Fighting Ransomware with Guided Undo

Matthias Held and Marcel Waldvogel*

University of Konstanz, Konstanz, Germany
{matthias.held, marcel.waldvogel}@uni-konstanz.de

Abstract

Ransomware attacks are rare, yet catastrophic. On closer inspection, they differ from other malware infections: Given appropriate preparation, they do not need to be detected and prevented, but could be recovered later. However, current ransomware protection follows the beaten path of anti-malware copying their fallacies. We show how the move to personal cloud storage allows for a paradigm shift in ransomware protection: exceptional attack isolation, perfect elimination of false positive alerts, and simplified recovery.

In this paper, we analyze the necessary operations for ransomware, extend existing ransomware taxonomy, and verify them against real-world malware samples. We analyze the costs and benefits of moving ransomware detection to versioned personal cloud storage. Our content, meta data, and behavior analysis paired with a 'guilt by association' capability greatly improve the false positive rate, but the guided undo make this rate all but inconsequential. Even though the user now carries a new burden, it comes with clear responsibilities and benefits, while being freed from questionable duties, resulting in a win-win situation for user experience and detection quality.

1 Introduction

Ransomware attacks are most effective when they can strike when no backups of the data exist, and the victim notices the encryption of his personal data after modifying files just a few hours ago. Tempting the victims to pay ransom in the hope of getting relief, and thus financing organized crime or losing all the personal data generating an economic loss for organizations and companies. It is therefore imperative for users and society to reduce the impact of ransomware to a rare event, causing no more than a brief annoyance. Current ransomware detection is integrated with or modeled after other malware protection: The key to defense is software running on the victim's machine, trying to identify malware when it is downloaded or first activated. Identifying the purpose of software from its bytestream is impossible [24], and once the software is running, anti-malware is not more powerful or bug-free than the operating system, whose duty includes protection from malicious software. This model cannot eliminate the problems of malware circumventing the detection software, thus leading to limitations which reduce the identification and recovery quality [23, 22].

Traditional ransomware detection analyzes program code and file operations, among other indicators, to detect potential malware activity. Either of them can result in false positives, triggering the user to react, often with panic. The anti-malware developer's goal therefore is to minimize the number of false positives, resulting in false negatives. If malware goes undetected, the data, even on the backup disk, when connected, may be encrypted or destroyed; an essentially unsolvable dilemma.

A more suitable solution would be an undo operation to the state of the files before the attack, with a backup strategy that cannot be circumvented. The ongoing move to personal cloud storage, however, opens the possibility of a paradigm shift in ransomware detection. Having

*The author presented this paper at the NISK 2018 conference.

Proceedings of the 11th Norwegian Information Security Conference, Sep. 19-20, 2018.
NISK Journal, e-ISSN 1894-7735. https://ojs.bibsys.no/index.php/NISK

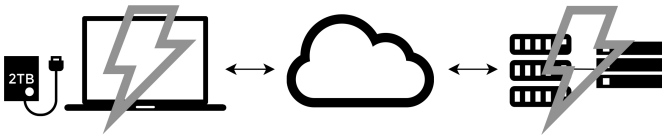

Figure 1: Possible victim machines

an up-to-date copy of all important data on an unrelated system where no ordinary user can install or run additional software provides for exceptional attack isolation, perfect elimination of false positive alerts, and simplified recovery. These benefits from attack isolation come at a cost: the weaker coupling to the events on the victim's machine reduces information about ongoing file operations as part of the malware, such as the sequence of actual file operations, and details including file offsets.

The missing real-time information has some impact on *content-based* and *metadata-based* indicators, which may analyze short-term (on the order of seconds) event sequences; however, analyzing long-term *behavior* remains unaffected. We shift the classification towards this indicator, supported by the file versioning to observe file operations without data loss. This shift improves the separation between processes of benign software which work with high entropy files and ransomware attacks.

The independence gained by the weak coupling between victim machine and cloud storage more than offsets the analysis limitations, as the detection software and archived file versions cannot be manipulated by the ransomware.

We propose the following data security model to protect against many incidents, including ransomware attacks (Figure 1): The user's machine (left) does a traditional backup to a locally-attached storage (far left). It also performs a real-time synchronization to a personal cloud storage, such as provided by a personal machine of the Raspberry Pi class or a cloud provider. Important is that the cloud storage be versioned, so that previous, unaffected, states of the data may be restored. As long as at most one of the two systems (user machine, personal cloud) is infected, data can be recovered. In the unlikely event that the cloud storage is the one being infected, synchronization of destroyed data may happen back to the user machine. This is guaranteed not to affect the backup, as it is outside of the synchronization scope.

As a result, unlike most other security incidents, ransomware does not necessarily need to be detected on first sight. Relaxing the requirements by using a delayed detection and recovery approach assisted by the personal cloud storage, allows us to remove the need of having a 100% detection rate without any false positives, something infeasible with real-time detection.

Offering such a model should also bring a usability improvement for the user by simplifying the recovery process by using the proposed indicators to analyse and classify the files and offering an color guidance for the user with an easy-to-use one-click recovery.

This paper makes the following contributions:

1. We define the general problems fighting ransomware using generic indicators.

2. We formulate a taxonomy of requirements for operation classes as well as indicator categories and verify it with real-world malware samples.

3. We define a set of *content-based*, *metadata-based* and *behavior-based* indicators which are a set of already suggested indicators and indicators extended or improved by us.

4. Based on these indicators, we implement a ransomware detection in an app on top of

Nextcloud cloud storage, which assists making the right selection when recovering from an attack.

2 Problems fighting ransomware

Ransomware attacks are rare events. However, is it necessary to analyze every file operation to detect them. Once we use an automatic recovery, we need to have a detection rate without any false positives since a – still so small – false positive rate would lead to many false recoveries due to many file operations performed on a computer system [18, 1].

