悪性文書ファイル検知のためのファイル構造検査の長期有効性

大坪雄平 †‡　　三村守 ‡　　田中英彦 ‡

† 警察庁
100-8974 東京都千代田区霞が関 2-1-2
mjp11001@grips.ac.jp

‡ 情報セキュリティ大学院大学
221-0835 神奈川県横浜市神奈川区鶴屋町 2-14-1

あらまし　実行ファイルが埋め込まれた悪性文書ファイルは，しばしば標的型攻撃に用いられる．我々は文書（Rich Text，Compound File Binary および PDF）ファイルに実行ファイルを埋め込んだ場合に，ファイルの構造に不整合が生じる特徴を検査することで悪性文書ファイルを高速かつ高確率で検知する手法を提案した．本論文では，その長期的な有効性を評価するため，新たに 2013 年から 2014 年までに標的型攻撃に用いられた悪性文書ファイルを用いて実験した．その結果，平均 0.253s で 96.1%の実行ファイルが埋め込まれた悪性文書ファイルを検知することができた，

Long-term Effectiveness of File Structure Inspection to Detect Malicious Document Files

Yuhei Otsubo†‡　　Mamoru Mimura‡　　Hidehiko Tanaka‡

†National Police Agency, Japan
2-1-2 Kasumigaseki, Chiyoda-ward, Tokyo 100-8974, JAPAN
mjp11001@grips.ac.jp
‡Institute of Information Security
2-14-1 Tsuruya-cho, Kanagawa-ward, Yokohama 221-0835, JAPAN

Abstract　Document files containing executable files are often used in targeted attacks. We proposed methods to fast and high detect malicious document files (Rich Text, Compound File Binary and PDF) using file structure inspection. Our methods can detect file structure syntax error contained in the malicious documents files containing executable files. In this paper, we examined new malicious document files that contained executable files to evaluate long-term effectiveness of our methods. These were used in targeted attacks from 2013 to 2014. Then, we could detect 96.1% of the malicious document files on average 0.253 seconds.

1　はじめに

近年では，特定の組織や個人を狙って情報窃取等を行う標的型攻撃が顕在化している．2015 年には，多くの組織で標的型メールを受信するだけでなく，マルウェア感染による情報漏洩にまで至っていることが発覚し，大きな社会問題となった．標的型攻撃では，受信者が不審に思わないような件名および本文の電子メールにマルウェアが添付されて送付されることが多い．トレンドマイクロ株式会社によると，2014 年に国内の標的型攻撃に用いられたファイルの拡張子の割合は，実行ファイルが約 7 割で残りの約 3 割が文書ファイルとなっている [1]．送付されたマルウェアが実行ファイルであれば，受信者が拡張子を確認することで不審なものであるか

否かある程度判断することができる．一方，標的型攻撃に文書ファイルが用いられた場合，受信者が悪性文書ファイルと通常の文書ファイルとを区別することは困難である．

標的型メール攻撃に用いられる悪性文書ファイルの多くには，実行ファイルやダミー表示用の文書ファイルが埋め込まれている．そこで，我々は文書（Rich Text，Compound File Binary（CFB）および PDF）ファイル構造を検査してその特徴を把握することで，実行ファイルが埋め込まれた悪性文書ファイルを検知する手法を提案した [2][3]．この手法では，悪性文書ファイルに埋め込まれた exploit（閲覧ソフトの脆弱性を攻撃するコード）や実行ファイル（exe や dll）等の不正なコードの分析はしない．その代わり，文書ファイルフォーマットの整合性を検査するファイル構造検査のみを実施することで，実行ファイルが埋め込まれた悪性文書ファイルの多くを検知することができる．2009 年から 2012 年までに標的型攻撃に用いられた実行ファイルが埋め込まれた悪性文書ファイルに対し実験した結果，高速かつ高確率で悪性文書ファイルを検知することができた．

