I.J. Information Technology and Computer Science, **2018, 1, 40-51**
Published Online January 2018 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2018.01.05

Towards Data Resilience: The Analytical Case of Crypto Ransomware Data Recovery Techniques

Aaron Zimba, Zhaoshun Wang
University of Science and Technology Beijing/Computer Science and Technology, Beijing, 100083, China
E-mail: azimba@xs.ustb.edu.cn

Luckson Simukonda
Gdańsk University of Technology/Electronics, Telecommunications & Informatics, Gdańsk, 80-233, Poland
E-mail: thezo1992@gmail.com

Received: 08 October 2017; Accepted: 17 November 2017; Published: 08 January 2018

***Abstract*—**Crypto ransomware has earned an infamous reputation in the malware landscape and its sound sends a lot of shivers to many despite being a new entrant. The media has not helped matters even as the myths and inaccuracies surrounding crypto ransomware continue to deepen. It's been purported that once crypto ransomware attacks, the victim is left with no option but to pay in order to retrieve the encrypted data, and that without a guarantee, or risk losing the data forever. Security researchers are inadvertently thrown into a cat-and-mouse chase to catch up with the latest vices of the aforesaid in order to provide data resilience. In this paper, we debunk the myths surrounding loss of data via a crypto ransomware attack. Using a variety of crypto ransomware samples, we employ reverse engineering and dynamic analysis to evaluate the underlying attack structures and data deletion techniques employed by the ransomware. Further, we expose the data deletion techniques used by ransomware to prevent data recovery and suggest how such could be countered. From the results, we further present observed sandbox evasion techniques employed by ransomware against both static and dynamic analysis in an effort to obfuscate its operations and subsequently prevent data recovery. Our analyses have led us to the conclusion that no matter how devastating a crypto ransomware attack might appear, the key to data recovery options lies in the underlying attack structure and the implemented data deletion methodology.

***Index Terms*—**Ransomware, Data Resilience, Recovery, Data Deletion, Attack Structure.

I. Introduction

Regardless of the environment in which a user interacts with his/her data, whether in the cloud, the Internet, private network or otherwise, the overall desire is to have secure and resilient data. Since systems are not inherently secure, Confidentiality, Integrity and Availability (CIA) mechanisms ensure that the aforementioned requirements are achieved. Confidentiality mechanisms employ encryption to ensure that data is revealed only to the intended recipients [1] whereas authentication mechanisms provide for data authenticity, hence Integrity [2]. On the other hand, countering data breaches against Availability isn't as straight forward owing to the evolution and diverse nature of the associated attack vectors. Most attacks on Availability come in form of Denial of Service (DoS) [3] and are usually directed on networks that be. Furthermore, a new DoS attack vector has emerged in the name of Ransomware [4]. This is an attack on data Availability that has come closest to the user. The user's files or system are held at ransom and are only made usable when a ransom demand is met. This type of attack has not spared any sector of the economy not limited to education, services, manufacturing, transport etc [5]. This has cast a lot of doubt on the data resilience of most systems even as security analysts are left with an uphill battle in trying to catch up with the ever mutating ransomware malware. Encryption mechanisms that are meant to counter breaches against Confidentiality are abused by cyber-criminals to carry out very effective ransomware attacks [6]. Since its inception, ransomware has been on the rise with newer strains employing complex attack methodologies thus casting even a darker shadow of doubt on data resilience on connected systems. However, there exists a mist of uncertainty surrounding the operations and the extent of the damage caused by this type of malware and the possible recovery approaches if any. Mainstream media has not helped matters even as facts are distorted with impunity to the detriment of the benign user. Ransomware is known to affect major operating systems not limited to Windows, Linux, Macintosh etc [7]. Antiviruses and Intrusion Detection and Prevention Systems (IDS & IPS) are thrown into a cat-and-mouse race to cope with the fast changing ransomware landscape [8]. System administrators are left in a limbo on how to address ransomware attack incidents. This has inadvertently called for an analytical evaluation of the attack structures and recovery techniques of ransomware for data resilience.

In light of this, in this paper, we endeavor to evaluate ransomware attack recovery techniques from an attack

structure point of view. We contend that ransomware strains with the same attack structure can be addressed with a similar recovery technique provided such a one exists. We experiment and analyze a myriad of samples and suggest a mitigation approach depending on the observed attack structure.

The rest of the paper is outlined as follows: Related works are discussed in Section II. Section III discusses the common attack structures and related concepts while the experimental setup and methodology are brought forth in Section IV. Results and the analyses thereof are presented in Section V while recommendations and best practices are brought forth in Section VI and we draw the conclusion in Section VII.

II. Related Works

Ahmadian et al. [23] present an approach for detecting and preventing highly survivable ransomware based on a principal feature discovered in the taxonomy. The focus of their work is to provide data resilience via monitoring and preventing the encryption of the victim's data. Based on dynamic malware analysis, the prevention methodology is successful when the ransomware tries to contact the C2 to download the encryption key. Though the findings are interesting, it clear that this prevention mechanism only works for ransomware that do not have embedded encryption keys. The work does not detail how the ransomware deletes the target files after encryption. This is important because even if ransomware manages to encrypt a victim's data, it's still possible to recover the data without paying the ransom depending on the deletion techniques employed.

Scaife et al. [24] present CryptoDrop, a dynamic analysis based early-warning detection system that informs a victim upon discovery of suspicious file activity. The system stops a process tampering with large amounts of the user's data. Their tests provide interesting results of a median loss of only 10 files from 5,100 files. However, the whole concept is based on the premise that multiple file access is an indicator of compromise. Further, there is no provision as to what type of deletion methodology is implemented by the observed samples because even the 10 lost files could be recovered depending on the deletion methodology.

Kirda [25] approaches ransomware attacks from a qualitative point of view and argues that the majority of ransomware isn't as complex as portrayed. He argues that behavioral characteristic which are common in all ransomware can be used to detect it. The author addresses some of the deletion techniques and some of the strong encryptions used. The author contends that even access to CryptoAPI functions which is used by most crypto ransomware can be monitored. However, there is no formulation of the attack model and ransomware is broadly addressed to refer to both crypto and non-crytpo ransomware. Further deletion of the shadow volume copy and privilege escalation are not put into consideration which in essence are part of the methodologies which ransomware uses to make data recovery almost impossible.