Perfect protection can be achieved with a system that will not lose any data during a ransomware attack and will always recover the data that was destroyed by ransomware and not removed by the user himself. Educating the user through an easy-to-use recovery, guiding the user with informations gathered by analysing and classifying file operation sequences, will lead to such a mechanism where the user is in control. Since ransomware attacks are so rare, it is easier for the user to take a spare moment to recover if an attack happens instead of taking many moments struggling with false positives.

This change can be compared with the usability step as in the 80s where the move from file-based manual versioning to the simple undo method was established. This reduced the effort for the user drastically. Furthermore, it opened up the file versioning for the group of unsophisticated users by simplifying the recovery. This point is important in companies, universities or organizations which offer a "bring your own device (BYOD)" integration, where our usability concept can reduce the effort of the IT-support by allowing the users to perform the recovery by themselves [5].

This approach should not just be seen as another step in the arms race between defenders and attackers. Using backups on an autonom system is only existing solution which can be merely circumvented with tremendous effort: Automated simultaneous attacks of two related autonom systems with the goal to disable the backup system and encrypt the data. Extending this solution with our easy-to-use recovery improves it drastically by educating the user and allowing him to quickly restore all changes. Bypassing this recovery by exploiting the insight in our metrics would lead to additional effort, we might assume that this is comparable with the popular and well-known *Nigerian scam*. Where the attacker as *Nigerian royalty* asks for money by sending a poorly worded email. Cormac Herley in his paper *Why do Nigerian Scammers Say They are from Nigeria?* showed that not only defenders have false positives also the attackers suffer from false positive. This is a result of the trade off between profit and investment based on the effort performed. Cormac Herley stated that the typos in the emails are a method to reduce the false positives on the attacker side by removing the users who would answer these emails and still not pay the money from the pool of victims – generating unrewarded effort for the attackers – on the first sight leaving only the users who are naively enough to pay the money with or without the typos [7]. Transferring this to our problem allows us to assume that additional effort leads to a worse investment-profit-balance which does not result in an improvement of the ransomware but a more specific targeting of users who do not protect them self.

The generic real-time ransomware detection approaches proposed in different papers offer a real-time detection together with an automatic recovery proclaiming a nearly perfect detection rate with very few false positives [2, 3, 11, 9]. The rarity of a ransomware attack and automatic recovery make theses good detection rates problematic: Assuming we have a ransomware detection software, which monitors every of the 100,000 file operations a day on the file system with a false positive rate of 0.1%. This would result in 100 falsely triggered recoveries leading

to 100 unnecessary user interactions reverting these false recoveries. Additionally, the efforts for reducing the false positives lead to a very concrete set of indicators allowing the ransomware authors to build the malware against these indicators avoiding the detection [1].

For blocking decisions, we therefore should include every possible generic property. It is unavoidable to integrate a file versioning mechanism to be able to consider more file operations. This can also be done in real-time and with automatic recovery [3] but is limited by the base rate fallacy.

This cannot be considered reliable due to ransomware circumventing local backup software and detection software by disabling them or overwriting the Master Boot Record to perform the encryption in a state of the pc where only the malware is running [23, 22].

3 Related work

Existing ransomware indicators were used in different attempts to detect malicious software, while having several limitations like false positives in form of benign software, which fulfilled the same indicators e.g. high entropy files created by compression tools [17] or bypassing the monitoring by avoiding the indicators thresholds or hiding the malicious process [3, 8]. Long-term observations of ransomware showed their proceeding and possible indicators to prevent zero-day attacks of ransomware [10]. This already constructed a wide-spread set of indicators for detecting ransomware. Possible improvements for the indicators were evaluated: Such as the analysis of the entropy of files to improve it by being able to distinguish between encryption and compression, which are both high entropy files, without leading to a satisfying result [4].

Additionally, the long-term observation *Symantec* published several reports with information and statistics of ransomware over the last several years showing the spreading, behavior, attack methods and how to prevent attacks [19, 21, 20, 16].

Also other researchers published in-depth analyses of ransomware offering informations to many different families [15, 12].

Independent from such indicators, there are techniques like controlling the access to the command-and-control servers to prevent the execution of the ransomware [2] and the detection and escrowing of the encryption key to later decrypt the files without having to pay the money [11].

These indicators were already used to build techniques for detecting and preventing ransomware in real-time [8, 9, 11, 17]. In addition, there is a concept based on sequences and a backup strategy but it acts on the local system without the need of interaction of the user [3]. Limitations of this approach are mainly the bypassing of the monitor by tampering the Kernel or multiprocess malware.

Additionally, the base rate fallacy is a huge problem due to the rarity of ransomware attacks with just a few true positives and potentially many false positives due to plenty non-ransomware-specific file operations leading to a burden for the user [18, 1].

Our work is unlike the real-time approaches and can be described as *delayed detection and recovery*. It presents a method using a personal cloud storage by providing easy-to-use recovery from ransomware attacks including taking the users into control of the recovery. The need for such a concept was also indicated by a support page in the *Dropbox* help center [5]. They tackle this problem by requesting the user to recover every single file by hand or sending a list of encrypted files to the *Dropbox* support.

This approach is based on the personal cloud storage *Nextcloud* and utilizes the integrated file versioning and trash bin methods [13, 14]. The integrated methods use additional logical

file storages or directories to backup modified files with database links adding the necessary informations of the file version.

4 Background: Classes

Ransomware activity is not a static set of operations, which is performed in the same way in every ransomware family [10, 15, 12, 16]. The reasons for this are on one hand the race between security researchers and ransomware programmers to detect and hide the activities by changing the operations, and on the other hand the huge code base of existing ransomware to copy from. This leads to several implementations and many different families. Based on our analysis of ransomware operations, we also believe that some of the unusual, inefficient behavior we have seen is due to the high division of labor or just due to oversight or stupidity of the ransomware developer.

