phase, the malware probes the targeted environment, fingerprints the host's ecosystem, and implements stealth countermeasures to sustain and configure the attack clandestinely to avert detection;
- Target Harvesting: Probing the environment to map any of the following targets: User's files, Master File Table (MFT), MBR, file-system volumes or partitions, and the entire hard-drives by implementing various search techniques. At this stage, the victimized host may contact the C&C servers to retrieve further instructions or an encryption key, if necessary;
- Cryptographic Infection: Encrypting the targeted resource;
- Extortion Demand: Displaying a message on the screen warning that the host is a victim of a crypto extortion and providing info about paying of the ransom.

Although this behavior is presumed to be typical, the problem of behavioral signature variance of CRW is a fundamental issue that detection solutions are most likely to encounter on the field [5, 12]. For instance, a WannaCry strain upon being executed, it places itself in sleep-mode for 18 minutes and 47 seconds, TeslaCrypt stalls for 4 minutes and 20 seconds, whereas Cerber does not stall at all [8, 15, 16]. Furthermore, the intrusion phases are not consistent among the CRW families or strains. For example, WannaCry starts its infection cycle with mapping the target's file-system and then displaying the ransom message on the desktop followed by the encryption phase, TeslaCrypt starts its infection cycle with fingerprinting the target environment to weaponize the attack followed by communicating with the C&C server to retrieve instructions [16, 17]. **Table 1** illustrates the variations of the sequence of the encryption phase of the attack cycle among selected CRW families.

TABLE 1. Order of the encryption phase within CRW attack cycle

Ransomware Family	Encryption Operation Sequence
WannaCry	3rd phase
TeslaCrypt	5th phase
Cerber	5th phase
Locky	2nd phase
CryptoWall	4th phase

Moreover, variations exist as to what scheme CRW may leverage to encrypt a user's files: CRW may either encrypt the original files directly, or copy files onto another destination and subsequently encrypt them and overwrite the original with the encrypted versions or delete them [16, 18].

All those variations of the infection algorithms of CRW complicate the proposed solutions or degrade their effectiveness or robustness, as the behavioral signatures of CRW are steadily variant. This matter is aggravated even further as malware, in general, is evolutionary and polymorphic [3, 14, 16].

IV. RANSOMWARE Anti-Analysis and Evasive Techniques

Since Ransomware has inherited the traits and behavior of its ancestor, the malware, it has been validated empirically that Crypto-Ransomware implements the anti-analysis and stealth techniques employed by other malware categories during intrusion [11, 12, 14-17, 19, 20].

In order for malware to mount assaults against the target stealthily, it may elect to employ evasive schemes to evade detection and identification. For instance, polymorphic Ransomware may create numerous decryptor engines and employ various methods for obfuscating its structure to avert analysis [12, 21]. The metamorphic strains of malware employ permutation engines to mutate its code body for innumerable clones [12, 22]. It is also important to observe that malware employs various obfuscation mechanisms to evade being analyzed and reverse-engineered by researchers or anti-malware tools [12]. In addition, malware may utilize steganography and covert communication channels to masquerade its logistics and modus operandi [23]. If the malware detects an anti-analysis environment, it would alter its processing logic to avert detection by employing one or more of the following tactics [12, 23-25]:

- It suspends its processing and places itself into hibernation mode;
- The injection of malcode into other processes would be delayed;
- Operations of downloading malcode from external repositories would be aborted;
- Obfuscating the dynamic behavior of its malcode;

- Operations of migrating and hopping through the network would be suspended;
- Sustaining live connections with the malware's command & control center would be avoided.

The following subsections explore Ransomware's anti-analysis and anti-detection techniques and delineate their adverse impact on the efficacy of proposed defences; Figure 2 illustrates those techniques:

A. *Obfuscation and Packing*

To obstruct analysis and detection, malcode obfuscation is deployed via multiple techniques of code transformation, such as instruction reordering, data-block reordering, inlining and outlining of routines/statements, register renaming, instruction permutation, code expansion or compaction, subroutine interleaving, and dead code insertion [20, 24]. The obfuscated malcode would be re-compiled to produce a new executable that is characteristically different from the original image [26, 27]. For instance, in order to implement obfuscation, the Ransomware Locky injects multiple files into its application code body to create a "code spaghetti" and loads it at run-time to sabotage the malware-analysis and detection mechanisms [2, 28]. A clever packing technique utilized by CryptoWall v3 family involves embedding a malicious PE inside a legitimate file [29].

