computers

MDPI

Article

In-Depth Analysis of Ransom Note Files

Yassine Lemmou [1,*,†] and Jean-Louis Lanet [2,†] and El Mamoun Souidi [1,†]

[1] Faculty of Sciences, Mohammed V University in Rabat, LabMIASI, BP 1014 RP, Rabat 10000, Morocco; emsouidi@gmail.com
[2] INRIA, LHS-PEC, 35042 Rennes, France; jean.louis.lanet@free.fr
* Correspondence: yassine.lemmou@gmail.com
† These authors contributed equally to this work.

Abstract: During recent years, many papers have been published on ransomware, but to the best of our knowledge, no previous academic studies have been conducted on ransom note files. In this paper, we present the results of a depth study on filenames and the content of ransom files. We propose a prototype to identify the ransom files. Then we explore how the filenames and the content of these files can minimize the risk of ransomware encryption of some specified ransomware or increase the effectiveness of some ransomware detection tools. To achieve these objectives, two approaches are discussed in this paper. The first uses Latent Semantic Analysis (LSA) to check similarities between the contents of files. The second uses some Machine Learning models to classify the filenames into two classes—ransom filenames and benign filenames.

Keywords: ransomware; ransom note file; detection; identification; Latent Semantic Analysis; Machine Learning

Citation: Lemmou, Y.; Lanet, J.-L.; Souidi, E.M. In-Depth Analysis of Ransom Note Files. *Computers* **2021**, *10*, 145. https://doi.org/10.3390/computers10110145

Academic Editor: Paolo Bellavista

Received: 22 September 2021
Accepted: 29 October 2021
Published: 8 November 2021

1. Introduction

Ransomware has spread rapidly over the last five years causing significant damage, especially in Windows environments. This category of malware covers two classes—Locker ransomware and Crypto ransomware. The first denies or blocks access to files. The second encrypts important data on a target machine, including files and backups. Generally, in both classes, a ransom to pay is demanded. This paper examines only the second class because it is more spread than the first class and the ransom files are more related to Crypto-ransomware than to Locker-ransomware.

Ransom note files are created by ransomware, then added in several directories on the target machine to notify the victims that their files are encrypted. They provide payment information, show how to send payment, the amount requested, and what are the consequences if if you do not pay. These files are the single links between the ransomware developers and their victims. The behavior of adding the same ransom files in directories is repeated recursively in each target directory. Only a few papers like [1–3] have mentioned the use of the ransom files to detect or identify ransomware. For example, Endgame team [1] developed a classifier based on the Naïve Bayes model to classify the ransom files. The whitepaper of MWR labs [2] presented some behaviors to detect ransomware. Among the proposed behaviors in this whitepaper is the use of a bag of most used terms in the ransom files. While this idea of monitoring the added ransom files can be effective, it can take time if it is based only on reading the whole content of any created file on the machine, searching for ransom content, then deciding if this file is truly a ransom file or not. Therefore, we found that monitoring the filenames of the created files, searching for ransom filenames, then checking the content of any file named like a ransom filename can help to spot ransomware or minimize its damage. Indeed, as shown in the following sections, the used terms in the filenames or the content of the ransom files are limited to a few terms.

The likelihood of spotting a benign file named like the filename of a ransom file is small in a computer. Furthermore, the false positive files described in Section 4.2 (the `readme` or `help` files) are not created recursively in each directory compared to the ransom files which in several cases are recursively duplicated. Finally, any solution to detect ransomware must contain several indicators based on different ransomware behaviors. In other words, this proposition cannot detect all ransomware and cannot be used alone in a ransomware detector. Indeed, many ransomware families do not add the ransom files in every target directory, some families encode the content of their ransom files and others add only one ransom file at the end of their infection (e.g., `Spora` ransomware [4]). For these reasons, we suggest that using the ransom files must be monitored with a corpus of ransomware behaviors. In summary, this paper presents the following contributions:

- We make the first and probably unique available collection of ransom files. This collection contains more than 170 ransom files of 62 different ransomware families. This collection is shared in `GitHub` (https://github.com/lemmou/RansomNoteFiles, accessed on 15 September 2021) for evaluation and study by the research community. It can also be used in the area of criminology and profiling ransomware developers. These ransom files were collected manually from different sources:
  - From *Malware Traffic Analysis* Project (https://www.malware-traffic-analysis.net/, accessed on 15 September 2021);
  - From *Hybrid-Analysis*, looking for the ransom note filenames (https://hybrid-analysis.com/, accessed on 15 September 2021);
  - By running some ransomware samples in our virtual machines;
  - From *The Week in Ransomware* of BleepingComputer (https://www.bleepingcomputer.com/, accessed on 15 September 2021);
  - From the *Pastebins* of the owner of ID-Ransomware project (https://pastebin.com/u/Demonslay335, accessed on 15 September 2021).
- We present a depth analysis of the ransom files including some statistics based on their filenames and contents;
- We propose a prototype to identify the ransomware family associated with a given ransom file;
- We apply an approach to classify the ransom files and benign files and another approach to check their similarities using some models of Machine Learning and Latent Semantic Analysis (LSA);
- The results of the previous items allow us to propose a new approach to detecting ransomware using the filenames and the content of the ransom files. This approach can be used with other behaviors to make a ransomware detector or added to the currently available solutions of ransomware detection, identification or prevention. Moreover, several ransomware families can be detected or identified using only this approach before their damage. In particular, the ones that add their ransom files before encryption;
- We compare the effectiveness of our approach on some ransomware detection tools at the end of this paper.

Table A1 in Appendix A (All theses families are crypto ransomware despite the `Locker` tag in the names of some ransomware) shows the used ransom files in this paper. This paper is structured as follows: ransom files description and related works are presented in Section 2. Section 3 describes how the ransom files can be identified and the proposed prototype to identify these files. The results of our study on the filenames of the ransom files are outlined in Section 4. The same study of Section 4 is applied to the content of the ransom files in Section 5. Finally, the conclusion is drawn in Section 6.

2. Ransom Note Files and Related Works

The ransom files are a unique way for the victims to take the first steps to restore their encrypted files. The ransom notes provide information to the victims such as the payment

information, fee, deadline, free files decryption, and various instructions to pay the ransom, then recover the encrypted files. The collected dataset of the ransom files includes the ransom notes of different known families like `Locky`, `Cerber`, and `TeslaCrypt`. Table A1 in the Appendix A shows the collected ransom files. The graph in Figure 1 shows the number of families for each extension of the ransom files. The most used extension is `txt` followed by `htm` and `html` (`htm(l)`). More than 30 families use `txt` or `htm(l)` files as a ransom note. Other ransomware like `Blind`, `Mr.Dec` and `NemucodAES` ransomware use `hta` files. `Matrix` ransomware family is the single family in our collection that uses `rtf` files. Generally, few ransomwares, such as `Qinynore` and `Cypher`, choose `rtf` files for their ransom files.

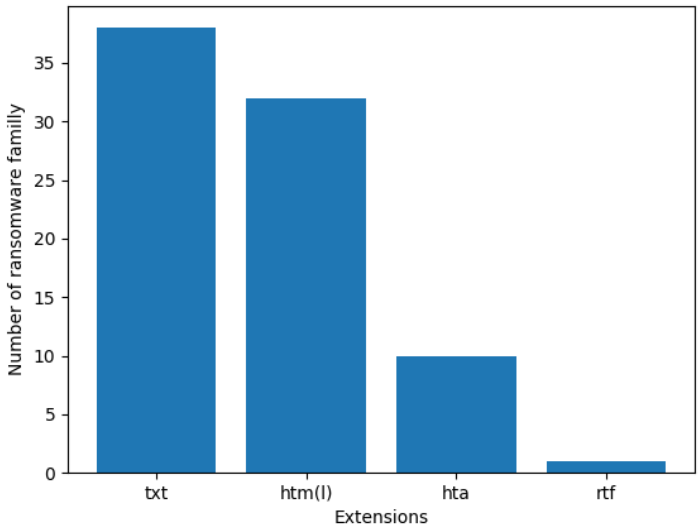

Figure 1. Number of Ransomware Families for Each Extension.

Ransomware families change the extensions and filenames of their ransom files between the versions. Other families add their names, the extension of encrypted files or the victim identity (`ID`) to the filenames. In another way, some ransomware families like `Dharma`, `Jaff` or `HC7` keep the same filename and extension of their ransom files in all their versions. Generally, the extensions of the ransom notes are limited to few extensions. Moreover, despite the changes in the filenames between the versions of the same ransomware, we prove in Section 4.1 that the used terms in the filenames are also limited to few terms.

`WannaCry` is a well-known ransomware that spread in May 2017 by exploiting a vulnerability in the SMB protocol. It infected more than 200,000 machines in the world [5]. This Crypto worm copies the content of the file `r.wnry` into a file named `@Please_Read_Me@.txt` before encrypting the files. Lemmou et al. [6] found that three files (`txt`, `html` and `url`) with the same filename `!_HOW_TO_RESTORE_[extension]` were replicated by `PrincessLocker` ransomware in target directories before encryption. Moreover, by exploring some recent ransomware, we found that many ransomware families like `JSWORM`, `ChaCha`, `StopDjvu`, `LockerGoga` and `GlobImposter` add their ransom files before encryption. Some versions of `TeslaCrypt`, `CryptoLocker`, and `Cerber` add their ransom files in each target directory after encrypting only the content of this directory. The filenames of all these ransom notes can be easily identified. For example, `JSWORM-DECRYPT.hta` of `JSWORM` ransomware, `README_LOCKED.txt` of `LockerGoga` or `DECRYPT-FILES.html` of `ChaCha` ransomware. For these reasons, using the ransom files to detect the ransomware can:

- Minimize the ransomware damage to few encrypted files if the ransomware encrypts the files of a target directory then puts the ransom files in this directory;
- Detect the ransomware without encrypting any file if the ransomware creates its ransom files before encryption.