The process of constructing an efficient and working way to detect ransomware includes formulating classes for the different types of ransomware. These will then be used to define indicators.

The classes in this section are based on the behavior of existing ransomware families and possible – not yet existing – implementations. For a better overview, the ransomware classes are separated into the following subclasses: **File destruction**, **operation sequence**, **file type funnelling** and **Entropy funnelling**. In addition, the **operation sequence** classes are described for the local file system and the cloud storage.

A ransomware belongs to each of these subclasses and fulfils only one of the properties of the according subclass.

4.1 File destruction

The main behavior is the encryption of user files, therefore the ransomware must read the data, write the data and remove the original data afterwards.

This behavior is implemented in various ways by different ransomware variants. The diverse implementations can be classified in two classes:

Class A Replaces the data of a file *in-place*, thus it first reads the data from the file then encrypts it, writes the encrypted data back to the file, and closes it. Afterwards, the file is optionally renamed.

Class B Creates a new file and moves the data to the new file by reading the content from the original file, encrypting it and writing the encrypted data to the new file, deleting the original file after closing the files.

Existing literature classifies ransomware in three classes, where **Class A** and **Class B** remain the same [17]. They additionally introduced a class, which is an extension of **Class A**, moves the files in an extra folder before encrypting the files *in-place* and then moving them back. We treat this class as a subclass of **Class A**, because the main indicator for this class is the *in-place* encryption and the movement of the files is an unnecessary extension, which does not change the fundamental characteristics.

4.2 Operation sequence

The file destruction can be performed in different operation sequences where every sequence is unlike the others due to the order of operations. The ordering can change with every system

hence we divide the operation sequence classes into two different scenarios: The operations performed on a local file system and the operations performed during a synchronization process with a cloud storage.

The differentiation between those two scenarios have the simple reason, that the synchronization is performed by a client software, which reorders the file operations that happen during a synchronization interval leading to different operation orders.

Thus the classes for the local file system are defined as following:

Batch: Batch WRITE followed by batch DELETE/RENAME-with-overwrite.
Interleaved: Interleaved WRITE – DELETE/RENAME-with-overwrite.
In-place-overwrite: In-place-overwrite

The classes for the file operations visible during the synchronization are defined like this:

Batch: Batch WRITE followed by batch DELETE/RENAME.
Inversed batch: Batch DELETE/RENAME followed by batch WRITE.
Interleaved: Interleaved WRITE – DELETE/RENAME.
Chaos: Chaos WRITE – DELETE/RENAME.

Class **Inversed batch** represents a regroup of class **Batch** and class **Chaos** depicts a regroup and arbitrary split of class **Interleaved**. As mentioned in the previous section the operations can be preceded by first moving the files to a special directory.

4.3 File type funnelling

Other options include whether the file extensions of encrypted files are random strings, a ransomware-specific extension, or the file type is unchanged from the original file. Similar things may also happen to the file name [6].

File type funnelling describes the process of reading many files with known file extensions and writing files, which can be assigned to the following classes:

1. All file extensions are unknown and the same.
2. All file extensions are unknown and every extension is distinct.
3. All file extensions are unknown and mixed.
4. All file extensions are known, but files are corrupted.

We define "known" in this enumeration as: File extensions which are known well or file extensions with well-known file types. Additionally, class 3 and 4 are possible but so far non-existing implementations of file type funnelling.

4.4 Entropy funnelling

All ransomware families encrypt the data of the user files to extort the user. Hence the entropy of the files is increased – except if all the original files were compressed or also encrypted – thus all current ransomware families show the property of entropy funnelling, where the entropy level of all files is changed to become nearly the same.

In theory, there are options to reduce the entropy of the files written or have them stay nearly the same. Such malware has not been described nor seen in the wild.

5 Background: Indicators

The discussed indicators in this section capture the destructive properties of ransomware regarding a single file or a sequence of files and not modifications of the operation system. The described indicators were chosen to be general enough to work for all ransomware variants and try to ignore specific identifiers like specific strings or network traffic to the command-and-control server.

We divide these into three groups of *content-based*, *metadata-based* and *behavior-based* properties, where the *content-based* indicators capture the properties of the data, *metadata-based* indicators express the properties of the file and the *behavior-based* ones describe the properties of the sequence. They are described in more detail in [6].

Most of these indicators are well discussed in [8, 17, 3]. They also showed that the single indicator is not able to determine between benign software and ransomware, but several of those indicators together are able to do so.

The goal of the following chapter is to improve the significance of the Shannon Entropy and to append yet not discussed indicators, which are described in operation timing and operation quantities together with size quantities.

5.1 Content-based

5.1.1 Shannon entropy

The Shannon entropy is a measurement of information in a file. Files with a high entropy are compressed and encrypted files, where the information level is reduced to gain a specific property of compression or encryption. Thus a ransomware attack should change the entropy of files to a higher average due to the encryption of files.

The definition of the Shannon entropy for a byte array of 256 bytes is as follows:

$$e = \sum_{i=0}^{255} P_{B_i} log_2 \frac{1}{P_{B_i}}$$

for $P_{B_i} = \frac{F_i}{totalbytes}$ and F_i, the number of instances of byte value i in the array. This sum expresses a value between 0 and 8, where 8 is the perfectly even distribution of the byte values in the array [17].

This is an indicator for malware files, as mentioned encrypted files have a nearly 8 distribution. Compression also tends to a nearly 8 distribution. Therefore, we analysed high entropy files with the goal to find a method to distinguish between encrypted and compressed files.

5.1.2 Improved Shannon entropy

To reduce the false positives of compressed files we use the standard deviation of the entropy of the data to distinguish between compressed and encrypted files. This method is based on the differences in the variance of high entropy blocks in the file data which are depicted in the box plots 3 an 2 of file encrypted with *AES* and files compressed with *Deflate*. The box plots are asymptotic because with larger data blocks the entropy of the blocks approaches the expected value and therefore reduces the standard deviation and larger blocks compensate areas with low entropy better than smaller blocks.