その後，2014 年 11 月には一太郎のゼロデイ（未修正の脆弱性）[4] を利用した攻撃が確認される [1] など，マルウェアは日々新たなものが出現している．その結果，提案手法で検知ができない新たな実行ファイルが埋め込まれた悪性文書が出現している可能性がある．そこで，本論文では，検体として，2013 年から 2014 年に標的型攻撃に用いられた実行ファイルが埋め込まれた悪性文書ファイルを実験に用いることで，提案手法の再評価を行うとともに，今後とも提案手法が長期的に悪性文書ファイルを高い精度で検知できるか評価することを目的とする．

2　関連研究

我々の提案手法は，文書ファイルを対象に悪性であるか否かを，exploit を含む不正なコードを動作させずに検査する．この検査では実際にマルウェアは動作しないため，我々の提案手法は静的解析の一種と言える．静的解析によって文書型のマルウェアを検知する既存の手法としては，例えば，マルウェアに対応したパターンファイルを用いた検知手法，exploit に着目した検知手法，実行ファイルに着目した検知手法，文書ファイルの構造に着目した検知手法がある．我々の提案手法は文書ファイルの構造に着目した検知手法に分類される．以下，文書ファイルの構造に着目した検知手法に関連する先行研究について述べる．

2.1　ファイル構造全体に着目したもの

文献 [5] では，潜在的に危険なアクションを伴うフィルタを対象に機械学習を適用することで，不審な PDF ファイルを高速に検知する手法が提案されている．文献 [6] では，PDF のドキュメント階層構造を対象に機械学習を適用することで，不審な PDF ファイルを高速に検知する手法が提案されている．これらの手法は，悪性 PDF ファイル全般を対象としており，実行ファイルが埋め込まれていない PDF ファイルも検知することが可能である．しかしながら，教師あり学習モデルを用いているため，学習するためのサンプルを集める必要があるだけでなく，その精度は学習のサンプルに依存する．我々の提案手法では学習を必要としないため，サンプルは不要である．また，標的型メール攻撃によく用いられる実行ファイルが埋め込まれた悪性 PDF ファイル特有のファイル構造に絞ってファイル構造を検査することで，悪性 PDF ファイルを検知する精度を高めている．

2.2　データの秘匿に利用できるファイル構造に着目したもの

文献 [7] では，CFB ファイルの構造を検査することにより，CFB ファイルに埋め込まれた，表示内容と関係のないデータを解析するツールが提案されている．このツールは CFB ファイル内でデータが秘匿される可能性がある 4 種類の場所を表示する．しかしながらこのツールは解析を目的としており，仮にこのツールを悪性 CFB ファイル検知に活用した場合，悪性 CFB

ファイルだけでなく通常の CFB ファイルも検知してしまうという課題がある．我々の提案手法では，表示内容と関係のないデータではなく，実行ファイルが埋め込まれた悪性 MS 文書ファイル特有のファイル構造に絞ってファイル構造を検査することで，悪性 CFB ファイルを検知する確率を高くしている．

2.3 ファイル構造の整合性に着目したもの

Microsoft 社が Microsoft Office 2010 から導入した Office ファイル検証機能 [8] は，ファイル構造を検証して悪性文書ファイルを検知するという，我々の提案と同じ目的の機能である．この Office ファイル検証機能がどのようなファイル構造の特徴を検証しているかは不明であるものの，Office ファイル（xls，doc，ppt，pub 拡張子）を開こうとした場合に，正常なファイル構造の Office ファイルか否かを検証し，正常なファイル構造でない場合に閲覧するか否か確認するポップアップウインドウが表示される．文献 [2] で両者の検知率を比較した結果，Office ファイル検証機能は我々の提案手法で検査している特徴と異なるものを検査対象としていると思われる．