Mercaldo et al. [26] present a static analysis method to automatically inspect ransomware sample based on Android. The dissection is executed with the goal of testing whether the malicious behavior represent a class of ransomware functionalities. Though the work is largely based on ransomware for Android systems, the deletion mechanism is not clear. Further, it's not evident how the ransomware seeks to prevent data recovery.

Cabaj and Mazurczyk [27] propose the use of SDN to mitigate ransomware attacks. The approach is based on dynamic analysis of CryptoWall whereupon two real-time mitigation approaches are proposed. The proposed SDN-based applications rely on updated databases of proxy servers of ransomwares. The work does not detail how ransomware achieves data recovery prevention. Moreover, like the other previous work, this approach only addresses those families of ransomware that need to contact the C2 server to effectuate the encryption.

III. Attack Structures and Related Concepts

Crypto ransomware attack methodologies come in two flavors; those employing symmetric key cryptosystems and those employing hybrid crypto systems. The former uses standard and custom symmetric key encryption algorithms for encrypting the target payload whereas the latter combines both symmetric and asymmetric key encryption algorithms to achieve the same. However, attack structures utilizing only symmetric key and custom made algorithms have been found to be weak and easily crackable to recover the data [9]. Attackers have thus shifted towards using the hybrid approach where standard symmetric algorithms, e.g. DES, AES, 3DES etc are used to encrypt the target data and the asymmetric encryption, e.g. RSA, ECC is used to encrypt the symmetric key. It's quite obvious that symmetric algorithms are faster than their asymmetric counterparts hence their use for encrypting the payload [10]. The majority of ransomware samples are known to use the CryptoAPI from operating system's Cryptographic Service Provider (CSP) to access cryptographic services for use in the ransomware payload. In light of the aforementioned, the attacker thus seeks to secure the private key from the asymmetric key pair. This leaves the attacker with two options:

(1) whether to ***locally generate*** the asymmetric key pair on the victim after infection,

(2) or whether to contact the Command & Control (C2) servers to ***download*** the asymmetric key pair [11].

Local Key Generation

This is the most practical and efficient method of handling the asymmetric key pair. It does not need third party communications and takes off overhead from the C2 severs. It is faster to implement as it uses the target operating system's CryptoAPI. The drawback however is that since the seed to the asymmetric key pair generation is gotten from the local machine, it is possible, via memory analysis of persistent memory to retrieve these

parameters and rebuild the key via reverse engineering. This was the major implementation weakness of WannaCry ransomware that made it possible to retrieve the RSA primes used in asymmetric key generation [12].

C2 Server Key Download

This is the ideal approach as it is practically infeasible to derive the matching private key of a public key download from the C2 servers. This was the approach implemented in Cryptowall-4 [13]. Nonetheless, this approach encounters the challenge of beaconing back to the C2 servers after a successful infection. This is only possible on the pretext that the victim has an active and steady Internet connection which can never be guaranteed. Furthermore, it requires a secure communication channel between the C2 servers and the ransomware agent.

Since the spectrum of infection vectors is so wide (from both online and offline attack vectors), it is impossible to guarantee absolute Internet connection on the victims end. This has led to attackers generating asymmetric key pairs locally on the victim as evidenced in many ransomware samples. Once the infection occurs and the subsequent encryption, ransomware seeks to incapacitate data resilience and make recovery efforts futile by deleting the original copies of the files. This is mainly done in any one of the following: (i) deleting the original file after encryption thus leaving only the resultant cipher-text (ii) overwriting the original file with random data to make restoration impossible (iii) deleting volume shadow copy snapshots in the system via vssadmin.exe. This scenario subdivides four attack instances on data resilience as elaborated in figure 1 below.

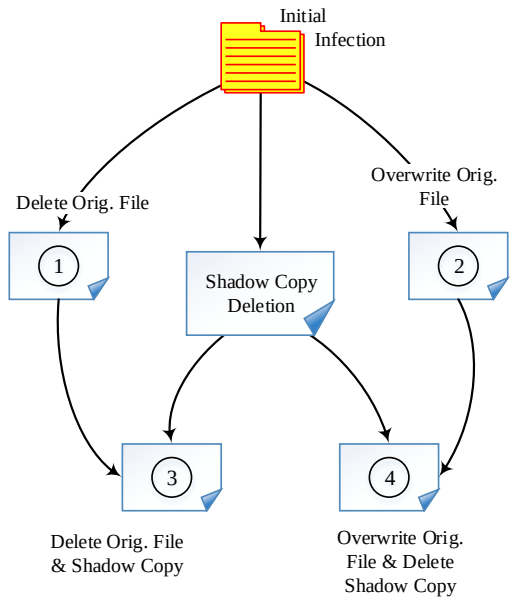


Fig.1. Data recovery attack structures

A ransomware attack on data recovery can implement any of the four attack instances depicted in figure 1. Attack methodologies employing instances 1 and 3 can easily be countered by recovery application such as Photorec or Recuva [14] which seek to recover lost directory structures and the lost meta-data of the deleted files. In such scenarios, there's no need to retrieve the encryption key since the deleted files can be recovered. Notwithstanding the aforementioned, most attackers employ attack structures that utilize instances 2 and 4 where recovery of deleted files using the aforesaid software resources does not apply. In such instances, recovery of the original data can only be done through decryption using the applicable encryption keys. It's important to note that attack instances 2 and 4 are independent and so are 1 and 3.

The attack model

Using the attack structures in figure 1, we now endeavor to characterize the attack model which describes data deletion and the recovery prevention thereof as used in this paper. We fuse an attack graph and a finite state machine to express the deletion attack of target data on a victim's system as shown in figure 2 below. The resultant attack graph is a Directed Acyclic Graph (DAG) defined as:

$$G_{i,j}^{del} = (N_i, E_{i,j}) \quad (1)$$

where the nodes $N_i = \{n_i \mid i = 0,1,2,3,\dots\}$, is a set of vertices $n_i \in N_i$ traversed to reach the goal (data deletion) and the edges $E_{i,j} = \{e_{i,j} \mid i,j = 0,1,2,3,\dots\}$, is a set of arcs $e_{i,j} \in E_{i,j}$ representing attack actions yielding the traversal. The system starts in S_0 (*Secure State - SS*) represented by n_0 whereupon infection by the ransomware transitions through t_1 to state S_1 (*Privilege Escalation - PE)* representative of the node set $\{n_1, n_2\}$. The edge $e_{0,1}$ leading to n_1 represents a malware infection into a user profile with no elevated privileges to enable execution of the ransomware. The ransomware in most cases inserts itself into a legitimate process in order to attain the required privileges thus transitioning to n_2 via the edge $e_{1,2}$. Alternatively, some infections land the ransomware into a privileged user profile and this is depicted by the attack edge $e_{0,2}$ to node n_2.