For instance, the CryptoWall v3, CrypVault, Cerber, and Petya families of Crypto-Ransomware implement various obfuscation techniques [20, 29, 31]. As for the packing techniques, most of the families of Crypto-Ransomware, such as GandCrab, CryptXXX, Cerber, Locky, Teerac, Crysis, CryptoWall, CTB-Locker, Cryptesla, and CriLock, are known to have leveraged packing [3, 17, 20, 32]. The cryptographic obfuscation techniques of the malcode are widely leveraged by the families of CryptoBit, Cerber 5.0.1, and Locky to subvert analysis and detection mechanisms [28, 33].

B. *Code Polymorphism & Metamorphism*

One of the viable approaches that Malware employs to obfuscate its behavior or structure is cryptography. A cryptographic module is comprised of a cryptographic engine component (encryptor-decryptor) and an encrypted payload. Malware may mutate the structure of the malcode or the crypto-engine or both [12, 30].

Polymorphic Malware alters the cryptographic engine for each new generation of a malcode instance. Therefore, this mechanism may generate a large number of distinct crypto-engines by applying permutation against the encryption engine algorithm or primitives. As for the Metamorphic malware, it applies transformation techniques against its crypto-engine (mutating engine) and the malcode structure as well [12, 34, 35].

New Ransomware morphs or mutants have steadily been developed [4, 36] reaching 21,239 unique mutants as intercepted by Kaspersky Labs alone for the first half of 2019 [1]. Kaspersky Labs detected 11,971 unique Crypto-Miner attacks during the first quarter of 2019 [1].

Hence, evolutionary Crypto-Ransomware, polymorphic or metamorphic, constitutes a continuous obstacle towards realizing effective detection solutions, as the structural and behavioral signatures are variant. Almost all Crypto-Ransomware families are evolutionary; for instance, Reveton, Winlock, and Urausy families are notorious examples of vigorous utilization of polymorphism techniques [5, 7, 12].

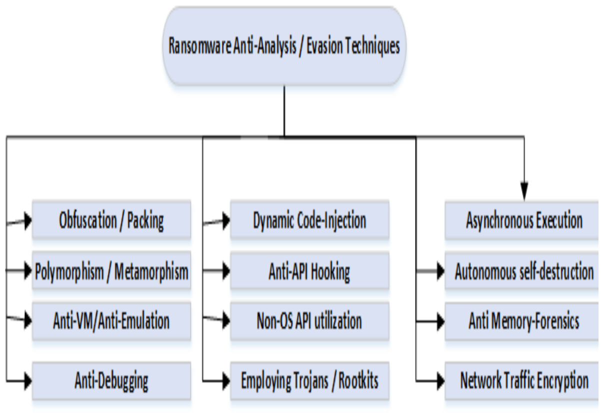

Fig. 2: Crypto-Ransomware evasion techniques

C. *Anti-Virtualization & Anti-Emulation*

Virtualization and emulation appliances are leveraged to virtualize or emulate an execution environment for Crypto-Ransomware agents without causing an infection to the host machine [37, 38, 42]. A list of the mechanisms that malware might employ to defeat virtualization and emulation appliances are enumerated as follows [21, 24, 37, 39-41]; however, the list is not exhaustive:

1) Locating Windows registry-keys of a virtual system. For instance, there are over 300 references in the Windows registry pointing to VMware; for example: "HKLM\SOFTWARE\Vmware Inc.\Vmware Tools".

2) Exploring the presence of a VM system installation. For instance, the presence of "C:\windows\System32\Drivers\VMToolsHook.dll" points to VMware; the presence of "C:\windows\System32\Drivers\VBoxSF.sys" points to Oracle VirtualBox.

3) Exploring the presence of a VM system's processes and services. For example, "VMwareTray" & "Vmscsi" of the VMware system and "vboxtray" of Oracle VirtualBox.

