ダミーファイルを利用した 暗号化型ランサムウェア対策システムの実装

田中 智也[1,a)]　小池 一樹[1]　小林 良太郎[1]　加藤 雅彦[2]

概要：近年ランサムウェアの被害は減少傾向にあるが未だ大きな脅威である．中でも, 暗号化型ランサムウェアである WannaCry は世界中で猛威を振るった．しかし, 感染経路となった脆弱性に対するパッチは事前に配布済みであったことが判明している．従って, パッチ適用前の状態でも暗号化型ランサムウェアに対応可能な予防策が求められる．本研究では, 暗号化型ランサムウェアの暗号化順序に着目し, 最初に暗号化されるパスにダミーファイルを設置することで, 存在を知らせ, ランサムウェアによる暗号化を遮断する新しい手法を提案する．本稿では, 実際に仮想マシン上でランサムウェアを感染させ本手法の有効性を検証した．

キーワード：ランサムウェア, 暗号化順, ダミーファイル

Implementation of encryption type ransomware protection system using dummy file

Tanaka Tomoya[1,a)]　Koike Kazuki[1]　Kobayashi Ryotaro[1]　Kato Masahiko[2]

Abstract: Ransomware damage has been decreasing in recent years, but is still a major threat. In particular, an encryption type WannaCry raged around the world. However, it has been found that patches for vulnerabilities that became the infection route had been distributed in advance. Therefore, there is a need for preventive measures that can handle an encryption type ransomware even before patch application. In this research, we focus on the encryption order of encryption type ransomware and propose a new method of notifying existence and blocking encryption due to ransomware by placing a dummy file in the path to be encrypted first. In this paper, we actually infect ransomware on a virtual machine and verified the effectiveness of this method.

Keywords: ransomware, encryption order, dummy file

1. はじめに

2014 年から 2017 年にかけてランサムウェアによる被害は世界的に増大した [1]．その後, 被害は減少傾向にあるが未だに多くの被害が確認されている．IPA が発表している“情報セキュリティ 10 大脅威 2019” では, 個人に対する脅威で 9 位, 組織に対する脅威で 3 位に位置しており依然として重大な脅威といえる [2]．また, この結果から攻撃手法が個人向けのばらまき型から組織向けの標的型へ広がってきているといえる．

このような攻撃手法の違いだけでなく, 感染後の挙動にも大きく分けて 2 種類のタイプが存在する．1 種類目は, 暗号化型である．暗号化型とは, 感染するとパソコン内のデータを暗号化して使えなくした後, データを復号する引き換えに ransom(身代金) を要求するマルウェアである．また, 暗号化されたファイルは独自の拡張子に変化する特徴があ

[1] 工学院大学
Kogakuin University 1-24-2, Nishi-shinjuku, Shinjuku-ku, Tokyo, Japan
[2] 長崎県立大学
University of Nagasaki 1-1-1 Manabino, Nagayo-cho, Nishi-Sonogi-gun, Nagasaki, Japan
[a)] j116180@ns.kogakuin.ac.jp

る．2 種類目は, 画面ロック型である．画面ロック型とは, ユーザが操作できないように画面をロックした後, 解除することと引き換えに身代金を要求するマルウェアである．画面ロックは, 起動時に読み込まれるデータを暗号化し OS の起動プロセスを妨害することで可能にしている．この 2 つのタイプの大きな違いは, 暗号化する対象である．暗号化型の対象は, データ自身であるため復号鍵が不明な限りデータを取り戻すことは難しい．一方画面ロック型の対象は, 起動プロセスを妨害することに重点を置いているため, 重要データ自体を暗号化している暗号化型に比べ, 復旧できる可能性は高くなる．

近年では, 復号することが困難な暗号化型が主流となってきている．その中でも, 2017 年に猛威を振るった暗号化型ランサムウェアである “WannaCry” は世界中で注目を集めた．WannaCry は通常動作である暗号化機能に加え, 自己増殖するワーム機能も備えていたことにより人を介さず増殖し, 日本を含め世界中に被害が広まった．従来, ランサムウェアはメールの添付ファイルや悪意のある web サイト経由で感染することが多かった．一方, WannaCry の侵入方法は Windows のファイル共有プロトコル “SMBv1” の脆弱性を悪用したものであったため, 知らぬ間に感染する事態が起きた [3]．しかし, この脆弱性に対するセキュリティパッチは WannaCry の感染被害が広がる前に配布済みであったことが判明している [4]．それゆえ, セキュリティアップデートを指示通り適用していれば感染することはない．しかし, 実際には多くの被害が発生した．