3 提案手法の検査項目

標的型攻撃に用いられる実行ファイルが埋め込まれた悪性文書ファイルでは，埋め込まれた実行ファイルの取り出しを shellcode で行う．shellcode はサイズが制限されることが多いため，実行ファイルは復号に必要なコードを小さくできる単純な方式を用いて文書ファイルに埋め込まれることが多い．一方，文書ファイルのフォーマットの規約に従って埋め込まれた実行ファイルを復号しようとした場合，必要なコードが大きくなることがほとんどである．このことから，実行ファイルが埋め込まれた悪性文書ファイルには，文書ファイルのフォーマットの規約から外れるファイル構造上の特徴が表れる．我々の提案手法では，これらの特徴がないかファイル構造を検査する．

検査項目は Rich Text で 1 つ，Microsoft Word 等で使われる CFB で 4 つ，PDF で 3 つの合計 8 つである．詳細については文献 [2][3] に示すとおりであり，概要を以下に示す．

- ファイルの終端を示す記号の後にデータが追加されているか（Rich Text）
- ファイルサイズが 512 の倍数でないか（CFB）
- 管理領域で参照できない領域にデータが追加されているか（CFB）
- ファイル末端のデータが管理領域で未使用と定義されているか（CFB）
- ヘッダ等から用途を分類できないデータがないか（CFB）
- コメント，本体，相互参照テーブルおよびトレーラの 4 つのセクションに分類できないデータがあるか（PDF）
- どこからも参照されないことで用途不明のオブジェクト（数値，文字列，バイナリデータ等）があるか（PDF）
- Stream（バイナリデータを格納するオブジェクト）のデコードに失敗するかまたはデコードに使用しない用途不明のデータが追加されているか（PDF）

4 実験

4.1 試験プログラムの概要

これまでに示したファイル構造上の特徴を検知するプログラムを，試験プログラムとして実験に使用した．この試験プログラムは文献 [2][3] で使用したものと同一のものであり，オープンソースのプログラミング言語である Python を用いて実装されている．各特徴に合致するか独立で判定し，いずれか 1 つでも特徴に合致した場合に悪性文書ファイルと判定する．

4.2 実験環境

実験を実施する環境は表 1 に示すとおりであり，実験はすべて仮想マシン上で行った．

表 1: 実験環境

CPU	Core i5-3450 3.1GHz
Memory	8.0GB
OS	Windows 7 SP1
Memory(VM)	512MB
OS(VM)	Windows XP SP3
Interpreter(VM)	Python 2.7.6

4.3 実験 1 の内容

提案手法の有効性を再評価するため，2013 年 1 月以降に発生した標的型攻撃に使用された悪性文書ファイルを試験プログラムに入力して結果を分析する．検体の採取期間は 2013 年 1 月から 2014 年 12 月までとし，複数の組織から採取した．実験の対象となる文書ファイルの概要を表 2 に示す．表の pps は，Microsoft PowerPoint でスライドショー形式で保存した場合の拡張子で，CFB のファイル構造を用いている．これらの検体は，分析により実行ファイルが埋め込まれていることをあらかじめ確認しているものである．ただし，拡張子と実際の中身が一致していないものが 42 個あり，それらについては実際の中身に基づいて分類した．特定の脆弱性の種類，検知名，RAT の種類等について，同一のものが多数含まれるといった検体の偏りが生じるのを防ぐため，検体の採取期間に標的型攻撃に用いられたメールとして提供を受けたもの全てから添付ファイルを取り出し，拡張子が文書ファイルのものを機械的に選定した．その上で，同一のハッシュ値を持つものは取り除いた．検体に悪用された脆弱性を表 3 に示す．最も悪用された脆弱性は MS12-027 であった．なお，1 つの検体で複数の脆弱性を悪用するものがあるため，表 3 の数字の合計は 100%にならない．また，検体を入手した時点でゼロデイであったものは 23 個あった．これらの検体を試験プログラムに入力し，検知率および平均実行時間を求める．この結果と文献 [2][3] での実験の結果を比較する．また，試験プログラムの検知率と採取した当時の最新パターンファイルを適用した大手ベンダのウイルス対策ソフトの検知率を比較する．