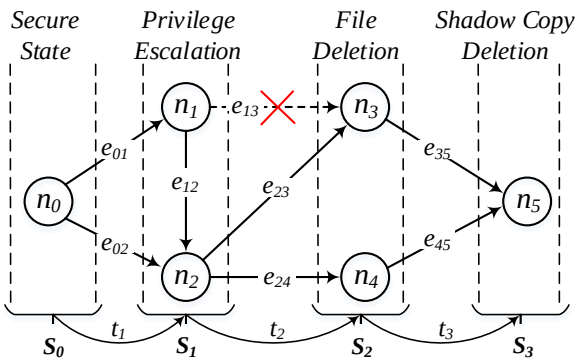


Fig.2. Attack model state transitions

Upon attainment of privileged mode in state S_1, the ransomware proceeds to delete the original files after

encryption. This is depicted by attack edge $e_{2,3}$ to node n_3 and $e_{2,4}$ to n_4. This transitions the system to state S_2 (*File Deletion - FD*) representative of the node set $\{n_3, n_4\}$ via transition t_2. The edge $e_{1,3}$ to node n_3 is infeasible because we suppose that the target system implements standard Discretionary Access Control (DAC). This would require the ransomware to be added to the target file's Access Control List (ACL) or to take the identity of the target file's owner, hence the privilege escalation via $e_{1,2}$.

The node n_3 represents system state upon normal file deletion corresponding to attack instance 1 of the attack structure in figure 1. Conversely, n_4 represents system state upon file deletion which encompasses overwriting the original file with random data which in essence corresponds to attack instance 2 of the attack structure in figure 1. Some ransomware are known to end here as their final goal whereas other tend to further seek to incapacitate data recover by deleting shadow copies [15]. The latter is depicted by attack edges $e_{3,5}$ and $e_{4,5}$ which necessitates transition t_3 leading to node n_5. The resultant is state S_3 (*Shadow Copy Deletion - SCD*) representative of a state where original data has been deleted and the corresponding shadow copies. In order characterize the linkages between the nodes, we deduce the adjacency matrix A_M of the DAG in figure 2:

$$A_M = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

It's vivid from the adjacency matrix that node n_2 is the isthmus linking the passive infection phase and the active attack process which is the actual file deletion. Thus elimination of this node ought to be prioritized in the mitigation process. Following from the Equation (2), we deduce the possible attack paths from n_0 in the secure state S_0 to n_5 in the breached state S_3.

$$\begin{aligned} P_1&: n_0 \to n_2 \to n_3 \to n_5 \\ P_2&: n_0 \to n_2 \to n_4 \to n_5 \\ P_3&: n_0 \to n_1 \to n_2 \to n_3 \to n_5 \\ P_4&: n_0 \to n_1 \to n_2 \to n_4 \to n_5 \end{aligned} \quad (3)$$

Even though path P_1 appears to be a subset of P_3 and path P_2 a subset of P_4, they are all independent unique paths. The similarity entails the two possible states in S_1 and S_2 represented by the node set $\{n_1, n_2\}$ and $\{n_3, n_4\}$ respectively. If we let the parameters $\mu_k, \varphi_k, \omega_k$ to denote *PE*, *FD* and *SCD* respectively, where k is such that $k \in [-, +]$, then we can express any given state as a function:

$$\boldsymbol{S_n}(\mu_k, \varphi_k, \omega_k) \text{ where } n \in \{ \mathbb{R}^+ \cup \{0\}\} \quad (4)$$

Since state $\boldsymbol{S_3}$ has only one possible state represented by n_5, the corresponding status equation can be expressed as:

$$\boldsymbol{S_3}(\mu_k, \varphi_k, \omega_k) = \omega \quad (5)$$

If we let μ_- and μ_+ to represent *privileged* and *unprivileged* mode respectively, then the state equation for state $\boldsymbol{S_1}$ is an exclusive OR operation expressed as:

$$\boldsymbol{S_1}(\mu_k, \varphi_k, \omega_k) = \mu_- \oplus \mu_+ \quad (6)$$

In the same vein, if φ_- and φ_+ represent *original file deletion* and *original file overwriting*, the state equation for state $\boldsymbol{S_2}$ is thus expressed as:

$$\boldsymbol{S_2}(\mu_k, \varphi_k, \omega_k) = \varphi_- \oplus \varphi_+ \quad (7)$$

The exclusivity in Equation (6) and (7) entail that although path P_1 appears to be a subset of P_3 and path P_2 a subset of P_4, they are independent events. Even though it is possible for ransomware to run two distinct processes; one in unprivileged and another in privileged mode, it defies attack logic and it's only reasonable to postulate that the attacker will want to run in privileged mode or otherwise but not both modes. In the same manner, most ransomware will either delete or overwrite the original file after encryption [16]. Having described the attack model, we now endeavor to demonstrate the ransomware attacks that seek to breach data availability via deletion after encryption and prevent recovery efforts.

IV. Experiment Set-up and Methodology

We employ two approaches to all ransomware samples in our experiments: reverse engineering (static analysis) and behavioral analysis (dynamic analysis). These analyses are performed in a confined sandbox environment in conformity to recommended best practices [17].

Static Analysis

In static analysis, we perform an autopsy to extract malware features without running it and analyze the source code fragments for functions and directives related to data deletion and recovery prevention. In the same manner, we look at some features that counter malware analysis since it's through such analysis that we seek to implement data recovery. Further in this analysis, we use a range of malware analysis disassembly tools not limited to Ollydebug, IDA Pro, Dependency Walker etc for dissection of the malware code internal logic. The diagram depicted in figure 3 below illustrates the stages we employ in reverse engineering.