Over recent years, the frequency of ransomware activities and the number of new ransomware families have grown since 2015. Simultaneously, many ransomware analysis,

detection, and prevention approaches have been published to detect ransomware. The first ransomware analysis was published by Gazet in 2010 [7]. He presented a comparative analysis of four ransomware families. More similar analyses have been published during the last four years. For example, the works of D. Caivano et al. [8] on the common ransomware characteristics of 76 samples and the works of Lemmou et al. [9] that extracted the ransomware behaviors of more than 20 ransomware families. In addition, in their paper they discussed some behaviors that were automatically searched by analyzing more than 200 different ransomware collected during 2019.

On ransomware detection, different approaches have been proposed. For example, monitoring abnormal file system activities to detect ransomware was proposed by Kharraz et al. [10] in 2015. Their work was the first attempt to detect the ransomware and their analysis of several ransomware families allowed them one year later to propose a monitoring tool called `UNVEIL` which detects Crypto-ransomware. `ShieldFS` [11] is a ransomware detection tool based on the difference between file system I/O requests of ransomware and benign software. `DaD` [12] is a tool to detect ransomware based on monitoring file system activities of all userland threads. `CryptoDrop` [3] another ransomware detection tool, which alerts the user during suspicious activities using a set of ransomware behaviors. Until now, `UNVEIL` and `ShieldFS` were not released for download, but `CryptoDrop` and `DaD` were available for download until 2019.

Other approaches have been proposed to detect ransomware such as using decoy files or monitoring the API crypto. The first approach is based on monitoring some files named decoys in some directories. Any write access to these files alerts the monitoring tool. This approach was described by Jeonghwan et al. [13] and it has been used by several detection tools like `Antiransom V3, Padvish` and `Cybereason RansomFree`. Lemmou et al. [9] discussed how ensuring maximum efficiency of using decoy files to detect ransomware. Generally, using decoy files to detect ransomware is an interesting idea, but it must be used with other monitored ransomware behaviors. On monitoring the API Crypto, Palisse et al. [14] introduced two countermeasures to decrypt files based on the interception of the API Crypto calls and weak ransomware operations. In the same way, Eugene et al. proposed `PayBreak` [15] a tool that hooks the encryption functions of standard libraries and securely stores encryption keys in a key vault for a future decryption of the encrypted files by ransomware.

Traditional ransomware detection techniques are included in the current malware detection and are primarily based on signatures that malware developers can easily evade. Machine Learning models have good capabilities to detect ransomware. Bello et al. [16] presented a comprehensive survey on the detection of ransomware attacks using Machine Learning algorithms. Their study analyzed literature from different perspectives focusing on Machine Learning algorithms to detect ransomware. An example to detect ransomware using Machine Learning was proposed by Poudyal et al. [17]. They designed a ransomware detection prototype named `AIRaD` for AI-based Ransomware Detection. Their tool is based on a deep inspection approach for multi-level profiling of crypto ransomware, which captures the distinct features at Dynamic Link Library, function calls, and assembly levels. `PEDA` [18] is a pre-encryption detection algorithm for detecting crypto-ransomware proposed by Kok et al. `PEDA` uses two levels of detection. The first is a signature based detection to detect known ransomware. The second is a Machine learning model based on the API calls to detect known and unknown ransomware. `DNAact-Ran` [19] is another ransomware detection solution using Machine Learning. `DNAact-Ran` uses Digital DNA sequencing design constraints and k-mer frequency vector. This solution was evaluated on 582 ransomware and 942 benign applications and it showed its effectiveness to detect ransomware compared with other methods. Ketzaki et al. [20] proposed a detection procedure based on neural network methodologies to detect ransomware. The used features by the neural network model are extracted from monitoring in real-time the CPU, the memory, the disk space, the rate of reads and writes, the number of changed, created and deleted files.

The current paper offers a complementary to these works. For example, Scaif et al. [3] use three primary and two secondary indicators to detect the ransomware in their tool `CryptoDrop`. Monitoring the ransom files can be added as a secondary indicator among the other indicators to reach the threshold score of detection in `CryptoDrop`. Some ransomware add their ransom files during exploring directories before encryption. However, our idea to detect the ransomware using the ransom files can also be combined with monitoring the per-thread file system traversal suggested in [21]. Within Machine Learning, the ransom files can be used as a monitored feature in [20] like rate of reads/writes and the number of changed, created and deleted files. Related to monitoring the API Crypto or decoy files, a created ransom file by a specified process can confirm that this process is a ransomware if also uses the API Crypto. The same is true if this process accesses to a decoy file.

3. Identification of Ransom Note Files

In this section, we present the extracted markers that can be used to identify the ransom files and associate them with their related ransomware.

3.1. Addresses

Ransomware developers share their addresses (URL, Bitcoin, Email or others) to receive the ransom. The most used addresses are the URL and the email addresses. In our collection of ransom files, we identified four categories of addresses: Email addresses, Bitmessage addresses, Web payment addresses and Bitcoin addresses.

3.1.1. Email Addresses

Several ransomware families notify their victims to use the cited email addresses in the ransom files to receive more instructions about the decryption of the files. Generally, the ransomware developers do not use a single address for all their ransomware versions. The email address can be used to identify the current or the previous versions of the ransomware family and cannot generally identify or predict the next versions of the same family. The collected dataset of ransom files, contains 43 ransomware families that used the email addresses. Table 1 shows the used addresses by some families. `Dharma` family used 14 email addresses for 10 versions and these email addresses were the single way to communicate with the developers of `Dharma`.

Table 1. Email Addresses used by some Ransomware Families.

Family	Version	Email Address
Argus	-	`argusdecrypt@cock.li` `argusdecrypt@mailfence.com`
BTCWare	Aleta	`chines34@protonmail.ch`
	Gryphon	`oceannew_vb@protonmail.com`
	Payday	`checkzip@india.com`
Dharma	abido	`abibo@protonmail.com`
	arrow	`jamie_white25@aol.com, dot_faldo@aol.com`
	Bip	`emailpeekabooo@qq.com, peekabooo@qq.com`
	bkp	`bkp@cock.li`
	brrr	`paydecryption@qq.com`
	cmb	`paymentbtc@firemail.cc`
	manpecman	`manpecamet1974@aol.com, raxisubsro1977@aol.com`
	monro	`icrypt@cock.li`
	skynet	`skynet45@cock.li, skynet45@tutanota.com`
	stopencrypt	`stopencrypt@qq.com`

Some ransomware, such as `Triplem` and `Argus` ransomware, added their names to the email addresses, which can allow us to associate the ransom file with its ransomware.

Therefore, any ransom file identifier must search the ransomware names in the email address to identify the ransom note file.

3.1.2. Bitcoin Addresses

The second method to identify the ransom files is using Bitcoin addresses. `CryptoLocker` was the first ransomware that asked for payment using Bitcoin (Emerging Technology: True scale of Bitcoin ransomware extortion revealed, https://www.technologyreview.com/s/610803/true-scale-of-bitcoin-ransomware-extortion-revealed/, accessed on 15 September 2021). Before `CryptoLocker`, the earlier ransomware used other payment methods like pre-paid money vouchers, premium SMS, GoldMoney or buying from a site.

The ransomware developers add their Bitcoin addresses in the ransom files to receive the Bitcoin payments from the victims. This address is a `base58` encoded identifier of 26 to 35 alphanumeric characters starting with 1 or 3 [22]. Our dataset contains ransomware that uses the Bitcoin addresses to receive the payment. For example, `BadBlock`, `TripleM` used two different Bitcoin addresses in two versions and `Cerber` the same Bitcoin address five versions. Generally, the Bitcoin address is used in many versions of the same ransomware. Table 2 shows the Bitcoin addresses of some ransomware families.

Table 2. Bitcoin Addresses of some Ransomware Families.

Family	Bitmessage Address
BadBlock	`19zvmsm7qsqgfxcckxbjstdvdbt99zuwbp`
Comonransomware	`35m1zjhtati4iduufzena75ibyjoq9ibgf`
Cryptolocker	`1LfX1pFa2uSH6HDfH47zRDZgre4Ms7uZTk`
Diamond	`1L6PpSehR8V7YsZTc3L3F5RwbWoNma1nno`

3.1.3. Bitmessage Address

Another method used by ransomware developers to communicate with their victims is `Bitmessage` (Bitmessage:A PeertoPeer Message Authentication and Delivery System, https://bitmessage.org/bitmessage.pdf, accessed on 15 September 2021). It is a peer-to-peer and decentralized communication protocol used by users to communicate anonymously with other users. This method is used only by few ransomware. This address of 32 to 34 characters is a `Base58` encoded public key hash that starts with the term `BM-`. `Scarab` ransomware family is one of the seven families in our dataset that used this method to communicate with the victims. It used the same address in the version seen in November 2017 and the version seen in October 2018. Table 3 shows three families of the seven families that used the Bitmessage addresses in their ransom files.

Table 3. Bitmessage Addresses Used by Three Ransomware Families.

Family	Version	Bitmessage Address
Amnesia	-	`BM-NBdUQmYVn43e3nK4amuoeaSm4ZStr8oZ`
CryptoBit	-	`BM-NAxZ29ouecw2Y7ibaXKus1vxDRDfheW6`
Crypton	-	`BM-2cwzhonfbjq3x8puliwsykhc6dedq54zq1`

3.1.4. Web Payment Address

Other ransomware creates an onion web page to give the victims more instructions to recover the encrypted files. The onion address is mentioned in the ransom file with other instructions about how to access the onion page. Using the most used terms in the web payment addresses like `onion, torstorm, tor2web, onion2web, top` and others, we extracted at least one onion address correctly from all the 28 ransomware families that added an onion address in their ransom files. Each extracted address identified correctly the ransomware family associated with the ransom file. Generally, the ransomware does not keep the same onion address during all their versions. Therefore, the extracted onion URLs can be used only to identify the ransom files that are currently or were previously

saved by the ransom note identifier. However, in some cases, they can be used to identify the ransom files of the next versions. Table 4 shows some ransomware families that used the same onion address in several versions.

Ransomware Tracker is an online service that tracks the IP addresses, domain names, and URLs including C&C servers, botnet and payment sites that are linked to a specified ransomware. We used the Python API of Ransomware Tracker available in GitHub (PaulWebSec: Python API for Ransomware Tracker. https://github.com/PaulSec/ransomware-tracker, accessed on 15 September 2021) to track the extracted URLs from the collected ransom files. We were able to track only the used URLs in the ransom files of two versions of Locky ransomware (Odin and Zepto). The ransom note identifier can use this service to identify the extracted and unidentified URLs from the ransom files, then track these addresses to associate them with a specified ransomware.