表 2: 検体の概要

拡張子	検体数	平均容量 (KB)	偽装
rtf	44	554.1	40
doc	35	193.8	2
xls	15	277.3	0
pps	1	1210.0	0
jtd	25	437.0	0
pdf	7	753.0	0
合計	127	415.2	42

表 3: 検体が利用する脆弱性

脆弱性	個数	割合
MS10-087	2 / 127	1.6%
MS12-027	71 / 127	55.9%
MS14-017	3 / 127	2.4%
JS13003	15 / 127	11.8%
JS14003	19 / 127	15.0%
APSB10-21	4 / 127	3.2%
APSB11-08	4 / 127	3.2%
APSB11-30	4 / 127	3.2%
APSB12-22	1 / 127	0.8%
APSB13-07	3 / 127	2.4%
なし	3 / 127	2.4%
不明	6 / 127	4.7%

4.4 実験 1 の結果

検体の拡張子ごとの検知率を表 4 に示す．この表における検知数の欄は，検知数/検体数 を表している．検知率は全体で 96.1%で，平均実行時間は 0.263s であった．また，2009 年 1 月から 2012 年 12 月までに標的型攻撃に用いられた実行ファイルが埋め込まれた悪性文書ファイル 364 個に対し実験した結果を表 5（文献 [2][3] を元に作成）に示す．この表における検知数の欄は，検知数/検体数 を表している．検知率は全体で 98.9%で，平均実行時間は 0.339s であった．

次に，試験プログラムの検知率と，大手ベンダのウイルス対策ソフトの検知率との比較結果を表 6 に示す．この表における検知数の欄は，検知数/検体数 を表している．実験に用いたウイルス対策ソフトのパターンファイルは毎日最新のものに更新しており，我々が検体を入手した

表 4: 提案手法の検知率 (2013〜2014)

拡張子	検知数	検知率	平均実行時間
rtf	41 / 44	93.2%	0.226s
doc	34 / 35	97.1%	0.170s
xls	15 / 15	100.0%	0.184s
pps	0 / 1	0%	0.171s
jtd	25 / 25	100.0%	0.194s
pdf	7 / 7	100.0%	1.382s
合計	122 / 127	96.1%	0.263s

表 5: 提案手法の検知率 (2009〜2012)

拡張子	検知数	検知率	平均実行時間
rtf	97 / 98	99.0%	0.021s
doc	35 / 36	97.2%	0.062s
xls	48 / 49	98.0%	0.051s
jtd/jtdc	17 / 17	100.0%	0.201s
pdf	163 / 164	99.4%	0.690s
合計	360 / 364	98.9%	0.339s

時点でマルウェアを検知するか否かを確認した．実験に用いた検体に対しては，採取した当時の最新のパターンファイルを適用した大手ベンダのウイルス対策ソフトでも 18.1%から 36.2%の低い確率でしかマルウェアを検知できなかった．しかも，ウイルス対策ソフトで検知できるマルウェアの種類には重複があったため，3 種類のウイルス対策ソフトを組み合わせて，どれか 1 つでも検知した場合（T，S，M 社 AV）をとっても，検知率は 50.4%であった．

表 6: ウイルス対策ソフトとの検知率の比較

	検知数	検知率
試験プログラム	122 / 127	96.1%
T 社 AV	46 / 127	36.2%
S 社 AV	30 / 127	23.6%
M 社 AV	23 / 127	18.1%
T，S，M 社 AV	64 / 127	50.4%

表 7: 検体の概要

拡張子	検体数	平均容量 (KB)
rtf	69	487.7
doc	61	259.0
xls	9	298.4
ppt	2	480.5
pdf	86	653.5
合計	227	481.5