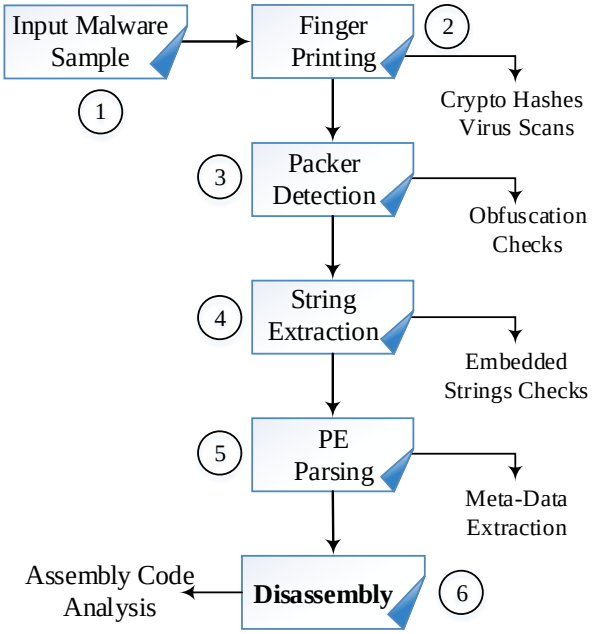


Fig.3. Summarized static analysis stages

In stage 1, we ready our ransomware samples. We select a wide range of specimens including the latest samples at the time of writing not limited to WannaCry, Cerber, Cryptowall, Locky etc which dominate the ransomware landscape accounting for 90% of the attacks [18]. We begin the extraction of external features in stage 2 where we compute the samples' cryptographic hashes to determine their identities. We further solidify the samples' authenticity and extract extra features by submitting them to reputed malware databases Virustotal.com, Malwr.com and Virscan.org. This removes false negatives and detects any changes in the original source code. We counter-check for obfuscation in stage 3 to determine whether the malware author compressed, encrypted or altered the ransomware contents. Thus we check whether the sample has been packed to disguise the internal program logic. In stage 4, we check for embedded strings related to file deletion and recovery prevention and any other relevant strings. We extract the meta-data in stage 5 by parsing the executables through corresponding tools (PEView and Resource Hacker). Finally, after having extracted all these external features, we proceed to stage 6 where we disassemble and debug the code using IDA Pro and Ollydebug respectively and hunt for techniques employed to delete target data and prevent the recovery thereof as elaborated in the preceding section.

Dynamic Analysis

We perform the second class of tests where we actively interact with the malware by running it in a sandbox environment and polling the associated behavioral features. The test-bed for dynamic analysis, as shown in figure 4 below, comprises two components: the server-side and client-side component.

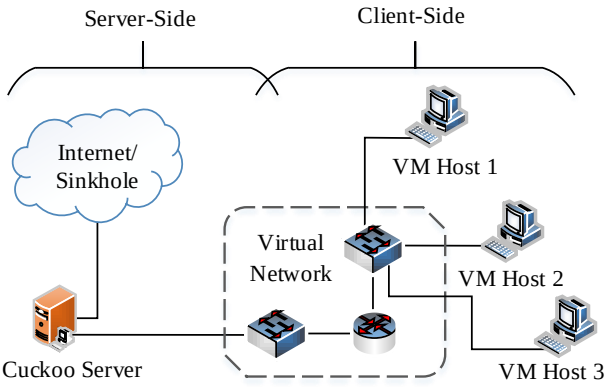


Fig.4. Dynamic analysis test-bed

The server-side component is a Linux system (Ubuntu) running Cuckoo sandbox and Volatility whereas the client-side comprises virtual machines of different versions of the Windows operating system. We limit our scope to the Windows desktop edition considering its widespread use [19]. We deliver the ransomware to the victim using the Cuckoo server console and collect all the behavioral features of the ransomware from the generated report. We seek limit the network activity to "host-only" and disable DNS resolutions for those samples with kill-switch domains as discovered during the static analysis stage. Since the infection vector is not relevant according to the attack model, we iteratively use the Cuckoo server console to deliver the ransomware to all the victims by specifying the IP addresses of the victim virtual machines. We collect all relevant features from both static and dynamic analysis and tabulate them in the proceeding results' section.

V. Results and Analyses

In order to ensure that data recovery efforts stay futile after a ransomware attack, malware authors employ two main approaches; one directed towards a victimized user seeking data recovery and one towards a malware analyst experimenting on the ransomware in a contained environment. Attack techniques directed to a victimized user include deleting the original file after encryption, overwriting it, deletion of volume shadow copies and the use of resilient standard encryption to make decryption of the victimized files impossible. In the case of WannaCry, the procedure just before deletion of the original files is shown in the code snippet in figure 5 below.


```
GetWindowsDirectoryW(pathToMoveFiles, MAX_PATH);
if ( *pathToMoveFiles == diskLabel + 'A' )
{
  GetTempPathW(MAX_PATH, pathToMoveFiles);
  if ( wcslen(pathToMoveFiles) && pathToMoveFiles[wcslen(pathToMoveFiles) - 1]
  {
    pathToMoveFiles[wcslen(pathToMoveFiles) - 1] = 0;
    return pathToMoveFiles;
  }
}
else
{
  swprintf(pathToMoveFiles, L"%C:\\%s", (diskLabel + 'A'), L"$RECYCLE");
  CreateDirectoryW(pathToMoveFiles, 0);
  sprintf(&lpCommandLine, "attrib +h +s %C:\\%s", diskLabel + 'A', "$RECYCLE");
  CreateProcess(&lpCommandLine, 0, 0);
```

Fig.5. Procedure for handling files designated for deletion

The last part of the code creates a temporal directory *"$RECYCLE"* where targeted user files meeting the conditions stated in the first part of the code are moved to. The ransomware obfuscates this directory using the attribute *"attrib +h +s"* in order to hide it from the Windows Explorer. Upon proper synchronization in the code, the targeted files are deleted and not overwritten. However, if errors occur in the synchronization process, targeted files might not actually get deleted but will remain hidden in the *"$RECYCLE"* directory or might not even be moved at all. This un-thorough way of deleting files is unsecure and targeted files can be recovered using recovery software discussed in the preceding section.