この事態は，必ずしもユーザが配布済みのパッチを適用するとは限らない，ということを意味している．従って，パッチ適用前の状態でも暗号化型ランサムウェアに対応可能な予防策が求められる．

そこで, 本研究ではダミーファイルを用いることでランサムウェアによる暗号化を塞き止め, パッチや定義ファイルによらない対策手法を提案する．

第 2 章では本研究の重要な要素であるランサムウェアの暗号化操作について，第 3 章では提案手法のメインアイデアについて述べる．第 4 章では提案手法の実装について説明し，第 5 章で検証結果を述べる．第 6 章で結果から考えられる考察を行い，第 7 章で本論文をまとめる．

2. ランサムウェアの暗号化操作

本章では，ランサムウェアが行う暗号化の対策を考案するために行った，暗号化に伴う挙動の調査結果について述べる．調査対象としたランサムウェアはハニーポットから収集した以下の 4 つである．

- WannaCry
- Jigsaw
- Cerber
- TeslaCrypt

der, pfx, key, crt, csr, p12, pem, odt, ott, sxw, stw, uot, 3ds, max, 3dm, ods, ots, sxc, stc, dif, slk, wb2, odp, otp, sxd, std, uop, odg, otg, sxm, mml, lay, lay6, asc, sqlite3, sqlitedb, sql, accdb, mdb, db, dbf, odb, frm, myd, myi, ibd, mdf, ldf, sln, suo, cs, c, cpp
, pas, h, asm, js, cmd, bat, , ps1, vbs, vb, pl, dip, dch, , sch, brd, jsp, php, asp, rb, java, jar, class, sh, mp3, wav, swf, fla, wmv, mpg, vob, mpeg, asf, avi, mov, mp4, 3gp, mkv, 3g2, flv, wma, mid, m3u, m4u, djvu, svg, ai, psd, nef, tiff, tif, cgm
, raw, gif, png, bmp, jpg, jpeg, vcd, iso, backup, zip, rar, 7z, gz, tgz, tar, bak, tbk, bz2, PAQ, ARC, aes, gpg, vmx, vmdk, vdi, sldm, sldx, sti, sxi, 602, hwp, snt, onetoc2, dwg, pdf, wk1, wks, 123, rtf, csv, txt, vsdx, vsd, edb, eml, msg, ost, pst, potm, potx, ppam, ppsx, ppsm
, pps, pot, pptm, pptx, ppt, xltm, xltx, xlc, xlm, xlt, xlw, xlsb, xlsm, xlsx, xls, dotx, dotm, dot, docm, docb, docx, doc

図 1 WannaCry が暗号化対象としている拡張子一覧
Fig. 1 List of extensions WannaCry uses as encryption targets

まず, WannaCry を例に挙げて暗号化対象とする拡張子について述べる．次に調査対象のランサムウェアがどのような順番でファイルを暗号化していくのか述べる．最後に調査対象のランサムウェアが暗号化対象としているファイルサイズを述べる．

2.1 暗号化対象の拡張子

暗号化される拡張子はランサムウェアごとに全く異なる．中でも, WannaCry は 176 種類に及ぶ拡張子を暗号化の対象としている．その中には, Microsoft Office ファイルや圧縮ファイル, データベース, メディアファイル, 実行ファイル, プログラムのソースファイルなどが含まれている．解析により判明した拡張子の一覧を図 1 に示す．

2.2 フォルダ単位の暗号化順

暗号化型ランサムウェアはデータ自体を暗号化する特性があるため, 標的のフォルダ構造を把握する必要がある．フォルダ構造が不明ならば, 対象のファイルが認識できず暗号化処理に入れなくなるからである．従って, 各ランサムウェアは取得したフォルダの階層情報通りに移動していることから, 暗号化の順番は始めから決まっていると仮定する．