Table 4. Onion Addresses of some Ransomware Families.

Family	Versions	Onion address
Cerber	2017-01-05, 2017-01-26 2017-03-15, 2017-05-12	p27dokhpz2n7nvgr.onion
CryptoWall	2016-01-17, 2016-02-05	3wzn5p2yiumh7akj.onion
GandCrab	v1 and v2 v5.0.1, v5.0.2, v5.0.3, v5.0.4	gdcbghvjyqy7jclk.onion.top gandcrabmfe6mnef.onion

3.2. Keyword

Some ransomware families add their names and some specified keywords like the extension of the encrypted files in the content of their ransom files. Any ransom note file identifier must search for ransomware names and other specified keywords in the content of these files. The names of the ransomware can be obfuscated in the content of the ransom files. For example, some versions of Cerber seen in May 2017 used the word Cerber instead of Cerber in their hta ransom files. The same ransomware used the term crbr in the hta and the txt ransom files of its version seen in August 2017. In another way, some ransomware like CryptoShield ransomware add the name of other ransomware in the content. For these reasons:

- The identifier must be aware of the encoded ransomware names. We suggest storing the previously encoded names by the ransomware in the ransom note identifier database;
- Take the most cited ransomware names in the ransom file or the first cited name in the case of equality between many cited ransomware names in the content;
- The identifier can search for the used extensions of the encrypted files in the content. Indeed, some ransomware adds these extensions to the content of their ransom files. For example, Blind ransomware added the extension NAPOLEON and BTCWare ransomware added the extension Gryphon.

Using the previous items, we tried to identify some ransom files using only the cited ransomware names in the content. We were able to identify correctly 29 families from 31 families that added their names in their ransom files with two false predictions:

- The version of TeslaCrypt that was seen in July 2015. Our script identified this version as a ransom file of CryptoWall ransomware. The reason for this false prediction is that the ransom file of this version is an exact copy of the ransom file of CryptoWall 3.0 ransomware;
- An unknown version of CrypMic. Our script identified this version as a ransom file of Alpha ransomware. This is due to the injected terms alpha in the content.

ID-Ransomware does not identify the ransom file of TeslaCrypt. It also cannot identify the ransom file of CrypMic correctly. Moreover, ID-Ransomware fails to identify correctly the ransom files of 29 families that add the name of the ransomware in the content.

For these reasons, we suggest that the ID-Ransomware service adapts its process to identify the ransom files by searching for the ransomware names in the content to improve its efficiency in identifying the ransom files.

3.3. Names and Content of the Ransom Note Files

Several ransomwares name their ransom files with some specified names or a specified combination of terms that make the identification of the ransom files possible using the filenames. For example, `Argus, Chip` and `GandCrab` ransomware added their names in the filenames of their ransom files. The ransom note identifier can search for the names of the known ransomware in the filenames of a given ransom file to identify it. The filenames of the ransom files of some ransomware contain the added extension to the encrypted files. For example, the `hta` ransom file of the `payday` version of `BTCWare` ransomware is `Payday.hta` and the `htm` ransom file of the `asasin` version of `Locky` ransomware is `asasin.htm`. For this reason, we suggest that the ransom note identifier searches for some used extensions by the ransomware in the filenames. As described in Section 4.2, the filenames of the ransom files cannot classify the ransomware effectively into families because multiple ransomware families change the filenames of their ransom notes in each version. Despite this weakness, we think that the ransom note identifier can check the similarity of a given ransom note filename and the other ransom filenames in its database as a secondary indicator to identify the ransom files.

Moreover, another method to identify the ransom files is using their content. Indeed, many ransomwares keep all or the same parts of the content of their ransom files for several versions.

3.4. Ransom Note Files Identifier

To the best of our knowledge, ID-Ransomware (https://id-ransomware.malwarehunterteam.com/, accessed on 15 September 2021) is the unique service for the victims and researchers to identify the ransom files. It is a free service to identify which ransomware encrypted a submitted file to the service or the ransomware associated with a given ransom file. The service is able to identify many known ransomware families like `Cerber`, `TeslaCrypt` and the recent families like `StopDjvu` or `REvil`. ID-Ransomware identifies the ransomware families using extensions, patterns, filenames, email, Bitcoin addresses and others.

The contributions of this paper include some suggestions to make a ransomware identifier or to improve the effectiveness of ID-Ransomware. In some cases, this service fails to identify the associated ransomware with a given ransom file. Until now, ID-Ransomware does not use the content of the ransom file to identify the ransom files. The responsible of this service mentioned in a private conversation with us that using the content to identify the ransom files is a future work for the owners of this service. Thus, we suggest using Latent Semantic Analysis (LSA) as a solution to check similarities between the content of the ransom files and the mentioned markers above to increase the effectiveness of this service.

We developed some Python scripts to evaluate our ideas to identify the ransom files in our dataset of 71 ransomware families. We get the following results:

- Using only the email addresses: 65 ransom files were correctly identified. ID-Ransomware failed to identify seven ransom files;
- Using only the Bitmessage addresses: eight ransom files were correctly identified. ID-Ransomware identified correctly only five ransom files and the others were identified using the email addresses embedded in the content;
- Using only the URL addresses: 102 ransom files were correctly identified. ID-Ransomware failed to identify 18 ransom files and 33 ransom files were identified using other markers like the Bitcoin addresses and some custom rules;

- Using only the Bitcoin addresses: 25 ransom files were correctly identified. ID-Ransomware identified 14 ransom files, it failed to identify three ransom files and eight ransom files were identified using other markers;
- Using only the ransomware tracker: two ransom files of one family were identified. ID-Ransomware was able to identify more ransom files using the ransomware tracker. We think that our script to identify the ransom files using the ransomware tracker must be improved to identify the ransom files;
- Using only LSA on the content of the ransom files (with 0.99995 as a threshold of similarity between the ransom files), only 13 ransom files (7% of 182 ransom files) were not correctly identified. Table 5 shows these false positives. LSA matched the ransom files of different versions of the same ransomware family. For example, the `txt` ransom file of the version seen during May 2016 of `Cerber` has high similarity with the ransom file of the version seen during August 2016. The `hta` ransom file of the `abido` version of `Dharma` has high similarity with all the `hta` ransom files of the other versions. Moreover, we tested LSA on the content of the ransom files of some new versions have been seen during 2021 of `Dharma` and `GlobImposter`. We found high similarity between the content of these ransom files and the content of their old ransom files.

Table 5. False Positives Labels Using LSA on the Content.

Family	Version	Ransom Note	Result by LSA (Label)
Cryptowall	3.0	`HELP_DECRYPT.txt`	TeslaCrypt (2 false ransom notes)
CryptoLocker	-	`HELP_RESTORE_FILES.txt`	TeslaCrypt & Alpha Crypt
Alpha Crypt	2015-04-30	`HELP_TO_SAVE_FILES.txt`	CryptoLocker & TeslaCrypt
TeslaCrypt	2015-04-03	`HELP_RESTORE_FILES.txt`	Alpha Crypt & CryptoLocker
	2015-10-23	`howto_recover_file_[].txt`	CryptoWall
	V2.1	`HOWTO_RESTORE_FILES.txt`	CryptoWall
CryptoMix	2016-11-28	-	Cryptfile2
Cryptfile2	-	-	CryptoMix
CrypMIC	-	`README.txt`	Cryptxxx (2 false ransom notes)
Rapid	-	`_READ_ME_FOR_DECRYPT.txt`	StorageCrypt
Cryptxxx	2016-05-05	`de_crypt_readme.html`	CrypMic
		`de_crypt_readme.txt`	CrypMic
StorageCrypt	-	-	Rapid
		13 ransom notes files	**18 FP labels**

ID-Ransomware searches for many markers to identify the ransom files without searching for correlations between the used phrases in the content or analyzing the content entirely. We tested the 182 ransom files in ID-Ransomware. It failed to identify or correctly identify 24 ransom files. The combination of our scripts (without checking the filenames of the ransom files) was able to identify 181 ransom files, including the unidentified ransom files by ID-Ransomware. Only one ransom file could not be identified using our scripts. It is the `txt` ransom file of `Kee` ransomware. This ransom file does not have high LSA similarity on the content (similarity ≥ 0.99995) with the other ransom files. Related to these results, we suggest that the ransom files identifier must combine and check all the discussed markers in this section to identify the ransom files, including the content and filenames of the ransom files. Indeed, the `Kee` ransomware can be identified by checking the filename

of its ransom file (`Hello There! Fellow @kee use!.txt`) that contains its name. For these reasons, we propose that ID-Ransomware or any new ransom note identifier service analyses also the filenames and the content of the ransom files using for example LSA and following this scenario:

1. The ransom note file identifier starts searching, then checking the Bitcoin addresses in the content to associate them with the ransomware that used these addresses using the saved Bitcoin addresses in its database or using any external service. We choose that the ransom note identifier starts searching for Bitcoin addresses in the content because several ransomware families are known by their Bitcoin addresses and they keep them in several versions;
2. The identifier checks the Bitmessage addresses if it is not able to associate the Bitcoin address with any known ransomware or the ransom file does not contain Bitcoin addresses. Generally, the ransomware keeps the Bitmessage addresses for a long period of time like the Bitcoin addresses despite that only few ransomware use them;
3. The identifier searches for any email address in the content. Moreover, the ransomware identifier can search for any known ransomware name in the extracted email addresses to identify the ransom file;
4. In the fourth step, the identifier searches for the URL addresses in the content. If it fails to identify the ransom note from the extracted URLs, it submits them to a ransomware tracker service;
5. If the identifier fails to identify the ransom file using the previous items, it can search for the known ransomware names or any known keyword (extensions, patterns, . . .) in the content. It can use the filename of the ransom file as a secondary indicator to prove its choice between many suggestions;
6. If the identifier fails to identify the ransom file, it uses LSA (or other methods) on the content to search any high similarity with the content of any known ransom file.