We use the property that the entropy of data blocks of encrypted files are very dense distributed, where in contrast the entropy of data blocks of compressed files is not very dense distributed.

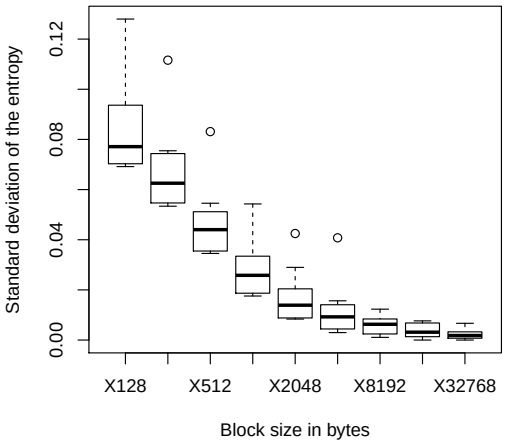

Figure 2: Boxplot of the standard deviation of files compressed with *Deflate*.

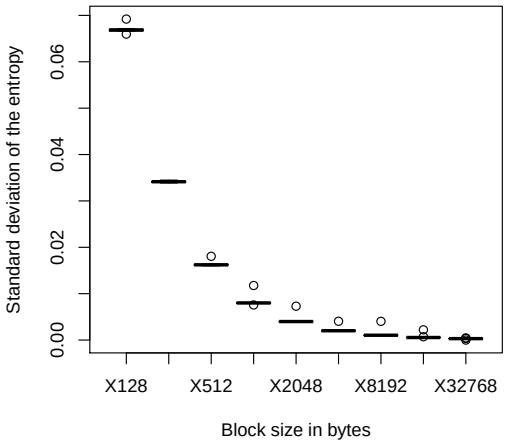

Figure 3: Boxplot of the standard deviation of files encrypted with *AES*.

This property is depicted by the standard deviation if we cut the data of the file in 256 bytes blocks and calculate the entropy of every block and the standard deviation of the entropy. The comparison of the standard deviation can be examined in figure 4. We choose a threshold of 0.06 for a data block size of 256 bytes for differentiating between encrypted and compressed files. The classification of this method does have a false positive rate of 10% but improves the indicator of the entropy of the data a lot due to the usage of a guilt by association presumption.

The false positive rate results from compression algorithms that have a very dense distribution with less outliers e.g. *LZMA* (see figure 5). In contradiction, all evaluated encryption algorithms generated a dense distribution hence we have less false negatives. Files classified as encrypted are more suspicious then other high entropy files thus we increase the suspicious classification of the file.

This means we have only a few false negatives and some false positives: If we classify a sequence we can use the 'guilt by association' conclusion to assure our suspected case.

6 Application implementation

The ransomware detection implementation is based on the personal cloud storage *Nextcloud*. The reason for this storage are the good file versioning and trash bin methods, which are integrated into this cloud storage and are used here to recover from a detected ransomware attack.

The functionality and limitations of the underlying file versioning and trash bin methods of *Nextcloud* will be described in the limitations.

To describe the integration into the *NextCloud* server, we describe the monitoring of file operations in the first subsection, followed by the integration of the sequence analysis. The last subsection describes the recovery method which uses the integrated file versioning and trash

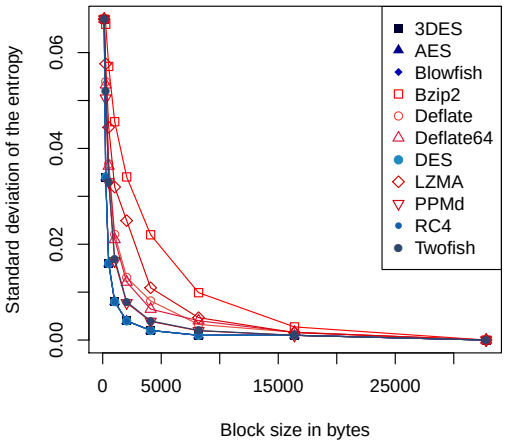

Figure 4: Comparison between the compressed files in red and encrypted files in blue.

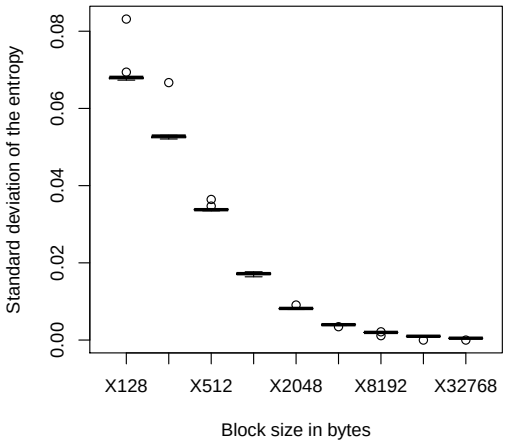

Figure 5: Boxplot of the standard deviation of files compressed with *LZMA*.

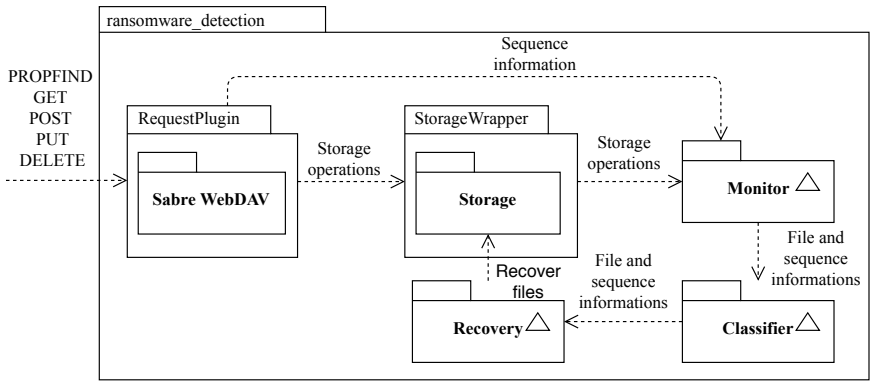

Figure 6: Architecture and integration of the ransomware detection application, which separates the concerns of monitoring, classifying and recovering files and file sequences.

bin of *Nextcloud*.