そこで暗号化順を明確にするため各フォルダ内に同様のファイルを設置し, 暗号化順に関する調査をした．ランサムウェアの暗号化は広範囲に及ぶ可能性があるため, 本調査では一般的に重要ファイルが保存される user フォルダより下位の階層に限定して行った．

調査結果に基づいて, 表 1 から表 4 に上記 4 つのランサムウェアの暗号化順を示す．

各マルウェアごとに 3 回実験を行ったところ，暗号化処理を行うフォルダの順番に変化は見られなかった．このことから, ランサムウェアの暗号化順は変化しないと考えられる．

一方で異なる点も確認できた．各ランサムウェアの暗号化順には共通点が見られなかった上に, 暗号化範囲にも違いがあった．WannaCry,Jigsaw,TeslaCrypt はいずれも 21 個のフォルダを暗号化対象にしている．一方, Cerber は 9 個のフォルダに留まっている．以上より, 開発者の意図によって優先フォルダや暗号化範囲は変わってくるといえる．

表 1　WannaCry の暗号化順
Table 1　WannaCry encryption order

1	C:\Users\%USERNAME%\Desktop
2	C:\Users\%USERNAME%\Documents
3	C:\Users\Public\Documents
4	C:\Users\All Users
5	C:\Users\Public\Downloads
6	C:\Users\Public\Favorites
7	C:\Users\Public\Libraries
8	C:\Users\Public\Music
9	C:\Users\Public\Pictures
10	C:\Users\Public\Recorded TV
11	C:\Users\Public\Videos
12	C:\Users\%USERNAME%
13	C:\Users\%USERNAME%\AppData
14	C:\Users\%USERNAME%\Contacts
15	C:\Users\%USERNAME%\Downloads
16	C:\Users\%USERNAME%\Favorites
17	C:\Users\%USERNAME%\Links
18	C:\Users\%USERNAME%\Music
19	C:\Users\%USERNAME%\Pictures
20	C:\Users\%USERNAME%\Saved Games
21	C:\Users\%USERNAME%\Searches

表 2　Jigsaw の暗号化順
Table 2　Jigsaw encryption order

1	C:\Users\All Users
2	C:\Users\Public
3	C:\Users\%USERNAME%
4	C:\Users\Public\Documents
5	C:\Users\Public\Downloads
6	C:\Users\Public\Music
7	C:\Users\Public\Pictures
8	C:\Users\Public\Recorded TV
9	C:\Users\Public\Videos
10	C:\Users\%USERNAME%\AppData
11	C:\Users\%USERNAME%\Contacts
12	C:\Users\%USERNAME%\Desktop
13	C:\Users\%USERNAME%\Documents
14	C:\Users\%USERNAME%\Downloads
15	C:\Users\%USERNAME%\Favorites
16	C:\Users\%USERNAME%\Links
17	C:\Users\%USERNAME%\Music
18	C:\Users\%USERNAME%\Pictures
19	C:\Users\%USERNAME%\Saved Games
20	C:\Users\%USERNAME%\Searches
21	C:\Users\%USERNAME%\Videos

2.3　ファイル単位の暗号化順

2.2 節では, フォルダ単位で暗号化順について述べた. 本節では着眼点を変え, 同フォルダ内に複数のファイルがあった場合の暗号化順について述べる.

表 3　Cerber の暗号化順
Table 3　Cerber encryption order

1	C:\Users\Public
2	C:\Users\Public\Documents
3	C:\Users\Public\Downloads
4	C:\Users\Public\Music
5	C:\Users\Public\Pictures
6	C:\Users\Public\Recorded TV
7	C:\Users\Public\Videos
8	C:\Users\%USERNAME%\Documents
9	C:\Users\%USERNAME%\Desktop