4.5 実験 2 の内容

提案手法の対象を，実行ファイルが埋め込まれたものに限定せず，悪性文書ファイル全般に拡大した場合の有効性を評価するため，VirusTotal[9] に 2013 年以降に登録された悪性文書ファイルを試験プログラムに入力して結果を分析する．実験の対象となる文書ファイルの概要を表 7 に示す．これらの検体は，VirusTotal において，2013 年から 2014 年の間に登録され，スキャンの結果マルウェアと判定されたファイルのうち拡張子が rtf,doc,xls,ppt および pdf であるものという条件で検索し，検索結果を機械的に選定したものである．ただし，拡張子は実際の中身に基づいて分類した．これらの検体を試験プログラムに入力し，検知率を求める．また，試験プログラムの検知率と OfficeMalScanner（OMS）[10] の検知率および Handy Scissors（HS）[11][12] の検知率を比較する．

OMS は，主に exploit に着目したツールで，Rich Text ファイルおよび CFB ファイルから，不正なコードによく利用されるコードを検索等することにより，悪性文書ファイルを検知することができるツールである．HS は，実行ファイルに着目したツールで，様々な形式の悪性文書ファイルに埋め込まれた実行ファイルを自動的に抽出するツールである．

4.6 実験 2 の結果

試験プログラムの検知率と OMS の検知率および HS の検知率との比較結果を表 8 に示す．この表では，検知数/検体数 で検知数を表している．実験に使用した OMS のバージョンは v0.58

表 8: 悪性文書ファイル全般の検知率

拡張子	試験プログラム	OMS	HS
rtf	34 / 69	12 / 69	7 / 69
doc	44 / 61	31 / 61	28 / 61
xls	2 / 9	3 / 9	1 / 9
ppt	0 / 2	0 / 2	1 / 2
pdf	40 / 86	-	8 / 86
合計	120 / 227	46 / 141	46 / 227
検知率	52.9%	32.6%	20.3%

であり，CFB ファイルについては，一般的な shellcode のパターンを検索する “SCAN” オプションおよび総当たりで実行ファイル等を検索する “BRUTE” オプションを使用して実行した．また，Rich Text ファイルについては，OMS に同封されている RTFScan.exe を，“SCAN” オプションを使用して実行した．表中の OMS の検知数は OfficeMalScanner.exe，RTFScan.exe いずれかで検知した数を示す．試験プログラムの検知率は 52.9%であり，OMS の検知率は 32.6%，HS の検知率は 20.3%であった．

5 考察

5.1 実験 1 で検知に失敗した原因

試験プログラムが実験 1 で検知に失敗した検体 5 個を分析した結果，検知失敗の原因は提案手法の検査項目に合致しないように実行ファイルが文書ファイルに埋め込まれていたことに集約された．その埋め込み方式は以下の 3 種類に分類された．

5.1.1 正規の方法で埋め込まれている

実行ファイルが正規のオブジェクトとして文書ファイルの中に埋め込まれ，閲覧ソフトの脆弱性を利用せずに実行ファイルを閲覧者に実行させる場合，文書ファイルには本論文で論じたような特徴は現れない．このため，我々の提案手法では検知することはできない．しかしながら，実行ファイルの実行には，閲覧者がオブジェクトをダブルクリックしたり，警告のポップアップウインドウを閉じたりする等の閲覧者による明示的な操作が必要である．このため，この方式の悪性文書ファイル全体に占める割合は少なくなっている．

5.1.2 shellcode の中に埋め込まれている

shellcode は閲覧ソフトが通常読み込む部分に埋め込まれることが多い．このため，shellcode が埋め込まれた部分には本論文で論じたような特徴が現れないことが多い．shellcode はサイズが制限されることが多いが，利用する脆弱性によっては実行ファイルを埋め込めるほど大きなサイズの shellcode も存在する．その場合，我々の提案手法では検知することはできない．この方式は，使用できる脆弱性が限定されるため，悪性文書ファイル全体に占める割合は少なくなっている．