As we have shown in this section, the ransom note files can be associated with their ransomware using the shared addresses by the ransomware developers to their victims. The Bitcoin/Bitmessage/Email/URL addresses, the ransomware tracker service, the filenames and contents must be combined correctly to make an efficient ransom file identifier. Generally, the identifier can add other methods, patterns or custom rules to increase its effectiveness.

4. Filenames of the Ransom Files

The filenames of the ransom files of many ransomwares are in uppercase, including some symbols like `#, !, @` or `=`. Some ransomware like `Cryptowall`, `CryptXXX` include the victim `ID`. Others add to the filenames the used extension for the encrypted files. During our research, we found that `Scarab` ransomware has the long ransom note filename: `IF YOU WANT TO GET ALL YOUR FILES BACK, PLEASE READ THIS`.

4.1. Ransom Names Pre-analysis

The focus of this part is to create some statistics on the used terms in the filenames before the semantic analysis in the following parts of this section. The filenames of the ransom notes in our dataset were tokenized to involve the unit of a filename to analyze. Each unit of the filename (named word or term) was standardized, cleaned, removed stop words from the units, then stemmed or lemmatized. These steps are often used in text preparing and pre-processing that precede text analysis of a documents corpus:

1. Tokenization: extension, ransomware name, victim `ID`, unknown or random characters used in the filenames were changed to `[ext]`, `[ransomwarename]`, `[id]`, `[unknown]` or `[random]`. Some filenames such as `_R_E_A_D___T_H_I_S__[random]` of `Cerber` were changed to `_READ___THIS__[random]`. After these modifications, we used each word in the filename as a unit in our analysis;
2. Standardization and Cleaning: the words were converted to lowercase. We do not want `Read me`, `READ ME`, and `READ Me` to be considered as separate terms. The non-

alphanumeric characters were stripped, concatenated terms in the filenames were split (`readme` to `read me`). Some filenames were corrected, for example `Thi$` to `This` or `bl0cked` to `blocked`;

3. Stop word removal: this step is used to drop frequently used words that did not add value to our test. Firstly, we used a predefined stop words list from the Python library `Nltk`, but it removed some interesting words like the interrogative pronouns. For this reason, we made a list containing the words to avoid: `to, me, your, this, if, you, all, my, it, for` and `with`;
4. Stemming or lemmatization: this step tries to transform the evaluated words to their root form [23]. Table 6 displays an example of stemming using `PorterStemmer` and lemmatization using `WordNetLemmatizer`. We applied `PorterStemmer`, `WordNetLemmatizer`, `LancasterStemmer` and `SnowballStemmer` from `Nltk` on the words. The latter gave us the desired results.

Table 6. Example Using `PorterStemmer` and `WordNetLemmatizer`.

	Before	After
Stemming	`restoring files`	`restor file`
Lemmatization	`restoring files`	`restoring file`

Figure 2 presents the most used terms in the filenames of the ransom notes. The word `file` is the most used word, followed by the word `decrypt`. In some filenames, the ransomware developers ask a theoretical question which consists in asking a question that does not wait for an answer from the victims like `how return/restore/back... my files`. The answer is known by the developers and mentioned in the content of the ransom files. Almost 20 samples of 13 families used this question in the filenames. Around 45 ransomwares mentioned the extension of the encrypted files or the `ID` victim in the filenames. Generally, the number of ransomware that adds the extension to the filenames is greater than the number of ransomware that adds the `ID` victim. Other ransomware include the term `please` in the filenames. Including this term in the filenames is only a social norm from the ransomware developers to establish a relationship with the victim to pay the ransom. In summary, the used words in the filenames of the ransom notes are limited to only a few words. These words are not often used by the users in the filenames of their files. For these reasons, we think that the ransom files can be detected during their creations using their filenames. The following part presents some tests to prove the discrimination between the filenames of the ransom files and benign files.

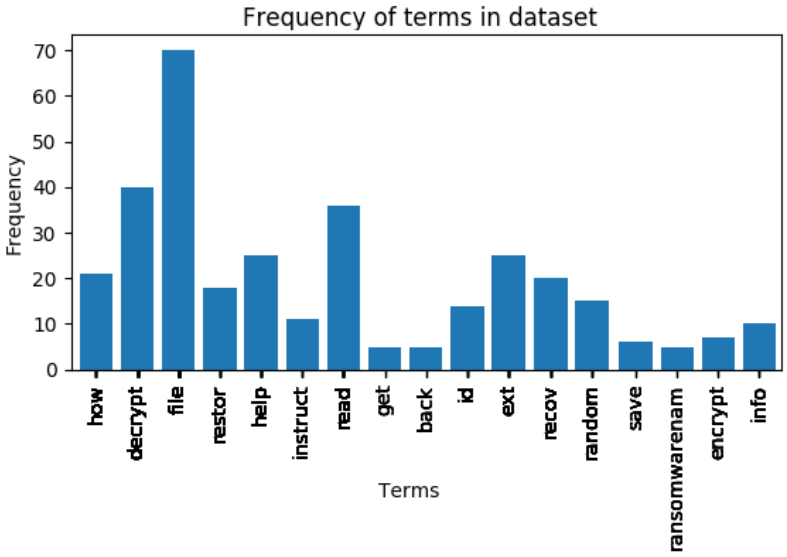

Figure 2. Term Frequency of the Most Used Terms in the Filenames.

4.2. Unsupervised and Classification Analysis on the Ransom Filenames

Unsupervised analysis, such as Latent Semantic Analysis LSA, is different from the classification analysis. The latter is a type of supervised learning model used to automatically classify data (sentences, malware, documents, reviews...) and build predictive models when the true classification of data is known. In contrast, no prior knowledge on the real categories is used in the unsupervised analysis methods. In this section, we recall the Latent Semantic Analysis (LSA) from [24]. Using the text pre-analysis in the previous part, we built a filename-term matrix based on the filenames of the ransom notes the rows of which correspond to the filenames and columns correspond to the extracted terms from the filenames. To evaluate the importance of each extracted term in the filename individually and in the corpus of filenames, *Tf-Idf* (Term Frequenc-Inverse document Frequency) weighting that assembles local and global weighting was used to the Filename-Term Matrix (NTM). It is often used in LSA and text analytics.

The weighted matrix was decomposed into three matrices using Singular Value Decomposition (SVD). Indeed, LSA performs an analysis of co-occurrence across the corpus of filenames using SVD to reduce the dimensionality of the Filename-Term matrix to a dimension k. This dimension is lower than the minimal number between the number of rows and columns of our NTM. This reduction has the effect of preserving the strongest semantic information in the filenames and it throws away the noise. After this step, LSA identifies the related terms and puts them under one topic (among the k topics) without any prior knowledge on their real classifications. An interesting example that supposes the case of each word means only one concept, and each concept is described by one word. In this case, there is a simple mapping from words to concepts. In reality, however, this mapping is impossible because each language has different words that mean the same thing, named synonyms. Therefore, LSA tries to identify similar and related terms and puts them under one topic. We present the result of applying LSA in three parts:

1. On a corpus of filenames of the ransom files;
2. On a corpus of filenames of the ransom files and user files;
3. On a corpus of filenames of files collected from an infected machine by ransomware.

4.2.1. LSA on a Corpus of Filenames of the Ransom Files

The filename preparation and pre-analysis applied in this part have some differences to the pre-analysis of Section 4.1. We changed the random characters, the `ID` or the Bitcoin addresses in the filenames to `[]`. Therefore, the filenames that contain only these markers were removed from the list (empty filenames). After some LSA tests, we removed from the filenames the added extensions of the encrypted files. We did not know if given ransomware keeps the same extension in all its infections, generated for each target machine or it is a random extension. To obtain only alphanumeric filenames, we substituted a space for any special character. We replaced the uppercase letters with their lowercase form. Then the obtained results were filtered to avoid the digit filenames and any term in the filenames with one character. To split the terms, we used probabilistic splitter words (Wordninja: https://github.com/keredson/wordninja, accessed on 15 September 2021). We also tested another splitter (A compound word splitter for Python: https://github.com/TimKam/compound-word-splitter, accessed on 15 September 2021) named Compound Word Splitter. In our case, we found that the first splitter split the terms more correctly than the second splitter did. We did not use the stop words removal because these words are mostly used in the filenames. Finally, the obtained terms were stemmed using `Snowball` stemmer.

The results of LSA are related to the number k of dimensions (also named singular values or latent factors). Choosing few singular values can lose valuable information. On the contrary, choosing many latent factors may not be meaningful. Among the proposed solutions to find the number of latent factors, we used the percentage of variance explained by retaining k factors [23]. Then we visualized the results to determine how many latent factors to keep in our LSA by looking for an elbow in the plot. The results appear in

Figure 3. We found that the value $k = 16$ is an optimal dimension. The LSA similarities between the filenames shows that we cannot find large similarities, neither between all the filenames of the collected ransom files, nor between the filenames of the ransom files used by the same family in all its versions. Indeed, the filenames of the ransom files of some families have high similarities with the ransom filenames of other families.

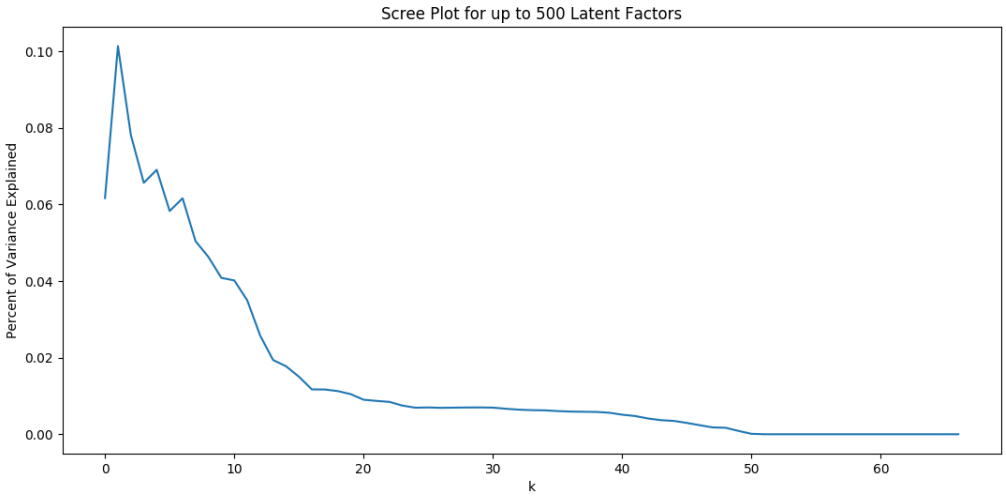

Figure 3. Variance Explained by Number of Singular Vectors.