表 4　TeslaCrypt の暗号化順
Table 4　TeslaCrypt encryption order

1	C:\Users\All Users
2	C:\Users\Public
3	C:\Users\Public\Documents
4	C:\Users\Public\Downloads
5	C:\Users\Public\Music
6	C:\Users\Public\Pictures
7	C:\Users\Public\Recorded TV
8	C:\Users\Public\Videos
9	C:\Users\%USERNAME%
10	C:\Users\%USERNAME%\AppData
11	C:\Users\%USERNAME%\Contacts
12	C:\Users\%USERNAME%\Desktop
13	C:\Users\%USERNAME%\Documents
14	C:\Users\%USERNAME%\Downloads
15	C:\Users\%USERNAME%\Favorites
16	C:\Users\%USERNAME%\Links
17	C:\Users\%USERNAME%\Music
18	C:\Users\%USERNAME%\Pictures
19	C:\Users\%USERNAME%\Saved Games
20	C:\Users\%USERNAME%\Searches
21	C:\Users\%USERNAME%\Videos

標的ファイルを暗号化するためには事前にファイル情報を取得する必要があるため, ファイル自体の暗号化順序は利用されているリスト取得の処理によって決まると仮説を立てた.

今回はランサムウェアが python で記述された場合を想定して考える. python のリスト取得には, listdir メソッドと os.walk メソッドが広く使われている. まず, listdir メソッドはカレントディレクトリの 1 つ配下にあるファイル名とフォルダ名のみ読み取るため, ランサムウェアのような広範囲のリスト取得には不向きである.

一方, os.walk メソッドはフォルダ内のサブフォルダと配下のファイル全てを調べることができるため, ランサムウェアに向いている. さらに, user をカレントディレクトリとすれば, ユーザが操作する全ファイルの情報を一度で取得できる. 以上の点から, os.walk メソッドが使われてい

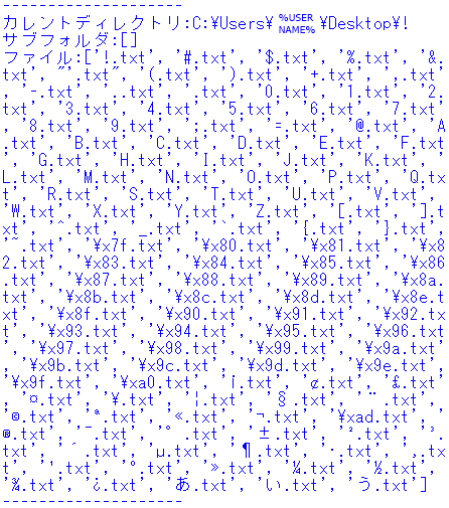

図 2　os.walk() のリスト取得順を示した図

Fig. 2　Figure showing the list acquisition order of os.walk ()

表 5　ファイルサイズにおいての暗号化範囲一覧

Table 5　List of encryption ranges in file size

調査対象	下限	上限
WannaCry	0B	なし
Jigsaw	0B	9.5MB
Cerber	2KB	なし
TeslaCrypt	0B	256MB

ると考え, os.walk メソッドのファイル取得順を調査した. 結果を図 2 に示す.

図 2 より, os.walk メソッドによるファイルリストは Unicode 順で出力されることが判明した. 従って, ランサムウェアにおいてもファイル名に応じて暗号化に順序があると仮定して, さらなる調査をした.

ファイル名を Unicode 順にした同サイズのファイルを同様のフォルダ内に設置し WannaCry, Jigsaw, Cerber, TeslaCrypt に対し調査を行った. その結果, 図 2 同様 Unicode 順となった. 以上のことから, フォルダ移動後の暗号化もファイル名に応じて順序が決まっているといえる.

2.4　暗号化対象のファイルサイズ

ランサムウェアには特定量のファイルのみを暗号化するものが存在する [5]. そのため, ランサムウェアはフォルダ単位で暗号化範囲を決めているだけでなく, ファイルサイズにおいても暗号化範囲を決めている可能性がある. 2.2 節で調査対象としたランサムウェアを用いてファイルサイズにおいて暗号化範囲を調査した. なお, ファイルサイズの上限は 100GB とした. 調査結果を表 5 に示す.

表 5 より, ファイルサイズに応じて暗号化範囲を決めていることが明らかとなった. Cerber のみ下限が設定されていた.