4) Identifying the BIOS serial number or MAC address of a VM system. Virtual network adapters' addresses reveal the identity of a VM vendor. For example, if the prefix of MAC addresses is equal to "00-0C-29", they are associated with VMware. If the prefix is equal to 08:00:27, they are associated with Oracle VirtualBox.

5) Locating the memory addresses of SIDT, SLDT, and STR. The Store Interrupt Descriptor Table (SIDT), Store Local Descriptor Table (SLDT) and Store Task Register (STR) tables are located in different regions of memory for physical and virtual machines.

6) Validating the physical/virtual platforms' unique parameters. Malware may query various parameters, such as serial numbers, of the chipset, system-motherboard, processor, SCSI controller to differentiate between physical and virtual machines.

7) Probing for special flags stored in microprocessor's registers. For instance, the CPUID register flag's 31st bit on a physical machine reads 0, whereas it reads 1 on a VM. The VM vendor strings, "Microsoft HV" and "VMware", could be validated, as they are stored in the CPU registers EAX, ECX, and EDX. VMware dedicates a specific I/O port to enable communication with the host operating system.

8) Validating the absence of particular microprocessor's special instruction sets. For instance, the Intel MMX instruction set is not supported by virtual machine environments.

9) Employing system timers to calculate the cost of processing a particular API or a routine or an instruction in the host environment by the invasive Ransomware. The computed cost differential would constitute an indicator of the existence of an emulation or virtualization session to the malware. For instance, the Windows APIs, GetProcessHeap and CloseHandle, are utilized by the Locky Crypto-Ransomware to determine if it is being executed in a virtual machine [28]. The virtual machines typically consumes a high number of CPU cycles, versus the physical machines, to perform MS Windows API routines, such as those two APIs [24]. Based on this observation, the malware would ascertain that it is being executed in a virtual machine; consequently, it would deploy particular stealth techniques to evade detection or it may stall executing its malcode for a period of time.

10) In case of an emulation appliance, malware probes the appliance to ascertain the absence of a certain OS API or system-call, as only a limited subset of an OS APIs and system-calls are emulated of a particular platform.

One of the most popular Ransomware analysis frameworks among researchers is the Cuckoo Sandbox [5, 12, 42, 43]. However, this framework could be subverted by Ransomware with a relative ease [24, 44, 45]. This adverse situation would be added to the compounded challenges encountering the proposed anti-Crypto-Ransomware solutions.

It is noteworthy to indicate that almost all the families of Crypto-Ransomware exploit anti-sandbox techniques to subvert detection; some examples of which are Cerber SFX, Cerber v6, UIWIX, WannaCry, and CryptXXX [14-17, 28, 33].

D. Anti-Debugging

Ransomware analysis utilities may involve leveraging a debugger to execute the malcode for discerning its anomalous behavior. Since a debugger invokes multiple Windows APIs and sets a debug flag in a microprocessor register, the malware would subsequently detect the debugger and it may stall or alter its vicious maneuvers [41, 46, 47].

Another tactic that Ransomware may utilize for detecting a debugger is checking for the current visual dialogs' handles. If the window's handle belongs to a debugger, such as OllyDbg or Immunity Debugger, the malware would instantly change course of action or terminate itself [20, 41, 46]. In addition, a clever mechanism for detecting a debugging environment by malware is invoking the Windows API GetTickCount to retrieve the system clock readings at the beginning and ending of tasks. If the processing time slice retrieved is longer than normal, the malware concludes that it is being debugged [15, 41, 46, 47].

It has been observed that most of the Crypto-Ransomware families employ anti-debugging to defeat analysis, such as Jigsaw, TeslaCrypt, CTB-Locker, Locky, CryptoWall, and TorrentLocker [16, 28, 29].