Table 7 shows an example. The ransom filename `HELP_DECRYPT` of `CryptoWall 3.0` does not have high similarities with the ransom filenames of the other versions of this ransomware. On the other hand, this filename has high similarities with the ransom filenames of other ransomware families like `Cerber`, `Sad` and `Mr.Dec`. Therefore, the identification or the classification of the ransom files using only their filenames can give wrong results. Related to Section 3.4, ID-Ransomware or any ransom notes identifier must be aware about these wrong identifications using only the filenames of the ransom files. The top ten terms of the 16 topics ($k = 16$) are mentioned in Table A2 in Appendix A. We found that several terms like `decrypt`, `help` and `readm` are into more than one topic. Viewing several terms in more than one topic means that they are related between them in several filenames. In other words, this result can allow us to classify the filenames into two groups: ransom filenames and benign filenames.

Table 7. Similar Filenames to `HELP_DECRYPT` of `CryptoWall 3.0`.

Ransom Note	Sim.	Ransom Note	Sim.
HELP_YOUR_FILES (Cryptowall 2016-01-17)	0.500	_HELP_HELP_HELP_[random]_ (Cerber 2017-01-26)	0.770
DECRYPT_INSTRUCTION (Cryptowall 2.0)	0.350	_HELPME_DECRYPT_ (Sad)	0.750
INSTRUCTIONS_[ID] (Cryptowall 2016-02-05)	0.005	Decoding help (Mr.Dec)	0.740

4.2.2. LSA Applied on Files of an Uninfected Machine

We took 3000 filenames of several files from an uninfected machine running Windows 7. We added to this list the filenames of the ransom notes of our dataset. A pre-analysis process like the previous subsection was applied to all these filenames. Then we applied LSA using a splitter and without splitter on this corpus of filenames to extract any similarity between the benign filenames and ransom filenames. The percentage of variance explained by retaining k factors in both cases did not display a clear elbow to find an optimal k. For these reasons, we calculated the similarities between the words defined as ransom

filenames and the words defined as benign filenames. Then we took for different singular values the number of queries that corresponded to some cases of similarities. Table 8 shows the results of our tests. For example, the gray cell means that for $k = 1100$ without splitter, we have 209 queries that have similarities between 0.5 and 0.8. For our test, we consider only the similarities $s > 0.8$:

- In the case with splitter, the number of queries tends to be 62 queries that have similarities > 0.8 approximately from $k = 943$;
- Without splitter, the number of queries tends to be 60 queries that have similarities > 0.8 approximately from $k = 1509$.

Table 8. Number of Queries with Similarity < 0.5, and ≥ 0.5.

		Number of Queries				
		$s \leq 0.5$	$0.5 < s \leq 0.8$	$0.8 < s \leq 0.9$	$0.9 < s < 1$	$s = 1$
$k = 1000$	S	240245 (96.92%)	311 (0.129%)	1 (0.0004%)	1 (0.0004%)	60 (0.025%)
	NS	247556 (99.87%)	214 (0.089%)	11 (0.004%)	19 (0.007%)	60 (0.025%)
$k = 1100$	S	240245 (96.92%)	311 (0.129%)	1 (0.0004%)	2 (0.0008%)	59 (0.024%)
	NS	247561 (99.88%)	209 (0.086%)	11 (0.004%)	23 (0.009%)	56 (0.023%)
$k = 1400$	NS	247590 (99.89%)	208 (0.086%)	1 (0.0004%)	5 (0.002%)	56 (0.023%)
$k = 1700$	NS	247634 (99.90%)	166 (0.069%)	0 (0.0%)	46 (0.019%)	14 (0.005%)

s = similarity, NS = without splitter, S = with splitter.

Even though there were no ransom files within the benign files and the number of queries was a minimum for k maximal, some benign filenames were similar to ransom filenames (false positives). In this case, the ransom note detector can use other ransom file characteristics like the content to avoid these false positives. Table 9 shows for a similarity > 0.8 the ransom note filenames and their similar benign filenames for $k = 943$ with splitter and $k = 1509$ without splitter. The number of queries that have similarities > 0.8 in the case without the splitter is less (60 queries) than the case with the splitter (62 queries). In other words, the case with splitter can add other false positives. For example, the word `asasin` is divided into the words `as as in`, then it is similar to the benign file named ‘‘`ASF.pm`’’ (‘‘`asf`’’ is divided into `as f`). In summary, despite the machine not being infected by ransomware, we found some false positives for LSA. Compared to the number of benign filenames (3000 filenames), we had only 20 filenames in the case with the splitter and 18 filenames without the splitter, which makes our idea promising to be used with other monitored behaviors to detect the ransomware like checking the content of these files. The next subsection presents the same test, but on an infected machine by a ransomware.

4.2.3. LSA Applied on Files of an Infected Machine

We infected the same machine by `zoro` ransomware which adds in each target directory three `txt` ransom files: `!-GET_MY_FILES-!-.txt`, `#RECOVERY-PC#.txt` and `@_RESTORE-FILES_@.txt`. Our dataset does not contain the second and the third ransom files. We tried to find the ransom files of `Zoro` using LSA.

- All ransom files `!-GET_MY_FILES-!.txt` and `@_RESTORE-FILES_@.txt` were detected. However, the `txt` files of `#RECOVERY-PC#.txt` were not detected because their filenames contain the word `PC`, which did not exist in any ransom filename of our dataset and the chosen threshold of similarity (>0.8) was not sufficient to detect this filename without false positives. Generally, we can detect this ransom file by adjusting the used stemmer to include `recovery` and `recover` in the some word `recov`. Therefore, the stemming method on the filenames has some effects on the results: using one word, we can detect other related words.

Table 9. Ransom Filenames and their Similar Filenames.

With Splitter ($k = 943$)			Without Splitter ($k = 1509$)		
Ransom Note	**Nbr of Queries**	**Similar Files**	**Ransom Note**	**Nbr of Queries**	**Similar Files**
‘‘ReadMe’’ ‘‘_[]_README_’’ ‘‘README’’ ‘‘@README’’	14 for each ransom name	ExifTool\README Recent\readme.lnk hexrays\readme.txt IDAscope\README.md ExeinfoPE\readme.txt plugsdk\readme.txt PeiD\readme.txt WebAdmin\Readme.html Doc\ReadMe.txt PESpin\ReadMe.txt odbg\readme.txt upx\README upx\README.1ST reghost\readme.txt	‘‘ReadMe’’ "_[]_READ ME_" ‘‘README’’ ‘‘@README’’	14 for each ransom name	same files like with splitter
‘‘_HELP_HELP_ HELP_[]_’’	5	Mnt\Help.lnk epydoc\help.html IDA\idahelp.chm odbg\help.pdf odbg2\help.pdf	‘‘_HELP _HELP_ HELP_[]_’’	4	same files like with splitter case but without idahelp.chm
‘‘asasin’’	1	ExifTool\ASF.pm			

Table 10 shows the false positive filenames in both states: infected machine state and uninfected state (FP means that the file was detected as a false positive ransom file. TN means that the file is a true negative file). Comparing the two states, the last 11 files that were detected in the uninfected state as false positives are detected in the infected state as true negatives. The reason for these changes is related to the added extension to the encrypted files. Indeed, Zoro ransomware adds the extension aurora. A file named book.txt will be book.txt.aurora. This behavior of adding new extensions to the encrypted files is helpful for any ransom filename detector supposes that the filename is all the words in the filename until the last dot. Indeed, the detector will remove the extension (last word after the last dot), then takes the rest as a filename. For example, the filename of a file not encrypted by ransomware like test.sys is test and the filename of an encrypted file like book.txt.aurora is book.txt. In our test, this effect reduced the number of false positives seen in the previous part like the readme or help files of some tools. A ransom filename searched by the detector, such as readme.html, is readme which had a different LSA similarity to the filename of an encrypted file readme.txt.aurora. The last was evaluated by (LSA) as a readme.txt. The false positive filenames in the infected case are less than the uninfected case which means that our approach to use the ransom filenames to detect the ransomware is correct.

Table 10. FP and TN files in the Infected/Uninfected State with Splitter.

File Name	Uninfected State	Infected State
..\Maintenance\Help.lnk	FP	FP
..\epydoc\help.html	FP	FP
..\ExifTool\README	FP	FP
..\Recent\readme.lnk	FP	FP
..\IDAscope\README.md	FP	FP
..\WebAdmin\Readme.html	FP	FP
..\upx\README	FP	FP
..\upx\README.1ST	FP	FP
..\Recover My Files.lnk	TN	FP
..\Recover My Files v5.lnk	TN	FP
..\ExifTool\ASF.pm	FP	FP
..\hexrays_sdk\readme.txt	FP	TN
..\ExeinfoPe\readme.txt	FP	TN
..\pluginsdk\readme.txt	FP	TN
..\PeiD\readme.txt	FP	TN
..\Documentation\ReadMe.txt	FP	TN
..\PESpin\ReadMe.txt	FP	TN
..\odbg\readme.txt	FP	TN
..\regshot\readme.txt	FP	TN
..\IDA\idahelp.chm	FP	TN
..\odbg\help.pdf	FP	TN
..\odbg2\help.pdf	FP	TN

4.2.4. Machines Learning on Ransom Names

In this section, we describe how we used Machine Learning models to classify filenames as ransom filenames or benign filenames. The same pre-analysis presented above was used. The evaluated filenames were separated into two groups based on the chosen split rule: training set and testing set. A popular split [23] is to put 70% of the filenames in the training set and the rest in the testing set. The distribution of the filenames in our dataset was imbalanced, more benign filenames than ransom filenames. To solve this problem, we used Synthetic Minority Over-sampling Technique (SMOTE) [25] and Neighborhood Cleaning Rule (NCR) [26]. The first generated minor examples to be added to the original set. The second performs under-sampling based on the condensed nearest neighbor ($k = 1$ default case) method. The chosen learning models in this test were: Logistic Regression, Decision Tree, K-Nearest Neighbor, Naive Bayes, Random Forests and Support Vector Machine. The classification method was applied to the *Tf-Idf* weighted NTM (Filenames-Term Matrix) of the training set. Once the classifier was created, it was applied to the *Tf-Idf* weighted NTM of the testing set.