3.　提案手法

ランサムウェアが最初に暗号化を行うフォルダにダミーファイルを設置することで, 暗号化活動の検知, 妨害を行う手法を提案する. ダミーファイルとは, 何も機能や意味を持たず実際のファイルに模倣したものである.

まず, 最初に暗号化されるフォルダ内に空のダミーファイルが配置してあるフォルダを設置する. そして, そのフォルダ内を監視しダミーファイルに対して何かしらの操作が行われた場合, システムが起動し暗号化を遮断する.

3.1　ランサムウェアの検知方法

2 章で述べたように, ランサムウェアは暗号化する順番が予め決まっており変化がない. そのため, 最初に暗号化されるフォルダへダミーファイルを設置し監視することで, 重要ファイルへ感染する前に侵入を検知できる. 本研究でダミーファイルを設置するフォルダは, 表 1 より WannaCry が C:\Users\%USERNAME%\Desktop である. 表 2 , 表 4 より Jigsaw と TeslaCrypt は C:\Users\All Users である. 表 3 より Cerber は C:\Users\Public である. ファイル名は図 2 より最初に暗号化されるファイル “!. 拡張子” とする.

また, ランサムウェアによる暗号化が開始されたことを検知するためには暗号後の変化を読み取る必要がある. 感染後に起きるフォルダ内の変化は以下である [6].

- 拡張子の変化
- 暗号化によりファイル内容が変化
- 暗号化済みの新規ファイルを作成
- 元ファイルの削除
- 暗号化済みファイルへ上書き

上記 5 つの変化を発見するため, FileSystemWatcher Class[7] を利用してファイルを監視する. FileSystemWatcher Class は, 指定したフォルダ内のファイルおよびサブディレクトリの変化を監視でき, 変化が起きたときイベントを発生させるものである.

3.2　検知後のダミーファイルの挙動

3.1 節の方法で感染を検知した場合, 新たなダミーファイルを同フォルダ内に作成する. その後, 暗号化された不要なダミーファイルを削除する. この動きを繰り返すことでランサムウェアの暗号化を塞き止めることが可能となる. なぜなら, ランサムウェアはダミーファイルが存在する限り, 実際の標的ファイルを暗号化することは不可能だからである.

検知後の動作順序を以下に示す.

(1) ダミーファイルに対してファイル操作がされる
(2) 警告文を出す
(3) 新規ダミーファイルを複数作成

(4) 感染したダミーファイルを削除
(5) 余分な新規ダミーファイルを削除

上記の処理がランサムウェアの暗号化処理に追い抜かれることを防ぐため, 新規ダミーファイルを余分に確保する必要がある. 以上より, 新規ダミーファイルを複数作成することとした. なお, ダミーファイルの作成数は固定でなく, 暗号化速度に応じて適切に設定する必要があることに加え, 余分なダミーファイルはディスク容量を圧迫するため削除することが必要である.

3.3 ダミーファイル作成時の問題点と解決方法

ランサムウェアは, 各フォルダ内のファイルリストを取得した後, そのファイルリストに従って暗号化を始める. そのため, 暗号化処理中に新規ファイルを作成した場合, その新規ファイルは暗号化されない問題が起きる. 暗号化処理前にファイルリストを取得してしまうため, 暗号化処理中に作成されるファイルは存在しないものとして処理されてしまうからである.

この問題を解決するために以下の手段を取る.

(1) 暗号化処理前に空のダミーファイルをフォルダ内に設置しておく. (空のダミーファイルは有限個であり, 設置の方法は次節で述べる.)
(2) 暗号化が開始されたとき, 空のダミーファイルを十分に容量があるダミーファイルへ上書きする.

上記の解決策により, ダミーファイルをランサムウェアのファイルリストに追加させることが可能となる. 加えて, 事前に設置しておくダミーファイルは空ファイルであるため, ディスク容量を圧迫することはない.

しかし, 表 5 より Cerber には下限が設けられているため, 0B の空ファイルは回避されてしまう. 従って, 下限を設けているランサムウェアは, 下限のサイズを空ファイルとして設置する必要がある.