E. Dynamic Code-Injection

One of the dynamic stealth technologies employed by malware is code-injection [3, 39]. There exists a plethora of code-injection implementation techniques of this technology [39, 48]; however, their functional denominator is constant: enabling the malcode to execute within the memory map of an OS process, or any benign one, and inheriting its security profile and features [48]. Hence, a malware agent may invade an external process by injecting foreign threads into its memory space. Of the primary targets of the code-injection attacks under MS Windows are the system processes "explorer" and "svcHost" whereby malware masquerades as being those system processes to shield itself against analysis and detection [7, 48]. It is imperative to observe that almost all Crypto-Ransomware families resort to code-injection techniques with varying styles of implementation [8, 15, 48, 49].

F. Anti-API-Hooking

Current Crypto-Ransomware detection techniques place amplified emphasis upon the semantic behavior of this malware; that is, relying on operating-system's API-call tracking via API-hooking mechanisms implemented in the User-Mode [6, 23, 42, 50]. Hence, so as to evade detection, Ransomware may implement anti-API-hooking tactics to subvert the tracking of its behavioral events at run-time [44, 51, 52].

G. Utilization of Non-OS API

As current Crypto-Ransomware detection techniques rely heavily on operating-systems' API-hooking [8, 14, 23, 42, 50], Crypto-Ransomware may sabotage this detective approach by employing non-OS API altogether; that is, it would leverage third-party cryptographic API, such as OpenSSL and Crypto++ [53].

H. Employing of Trojans & Rootkit Stealth

Since a rootkit has been proven to provide stealth services and concealment for other categories of malware upon infection [54, 55], the potential exists for Crypto-Ransomware to seek refuge behind rootkits at the Kernel-Mode as an evasive measure against Ransomware detective solutions. Although we are not aware of a case whereby Crypto-Ransomware has utilized rootkits to provide stealth, we anticipate that this malware would adopt this approach. However, Ransomware has effectively employed Trojans to implement its intrusions [4].

I. Asynchronous Execution

Crypto-Ransomware may unleash its destructive payload contingent on a timer or an event trigger. Therefore, the malware may delay or stall its destructive processing and it would subsequently stand dormant for an unspecified period of time. This temporal and event-centric behavior constitutes a challenge for the detective solutions. It should be noted that WannaCry, KeRanger, Reveton, CryptXXX, and Cerber leverage this stealth technique [15, 17, 29, 33].

J. Autonomous Destruction of Malicious Evidence

Some of Crypto-Ransomware families may implement various strategies of destroying their residual artifacts and the forensic evidence of their intrusion so as to obstruct detection. The Ransomware families Cerber, CryptoWall 3.0, Locky, and TeslaCrypt are observed to have implemented this anti-analysis technique [16, 17, 29, 33].

K. Implementation of Fileless Infections

Some Crypto-Ransomware families or strains have implemented "fileless" intrusive techniques by loading their malcode into the system memory directly without leaving any residual traces on the file-system of the victimized environment [1, 36, 56].

L. Anti-Memory Dumping

Some of the families of Ransomware may elect to obstruct the efficacy of the forensic technique of dumping their processes' memory-images. To sabotage the analysis, the malware may modify the PE assembly's headers and/or randomize the process's address-space; TeslaCrypt and CTB Locker are reported to have implemented this stealth scheme [8, 16].

M. Network Traffic Encryption

Crypto-Ransomware may encrypt the networking packets so as to evade analysis and detection by the anti-malware appliances; for instance, CryptoWall and Locky families encrypt their networking traffic [28, 29].

All the defensive techniques that were enumerated earlier are so exploitable by Ransomware [3, 7, 8, 48]. The sole issue with which Ransomware diverges from non-Ransomware malware behavior is sabotaging the OS kernel, as Ransomware requires the OS to be healthy so as to proceed and enable the victim to retrieve Ransomware instructions and complete the transaction of the ransom-payment [6, 11, 23, 57].