Among several ways to evaluate which model to choose, we have the accuracy of the models. Table 11 presents the overall accuracy in descending order (with F-measure) in the case without a splitter. We conclude that the Random Forests model is an optimal choice to detect the ransom filenames in our dataset due to its best accuracy value. Random forests (RF) are a part of the tree based model family. These models try to build a combination of tree predictors with different initial variables. Precision, recall and F-measure are described in Table 12 in the case without splitter. The overall accuracy of this model is 98.32%, and the overall F-measure is 0.920. The model misclassifies 13 filenames: nine ransom filenames are classified as benign filenames and four benign filenames are classified ransom filenames.

In summary, the detection of the ransom files using LSA or Machine Learning models on the filenames is possible. However, it must be combined with other indicators to detect the ransomware. ID-Ransomware service uses the saved filenames in its database without any check of similarity of the filenames to identify the associated ransomware to a given ransom file. However, checking the similarity of the ransom filenames can be used (with other patterns in the ransom files) to identify the ransomware associated to a given ransom file.

Table 11. Overall Accuracy in Descending Order.

Order	Model	Overall F-Measure	Overall Accuracy
1	Random Forests	0.920	98.32%
2	Support Vector Machine	0.895	97.81%
3	Decision Tree	0.900	97.81%
4	Naive Bayes	0.900	97.68%
5	Logistic Regression	0.890	97.55%
6	k-Nearest Neighbor	0.425	52.89%

Table 12. RF Contingency Matrix and Goodness of F-Measure (without splitter).

		Actual	
		(47 Ransom Filenames + 730 File Filenames)	
		Ransom Filename	**Benign Filename**
Predicted	**Ransom Name**	38	4
	Benign Name	9	726
	Precision	**Recall**	**F-measure**
Ransom Filename	0.90	0.80	0.85
Benign Filename	0.98	0.99	0.99

5. LSA on the Content of Ransom Files

Ransomware developers use the ransom files to contact their victims and notify them that their files have been encrypted. In some cases, the ransomware developers try to be creative in the content and the receipt addressed to their victims. The ransom files have a fixed objective which is to draw the attention of the victim in order to pay the ransom. Some ransomware families distinguish themselves from other families by adding some features in their ransom files. For example, `Spora` ransomware conducts its victims to a payment portal. This portal offers many services to the victims like deadline extensions and free decryption for some files. The most interesting feature is the real-time chat window where victims can contact the `Spora` developers. It was a unique feature compared to other ransomware seen in 2017. Another example is `White Rose` ransomware that was spread at the end of March 2018. The ransom note of this ransomware tells a strange story about its developer. He offers to share a white roses with the victims which results in encrypting the files. Despite the fact that there are several ransomware families in the world, the main aim of the ransom files of all ransomware has a fixed objective to inform the victim that his files are encrypted and the single way to restore the files is the payment of the ransom. This section is a continuation of the previous section that suggests the use of LSA on the filenames of the ransom files to identify the ransom files and detect the ransomware.

5.1. The Content to Identify the Ransom File

The exploration and application of the previous pre-analysis done above was applied, with some differences, in the stop word removal step. Contrary to the pre-analysis of the filenames where we used a created stop word list, we used the predefined stop words dictionary from the Python module `Nltk`. Table 13 presents the top ten of the used terms in the content of the ransom files.

The most used term is `file` and this term is the most used in the filenames. This term is followed by `tor, browser, encrypt`, then `decrypt`. These terms summarize the content of most ransom files: *files are encrypted to decrypt them and use the address in the Tor browser then follow the instructions.*

Table 13. Top 10 of the Used Terms in the Content of the Ransom Files.

	Term	Count		Term	Cont
1	file	1417	6	instruct	409
2	tor	951	7	key	371
3	browser	910	8	instal	359
4	decrypt	617	9	address	353
5	encrypt	419	10	internet	346

As in Section 4.2.1, we used the percentage of variance explained by retaining k factors. Then we visualized the results to determine how many latent factors to keep in our LSA by looking for an elbow in the plot. The results appear in Figure 4. We found that 3 and 6 are two optimal values for k. Figure 5 shows a part of the document–document (ransom file–ransom file) similarities in the six-dimensional LSA space. An example of pairs of documents with highest similarity value is the pair of the ransom file `[HOW_TO_DECRYPT_FILES].html` of `Locker` ransomware and the two ransom files `ReadMe.txt/.html` of `Sigma` ransomware. Other examples are framed in the same figure. Many ransom files of the same family (multiple versions) have high similarity between them. The results seem reasonable because the little reformulations/modifications in the content of these files have no effect on the results of LSA.

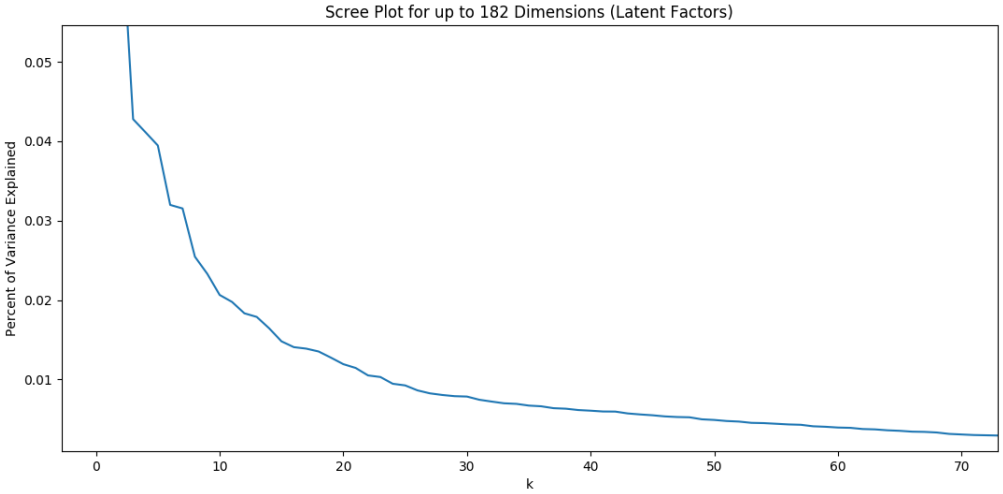

Figure 4. Variance Explained by the Number of Singular Vectors (k).

Section 3.4 describes the proposed process to identify the ransom files and to associate them with their ransomware by checking the similarity of the content in the last step of the process. The reason that checking the similarities of the content in the last step is due to few false positives ransom files. Indeed, we calculated for different thresholds $s \geq 0.9$ and for $k \in \{3, 6\}$ the number of similar ransom files for each ransom file in the corpus. Table 14 shows the number of false positives (FP) and true positives (TP) for some chosen thresholds of similarities (s).

	LockeR//[HOW_	CryptOl/HOW_T	Mole/_H/_HELP	Kee/Hel/Hello	Qweuirt/!!!Re	Sigma/R/ReadM	Sigma/y/yRead
LockeR//[HOW_	1.000000	0.959771	0.749207	0.684888	0.499212	0.966928	0.967410
CryptOl/HOW_T	0.959771	1.000000	0.710572	0.699417	0.588178	0.939850	0.938243
Mole/_H/_HELP	0.749207	0.710572	1.000000	0.283769	0.357602	0.650225	0.652635
Kee/Hel/Hello	0.684888	0.699417	0.283769	1.000000	0.851072	0.584249	0.579862
Qweuirt/!!!Re	0.499212	0.588178	0.357602	0.851072	1.000000	0.394422	0.387062
Sigma/R/ReadM	0.966928	0.939850	0.650225	0.584249	0.394422	1.000000	0.999934
Sigma/y/yRead	0.967410	0.938243	0.652635	0.579862	0.387062	0.999934	1.000000

Figure 5. Part of the Doc–Doc Similarities in the Six-dimensional LSA Space.

The optimal values of s and k can be found approximately from a threshold $s \geq 0.999$ for $k = 3$ and from $s \geq 0.99$ for $k = 6$. Indeed, approximately from these thresholds, the number of false positive is less than the number of true positive. The two cases tend to the

same number of false positives and true positives. We can say that the case $k = 6$ converges more quickly to eight false positives than case $k = 3$.

Table 14. Number of FP and TP Labels for $k = 3$ and $k = 6$.

Threshold	$k = 3$		$k = 6$	
	FP	**TP**	**FP**	**TP**
$0.9 \leq s$	6926	1180	2586	954
$0.99 \leq s$	1512	826	390	742
$0.999 \leq s$	332	582	86	486
$0.999995 \leq s$	14	376	8	372
$0.999999 \leq s$	8	372	8	372
$0.9999999 \leq s$	8	372	8	372

Table 15 shows the eight false positives. For example, the ransom file `HELP_RESTORE_FILES.txt` of `CryptoLocker` is labeled as a ransom file of `TeslaCrypt` or `Alpha` ransomware. For this reason, we choose that checking the content of the ransom files is the last used step by the ransom note identifier to identify the ransom files. On the other hand, it can be used to choose between multiple ransomware suggested by the ransomware identifier.

Table 15. The Eight False Positive Labels.