However, if the targeted files reside in directories classified as *"priority"*, the ransomware alters the deletion methodology. This is shown in figure 6 below.

```
SHGetFolderPathW(0, CSIDL_DESKTOP, 0, 0, &pszPath);
if ( wcslen(&pszPath) )
  ProcessDir(wannaCtx, &pszPath, 1);
LOWORD(pszPath) = 0;
SHGetFolderPathW(0, CSIDL_PERSONAL, 0, 0, &pszPath);
if ( wcslen(&pszPath) )
  ProcessDir(wannaCtx, &pszPath, 1);
ProcessDrives(CSIDL_COMMON_DESKTOPDIRECTORY, callback_Proc
ProcessDrives(CSIDL_COMMON_DOCUMENTS, callback_ProcessDir
```

Fig.6. Deletion of "priority files" via overwriting

In the code snippet above, the ransomware classifies files in the Desktop and Documents directories as priority. The original files meeting the above condition (residing either Desktop or Documents) are overwritten with random data and it's virtually impractical to retain the data. The rationale behind prioritizing files for deletion can only be speculated. We contend that it's logical that most users keep their working files either on their Desktop or in their Documents directory but malware authors could have had other reasons for doing so. The ransomware targets user files and not system files. Thus, file extensions such as .exe, .dll and other critical directory likewise are skipped.

After having deleted the files as specified above, the ransomware goes further to prevent any system recovery by deleting volume shadow copies with the command:

cmd.exe /c vssadmin delete shadows /all /quiet & wmic shadowcopy delete & bcdedit /set {default} bootstatuspolicy ignoreallfailures & bcdedit /set {default} recoveryenabled no & wbadmin delete catalog -quiet

This deletes volume shadow copies and all created system restore points and further disables any debugging of startup errors. However, this is a privileged command and for systems configured with User Account Control (UAC), the following window pops up:

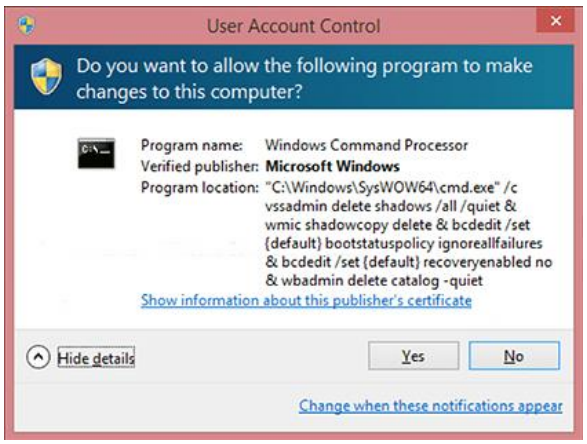


Fig.7. UAC seeking volume shadow copy deletion

It's only at this time window that the victim might have a chance to restore their data depending on their action. However, if the command is invoked in root mode or if UAC is disabled (as specified in the attack model), the victim might not encounter this pop up.

At this point, the data recovery prevention techniques by the ransomware are complete. It proceeds to make itself persistent by adding corresponding registry entries and enlisting in the startup so that the malware runs each time the system is restarted. There are other actions that the ransomware performs such as network activity, checking if the victim is already infected, trying to propagate on port 445 to exploit the SMB vulnerability [20] on unpatched systems etc.

The ransomware observed herein mostly used functions present in the operating system to effectuate the actual encryption. The most commonly observed encryption algorithms were RSA and AES. The RSA was mainly used to encrypt the symmetric AES key whereas

the AES itself was used to encrypt the actual files. The diagram below in figure 8 shows one of the observed cryptographic algorithms used by some samples. We collect a couple of external features of the samples from the analysis methodology presented thus far. Some samples of the same family had different variations depending on year in which they were released in the wild. Such variations include update to loopholes and errors in the code that made data recovery possible.

```
aMicrosoftEnhan db 'Microsoft Enhanced RSA and AES Cryptographic Provider'
                                        ; DATA XREF: sub_40182C+14↑o
                align 4
; CHAR aCryptgenkey[]
aCryptgenkey    db 'CryptGenKey',0      ; DATA XREF: sub_401A45+68↑o
; CHAR aCryptdecrypt[]
aCryptdecrypt   db 'CryptDecrypt',0     ; DATA XREF: sub_401A45+5B↑o
                align 10h
; CHAR aCryptencrypt[]
aCryptencrypt   db 'CryptEncrypt',0     ; DATA XREF: sub_401A45+4E↑o
                align 10h
; CHAR aCryptdestroyke[]
aCryptdestroyke db 'CryptDestroyKey',0  ; DATA XREF: sub_401A45+41↑o
; CHAR aCryptimportkey[]
aCryptimportkey db 'CryptImportKey',0   ; DATA XREF: sub_401A45+34↑o
                align 10h
; CHAR aCryptacquireco[]
aCryptacquireco db 'CryptAcquireContextA',0 ; DATA XREF: sub_401A45+2C↑o
                align 4
                dd offset a_doc         ; ".doc"
```

Fig.8. Ransomware encryption routines

Table 1 below shows external features of 10 different samples used in the experiment. We specify the methodology used by each sample to extract the encryption key, whether to download from the C2 server after initial beaconing or use the operating system's CryptoAPI.

Table 1. External features for various samples

Sample	Key Gen. Method	Public Key	Private Key	CSP	File Size
Sample1 (Cerber)	Local	RSA	RC4	ECP	244 KB
Sample2 (CryptoWall)	C2 Download	RSA	RSA	ECP	404 KB
Sample3 (WannaCry)	Local Generation	RSA	AES	ECP	3636 KB
Sample4 (Locky)	C2 Download	RSA	AES	ECP	116 KB
Sample5 (TeslaCrypt)	Local	ECC	AES	BCP	384 KB
Sample6 (Petya)	Local	ECC	Salsa20	-	788 KB
Sample7 (Gpcode)	C2 Download	RSA	AES	ECP	29 KB
Sample8 (CTB-Locker)	Local	ECC	AES	ECP	1173 KB
Sample9 (CryptoLocker)	C2 Download	RSA	AES	ECP	549 KB
Sample10 (NotPetya)	Local	RSA	AES	-	309 KB

We observed in the samples that the majority generated encryption keys locally. Local key generation accounted for 60% while C2 Download accounted for the remaining 40%. It was noted however that most newer versions of those that originally downloaded the encryption key were able to begin the attack process even without communicating to the C2. A shift towards ransomware attacking without contacting the C2 servers could only be explained that it's not always that the target host will have an Internet connection or firewalls and IDS might actually restrict network access depending on the scenario.

Almost all the samples used a public key to encrypt the symmetric key that encrypted the files. Those that did not need to contact the C2 to attack came with an embedded public key of which RSA accounted for 70% while the remaining 30% was ECC. It's observed that most recent ransomware versions are shifting towards using RSA for asymmetric key encryption. The RSA public key is usually not used to encrypt user files due to its computational intensiveness resulting in significant CPU like in the case of CryptoWall. The majority of ransomware (80%) used symmetric block ciphers for file encryption with AES accounting for 70% while the stream ciphers accounted for 20% which included RC4 and Salsa20.