V. Open Research Problems

Based on this paper's enumeration of the various problems encountered by the current analytic and detective solutions of Ransomware, it is evident that considerable research opportunities exist. The Anti-emulation and anti-virtualization nefarious techniques deployed by all malware categories are still problems that require effective resolutions. In addition, other extant problems demand extended rigorous research efforts, some of which follow. The inability of the current solutions to differentiate between a cryptographic process induced by the current logged-in user and an invasive one during the process-creation phase. Identifying and locating the cryptographic routines and primitives in the address-space of a nefarious process preemptively and effectively represents a challenging issue. The problem of Fileless Ransomware infections represents a formidable challenge as well and creates new opportunities for further investigations. Due to the vastness of the infection plane of Rasomware and the evolutionary nature of its families and strains, intelligent analytic/detective engines are in demand to provide a comprehensive immunity corresponding to the targeted multi-dimensional scopes. Since Ransomware adopts heterogeneous techniques for implementing its attacks and utilizing variant time-lines to deploy them, there is a research incentive to design solutions to account for the existing variations and adapt to the novel ones with acceptable efficacy and resource-complexity. In addition, one of the outstanding problems is that there exists significant resemblance between the behavioral signature of Crypto-Ransomware and that of some benign applications, such as the packing and cryptographic utilities. The challenging task is, therefore, as to how to effectively discriminate between the benign processes and those of Crypto-Ransomware if both types utilize the same cryptographic infrastructures of the operating system and work against the same type of users' computing resources. A problem that has the potential to render all the OS API-centric solutions ineffective is evading the utilization of the OS crypto-API by Ransomware.

VI. Conclusion

This paper presented a plethora of technical challenges that hinder producing robustly effective detection solutions of Crypto-Ransomware. Some of those challenges are attributed to the platforms' architectures and the operational constraints of the operating systems. Other problems emanate from the nature of this particular Malware or inherited from its ancestors. Considerable number of technical problems remain intractable. The technical issues explored in this paper should motivate the research community to devise more effective and efficient anti-Ransomware algorithms and procedures, most especially that the world is now geared towards adopting more than ever digital styles of performing work and attending schools.

References

[1] Kaspersky Labs. Kaspersky Security Bulletin 2019. Statistics. 2020; Available from: https://securelist.com/kaspersky-security-bulletin-2019-statistics/95475/.

[2] Kaspersky Labs. Malware Reports: IT Threat Evolution Q3/2020. 2020; Available from: https://securelist.com/it-threat-evolution-q3-2020-non-mobile-statistics/99404/.

[3] BlackFog.com. The State of Ransomware in 2020. 2020; Available from: https://www.blackfog.com/the-state-of-ransomware-in-2020/.

[4] SOPHOS.com. Sophos 2021 Threat Report. 2020; Available from: https://www.sophos.com/en-us/medialibrary/pdfs/technical-papers/sophos-2021-threat-report.pdf.

[5] Zimba, A., et al., Recent advances in cryptovirology: State-of-the-art crypto mining and crypto ransomware attacks. TIIS, 2019. 13(6): p. 3258-3279.

[6] Al-rimy, B.A.S., M.A. Maarof, and S.Z.M. Shaid, Ransomware Threat Success Factors, Taxonomy, and Countermeasures: A Survey and Research Directions. Computers & Security, 2018. 74: p. 144-166.

[7] Tandon, A. and A. Nayyar, A Comprehensive Survey on Ransomware Attack: A Growing Havoc Cyberthreat, in Data Management, Analytics and Innovation. 2019, Springer. p. 403-420.

[8] Chittooparambil, H.J., et al. A Review of Ransomware Families and Detection Methods. in International Conference of Reliable Information and Communication Technology. 2018. Springer.

[9] Nadir, I. and T. Bakhshi. Contemporary cybercrime: A taxonomy of ransomware threats & mitigation techniques. in 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). 2018. IEEE.

[10] Yazdinejad, A., et al., Cryptocurrency malware hunting: A deep recurrent neural network approach. Applied Soft Computing, 2020. 96: p. 106630.

[11] Kharraz, A., W. Robertson, and E. Kirda, Protecting against ransomware: A new line of research or restating classic ideas? IEEE Security & Privacy, 2018. 16(3): p. 103-107.

[12] Popli, N.K. and A. Girdhar, Behavioural Analysis of Recent Ransomwares and Prediction of Future Attacks by Polymorphic and Metamorphic Ransomware, in Computational Intelligence: Theories, Applications and Future Directions-Volume II. 2019, Springer. p. 65-80.

[13] Yu, B., et al., A survey of malware behavior description and analysis. Frontiers of Information Technology & Electronic Engineering, 2018. 19(5): p. 583-603.