Ransom Note File	False Positive Ransomware Label
`HELP_RESTORE_FILES.txt` of CryptoLocker	TeslaCrypt & Alpha Crypt
`HELP_TO_SAVE_FILES.txt` of the version seen on 2015-04-30 of Alpha Crypt	TeslaCrypt & CryptoLocker
`HELP_RESTORE_FILES.txt` of the version seen on 2015-04-03 of TeslaCrypt	CryptoLocker & Alpha Crypt
`_READ_ME_FOR_DECRYPT.txt` of Rapid	StorageCrypt
`_READ_ME_FOR_DECRYPT` of StorageCrypt	Rapid

5.2. The Content to Detect the Ransomware

Our proposition to increase the effectiveness of the ransomware detection tools can be based on the following scenario. The reader can use more convincing and elaborate description of its detection/prevention scenario than our proposed scenario:

Let us suggest a ransomware detector that detects the ransomware using its malicious behaviors. For each behavior, a value v is added to a malice score m of a given monitored process. This detector concludes that the monitored process is a ransomware if m reaches a threshold t:

1. By checking the filenames of the created `txt`, `htm(l)`, `hta` and `rtf` files by LSA, the detector focuses only on the filenames of these files than focusing on the whole content of all the created files;
2. The detector applies LSA again on the content of any created file marked by the first item as a ransom filename;
3. If the content is marked as a ransom note content, the owner process that created this file is suspicious and it is probably ransomware. In this case, v is added to the malice score of this process. For this behavior, we suggest that the added value v is high such that m reaches the threshold t.

As described in Table 14, the similarity between the ransom files of different families is high despite a threshold $s \geq 0.9$. For this reason, we have calculated using LSA the similarities between the ransom files and benign files. The benign files were collected from different machines searching for any `txt` or `pdf` file named `readme` or containing the term `help` in the filename. We added to these files several files from different corpora like [26]

or [27] to construct a corpus of 200 randomly benign files. Using the same pre-analysis of Section 4.2.1, we found three optimal values for k, which are 6, 10 and 11. Table 16 shows the number of similar ransom files to benign files for each k.

Table 16. Number of Similar Ransom Files to Benign Files for some Thresholds.

Threshold of Similarity	Dimensions		
	6	**10**	**11**
$s \geq 0.90$	234	3	1
$s \geq 0.91$	199	1	1
$s \geq 0.92$	160	1	1
$s \geq 0.94$	94	1	1
$s \geq 0.95$	46	0	0
$s \geq 0.97$	6	0	0
$s \geq 0.99$	0	0	0

As shown in Table 16, we can differentiate between the benign files and ransom files. Indeed, the number of similar ransom files to benign files is null for $k = 6$ from a threshold $s \geq 0.99$ and $k \in \{10, 11\}$ from $s \geq 0.95$. Effectively, the content of the ransom files cannot evade from their objective and their context. To prove our idea, we collected three spread ransomware during 2020 and 2021. Then we tried to check their similarities with the benign files and ransom files of the same dataset. The collected ransomware were:

- `CryptoMix` that adds a ransom note file named `_HELP_INSTRUCTION.txt` at the beginning of infection (before encryption) in the root directory;
- `French101` adds the ransom file `HOW TO RECOVER ENCRYPTED FILES.txt` in any target directory containing target files before encrypting its content;
- `StopDjvu` ransomware adds `_readme.txt` in the Desktop of the current user before encrypting its content.

The three ransomwares add their ransom files before encryption. We compared the similarities of the ransom files of these ransomware with our dataset of benign files and ransom files. Table 17 shows for $k = 10$ and $s \geq 0.95$ the files that are similar to these ransom files. These ransom files of these ransomwares are similar only to the other ransom files. No similarities to benign files were seen.

We explored other ransomware seen in 2020 and 2021 that add their ransom files before encryption like `ChaCha`, `JSWorm`, `GlobImposter`, `StopDjvu` and `Buran` ransomware. Their ransom files have high LSA similarity ($\geq$0.90) with the other ransom files like the ransom files of `Dharma`, and `Querty`. Their LSA similarity with the benign files is low. Moreover, we tried to detect these ransomware using some ransomware detection tools like `AntiRansomV3` (Security Projects: AntiRansom,http://www.security-projects.com/?Anti_Ransom, accessed on 15 September 2021), `CryptoDrop` [3], `DaD` [12], `Acronis Ransomware protection` (Acronis Ransomware Protection, https://www.acronis.com/fr-fr/personal/free-data-protection/, accessed on 15 September 2021), `Kaspersky Anti-Ransomware` (Kaspersky Anti-Ransomware Tool, https://www.kaspersky.fr/anti-ransomware-tool, accessed on 15 September 2021) (without updates) and `CyberReason RansomFree` (Cybereason: Cybereason RansomFree, (https://www.cybereason.com/solutions/ransomware-protection, accessed on 15 September 2021). Most of these tools detected these ransomware after encrypting some files in the target directories and after they created the ransom files. Only the updated version of `Kaspersky Anti-Ransomware` detected these ransomware. This tool uses the signature scans with a behavioral analysis to detect the ransomware.

Table 17. Similar Files for $k = 10$ and $s \geq 0.95$.

Ransom Note Name	Sim.	Similar Files
`_HELP_INSTRUCTION.txt` (CryptoMix)	0.9858	`helloreadmenow23.txt` of CryptoBit ransomware
`HOW TO RECOVER ENCRYPTED FILES.txt` (French101)	0.9817	`READ THIS IF YOU WANT TO GET ALL YOUR FILES BACK.txt` of Omerta ransomware
	0.9872	`How to decrypt files.txt` of Vendetta ransomware
	0.9869	all `info.hta` files of Dharma ransomware
	0.9902	`!! RETURN FILES !!.txt` and `Payday.hta` of BTCWare ransomware
	0.9884	`!#_READ_ME_#!.hta` BTCWare ransomware
	0.9974	`HOW TO RECOVER ENCRYPTED FILES.txt` of Scarab ransomware (2018-10-12)
	0.9930	`IF YOU WANT TO GET ALL YOUR FILES BACK, PLEASE READ THIS.txt` of scarab ransomware (2017-11-23)
	0.9923	`DECRYPTING.txt` of Comonransomware ransomware
`_readme.txt` (StopDjvu)	0.9829	`README_DECRYPT.html` of striked ransomware
	0.9592	`RECOVER.txt` of hc7 ransomware
	0.9755	`ransom_pay.html` (unknown ransomware)
	0.9813	`how_to_back_files.html` of GlobImposter
	0.9768	`How to restore your files.hta` of GlobImposter ransomware
	0.9503	`[unknown].txt` of Amnesia
	0.9706	`DECRYPT.[].txt` of rapid
	0.9734	`README_BACK_FILES.htm` of Eq

In this section, the LSA tests on the content of the ransom files is detailed. Applying this method must be preceded by a pre-analysis on the content to filter it and keep only the significant terms. The results of applying LSA on the content are promising. Indeed, the identification of the ransom files using LSA as the last method to identify the ransom files can give more interesting results using a large database of ransom files. On another hand, the high similarities between the ransom files differentiate them from the benign files, which make their detection possible. Generally, a ransom note remains a file that notifies the victim to pay a ransom and the same content is repeated in most ransom files.

6. Conclusions

Through LSA and the extracted markers from the ransom files, we present our approaches to identify the ransom files and associate them with their ransomware. The results are promising and can be very promising by assembling all markers by checking the similarities of the filenames and the content of the ransom files in a real ransom note identifier with a large database of ransom files. On the other hand, we found that the used terms in the filenames of the ransom files are limited to a few terms, which make their

detection possible compared to the filenames of benign files on a target machine. However, using LSA or Machine Learning models such as Random Forest to differentiate between the filenames of benign files and ransom files shows interesting and encouraging results. Our idea to detect ransomware is based on using the ransom files. Indeed, we propose the use of LSA for the content of any file labeled as a ransom file by checking its filename. Most ransom files keep the same context in their ransom files to notify the users and pay the ransom. For this reason, the LSA similarity between the content of the ransom files and the content of benign files is not high. The continuation of this paper and future works should focus on:

- Assembling all the markers cited in the third section to make a complete and automatic ransomware identifier;
- Proving our approach to detect ransomware using the ransom files with other ransomware suspicious behaviors in a real ransomware detection tool like `CryptoDrop`, `ShieldFS`, `DaD` and other tools. We have started the first tests on `DaD` by checking the filenames of the ransom files. Firstly, we tested this detection tool on 60 arbitrary samples of several ransomware families, including some known families like `Dharma`, `Spora`, `StopDjvu`, `GandGrab` and `GlobImposter`. `Dad` detected all the 60 ransomware except five ransomware. One of these ransomware adds the ransom file before encryption. Three ransomware add the ransom file after encrypting the content of one target directory. The last ransomware is a multi-threading ransomware that adds the ransom file after encrypting at least one file. By adding our approach to `DaD`, we were able to detect all these ransomwares with few encrypted files. The result of using `DaD` and checking the ransom files to detect ransomware will be published in our future work;
- Using LSA on other features such as dumps memory of the running process on a target machine. Indeed, until the date of writing this paper, most ransomwares use the ransom files to notify the victims. However, with our detection technique using LSA on the filenames, the content of the ransom files is useful for fighting future ransomware. This detection technique can show some limits to detecting more ingenious future ransomware such as the ones that use encoded content for their ransom files or new communication channels through URL links. As proven in this paper, LSA has shown its effectiveness to differentiate between benign files and ransom files. For these reasons, we suppose that LSA can differentiate between the dump memory of a ransomware process and other benign processes.

Author Contributions: Formal analysis, Y.L. and J.-L.L.; Methodology, Y.L., J.-L.L. and E.M.S.; Project administration, E.M.S.; Resources, Y.L.; Software, J.-L.L.; Supervision, J.-L.L. and E.M.S.; Validation, J.-L.L. and E.M.S.; Writing—original draft, Y.L.; Writing—review & editing, Y.L. and E.M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors would like to express their acknowledgment and deep gratitude to both reviewers for their insightful comments, constructive remarks, and efforts towards improving their manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1 shows the used ransom files in this paper organized by ransomware families and extensions. The most used extension of the ransom files is `txt` followed by `htm(l)`. More than 30 families use the `txt` or htm(l) files as ransom note. Other ransomware like `Blind`, `Mr.Dec` and `NemucodAES` ransomware use `hta` files. `Matrix` ransomware family is the unique family in our collection that uses `rtf` files.

Table A1. Used Ransom Files.