5.1.3 正規の Stream に偽装されている

実行ファイルが正規の Stream に偽装して CFB ファイルに埋め込まれた場合，CFB ファイルには本論文で論じた特徴は現れない，このため，我々の提案手法では検知することはできない．ただし，偽装された Stream を閲覧ソフトが処理すると，閲覧ソフトが誤動作したり，表示される内容がいわゆる文字化け状態になったりしてしまう．そのため，偽装された Stream は閲覧ソフトに処理されないように，他のどの Stream からも参照されないことが多いと考えられる．したがって，我々の提案手法のうち，悪性 PDF ファイルの検知で行っているオブジェクト間の参照を検査する手法を悪性 CFB ファイルの検知に応用させれば，この方式の CFB ファイルは検知可能と思われる．

また，文書ファイルの構造を構文解釈して中身を書き換えるオープンソースのツールが，PDF の場合は複数普及しているが，CFB の場合はほとんど普及していない．CFB ファイルにこの方式で実行ファイルを埋め込むためには，攻撃者は CFB のファイル構造を習熟する必要がある．そのため，この方式の悪性文書ファイル全体に占める割合は少なくなっている．

5.2 提案手法の長期的効果

文献 [2][3] において実行ファイルが埋め込まれた悪性文書ファイルで実験したところ，試験プログラムは平均 0.339s という短い時間かつ 98.9%という高い確率で悪性文書ファイルを検知することができた．さらに本論文では，より後の時期に標的型攻撃で使用された実行ファイルが埋め込まれた悪性文書ファイルで実験したところ，試験プログラムは平均 0.263s という短い時間かつ 96.1%という引き続き高い確率で悪性文書ファイルを検知することができた．ファイル構造の仕様は攻撃者の意志で変更することが困難であることから，我々の提案手法は長期にわたり高い確率で悪性文書ファイルを検知できることが期待される．検知できた悪性文書ファイルの中にはゼロデイのもあり，最新の悪性文書ファイルが含まれていたが，文書ファイルに実行ファイルを埋め込む方式に大きな変化はなく，提案手法で高い確率で検知することができた．

5.3 提案手法の応用

マルウェアは日々新たなものが出現している．提案手法は，パターンファイルを用いずに 96.1%という高い確率で，実行ファイルが埋め込まれた悪性文書ファイルを検知することに成功した．さらに，検査時間の平均値はわずか 0.263s であった．一般的に，サンドボックス型の標的型メール対策ソフトの検査時間は分単位で時間がかかることを考えると，試験プログラムは高速に検査が完了する．したがって，試験プログラムを組織内のメールサーバ等で自動実行させれば，組織内に到達するメールを高速で検査することが可能である．

一方，組織内に到達する悪性文書ファイルは実行ファイルが埋め込まれた文書ファイルだけではないことから，悪性文書ファイル全般を想定し，実験 2 において，VirusTotal に登録された悪性文書ファイルで実験を行った．その結果，検知率は 52.9%であった．実行ファイルが埋め込まれていない悪性文書ファイルには，我々の提案手法の検査対象であるファイル構造の不整合が生じないことが多い．このため，検知率の向上には，我々の提案手法単体の運用ではなく，実行ファイルが埋め込まれていない悪性文書ファイルを検知できる手法と組み合わせた運用について検討する必要がある．例えば，OMS の場合，検知率は試験プログラムと比較すると低くなっているが，試験プログラムで検知できなかった悪性文書ファイルを検知することができたため，試験プログラムと OMS を組み合わせて，いずれかで検知した場合をとると検知率の向上が見込める．

5.4 提案手法の制約

我々の提案手法は，ファイルフォーマットに依存しており，Rich Text，CFB および PDF にのみに対応している．提案手法は OOXML（Office Open XML）（docx，xlsx，pptx の拡張子）の文書ファイルは検知対象としていない．しかしながら，正規のオブジェクトとして実行ファイルを埋め込む以外の方法で実行ファイルが埋め込まれた OOXML の文書ファイルは確認できていない．その原因として 2 つの理由が考えられる．1 つ目は，OOXML の文書ファイルに埋め込まれた不正なファイルの検知を容易にする仕組みが導入されていることが考えられる．OOXML の文書ファイルの中には複数のファイルが zip 圧縮されて格納されている．格納されている各ファイル相互の関連性を指定するリレーションシップという仕組みがあり，この仕組みを利用することで不正なファイルが埋め込まれた文書ファイルは閲覧ソフトで開くことができない．2 つ目は，仮にリレーションシップの仕組みを突破して実行ファイルを OOXML の文書ファイルに埋め込んだとしても，サイズが制限されることが多い shellcode に zip 圧縮をデコードするコードを組み込むことは困難であることが考えられる．このことから，実行ファイルが埋め込まれた OOXML の文書ファイルを利用した標的型攻撃もほとんど確認できていない．しかしながら，今後 OOXML の悪性文書ファイルを用いた標的型攻撃が増加した場合には，その対応についても検討する必要がある．