6.1 Monitoring

To monitor all file operations, the data storage of *Nextcloud* is wrapped, piping all file operations through our analyser, gaining the needed operations by analysing the operations with indicators defined in section 5.

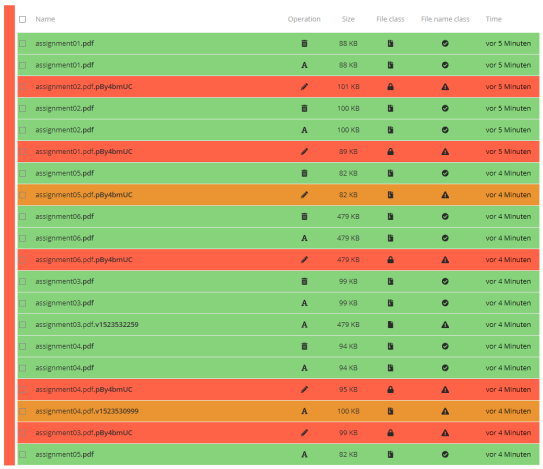

Figure 7: Colouring of a sequence of file operations created by the ransomware *GoldenEye*.

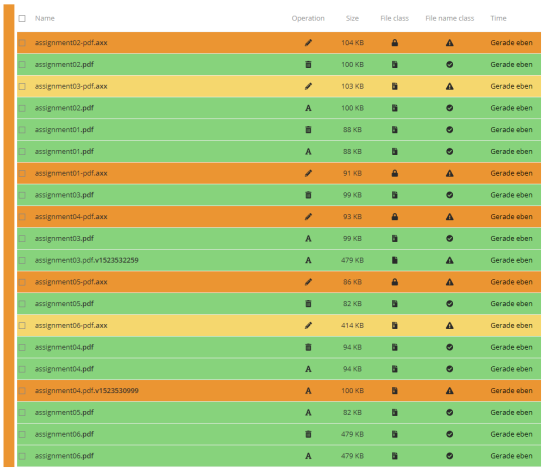

Figure 8: Colouring of a sequence of file operations created by the file encryption tool *AxCrypt*.

6.2 Sequence analysis

The sequence analysis puts file operations together to a sequence by looking for time intervals where nothing happens. In the context of *Nextcloud* this is equivalent with synchronization requests with no changes. Thus we are detecting all synchronization requests of a client without file storage access and if there is a succession of six synchronization requests without changes, we start a new sequence of file operations. The succession of six synchronization requests conforms three minutes where no changes are synchronized. This time interval is relevant for ransomware families (e.g. *WannaCry*) which work by batch writing all files and afterwards batch deleting all files and would lose the correct classification if the sequence of file operations would be split. During our evaluation, *WannaCry* delayed the file deletion by 10 to 90 seconds depending on the host system, therefore we defined a reasonably long enough time interval which reflects a break of file operations. However, if this threshold fails the user, in the situation of a ransomware attack, should nonetheless be able to find the split sequence easily by looking for the files which he knows were encrypted and recover both with an additional operation.

6.3 Classification

The results of the analysis performed during the monitoring are used to classify file operations and sequences of them. These results are used in recovery interface to give the user a color guide, which leads to the following depiction of file operation sequences (see figure 7, 8, 9 and 10).

6.4 Recovery

To recover from ransomware attacks the selected sequence of file operations will be reverted; that means new created files will be deleted and deleted or renamed files will be restored. This is done by using the integrated file versioning methods of *Nextcloud*.

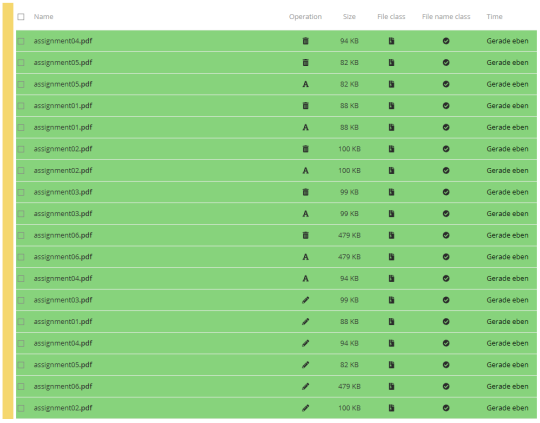

Figure 9: Colouring of a harmless sequence of file operations created by manual batch DELETE/RENAME followed by batch WRITE.

Figure 10: Colouring of a harmless sequence of file operations created by writing many files.

Family	File suspicion score	Quantities	File type funnelling	Entropy funnelling	Sequence suspicion score
BTCWare	0.88	2.00	2.00	2.00	6.88
Cerber	0.75	2.00	2.00	2.00	6.75
Evasive	0.83	2.00	2.00	2.00	6.83
Evader	0.50	2.00	2.00	2.00	6.50
GlobeImposter	0.63	2.00	2.00	2.00	6.63
WannaCry	0.81	2.00	2.00	2.00	6.81
Mamba	0.79	2.00	2.00	2.00	6.79
GoldenEye	0.93	2.00	2.00	2.00	6.93
Median	**0.63**	**2.00**	**2.00**	**2.00**	**6.63**

Table 1: A list of ransomware families and their *behavior-based* indicator values with the sequence suspicion score.

7 Experiments

In this section the application is reviewed in different aspects to evaluate the functionality and effectiveness. Therefore the classification of the sequences and the effectiveness of the indicators are evaluated.