Families	Number of Ransom Files	Extensions			
		TXT	HTM (L)	HTA	RTF
Alpha	1	1	-	-	-
Alpha Crypt	2	2	-	-	-
Argus	1	-	1	-	-
BadBlock	1	-	1	-	-
Blind	1	-	-	1	1
BTCWare	4	2	-	2	-
Cerber	17	6	3	8	-
Chip	1	1	-	-	-
Comonransomware	1	1	-	-	-
CrypMIC	2	1	1	-	-
Crypt0l0cker	1	-	1	-	-
Cryptfile2	1	1	-	-	-
CryptoBit	1	1	-	-	-
Cryptolocker	1	1	-	-	-
CryptoMix	2	2	-	-	-
Crypton	1	-	1	-	-
CryptoShield	6	3	3	-	-
CryptoWall	7	4	3	-	-
Cryptxxx	13	6	7	-	-
Dharma	16	6	-	10	-
Diamond	1	-	1	-	-
Dr.Fucker	1	-	1	-	-
Eq	1	-	1	-	-
Everbe	1	-	1	-	-
Evil locker	1	1	-	-	-
GandCrab	7	6	1	-	-
Gibon	1	1	-	-	-
GlobeImposter	4	-	3	1	-
HC7	2	2	-	-	-
Hermes	1	-	1	-	-
HydraCrypt	1	1	-	-	-
JAFF	4	2	2	-	-
Keyholder	1	-	1	-	-
LockeR	1	-	1	-	-
Locky	8	1	7	-	-
Matrix	4	-	-	2	1
MMM	2	-	2	-	-
Mole	1	1	-	-	-
Mr.Dec	1	-	-	1	-
NemucodAES	1	-	-	1	-
Omerta	1	1	-	-	-
PrincessLocker	4	2	2	-	-
Qwerty	1	1	-	-	-
Rapid	1	1	-	-	-
RaRansomware	1	-	1	-	-
Sad	3	1	1	1	-
Sage	2	-	1	1	-
Satana	1	1	-	-	-
Saturn	1	1	-	-	-
Scarab	2	2	-	-	-
Sigma	2	1	1	-	-
Sigrun	1	-	1	-	-
Spora	2	-	2	-	-
Striked	1	-	1	-	-
TeslaCrypt	18	11	7	-	-

Table A1. *Cont.*

Families	Number of Ransom Files	Extensions			
		TXT	HTM (L)	HTA	RTF
UIWIX	1	1	-	-	-
Unknown	5	4	1	-	-
Velso	1	1	-	-	-
Vendetta	1	1	-	-	-
WhiteRose	1	1	-	-	-
X3m	1	-	1	-	-
Zoro	1	1	-	-	-
Total	176	84	62	28	2

The top ten terms of the sixteen topics ($k = 16$) are mentioned in Table A2. We found that several terms like `decrypt`, `help` and `readm` into more than one topic.

Table A2. Top Ten Terms in the 16 Dimensions.

Dim.	Terms
0	*decrypt, file, how, restor, to, help, my, readm, your, encrypt*
1	*decrypt, help, my, readm, me, inform, argus, de, crypt, sos*
2	*readm, save, to, your, crypt, de, back, cke, bl, 23*
3	*info, as, in, get, this, do, text, repair, use, payday*
4	*recoveri, how, decrypt, to, my, ra, ware, ransom, do, text*
5	*recoveri, instruct, help, recov, your, readm, me, save, decod, sos,*
6	*recov, how, instruct, decrypt, to, encrypt, my, for, it, read,*
7	*encrypt, file, read, my, this, your, get, me, recoveri, thi,*
8	*restor, instruct, my, file, readm, recoveri, encrypt, decrypt, run, sig,*
9	*read, me, this, instruct, restor, thi, how, now, if, you,*
10	*instruct, encrypt, to, how, my, save, back, get, file, this,*
11	*encrypt, me, how, decrypt, help, restor, readm, read, now, inform,*
12	*my, me, help, how, readm, encrypt, get, decod, now, sos,*
13	*me, save, to, your, restor, encrypt, now, repair, use, my,*
14	*your, back, how, me, get, about, if, you, all, want,*
15	*back, get, repair, use, to, if, you, want, all, this*

References

1. Mager, M. Stop and Step Away from the Data: Rapid Anomaly Detection via Ransom Note File Classification. Endgame. 2018. Available online: https://www.elastic.co/fr/blog/stop-and-step-away-data-rapid-anomaly-detection-ransom-note-file-classification (accessed on 15 September 2021).
2. Nieuwenhuizen, D. A Behavioural-Based Approach to Ransomware Detection. MWR Labs Whitepaper. 2017. Available online: https://labs.f-secure.com/assets/resourceFiles/mwri-behavioural-ransomware-detection-2017-04-5.pdf (accessed on 15 September 2021).
3. Scaife, N.; Carter, H.; Traynor, P.; Butler, K.R.B. CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data. In Proceedings of the 36th IEEE International Conference on Distributed Computing Systems, ICDCS 2016, Nara, Japan, 27–30 June 2016; pp. 303–312.
4. Lemmou, Y.; Souidi, E. An Overview on Spora Ransomware. In *Security in Computing and Communications*; Springer: Singapore, 2017; pp. 259–275. [CrossRef]
5. Perekalin, A. WannaCry: Are You Safe? Kaspersky. 2017. Available online: https://www.kaspersky.com/blog/wannacry-ransomware/16518/ (accessed on 15 September 2021).
6. Lemmou, Y.; Souidi, E.M. PrincessLocker analysis. In Proceedings of the 2017 International Conference on Cyber Security And Protection of Digital Services (Cyber Security), London, UK, 19–20 June 2017; pp. 1–10.
7. Gazet, A. Comparative analysis of various ransomware virii. *J. Comput. Virol.* **2010**, *6*, 77–90. [CrossRef]
8. Caivano, D.; Canfora, G.; Cocomazzi, A.; Pirozzi, A.; Visaggio, C.A. Ransomware at X-rays. In Proceedings of the 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, UK, 21–23 June 2017; pp. 348–353.

9. Lemmou, Y.; Lanet, J.L.; Souidi, E.M. A behavioural in-depth analysis of ransomware infection. *IET Inf. Secur.* **2021**, *15*, 38–58. [CrossRef]
10. Kharraz, A.; Robertson, W.K.; Balzarotti, D.; Bilge, L.; Kirda, E. Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks. In Proceedings of the Detection of Intrusions and Malware, and Vulnerability Assessment—12th International Conference, DIMVA 2015, Milan, Italy, 9–10 July 2015; pp. 3–24.
11. Continella, A.; Guagnelli, A.; Zingaro, G.; Pasquale, G.D.; Barenghi, A.; Zanero, S.; Maggi, F. ShieldFS: A self-healing, ransomware-aware filesystem. In Proceedings of the 32nd Annual Conference on Computer Security Applications, ACSAC 2016, Los Angeles, CA, USA, 5–9 December 2016; pp. 336–347.
12. Palisse, A.; Durand, A.; Le Bouder, H.; Le Guernic, C.; Lanet, J.L. Data Aware Defense (DaD): Towards a Generic and Practical Ransomware Countermeasure. In Proceedings of the NordSec2017: 22nd Nordic Conference on Secure IT Systems, LNCS, Tartu, Estonia, 8–10 November 2017; Volume 10674, pp. 192–208. [CrossRef]
13. Lee, J.; Lee, J.; Hong, J. How to Make Efficient Decoy Files for Ransomware Detection? In Proceedings of the International Conference on Research in Adaptive and Convergent Systems, RACS'17, Krakow, Poland, 20–23 September 2017; pp. 208–212. [CrossRef]
14. Palisse, A.; Bouder, H.L.; Lanet, J.; Guernic, C.L.; Legay, A. Ransomware and the Legacy Crypto API. In Proceedings of the Risks and Security of Internet and Systems—11th International Conference, CRiSIS 2016, Roscoff, France, 5–7 September 2016; pp. 11–28.
15. Kolodenker, E.; Koch, W.; Stringhini, G.; Egele, M. PayBreak: Defense Against Cryptographic Ransomware. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 599–611.
16. Bello, I.; Chiroma, H.; Ali, U.; Jauro, F.; Khan, A.; Okesola, J.; Abdulhamid, S. Detecting ransomware attacks using intelligent algorithms: Recent development and next direction from deep learning and big data perspectives. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 8699–8717. [CrossRef]
17. Poudyal, S.; Dasgupta, D. Analysis of Crypto-Ransomware Using ML-Based Multi-Level Profiling. *IEEE Access* **2021**, *9*, 122532–122547. [CrossRef]
18. Kok, S.; Azween, A.; Jhanjhi, N. Evaluation metric for crypto-ransomware detection using machine learning. *J. Inf. Secur. Appl.* **2020**, *55*, 102646. [CrossRef]
19. Khan, F.; Ncube, C.; Ramasamy, L.K.; Kadry, S.; Nam, Y. A Digital DNA Sequencing Engine for Ransomware Detection Using Machine Learning. *IEEE Access* **2020**, *8*, 119710–119719. [CrossRef]
20. Ketzaki, E.; Toupas, P.; Giannoutakis, K.M.; Drosou, A.; Tzovaras, D. A Behaviour based Ransomware Detection using Neural Network Models. In Proceedings of the 2020 10th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, Germany, 16–18 September 2020; pp. 747–750. [CrossRef]
21. Moussaileb, R.; Bouget, B.; Palisse, A.; Le Bouder, H.; Cuppens, N.; Lanet, J.L. Ransomware's Early Mitigation Mechanisms. In Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES, Hamburg, Germany, 27–30 August 2018. [CrossRef]
22. Jawaheri, H.A.; Sabah, M.A.; Boshmaf, Y.; Erbad, A. Deanonymizing Tor hidden service users through Bitcoin transactions analysis. *Comput. Secur.* **2020**, *89*, 101684. [CrossRef]
23. Anandarajan, M.; Hill, C.; Nolan, T. *Practical Text Analytics: Maximizing the Value of Text Data*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2018.
24. Cosma, G.; Joy, M. An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis. *IEEE Trans. Comput.* **2012**, *61*, 379–394. [CrossRef]
25. Chawla, N.; Bowyer, K.; Hall, L.; Kegelmeyer, W. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
26. Open American National Corpus: MASC. Available online: http://www.anc.org/data/masc/downloads/data-download/ (accessed on 15 September 2021).
27. Schler, J.; Koppel, M.; Argamon, S.E.; Pennebaker, J.W. Effects of Age and Gender on Blogging. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*; American Association for Artificial Intelligence: Menlo Park, CA, USA, 2006.