Almost all the samples made use of the CryptoAPI provided Microsoft Cryptographic Services. The Enhanced Cryptographic Provider (ECP) library accounted for 70% due to its provision of stronger security via longer keys and additional algorithms while

the Base Cryptographic (BCP) was observed in one sample. The average file size of the ransomware payload was about 750 KB. Such a small size of less than a Megabyte is easily downloadable and unnoticeable. The ransomware once run tries to hide by assuming the identity of a trusted process or injects itself into one.

We further tabulate behavioral characteristics of the samples that seek to make data recovery impossible and present them in Table 2 below with other related characteristics of the samples.

Sandbox Evasion

It was observed that a number of the sample employed some techniques to conceal their operations and thus make data recovery even harder. The ransomware checks against a number of conditions and would not run if a condition is satisfied. Others actually would go ahead to crash the debugger once detected. The diagram in figure 9 below shows sandbox evasion against Virtual Box. The malware checks if the VBoxService.exe is running and if it detects such a process, it will switch into sleep mode and there will be no further analysis. This will inhibit efforts to find out how the malware operates and that subsequently thwart any sought data recovery attempts whatsoever in this respect.

```
; Attributes: bp-based frame

sub_408B09 proc near

var_18= dword ptr -18h

push    ebp
mov     ebp, esp
sub     esp, 18h
mov     [esp+18h+var_18], offset aVboxservice_ex ; "VBoxService.exe"
call    sub_408A28
leave
retn
sub_408B09 endp
```

Fig.9. Sandbox evasion against virtual box

```
; Attributes: bp-based frame

sub_408B1D proc near

var_18= dword ptr -18h

push    ebp
mov     ebp, esp
sub     esp, 18h
mov     [esp+18h+var_18], offset aWireshark_exe ; "Wireshark.exe"
call    sub_408A28
leave
retn
sub_408B1D endp
```

Fig.10. Sandbox evasion against Wireshark

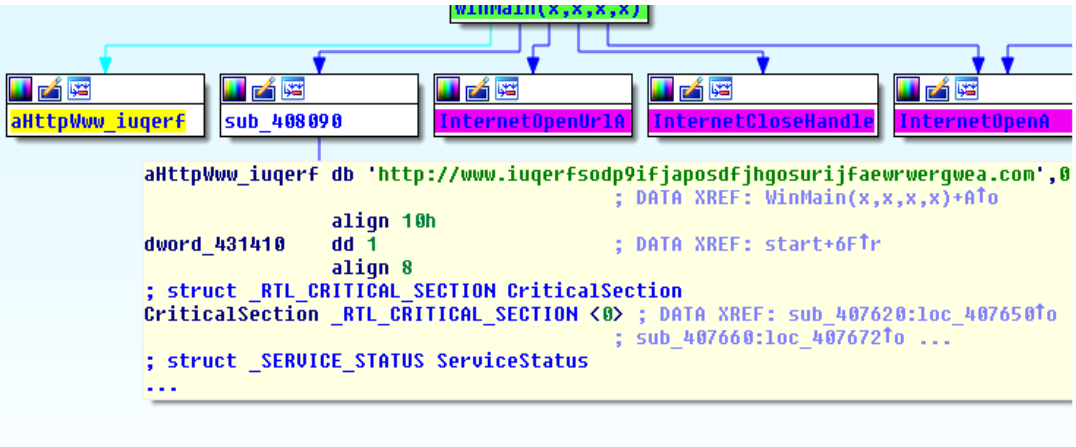


Fig.11. Sandbox evasion against kill switch

Table 2. Behavioral features and Characteristics

Sample	Delete Orig. File	Overwrite Orig. File	Delete Shadow Copy	C2 Beaconing	Sandbox Evasion	Persistent Presence	First Appeared
Sample1 (Cerber)	✓	✓	✓	-	✓	✓	2016
Sample2 (CryptoWall)	-	✓	✓	✓	✓	✓	2014
Sample3 (WannaCry)	✓	✓	✓	-	✓	✓	2017
Sample4 (Locky)	-	✓	✓	✓	✓	✓	2016
Sample5 (TeslaCrypt)	-	✓	✓	-	✓	✓	2015
Sample6 (Petya)	-	✓	✓	-	✓	✓	2016
Sample7 (Gpcode)	✓	✓	-	✓	-	-	2004
Sample8 (CTB-Locker)	✓	✓	-	-	✓	✓	2014
Sample9 (CryptoLocker)	✓	✓	-	✓	✓	✓	2013
Sample10 (NotPetya)	✓	✓	✓	-	✓	✓	2017

```
; Attributes: bp-based frame

sub_408B60 proc near

var_128= dword ptr -128h
var_124= dword ptr -124h
var_10C= dword ptr -10Ch

push    ebp
mov     ebp, esp
sub     esp, 128h           ; lpClassName
lea     eax, [ebp+var_10C]
mov     dword ptr [eax], 594C4C4Fh
mov     dword ptr [eax+4], 474244h
mov     [esp+128h+var_124], 0
mov     [esp+128h+var_128], eax
call    FindWindowA
sub     esp, 8              ; hObject
test    eax, eax
jz      short locret_408BA0
```

Fig.12. Sandbox evasion against Ollydebug

```
Explicit PID: 3472

Public key (00000000.pky) is in current directory, let's use it Modulus
B7825005FA2C91C0C57DE587FD0196816A6F560BC89662F9928E1EF6E2AFD1EC1F2B41FDB526
C06B8289E0F6F53C8084A962D23F927.....

Exponent 010001

Searching for primes numbers in memory.....

Prime1
F6F35985D35E1D232AAAECC3712E56D8EC1E8B180FEFAAE6E27D9EB1B31D24A97B5669A6C7B6
BCC77D1FF30102F91DD6.....

Prime2
BE3BD0E14336D6EAADA58195471132DC8377ADF925A16D9BD7BC009694E7A68B6A3A3B0A0CF4
7C50D5D3FEC41A3213BFFA365B.....

Let's save private key blob in 00000000.dky file Using raw user private key!

File c:\AUTOEXEC.BAT.WNCRY -- OK

File c:\Documents and Settings\All Users\Application Data\Microsoft\User
Account Pictures\Default Pictures\airplane.bmp.WNCRY -- OK

File c:\Documents and Settings\All Users\Application Data\Microsoft\User
Account Pictures\Default Pictures\astronaut.bmp.WNCRY -- OK
```

Fig.13. Target file decryption using primes left in the ransomware allocated memory

In the same manner, some malware look for other virtualizations services such as VMware and Qemu. Some malware implement sandbox evasion against efforts to sniff commutations to C2 servers as show in figure 10 above. Since Wireshark is one of the most used packet sniffers, the malware checks to see if such a process is running and if so detected, it won't run. However, this check is done on the guest running the sandbox, so a solution would be to run the packet sniffer from another host but in the same subnet. The WannaCry sample we ran checks against a kill-switch to check if it was running in a sandbox environment. However, a newer version of the ransomware does not include the kill switch domain.