[14] O'Kane, P., S. Sezer, and D. Carlin, Evolution of ransomware. IET Networks, 2018. 7(5): p. 321-327.

[15] Akbanov, M., V.G. Vassilakis, and M.D. Logothetis, WannaCry Ransomware: Analysis of Infection, Persistence, Recovery Prevention and Propagation Mechanisms. Journal of Telecommunications and Information Technology, 2019.

[16] Lemmou, Y. and E.M. Souidi. Infection, self-reproduction and overinfection in ransomware: the case of teslacrypt. in 2018 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security). 2018. IEEE.

[17] Cyberbit.com. Locky Ransomware: New Evasion Techniques Discovered. 2016; Available from: https://www.cyberbit.com/blog/endpoint-security/locky-ransomware-new-evasion-techniques-discovered/.

[18] Kharaz, A., et al. UNVEIL: A large-scale, automated approach to detecting ransomware. in 25th {USENIX} Security Symposium ({USENIX} Security 16). 2016.

[19] Dhawan, S. and B. Narwal. Unfolding the mystery of ransomware. in International Conference on Innovative Computing and Communications. 2019. Springer.

[20] Hurtuk, J., et al. Case Study of Ransomware Malware Hiding Using Obfuscation Methods. in 2018 16th International Conference on Emerging eLearning Technologies and Applications (ICETA). 2018. IEEE.

[21] Oyama, Y., Trends of anti-analysis operations of malwares observed in API call logs. Journal of Computer Virology and Hacking Techniques, 2018. 14(1): p. 69-85.

[22] Irshad, M., et al., Effective methods to detect metamorphic malware: a systematic review. International Journal of Electronic Security and Digital Forensics, 2018. 10(2): p. 138-154.

[23] Zimba, A. and M. Chishimba, Understanding the Evolution of Ransomware: Paradigm Shifts in Attack Structures. International Journal of Computer Network & Information Security, 2019. 11(1).

[24] M. F. Botacin, V.F.d.R., P. L. de Geus, A. R. A. Grégio, Analysis, Anti-Analysis, Anti-Anti-Analysis: An Overview of the Evasive Malware Scenario, in Anais do SBSeg'17, XVII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. 2017. p. 250-263.

[25] Maniath, S., P. Poornachandran, and V. Sujadevi. Survey on Prevention, Mitigation and Containment of Ransomware Attacks. in International Symposium on Security in Computing and Communication. 2018. Springer.

[26] Szor, P., The Art of Computer Virus Research and Defense: ART COMP VIRUS RES DEFENSE _p1. 2005: Pearson Education.

[27] Deka, D., N. Sarma, and N.J. Panicker. Malware detection vectors and analysis techniques: A brief survey. in 2016 International Conference on Accessibility to Digital World (ICADW). 2016. IEEE.

[28] MacRae, J. and V.N. Franqueira. On Locky Ransomware, Al Capone and Brexit. in International Conference on Digital Forensics and Cyber Crime. 2017. Springer.

[29] Varonis.com. The State of CryptoWall in 2018. 2018; Available from: https://www.varonis.com/blog/cryptowall/.

[30] Wong, W. and M. Stamp, Hunting for metamorphic engines. Journal in Computer Virology, 2006. 2(3): p. 211-229.

[31] Fayi, S.Y.A., What Petya/NotPetya ransomware is and what its remidiations are, in Information Technology-New Generations. 2018, Springer. p. 93-100.

[32] Genç, Z.A., G. Lenzini, and P. Ryan, The Cipher, the Random and the Ransom: A Survey on Current and Future Ransomware. Advances in Cybersecurity 2017, 2017.

[33] FortiNet Inc. Cerber 5.0.1 Arrives with New Multithreading Method. 2016; Available from: https://www.fortinet.com/blog/threat-research/cerber-5-0-1-arrives-with-new-multithreading-method.

[34] Lin, D. and M. Stamp, Hunting for undetectable metamorphic viruses. Journal in computer virology, 2011. 7(3): p. 201-214.

[35] Hofheinz, D., J. Malone-Lee, and M. Stam, Obfuscation for cryptographic purposes. Journal of Cryptology, 2010. 23(1): p. 121-168.