7.1 Classification

The sequence classification was tested with a dataset of multiple ransomware families which were submitted to *VirusTotal* in the last months.

The table shows the weighted sum of the *content-based* and *metadata-based* indicators notated as file suspicion score, followed by the *behavior-based* indicators. The sequence suspicion score is the sum of all indicators and the file suspicion score, for more details see [6].

For every of those samples the classification reached a critical sequence suspicion score leading to a highly suspicious sequence.

AxCrypt also reached a higher suspicion level compared to the other benign processes. The reason for this is that this is file encryption software which has the same behavior as ransomware.

Program	File suspicion score	Quantities	File type funnelling	Entropy funnelling	Sequence suspicion score
Batch image resize	0.25	0.00	0.00	0.00	0.25
Application update process	0.25	0.00	0.00	0.00	0.25
AESCrypt	0.72	0.00	2.00	2.00	4.72
AxCrypt	0.68	1.00	2.00	2.00	5.68
WinRAR compress	0.00	0.00	0.00	0.00	0.00
WinRAR decompress	0.38	0.00	0.00	0.00	0.38
Zip compress	0.00	0.00	0.00	0.00	0.00
Zip decompress	0.38	0.00	0.00	0.00	0.38
Git	0.25	0.00	0.00	0.00	0.25
Median	**0.25**	**0.00**	**0.00**	**0.00**	**0.25**

Table 2: A list of benign application and their *behavior-based* indicator values additionally with the sequence suspicion score.

7.2 Indicator effectiveness

Comparing the indicators and their scores between the benign software and the ransomware we notice that the quantities and the file type funnelling clearly separate them. Additionally, the file suspicion level of benign software is lower but it is only useful in the union with the other indicators due to the classification of the file name, file extension and file class which can be easily faked.

In contrast the other two *behavior-based* indicators are less burdened with false positives, since it is much harder to hide those operations from the monitor, and capture multiple properties which are significant for ransomware.

8 Usability

To evaluate the usability of our approach we performed an online survey without any limitations for the participations. The reason for this is that our approach tries to be usable for everybody.

We had 30 participants between the age of 20 and 46 years with 57,1% being male and 42,9% being female.

The goal was to evaluate the usability of our approach compared to restoring all files file-by-file with the use of the trash bin of a personal cloud storage and in addition the support of color guidance for detection ransomware attacks.

The survey was structured as following: Firstly, we collected personal information and the foreknowledge followed by the preference about single file recovery or sequence recovery by asking simple multiple choice questions after describing the scenario to the participants. Thirdly, we asked the participants to decide, which of the given three sequences depicts a ransomware attack. This was done for three sets of three sequences each with and without color guidance. For each set of sequences without color guidance the colors were removed and was presented to the user before presenting the sets with color guidance. To assist the participants spotting the ransomware attacks, two behavior characteristics were given before displaying the sets of sequences. Each set was presented alone and the decision was collected directly afterwards. The sequences were displayed as screenshot similar to figures 7, 8, 9 and 10. Afterwards, the feedback and criticism of the color guidance and the whole approach was collected with a text field and we also asked – for participants who preferred single file recovery over sequence recovery – if they changed their mind about.

50% did not know something about ransomware before taking the survey but two did not answer this question. Asked if the participants would prefer to recover all files file-by-file or as a sequence of files – 96,7% preferred the sequence recovery. Reasons for this were the easy usage and the quicker approach but it was also demanded to be able to select single files. The one

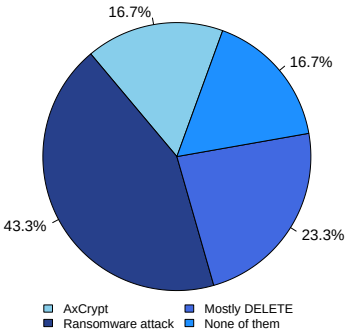

Figure 11: Unguided sequences classified as ransomware attack for sequence set 1.

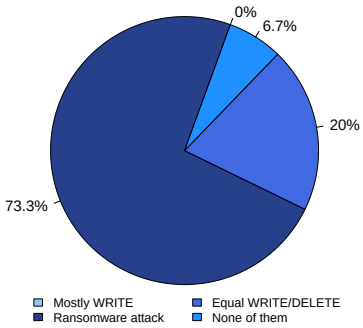

Figure 12: Unguided sequences classified as ransomware attack for sequence set 2.

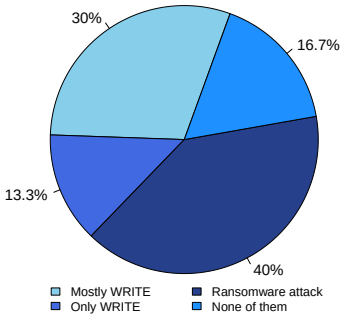

Figure 13: Unguided sequences classified as ransomware attack for sequence set 3.

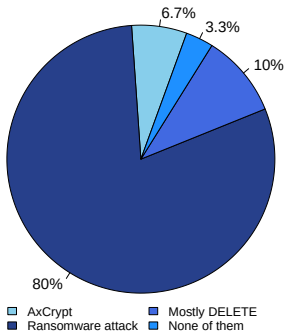

Figure 14: Guided sequences classified as ransomware attack for sequence set 1.

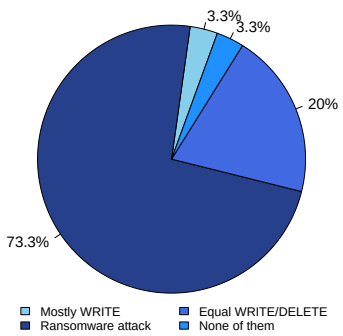

Figure 15: Guided sequences classified as ransomware attack for sequence set 2.