3.4 意図した順番で暗号化させる方法

2.3 節で述べた通り暗号化は Unicode 順で進められる. そのため, ダミーファイルのファイル名を Unicode 順で並べることにより, 意図した順番でファイルを暗号化させることが可能となる. しかし, リスト取得の処理はファイル名が 2 文字以上になったとき, 左から一つずつ独立した文字として Unicode 順で判定するため, 数字を用いてファイル名を決める場合は必ずしも昇順通りにならないことを考慮しなければならない.

以上のことを踏まえ, 本研究ではファイル名を 1 ずつ増加させダミーファイルを作成するシステムを構築した. このシステムを用いて昇順通り暗号化させるためには, 表 6 のように配置する必要がある.

表 6 の 2 行目を抜粋して説明する. 1.txt からはじめた場合, 昇順通り暗号化をさせるためには 9.txt までしか作成できない. 10.txt まで作成した場合, 1,10,2,3,4,5,6,7,8,9 の順番で暗号化されてしまうからである. このように, 昇順通り暗号化させたい場合は桁上がりをしないように設置しなければならない. 本手法を実現するためには上記のことを十分考慮しダミーファイルを設置する必要がある.

表 6 昇順通り暗号化される範囲
Table 6 Range that is encrypted in ascending order

始まり	終わり	確保できるファイル数
1.txt	9.txt	9 個
10.txt	99.txt	90 個
100.txt	999.txt	900 個
1000.txt	9999.txt	9000 個
10000.txt	99999.txt	90000 個

表 7 実装環境
Table 7 Implementation environment

環境	OS	CPU	RAM	Strage
ホスト	Ubuntu18.04 LTS	Intel® Core™ i7-8700 CPU	16GB	512GB
ゲスト 1	Windows 7 Professional	割り当て 4 コア	4GB	200GB
ゲスト 2	Windows 7 Professional	割り当て 2 コア	2GB	200GB

4. 実装

本章では, 提案手法の実装について述べる. 実現性を証明するため, 本手法を実装した仮想マシン上で実際のランサムウェアを実行し以下の検証を行った. 検証対象のランサムウェアは WannaCry, Jigsaw, Cerber, TeslaCryptr の 4 つである.

- 最初に暗号化されるフォルダへダミーファイルを設置し, ランサムウェア検体の実行時から, ターゲットファイルが暗号化されるまでの時間を計測する. (ターゲットファイルの設置場所はランサムウェア検体が暗号化するフォルダ内とする.)
- 仮想マシン上の性能を変え再度時間を計測する.

以上より, 本検証では最初に暗号化されるフォルダ内へダミーファイルを設置し, C:\Users \USERNAME \Documents のフォルダ内へターゲットファイルを設置する. 表 5 よりダミーファイルの容量は WannaCry, Cerber, TeslaCrypt が 256MB, Jigsaw は上限の 9.5MB とする.

4.1 実装環境

本節では実装環境について述べる. 実装環境を表 7 に示す.

ランサムウェア検体がいずれも正常に動作する必要があるため, ゲスト環境の OS は Windows 7 Professional を使用する. 1 回目の実装ではゲスト 1 環境を使い, 2 回目

表 8　検証結果一覧

Table 8　List of verification results

ランサムウェア検体	実装環境	耐久時間 (h)
WannaCry	ゲスト 1	26.3
	ゲスト 2	70.3
Jigsaw	ゲスト 1	Error
	ゲスト 2	Error
Cerber	ゲスト 1	28.1
	ゲスト 2	67.1
TeslaCrypt	ゲスト 1	52.9
	ゲスト 2	25.1 (!31169.txt 時点)

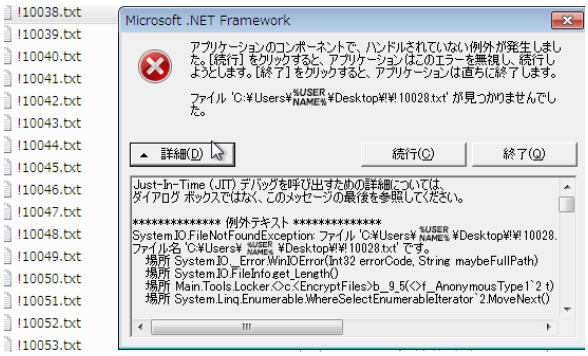

図 3　Jigsaw 検証時のエラー画面

Fig. 3　Error screen during Jigsaw verification

は CPU コア数と RAM を変えたゲスト 2 環境を使用する．ディスク容量は結果に影響しないと判断し 200GB 固定とした．また，本手法の暗号化妨害処理は，PowerShell スクリプトで記述，実装する．