Figure 11 above shows a check against such a kill switch. The ransomware checks to see if the fake non-existent domain is reachable. Such a domain would only be reachable in a sandbox environment where automatic valid DNS responses are present.

Kill switch domains were eliminated in new WannaCry strains because security researchers registered some of these fake domains consequently leading to the ransomware not running when it beacons to such domains.

The sandbox evasion techniques discussed thus far are directed towards dynamic analysis. Newer ransomware strains, however, employ sandbox evasion techniques against static analysis as. These target analysis and debugging tools as shown in figure 12 above.

In the above code, the targeted analysis tool is even encoded. The arguments *594C4CFh* and *474244h* when converted to ASCII read *"YLLO"* and *"GBD"* respectively which is equivalent to Ollydebug when read in reverse. Basically the malware checks for an Ollydbg window via the *call FindWindowA* and prevents any further analysis if the condition is met.

We observe in Table 2 that almost all the samples employed some form of either dynamic or static analysis sandbox evasion, with the majority against dynamic analysis. Earlier versions of Gpcode showed fewer signs of sandbox evasion if any at all, apparently due to the fact that it represents one of the earliest forms and not a new form of ransomware. It's only in later versions that it started to include complicated techniques such as RSA and AES encryption. It can be observed form Table 2 that as ransomware matures, so do the techniques employed to make data recovery impossible. This is evidenced throughout the timeline where earlier versions did not give priority to shadow copy deletion and use of unsecure deletion techniques. Furthermore, recent versions likewise do not need to contact the C2 server to effectuate an attack. They come with embedded public keys, mostly RSA, with which they encrypt the public key of the sub-pair of the locally generated RSA keys after it encrypts the AES key which actually encrypted the original user data.

However, the tendency to generate an RSA key pair on the targeted host has brought the possibility of fetching off from running memory the seeds used in such key generations. The figure above in figure 13 shows the successful decryption of WannaCry victim files using Wanakiwi [21] after having fetched from memory the primes used in RSA sub-key pair generation.

It's worth noting however that the above method of data recovery only works if the memory allocated to the ransomware process is not flushed or overwritten. That is to say that the victim shouldn't restart his machine among other things, which is quite the contrary in the event of an unexpected attack.

VI. Recommendations and Best Practices

Ransomware seems not to be leaving us anytime soon. If anything, with the advent of Ransomware as a Service (RaaS) [22], its activities are only expected to be on the rise and users should brace themselves against such attacks. Ransomware is best prevented than cured. That is to imply that security measures against ransomware ought to be proactive and not reactive. A reactive approach will always force the victim to seek recovery efforts which are not 100% percent guaranteed. Mitigating ransomware requires a different approach as opposed to that when dealing with conventional malwares because subtle errors can lead to indefinite loss of data. Offline backup is strongly recommended because new malware strains such as WannaCry and NotPetya introduce worm capabilities enabling them to traverse a compromised network and attack any online backup on the network. Users should always keep their systems updated and have their security vulnerabilities patched because new malware versions capitalize on such loopholes. Considering the diversity of attack and infection vectors, backups alone aren't enough. Network and local security should be implemented which should not allow unsanctioned privilege escalation since it's noticed that deletion of volume shadow copies requires privileged access. In the same manner, since ransomware on average seeks to delete volume shadow copies, offline backup of volume shadow copies is highly recommended since it's possible to restore a system from offline backed up volume shadow copies. Last but not the least, user training and awareness should always be factored in when implementing a security plan. It's the benign actions of users that lead to ransomware attacks because on average, ransomware requires some form of user action. This could be as simple as opening an attachment, visiting a webpage harboring an exploit kit, running a macro in an office application or simply opening up a PDF document.

VII. Conclusion

The approach used to evaluate data deletion attacks by ransomware, as presented in this paper, shows that data recovery after a ransomware attack is possible depending on the attack structure. Ideal attack structures have shown that public keys ought to be generated on the C2 server and later propagated to the ransomware upon successful beaconing after an attack. This, coupled with secure data deletion techniques would make data recovery almost impossible. However, we observed that this is not the

attack structure implemented by the majority of ransomware families, especially the recent ones, owing to the complexity of interlinked factors. The other attack structures are prone to full data recovery the degree of which is dictated by the data deletion methodology implemented by the ransomware. Though almost all ransomware variants seem to seek to delete volume shadow copies, offline backup of shadow copies before ransomware attacks and their subsequent restoration would restore data to the same level of the backup. Though almost all ransomware employs some form of sandbox evasion techniques directed against dynamic and static analysis, it is still possible to restore ransomed data especially in with the current methodology of generating public key pairs using the victimized host. From our study, we are led to conclude that no matter how devastating a ransomware attack might appear, the key to data recovery options lies in the underlying attack structure and the implemented data deletion methodology.

REFERENCES

[1] M. Belkadi, R. Aoudjit, M. Daoui, and M. Lalam. "Energy-efficient secure directed diffusion protocol for wireless sensor networks." International Journal of Information Technology and Computer Science (IJITCS) 6, no. 1 (2013): pp.50.

[2] F. Nadeem. "A Taxonomy of Data Management Models in Distributed and Grid Environments." International Journal of Information Technology and Computer Science (IJITCS) 8, no. 3 (2016): pp.19.

[3] C. Sheth and R. Thakker. "Performance evaluation and comparison of network firewalls under DDoS attack." International Journal of Computer Network and Information Security (IJCNIS) Vol.5, Iss. No. 12 (2013): pp.60-67..

[4] Hilarie Orman. "Evil Offspring-Ransomware and Crypto Technology." IEEE Internet Computing 20, no. 5 (2016): pp.89-94.