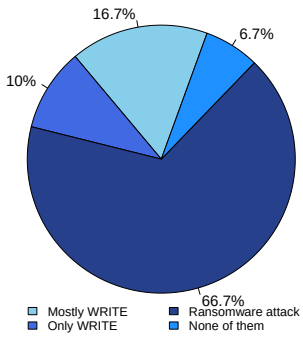

Figure 16: Guided sequences classified as ransomware attack for sequence set 3.

person preferring single file recovery over sequence recovery changed their mind after learning about the sequence recovery.

The participants correctly recovered 52% of the ransomware attacks without guidance compared to 73% correct recovered ransomware attacks with guidance. Thus we have an increase by 21% with the help of the color guidance. This is also confirmed by 73% of the participants who said that the color guidance helped them spotting the ransomware attacks quicker and easier due to the clear color choice and the direct comparison. One person was confused by the guidance and the approach as whole. Two participants were not able to spot the ransomware attack neither without guidance nor with the guidance and did not perceive the guidance as simplification.

Without the guidance the ransomware attack must fulfil one of the both given behavior

characteristics very clearly to be classified as ransomware attack by the participants. If this is not given or a sequence looks similar, the percentage of the correctly as ransomware classified sequence by the participants is only half (see figures 11, 12 and 13).

In contrast the sequences with guidance were classified correctly in over 66% of all questions (see figures 14, 15 and 16).

The foreknowledge does not help the user when it comes to spotting the ransomware attacks. We assume the reason for this is that many reports about ransomware do explain the basic behavior but the most readers cannot transfer their knowledge into a concrete process on a computer.

Conclusively, the recovery interface with the file sequences and the color coding are easily adopted and well understandable thus offer a very good usability. The color coding reduces the uncertainty of the user which seem to exist without it. The participants liked the quick recovery of file sequences making the extra amount of work – in contrast to automatic recovery approaches – not problematic.

More details of the survey are given in [6].

9 Limitations

Although we extract the false positives by handing over the decision to the educated user guided by the information gathered by our analysis and classification methods, we can separate the limitations in the non-technical user responsibilities and the technical ones.

The non-technical user responsibilities are to be creative, adaptive and sceptical to non-suspicious sequences which can be created with bypassing our technical approaches [18].

Independent from the non-technical user responsibilities, there are some technical limitations, which can be used to evade our detection mechanisms and the according countermeasures.

9.1 Attacking the file classification score

The file classification score consists of three parts: The file name classification, the file extension classification, including the file corruption, and the entropy of the file data.

For every part simulating a normal file takes effect and allows the attacker to keep the suspicion level below the threshold. The simulation of a normal file is simple for the file name and file extension but more extensive when it comes to the data entropy. This could be done by adding garbage data to it. This would lead to additional effort, for adding and removing the garbage data, and would also increase the file size. Hence we may gain additional detection patterns and encrypted files with the same reduced entropy or additional header informations about the entropy reduction if done randomly for every file.

All three bypassing methods are possible and would reduce the file suspicion level drastically but would be detected in the sequence classification score because the attacker has to encrypt many files on the system – optimally all files on the system – to launch an successful ransomware attack.

9.2 Attacking the sequence classification score

Attacking the sequence classification score is much more complex then attacking the file classification score since it is not based on *content-based* or *metadata-based* indicators but rather on *behavior-based* indicators. The behavior of ransomware is unique and is hardly changed.

We use four *behavior-based* detection techniques to classify a sequence of file operations: operation quantities and size quantities can be circumvented by delaying operations long enough that the monitor creates a new sequence or by writing additional files. However, delaying operations are contradictory to the needs of ransomware and increase the risk of being detected in the meantime. This risk is targeted by our approach by defining a long enough time interval between sequences. Although some ransomware families keep the file extension from the victims file, keeping the header informations of the file to bypass the corruption detection is special and would make the encryption and decryption process more complex. Writing additional files or increasing the file size by adding garbage data leads to additional effort because it must be added systematically that it can be removed before decrypting the file. The file type funnelling can be bypassed by using the same file extension and the same header informations as the victim file. To bypass the entropy funnelling the attacker must reduce the entropy of the file. The problems with this were already explained in the last subsection.

All these attacks are possible but are only effective as an union. Implementing only one or two would lead to a lower suspicion but the sequence could still be easily spotted as ransomware attack by the user.

9.3 File versioning

File versioning and trash bin implementations in personal cloud storages comprise some advantages in contrast to a classical backup strategy: A cloud storage separates the ransomware host from the file storage helping to guard the detection mechanism and the files. Additionally, we have double existence of the data thus a disruption of the synchronization – be it because of ransomware or by the user – allows us to always recover the files also if it is a selective disruption. Furthermore, people who locally backup regularly can be a victim of backup encryption by ransomware, a disk failure or a loss of the backup [23, 22].

Together with these features there are some negative aspects in terms of storage space, speed and complexity. *Nextcloud* uses a simple file versioning and trash bin functionality – by copying the file to a special directory before performing any changes to it – which can use up to 50% of the user's availably free space but uses clean up methods to reduce the space used. Furthermore, for every synchronized file modification a constant number of additional operations have to be performed to create a file version and backup the file accordingly in the correct storage. This usage of multiple backup storages and file management methods increases the complexity of storing and synchronizing files of the system [13, 14].

Compared to the gain of file versioning and the trash bin, the effort is reasonably small.

9.4 Local changes

Changes applied on the local system which are then later encrypted by a ransomware attack can be a problem depending on when the synchronization process takes place: If the synchronization takes place before the attack no data will be lost. If the synchronization takes place after the encryption the changes will be lost. The time interval between the modification of the data and the synchronization is small and so will be the changes. Thus we can assume that the user is currently working on the data and will immediately notice that something is wrong with the files. Therefore the user will be able to recover all the data and manual reapply the changes.