4.2　空ファイル設置

3.3 節で述べたように下準備として空ファイルを設置する．ただし，設置数により検証結果が変わってしまうため，前もって設置数を決めておかなければならない．本研究ではランサムウェア検体がいずれも正常に動作し，かつより長く妨害できる 9 万個を設置して検証する．

5.　結果

本検証では各ランサムウェア検体，各実装環境で検証を行った．その内，耐久時間がより長かった結果を使用した．

本手法の検証結果を表 8 に示す．なお，表 8 の耐久時間の単位は hour とする．加えて，耐久時間の値は少数点第二位を四捨五入し，小数点第一位までの値を表示させている．

ゲスト 2 環境下での TeslaCrypt の検証結果は !31169.txt 時点で TeslaCrypt の暗号化処理に追い抜かれてしまったため (!31169.txt 時点) と記載した．また，Jigsaw に関してはエラーが発生し暗号化処理の動作が停止したため，表 8 の耐久時間を Error とした．エラーが発生したときの様子を図 3 に示す

表 8 から，TeslaCrypt, Cerber, WannaCry の順で耐久時間が長くなることを確認できた．Jigsaw のみ計測不能となったが，Jigsaw 自体の暗号化処理は停止していたためターゲットファイルが暗号化されることはなかった．

6.　考察

WannaCry, Cerber, TeslaCrypt の実装結果より，本手法がランサムウェアによる暗号化を長時間遮断できることが確認できた．一方，Jigsaw の暗号化処理は止まっていたため重要ファイルを守り抜くことは達成していた．以上より，本手法は感染後の適切な対応を行えるだけの時間を稼ぐことができるといえる．

6.1　各ランサムウェア検体の比較

表 8 の検証結果から，各ランサムウェア検体間で耐久時間に大きな開きが見られた．その理由としては，ランサムウェアの暗号化処理にある．ランサムウェアは各々異なる暗号化アルゴリズム，暗号化ライブラリを用いて暗号化処理を行っている [8]．仮にランサムウェアが全て同じ暗号化処理を扱っていた場合，容易に対策されてしまうことに起因する．従って，暗号化処理の違いから暗号化速度に差が生じ耐久時間に開きが表れたと考えられる．

6.2　Jigsaw 検証中にエラーが発生した原因

Jigsaw は検証中にエラーが発生し動作が停止してしまったため計測不能となった．本節では，Jigsaw の検証中にエラーが発生した原因を述べる．まず，図 3 の内容を確認すると FileNotFoundException[9] が要因であることが確認できる．FileNotFoundException Class はディスク上に存在しないファイルにアクセスしようとして失敗したときに発生する例外処理である．本検証中では，!.10028.txt ファイルに対して FileNotFoundException エラーが発生した．以上より，本手法がファイルリスト取得時に存在した!.10028.txt ファイルを暗号化操作の直前に余分なファイルとして削除したことが原因であると考えられる．Jigsaw のファイルアクセスのタイミングと余分なファイルを削除する処理のタイミングが一致しなければ FileNotFoundException エラーが発生することはないと考えられるが，互いの処理が非常に早いため FileNotFoundException エラーが発生しない検証結果を出すことは期待できないといえる．

解決策としては，原因となった操作の改善，暗号化処理が完了するまでファイルを削除しないことが挙げられる．しかし，暗号化処理が非常に早いため暗号化処理が完了するまで待機することは困難であり，確認動作を増やすことは本手法の性能低下につながる．また，他のランサムウェア検体に同様のエラーが発生しなかった要因としては，エラーに対して何らかの対策が施されていたことが挙げられる．

6.3 ゲスト 1 環境とゲスト 2 環境の結果比較

本節では, ゲスト 1 環境の結果とゲスト 2 環境の結果を比較し考察する. まず表 8 にある WannaCry, Cerber の結果に着目すると, ゲスト 2 環境の方が長い時間耐えることが確認できた. 長く耐えた要因としては, CPU と RAM の性能が半減したことが挙げられる. なぜなら, 性能低下によりランサムウェア検体の暗号化処理が遅くなったことに加え, 本手法の処理速度も遅くなり, 有限個であった空ファイルの消耗が抑えられたためである. 以上より, 性能がより低い環境ほど空ファイルの消耗を抑えることができるといえる.