6 おわりに

本論文では，ファイル構造検査により悪性文書ファイルを検知するという，我々の提案手法の有効性を新たに採取した悪性文書ファイルで再評価した結果，平均実行時間 0.263s で 96.1%の悪性文書ファイルを検知することができた．ファイル構造は攻撃者の意志で仕様を変更することが困難であり，我々の提案手法は長期にわたり有効であることが確認できた．

今後の課題としては，セキュリティオペレーションセンター（SOC）で監視・分析の対象となる機器（監視センサー）への適用が挙げられる．文献 [2][3] で通常の文書ファイルに対する誤検知率を評価したが，監視センサーで観測される文書ファイルの数は膨大であり誤検知の影響は明らかでない．また，悪性文書ファイル全般に対する有効性についても継続して評価する必要がある．

参考文献

[1] トレンドマイクロ株式会社 : 国内標的型サイバー攻撃分析レポート 2015 年板〜「気付けない攻撃」の高度化が進む〜，トレンドマイクロ株式会社（2015）

[2] 大坪雄平，三村守，田中英彦 : ファイル構造検査による悪性 MS 文書ファイルの検知，情報処理学会論文誌，Vol.55，No.5，pp.1530-1540 (2014).

[3] 大坪雄平，三村守，田中英彦 : ファイル構造検査の悪性 PDF ファイル検知への応用，情報処理学会論文誌, Vol.55, No.10, pp.2281-2289 (2014).

[4] ジャストシステム : 一太郎の脆弱性を悪用した不正なプログラムの実行危険性について (online), https://www.justsystems.com/jp/info/js14003.html (2015-07-16).

[5] Xu, W. Wang, X. Zhang, Y. and Xie, H.: A Fast and Precise Malicious PDF Filter, *Proceedings of the 22nd Virus Bulletin International Conference*, pp.14-19 (2012).

[6] Nedim, S. Pavel, L.: Detection of Malicious PDF Files Based on Hierarchical Document Structure, 20th Annual Network & Distributed System Security Symposium, (2013).

[7] Hyukdon, K. Yeog, K. Sangjin, L. and Jongin, L. : A Tool for the Detection of Hidden Data in Microsoft Compound Document File Format，*ICISS '08 Proceedings of the 2008 International Conference on Information Science and Security*, pp.141-146 (2008).

[8] Microsoft : Microsoft Office 向けの Microsoft Office ファイル検証機能の公開 (online)，http://technet.microsoft.com/ja-jp/security/advisory/2501584 (2015-07-16).

[9] VirusTotal : VirusTotal(online)，https://www.virustotal.com/ja/ (2015-07-16).

[10] Boldewin, F. : Analyzing MSOffice malware with OfficeMalScanner(online)，http://www.reconstructer.org/papers/Analyzing%20MSOffice%20malware%20with%20OfficeMalScanner.zip (2015-07-16).

[11] 三村守，田中英彦 : Handy Scissors : 悪性文書ファイルに埋め込まれた実行ファイルの自動抽出ツール，情報処理学会論文誌, Vol.54, No.3, pp.1211-1219 (2013).

[12] 三村守，大坪雄平，田中英彦 : 悪性文書ファイルに埋め込まれた RAT の検知手法，情報処理学会論文誌, Vol.55, No.2, pp.1089-1099 (2014).