10 Conclusions

In this paper, we defined general problems fighting ransomware with generic properties. This included the base rate fallacy and the problem of missing significant generic indicators regarding a single file for blocking them in real-time.

We also described ransomware classes based on the behavior of the file destruction, the operations sequence, the file type funnelling and the entropy funnelling. Following, we formulated ransomware indicators and categorised them into three classes: *Content-based*, *metadata-based* and *behavior-based* ones.

We observed that we are able to separate encrypted from compressed files on the basis of the standard deviation of the entropy once we split the file in data blocks, allowing us to increase the validity of the entropy indicator.

Justified through the base rate fallacy of automatic real-time detection and recovery of ransomware attacks, we propose a delayed detection and recovery based on a personal cloud storage with file versioning permitting us to remove false positives and increase the usability by taking the user into responsibility. This approach is supported by the guidance of the classification of the file operation sequence based on the ransomware indicators.

This method is supported by the idea that we have no need to recover from ransomware attacks in real-time as long as we do not lose data in the mean-time, we can increase the quality of the classification and the usability of the recovery for the user.

The usage of a strategy with file versioning opens up the use of *behavior-based* indicators to classify the whole sequence file operations according to a ransomware attack. This strategy reflects an improvement regarding the already proposed classification approaches.

Additionally, the implementation of the ransomware detection on the basis of a personal cloud storage removes the threat of the monitor being attacked and also takes care that there is always a backup of the files.

11 Acknowledgements

The authors would like to thank VirusTotal for providing malware samples.

References

[1] BBC UK. A scanner to detect terrorists. [online], 2009. http://news.bbc.co.uk/2/hi/uk_news/magazine/8153539.stm (accessed March 28th, 2018).

[2] Krzysztof Cabaj and Wojciech Mazurczyk. Using Software-Defined Networking for Ransomware Mitigation: The Case of CryptoWall. *Netwrk. Mag. of Global Internetwkg.*, 30(6):14–20, November 2016.

[3] Andrea Continella, Alessandro Guagnelli, Giovanni Zingaro, Giulio De Pasquale, Alessandro Barenghi, Stefano Zanero, and Federico Maggi. ShieldFS: A Self-healing, Ransomware-aware Filesystem. In *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ACSAC '16, pages 336–347, New York, NY, USA, 2016. ACM.

[4] Craig from /dev/ttyS0. Differentiate Encryption From Compression Using Math. [online], 2013. http://www.devttys0.com/2013/06/differentiate-encryption-from-compression-using-math/ (accessed March 28th, 2018).

[5] Dropbox Inc. What to do if your files were corrupted or renamed by ransomware. [online], 2017. https://www.dropbox.com/help/security/ransomware-recovery (accessed March 28th, 2018).

[6] Matthias Held. Detecting ransomware. Master's thesis, University of Konstanz, Konstanz, Germany, 2018.

[7] Cormac Herley. Why do nigerian scammers say they are from nigeria? *WEIS*, June 2012.

[8] Amin Kharaz, Sajjad Arshad, Collin Mulliner, William Robertson, and Engin Kirda. UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 757–772, Austin, TX, 2016. USENIX Association.

[9] Amin Kharraz and Engin Kirda. Redemption: Real-Time Protection Against Ransomware at End-Hosts. In *RAID*, 2017.

[10] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin Kirda. Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks. In *Proceedings of the 12th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment - Volume 9148*, DIMVA 2015, pages 3–24, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

[11] Eugene Kolodenker, William Koch, Gianluca Stringhini, and Manuel Egele. PayBreak: Defense Against Cryptographic Ransomware. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pages 599–611, New York, NY, USA, 2017. ACM.

[12] Vadim Kotov and Mantej Singh Rajpal. Understanding Crypto-Ransomware: In-Depth Analysis of the Most Popular Malware Families. Technical report, Tech. rep. Bromium, 2014.

[13] Nextcloud GmbH. Nextcloud 13 user manual. [online], 2018. https://docs.nextcloud.com/server/13/user_manual/files/version_control.html (accessed June 4th, 2018).

[14] Nextcloud GmbH. Nextcloud 13 user manual. [online], 2018. https://docs.nextcloud.com/server/13/user_manual/files/deleted_file_management.html (accessed June 4th, 2018).

[15] Segun I Popoola, Ujioghosa B Iyekekpolo, Samuel O Ojewande, Faith O Sweetwilliams, and AA Atayero. Ransomware: Current Trend, Challenges, and Research Directions. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1, pages 169–174, 2017.

[16] Kevin Savage, Peter Coogan, and Hon Lau. The evolution of ransomware. *Symantec, Mountain View*, 2015.

[17] N. Scaife, H. Carter, P. Traynor, and K. R. B. Butler. CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 303–312, June 2016.

[18] Bruce Schneier. *Beyond Fear: Thinking Sensibly About Security in an Uncertain World*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[19] Symantec Corperation. Internet Security Threat Report: Ransomware and Businesses 2016. [online], 2016. https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/ISTR2016_Ransomware_and_Businesses.pdf (accessed March 28th, 2018).

[20] Symantec Corperation. Internet Security Threat Report: Volume 22. [online], 2016. https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf (accessed March 28th, 2018).

[21] Symantec Corperation. Internet Security Threat Report: Ransomware 2017. [online], 2017. https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/istr-ransomware-2017-en.pdf (accessed March 28th, 2018).

[22] Symantec Security Response. BadRabbit: New strain of ransomware hits Russia and Ukraine. [online], 2017. https://www.symantec.com/connect/blogs/badrabbit-new-strain-ransomware-hits-russia-and-ukraine (accessed March 28th, 2018).

[23] Symantec Security Response. Petya ransomware outbreak: Heres what you need to know. [online], 2017. https://www.symantec.com/blogs/threat-intelligence/petya-ransomware-wiper (accessed March 28th, 2018).

[24] Alan Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230265, 1937.