[36] Kaspersky Labs. Kaspersky Labs IT Threat Evolution Q2, 2019. 2020; Available from: https://securelist.com/it-threat-evolution-q2-2019/91994/.

[37] Pearce, M., S. Zeadally, and R. Hunt, Virtualization: Issues, security threats, and solutions. ACM Computing Surveys (CSUR), 2013. 45(2): p. 1-39.

[38] Kruegel, C. Full system emulation: Achieving successful automated dynamic analysis of evasive malware. in Proc. BlackHat USA Security Conference. 2014.

[39] McAfee Inc., McAfee Lab Report: Reviews of 30 Year Evolution of Evasion Techniques. 2017.

[40] Issa, A., Anti-virtual machines and emulations. Journal in Computer Virology, 2012. 8(4): p. 141-149.

[41] Branco, R.R., G.N. Barbosa, and P.D. Neto, Scientific but not academical overview of malware anti-debugging, anti-disassembly and anti-vm technologies. Black Hat, 2012. 1.

[42] CuckooSandbox.org. Cuckoo Sandbox. 2020; Available from: https://cuckoosandbox.org/.

[43] Shaukat, S.K. and V.J. Ribeiro. RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning. in 2018 10th International Conference on Communication Systems & Networks (COMSNETS). 2018. IEEE.

[44] Chailytko, A. and S. Skuratovich. Defeating sandbox evasion: how to increase the successful emulation rate in your virtual environment. in ShmooCon 2017. 2017.

[45] Ferrand, O., How to detect the cuckoo sandbox and to strengthen it? Journal of Computer Virology and Hacking Techniques, 2015. 11(1): p. 51-58.

[46] Zhang, F., et al., Towards transparent debugging. IEEE Transactions on Dependable and Secure Computing, 2016. 15(2): p. 321-335.

[47] Chen, P., et al. Advanced or not? A comparative study of the use of anti-debugging and anti-VM techniques in generic and targeted malware. in IFIP International Conference on ICT Systems Security and Privacy Protection. 2016. Springer.

[48] BlackHAT.com. Windows Process Injection in 2019. 2019; Available from:https://i.blackhat.com/USA-19/Thursday/us-19-Kotler-Process-Injection-Techniques-Gotta-Catch-Them-All-wp.pdf.

[49] EndGame.com. Ten Process-Injection Techniques: A Technical Survey of Common and Trending Process Injection Techniques. 2017; Available from: https://www.endgame.com/blog/technical-blog/ten-process-injection-techniques-technical-survey-common-and-trending-process-2017.

[50] Sharma, H. and S. Kant, Early Detection of Ransomware by Indicator Analysis and WinAPI Call Sequence Pattern, in Information and Communication Technology for Intelligent Systems. 2019, Springer. p. 201-211.

[51] Lopez, J., et al., A survey on function and system call hooking approaches. Journal of Hardware and Systems Security, 2017. 1(2): p. 114-136.

[52] Shaid, S.Z.M. and M.A. Maarof. In memory detection of Windows API call hooking technique. in 2015 International conference on computer, communications, and control technology (I4CT). 2015. IEEE.

[53] CryptoPP.com. Crypto++ Library. 2020; Available from: https://www.cryptopp.com/.

[54] Rudd, E.M., et al., A survey of stealth malware attacks, mitigation measures, and steps toward autonomous open world solutions. IEEE Communications Surveys & Tutorials, 2016. 19(2): p. 1145-1172.

[55] Homem, I., S. Tanda, and I. Korkin. Detect Kernel-Mode Rootkits via Real Time Logging & Controlling Memory Access. in Proceedings of the Conference on Digital Forensics, Security and Law. 2017. Association of Digital Forensics, Security and Law.

[56] Dargahi, T., et al., A cyber-kill-chain based taxonomy of crypto-ransomware features. Journal of Computer Virology and Hacking Techniques, 2019. 15(4): p. 277-305.

[57] Berrueta, E., et al., A survey on detection techniques for cryptographic ransomware. IEEE Access, 2019. 7: p. 144925-144944.