[5] Symantec Security Response. "An ISTR Special Report: Ransomware and Businesses 2016." [Online] Available:http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/ISTR2016_Ransomware_and_Businesses.pdf [Accessed 24th July 2017]

[6] A.L. Young, and M. Yung. "Cryptovirology: The birth, neglect, and explosion of ransomware." Communications of the ACM 60, no. 7 (2017): pp.24-26.

[7] F. Lombardi, and R. Di Pietro. "Heterogeneous Architectures: Malware and Countermeasures." In Secure System Design and Trustable Computing, pp. 421-438. Springer International Publishing, 2016.

[8] M. Carpenter, T. Liston, and E. Skoudis. "Hiding virtualization from attackers and malware." IEEE Security & Privacy 5, no. 3 (2007).

[9] D. Emm. "Cracking the code: The history of Gpcode." Computer Fraud & Security 2008, no. 9 (2008): 15-17.

[10] E. Fujisaki, and T. Okamoto. "Secure integration of asymmetric and symmetric encryption schemes." In Crypto, vol. 99, no. 32, pp. 537-554. 1999.

[11] K. Cabaj, P. Gawkowski, K. Grochowski, and D. Osojca. "Network activity analysis of CryptoWall ransomware." Przeglad Elektrotechniczny 91, no. 11 (2015): 201-204.

[12] "French researchers find way to unlock “WannaCry” without ransom" (2017). Available [Online] Read more at: https://www.vanguardngr.com/2017/05/french-researchers-find-way-unlock-wannacry-without-ransom/" [19th May, 2019]

[13] R. Brewer. "Ransomware attacks: detection, prevention and cure." Network Security 2016, no. 9 (2016): 5-9. Elsevier Publishing.

[14] "How to Remove Crypt888 Ransomware" (2016). Available [Online]: http://botcrawl.com/how-to-remove-crypt888-ransomware/ [7th December, 2016]

[15] M. Weckstén, J. Frick, A. Sjöström, and E. Järpe. "A novel method for recovery from Crypto Ransomware infections." In Computer and Communications (ICCC), 2016 2nd IEEE International Conference on, pp. 1354-1358. IEEE, 2016.

[16] A. Zimba, Z.Wang, and H. Chen "Reasoning crypto ransomware infection vectors with Bayesian networks." In Intelligence and Security Informatics (ISI), 2017 IEEE International Conference on, pp. 149-151. IEEE, 2017.

[17] C. Rossow et al. "Prudent practices for designing malware experiments: Status quo and outlook." Security and Privacy (SP), 2012 IEEE Symposium on. IEEE, 2012.

[18] "Locky, Cryptowall and Cerber account for '90 per cent of ransomware attacks" (2017). Available [Online]: https://www.theinquirer.net/inquirer/news/3005154/locky-cryptowall-and-cerber-account-for-90-per-cent-of-ransomware-attacks [22nd February 2017]

[19] "Today's most popular operating systems" (2017). Available [Online]: http://www.zdnet.com/article/todays-most-popular-operating-systems/ [9th January, 2017]

[20] "Five Lessons Learned from Recent Cyber Attacks." (2017). IEEE Innovation at Work. Available [Online]: [Accessed 3rd September, 2017]

[21] A. Zimba, L. Simukonda, M. Chishimba. "Demystifying Ransomware Attacks: Reverse Engineering and Dynamic Malware Analysis of WannaCry for Network and Information Security." In Information and Communications Technologies (ICICT), 2017 IEEE International Conference. IEEE, 2017.

[22] D.S. Wall. "Dis-organised crime: Towards a distributed model of the organization of cybercrime." (2015).

[23] MM Ahmadian, HR Shahriari, and SM Ghaffarian. "Connection-monitor & connection-breaker: A novel approach for prevention and detection of high survivable ransomwares." In Information Security and Cryptology (ISCISC), 2015 12th International Iranian Society of Cryptology Conference on, pp. 79-84. IEEE, 2015.

[24] N Scaife,H Carter, P Traynor, and KRB Butler. "Cryptolock (and drop it): stopping ransomware attacks on user data." In Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on, pp. 303-312. IEEE, 2016.

[25] Kirda, E., 2015. Most Ransomware Isn’t As Complex As You Might Think Yes, we should be able to detect most of it. DIMVA.

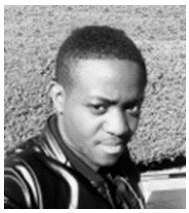
[26] F Mercaldo,, V Nardone, and A Santone. "Ransomware Inside Out." In Availability, Reliability and Security (ARES), 2016 11th International Conference on, pp. 628-637. IEEE, 2016.

[27] K Cabaj and W Mazurczyk. "Using software-defined networking for ransomware mitigation: the case of cryptowall." IEEE Network 30, no. 6 (2016): 14-20.

Authors' Profiles

Aaron Zimba is currently a PhD candidate at the University of Science and Technology Beijing in the Department of Computer Science and Technology. He received his Master and Bachelor of Science degree from the St Petersburg Electrotechnical University in St Petersburg in 2009 and 2007 respectively. He is also a member of the IEEE. His main research interests include Network and Information Security, Cloud Computing Security and Network Security Models.

Zhaoshun Wang is a Professor and the Associate Head of the Department of Computer Science and Technology at the University of Science and Technology Beijing. He graduated from Department of Mathematics at Beijing Normal University in 1993. He received his PhD from Beijing University of Science and Technology in 2002. He completed postdoctoral research work at the Graduate School of the Chinese Academy of Sciences in 2006. He holds patents and has many awards to his name. His main research areas include Information Security, Computer Architecture and Software Engineering.



Luckson Simukonda is currently pursuing a masters degree at Gdańsk University of Technology studying Control Engineering and Robotics in the Faculty Of Electronics, Telecommunications and Informatics. He holds a Bachelors Degree in Computer Science from Mulungushi University which was awarded to him in the year 2015. His research interests are Artificial Intelligence and Robotics and Computer Networks and Security.

How to cite this paper: Aaron Zimba, Zhaoshun Wang, Luckson Simukonda, "Towards Data Resilience: The Analytical Case of Crypto Ransomware Data Recovery Techniques", International Journal of Information Technology and Computer Science(IJITCS), Vol.10, No.1, pp.40-51, 2018. DOI: 10.5815/ijitcs.2018.01.05