次に, 表 8 にある TeslaCrypt の結果に着目すると, ゲスト 1 環境の方が長い時間耐えることが確認できた. しかし, ゲスト 2 環境の結果は!31169.txt 以降のダミーファイルに関しては考慮されていない. そのため, 最後のダミーファイルまで使用できていた場合 WannaCry, Cerber の結果と同様ゲスト 2 環境の方が長く耐えれていたと考えられる. このことから, 性能がより低い環境ほど本手法の信頼性が損なわれる可能性がある.

7. まとめ

本稿では, 未だ被害が多く確認されている暗号化型ランサムウェアに対し, ランサムウェアが最初に暗号化を行うフォルダにダミーファイルを設置することで, 暗号化活動の検知, 妨害を行う手法を提案した. また, 実装結果から本手法でランサムウェアの暗号化を長時間塞き止められることを示した.

今後の課題としては, 本手法では空ファイルが有限個であり, 耐久時間に限界があるため, 空ファイルを使用せずともダミーファイルを暗号化させる方法を検討することが挙げられる. 他には, 暗号化速度が早いランサムウェアに対応するためファイルを余分に作成する方法を取ったが, Jigsaw 検証時に発生したエラーの原因となってしまったため, 余分なファイルを作成せずともランサムウェアの暗号化処理に追い抜かれない方法を検討することが挙げられる.

謝辞 本研究の一部は,JSPS 科研費 17K00076, 19K11968, 19H04108 の支援により行った.

参考文献

[1] 中尾 康二, 歴史を紐解くセキュリティ技術, その現在, そして未来デジタルプラクティス,Vol.9,No.3,pp.596-608,2018.

[2] 独立行政法人情報処理推進機構 (IPA) , 情報セキュリティ 10 大脅威 2019 〜局面ごとにセキュリティ対策の最善手を〜, https://www.ipa.go.jp/files/000072668.pdf (Accessed 2019-07-05).

[3] “ランサムウェア WannaCrypt 攻撃に関するお客様ガイダンス”.Microsoft Security Response Center , https://msrc-blog.microsoft.com/2017/05/14/ransomware-wannacrypt-customer-guidance/ (Accessed 2019-06-08).

[4] 寺田 剛陽, 稲葉 緑, ユーザのセキュリティパッチ適用行動を促す心理学アプローチの検討研究報告セキュリティ心理学とトラスト (SPT),Vol.2019-SPT-32,No.15,pp.1-7,2019.

[5] Research Engineer - Eduardo Altares II (2014), “ファイル暗号化で「脅迫」するランサムウェア, 複数の新種を確認”, https://blog.trendmicro.co.jp/archives/9535 (Accessed 2019-08-13).

[6] 本多 俊貴, 向山 浩平, 白井 丈晴, 西垣 正勝, ユーザの文書編集操作を考慮したランサムウェア検知に関する一検討研究報告コンピュータセキュリティ (CSEC),Vol.2017-CSEC-77,No.14,pp.1-8,2017.

[7] “FileSystemWatcher Class (System.IO) — Microsoft Docs” https://docs.microsoft.com/ja-jp/dotnet/api/system.io.filesystemwatcher?view=netframework-4.8 (Accessed 2019-08-10).

[8] 重田 貴成, 森井 昌克, 長谷川 智久, 池上 雅人, 石川 堤一ランサムウェアにおける暗号化処理についてコンピュータセキュリティシンポジウム 2016 論文集,Vol.2016,No.2,pp.134-137,2016.

[9] “FileNotFoundException Class (System.IO) — Microsoft Docs” https://docs.microsoft.com/ja-jp/dotnet/api/system.io.filenotfoundexception?view=netframework-4.8